การศึกษาและการประยุกต์ใช้งานการเรียนรู้ของเครื่องบนไมโครคอนโทรลเลอร์

APPLICATION AND LEARNING OF MACHINE LEARNING ON MICROCONTROLLER

เกสราวดี อักษรรัตน์

KESARAWADEE AUGSORNRAT

อดินันท์ ตุ้มฉาย

ADINAN TUMCHAY

อารยา ปราบภัย

ARAYA PRABPHAI

ปริญญานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาวิศวกรรมศาสตรบัณฑิต

สาขาวิชาวิศวกรรมสารสนเทศ

คณะวิศวกรรมศาสตร์

สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

ปีการศึกษา 2565

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

APPLICATION AND LEARNING OF MACHINE LEARNING ON MICROCONTROLLER

KESARAWADEE AUGSORNRAT

ADINAN TUMCHAY

ARAYA PRABPHAI

THIS THESIS IS SUBMITTED IN PARTIAL FULFILLMENT

OF THE REQUIREMENT FOR THE DEGREE OF

BACHELOR OF ENGINEERING IN INFORMATION ENGINEERING

SCHOOL OF ENGINEERING

KING MONGKUT'S INSTITUTE OF TECHNOLOGY LADKRABANG

ACADEMIC YEAR 2022

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

หัวข้อปริญญานิพนธ์	การศึกษาและการประยุกต์ใช้งานการเรียนรู้ของเครื่องบนไมโครคอนโทรลเลอร์	
รายชื่อนักศึกษา	นางสาวเกสราวดี อักษรรัตน์	รหัสนักศึกษา 62010076
	นายอดินันท์ ตุ้มฉาย	รหัสนักศึกษา 62010999
	นางสาวอารยา ปราบภัย	รหัสนักศึกษา 62011065
ปริญญา	วิศวกรรมศาสตรบัณฑิต	
สาขาวิชา	วิศวกรรมสารสนเทศ	
พ.ศ.	2565	
อาจารย์ที่ปรึกษาปริญญานิพนธ์	รศ.ดร.บุณย์ชนะ ภู่ระหงษ์	
	ดร.ธนวิชญ์ อนุวงศ์พินิจ	

ปริญญานิพนธ์ฉบับนี้ ได้รับการอนุมัติให้เป็นส่วนหนึ่งของการศึกษา ตามหลักสูตรวิศวกรรมศาสตร์บัณฑิต คณะวิศวกรรมศาสตร์ สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหาร ลาดกระบัง

.....

(รศ.ดร.บุณย์ชนะ ภู่ระหงษ์)

.....

(ดร.ธนวิชญ์ อนุวงศ์พินิจ)

อาจารย์ผู้ควบคุมปริญญานิพนธ์

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

หัวข้อปริญญานิพนธ์ การศึกษาและการประยุกต์ใช้งานการเรียนรู้ของเครื่องบนไมโครคอนโทรลเลอร์

รายชื่อนักศึกษา นางสาวเกสราวดี อักษรรัตน์ รหัสนักศึกษา 62010076
นายอดินันท์ ตุ้มฉาย รหัสนักศึกษา 62010999
นางสาวอารยา ปราบภัย รหัสนักศึกษา 62011065

ปริญญา วิศวกรรมศาสตรบัณฑิต

สาขาวิชา วิศวกรรมสารสนเทศ

พ.ศ. 2565

อาจารย์ที่ปรึกษาปริญญานิพนธ์ รศ.ดร.บุณย์ชนะ ภู่ระหงษ์
ดร.ธนวิชญ์ อนุวงศ์พินิจ

บทคัดย่อ

ในปัจจุบันการเรียนรู้ของเครื่องหรือ Machine Learning เป็นเทคโนโลยีที่มีการใช้แพร่หลาย แต่ก็มีข้อจำกัดในด้านต้นทุนที่สูงและอุปกรณ์ที่ใช้ในการพัฒนาจำเป็นต้องมีกำลังประมวลผลที่สูง ดังนั้นจึงได้มีการพัฒนาเทคโนโลยีที่ทำให้สามารถใช้งาน Machine Learning บนอุปกรณ์ปลายทางขนาดเล็กหรือไมโครคอนโทรลเลอร์ได้หรือ TinyML วัตถุประสงค์ของการวิจัยนี้คือการศึกษา TinyML ด้วยการนำความรู้เรื่องการเรียนรู้ของเครื่องและไมโครคอนโทรลเลอร์มาศึกษาและเรียนรู้ในการต่อยอดใช้ร่วมกัน โดยเริ่มศึกษาเปรียบเทียบผลลัพธ์ของ TensorFlow ซึ่งเป็นเครื่องมือสำหรับสร้าง Machine Learning และ TensorFlow lite ที่มีไว้สำหรับแปลงขนาดของโมเดลให้สามารถใช้งานบนอุปกรณ์ไมโครคอนโทรลเลอร์ได้พบว่าหลังผ่านกระบวนการแปลงขนาดของโมเดลเล็กลงและสามารถทำงานบนบอร์ดขนาดเล็ก เช่น Arduino nano BLE 33 sense หรือ Xiao nRF52840 Sense เป็นต้น แต่ความแม่นยำที่ได้จะลดลงในระดับที่สามารถนำไปใช้งานได้ ต่อมาได้ศึกษาและพบ edge impulse ซึ่งเป็นเครื่องมือสำหรับทำ Machine Learning และแปลงให้สามารถทำงานบนอุปกรณ์ปลายทางชนิดต่าง ๆ ในแพลตฟอร์มเดียวได้ ผู้จัดทำจึงได้เลือกที่จะนำ edge impulse มาใช้เป็นต้นแบบในการศึกษาและทดสอบ โดยเลือกการชี้คำเฉพาะ การตรวจจับทางท่า และการตรวจจับวัตถุมาเป็นตัวอย่างสำหรับเป็นต้นแบบในการทำวิจัย โดยหลังการทดสอบด้วยการ training บน edge impulse และทำการติดตั้งลงบนบอร์ด Xiao nRF52840 Sense สำหรับ การชี้คำเฉพาะ และการตรวจจับท่าทาง จะเก็บชุดข้อมูลด้วยเซนเซอร์ที่อยู่บนบอร์ดได้ผลความแม่นยำในระหว่างการ training อยู่ที่ 80-100% แต่เมื่อนำไปใช้จริงยังพบว่ามีความผิดพลาดในการใช้งานอยู่บ้าง ส่วนการทดสอบการตรวจจับวัตถุจะใช้กล้องที่ติดกับคอมพิวเตอร์หรืออุปกรณ์มือถือและติดตั้งลงบนอุปกรณ์ปลายทางที่มีกล้อง ได้ผลลัพธ์ความแม่นยำที่ 80-90% และผลลัพธ์การทดสอบที่น่าพึงพอใจพร้อมใช้งานจริง ในท้ายสุดผู้จัดทำยังได้นำสิ่งที่ได้ศึกษามาออกแบบเป็นแบบเรียนสำหรับพัฒนาให้ผู้เริ่มต้นที่สนใจจะมาเรียนรู้เกี่ยวกับ TinyML สามารถทำความเข้าใจพื้นฐานและภาพรวมของ TinyML ในเบื้องต้นและสามารถนำไปประยุกต์ใช้ต่อกับงานของตนได้

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

Thesis Title	Application and Learning of Machine Learning on Microcontroller	
Student	Miss Kesarawadee Augsornrat	Student ID. 62010076
	Mr. Adinan Tumchay	Student ID. 62010999
	Miss Araya Prabphai	Student ID. 62011065
Degree	Bachelor of Engineering	
Program	Information Engineering	
Year	2022	
Thesis Advisor	Assoc.Prof.Dr.Boonchana Purahong	
	Dr.Thanavit Anuwongpinit	

ABSTRACT

Nowadays, machine learning is a technology that is widely used, but it is limited in terms of high cost and high computing power required to used it. Therefore, a technology that enables Machine Learning to be implemented on small devices or microcontrollers, called TinyML, has been developed. The purpose of this research is to study machine learning on microcontrollers by applying machine learning and microcontrollers knowledge to study and learn to use TinyML, start with compare the results of TensorFlow, a machine learning tool, and TensorFlow lite, which is intended for converting model sizes to use on microcontroller devices, found that after the conversion process, the model size is smaller and can run on the small board, but the accuracy is reduced to the level that can be used. Later, we studied edge impulse, which is a tool for doing Machine Learning and converting it to be able to work on different types of endpoints in a single platform. We have chosen to use the edge impulse as a model for study and testing. By selecting a keywords spotting, gesture detection and object detection Come as an example for a prototype to do research. After training on the edge impulse and installing it on the Xiao nRF52840 Sense board for keywords spotting and gesture detection, it collects datasets with on-board sensors and results in 80-100% accuracy during training, but in actual use there are still some errors in use. An object detection test uses a camera attached to a computer or mobile device and mounted on a terminal equipped with a camera. Achieve 80-90% accuracy results and satisfactory test results are ready for use. In the end, we also brought what has been studied to design a developmental course for beginners who are interested in learning about TinyML to be able to understand the basics and overview of TinyML in the beginning and be able to apply it.

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

กิตติกรรมประกาศ

ปริญญานิพนธ์ฉบับนี้จะไม่สามารถสำเร็จลุล่วงไปได้ด้วยดี หากไม่ได้รับความสนับสนุนจากอาจารย์หลาย ๆ ท่านในภาควิชาวิศกรรมสารสนเทศ คณะวิศวกรรมศาสตร์ สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง โดยเฉพาะอย่างยิ่ง อาจารย์บุณย์ชนะ ภู่ระหงษ์ และอาจารย์ธนวิชญ์ อนุวงศ์พินิจ ผู้เป็นที่ปรึกษาในโครงงานนี้ ที่ได้ให้คำแนะนำที่เป็นประโยชน์ตลอดมา

คณะผู้จัดทำ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

สารบัญ

บทคัดย่อ ..... ก

ABSTRACT ..... ข

กิตติกรรมประกาศ ..... ค

บทที่ 1 บทนำ 1

1.1 แนวคิดและที่มา 1

1.2 วัตถุประสงค์การวิจัย 2

1.3 ความสำคัญของการวิจัย 2

1.4 สมมติฐานการวิจัย 2

1.5 ขอบเขตของการวิจัย 2

บทที่ 2 ทฤษฎีที่เกี่ยวข้อง 3

2.1 Microcontroller 3

2.2 Microprocessor 5

2.3 ข้อแตกต่างระหว่าง Microcontroller กับ Microprocessor 5

2.4 Machine Learning 6

2.5 TensorFlow 9

2.6 Tiny Machine Learning 10

2.6.1 ข้อดีและข้อเสียของ TINY MACHINE LEARNING 10

2.6.2 แนวทางการใช้ TINY MACHINE LEARNING ในปัจจุบัน 11

2.6.2.1 Arduino nano 33 BLE 12

2.6.2.2 Seeed Studio XIAO nRF52840 Sense 13

2.6.2.3 Syntiant’s Tiny Machine Learning (NDP101) 14

2.6.2.4 Arducam Pico4ML 15

2.6.2.5 SparkFun Edge 16

2.6.2.6 Arduino Portenta H7 17

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

2.6.2.7 Wio Terminal(ATSAMD51P19) 18
2.7 Edge Impulse 19
บทที่ 3 วิธีดำเนินการวิจัย 20
3.1 ศึกษาเนื้อหาที่เกี่ยวข้อง 22
3.1.1 ความแตกต่างของไมโครคอนโทรลเลอร์และไมโครโพรเซสเซอร์ 22
3.1.2 MACHINE LEARNING และ TENSORFLOW 22
3.1.3 TINY MACHINE LEARNING 22
3.1.4 EDGE IMPULSE 22
3.2 การวิเคราะห์การทำงาน 23
3.2.1 อุปกรณ์ไมโครคอนโทรลเลอร์ที่เลือกมาใช้ในการควบคุมการเรียนรู้ของเครื่อง 23
3.2.2 กระบวนการแปลง TENSORFLOW ด้วย TENSORFLOW LITE 24
3.2.2.1 Convert Model 24
3.2.2.2 Optimization 24
3.2.2.2.1 Automatic Optimization 25
3.2.2.2.2 Quantization 25
3.2.2.3 Quantization-aware Training 25
3.2.3 TENSORFLOW LITE MICRO 25
3.2.3.1 นิยามของ TensorFlow lite Micro 26
3.2.4 EDGE IMPULSE 26
3.2.4.1 Train tiny model 26
3.2.4.2 Model optimization 27
3.2.4.2 Deployment 28
3.3 การทดลองใช้งาน 29
3.3.1 การทดสอบชี้เฉพาะคำสำคัญด้วยการเรียนรู้ของเครื่องบนไมโครคอนโทรลเลอร์ 29
3.3.1.1 Collecting Data 29
3.3.1.2 Preprocess Data 29

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

3.3.1.3 Design Model 30
3.3.1.4 Training Model 30
3.3.1.5 Evaluate Optimize และ Convert Model 30
3.3.1.6 Deploy Model 31
3.3.2 การทดสอบคาดเดาท่าทางการขยับด้วยการเรียนรู้ของเครื่องด้วยหน่วยตรวจวัดการเคลื่อนไหวภายใน 31
3.3.2.1 Collecting Data 31
3.3.2.2 Preprocess Data 31
3.3.2.3 Design Model 32
3.3.1.4 Training Model 32
3.3.1.5 Evaluate Optimize และ Convert Model 32
3.3.1.6 Deploy Model 32
3.3.3 การทดสอบการตรวจจับวัตถุด้วยการเรียนรู้ของเครื่องบนอุปกรณ์ไมโครคอนโทรลเลอร์ 32
3.3.3.1 Collecting Data 32
3.3.2.3 Design Model 32
3.3.1.4 Training Model 32
3.3.1.5 Evaluate Optimize และ Convert Model 33
3.3.1.6 Deploy Model 33
3.4 การพัฒนาต้นแบบ 33
บทที่ 4 ผลการดำเนินการวิจัย 34
4.1 การเปรียบเทียบผลที่ได้ระหว่าง TensorFlow และ TensorFlow lite 34
4.1.1 ผลจากการ CONVERT MODEL 34
4.1.2 ผลที่ได้จากการ OPTIMIZATION และ QUANTIZING 38
4.2 การทดลองใช้งาน TensorFlow lite และนำไปใช้บนบอร์ด Arduino nano 33 BLE 41
4.2.1 ขั้นตอนการทดสอบชี้เฉพาะคำสำคัญด้วยการเรียนรู้ของเครื่องบนไมโครคอนโทรลเลอร์ 41
4.2.1.1 Collecting Data 41

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

4.2.1.2 Preprocessing and Model 41
4.2.1.3 Training Model 44
4.2.1.4 Evaluate Optimize และ Convert Model 44
4.2.1.5 Set up Code 46
4.2.2 ผลการทดสอบชี้เฉพาะคำสำคัญด้วยการเรียนรู้ของเครื่องบนไมโครคอนโทรลเลอร์ 52
4.2.3 ขั้นตอนการทดสอบคาดเดาท่าทางการขยับด้วยการเรียนรู้ของเครื่องด้วยหน่วยตรวจวัดการเคลื่อนไหวภายใน 55
4.2.3.1 Collecting Data 55
4.2.1.2 Preprocessing and Model 55
4.2.2.3 Training Model 60
4.2.1.4 Evaluate Optimize และ Convert Model 61
4.2.1.5 Set up Code 63
4.2.4 ผลการทดสอบคาดเดาท่าทางการขยับด้วยการเรียนรู้ของเครื่องด้วยหน่วยตรวจวัดการเคลื่อนไหวภายใน 69
4.3 การทดลองใช้งาน Edge Impulse 71
4.3.1 ขั้นตอนการทดสอบชี้เฉพาะคำสำคัญด้วยการเรียนรู้ของเครื่องบนไมโครคอนโทรลเลอร์โดย EDGE IMPULSE 71
4.3.1.1 Collecting Data 71
4.3.1.2 Training Model 75
4.3.1.3 Deploy Code 77
4.3.1.4 ผลการทดสอบชี้เฉพาะคำสำคัญด้วยการเรียนรู้ของเครื่องบนไมโครคอนโทรลเลอร์โดย EDGE IMPULSE 80
4.3.2 ขั้นตอนการทดสอบคาดเดาท่าทางการขยับด้วยการเรียนรู้ของเครื่องด้วยหน่วยตรวจวัดการเคลื่อนไหวภายในโดย EDGE IMPULSE 81
4.3.2.1 Collecting Data 81
4.3.2.2 Training Model 82
4.3.2.3 Deploy Code 83

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

4.4 การทดลองการตรวจจับวัตถุด้วยการเรียนรู้ของเครื่องบนอุปกรณ์ไมโครคอนโทรลเลอร์ 84
4.4.1 COLLECTING DATA 84
4.4.2 TRAINING MODEL 85
4.3.1.2 Deploy Code 86
4.5 การจัดทำแบบเรียนเพื่อการศึกษาการเรียนของเครื่องบนอุปกรณ์ไมโครคอนโทรลเลอร์ 87
4.6 การนำไปประยุกต์ใช้ 151
บทที่ 5 สรุปผลและข้อเสนอแนะ 153
5.1 สรุปผล 153
5.2 ปัญหาที่พบในการวิจัย 154
5.3 ข้อเสนอแนะ 154
บรรณานุกรม 155

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

สารบัญตาราง

ตาราง 2.1 ตารางเปรียบเทียบ Microcontroller และ Microprocessor 5
ตาราง 2.2 ตารางข้อดีและข้อเสียของ Machine Learning 9
ตาราง 4.1 ตารางผลการทดลอง 152

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

สารบัญรูป

ภาพที่ 2.1 โครงสร้างทั่วไปของไมโครคอนโทรลเลอร์ 3
ภาพที่ 2.2 Arduino nano 33 BLE 12
ภาพที่ 2.3 Seeed Studio XIAO nRF52840 Sense 13
ภาพที่ 2.4 Syntiant’s Tiny Machine Learning (NDP101) 14
ภาพที่ 2.5 Arducam Pico4ML 15
ภาพที่ 2.6 SparkFun Edge 16
ภาพที่ 2.7 Arduino Portenta H7 17
ภาพที่ 2.8 Wio Terminal(ATSAMD51P19) 18

ภาพที่ 3.1 Arduino nano 33 BLE 23
ภาพที่ 3.2 Seeed Studio XIAO nRF52840 (Sense) 23
ภาพที่ 3.3 แผนภาพกระบวนการแปลง TensorFlow ด้วย TensorFlow lite 24
ภาพที่ 3.4 ภาพแสดงการ Convert Model 24
ภาพที่ 3.5 การ Quantization 25
ภาพที่ 3.6 Edge Impulse 26
ภาพที่ 3.7 Train tiny model 27
ภาพที่ 3.8 Model optimization 28
ภาพที่ 3.9 Deployment 28
ภาพที่ 3.10 Keyword Spotting (KWS) 29
ภาพที่ 3.11 การเปลี่ยนรูปแบบแกนให้เป็นแกนความถี่ 29
ภาพที่ 3.12 การสร้าง Mel filter bank เพื่อรวมกลุ่มความถี่สูง 30
ภาพที่ 3.13 Design Model 30
ภาพที่ 3.14 Inertial measurement unit (IMU) 31
ภาพที่ 3.15 Object detection 32

ภาพที่ 4.1 และภาพที่ 4.2 ผลจากการ Convert Model (1) 34
ภาพที่ 4.3 ผลจากการ Convert Model (2) 35
ภาพที่ 4.4 ผลจากการ Convert Model (3) 35
ภาพที่ 4.5 ผลจากการ Convert Model (4) 35
ภาพที่ 4.6 ผลจากการ Convert Model (5) 36

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ภาพที่ 4.7 ผลจากการ Convert Model (6) 36
ภาพที่ 4.8 ผลจากการ Convert Model (7) 37
ภาพที่ 4.9 ผลจากการ Convert Model (8) 37
ภาพที่ 4.10 ผลจากการ Convert Model (9) 38
ภาพที่ 4.11ผลจากการ Convert Model (10) 38
ภาพที่ 4.12 ผลจากการ Convert Model (11) 38
ภาพที่ 4.13 ผลจากการ Convert Model (12) 39
ภาพที่ 4.14 ผลจากการ Convert Model (13) 39
ภาพที่ 4.15 ผลจากการ Convert Model (14) 40
ภาพที่ 4.16 ผลจากการ Convert Model (15) 40
ภาพที่ 4.17 Open Speech Recording 41
ภาพที่ 4.18 Preprocessing and Model (16) 41
ภาพที่ 4.19 Preprocessing and Model (17) 42
ภาพที่ 4.20 Preprocessing and Model (18) 42
ภาพที่ 4.21 Preprocessing and Model (19) 42
ภาพที่ 4.22 Preprocessing and Model (20) 43
ภาพที่ 4.23 Preprocessing and Model (21) 43
ภาพที่ 4.24 และภาพที่ 4.25 Training Model (22) 44
ภาพที่ 4.26 Evaluate Optimize (1) 44
ภาพที่ 4.27 Evaluate Optimize (2) 45
ภาพที่ 4.28 Convert Model (1) 45
ภาพที่ 4.29 Convert Model (2) 45
ภาพที่ 4.30 Set up Code (1) 46
ภาพที่ 4.31 Set up Code (2) 47
ภาพที่ 4.32 Set up Code (3) 48
ภาพที่ 4.33 Set up Code (4) 48
ภาพที่ 4.34 Set up Code (5) 48
ภาพที่ 4.35 Set up Code (6) 49
ภาพที่ 4.36 Set up Code (7) 49
ภาพที่ 4.37 Set up Code (8) 49
ภาพที่ 4.38 Set up Code (9) 50
ภาพที่ 4.39 Set up Code (10) 50
ภาพที่ 4.40 Set up Code (11) 50

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ภาพที่ 4.41 Set up Code (12) 51
ภาพที่ 4.42 Set up Code (13) 51
ภาพที่ 4.43 ผลการทดสอบ KWS (1) 52
ภาพที่ 4.44 ผลการทดสอบ KWS (2) 53
ภาพที่ 4.45 ผลการทดสอบ KWS (3) 53
ภาพที่ 4.46 ผลการทดสอบ KWS (4) 54
ภาพที่ 4.47 หน้าต่างเว็บไซต์ที่ใช้เก็บข้อมูล IMU 55
ภาพที่ 4.48 Preprocessing and Model (1) 55
ภาพที่ 4.49 Preprocessing and Model (2) 56
ภาพที่ 4.50 Preprocessing and Model (3) 56
ภาพที่ 4.51 Preprocessing and Model (4) 57
ภาพที่ 4.52 Preprocessing and Model (5) 58
ภาพที่ 4.53 Preprocessing and Model (6) 58
ภาพที่ 4.54 Preprocessing and Model (7) 59
ภาพที่ 4.55 Preprocessing and Model (8) 59
ภาพที่ 4.56 Preprocessing and Model (9) 60
ภาพที่ 4.57 Training Model 60
ภาพที่ 4.58 Evaluate Optimize (1) 61
ภาพที่ 4.59 Evaluate Optimize (2) 61
ภาพที่ 4.60 Evaluate Optimize (3) 62
ภาพที่ 4.61 Convert Model 62
ภาพที่ 4.62 Set up Code (1) 63
ภาพที่ 4.63 Set up Code (2) 64
ภาพที่ 4.64 Set up Code (3) 65
ภาพที่ 4.65 Set up Code (4) 66
ภาพที่ 4.66 Set up Code (5) 66
ภาพที่ 4.67 Set up Code (6) 66
ภาพที่ 4.68 Set up Code (7) 67
ภาพที่ 4.69 Set up Code (8) 67
ภาพที่ 4.70 Set up Code (9) 68
ภาพที่ 4.71 Set up Code (10) 68
ภาพที่ 4.72 Set up Code (11) 68
ภาพที่ 4.73 ผลลัพธ์จากการทดสอบ IMU (1) 69

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ภาพที่ 4.74 ผลลัพธ์จากการทดสอบ IMU (2) 69
ภาพที่ 4.75 ผลลัพธ์จากการทดสอบ IMU (3) 70
ภาพที่ 4.76 การเชื่อมต่อกับอุปกรณ์ 71
ภาพที่ 4.77 การเชื่อมต่อเรียบร้อย 71
ภาพที่ 4.78 หน้าต่างในอุปกรณ์โทรศัพท์มือถือ 72
ภาพที่ 4.79 หน้าต่าง dataset 72
ภาพที่ 4.80 หน้าต่างเมนูตั้งค่าข้อมูล 73
ภาพที่ 4.81 หน้าต่าง Split sample 73
ภาพที่ 4.82 ข้อมูลหลังการ Split sample 74
ภาพที่ 4.83 การอัดข้อมูลเพิ่มเติม 74
ภาพที่ 4.84 หน้าต่าง Create impulse 75
ภาพที่ 4.85 หน้าต่าง MFCC 75
ภาพที่ 4.86 หน้าต่าง Generate features 76
ภาพที่ 4.87 หน้าต่าง Classifier 76
ภาพที่ 4.88 หน้าต่าง Model testing 77
ภาพที่ 4.89 เลือกอุปกรณ์ที่ต้องการ Deployment 77
ภาพที่ 4.90 Build library 78
ภาพที่ 4.91 ภาพการ Add library 78
ภาพที่ 4.92 การเลือก Examples 79
ภาพที่ 4.93 Code ที่เติมเข้าไป 79
ภาพที่ 4.94 ผลที่ได้จากการทดลองการจับคำเฉพาะ 80
ภาพที่ 4.95 การ Download edge impulse CLI 81
ภาพที่ 4.96 การส่งข้อมูลไป edge impulse 81
ภาพที่ 4.97 Dataset ที่เก็บข้อมูลมาได้ 81
ภาพที่ 4.98 หน้าต่าง Create impulse 82
ภาพที่ 4.99 หน้าต่าง Model testing 82
ภาพที่ 4.100 เลือกอุปกรณ์ที่ต้องการ Deployment 83
ภาพที่ 4.101 ผลลัพธ์การทดลองตรวจจับท่าทาง 83
ภาพที่ 4.102 การเก็บข้อมูลกล้อง 84
ภาพที่ 4.103 Dataset ข้อมูลใบหน้า 84
ภาพที่ 4.104 การ train model 85
ภาพที่ 4.105 การ test model 85
ภาพที่ 4.106 เลือกอุปกรณ์ที่ต้องการ Deployment 86

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ภาพที่ 4.107 ผลลัพธ์การทดสอบการตรวจจับใบหน้า 86
ภาพที่ 4.108 ตัวอย่างการตั้งค่าบอร์ด 88
ภาพที่ 4.109 หน้าต่าง Preferences 89
ภาพที่ 4.110 การตั้งค่าบอร์ด 89
ภาพที่ 4.111 หน้าต่าง Board Manager 90
ภาพที่ 4.112 เสร็จสิ้นการตั้งค่าบอร์ด 90
ภาพที่ 4.113 หน้าแรก edge impulse 91
ภาพที่ 4.114 หน้าต่างสร้าง account edge impulse 91
ภาพที่ 4.115 เสร็จสิ้นการสร้าง account 92
ภาพที่ 4.116 หน้าต่าง Dashboard edge impulse 92
ภาพที่ 4.117 ตัวอย่างการสร้าง project edge impulse (1) 93
ภาพที่ 4.118 ตัวอย่างการสร้าง project edge impulse (1) 93
ภาพที่ 4.119 ตัวอย่างการเชื่อมต่ออุปกรณ์เก็บข้อมูลเสียง (1) 94
ภาพที่ 4.120 ตัวอย่างการเชื่อมต่ออุปกรณ์เก็บข้อมูลเสียง (2) 94
ภาพที่ 4.121 ตัวอย่างการเชื่อมต่ออุปกรณ์เก็บข้อมูลเสียง (3) 95
ภาพที่ 4.122 ตัวอย่างการเชื่อมต่ออุปกรณ์เก็บข้อมูลเสียง (4) 95
ภาพที่ 4.123 ตัวอย่างการเก็บข้อมูลเสียง 96
ภาพที่ 4.124 ตัวอย่างการ Split sample (1) 96
ภาพที่ 4.125 ตัวอย่างการ Split sample (2) 97
ภาพที่ 4.126 ตัวอย่างการ Split sample (3) 97
ภาพที่ 4.127 ตัวอย่างการเก็บข้อมูลเพิ่มเติม 98
ภาพที่ 4.128 ตัวอย่างการ perform split (1) 98
ภาพที่ 4.129 ตัวอย่างการ perform split (2) 99
ภาพที่ 4.130 ตัวอย่างการ Create impulse (1) 99
ภาพที่ 4.131 ตัวอย่างการ perform split (2) 100
ภาพที่ 4.132 ตัวอย่างการ perform split (3) 100
ภาพที่ 4.133 ตัวอย่างการ perform split (4) 101
ภาพที่ 4.134 ตัวอย่างหน้าต่าง MFCC (1) 101
ภาพที่ 4.135 ตัวอย่างหน้าต่าง MFCC (2) 102
ภาพที่ 4.136 ตัวอย่างหน้าต่าง Classifier (1) 102
ภาพที่ 4.137 ตัวอย่างหน้าต่าง Classifier (2) 103
ภาพที่ 4.138 ตัวอย่างหน้าต่าง Model testing (1) 103
ภาพที่ 4.139 ตัวอย่างหน้าต่าง Model testing (2) 104

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ภาพที่ 4.140 ตัวอย่างหน้าต่าง Deploymemnt (1) 104
ภาพที่ 4.141 ตัวอย่างหน้าต่าง Deployment (2) 105
ภาพที่ 4.142 ตัวอย่างหน้าต่าง Deployment (3) 105
ภาพที่ 4.143 ตัวอย่างการ Add Library (1) 106
ภาพที่ 4.144 ตัวอย่างการ Add Library (2) 106
ภาพที่ 4.145 ตัวอย่างการ Upload Code 109
ภาพที่ 4.146 ตัวอย่างการเปิด Serial port 109
ภาพที่ 4.147 ตัวอย่างผลลัพธ์การชี้คำเฉพาะ (1) 110
ภาพที่ 4.148 หน้าเว็บไซต์สำหรับโหลด dataset 111
ภาพที่ 4.149 ตัวอย่างการ Upload dataset (1) 111
ภาพที่ 4.150 ตัวอย่างการ Upload dataset (2) 112
ภาพที่ 4.151 ตัวอย่างการ Upload dataset (3) 112
ภาพที่ 4.152 ตัวอย่างการ Create impulse อีกครั้ง 113
ภาพที่ 4.153 ตัวอย่างการลบ library 113
ภาพที่ 4.154 ตัวอย่างการเพิ่ม library ใหม่ (1) 114
ภาพที่ 4.155 ตัวอย่างการเพิ่ม library ใหม่ (2) 114
ภาพที่ 4.156 ตัวอย่างผลลัพธ์การชี้คำเฉพาะ (2) 118
ภาพที่ 4.157 ภาพประกอบการทดลอง 118
ภาพที่ 4.158 ตัวอย่างการสร้างโปรเจคใหม่ 119
ภาพที่ 4.159 ตัวอย่างการตั้งค่า Dashboard 120
ภาพที่ 4.160 ตัวอย่างการเชื่อมต่ออุปกรณ์เก็บข้อมูลภาพ (1) 120
ภาพที่ 4.161 ตัวอย่างการเชื่อมต่ออุปกรณ์เก็บข้อมูลภาพ (2) 121
ภาพที่ 4.162 ตัวอย่างการเชื่อมต่ออุปกรณ์เก็บข้อมูลภาพ (3) 121
ภาพที่ 4.163 ตัวอย่างการเก็บข้อมูลภาพ (1) 122
ภาพที่ 4.164 ตัวอย่างการเก็บข้อมูลภาพ (2) 122
ภาพที่ 4.165 ตัวอย่างการตั้ง label (1) 123
ภาพที่ 4.166 ตัวอย่างการตั้ง label (2) 123
ภาพที่ 4.167 ตัวอย่างการตั้ง label (3) 124
ภาพที่ 4.168 ตัวอย่างการ Create impulse (1) 124
ภาพที่ 4.169 ตัวอย่างการ Create impulse (2) 125
ภาพที่ 4.170 ตัวอย่างการ Create impulse (3) 125
ภาพที่ 4.171 ตัวอย่างการ Create impulse (4) 125
ภาพที่ 4.172 ตัวอย่างหน้าต่าง image (1) 126

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ภาพที่ 4.173 ตัวอย่างหน้าต่าง image (2) 126
ภาพที่ 4.174 ตัวอย่างหน้าต่าง Object detection (1) 127
ภาพที่ 4.175 ตัวอย่างหน้าต่าง Object detection (2) 127
ภาพที่ 4.176 ตัวอย่างหน้าต่าง model testing 128
ภาพที่ 4.177 ตัวอย่างหน้าต่าง Deployment 128
ภาพที่ 4.178 ตัวอย่างผลลัพธ์การตรวจจับใบหน้า (1) 129
ภาพที่ 4.179 ตัวอย่างผลลัพธ์การตรวจจับใบหน้า (2) 129
ภาพที่ 4.180 ตัวอย่างการเก็บข้อมูลภาพเพิ่มเติม 130
ภาพที่ 4.181 ตัวอย่างการตั้งค่า label เพิ่มเติม (1) 130
ภาพที่ 4.182 ตัวอย่างการตั้งค่า label เพิ่มเติม (2) 131
ภาพที่ 4.183 ตัวอย่างการตั้งค่า Node.js 132
ภาพที่ 4.184 ตัวอย่างการติดตตั้ง edge impulse CLI 132
ภาพที่ 4.185 ตัวอย่างการ Add Library 133
ภาพที่ 4.186 ตัวอย่างการสร้างโปรเจคใหม่ 135
ภาพที่ 4.187 ตัวอย่างการเชื่อมต่อ edge impulse CLI (1) 135
ภาพที่ 4.188 ตัวอย่างการเชื่อมต่อ edge impulse CLI (2) 135
ภาพที่ 4.189 ตัวอย่างการเชื่อมต่อ edge impulse CLI (3) 136
ภาพที่ 4.190 ตัวอย่างการเชื่อมต่อ edge impulse CLI (4) 136
ภาพที่ 4.191 ตัวอย่างการเชื่อมต่อ edge impulse CLI (5) 136
ภาพที่ 4.192 ตัวอย่างการตั้งค่า label ข้อมูล IMU (1) 136
ภาพที่ 4.193 ตัวอย่างการตั้งค่า label ข้อมูล IMU (2) 137
ภาพที่ 4.194 ตัวอย่างการตั้งค่า label ข้อมูล IMU (3) 137
ภาพที่ 4.195 ตัวอย่างการ Create impulse (1) 138
ภาพที่ 4.196 ตัวอย่างการการ Create impulse (2) 138
ภาพที่ 4.197 ตัวอย่างการการ Create impulse (3) 138
ภาพที่ 4.198 ตัวอย่างการตั้งค่า Spectral features (1) 139
ภาพที่ 4.199 ตัวอย่างการตั้งค่า Spectral features (2) 139
ภาพที่ 4.200 ตัวอย่างการ Classifier (1) 140
ภาพที่ 4.201 ตัวอย่างการ Classifier (2) 140
ภาพที่ 4.202 ตัวอย่างการ test model 141
ภาพที่ 4.203 ตัวอย่างการ Deploy (1) 141
ภาพที่ 4.204 ตัวอย่างการ Deploy (2) 142
ภาพที่ 4.205 ตัวอย่างการ Add Library 142

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ภาพที่ 4.206 ตัวอย่างการเปลี่ยน Library 149
ภาพที่ 4.207 ตัวอย่างผลลัพธ์การตรวจจับท่าทาง 150
ภาพที่ 4.208 วงจรระบบการควบคุมหลอดไฟด้วยเสียง 151
ภาพที่ 4.209 ภาพรวมการทำงาน 151
ภาพที่ 4.210 วงจรที่ต่อเสร็จเรียบร้อยแล้ว 152

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

บทที่ 1
บทนำ

1.1 แนวคิดและที่มา

ในปัจจุบันมีการนำเทคโนโลยี Machine learning มาใช้อย่างแพร่หลายในอุตสาหกรรมทุก ๆ ด้าน เนื่องจากกำลังในการคำนวณของคอมพิวเตอร์นั้นมีการพัฒนาอย่างไม่หยุดยั้ง ทำให้เทคโนโลยี Machine learning ก็พัฒนาเช่นกัน มีการใช้ทั้งในทางด้านการแพทย์เพื่อช่วยในการวินิจฉัยโรคต่าง ๆ หรือด้านการตลาดที่นำ Machine learning มาใช้คาดการณ์แนวโน้มของราคา หรือยอดกำไร

ถึงแม้ Machine learning จะมีประโยชน์และถูกนำไปประยุกต์ใช้งานกับสายงานต่าง ๆ จำนวนมาก แต่ Machine learning ก็ยังมีข้อจำกัดสำคัญบางประการอยู่ หนึ่งในนั้นคือยิ่งมีปริมาณข้อมูลที่มากเท่าไหรก็ยิ่งจำเป็นที่ต้องใช้เวลาและกำลังในการประมวลผลที่มากขึ้นตาม สิ่งที่ตามมาคืองบประมาณที่ใช้ในการทำ Machine learning ทั้งค่าพลังงานและค่าบำรุงรักษา ทำให้ในการทำงานที่มีอัตราส่วนข้อมูลขนาดใหญ่บางครั้งจึงจำเป็นที่ต้องใช้งบลงทุนที่มหาศาล ทำให้ Machine learning ไม่เหมาะสมกับงานบางประเภท

ในเวลาต่อมาจึงได้มีการพัฒนาเทคโนโลยี Machine learning ที่ไม่จำเป็นที่จะต้องใช้พลังงานและระบบประมวลผลที่กำลังสูง โดยใช้การบีบอัดข้อมูลหรือตัดบางส่วนของข้อมูลที่ไม่ได้มีความสำคัญอย่างเห็นได้ชัดออกไป โดยเน้นไปที่ความเร็วในการประมวลผล และกินพลังงานต่ำ โดยมีการนิยาม Machine learning ขนาดเล็กนี้ว่า TinyML โดยเมื่อนำมาใช้กับอุปกรณ์ไมโครคอนโทรลเลอร์ จึงทำให้สามารถเพิ่มขอบเขตการใช้งาน Machine learning ให้สามารถประยุกต์ใช้งานกับงานได้อีกหลายแขนง เนื่องด้วยการกินพลังงานที่น้อยของตัวอุปกรณ์ไมโครคอนโทรลเลอร์ยังทำให้ตัวอุปกรณ์ทั้งหมดสามารถใช้ติดต่อกันได้เป็นเวลานานโดยที่ไม่ต้องมีการปิดการใช้งานอุปกรณ์อีกด้วย

ทั้งนี้ TinyML ยังเป็นเทคโนโลยีที่ค่อนข้างใหม่และยังไม่ได้มีการใช้อย่างแพร่หลาย ทางคณะผู้จัดทำจึงต้องการที่จะศึกษาถึงข้อเด่น ข้อด้อยของ TinyML และเปรียบเทียบการทำงานของ TinyML และ Machine learning ทั่วไป เพื่อหาถึงวิธีการใช้งานและการนำไปประยุกต์ใช้งานที่เหมาะสมในด้านต่าง ๆ เพื่อนำไปเป็นแนวทางในการประกอบการตัดสินใจและเลือกใช้งานการเรียนรู้ของเครื่องที่เหมาะสมต่องานหรือโครงการในภายภาคหน้าต่อไป

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

1.2 วัตถุประสงค์การวิจัย

1. เพื่อศึกษาการเรียนรู้ของเครื่องบนไมโครคอนโทรลเลอร์

2. เพื่อเปรียบเทียบการเรียนรู้ของเครื่องบนไมโครคอนโทรลเลอร์ กับอุปกรณ์ควบคุม หรือการเรียนรู้ของเครื่องบนอุปกรณ์อื่น ๆ

1.3 ความสำคัญของการวิจัย

1. ได้ทราบถึงหลักการทำงานการเรียนรู้ของเครื่องบนไมโครคอนโทรลเลอร์

2. เข้าใจความแตกต่างระหว่าง Machine Learning และ TinyML

3. เป็นประโยชน์ต่อผู้ที่กำลังศึกษาด้าน Machine Learning และผู้ที่ต้องการนำ Machine Learning ไปประยุกต์ใช้งาน

4. ผลจากการวิจัยจะเป็นประโยชน์ในการประกอบการพิจารณาการประเมินคุณภาพการศึกษา

1.4 สมมติฐานการวิจัย

1. สามารถเข้าใจหลักการและการทำงานของการเรียนรู้ของเครื่องบนไมโครคอนโทรลเลอร์

2. ทราบถึงข้อดีและข้อเสียในการประยุกต์ใช้งานการเรียนรู้ของเครื่องบนอุปกรณ์แต่ละประเภท

1.5 ขอบเขตของการวิจัย

1. ศึกษาระบบการทำงานของการเรียนรู้ของเครื่องบนไมโครคอนโทรลเลอร์

2. นำมาประยุกต์ใช้งานกับอุปกรณ์อิเล็กทรอนิกส์

3. เปรียบเทียบคุณภาพผลลัพธ์ของการเรียนรู้ของเครื่องบนไมโครคอนโทรลเลอร์กับการเรียนรู้ของเครื่องรูปแบบอื่น ๆ หรือการทำงานของอุปกรณ์ไมโครคอนโทรลเลอร์ทั่วไป

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

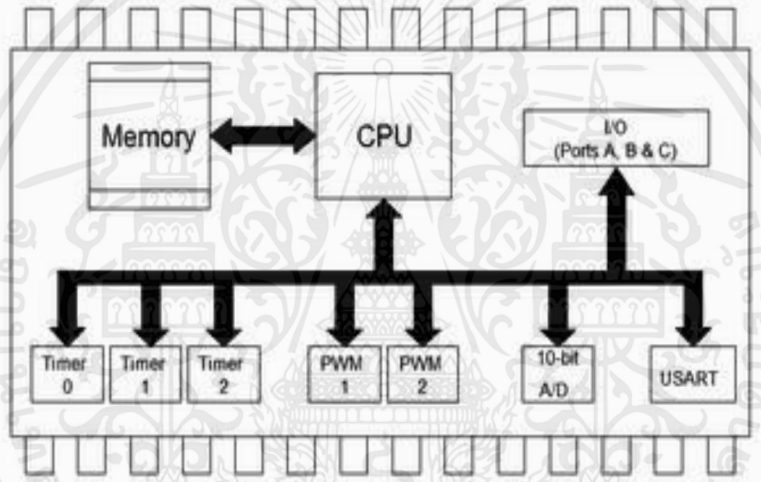
บทที่ 2
ทฤษฎีที่เกี่ยวข้อง

2.1 Microcontroller

ไมโครคอนโทรลเลอร์ (Microcontroller ตัวย่อ uC, µC หรือ MCU) เกิดจากการรวมคำว่า ไมโคร (Micro) หรือ ‘ขนาดเล็ก’ และคำว่า คอนโทรลเลอร์ (Controller) หรือ ‘อุปกรณ์ควบคุม’ ดังนั้น ไมโครคอนโทรลเลอร์ จึงหมายถึง ‘อุปกรณ์ควบคุมที่มีขนาดเล็ก’ โดยภายในจะบรรจุการทำงานที่คล้ายกับระบบปฏิบัติการคอมพิวเตอร์ ประกอบด้วย CPU หน่วยความจำ และ Port ในตัวถังเดียวกัน

โครงสร้างทั่วไปของ Microcontroller

โครงสร้างทั่วไปของไมโครคอนโทรลเลอร์ สามารถแบ่งออกมาได้เป็น 5 ส่วนใหญ่ ๆ



ภาพที่ 2.1 โครงสร้างทั่วไปของไมโครคอนโทรลเลอร์

1. หน่วยประมวลผลกลางหรือ CPU (Central Processing Unit)

หน่วยประมวลผลกลางคือวงจรอิเล็กทรอนิกส์ที่ประมวลผลตามชุดคำสั่งจากซอฟต์แวร์ เรียกได้ว่าเป็นมันสมองของคอมพิวเตอร์เพื่อทำหน้าที่ตัดสินใจหรือคำนวณตัวเลขจากคำสั่ง เช่น การบวกลบทางคณิตศาสตร์ โดยกระบวนการทำงานคือ

- Fetch คือ การอ่านชุดคำสั่ง
- Decode คือ การตีความชุดคำสั่ง
- Execute คือ การประมวลผลชุดคำสั่ง
- Memory คือ การอ่านข้อมูลจากหน่วยความจำ
- Write back คือ การเขียนข้อมูล/ส่งผลจากการประมวลผลกลับ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

2. หน่วยความจำ (Memory) แบ่งออกเป็น 2 ส่วน

ส่วนที่หนึ่งคือ หน่วยความจำสำหรับเก็บโปรแกรมหลัก (Program Memory)เสมือนกับฮาร์ดดิสก์ของคอมพิวเตอร์ตั้งโต๊ะ ข้อมูลที่ถูกเก็บไว้ในนี้จะไม่สูญหายไปแม้จะไม่มีไฟเลี้ยง

ส่วนที่สองคือ หน่วยความจำข้อมูล (Data Memory) เป็นที่พักของข้อมูลแบบชั่วคราวในขณะที่ CPU ทำงาน และใช้เป็นเหมือนกระดาษทด ถ้าหากไม่มีไฟเลี้ยงข้อมูลนั้นก็จะหายไปคล้ายกับหน่วยความจำแรม (RAM) ในคอมพิวเตอร์ทั่วไป แต่ไมโครคอนโทรลเลอร์สมัยใหม่จะมีหน่วยความจำข้อมูลทั้งแบบแรม ซึ่งข้อมูลจะหายไปเมื่อไม่มีไฟเลี้ยง และหน่วยความจำแบบอีอีพรอม (EEPROM : Electrically Erasable Programmable Read-Only Memory) ซึ่งสามารถเก็บข้อมูลไว้ได้ถึงแม้จะไม่มีไฟเลี้ยงก็ตาม

3. พอร์ต (Port)

พอร์ต หรือส่วนสำหรับติดต่อกับอุปกรณ์ภายนอก ถือเป็นส่วนที่สำคัญมาก มี 2 ประเภทคือ พอร์ตอินพุต (Input Port) ที่ใช้สำหรับรับสัญญาณจากภายนอก และพอร์ตเอาต์พุต (Output Port) หรือพอร์ตส่งสัญญาณ โดยทั้งสองพอร์ตจะทำงานร่วมกันโดยพอร์ตอินพุตจะรับสัญญาณเพื่อนำไปประมวณผลและส่งไปพอร์ตเอาต์พุตเพื่อแสดงผล เช่น การกดสวิตช์เพื่อควบคุมการเปิดปิดของหลอดไฟ

4. บัส (BUS)

บัส หรือช่องทางการเดินสัญญาณ คือเส้นทางสำหรับแลกเปลี่ยนสัญญาณข้อมูลระหว่าง CPU หน่วยความจำ และพอร์ต มีสักษณะเป็นสายสัญญาณจำนวนมากอยู่ภายในตัวไมโครคอนโทรลเลอร์ มีทั้งหมด 3 ประเภทคือ บัสข้อมูล (Data Bus) บัสแอดเดรส (Address Bus) และบัสควบคุม (Control Bus) บัสข้อมูล (Data Bus) เป็นสายสัญญาณที่บรรจุข้อมูลทั้งหมดที่ใช้ในการประมวลผล โดยขนาดของบัสจะขึ้นอยู่กับความสามารถในการประมวลผลของ CPU สำหรับงานทั่วไปขนาดของบัสข้อมูลจะอยู่ที่ 8 บิต และในปัจจุบันมีขนาดสูงสุดได้ถึง 64 บิต บัสแอดเดรส (Address Bus) เป็นสายสัญญาณที่บรรจุข้อมูลตำแหน่งของหน่วยความจำ ซึ่งการที่จะติดต่อกับหน่วยความจำนั้น CPU จะต้องกำหนดตำแหน่งที่ต้องการอ่านหรือเขียนก่อน สายสัญญาณของแอดเดรสนั้น ยิ่งมีมากเท่าไหร่ก็จะยิ่งสามารถติดต่อหน่วยความจำได้ขนาดใหญ่มากขึ้นไปด้วย โดยวิธีการคำนวณจำนวนแอดเดรสของหน่วยความจำคือ 2 ยกกำลัง n (n คือจำนวนเส้นทาง) ตัวอย่าง ไมโครคอนโทรลเลอร์ตัวหนึ่งมีสายแอดเดรส 11 เส้น ดังนั้นไมโครคอนโทรลเลอร์ตัวนี้จะสามารถติดต่อกับหน่วยความจำได้ 2,048 ตำแหน่ง (2 ยกกำลัง 11) และถ้าหากต้องการทราบความจุของหน่วยความจำจะต้องทราบถึงขนาดของบัสข้อมูลก่อนว่ามีขนาดเท่าใด หากบัสข้อมูลมีขนาด 8 บิต หน่วยความจำที่มีสายแอดเดรส 11 เส้น จะมีความจุ 8x2084 = 16,384 บิต และ 1 กิโลไบต์เท่ากับ 1,024 ไบต์ ดังนั้นไมโครคอนโทรลเลอร์ตัวนี้จึงมีความจุของหน่วยความจำเท่ากับ 16,384 บิต หรือ 2084 ไบต์ หรือ 2 กิโลไบต์

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

บัสควบคุม (Control Bus) เป็นกลุ่มของสายสัญญาณที่มีหน้าที่ควบคุมการติดต่อของ CPU กับหน่วยความจำและพอร์ต โดยจะมีสายสัญญาณควบคุมหลักได้แก่ สายสัญญาณเลือกอ่าน-เขียนหน่วยความจำ และสายสัญญาณเลือกอ่าน-เขียนข้อมูลกับพอร์ต

5. วงจรกำเนิดสัญญาณนาฬิกา (Clock)

วงจรกำเนิดสัญญาณนาฬิกา เป็นองค์ประกอบที่สำคัญมากอีกส่วนหนึ่ง เนื่องจากการทำงานที่เกิดขึ้นภายในตัวไมโครคอนโทรลเลอร์นั้นจะขึ้นอยู่กับการกำหนดจังหวะ ดังนั้นหากสัญญาณนาฬิกามีความถี่สูง จังหวะการทำงานก็ถี่ขึ้น ส่งผลให้ไมโครคอนโทรลเลอร์มีความเร็วในการประมวลผลสูงขึ้นด้วย

2.2 Microprocessor

ไมโครโพรเซสเซอร์ (Microprocessor) คือ คอมพิวเตอร์โพรเซสเซอร์บนไมโครชิพ มันเป็นเอนจิ้น (Engine) ซึ่งจะเริ่มทำงานก็ต่อเมื่อเปิดสวิตช์เครื่อง ไมโครโพรเซสเซอร์ถูกออกแบบมาเพื่อการประมวลผลคณิตศาสตร์และตรรกศาสตร์ อาศัยตัวเก็บข้อมูลขนาดเล็กที่เรียกว่ารีจีสเตอร์(Register) ในการเก็บข้อมูลรอนำส่งและเก็บผลลัพธ์ชั่วคราว

ไมโครโพรเซสเซอร์มีลักษณะการประมวลผลได้แก่ การบวก การลบ การเปรียบเทียบตัวเลข 2 จำนวน และการดึงข้อมูลจากที่หนึ่งไปยังอีกที่หนึ่ง เป็นผลจากชุดคำสั่งซึ่งเป็นส่วนหนึ่งของโครงสร้างไมโครโพรเซสเซอร์

2.3 ข้อแตกต่างระหว่าง Microcontroller กับ Microprocessor

ไมโครโพรเซสเซอร์ที่ใช้อยู่ในปัจจุบันจะไม่มีหน่วยความจำ RAM ROM และ Port อยู่ในตัวเอง จึงต้องต่อหน่วยความจำโปรแกรมภายนอก และต้องใช้ ICs ในการขยายพอร์ตเพิ่ม ข้อดีคือสามารถเพิ่มหน่วยความจำได้ตลอด ส่วนไมโครคอนโทรลเลอร์นั้นจะมีวงจรพื้นฐานอยู่แล้วภายในชิป (เช่น หน่วยความจำ RAM ROM และ I/O Port) เพราะเหตุนี้ไมโครคอนโทรลเลอร์จึงมีขนาดเล็กกว่าและราคาต่ำกว่าไมโครโพรเซสเซอร์

ตาราง 2.1 ตารางเปรียบเทียบ Microcontroller และ Microprocessor

Microcontroller	Microprocessor
ประกอบด้วย CPU, หน่วยความจำ, I/O ทั้งหมดรวมอยู่ในชิปตัวเดียว	ประกอบด้วยหน่วยประมวลผลกลางเท่านั้น จำเป็นต้องต่ออุปกรณ์เพิ่มเพื่อให้สามารถใช้งานได้สมบูรณ์
ใช้บัสควบคุมภายในเป็นแบบระบบฝังตัว อัพเกรดส่วนประกอบต่าง ๆ ได้ยาก แต่ทำให้ระบบมีมาตรฐาน และถูกรบกวนได้ยากกว่า	ใช้บัสภายนอกเพื่อเชื่อมต่อกับ RAM, ROM และอุปกรณ์ต่อพ่วงอื่น ๆ ทำให้สามารถอัพเกรดส่วนประกอบต่างๆ ได้ง่าย

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

การทำงานเป็นแบบตรงไปตรงมา โดยมีคำสั่งในการประมวลผลน้อยกว่า ทำให้ใช้ระยะเวลาและพลังงานน้อยกว่า	ใช้คำสั่งจำนวนมากในการประมวลผล ทำให้ใช้ระยะเวลาและพลังงานมากกว่า
วิศวกรจะเขียนและคอมไพล์โค้ดที่มีไว้สำหรับแอพลิเคชันเฉพาะ และอัพโหลดลงในไมโครคอนโทรลเลอร์ ซึ่งบรรจุฟีเจอร์และส่วนประกอบการประมวลผลที่จำเป็นทั้งหมดไว้ภายในเพื่อรันโค้ด จึงมักต้องการหน่วยความจำน้อยกว่า พลังการประมวลผลน้อยกว่า และความซับซ้อนโดยรวมน้อยกว่าไมโครโปรเซสเซอร์ ดังนั้นต้นทุนจึงต่ำกว่า	มักผลิตขึ้นเพื่อใช้กับอุปกรณ์ราคาแพงกว่า ซึ่งจะใช้ประโยชน์จากอุปกรณ์ต่อพ่วงภายนอกเพื่อเพิ่มประสิทธิภาพ มีความซับซ้อนกว่าอย่างเห็นได้ชัด เนื่องจากมีไว้สำหรับทำงานด้านการคำนวณที่หลากหลาย ดังนั้นจึงต้องการแหล่งหน่วยความจำภายนอกที่ทนทาน เพื่อรองรับงานการคำนวณที่ซับซ้อนมากขึ้น
มีไว้เพื่อจัดการกับงานหรือแอพลิเคชันเฉพาะ	มีไว้สำหรับงานคอมพิวเตอร์ที่ซับซ้อน แข็งแกร่ง และคาดเดาไม่ได้
สามารถปรับให้เหมาะสมเพื่อเรียกใช้โค้ดสำหรับงานเฉพาะ คือใช้ความเร็วและกำลังในปริมาณที่เหมาะสม ไม่มากไป ไม่น้อยไป	ทำงานด้วยความเร็วและกำลังสูงสุดตลอดเวลา จึงมักสิ้นเปลืองพลังงานมากกว่า

2.4 Machine Learning

Machine Learning (ML) เป็นการเรียนรู้การวิเคราะห์ข้อมูลที่ทำให้การสร้างแบบจำลองการวิเคราะห์เป็นไปโดยอัตโนมัติ สามารถช่วยให้แอปพลิเคชันซอฟต์แวร์มีความแม่นยำมากขึ้น เป็นแขนงหนึ่งของปัญญาประดิษฐ์ที่มีแนวคิดว่าระบบสามารถเรียนรู้ วิเคราะห์ หรือแม้แต่คาดเดาคาดคะเนประมวลผลจากข้อมูล สามารถนำผลของข้อมูลมาเรียนรู้ พัฒนา ระบุรูปแบบและทำการตัดสินใจด้วยตัวเองโดยมีการแทรกแซงจากภายนอกน้อยที่สุด

การทำงานของ Machine Learning เป็นการทำงานที่คล้ายกับวิธีการที่สมองมนุษย์ทำงาน คือการได้รับความรู้และเข้าใจในความรู้ การเรียนรู้ของเครื่องก็ต้องอาศัยอินพุตข้อมูลเพื่อทำการวิเคราะห์ สังเคราะห์ประมวลผลเชื่อมโยงข้อมูลตามการกำหนดค่าที่ต้องการเพื่อให้การเรียนรู้เชิงลึกสามารถเริ่มต้นการเรียนรู้ได้ โดยมีจุดหมายหลักเพื่ออนุญาตให้คอมพิวเตอร์สามารถเรียนรู้ได้ด้วยตัวเองโดยปราศจากการแทรกแซงหรือความช่วยเหลือของมนุษย์ มีความเป็นไปได้ที่จะสร้างแบบจำลองได้รวดเร็วและอัตโนมัติที่สามารถวิเคราะห์ข้อมูลที่ใหญ่กว่าและซับซ้อนมากขึ้น และให้ผลลัพธ์ที่รวดเร็วและแม่นยำยิ่งขึ้นแม้ว่าจะมีขนาดใหญ่มากก็ตาม และด้วยการสร้างแบบจำลองที่แม่นยำ ผู้ใช้งานหรือองค์กรที่นำไปใช้งานจะมีโอกาสมากขึ้นในการคาดเดาหรือทำนาย เพื่อหลีกเลี่ยงความเสี่ยงที่ไม่รู้จัก

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ระบบการเรียนรู้ของอัลกอริทึมแบ่งออกเป็นสามส่วนหลัก

- หนึ่งคือกระบวนการการตัดสินใจ โดยทั่วไปอัลกอริทึมจะใช้เพื่อการทำนายหรือการจัดหมวดหมู่ อัลกอริทึมจะสร้างค่าประมาณเกี่ยวกับรูปแบบในข้อมูลโดยอ้างอิงจากอินพุตของข้อมูลบางอย่างซึ่งสามารถกำหนดค่าหรือไม่ก็ได้
- สองฟังก์ชันข้อผิดพลาด ประเมิณการคาดคะเนของแบบจำลอง หากมีตัวอย่างที่ทราบฟังก์ชันข้อผิดพลาดสามารถทำการเปรียบเทียบเพื่อประเมิณความแม่นยำของแบบจำลองได้
- สามกระบวนการเพิ่มประสิทธิภาพโมเดล หากโมเดลเหมาะสมกับจุดข้อมูลในชุดการฝึกได้ดีขึ้น การปรับค่าเพื่อลดความคลาดเคลื่อนระหว่างตัวอย่างที่ทราบและค่าประมาณของโมเดลอัลกอริทึมจะทำกระบวนการประเมิณและเพิ่มประสิทธิภาพนี้ซ้ำโดยเพิ่มค่าน้ำหนักโดยอัตโนมัติจนกว่าความแม่นยำจะถึงเกณฑ์

โมเดลของ Machine Learning อัลกอริทึมที่ใช้แบ่งออกได้เป็นสามประเภทโดยประมาณได้แก่

- การเรียนรู้แบบมีผู้สอน (Supervised Learning) การใช้ชุดข้อมูลที่มีป้ายกำกับ ชุดข้อมูลเหล่านี้ออกแบบมาเพื่อฝึกอัลกอริทึมในการจำแนกข้อมูลหรือทำนายผลลัพธ์อย่างแม่นยำเมื่อใช้อินพุตและเอาต์พุตที่ติดป้าย แบบจำลองสามารถวัดความแม่นยำและเรียนรู้เมื่อเวลาผ่านไป
- การเรียนรู้แบบไม่มีผู้สอน (Unsupervised Learning) ใช้อัลกอริทึมของ Machine Learning เพื่อวิเคราะห์และจัดกลุ่มชุดข้อมูลที่ไม่มีป้ายกำกับ อัลกอริทึมเหล่านี้ค้นพบรูปแบบที่ซ่อนอยู่ในข้อมูลโดยไม่จำเป็นต้องมีการแทรกแซงจากมนุษย์
- การเรียนรู้แบบเสริมกำลัง (Reinforcement Learning) ข้อมูลการฝึกมีป้ายกำกับแต่สามารถตัดสินใจได้ว่าจะปฏิบัติงานอย่างไร

