แทรชชี่ ถังขยะอัจฉริยะเพื่อการรีไซเคิล

Trashy : An IoT - based Smart Recycling Bin System

นางสาวสุภาวินี บัวทอง

Suphawinee Buathong

นางสาวพิมพ์ฝัน แสนเรืองเดช

Pimphan Sanruagdech

นางสาวขวัญชนก ผนวกสุข

Kwanchanok Panuaksuk

โครงงานนี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรวิทยาศาสตรบัณฑิต

ภาควิชาวิทยาการคอมพิวเตอร์ คณะวิทยาศาสตร์

มหาวิทยาลัยศรีนครินทรวิโรฒ ปีการศึกษา พ.ศ. 2562

คณะวิทยาศาสตร์ มหาวิทยาลัยศรีนครินทรวิโรฒ

โครงงาน	แทรชชี่ ถังขยะอัจฉริยะเพื่อการรีไซเคิล Trashy : An IoT - based Smart Recycling Bin System
นิสิต	นางสาวสุภาวินี บัวทอง 59102010296 นางสาวพิมพ์ฝัน แสนเรืองเดช 59102010271 นางสาวขวัญชนก ผนวกสุข 59102010563
ปริญญา	วิทยาศาสตรบัณฑิต(วท.บ.)
ภาควิชา	วิทยาการคอมพิวเตอร์
อาจารย์ที่ปรึกษาโครงงาน	ผู้ช่วยศาสตราจารย์ ดร.จันตรี ผลประเสริฐ

ลงชื่อ.....จันตรี ผลประเสริฐ.....

(ผู้ช่วยศาสตราจารย์ ดร.จันตรี ผลประเสริฐ)

อาจารย์ที่ปรึกษาโครงงาน

บทคัดย่อ

การจัดการขยะเป็นหนึ่งในปัญหาสำคัญของโลกในขณะนี้ การจัดการขยะที่ไม่เหมาะสมจะก่อให้เกิดปัญหาอื่นๆ ตามมาอีกมากมาย เช่น ปัญหาทางสุขอนามัย เพราะขยะเป็นแหล่งสะสมของเชื้อโรค หรือปัญหาภาวะโลกร้อน อันเกิดจากการกำจัดขยะอย่างไม่มีประสิทธิภาพโดยการเผาไหม้ แนวทางหนึ่งในการจัดการขยะที่เป็นไปได้ คือการแยกประเภทขยะอย่างเหมาะสม เพื่อให้สามารถจัดการขยะได้อย่างมีประสิทธิภาพ เช่น การนำขยะบางประเภทไปฝังกลบเพื่อทำเป็นปุ๋ย หรือการนำขยะบางประเภทมาใช้ใหม่ให้มีประสิทธิภาพสูงสุด อย่างไรก็ตาม ระบบการแยกขยะในหลายๆ ที่ยังไม่มีประสิทธิภาพเพียงพอ อีกทั้งบุคลากรที่เกี่ยวข้องยังขาดความรู้ความเข้าใจในการแยกขยะส่งผลให้การแยกขยะ เพื่อจัดการอย่างเหมาะสมยังทำได้ไม่มีประสิทธิภาพเท่าใดนัก โครงงานแทรชชี่ ถังขยะอัจฉริยะเพื่อการรีไซเคิล (Trashy : An IoT - based Smart Recycling Bin System) เป็นการพัฒนาต้นแบบถังขยะอัจฉริยะที่สามารถคัดแยกขยะได้โดยอัตโนมัติตามประเภทขยะ ได้แก่ แก้ว พลาสติก กระป๋องโลหะ และขยะอื่นๆ เพื่ออำนวยความสะดวกต่อบุคลากรในการจัดการขยะ เพื่อนำมารีไซเคิลต่อไป โดยต้นแบบนี้ใช้ Raspberry Pi 4 สำหรับเป็นตัวควบคุมการทำงานของถังขยะอัจฉริยะ โดยจะทำการแยกขยะจากข้อมูลภาพถ่าย และจะมีมอเตอร์ควบคุมในการทิ้งขยะให้ถูกประเภท ใช้ Ultrasonic sensor ในการตรวจวัดปริมาณขยะในถัง และมีการแจ้งเตือนเจ้าหน้าที่ให้มาเก็บขยะผ่าน Line Notify เมื่อขยะใกล้เต็มถังขยะ โดยข้อมูลของขยะทั้งหมดจะถูกเก็บอยู่บนฐานข้อมูลบนคลาวด์ เพื่อใช้ในการทำ Business Intelligence วิเคราะห์ปริมาณขยะในแต่ละพื้นที่ต่อไป ในส่วนของการคัดแยกขยะ ต้นแบบถังขยะอัจฉริยะนี้ใช้ model การเรียนรู้เชิงลึก (Deep learning) แบบ resnet50 (Residual neural network 50) ซึ่งจะทำการวิเคราะห์ภาพถ่ายและทำการจำแนกขยะออกเป็น 4 ประเภท แก้ว พลาสติก กระป๋องโลหะ และขยะอื่นๆ โดยเมื่อระบบทำการวิเคราะห์ประเภทของขยะแล้วก็จะทำการทิ้งขยะตามประเภทของขยะในแต่ละถังโดยอัตโนมัติ จากการทดสอบประสิทธิภาพของถังขยะต้นแบบเบื้องต้น พบว่าถังขยะอัจฉริยะนี้สามารถจำแนกประเภทขยะได้ถูกต้องร้อยละ 98 ตามผลที่ได้จากการเรียนรู้ข้อมูล แต่ทางด้านการนำไปประยุกต์ใช้จริงสามารถจำแนกได้ถูกต้องประมาณร้อยละ 84 เนื่องจากมีตัวแปรตามอื่นๆ ที่มีผลต่อการทำนาย เช่น แสงจากภายนอก แสงสะท้อน เงาตกกระทบ เป็นต้น ดังนั้นจึงมีความผิดพลาดเกิดขึ้นได้ โดยในการทำงานจริงจะใช้เวลาในการประมวลผลประมาณ 12 - 13 วินาที โดยการถ่ายภาพขยะในกล่องเป็นเวลา 3 วินาที, การทำนายผลในการจำแนกขยะเป็นเวลาประมาณ 5 - 6 วินาที, การดันขยะให้ลงถังเป็นเวลา 4 วินาที ส่วนที่เหลือคือการทำงานของมอเตอร์ที่แตกต่างกันไป ขึ้นอยู่กับระยะทางไปยังถังขยะประเภทนั้นๆ

Abstract

Recently, waste management is one of the biggest problems encountered by many countries. Improper waste management leads to many ensuing problems for society such as health issues due to germs and global warming since many people destroy waste by burning. One possible solution to alleviate this problem is to properly classify the type of garbage and handle each type efficiently. For example, some organic waste could be fermented to be used as fertilizer or some waste could be recycled. However, waste classification in many countries is neglected due to many factors such as incompetent authority or lack of waste management awareness leading to inefficient waste management system. In this project, we propose 'Trashy' (An IoT - based Smart Recycling Bin System), an intelligent waste classification bin that can accurately and automatically classify trash into 4 types: glass, metal, plastic and others. We employ Raspberry Pi4 (RPi4) as the main processor to control the overall system. The bin utilized visual information to classify garbage where a deep learning model called resnet50 (Residual neural network 50) is employed. Once the type of garbage is decided, RPi4 controls a motor to send the trash into the appropriate bin type. The trash employs Ultrasonic sensors to measure the level of trash in the bin and informs responsible authorities through Line messenger to pick up the trash. All trash information is stored in cloud database for further analysis. Our model yields 98% classification accuracy for training dataset but achieves 84% classification accuracy for testing dataset. Performance drops on the testing dataset are due to several factors such as external light and shadow. Image classification using resnet50 takes approximately 5-6 seconds. Future works will focus on improving classification speed using less complex models.

กิตติกรรมประกาศ

โครงงานแทรชชี่ ถังขยะอัจฉริยะเพื่อการรีไซเคิล (Trashy : An IoT - based Smart Recycling Bin System) สามารถดำเนินการได้สำเร็จลุล่วง โดยได้รับความช่วยเหลือ ร่วมมือและความอนุเคราะห์ จากหลายฝ่ายเป็นอย่างดี จึงใคร่ขอขอบพระคุณบุคคล และหน่วยงานต่างๆ ดังต่อไปนี้

ขอขอบพระคุณ ผู้ช่วยศาสตราจารย์ดร.จันตรี ผลประเสริฐ อาจารย์ที่ปรึกษาโครงงานผู้ให้ความรู้ และคำแนะนำเกี่ยวกับแหล่งค้นคว้าศึกษาข้อมูลที่เป็นประโยชน์ รวมถึงช่วยแนะนำแนวทาง ในการแก้ไข ปัญหาต่างๆ ในระหว่างการจัดทำโครงงานตลอด จนการตรวจสอบแก้ไขข้อบกพร่องในการทำโครงงาน และทุนสนับสนุนในการทำโครงงานแทรชชี่ ถังขยะอัจฉริยะเพื่อการรีไซเคิล (Trashy : An IoT - based Smart Recycling Bin System)

ขอขอบพระคุณอาจารย์ทุกท่านในภาควิชาวิทยาการคอมพิวเตอร์ มหาวิทยาลัยศรีนครินทรวิโรฒ ที่กรุณาให้คำแนะนำ ข้อคิดเห็น เสนอแนวทางการแก้ไขปัญหาในการจัดทำโครงงาน และให้ความรู้ ต่างๆ ที่เป็นประโยชน์ต่อการจัดทำโครงงานแทรชชี่ ถังขยะอัจฉริยะเพื่อการรีไซเคิล (Trashy : An IoT - based Smart Recycling Bin System)

ขอขอบคุณเพื่อนๆ ภาควิชาวิทยาการคอมพิวเตอร์ มหาวิทยาลัยศรีนครินทรวิโรฒ ที่ได้แนะนำวิธีการด้านเทคนิคและให้คำปรึกษาในการพัฒนาโครงงานแทรชชี่ ถังขยะอัจฉริยะเพื่อการรีไซเคิล (Trashy : An IoT - based Smart Recycling Bin System)

ขอขอบพระคุณทุนอุดหนุนโครงงานการแข่งขันพัฒนาโปรแกรมคอมพิวเตอร์แห่งประเทศไทย ครั้งที่ ๒๒ จากสำนักงานพัฒนาวิทยาศาสตร์และเทคโนโลยีแห่งชาติ ที่ได้ให้ทุนอุดหนุนสำหรับการพัฒนาโครงงานแทรชชี่ ถังขยะอัจฉริยะเพื่อการรีไซเคิล (Trashy : An IoT - based Smart Recycling Bin System)

สุดท้ายนี้ขอขอบพระคุณบิดามารดาและครอบครัว ที่เปิดโอกาสให้ศึกษาเล่าเรียน รวมถึงให้ความช่วยเหลือและให้กำลังใจคณะผู้จัดทำจนโครงงานประสบความสำเร็จเป็นอย่างดี

คณะผู้จัดทำ

สารบัญ

สารบัญภาพ

สารบัญตาราง

บทที่ 1

บทนำ

1.1 ที่มาและความสำคัญของปัญหา

ปัจจุบันประเทศไทยมีปริมาณขยะเพิ่มสูงขึ้นอย่างต่อเนื่อง จากในปี 2561 ประเทศไทยมีปริมาณขยะมูลฝอยที่เกิดขึ้นทั่วทั้งประเทศประมาณ 27.8 ล้านตัน เมื่อเปรียบเทียบกับปี 2560 มีปริมาณเพิ่มขึ้นร้อยละ 1.64 เนื่องจากการขยายตัวของชุมชนเมือง และการปรับเปลี่ยนวิถีชีวิตจากสังคมเกษตรกรรมสู่สังคมเมือง การเพิ่มขึ้นของประชากร การส่งเสริมการท่องเที่ยว การบริโภคที่เพิ่มมากขึ้น ส่งผลให้ปริมาณขยะมูลฝอยในหลายพื้นที่เพิ่มมากขึ้น และยังมีการจัดการขยะที่ถูกกำจัดอย่างไม่ถูกต้องประมาณ 7.36 ล้านตัน หรือคิดเป็นร้อยละ 27 [1] เมื่อพิจารณาพบว่าปริมาณขยะมูลฝอยมีแนวโน้มเพิ่มขึ้น

จากการสรุปสถานการณ์มลพิษของประเทศไทยปี 2561 ที่ผ่านมาพบว่าขยะมูลฝอยชุมชน 27.8 ล้านตัน ได้ถูกนำไปกำจัดอย่างถูกต้องประมาณ 10.88 ล้านตัน หรือคิดเป็นร้อยละ 39 ของปริมาณขยะมูลฝอยทั้งหมด และนำกลับไปใช้ประโยชน์ 9.58 ล้านตัน หรือคิดเป็นร้อยละ 34 เพิ่มขึ้นจากปีที่ผ่านมาร้อยละ 13 ส่วนใหญ่เป็นการใช้ประโยชน์จากขยะรีไซเคิลและทำปุ๋ยอินทรีย์ ขยะมูลฝอยชุมชนส่วนที่เหลือเป็นขยะที่ถูกกำจัดอย่างไม่ถูกต้องประมาณ 7.36 ล้านตัน หรือคิดเป็นร้อยละ 27 [1] เช่น เทกอง การเผากำจัดกลางแจ้ง การเผากำจัดในเตาเผาขนาดเล็กที่ไม่มีการบำบัดมลพิษทางอากาศ และการลักลอบทิ้งในพื้นที่ต่างๆ [2]

ในการดำเนินชีวิตประจำวันของมนุษย์ต้องเกี่ยวข้องกับการอุปโภค บริโภค หรือการทำกิจกรรมต่างๆ ของมนุษย์ ก่อให้เกิดขยะมูลฝอย ทั้งขยะอันตราย ขยะที่ย่อยสลายได้ และขยะที่ย่อยสลายไม่ได้ ปัญหาขยะเป็นปัญหาใหญ่ระดับต้นๆ ของหลายประเทศที่ยากจะแก้ไขภายในวันเดียว เพราะขยะเกิดขึ้นทุกวันและมีจำนวนมากขึ้นเรื่อยๆ อีกทั้งการทำลายหรือกำจัดขยะที่ไม่มีประสิทธิภาพ อาจส่งผลให้ขยะตกค้าง หรือเกิดปัญหาอื่นตามมา เช่น การทิ้งถุงพลาสติกกับขวดน้ำ หากเราแยกขวดพลาสติกกับถุงพลาสติกได้ ขวดก็สามารถนำไปรีไซเคิลและใช้ประโยชน์ต่อไปได้ แต่ถุงพลาสติกต้องอาศัยการกำจัดที่ถูกวิธี เพราะการย่อยสลายพลาสติกตามธรรมชาติต้องใช้การย่อยสลายเป็นเวลานาน การทำลายพลาสติกโดยการเผาไหม้ อาจทำให้เกิดควันซึ่งเป็นปัญหาทางมลพิษทางอากาศ จากตัวอย่างปัญหาต่างๆ ที่กล่าวมาคณะผู้จัดทำตระหนักถึงความสำคัญของปัญหานี้ จึงนำเสนอแนวทางหนึ่งในการแก้ไขปัญหาขยะ โดยการทำเครื่องสำหรับแยกประเภทขยะ เพื่อนำขยะเหล่านั้นไปผ่านกระบวนการกำจัดอย่างถูกวิธี นำไปรีไซเคิล นำไปใช้ประโยชน์ และทำให้เกิดความสะดวกต่อผู้ที่นำขยะไปทิ้งให้มากที่สุด

จากอัตราการกำจัดขยะอย่างถูกต้องที่มีเพียงร้อยละ 39 อาจเกิดจากการจัดการขยะที่ไม่เอื้อต่อการนำไปรีไซเคิล จึงได้มีการรณรงค์ต่างๆ เช่น การแยกขยะเพื่อนำไปรีไซเคิล แต่การกระทำของมนุษย์

- มักง่ายและขาดจิตสำนึก
- ขาดความรู้หรือความใส่ใจในการแยกขยะ
- การเก็บหรือทำลายขยะอย่างไม่มีประสิทธิภาพ

จากปัญหาดังกล่าว โครงงานนี้นำเสนอถังขยะอัจฉริยะที่สามารถแยกขยะได้ โดยเราจะทำการแยกขยะออกเป็น 4 ประเภท คือ แก้ว พลาสติก กระป๋องโลหะ และขยะอื่นๆ ซึ่งจะเน้นไปที่ขยะที่สามารนำไปรีไซเคิลได้ ประสิทธิภาพเบื้องต้นของการคัดแยกประเภทขยะอยู่ที่ประมาณร้อยละ 98 มีการใช้ Ultrasonic sensor เพื่อวัดปริมาณขยะภายในถัง โดยจะมีการแจ้งเตือนให้เจ้าหน้าที่ทราบ เมื่อปริมาณขยะใกล้เต็มถังและมีการเชื่อมต่อฐานข้อมูล AWS S3 เพื่อเก็บข้อมูลภาพหลังจากการแยกประเภทนั้นๆ แล้ว

1.2 วัตถุประสงค์ของงานวิจัย

1.2.1 พัฒนาถังขยะที่สามารถจำแยกขยะประเภทรีไซเคิลทั้ง 4 ประเภท ได้อย่างมีประสิทธิภาพ

1.2.2 ศึกษาการใช้ Deep learning เพื่อมาวิเคราะห์ภาพในการจำแนกขยะ

1.2.3 ศึกษา Embedded system เพื่อใช้ในการควบคุมถังขยะ

1.2.4 ศึกษาเทคโนโลยีเซนเซอร์ที่ใช้ในการตรวจจับขยะและวัดปริมาณขยะ

1.2.5 ศึกษาแนวทางการเก็บข้อมูลภาพขยะลงบนระบบ Cloud

1.3 ขอบเขตของโครงงาน

1.3.1 สามารถตรวจจับปริมาณขยะในถังได้ว่าจะเต็มหรือยัง (ความคลาดเคลื่อนไม่เกิน 2 - 3 cm)

1.3.2 สามารถจำแนกขยะเป็นแก้ว พลาสติก กระป๋องโลหะ และขยะอื่นๆ ได้

1.3.3 สามารถรับขยะได้เพียง 1 ประเภท ต่อ 1 รอบการทำงานของถังขยะอัจฉริยะ

1.3.4 ใช้ข้อมูลจากภาพเพียงอย่างเดียวในการจำแนกขยะ

1.3.5 ถังขยะจะใช้พลังงานจากไฟฟ้ากระแสสลับ (AC) จากการเสียบปลั๊ก

1.4 ประโยชน์ที่คาดว่าจะได้รับ

1.4.1 เรียนรู้เกี่ยวกับ Machine Learning

1.4.2 เรียนรู้เกี่ยวกับ IoT

1.4.3 ช่วยลดปัญหาการจัดการขยะที่ไม่ถูกต้อง

1.4.4 ช่วยลดปัญหาขยะล้นถัง

1.4.5 ช่วยลดเวลาในการคัดแยกขยะ

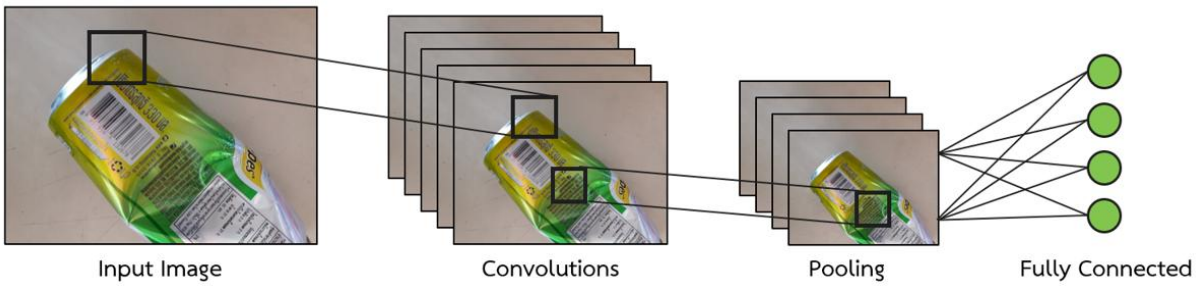
บทที่ 2

ทฤษฎีและแนวคิดที่ใช้ในการทำวิจัย

2.1 ทฤษฎีและแนวคิดที่ใช้ในการทำโครงงาน

2.1.1 Convolutional Neural Network (CNN)

CNN จะจำลองการมองเห็นของมนุษย์ที่มองพื้นที่เป็นที่ย่อยๆ และนำกลุ่มของพื้นที่ย่อยๆ มาผสานกัน เพื่อดูว่าสิ่งที่เห็นอยู่เป็นอะไร โดยการมองพื้นที่ย่อยของมนุษย์จะมีการแยกคุณลักษณะ (feature) ของพื้นที่ย่อยนั้น เช่น ลายเส้น และการตัดกันของสี ซึ่งการที่มนุษย์รู้ถึงคุณลักษณะของพื้นที่ย่อยนี้ เพราะมนุษย์ดูทั้งจุดที่สนใจและบริเวณรอบๆ ประกอบกัน [3] ดังตัวอย่างในภาพที่ 1

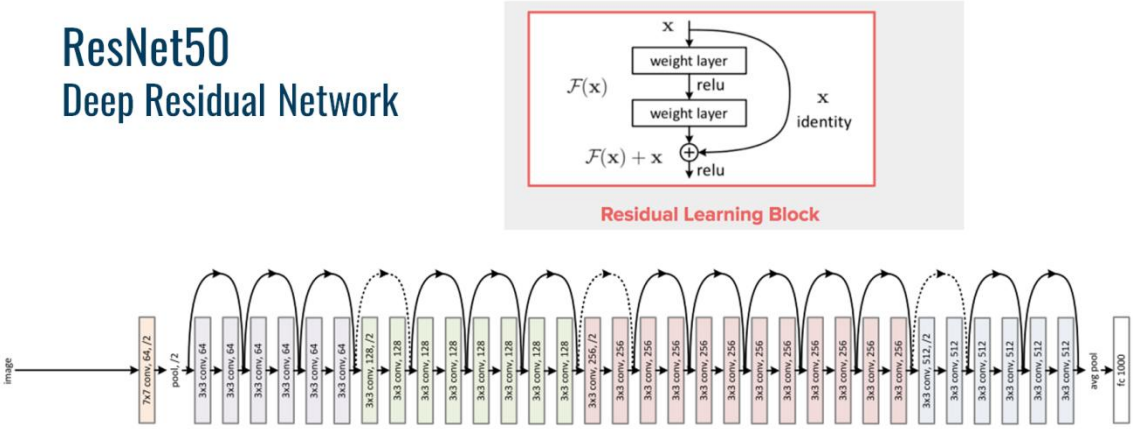


ภาพที่ 1 : การทำงานของ CNN

2.1.2 Residual Network (ResNet)

Residual Network (ResNet) เป็นเครือข่ายประสาทเทียมเชิงลึกที่สามารถจำแนกภาพ จากการดึงคุณลักษณะของภาพ มีโครงสร้างที่ลึกมาก เหมาะกับข้อมูลที่มีขนาดใหญ่ โดย ResNet ได้เสนอให้ Convolution layer บางชั้นนั้นอาจจะไม่จำเป็น และเสนอให้เพิ่มการ skip ข้าม layer ไว้ ถ้าชั้นหนึ่งๆ ไม่จำเป็นจริงๆ ค่าถ่วงน้ำหนักบน layers เหล่านั้นก็จะถูก train ให้เข้าใกล้ 0 ไป ส่วนข้อมูลที่จำเป็นก็ยังสามารถ flow ผ่าน skip connection ไปชั้นต่อไปได้ [4]

โดยคณะผู้จัดทำได้เลือกใช้ ResNet50 เพราะจากผลการทดลองจากหลายๆ โมเดล โมเดลนี้มีความถูกต้องในการนำไปประยุกต์ใช้จริงมากกว่าโมเดลอื่นๆ ซึ่งโมเดลนี้มีโครงสร้างของ ResNet ที่ประกอบด้วย 50 layers การนับจำนวนชั้นก็จะต่างกันไปในแต่ละคนที่ออกแบบดังภาพที่ 2 จะเห็นได้ว่ามีการทำงาน 50 layers แต่ละ layer จะมีการกำหนดค่าต่าง เช่น Convolution, Activation function และอื่นๆ รวมถึงการปรับค่าอย่าง Learning rate เป็นต้น

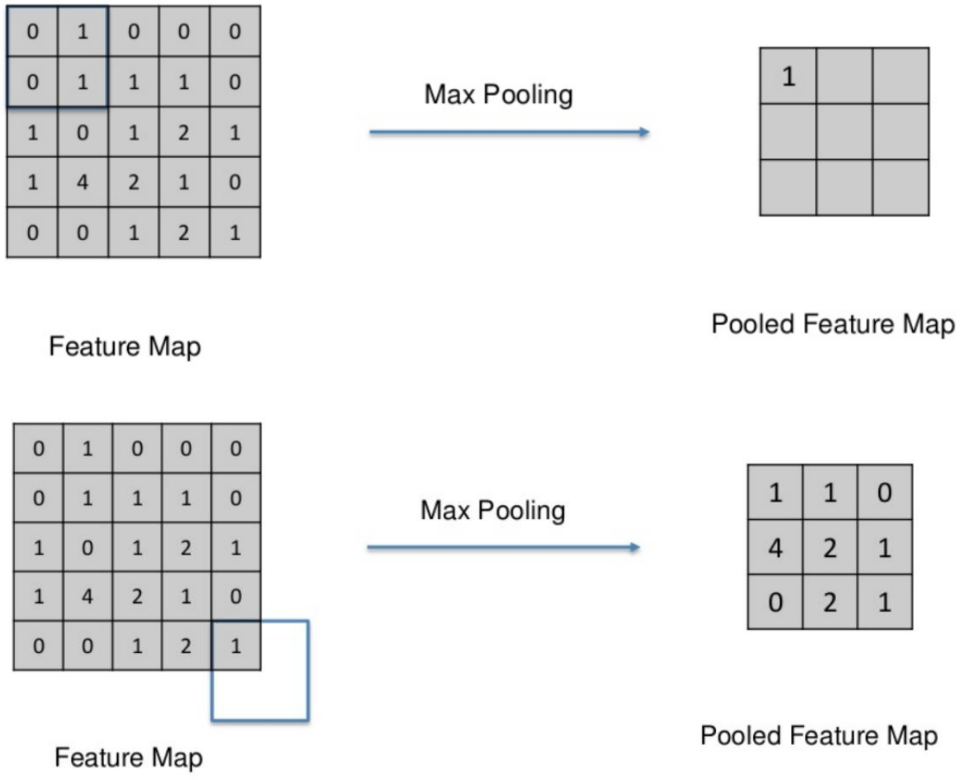


ภาพที่ 2 : Architecture of ResNet50

ที่มา : https://kharshit.github.io/blog/2018/09/07/skip-connections-and-residual-blocks