Neural Networks หรือโครงข่ายประสาทเป็นส่วนหนึ่งของ Machine Learning เช่นเดียวกับโครงข่ายประสาทเทียม (Artificial neural networks (ANNs)) และโครงข่ายประสาทจำลอง (Simulated neural networks (SNNs)) และเป็นหัวใจหลักของการเรียนรู้เชิงลึก (Deep Learning) โดยได้รับแรงบันดาลใจมาจากสมองของมนุษย์ ทำการเลียนแบบขั้นตอนและกระบวนการทำงานของเซลล์ประสาทเชิงชีววิทยา

โครงข่ายประสาทเทียมประกอบด้วยชั้นโหนดที่มีชั้นอินพุต ชั้นซ่อนหนึ่งชั้นหรือมากกว่า และชั้นเอาต์พุต แต่ละโหนดหรือเซลล์ประสาทเทียมเชื่อมต่อกับอีกโหนดหนึ่งและมีน้ำหนักและเกณฑ์ที่สัมพันธ์กัน หากเอาต์พุตของแต่ละโหนดสูงกว่าค่าเกณฑ์ที่ระบุ โหนดนั้นจะถูกเปิดใช้งาน โดยส่งข้อมูลไปยังเลเยอร์ถัดไปของเครือข่าย ถ้าไม่อย่างนั้นการส่งข้อมูลไปยังเลเยอร์ถัดไปอาจมีความผิดพลาด

โครงข่ายประสาทเทียมสามารถใช้ข้อมูลการฝึกอบรมเพื่อเรียนรู้และปรับปรุงความแม่นยำเมื่อเวลาผ่านไป และเมื่ออัลกอริทึมการเรียนรู้เหล่านี้ได้รับการปรับแต่งเพื่อความแม่นยำแล้ว ก็จะเป็นเครื่องมือที่มีประสิทธิภาพในด้านวิทยาการคอมพิวเตอร์และปัญญาประดิษฐ์ทำให้เราสามารถจัดประเภทและจัดกลุ่มข้อมูล

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ด้วยความเร็วสูง งานด้านการรู้จำเสียงหรือการรู้จำภาพอาจใช้เวลาเป็นนาทีเทียบกับชั่วโมงเมื่อเทียบกับการระบุด้วยตนเองโดยมนุษย์ หนึ่งในเครือข่ายประสาทที่เป็นที่รู้จักมากที่สุดคืออัลกอริทึมการค้นหาของ Google

ประเภทของโครงข่ายประสาท

- Perceptron เป็นโครงข่ายประสาทที่เก่ามากที่สุด มีเซลล์ประสาทเพียงหนึ่งเซลล์และเป็นรูปแบบง่าย
- Feedforward neural networks หรือ multi-layer perceptrons (MLPs) ประกอบด้วยชั้นอินพุต ชั้นหรือชั้นที่ซ่อนอยู่ และชั้นเอาต์พุตมีเซลล์ประสาทซิกมอยด์ รากฐานสำหรับการมองเห็นของคอมพิวเตอร์การประมวลผลภาษาธรรมชาติ
- Convolutional Neural Network (CNN) หรือโครงข่ายประสาทแบบคอนโวลูชัน เป็นหนึ่งโครงข่ายประสาทเทียม โดยจะจำลองการมองเห็นของมนุษย์ที่มองพื้นที่เป็นที่ย่อย และนำกลุ่มพื้นที่ย่อยมาผสานกัน เพื่อแสดงผลลัพธ์ที่เห็น

และโครงข่ายประเภทอื่น ๆ

Deep Learning (DL) การเรียนรู้เชิงลึกเป็นรู้แบบการเรียนรู้ที่ต้องใช้โครงข่ายใยประสาทเสมือน (Artificial Neural Network (ANN)) ซึ่งจำลองการทำงานของระบบประสาทในสมองมนุษย์ โครงข่ายเหล่านี้มีเซลล์ประสาทที่เชื่อมต่อกันเป็นระบบประสาทและสื่อสารกันโดยใช้วิธีประมวลผลแบบขนาน (Parallel Processing) เพื่อทำให้มันสามารถเข้าใจและเรียนรู้จากข้อมูลจำนวนมากที่ได้รับอย่างต่อเนื่อง การกำหนดค่าที่ต้องการสิ่งต่าง ๆ แบ่งแยกออกเป็นหมวดหมู่เมื่อใดก็ตามที่ได้รับข้อมูลใหม่ การเปรียบเทียบระหว่างข้อมูลใหญ่และสิ่งที่ได้เรียนรู้ไปแล้วก่อนหน้าก็จะเกิดขึ้นก่อนจะได้ทำการวิเคราะห์หรือสังเคราะห์ข้อมูล

ความแตกต่างระหว่าง Machine Learning และ Deep Learning

- DL เป็นส่วนย่อยของ ML ความแตกต่างหลักๆ ขึ้นอยู่กับวิธีที่แต่ละอัลกอริทึมเรียนรู้และจำนวนข้อมูลที่ต้องการ
- DL ต้องการการแทรกแซงของมนุษย์น้อยกว่ามากเนื่องจากทำการแยกคุณสมบัติอัตโนมัติจำนวนมาก ในขณะเดียวกัน ML ขึ้นอยู่กับการแทรกแซงของมนุษย์ในการเรียนรู้มากกว่า ผู้เชี่ยวชาญที่เป็นบุคคลกำหนดลำดับชั้นของคุณสมบัติเพื่อทำความเข้าใจความแตกต่างระหว่างข้อมูลที่ป้อนเข้า

ซึ่งหมายความว่าอัลกอริทึม Machine Learning ใช้ประโยชน์จากข้อมูลที่มีโครงสร้างและมีป้ายกำกับเพื่อทำการทำนาย คุณสมบัติเฉพาะถูกกำหนดจากข้อมูลที่ป้อนเข้า และว่าหากใช้ข้อมูลที่ไม่มีโครงสร้าง โดยทั่วไปจะต้องผ่านการประมวลผลล่วงหน้าเพื่อจัดระเบียบข้อมูลให้อยู่ในรูปแบบที่มีโครงสร้าง

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ในขณะเดียวกัน Deep Learning สามารถใช้ประโยชน์จากชุดข้อมูลที่มีป้ายกำกับ (ผ่านการเรียนรู้ที่มีผู้สอน) เพื่อแจ้งอัลกอริทึม แต่ไม่จำเป็น นอกจากนี้ Deep Learning ยังสามารถรับข้อมูลที่ไม่มีโครงสร้างในรูปแบบดิบและกำหนดชุดคุณลักษณะโดยอัตโนมัติซึ่งแยกรายการต่างๆ ออกจากกัน

ข้อดีและข้อเสียของ Machine Learning

ตาราง 2.2 ตารางข้อดีและข้อเสียของ Machine Learning

ข้อดี	ข้อเสีย
- สามารถวิเคราะห์จดจำรูปแบบและการไหลของข้อมูลได้อย่างง่ายดาย - เป็นอัตโนมัติเมื่อดำเนินการฝึกระบบโมเดล จะสามารถทำงานด้วยตัวเองได้เมื่อกระบวนการเสร็จสิ้น - มีความหลากหลายสามารถนำไปใช้งานได้ในทุกสาขา - สามารถจัดการข้อมูลได้อย่างมหาศาล - สามารถดำเนินการทำงานหลายประเภทได้พร้อมกัน - มีขอบเขตการเรียนรู้ที่กว้างขวางไม่มีสิ้นสุดตราบใดที่ยังมีการเรียนรู้ - ลดเวลาและความซับซ้อนในการทำงาน - สามารถคาดเดาหรือทำนายแนวโน้มจากข้อมูลได้	- การได้ข้อมูลมาทำการฝึกนั้นยาก อาจมีข้อผิดพลาดในการรวบรวมข้อมูล ข้อมูลขนาดใหญ่ใช้เวลาในการประมวลผลนานจนอาจเกิดความผิดปกติ ทำให้การฝึกเกิดข้อผิดพลาดได้ง่าย - ใช้เวลานานในการสร้างฝึกแต่ละครั้ง - ต้องลงทุนด้วยเม็ดเงินและทรัพยากรข้อมูลมหาศาล ในการทำให้สมบูรณ์

2.5 TensorFlow

TensorFlow เป็นไลบรารีโอเพ่นซอร์สสำหรับการคำนวณเชิงตัวเลขที่ทำให้ Machine Learning และการพัฒนาโครงข่ายประสาทเทียมเร็วขึ้นและง่ายขึ้นของ Google ที่ทำให้กระบวนการรับข้อมูล การฝึกอบรมโมเดล การแสดงการคาดการณ์ และการปรับแต่งผลลัพธ์ในอนาคตง่ายขึ้น

TensorFlow รวม Machine Learning โมเดลการเรียนรู้เชิงลึกและอัลกอริทึมเข้าด้วยกัน สามารถใช้ Python หรือ JavaScript โดยมี API (Application Programming Interfaces) หลายตัวซึ่งสามารถแบ่งได้เป็นสองประเภท

- Low level API: การควบคุมการเขียนโปรแกรมที่สมบูรณ์ มีคำแนะนำสำหรับผู้ที่ต้องการเรียนรู้ Machine Learning สามารถให้การควบคุมโมเดลได้ในระดับที่ดี อย่างเช่น TensorFlow Core

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

- High level API: สร้างขึ้นบน TensorFlow Core สามารถเรียนรู้และใช้งานได้ง่ายกว่า TensorFlow Core ทำให้ทำงานซ้ำง่ายขึ้นและสอดคล้องกันมากขึ้นระหว่างผู้ใช้ที่แตกต่างกัน เช่น tf.contrib.learn

การฝึกโมเดล Machine Learning โดยใช้ TensorFlow

1. นำเข้าและแยกวิเคราะห์ชุดข้อมูล
2. เลือกชนิดแบบจำลองที่ใช้ให้เข้ากับข้อมูล เป็นตัวกำหนดรูปแบบการวิเคราะห์ข้อมูล
3. ฝึกโมเดล มีเป้าหมายคือการเรียนรู้โครงสร้างของชุดข้อมูลให้เพียงพอเพื่อคาดการณ์ข้อมูล
4. ประเมินประสิทธิภาพของแบบจำลอง เพื่อหาความเสถียรและความแม่นยำของโมเดลที่ฝึก
5. ใช้แบบจำลองที่ได้รับจากการฝึกมาทำนาย

TensorFlow Lite คือ โมเดล TensorFlow ที่ผ่านการฝึกสามารถนำไปใช้กับการประมวลผลที่ชอบหรืออุปกรณ์พกพา เช่น ระบบ iOS หรือ Android ชุดเครื่องมือ TensorFlow Lite ปรับแต่งโมเดล TensorFlow เพื่อให้ทำงานได้ดีบนอุปกรณ์ดังกล่าว โดยอนุญาตให้ทำการแลกเปลี่ยนระหว่างขนาดโมเดลและความแม่นยำ โมเดลที่เล็กกว่าจะมีความแม่นยำน้อยกว่า แต่โดยทั่วไปแล้วการสูญเสียความแม่นยำจะมีเพียงเล็กน้อย และมากกว่าการชดเชยด้วยความเร็วและประสิทธิภาพพลังงานของโมเดล

TensorFlow Lite Micro คือ ชุดเครื่องมือ TensorFlow Lite ที่ได้รับการออกแบบมาเพื่อเรียกใช้โมเดลบนเครื่องไมโครคอนโทรลเลอร์ สามารถรันโมเดลพื้นฐานได้หลายรุ่นไม่ต้องการการสนับสนุนระบบปฏิบัติการมาตรฐาน

2.6 Tiny Machine Learning

Tiny Machine Learning คือ วลีที่นิยามถึงการนำการเรียนรู้ของเครื่องมาประยุกต์ใช้บนไมโครคอนโทรลเลอร์ หรือระบบสมองกลฝังตัว (embedded system) โดยมีจุดประสงค์เพื่อให้การเรียนรู้ของเครื่องสามารถใช้งานบนอุปกรณ์ที่ใช้พลังงานต่ำ ภายในหลักมิลิวัตต์ มีหน่วยความจำและกำลังในการประมวลผลที่ต่ำ มีราคาถูกและสามารถใช้งานได้อย่างต่อเนื่องเป็นเวลานาน โดยสามารถรับและประมวลผลข้อมูลได้ตามเวลาจริง รวมไปถึงพร้อมที่จะทำงานร่วมกับอุปกรณ์ปลายทางอื่น ๆ ในระบบหรือในเครือข่ายและบนคลาวน์

2.6.1 ข้อดีและข้อเสียของ Tiny Machine Learning

แต่เดิมการเรียนรู้ของเครื่องโดยทั่วไปจะถูกพัฒนาเพื่อแก้ไขหรือคาดเดาผลลัพธ์ในโจทย์หรือปัญหาที่มีความเฉพาะทาง Tiny Machine Learning ก็ไม่ต่างกัน แต่ในขณะที่การเรียนรู้ของเครื่องจะใช้ปริมาณข้อมูลและกำลังประมวลผลของอุปกรณ์ที่ใช้ในการคำนวณเป็นย่างมาก ในทางกลับกัน Tiny Machine Learning จะทำงานอยู่บนอุปกรณ์ไมโครคอนโทรลเลอร์ที่ใช้พลังงานน้อยแต่สามารถทำงานได้เป็นเวลานาน แต่จะไม่สามารถนำมาใช้กับงานที่ต้องใช้กำลังในการประมวลผลที่สูงได้ Tiny Machine Learning จึงเหมาะสมกับงานเพียงบางประเภทเท่านั้น เช่น การตรวจจับคำพูดเฉพาะ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

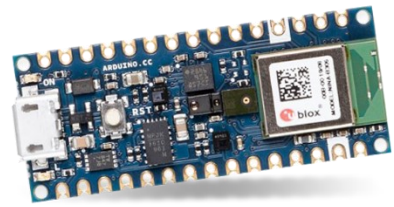
หรือการตรวจจับท่าทาง เป็นต้น ในแนวทางเดียวกัน การเรียนรู้ของเครื่องทั่วไปก็จะถูกออกแบบมาให้ใช้ในงานที่ Tiny Machine Learning ไม่สามารถทำได้หรือไม่เหมาะสมที่จะทำ Tiny Machine Learning จึงไม่ใช่การทดแทนแต่เป็นการประยุกต์ให้สามารถเพิ่มขอบเขตความสามารถของการเรียนรู้ของเครื่องให้ครอบคลุมงานที่มากขึ้น

2.6.2 แนวทางการใช้ Tiny Machine Learning ในปัจจุบัน

โดยทั่วไปแล้วการเรียนรู้ของเครื่องจะเป็นการ Training Model บน platform ต่าง ๆ เช่น TensorFlow บนคลาวน์ ทำให้สามารถประมวลผลข้อมูลขนาดใหญ่บน web platform เช่น Google Colab ได้ แต่การนำการเรียนรู้บนเครื่องไปใช้บนระบบประมวลผลขนาดเล็ก เช่น Android หรือไมโครคอนโทรลเลอร์ จำเป็นที่จะต้องมีวิธีการที่จะทำให้ขนาดของ Model หลังการ Training เล็กลง หนึ่งในนั้นคือ TensorFlow lite ซึ่งคือ framework ที่ไว้ใช้ทำให้สามารถทำการ Inference บนอุปกรณ์อุปกรณ์ edge ต่าง ๆ เช่น อุปกรณ์ Android หรือ IOS ได้ แต่ถึงอย่างนั้นก็ยังไม่สามารถนำไปใช้บนไมโครคอนโทรลเลอร์ได้เพราะโดยส่วนใหญ่แล้วไมโครคอนโทรลเลอร์มักจะมีหน่วยจำเพียง 1 MB และ RAM 256 KB จึงได้มีการพัฒนา TensorFlow lite Micro ซึ่งจะเป็นการลดขนาดของ library ของ TensorFlow ให้มีขนาดเล็กลง โดยจะทำการตัด Operators ที่ไม่จำเป็นต่องานปัจจุบันออกไปและทำให้สามารถทำงานบน embedded system ได้

ในปัจจุบันมีบริษัทพัฒนาอุปกรณ์ไมโครคอนโทรลเลอร์จำนวนมากที่ได้ออกผลิตภัณฑ์เพื่อมาใช้กับ Tiny Machine Learning

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



ภาพที่ 2.2 Arduino nano 33 BLE

2.6.2.1 Arduino nano 33 BLE

บอร์ดชุด nano พัฒนาโดย Arduino มาพร้อมกับเซนเซอร์วัดค่าต่าง ๆ ในตัว รองรับการเชื่อมต่อ Bluetooth เหมาะแก่การมาใช้เป็นตัวต้นแบบของระบบ Embedded

ข้อมูลทั่วไปของบอร์ด

ราคา : 3125 บาทโดยประมาณ

ขนาด : 45*18 mm

Microcontroller : nRF52840

CPU Flash Memory : 1 MB

SRAM : 256 KB

I/O pin : 14

IMU : LSM9DS1

Microphone : MP34DT05

Gesture, light, proximity : APDS9960

Barometric pressure : LPS22HB

Temperature, humidity : HTS221

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ภาพที่ 2.3 Seeed Studio XIAO nRF52840 Sense

2.6.2.2 Seeed Studio XIAO nRF52840 Sense

บอร์ดโมเดล XIAO พัฒนาโดย Seeed พัฒนาเพื่อใช้กับเทคโนโลยี TensorFLow lite มาพร้อมกับไมโครโฟน/IMU และรองรับการเชื่อมต่อ Bluetooth

ข้อมูลทั่วไปของบอร์ด

ราคา : 550 บาทโดยประมาณ

ขนาด : 21*17.5 mm

Microcontroller : nRF52840

CPU Flash Memory : 1 MB

SRAM : 256 KB

I/O pin : 11

IMU : LSM6DS3TR-C

Microphone : MSM261D3526H1CPM

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ภาพที่ 2.4 Syntiant’s Tiny Machine Learning (NDP101)

2.6.2.3 Syntiant’s Tiny Machine Learning (NDP101)

บอร์ดพัฒนา TinyML พร้อมใช้งานสำหรับการสร้างแอปพลิเคชันเสียงและเซ็นเซอร์ที่ใช้พลังงานต่ำโดยใช้แพลตฟอร์ม ML แบบฝังตัวของ Edge Impulse

ข้อมูลทั่วไปของบอร์ด

ราคา : 1,245 บาทโดยประมาณ

ขนาด : 24*28 mm

Neural Decision Processor : NDP101

Host processor: SAMD21 Cortex-M0+ 32bit low power ARM MCU

CPU flash memory : 256KB

SRAM host processor : 32KB

Board power supply: 5V micro-USB or 3.7V LiPo battery

Digital I/Os : 5 compatible with Arduino MKR series boards

UART interface : 1 (included in the digital I/O Pins)

I2C interface : 1 (included in the digital I/O Pins)

on-board serial flash : 2MB

48MHz system clock

RGB LED : 1 user

uSD card slot

6 axis motion sensor : BMI160

Microphone : SPH0641LM4H

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ภาพที่ 2.5 Arducam Pico4ML

2.6.2.4 Arducam Pico4ML

บอร์ด Raspberry Pi Pico สำหรับ TensorFlow Lite Micro ที่พัฒนาโดยทีมงาน Arducam ได้ใช้ Raspberry Silicon (หรือที่เรียกว่าชิป RP2040) และสร้าง Pico4ML ซึ่งเป็นบอร์ดพัฒนาไมโครคอนโทรลเลอร์ที่สร้างขึ้นมาสำหรับการรันและฝึกตัวอย่าง ML โดยเฉพาะ

ข้อมูลทั่วไปของบอร์ด

ราคา : 902 บาทโดยประมาณ

ขนาด : 51*21 mm

Microcontroller: Raspberry Pi RP2040

IMU: ICM-20948

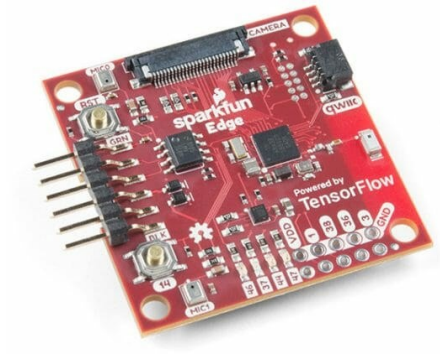
Camera Module: HiMax HM01B0, Up to QVGA (320 x 240 @60fps).

Screen: 0.96 inch LCD SPI Display (160 x 80, ST7735)

Operating Voltage: 3.3V

Input Voltage: 5V +/- 10% via VBUS, Max 5.5V via VSYS

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



ภาพที่ 2.6 SparkFun Edge

2.6.2.5 SparkFun Edge

บอร์ดพัฒนา SparkFun Edge สำหรับเริ่มต้นใช้งาน TensorFlow Lite ที่พัฒนาขึ้นโดย Ambiq Micro Apollo3 Blue Bluetooth MCU ซึ่งมีแกน Arm Cortex-M4F ที่มีประสิทธิภาพสูงสุดสามารถเรียกใช้ TensorFlow Lite โดยใช้เพียง 6uA/MHz

ข้อมูลทั่วไปของบอร์ด

ราคา : 520 บาทโดยประมาณ

ขนาด : 40.6*40.6*8.9 mm

Microcontroller : 32-bit ARM Cortex-M4F processor with Direct Memory Access 48MHz CPU clock, 96MHz with TurboSPOT™

Extremely low-power usage : 6uA/MHz

Flash memory : 1MB

SRAM host processor : 384KB

Connectivity : Bluetooth LE 5 (on-chip) + Bluetooth antenna

Camera : OV7670 camera connector

Audio : 2x MEMS microphones with operational amplifier

Sensor : STMicro LIS2DH12 3-axis accelerometer

Expansion : Qwiic connector, 4x GPIO header,

Debugging : FTDI-style serial header for programming

Misc : 4x user LEDs, 1x user button

Power Supply : CR2032 coin cell holder for battery operation, VDD (1.8 to 3.6V) / GND pins

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ภาพที่ 2.7 Arduino Portenta H7

2.6.2.6 Arduino Portenta H7

บอร์ด Arduino สำหรับการจัดการอุปกรณ์และตระกูลบอร์ดระดับอุตสาหกรรม Portenta ใหม่ที่ขับเคลื่อนโดย ไมโครคอนโทรลเลอร์ Arm Cortex-M7/M4 แบบดูอัลคอร์ STMicro

ข้อมูลทั่วไปของบอร์ด

ราคา : 3,618 บาทโดยประมาณ

ขนาด : 66*25 mm

Microcontroller : STM32H747XI dual Cortex®-M7+M4 32bit low power Arm® MCU

Radio module : Murata 1DX dual WiFi 802.11b/g/n 65 Mbps and Bluetooth® (Bluetooth® Low Energy. 5 via Cordio stack, Bluetooth® Low Energy 4.2 via Arduino Stack)

Security : NXP SE0502 Secure Element

Board Power Supply (USB/VIN) : 5V

Supported Battery : Li-Po Single Cell, 3.7V, 700mAh Minimum (integrated charger)

Circuit Operating Voltage : 3.3V

Display Connector : MIPI DSI host & MIPI D-PHY to interface with low pin count large display

GPU : Chrom-ART graphical hardware Accelerator™

Timers : 22x timers and watchdogs

UART : 4x ports (2 with flow control)

Ethernet PHY : 10 / 100 Mbps

SD Card : Interface for SD Card connector

Operational Temperature : -40 °C to +85 °C

MKR Headers : Use any of the existing industrial MKR shields on it

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

High-density Connectors : Two 80 pin connectors will expose all of the board's peripherals to other devices

Camera Interface : 8-bit, up to 80 MHz

ADC : 3× ADCs with 16-bit max. resolution (up to 36 channels, up to 3.6 MSPS)

DAC : 2× 12-bit DAC (1 MHz)

USB-C : Host / Device, DisplayPort out, High / Full Speed, Power delivery

ภาพที่ 2.8 Wio Terminal(ATSAMD51P19)

2.6.2.7 Wio Terminal(ATSAMD51P19)

เป็นบอร์ดไมโครคอนโทรลเลอร์ที่มีการติดตั้งวงจรสื่อสารข้อมูลไร้สายแบบ WiFi และ BLE (บลูทูธกำลังงานต่ำ) พัฒนาโดย Seeedstudio มีการเชื่อมต่ออุปกรณ์อินพุตเอาต์พุตหลากหลาย เพื่อให้นำไปประยุกต์ใช้สร้างระบบควบคุมที่มีการติดต่อแบบไร้สายได้อย่างสะดวก

ข้อมูลทั่วไปของบอร์ด

ราคา : 1,712 บาทโดยประมาณ

ขนาด : 72*57*12 mm (วัสดุ ABS+PC)

MCU – Microchip SAMD51 (ATSAMD51P19) Arm Cortex-M4F microcontroller @ 120 MHz (can be overclocked to 200 MHz) with 192KB RAM,

Flash memory : 512KB

Storage : 4MB external SPI flash, MicroSD card slot up to 16GB

Display : 2.4″ LCD screen with 320×240 resolution (ILI9341 driver)

Audio : Microphone and buzzer

Connectivity : Dual-band WiFi 4 802.11b/g/n and Bluetooth 5.0 via Realtek RTL8720DN COrtex-M4/M0 WiSoC

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

USB : 1x USB-C OTG port

Sensors : LIS3DHTR accelerometer, light sensor, IR emitter

Expansion : 40-pin Raspberry Pi Compatible female header to connect to Raspberry Pi board, 2x multi-function Grove connectors, 20-pin FPC connector (no details provided)

Misc : 5-Way Switch, Power/Reset switch, 3x user-defined buttons

Power Supply : 5V via USB-C port

Temperature Range : -40°C ~ 85°C (TA)

2.7 Edge Impulse

เป็นแพลตฟอร์มการพัฒนาการเรียนรู้ของเครื่องบนไมโครคอนโทรลเลอร์หรือ TinyML ที่ใช้งานได้ฟรี ได้รับการปรับปรุงและพัฒนาจากวิศวกรจึงได้รับความไว้วางใจจากองค์กรและบริษัทต่าง ๆ ทั่วโลกสำหรับทั้งผู้พัฒนาและผู้ผลิตอุปกรณ์สำหรับการเรียนรู้ของเครื่องบนไมโครคอนโทรลเลอร์ รับประสบการณ์ในการพัฒนาและปรับใช้การเรียนรู้ของเครื่องบนไมโครคอนโทรลเลอร์โดยเน้นที่เซ็นเซอร์ระบบเสียงและคอมพิวเตอร์วิทัศน์ ทำให้สามารถแก้ปัญหาหรือพัฒนาการเรียนรู้ของเครื่องบนไมโครคอนโทรลเลอร์ได้อย่างง่ายดายโดยจำเป็นต้องเรียนจบในสาขาวิชาเฉพาะโดยตรง

แม้ Tiny Machine Learning จะเป็นเทคโนโลยีที่เพิ่งถูกพัฒนาขึ้นมาได้ไม่นานและไม่ได้แพร่หลายมาก แต่ก็มีการนำไปใช้อย่างมีนัยสำคัญในอุปกรณ์บางชนิด เช่น เทคโนโลยี keyword spotting เพื่อใช้ในการสั่งการทำงานของ Assistant AI ของ Google หรือ Apple เป็นต้น

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

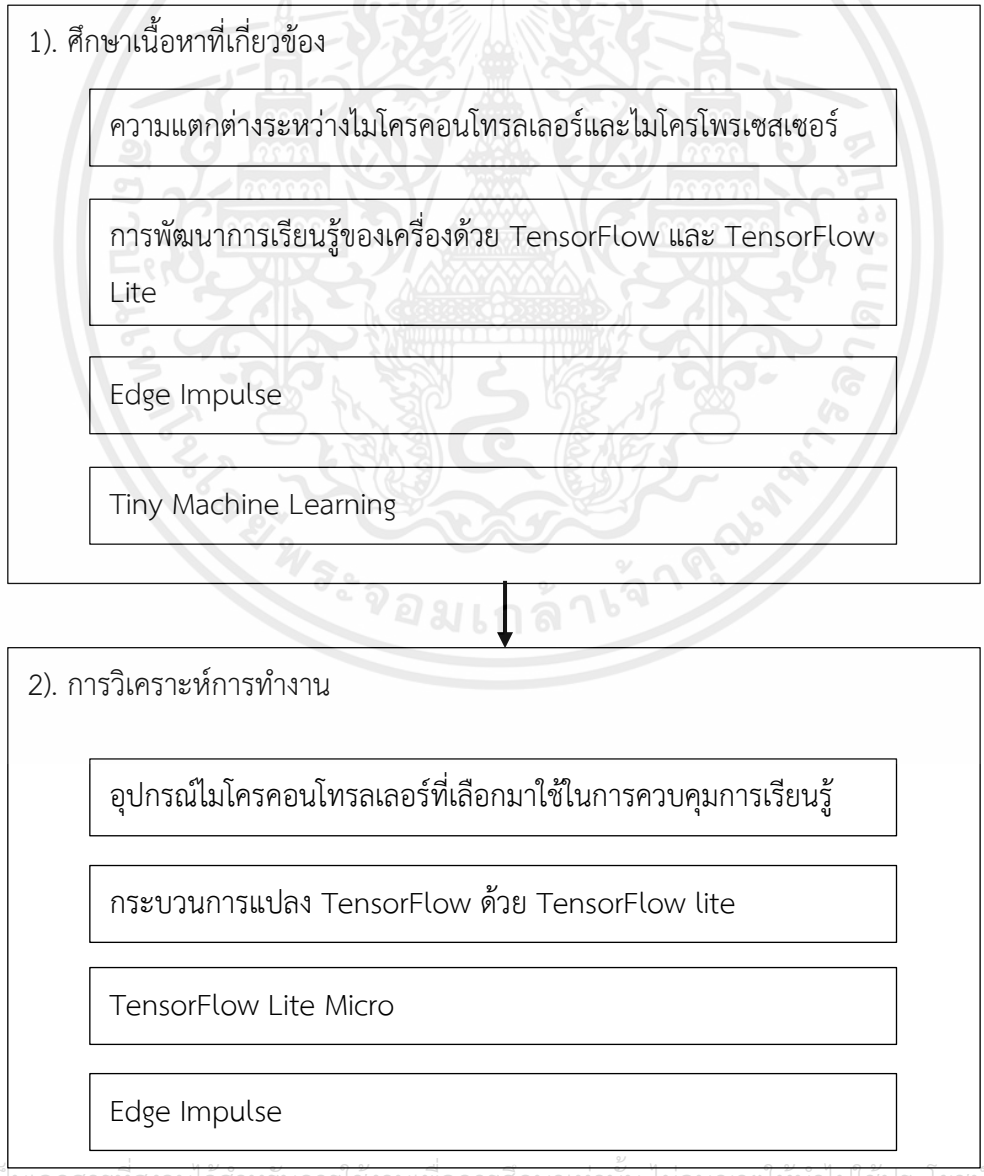
บทที่ 3

วิธีดำเนินการวิจัย

การวิจัยเพื่อศึกษาและการประยุกต์ใช้งานการเรียนรู้ของเครื่องบนไมโครคอนโทรลเลอร์มีจุดประสงค์เพื่อศึกษาการทำงานของการเรียนรู้ของเครื่องบนไมโครคอนโทรลเลอร์และนำไปประยุกต์ใช้ โดยทางผู้วิจัยมีขั้นตอนในการดำเนินการวิจัยใหญ่ ๆ ดังนี้

1) การศึกษาเนื้อหาที่เกี่ยวข้อง
2) การวิเคราะห์การทำงาน
3) การทดลองใช้งาน
4) การพัฒนาแบบเรียน

โดยสามารถนำไปเขียนแผนภาพแสดงรายละเอียดในการทำงานได้ดังนี้



เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

3). การทดลองใช้งาน

การทดสอบชี้เฉพาะคำสำคัญด้วยการเรียนรู้ของเครื่องบนไมโครคอนโทรลเลอร์

การทดสอบคาดเดาท่าทางการขยับด้วยการเรียนรู้ของเครื่องด้วยหน่วยตรวจวัดการเคลื่อนไหวภายใน

การทดสอบการตรวจจับวัตถุด้วยการเรียนรู้ของเครื่องบนอุปกรณ์ไมโครคอนโทรลเลอร์

4). การพัฒนาต้นแบบ

การนำสิ่งที่ได้ศึกษามาทั้งหมดมาประยุกต์ใช้เป็นต้นแบบของอุปกรณ์การเรียนรู้ของเครื่องบนไมโครคอนโทรลเลอร์

การนำสิ่งที่ได้เรียนรู้มาจัดทำแบบการเรียนรู้

เมื่อนำแผนผังการทำงานมาแจกแจงรายละเอียดต่าง ๆ จะได้ดังนี้

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

3.1 ศึกษาเนื้อหาที่เกี่ยวข้อง

3.1.1 ความแตกต่างของไมโครคอนโทรลเลอร์และไมโครโพรเซสเซอร์

ไมโครโพรเซสเซอร์ที่ใช้อยู่ในปัจจุบันจะไม่มีหน่วยความจำ RAM ROM และ Port อยู่ในตัวเอง จึงต้องต่อหน่วยความจำโปรแกรมภายนอก และต้องใช้ ICs ในการขยายพอร์ตเพิ่ม ข้อดีคือสามารถเพิ่มหน่วยความจำได้ตลอด ส่วนไมโครคอนโทรลเลอร์นั้นจะมีวงจรพื้นฐานอยู่แล้วภายในชิป (เช่น หน่วยความจำ RAM ROM และ I/O Port) เพราะเหตุนี้ไมโครคอนโทรลเลอร์จึงมีขนาดเล็กกว่าและราคาต่ำกว่าไมโครโพรเซสเซอร์

3.1.2 Machine Learning และ TensorFlow

Machine Learning คือ เทคโนโลยีที่ถูกนำมาใช้อย่างแพร่หลายในปัจจุบันหลายสายงานดังที่กล่าวไว้ในข้างต้น และ TensorFlow คือ library ที่พัฒนาขึ้นมาเพื่อทำให้การศึกษาลพัฒนา Machine Learning เป็นไปได้ง่ายและสะดวกมากขึ้น แต่ถึงแบบนั้นทั้ง Machine Learning และ TensorFlow ก็มีสาขางานที่ไม่สามารถนำมาใช้ร่วมกันได้เนื่องด้วยของจำกัดทางด้านพื้นที่ หนึ่งในนั้นคือ embedded system จึงได้มีการพัฒนา TensorFlow lite และ TensorFlow lite Micro เพื่อให้สามารถนำ Machine Learning มาใช้งานบนไมโครคอนโทรลเลอร์ได้และเรียกว่า Tiny Machine Learning หรือการเรียนรู้ของเครื่องขนาดเล็กนั้นเอง

3.1.3 Tiny Machine Learning

การนำการเรียนรู้ของเครื่องมาประยุกต์ใช้บนไมโครคอนโทรลเลอร์ หรือระบบสมองกลฝังตัว โดยมีจุดประสงค์เพื่อให้การเรียนรู้ของเครื่องสามารถใช้งานบนอุปกรณ์ที่ใช้พลังงานต่ำ ภายในหลักมิลิวัตต์ มีหน่วยคามจำและกำลังในการประมวลผลที่ต่ำ มีราคาถูกและสามารถใช้งานได้อย่างต่อเนื่องเป็นเวลานาน

3.1.4 Edge Impulse

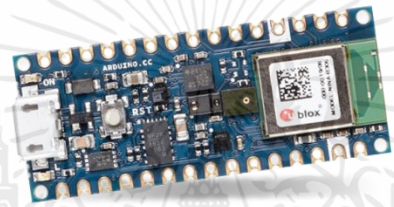
เป็น AI platform ที่ใช้ในการปรับโมเดลที่ได้จากการเรียนรู้ของเครื่องมาทำการปรับใช้ลงบนไมโครคอนโทรลเลอร์หรือระบบสมองกลฝังตัวอย่างง่าย เป็นเครื่องมือการใช้งานกระบวนการลอจิสติกส์แบบครบวงจร ใช้ TensorFlow สำหรับการฝึกข้อมูลเพิ่มประสิทธิภาพ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

3.2 การวิเคราะห์การทำงาน

3.2.1 อุปกรณ์ไมโครคอนโทรลเลอร์ที่เลือกมาใช้ในการควบคุมการเรียนรู้ของเครื่อง

ในปัจจุบันมีอุปกรณ์จำนวนมากที่ถูกพัฒนามาเพื่อใช้กับการเรียนรู้ของเครื่องบนอุปกรณ์ไมโครคอนโทรลเลอร์ แต่ทางคณะผู้จัดทำเลือกจะใช้บอร์ด Arduino nano 33 BLE ซึ่งเป็นบอร์ดชุด nano พัฒนาโดย Arduino เนื่องจากตัวบอร์ดมาพร้อมชิพ nRF52840 กับเซนเซอร์วัดค่าต่าง ๆ และไมโครโฟน ในตัว รองรับการเชื่อมต่อ Bluetooth เหมาะแก่การมาใช้เป็นตัวต้นแบบของระบบ Embedded มีคุณภาพสูง สามารถหาซื้อได้ง่าย ใช้อย่างแพร่หลายและมีชุมชนพร้อมให้การสนับสนุน ถึงแม้ตัวบอร์ดจะมีหน่วยความจุ Flash Memory เพียง 1 MB แต่ก็เหมาะที่จะนำมาเพื่อใช้ทดสอบการเรียนรู้ของเครื่องบนสภาวะจำกัดทางด้านทรัพยากร



ภาพที่ 3.1 Arduino nano 33 BLE

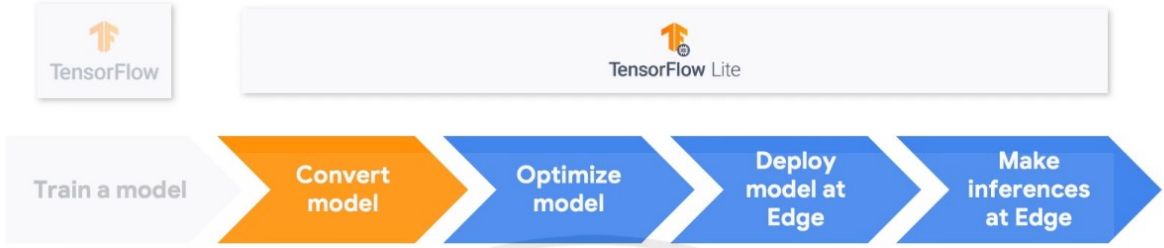
และบอร์ด Seeed Studio XIAO nRF52840 (Sense) ที่ตัวบอร์ดมาพร้อมกับชิพ nRF52840 รองรับการเชื่อมต่อ Bluetooth เหมาะแก่การมาใช้เป็นตัวต้นแบบของระบบ Embedded มีคุณภาพสูง สามารถหาซื้อได้ง่ายราคาถูก

ภาพที่ 3.2 Seeed Studio XIAO nRF52840 (Sense)

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

3.2.2 กระบวนการแปลง TensorFlow ด้วย TensorFlow lite

หลังจากที่ทำการ Train Model สำหรับใช้งานทั่วไปบน TensorFlow เรียบร้อยแล้ว ต่อมาจะเป็นการนำ Model ที่ได้ไปผ่านกระบวนการ TensorFlow lite เพื่อให้สามารถนำไปใช้บน edge device ได้ โดยสามารถดูขั้นตอนภาพรวมได้ดังแผนภาพต่อไปนี้

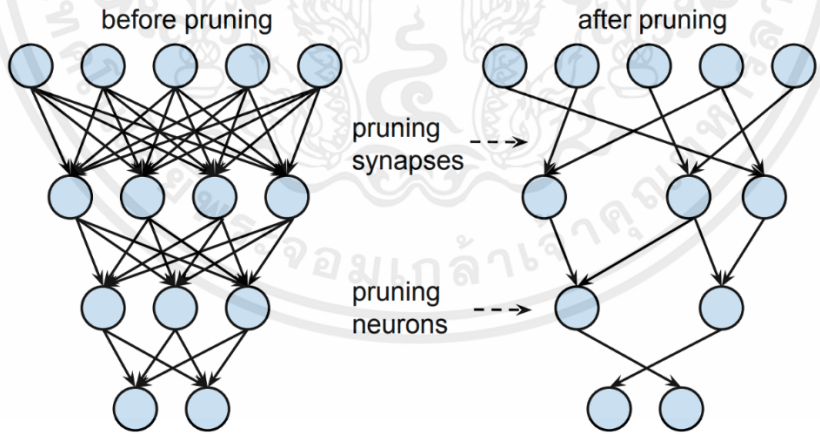


ภาพที่ 3.3 แผนภาพกระบวนการแปลง TensorFlow ด้วย TensorFlow lite

เมื่อนำแผนภาพนี้ไปขยายความจะได้ขั้นตอนย่อย ๆ ต่าง ๆ ต่อไปนี้

3.2.2.1 Convert Model

เป็นการเปลี่ยน format ของ Model จาก TensorFlow เดิม ให้เป็น TensorFlow lite โดยการใช้การใช้ API ของ TensorFlow lite โดยทั่วไปจะคือการ pruning (การตัดแต่ง) อธิบายคือการนำ synapses และ neurons ที่ไม่ส่งผลต่อผลลัพธ์หลังการ Convert อย่างเห็นได้ชัดออกไปทำให้โมเดลที่ได้มามีขนาดที่เล็กลง



ภาพที่ 3.4 ภาพแสดงการ Convert Model

3.2.2.2 Optimization

ในขั้นตอนการ Convert นั้น สามารถทำการตั้งค่าเพิ่มเติมเพื่อให้สามารถลดขนาดของ Model ให้เล็กลงไปอีก โดยมีหลัก ๆ สองวิธีคือ

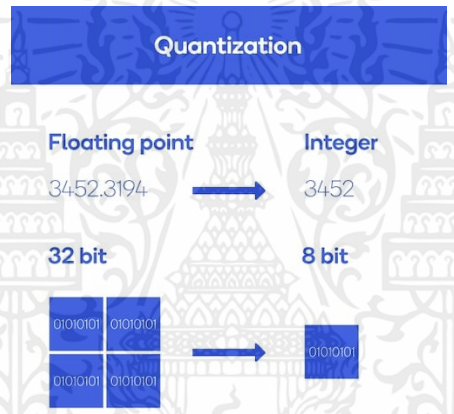
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

3.2.2.2.1 Automatic Optimization

คือ การใช้ API ของ TensorFlow lite เพื่อ Optimize Model โดยสามารถตั้งค่าได้ว่าจะทำการ Optimize ในส่วนใด เช่น ขนาดของ Model หรือค่าความหน่วง ขั้นตอนนี้จะทำให้ขนาดของโมเดลลดลง 3 ถึง 4 เท่าของขนาดไฟล์เดิม และทำให้ความหน่วงเวลาของการ Inference ลดลงได้ 2 ถึง 3 เท่า แต่ความแม่นยำของโมเดลจะลดลงเช่นกัน

3.2.2.2.2 Quantization

คือการเปลี่ยนรูปแบบข้อมูลจาก float32 เป็น int8 โดยทำเพื่อลดขนาดพื้นที่ที่ใช้ในการเก็บข้อมูล และลดความหน่วงเวลาที่ใช้ในการ Interference ส่งผลให้พลังงานที่ใช้ในการคำนวณลดลงเช่นกัน แต่เนื่องจากการ Quantize ทำให้วามละเอียดของโมเดลลดลงไปเช่นกันจะส่งผลให้ความแม่นยำของโมเดลจะลดลง



ภาพที่ 3.5 การ Quantization

3.2.2.3 Quantization-aware Training

คือการให้โมเดลที่อยู่ในระหว่างการ Training รับรู้การ Error ที่เกิดจากการ Quantizing เพื่อให้โมเดลที่ผ่านการ Training สามารถปรับรูปแบบต่าง ๆ ของ Model ให้ปรับตัวกับ Error รองรับการ Quantizing ที่จะเกิดขึ้นทำให้ผลที่ได้หลังการ Quantizing มีความแม่นยำมากขึ้น แต่ทั้งนี้การทำ Quantization-aware Training ไม่ได้การันตีเสมอไปว่า Model ที่ได้แม้จะมีขนาดเล็กลงแต่จะมีผลที่แม่นยำมากขึ้นเมื่อเทียบกับ Model ที่มได้ผ่าน Quantization-aware Training หลังการ Convert

3.2.3 TensorFlow lite Micro

หลังผ่านการ Convert และ Optimize เรียบร้อยแล้วต่อมาคือการ deploy Model ที่ผ่านการย่อขนาดลงบน Edge device เพื่อนำไปใช้งานต่อไป

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

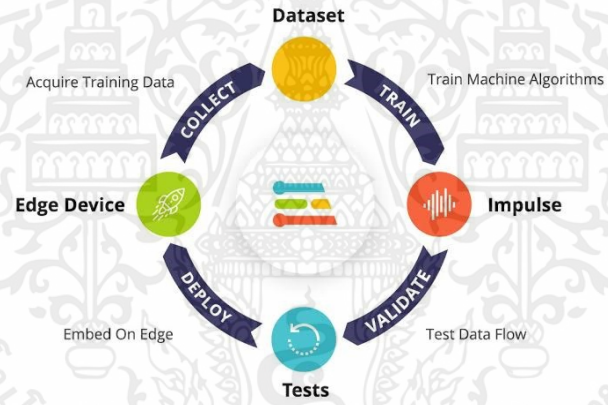
โดยทั่วไปแล้ว Edge device ประเภท Android หรือ IOS ที่มีระบบปฏิบัติการขนาดเล็กในตัวจะสามารถนำ Model ที่ผ่านการ Convert แล้วไปใช้ได้เลย แต่เมื่อต้องนำไปใช้บนอุปกรณ์ที่มีขนาดเล็กลงมาเช่น บอร์ด Arduino nano 33 BLE จำเป็นที่จะต้องมีการทำให้ตัว Model มีขนาดเล็กลงไปอีกหรือปรับให้สามารถใช้งานบนสภาวะที่มีทรัพยากรพื้นที่อย่างจำกัดได้

3.2.3.1 นิยามของ TensorFlow lite Micro

คือ library ที่จะทำการปรับให้ TensorFlow lite สามารถใช้งานบนอุปกรณ์ที่มีขนาดเล็กลงไปอีก เช่น Arduino nano 33 BLE เป็นต้น โดยจะเน้นไปที่การทำให้โมเดลสามารถ Inference บนอุปกรณ์ที่มีทรัพยากรจำกัดโดยการจำกัดพื้นที่สูงสุดที่ใช้ได้ และตัด Operators ต่าง ๆ ที่ไม่จำเป็นต่อ Model ของ TensorFlow ทำให้พื้นที่ใช้งานลดลง

3.2.4 Edge Impulse

Edge Impulse ใช้ประโยชน์การ TensorFlow สำหรับการฝึกโมเดลผ่านกระบวนการเพื่อให้สามารถนำไปใช้บนไมโครคอนโทรลเลอร์ได้อย่างง่าย

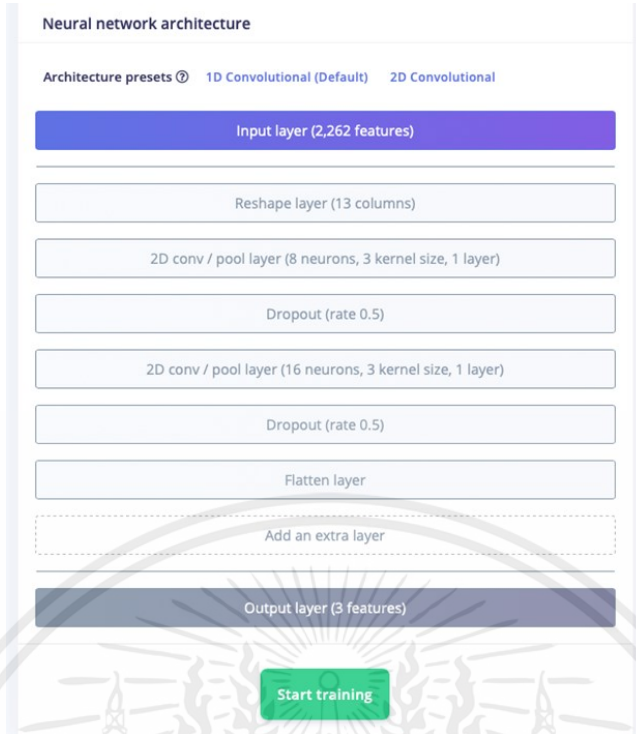


ภาพที่ 3.6 Edge Impulse

3.2.4.1 Train tiny model

ภายใน Edge Impulse สามารถใช้ไลบรารีโมเดลที่มีอยู่ซึ่งออกแบบมาสำหรับการใช้ในอุปกรณ์ไมโครคอนโทรลเลอร์ สามารถปรับแต่งพารามิเตอร์ เลเยอร์ใหม่เพื่อเข้าถึงได้โดยใช้ Python ผ่าน Keras API ของ TensorFlow ซึ่งง่ายต่อการเพิ่มโปรแกรมในการฝึกอบรม ใช้ไปป์ไลน์แบบกระบวนการลอจิสติกส์ครบวงจร ดำเนินการผ่านระบบคลาวด์ โมเดลที่ฝึกจะได้รับการปรับให้เหมาะสมสำหรับการฝังโดยผสานคุณประโยชน์ของ TensorFlow และเทคโนโลยีของ Edge Impulse

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

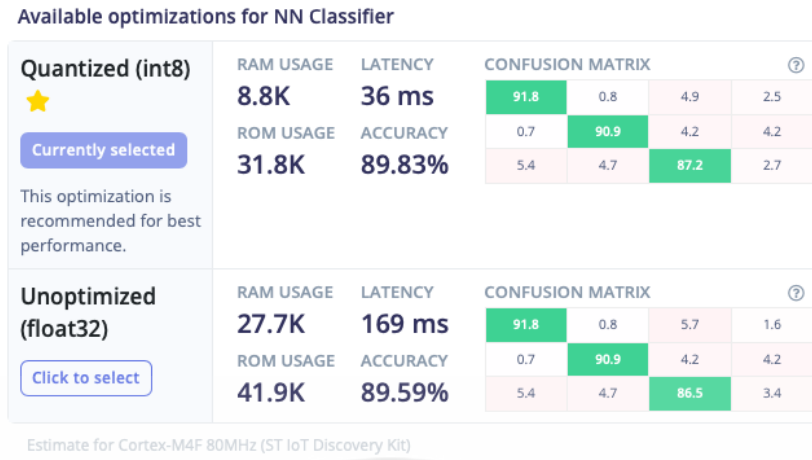


ภาพที่ 3.7 Train tiny model

3.2.4.2 Model optimization

การใช้โมเดลการเปลี่ยนรูปแบบข้อมูลจาก float32 เป็น int8 แบบเดียวกับ TensorFlow เพื่อวัดปริมาณโมเดล ลดความแม่นยำโดยให้ได้รับผลกระทบน้อยที่สุด ใช้ TensorFlow lite Micro ร่วมกับซอฟต์แวร์จำลอง Renode ทำให้ Edge Impulse สามารถประเมินเวลาแฝงและการใช้หน่วยความทรงจำในการฝังตัวของโมเดลได้อย่างแม่นยำ ง่ายต่อการกำหนดผลกระทบของการเพิ่มประสิทธิภาพ เพิ่มความยืดหยุ่นสูงสุดและความเข้ากันได้กับเวิร์กโฟลว์ที่มีอยู่ มีโมเดลที่ฝึกอบรมแล้วหลายรูปแบบ และสามารถส่งออกโมเดลเป็นแบบ TensorFlow หรือแบบที่ปรับให้เหมาะสมกับโมเดลแล้วอย่าง TensorFlow lite ได้

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

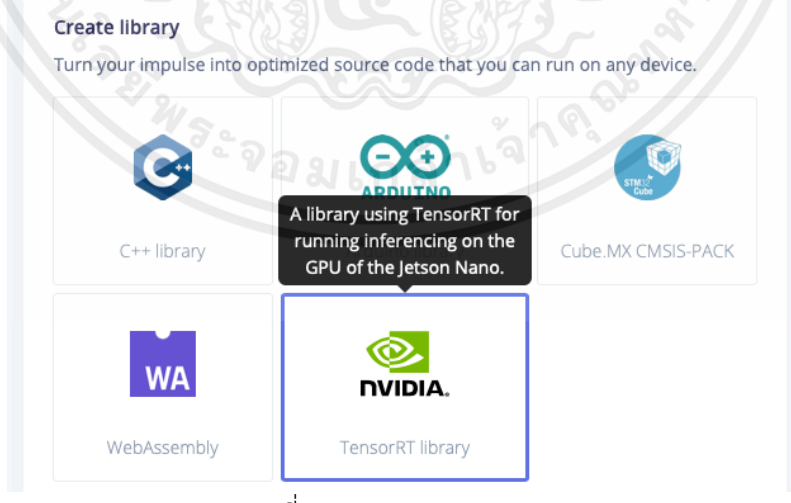


ภาพที่ 3.8 Model optimization

3.2.4.2 Deployment

เมื่อโมเดลผ่านการฝึกอบรมและทดสอบประสิทธิภาพแล้ว สามารถนำไปใช้กับไมโครคอนโทรลเลอร์ที่ทำงานร่วมกับภาษา C หรือ C++ โดยการใช้ชุดเครื่องมือในการปรับไลบรารีของโปรแกรมให้เหมาะสมซึ่งใช้ไปป์ไลน์การประมวลผลการการเรียนรู้ของเครื่องเชิงลึก

โดยใช้เทคโนโลยี TensorFlow คือ EON Compiler ของ Edge Impulse ในการแปลงโมเดล TensorFlow lite เป็นภาษา C++ สำหรับไมโครคอนโทรลเลอร์ โดยเรียกใช้ Kernal ที่มีประสิทธิภาพโดยไม่ต้องแปลงภาษาอัตโนมัติสำหรับอุปกรณ์ ไลบรารีจะถูกใช้ตามความเหมาะสม ส่งผลให้ประหยัดหน่วยความจำได้ห้าสิบเปอร์เซ็นต์ หรือใช้เครื่องมือในการกำหนดเครื่องมือที่จะฝังโมเดลที่ผ่านการฝึกมาแล้วที่มีอยู่อย่างหลากหลายได้



ภาพที่ 3.9 Deployment

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

3.3 การทดลองใช้งาน

3.3.1 การทดสอบชี้เฉพาะคำสำคัญด้วยการเรียนรู้ของเครื่องบนไมโครคอนโทรลเลอร์

การชี้เฉพาะคำสำคัญหรือ Keyword Spotting (KWS) เป็นการให้อุปกรณ์ที่สามารถรับเสียงจากผู้ใช้งานแล้วสามารถบอกได้ว่าผู้ใช้งานพูดคำที่กำหนดเอาไว้เมื่อไหร ซึ่งถูกนำไปใช้อย่างแพร่หลายบนอุปกรณ์ขนาดพกพาที่มีผู้ช่วยอัจฉริยะ โดยใช้เป็นคำสั่งในการเรียกใช้งาน เช่น Google Assistant , Siri เป็นต้น

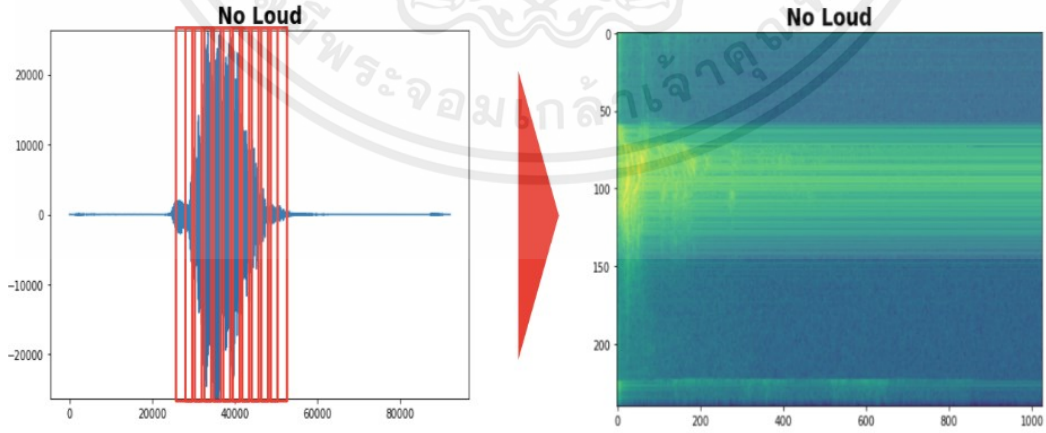
ภาพที่ 3.10 Keyword Spotting (KWS)

3.3.1.1 Collecting Data

ในส่วนแรกของการทำการชี้เฉพาะคำสำคัญคือการรวบรวมชุดข้อมูลของคำที่เราต้องการ โดยยิ่งถ้าตัวชุดข้อมูลมีความหลากหลายของโทนเสียงและสภาพแวดล้อมจะยิ่งทำให้ Model ที่ Train มามีความแม่นยำและสามารถนำไปใช้ได้จริงได้ดีขึ้น

3.3.1.2 Preprocess Data

ในการนำชุดข้อมูลไปใช้เพื่อ Train Model นั้น เมื่อได้ไฟล์เสียงมาแล้ว จะอยู่ในรูปแบบแกนเวลาซึ่งนำไปใช้กับการเรียนรู้ของเครื่องได้ยาก จึงต้องทำการเปลี่ยนรูปแบบแกนให้เป็นแกนความถี่ก่อนโดยการใช้ Fast Fourier Transform ทำให้แกนอยู่ในรูปของ Spectrogram

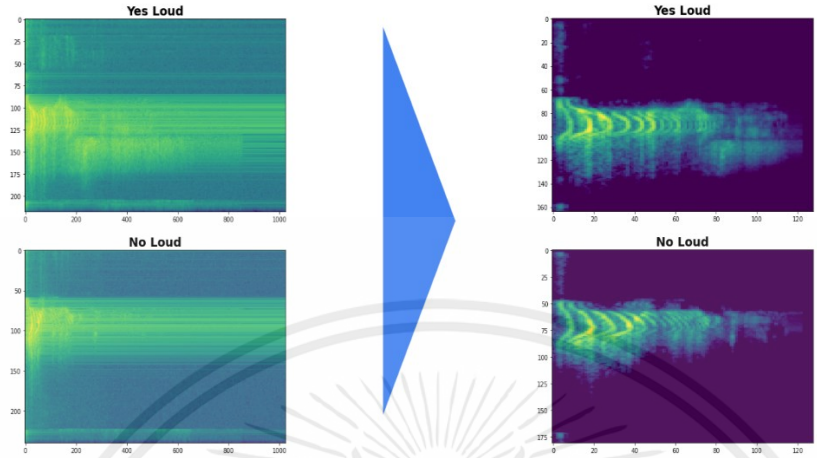


ภาพที่ 3.11 การเปลี่ยนรูปแบบแกนให้เป็นแกนความถี่

เมื่ออยู่ในรูปแบบ Spectrogram การเรียนรู้ของเครื่องจะสามารถแยกความแตกต่างของคำที่ผู้ใช้งานพูดได้ง่าย

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ต่อมาเมื่อพิจารณาตามหลักของการได้ยินของมนุษย์จะพบว่ามนุษย์สามารถแยกแยะเสียงที่มีความถี่ต่ำได้ดีกว่าเสียงที่มีความถี่สูงตามหลักของ Mel Scale จึงได้ทำการสร้าง Mel filter bank เพื่อรวมกลุ่มความถี่สูง และนำไปใช้กับการเรียนรู้ของเครื่อง



ภาพที่ 3.12 การสร้าง Mel filter bank เพื่อรวมกลุ่มความถี่สูง

3.3.1.3 Design Model

ในการทำ Neural Network เลือกใช้ Convolution Neural Network เพื่อใช้ในการสร้าง Model แต่ถ้าใช้ Convolution Model ขนาดของ Model ที่ผ่านการ Train มาแล้วจะยังมีขนาดที่ใหญ่เกินไปอยู่จึงมีการใช้ tiny_conv Model ที่เป็น Convolution Model ที่มีขนาดเล็กกว่าแต่ประสิทธิภาพที่ได้มาจะด้อยกว่า

ภาพที่ 3.13 Design Model

3.3.1.4 Training Model

นำ Model ที่ได้มา Train ใน TensorFlow

3.3.1.5 Evaluate Optimize และ Convert Model

ทำให้ Model ที่ผ่านการ Train มีขนาดเล็กลงด้วยการ Convert และ Optimize

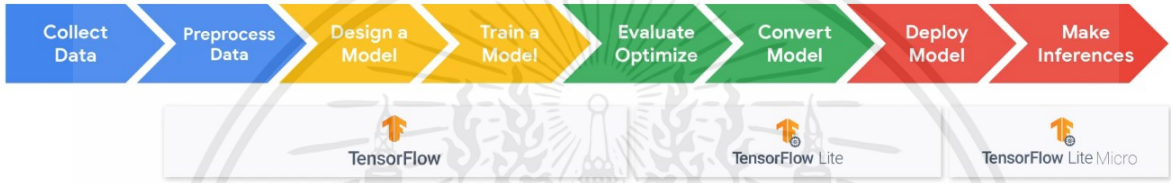
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