โดยจะประกอบไปด้วย layer หลักๆดังนี้

- Convolution layer (CONV) จะใช้ filters ที่ช่วยดึงคุณลักษณะของอินพุตที่ใช้ในการรู้จำวัตถุออก โดยมีการกำหนด hyperparameters ด้วย filter หรือ padding ตามลักษณะอินพุต [5]
- Max Pooling คือ ตัวกรอกที่ใช้ในการเลือกค่าที่มากที่สุดในบริเวณเมทริกซ์เดียวกัน [5] ดังภาพที่ 3

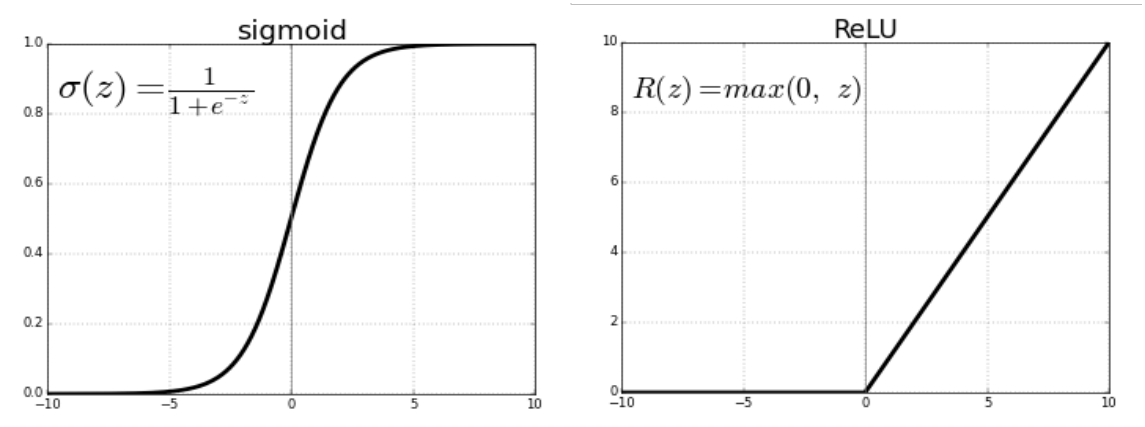


ภาพที่ 3 : การทำ Max Pooling

ที่มา : https://medium.com/@pradyasin/what-is-convolution-neural-network-bf2e525089f5

- Batch Normalizing (BatchNorm) คือ เทคนิคที่ใช้ระหว่างการเทรน Machine Learning เพื่อปรับ Shift, Scale ให้ Activation ที่อยู่ภายใน Hidden Layer ของ Deep Neural Network ให้มีขนาดเหมาะสม ไม่เล็กหรือไม่ใหญ่จนเกินไป โดยดูเทียบจากค่า Mean และ Standard Deviation ของทุก Activation ใน Layer ของทั้ง Batch นั้น และมีการเสริมด้วย Learning Parameter เพื่อให้ model เรียนรู้ที่จะปรับ Activation ให้เป็นที่ต้องการได้เอง Batch Normalization ทำให้แต่ละ layer ใน Neural Network สามารถเรียนรู้ได้ด้วยตัวเองลดการผูกติดกับ layer อื่นๆ [6]

- Rectified Linear Unit (ReLU) คือ ฟังก์ชันเส้นตรงที่ถูกปรับแก้ Rectified ไม่ได้ ดังภาพที่ 4 ซึ่งจะเป็นฟังก์ชันที่เรียบง่ายกว่าทุก activation function อื่นๆ โดยถ้าค่า input เป็นบวก slope จะเท่ากับ 1 ตลอดกาล ทำให้ gradient ไม่หาย (ไม่เกิด vanishing gradient) ส่งผลให้เราเทรน model ได้เร็วขึ้นมาก [7]



ภาพที่ 4 : Sigmoid and ReLU Activation Function

ที่มา : https://www.bualabs.com/archives/1355/what-is-relu-function-why-popular-deep-learning-training-deep-neural-network-activation-function-ep-3/

2.1.3 Cloud Computing

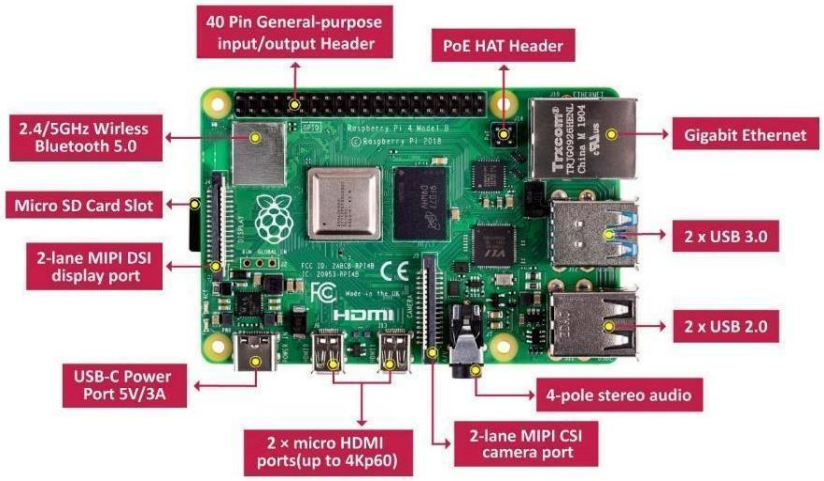
Cloud computing คือเทคโนโลยีแบบใหม่ในการเก็บข้อมูล ซึ่งทำในรูปแบบออนไลน์ ซึ่งสามารถเข้าถึงได้ทั้งผ่านคอมพิวเตอร์ มือถือ แท็บเล็ตหรืออุปกรณ์อื่นๆ ซึ่งทำให้เข้าถึงได้ทุกที่ทุกเวลา Cloud computing ยังเป็นตัวแทนของโครงสร้างอินเตอร์เน็ตปัจจุบัน

โดยคณะผู้จัดทำได้เลือกใช้ Amazon Web Services (AWS) เป็นแพลตฟอร์มระบบคลาวด์ที่ครอบคลุมและใช้งานกันมากที่สุดมีขอบเขตการให้บริการที่กว้างขวาง และยังมีการทำงานที่ครบวงจรที่สุดในบริการเหล่านั้น ตัวอย่างเช่น [8]

Amazon Simple Storage Service (Amazon S3) คือ บริการพื้นที่จัดเก็บอ็อบเจ็กต์ ที่มอบการปรับขนาด ความพร้อมใช้งานของข้อมูล ความปลอดภัย และประสิทธิภาพชั้นนำ เช่น เว็บไซต์ แอปพลิเคชันมือถือ การสำรองข้อมูล และการคืนค่า การเก็บถาวร การใช้งานในองค์กร อุปกรณ์ IoT และการวิเคราะห์ Big Data Amazon S3 มอบคุณสมบัติการจัดการที่ใช้งานได้ง่าย เพื่อให้สามารถจัดระเบียบข้อมูลและกำหนดค่าการควบคุมการเข้าถึงที่ตรงกัน เพื่อตอบสนองต่อความต้องการ [9]

2.1.4 Raspberry Pi

Raspberry Pi คือบอร์ดคอมพิวเตอร์ขนาดเล็กที่สามารถเชื่อมต่อกับจอมอนิเตอร์ คีย์บอร์ด เมาส์ และอีกทั้งบอร์ด Raspberry Pi ยังรองรับระบบปฏิบัติการลินุกซ์ (Linux Operating System) ได้หลายระบบ เช่น Raspbian (Debian), Pidora (Fedora) และ Arch Linux เป็นต้น โดยติดตั้งบน SD Card บอร์ด Raspberry Pi นี้ถูกออกแบบมาให้มี CPU, GPU และ RAM อยู่ภายในชิปเดียวกัน ดังแสดงในภาพที่ 5 มีจุดเชื่อมต่อ GPIO ให้ผู้ใช้สามารถนำไปใช้ร่วมกับอุปกรณ์อิเล็กทรอนิกส์อื่นๆ ได้ [10]

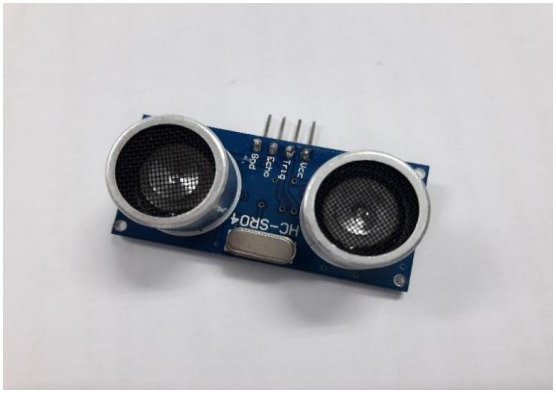


ภาพที่ 5 : Raspberry Pi 4

ที่มา : https://raw.githubusercontent.com/SeeedDocument/Raspberry-Pi-4/master/img/hardware-overview-1400.jpg

2.1.5 Ultrasonic Sensor

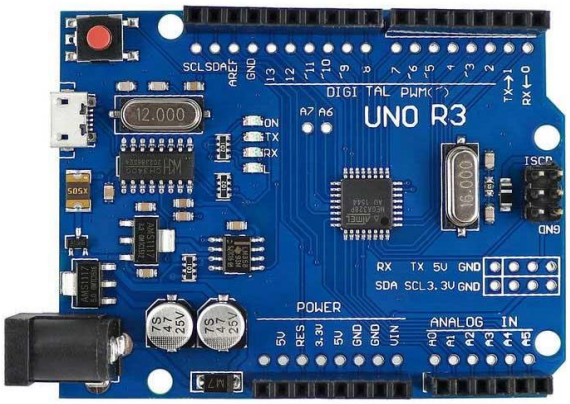
Ultrasonic Sensor เป็นเซนเซอร์ที่ทำงานโดยอาศัยคลื่นเสียงที่มีความถี่สูงกว่า 20 กิโลเฮิร์ต (kHz) ซึ่งเป็นคลื่นในย่านที่มนุษย์ไม่สามารถได้ยินเสียง เซนเซอร์ชนิดอัลตราโซนิกทำงานโดยอาศัยการกระจาย หรือการเคลื่อนที่ของคลื่นเสียงไปกระทบกับพื้นผิวของตัวกลาง และบางส่วนของคลื่นเสียง จะแทรกผ่านเข้าไปในตัวกลางนั้น และส่วนใหญ่ของคลื่นความถี่สูงนี้ จะสะท้อนกลับเรียกว่า "Echo" โดยช่วงเวลาของการสะท้อนกลับของคลื่นเสียงเป็นสัดส่วนโดยตรงกับระยะห่างระหว่างวัตถุกับเซนเซอร์ [11] ดังแสดงในภาพที่ 6



ภาพที่ 6 : Ultrasonic Sensor

2.1.6 Arduino Uno R3

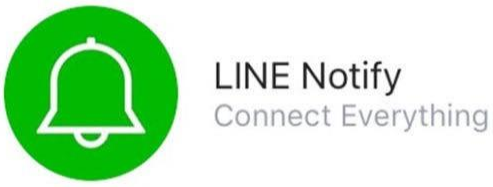
บอร์ดไมโครคอนโทรลเลอร์แบบ Open-source บนแพลตฟอร์ม Arduino โดยใช้ชิป ATmega328 รันที่ความถี่ 16 MHz หน่วยความจำแฟลช 32 KB แรม 2 KB บอร์ดใช้ไฟเลี้ยง 7 ถึง 12V มีระดับแรงดันไฟฟ้าในการทำงานและขาสัญญาณอยู่ที่ 5V (TTL) มี Digital Input / Output 14 ขา (เป็น PWM ได้ 6 ขา) มี Analog Input 6 ขา Serial UART 1 ชุด I2C 1 ชุด SPI 1 ชุด เขียนโปรแกรมบนซอฟท์แวร์ Arduino IDE และโปรแกรมผ่านพอร์ต USB ดังภาพที่ 7 [12]



ภาพที่ 7 : Arduino Uno R3 ATmega328P

ที่มา : https://img.dxcdn.com/productimages/sku_370842_1.jpg

2.1.7 Line notify



ภาพที่ 8 : บริการ LINE Notify

ที่มา : https://images.app.goo.gl/qjbgekKTWmjmXMvK9

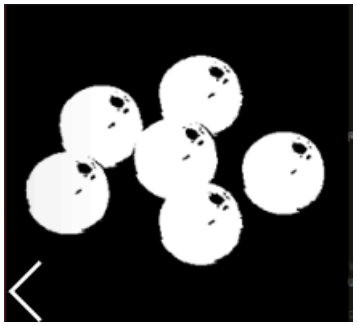
LINE Notify คือ บริการที่รับข้อความแจ้งเตือนจากเว็บเซอร์วิสต่างๆ ที่สนใจได้ทาง LINE โดยหลังเสร็จสิ้นการเชื่อมต่อกับทางเว็บเซอร์วิสแล้ว จะได้รับการแจ้งเตือนจากบัญชีทางการของ “LINE Notify” ซึ่งให้บริการโดย LINE สามารถเชื่อมต่อกับบริการที่หลากหลาย และยังสามารถรับการแจ้งเตือนทางกลุ่มได้อีกด้วย

การใช้งานโดยรวมของ LINE notify จะมีรูปแบบดังนี้ คือ เริ่มแรกต้องไปสร้าง token ของ account ในระบบของ Line ก่อน จากนั้นเก็บ token นี้เอาไว้ เมื่อต้องการที่จะส่งข้อความแจ้งเตือนต่างๆ จะใช้ token นี้เพื่อส่งข้อความแจ้งเตือน ผ่านทาง http post [13]

2.1.8 Background Subtraction

การทำ Background Subtraction เป็นการหาความต่างของภาพ foreground กับภาพ background โดยการนำภาพทั้งสองมาลบกัน หลังจากนั้นก็นำผลลัพธ์จากการลบกันของ 2 ภาพ มาทำ Threshold ซึ่งเป็นการใช้ค่าคงที่ค่าหนึ่ง ในการเปรียบเทียบกับค่าของ Pixel ในแต่ละพื้น ถ้าค่าของ Pixel ในพื้นที่นั้นมีค่าน้อยกว่าค่าคงที่ก็จะเปลี่ยนค่า Pixel ของพื้นที่นั้นเป็น 0 แต่ถ้าค่าของ Pixel ในพื้นที่นั้นมีค่ามากกว่าก็จะเปลี่ยนค่า Pixel ของพื้นที่นั้นเป็น 255 โดยในงานวิจัยนี้เราใช้เป็น Threshold Binary หลังจากนั้นนำภาพที่ได้จากการทำ Threshold มาทำการลบ noises ด้วยการทำ Median Blurring และ Gaussian Blurring และหลังจากนั้นนำภาพที่ได้มาทำการ sum ค่าจุดสีขาว เพื่อหาว่ามีวัตถุหรือไม่ โดยภาพที่ไม่มีวัตถุจะเกิดจากการที่ภาพ foreground กับภาพ background เป็นภาพแบบเดียวกันจะได้ภาพสุดท้ายเป็นภาพสีดำล้วนดังภาพที่ 9 และภาพที่มีวัตถุจะได้ภาพสุดท้ายที่มีบริเวณรอบๆ วัตถุเป็นสีดำ และวัตถุเป็นสีขาว ดังภาพที่ 10

ภาพที่ 9 : ภาพที่ไม่มีวัตถุ



ภาพที่ 10 : ภาพที่มีวัตถุ

ที่มา : https://docs.opencv.org/2.4/doc/tutorials/imgproc/threshold/threshold.html#threshold-binary

2.2 งานวิจัยที่เกี่ยวข้อง

2.2.1 Multilayer Hybrid Deep-Learning Method for Waste Classification and Recycling [14]

ผู้แต่ง : Yinghao Chu, Chen Huang, Xiaodan Xie, Bohai Tan, Shyam Kamal and Xiaogang Xiong

งานวิจัยนี้เป็นการศึกษาเกี่ยวกับการแยกขยะ โดยใช้รูปภาพนำมาผ่านกระบวนการมัติเลเยอร์ไฮบริดเลิร์นนิ่ง โดยกระบวนการไม่ได้มีเฉพาะอัลกอริทึมมาช่วย ในการจำแนกขยะเพียงเท่านั้น แต่ยังมี Sensor ต่างๆ ที่นำมาช่วยให้การจำแนกประสิทธภาพในการจำแนกขยะเพิ่มขึ้น ได้แก่ Bridge sensor ช่วยในเรื่องของการวัดน้ำหนัก Inductor ใช้สำหรับวัดโลหะ

2.2.2 RecycleNet: Intelligent Waste Sorting Using Deep Neural Networks [15]

ผู้แต่ง : Cenk Bircanoğlu, Meltem Atay, Fuat Beşer, Özgün Genç and Merve Ayyüce Kızrak

งานวิจัยนี้เป็นการนำ Convolution Neural Network มาใช้ในการจำแนกขยะของชุดข้อมูล TrashNet ซึ่งจากการวิจัยนี้ได้ลองทดสอบโดยใช้ CNN แบบต่างๆ มาเปรียบเทียบ แล้วได้ทำการบันทึกผล ความแม่นยำของ model เหล่านั้น ซึ่งการวิจัยนี้ได้ให้ความเห็นว่า model เหล่านั้นมีความแม่นยำแต่ เวลาในการทำนายค่อนข้างช้า ทางคณะผู้จัดทำจึงได้มีการนำเสนอ model แบบใหม่ที่มีความรวดเร็วในการทำนายมากขึ้นเพื่อให้เหมาะกับการใช้งานแบบเรียลไทม์

2.2.3 A LoRa-based IoT Sensor Node for Waste Management Based on a Customized Ultrasonic Transceiver [16]

ผู้แต่ง : Tommaso Addabbo, Ada Fort, Alessandro Mecocci, Marco Mugnaini, Stefano Parrino, Alessandro Pozzebon and Valerio Vignoli

งานวิจัยนี้กล่าวถึงการใช้ Ultrasonic Sensor ราคาไม่แพง เพื่อมาตรวจวัดปริมาณขยะในถังขยะ ซึ่ง Ultrasonic Sensor ที่ราคาไม่แพงจะมีการตรวจวัดที่ไม่ค่อยแม่นยำมากนัก ซึ่งงานวิจัยนี้ได้นำ Ultrasonic Sensor ที่ราคาไม่แพงมาปรับปรุง ให้มีการวัดปริมาณขยะในถังขยะให้แม่นยำมากขึ้น และได้ทดลองนำถังขยะที่มีการปรับแต่งแล้วไปติดตั้งในเมืองฟลอเรนซ์ ประเทศอิตาลี

2.2.4 ระบบติดตามถังขยะอัจฉริยะ Smart Trash Tracking System [17]

ผู้แต่ง : ทิพานัน พงษ์สุวรรณ, อนุพงษ์ ติตะ และ ภานุวัตร อุทัยบาล

ระบบติดตามถังขยะอัจฉริยะ เป็นระบบที่ใช้ในการจัดการปัญหาขยะที่เต็มก่อนกำหนด และมีกลิ่นเหม็น โดยมีการแจ้งเตือนส่วนกลางให้ทราบถึงถังขยะที่ใกล้จะเต็มแล้ว ขยะมีกลิ่นเหม็น หรือตรวจสอบว่าขยะที่เต็มแล้วนั้นถูกวางไว้จุดใด โดยการใช้เซนเซอร์แสงอินฟราเรดติดที่จุดต่างๆ เพื่อวัดปริมาณขยะในถัง การนำเซนเซอร์วัดคุณภาพอากาศเพื่อตรวจสอบกลิ่นเหม็นที่ออกมาจากขยะ จากนั้นจะส่งตำแหน่งของถังขยะ โดยใช้ระบบนำทาง (GPS) ข้อมูลปริมาณและกลิ่นของขยะผ่านทางอุปกรณ์ตรวจจับแล้วนำส่งข้อมูลไปยังเครื่องแม่ข่าย (Server) และมีระบบรายงานแจ้งเตือนทั้งทางเว็บ และแอปพลิเคชันบนมือถือ เพื่อบริหารจัดการและแก้ไขปัญหาขยะในห้องสมุด

บทที่ 3

วิธีการดำเนินงานศึกษาค้นคว้า

3.1 วิธีการดำเนินการวิจัย

3.1.1 ศึกษาค้นคว้าเอกสาร และงานวิจัยที่เกี่ยวข้อง

เป็นขั้นตอนการสืบค้นเอกสาร และงานวิจัยที่เกี่ยวข้องเกี่ยวกับเรื่องที่คณะผู้จัดทำสนใจ และทำการคัดเลือกงานวิจัยที่เกี่ยวข้อง เพื่อนำมาเป็นแนวทางในการทำโครงงานและกำหนดหัวข้อโครงงาน

3.1.2 ศึกษาวิธีการทำงานของ CNN เพื่อมาประยุกต์ใช้กับชุดข้อมูลภาพขยะ

จากการได้ศึกษาเอกสารงานวิจัยที่เกี่ยวข้องกับโครงงาน ในเอกสารงานวิจัยได้มีการใช้ CNN จึงต้องทำการศึกษา CNN เพื่อความเข้าใจและนำไปประยุกต์ใช้ในกระบวนการทำงานในส่วนการสร้าง model

3.1.3 ศึกษาวิธีการใช้งาน Raspberry pi ในการเชื่อมต่อมอเตอร์และ Cloud

เนื่องจากคณะผู้จัดทำต้องการเก็บฐานข้อมูลของภาพขยะที่ทำการทำนายแล้ว ในระยะยาวต้องอาศัยพื้นที่เก็บจำนวนมาก และสะดวกในการจัดการภาพในอนาคต จึงได้ศึกษาการเชื่อมต่อระหว่าง Raspberry Pi ในการเชื่อมต่อมอเตอร์และ Cloud เพื่อความยืดหยุ่นในการเก็บข้อมูลและให้เครื่องมือทำงานร่วมกันได้

3.1.4 มีการออกแบบถังขยะ เพื่อให้ตรงตามการใช้งาน

เป็นขั้นตอนในการออกแบบรูปแบบถังและการออกแบบกระบวนการทิ้งของถังขยะ โดยคำนึงถึงสภาพแวดล้อมการทำงานจริงกับอุปกรณ์ที่ได้มีการติดตั้งให้สามารถดำเนินการได้ตามสภาพแวดล้อมต่างๆ และงบประมาณในการจัดทำถังขยะอัจฉริยะ

3.1.5 ทดสอบการทำงานของถังขยะ

ในขั้นตอนการ Train model ต้องทำการทดสอบประสิทธิภาพของ model เพื่อเทียบประสิทธิภาพในการทำงานจริงออกมาสรุปผลการทำงาน เนื่องจากสถาพแวดล้อมของภาพที่ใช้ในการ Train ไม่เหมือนกับภาพในสถานการณ์จริง จึงอาจทำให้มีการทำนายผิดพลาดได้

3.1.6 สรุปผลการวิจัย เสนอแนะ และจัดทำรูปเล่ม

3.2 แผนการดำเนินงานตลอดโครงงาน

ตารางที่ 1 : แผนการดำเนินงาน

แผนดำเนินงานโครงงานวิจัย								
ขั้นตอนดำเนินการ	ต.ค. 62	พ.ย. 62	ธ.ค. 62	ม.ค. 63	ก.พ. 63	มี.ค. 63	เม.ย. 63	พ.ค. 63
1. เริ่มต้นและวางแผนโครงงาน								
1.1 วางแผนจัดทำโครงงาน								
1.2 หาข้อมูลศึกษาแนวคิดและทฤษฎี								
1.3 เลือกเครื่องมือ								
1.4 ศึกษาระบบงานที่เกี่ยวข้อง								
2. การวิเคราะห์ระบบ								
2.1 ทดลองนำ CNN มา Trian และ Test model โดยใช้ชุดข้อมูล “Trashnet”								
2.2 เก็บรวบรวมชุดข้อมูลภาพขยะใหม่ สำหรับการ นำมา Test model								
2.3 ปรับค่า Learning rate, epochs เพื่อหาค่าที่ทำให้ได้ค่า accuracy ที่ดีที่สุด								
2.4 ลงระบบปฏิบัติการลง Raspberry Pi และนำ model ที่ได้แล้วมาใส่ Raspberry Pi								
3. การออกแบบระบบ								
3.1 ออกแบบตัวถังขยะ								
3.2 ออกแบบระบบการทำงานของถังขยะ								
4. การพัฒนาและติดตั้งระบบ								
4.1 เชื่อมต่อ Raspberry Pi กับ กล้อง, AWS S3, Ultrasonic Sensor และ Arduino								

4.2 ประกอบตัวถังขยะ								
4.3 ทดสอบการใช้งานถังขยะ								
4.4 หาข้อผิดพลาดและแก้ไขปรับปรุง								
4.5 จัดทำเอกสารการติดตั้งระบบและคู่มือการใช้งาน								
5. สรุปและเผยแพร่งานวิจัย								

3.3 อุปกรณ์และเครื่องมือที่ใช้ในการวิจัย

3.3.1 ฮาร์ดแวร์ (Hardware)

1. คอมพิวเตอร์

   <u>คอมพิวเตอร์เครื่องที่ 1</u>

   - ระบบปฏิบัติการ: Windows 10 pro (64-bit)
   - หน่วยประมวผล: Intel Core i5-3470
   - หน่วยความจำ: RAM 16 GB

   <u>คอมพิวเตอร์เครื่องที่ 2</u>

   - ระบบปฏิบัติการ: MacOS Mojave
   - หน่วยประมวผล: Intel Core i5
   - หน่วยความจำ: RAM 8 GB

   <u>Server</u>

   - ระบบปฏิบัติการ: Linux/Ubuntu 18.04.1
   - หน่วยประมวผล: Intel Core i7-8700
   - หน่วยความจำ: RAM 62 GiB

2. Ultrasonic sensor รุ่น HC-SR04
3. Raspberry Pi 4 model B Ram 4 GB
4. Raspberry Pi camera V2.1
5. Arduino Uno R3 ATmega328P
6. Motor
7. Relay 5V 2-CH 10A 250VAC
8. LED strip
9. DRV 8255 Stepping motor Drive

3.3.2 ซอฟแวร์ (Software)

1. Google Colaboratory (ในการ train และ test Model)
2. Jupyter Notebook
3. Amazon Simple Storage Service (Amazon S3)

3.3.3 ภาษาที่ใช้ในการพัฒนา (Language)

1. Python 3.7.3
2. C

3.4 การออกแบบและพัฒนา

3.4.1 การเตรียมชุดข้อมูล (Data Set)

ตารางที่ 2 : ชุดข้อมูล Trashnet

ประเภทของขยะ	จำนวนภาพ ในแต่ละประเภทขยะ
กระดาษ	594
แก้ว	501
พลาสติก	482
กระป๋องโลหะ	410
กระดาษแข็ง	403
ขยะอื่นๆ	137
รวม	**2,527**

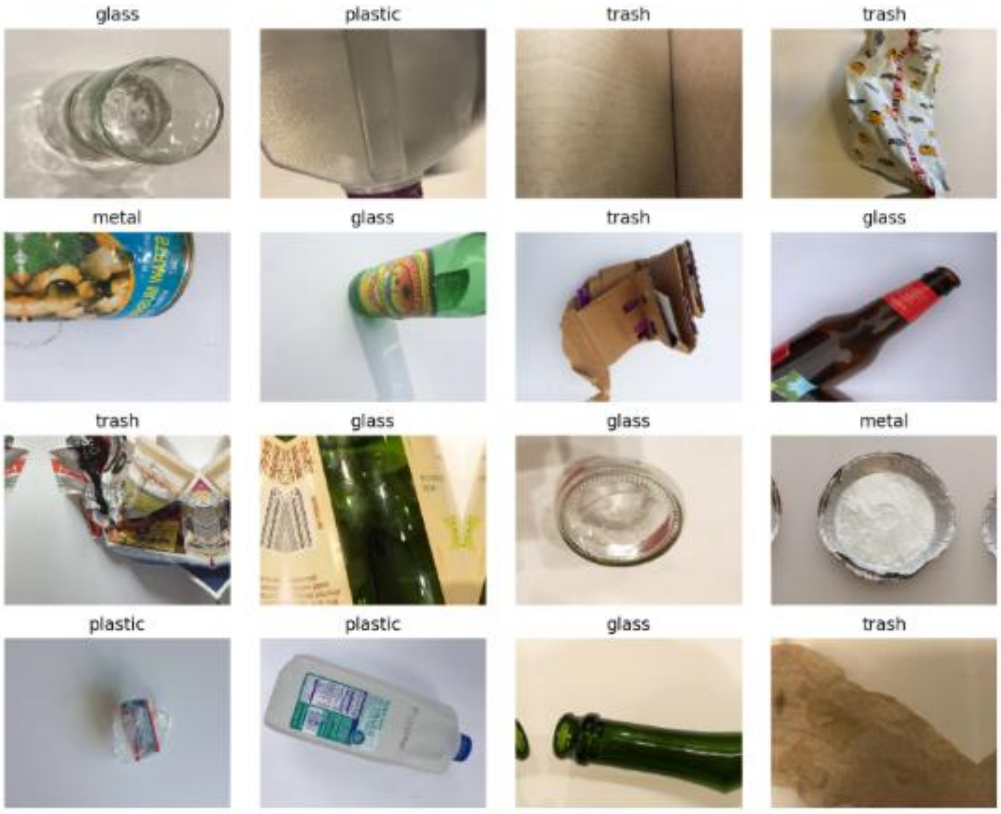
จากตารางที่ 2 เป็นชุดข้อมูล Trashnet ที่มีทั้งหมด 2,527 รูป และแบ่งเป็น 6 ประเภท ได้แก่ กระดาษ แก้ว พลาสติก กระป๋องโลหะ กระดาษแข็ง และขยะอื่นๆ ซึ่งทางคณะผู้จัดทำได้นำชุดข้อมูลดังกล่าวมาทำการดัดแปลง เพื่อให้เหมาะสมกับงานวิจัยนี้ โดยการลดประเภทขยะที่จะคัดแยกให้เหลือ 4 ประเภท ได้แก่ แก้ว พลาสติก กระป๋องโลหะ และขยะอื่นๆ และได้มีการเพิ่มภาพขยะใหม่เข้าไปบางส่วน เพื่อให้ข้อมูลภาพขยะแต่ละประเภทมีจำนวนที่สมดุลกันในทุกประเภท ให้สามารถนำมาใช้งานได้อย่างมีประสิทธิภาพ โดยจะมีการจัดสภาพแวดล้อมของภาพใหม่ให้มีสภาพแวดล้อมใกล้เคียงกับภาพในชุดข้อมูล Trashnet

ตารางที่ 3 : เปรียบเทียบจำนวนภาพในชุดข้อมูล

ประเภทของขยะ	**จำนวนภาพในชุดข้อมูล Trashnet**	**จำนวนภาพในชุดข้อมูล Trashyset**
แก้ว	501	950
พลาสติก	482	950
กระป๋องโลหะ	410	950
ขยะอื่นๆ	1,134	950
รวม	**2,527**	**3,800**

จากตารางที่ 3 แสดงให้เห็นถึงจำนวนภาพในชุดข้อมูล Trashnet ที่ได้ทำการดัดแปลง โดยได้ทำการเพิ่มจำนวนภาพแก้ว 449 รูป, เพิ่มจำนวนภาพพลาสติก 468 รูป, เพิ่มจำนวนภาพกระป๋องโลหะ 540 รูป และลดจำนวนภาพขยะอื่นๆ 184 รูป

โดยจะได้ชุดข้อมูลใหม่คือ Trashyset โดยมีจำนวนทั้งหมด 3,800 รูป โดยมีตัวอย่างดังภาพที่ 11 โดยจะแบ่งข้อมูล ดังตารางที่ 4 ซึ่งจะนำไป ทำการ train ต่อไป เพื่อให้ได้ model ที่สามารถแยกออกมาเป็น 4 คลาส ได้แก่ แก้ว พลาสติก กระป๋องโลหะ และ ขยะอื่นๆ โดยการสุ่มแบ่งชุดข้อมูลภาพเป็นการ Train ร้อยละ 80, Validation ร้อยละ 10 และ Test ร้อยละ 10

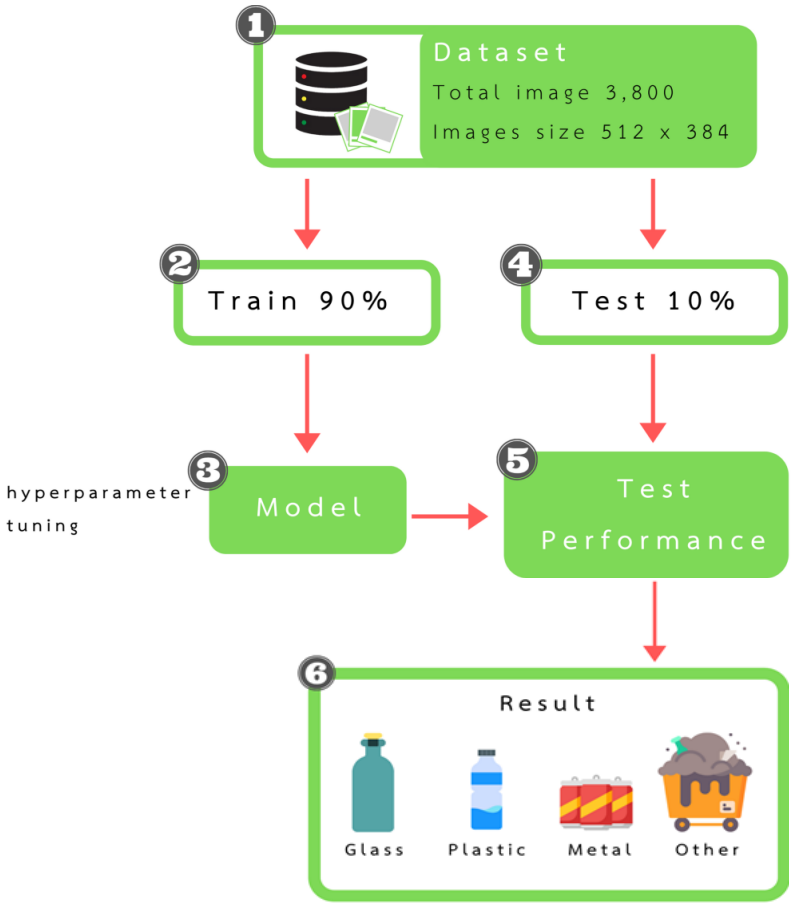


ภาพที่ 11 : ตัวอย่างชุดข้อมูล Trashyset

ตารางที่ 4 : ชุดข้อมูล Trashyset

ประเภทของขยะ	**จำนวนภาพในแต่ละประเภทขยะ**	**Train (80%)**	**Validation (10%)**	**Test (10%)**
แก้ว	950	760	95	95
พลาสติก	950	760	95	95
กระป๋องโลหะ	950	760	95	95
ขยะอื่นๆ	950	760	95	95
รวม	**3,800**	**3,040**	**380**	**380**

3.4.2 การ Train Model



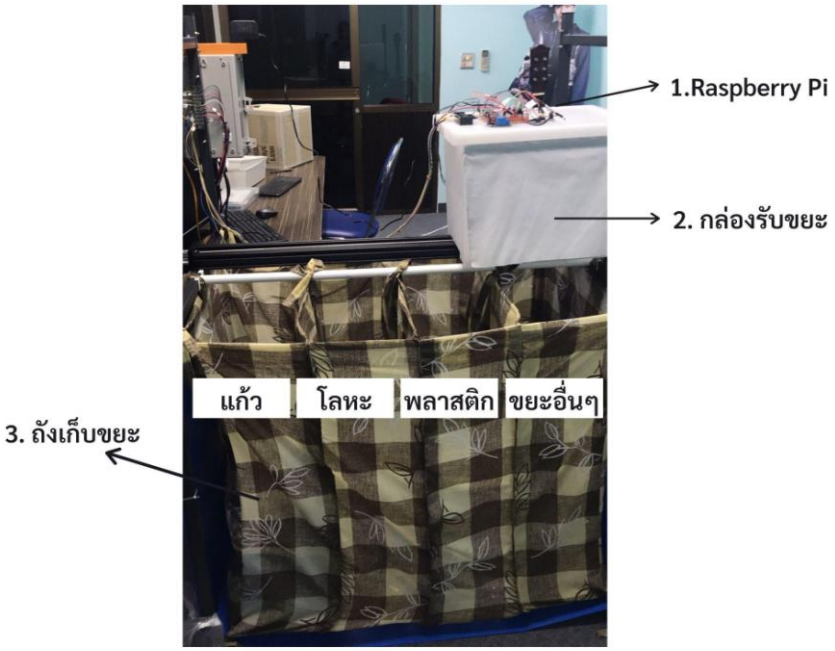
ภาพที่ 12 : กระบวนการในการ train ข้อมูลภาพในการแยกขยะ

จากภาพที่ 12 ได้นำชุดข้อมูล Trashyset จากในตารางที่ 4 จำนวนทั้งหมด 3,800 รูป มาทำการ train เพื่อให้ได้ model ที่สามารถแยกออกมาเป็น 4 คลาส ได้แก่ แก้ว พลาสติก กระป๋องโลหะ และขยะอื่นๆ โดยการสุ่มแบ่งชุดข้อมูลภาพเป็นการ Train ร้อยละ 80, Validation ร้อยละ 10 และ Test ร้อยละ 10 หลังจากนั้นมีการหาค่า Learning rate โดยกำหนดช่วงในการหาค่า โดยอยู่ในช่วง 1e-6 - 1e1 เมื่อได้ค่า Learning Rate ที่เหมาะสม จึงเลือกนำค่านั้นมา Fit method แล้วเพิ่มรอบการคำนวณ ซึ่งเราทำการคำนวณ 40 รอบ เมื่อได้ model แล้วจึงนำ model ไป Test performance ว่ามีประสิทธิภาพในการจำแนกหรือไม่ โดยวัดประสิทธิภาพการทำงานของ model แบบ Confusion matrix แล้วนำ model นั้นไปลง Raspberry pi เพื่อนำไปรอการประมวลผลในการจำแนกขยะให้สามารถทำงานร่วมกับถังขยะที่เตรียมไว้ได้

ในการ train data เราได้มี source code มาจากเว็บ

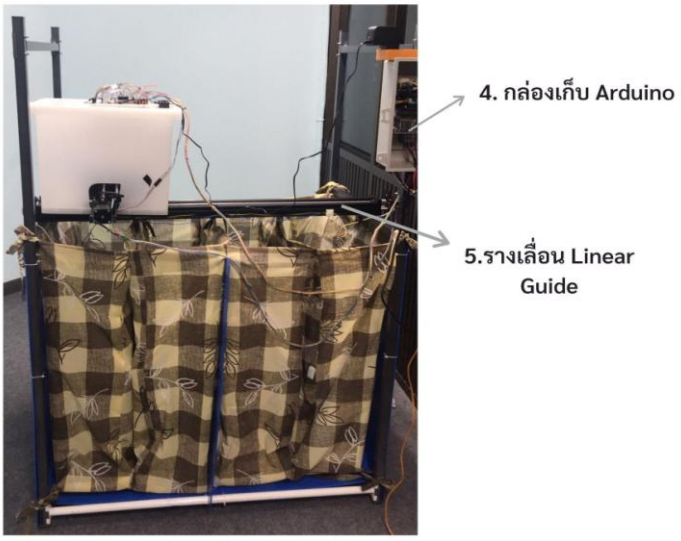
https://github.com/collindching/Waste-Sorter/blob/master/Waste%20sorter.ipynb

3.4.3 โครงสร้างของถังขยะ



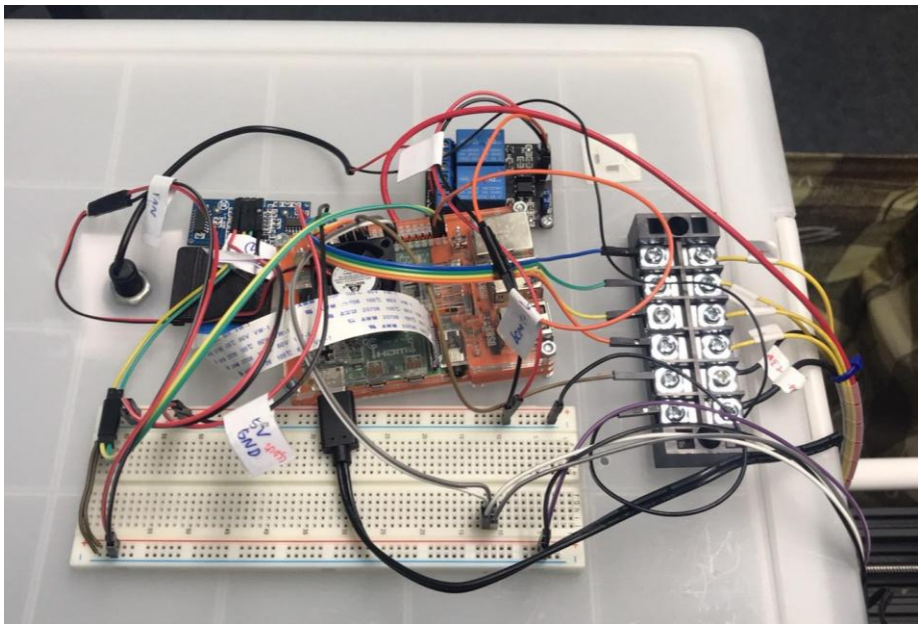
ภาพที่ 13 : ด้านหน้าของถังขยะ

จากภาพที่ 13 ขนาดของถังเก็บขยะแต่ละประเภท กว้าง 25 เซนติเมตร สูง 85 เซนติเมตร ลึก 60 เซนติเมตร และขนาดของกล่องรับขยะ กว้าง 40 เซนติเมตร สูง 30 เซนติเมตร ลึก 20 เซนติเมตร



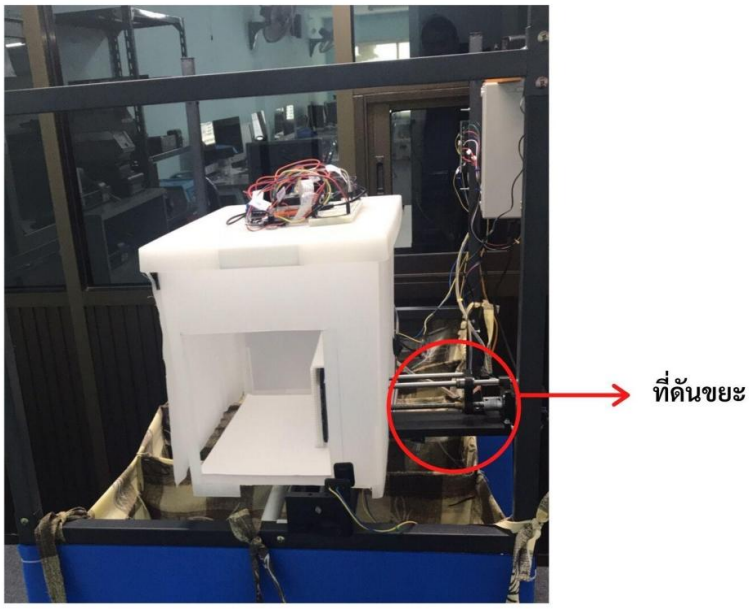
ภาพที่ 14 : ด้านหลังของถังขยะ

1. Raspberry Pi เป็นการต่อวงจร เพื่อควบคุมการทำงานกับอุปกรณ์ต่างๆ ได้แก่ Raspberry Pi camera และการเชื่อมต่อผ่าน pin ได้แก่ พัดลม relay ไฟ LED Ultrasonic sensor 2 ส่วน คือ ส่วนของการตรวจสอบการรับขยะ และวัดปริมาณขยะในถัง ดังภาพที่ 15



ภาพที่ 15 : วงจรการเชื่อมต่อRaspberry Pi

2. กล่องรับขยะ ทำหน้าที่รับขยะจากผู้ใช้มาและนำขยะไปทิ้งโดยการดันขยะลงในถัง ดังภาพที่ 16



ภาพที่ 16 : กล่องรับขยะ

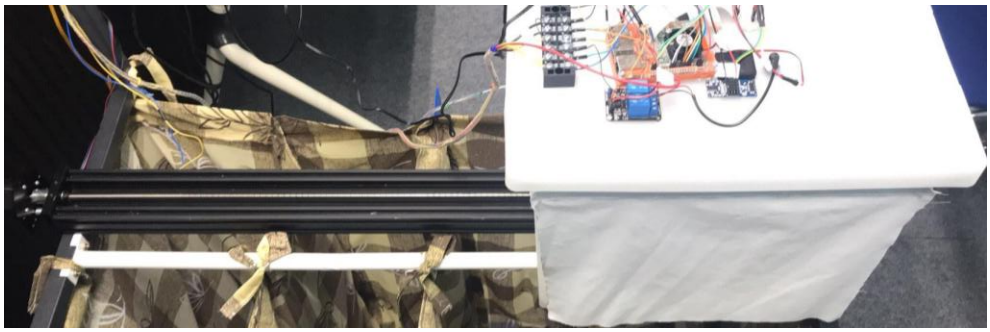
3. ถังเก็บขยะทำหน้าที่เก็บขยะที่คัดแยกแล้ว โดยจะแบ่งเป็น 4 ถัง คือ แก้ว โลหะ พลาสติก และขยะอื่นๆ ดังภาพที่ 17

ภาพที่ 17 : ภายในถังเก็บขยะ

4. กล่องเก็บ Arduino Uno จะเก็บ Arduino Uno, Power supply และ DRV8255 Stepping motor Drive ทั้ง 2 อัน ดังภาพที่ 18

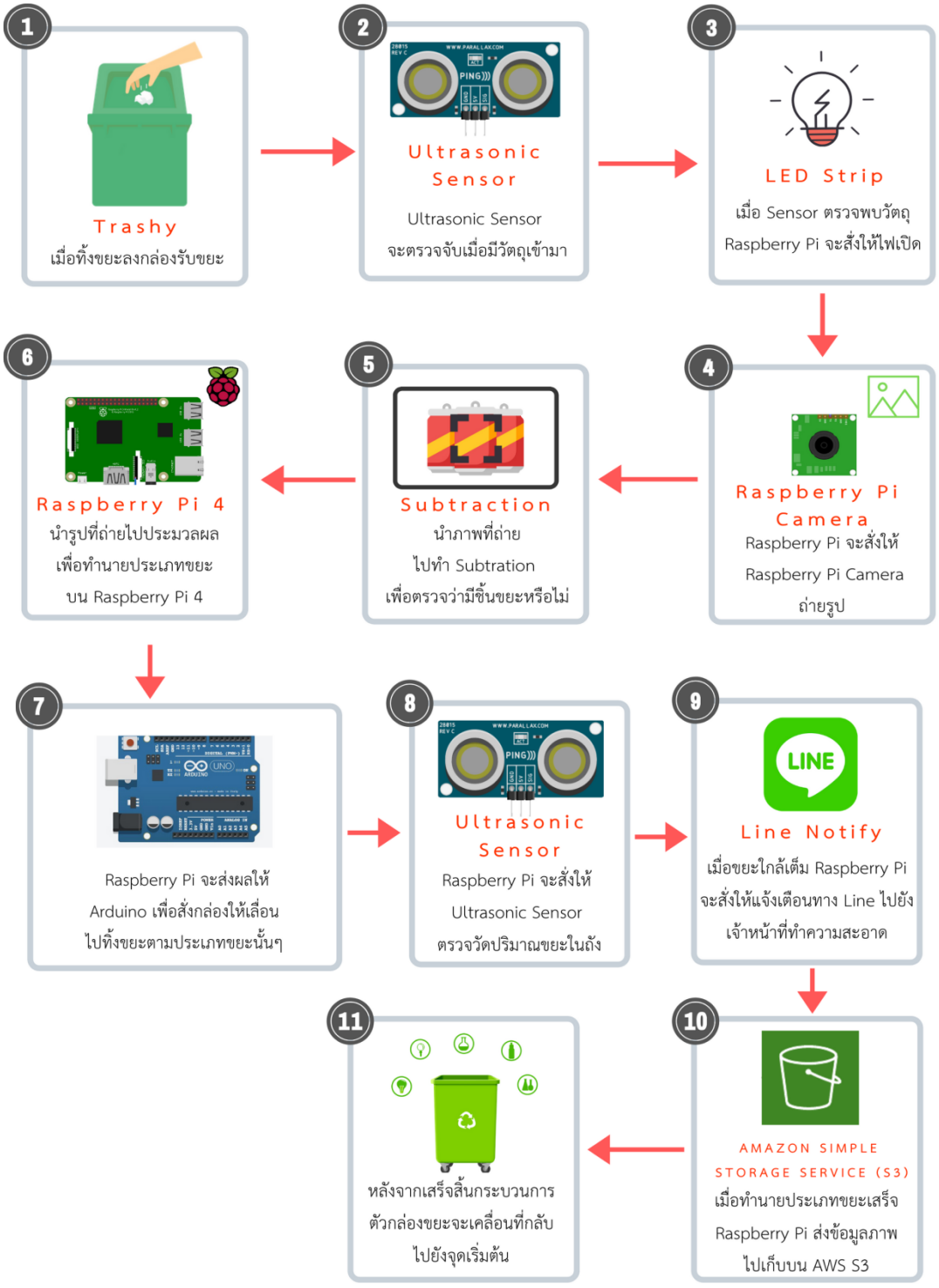
ภาพที่ 18 : ภายในกล่องเก็บ Arduino Uno

5. รางเลื่อน Linear Guide ทำหน้าที่ เลื่อนกล่องรับขยะไปทิ้งยังถังขยะประเภทนั้นๆ ดังภาพที่ 19