3.3.1.6 Deploy Model

Deploy Model ลงบนบอร์ดเพื่อนำไปใช้งานต่อไป

3.3.2 การทดสอบคาดเดาท่าทางการขยับด้วยการเรียนรู้ของเครื่องด้วยหน่วยตรวจวัดการเคลื่อนไหวภายใน

หน่วยตรวจวัดการเคลื่อนไหวภายในหรือ Inertial measurement unit (IMU) โดยปกติแล้วจะมีไว้เพื่อประมวลผลลักษณะการขยับในทิศทาง 6 แกน โดยทั่วไปจะถูกใช้เพื่อการวัดความสมดุลบนรถจักรยานยนต์หรือบังคับทิศทางของการบังคับของอุปกรณ์เคลื่อนที่ 6 แกน โดยในการทดสอบนี้จะนำมาประยุกต์ใช้กับการเรียนรู้ของเครื่องเพื่อนำไปวิเคราะห์ว่าการขยับนั้นขยับเป็นรูป ตัวอักษร หรือเลขอะไร



ภาพที่ 3.14 Inertial measurement unit (IMU)

3.3.2.1 Collecting Data

ในส่วนแรกคือการรวบรวมชุดข้อมูล โดยจะเก็บชุดข้อมูลการขยับ IMU เป็นลักษณะต่าง ๆ แล้วบันทึกว่าต้องการให้เป็นรูปอะไร ยิ่งชุดข้อมูลมีจำนวนมากและลักษณะความแตกต่างในการขยับของแต่ละรูปมีมากก็ยิ่งดีต่อการ Train Model

3.3.2.2 Preprocess Data

หลักการสำคัญคือเวลาที่มนุษย์จะขยับเป็นลักษณะท่าต่าง ๆ จะมี pattern ที่แน่นอนอยู่ เช่น การขยับมือเป็นรูปเลข 1 หรือตัวอักษร I จะเป็นการขยับขึ้นลงในแกนเดียว แต่ในความเป็นจริงนั้นติให้มี pattern ที่แน่นอนแต่การขยับของมนุษย์นั้นจะไม่ได้ออกมาเป็นองศาที่สวยงามสมบูรณ์แบบจึงมีการหยิบการเรียนรู้ของเครื่องมาใช้ในส่วนนี้เพื่อเปรียบเทียบส่วนที่ผิดพลาดเพียงเล็กน้อยแต่มีจำนวนที่มากแล้วประมวลผลออกมาว่าท่าทางที่ขยับนั้นเป็นลักษณะอะไร

โดยทั่วไปแล้ว IMU จะสามารถนำมาใช้เพื่อประมวลผลการขยับทั้ง 6 แกนทำให้ภาพที่ได้มีลักษณะเป็นสามมิติได้ แต่ในแง่การนำมาใช้งานเพื่อประมวลผลการขยับทางท่าในขั้นต้นนั้น จะมีความซับซ้อนและยุ่งยากต่อการนำไป Train จึงจะมีการแปลงข้อมูลที่ได้ให้อยู่ในรูปแบบภาพสองมิติก่อนการนำไป Train และประมวลผล

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

3.3.2.3 Design Model

ในการทำ Neural Network เลือกใช้ Convolution Neural Network เพื่อใช้ในการสร้าง Model เนื่องจากสิ่งทำหลัก ๆ ก็คือการ Image Classification

3.3.1.4 Training Model

นำ Model ที่ได้มา Train ใน TensorFlow

3.3.1.5 Evaluate Optimize และ Convert Model

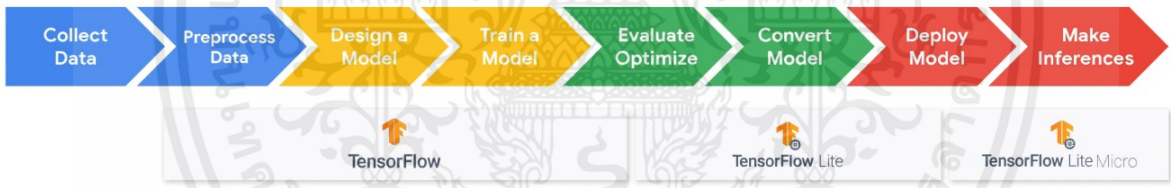
ทำให้ Model ที่ผ่านการ Train มีขนาดเล็กลงด้วยการ Convert และ Optimize

3.3.1.6 Deploy Model

Deploy Model ลงบนบอร์ดเพื่อนำไปใช้งานต่อไป

3.3.3 การทดสอบการตรวจจับวัตถุด้วยการเรียนรู้ของเครื่องบนอุปกรณ์ไมโครคอนโทรลเลอร์

การตรวจจับวัตถุคือการที่โปรแกรมสามารถประมวลผลภาพและบอกได้ว่าองค์ประกอบในภาพนั้นมีวัตถุอะไรอยู่บ้าง โดยทั่วไปการตรวจจับวัตถุจะมีไว้ใช้เพื่อระบุวัตถุในภาพจากกล้อง เช่น กล้องวงจรปิด และนำไปประยุกต์ใช้ต่อในการใช้งานต่าง ๆ



ภาพที่ 3.15 Object detection

3.3.3.1 Collecting Data

ในส่วนแรกคือการรวบรวมชุดข้อมูล โดยจะเก็บชุดข้อมูลรูปภาพของวัตถุที่อยากจะตรวจสอบและนำไป train model

3.3.2.3 Design Model

ในการทำ Neural Network เลือกใช้ Convolution Neural Network เพื่อใช้ในการสร้าง Model เนื่องจากสิ่งทำหลัก ๆ ก็คือการ Image Classification

3.3.1.4 Training Model

นำ Model ที่ได้มา Train ใน TensorFlow

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

3.3.1.5 Evaluate Optimize และ Convert Model

ทำให้ Model ที่ผ่านการ Train มีขนาดเล็กลงด้วยการ Convert และ Optimize

3.3.1.6 Deploy Model

Deploy Model ลงบนบอร์ดเพื่อนำไปใช้งานต่อไป

3.4 การพัฒนาต้นแบบ

ขั้นตอนนี้จะเป็นการนำสิ่งที่ศึกษาทั้งหมด ไม่ว่าจะเป็นการเปรียบเทียบการทำงานของ TensorFlow และ TensorFlow lite กระบวนการแปลงโมเดลของ TensorFlow ให้มีขนาดเล็กลงด้วย TensorFlow lite กระบวนการ Optimize เพื่อการติดตั้งด้วย TensorFlow lite Micro การใช้งาน tools ต่าง ๆ เช่น edge impulse รวมไปถึงการทดลองใช้งานจริง เช่น การชี้คำเฉพาะ การตรวจจับวัตถุด้วยภาพ มาเลือกสรร จัดทำและดัดแปลงเป็นต้นแบบเพื่อการศึกษา โดยมีจุดประสงค์เพื่อเรียบเรียงสิ่งที่ได้ศึกษามาทั้งหมดและเผยแพร่ในรูปแบบใดรูปแบบหนึ่งแก่สาธารณะชน

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

บทที่ 4
ผลการดำเนินการวิจัย

4.1 การเปรียบเทียบผลที่ได้ระหว่าง TensorFlow และ TensorFlow lite

ในกระบวนการแปลงจาก TensorFlow ให้อยู่ในรูปแบบสำหรับ Deploy ลงบนอุปกรณ์ไมโครคอนโทรลเลอร์ จำเป็นที่ต้องใช้ TensorFlow lite โดยการทดลองนี้จะนำผลลัพธ์ทั้งก่อนและหลังการแปลงมาเปรียบเทียบกัน เพื่อทดสอบว่ากระบวนการแปลงนั้น ส่งผลต่อความแม่นยำ ขนาดหรือผลลัพธ์หลังการ inference

4.1.1 ผลจากการ Convert Model

ทำการทดสอบกับ Model Image Classification ของภาพแมวและสุนัข โดยใช้ dataset ของรูปภาพแมวและสุนัขของ Kaggle Cats and Dogs Dataset (https://www.microsoft.com/en-us/download/details.aspx?id=54765)

โดยทำการโหลด dataset และ Train Model โดยใช้ Epoch = 5

```
import numpy as np
import matplotlib.pylab as plt

import tensorflow as tf
import tensorflow_hub as hub
import tensorflow_datasets as tfds
setattr(tfds.image_classification.cats_vs_dogs, '_URL',"https://download.microsoft.com/download/3/E/1/3E1C3F21-ECDB-4869-8368-6DFEBA77B919F/kagglecatsanddogs_5340.zip")


def format_image(image, label):
    image = tf.image.resize(image, (224, 224)) / 255.0
    return  image, label


(raw_train, raw_validation, raw_test), metadata = tfds.load(
    'cats_vs_dogs',
    split=['train[:80%]', 'train[80%:90%]', 'train[90%:]'],
    with_info=True,
    as_supervised=True,
)

num_examples = metadata.splits['train'].num_examples
num_classes = metadata.features['label'].num_classes
print(num_examples)
print(num_classes)

BATCH_SIZE = 32
train_batches = raw_train.shuffle(num_examples // 4).map(format_image).batch(BATCH_SIZE).prefetch(1)
validation_batches = raw_validation.map(format_image).batch(BATCH_SIZE).prefetch(1)
test_batches = raw_test.map(format_image).batch(1)

for image_batch, label_batch in train_batches.take(1):
    pass

image_batch.shape

module_selection = ("mobilenet_v2", 224, 1280)
handle_base, pixels, FV_SIZE = module_selection
MODULE_HANDLE ="https://tfhub.dev/google/tf2-preview/{}/feature_vector/4".format(handle_base)
IMAGE_SIZE = (pixels, pixels)
print("Using {} with input size {} and output dimension {}".format(MODULE_HANDLE, IMAGE_SIZE, FV_SIZE))

feature_extractor = hub.KerasLayer(MODULE_HANDLE,
                                   input_shape=IMAGE_SIZE + (3,),
                                   output_shape=[FV_SIZE],
                                   trainable=False)

print("Building model with", MODULE_HANDLE)

model = tf.keras.Sequential([
       feature_extractor,
       tf.keras.layers.Dense(num_classes, activation='softmax')
])

model.summary()

model.compile(optimizer='adam',
                loss='sparse_categorical_crossentropy',
                metrics=['accuracy'])

EPOCHS = 5

hist = model.fit(train_batches,
                 epochs=EPOCHS,
                 validation_data=validation_batches)
```

ภาพที่ 4.1 และภาพที่ 4.2 ผลจากการ Convert Model (1)

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
import pathlib
CATS_VS_DOGS_SAVED_MODEL = "exp_saved_model"
tf.saved_model.save(model, CATS_VS_DOGS_SAVED_MODEL)
```

```
import pathlib
converter = tf.lite.TFLiteConverter.from_saved_model(CATS_VS_DOGS_SAVED_MODEL)

tflite_model = converter.convert()
tflite_models_dir = pathlib.Path("/tmp/")

tflite_model_file = tflite_models_dir/'model1.tflite'
tflite_model_file.write_bytes(tflite_model)
# This will report back the file size in bytes
```

ภาพที่ 4.3 ผลจากการ Convert Model (2)

ทำการ save Model ที่ได้และนำไป Convert เป็น TensorFlow lite แล้วจึงดูขนาดของไฟล์ที่ได้มาเป็นผลลัพธ์

```
tflite_model_file = tflite_models_dir/'model1.tflite'
tflite_model_file.write_bytes(tflite_model)
# This will report back the file size in bytes
```

```
8877856
```

ภาพที่ 4.4 ผลจากการ Convert Model (3)

จากภาพจะเห็นได้ว่าขนาดไฟล์หลังผ่านการ Convert แล้วจะอยู่ที่ประมาณ 8.8 MB จากนั้นนำ Model ที่ผ่านการ Convert มาวัดความแม่นยำ

```
from tqdm import tqdm
# Load TFLite model and allocate tensors.
tflite_model_file = '/tmp/model1.tflite'
interpreter = tf.lite.Interpreter(model_path=tflite_model_file)
interpreter.allocate_tensors()

input_index = interpreter.get_input_details()[0]["index"]
output_index = interpreter.get_output_details()[0]["index"]

predictions = []

# This will report how many iterations per second, where each
# iteration is 100 predictions
test_labels, test_imgs = [], []
for img, label in tqdm(test_batches.take(100)):
    interpreter.set_tensor(input_index, img)
    interpreter.invoke()
    predictions.append(interpreter.get_tensor(output_index))

    test_labels.append(label.numpy()[0])
    test_imgs.append(img)


# This will tell you how many of the predictions were correct
score = 0
for item in range(0,len(predictions)):
  prediction=np.argmax(predictions[item])
  label = test_labels[item]
  if prediction==label:
    score=score+1

print("Out of 100 predictions I got " + str(score) + " correct")
```

ภาพที่ 4.5 ผลจากการ Convert Model (4)

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
100%|██████████| 100/100 [00:01<00:00, 88.77it/s]Out of 100 predictions I got 100 correct
```

ภาพที่ 4.6 ผลจากการ Convert Model (5)

จากผลจะพบว่ามีความแม่นยำที่ 100%

ต่อมาได้ทำการปรับค่า Epoch เป็น 20

```
module_selection = ("mobilenet_v2", 224, 1280)
handle_base, pixels, FV_SIZE = module_selection
MODULE_HANDLE ="https://tfhub.dev/google/tf2-preview/{}/feature_vector/4".format(handle_base)
IMAGE_SIZE = (pixels, pixels)
print("Using {} with input size {} and output dimension {}".format(MODULE_HANDLE, IMAGE_SIZE, FV_SIZE))

feature_extractor = hub.KerasLayer(MODULE_HANDLE,
                                   input_shape=IMAGE_SIZE + (3,),
                                   output_shape=[FV_SIZE],
                                   trainable=False)

print("Building model with", MODULE_HANDLE)

model = tf.keras.Sequential([
        feature_extractor,
        tf.keras.layers.Dense(num_classes, activation='softmax')
])

model.summary()

model.compile(optimizer='adam',
                  loss='sparse_categorical_crossentropy',
                  metrics=['accuracy'])

EPOCHS = 20

hist = model.fit(train_batches,
                 epochs=EPOCHS,
                 validation_data=validation_batches)
```

ภาพที่ 4.7 ผลจากการ Convert Model (6)

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ขนาดไฟล์ที่ได้หลังการ Convert มีขนาดประมาณ 8.8 MB

```
import pathlib
CATS_VS_DOGS_SAVED_MODEL = "exp_saved_model"
tf.saved_model.save(model, CATS_VS_DOGS_SAVED_MODEL)
```

```
import pathlib
converter = tf.lite.TFLiteConverter.from_saved_model(CATS_VS_DOGS_SAVED_MODEL)

tflite_model = converter.convert()
tflite_models_dir = pathlib.Path("/tmp/")

tflite_model_file = tflite_models_dir/'model1.tflite'
tflite_model_file.write_bytes(tflite_model)
# This will report back the file size in bytes
```

```
8877876
```

ภาพที่ 4.8 ผลจากการ Convert Model (7)

เมื่อนำมาวัดความแม่นยำจะพบว่ามีความแม่นยำที่ 99%

```
from tqdm import tqdm
# Load TFLite model and allocate tensors.
tflite_model_file = '/tmp/model1.tflite'
interpreter = tf.lite.Interpreter(model_path=tflite_model_file)
interpreter.allocate_tensors()

input_index = interpreter.get_input_details()[0]["index"]
output_index = interpreter.get_output_details()[0]["index"]

predictions = []

# This will report how many iterations per second, where each
# iteration is 100 predictions
test_labels, test_imgs = [], []
for img, label in tqdm(test_batches.take(100)):
    interpreter.set_tensor(input_index, img)
    interpreter.invoke()
    predictions.append(interpreter.get_tensor(output_index))

    test_labels.append(label.numpy()[0])
    test_imgs.append(img)


# This will tell you how many of the predictions were correct
score = 0
for item in range(0,len(predictions)):
  prediction=np.argmax(predictions[item])
  label = test_labels[item]
  if prediction==label:
    score=score+1

print("Out of 100 predictions I got " + str(score) + " correct")
```

```
100%|██████████| 100/100 [00:01<00:00, 88.05it/s]Out of 100 predictions I got 99 correct
```

ภาพที่ 4.9 ผลจากการ Convert Model (8)

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

4.1.2 ผลที่ได้จากการ Optimization และ Quantizing

ทำการทดลองโดยใช้ dataset เดียวกับการทดลอง 4.1.1 โดยใช้ Model ที่ผ่านการ Convert แล้วเป็นตัวเปรียบเทียบ

```
import pathlib
converter = tf.lite.TFLiteConverter.from_saved_model(CATS_VS_DOGS_SAVED_MODEL)

tflite_model = converter.convert()
tflite_models_dir = pathlib.Path("/tmp/")

tflite_model_file = tflite_models_dir/'model1.tflite'
tflite_model_file.write_bytes(tflite_model)
```

ภาพที่ 4.10 ผลจากการ Convert Model (9)

ขนาดของไฟล์ที่ได้หลังการ Convert มีเท่าเดิมคือ 8.8 MB

```
tflite_model_file = tflite_models_dir/'model1.tflite'
tflite_model_file.write_bytes(tflite_model)
# This will report back the file size in bytes

8877856
```

ภาพที่ 4.11ผลจากการ Convert Model (10)

ทำการ Optimization ด้วย API ของ TensorFlow lite เมื่อดูขนาดของไฟล์จะพบว่าขนาดของไฟล์จะเป็น 2.5 MB

```
import pathlib
converter = tf.lite.TFLiteConverter.from_saved_model(CATS_VS_DOGS_SAVED_MODEL)
converter.optimizations = [tf.lite.Optimize.DEFAULT]

tflite_model = converter.convert()
tflite_models_dir = pathlib.Path("/tmp/")

tflite_model_file = tflite_models_dir/'model2.tflite'
tflite_model_file.write_bytes(tflite_model)

2512200
```

ภาพที่ 4.12 ผลจากการ Convert Model (11)

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ต่อมาทำการ Quantizing จะได้ว่าขนาดของไฟล์อยู่ที่ 2.7 MB

```
import pathlib
converter = tf.lite.TFLiteConverter.from_saved_model(CATS_VS_DOGS_SAVED_MODEL)
converter.optimizations = [tf.lite.Optimize.DEFAULT]
def representative_data_gen():
     for input_value, _ in test_batches.take(100):
        yield [input_value]
converter.representative_dataset = representative_data_gen
converter.target_spec.supported_ops = [tf.lite.OpsSet.TFLITE_BUILTINS_INT8]

tflite_model = converter.convert()
tflite_models_dir = pathlib.Path("/tmp/")

tflite_model_file = tflite_models_dir/'model3.tflite'
tflite_model_file.write_bytes(tflite_model)
```

```
2715448
```

ภาพที่ 4.13 ผลจากการ Convert Model (12)

นำ Model ที่ผ่านการ Train ทั้ง 3 ไปทดสอบความแม่นยำ

```
from tqdm import tqdm

tflite_model_file = '/tmp/model.tflite'
interpreter = tf.lite.Interpreter(model_path=tflite_model_file)
interpreter.allocate_tensors()

input_index = interpreter.get_input_details()[0]["index"]
output_index = interpreter.get_output_details()[0]["index"]

predictions = []

test_labels, test_imgs = [], []
for img, label in tqdm(test_batches.take(100)):
    interpreter.set_tensor(input_index, img)
    interpreter.invoke()
    predictions.append(interpreter.get_tensor(output_index))

    test_labels.append(label.numpy()[0])
    test_imgs.append(img)

score = 0
for item in range(0,100):
  prediction=np.argmax(predictions[item])
  label = test_labels[item]
  if prediction==label:
    score=score+1

print("Out of 100 predictions I got " + str(score) + " correct")
```

```
100%|██████████| 100/100 [00:01<00:00, 57.54it/s]Out of 100 predictions I got 100 correct
```

ภาพที่ 4.14 ผลจากการ Convert Model (13)

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
from tqdm import tqdm
# Load TFLite model and allocate tensors.
tflite_model_file = '/tmp/model2.tflite'
interpreter = tf.lite.Interpreter(model_path=tflite_model_file)
interpreter.allocate_tensors()

input_index = interpreter.get_input_details()[0]["index"]
output_index = interpreter.get_output_details()[0]["index"]

predictions = []

test_labels, test_imgs = [], []
for img, label in tqdm(test_batches.take(100)):
    interpreter.set_tensor(input_index, img)
    interpreter.invoke()
    predictions.append(interpreter.get_tensor(output_index))

    test_labels.append(label.numpy()[0])
    test_imgs.append(img)

score = 0
for item in range(0,100):
  prediction=np.argmax(predictions[item])
  label = test_labels[item]
  if prediction==label:
    score=score+1

print("Out of 100 predictions I got " + str(score) + " correct")
```

```
100%|██████████| 100/100 [00:03<00:00, 29.95it/s]Out of 100 predictions I got 100 correct
```

ภาพที่ 4.15 ผลจากการ Convert Model (14)

```
from tqdm import tqdm
# Load TFLite model and allocate tensors.
tflite_model_file = '/tmp/model3.tflite'
interpreter = tf.lite.Interpreter(model_path=tflite_model_file)
interpreter.allocate_tensors()

input_index = interpreter.get_input_details()[0]["index"]
output_index = interpreter.get_output_details()[0]["index"]

predictions = []

test_labels, test_imgs = [], []
for img, label in tqdm(test_batches.take(100)):
    interpreter.set_tensor(input_index, img)
    interpreter.invoke()
    predictions.append(interpreter.get_tensor(output_index))

    test_labels.append(label.numpy()[0])
    test_imgs.append(img)

score = 0
for item in range(0,100):
  prediction=np.argmax(predictions[item])
  label = test_labels[item]
  if prediction==label:
    score=score+1

print("Out of 100 predictions I got " + str(score) + " correct")
```

```
100%|██████████| 100/100 [00:04<00:00, 21.92it/s]Out of 100 predictions I got 100 correct
```

ภาพที่ 4.16 ผลจากการ Convert Model (15)

จะพบว่าทั้งสาม Model ได้ความแม่นยำเป็น 100%

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

4.2 การทดลองใช้งาน TensorFlow lite และนำไปใช้บนบอร์ด Arduino nano 33 BLE

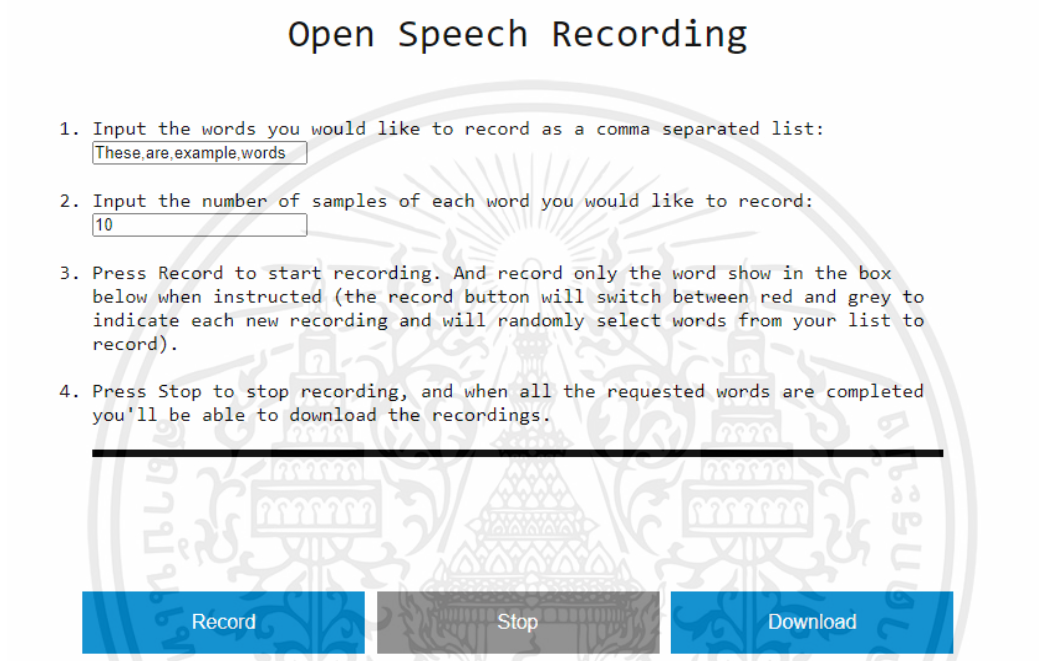
4.2.1 ขั้นตอนการทดสอบชี้เฉพาะคำสำคัญด้วยการเรียนรู้ของเครื่องบนไมโครคอนโทรลเลอร์

4.2.1.1 Collecting Data

ทำการเก็บบันทึกเสียงในรูปแบบ .ogg ไฟล์ผ่านเว็บไซต์

https://tinyml.seas.harvard.edu/open_speech_recording/

โดยทำการอัดเสียงคำที่ต้องการอย่างละ 3-4 ครั้ง



ภาพที่ 4.17 Open Speech Recording

4.2.1.2 Preprocessing and Model

ทำการ define และ import library ที่ใช้

```
import tensorflow as tf
#assert(tf.VERSION == "1.15.0") #Make sure you install Tensorflow 1.15.0 and restart the runtime or this will fail!
# Now that the runtime is set import things!
import sys
# We add this path so we can import the speech processing modules.
sys.path.append("/content/tensorflow/tensorflow/examples/speech_commands/")
import input_data
import models
import numpy as np
import glob
import os
import re
import shutil
from google.colab import files
!pip install ffmpeg-python &> 0
!apt-get update && apt-get -qq install xxd
```

ภาพที่ 4.18 Preprocessing and Model (16)

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

แปลงไฟล์ที่อัพโหลดให้เป็นรูปแบบ .wav

```
# convert the ogg files to wavs
!mkdir wavs
!find *.ogg -print0 | xargs -0 basename -s .ogg | xargs -I {} ffmpeg -i {}.ogg -ar 16000 wavs/{}.wav
!rm -r -f *.ogg

# then use pete's tool to only extract 1 second clips from them for use with the KWS pipeline
!mkdir trimmed_wavs
!git clone https://github.com/petewarden/extract_loudest_section.git
!make -C extract_loudest_section/
!/tmp/extract_loudest_section/gen/bin/extract_loudest_section 'wavs/*.wav' trimmed_wavs/
!rm -r -f /wavs
```

```
# Store them in the appropriate folders
data_index = {}
os.chdir('trimmed_wavs')
search_path = os.path.join('*.wav')
for wav_path in glob.glob(search_path):
    original_wav_path = wav_path
    parts = wav_path.split('_')
    if len(parts) > 2:
        wav_path = parts[0] + '_' + ''.join(parts[1:])
    matches = re.search('([^/_]+)_([^/_]+)\.wav', wav_path)
    if not matches:
        raise Exception('File name not in a recognized form:"%s"' % wav_path)
    word = matches.group(1).lower()
    instance = matches.group(2).lower()
    if not word in data_index:
      data_index[word] = {}
    if instance in data_index[word]:
        raise Exception('Audio instance already seen:"%s"' % wav_path)
    data_index[word][instance] = original_wav_path

output_dir = os.path.join('..', 'dataset')
try:
    os.mkdir(output_dir)
except:
    pass
for word in data_index:
  word_dir = os.path.join(output_dir, word)
  try:
      os.mkdir(word_dir)
      print('Created dir: ' + word_dir)
  except:
      print('Storing in existing dir: ' + word_dir)
  for instance in data_index[word]:
    wav_path = data_index[word][instance]
    output_path = os.path.join(word_dir, instance + '.wav')
    shutil.copyfile(wav_path, output_path)
os.chdir('..')
!rm -r -f trimmed_wavs
```

ภาพที่ 4.19 Preprocessing and Model (17)

กำหนดคำหรือไฟล์คำที่ต้องการ Train

```
WANTED_WORDS = "yes,no"
```

ภาพที่ 4.20 Preprocessing and Model (18)

กำหนด Training step และ Learning rate

```
TRAINING_STEPS = "12000,3000"
LEARNING_RATE = "0.001,0.0001"
```

ภาพที่ 4.21 Preprocessing and Model (19)

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

กำหนด Model เป็น tiny_conv

```
MODEL_ARCHITECTURE = 'tiny_conv'
```

ภาพที่ 4.22 Preprocessing and Model (20)

```
# Calculate the percentage of 'silence' and 'unknown' training samples required
# to ensure that we have equal number of samples for each label.
number_of_labels = WANTED_WORDS.count(',') + 1
number_of_total_labels = number_of_labels + 2 # for 'silence' and 'unknown' label
equal_percentage_of_training_samples = int(100.0/(number_of_total_labels))
SILENT_PERCENTAGE = equal_percentage_of_training_samples
UNKNOWN_PERCENTAGE = equal_percentage_of_training_samples

# Constants used during training only
VERBOSITY = 'DEBUG'
EVAL_STEP_INTERVAL = '1000'
SAVE_STEP_INTERVAL = '1000'

# Constants for training directories and filepaths
LOGS_DIR = 'logs/'
TRAIN_DIR = 'train/' # for training checkpoints and other files.

# Constants for inference directories and filepaths
import os
MODELS_DIR = 'models'
if not os.path.exists(MODELS_DIR):
  os.mkdir(MODELS_DIR)
MODEL_TF = os.path.join(MODELS_DIR, 'KWS_custom.pb')
MODEL_TFLITE = os.path.join(MODELS_DIR, 'KWS_custom.tflite')
FLOAT_MODEL_TFLITE = os.path.join(MODELS_DIR, 'KWS_custom_float.tflite')
MODEL_TFLITE_MICRO = os.path.join(MODELS_DIR, 'KWS_custom.cc')
SAVED_MODEL = os.path.join(MODELS_DIR, 'KWS_custom_saved_model')

# Constants which are shared during training and inference
PREPROCESS = 'micro'
WINDOW_STRIDE = 20

# Constants for Quantization
QUANT_INPUT_MIN = 0.0
QUANT_INPUT_MAX = 26.0
QUANT_INPUT_RANGE = QUANT_INPUT_MAX - QUANT_INPUT_MIN

# Constants for audio process during Quantization and Evaluation
SAMPLE_RATE = 16000
CLIP_DURATION_MS = 1000
WINDOW_SIZE_MS = 30.0
FEATURE_BIN_COUNT = 40
BACKGROUND_FREQUENCY = 0.8
BACKGROUND_VOLUME_RANGE = 0.1
TIME_SHIFT_MS = 100.0

# Use the custom local dataset and set the tes/val/train split
DATA_URL = ''
VALIDATION_PERCENTAGE = 10
TESTING_PERCENTAGE = 10
```

ภาพที่ 4.23 Preprocessing and Model (21)

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

4.2.1.3 Training Model

ทำการ Training Model ที่ได้มา

```
python tensorflow/tensorflow/examples/speech_commands/train.py \
 --data_dir={DATASET_DIR} \
 --data_url={DATA_URL} \
 --wanted_words={WANTED_WORDS} \
 --silence_percentage={SILENT_PERCENTAGE} \
 --unknown_percentage={UNKNOWN_PERCENTAGE} \
 --preprocess={PREPROCESS} \
 --window_stride={WINDOW_STRIDE} \
 --model_architecture={MODEL_ARCHITECTURE} \
 --how_many_training_steps={TRAINING_STEPS} \
 --learning_rate={LEARNING_RATE} \
 --train_dir={TRAIN_DIR} \
 --summaries_dir={LOGS_DIR} \
 --verbosity={VERBOSITY} \
 --eval_step_interval={EVAL_STEP_INTERVAL} \
 --save_step_interval={SAVE_STEP_INTERVAL}
```

```
!rm -rf {SAVED_MODEL}
!python tensorflow/tensorflow/examples/speech_commands/freeze.py \
--wanted_words=$WANTED_WORDS \
--window_stride_ms=$WINDOW_STRIDE \
--preprocess=$PREPROCESS \
--model_architecture=$MODEL_ARCHITECTURE \
--start_checkpoint=$TRAIN_DIR$MODEL_ARCHITECTURE'.ckpt-'{TOTAL_STEPS} \
--save_format=saved_model \
--output_file={SAVED_MODEL}
```

ภาพที่ 4.24 และภาพที่ 4.25 Training Model (22)

4.2.1.4 Evaluate Optimize และ Convert Model

ทำการ Convert และ Optimize Model ด้วย TensorFlow lite

```
model_settings = models.prepare_model_settings(
    len(input_data.prepare_words_list(WANTED_WORDS.split(','))),
    SAMPLE_RATE, CLIP_DURATION_MS, WINDOW_SIZE_MS,
    WINDOW_STRIDE, FEATURE_BIN_COUNT, PREPROCESS)
audio_processor = input_data.AudioProcessor(
    DATA_URL, DATASET_DIR,
    SILENT_PERCENTAGE, UNKNOWN_PERCENTAGE,
    WANTED_WORDS.split(','), VALIDATION_PERCENTAGE,
    TESTING_PERCENTAGE, model_settings, LOGS_DIR)
```

ภาพที่ 4.26 Evaluate Optimize (1)

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
REP_DATA_SIZE = 100
with tf.Session() as sess:
  # with tf.compat.v1.Session() as sess: #replaces the above line for use with TF2.x
  float_converter = tf.lite.TFLiteConverter.from_saved_model(SAVED_MODEL)
  float_tflite_model = float_converter.convert()
  float_tflite_model_size = open(FLOAT_MODEL_TFLITE, "wb").write(float_tflite_model)
  print("Float model is %d bytes" % float_tflite_model_size)

  converter = tf.lite.TFLiteConverter.from_saved_model(SAVED_MODEL)
  converter.optimizations = [tf.lite.Optimize.DEFAULT]
  converter.inference_input_type = tf.lite.constants.INT8
  # converter.inference_input_type = tf.compat.v1.lite.constants.INT8 #replaces the above line for use with TF2.x
  converter.inference_output_type = tf.lite.constants.INT8
  # converter.inference_output_type = tf.compat.v1.lite.constants.INT8 #replaces the above line for use with TF2.x
  def representative_dataset_gen():
    for i in range(REP_DATA_SIZE):
      data, _ = audio_processor.get_data(1, i*1, model_settings,
                                         BACKGROUND_FREQUENCY,
                                         BACKGROUND_VOLUME_RANGE,
                                         TIME_SHIFT_MS,
                                         'testing',
                                         sess)
      flattened_data = np.array(data.flatten(), dtype=np.float32).reshape(1, 1960)
      print(i)
      yield [flattened_data]
  converter.representative_dataset = representative_dataset_gen
  tflite_model = converter.convert()
  tflite_model_size = open(MODEL_TFLITE, "wb").write(tflite_model)
  print("Quantized model is %d bytes" % tflite_model_size)
```

ภาพที่ 4.27 Evaluate Optimize (2)

แปลง Model ให้เป็นรูปแบบที่สามารถนำไปใช้บน Microcontroller ได้ด้วย TensorFlow lite Micro โดยทำการแปลงให้สามารถใช้กับ C source file

```
!xxd -i {MODEL_TFLITE} > {MODEL_TFLITE_MICRO}
REPLACE_TEXT = MODEL_TFLITE.replace('/', '_').replace('.', '_')
!sed -i 's/'{REPLACE_TEXT}'/g_model/g' {MODEL_TFLITE_MICRO}
```

ภาพที่ 4.28 Convert Model (1)

```
!cat {MODEL_TFLITE_MICRO}
```

ภาพที่ 4.29 Convert Model (2)

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

4.2.1.5 Set up Code

นำโมเดลที่แปลงรูปแบบแล้วมาใส่ในไฟล์ของ Arduino

```
#include "micro_features_model.h"

// We need to keep the data array aligned on some architectures.
#ifdef __has_attribute
#define HAVE_ATTRIBUTE(x) __has_attribute(x)
#else
#define HAVE_ATTRIBUTE(x) 0
#endif
#if HAVE_ATTRIBUTE(aligned) || (defined(__GNUC__) && !defined(__clang__))
#define DATA_ALIGN_ATTRIBUTE __attribute__((aligned(4)))
#else
#define DATA_ALIGN_ATTRIBUTE
#endif

const unsigned char g_model[] DATA_ALIGN_ATTRIBUTE = {
    0x20, 0x00, 0x00, 0x00, 0x54, 0x46, 0x4c, 0x33, 0x00, 0x00, 0x00, 0x00,
    0x00, 0x00, 0x12, 0x00, 0x1c, 0x00, 0x04, 0x00, 0x08, 0x00, 0x0c, 0x00,
    0x10, 0x00, 0x14, 0x00, 0x00, 0x00, 0x18, 0x00, 0x12, 0x00, 0x00, 0x00,
    0x03, 0x00, 0x00, 0x00, 0x94, 0x48, 0x00, 0x00, 0x34, 0x42, 0x00, 0x00,
    0x1c, 0x42, 0x00, 0x00, 0x3c, 0x00, 0x00, 0x00, 0x04, 0x00, 0x00, 0x00,
    0x01, 0x00, 0x00, 0x00, 0x0c, 0x00, 0x00, 0x00, 0x08, 0x00, 0x0c, 0x00,
    0x04, 0x00, 0x08, 0x00, 0x08, 0x00, 0x00, 0x00, 0x08, 0x00, 0x00, 0x00,
    0x0b, 0x00, 0x00, 0x00, 0x13, 0x00, 0x00, 0x00, 0x6d, 0x69, 0x6e, 0x5f,
    0x72, 0x75, 0x6e, 0x74, 0x69, 0x6d, 0x65, 0x5f, 0x76, 0x65, 0x72, 0x73,
    0x69, 0x6f, 0x6e, 0x00, 0x0c, 0x00, 0x00, 0x00, 0xd4, 0x41, 0x00, 0x00,
    0xb4, 0x41, 0x00, 0x00, 0x24, 0x03, 0x00, 0x00, 0xf4, 0x02, 0x00, 0x00,
    0xec, 0x02, 0x00, 0x00, 0xe4, 0x02, 0x00, 0x00, 0xc4, 0x02, 0x00, 0x00,
    0xbc, 0x02, 0x00, 0x00, 0x2c, 0x00, 0x00, 0x00, 0x24, 0x00, 0x00, 0x00,
    0x1c, 0x00, 0x00, 0x00, 0x04, 0x00, 0x00, 0x00, 0x16, 0xbd, 0xff, 0xff,
    0x04, 0x00, 0x00, 0x00, 0x05, 0x00, 0x00, 0x00, 0x31, 0x2e, 0x35, 0x2e,
    0x30, 0x00, 0x00, 0x00, 0x94, 0xba, 0xff, 0xff, 0x98, 0xba, 0xff, 0xff,
    0x32, 0xbd, 0xff, 0xff, 0x04, 0x00, 0x00, 0x00, 0x80, 0x02, 0x00, 0x00,
    0xfa, 0xee, 0x28, 0xc4, 0xee, 0xfe, 0xcf, 0x0f, 0x1e, 0xf7, 0x1f, 0x06,
    0x0d, 0xed, 0xe9, 0x83, 0x5c, 0xc9, 0x18, 0xe3, 0xf9, 0x14, 0x28, 0x2a,
```

ภาพที่ 4.30 Set up Code (1)

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ทำการ Include library และ กำหนดตัวแปรต่าง ๆ

```
#include <TensorFlowLite.h>

#include "main_functions.h"

#include "audio_provider.h"
#include "command_responder.h"
#include "feature_provider.h"
#include "micro_features_micro_model_settings.h"
#include "micro_features_model.h"
#include "recognize_commands.h"
#include "tensorflow/lite/micro/micro_error_reporter.h"
#include "tensorflow/lite/micro/micro_interpreter.h"
#include "tensorflow/lite/micro/micro_mutable_op_resolver.h"
#include "tensorflow/lite/schema/schema_generated.h"
#include "tensorflow/lite/version.h"

// Globals, used for compatibility with Arduino-style sketches.
namespace {
tflite::ErrorReporter* error_reporter = nullptr;
const tflite::Model* model = nullptr;
tflite::MicroInterpreter* interpreter = nullptr;
TfLiteTensor* model_input = nullptr;
FeatureProvider* feature_provider = nullptr;
RecognizeCommands* recognizer = nullptr;
int32_t previous_time = 0;

// Create an area of memory to use for input, output, and intermediate arrays.
// The size of this will depend on the model you're using, and may need to be
// determined by experimentation.
constexpr int kTensorArenaSize = 10 * 1024;
uint8_t tensor_arena[kTensorArenaSize];
int8_t feature_buffer[kFeatureElementCount];
int8_t* model_input_buffer = nullptr;
}  // namespace
```

ภาพที่ 4.31 Set up Code (2)

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ใน void setup ทำการ load model

```
void setup() {
  // Set up logging. Google style is to avoid globals or statics because of
  // lifetime uncertainty, but since this has a trivial destructor it's okay.
  // NOLINTNEXTLINE(runtime-global-variables)
  static tflite::MicroErrorReporter micro_error_reporter;
  error_reporter = &micro_error_reporter;

  // Map the model into a usable data structure. This doesn't involve any
  // copying or parsing, it's a very lightweight operation.
  model = tflite::GetModel(g_model);
  if (model->version() != TFLITE_SCHEMA_VERSION) {
    TF_LITE_REPORT_ERROR(error_reporter,
                         "Model provided is schema version %d not equal "
                         "to supported version %d.",
                         model->version(), TFLITE_SCHEMA_VERSION);
    return;
  }
```

ภาพที่ 4.32 Set up Code (3)

กำหนด Operator ที่ใช้ โดยส่วนนี้เป็นส่วนหนึ่งของ TensorFlow lite Micro

```
  static tflite::MicroMutableOpResolver<4> micro_op_resolver(error_reporter);
  if (micro_op_resolver.AddDepthwiseConv2D() != kTfLiteOk) {
    return;
  }
  if (micro_op_resolver.AddFullyConnected() != kTfLiteOk) {
    return;
  }
  if (micro_op_resolver.AddSoftmax() != kTfLiteOk) {
    return;
  }
  if (micro_op_resolver.AddReshape() != kTfLiteOk) {
    return;
  }
```

ภาพที่ 4.33 Set up Code (4)

กำหนด Interpreter ที่ใช้

```
  static tflite::MicroInterpreter static_interpreter(
      model, micro_op_resolver, tensor_arena, kTensorArenaSize, error_reporter);
  interpreter = &static_interpreter;
```

ภาพที่ 4.34 Set up Code (5)

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

กำหนดขนาด Arena หรือขนาดพื้นที่สูงสุดที่ Model ใช้ได้

```
// Allocate memory from the tensor_arena for the model's tensors.
TfLiteStatus allocate_status = interpreter->AllocateTensors();
if (allocate_status != kTfLiteOk) {
  TF_LITE_REPORT_ERROR(error_reporter, "AllocateTensors() failed");
  return;
}
```

ภาพที่ 4.35 Set up Code (6)

กำหนดค่าต่าง ๆ ของ Model Input

```
// Get information about the memory area to use for the model's input.
model_input = interpreter->input(0);
if ((model_input->dims->size != 2) || (model_input->dims->data[0] != 1) ||
    (model_input->dims->data[1] !=
     (kFeatureSliceCount * kFeatureSliceSize)) ||
    (model_input->type != kTfLiteInt8)) {
  TF_LITE_REPORT_ERROR(error_reporter,
                       "Bad input tensor parameters in model");
  return;
}
model_input_buffer = model_input->data.int8;
```

ภาพที่ 4.36 Set up Code (7)

Set up ค่าต่าง ๆ ก่อน main loop เช่น feature provider เพื่อใช้ในการสร้าง spectrogram เป็นต้น

```
  // Prepare to access the audio spectrograms from a microphone or other source
  // that will provide the inputs to the neural network.
  // NOLINTNEXTLINE(runtime-global-variables)
  static FeatureProvider static_feature_provider(kFeatureElementCount,
                                                 feature_buffer);
  feature_provider = &static_feature_provider;

  static RecognizeCommands static_recognizer(error_reporter);
  recognizer = &static_recognizer;

  previous_time = 0;
}
```

ภาพที่ 4.37 Set up Code (8)

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ใน void loop ทำการกำหนดฟังก์ชั่นที่ทำให้สามารถสร้าง spectrogram ได้

```
void loop() {
  // Fetch the spectrogram for the current time.
  const int32_t current_time = LatestAudioTimestamp();
  int how_many_new_slices = 0;
  TfLiteStatus feature_status = feature_provider->PopulateFeatureData(
      error_reporter, previous_time, current_time, &how_many_new_slices);
  if (feature_status != kTfLiteOk) {
    TF_LITE_REPORT_ERROR(error_reporter, "Feature generation failed");
    return;
  }
  previous_time = current_time;
  // If no new audio samples have been received since last time, don't bother
  // running the network model.
  if (how_many_new_slices == 0) {
    return;
  }
```

ภาพที่ 4.38 Set up Code (9)

ทำการนำข้อมูลเสียงที่แปลงเป็น spectrogram แล้ว เข้าสู่กระบวนการ Inference ด้วยคำสั่ง invoke

```
  // Copy feature buffer to input tensor
  for (int i = 0; i < kFeatureElementCount; i++) {
    model_input_buffer[i] = feature_buffer[i];
  }

  // Run the model on the spectrogram input and make sure it succeeds.
  TfLiteStatus invoke_status = interpreter->Invoke();
  if (invoke_status != kTfLiteOk) {
    TF_LITE_REPORT_ERROR(error_reporter, "Invoke failed");
    return;
  }
```

ภาพที่ 4.39 Set up Code (10)

นำข้อมูลที่ได้มาหลังกระบวนการ process เพื่อนำไปใช้ต่อไป

```
  // Obtain a pointer to the output tensor
  TfLiteTensor* output = interpreter->output(0);
  // Determine whether a command was recognized based on the output of inference
  const char* found_command = nullptr;
  uint8_t score = 0;
  bool is_new_command = false;
  TfLiteStatus process_status = recognizer->ProcessLatestResults(
      output, current_time, &found_command, &score, &is_new_command);
  if (process_status != kTfLiteOk) {
    TF_LITE_REPORT_ERROR(error_reporter,
                         "RecognizeCommands::ProcessLatestResults() failed");
    return;
  }
```

ภาพที่ 4.40 Set up Code (11)

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ทำการใช้ฟังก์ชั่น RespondToCommand เพื่อให้คำสั่งหลังจากนั้นทำงานต่อไป

```
  RespondToCommand(error_reporter, current_time, found_command, score,
                   is_new_command);
}
```

ภาพที่ 4.41 Set up Code (12)

โดยในฟังก์ชั่น RespondToCommand จะมีการตรวจสอบว่าคำที่จับได้นั้นเป็นคำอะไรแล้วจึงสั่งให้ LED ทำงาน

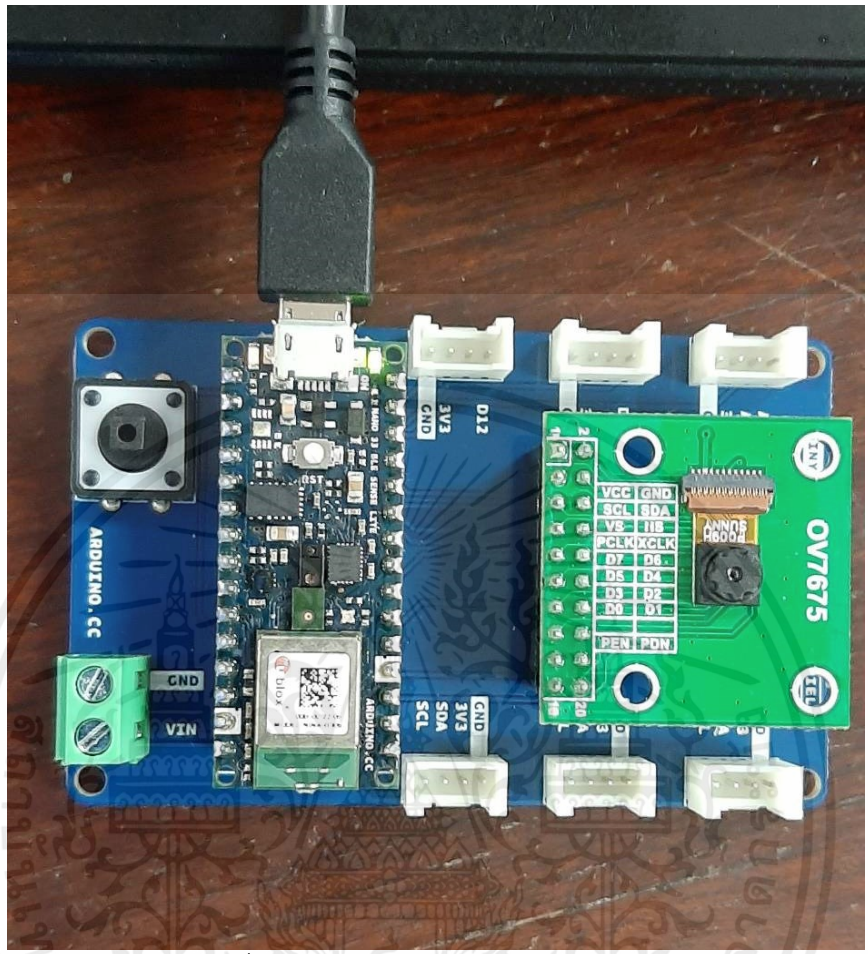
```
void RespondToCommand(tflite::ErrorReporter* error_reporter,
                      int32_t current_time, const char* found_command,
                      uint8_t score, bool is_new_command) {
  static bool is_initialized = false;
  if (!is_initialized) {
    pinMode(LED_BUILTIN, OUTPUT);
    // Pins for the built-in RGB LEDs on the Arduino Nano 33 BLE Sense
    pinMode(LEDR, OUTPUT);
    pinMode(LEDG, OUTPUT);
    pinMode(LEDB, OUTPUT);
    // Ensure the LED is off by default.
    // Note: The RGB LEDs on the Arduino Nano 33 BLE
    // Sense are on when the pin is LOW, off when HIGH.
    digitalWrite(LEDR, HIGH);
    digitalWrite(LEDG, HIGH);
    digitalWrite(LEDB, HIGH);
    is_initialized = true;
  }
  static int32_t last_command_time = 0;
  static int count = 0;
  static int certainty = 220;

  if (is_new_command) {
    TF_LITE_REPORT_ERROR(error_reporter, "Heard %s (%d) @%dms", found_command,
                         score, current_time);
    // If we hear a command, light up the appropriate LED
    if (found_command[0] == 'y') {
      last_command_time = current_time;
      digitalWrite(LEDG, LOW);  // Green for yes
    }

    if (found_command[0] == 'n') {
      last_command_time = current_time;
      digitalWrite(LEDR, LOW);  // Red for no
    }

    if (found_command[0] == 'u') {
      last_command_time = current_time;
      digitalWrite(LEDB, LOW);  // Blue for unknown
    }
  }
```

ภาพที่ 4.42 Set up Code (13)

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

4.2.2 ผลการทดสอบชี้เฉพาะคำสำคัญด้วยการเรียนรู้ของเครื่องบนไมโครคอนโทรลเลอร์

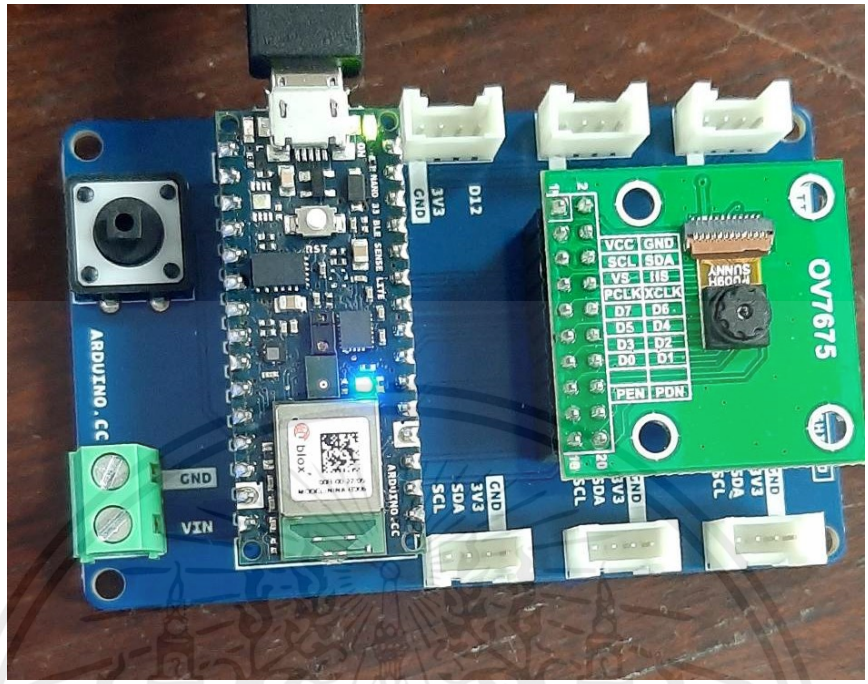
เมื่ออยู่ในสถานะรอคำพูด LED จะกระพริบเป็นสีส้ม



ภาพที่ 4.43 ผลการทดสอบ KWS (1)

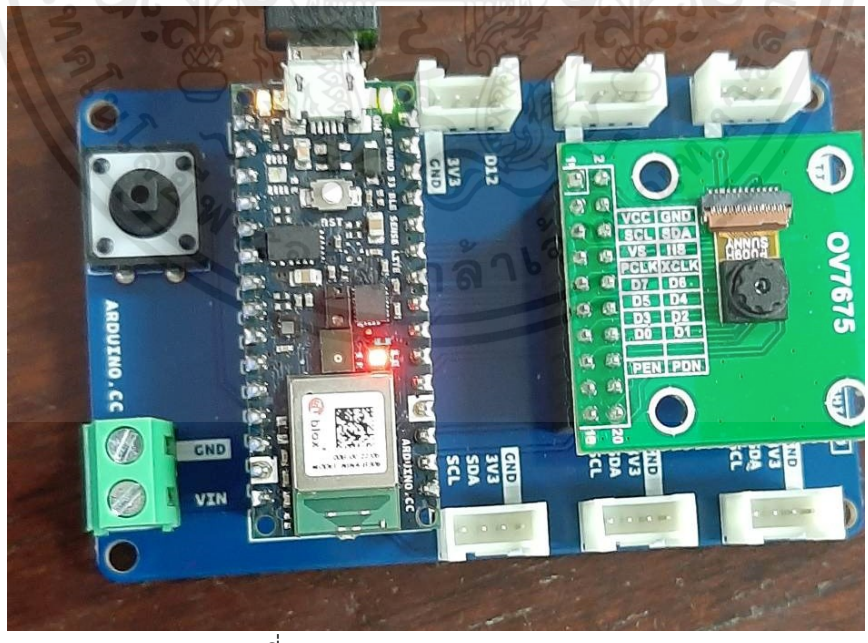
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

เมื่อคำที่ได้อยู่ในสถานะ unknown LED จะเป็นสีฟ้า



ภาพที่ 4.44 ผลการทดสอบ KWS (2)

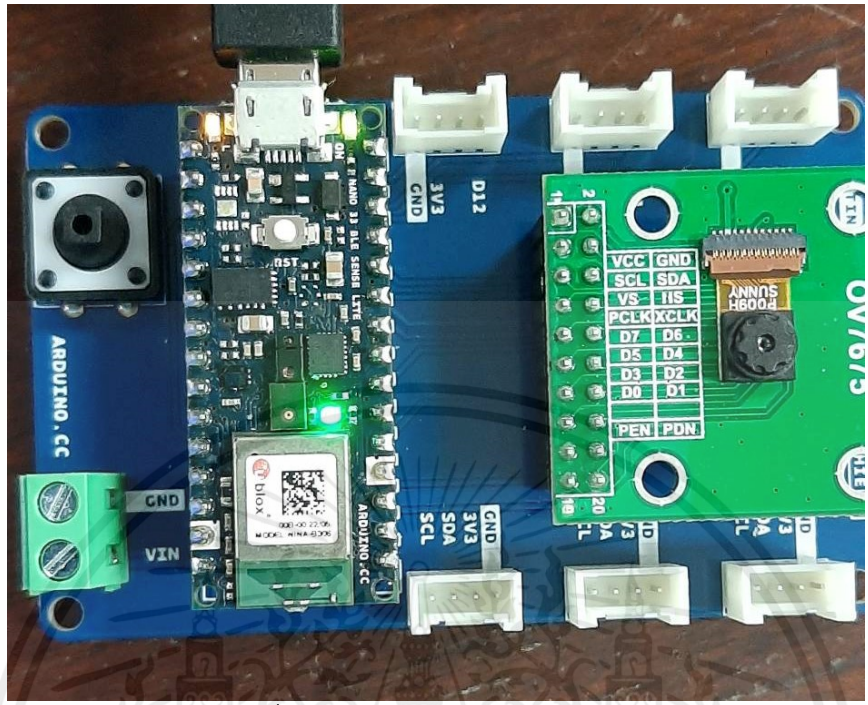
เมื่อได้ยินคำว่า no LED จะเป็นสีแดงตามที่ได้ตั้งค่าเอาไว้



ภาพที่ 4.45 ผลการทดสอบ KWS (3)

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

เมื่อได้ยินคำว่า Yes LED จะเป็นสีเขียวตามที่ได้ตั้งค่าเอาไว้



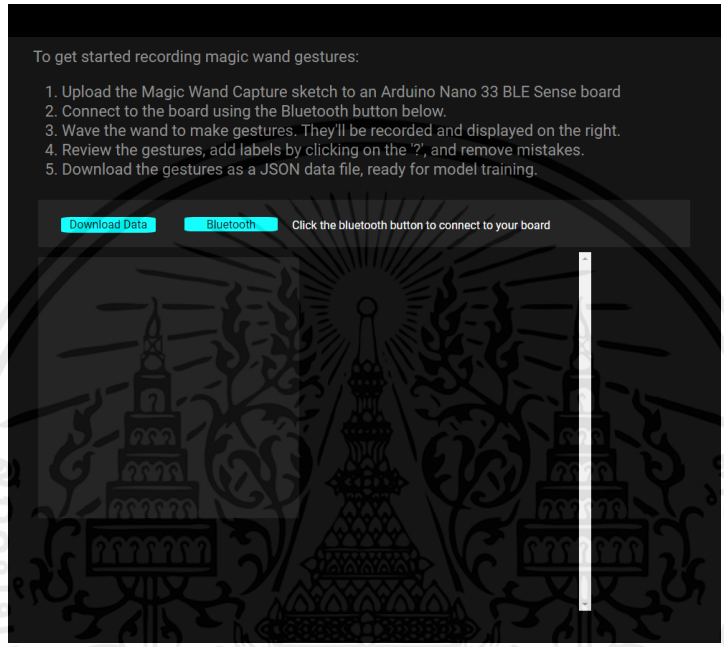
ภาพที่ 4.46 ผลการทดสอบ KWS (4)

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

4.2.3 ขั้นตอนการทดสอบคาดเดาท่าทางการขยับด้วยการเรียนรู้ของเครื่องด้วยหน่วยตรวจวัดการเคลื่อนไหวภายใน

4.2.3.1 Collecting Data

ทำการเก็บข้อมูลผ่านเว็บไซต์ https://tinyml.seas.harvard.edu/magic_wand/ โดยเก็บข้อมูลการขยับด้วยการเชื่อม Bluetooth กับบอร์ดที่มี IMU เมื่อเก็บข้อมูลประมาณ ท่าละ 20 ตัวอย่าง ก็โหลดข้อมูลเก็บไว้ในรูปแบบ .json file



ภาพที่ 4.47 หน้าต่างเว็บไซต์ที่ใช้เก็บข้อมูล IMU

4.2.1.2 Preprocessing and Model

ทำการ define และ import library ที่ใช้

```
%tensorflow_version 2.x
import tensorflow as tf
from tensorflow import keras
from tensorflow.keras import layers
from tensorflow.keras.preprocessing import image_dataset_from_directory
from google.colab import files
from IPython.display import Image, display
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
from pathlib import Path
import shutil
import PIL
import math
import glob
import json
import os
!apt-get update && apt-get -qq install xxd
```

ภาพที่ 4.48 Preprocessing and Model (1)