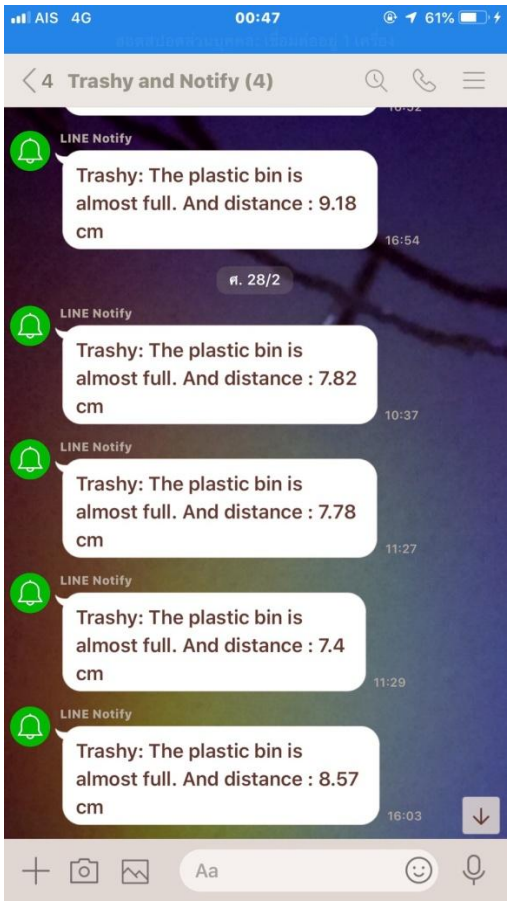
ภาพที่ 19 : รางเลื่อน Linear Guide

3.4.4 ขั้นตอนการทำงานของระบบ



ภาพที่ 20 : ขั้นตอนการทำงานของระบบ

จากภาพที่ 20 แสดงการทำงานของถังขยะที่ติดตั้งอุปกรณ์แล้ว ภายใน 1 ถัง จะมี 4 ช่องเก็บขยะ ตามประเภท ดังนี้ แก้ว พลาสติก กระป๋องโลหะ และขยะอื่นๆ เช่น กระดาษ ถุงพลาสติก ภายในถังจะมีกล่องรับขยะไว้เป็นที่พักเพื่อรอการประมวลผล ซึ่งการทำงานของถังจะเป็นดังนี้ เมื่อมีการทิ้งขยะในกล่องรับขยะ Ultrasonic sensor จะทำการตรวจสอบเบื้องต้นว่ามีขยะเข้ามาในกล่องหรือไม่ เมื่อ sensor ตรวจจับว่ามีวัตถุเข้ามา LED Strip จะเปิดไฟ และกล้อง Raspberry pi camera จะถ่ายภาพเพื่อนำภาพไปตรวจสอบอีกครั้งว่ามีวัตถุเข้ามาจริง เนื่องจากที่ต้องมีการตรวจสอบอีกครั้ง เพราะว่า Ultrasonic sensor ที่กลุ่มวิจัยใช้มีรัศมีการตรวจจับที่แคบทำให้บางจุดในกล่องรับไม่สามารถตรวจจับได้ หลังจากนั้นจะนำภาพที่ถ่ายไปทำนายผลบน Raspberry Pi 4 เมื่อได้ผลลัพธ์ของประเภทขยะมา ตัวกล่องรับจะทำการเลื่อนกล่องไปตามรางแล้วนำขยะไปทิ้งตามประเภทนั้นๆ หลังจากที่ทิ้งขยะเสร็จแล้ว Ultrasonic sensor ที่ติดอยู่ข้างล่างกล่องรับ จะทำการตรวจสอบปริมาณขยะในถังว่าใกล้เต็มหรือไม่ ถ้าใกล้เต็มจะมีการแจ้งให้เจ้าหน้าทราบผ่าน Line Notify ดังภาพที่ 21 หลังจากนั้นจะมีการอัพโหลดรูปภาพขึ้น AWS S3 หลังจากนั้นตัวกล่องรับจะเคลื่อนที่กลับไปยังจุดเริ่มต้นอีกครั้ง

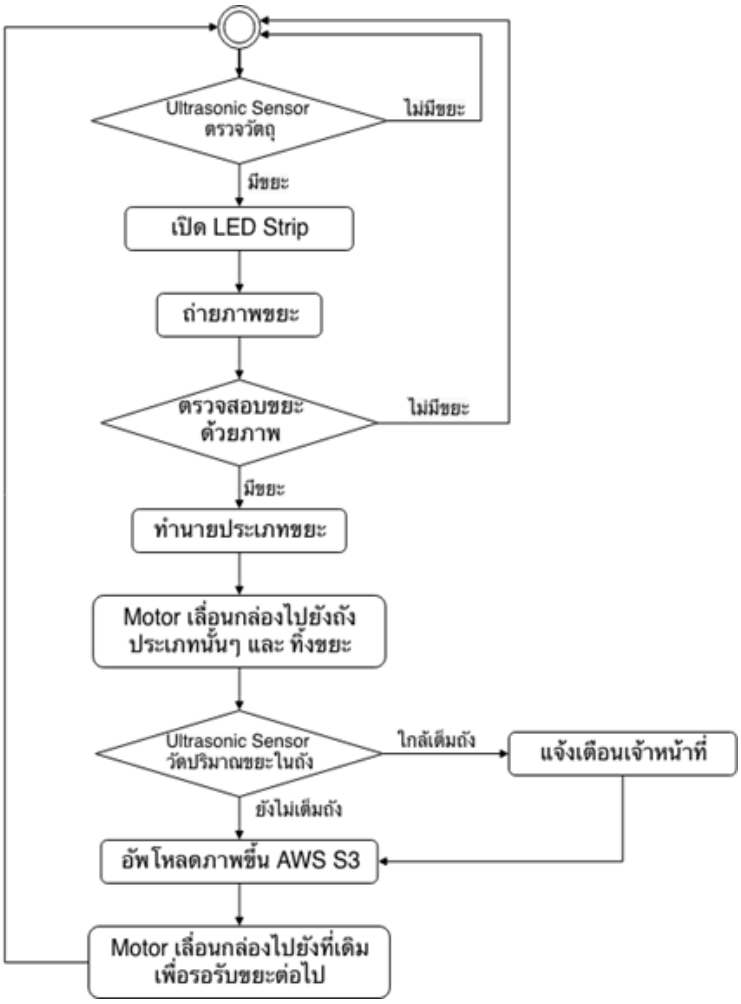


ภาพที่ 21 : Line Notify

3.4.5 การควบคุมการทำงานที่ควบคุมโดย Raspberry Pi

1. รอการตรวจสอบวัตถุด้วย Ultrasonic Sensor โดยจะมีการปล่อยคลื่นออกมาเรื่อยๆ
2. ควบคุมการเปิด LED Strip เมื่อมีวัตถุเข้ามา
3. การถ่ายภาพขยะหลังจากวัตถุเข้ามา 3 วินาที
4. ตรวจสอบขยะจากภาพว่ามีอยู่จริง ถ้าไม่มีจะกลับไปตรวจสอบวัตถุอีกครั้ง
5. การทำนายประเภทขยะ
6. Ultrasonic Sensor วัดปริมาณขยะในถังนั้นๆ ถ้าใกล้เต็มจะมีการส่งแจ้งเตือนให้ทราบผ่าน Line Notify
7. อัพโหลดภาพขึ้นบนฐานข้อมูล AWS S3

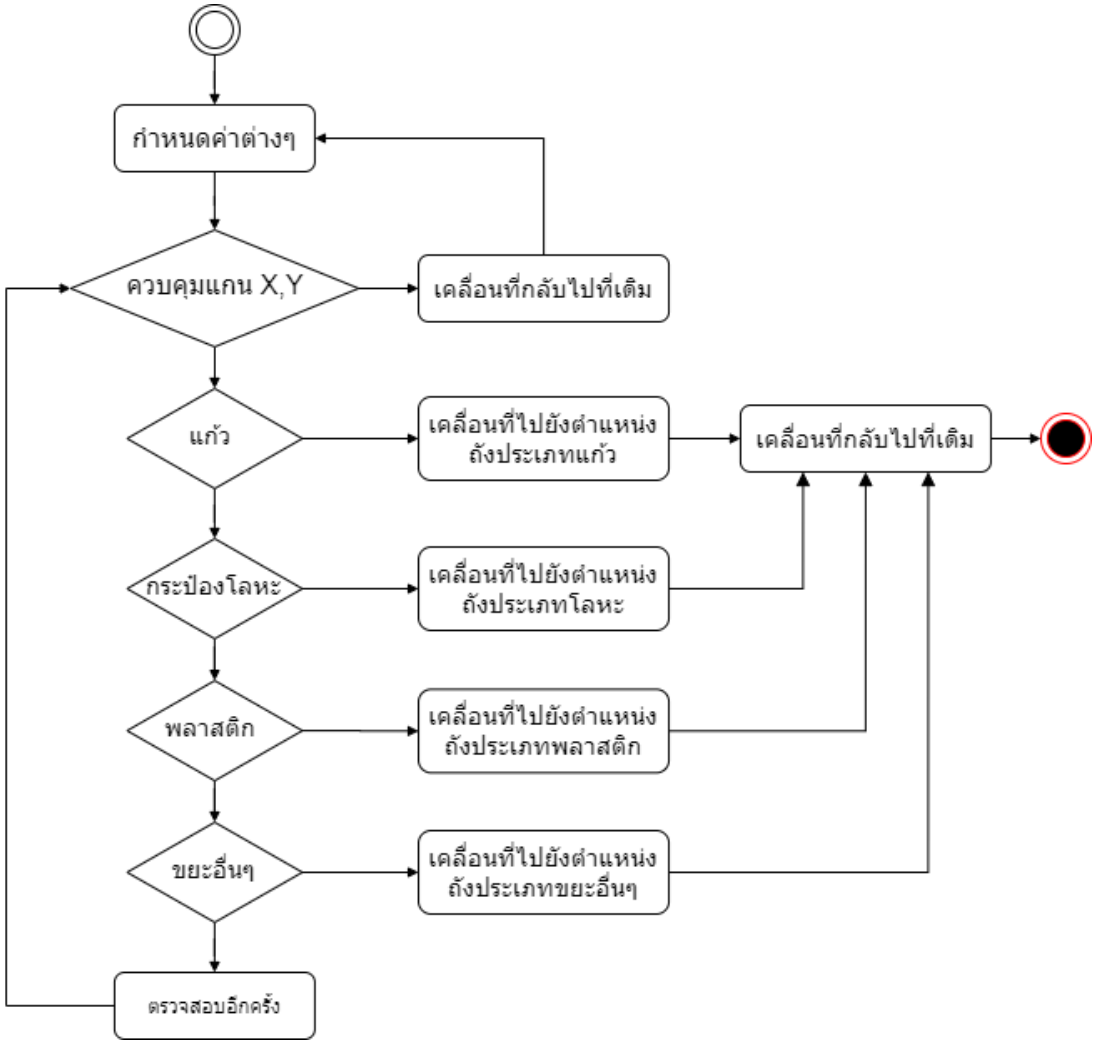
โดยการทำงานภายใน Raspberry Pi เป็นดังภาพที่ 22



ภาพที่ 22 : โครงสร้างการทำงานกับการควบคุม

3.4.6 การควบคุมการทำงานที่ควบคุมโดย Arduino UNO

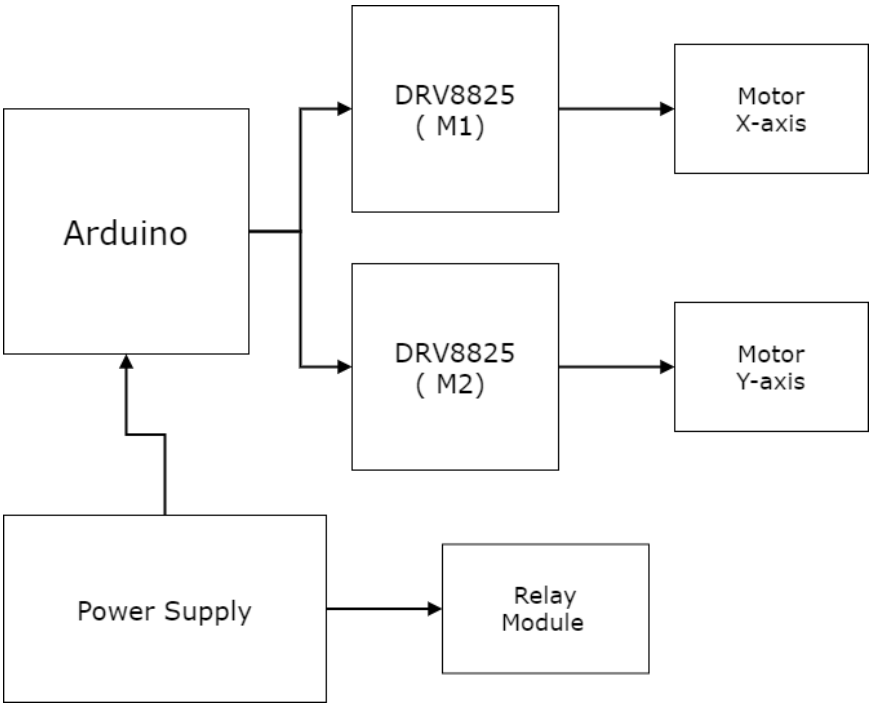
1. ส่วนของ Motor ให้เคลื่อนที่ไปยังถังที่ตรงกับที่ทำนายผลได้
2. การเคลื่อนกลับมาของ Motor ที่จุดเดิมเพื่อมารอรับขยะต่อไป



ภาพที่ 23 : การเคลื่อนที่ของ Motor

จากภาพที่ 23 ระบบการเคลื่อนที่แบบสองแกนซึ่งประกอบไปด้วยแกนแนวนอน (X -axis) และแกนแนวตั้ง (Y-axis) ควบคุมการทำงานด้วยระบบไมโครคอนโทรลเลอร์แบบ 8 บิท ที่รับคำสั่งจากระบบการตรวจจับภาพแบบ Image processing จาก Raspberry Pi ที่ทำนายผลได้

3.4.7 การควบคุมมอเตอร์โดย Arduino



ภาพที่ 24 : วงจรการเชื่อมต่อระหว่าง Arduino และ Motor

Arduino Board

จากภาพที่ 24 คอยควบคุมทิศทางการเคลื่อนที่ ความเร็ว และมีการกำหนดให้ทุกครั้งที่เปิดแกนมอเตอร์ทั้งสองแกนจะกลับไปยังตำแหน่งเริ่มต้นโดยอัตโนมัติ เพื่อลดกรณีการใช้ก่อนหน้าเกิดค้างระหว่างทำงานหรือไฟดับ

DRV8255 Stepping motor Drive

ควบคุมมอเตอร์ให้มีการขับเคลื่อนโดย Arduino ซึ่งส่วนของถังขยะจะมีการขับเคลื่อน 2 แกน ดังนี้

- (M1) แกน X ควบคุมในการเคลื่อนที่ของกล่องรับขยะ
- (M2) แกน Y ควบคุมในการเคลื่อนที่ของตัวดันขยะ

Motor X-axis, Motor Y-axis

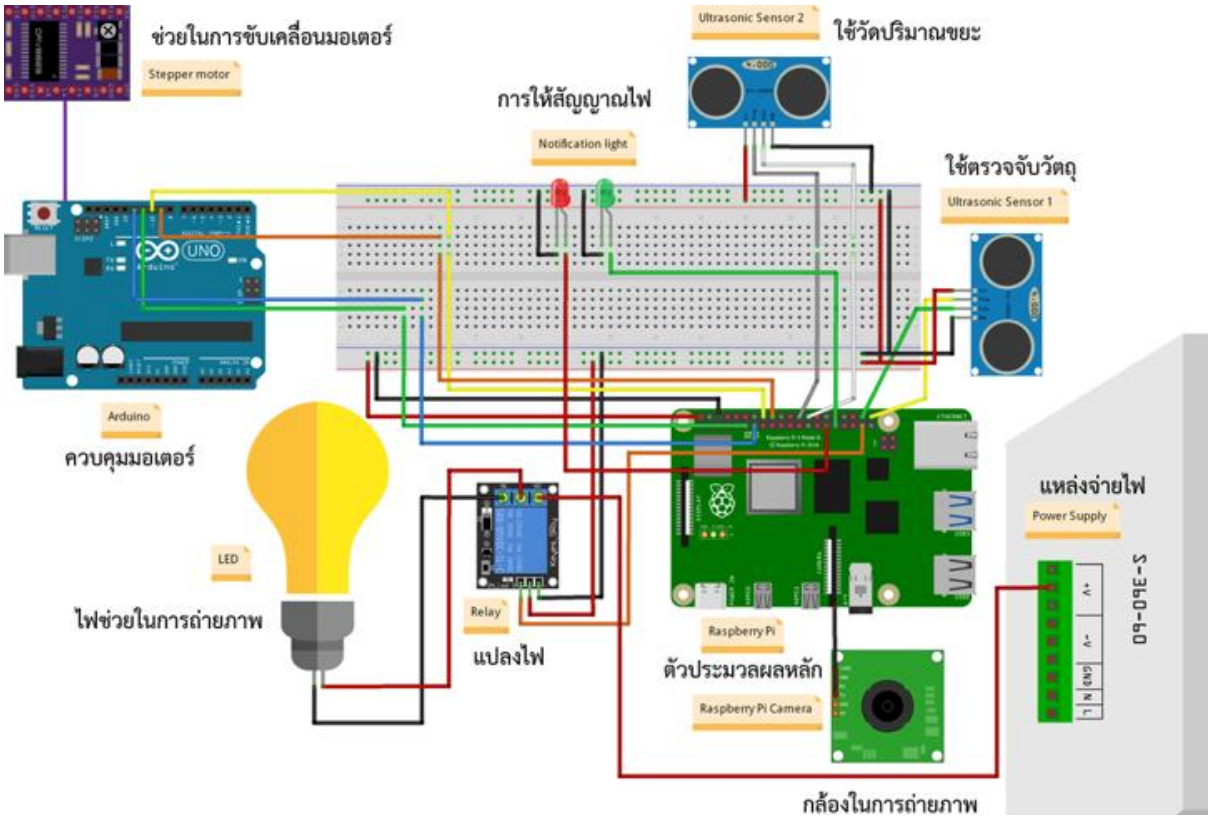
เคลื่อนที่ทวนเข็มนาฬิกาบนแกนลิเนียร์โดย

- แกน X ทำการเลื่อนตามคำสั่งให้ตรงถังที่กำหนด
- แกน Y ทำการเลื่อนตามคำสั่งให้ตัวดันขยะให้ลงถัง

Power supply

แหล่งจ่ายไฟ 12V คอยจ่ายพลังงานให้มอเตอร์ทำงาน จ่ายพลังงานให้ Relay เพื่อนำพลังงานไปเลี้ยง ใช้ในส่วนของไฟ LED ช่วยในการถ่ายภาพ ควบคุมการเปิดปิดโดย Raspberry Pi และ เป็นแหล่งจ่ายไฟให้ Arduino สามารถทำงานได้ ในกล่องแหล่งจ่ายไฟนี้ยังมีสวิตช์ไฟ คอยเปิดปิดการทำงานของมอเตอร์

3.4.8 การเชื่อมต่อวงจรการทำงานหลัก



ภาพที่ 25 : การเชื่อมต่อวงจรบนกล่องรับขยะ

การเชื่อมต่อวงจรการทำงานหลัก จากภาพที่ 25

ในส่วนของ Camera ได้ทำการเชื่อมต่อ Raspberry Pi Camera Module เข้ากับช่อง Camera Module port บน Raspberry Pi เพื่อใช้ในการถ่ายภาพ

ในส่วนของ Ultrasonic Sensor 2 ตัว ได้แก่

- ตัวที่ 1 ทำหน้าที่ในการตรวจจับวัตถุที่เข้ามาในกล่องรับขยะ ที่ PIN Trig - GPIO21 และ Echo - GPIO20
- ตัวที่ 2 ทำหน้าที่ในการวัดปริมาณขยะในถังนั้นๆ ที่ PIN Trig – GPIO8 และ Echo – GPIO7 และ Ultrasonic Sensor ทั้ง 2 ตัว ได้ทำการเชื่อมต่อไฟ 5V และสายดิน

ในส่วนของ Relay และ LED ได้มีการเชื่อมต่อไฟ 10V จาก power Supply และสายดิน โดยการใช้ Relay เพื่อแปลงไฟให้ LED โดยจ่ายไฟในปริมาณ 5V ซึ่ง Relay ได้ทำการแปลงไฟจาก 10V เป็น 5V และมีการกำหนดการทำงานในการเปิด/ปิดไฟ ของ LED ผ่าน GPIO26

ในส่วนของ Notification light คือไฟแจ้งเตือนเพื่อให้ผู้ใช้งานทราบว่าถังขยะพร้อมใช้งานหรือไม่ โดยการกำหนด PIN ไฟเขียวแสดงว่าพร้อมใช้งานที่ GPIO6 และ ไฟแดงแสดงว่าพร้อมใช้งานที่ GPIO5

ในส่วนของการทำงานที่เชื่อมต่อระหว่าง Raspberry Pi กับ Arduino เพื่อให้สามารถส่งผลลัพธ์จากการทำนายไปให้มอเตอร์สามารถทำงานและเคลื่อนที่ไปยังถังนั้นๆ ได้ โดยมีการกำหนด PIN ดังตารางที่ 5

ตารางที่ 5 : การเชื่อมต่อ Pin ระหว่าง Raspberry Pi และ Arduino

ประเภทขยะ	**PIN บน Raspberry Pi**	**PIN บน Arduino**
ขยะอื่น ๆ	GPIO24	GPIO9
พลาสติก	GPIO23	GPIO10
กระป๋องโลหะ	GPIO17	GPIO11
แก้ว	GPIO27	GPIO12

จากการเชื่อมต่อ Raspberry Pi กับ Arduino จะทำงานร่วมกัน หากผลลัพธ์ไปที่ถังขยะประเภทไหน จะมีการกำหนดส่งค่าแรงดันไฟจาก Raspberry Pi โดยจะให้ค่านั้นเป็น LOW แล้วที่เหลือเป็น HIGH เช่น ทำนายผลได้ว่าเป็นแก้ว การกำหนดค่าแรงดันไฟของ PIN ตามลำดับ จะได้ค่า HIGH, HIGH, HIGH, LOW ซึ่งจากนั้น Arduino จะรับแล้วทำการตรวจสอบตามเงื่อนไขแล้วควบคุมมอเตอร์ให้ทำงานต่อไป

บทที่ 4

ผลการดำเนินงาน

4.1 ผลการเปรียบเทียบการ Train model และ Test model

4.1.1 ผลการเปรียบเทียบการ Train model

จากตารางที่ 6 ได้นำชุดข้อมูล Trashyset มาทำการ Train model 10 model ได้แก่ squeezenet1_0, alexnet, vgg16_bn, vgg19_bn, densenet121, ResNet34, ResNet50, ResNet18, ResNet101 และ squeezenet1_1 โดยได้ทำการ Train model model ละ 10 รอบ รวมเป็นการ Train model ทั้งหมด 100 รอบ* ในการ Train model แต่ละรอบ มีรอบการทำงาน 40 รอบการทำงาน (Epochs) พบว่าผลการทดลองในการแบ่งชุดข้อมูลเป็น Train, Validation, Test ที่มีอัตราส่วน 80 : 10 : 10 ของแต่ละ model ส่วนใหญ่มีค่าเฉลี่ย accuracy ที่ค่อนข้างสูงกว่าการแบ่งชุดข้อมูลในอัตราส่วน 50 : 25 : 25 หรือการแบ่งชุดข้อมูลในอัตราส่วน 60 : 20 : 20 และการแบ่งชุดข้อมูลในอัตราส่วน 70 : 15 : 15 จึงได้เลือกการแบ่งข้อมูลในอัตราส่วน 80 : 10 : 10 และพบว่าการแบ่งชุดข้อมูลในอัตราส่วน 80 : 10 : 10 ที่มีค่าเฉลี่ย accuracy ทั้งหมดเป็นค่าเฉลี่ยสูงสุด 3 อันดับ จาก 10 รอบของการ Train model ได้แก่ vgg19_bn ค่าเฉลี่ย accuracy ประมาณร้อยละ 98.59, densenet121 ค่าเฉลี่ย accuracy ประมาณร้อยละ 98.41, ResNet34 ค่าเฉลี่ย accuracy ประมาณร้อยละ 98.07, ResNet50 ค่าเฉลี่ย accuracy ประมาณร้อยละ 98.68, ResNet101 ค่าเฉลี่ย accuracy ประมาณร้อยละ 98.59 และ squeezenet1_1 ค่าเฉลี่ย accuracy ประมาณร้อยละ 92.1 สามารถเรียงค่าค่าเฉลี่ย accuracy จากมากไปน้อยตามลำดับได้ดังนี้ ResNet50, vgg19_bn, Resnet101, densenet121 และ ResNet34 ซึ่งจะเห็นได้ว่า model ทั้ง 5 model มีค่าเฉลี่ย accuracy มากกว่าร้อยละ 98 และจะเห็นได้ว่า Resnet50 มีค่าเฉลี่ย accuracy สูงที่สุด

ตารางที่ 6 : อันดับ model ที่มีค่าเฉลี่ย Accuracy 3 อันดับแรกดีสุด

* หมายเหตุ Model ทั้งหมดอยู่ในภาคผนวก ค ตารางที่ ค - 1

Model	ค่าเฉลี่ย Accuracy (%)	Model size
ResNet50	98.68	98 MB
vgg19_bn	98.59	82.4 MB
Resnet101	98.59	179 MB
densenet121	98.41	32.6 MB
ResNet34	98.07	87.4 MB

จากตารางที่ 7 แสดงข้อมูลการ Train model ResNet50 ทั้ง 3 รอบ

ตารางที่ 7 : การ Train model ResNet50 ทั้ง 3 รอบ

Model	Learning rate	Accuracy (%)	Training time (min)	Model size
ResNet50	7.41E-04	98.15	30.48	98 MB
ResNet50	8.32E-04	99.21	30.52	98 MB
ResNet50	5.75E-04	98.68	30.65	98 MB

4.1.2 ผลการเปรียบ model ที่ได้ทำการ Test กับ ภาพขยะจริง

หลังจากการ Train model ทางคณะผู้จัดทำได้ Export model ของการ Train แต่ละครั้ง ที่มีนามสกุลไฟล์คือ .pkl เพื่อนำไปใช้ในการทดสอบกับขยะจริง โดยได้ทำการทดสอบจำนวน 100 ครั้ง** เท่ากับจำนวนการ Train ทั้งหมด ในแต่ละ model ได้มีการทดสอบกับภาพทดสอบทั้งหมด 100 รูป ได้แก่ แก้ว 25 รูป, กระป๋องโลหะ 25 รูป, พลาสติก 25 รูป และ ขยะอื่นๆ 25 รูป ซึ่งรูปทั้งหมดได้มาจากการถ่ายภาพในขณะที่ทดลองในถังขยะจริง จากตารางที่ 8 แสดงให้เห็นว่า Model Rasnet50 ในการทดลองรอบที่ 68 สามารถทำนายจากภาพขยะจริงได้ถูกต้องมากที่สุดที่ร้อยละ 84 โดยทำนายถูกต้องที่ แก้ว 23 รูป, กระป๋องโลหะ18 รูป, พลาสติก 22 รูป และขยะอื่นๆ 21 รูป

ตารางที่ 8 : ผลของการนำ model มาทำการ Test กับ ภาพขยะจริง

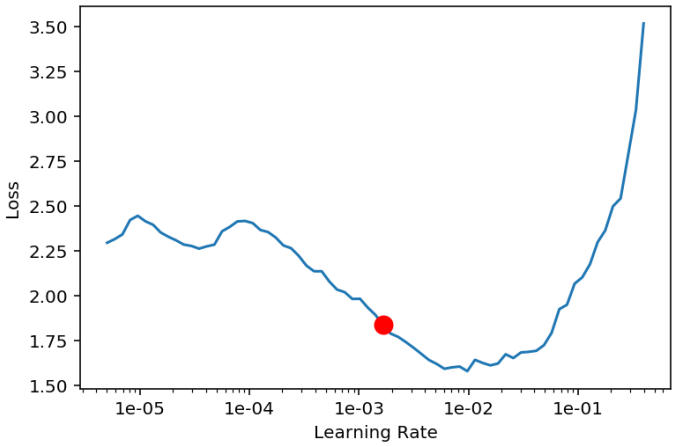
**หมายเหตุ Model ทั้งหมดอยู่ในภาคผนวก ค ตารางที่ ค – 2

No.	Model	Glass (25 images)	Metal (25 images)	Plastic (25 images)	Trash (25 images)	Accuracy (%)
68	ResNet50	23	18	22	21	84
69	ResNet50	20	19	22	19	80
70	ResNet50	21	16	22	18	77

4.2 ผลการ Train และ Test model ที่ถูกเลือก

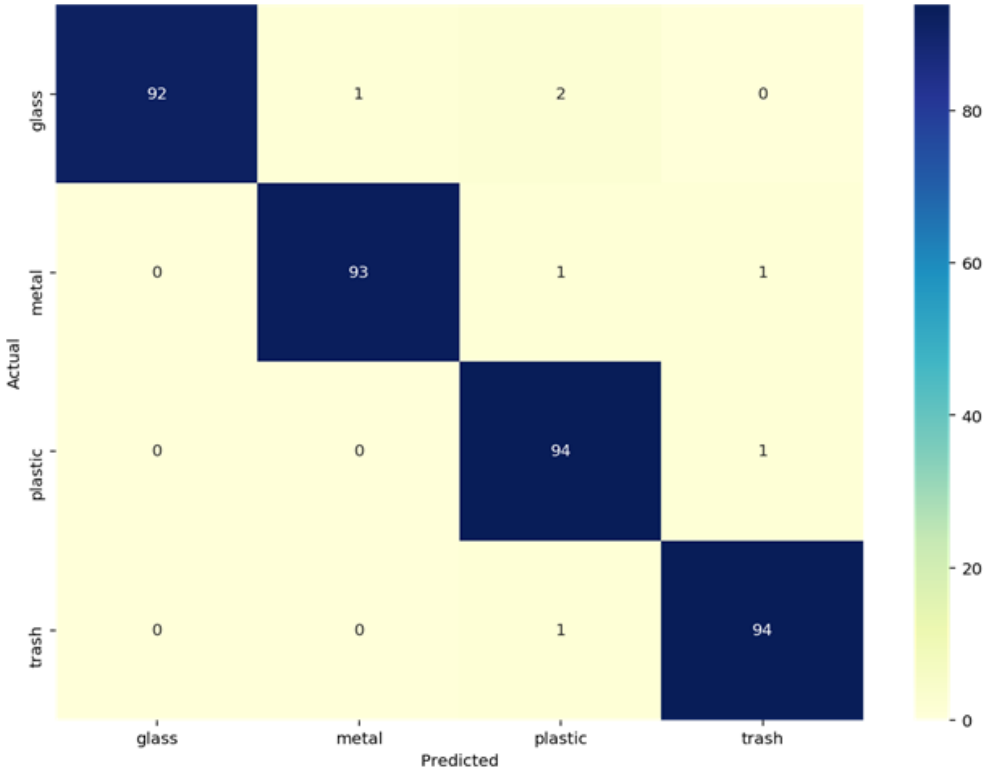
4.2.1 ผลการ Train model

Model ที่ทางคณะผู้จัดทำได้เลือกมา คือ ResNet50 ซึ่งมีโครงสร้างของ ResNet ที่ประกอบไปด้วยประมาณ 50 layers โดยเราได้มีการหาค่า Learning Rate สำหรับ Gradient descent เพื่อทำให้มั่นใจว่ารวดเร็วเพียงพอและมีโอกาสผิดพลาดน้อย และได้ทำการหาค่าที่ช่วง 1e-6 - 1e1 ดังภาพที่ 26 จึงได้ค่าที่เหมาะสม คือ ค่าที่ 7.41E-04



ภาพที่ 26 : การหา learning Rate ได้ค่าที่เหมาะสม 7.41E-04

เมื่อได้ model มาแล้วจึงได้นำ model ไป test performance ว่ามีประสิทธิภาพในการจำแนกหรือไม่ โดยวัดประสิทธิภาพการทำงานของ model แบบ Confusion matrix (Accuracy) ดังภาพที่ 27 จะได้ค่าความถูกต้องจากการ Test ร้อยละ 98.15



ภาพที่ 27 : Confusion matrix จากการทดสอบ บน Server

จากภาพที่ 27 แสดงให้เห็นถึงการทำนายขยะผิดจำนวน 7 รูป จากขยะทั้งหมดสำหรับการทดสอบ 380 รูป ได้แก่

- แก้ว ทำนายเป็น กระป๋องโลหะ 1 รูป พลาสติก 2 รูป
- กระป๋องโลหะ ทำนายเป็น พลาสติก 1 รูป ขยะอื่นๆ 1 รูป
- พลาสติก ทำนายเป็น ขยะอื่นๆ 1 รูป
- ขยะอื่นๆ ทำนายเป็น พลาสติก 1 รูป

ซึ่งจะเห็นได้ว่า จากการทดสอบการทำนายนี้ยังมีความผิดพลาดในการทำนายผลอยู่ประมาณร้อยละ 2

4.2.2 ผลการ Test model กับภาพขยะจริง

ในการ Test กับ ภาพขยะจริง มีตัวอย่างชุดข้อมูลจริง ดังภาพที่ 28

ภาพที่ 28 : ตัวอย่างชุดข้อมูลจริง

จากผลการทดลองการเปรียบเทียบ model ดังในตารางที่ 9 คณะผู้จัดทำได้เลือก model ที่ดีที่สุดมาซึ่งมีค่าความถูกต้องร้อยละ 84

ตารางที่ 9 : ผลลัพธ์จากการทดสอบภาพขยะจริง ในรูป Confusion matrix

ค่าที่ทำนายได้ / ค่าจริง	แก้ว	พลาสติก	กระป๋องโลหะ	ขยะอื่นๆ
แก้ว	23	1	-	1
พลาสติก	3	22	-	-
กระป๋องโลหะ	-	2	18	5
ขยะอื่นๆ	1	3	-	21

จากตารางที่ 9 แสดงให้เห็นผลลัพธ์จากการทดสอบภาพขยะจริงทั้งหมด 100 ชิ้น มีการทำนายขยะผิดทั้งหมด 16 รูป โดยแบ่งเป็น

- แก้ว 25 รูป ทำนายผิดเป็น พลาสติก 1 รูป ขยะอื่นๆ 1 รูป
- พลาสติก 25 รูป ทำนายผิดเป็น แก้ว 3 รูป
- กระป๋องโลหะ 25 รูป ทำนายผิดเป็น พลาสติก 2 รูป ขยะอื่นๆ 5 รูป
- ขยะอื่นๆ 25 รูป ทำนายผิดเป็น แก้ว 1 รูป พลาสติก 3 รูป

ซึ่งจะเห็นได้ว่า จากการทดสอบการทำนายนี้ยังมีความผิดพลาดในการทำนายผลอยู่ประมาณร้อยละ 16

จากผลลัพธ์การ Train model และการนำ model ไปใช้กับถังขยะจริง แสดงถึงค่า accuracy ที่ต่างกันมาก ซึ่งอาจเกิดจากสภาพแวดล้อมการถ่ายภาพขยะในถังจริงนั้นต่างจากภาพในการ Train model โดยภาพที่ถ่ายจากถังขยะจริงอาจมีแสงจากภายนอกที่เข้ามาในกล่องรับ เงาตกกระทบ แสงสะท้อน และเกิดจากการที่ภาพที่นำมา ทดสอบกับถังจริงบางมุมอาจคล้ายกับขยะอีกประเภท รูปทรงของขยะบางชิ้นเป็นทรงคล้าย ซึ่งส่งผลให้การทำนายผิดพลาดไป ดังจะเห็นได้จากภาพที่ 29 – 32

ภาพที่ 29 : ภาพกระป๋องที่ทำนายผิด

จากภาพที่ 29 เป็นภาพโลหะ ทำนายผิดเป็น พลาสติก เพราะภาพในมุมนี้ถ่ายเห็นชิ้นขยะบางส่วน มีสีที่ไม่โดดเด่นจากพื้นหลัง และภาพโลหะ ทำนายเป็น ขยะอื่นๆ เพราะอาจเกิดจากสีของกระป๋องมีความเข้มเวลาแสงตกกระทบกับกระป๋องโลหะทำให้เกิดความวาวน้อยกว่าภาพที่ทำการ Train หรือกระป๋องที่มีสีอ่อนกว่า

ภาพที่ 30 : ภาพพลาสติกที่ทำนายถูกและผิด

จากภาพที่ 30 พลาสติก ทำนายผิดเป็นแก้ว เพราะพลาสติกบางชิ้นมีลักษณะใสมีความมันวาวคล้ายแก้ว และตำแหน่งของภาพอาจจะที่เห็นภาพเพียงส่วนน้อยทำให้เห็นบางส่วนคล้ายแก้วบวกกับพื้นหลังที่มีความกลมกลืนกับชิ้นส่วนขยะทำให้ทำนายผิดได้

ภาพที่ 31 : ภาพแก้วที่ทำนายถูกและผิด

จากภาพที่ 31 เป็นภาพแก้ว ทำนายผิดเป็น พลาสติก เพราะถ่ายมุมที่ไม่เห็นรูปทรงของแก้วหรือส่วนของแก้ว และภาพแก้ว ทำนายผิดเป็น ขยะอื่นๆ เพราะภาพเห็นส่วนใหญ่ที่เป็นฉลาก ทำให้ไม่เห็นความวาวของแก้ว อาจทำให้ทำนายผิดได้

ภาพที่ 32 : ภาพขยะอื่นๆ ที่ทำนายถูกและผิด

จากภาพที่ 32 แสดงภาพขยะอื่นๆ ทำนายผิดเป็น แก้ว เพราะไม่มีชุดข้อมูลภาพที่คล้ายกับภาพหรือมีข้อมูลไม่เพียงพอในตอน Train และขยะอื่นๆ ทำนายผิดเป็น พลาสติก เพราะ ส่วนใหญ่ของขยะมีวัสดุเป็นพลาสติก

4.3 สรุปผลเวลาที่ใช้ในการทำงานทั้งหมดของถังขยะ

จากการนำ Model ที่เราเลือกใส่ลง Raspberry Pi เพื่อให้เกิดการประมวลผลในการทำนายประเภทของขยะ แล้วเชื่อมต่อ Arduino เพื่อควบคุมการทำงานของมอเตอร์ ให้ถังขยะสามารถใช้งานได้อย่างอัตโนมัติ โดยในการทำงานตั้งแต่เริ่มต้นจนสิ้นสุดของขยะแต่ละประเภทจะใช้เวลาในการทิ้งขยะที่ต่างกันไป ดังตารางที่ 10

ตารางที่ 10 : ผลเวลาที่ใช้ในการทำงานทั้งหมดของถังขยะแต่ละถัง

ประเภทของขยะ	**การเคลื่อนที่ไป และกลับยังถัง (วินาที)**	**ทั้งหมด (วินาที)**
ขยะอื่นๆ	10	21
พลาสติก	18	29
กระป๋องโลหะ	28	39
แก้ว	42	53

จะเห็นได้ว่าจะใช้เวลาค่อนข้างเยอะ หากถังขยะอยู่ไกลจากตำแหน่งรับขยะมาก ซึ่งจากเวลาในการทำงานทั้งหมด จะมีการทำงานย่อยที่เวลาที่เท่ากันคือ การถ่ายภาพขยะในกล่องเป็นเวลา 3 วินาที, การทำนายผลในการจำแนกขยะเป็นเวลาประมาณ 5 - 6 วินาที, การดันขยะให้ลงถังเป็นเวลา 4 วินาที ส่วนที่เหลือคือการทำงานของมอเตอร์ที่แตกต่างกันไป ขึ้นอยู่กับระยะทางไปยังถังขยะประเภทนั้นๆ

บทที่ 5

สรุปผลการดำเนินงาน

5.1 สรุปผลการดำเนินงาน

จากการพัฒนาถังขยะ Trashy : An IoT - based Smart Recycling Bin System ได้มีการนำ IoT มาใช้ร่วมกับ AI เพื่อคัดแยกขยะ ที่สามารถนำไปรีไซเคิลได้โดยแบ่งเป็น 4 ประเภท ได้แก่ แก้ว พลาสติก กระป๋องโลหะ และขยะอื่นๆ โดย model ที่ทางคณะผู้จัดทำได้เลือกมาคือ ResNet50 ในช่วงค่าที่เหมาะสมที่ค่า 7.41E-04 ซึ่งเป็น model ที่ดีที่สุดจากการทดลองทั้ง 10 model ในส่วนของการ Train model นั้นได้ ใช้ชุดข้อมูล Trashyset โดยมีการนำข้อมูลจาก Trashnet มาปรับปรุง โดยมีการคัดแยกภาพบางส่วน เพิ่มภาพใหม่เข้าไป นำมาหา model ที่เหมาะสมในการนำมาใช้ในการจำแนกขยะ พบว่าในการแบ่งข้อมูล Train, Validation และ Test ในอัตราส่วน 80 : 10 : 10 มีค่าประสิทธิภาพการเรียนรู้ขยะที่ดีในการจำแนกขยะได้อย่างถูกต้องที่ร้อยละ 98 และเมื่อนำไปประยุกต์ใช้จริงสามารถจำแนกได้ถูกต้องที่ประมาณร้อยละ 84 ซึ่งเป็นค่าที่มากกว่า model อื่นๆ และเวลาที่ใช้ในการทำนายผลโดยประมาณ 5 - 6 วินาที ซึ่งเป็นเวลาที่ค่อนข้างน่าพอใจ

หลังจากได้ model ก็ได้นำไปติดตั้งใน Raspberry Pi 4 ซึ่ง Raspberry Pi จะติดต่อกับ Arduino Uno ในการควบคุม motor เพื่อนำขยะไปทิ้งยังถังขยะนั้นๆ ได้อย่างอัตโนมัติ โดยเวลาที่ใช้ในการนำขยะไปทิ้งยังถังขยะประเภทนั้นๆ และเคลื่อนที่กลับมายังจุดเริ่มต้น เพื่อให้พร้อมใช้งานในรอบต่อไป มีการใช้เวลามากที่สุดที่ 53 วินาที ซึ่งเป็นเวลาที่ยังค่อนข้างช้าอยู่

5.2 ปัญหาและอุปสรรคในการดำเนินงาน

ตารางที่ 11 : ปัญหาและแนวทางการแก้ไข

ปัญหา	แนวทางการแก้ไข
1. เนื่องจากภาพที่ใช้ในการ Train model ต่างจากภาพที่ใช้ในการทำ Field test โดยเกิดจากสภาพแวดล้อมที่ต่างกันในขณะที่ถ่ายรูปภาพ เช่น แสงจากภายนอก แสงสะท้อน เป็นต้น จึงทำให้ยังมีการทำนายผิดพลาดได้เมื่อนำไปใช้กับเครื่องจริง	● อาจมีการเพิ่มเซนเซอร์ให้มีการวัดน้ำหนักของวัตถุ ● อาจมีการปรับแสงหรือสีของภาพถ่าย
2. มีการทำงานหลายขั้นตอน เช่น การถ่ายภาพเพื่อตรวจจับวัตถุที่เข้ามา การประมวลผลการคัดแยกประเภทขยะ การเคลื่อนที่ของกล่อง จึงทำให้ถังขยะยังใช้เวลามากในการทำงานทั้งหมด โดยเวลามากสุดที่ประมาณ 53 วินาที	● อาจมีการปรับลดขั้นตอนบางอย่างลง ● อาจปรับจูล model ให้ประมวลผลได้ไวกว่านี้
3. การทำงานร่วมกับระบบคลาวด์ไม่เป็นไปตามแผนที่กำหนดไว้ คือจะให้ model ประมวลผลบนคลาวด์ จึงมีการปรับแผนซึ่งทำให้งานล่าช้า	● ปรับให้ model มาประมวลผลที่ Raspberry Piแทน และลดให้เหลือแค่การเก็บ Dataset ไว้ที่ AWS S3
4. เซนเซอร์ตรวจจับวัตถุที่เข้ามามีความคลาดเคลื่อน ซึ่งทำให้เกิดความผิดพลาดของการใช้งานได้	● ปรับให้มีการการถ่ายภาพ แล้วนำไปทำ subtraction เพื่อตรวจจับวัตถุที่เข้ามาอีกครั้ง
5. ในระหว่างการพัฒนา Raspberry Pi camera เกิดชำรุด	● ได้มีการส่งเคลมกับทางร้าน

5.3 ข้อเสนอแนะ

จากการพัฒนาโครงงานแทรชชี่ ถังขยะอัจฉริยะเพื่อการรีไซเคิล (Trashy : An IoT - based Smart Recycling Bin System) เพื่อช่วยในการแยกขยะให้สามารถนำกลับไปใช้ใหม่ได้อย่างมีประสิทธิภาพ โดยตัวถังจะมีการคัดแยกขยะ เพื่อนำไปรีไซเคิลเป็น 4 ประเภท ได้แก่ แก้ว พลาสติก กระป๋องโลหะ และขยะอื่นๆ โดยmodel ที่ทางคณะผู้จัดทำเลือกใช้คือ ResNet50 ซึ่งมีการนำชุดข้อมูลมาจากชุดข้อมูล “Trashnet” และได้มีการนำมาปรับปรุง โดยการคัดแยกภาพบางส่วน และเพิ่มภาพใหม่เข้าไป เพื่อให้เหมาะสมกับโปรเจคนี้ จากการ train model ได้ค่าประสิทธิภาพการรู้จำขยะที่ดีที่สุดด้วยประสิทธิภาพร้อยละ 98 และเวลาที่ใช้ในการทำนายผลโดยประมาณ 6 วินาที ซึ่งเป็นค่าที่ค่อนข้างพอใจ และอาจมีการพัฒนาให้สามารถทำนายผลให้รวดเร็วมากขึ้นต่อไป

บรรณานุกรม

[1] กรมควบคุมมลพิษและกระทรวงทรัพยากรธรรมชาติและสิ่งแวดล้อม. (2559). **แผนแม่บท การบริหารจัดการขยะมูลฝอยของประเทศ พ.ศ. 2559 – 2564.** สืบค้นเมื่อ 21 พฤศจิกายน 2562, จาก http://infofile.pcd.go.th/waste/WasteMasterPlan.pdf?CFID=2748363&CFTOKEN=86290581

[2] กรมควบคุมมลพิษ. (2560). **รายงานสถานการณ์ขยะมูลฝอยชุมชนของประเทศไทย ปี 2559 สำนักจัดการกากของเสียและสารอันตราย กรมควบคุมมลพิษ.** สืบค้นเมื่อ 21 พฤศจิกายน 2562, จาก http://infofile.pcd.go.th/waste/wsthaz_annual59.pdf?CFID=2568453&CFTOKEN=12162690

[3] Aggregated news around AI and co. (2561). **มาลองดูวิธีการคิดของ CNN กัน.** สืบค้นเมื่อ 16 พฤศจิกายน 2562, จาก https://mc.ai/มาลองดูวิธีการคิดของ-cnn-ก/

[4] Sanparith Marukatat. (2560). **สนุกกับ Neural Network (2).** สืบค้นเมื่อ 24 พฤศจิกายน 2562, จาก https://medium.com/@sanparithmarukatat/%E0%B8%AA%E0%B8%99%E0%B8%B8%E0%B8%81%E0%B8%81%E0%B8%B1%E0%B8%9A-neural-network-2-11a7194ed 236

[5] Chatchawan Niyomthum. (2562). **Convolution Neural Network คืออะไร.** สืบค้นเมื่อ 14 เมษายน 2563, จาก https://medium.com/@pradyasin/what-is-convolution-neural-network-bf2e525089f5

[6] Keng Surapong. (2562). **BatchNorm คืออะไร.** สืบค้นเมื่อ 14 เมษายน 2563, จาก https://www.bualabs.com/archives/2617/what-is-batchnorm-teach-batch-normalization-train-machine-learning-model-deep-convolutional-neural-network-convnet-ep-5/

[7] Keng Surapong. (2562). **ReLU Function คืออะไร.** สืบค้นเมื่อ 14 เมษายน 2563, จาก https://www.bualabs.com/archives/1355/what-is-relu-function-why-popular-deep-learning-training-deep-neural-network-activation-function-ep-3/

[8] Amazon Web Services. (2545). **การประมวลผลบนระบบคลาวด์ด้วย AWS.** สืบค้นเมื่อ 16 พฤศจิกายน 2562, จาก https://aws.amazon.com/th/what-is-aws/

[9] Amazon Web Services. (2545). **Amazon S3.** สืบค้นเมื่อ 16 พฤศจิกายน 2562, จาก https://aws.amazon.com/th/s3/?nc2=h_ql_prod_st_s3

[10] ThaiEasyElec. (2559). **บทความการพัฒนาโปรแกรมบน Raspberry Pi ด้วย Qt.** สืบค้นเมื่อ 16 พฤศจิกายน 2562, จาก https://www.thaieasyelec.com/article-wiki/embedded-electronics-application/raspberry-pi- programming-with-qt-ch1.html

[11] ผศ.ดร.พิมพ์เพ็ญ พรเฉลิมพงศ์ , ดร.นวภัทรา หนูนาค. (2557). **Ultrasonic sensor/เซนเซอร์ชนิดใช้เสียงหรือเซนเซอร์ชนิดอัลตราโซนิก.** สืบค้นเมื่อ 16 พฤศจิกายน 2562, จาก http://www.foodnetworksolution.com/wiki/word/4348/ultrasonic-sensor-เซนเซอร์ชนิดใช้เสียง-หรือเซนเซอร์ชนิดอัลตราโซนิก.

[12] Thaieasyele. (2563). **Arduino Uno R3.** สืบค้นเมื่อ 15 เมษายน 2563, จาก https://www.thaieasyelec.com/arduino-uno-r3.html

[13] Geng. (2560). **Line notify**. สืบค้นเมื่อ 24 เมษายน 2563, จาก https://graphicbuffet.co.th/line-notify-ตัวช่วยใหม่/

[14] Y. Chu, C. Huang, X. Xie, B. Tan, S. Kamal, and X. Xiong, “Multilayer hybrid deep-learning method for waste classification and recycling,” Comput. Intell. Neurosci., vol. 2018, 2018.

[15] C. Bircanoglu, M. Atay, F. Beser, O. Genc, and M. A. Kizrak, “RecycleNet: Intelligent Waste Sorting Using Deep Neural Networks,” 2018 IEEE Int. Conf. Innov. Intell. Syst. Appl. INISTA 2018, 2018.

[16] T. Addabbo et al., “A LoRa-based IoT Sensor Node for Waste Management Based on a Customized Ultrasonic Transceiver,” SAS 2019 - 2019 IEEE Sensors Appl. Symp. Conf. Proc., pp. 1–6, 2019.

[17] ทิพานัน พงษ์สุวรรณ, อนุพงษ์ ติตะ, ภานุวัตร อุทัยบาล, “ระบบติดตามถังขยะอัจฉริยะ Smart Trash Tracking System,” Tue 3 Jul. 2018. Khon Kaen University Library.