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
# Define filenames and set up directory structure
MODELS_DIR = 'models'
if not os.path.exists(MODELS_DIR):
  os.mkdir(MODELS_DIR)
SAVED_MODEL_FILENAME = os.path.join(MODELS_DIR, "magic_wand")
FLOAT_TFL_MODEL_FILENAME = os.path.join(MODELS_DIR, "magic_wand_float.tfl")
QUANTIZED_TFL_MODEL_FILENAME = os.path.join(MODELS_DIR, "magic_wand.tfl")
TFL_CC_MODEL_FILENAME = os.path.join(MODELS_DIR, "magic_wand.cc")

DATASET_DIR =  'dataset'
if not os.path.exists(DATASET_DIR):
  os.mkdir(DATASET_DIR)
TRAIN_DIR = os.path.join(DATASET_DIR, "train")
VAL_DIR = os.path.join(DATASET_DIR, "validation")
TEST_DIR = os.path.join(DATASET_DIR, "test")
!rm -rf sample_data

CHKPT_DIR =  'checkpoints'
if not os.path.exists(CHKPT_DIR):
  os.mkdir(CHKPT_DIR)

# Train Split
TEST_PERCENTAGE = 10
VALIDATION_PERCENTAGE = 10
TRAIN_PERCENTAGE = 100 - (TEST_PERCENTAGE + VALIDATION_PERCENTAGE)
```

```
# Upload your files
os.chdir("/content/dataset")
uploaded = files.upload()
os.chdir("/content")
```

ภาพที่ 4.49 Preprocessing and Model (2)

กำหนดจำนวนท่าทางที่ต้องการ

```
NUM_GESTURES = 10 #
```

แปลง json file เป็น python object

```
dataset_jsons = DATASET_DIR + "/*.json"
strokes = []
for filename in glob.glob(dataset_jsons):
  with open(filename, "r") as file:
    file_contents = file.read()
  file_data = json.loads(file_contents)
  for stroke in file_data["strokes"]:
    stroke["filename"] = filename
    strokes.append(stroke)
```

ภาพที่ 4.50 Preprocessing and Model (3)

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

Preprocess ข้อมูล

```
FIXED_POINT = 256

def mul_fp(a, b):
  return (a * b) / FIXED_POINT

def div_fp(a, b):
  if b == 0:
    b = 1
  return (a * FIXED_POINT) / b

def float_to_fp(a):
  return math.floor(a * FIXED_POINT)

def norm_to_coord_fp(a, range_fp, half_size_fp):
  a_fp = float_to_fp(a)
  norm_fp = div_fp(a_fp, range_fp)
  return mul_fp(norm_fp, half_size_fp) + half_size_fp

def round_fp_to_int(a):
  return math.floor((a + (FIXED_POINT / 2)) / FIXED_POINT)

def gate(a, min, max):
  if a < min:
    return min
  elif a > max:
    return max
  else:
    return a
```

```
def rasterize_stroke(stroke_points, x_range, y_range, width, height):
  num_channels = 3
  buffer_byte_count = height * width * num_channels
  buffer = bytearray(buffer_byte_count)

  width_fp = width * FIXED_POINT
  height_fp = height * FIXED_POINT
  half_width_fp = width_fp / 2
  half_height_fp = height_fp / 2
  x_range_fp = float_to_fp(x_range)
  y_range_fp = float_to_fp(y_range)

  t_inc_fp = FIXED_POINT / len(stroke_points)

  one_half_fp = (FIXED_POINT / 2)

  for point_index in range(len(stroke_points) - 1):
    start_point = stroke_points[point_index]
    end_point = stroke_points[point_index + 1]
    start_x_fp = norm_to_coord_fp(start_point["x"], x_range_fp, half_width_fp)
    start_y_fp = norm_to_coord_fp(-start_point["y"], y_range_fp, half_height_fp)
    end_x_fp = norm_to_coord_fp(end_point["x"], x_range_fp, half_width_fp)
    end_y_fp = norm_to_coord_fp(-end_point["y"], y_range_fp, half_height_fp)
    delta_x_fp = end_x_fp - start_x_fp
    delta_y_fp = end_y_fp - start_y_fp

    t_fp = point_index * t_inc_fp
    if t_fp < one_half_fp:
      local_t_fp = div_fp(t_fp, one_half_fp)
      one_minus_t_fp = FIXED_POINT - local_t_fp
      red = round_fp_to_int(one_minus_t_fp * 255)
      green = round_fp_to_int(local_t_fp * 255)
      blue = 0
    else:
      local_t_fp = div_fp(t_fp - one_half_fp, one_half_fp)
      one_minus_t_fp = FIXED_POINT - local_t_fp
      red = 0
      green = round_fp_to_int(one_minus_t_fp * 255)
      blue = round_fp_to_int(local_t_fp * 255)
    red = gate(red, 0, 255)
    green = gate(green, 0, 255)
    blue = gate(blue, 0, 255)
```

ภาพที่ 4.51 Preprocessing and Model (4)

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
    if abs(delta_x_fp) > abs(delta_y_fp):
      line_length = abs(round_fp_to_int(delta_x_fp))
      if delta_x_fp > 0:
        x_inc_fp = 1 * FIXED_POINT
        y_inc_fp = div_fp(delta_y_fp, delta_x_fp)
      else:
        x_inc_fp = -1 * FIXED_POINT
        y_inc_fp = -div_fp(delta_y_fp, delta_x_fp)
    else:
      line_length = abs(round_fp_to_int(delta_y_fp))
      if delta_y_fp > 0:
        y_inc_fp = 1 * FIXED_POINT
        x_inc_fp = div_fp(delta_x_fp, delta_y_fp)
      else:
        y_inc_fp = -1 * FIXED_POINT
        x_inc_fp = -div_fp(delta_x_fp, delta_y_fp)
    for i in range(line_length + 1):
      x_fp = start_x_fp + (i * x_inc_fp)
      y_fp = start_y_fp + (i * y_inc_fp)
      x = round_fp_to_int(x_fp)
      y = round_fp_to_int(y_fp)
      if (x < 0) or (x >= width) or (y < 0) or (y >= height):
        continue
      buffer_index = (y * width * num_channels) + (x * num_channels)
      buffer[buffer_index + 0] = red
      buffer[buffer_index + 1] = green
      buffer[buffer_index + 2] = blue

  np_buffer = np.frombuffer(buffer, dtype=np.uint8).reshape(height, width, num_channels)

  return np_buffer
```

ภาพที่ 4.52 Preprocessing and Model (5)

```
X_RANGE = 0.6
Y_RANGE = 0.6

def ensure_empty_dir(dirname):
  dirpath = Path(dirname)
  if dirpath.exists() and dirpath.is_dir():
    shutil.rmtree(dirpath)
  dirpath.mkdir()

def augment_points(points, move_range, scale_range, rotate_range):
  move_x = np.random.uniform(low=-move_range, high=move_range)
  move_y = np.random.uniform(low=-move_range, high=move_range)
  scale = np.random.uniform(low=1.0-scale_range, high=1.0+scale_range)
  rotate = np.random.uniform(low=-rotate_range, high=rotate_range)

  x_axis_x = math.cos(rotate) * scale
  x_axis_y = math.sin(rotate) * scale

  y_axis_x = -math.sin(rotate) * scale
  y_axis_y = math.cos(rotate) * scale

  new_points = []
  for point in points:
    old_x = point["x"]
    old_y = point["y"]
    new_x = (x_axis_x * old_x) + (x_axis_y * old_y) + move_x
    new_y = (y_axis_x * old_x) + (y_axis_y * old_y) + move_y
    new_points.append({"x": new_x, "y": new_y})

  return new_points
```

ภาพที่ 4.53 Preprocessing and Model (6)

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```python
def save_strokes_as_images(strokes, root_folder, width, height, augment_count):
  ensure_empty_dir(root_folder)
  labels = set()
  for stroke in strokes:
    labels.add(stroke["label"].lower())
  for label in labels:
    label_path = Path(root_folder, label)
    ensure_empty_dir(label_path)

  label_counts = {}
  for stroke in strokes:
    points = stroke["strokePoints"]
    label = stroke["label"].lower()
    if label == "":
      raise Exception("Missing label for %s:%d" % (stroke["filename"], stroke["index"]))
    if label not in label_counts:
      label_counts[label] = 0
    label_count = label_counts[label]
    label_counts[label] += 1
    raster = rasterize_stroke(points, X_RANGE, Y_RANGE, width, height)
    image = PIL.Image.fromarray(raster)
    image.save(Path(root_folder, label, str(label_count) + ".png"))
    for i in range(augment_count):
      augmented_points = augment_points(points, 0.1, 0.1, 0.3)
      raster = rasterize_stroke(augmented_points, X_RANGE, Y_RANGE, width, height)
      image = PIL.Image.fromarray(raster)
      image.save(Path(root_folder, label, str(label_count) + "_a" + str(i) + ".png"))
  return labels
```

ภาพที่ 4.54 Preprocessing and Model (7)

```python
IMAGE_WIDTH = 32
IMAGE_HEIGHT = 32

shuffled_strokes = strokes
np.random.shuffle(shuffled_strokes)

test_count = math.floor((len(shuffled_strokes) * TEST_PERCENTAGE) / 100)
validation_count = math.floor((len(shuffled_strokes) * VALIDATION_PERCENTAGE) / 100)
test_strokes = shuffled_strokes[0:test_count]
validation_strokes = shuffled_strokes[test_count:(test_count + validation_count)]
train_strokes = shuffled_strokes[(test_count + validation_count):]

labels_test  = save_strokes_as_images(test_strokes, TEST_DIR, IMAGE_WIDTH, IMAGE_HEIGHT, 10)
labels_val   = save_strokes_as_images(validation_strokes, VAL_DIR, IMAGE_WIDTH, IMAGE_HEIGHT, 0)
labels_train = save_strokes_as_images(train_strokes, TRAIN_DIR, IMAGE_WIDTH, IMAGE_HEIGHT, 10)
```

ภาพที่ 4.55 Preprocessing and Model (8)

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

Define Model ที่ใช้

```python
def make_model(input_shape, num_classes):
    inputs = keras.Input(shape=input_shape)

    # Entry block
    x = layers.experimental.preprocessing.Rescaling(1.0 / 255)(inputs)
    x = layers.Conv2D(16, 3, strides=2, padding="same")(x)
    x = layers.BatchNormalization()(x)
    x = layers.Activation("relu")(x)
    x = layers.Dropout(0.5)(x)

    x = layers.Conv2D(32, 3, strides=2, padding="same")(x)
    x = layers.BatchNormalization()(x)
    x = layers.Activation("relu")(x)
    x = layers.Dropout(0.5)(x)

    x = layers.Conv2D(64, 3, strides=2, padding="same")(x)
    x = layers.BatchNormalization()(x)
    x = layers.Activation("relu")(x)
    x = layers.Dropout(0.5)(x)

    x = layers.GlobalAveragePooling2D()(x)
    activation = "softmax"
    units = num_classes

    x = layers.Dropout(0.5)(x)
    outputs = layers.Dense(units, activation=activation)(x)
    return keras.Model(inputs, outputs)

model = make_model(input_shape=(IMAGE_WIDTH, IMAGE_HEIGHT, 3), num_classes=NUM_GESTURES)
```

ภาพที่ 4.56 Preprocessing and Model (9)

4.2.2.3 Training Model

ทำการ Training Model ที่ได้มาด้วย Epochs = 30

```python
# How many epochs to train for, we have found ~30 to be a good starting point
EPOCHS = 30

# Callback to save model checkpoints for future inspection or training
checkpointFileLoc = CHKPT_DIR + "/save_at_{epoch:02d}.h5"
modelCheckpointCallback = keras.callbacks.ModelCheckpoint(checkpointFileLoc)

# Compile the model!
model.compile(
    optimizer=keras.optimizers.Adam(1e-3),
    loss="binary_crossentropy",
    metrics=["accuracy"],
)

# Run training
history = model.fit(train_ds, epochs=EPOCHS, validation_data=validation_ds,
                    callbacks=[modelCheckpointCallback])
```

ภาพที่ 4.57 Training Model

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ทำการ Save Model

```
# save the model file
model.save(SAVED_MODEL_FILENAME)
```

4.2.1.4 Evaluate Optimize และ Convert Model

ทำการ Convert และ Optimize Model ด้วย TensorFlow lite

```
converter = tf.lite.TFLiteConverter.from_saved_model(SAVED_MODEL_FILENAME)
model_no_quant_tflite = converter.convert()

# Save the model to disk
open(FLOAT_TFL_MODEL_FILENAME, "wb").write(model_no_quant_tflite)

def representative_dataset():
  for filename in glob.glob(TEST_DIR + "/*/*.png"):
    img = keras.preprocessing.image.load_img(filename, target_size=(IMAGE_WIDTH, IMAGE_HEIGHT))
    img_array = keras.preprocessing.image.img_to_array(img)
    img_array = tf.expand_dims(img_array, 0)  # Create batch axis for images, labels in train_ds.take(1):
    yield([img_array])
# Set the optimization flag.
converter.optimizations = [tf.lite.Optimize.DEFAULT]
# Enforce integer only quantization
converter.target_spec.supported_ops = [tf.lite.OpsSet.TFLITE_BUILTINS_INT8]
converter.inference_input_type = tf.int8
converter.inference_output_type = tf.int8
# Provide a representative dataset to ensure we quantize correctly.
converter.representative_dataset = representative_dataset
model_tflite = converter.convert()

# Save the model to disk
open(QUANTIZED_TFL_MODEL_FILENAME, "wb").write(model_tflite)
```

ภาพที่ 4.58 Evaluate Optimize (1)

เปรียบเทียบขนาดก่อนและหลังผ่าน TensorFlow lite โดยมีขนาดลดลงไปกว่า 0.5 MB จาก 0.6 MB และลดลงไปอีกเมื่อผ่านการ Quantize

```
def get_dir_size(dir):
  size = 0
  for f in os.scandir(dir):
    if f.is_file():
      size += f.stat().st_size
    elif f.is_dir():
      size += get_dir_size(f.path)
  return size

# Calculate size
size_tf = get_dir_size(SAVED_MODEL_FILENAME)
size_no_quant_tflite = os.path.getsize(FLOAT_TFL_MODEL_FILENAME)
size_tflite = os.path.getsize(QUANTIZED_TFL_MODEL_FILENAME)

# Compare size
pd.DataFrame.from_records(
    [["TensorFlow", f"{size_tf} bytes", ""],
     ["TensorFlow Lite", f"{size_no_quant_tflite} bytes ", f"(reduced by {size_tf - size_no_quant_tflite} bytes)"],
     ["TensorFlow Lite Quantized", f"{size_tflite} bytes", f"(reduced by {size_no_quant_tflite - size_tflite} bytes)"]],
     columns = ["Model", "Size", ""], index="Model")
```

Model	Size	
TensorFlow	655959 bytes	
TensorFlow Lite	100428 bytes	(reduced by 555531 bytes)
TensorFlow Lite Quantized	31240 bytes	(reduced by 69188 bytes)

ภาพที่ 4.59 Evaluate Optimize (2)

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ทดสอบความแม่นยำของ Model ก่อนและหลังการ Quantize

```
run_tflite_test(model_no_quant_tflite)

Testing Gesture:  3  with datasize:  143
Testing Gesture:  8  with datasize:  77
Testing Gesture:  1  with datasize:  88
Testing Gesture:  2  with datasize:  121
Testing Gesture:  9  with datasize:  121
Testing Gesture:  4  with datasize:  99
Testing Gesture:  7  with datasize:  132
Testing Gesture:  5  with datasize:  66
Testing Gesture:  0  with datasize:  132
Testing Gesture:  6  with datasize:  121
100.0% correct (N=915, 185 unknown)

 test the quantized model

run_tflite_test(model_tflite)

Testing Gesture:  3  with datasize:  143
Testing Gesture:  8  with datasize:  77
Testing Gesture:  1  with datasize:  88
Testing Gesture:  2  with datasize:  121
Testing Gesture:  9  with datasize:  121
Testing Gesture:  4  with datasize:  99
Testing Gesture:  7  with datasize:  132
Testing Gesture:  5  with datasize:  66
Testing Gesture:  0  with datasize:  132
Testing Gesture:  6  with datasize:  121
100.0% correct (N=924, 176 unknown)
```

ภาพที่ 4.60 Evaluate Optimize (3)

แปลง Model ให้เป็นรูปแบบที่สามารถนำไปใช้บน Microcontroller ได้ด้วย TensorFlow lite Micro โดยทำการแปลงให้สามารถใช้กับ C source file

```
# Convert to a C source file, i.e, a TensorFlow Lite for Microcontrollers model
!xxd -i {QUANTIZED_TFL_MODEL_FILENAME} > {TFL_CC_MODEL_FILENAME}
# Update variable names
REPLACE_TEXT = QUANTIZED_TFL_MODEL_FILENAME.replace('/', '_').replace('.', '_')
!sed -i 's/'{REPLACE_TEXT}'/g_magic_wand_model_data/g' {TFL_CC_MODEL_FILENAME}
```

ภาพที่ 4.61 Convert Model

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

4.2.1.5 Set up Code

นำโมเดลที่แปลงรูปแบบแล้วมาใส่ในไฟล์ของ Arduino

```
/* Copyright 2020 The TensorFlow Authors. All Rights Reserved.
Licensed under the Apache License, Version 2.0 (the "License");
you may not use this file except in compliance with the License.
You may obtain a copy of the License at
    http://www.apache.org/licenses/LICENSE-2.0
Unless required by applicable law or agreed to in writing, software
distributed under the License is distributed on an "AS IS" BASIS,
WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
See the License for the specific language governing permissions and
limitations under the License.
==============================================================================*/

unsigned char g_magic_wand_model_data[] = {
  0x1c, 0x00, 0x00, 0x00, 0x54, 0x46, 0x4c, 0x33, 0x14, 0x00, 0x20, 0x00,
  0x1c, 0x00, 0x18, 0x00, 0x14, 0x00, 0x10, 0x00, 0x0c, 0x00, 0x00, 0x00,
  0x08, 0x00, 0x04, 0x00, 0x14, 0x00, 0x00, 0x00, 0x1c, 0x00, 0x00, 0x00,
  0x90, 0x00, 0x00, 0x00, 0xc0, 0x00, 0x00, 0x00, 0xd0, 0x61, 0x00, 0x00,
  0xe0, 0x61, 0x00, 0x00, 0x54, 0x79, 0x00, 0x00, 0x03, 0x00, 0x00, 0x00,
  0x01, 0x00, 0x00, 0x00, 0x04, 0x00, 0x00, 0x00, 0x38, 0x87, 0xff, 0xff,
  0x10, 0x00, 0x00, 0x00, 0x18, 0x00, 0x00, 0x00, 0x28, 0x00, 0x00, 0x00,
  0x44, 0x00, 0x00, 0x00, 0x05, 0x00, 0x00, 0x00, 0x73, 0x65, 0x72, 0x76,
  0x65, 0x00, 0x00, 0x00, 0x0f, 0x00, 0x00, 0x00, 0x73, 0x65, 0x72, 0x76,
  0x69, 0x6e, 0x67, 0x5f, 0x64, 0x65, 0x66, 0x61, 0x75, 0x6c, 0x74, 0x00,
  0x01, 0x00, 0x00, 0x00, 0x04, 0x00, 0x00, 0x00, 0xc4, 0xff, 0xff, 0xff,
  0x0f, 0x00, 0x00, 0x00, 0x04, 0x00, 0x00, 0x00, 0x07, 0x00, 0x00, 0x00,
  0x64, 0x65, 0x6e, 0x73, 0x65, 0x5f, 0x31, 0x00, 0x01, 0x00, 0x00, 0x00,
  0x04, 0x00, 0x00, 0x00, 0x1e, 0x9e, 0xff, 0xff, 0x04, 0x00, 0x00, 0x00,
  0x07, 0x00, 0x00, 0x00, 0x69, 0x6e, 0x70, 0x75, 0x74, 0x5f, 0x32, 0x00,
  0x01, 0x00, 0x00, 0x00, 0x0c, 0x00, 0x00, 0x00, 0x08, 0x00, 0x0c, 0x00,
  0x08, 0x00, 0x04, 0x00, 0x08, 0x00, 0x00, 0x00, 0x11, 0x00, 0x00, 0x00,
  0x04, 0x00, 0x00, 0x00, 0x13, 0x00, 0x00, 0x00, 0x6d, 0x69, 0x6e, 0x5f,
  0x72, 0x75, 0x6e, 0x74, 0x69, 0x6d, 0x65, 0x5f, 0x76, 0x65, 0x72, 0x73,
  0x69, 0x6f, 0x6e, 0x00, 0x12, 0x00, 0x00, 0x00, 0x0c, 0x61, 0x00, 0x00,
  0x04, 0x61, 0x00, 0x00, 0xcc, 0x60, 0x00, 0x00, 0xb4, 0x60, 0x00, 0x00,
  0x64, 0x60, 0x00, 0x00, 0xd4, 0x5f, 0x00, 0x00, 0xc4, 0x5e, 0x00, 0x00,
  0x34, 0x5c, 0x00, 0x00, 0x24, 0x4a, 0x00, 0x00, 0x14, 0x02, 0x00, 0x00,
  0x54, 0x00, 0x00, 0x00, 0x4c, 0x00, 0x00, 0x00, 0x44, 0x00, 0x00, 0x00,
  0x3c, 0x00, 0x00, 0x00, 0x34, 0x00, 0x00, 0x00, 0x2c, 0x00, 0x00, 0x00,
  0x24, 0x00, 0x00, 0x00, 0x04, 0x00, 0x00, 0x00, 0xb2, 0x9e, 0xff, 0xff,
  0x04, 0x00, 0x00, 0x00, 0x10, 0x00, 0x00, 0x00, 0x31, 0x2e, 0x36, 0x2e,
  0x30, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
  0x20, 0x89, 0xff, 0xff, 0x24, 0x89, 0xff, 0xff, 0x28, 0x89, 0xff, 0xff,
  0x2c, 0x89, 0xff, 0xff, 0x30, 0x89, 0xff, 0xff, 0x34, 0x89, 0xff, 0xff,
  0xe6, 0x9e, 0xff, 0xff, 0x04, 0x00, 0x00, 0x00, 0xb0, 0x01, 0x00, 0x00,
  0xd9, 0xfc, 0x10, 0x95, 0x28, 0xe9, 0xec, 0x1d, 0xfa, 0xef, 0xa5, 0xbc,
  0xbf, 0x90, 0xf3, 0x46, 0x08, 0xe7, 0x81, 0x2a, 0xfa, 0xe3, 0xec, 0xdf,
  0x32, 0xfd, 0xfd, 0x13, 0x37, 0x9d, 0x32, 0x1a, 0xe4, 0x4b, 0xf4, 0x05,
  0x0a, 0x49, 0x81, 0x2b, 0x36, 0xe4, 0x5e, 0x05, 0xff, 0x04, 0x42, 0x95,
  0x19, 0x29, 0xfa, 0x36, 0x07, 0xfa, 0xa6, 0xf9, 0xfa, 0xe0, 0x0e, 0xf3,
  0xf8, 0x08, 0xfa, 0xb6, 0xea, 0xfe, 0xc8, 0x35, 0x81, 0xe1, 0x19, 0xbf,
  0x12, 0xda, 0x00, 0x05, 0xfa, 0xff, 0xf3, 0x0b, 0xeb, 0xc8, 0x32, 0x22,
  0x30, 0xa9, 0x92, 0xfb, 0x10, 0x0a, 0xe4, 0x29, 0x02, 0x4b, 0xa0, 0x81,
```

ภาพที่ 4.62 Set up Code (1)

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ทำการ Include library และกำหนดตัวแปรต่าง ๆ

```
#include <TensorFlowLite.h>

#include "tensorflow/lite/micro/micro_error_reporter.h"
#include "tensorflow/lite/micro/micro_interpreter.h"
#include "tensorflow/lite/micro/micro_mutable_op_resolver.h"
#include "tensorflow/lite/schema/schema_generated.h"
#include "tensorflow/lite/version.h"

#include "magic_wand_model_data.h"
#include "rasterize_stroke.h"
#include "imu_provider.h"

#define BLE_SENSE_UUID(val) ("4798e0f2-" val "-4d68-af64-8a8f5258404e")

namespace {

  const int VERSION = 0x00000000;

  // Constants for image rasterization
  constexpr int raster_width = 32;
  constexpr int raster_height = 32;
  constexpr int raster_channels = 3;
  constexpr int raster_byte_count = raster_height * raster_width * raster_channels;
  int8_t raster_buffer[raster_byte_count];

  // BLE settings
  BLEService        service                   (BLE_SENSE_UUID("0000"));
  BLECharacteristic strokeCharacteristic      (BLE_SENSE_UUID("300a"), BLERead, stroke_struct_byte_count);

  // String to calculate the local and device name
  String name;

  // Create an area of memory to use for input, output, and intermediate arrays.
  // The size of this will depend on the model you're using, and may need to be
  // determined by experimentation.
  constexpr int kTensorArenaSize = 30 * 1024;
  uint8_t tensor_arena[kTensorArenaSize];

  tflite::ErrorReporter* error_reporter = nullptr;
  const tflite::Model* model = nullptr;
  tflite::MicroInterpreter* interpreter = nullptr;

  // ------------------------------------------------------------------------- //
  // UPDATE THESE VARIABLES TO MATCH THE NUMBER AND LIST OF GESTURES IN YOUR DATASET  //
  // ------------------------------------------------------------------------- //
  constexpr int label_count = 10;
  const char* labels[label_count] = {"0", "1", "2", "3", "4", "5", "6", "7", "8", "9"};
```

ภาพที่ 4.63 Set up Code (2)

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ใน void setup() ทำการตั้งค่าเริ่มต้น IMU

```
void setup() {
  // Start serial
  Serial.begin(9600);
  Serial.println("Started");

  // Start IMU
  if (!IMU.begin()) {
    Serial.println("Failed to initialized IMU!");
    while (1);
  }
  SetupIMU();

  // Start BLE
  if (!BLE.begin()) {
    Serial.println("Failed to initialized BLE!");
    while (1);
  }
  String address = BLE.address();

  // Output BLE settings over Serial
  Serial.print("address = ");
  Serial.println(address);

  address.toUpperCase();

  name = "BLESense-";
  name += address[address.length() - 5];
  name += address[address.length() - 4];
  name += address[address.length() - 2];
  name += address[address.length() - 1];

  Serial.print("name = ");
  Serial.println(name);

  BLE.setLocalName(name.c_str());
  BLE.setDeviceName(name.c_str());
  BLE.setAdvertisedService(service);

  service.addCharacteristic(strokeCharacteristic);

  BLE.addService(service);
  BLE.advertise();
```

ภาพที่ 4.64 Set up Code (3)

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ทำการโหลด Model

```
// Set up logging. Google style is to avoid globals or statics because of
// lifetime uncertainty, but since this has a trivial destructor it's okay.
static tflite::MicroErrorReporter micro_error_reporter;  // NOLINT
error_reporter = &micro_error_reporter;

// Map the model into a usable data structure. This doesn't involve any
// copying or parsing, it's a very lightweight operation.
model = tflite::GetModel(g_magic_wand_model_data);
if (model->version() != TFLITE_SCHEMA_VERSION) {
  TF_LITE_REPORT_ERROR(error_reporter,
                       "Model provided is schema version %d not equal "
                       "to supported version %d.",
                       model->version(), TFLITE_SCHEMA_VERSION);
  return;
}
```

ภาพที่ 4.65 Set up Code (4)

กำหนด Operator ที่ใช้ โดยส่วนนี้เป็นส่วนหนึ่งของ TensorFlow lite Micro

```
static tflite::MicroMutableOpResolver<4> micro_op_resolver;  // NOLINT
micro_op_resolver.AddConv2D();
micro_op_resolver.AddMean();
micro_op_resolver.AddFullyConnected();
micro_op_resolver.AddSoftmax();
```

ภาพที่ 4.66 Set up Code (5)

กำหนด Interpreter ที่ใช้

```
// Build an interpreter to run the model with.
static tflite::MicroInterpreter static_interpreter(
    model, micro_op_resolver, tensor_arena, kTensorArenaSize, error_reporter);
interpreter = &static_interpreter;

// Allocate memory from the tensor_arena for the model's tensors.
interpreter->AllocateTensors();
```

ภาพที่ 4.67 Set up Code (6)

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

Set up Model Input และ Output

```
// Set model input settings
TfLiteTensor* model_input = interpreter->input(0);
if ((model_input->dims->size != 4) || (model_input->dims->data[0] != 1) ||
    (model_input->dims->data[1] != raster_height) ||
    (model_input->dims->data[2] != raster_width) ||
    (model_input->dims->data[3] != raster_channels) ||
    (model_input->type != kTfLiteInt8)) {
  TF_LITE_REPORT_ERROR(error_reporter,
                       "Bad input tensor parameters in model");
  return;
}

// Set model output settings
TfLiteTensor* model_output = interpreter->output(0);
if ((model_output->dims->size != 2) || (model_output->dims->data[0] != 1) ||
    (model_output->dims->data[1] != label_count) ||
    (model_output->type != kTfLiteInt8)) {
  TF_LITE_REPORT_ERROR(error_reporter,
                       "Bad output tensor parameters in model");
  return;
}


// make sure IMU data is available then read in data
const bool data_available = IMU.accelerationAvailable() || IMU.gyroscopeAvailable();
if (!data_available) {
  return;
}
int accelerometer_samples_read;
int gyroscope_samples_read;
ReadAccelerometerAndGyroscope(&accelerometer_samples_read, &gyroscope_samples_read);
```

ภาพที่ 4.68 Set up Code (7)

ประมวลผลที่ได้จาก IMU และแปลงให้เป็นสองมิติ

```
// Parse and process IMU data
bool done_just_triggered = false;
if (gyroscope_samples_read > 0) {
  EstimateGyroscopeDrift(current_gyroscope_drift);
  UpdateOrientation(gyroscope_samples_read, current_gravity, current_gyroscope_drift);
  UpdateStroke(gyroscope_samples_read, &done_just_triggered);
  if (central && central.connected()) {
    strokeCharacteristic.writeValue(stroke_struct_buffer, stroke_struct_byte_count);
  }
}
if (accelerometer_samples_read > 0) {
  EstimateGravityDirection(current_gravity);
  UpdateVelocity(accelerometer_samples_read, current_gravity);
}
```

ภาพที่ 4.69 Set up Code (8)

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

นำผลที่ได้มาผ่านฟังก์ชัน rasterize เพื่อลบรอยหยักบางส่วน

```
// Wait for a gesture to be done
if (done_just_triggered) {
  // Rasterize the gesture
  RasterizeStroke(stroke_points, *stroke_transmit_length, 0.6f, 0.6f, raster_width, raster_height, raster_buffer);
  for (int y = 0; y < raster_height; ++y) {
    char line[raster_width + 1];
    for (int x = 0; x < raster_width; ++x) {
      const int8_t* pixel = &raster_buffer[(y * raster_width * raster_channels) + (x * raster_channels)];
      const int8_t red = pixel[0];
      const int8_t green = pixel[1];
      const int8_t blue = pixel[2];
      char output;
      if ((red > -128) || (green > -128) || (blue > -128)) {
        output = '#';
      } else {
        output = '.';
      }
      line[x] = output;
    }
    line[raster_width] = 0;
    Serial.println(line);
  }
```

ภาพที่ 4.70 Set up Code (9)

นำข้อมูลที่ได้มาผ่าน Convolution Neural Network ที่ทำไว้

```
  // Pass to the model and run the interpreter
  TfLiteTensor* model_input = interpreter->input(0);
  for (int i = 0; i < raster_byte_count; ++i) {
    model_input->data.int8[i] = raster_buffer[i];
  }
  TfLiteStatus invoke_status = interpreter->Invoke();
  if (invoke_status != kTfLiteOk) {
    TF_LITE_REPORT_ERROR(error_reporter, "Invoke failed");
    return;
  }
  TfLiteTensor* output = interpreter->output(0);
```

ภาพที่ 4.71 Set up Code (10)

นำผลที่ได้มาแสดงผล

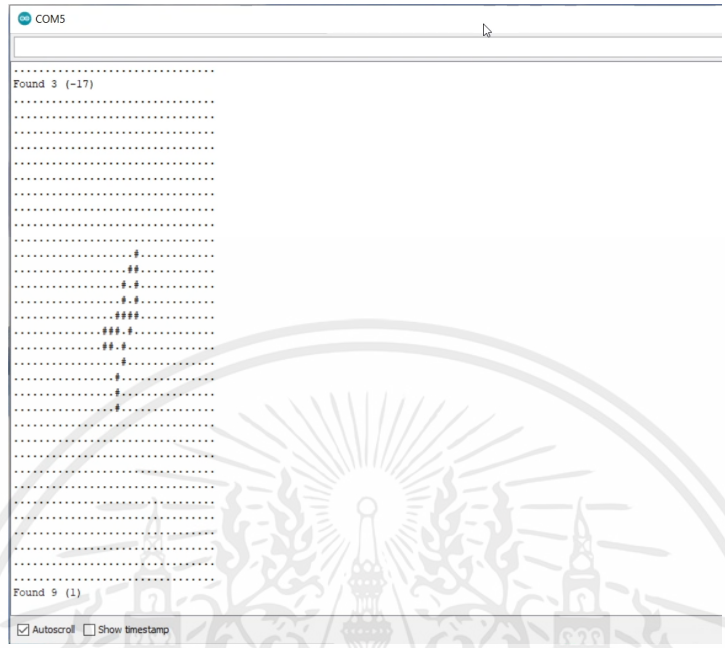
```
  // Parse the model output
  int8_t max_score;
  int max_index;
  for (int i = 0; i < label_count; ++i) {
    const int8_t score = output->data.int8[i];
    if ((i == 0) || (score > max_score)) {
      max_score = score;
      max_index = i;
    }
  }
  TF_LITE_REPORT_ERROR(error_reporter, "Found %s (%d)", labels[max_index], max_score);
}
}
```

ภาพที่ 4.72 Set up Code (11)

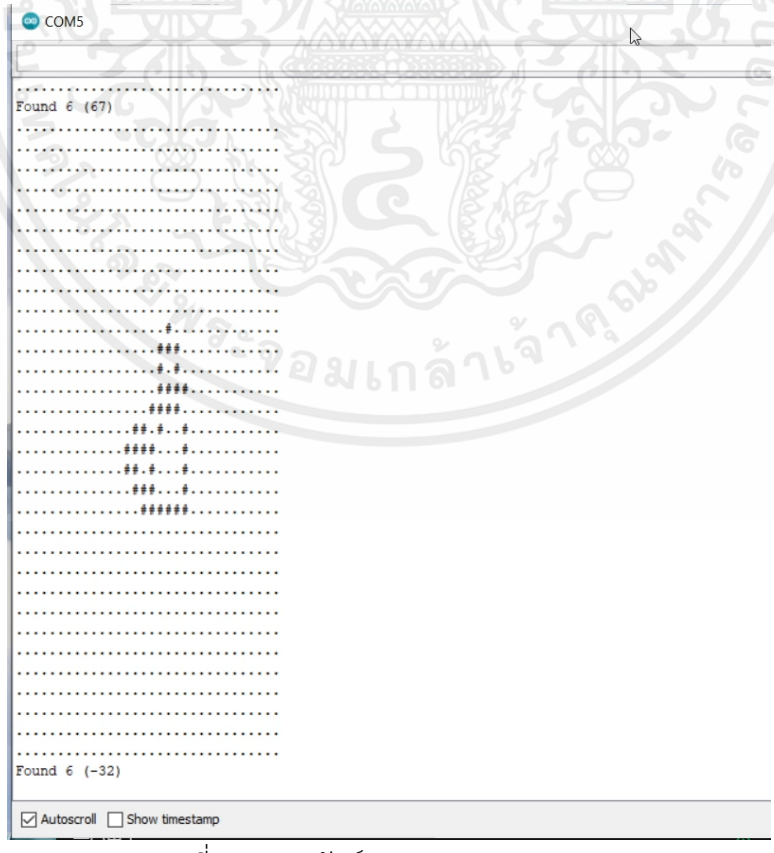
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

4.2.4 ผลการทดสอบคาดเดาท่าทางการขยับด้วยการเรียนรู้ของเครื่องด้วยหน่วยตรวจวัดการเคลื่อนไหวภายใน

เมื่อทำการขยับบอร์ดที่มี IMU โปรแกรมจะทำการคาดเดาตัวเลขที่วาด

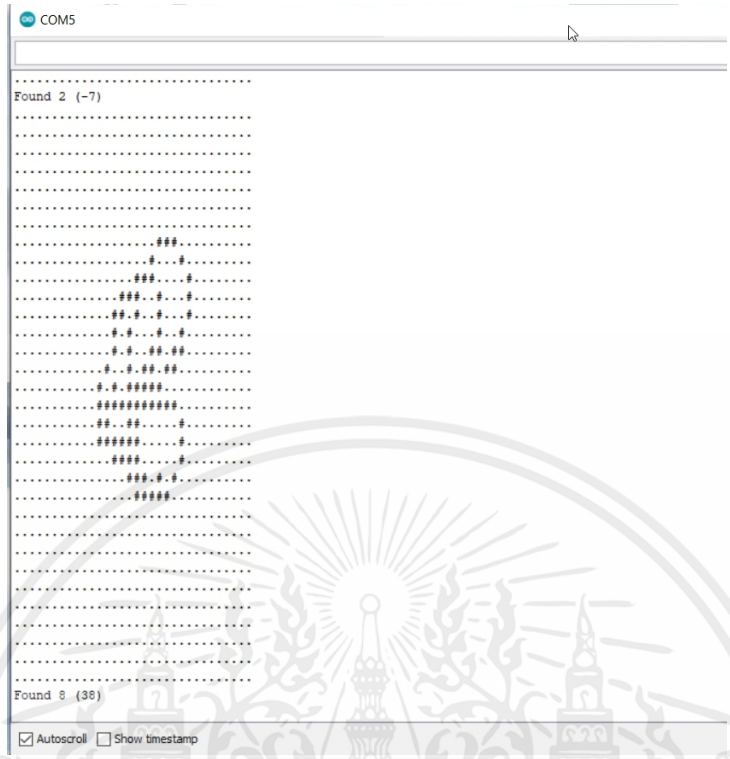


ภาพที่ 4.73 ผลลัพธ์จากการทดสอบ IMU (1)



ภาพที่ 4.74 ผลลัพธ์จากการทดสอบ IMU (2)

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



ภาพที่ 4.75 ผลลัพธ์จากการทดสอบ IMU (3)

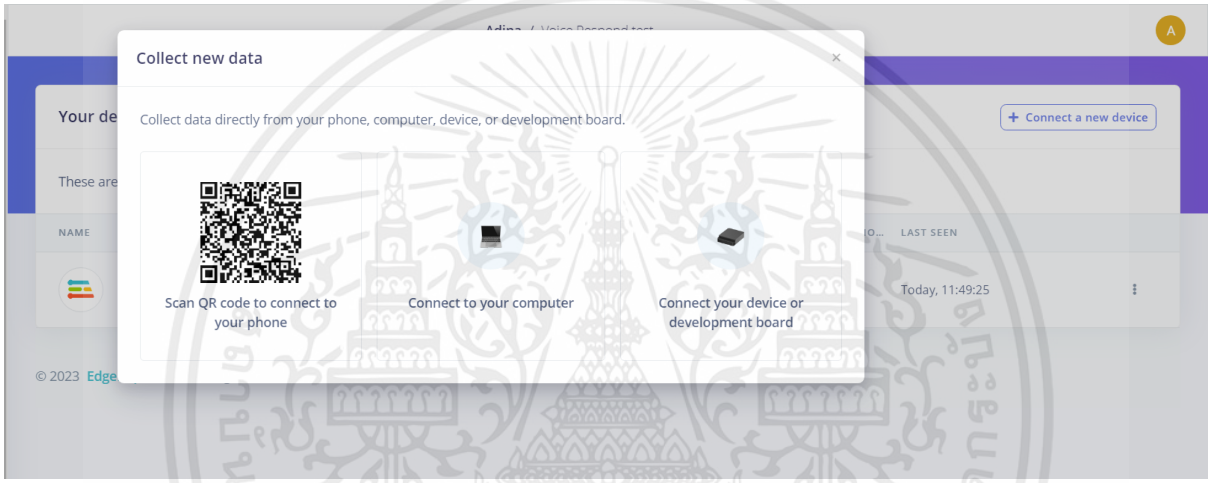
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

4.3 การทดลองใช้งาน Edge Impulse

4.3.1 ขั้นตอนการทดสอบชี้เฉพาะคำสำคัญด้วยการเรียนรู้ของเครื่องบนไมโครคอนโทรลเลอร์โดย edge impulse

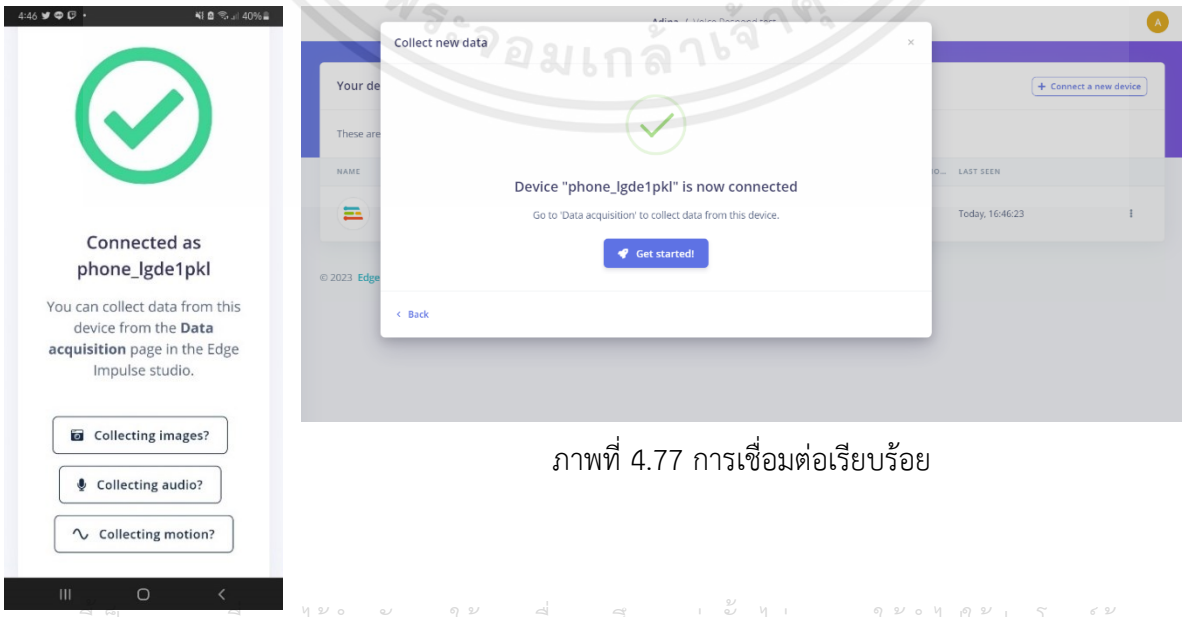
4.3.1.1 Collecting Data

หลังจากทำการสมัคร edge impulse account และสร้าง project ใหม่เรียบร้อยแล้ว ทำการเชื่อมต่อ edge impulse กับ web browser บนโทรศัพท์มือถือ โดยการสแกน QR code ที่ edge impulse สร้างมาให้



ภาพที่ 4.76 การเชื่อมต่อกับอุปกรณ์

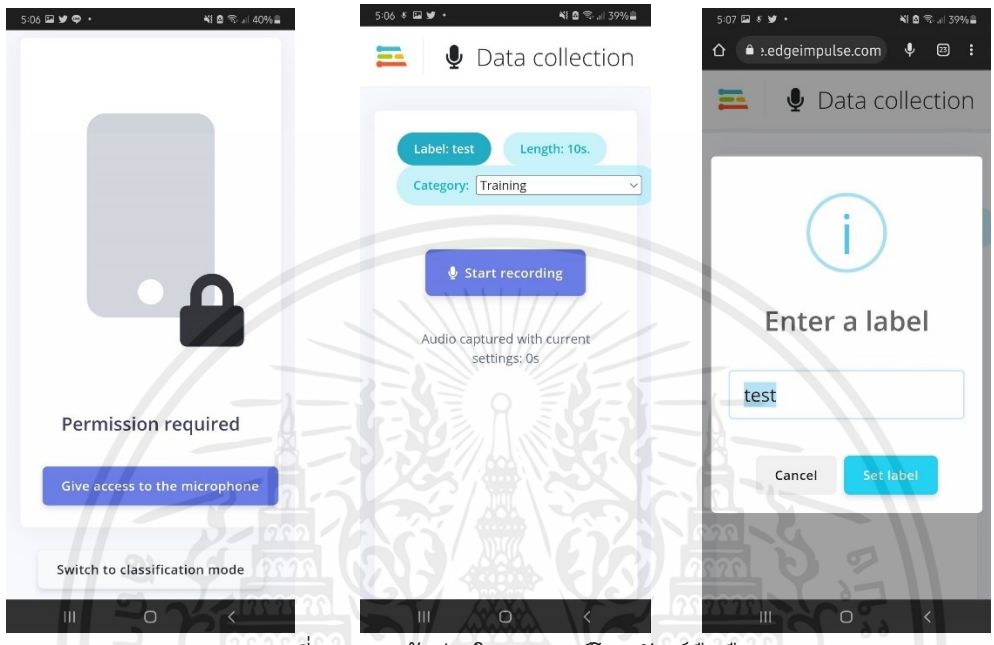
เมื่อทำการสแกนด้วยโทรศัพท์แล้วจะขึ้นอุปกรณ์ใหม่ในหน้า Devices ให้ทำการกด Collecting audio เพื่อเก็บข้อมูลจากไมค์ของโทรศัพท์มือถือ



ภาพที่ 4.77 การเชื่อมต่อเรียบร้อย

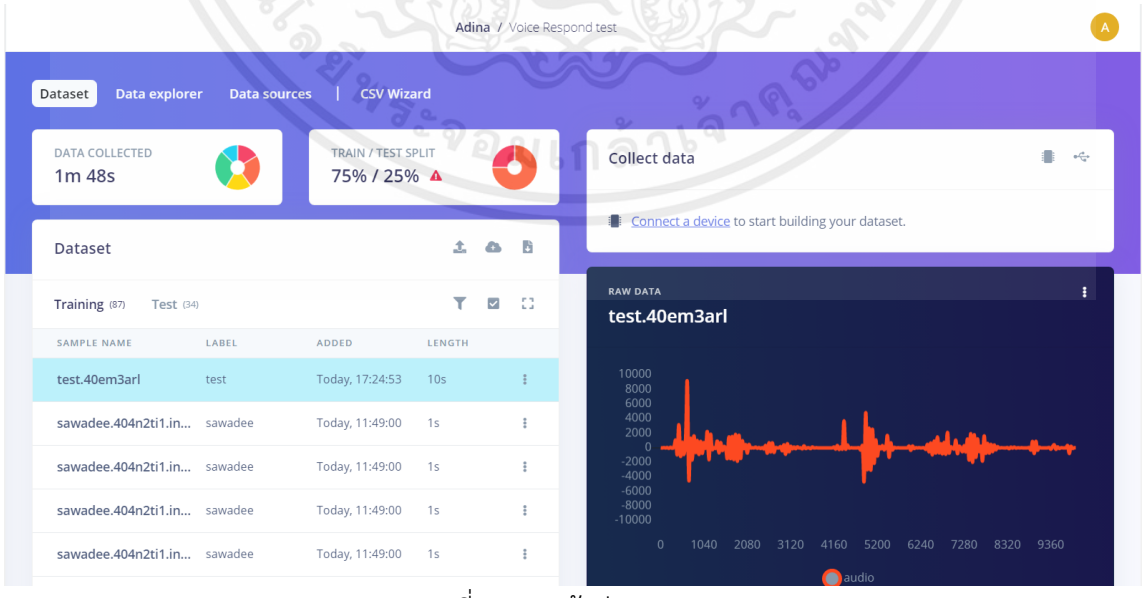
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ทำการให้ Permission ใช้ไมค์ จะเข้าสู่หน้าการอัดเสียงเพื่อเก็บข้อมูล ทำการเปลี่ยน label ให้เป็นชื่อที่เราต้องการบันทึกข้อมูล โดยสามารถกดที่เมนู length เพื่อกำหนดระยะเวลาในการอัดเสียงแต่ละครั้งได้ แล้วทำการ Start Recording label ละ 10 วินาที ทำการพูดคำที่ต้องการต่อเนื่องเป็นเวลา 10 วินาที



ภาพที่ 4.78 หน้าต่างในอุปกรณ์โทรศัพท์มือถือ

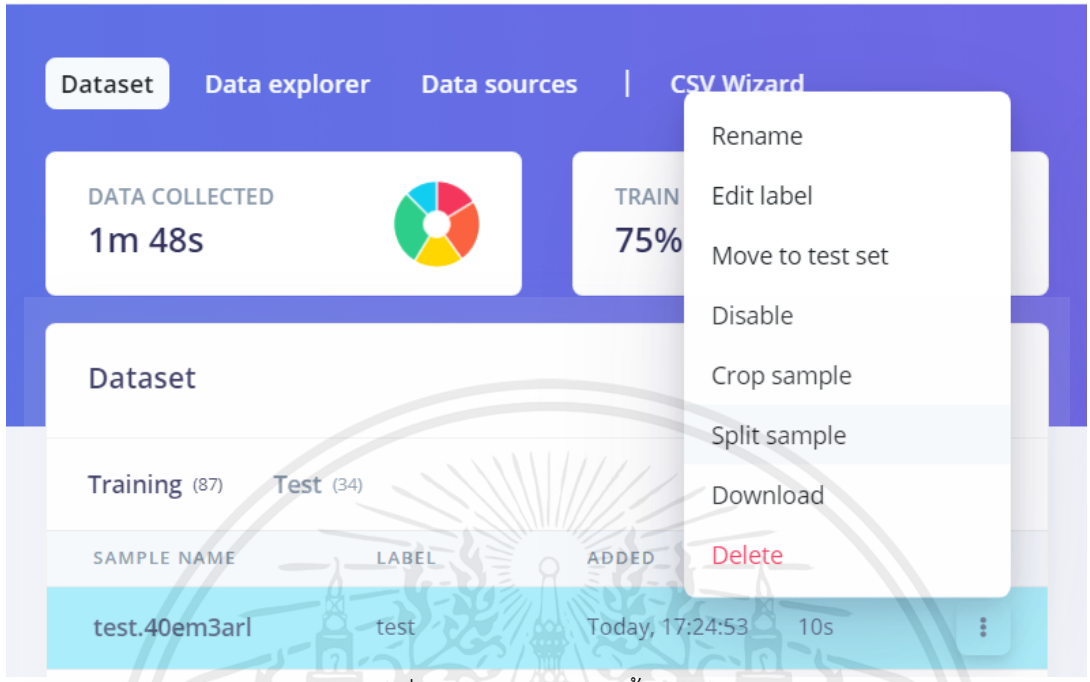
ในหน้า Data acquisition จะขึ้นข้อมูลเสียงที่ได้ทำการอัดไว้ ทำการกดที่ kebab menu



ภาพที่ 4.79 หน้าต่าง dataset

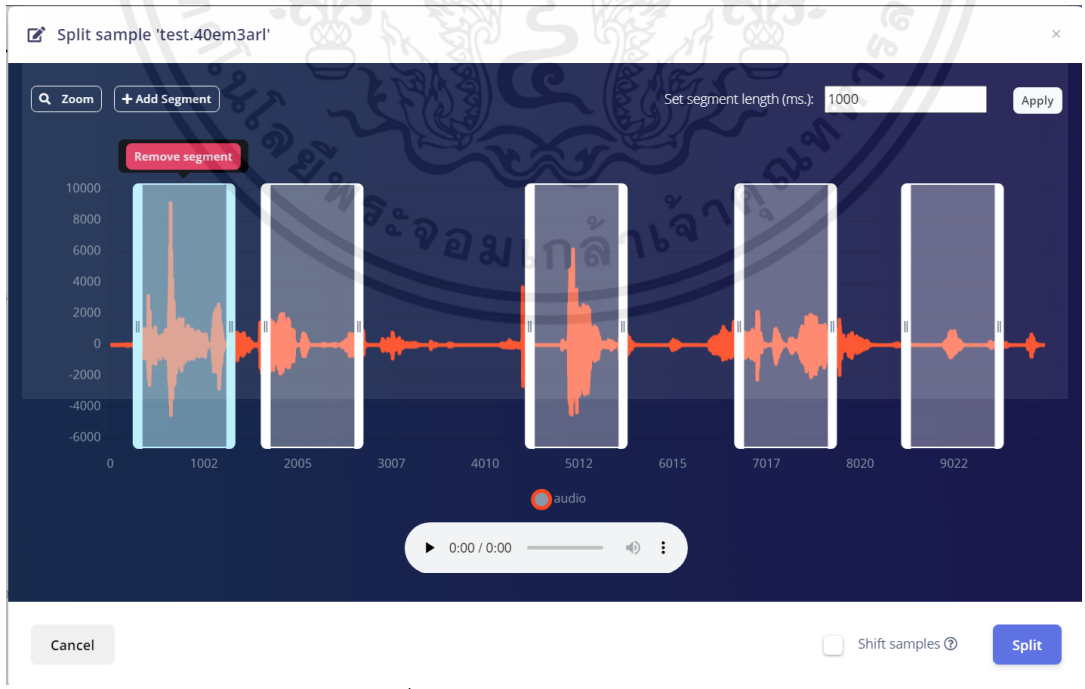
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ทำการกด Spilt sample เพื่อแยกเสียงที่ต้องการออกมาเป็นข้อมูลย่อย ๆ อีกที



ภาพที่ 4.80 หน้าต่างเมนูตั้งค่าข้อมูล

ทำการลาก Segment ให้ตรงกับเสียงที่ต้องการใช้และทำการกด Add Segment เมื่อต้องการเพิ่ม Segment เมื่อเสร็จแล้วให้ทำการกด Split



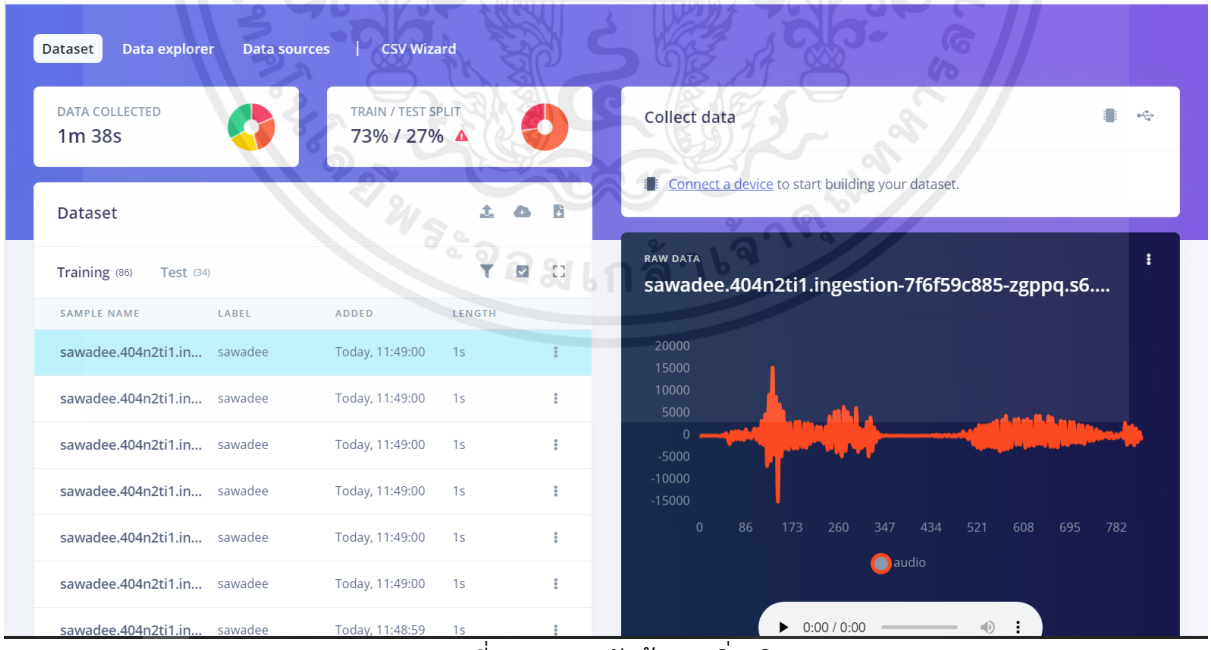
ภาพที่ 4.81 หน้าต่าง Split sample

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

จะทำให้ได้ข้อมูลย่อย ๆ ออกมาตามจำนวน Segment ที่ได้ใส่ลงไปอย่างละ 1 วินาที

ภาพที่ 4.82 ข้อมูลหลังการ Split sample

จากนั้นทำการอัดข้อมูลเสียงประมาณ 30-40 ตัวอย่าง แล้วจึงอัดข้อมูลเสียงเพิ่มอีกสอง label เพื่อทำให้สามารถจับได้หลายเสียง

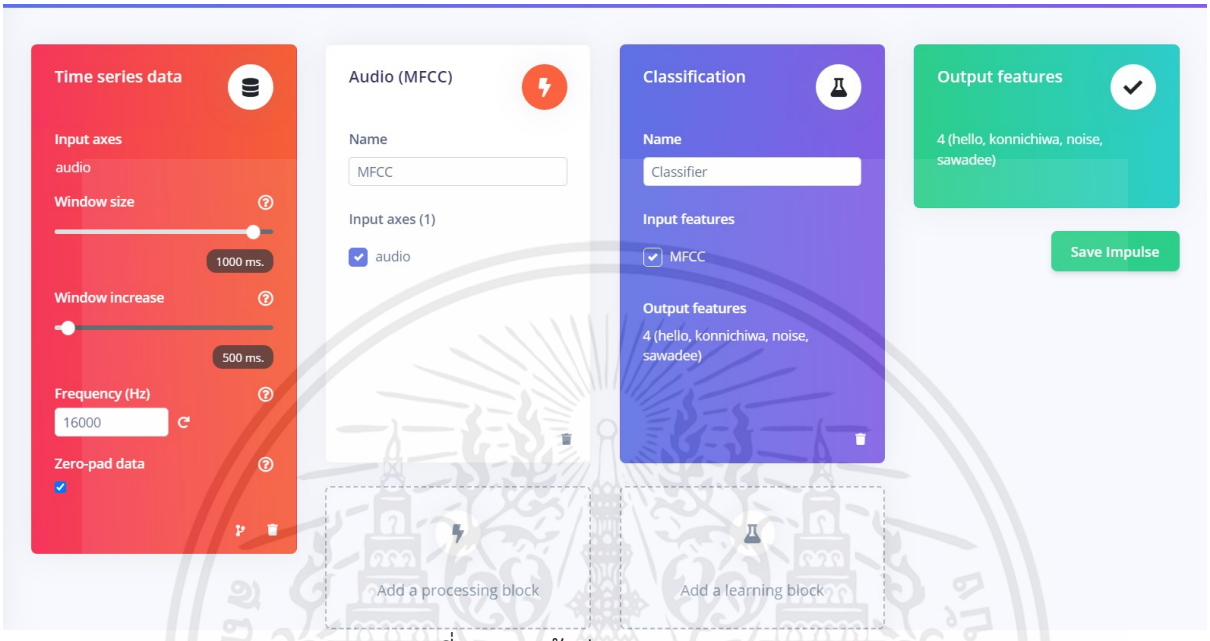


ภาพที่ 4.83 การอัดข้อมูลเพิ่มเติม

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

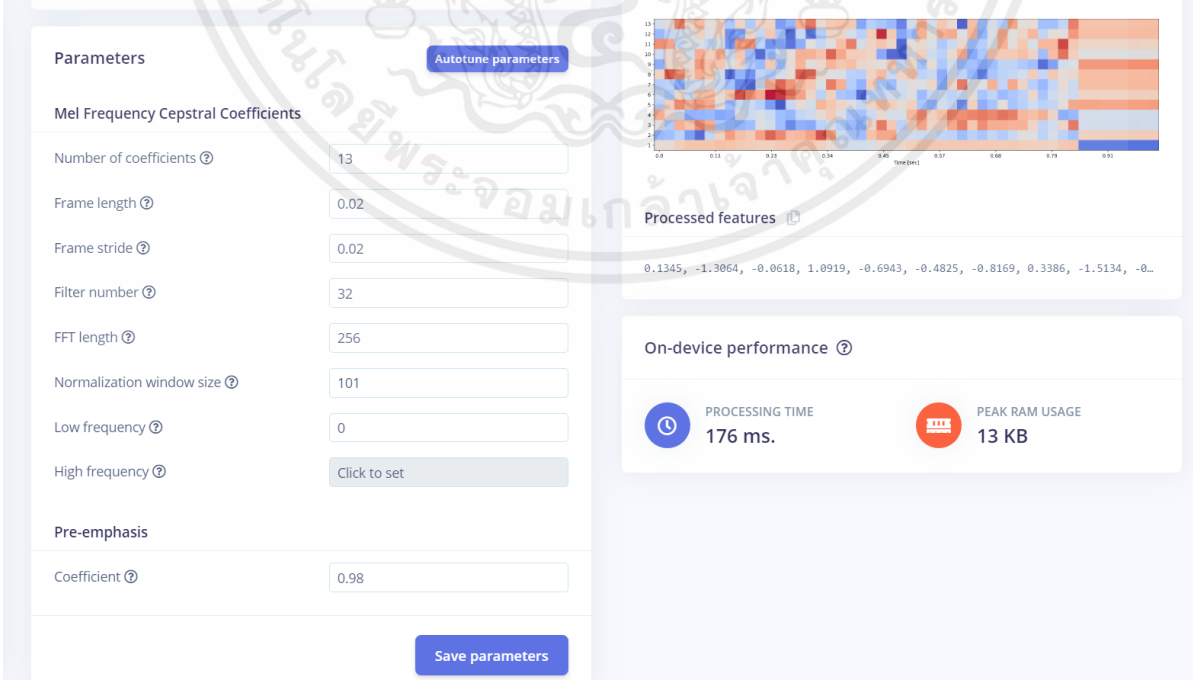
4.3.1.2 Training Model

เข้ามาในเมนู Create impulse ใน Add a processing block ทำการเลือก Audio(MFCC) และในเมนู Add a learning block เลือก Classification แล้วจึงกด Save impulse



ภาพที่ 4.84 หน้าต่าง Create impulse

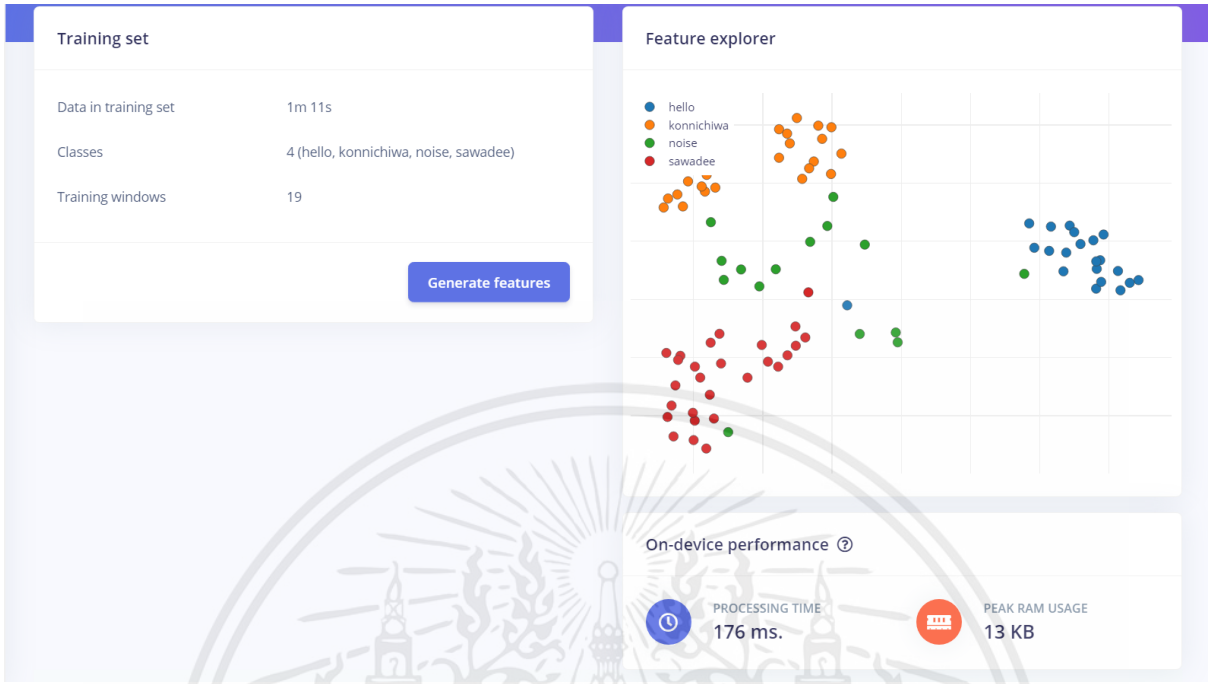
เข้ามาในเมนู MFCC แล้วทำการกด Save parameters



ภาพที่ 4.85 หน้าต่าง MFCC

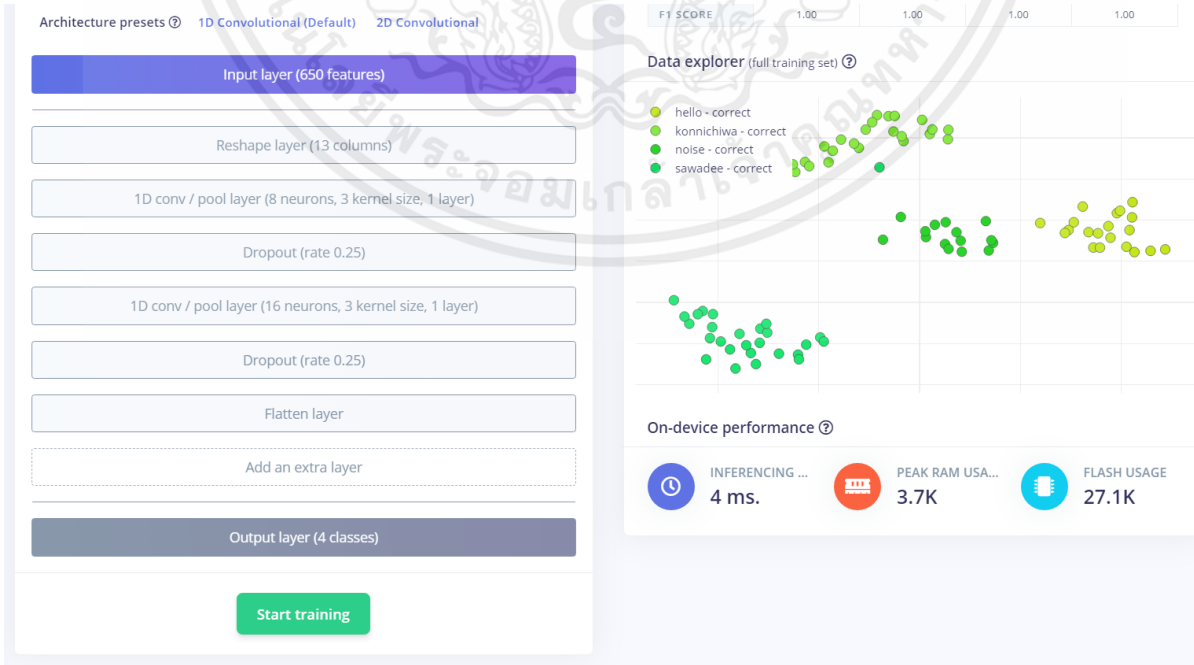
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

เมื่อกด Save parameters หน้าต่างจะเปลี่ยนเป็นหน้าต่างเมนู Generate features ทำการกด Generate features



ภาพที่ 4.86 หน้าต่าง Generate features

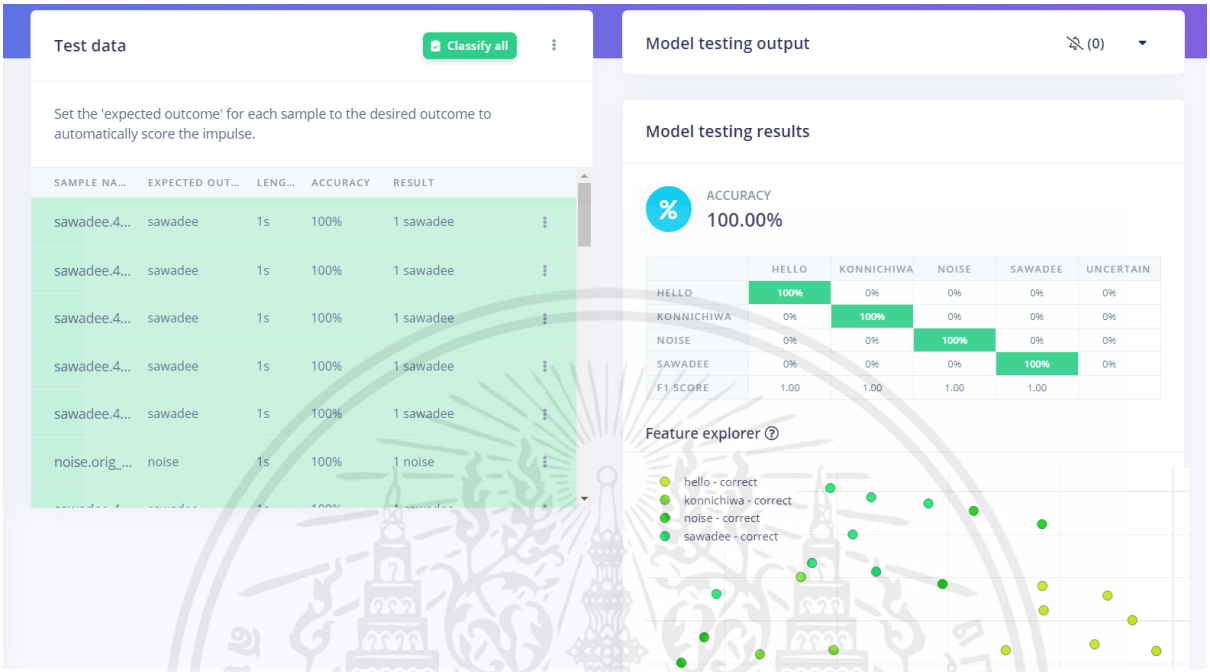
เข้ามาในเมนู Classifier ทำการกด Start training เมื่อ training เสร็จแล้วจะแสดงหน้าต่างผลลัพธ์ขึ้นมา



ภาพที่ 4.87 หน้าต่าง Classifier

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

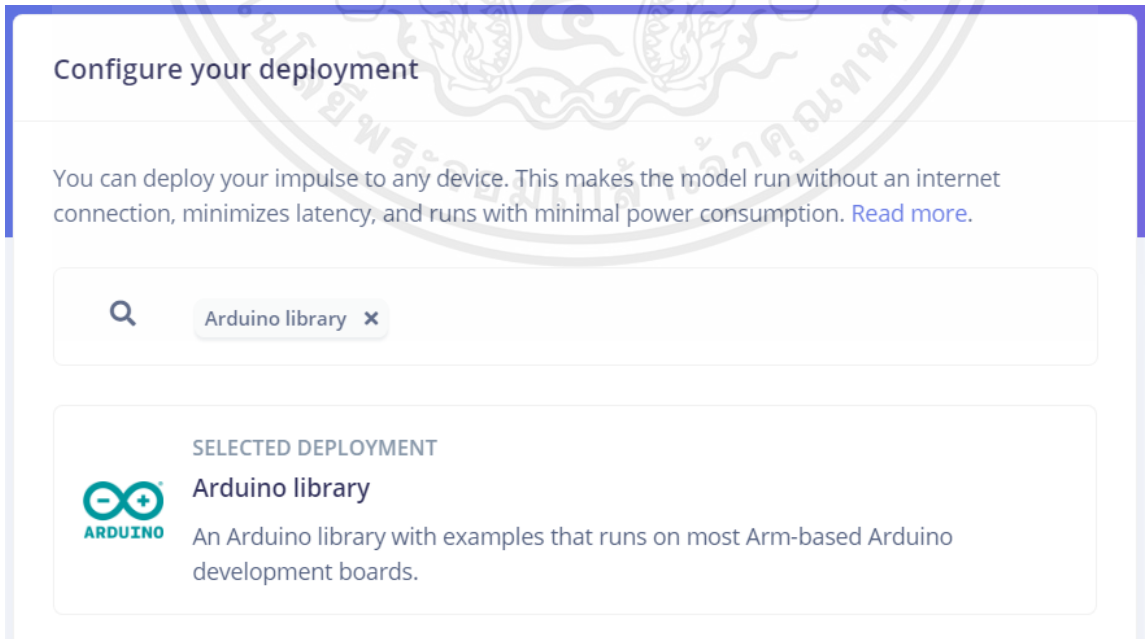
โดยจะสามารถทำการทดสอบ Model ได้ด้วยการเลือกเมนู Model testing โดยสามารถทำการ Perform train / test split ได้ในหน้า Dashboard เพื่อนำข้อมูลใน dataset ส่วนหนึ่งไปใช้เป็นข้อมูลสำหรับการทดสอบหรือจะทำการเก็บข้อมูลแยกก็ได้เช่นกัน



ภาพที่ 4.88 หน้าต่าง Model testing

4.3.1.3 Deploy Code

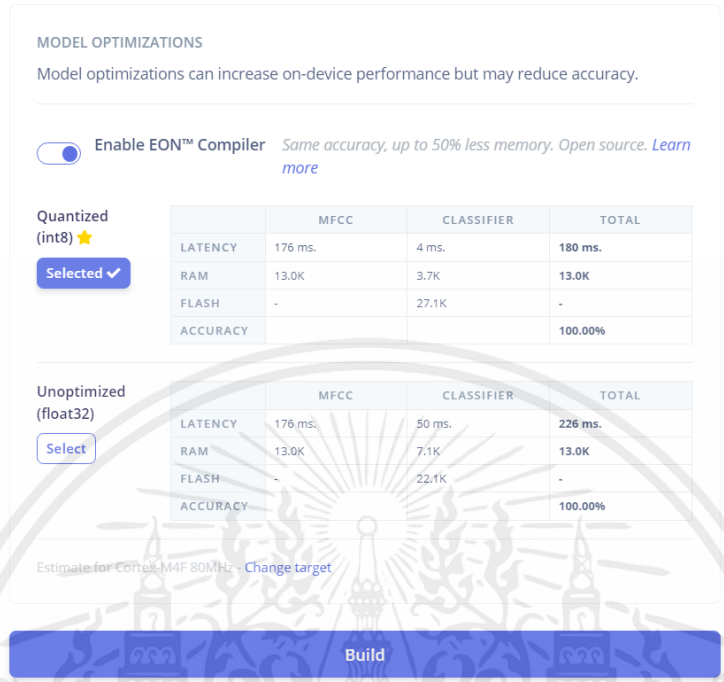
เข้าในเมนู Deployment ทำการค้นหาและเลือก Arduino library



ภาพที่ 4.89 เลือกอุปกรณ์ที่ต้องการ Deployment

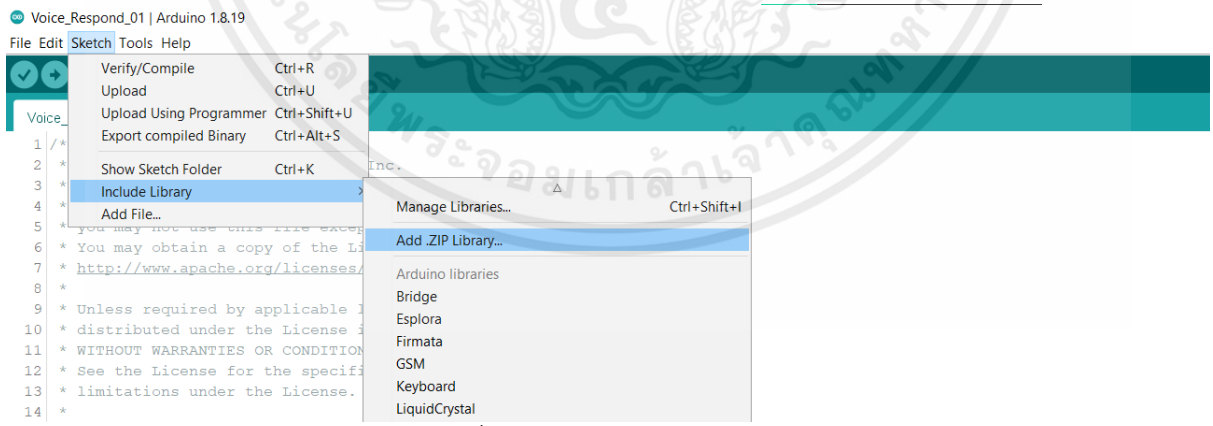
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

เมื่อเลือก Arduino library แล้ว ทำการกด build เพื่อสร้าง file zip ของ library และทำการ download



ภาพที่ 4.90 Build library

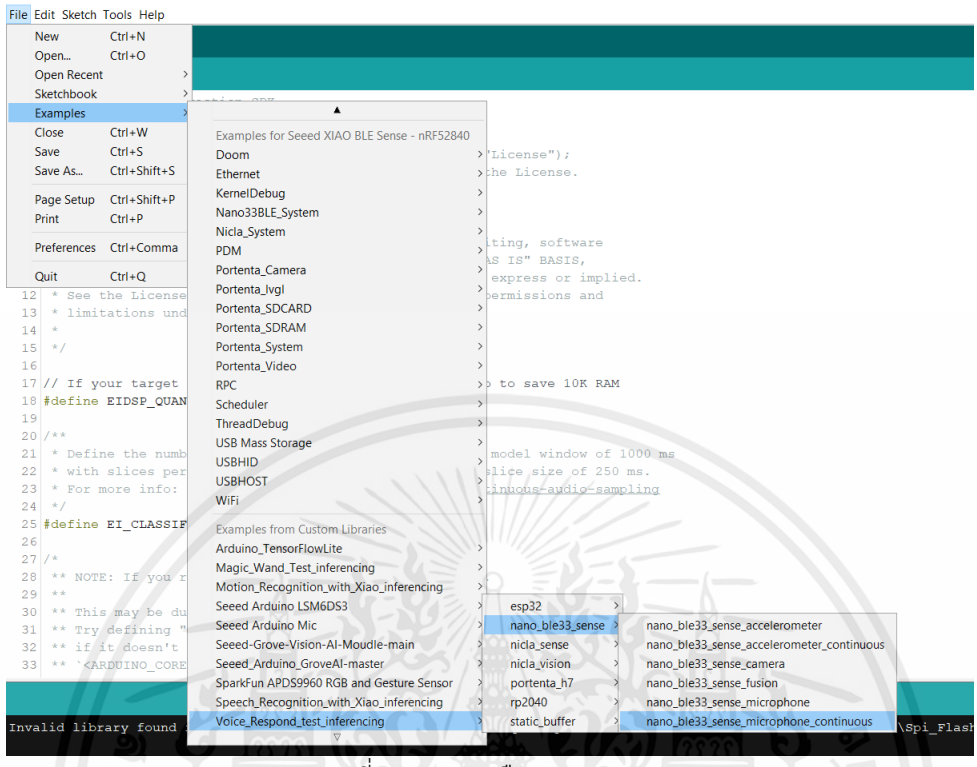
ในโปรแกรม Arduino ทำการเลือกเมนู Sketch และทำการ Include Library ด้วยการ Add .ZIP Library แล้วเลือกไฟล์ที่โหลดมา



ภาพที่ 4.91 ภาพการ Add library

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ไฟล์ที่ได้มาจะอยู่ในหมวดหมู่ Examples และอยู่ใน Library ที่โหลดมา



ภาพที่ 4.92 การเลือก Examples

เติม code ส่วนที่เป็นการสั่งให้ LED สว่างขึ้นตามคำที่ได้พูดมา