ภาคผนวก

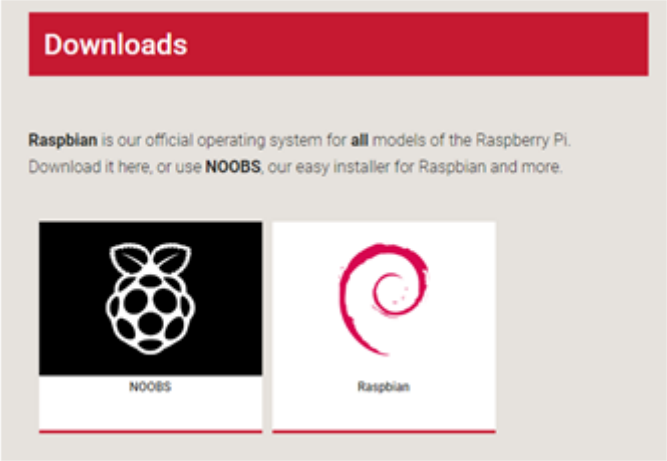
ภาคผนวก ก

คู่มือการติดตั้ง บน Raspberry Pi

1. ติดตั้ง Raspbian บน Raspberry PI 4

เข้าเว็บไซต์ https://www.raspberrypi.org/downloads/

ดาวน์โหลดไฟล์ Raspbian ลงในเครื่องคอมพิวเตอร์ ดังภาพที่ ก - 1



ภาพที่ ก - 1 : ขั้นตอนการดาวน์โหลดไฟล์ Raspbian

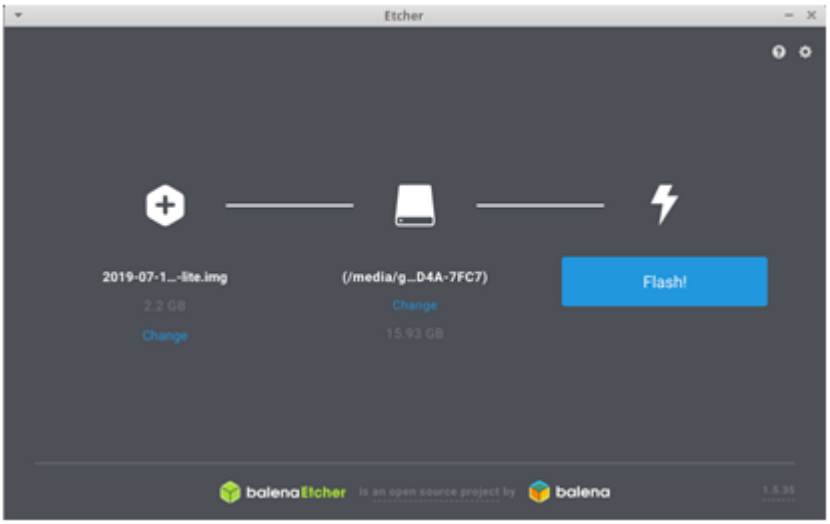
2. ติดตั้งโปรแกรม balena etcher บนเครื่องคอมพิวเตอร์

เข้าเว็บไซต์ https://www.balena.io/etcher/ ดาวน์โหลดไฟล์ balena etcher ดังภาพที่ ก - 2

ภาพที่ ก - 2 : ขั้นตอนการดาวน์โหลดไฟล์ balena etcher

3. เปิดโปรแกรม balena etcher

เลือกไฟล์ .img ที่จะติดตั้ง --> เลือก Micro SD Card ที่เตรียมเอาไว้ --> กด Flash ดังภาพที่ ก - 3



ภาพที่ ก - 3 : ขั้นตอนการติดตั้ง Raspbian โดยใช้ balena etcher

รอโหลดจนเสร็จ ดังภาพที่ ก - 4

ภาพที่ ก - 4 : รอโหลดจนสำเร็จ

4. ตั้งค่า Raspberry Pi 4

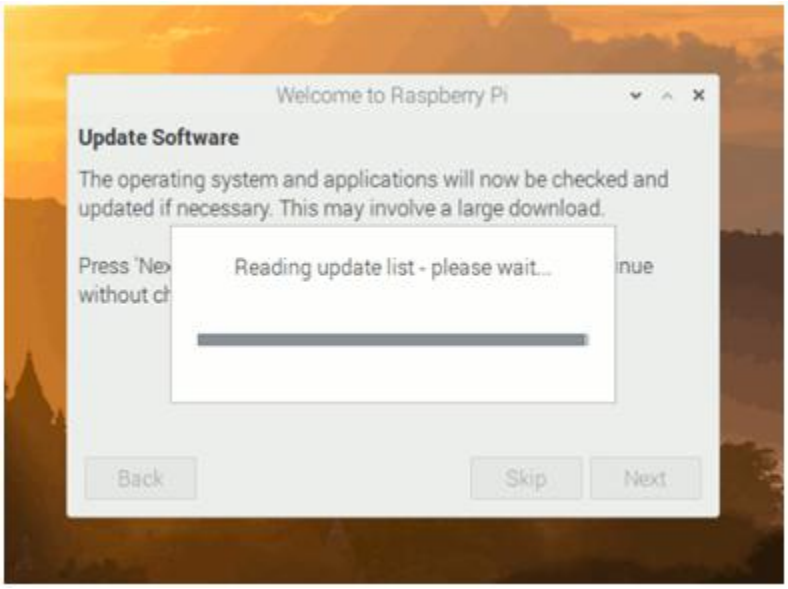
นำ SD Card ใส่ Raspberry Pi จากนั้นเปิด Raspberry Pi --> ตั้งค่าสถานที่และภาษา ดังภาพที่ ก - 5

ภาพที่ ก - 5 : ขั้นตอนการตั้งค่าสถานที่และภาษา

ตั้งค่า Password --> กด Next ดังภาพที่ ก - 6

ภาพที่ ก - 6 : ขั้นตอนการตั้งค่า Password

รอโหลดจนเสร็จ ดังภาพที่ ก - 7

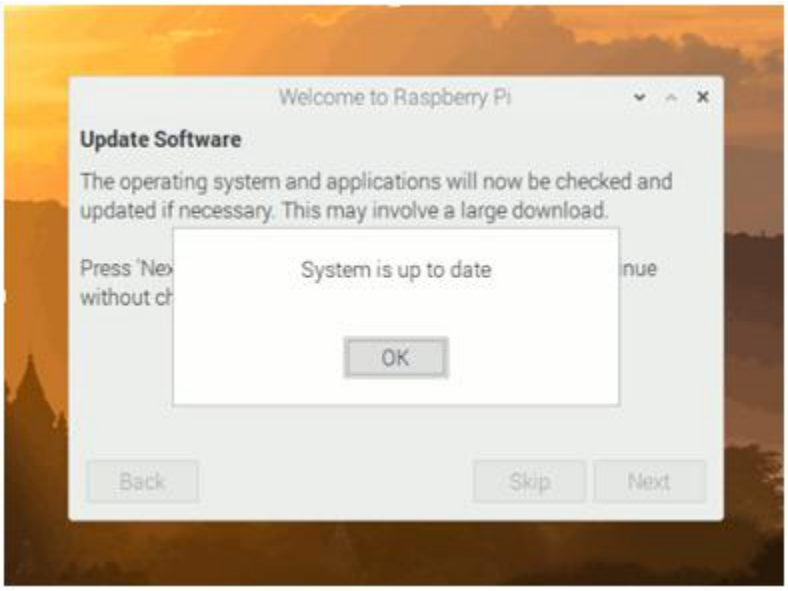


ภาพที่ ก - 7 : รอโหลดจนสำเร็จ

เชื่อมต่ออินเทอร์เน็ต --> กด Next ดังภาพที่ ก - 8

ภาพที่ ก - 8 : ขั้นตอนตั้งค่าอินเทอร์เน็ต

กด OK ดังภาพที่ ก - 9



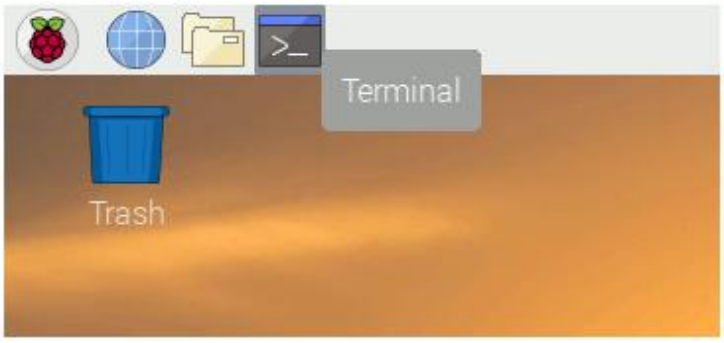
ภาพที่ ก - 9 : ขั้นตอนอัพเดทซอฟแวต์

กด Restart ดังภาพที่ ก - 10

ภาพที่ ก - 10 : ขั้นตอนรีสตาร์ท Raspberry Pi 4

5. Install fastai

เข้า terminal ดังภาพที่ ก - 11



ภาพที่ ก - 11 : ขั้นตอนเข้า Terminal

ติดตั้งคำสั่งต่อไปนี้

$ sudo apt-get -y update

$ sudo apt-get -y upgrade

$ sudo apt-get -y install python3-numpy python3-scipy python3-matplotlib python3-pandas

$ sudo apt-get -y install python3-pip python3-pil

$ sudo apt-get -y install build-essential python-dev git

$ sudo apt-get -y install python-requests python-dataclasses python-typing

$ sudo apt-get -y install g++ cmake python-dev

$ sudo apt-get -y install nodejs npm freetype2-demos

$ sudo apt-get -y install zlib1g-dev zip libjpeg8-dev libhdf5-dev libpng-devel libfreetype-devel

$ sudo apt-get -y install libhdf5-serial-dev hdf5-tools

$ sudo apt-get -y install cython python-cuda

$ sudo apt-get -y install pkg-config libfreetype6-dev libpng12-dev

$ pip3 install freetype-py

$ pip3 install pypng

```
$ pip3 install dataclasses bottleneck
$ pip3 install pynvx
$ pip3 install pandas
$ pip3 install fire
$ pip3 install jupyter jupyterlab
$ pip3 install fastprogress
$ pip3 install torchvision --no-deps
$ pip3 install spacy==2.0.18
$ pip3 install fastai --no-deps
$ mkdir .fastai
$ mkdir .fastai/data
$ echo "done with part1 - now logout, login again and run setup_jupyter.sh"
```

6. ติดตั้ง torch

โคลน git ที่ https://github.com/sungjuGit/PyTorch-and-Vision-for-Raspberry-Pi-4B.git

ติดตั้งคำสั่งต่อไปนี้

```
$ sudo pip3 install torch-1.4.0a0+f43194e-cp37-cp37m-linux_armv7l.whl
$ sudo pip3 install torchvision-0.5.0a0+9cdc814-cp37-cp37m-linux_armv7l.whl
```

7. ติดตั้ง boto

```
$ sudo apt install python3-boto
```

8. ติดตั้ง opencv

หมายเหตุ **:** ต้องเป็น python เวอร์ชั่น 3.7 ขึ้นไป

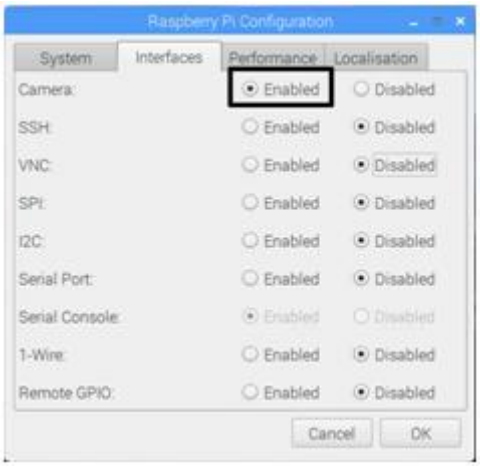
```
$ pip install opencv-contrib-python==4.1.0.25
```

9. ติดตั้ง Raspberry Pi Camera

เลือก Raspberry Pi Configuration ดังภาพที่ ก - 12

ภาพที่ ก - 12 : เข้า Raspberry Pi Configuration

เลือก Enabled ที่ Camera --> กด OK ดังภาพที่ ก - 13



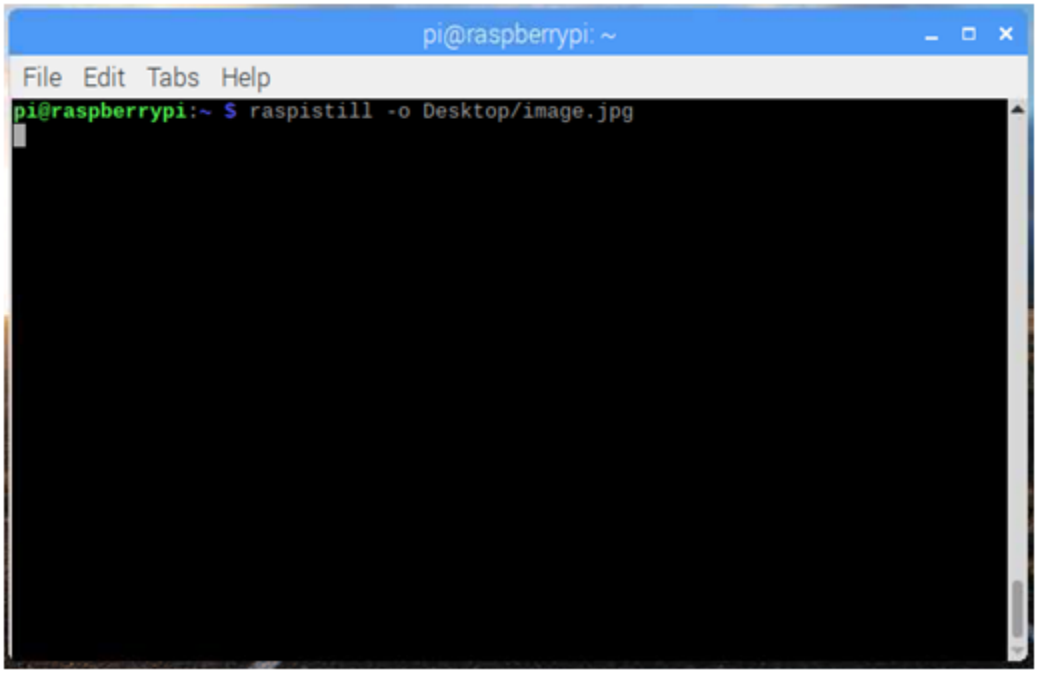
ภาพที่ ก - 13 : เข้า Raspberry Pi Configuration

ทดสอบถ่ายภาพ

เข้า Terminal ดังภาพที่ ก - 14

ภาพที่ ก - 14 : เข้า Terminal

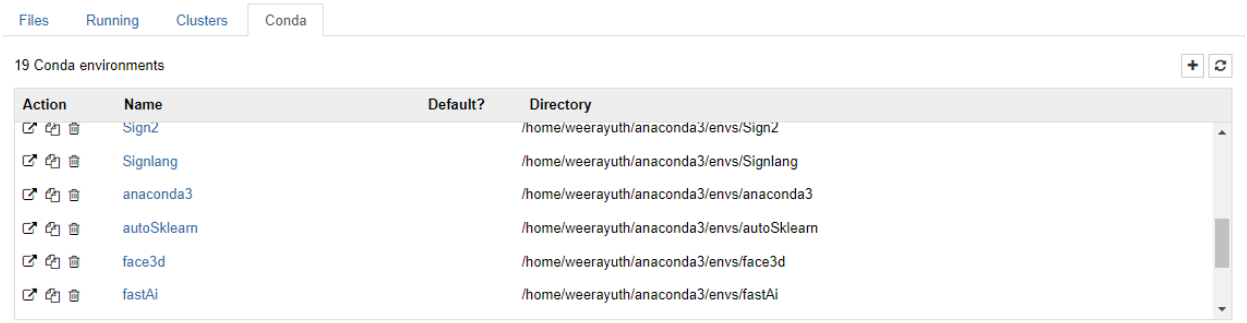
พิมพ์คำสั่งถ่ายภาพ $ raspistill -o Desktop/image.jpg ดังภาพที่ ก - 15



ภาพที่ ก - 15 : คำสั่งถ่ายภาพ

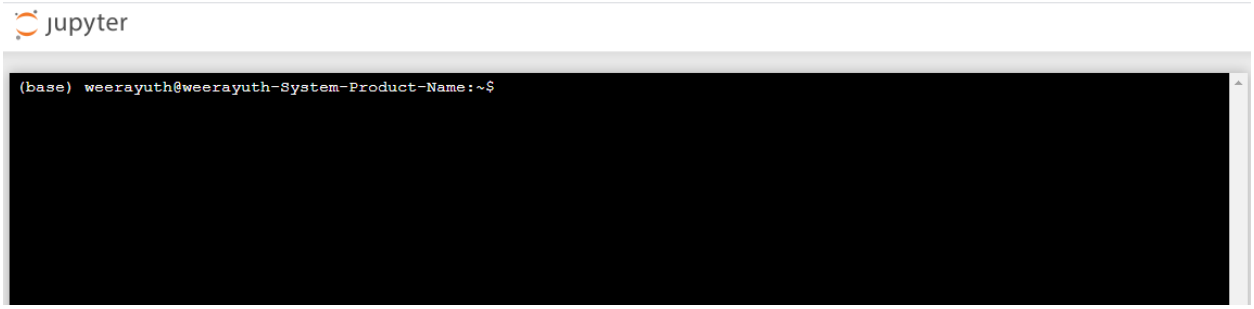
คู่มือการติดตั้ง Jupyter บน Server http://10.1.136.181:8888/tree

1. ไปที่ Conda เพื่อสร้าง New Environment ไว้สำหรับติดตั้ง Module/Package ต่างๆ ที่จำเป็นต้องใช้งานให้พร้อม โดยสร้างชื่อ fastAi ดังภาพที่ ก - 16



ภาพที่ ก - 16 : Conda on Server

2. เข้าหน้า Files ไปที่เมนู New แล้วเลือก Terminal เพื่อติดตั้ง จะแสดงหน้าจอ ดังภาพที่ ก - 17



ภาพที่ ก - 17 : Terminal on Server

ติดตั้งคำสั่งต่อไปนี้

$ conda activate fastAi

$ conda install matplotlin - y

$ conda install --no-deps -c fastai fastai

$ conda install yaml

$ pip install yaml

$ pip install pyyaml

$ pip install pandas

$ pip install scipy$ pip install requests

$ pip install fastprogress

$ pip install torch

$ conda install PIL

$ pip install PIL

$ pip install Pillow

$ pip install torchvision

$ pip install git+https://github.com/fastai/fastai.git

$conda install glob2

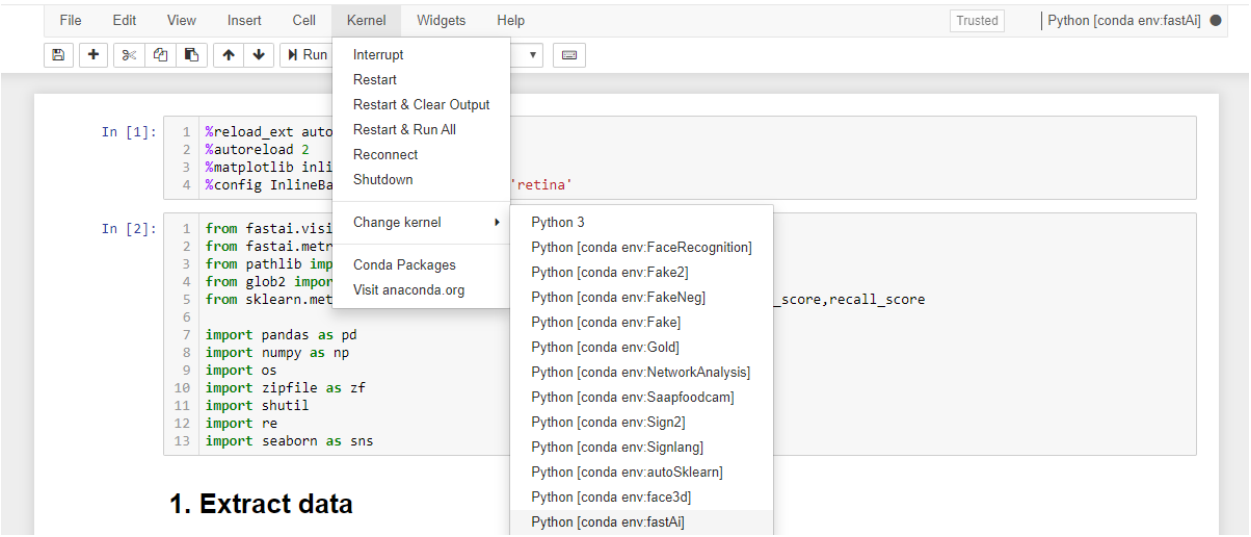
$ pip install glob2

$ pip install sklearn

$ pip install seaborn

$ pip3 install fastai

3. เมื่อเปิดใช้งานโค้ดให้ตั้งค่าที่ Kernel แล้วเลือก Environment ที่เราสร้างไว้ คือ Python [conda env:fastAi] เพื่อใช้งาน ดังภาพ ก - 18



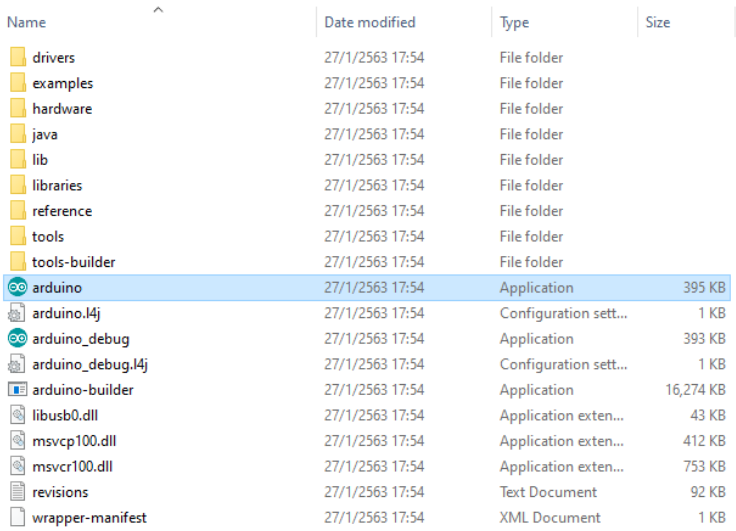
ภาพที่ ก - 18 : Change Kernel

คู่มือการติดตั้งบน Arduino

1. ดาวน์โหลด Arduino IDE ที่เว็บไซต์ https://www.arduino.cc/en/main/software แล้วเลือกตามระบบปฏิบัติการที่ใช้ ดังภาพ ก - 19

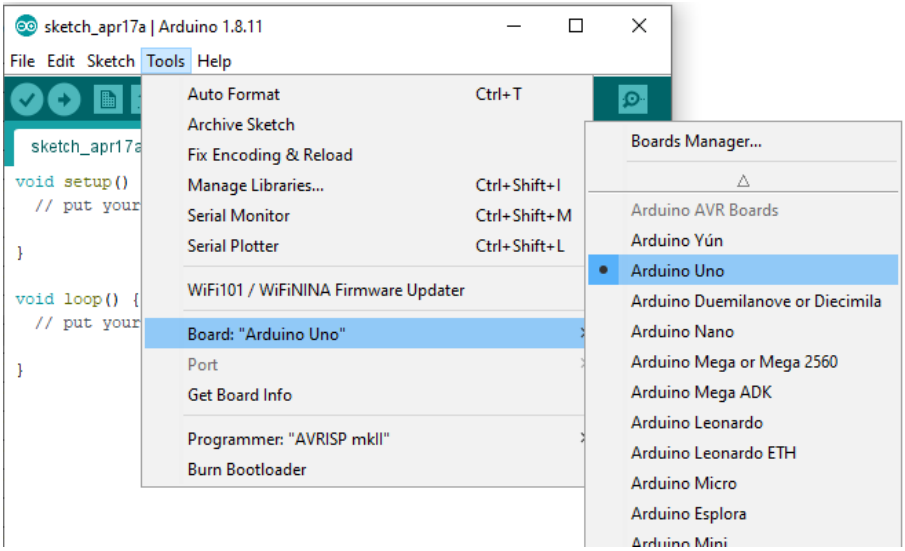
ภาพที่ ก - 19 : Download Arduino IDE

2. เปิดไฟล์ arduino.exe เพื่อเปิดโปรแกรม Arduino IDE ดังภาพ ก - 20



ภาพที่ ก - 20 : Open Arduino IDE Program

3. เมื่อเปิดโปรแกรมแล้วให้เลือกชนิดบอร์ดที่ต้องการจะนำโค้ดไปใส่ โดยไปที่ Tool เลือกเมนู board และทำการเลือก borad arduino ที่เราใช้ เช่น Arduino UNO ดังภาพที่ ก - 21



ภาพที่ ก - 21 : Select board arduino

4. ทำการเสียบ USB เข้ากับ board arduino แล้วเลือก port ให้ตรงกัน

5. เมื่อใส่โค้ดแล้วให้ Compile โค้ด เพื่อตรวจสอบว่าโค้ดไม่มีอะไรผิดพลาดจะขึ้น Done compiling

6. จากนั้นทำการ Upload เพื่อใส่โค้ดไปยัง Arduino รอจน Done uploading แสดงว่าลงเสร็จเรียบร้อย