```
void led(int pred_index){
  switch (pred_index){
    case 0:
      digitalWrite(LEDR, LOW);
      digitalWrite(LEDG, HIGH);
      digitalWrite(LEDB, HIGH);
      break;

    case 1:
      digitalWrite(LEDR, HIGH);
      digitalWrite(LEDG, LOW);
      digitalWrite(LEDB, HIGH);
      break;

    case 2: // environment voice
      digitalWrite(LEDR, HIGH);
      digitalWrite(LEDG, HIGH);
      digitalWrite(LEDB, HIGH);
      break;

    case 3:
      digitalWrite(LEDR, HIGH);
      digitalWrite(LEDG, HIGH);
      digitalWrite(LEDB, LOW);
      break;
  }
}
```

ภาพที่ 4.93 Code ที่เติมเข้าไป

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

4.3.1.4 ผลการทดสอบชี้เฉพาะคำสำคัญด้วยการเรียนรู้ของเครื่องบนไมโครคอนโทรลเลอร์โดย edge impulse

เมื่อทดลองพูดคำว่า Hello LED สีแดงจะทำงาน เมื่อพูด Sawadee LED สีฟ้าจะทำงาน เมื่อพูด Konichiwa LED สีเขียวจะทำงาน

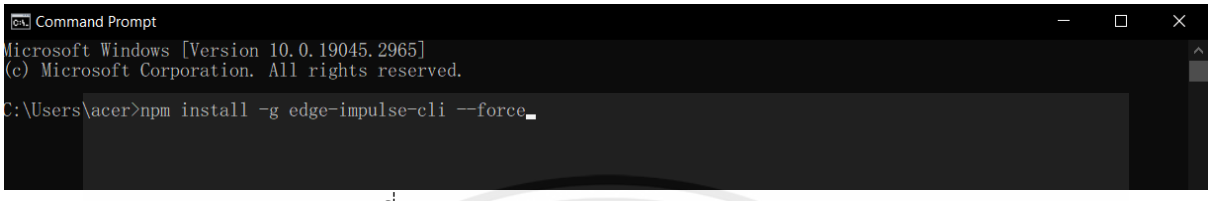
ภาพที่ 4.94 ผลที่ได้จากการทดลองการจับคำเฉพาะ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

4.3.2 ขั้นตอนการทดสอบคาดเดาท่าทางการขยับด้วยการเรียนรู้ของเครื่องด้วยหน่วยตรวจวัดการเคลื่อนไหวภายในโดย edge impulse

4.3.2.1 Collecting Data

ทำการ Download edge impulse CLI เพื่อสามารถ Upload ข้อมูลจากบอร์ดไปที่ edge impulse ได้ผ่าน Serial port

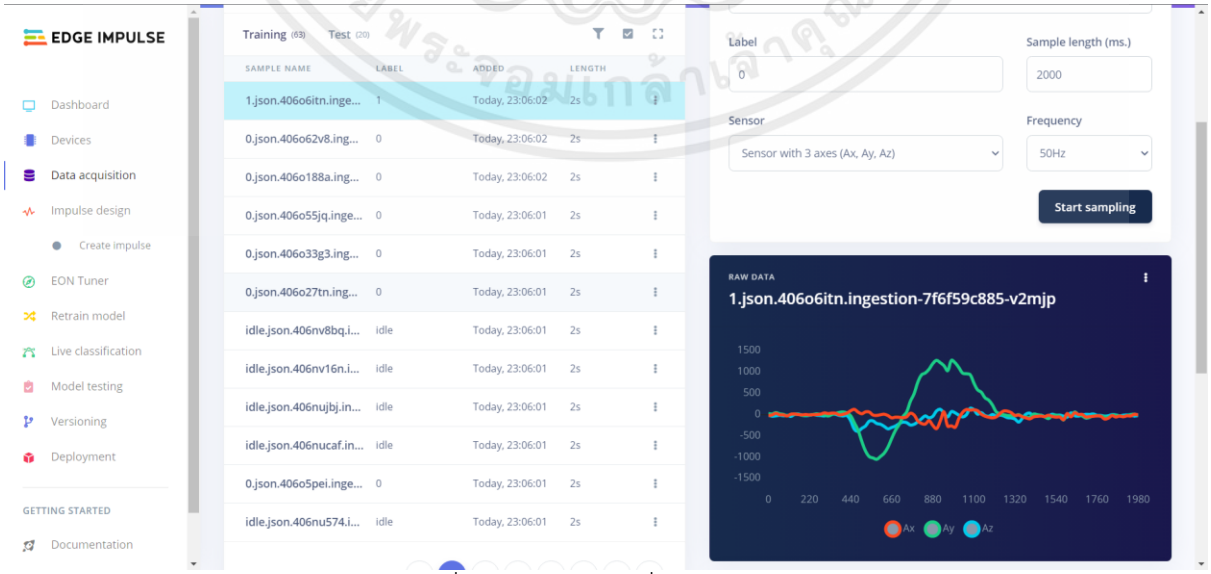


ภาพที่ 4.95 การ Download edge impulse CLI

ใช้คำสั่งเพื่อเชื่อมข้อมูลไปยัง edge impulse

ภาพที่ 4.96 การส่งข้อมูลไป edge impulse

ทำการเก็บข้อมูลการเคลื่อนไหวของแกน X,Y,Z

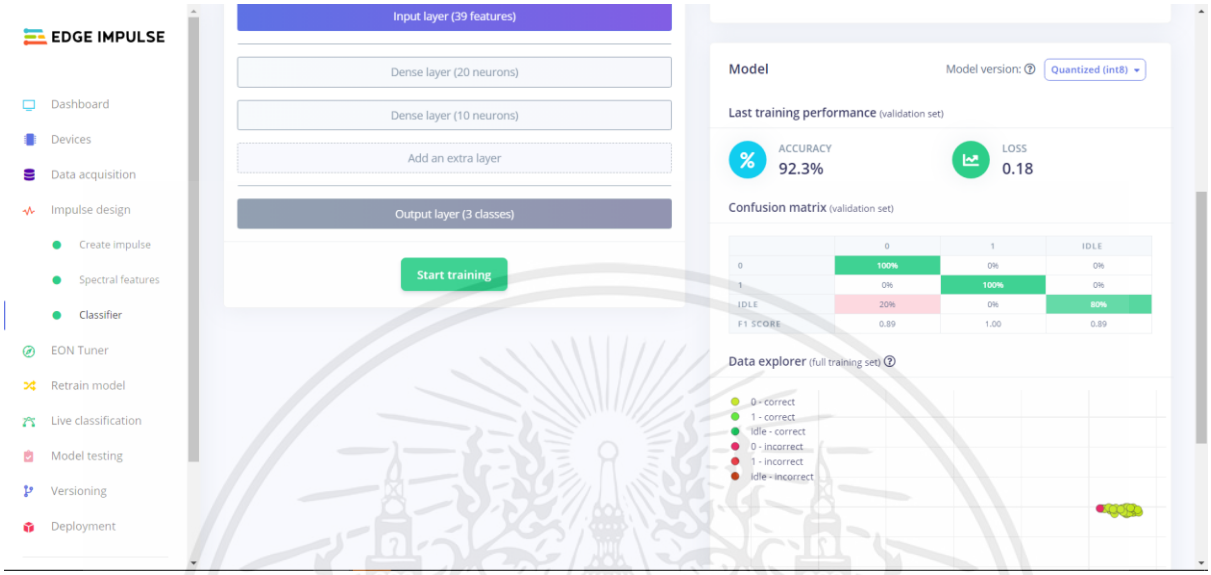


ภาพที่ 4.97 Dataset ที่เก็บข้อมูลมาได้

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

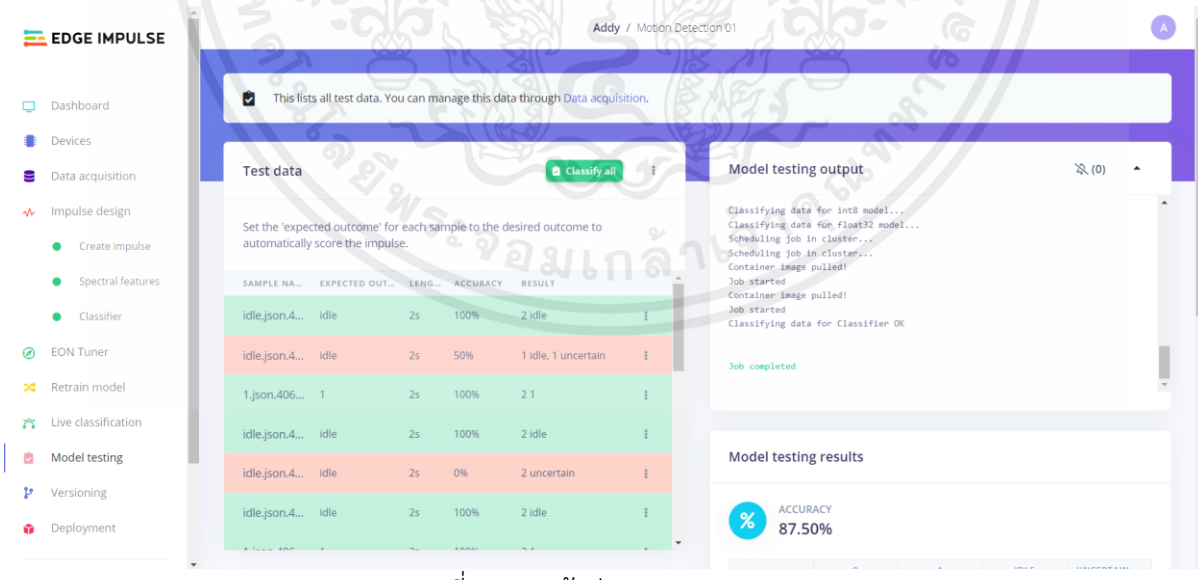
4.3.2.2 Training Model

เข้ามาในเมนู Create impulse ใน Add a processing block ให้เลือก Spectral Analysis ในส่วน Add a learning block ให้เลือก Classification แล้วจึงกด Save impulse จากนั้นทำการ train model



ภาพที่ 4.98 หน้าต่าง Create impulse

ทำการทดสอบความแม่นยำของ Model

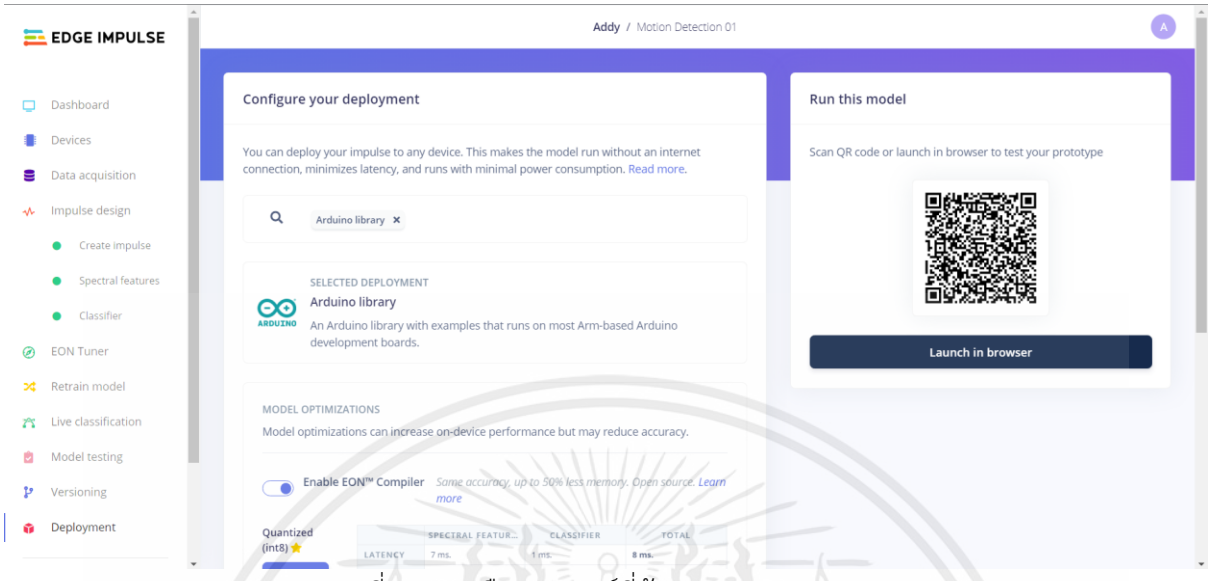


ภาพที่ 4.99 หน้าต่าง Model testing

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

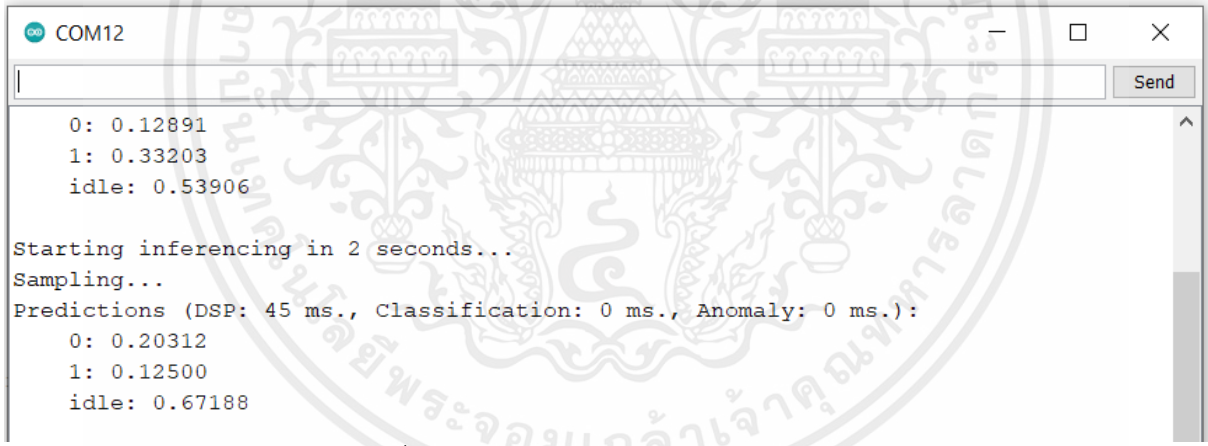
4.3.2.3 Deploy Code

เข้าในเมนู Deployment ทำการค้นหาและเลือก Arduino library



ภาพที่ 4.100 เลือกอุปกรณ์ที่ต้องการ Deployment

Upload code เพื่อใช้งาน model



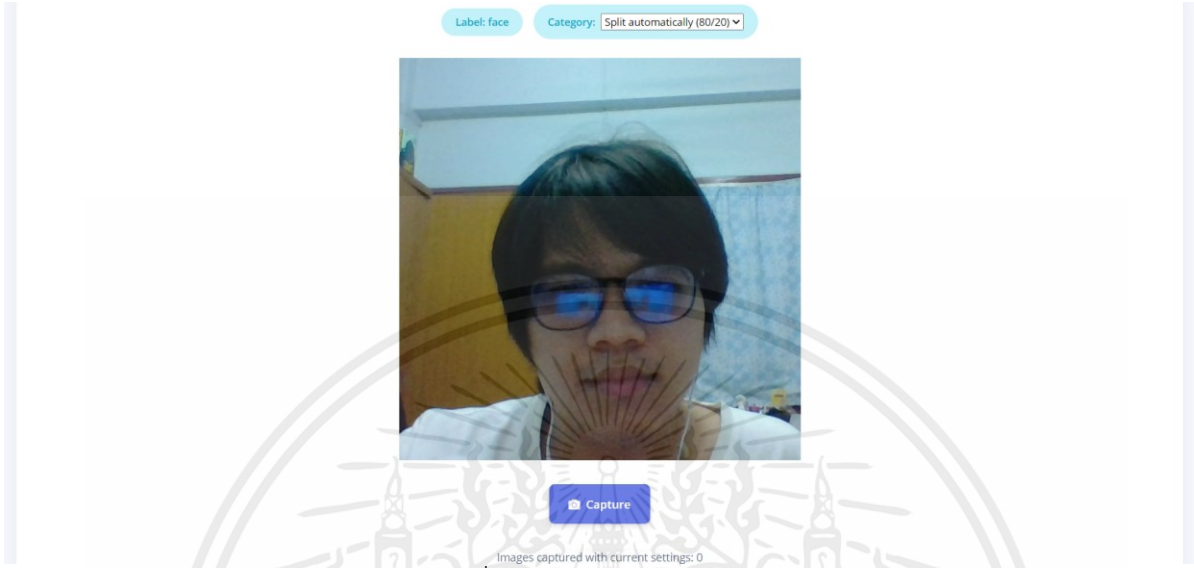
ภาพที่ 4.101 ผลลัพธ์การทดลองตรวจจับท่าทาง

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

4.4 การทดลองการตรวจจับวัตถุด้วยการเรียนรู้ของเครื่องบนอุปกรณ์ไมโครคอนโทรลเลอร์

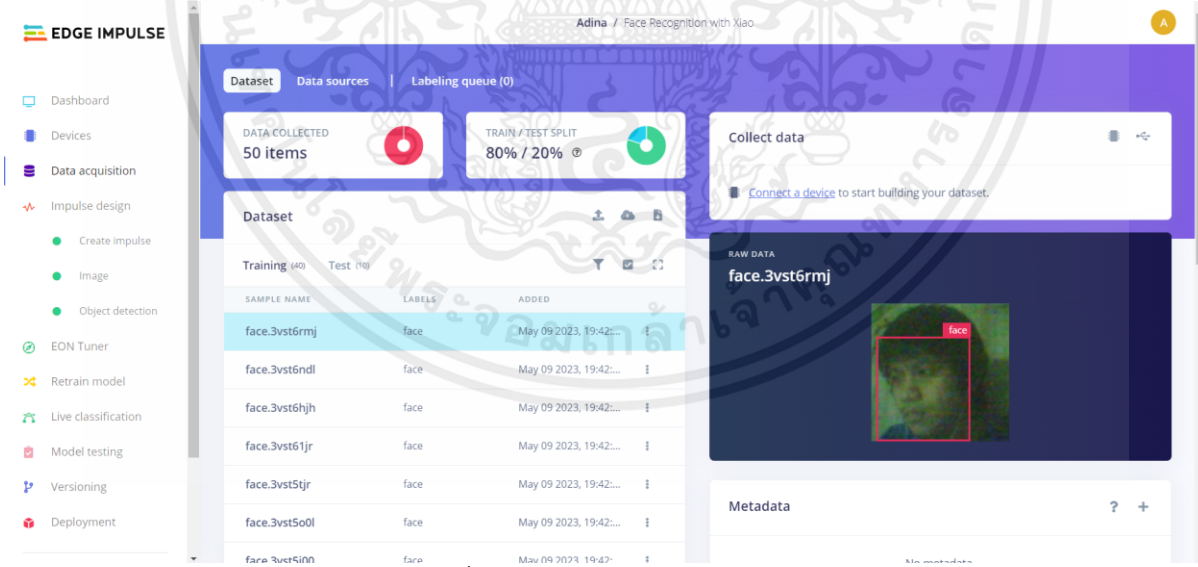
4.4.1 Collecting Data

ทำการเก็บข้อมูลจากกล้อง webcam หรือ กล้องมือถือ



ภาพที่ 4.102 การเก็บข้อมูลกล้อง

ทำการเก็บข้อมูลและบันทึกข้อมูล label ใบหน้า

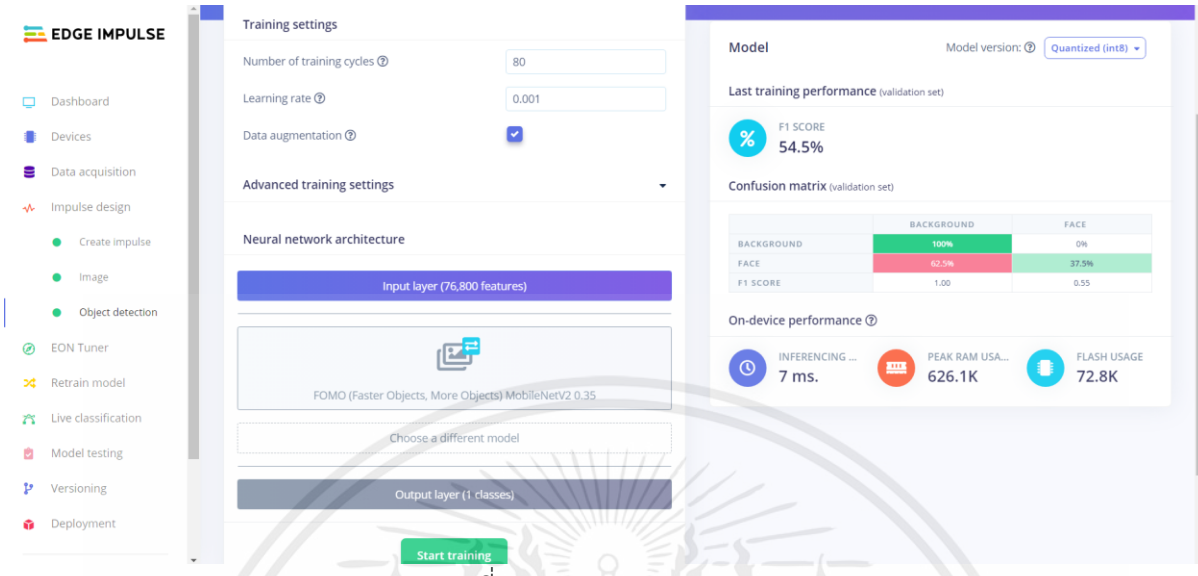


ภาพที่ 4.103 Dataset ข้อมูลใบหน้า

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

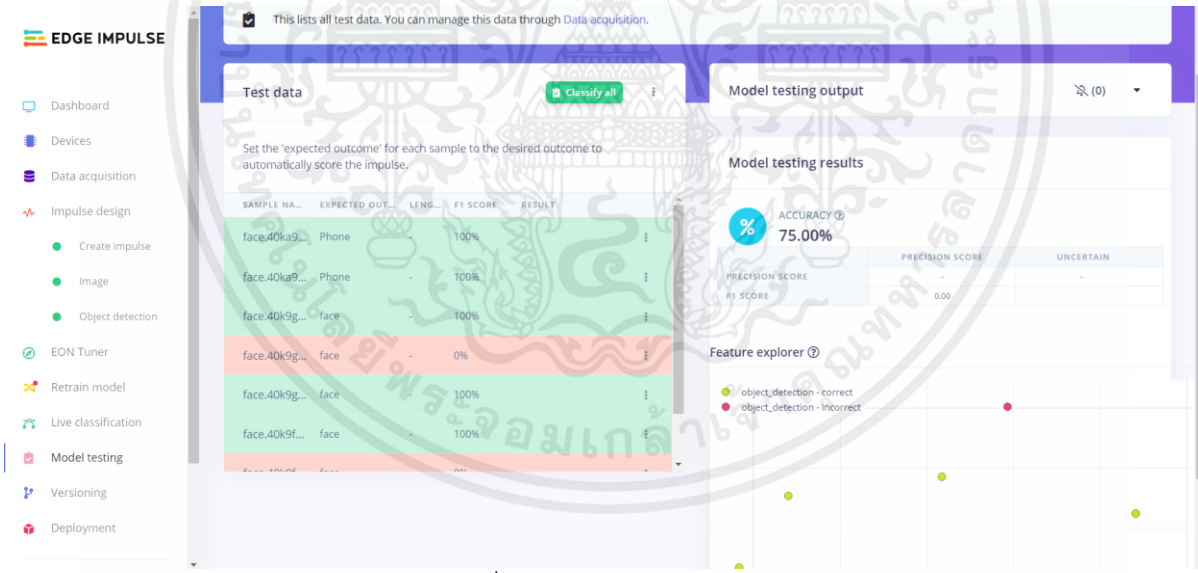
4.4.2 Training Model

ในเมนู Object detection ให้ทำการ Start training เพื่อเริ่ม train model



ภาพที่ 4.104 การ train model

ทำการ Test Model เพื่อดู Accuracy ของ Model ที่ได้

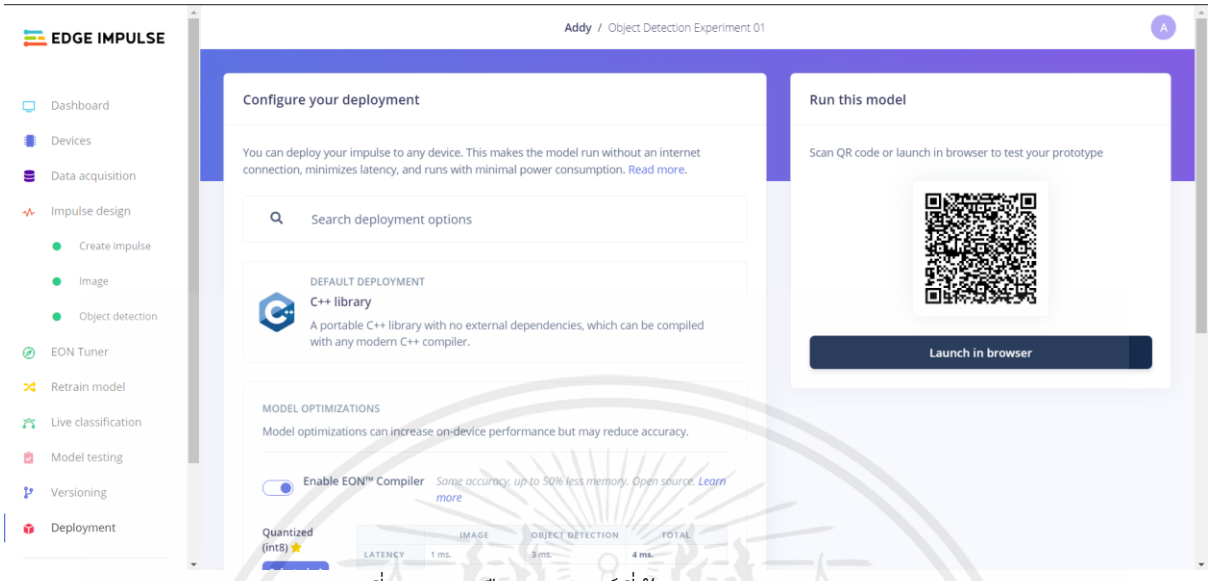


ภาพที่ 4.105 การ test model

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

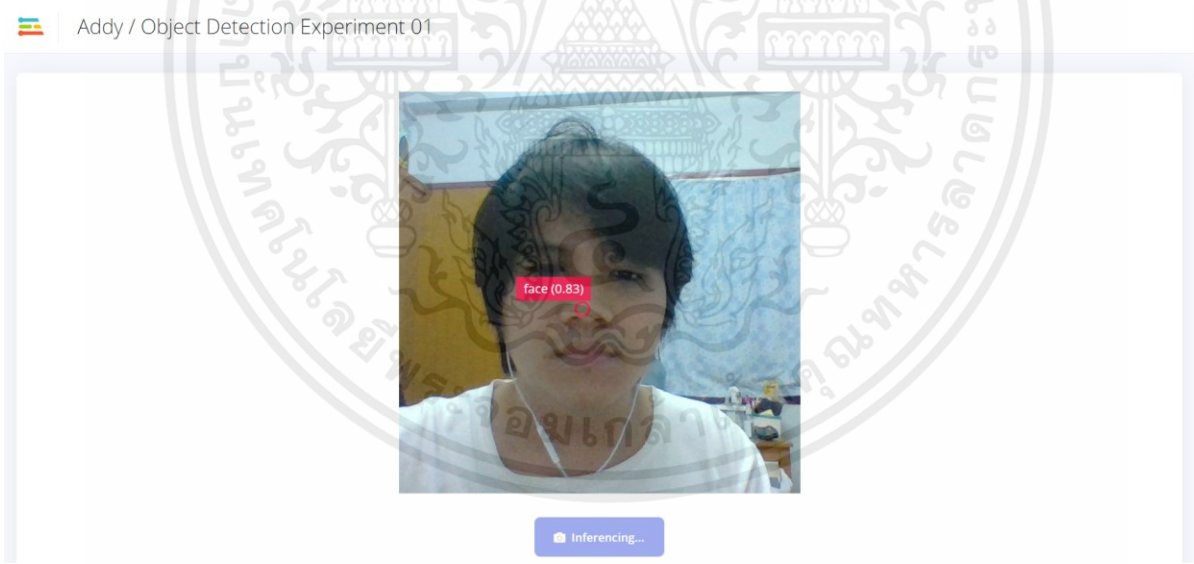
4.3.1.2 Deploy Code

ทำการเลือกเมนู Build และทำการใช้งานบนคอมพิวเตอร์หรือโทรศัพท์มือถือ



ภาพที่ 4.106 เลือกอุปกรณ์ที่ต้องการ Deployment

โดยโปรแกรมจะสามารถตรวจจับและบอกได้ว่าเป็นใบหน้าเมื่อเห็นใบหน้า



ภาพที่ 4.107 ผลลัพธ์การทดสอบการตรวจจับใบหน้า

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

4.5 การจัดทำแบบเรียนเพื่อการศึกษาการเรียนของเครื่องบนอุปกรณ์ไมโครคอนโทรลเลอร์

การเรียนรู้ของเครื่องบนอุปกรณ์ไมโครคอนโทรลเลอร์หรือ Tiny Machine Learning หรือโดยย่อ “TinyML” หมายถึง เทคโนโลยีที่นำ Machine Learning ซึ่งเป็นการเรียนรู้ของเครื่องคอมพิวเตอร์ให้โปรแกรมสามารถคาดเดาหรือเรียนรู้ถึงสิ่งที่เราต้องการ เช่น การแยกรูปคน สัตว์ สิ่งของ การคาดเดาสภาพอากาศหรือผลประกอบการของธุรกิจในไตรมาสถัดไป เป็นต้น ซึ่งปกติจำเป็นที่ต้องใช้ระบบประมวลผลที่มีกำลังการประมวลที่สูงให้สามารถนำมาใช้งานบนอุปกรณ์ที่มีกำลังในการประมวลผลที่ต่ำ เช่น โทรศัพท์มือถือหรืออุปกรณ์ไมโครคอนโทรลเลอร์ โดยมีวัตถุประสงค์เพื่อนำ Machine Learning ไปประยุกต์ใช้กับประเภทงานให้ได้กว้างมากขึ้น โดยตัวอย่างการใช้งาน TinyML ในชีวิตประจำวันที่โดดเด่นที่สุดก็คือ การปลุก Google assistant หรือ Siri ด้วยคำสั่งการพูดอย่าง Hey Siri และ OK Google

โดยกระบวนการสร้าง Machine Learning จะมีการใช้เครื่องมืออย่าง TensorFlow โดยจะได้เป็นโมเดลที่ผ่านการ train จากนั้นจึงนำโมเดลที่ได้มาผ่านกระบวนการย่อขนาดให้เล็กลงด้วยกระบวนการต่าง ๆ ด้วยเครื่องมือ TensorFlow Lite และ TensorFlow Lite Micro แล้วจึงนำโมเดลไปติดตั้งลงบนอุปกรณ์ปลายทาง

ในปัจจุบันในมีการพัฒนาเครื่องมืออีกหนึ่งอย่างที่ถูกพัฒนาเพื่อสนับสนุนการทำ TinyML ให้สามารถเข้าถึงได้ง่ายและสะดวกมากขึ้นชื่อว่า Edge Impulse ซึ่งถูกพัฒนาขึ้นมาให้ง่ายต่อการใช้งาน ในแบบเรียนฉบับนี้จะเน้นไปที่การใช้งาน Edge Impulse ในขั้นเบื้องต้น และตัวอย่างการนำไปใช้งานจริง และแบบฝึกทดลอง

อุปกรณ์ที่ต้องใช้ในการทดลอง

- บอร์ด Seeed Studio XIAO nRF52840 (Sense)

- คอมพิวเตอร์ที่มี webcam หรือโทรศัพท์มือถือ ที่สามารถเชื่อมต่ออินเตอร์เน็ตได้

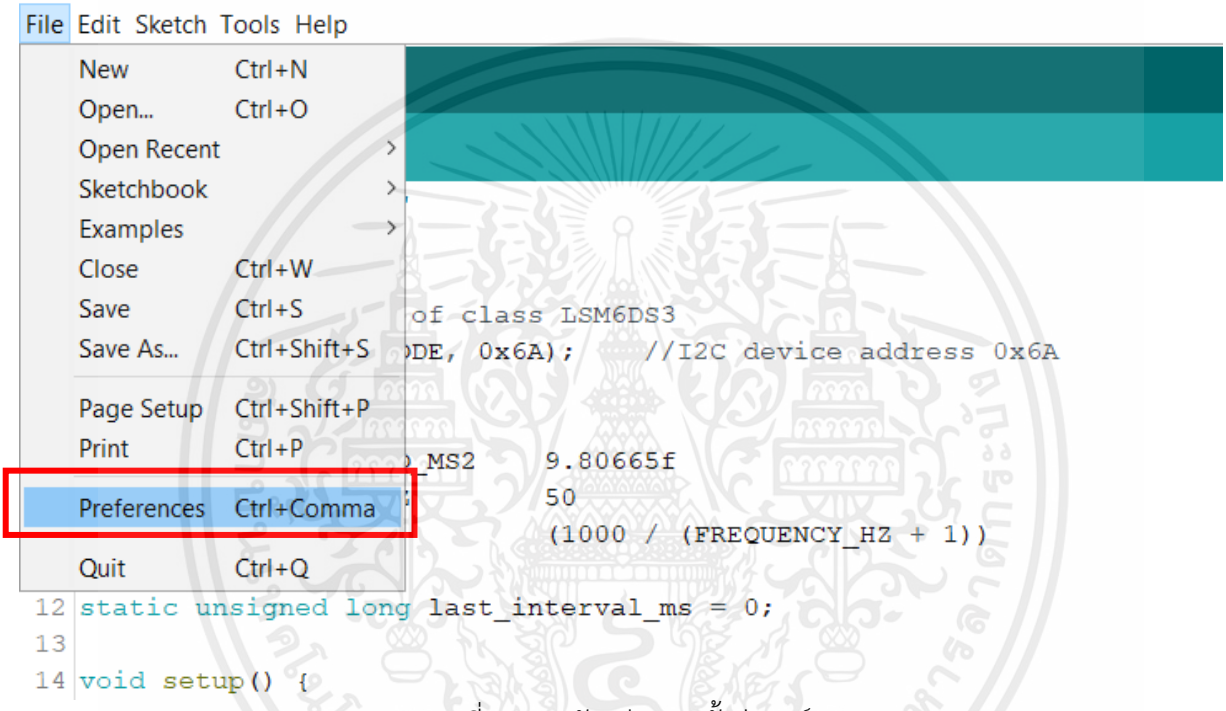
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

โปรแกรมที่ต้องใช้ในการทดลอง

- Arduino IDE
- Web browser เช่น Google Chrome
- Python 3
- Node.js เวอร์ชั่น 14 ขึ้นไป

การบันทึกบอร์ด Xiao บน Arduino IDE

ให้เข้าไปในเมนู File และเลือก Preferences

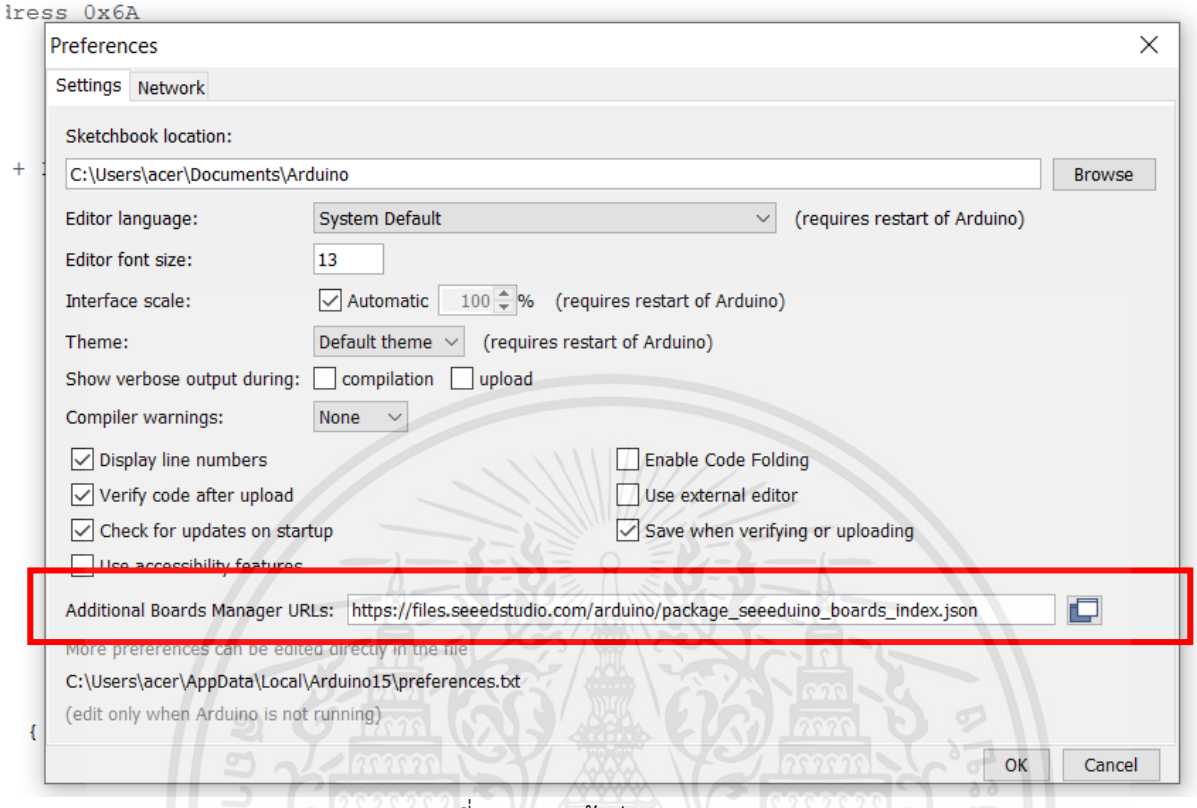


ภาพที่ 4.108 ตัวอย่างการตั้งค่าบอร์ด

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

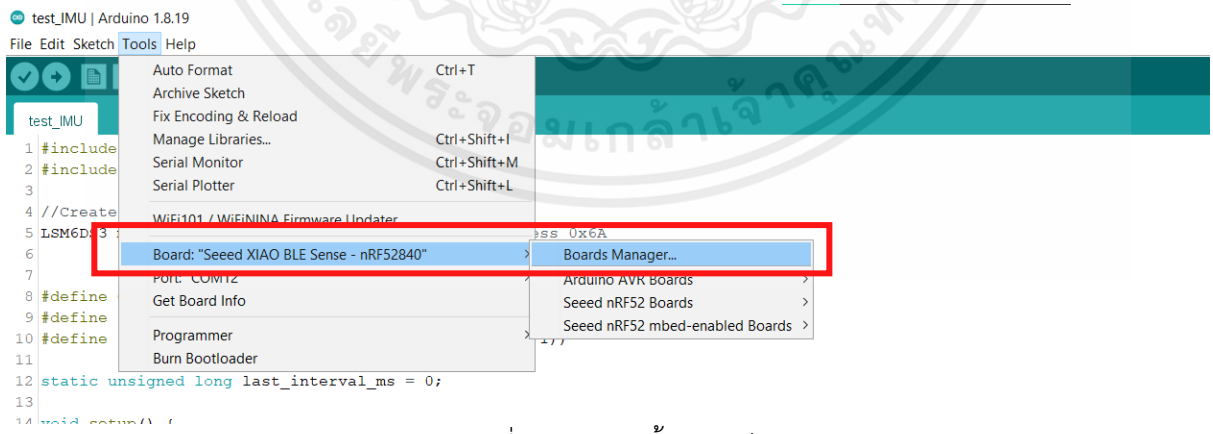
ในหน้าต่าง Preferences ในส่วน Additional Boards Manager URLs ให้ใส่

https://files.seeedstudio.com/arduino/package_seeeduino_boards_index.json



ภาพที่ 4.109 หน้าต่าง Preferences

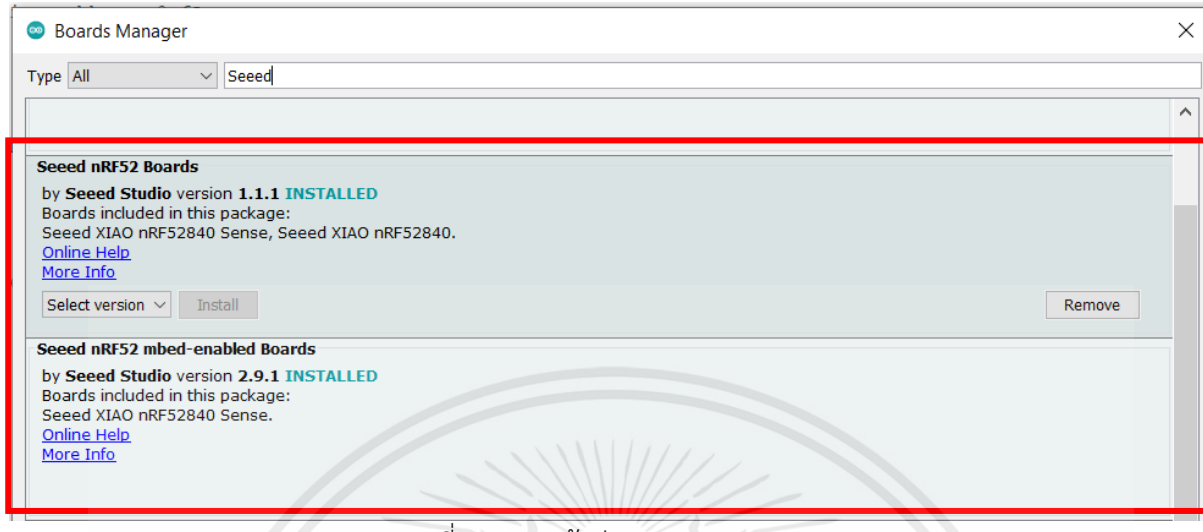
ต่อมาในเมนู Tools ไปที่ Board > Boards Manager...



ภาพที่ 4.110 การตั้งค่าบอร์ด

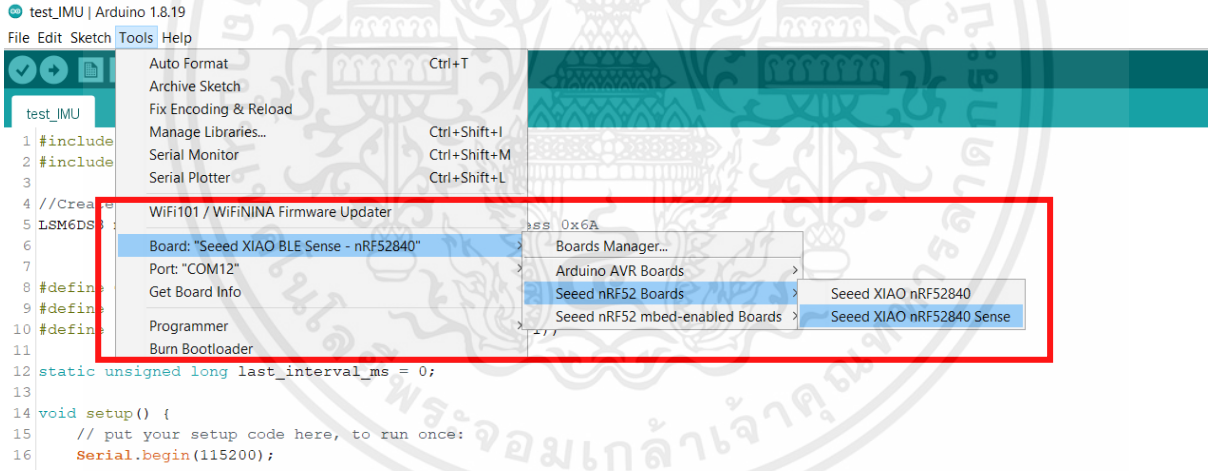
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ค้นหา Seeed nRF52 Boards และ Seeed nRF52 mbed-enabled Boards และโหลดเวอร์ชั่นล่าสุดมาทั้ง 2 อัน



ภาพที่ 4.111 หน้าต่าง Board Manager

ในเมนู Board จะสามารถเลือก Seed XAIO nRF52840 Sense ได้

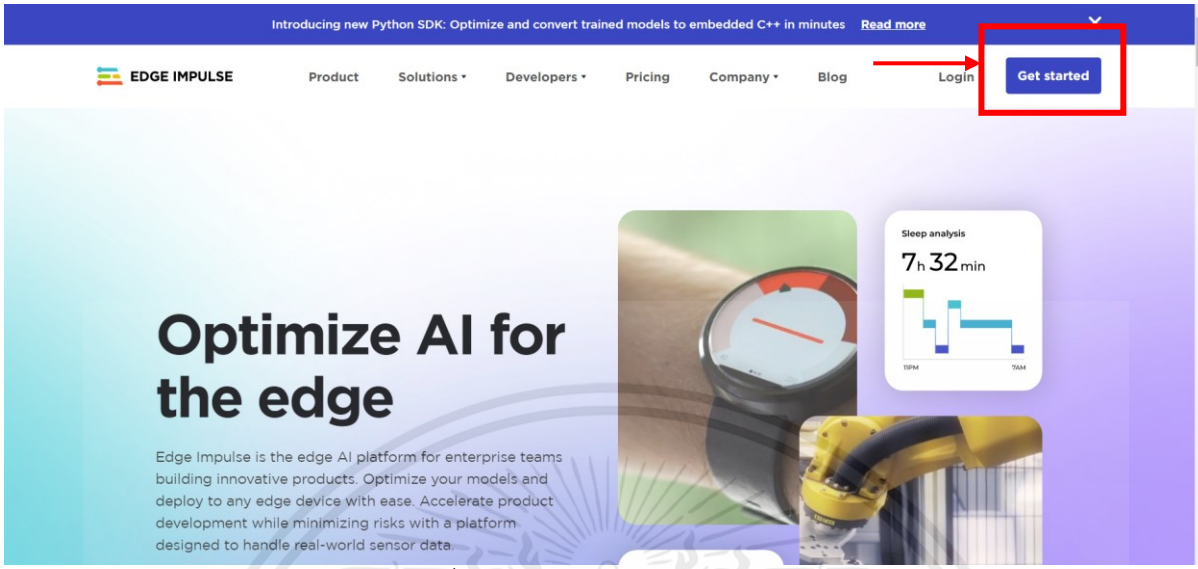


ภาพที่ 4.112 เสร็จสิ้นการตั้งค่าบอร์ด

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

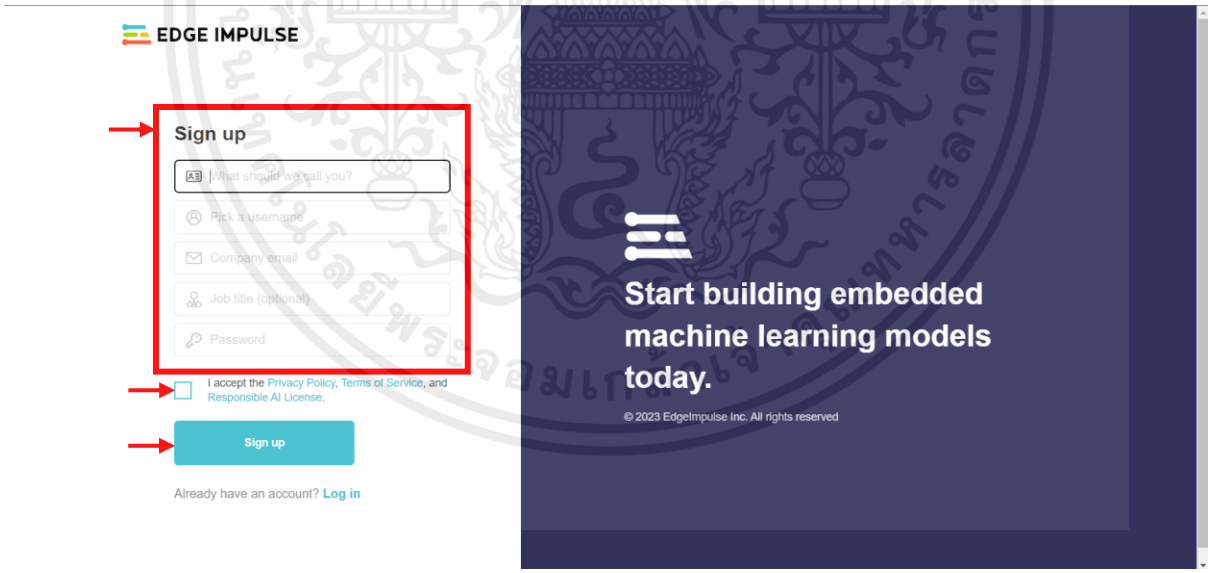
การสมัคร Edge Impulse

ให้ผู้ศึกษาทำการเข้าเว็บไซต์ของ Edge Impulse และทำการสมัคร Account



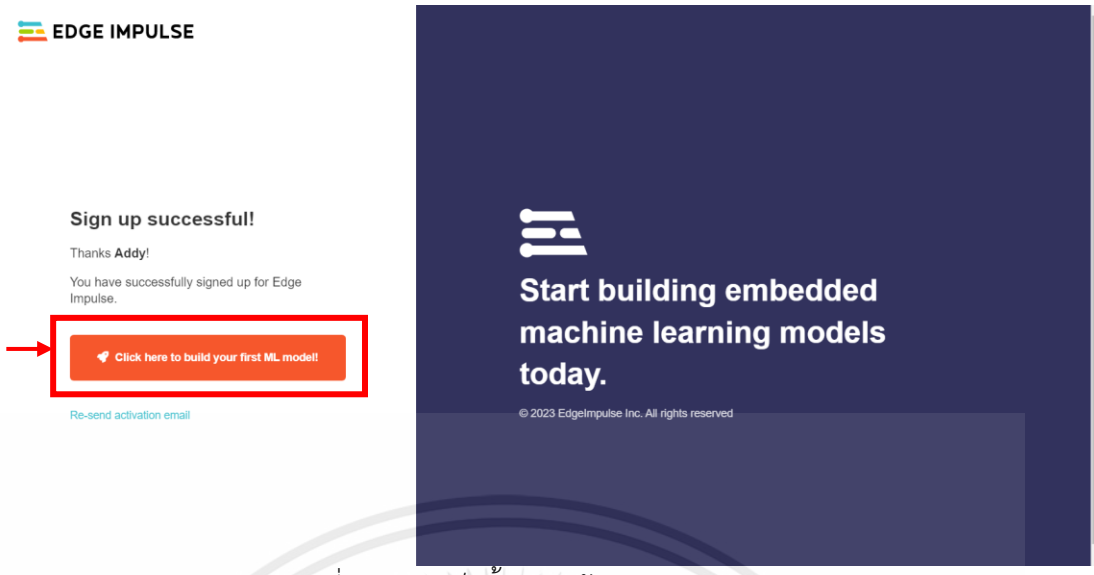
ภาพที่ 4.113 หน้าแรก edge impulse

กรอกรายละเอียดกดยอมรับเงื่อนไขและกด Sign up โดยจะมี E-mail สำหรับ Activate Account ส่งไปใน mail ที่กรอกเอาไว้



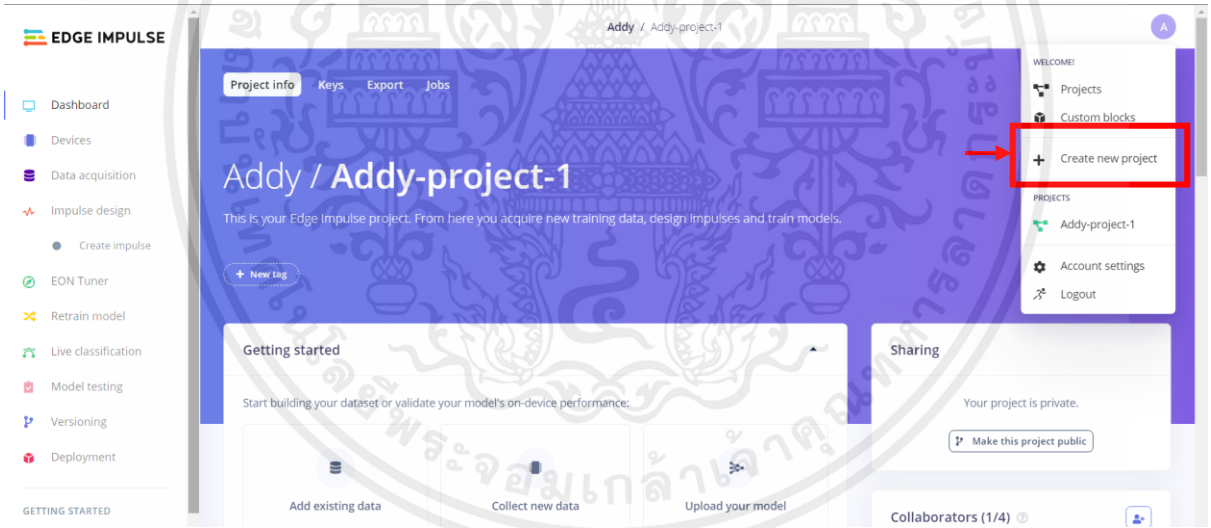
ภาพที่ 4.114 หน้าต่างสร้าง account edge impulse

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