ภาคผนวก ข

ชื่อไฟล์ : TrainModel_ResNet50.ipynb

ภาษาที่ใช้ : Python

คำอธิบาย : โค้ดการทำงานในการสร้าง model โดยให้เกิดการเรียนรู้จากภาพแล้วสามารถนำมาจำแนกได้ โค้ดจะเริ่มตั้งแต่การนำเข้าข้อมูลภาพ การแบ่งไฟล์ การเรียนรู้ของภาพ การประเมินประสิทธิภาพ และการส่งออกไฟล์ในรูปแบบที่สามารถไปใช้งานได้ในการทำนายเพื่อไม่ต้องเสียเวลาในการเรียนรู้หลายครั้ง

```
#การนำเข้า Module อื่น ๆ เพื่อนำมาเรียกใช้งานในโปรแกรม
%reload_ext autoreload
%autoreload 2
%matplotlib inline
%config InlineBackend.figure_format = 'retina'
from fastai.vision import *
from fastai.metrics import error_rate
from pathlib import Path
from glob2 import glob
from sklearn.metrics import confusion_matrix , accuracy_score,precision_score,recall_score
import pandas as pd
import numpy as np
import os
import zipfile as zf
import shutil
import re
import seaborn as sns

files = zf.ZipFile("data-set.zip",'r') # แตกไฟล์ .zip
files.extractall()
files.close()

def split_indices (folder,seed1,seed2): #การแบ่ง folder Train, Test, Validation
    n = len(os.listdir(folder))
    full_set = list(range(1,n+1))
    random.seed(seed1)
    train = random.sample(list(range(1,n+1)),int(.8*n))
    remain = list(set(full_set)-set(train))
    random.seed(seed2)
```

```
    valid = random.sample(remain,int(.5*len(remain)))
    test = list(set(remain)-set(valid))
    return(train,valid,test)


def get_names (waste_type,indices):
    file_names = [waste_type+" ("+str(i)+").jpg" for i in indices]
    return(file_names)
def move_files (source_files,destination_folder):
    for file in source_files:
        shutil.move(file,destination_folder)

subsets = ['train','valid']
waste_types = ['glass','metal','plastic','trash']
for subset in subsets: # จัดการ directory
    for waste_type in waste_types:
        folder = os.path.join('data',subset,waste_type)
        if not os.path.exists(folder):
            os.makedirs(folder)

if not os.path.exists(os.path.join('data','test')):
    os.makedirs(os.path.join('data','test'))

for waste_type in waste_types:
    source_folder = os.path.join('data-set',waste_type)
    train_ind, valid_ind, test_ind = split_indices(source_folder,1,1)

    train_names = get_names(waste_type,train_ind)
    train_source_files = [os.path.join(source_folder,name) for name in train_names]
    train_dest = "data/train/"+waste_type
    move_files(train_source_files,train_dest)

    valid_names = get_names(waste_type,valid_ind)
    valid_source_files = [os.path.join(source_folder,name) for name in valid_names]
    valid_dest = "data/valid/"+waste_type
```

```
    move_files(valid_source_files,valid_dest)

    test_names = get_names(waste_type,test_ind)
    test_source_files = [os.path.join(source_folder,name) for name in test_names]
    move_files(test_source_files,"data/test")


path = Path(os.getcwd())/"data" # ย้าย path ไปที่เราจะเรียกใช้

tfms = get_transforms(do_flip=True,flip_vert=True) # เพิ่มข้อมูลโดยการหมุนภาพแนวตั้งและนอน
data = ImageDataBunch.from_folder(path,test="test",ds_tfms=tfms,bs=16)

learn = create_cnn(data,models.resnet50,metrics=error_rate) # เลือกใช้ model “ResNet50”

learn.lr_find() # การหา learning rate
learn.recorder.plot(suggestion=True)

learn.fit_one_cycle(40,max_lr=7.41E-04) # การ Training 40 รอบการทำงาน

interp = ClassificationInterpretation.from_learner(learn) # การแสดงภาพที่ไม่ถูกต้องที่สุด
losses,idxs = interp.top_losses()
interp.plot_top_losses(9, figsize=(15,11))
interp.plot_top_losses
interp.plot_confusion_matrix(figsize=(12,12), dpi=60)
interp.most_confused(min_val=2)

preds = learn.get_preds(ds_type=DatasetType.Test) # ทำนายกับชุด Test
max_idxs = np.asarray(np.argmax(preds[0],axis=1))
yhat = []
for max_idx in max_idxs:
    yhat.append(data.classes[max_idx])

y = [] # ดึงค่าจริงของภาพมาเทียบความถูกต้อง
for label_path in data.test_ds.items:
    y.append(str(label_path))
```

```
pattern = re.compile("([a-z]+)(\s[(][0-9]+[)])")
for i in range(len(y)):
    y[i] = pattern.search(y[i]).group(1)

cm = confusion_matrix(y,yhat) # ดูประสิทธิภาพ model ในรูป confusion matrix
df_cm = pd.DataFrame(cm,waste_types,waste_types)
plt.figure(figsize=(11,8))



ax = sns.heatmap(df_cm,annot=True,fmt="g",cmap="YlGnBu")
bottom, top = ax.get_ylim()
ax.set_ylim(bottom + 0.5, top - 0.5)

plt.xlabel('Predicted')
plt.ylabel('Actual')
plt.show()

correct = 0
for r in range(len(cm)):
    for c in range(len(cm)):
        if (r==c):
            correct += cm[r,c]
accuracy = correct/sum(sum(cm))
accuracy # แสดงเปอร์เซ็นต์ความถูกต้อง

learn.export('TrashyNet.pkl') # Export model เพื่อนำไปใช้ต่อ
```

ชื่อไฟล์ : Automatrashsimple.ino

ภาษาที่ใช้ : C

คำอธิบาย : โค้ดการทำงานของถังขยะในส่วนที่ควบคุมมอเตอร์หลังจากที่มีการประมวลผลแล้วว่าขยะเป็นประเภทอะไร โดยโค้ดนี้จะทำงานบน Arduino ซึ่งจะทำงานควบคู่กับการใช้ Pin ที่มีการกำหนดค่าแรงดันไฟใน Raspberry Pi ให้ตรงกัน จึงสามารถควบคุมการทำงานของถังขยะให้ทิ้งลงช่องที่ต้องการได้ ประกอบไปด้วยการเซตค่า การเคลื่อนที่ต่าง ๆ ของตัวกล่องรับขยะ และตัวดันขยะ

```
// กำหนดการเชื่อมต่อมอเตอร์ตามขั้นตอนการทำงาน
// ตั้งค่าการเคลื่อนที่ของตัวกล่องรับขยะ
#define dirPin1 2
#define stepPin1 3

// ตั้งค่าการเคลื่อนที่ของตัวดันขยะ
#define dirPin2 4
#define stepPin2 5

// การปรับเปลี่ยนระยะการเคลื่อนที่ของกล่องรับถัง ให้ไปยังถังนั้น ๆ
#define stepsPerRevolution1 200 ///------- Step distance Trash A (Trash)
#define stepsPerRevolution2 5000 ///------- Step distance Trash B (Plastic)
#define stepsPerRevolution3 10700 ///------- Step distance Trash C (Metal)
#define stepsPerRevolution4 15870 ///------- Step distance Trash D (Glass)

// การปรับระยะตัวดันขยะ
#define stepsArm 3100

// การปรับระยะให้อยู่ในตำแหน่งเริ่มต้น
#define stepsPerRevolutionHome 1

// กำหนดค่านำเข้า
const int Ain1 = 9;
const int Ain2 = 10;
const int Ain3 = 11;
const int Ain4 = 12;

// ค่าคงที่จะไม่มีการเปลี่ยนแปลง โดยค่านี้จะเป็นการกำหนด Pin
const int MoveHome = 8; // Pin ของตัวกล่องรับขยะ
const int ArmHome = 6; // Pin ของตัวดันขยะ
```

```
// ตัวแปรที่จะมีการเปลี่ยนแปลง
int MoveHomeState = 0; // ตัวแปรเพื่อดูสถานะของปุ่มเช็คการกับจุดเริ่มต้นของตัวกล่องรับขยะ
int ArmHomeState = 0; // ตัวแปรเพื่อดูสถานะของปุ่มเช็คการกับจุดเริ่มต้นของตัวดันขยะ


// กำหนดตัวแปรกำหนดสถานะการนำเข้า
int StateA1 = 0;
int StateA2 = 0;
int StateA3 = 0;
int StateA4 = 0;

void setup () {
  // ประกาศ Pin เป็นการนำเข้า
  pinMode (Ain1, INPUT);
  pinMode (Ain2, INPUT);
  pinMode (Ain3, INPUT);
  pinMode (Ain4, INPUT);

  // ประกาศ Pin เป็นการส่งออก
  pinMode (stepPin1, OUTPUT);
  pinMode (dirPin1, OUTPUT);

  pinMode (stepPin2, OUTPUT);
  pinMode (dirPin2, OUTPUT);

  pinMode (MoveHome, INPUT);
  pinMode (ArmHome, INPUT);

  Serial.begin(9600);
  Serial.println("Automatrash system V1.0");
  delay (200);

// เซตให้ทุกครั้งก่อนการทำงานอยู่ที่ตำแหน่งเริ่มต้น เพื่อป้องกันกรณีไฟดับขณะใช้งาน และ ค้าง
    movetrayhome ();
    ArmtoHome ();
  }

// เงื่อนไขการเคลื่อนที่ไปยังถังที่ต้องการ โดยจะควบคุมแรงดันไฟของค่า 4 Pin โดยไปยังถังที่ Pin มีค่าเป็น LOW
void loop () { // อ่านค่าจาก Raspberry Pi
    StateA1 = digitalRead (Ain1); // ให้ตัวแปรสามารถอ่านค่าสถานะแรงดันไฟได้ โดยมีค่าเป็น LOW หรือ HIGH
    delay (10);
```

```
  StateA2 = digitalRead (Ain2);
  delay (10);
  StateA3 = digitalRead (Ain3);
  delay (10);
  StateA4 = digitalRead (Ain4);
  delay (10);


 // กำหนดเงื่อนไข ที่ตัวแปร StateA1 ตรวจสอบสถานะได้เป็น LOW
  if (StateA1 == LOW && StateA2 == HIGH && StateA3 == HIGH && StateA4 == HIGH) {
  Serial.println("Trash-1");
  delay (500);
  movetrayforward1();
  delay (500);
  Serial.println("Arm-push");
  movearm ();
  delay (500);
  ArmtoHome ();
  Serial.println("Back-home");
  movetrayhome ();
  delay (500);
  Serial.println("Finish");
  delay (1000);
  }

  // กำหนดเงื่อนไข ที่ตัวแปร StateA2 ตรวจสอบสถานะได้เป็น LOW
  else if (StateA1 == HIGH && StateA2 == LOW && StateA3 == HIGH && StateA4 == HIGH) {
  Serial.println("Trash-2");
  delay (500);
  movetrayforward2();
  delay (500);
  Serial.println("Arm-push");
  movearm ();
  delay (500);
  ArmtoHome ();
  Serial.println("Back-home");
  movetrayhome ();
  delay (500);
  Serial.println("Finish");
  delay (1000);
  }
```

```
  // กำหนดเงื่อนไข ที่ตัวแปร StateA3 ตรวจสอบสถานะได้เป็น LOW
    else if (StateA1 == HIGH && StateA2 == HIGH && StateA3 == LOW && StateA4 == HIGH) {
    Serial.println("Trash-3");
    delay (500);
    movetrayforward3();
    delay (500);
    Serial.println("Arm-push");
    movearm ();
    delay (500);
    ArmtoHome ();
    Serial.println("Back-home");
    movetrayhome ();
    delay (500);
    Serial.println("Finish");
    delay (1000);
    }

    // กำหนดเงื่อนไข ที่ตัวแปร StateA4 ตรวจสอบสถานะได้เป็น LOW
    else if (StateA1 == HIGH && StateA2 == HIGH && StateA3 == HIGH && StateA4 == LOW) {
    Serial.println("Trash-4");
    delay (500);
    movetrayforward4();
    delay (500);
    Serial.println("Arm-push");
    movearm ();
    delay (500);
    ArmtoHome ();
    Serial.println("Back-home");
    movetrayhome ();
    delay (500);
    Serial.println("Finish");
    delay (1000);
    }

    else {
        Serial.println("Wait-command");
    }
    delay (1000);
}
```

```
// การเคลื่อนที่ของตัวกล่องรับไปยังช่องถังขยะที่ 1
void movetrayforward1() {
  digitalWrite (dirPin1, LOW); // กำหนดการทำงานของ Pin ในสถานะ LOW หรือ HIGH
  for (int i = 0; i < stepsPerRevolution1; i++) {
                digitalWrite (stepPin1, HIGH);
                delayMicroseconds (500);
                digitalWrite (stepPin1, LOW);
                delayMicroseconds (500);
  }
}

// การเคลื่อนที่ของตัวกล่องรับไปยังช่องถังขยะที่ 2
void movetrayforward2() {
  digitalWrite (dirPin1, LOW);
  for (int i = 0; i < stepsPerRevolution2; i++) {
                digitalWrite (stepPin1, HIGH);
                delayMicroseconds (500);
                digitalWrite (stepPin1, LOW);
                delayMicroseconds (500);
  }
}

// การเคลื่อนที่ของตัวกล่องรับไปยังช่องถังขยะที่ 3
void movetrayforward3() {
  digitalWrite (dirPin1, LOW);
  for (int i = 0; i < stepsPerRevolution3; i++) {
                digitalWrite (stepPin1, HIGH);
                delayMicroseconds (500);
                digitalWrite (stepPin1, LOW);
                delayMicroseconds (500);
  }
}

// การเคลื่อนที่ของตัวกล่องรับไปยังช่องถังขยะที่ 4
void movetrayforward4() {
  digitalWrite (dirPin1, LOW);
  for (int i = 0; i < stepsPerRevolution4; i++) {
                digitalWrite (stepPin1, HIGH);
                delayMicroseconds (500);
                digitalWrite (stepPin1, LOW);
                delayMicroseconds (500);
```

```
  }
}

// การเคลื่อนที่ของตัวกล่องรับไปให้กลับมายังตำแหน่งเดิม
void movetrayhome () {
  do {MoveHomeState = digitalRead (MoveHome);
   digitalWrite (dirPin1, HIGH);
            for (int i = 0; i < stepsPerRevolutionHome; i++) {
              digitalWrite (stepPin1, HIGH);
              delayMicroseconds (500);
              digitalWrite (stepPin1, LOW);
              delayMicroseconds (500);
    }
  }
  while (MoveHomeState == LOW);
}

// การเคลื่อนที่ของตัวดันขยะให้มีการดัน
void movearm () {
  digitalWrite (dirPin2, LOW);
            for (int i = 0; i < stepsArm; i++) {
              digitalWrite (stepPin2, HIGH);
              delayMicroseconds (500);
              digitalWrite (stepPin2, LOW);
              delayMicroseconds (500);
    }
}
// การเคลื่อนที่ของตัวดันขยะให้กลับมายังตำแหน่งเดิม
void ArmtoHome () {
   do {ArmHomeState = digitalRead (ArmHome);
      digitalWrite (dirPin2, HIGH);
      for (int i = 0; i < stepsPerRevolutionHome; i++) {
                        digitalWrite (stepPin2, HIGH);
                        delayMicroseconds (500);
                        digitalWrite (stepPin2, LOW);
                        delayMicroseconds (500);
      }
    }
    while (ArmHomeState == LOW);
}
```

ชื่อไฟล์ : main.py

ภาษาที่ใช้ : Python

คำอธิบาย : โค้ดการทำงานของถังขยะในส่วนที่ควบคุมการรับภาพถ่ายจากขยะจริงเพื่อนำไปประมวลผลลัพธ์ว่าเป็นขยะประเภทไหน และกำหนดแรงดันไฟให้ตรงเพื่อส่งการทำงานต่อให้มอเตอร์ทิ้งขยะตามขยะนั้น ๆ ที่จำแนกได้ตามผลลัพธ์ โดยทำตามขั้นตอนการทำงานของโค้ดที่ Arduino ต่อไป

โดยโค้ดส่วนนี้เป็นตัวดำเนินการหลักของขั้นตอนการทำงานทั้งหมดตั้งแต่ต้นจนจบการทำงาน

```
#การนำเข้า Module อื่น ๆเพื่อนำมาเรียกใช้งานในโปรแกรมหลัก
import RPi.GPIO as GPIO
import time
import os
from TestModel import *
from Photo import *
from Motor import *
GPIO.setmode(GPIO.BCM) # การกำหนดให้การตั้งค่าอ้างอิงตามชื่อของ GPIO

TRIG = 21
ECHO = 20
pinled = 26
pin_ready = 6
pin_working = 5

testmodel = TestModel () # การกำหนดตัวแปรของ Module ที่จะนำมาใช้
photo = Photo ()
motor = Motor ()

GPIO.setwarnings(False) # ตั้งค่าแต่ละ Pin
GPIO.setup(pinled, GPIO.OUT)
GPIO.setup(pin_ready, GPIO.OUT)
GPIO.setup(pin_working, GPIO.OUT)
GPIO.setup(TRIG, GPIO.OUT)
GPIO.setup(ECHO, GPIO.IN)
GPIO.setwarnings(False)

try:
    while True:
        GPIO.output(pin_ready, GPIO.HIGH)
        GPIO.output(TRIG, False)
```

```
        time. sleep (0.25)
        GPIO.output(TRIG, True)
        time. sleep (0.0001)
        GPIO.output(TRIG, False)

        while GPIO.input(ECHO)==0: # เซนเซอร์ตรวจจับไม่พบอะไร
            time_start = time. time ()

        while GPIO.input(ECHO)==1: # เซนเซอร์ตรวจจับพบว่ามีวัตถุบางอย่างเข้ามา
            time_end = time. Time ()

        time_duration = time_end - time_start # การคำนวณระยะทางของเซนเซอร์
        distance = time_duration * 17150
        distance = round (distance, 2)

        if distance < 27: # ระยะทางน้อยกว่า 27 ซม. พบว่ามีวัตถุบางอย่าง
            GPIO.output(pin_ready, GPIO.LOW)
            GPIO.output(pin_working, GPIO.HIGH)
            GPIO.output(pinled, GPIO.LOW) # เปิดไฟ LED
            photo. capture () # ถ่ายภาพ โดยเรียกฟังก์ชันจาก class Photo
            photo. crop () # ตัดขอบภาพ
            photo. CheckTrash () # ตรวจสอบว่าการถ่ายภาพนั้นมีขยะแน่นอนหรือไม่
            if (photo. CheckTrash () == True): # มีขยะจริง
                predict = testmodel. predict ()
                motor. arduino(predict) # นำผลลัพธ์ที่ได้ไปกำหนดแรงดันไฟ เพื่อให้มอเตอร์ทำงานตามโค้ดใน Aruduino
                photo. rename(predict)
                photo.uploadtos3(predict) # อัพโหลดภาพขึ้น AWS S3
                ultrasonic. checkfull(predict) # ตรวจสอบว่าถังขยะเต็มหรือไม่
                GPIO.output(pinled, GPIO.HIGH) # เปิดไฟ LED
                GPIO.output(pin_working, GPIO.LOW)

            else: # ไม่มีขยะ
                os. remove('/home/pi/Desktop/test.jpg')
                GPIO.output(pin_working, GPIO.LOW)

except Keyboard Interrupt:
    GPIO.output(pin_ready, GPIO.LOW)
    GPIO.cleanup()
finally:
    GPIO.output(pin_ready, GPIO.LOW)
    GPIO.cleanup()
```

ชื่อไฟล์ : Photo.py

ภาษาที่ใช้ : Python

คำอธิบาย : โค้ดฟังก์ชันการทำงานของส่วนที่มีความเกี่ยวข้องกับการใช้ภาพถ่าย ที่มีการเชื่อมกับ Raspberry Pi Camera เป็นกล้องที่ใช้ในการภ่ายภาพอยู่ภายในตัวถัง และ Amazon Simple Storage Service (S3) ในการเก็บข้อมูลภาพขยะที่ถ่ายไป โดยจะมีฟังก์ชันย่อยๆ ดังนี้ คำสั่งถ่ายภาพ ตัดขอบภาพ ตรวจสอบวัตถุในภาพ เปลี่ยนชื่อไฟล์ภาพ อัพโหลดไฟล์ภาพ

โดยโค้ดส่วนนี้เป็นตัวดำเนินการย่อย ที่โปรแกรมหลักจะเรียกใช้งาน

```
#การนำเข้า Module อื่น ๆเพื่อนำมาเรียกใช้งานในโปรแกรม
from picamera import PiCamera
from time import sleep
import sys, os, glob, time
import os
import datetime
import RPi.GPIO as GPIO
from boto. s3. connection import S3Connection
from boto.s3.key import Key
import glob
from PIL import Image, ImageDraw, ImageFilter
import cv2
import numpy as np

from TestModel import *
testmodel = TestModel ()
pin = 16
GPIO.setmode(GPIO.BCM)
GPIO.setup(pin, GPIO.OUT)

class Photo:
    def __init__(self): # การกำหนดเพื่อใช้ในการเข้าถึง S3
        self.directory = '/home/pi/Desktop/'
        self.AWS_ACCESS = "AKIA4JSFZIFQHABSYHEQ"
        self.AWS_SECRET = "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"
        self.conn = S3Connection(self.AWS_ACCESS,self.AWS_SECRET)
        self.bucket = self.conn.get_bucket('trashy2')
```

```
    def capture(self): # การถ่ายภาพขยะ
        start = time.time ()
        camera = PiCamera ()
        camera.resolution = (1280,960)
        camera.start_preview()
        sleep(3)
        camera.capture ('/home/pi/Desktop/test.jpg')
        camera.stop_preview()
        camera.close ()

    def crop(self): # การตัดขอบภาพ
        im = Image.open('/home/pi/Desktop/test.jpg')
        thumb_width = 711
        thumb_height = 568
        img_width, img_height = im.size
        im_thumb = im.crop (((img_width - thumb_width) // 2, (img_height - thumb_height) // 2,
(img_width + thumb_width) // 2, (img_height + thumb_height) // 2))
        im_thumb.save ('/home/pi/Desktop/test.jpg')

    def rename(self,folder): # การเปลี่ยนชื่อไฟล์ภาพ
        Current_Date = datetime.datetime.now ()
        os.rename (r'/home/pi/Desktop/test.jpg',r'/home/pi/Desktop/'+ folder + '-' + str(Current_Date) + '.jpg')

    def percent_cb(self,complete, total):
        sys.stdout.write('.')
        sys.stdout.flush()

    def getFiles(self, dir):
        self.dir =dir
        return [os. path.basename(x) for x in glob.glob(str(self.dir) + '*.jpg')]

    def uploadtos3(self,folder): # อัพโหลดไฟล์ภาพขึ้น S3
        filenames = self.getFiles (self.directory)
        path_s3 = folder +'/'
        for f in filenames:
            print(f)
            k = Key (self.bucket)
            k.key = path_s3 +f
            GPIO.output(pin, GPIO.HIGH)
```

```
        k.set_contents_from_filename (self.directory + f, cb=self.percent_cb, num_cb=10)
        GPIO.output(pin, GPIO.LOW)
        os.remove (self.directory + f)


def CheckTrash(self): # ตรวจสอบวัตถุในภาพ โดยแปลงภาพเป็นภาพขาวดำก่อน
    bg = cv2.imread('/home/pi/Desktop/TestSubImg/bg.jpg')
    bg_grey = cv2.cvtColor(bg, cv2.COLOR_RGB2GRAY)

    frame = cv2.imread('/home/pi/Desktop/test.jpg')
    frame_grey = cv2.cvtColor(frame, cv2.COLOR_RGB2GRAY)

    difference = cv2.absdiff(bg_grey, frame_grey)
    _, final = cv2.threshold(difference, 20, 255, cv2.THRESH_BINARY)

    final = cv2.medianBlur(final,5)
    final = cv2.GaussianBlur(final, (3,3),0)

    arr2 = np.asarray (final)
    image_size = bg_grey.shape [0] * bg_grey.shape [1]

    final2 = arr2 / 255.0
    image_detect_size = np.sum(final2)

    if (image_detect_size > (0.02*image_size)):
        return True; # มีวัตถุ
    else:
        return False; # ไม่มีวัตถุ
```

ชื่อไฟล์ : TestModel.py

ภาษาที่ใช้ : Python

คำอธิบาย : โค้ดฟังก์ชันการทำงานของส่วนในการทำนายผลลัพธ์การแยกประเภทขยะจากภาพถ่ายโดยจะทำการดึงไฟล์ที่มีชื่อว่า Trashy_ResNet50_Model.pkl เป็นไฟล์ที่ได้จากการ Export หลังการ Train model เรียบร้อยแล้ว เพื่อให้สามารถนำมาใช้ได้รวดเร็วมากขึ้น

โดยโค้ดส่วนนี้เป็นตัวดำเนินการย่อย ที่โปรแกรมหลักจะเรียกใช้งาน

```
#การนำเข้า Module อื่น ๆเพื่อนำมาเรียกใช้งานในโปรแกรม
from fastai.vision import *
import time

class TestModel:
    def __init__(self): # การกำหนดที่อยู่ไฟล์ที่จะใช้งาน

        self.path = "/home/pi/Desktop/Trashy"
        self.learn = load_learner(self.path,'Trashy_ResNet50_Model.pkl')

    def predict (self): # การนำภาพเพื่อมาทำนายผลลัพธ์แล้วนำค่าเก็บในตัวแปร predict
        start = time.time ()
        data = open_image ('/home/pi/Desktop/test.jpg')
        data = data.resize ((3,384,512))
        predicted_class, predicted_index, outputs = self.learn.predict (data)
        predict = str (predicted_class)
        end = time.time()
        print ("time to predict : " + str(end - start))
        return predict
```