ภาพที่ 4.115 เสร็จสิ้นการสร้าง account

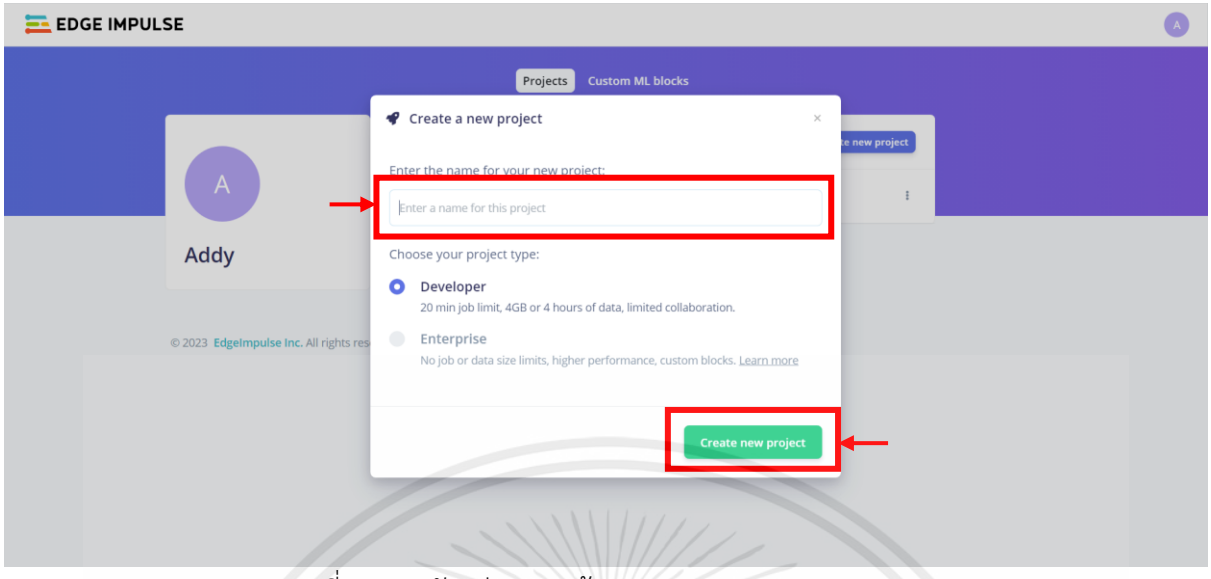
กดที่รูปโปรไฟล์เพื่อแสดงเมนู แล้วเลือก Create new project เพื่อสร้างโปรเจคใหม่



ภาพที่ 4.116 หน้าต่าง Dashboard edge impulse

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

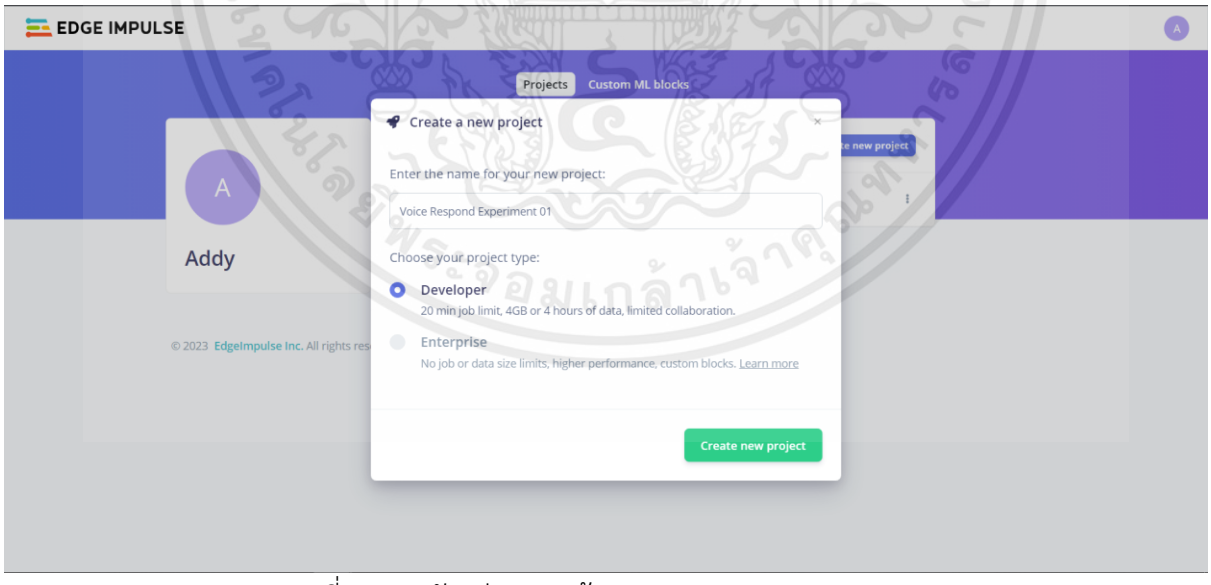
ใส่ชื่อโปรเจคและกด Create new project



ภาพที่ 4.117 ตัวอย่างการสร้าง project edge impulse (1)

1.Voice Responder (การทดสอบชี้เฉพาะคำสำคัญด้วยการเรียนรู้ของเครื่องบนไมโครคอนโทรลเลอร์)

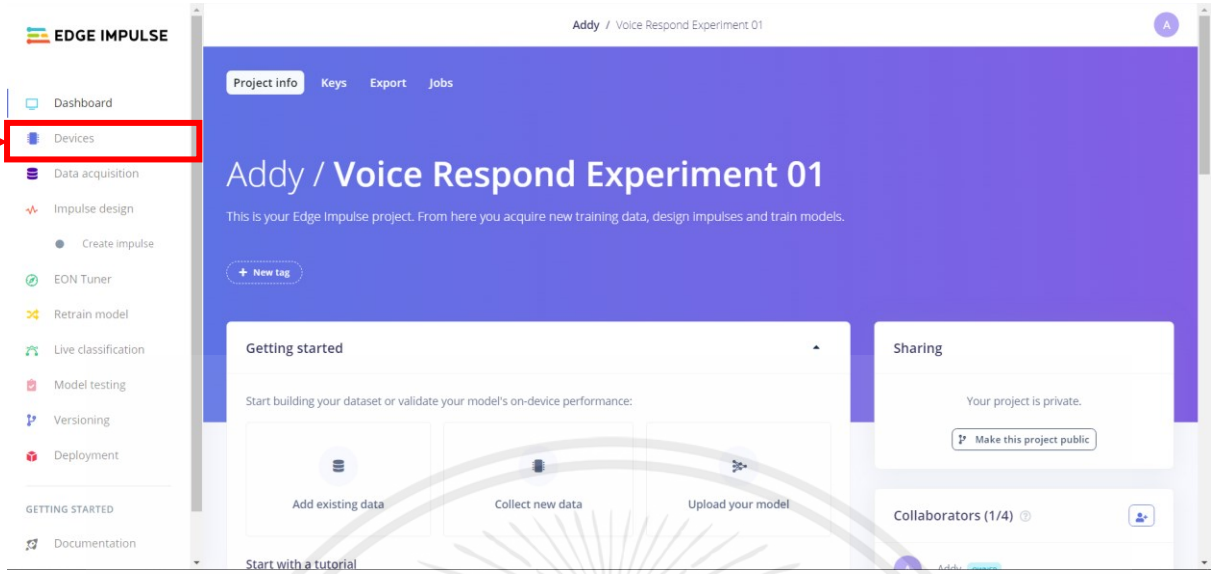
สร้างโปรเจคใหม่ชื่อ Voice Respond Experiment 01



ภาพที่ 4.118 ตัวอย่างการสร้าง project edge impulse (1)

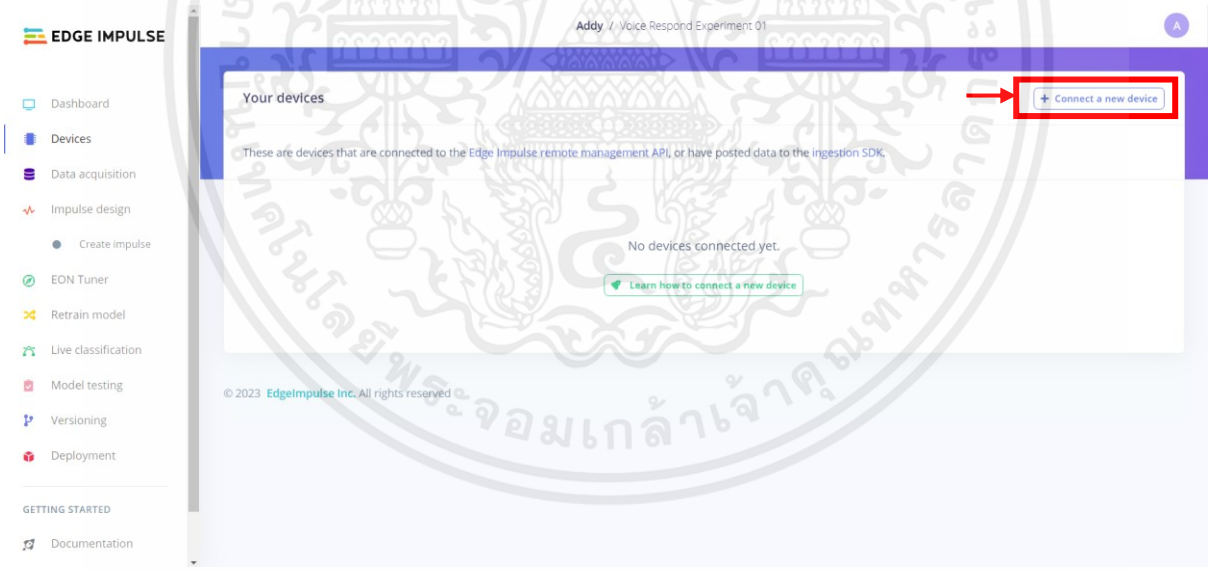
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ในหน้าหลักของโปรเจค กด Devices เพื่อเพิ่มอุปกรณ์เชื่อมต่อ



ภาพที่ 4.119 ตัวอย่างการเชื่อมต่ออุปกรณ์เก็บข้อมูลเสียง (1)

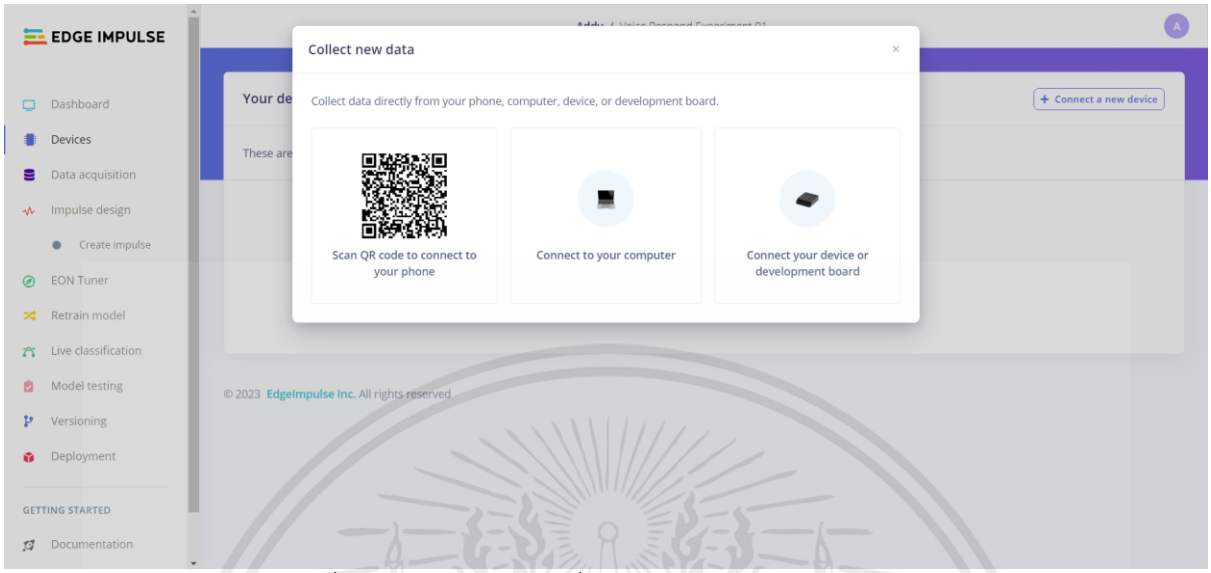
กด Connect a new device



ภาพที่ 4.120 ตัวอย่างการเชื่อมต่ออุปกรณ์เก็บข้อมูลเสียง (2)

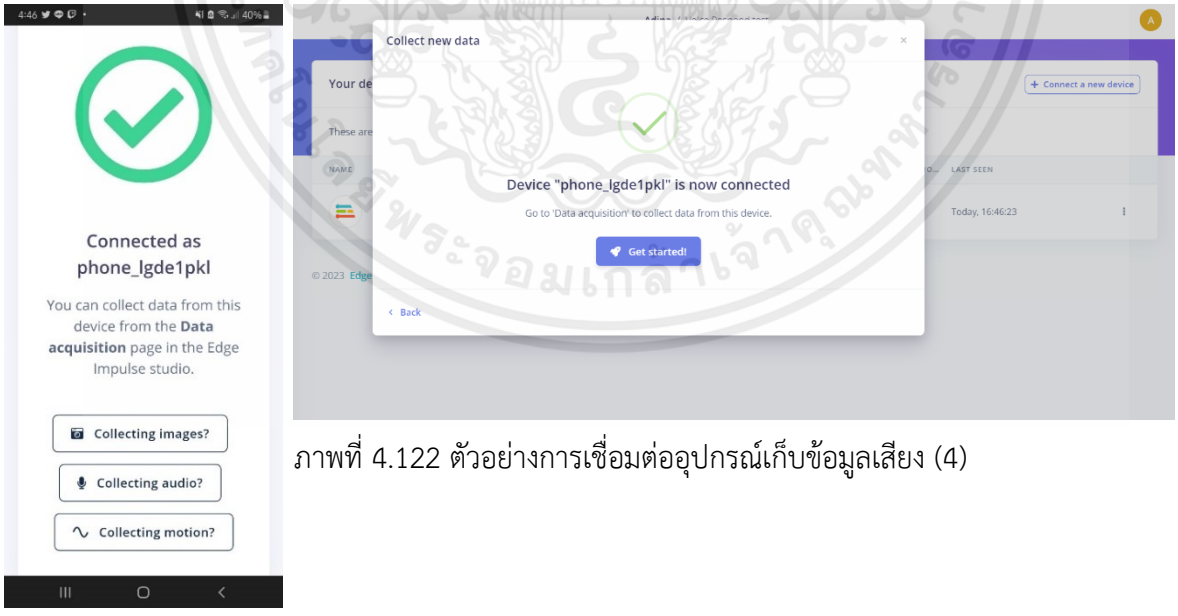
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

จะขึ้นหน้าต่างเลือกอุปกรณ์ที่จะเชื่อมขึ้นมา ให้ทำโทรศัพท์มือถือมาสแกน QR code เพื่อเชื่อมต่อ โดยจะขึ้นลิงค์ไปยังเว็บ edge impulse



ภาพที่ 4.121 ตัวอย่างการเชื่อมต่ออุปกรณ์เก็บข้อมูลเสียง (3)

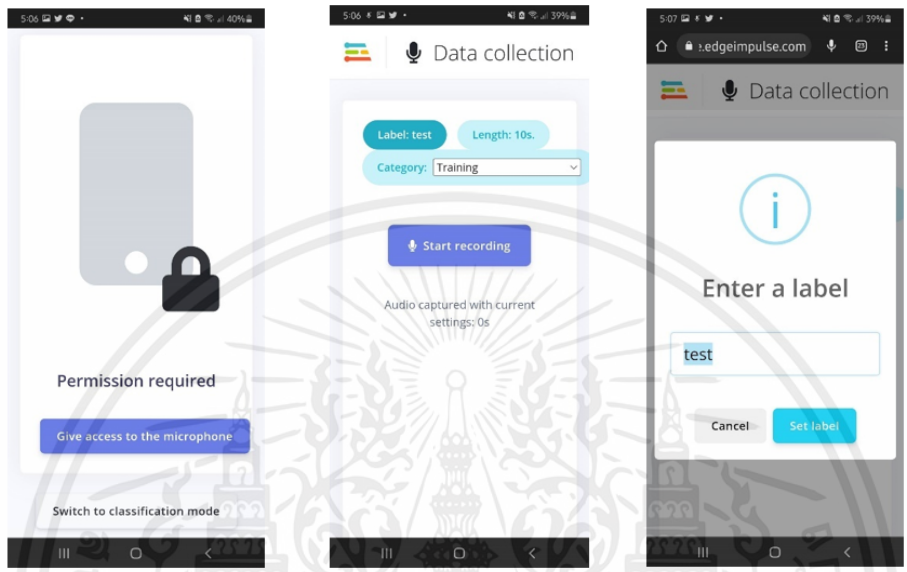
เมื่อเชื่อมต่อเรียบร้อยแล้วจะขึ้นชื่ออุปกรณ์ใหม่ในหน้าเมนู Devices เพื่อเก็บข้อมูลเสียงทำการกด Collecting audio?



ภาพที่ 4.122 ตัวอย่างการเชื่อมต่ออุปกรณ์เก็บข้อมูลเสียง (4)

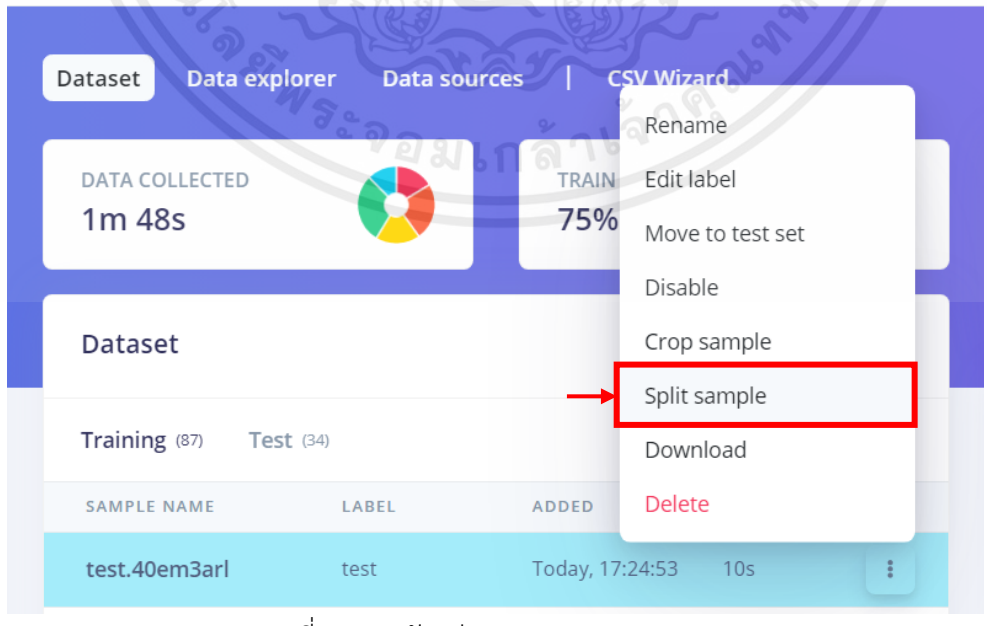
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ให้ permission เข้าถึง Microphone ต่อมาในหน้า recording จะมีเมนู label โดยกดเข้าไปเพื่อตั้งชื่อโดยให้ใส่ชื่อ Hello ส่วนเมนู length จะเป็นการกำหนดความยาวในการอัดเสียงแต่ละครั้ง โดยกำหนดให้เป็น 10 วินาที แล้วทำการกด Start recording จะทำการเริ่มอัดเสียง ให้ทดลองพูดคำว่า Hello จนกว่าจะหมดเวลา



ภาพที่ 4.123 ตัวอย่างการเก็บข้อมูลเสียง

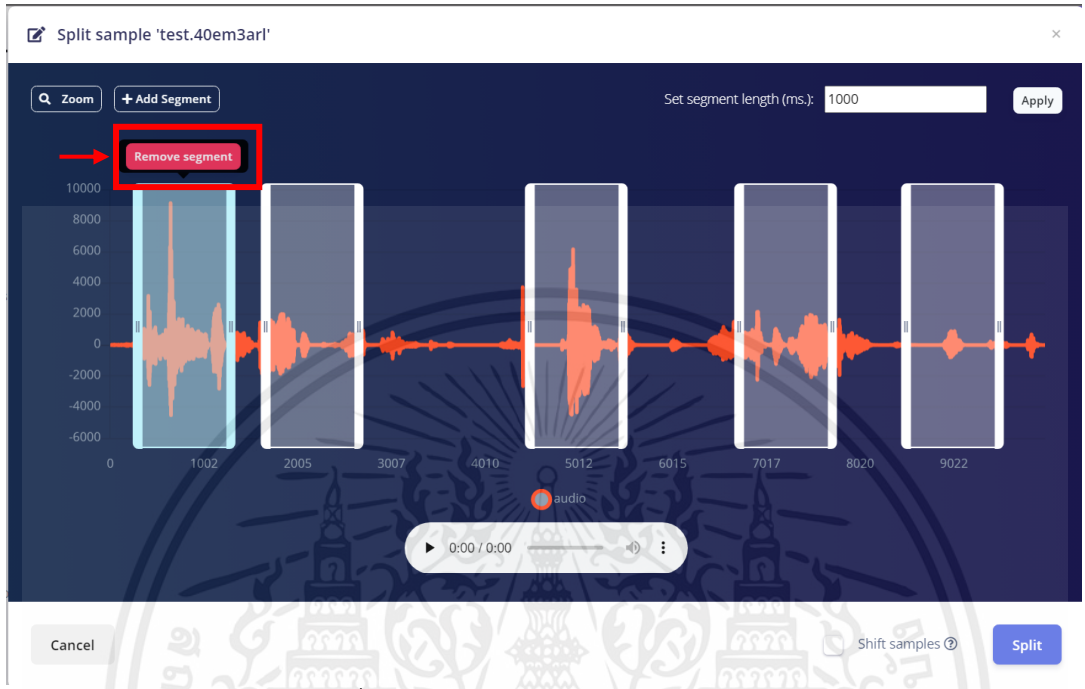
ในหน้าเมนู Data acquisition จะขึ้นแสดงข้อมูลเสียงที่เราได้อัดไว้ขึ้นมา กดที่เมนูจุด 3 จุด บริเวณด้านข้าง เลือก Split sample



ภาพที่ 4.124 ตัวอย่างการ Split sample (1)

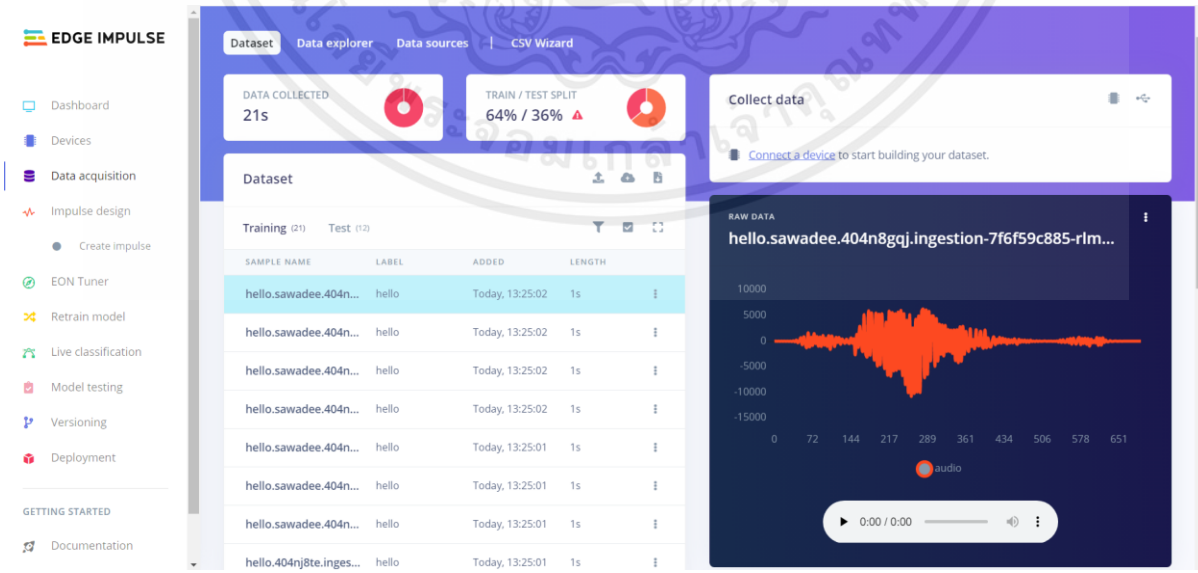
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

จะขึ้นหน้าต่างที่ใช้ในการ Split sample ซึ่งก็คือการตัดแบ่งข้อมูลตัวอย่าง โดยจะทำการแบ่งตัวอย่างเสียงที่อัดมาเป็นเสียงย่อย ๆ เพื่อเอาเฉพาะเสียงที่เราต้องการ โดยทำการกด Add Segment และกดที่กราฟเสียงเพื่อเพิ่ม Segment



ภาพที่ 4.125 ตัวอย่างการ Split sample (2)

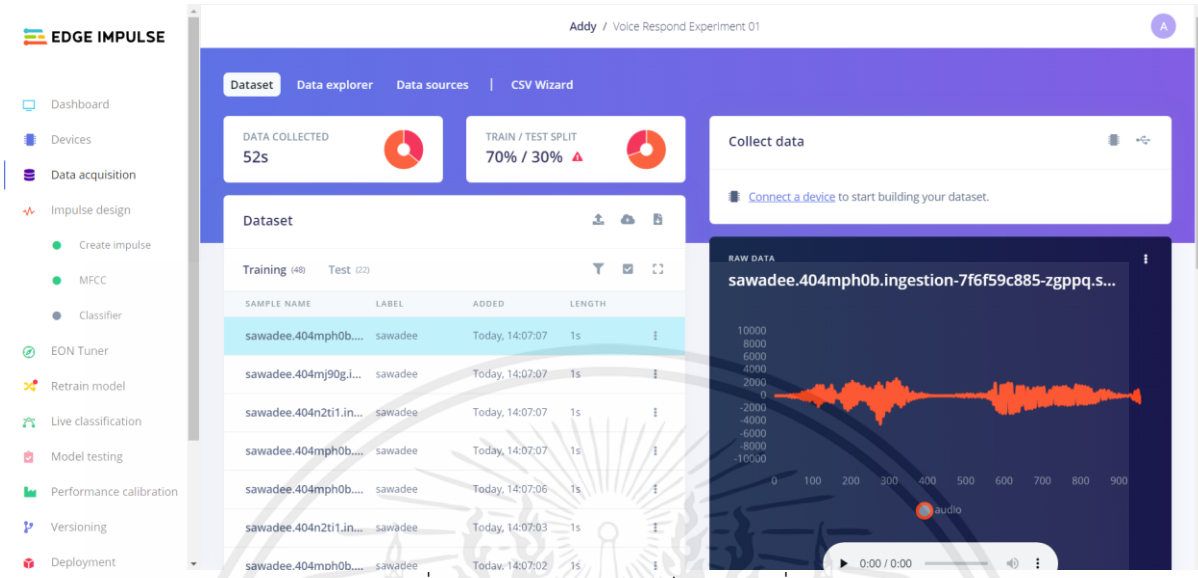
ต่อมาให้ทำการอัดเสียง label hello เพิ่มอีกและทำการ Split Sample ให้ได้ประมาณ 30 ถึง 40 ตัวอย่าง



ภาพที่ 4.126 ตัวอย่างการ Split sample (3)

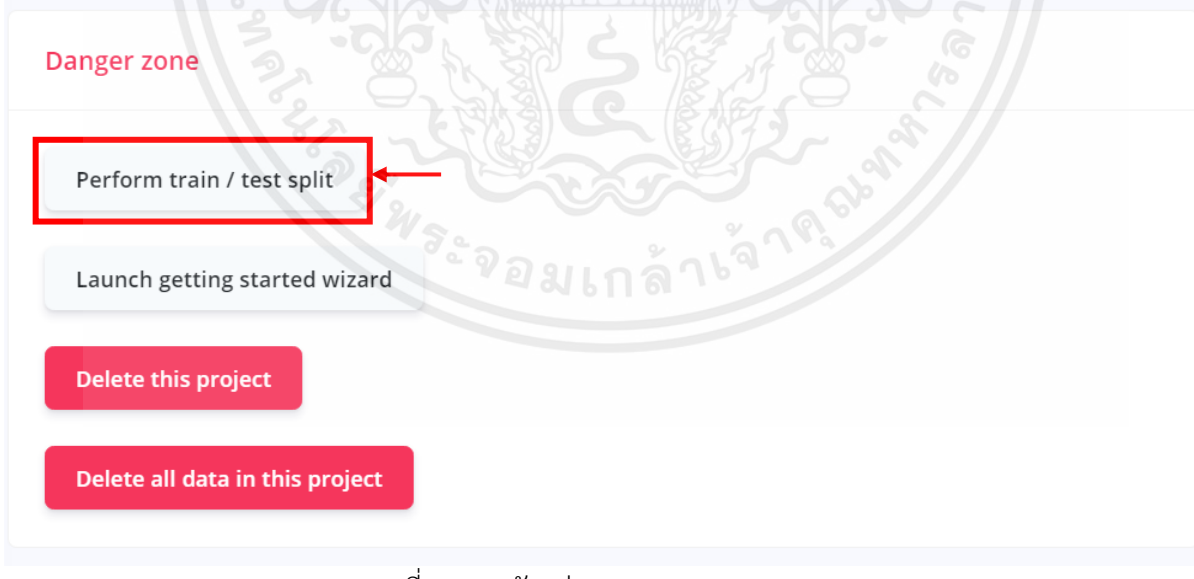
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ทำการอัดเสียงเพิ่มอีก label โดยครั้งนี้ให้ชื่อ sawadee และทำการอัดคำว่าสวัสดีและทำการ Split ให้ได้ประมาณ 30 ถึง 40 ตัวอย่าง



ภาพที่ 4.127 ตัวอย่างการเก็บข้อมูลเพิ่มเติม

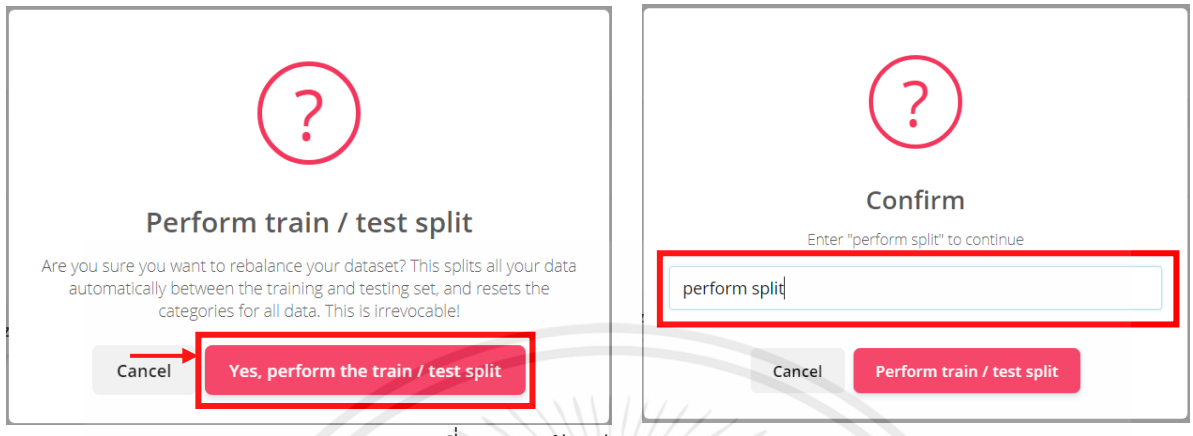
ไปที่หน้าเมนู Dashboard เลื่อนมาบริเวณ Danger zone เลือกเมนู Perform train / test split เพื่อแบ่งข้อมูลส่วนหนึ่งไว้ใช้สำหรับ Model testing



ภาพที่ 4.128 ตัวอย่างการ perform split (1)

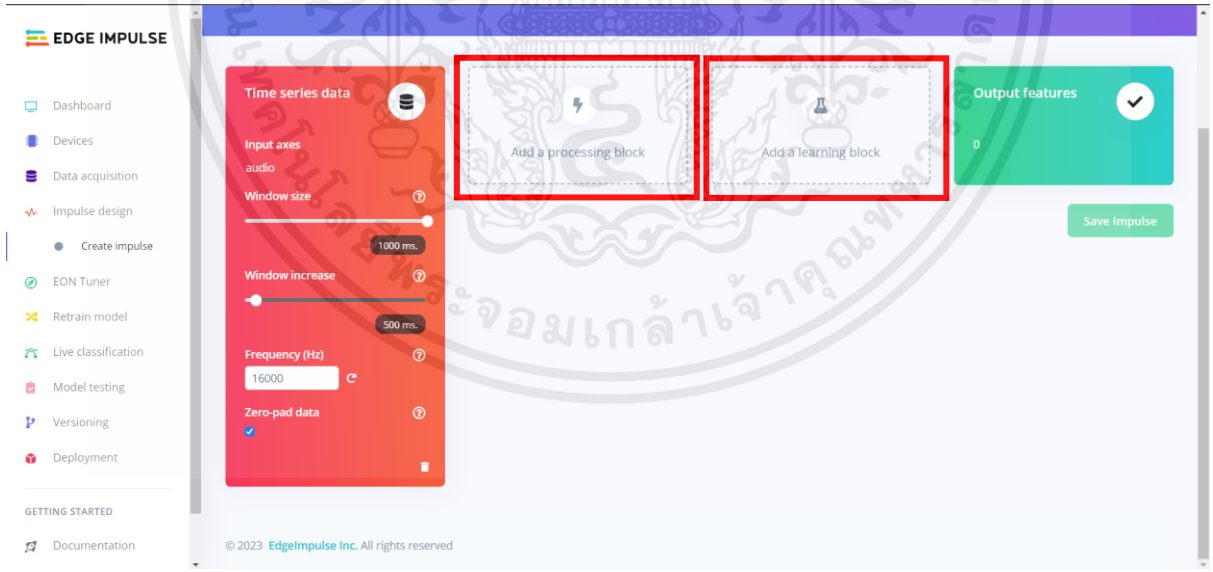
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

กดยืนยันการ perform train / test split และพิมพ์ perform split เพื่อยืนยัน



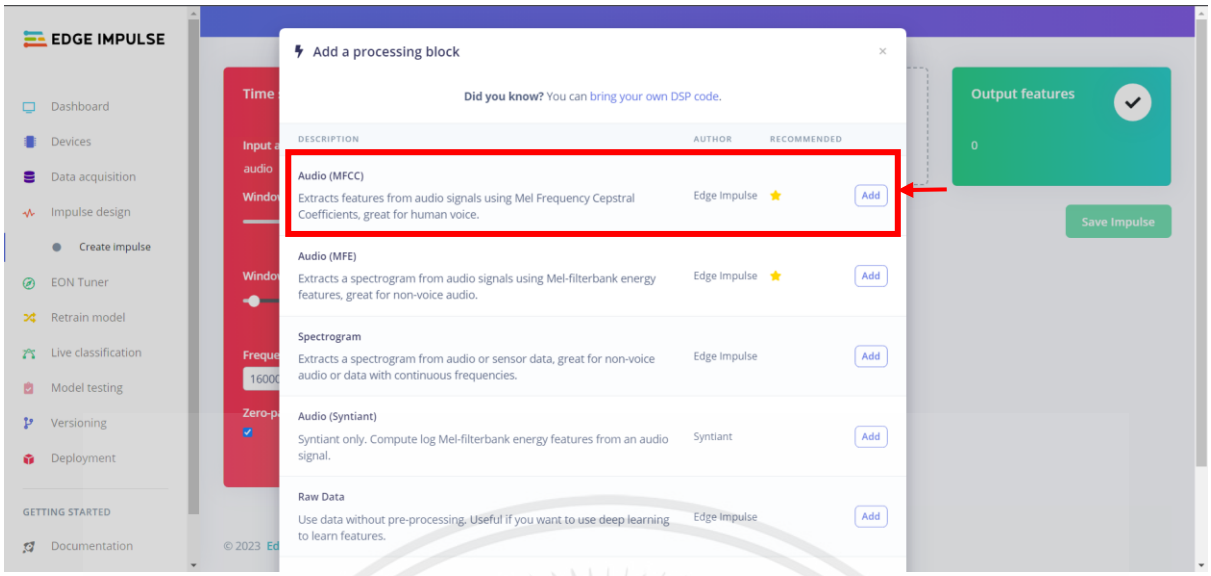
ภาพที่ 4.129 ตัวอย่างการ perform split (2)

เข้ามาที่หน้าต่าง Create impulse ทำการ Add a processing block โดย เลือกเป็น Audio(MFCC) ส่วน Add a learning block เลือกเป็น Classification และทำการ Save impulse

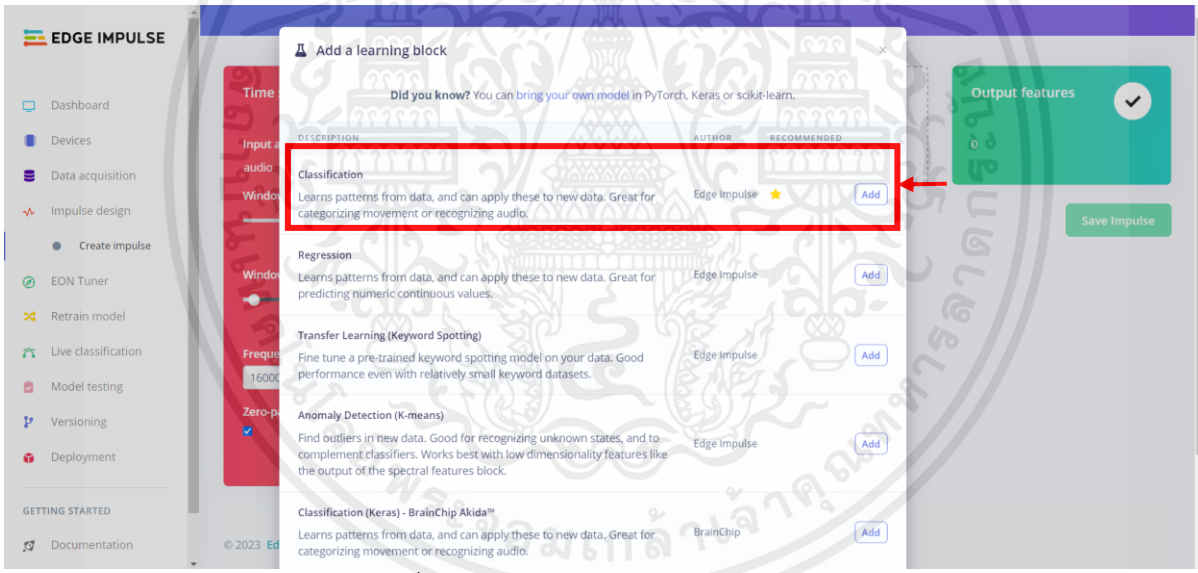


ภาพที่ 4.130 ตัวอย่างการ Create impulse (1)

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

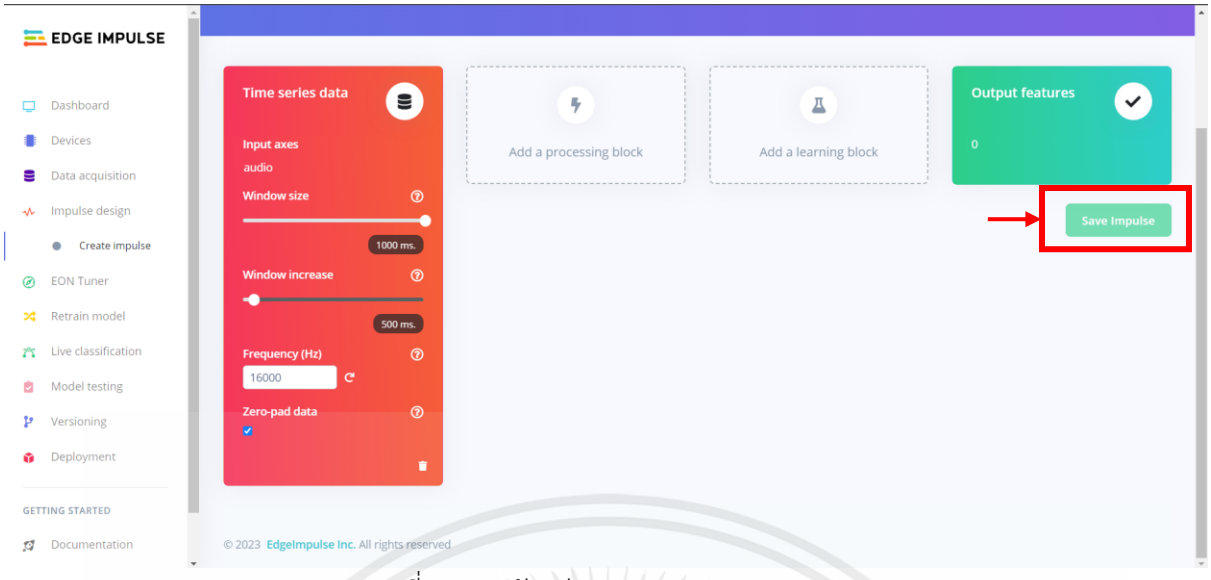


ภาพที่ 4.131 ตัวอย่างการ perform split (2)



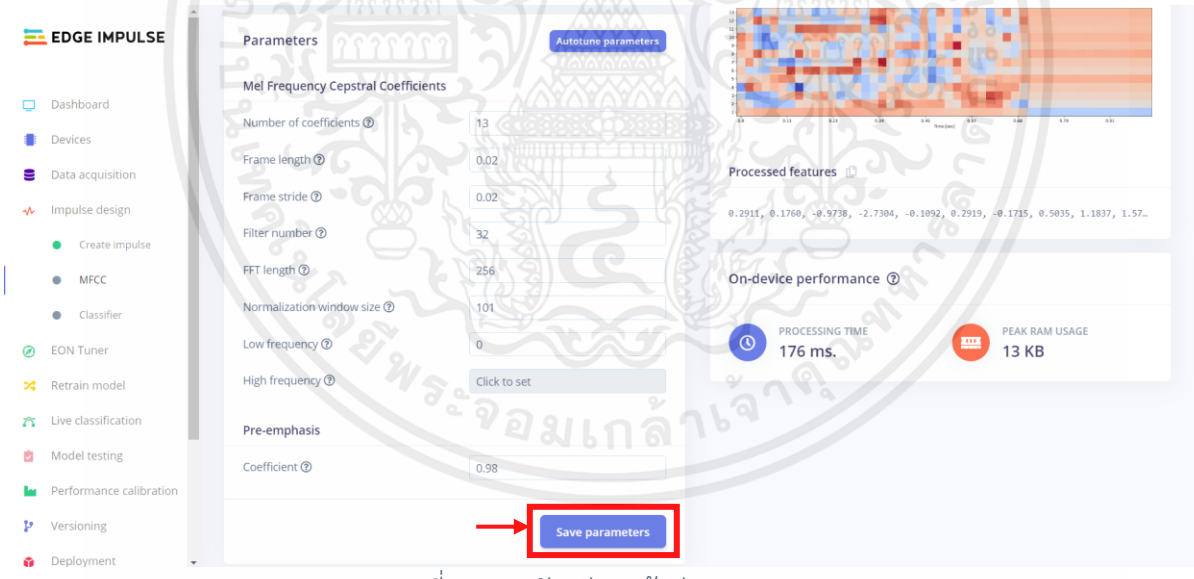
ภาพที่ 4.132 ตัวอย่างการ perform split (3)

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



ภาพที่ 4.133 ตัวอย่างการ perform split (4)

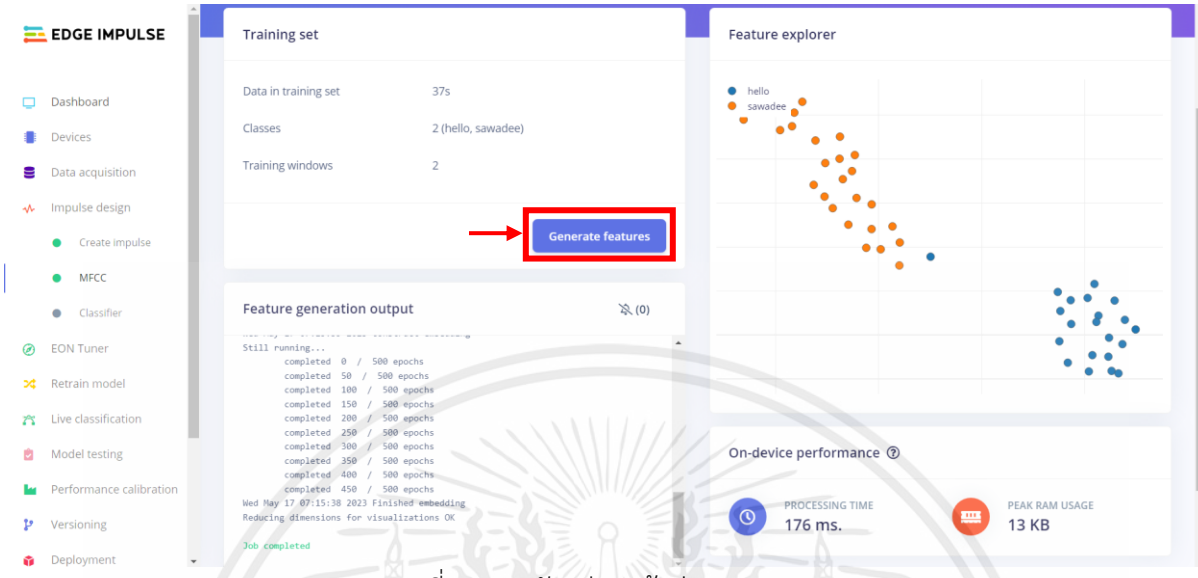
เมื่อกด Save impulse จะมีเมนู MFCC และ Classifier ขึ้นมาเข้าไปที่เมนู MFCC เลื่อนลงมาและกด Save parameters



ภาพที่ 4.134 ตัวอย่างหน้าต่าง MFCC (1)

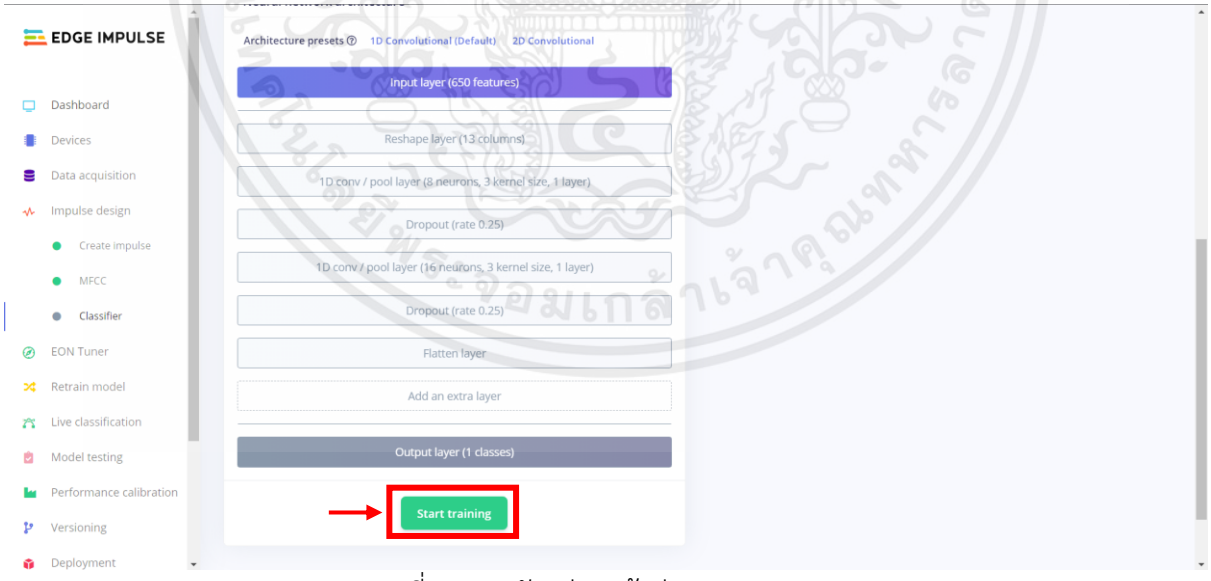
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

เมื่อกด Save impulse และจะแสดงหน้าต่าง Generates features กด Generate features



ภาพที่ 4.135 ตัวอย่างหน้าต่าง MFCC (2)

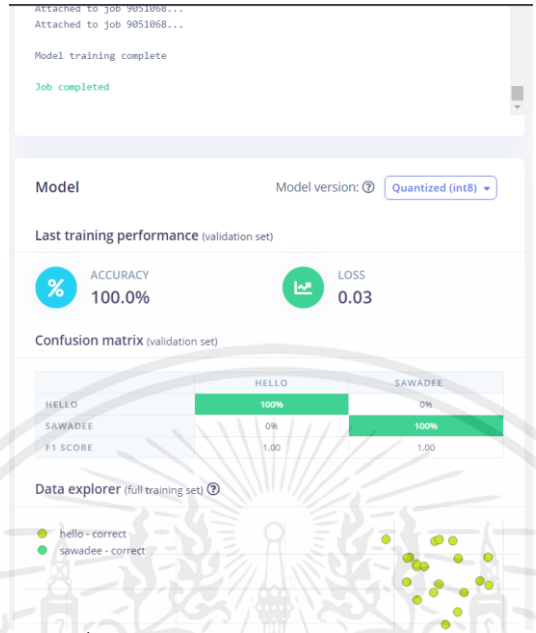
เข้ามาที่เมนู Classifier กด Start training เพื่อเริ่ม training model



ภาพที่ 4.136 ตัวอย่างหน้าต่าง Classifier (1)

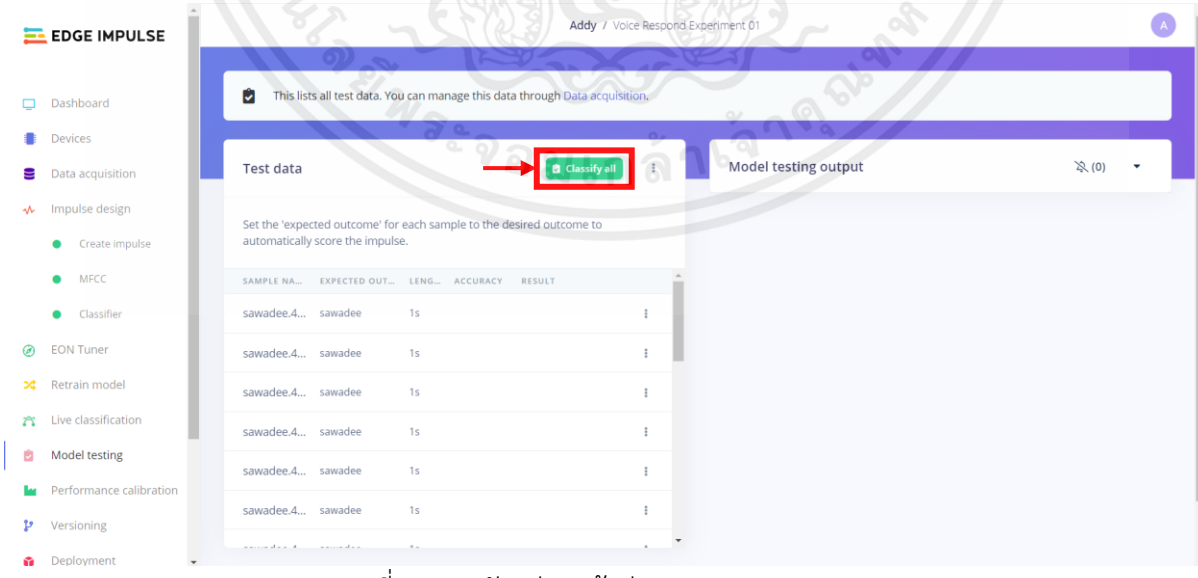
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

เมื่อ training model เสร็จจะขึ้น Job completed และแสดงผลลัพธ์ที่ได้จากกการ training ทั้ง Accuracy และ loss



ภาพที่ 4.137 ตัวอย่างหน้าต่าง Classifier (2)

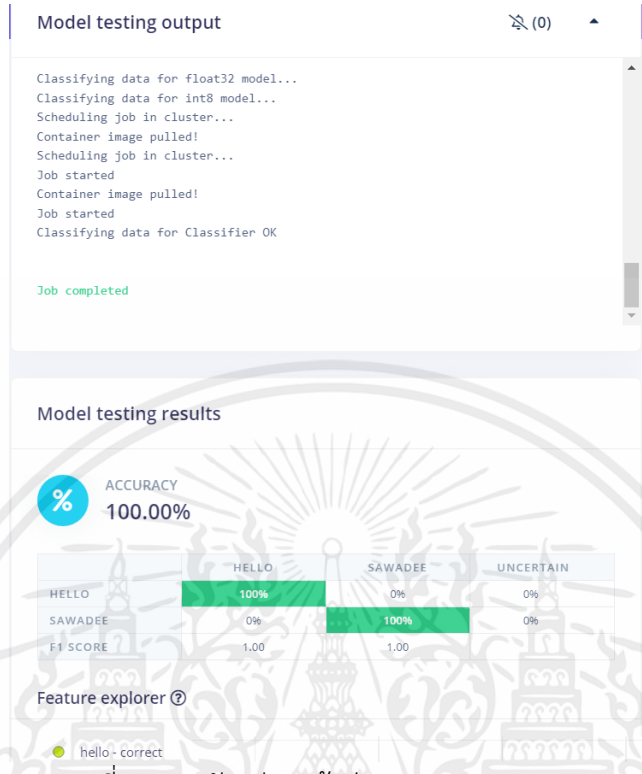
เพื่อทำการทดสอบว่า Model ที่ผ่านการ training มีประสิทธิภาพมากแค่ไหนให้เข้ามาที่เมนู Model testing เมื่อเข้ามาแล้วให้ทำการกด Classify all โปรแกรมจะทำการทดสอบโดยใช้ข้อมูลส่วน testing ที่ได้ทำการ Split ไปก่อนหน้านี้



ภาพที่ 4.138 ตัวอย่างหน้าต่าง Model testing (1)

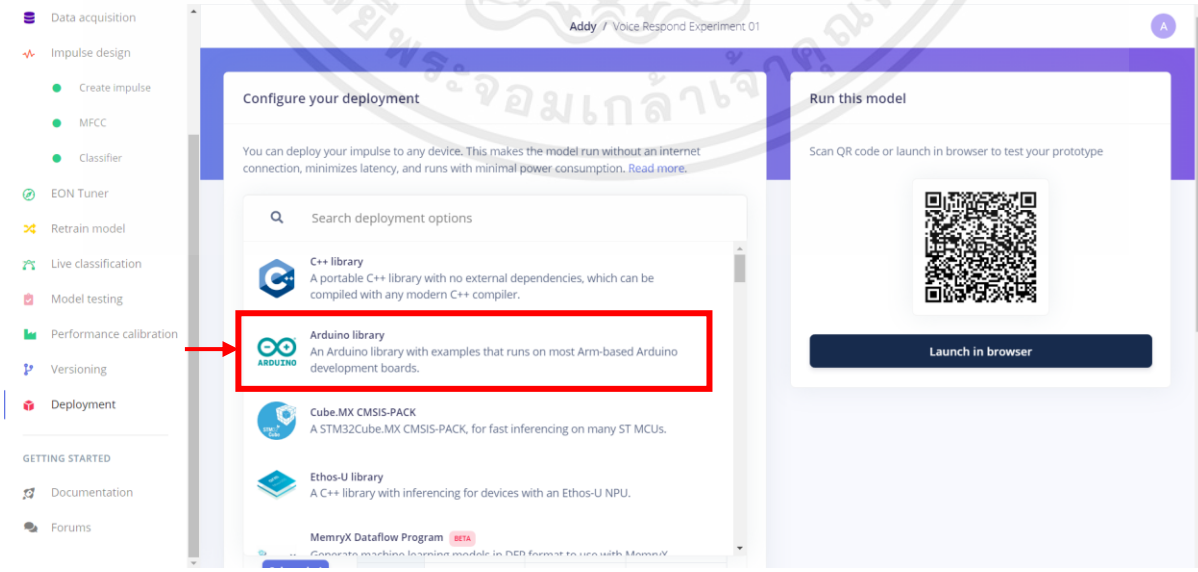
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

เมื่อทำการ testing เสร็จจะแสดงผลลัพธ์จากการ testing ออกมาเป็น Accuracy



ภาพที่ 4.139 ตัวอย่างหน้าต่าง Model testing (2)

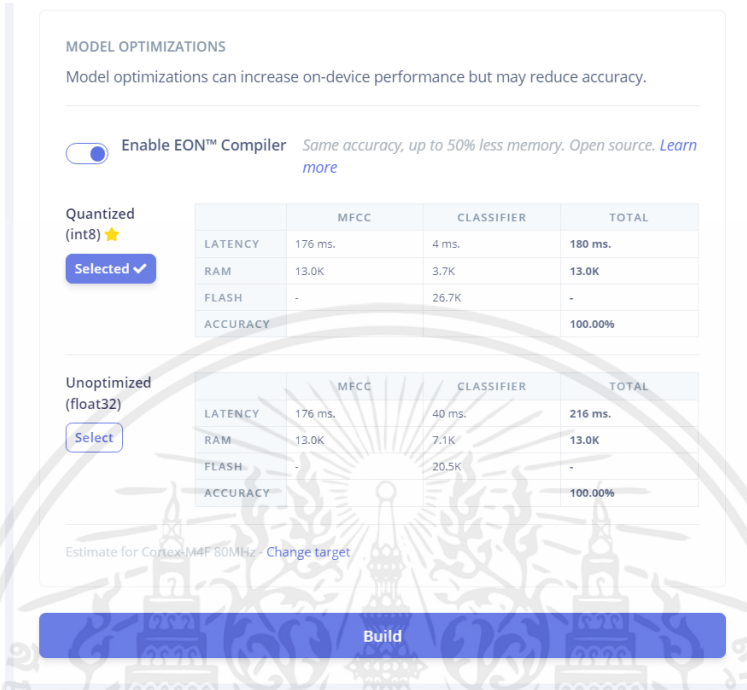
ต่อมาจะเป็นการนำ Model ที่ผ่านการ training ไป Deploy บนอุปกรณ์ที่ได้เตรียมไว้ เข้าไปที่เมนู Deployment และทำการค้นหา Arduino library



ภาพที่ 4.140 ตัวอย่างหน้าต่าง Deploymemnt (1)

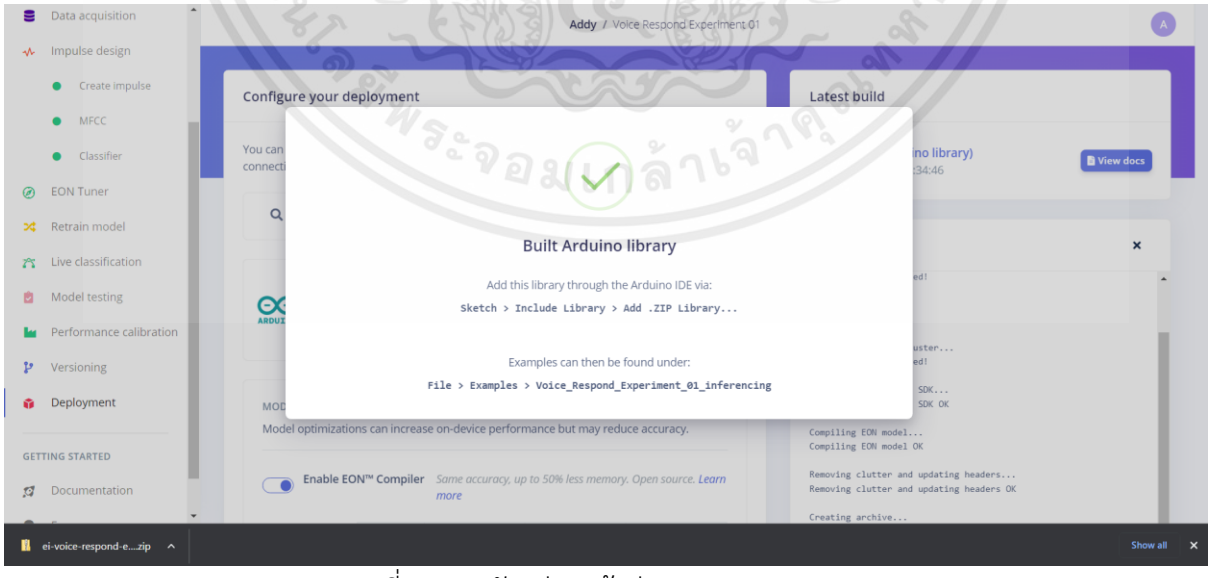
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

เมื่อเลือก Arduino library แล้วให้เลื่อนลงมาและกด Build เพื่อสร้าง library เป็น ZIP file



ภาพที่ 4.141 ตัวอย่างหน้าต่าง Deployment (2)

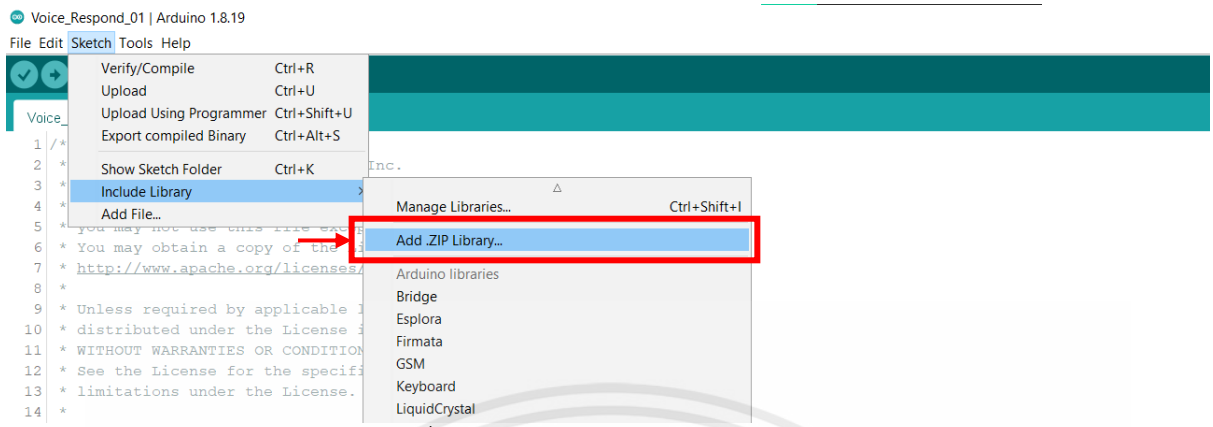
เมื่อ Build เสร็จแล้วจะทำการ download ZIP file โดยอัตโนมัติ



ภาพที่ 4.142 ตัวอย่างหน้าต่าง Deployment (3)

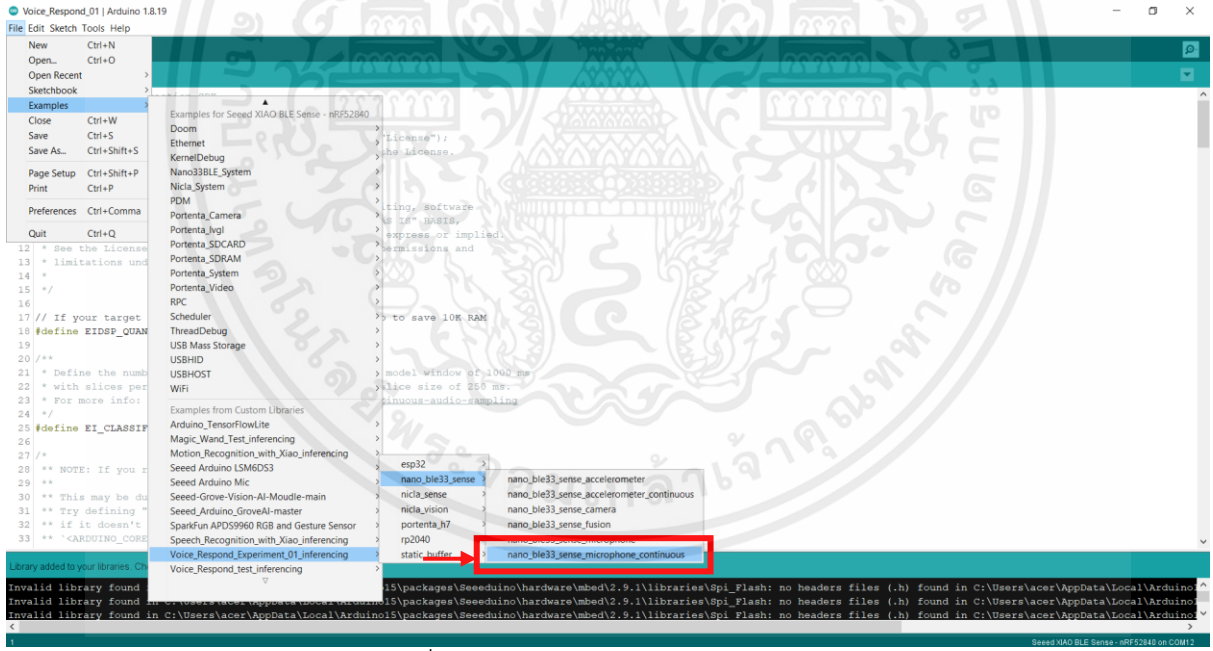
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

เปิดโปรแกรม Arduino ทำการเลือกเมนู Sketch และทำการ Include Library ด้วยการ Add .ZIP Library แล้วเลือกไฟล์ที่โหลดมา



ภาพที่ 4.143 ตัวอย่างการ Add Library (1)

Code ที่ต้องใช้จะอยู่ในไฟล์ Examples และอยู่ในส่วน Library ที่โหลดมา โดยเลือก nano_ble33_sense และ nano_ble33_sense_microphone_continous



ภาพที่ 4.144 ตัวอย่างการ Add Library (2)

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

เพื่อทำการทดสอบให้เพิ่ม Code ไปดังนี้ Code ซึ่งเป็น function เพื่อทำให้ LED ที่ติดตั้งอยู่บนบอร์ดแสดงผลตอนที่ได้ยินคำที่กำหนดไว้

โดยใน void set up() ให้เพิ่มการประกาศ pin output ไปดังนี้

```
pinMode(LED_BUILTIN, OUTPUT);
pinMode(LEDR, OUTPUT);
pinMode(LEDG, OUTPUT);
pinMode(LEDB, OUTPUT);
digitalWrite(LEDR, HIGH);
digitalWrite(LEDG, HIGH);
digitalWrite(LEDB, HIGH);
```

ใน void loop() ให้เปลี่ยน code ให้เป็นดังนี้

```
void loop()
{
    bool m = microphone_inference_record();
    if (!m) {
        ei_printf("ERR: Failed to record audio...\n");
        return;
    }

    signal_t signal;
    signal.total_length = EI_CLASSIFIER_SLICE_SIZE;
    signal.get_data = &microphone_audio_signal_get_data;
    ei_impulse_result_t result = {0};

    EI_IMPULSE_ERROR r = run_classifier_continuous(&signal, &result, debug_nn);
    if (r != EI_IMPULSE_OK) {
        ei_printf("ERR: Failed to run classifier (%d)\n", r);
        return;
    }
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

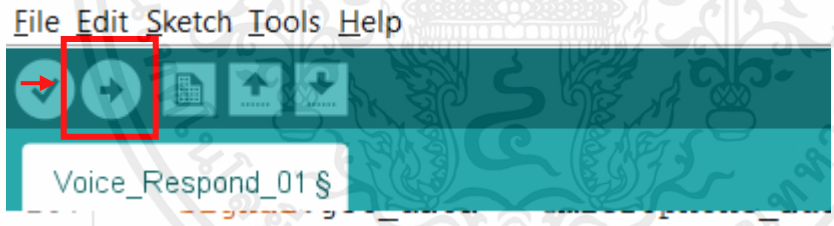
```
    if (++print_results >= (EI_CLASSIFIER_SLICES_PER_MODEL_WINDOW)) {
        // print the predictions
        ei_printf("Predictions ");
        ei_printf("(DSP: %d ms., Classification: %d ms., Anomaly: %d ms.)",
            result.timing.dsp, result.timing.classification, result.timing.anomaly);
        ei_printf(": \n");
        int pred_index = 2;      // added, Initialize pred_index
        float pred_value = 0.7;   // added, Initialize pred_value
        for (size_t ix = 0; ix < EI_CLASSIFIER_LABEL_COUNT; ix++) {
            ei_printf("    %s: %.5f\n", result.classification[ix].label,
                        result.classification[ix].value);
            if (result.classification[ix].value > pred_value)
        {
                pred_index = ix;
                 pred_value = result.classification[ix].value;



        }

          led(pred_index); //added, active function


        }

#if EI_CLASSIFIER_HAS_ANOMALY == 1
        ei_printf("    anomaly score: %.3f\n", result.anomaly);
#endif

        print_results = 0;
    }
}
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

และหลังบรรทัดสุดท้ายของ Code ทั้งหมดให้เพิ่ม function สำหรับออกคำสั่ง LED ลงไปดังนี้

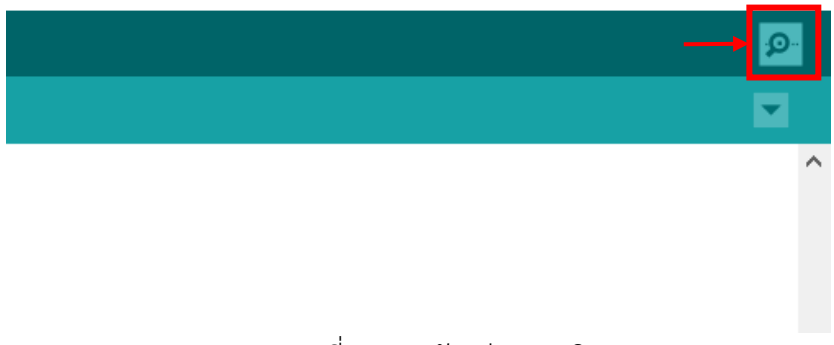
```
void led(int pred_index){
  switch (pred_index){
    case 0:
      digitalWrite(LEDR, LOW);
      digitalWrite(LEDG, HIGH);
      digitalWrite(LEDB, HIGH);
      break;

    case 1:
      digitalWrite(LEDR, HIGH);
      digitalWrite(LEDG, LOW);
      digitalWrite(LEDB, HIGH);
      break;
  }
}
```

ทำการ Upload Code ลงบอร์ดโดยกดที่ปุ่ม Upload



ภาพที่ 4.145 ตัวอย่างการ Upload Code

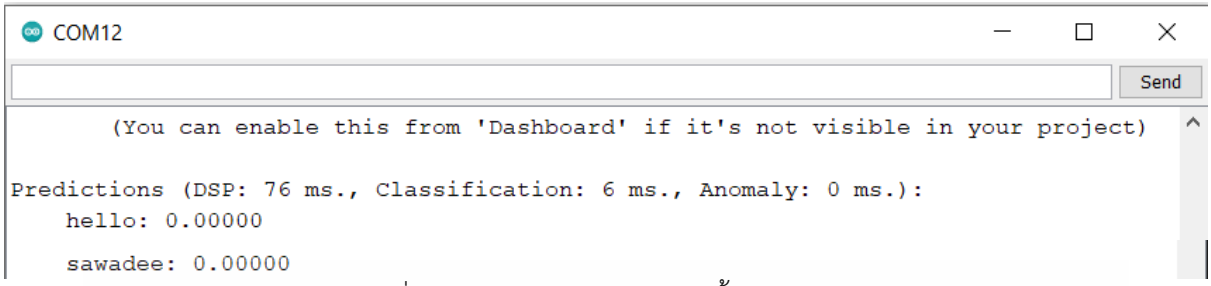
เมื่อ Upload เสร็จกดที่ Serial Monitor เพื่อดูข้อมูลที่ส่งมาทาง port



ภาพที่ 4.146 ตัวอย่างการเปิด Serial port

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

โดยใน Serial Monitor จะแสดงถึงผลลัพธ์การคาดเดาเสียงที่ได้ยินของ Model ประกอบกับบอร์ดจะแสดง LED เป็นสีแดงเมื่อได้ยินคำว่า Hello และสีเขียวเมื่อได้ยินคำว่า สวัสดี



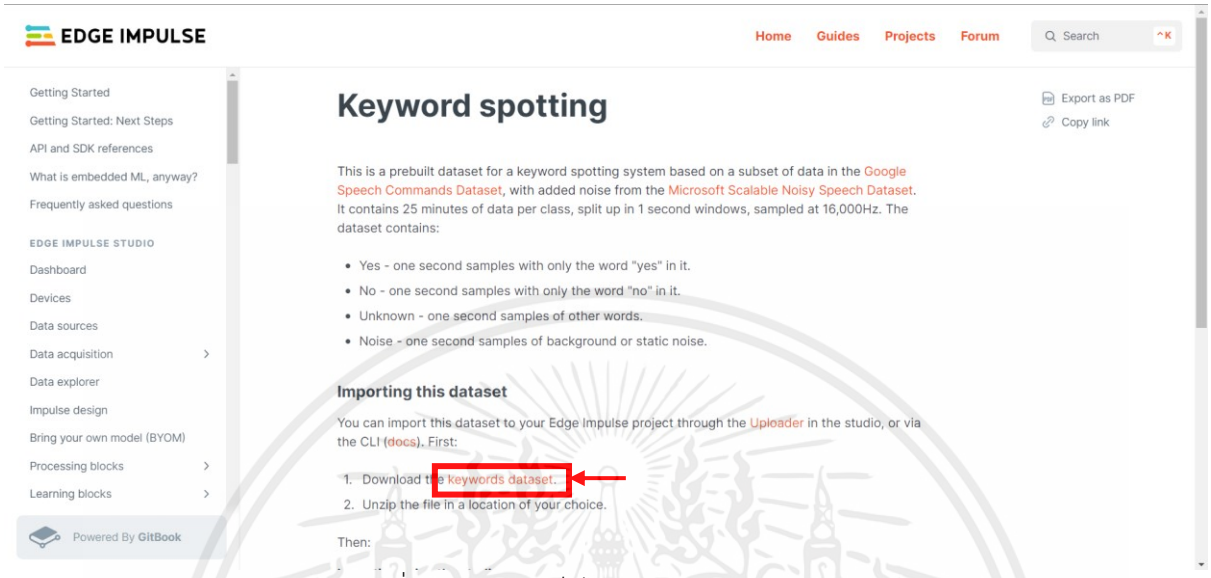
ภาพที่ 4.147 ตัวอย่างผลลัพธ์การชี้คำเฉพาะ (1)

ให้ผู้ศึกษาทำการทดลองดูว่าในการทดสอบพูดคำที่ต้องการ 10 ครั้งมีทั้งหมดกี่ครั้งที่บอร์ดสามารถแสดงผล LED ได้ตรงตามที่ต้องการ

คำถาม : การแสดงผลจาก LED สามารถแสดงผลได้ถูกต้องทั้งหมดหรือไม่ ถ้าไม่ผู้ศึกษาคิดว่าเป็นเพราะเหตุใด หรือมีอะไรส่งผลต่อการคาดเดาคำพูดของโปรแกรมบ้าง?

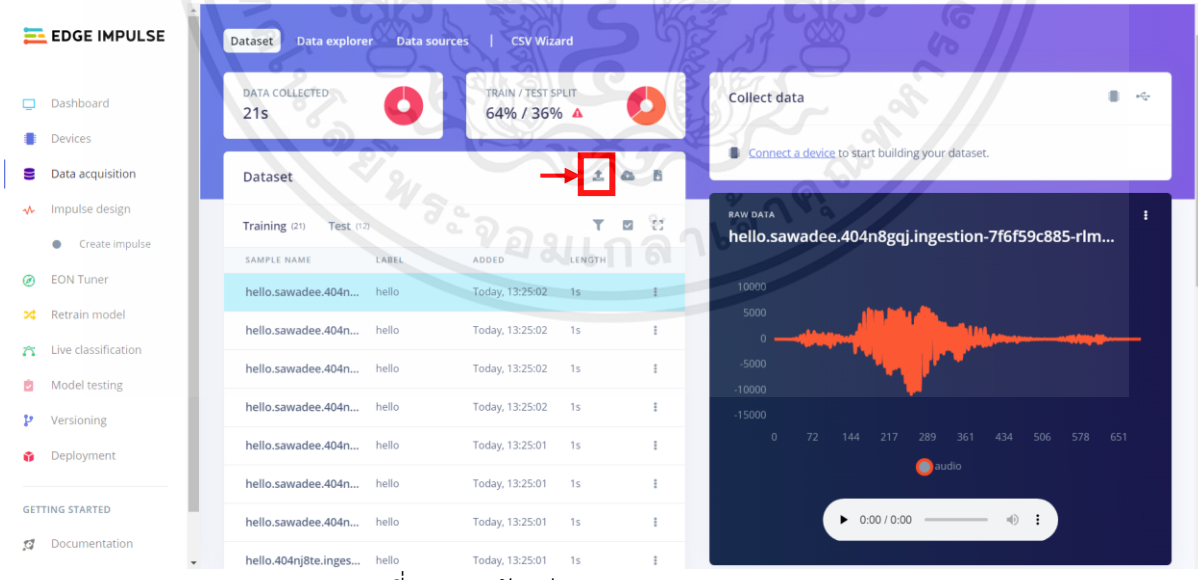
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ต่อมาให้ผู้ศึกษาทดลองเพิ่ม label ของเสียงรบกวนเข้าไปใน dataset ที่ได้ทำไว้ โดยทำการ download dataset จากลิงค์ : https://docs.edgeimpulse.com/docs/pre-built-datasets/keyword-spotting



ภาพที่ 4.148 หน้าเว็บไซต์สำหรับโหลด dataset

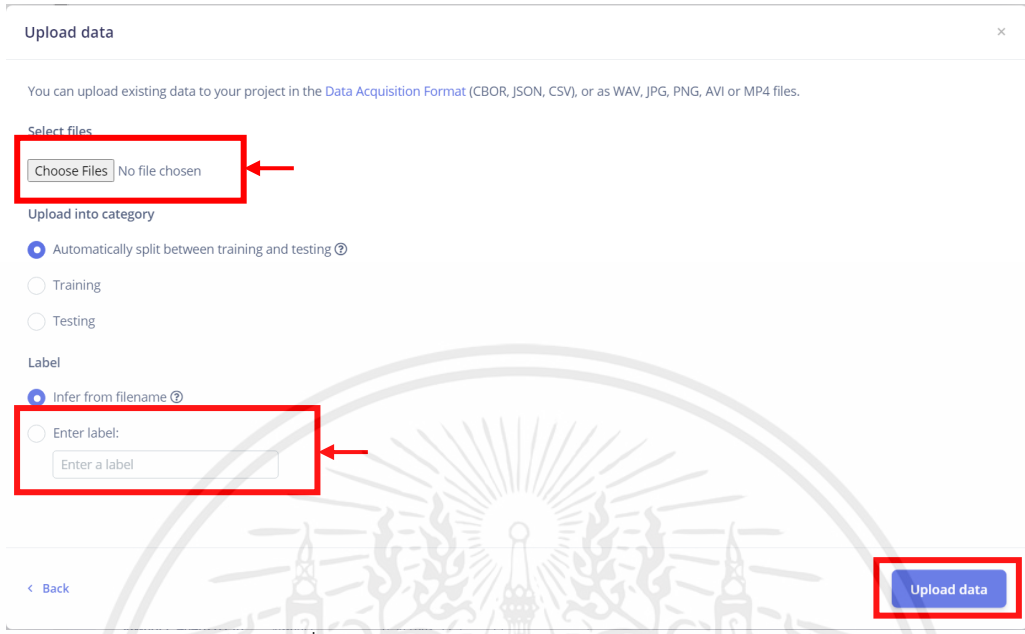
เมื่อ download แล้วทำการแตกไฟล์จะได้ไฟล์ keyword ให้ดูในโฟลเดอร์ที่ชื่อ noise จะมีไฟล์เสียงรบกวนอยู่จำนวนหนึ่ง โดยให้ทำการ Upload ไฟล์เหล่านั้นไปที่ dataset ของเราโดยกดที่ปุ่ม Upload ในหน้า Data acquisition



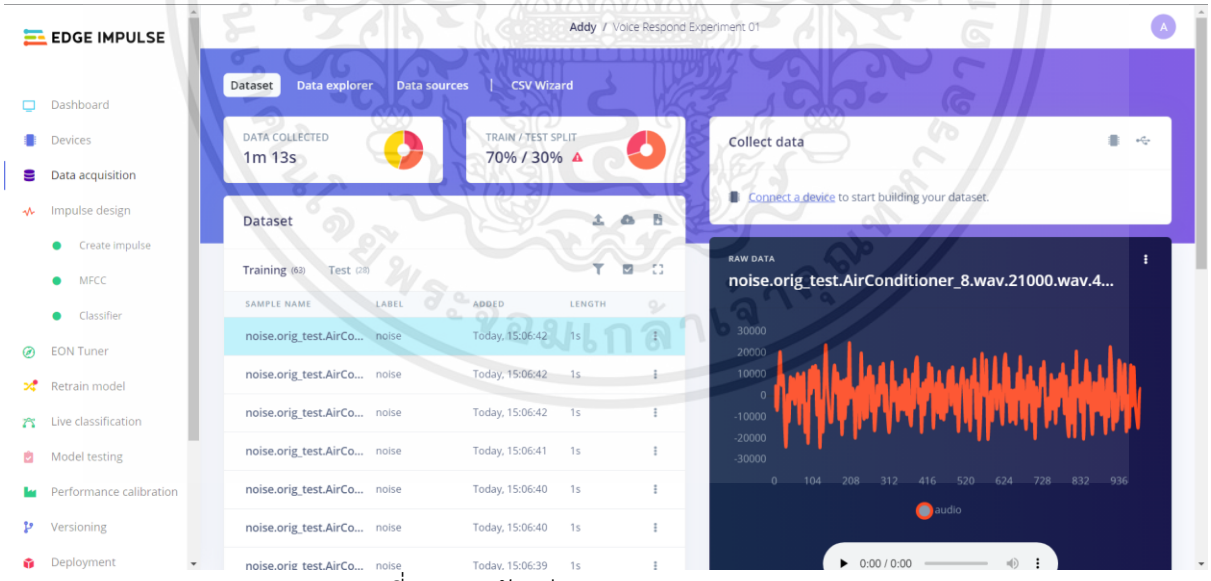
ภาพที่ 4.149 ตัวอย่างการ Upload dataset (1)

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ตั้งชื่อ label ว่า noise แล้วกด Choose Files แล้วเลือกไฟล์เสียงในโฟลเดอร์ noise มาประมาณ 30-40 ไฟล์ เมื่อประมวลผลเสร็จแล้วให้ทำการกด Upload data เพื่อ Upload ไฟล์



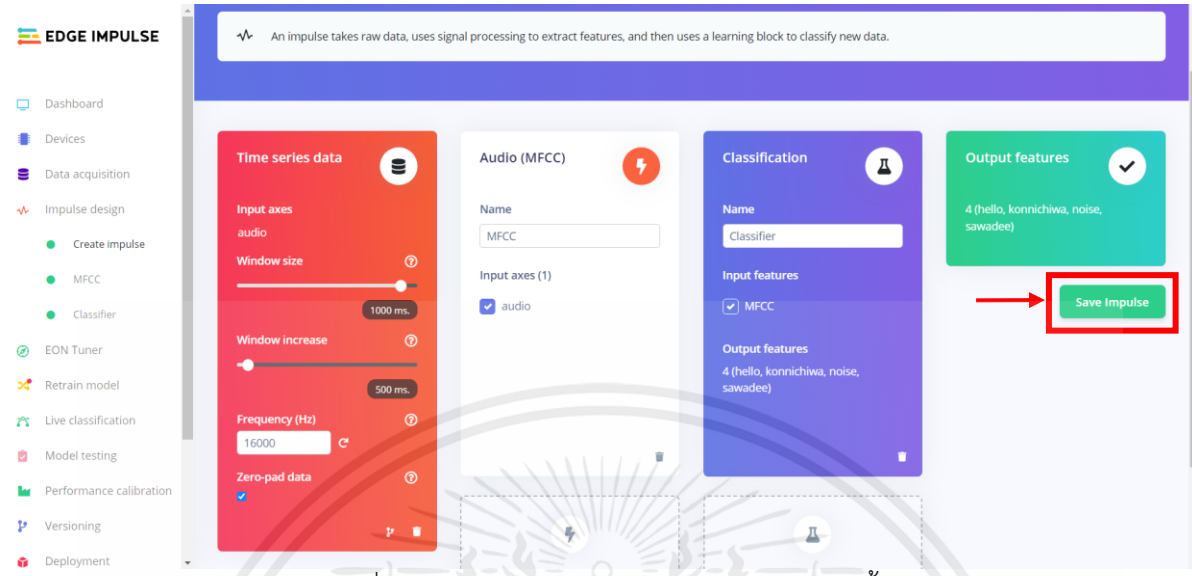
ภาพที่ 4.150 ตัวอย่างการ Upload dataset (2)



ภาพที่ 4.151 ตัวอย่างการ Upload dataset (3)

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

จากนั้นให้ทำตามขั้นตอนตั้งแต่การ Create impulse อีกครั้งโดยครั้งนี้สามารถกด Save parameters โดยใช้การตั้งค่าเดิมได้เลย

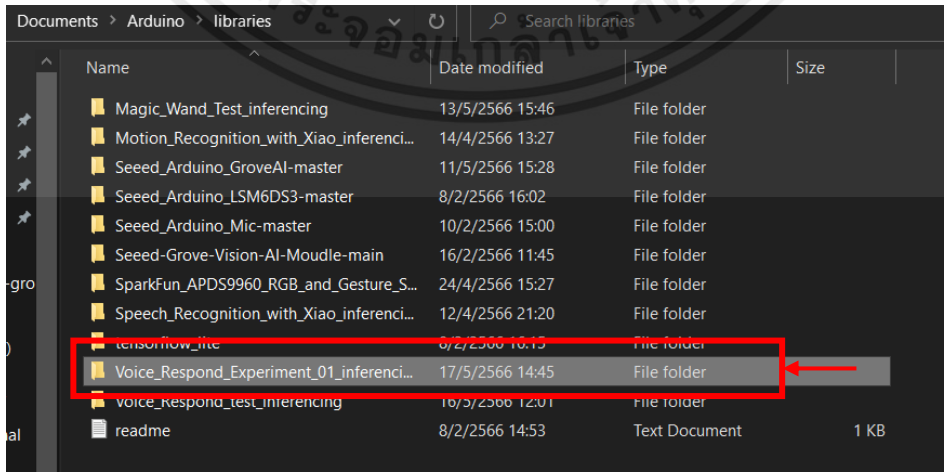


ภาพที่ 4.152 ตัวอย่างการ Create impulse อีกครั้ง

จากนั้นให้ทำตามขั้นตอนเดิมตั้งแต่

- Save Parameters
- Generates features
- Training Model
- Deployment

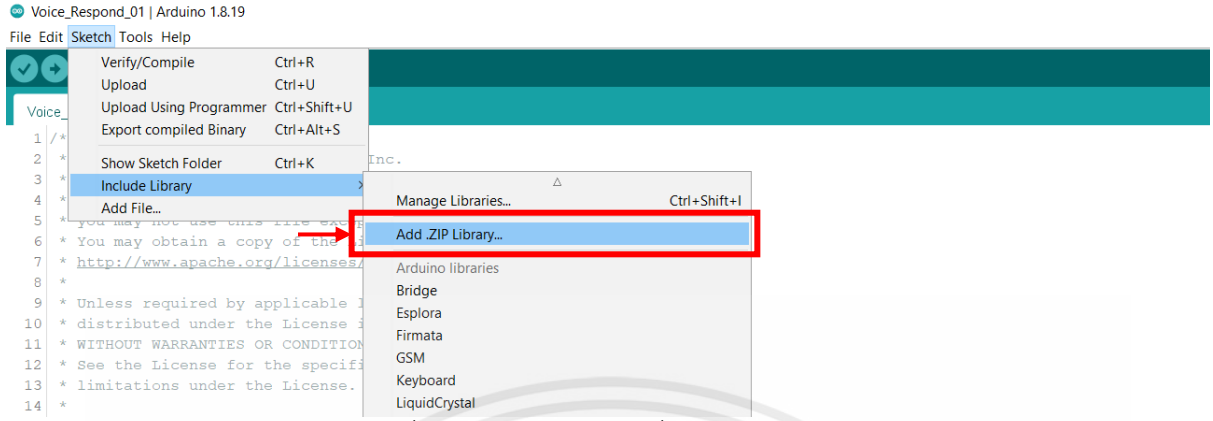
จนได้ไฟล์เพื่อ Upload ลงบอร์ด โดยก่อนที่จะทำการ Add .ZIP library ให้ทำการลบไฟล์ library เก่าที่มีอยู่ออกไปก่อน โดยไฟล์ library จะอยู่ในโฟลเดอร์ที่เก็บ Arduino ไว้ (ส่วนมากจะเป็น document) Arduino > libraries



ภาพที่ 4.153 ตัวอย่างการลบ library

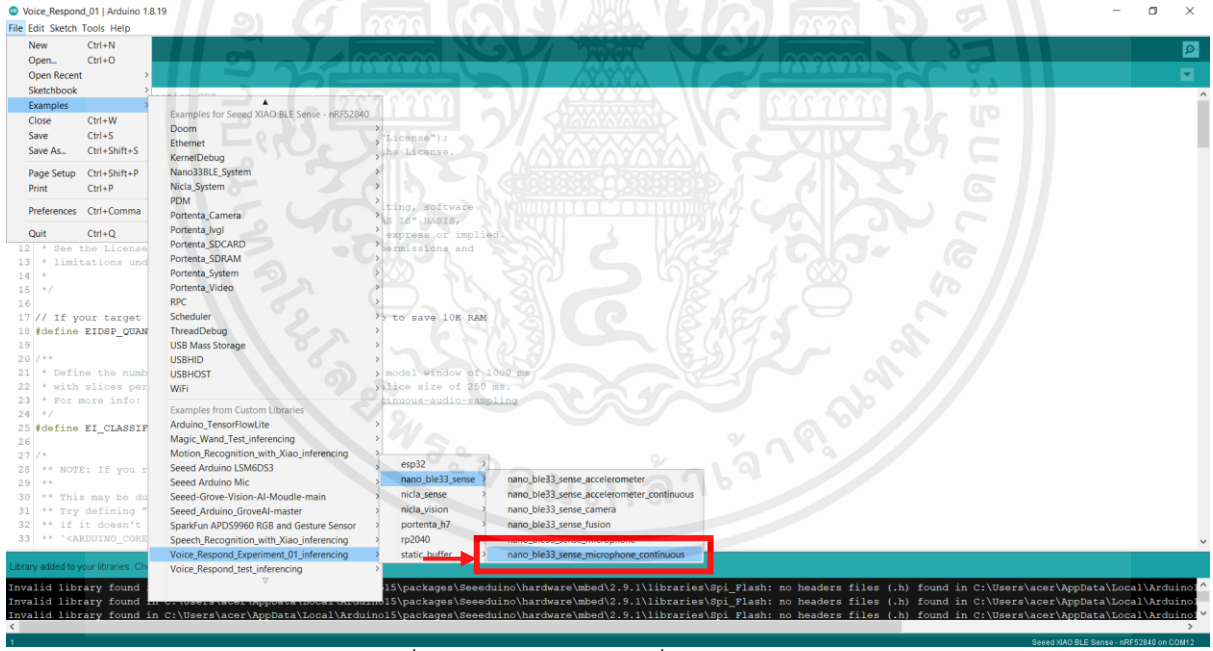
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

เปิดโปรแกรม Arduino เพื่อเพิ่ม library เหมือนในขั้นตอนที่ผ่านมาทำการเลือกเมนู Sketch และทำการ Include Library ด้วยการ Add .ZIP Library แล้วเลือกไฟล์ที่โหลดมา



ภาพที่ 4.154 ตัวอย่างการเพิ่ม library ใหม่ (1)

Code ที่ต้องใช้จะอยู่ในไฟล์ Examples และอยู่ในส่วน Library ที่โหลดมา โดยเลือก nano_ble33_sense และ nano_ble33_sense_microphone_continous



ภาพที่ 4.155 ตัวอย่างการเพิ่ม library ใหม่ (2)

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

เพื่อทำการทดสอบให้เพิ่ม Code ไปดังนี้ Code ซึ่งเป็น function เพื่อทำให้ LED ที่ติดตั้งอยู่บนบอร์ดแสดงผลตอนที่ได้ยินคำที่กำหนดไว้

โดยใน void set up() ให้เพิ่มการประกาศ pin output ไปดังนี้

```
pinMode(LED_BUILTIN, OUTPUT);
pinMode(LEDR, OUTPUT);
pinMode(LEDG, OUTPUT);
pinMode(LEDB, OUTPUT);
digitalWrite(LEDR, HIGH);
digitalWrite(LEDG, HIGH);
digitalWrite(LEDB, HIGH);
```

ใน void loop() ให้เปลี่ยน code ให้เป็นดังนี้

```
void loop()
{
    bool m = microphone_inference_record();
    if (!m) {
        ei_printf("ERR: Failed to record audio...\n");
        return;
    }

    signal_t signal;
    signal.total_length = EI_CLASSIFIER_SLICE_SIZE;
    signal.get_data = &microphone_audio_signal_get_data;
    ei_impulse_result_t result = {0};

    EI_IMPULSE_ERROR r = run_classifier_continuous(&signal, &result, debug_nn);
    if (r != EI_IMPULSE_OK) {
        ei_printf("ERR: Failed to run classifier (%d)\n", r);
        return;
    }
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
    if (++print_results >= (EI_CLASSIFIER_SLICES_PER_MODEL_WINDOW)) {
        // print the predictions
        ei_printf("Predictions ");
        ei_printf("(DSP: %d ms., Classification: %d ms., Anomaly: %d ms.)",
            result.timing.dsp, result.timing.classification, result.timing.anomaly);
        ei_printf(": \n");
        int pred_index = 2;      // added, Initialize pred_index
        float pred_value = 0.7;   // added, Initialize pred_value
        for (size_t ix = 0; ix < EI_CLASSIFIER_LABEL_COUNT; ix++) {
            ei_printf("    %s: %.5f\n", result.classification[ix].label,
                        result.classification[ix].value);
            if (result.classification[ix].value > pred_value)
        {
                pred_index = ix;
                 pred_value = result.classification[ix].value;



        }

          led(pred_index); //added, active function


        }

#if EI_CLASSIFIER_HAS_ANOMALY == 1
        ei_printf("    anomaly score: %.3f\n", result.anomaly);
#endif

        print_results = 0;
    }
}
```

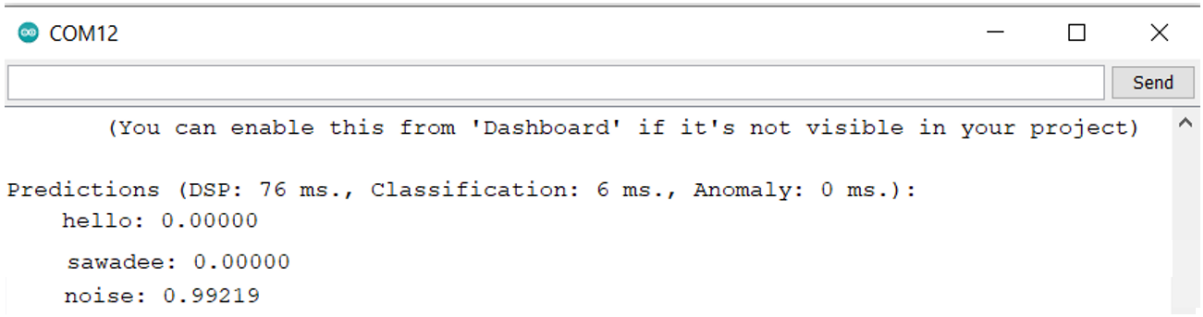
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

และหลังบรรทัดสุดท้ายของ Code ทั้งหมดให้เพิ่ม function สำหรับออกคำสั่ง LED โดยครั้งนี้จะเพิ่มส่วนของ noise โดยจะไม่ให้มีอะไรเกิดขึ้นเมื่ออุปกรณ์สามารถได้ยิน noise ได้

```
void led(int pred_index){
  switch (pred_index){
    case 0:
      digitalWrite(LEDR, LOW);
      digitalWrite(LEDG, HIGH);
      digitalWrite(LEDB, HIGH);
      break;

    case 1:
      digitalWrite(LEDR, HIGH);
      digitalWrite(LEDG, LOW);
      digitalWrite(LEDB, HIGH);
      break;

   case 2: // environment voice
      digitalWrite(LEDR, HIGH);
      digitalWrite(LEDG, HIGH);
      digitalWrite(LEDB, HIGH);
      break;
  }
}
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

เมื่อดูผลลัพธ์จาก Serial Monitor จะเห็นได้ว่ามี noise เพิ่มขึ้นมา



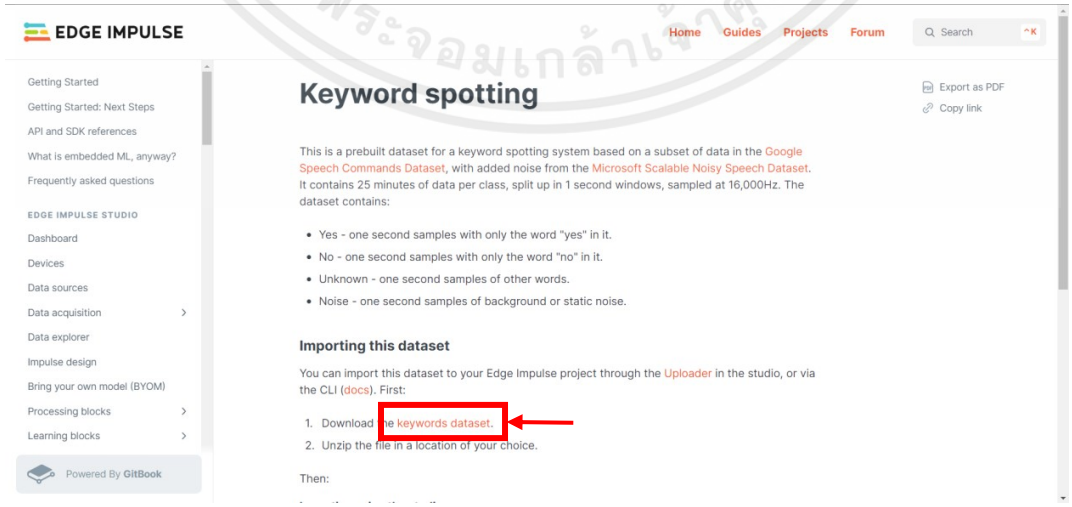
ภาพที่ 4.156 ตัวอย่างผลลัพธ์การชี้คำเฉพาะ (2)

ให้ผู้ศึกษาทำการทดลองดูว่าในการทดสอบพูดคำที่ต้องการ 10 ครั้งมีทั้งหมดกี่ครั้งที่บอร์ดสามารถแสดงผล LED ได้ตรงตามที่ต้องการ

คำถาม : การแสดงผลจาก LED สามารถแสดงผลได้ถูกต้องทั้งหมดหรือไม่ การที่เพิ่ม dataset noise ส่งผลต่อผลลัพธ์ทั้งหมดหรือไม่ อย่างไร?

การทดลอง 1 : ให้ผู้ศึกษาทำการเก็บข้อมูลคำใดก็ได้ตามใจชอบไปอีกคำหนึ่งนอกจากคำว่า Hello และ สวัสดีที่ได้ทดลองไปในขั้นตอนก่อน ๆ พร้อมทั้งเพิ่ม Code ที่จะทำให้สามารถแสดงผล LED สีฟ้าขึ้นมาเมื่อได้ยินคำที่เพิ่มขึ้นมา และบันทึกผลการทดลองว่าเมื่อทดสอบพูดคำที่ต้องการ 10 ครั้งมีทั้งหมดกี่ครั้งที่บอร์ดสามารถแสดงผล LED ได้ตรงตามที่ต้องการ

การทดลอง 2 : ให้ผู้ศึกษาใช้ Dataset ที่ได้มาจากไฟล์ keyword มาทำการทดลองโดยใช้คำ yes no และ noise ที่มีอยู่ในไฟล์และสร้างโปรแกรมขึ้นมาดังที่ได้ทำไปก่อนหน้า และบันทึกผลการทดลองว่าเมื่อทดสอบพูดคำที่ต้องการ 10 ครั้งมีทั้งหมดกี่ครั้งที่บอร์ดสามารถแสดงผล LED ได้ตรงตามที่ต้องการ พร้อมเปรียบเทียบว่ามีผลลัพธ์ต่างจากการทดลองที่ 1 อย่างไร



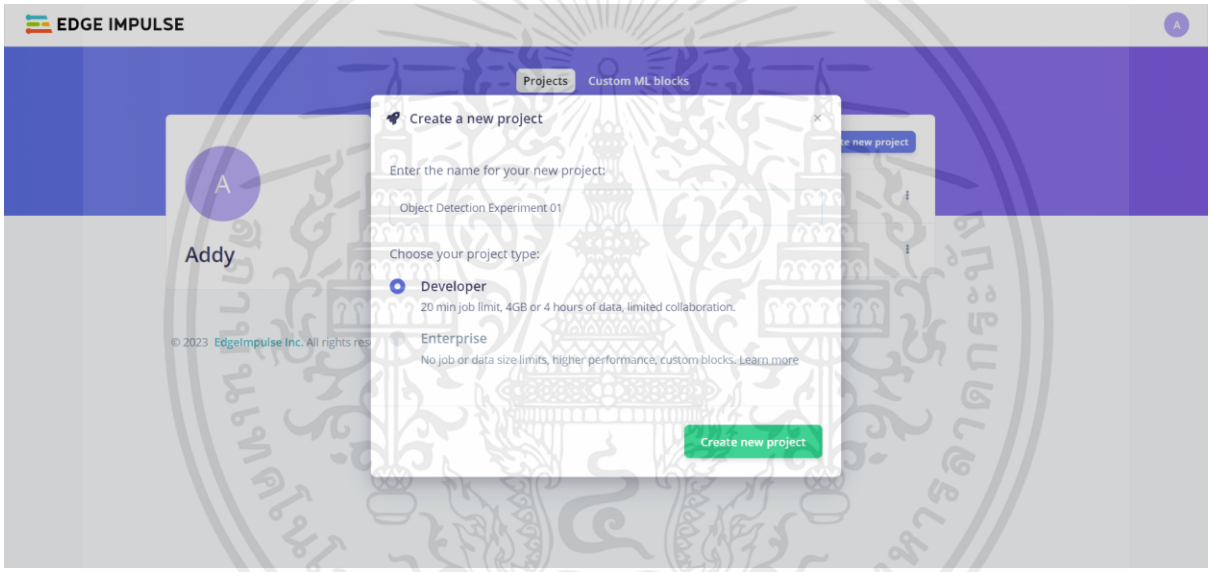
ภาพที่ 4.157 ภาพประกอบการทดลอง

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

การทดลอง 3 : ให้ผู้ศึกษาทดลองเก็บ dataset ด้วยคำใดก็ได้ตามใจชอบ หรือจะหา dataset จากเว็บไซต์อื่น ๆ เช่น Kaggle มาสร้างโปรแกรมตรวจจับเสียงในแบบของผู้ศึกษาเองอย่างน้อย 2 คำ

2.Object Detection (การทดสอบตรวจจับภาพวัตถุด้วยการเรียนรู้ของเครื่องบนไมโครคอนโทรลเลอร์)

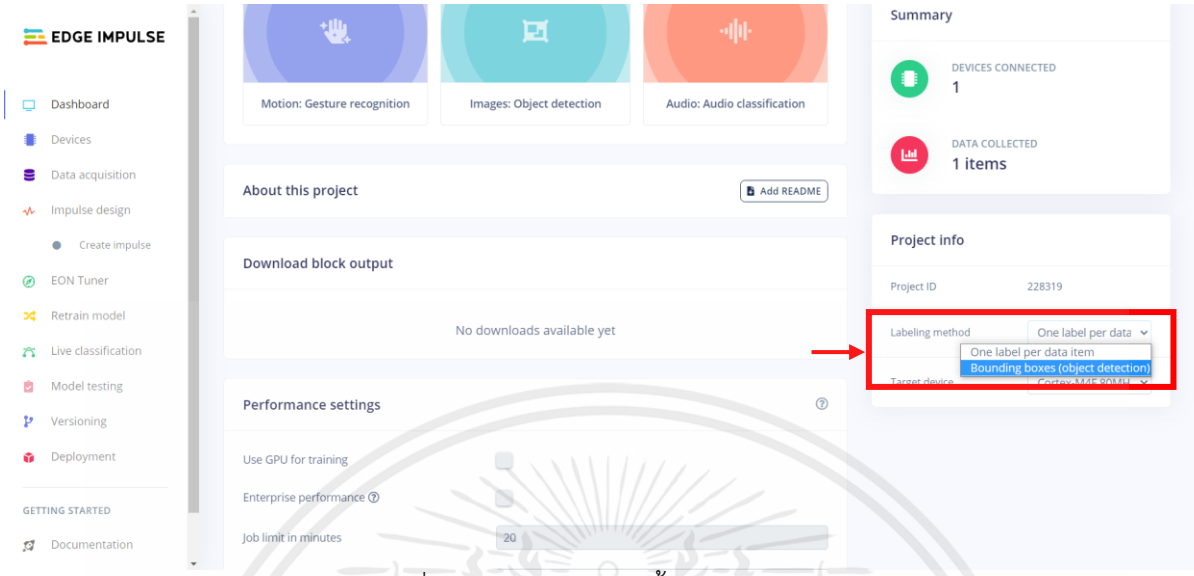
ต่อจากการทดลองที่ 1 ให้สร้างโปรเจคใหม่ใน edge impulse ตั้งชื่อว่า Object Detection Experiment 01



ภาพที่ 4.158 ตัวอย่างการสร้างโปรเจคใหม่

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ในหน้า Dashboard ให้เลื่อนไปที่ Project info และตั้งค่า labeling method ให้เป็น Bounding boxes(object detection)

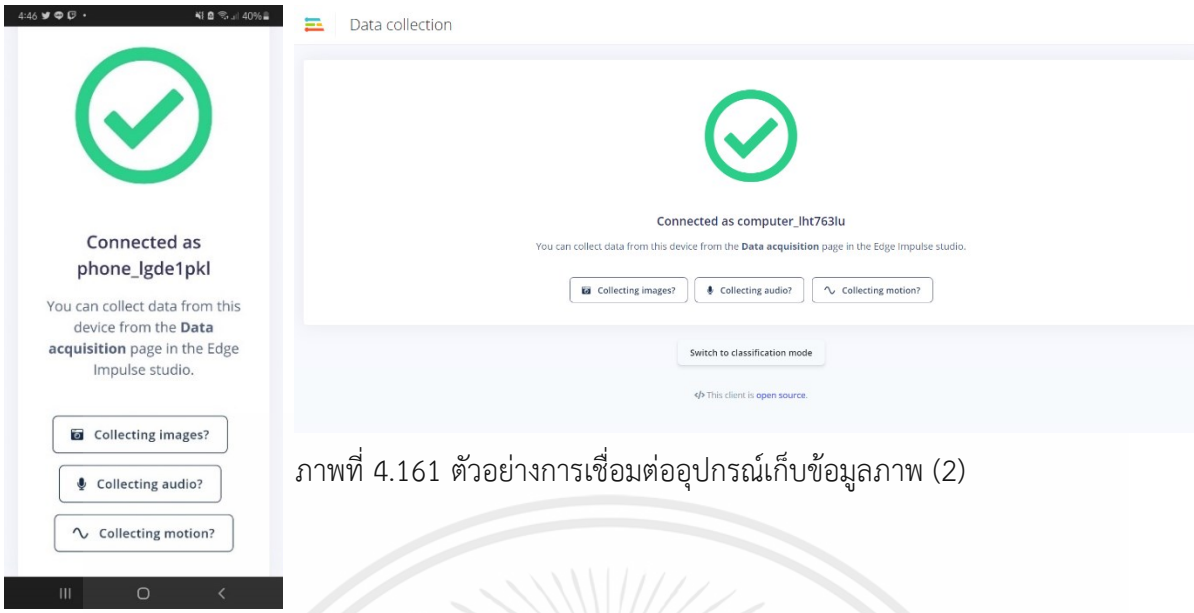


ภาพที่ 4.159 ตัวอย่างการตั้งค่า Dashboard

ในหน้า Devices ให้ทำการเชื่อมต่ออุปกรณ์คอมพิวเคอร์ที่มี webcam หรือโทรศัพท์มือถือ