ชื่อไฟล์ : Motor.py

ภาษาที่ใช้ : Python

คำอธิบาย : โค้ดฟังก์ชันการทำงานของส่วนที่จะทำการเคลื่อนกล่องรับขยะไปยังตำแหน่งของถังขยะที่ทำนายผลได้ มีการตั้งเงื่อนไขในกรณีผลลัพธ์แบบต่างๆ โดยจะตั้งค่าสถานะแรงดันไฟที่ LOW เพื่อให้สามารถสื่อสารกันได้ระหว่างโค้ดนี้กับโค้ดใน Arduino ที่ได้มีการตั้งค่าไว้แล้ว ให้มอเตอร์ทำงานตามต้องการ

โดยโค้ดส่วนนี้เป็นตัวดำเนินการย่อย ที่โปรแกรมหลักจะเรียกใช้งาน

```
#การนำเข้า Module อื่น ๆเพื่อนำมาเรียกใช้งานในโปรแกรม
import RPi.GPIO as GPIO
import time
GPIO.setmode(GPIO.BCM)
from Ultrasonic import *
ultrasonic = Ultrasonic ()

GPIO.setwarnings (False)
pintrash = 24
pinplast = 23
pinmetal = 17
pinglass = 27

time_ch1 = 5     # trash
time_ch2 = 9     # plastic
time_ch3 = 14    # metal
time_ch4 = 21    # glass

class Motor():
    def arduino (self,typebin): # การแยกตามประเภทขยะที่ได้จากการทำนายโดย predict = typebin ในโค้ดนี้
        if (typebin == 'trash'): # เป็นขยะอื่นๆ ตั้งค่าให้ pintrash มีสถานะ LOW
            GPIO.setup(pinplast,GPIO.OUT)
            GPIO.setup(pintrash,GPIO.OUT)
            GPIO.setup(pinmetal,GPIO.OUT)
            GPIO.setup(pinglass,GPIO.OUT)
            GPIO.output(pinplast,GPIO.HIGH)
            GPIO.output(pintrash,GPIO.LOW)
            GPIO.output(pinmetal,GPIO.HIGH)
            GPIO.output(pinglass,GPIO.HIGH)
            time.sleep(time_ch1)
```

```
        ultrasonic.checkfull(typebin)
        time.sleep(time_ch1-3)
        GPIO.output(pintrash,GPIO.HIGH)



    if (typebin == 'plastic'): # เป็นพลาสติก ตั้งค่าให้ pinplas มีสถานะ LOW
        GPIO.setup(pinplast,GPIO.OUT)
        GPIO.setup(pintrash,GPIO.OUT)
        GPIO.setup(pinmetal,GPIO.OUT)
        GPIO.setup(pinglass,GPIO.OUT)
        GPIO.output(pinplast,GPIO.LOW)
        GPIO.output(pintrash,GPIO.HIGH)
        GPIO.output(pinmetal,GPIO.HIGH)
        GPIO.output(pinglass,GPIO.HIGH)
        time.sleep(time_ch2)
        ultrasonic.checkfull(typebin)
        time.sleep(time_ch2-3)
        GPIO.output(pinplast,GPIO.HIGH)


    if (typebin == 'metal'): # เป็นโลหะ ตั้งค่าให้ pinmetal มีสถานะ LOW
        GPIO.setup(pinplast,GPIO.OUT)
        GPIO.setup(pintrash,GPIO.OUT)
        GPIO.setup(pinmetal,GPIO.OUT)
        GPIO.setup(pinglass,GPIO.OUT)
        GPIO.output(pinplast,GPIO.HIGH)
        GPIO.output(pintrash,GPIO.HIGH)
        GPIO.output(pinmetal,GPIO.LOW)
        GPIO.output(pinglass,GPIO.HIGH)
        time.sleep(time_ch3)
        ultrasonic.checkfull(typebin)
        time.sleep(time_ch3-3)
        GPIO.output(pinmetal,GPIO.HIGH)

    if (typebin == 'glass') : # เป็นแก้ว ตั้งค่าให้ pinglass มีสถานะ LOW
        GPIO.setup(pinplast,GPIO.OUT)
        GPIO.setup(pintrash,GPIO.OUT)
        GPIO.setup(pinmetal,GPIO.OUT)
        GPIO.setup(pinglass,GPIO.OUT)
        GPIO.output(pinplast,GPIO.HIGH)
```

```
        GPIO.output(pintrash,GPIO.HIGH)
        GPIO.output(pinmetal,GPIO.HIGH)
        GPIO.output(pinglass,GPIO.LOW)
        time.sleep(time_ch4)
        ultrasonic.checkfull(typebin)
        time.sleep(time_ch4-3)
        GPIO.output(pinglass,GPIO.HIGH)
```

ชื่อไฟล์ : Ultrasonic.py

ภาษาที่ใช้ : Python

คำอธิบาย : โค้ดฟังก์ชันการทำงานของส่วนที่จะตรวจสอบว่าปริมาณขยะในถังเต็มหรือไม่ โดย Ultrasonic Sensor เช่นเดียวกับการตรวจสอบว่ามีขยะเข้ามาในกล่องรับหรือยัง แต่จะใช้ในกรณีที่ถังขยะใกล้เต็ม หรือเต็ม จะมีการแจ้งเตือนผ่านทาง Line Notify เพื่อแจ้งให้เจ้าหน้าที่มาทำการเก็บขยะ

โดยโค้ดส่วนนี้เป็นตัวดำเนินการย่อย ที่โปรแกรมหลักจะเรียกใช้งาน

```
#การนำเข้า Module อื่น ๆเพื่อนำมาเรียกใช้งานในโปรแกรม
import RPi.GPIO as GPIO
import time
import requests

GPIO.setmode(GPIO.BCM)
TRIG = 8
ECHO = 7
GPIO.setup(TRIG,GPIO.OUT)
GPIO.setup(ECHO,GPIO.IN)
global msg

class Ultrasonic:
    def checkfull (self,typebin): # การกำหนดเพื่อใช้ในการเข้าถึง Line Notify และ เงื่อนไขกรณีถังขยะใกล้เต็ม/เต็ม
        url = 'https://notify-api.line.me/api/notify'
        token = 'RuVYrUg3xGwxSgeesncMkKbRTUeJ5JCwyaNwdy5i5Ql'
        headers = {'content-type':'application/x-www-form-urlencoded','Authorization':'Bearer '+token}
        GPIO.output(TRIG, False)
        time.sleep (0.01)
        GPIO.output(TRIG, True)
        time.sleep (0.0001)
        GPIO.output(TRIG, False)

        while GPIO.input(ECHO)==0:
            time_start = time.time()

        while GPIO.input(ECHO)==1:
            time_end = time.time()

        time_duration = time_end - time_start
```

```
    distance = time_duration * 17150
    distance = round (distance, 2)
        if (distance < 5): # เงื่อนไขเมื่อถังขยะเต็ม
        msg = 'The ' + typebin + ' bin is full. And distance : ' + str (distance) + ' cm'
        r = requests.post(url, headers=headers, data = {'message':msg})

elif (distance < 10): # เงื่อนไขเมื่อถังขยะใกล้เต็ม
        msg ='The ' + typebin + ' bin is almost full. And distance : ' + str (distance) + ' cm'
        r = requests.post (url, headers=headers, data = {'message':msg})
else:
        pass
```

ภาคผนวก ค

ผลการ Train ของทุก model โดยมี 100 model

ตารางที่ ค - 1 : ผลการ Train ของทั้งหมด 100 model

No.	Model	Train (%)	Validation (%)	Test (%)	Epochs	Learning rate	Accuracy	Training time (min)	Model size
1	squeezenet1_0	50	25	25	40	4.37E-03	90.65	141.53	5.1 MB
2	squeezenet1_0	50	25	25	40	5.75E-04	89.18	125.87	5.1 MB
3	squeezenet1_0	60	20	20	40	6.92E-04	91.18	157.09	5.1 MB
4	squeezenet1_0	60	20	20	40	5.25E-03	93.16	131.13	5.1 MB
5	squeezenet1_0	70	15	15	40	1.32E-02	92.31	149.58	5.1 MB
6	squeezenet1_0	70	15	15	40	4.37E-03	92.48	142.69	5.1 MB
7	squeezenet1_0	70	15	15	40	2.51E-03	93.35	153.55	5.1 MB
8	squeezenet1_0	80	10	10	40	8.32E-04	91.57	152.22	5.1 MB
9	squeezenet1_0	80	10	10	40	3.63E-03	92.37	150.65	5.1 MB
10	squeezenet1_0	80	10	10	40	5.75E-02	86.84	150.64	5.1 MB
11	alexnet	50	25	25	40	9.12E-03	87.92	5.68	11 MB
12	alexnet	50	25	25	40	5.25E-03	88.13	5.64	11 MB
13	alexnet	60	20	20	40	7.59E-03	90.39	6.29	11 MB
14	alexnet	60	20	20	40	7.59E-03	90	6.29	11 MB
15	alexnet	70	15	15	40	5.25E-03	90.2	48.36	11 MB
16	alexnet	70	15	15	40	1.74E-03	88.98	47.3	11 MB
17	alexnet	70	15	15	40	6.31E-03	90.55	49.02	11 MB
18	alexnet	80	10	10	40	3.63E-03	90.26	50.98	11 MB
19	alexnet	80	10	10	40	7.59E-03	88.94	51.49	11 MB
20	alexnet	80	10	10	40	3.63E-03	89.73	51.74	11 MB
21	vgg16_bn	50	25	25	40	5.25E-03	95.9	34.73	61.1 MB

22	vgg16_bn	50	25	25	40	1.74E-03	95.58	34.72	61.1 MB
23	vgg16_bn	60	20	20	40	1.45E-03	96.84	38.98	61.1 MB
24	vgg16_bn	60	20	20	40	5.25E-03	96.31	38.93	61.1 MB
25	vgg16_bn	70	15	15	40	1.74E-03	95.46	43.26	61.1 MB
26	vgg16_bn	70	15	15	40	2.51E-03	96.5	43	61.1 MB
27	vgg16_bn	70	15	15	40	5.25E-03	97.2	43.28	61.1 MB
28	vgg16_bn	80	10	10	40	2.09E-03	97.1	47.75	61.1 MB
29	vgg16_bn	80	10	10	40	5.25E-03	98.42	47.82	61.1 MB
30	vgg16_bn	80	10	10	40	7.59E-03	96.57	48.95	61.1 MB
31	vgg19_bn	50	25	25	40	3.02E-03	95.69	39.4	82.4 MB
32	vgg19_bn	50	25	25	40	4.37E-03	95.16	40.66	82.4 MB
33	vgg19_bn	60	20	20	40	5.25E-03	97.23	44.15	82.4 MB
34	vgg19_bn	60	20	20	40	3.63E-03	97.36	44.19	82.4 MB
35	vgg19_bn	70	15	15	40	2.51E-03	96.85	49.03	82.4 MB
36	vgg19_bn	70	15	15	40	3.02E-03	97.55	49.08	82.4 MB
37	vgg19_bn	70	15	15	40	9.12E-03	95.63	48.96	82.4 MB
38	vgg19_bn	80	10	10	40	5.25E-03	98.68	53.87	82.4 MB
39	vgg19_bn	80	10	10	40	5.25E-03	97.63	63.48	82.4 MB
40	vgg19_bn	80	10	10	40	4.37E-03	99.47	53.89	82.4 MB
41	densenet121	50	25	25	40	1.20E-03	96.84	25.82	32.6 MB
42	densenet121	50	25	25	40	3.63E-03	96.42	25.79	32.6 MB
43	densenet121	60	20	20	40	2.09E-03	97.89	29.37	32.6 MB
44	densenet121	60	20	20	40	3.31E-04	97.1	29.38	32.6 MB
45	densenet121	70	15	15	40	1.20E-03	97.9	32.96	32.6 MB
46	densenet121	70	15	15	40	2.09E-03	97.37	32.89	32.6 MB
47	densenet121	70	15	15	40	2.51E-03	97.9	33.59	32.6 MB
48	densenet121	80	10	10	40	1.74E-03	98.15	36.58	32.6 MB

49	densenet121	80	10	10	40	3.02E-03	98.94	36.25	32.6 MB
50	densenet121	80	10	10	40	1.74E-03	98.15	36.44	32.6 MB
51	ResNet34	50	25	25	40	9.12E-03	94.75	10.89	87.4 MB
52	ResNet34	50	25	25	40	2.09E-03	96.32	10.83	87.4 MB
53	ResNet34	60	20	20	40	2.51E-03	96.11	10.93	87.4 MB
54	ResNet34	60	20	20	40	1.45E-03	97.24	12.05	87.4 MB
55	ResNet34	70	15	15	40	8.32E-04	97.55	13.38	87.4 MB
56	ResNet34	70	15	15	40	1.32E-02	94.58	13.3	87.4 MB
57	ResNet34	70	15	15	40	9.12E-03	95.62	13.45	87.4 MB
58	ResNet34	80	10	10	40	1.45E-03	97.89	14.81	87.4 MB
59	ResNet34	80	10	10	40	1.74E-03	97.63	14.66	87.4 MB
60	ResNet34	80	10	10	40	2.09E-03	98.68	14.71	87.4 MB
61	ResNet50	50	25	25	40	3.63E-03	96.64	22.04	98 MB
62	ResNet50	50	25	25	40	4.79E-04	96.74	22.02	98 MB
63	ResNet50	60	20	20	40	6.92E-04	97.76	24.87	98 MB
64	ResNet50	60	20	20	40	5.75E-04	97.1	24.99	98 MB
65	ResNet50	70	15	15	40	3.63E-03	98.25	27.72	98 MB
66	ResNet50	70	15	15	40	5.75E-04	97.55	27.78	98 MB
67	ResNet50	70	15	15	40	4.79E-04	98.07	27.68	98 MB
68	ResNet50	80	10	10	40	7.41E-04	98.15	30.48	98 MB
69	ResNet50	80	10	10	40	8.32E-04	99.21	30.52	98 MB
70	ResNet50	80	10	10	40	5.75E-04	98.68	30.65	98 MB
71	ResNet18	50	25	25	40	5.25E-03	94.43	7.44	46.9 MB
72	ResNet18	50	25	25	40	2.09E-03	95.27	7.42	46.9 MB
73	ResNet18	60	20	20	40	6.31E-03	96.32	8.24	46.9 MB
74	ResNet18	60	20	20	40	1.10E-02	97.63	8.2	46.9 MB
75	ResNet18	70	15	15	40	1.45E-03	97.37	9.04	46.9 MB

76	ResNet18	70	15	15	40	1.20E-03	97.9	9.02	46.9 MB
77	ResNet18	70	15	15	40	2.51E-03	97.03	9.02	46.9 MB
78	ResNet18	80	10	10	40	3.63E-03	97.63	9.85	46.9 MB
79	ResNet18	80	10	10	40	3.63E-03	98.95	9.87	46.9 MB
80	ResNet18	80	10	10	40	1.32E-02	96.57	9.81	46.9 MB
81	ResNet101	50	25	25	40	4.79E-04	96.42	34.32	179 MB
82	ResNet101	50	25	25	40	5.75E-04	96	34.32	179 MB
83	ResNet101	60	20	20	40	8.32E-04	98.29	38.8	179 MB
84	ResNet101	60	20	20	40	3.02E-03	97.5	38.71	179 MB
85	ResNet101	70	15	15	40	2.75E-04	97.55	43.17	179 MB
86	ResNet101	70	15	15	40	1.74E-03	99.3	43.17	179 MB
87	ResNet101	70	15	15	40	1.20E-03	97.9	43.19	179 MB
88	ResNet101	80	10	10	40	1.20E-03	98.95	47.52	179 MB
89	ResNet101	80	10	10	40	1.20E-03	98.42	47.94	179 MB
90	ResNet101	80	10	10	40	8.32E-04	98.42	48.9	179 MB
91	squeezenet1_1	50	25	25	40	1.20E-03	88.55	69.32	5.05 MB
92	squeezenet1_1	50	25	25	40	3.02E-03	88.13	69.64	5.05 MB
93	squeezenet1_1	60	20	20	40	3.02E-03	90.92	73.77	5.05 MB
94	squeezenet1_1	60	20	20	40	1.20E-03	90.26	73.78	5.05 MB
95	squeezenet1_1	70	15	15	40	1.58E-02	91.25	82.67	5.05 MB
96	squeezenet1_1	70	15	15	40	9.12E-03	91.43	78	5.05 MB
97	squeezenet1_1	70	15	15	40	3.31E-04	90.56	79.6	5.05 MB
98	squeezenet1_1	80	10	10	40	3.63E-03	92.89	94.25	5.05 MB
99	squeezenet1_1	80	10	10	40	1.10E-02	91.84	96.74	5.05 MB
100	squeezenet1_1	80	10	10	40	8.32E-04	91.57	100.87	5.05 MB

ผลการ Test จริงของทุก model โดยมีทั้งหมด 100 model

ตารางที่ ค - 2 : ผลการ Test จริงของทั้งหมด 100 model

No.	Model	Glass (25 images)	Metal (25 images)	Plastic (25 images)	Trash (25 images)	Accuracy (%)
1	squeezenet1_0	22	11	20	8	61
2	squeezenet1_0	21	18	18	7	64
3	squeezenet1_0	22	14	19	5	60
4	squeezenet1_0	19	16	22	7	64
5	squeezenet1_0	18	15	22	8	63
6	squeezenet1_0	18	15	23	8	64
7	squeezenet1_0	17	15	23	8	63
8	squeezenet1_0	19	18	19	10	66
9	squeezenet1_0	18	19	21	8	66
10	squeezenet1_0	23	15	12	10	60
11	alexnet	6	21	21	9	57
12	alexnet	7	20	22	8	57
13	alexnet	8	21	22	8	59
14	alexnet	9	21	20	9	59
15	alexnet	6	22	24	6	58
16	alexnet	10	20	18	9	57
17	alexnet	9	22	23	6	60
18	alexnet	2	19	23	7	51
19	alexnet	4	22	22	6	54
20	alexnet	6	21	19	9	55
21	vgg16_bn	21	18	18	15	72
22	vgg16_bn	17	18	21	10	66

23	vgg16_bn	17	12	22	14	65
24	vgg16_bn	20	18	17	15	70
25	vgg16_bn	20	19	18	13	70
26	vgg16_bn	15	16	23	11	65
27	vgg16_bn	15	16	11	14	56
28	vgg16_bn	18	14	23	10	65
29	vgg16_bn	20	20	21	17	78
30	vgg16_bn	24	17	19	14	74
31	vgg19_bn	20	19	19	15	73
32	vgg19_bn	19	12	19	11	61
33	vgg19_bn	18	21	21	13	73
34	vgg19_bn	18	19	23	12	72
35	vgg19_bn	22	20	21	13	76
36	vgg19_bn	18	15	14	15	62
37	vgg19_bn	21	16	21	14	72
38	vgg19_bn	18	17	23	11	69
39	vgg19_bn	21	16	17	14	68
40	vgg19_bn	21	18	19	12	70
41	densenet121	19	22	24	14	79
42	densenet121	19	10	24	5	58
43	densenet121	17	13	25	8	63
44	densenet121	16	21	25	9	71
45	densenet121	20	16	20	10	66
46	densenet121	17	18	24	11	70
47	densenet121	17	21	22	8	68
48	densenet121	20	15	24	9	68
49	densenet121	19	17	22	8	66

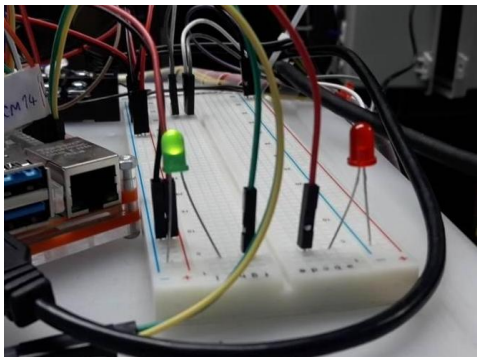
50	densenet121	17	14	24	10	65
51	ResNet34	17	11	24	11	63
52	ResNet34	15	16	24	6	61
53	ResNet34	17	15	24	13	69
54	ResNet34	16	16	25	9	66
55	ResNet34	21	14	23	13	71
56	ResNet34	17	15	17	20	69
57	ResNet34	22	22	14	14	72
58	ResNet34	22	17	22	9	70
59	ResNet34	20	14	23	13	70
60	ResNet34	22	11	19	18	70
61	ResNet50	19	20	20	18	77
62	ResNet50	20	13	25	10	68
63	ResNet50	22	18	25	16	81
64	ResNet50	15	12	25	12	64
65	ResNet50	16	18	24	16	74
66	ResNet50	22	16	23	19	80
67	ResNet50	20	14	23	16	73
68	**ResNet50**	**23**	**18**	**22**	**21**	**84**
69	ResNet50	20	19	22	19	80
70	ResNet50	21	16	22	18	77
71	ResNet18	15	9	23	13	60
72	ResNet18	24	14	22	12	72
73	ResNet18	14	11	25	13	63
74	ResNet18	16	17	22	9	64
75	ResNet18	21	15	20	14	70
76	ResNet18	18	17	23	10	68

77	ResNet18	22	12	22	12	68
78	ResNet18	17	11	21	17	66
79	ResNet18	19	15	21	14	69
80	ResNet18	15	14	23	15	67
81	ResNet101	20	18	25	16	79
82	ResNet101	19	18	22	19	78
83	ResNet101	21	21	19	20	81
84	ResNet101	17	20	22	15	74
85	ResNet101	22	21	22	12	77
86	ResNet101	19	14	21	11	65
87	ResNet101	19	23	22	16	80
88	ResNet101	22	19	22	15	78
89	ResNet101	16	23	24	14	77
90	ResNet101	23	22	22	13	80
91	squeezenet1_1	15	17	20	7	59
92	squeezenet1_1	13	15	22	10	60
93	squeezenet1_1	13	16	22	6	57
94	squeezenet1_1	13	14	21	7	55
95	squeezenet1_1	12	16	22	9	59
96	squeezenet1_1	10	20	18	14	62
97	squeezenet1_1	11	17	22	8	58
98	squeezenet1_1	14	15	21	9	59
99	squeezenet1_1	11	15	21	9	56
100	squeezenet1_1	13	17	23	9	62

ภาคผนวก ง

คู่มือการใช้งานถังขยะ

1. รอสัญญาณไฟสีเขียว แสดงว่าสามารถทิ้งขยะได้ ดังภาพที่ ง - 1

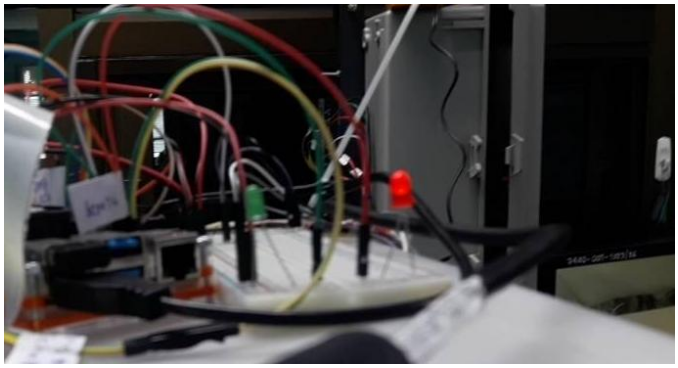


ภาพที่ ง - 1 : สัญญาณไฟแสดงว่าเครื่องพร้อมทำงาน

2. ใส่ขยะลงในช่องรับขยะให้ถูกต้อง โดยยื่นมือเข้าไปในกล่องแล้ววางขยะ เพื่อให้ Ultrasonic Sensor สามารถตรวจจับการนำเข้าของขยะได้ ดังภาพที่ ง - 2

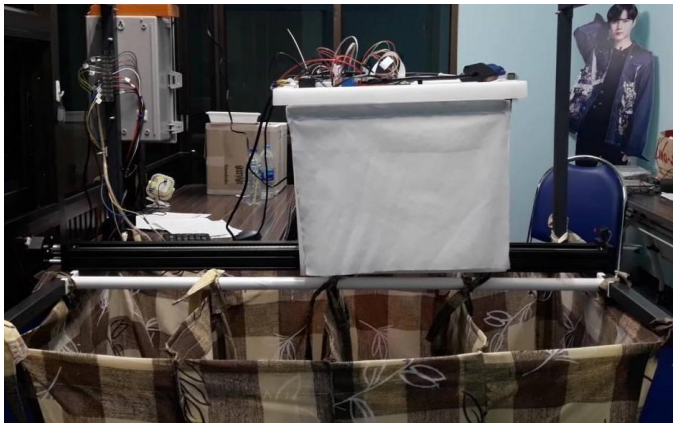
ภาพที่ ง - 2 : ใส่ขยะในช่องรับขยะ

3. หลังจากนำมือออก สัญญาณไฟสีเขียวจะเปลี่ยนเป็นสีแดง แสดงว่าอยู่ในระหว่างการประมวลผล ในระหว่างนี้ให้รอ ดังภาพที่ ง - 3



ภาพที่ ง - 3 : สัญญาณไฟเปลี่ยนเป็นสีแดง เครื่องกำลังประมวลผล

4. เครื่องจะทำงานตามกระบวนการโดยอัตโนมัติ แล้วกล่องรับจะเลื่อนไปตรงกับช่องถังของขยะประเภทนั้นๆ ดังภาพที่ ง - 4



ภาพที่ ง - 4 : กล่องรับขยะเคลื่อนที่ไปยังตำแหน่งถังที่ถูกต้อง

5. ตัวดันจะผลักขยะให้ลงถังแล้วเลื่อนกลับที่ตำแหน่งเดิม ดังภาพที่ ง - 5

ภาพที่ ง - 5 : ผลักขยะลงถังที่ทำนายได้

6. สัญญาณไฟสีแดงจะเปลี่ยนเป็นสีเขียวอีกครั้ง เพื่อพร้อมรับขยะชิ้นต่อไป ตามข้อที่ 1.