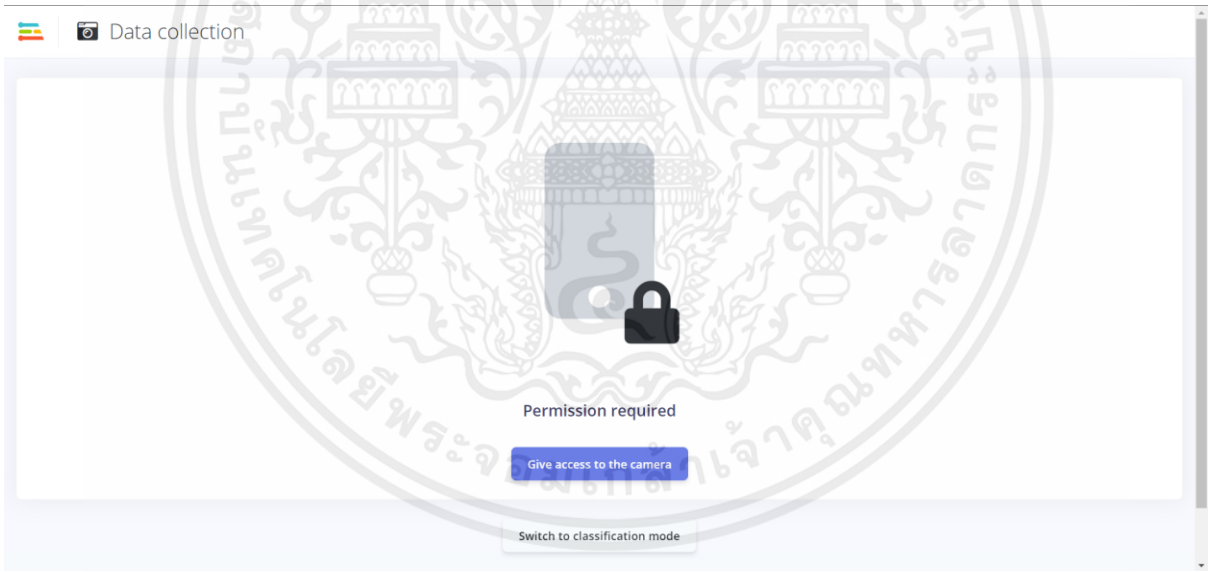
ภาพที่ 4.160 ตัวอย่างการเชื่อมต่ออุปกรณ์เก็บข้อมูลภาพ (1)

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



ภาพที่ 4.161 ตัวอย่างการเชื่อมต่ออุปกรณ์เก็บข้อมูลภาพ (2)

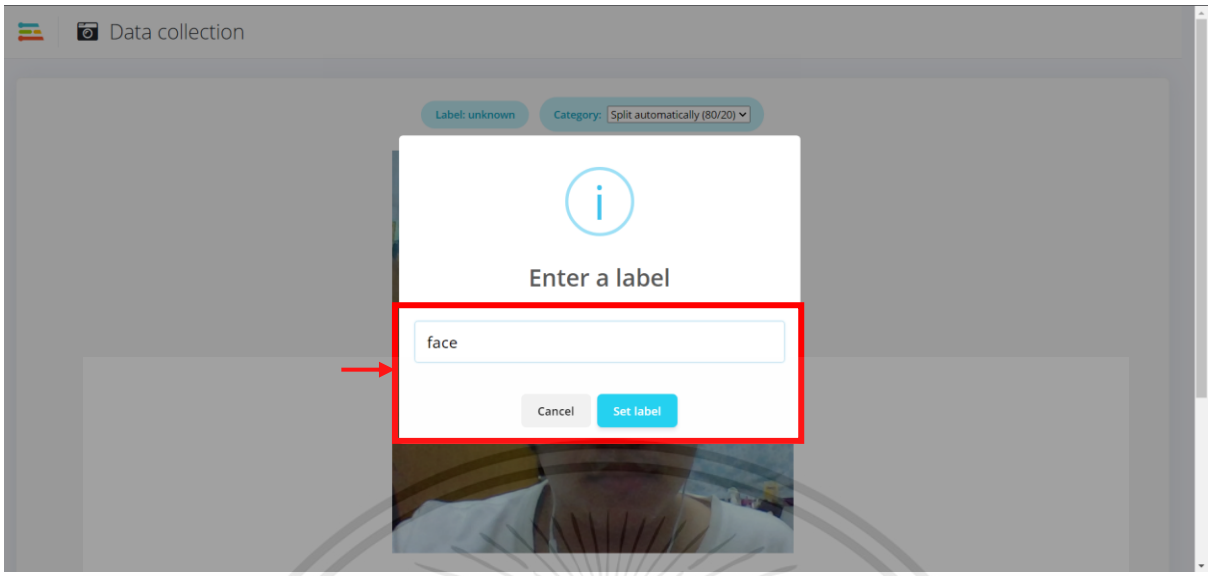
เมื่อเชื่อมต่อเรียบร้อยแล้วให้กด Collecting image? และให้ access permission



ภาพที่ 4.162 ตัวอย่างการเชื่อมต่ออุปกรณ์เก็บข้อมูลภาพ (3)

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ตั้งชื่อ label ว่า face และทำการ Capture รูปหน้าของตัวเอง 30-40 รูป

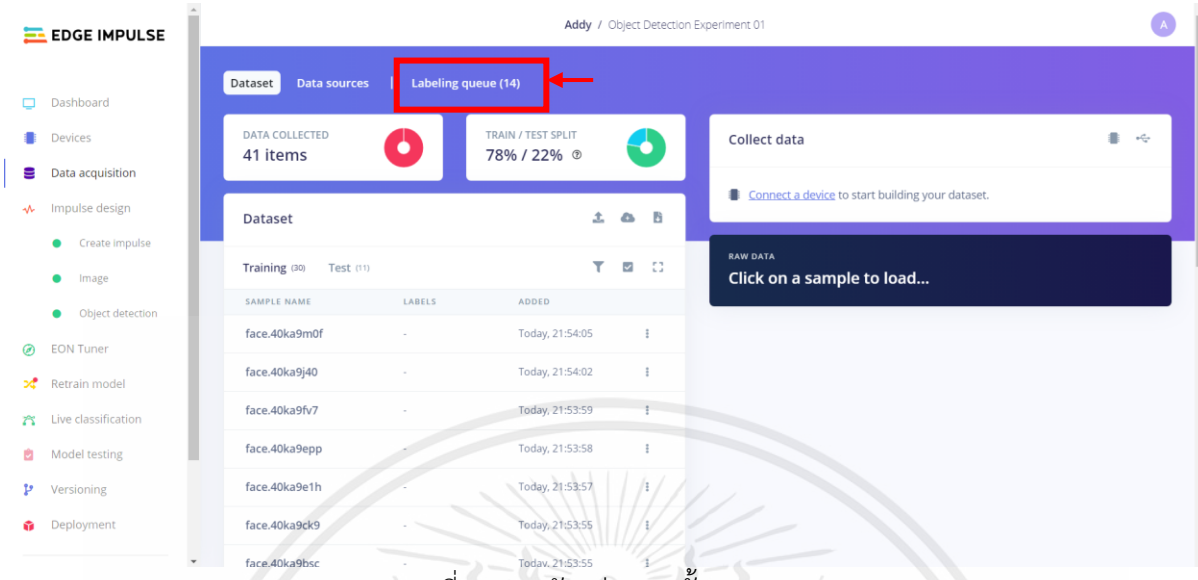


ภาพที่ 4.163 ตัวอย่างการเก็บข้อมูลภาพ (1)

ภาพที่ 4.164 ตัวอย่างการเก็บข้อมูลภาพ (2)

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

เมื่อเก็บข้อมูลเรียบร้อยแล้วในหน้าเมนู Data acquisition ให้เข้าไปยังหน้าต่าง Labeling queue



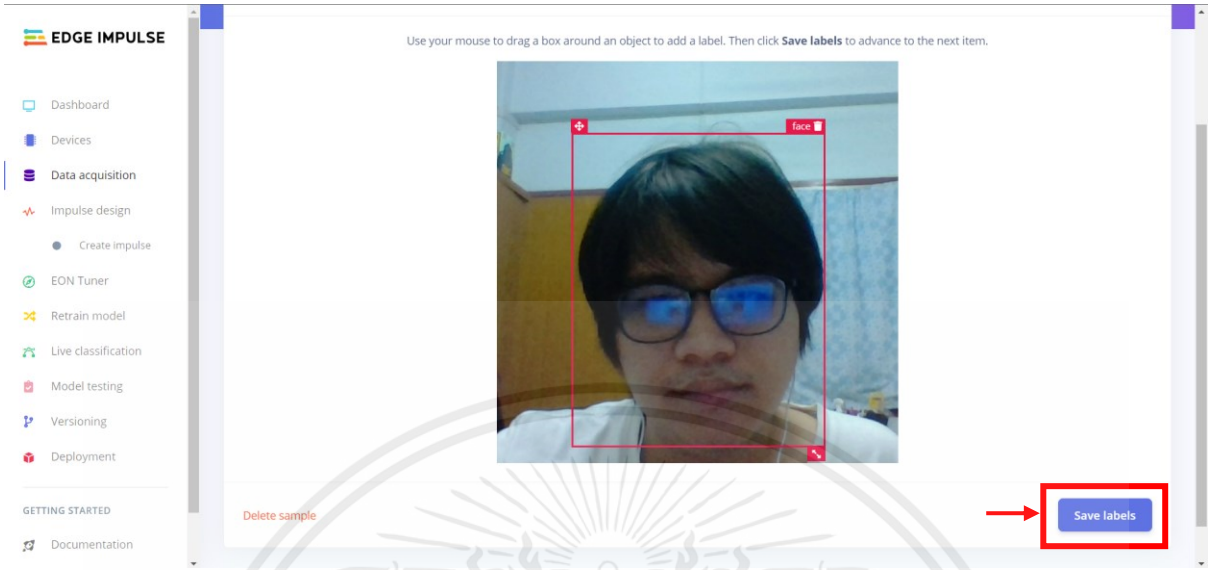
ภาพที่ 4.165 ตัวอย่างการตั้ง label (1)

โดยทำการลากกรอกสี่เหลี่ยมเพื่อกำหนดขอบเขตที่ต้องการให้ระบุเป็นใบหน้าและตั้งชื่อ label ว่า face

ภาพที่ 4.166 ตัวอย่างการตั้ง label (2)

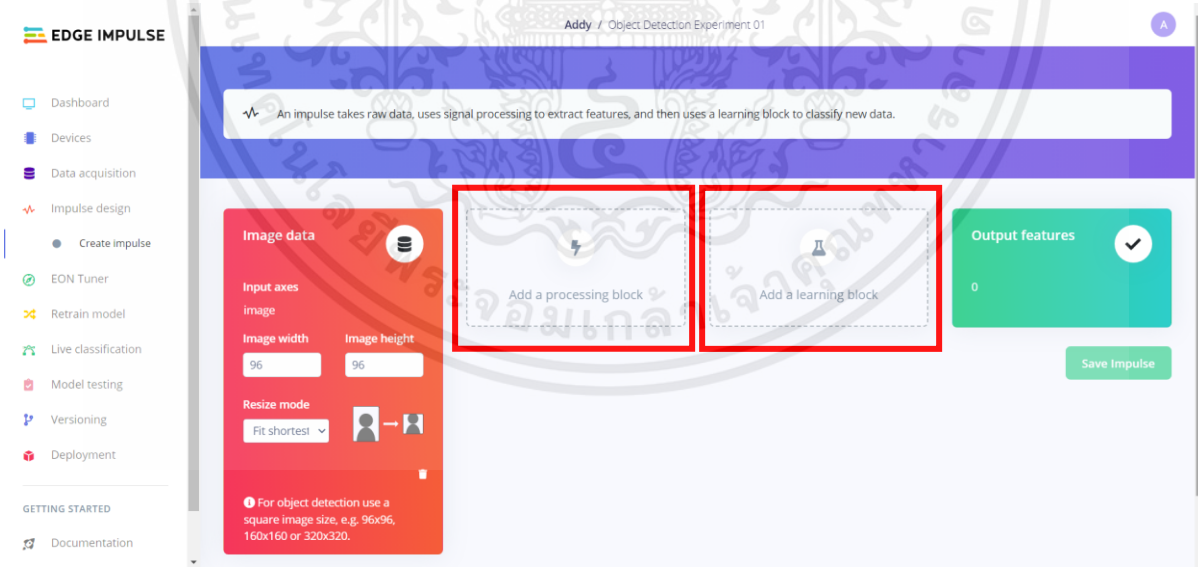
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

เมื่อลากกรอบและกำหนด label เรียบร้อยแล้วให้ทำการกด Save labels โดยทำแบบนี้กับรูปทั้งหมดใน dataset



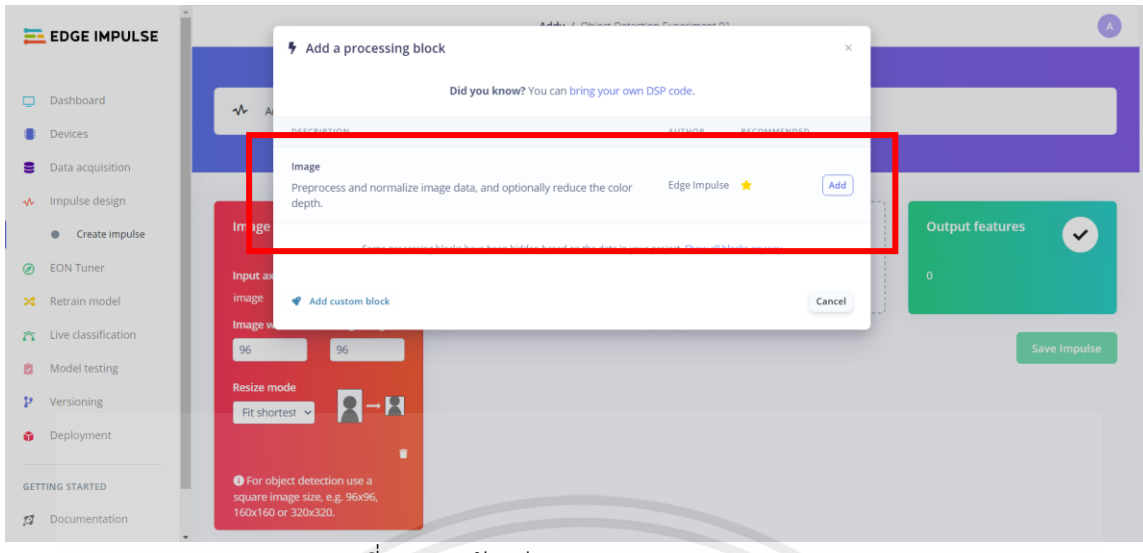
ภาพที่ 4.167 ตัวอย่างการตั้ง label (3)

ต่อมาในเมนู Create impulse Add a processing block ให้เลือก image และในส่วนของ Add a learning block ให้เลือก Object detection(Images) เมื่อเรียบร้อยแล้วให้ทำการ Save impulse

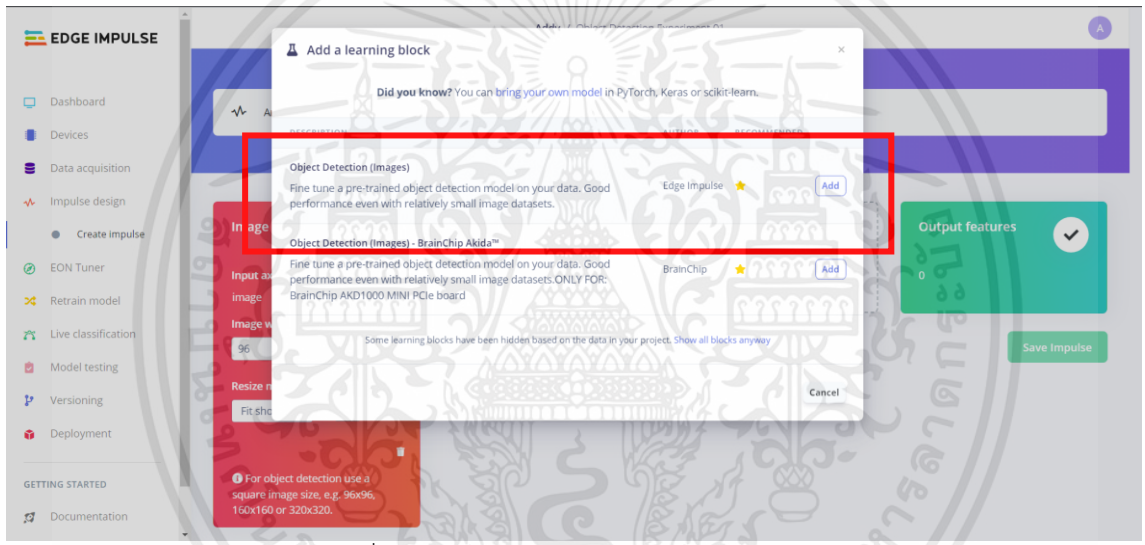


ภาพที่ 4.168 ตัวอย่างการ Create impulse (1)

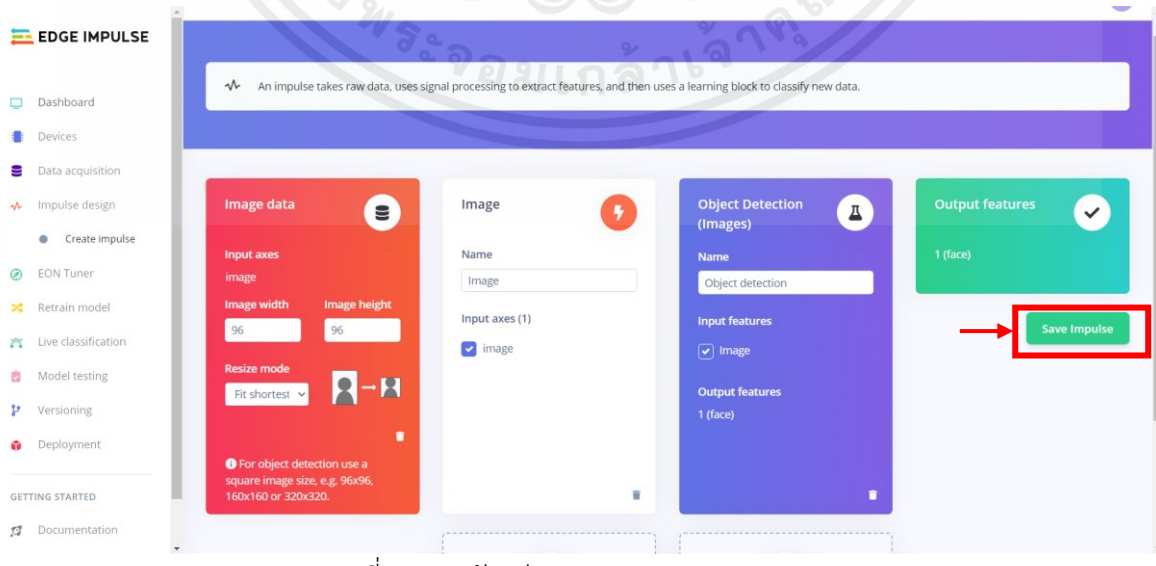
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



ภาพที่ 4.169 ตัวอย่างการ Create impulse (2)



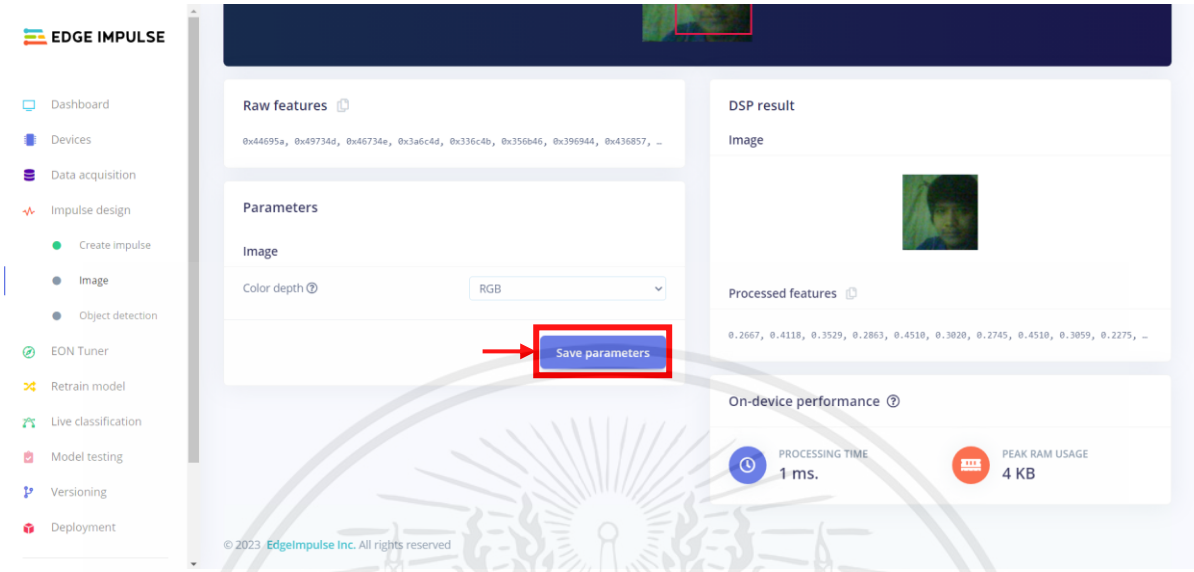
ภาพที่ 4.170 ตัวอย่างการ Create impulse (3)



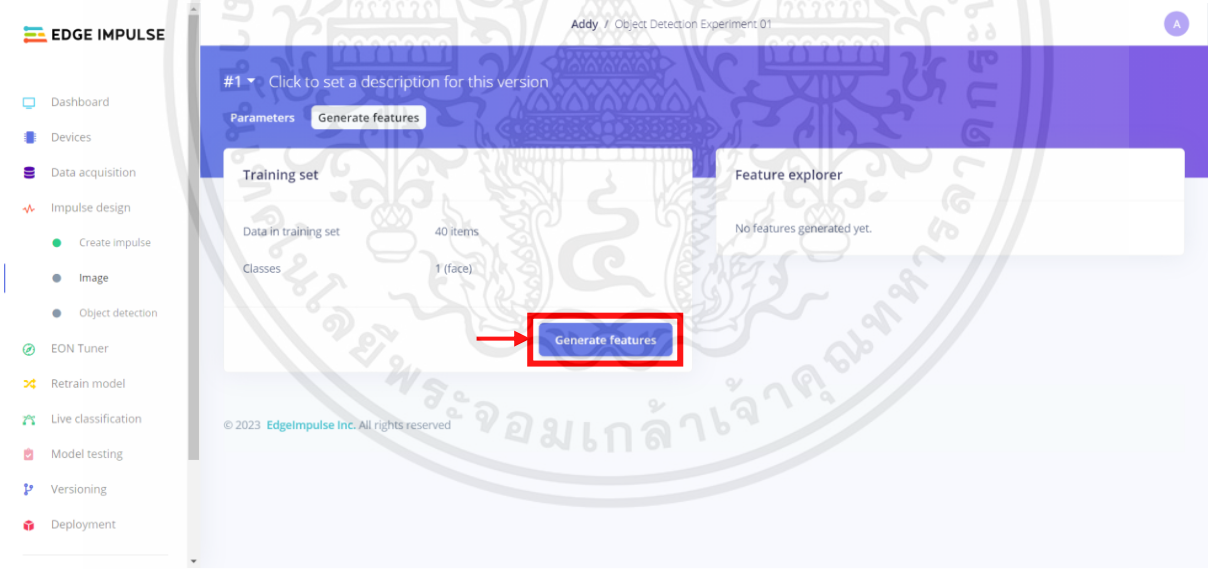
ภาพที่ 4.171 ตัวอย่างการ Create impulse (4)

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ในเมนู Image ทำการกด Save parameters เมื่อกดแล้วจะแสดงหน้าต่าง Generate features และกด Generate Features



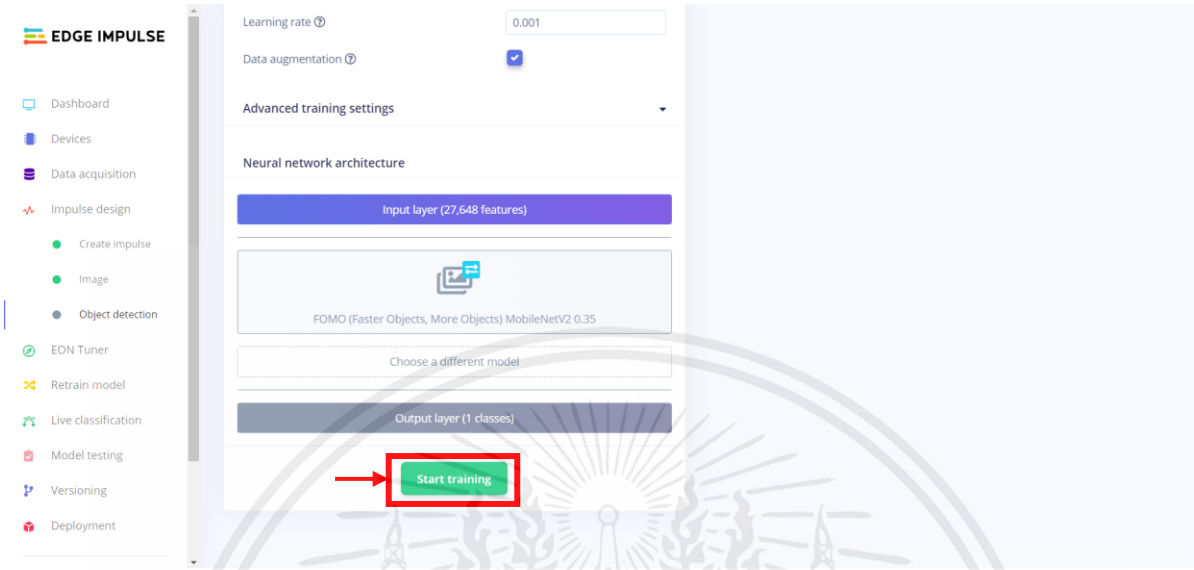
ภาพที่ 4.172 ตัวอย่างหน้าต่าง image (1)



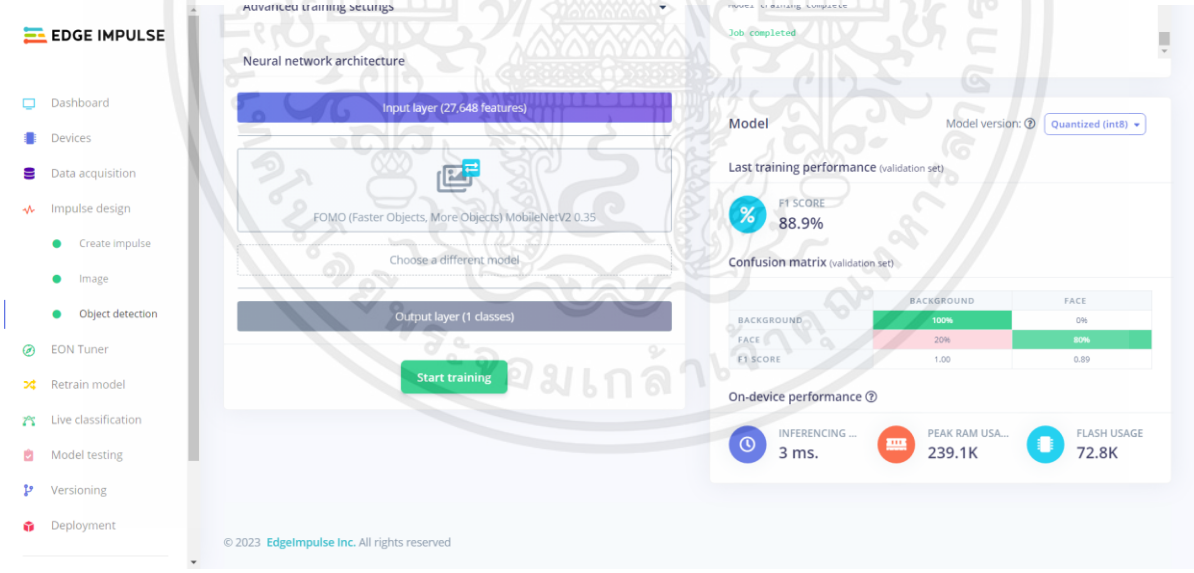
ภาพที่ 4.173 ตัวอย่างหน้าต่าง image (2)

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ในเมนู Object detection ทำการกด Start training เมื่อ training เสร็จจะแสดงผล Accuracy ที่ได้ออกมาด้วย



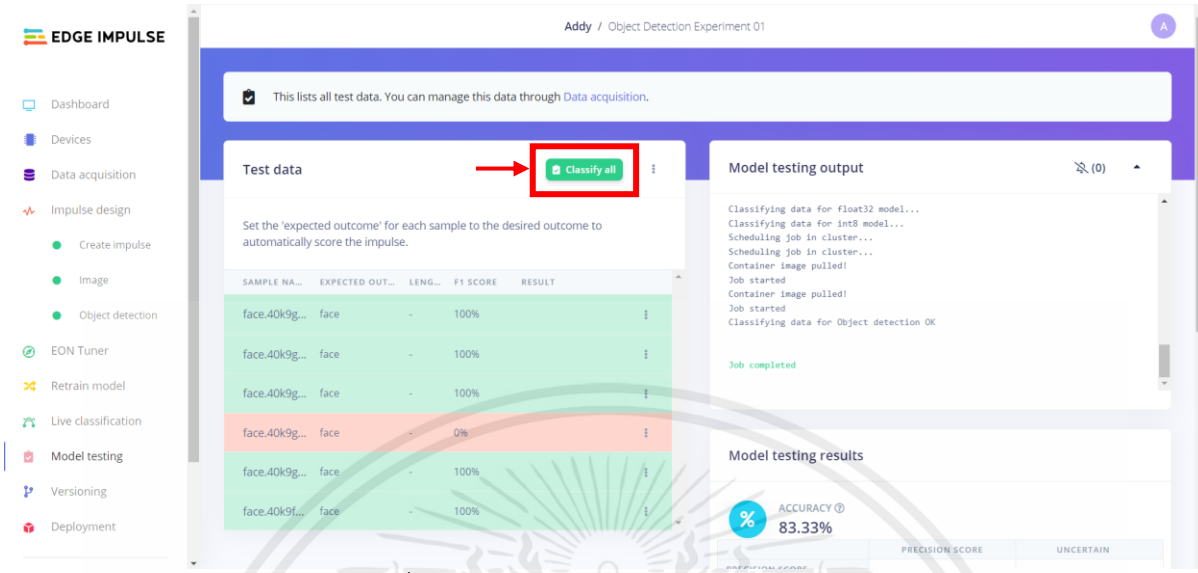
ภาพที่ 4.174 ตัวอย่างหน้าต่าง Object detection (1)



ภาพที่ 4.175 ตัวอย่างหน้าต่าง Object detection (2)

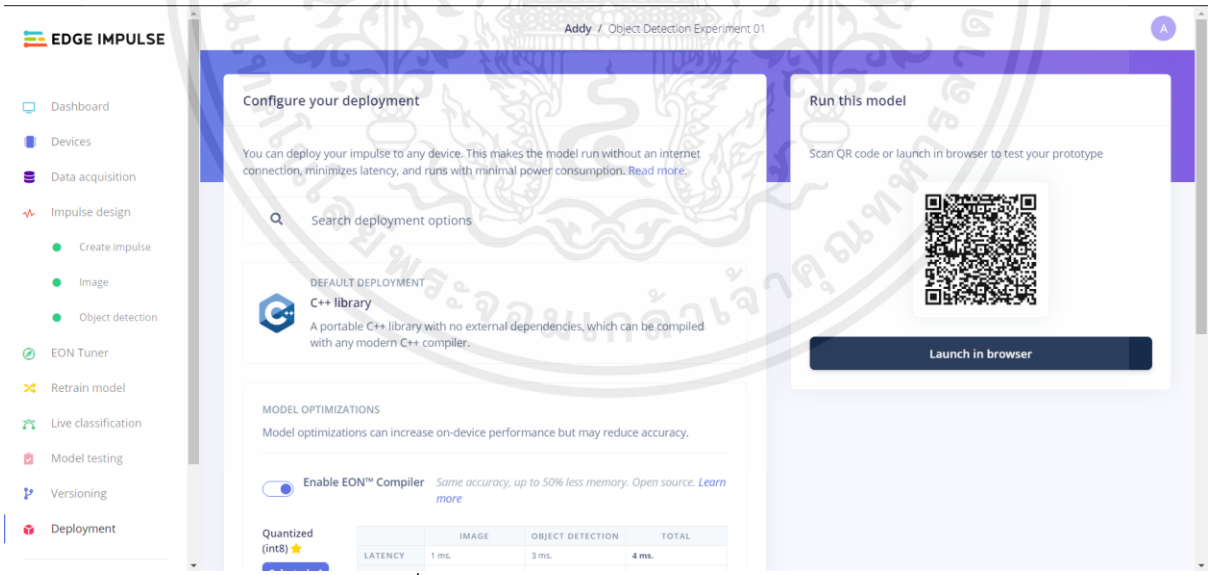
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ในหน้าต่าง Model testing ทำการกด Classify all เพื่อทำการทดสอบ Model และจะแสดงผลลัพธ์ที่ได้เป็นผล Accuracy



ภาพที่ 4.176 ตัวอย่างหน้าต่าง model testing

เพื่อทำการ Deploy ให้ไปในหน้า Deployment สแกน QR code สำหรับคนที่ใช้โทรศัพท์มือถือ และกด Launch in browser สำหรับ computer laptop ที่มี webcam

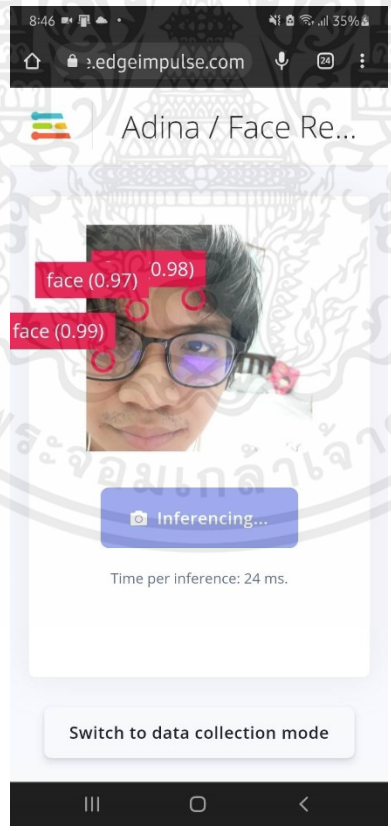


ภาพที่ 4.177 ตัวอย่างหน้าต่าง Deployment

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

โดยผลที่ได้โปรแกรมจะสามารถตรวจจับใบหน้าได้ออกมาเป็นผลลัพธ์ได้ทั้งบน Computer และโทรศัพท์มือถือ

ภาพที่ 4.178 ตัวอย่างผลลัพธ์การตรวจจับใบหน้า (1)

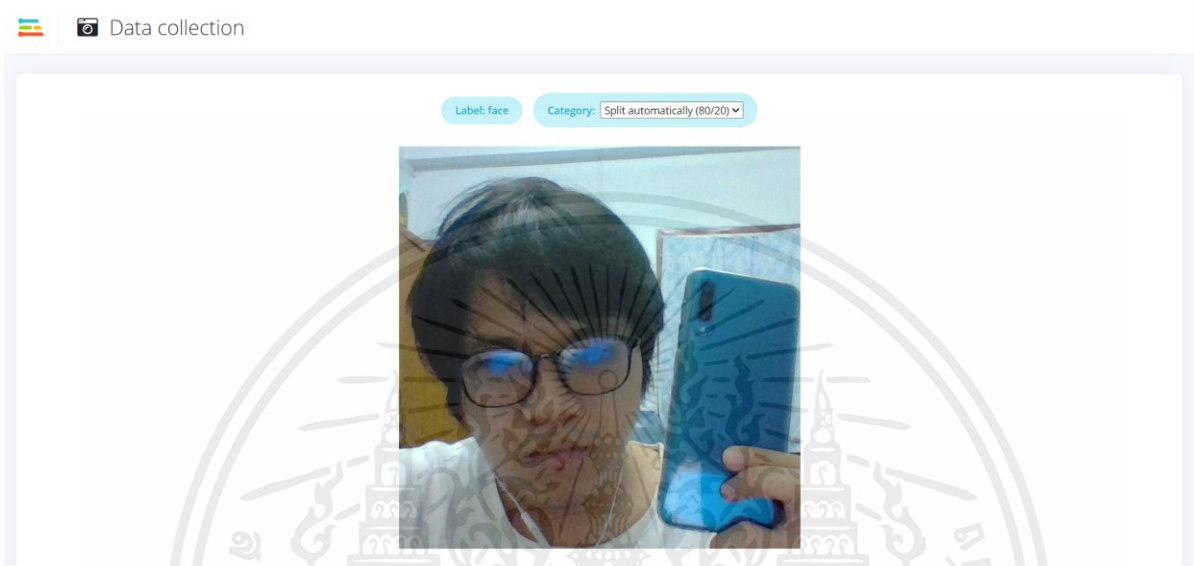


ภาพที่ 4.179 ตัวอย่างผลลัพธ์การตรวจจับใบหน้า (2)

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

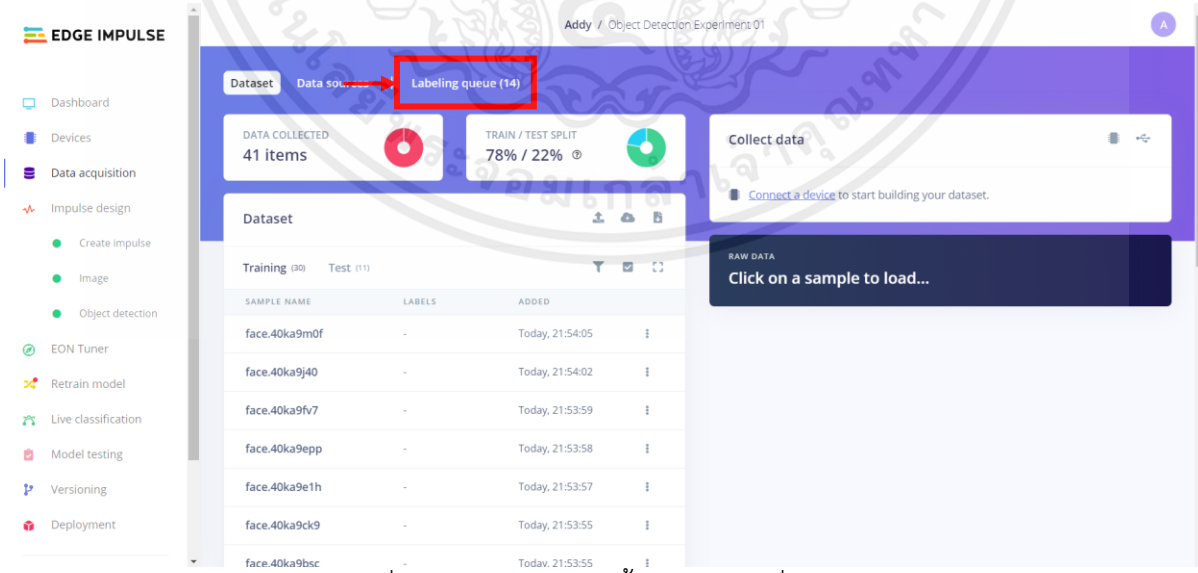
คำถาม : ให้ผู้ศึกษาทดลองดูว่าโปรแกรมสามารถตรวจจับใบหน้าของคนอื่นนอกจากผู้ศึกษาได้หรือไม่ ถ้าไม่ได้คิดว่าเกิดจากอะไร?

ในการทดลองต่อมาให้ผู้ศึกษาเก็บข้อมูลภาพเพิ่มโดยครั้งนี้ให้เก็บเป็นสิ่งของ เช่น โทรศัพท์มือถือ หรืออะไรที่มีขนาดเล็กสามารถใช้มือหยิบได้ โดยเก็บข้อมูลประมาณ 30-40 รูปเพิ่มไปจากเดิม



ภาพที่ 4.180 ตัวอย่างการเก็บข้อมูลภาพเพิ่มเติม

ในหน้าต่าง Data acquisition เข้าไปในหน้าต่าง Labeling queue



ภาพที่ 4.181 ตัวอย่างการตั้งค่า label เพิ่มเติม (1)

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

โดยให้ลากกรอบและกำหนด label ใหม่เป็นสิ่งของที่เราต้องการให้โปรแกรมตรวจสอบ ทำการ Save labels ทุกรูปใน dataset

ภาพที่ 4.182 ตัวอย่างการตั้งค่า label เพิ่มเติม (2)

จากนั้นให้ทำตามขั้นตอนเดิมตั้งแต่

- Save Parameters
- Generates features
- Training Model
- Deployment

คำถาม : โปรแกรมสามารถตรวจจับใบหน้าและวัตถุได้พร้อมกันหรือไม่? ถ้านำวัตถุนั้นมาในกล้องด้วยท่าทางอื่นโปรแกรมจะยังสามารถตรวจจับได้อยู่หรือไม่ ถ้าไม่ผู้ศึกษาคิดว่าเป็นเพราะอะไร?

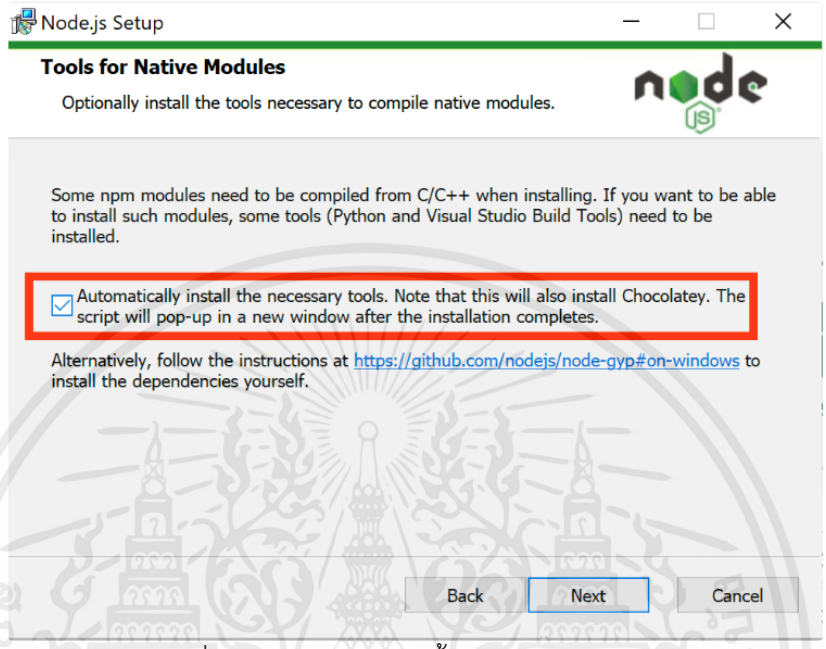
การทดลอง 1 : ให้ผู้ศึกษาทำการหารูปมนุษย์ที่ไม่ใช่ผู้ศึกษาจากที่ผู้ศึกษามีเองหรือจากอินเตอร์เน็ตและนำมา train เพื่อทำโปรแกรมตรวจจับใบหน้า และทดลองว่าโปรแกรมจะสามารถตรวจจับใบหน้าของผู้ศึกษาได้หรือไม่ และผู้ศึกษาคิดว่าผลลัพธ์ที่ได้นั้นเกิดจากอะไร

การทดลอง 2 : ให้ผู้ศึกษาทดลองสร้างโปรแกรมตรวจจับวัตถุต่างกัน 2 ชนิดพร้อมกันโดยให้ใช้วัตถุที่ผู้ศึกษามีอยู่และบันทึกผลการทดลอง

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

3.Motion Detection (การทดสอบตรวจจับภาพวัตถุด้วยการเรียนรู้ของเครื่องบนไมโครคอนโทรลเลอร์)

กรณีติดตั้ง Node.js ให้เลือกติดตั้ง tools เพิ่มเติมลงไปด้วย



ภาพที่ 4.183 ตัวอย่างการตั้งค่า Node.js

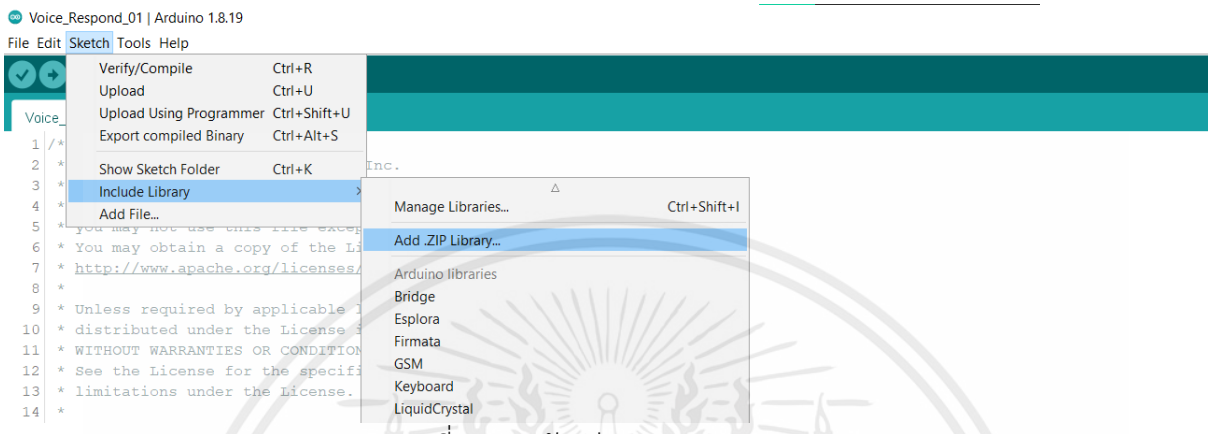
ใช้คำสั่ง npm install -g edge-impulse-cli –force เพื่อติดตั้ง edge impulse CLI จะทำให้สามารถส่งค่าจาก Serial port ไปยัง edge impulse ได้

ภาพที่ 4.184 ตัวอย่างการติดตตั้ง edge impulse CLI

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ให้ทำการโหลดไฟล์ library สำหรับ LSM6DS3 เพื่อทำให้สามารถวัดค่า Gyro แกน x,y,z ได้ https://files.seeedstudio.com/wiki/XIAO-BLE-Motion-Recognition/Seeed_Arduino_LSM6DS3-master.zip

เมื่อโหลดไฟล์ .ZIP เสร็จแล้วให้ทำการติดตั้ง library โดย include library และ Add .ZIP Library โดยเลือกไฟล์ที่โหลดมา



ภาพที่ 4.185 ตัวอย่างการ Add Library

ทำการต่อบอร์ดกับคอมพิวเตอร์และ Upload Code ต่อไปนี้

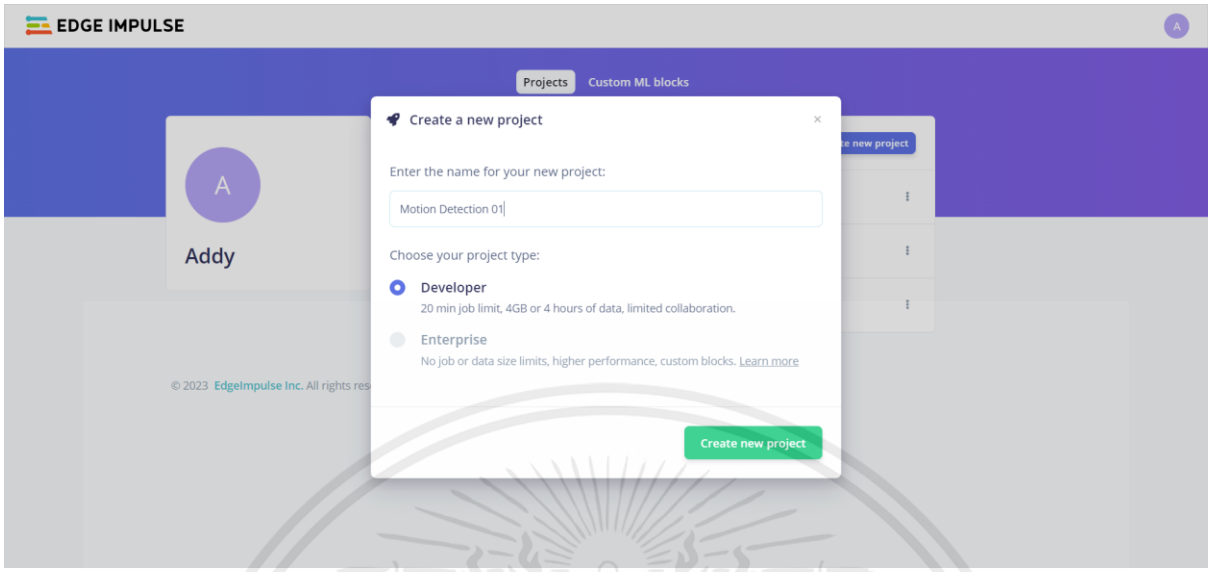
```
#include "LSM6DS3.h"
#include "Wire.h"

//Create a instance of class LSM6DS3
LSM6DS3 myIMU(I2C_MODE, 0x6A);    //I2C device address 0x6A


#define CONVERT_G_TO_MS2    9.80665f
#define FREQUENCY_HZ        50
#define INTERVAL_MS         (1000 / (FREQUENCY_HZ + 1))

static unsigned long last_interval_ms = 0;

void setup() {
    // put your setup code here, to run once:
    Serial.begin(115200);
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
    while (!Serial);
    //Call .begin() to configure the IMUs
    if (myIMU.begin() != 0) {
        Serial.println("Device error");
    } else {
        Serial.println("Device OK!");
    }
}

void loop() {

    if (millis() > last_interval_ms + INTERVAL_MS) {
        last_interval_ms = millis();

        Serial.print(myIMU.readFloatGyroX() * CONVERT_G_TO_MS2,4);
        Serial.print('\t');
        Serial.print(myIMU.readFloatGyroY() * CONVERT_G_TO_MS2,4);
        Serial.print('\t');
        Serial.println(myIMU.readFloatGyroZ() * CONVERT_G_TO_MS2,4);
    }
}
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ใน edge impulse ทำการสร้างโปรเจคใหม่ตั้งชื่อ Motion Detection Experiment 01



ภาพที่ 4.186 ตัวอย่างการสร้างโปรเจคใหม่

ใน cmd พิมพ์คำสั่ง edge-impulse-data-forwarder โดยระบบจะถาม e-mail และรหัสที่ใช้เชื่อมกับ edge impulse ทำการกรอก e-mail และรหัสลงไป

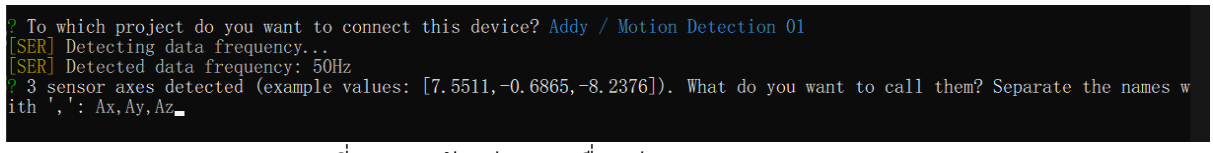
ภาพที่ 4.187 ตัวอย่างการเชื่อมต่อ edge impulse CLI (1)

เมื่อเชื่อมได้แล้วทำการเลือกโปรเจคที่ต้องการ โดยเลือก Motion Detection

ภาพที่ 4.188 ตัวอย่างการเชื่อมต่อ edge impulse CLI (2)

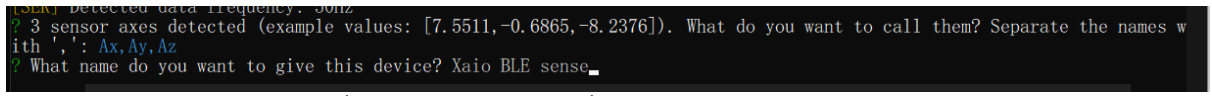
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ระบบจะให้ตั้งชื่อตัวแปรแกนทั้ง 3 แกน ให้ใส่ Ax,Ay,Az



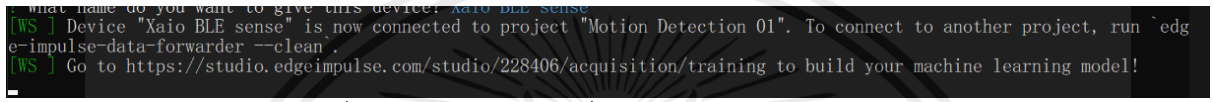
ภาพที่ 4.189 ตัวอย่างการเชื่อมต่อ edge impulse CLI (3)

ตั้งชื่ออุปกรณ์ว่า Xaio BLE sense



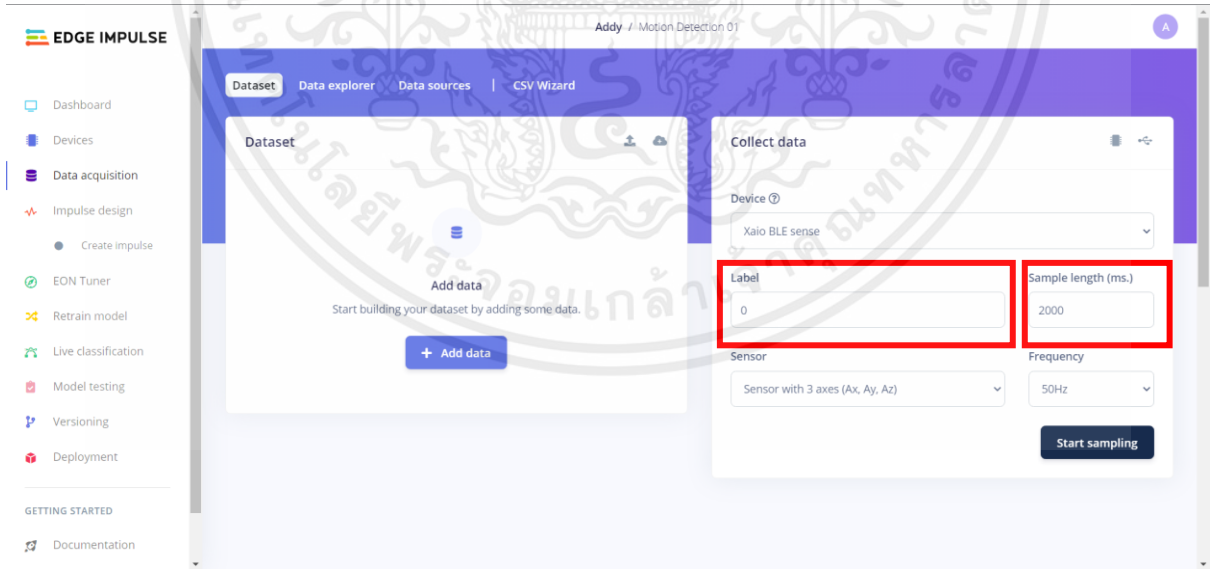
ภาพที่ 4.190 ตัวอย่างการเชื่อมต่อ edge impulse CLI (4)

เชื่อมต่อเรียบร้อย



ภาพที่ 4.191 ตัวอย่างการเชื่อมต่อ edge impulse CLI (5)

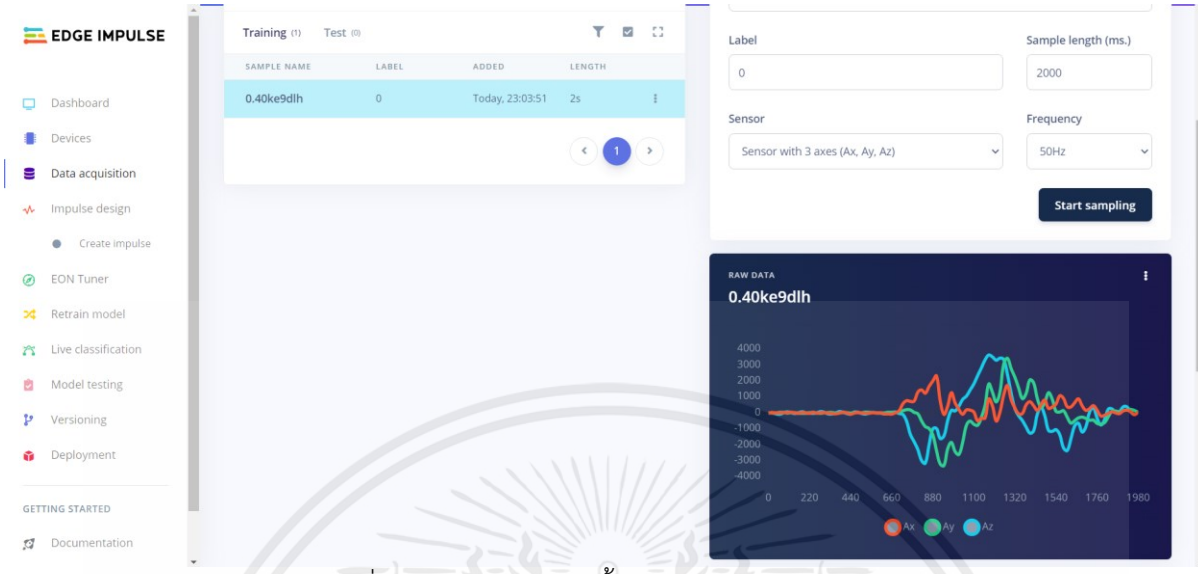
ต่อมาในเมนู Data acquisition จะมี sensor ขึ้นมา ให้ตั้งชื่อ label เป็น 0 และ Simple length เป็น 2000



ภาพที่ 4.192 ตัวอย่างการตั้งค่า label ข้อมูล IMU (1)

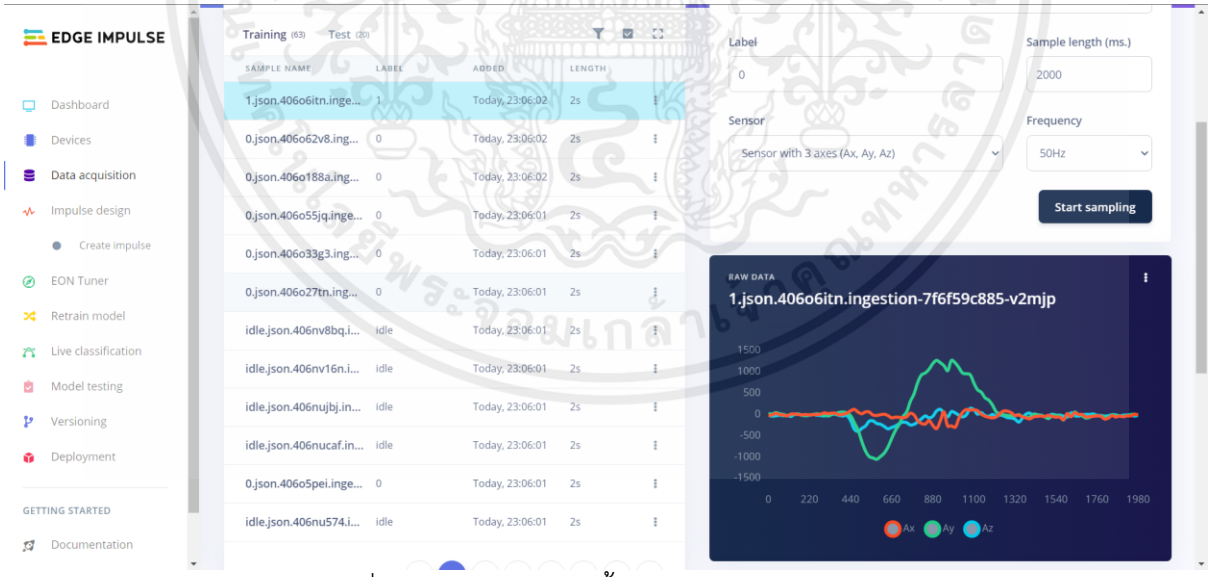
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

เมื่อตั้งค่าเสร็จให้ทำการกด Start sampling จะมีการนับถอยหลังและเริ่มเก็บข้อมูลโดยให้ถือบอร์ดเอาไว้และวาดเป็นเลข 0 โดยทำประมาณ 20-30 ครั้ง



ภาพที่ 4.193 ตัวอย่างการตั้งค่า label ข้อมูล IMU (2)

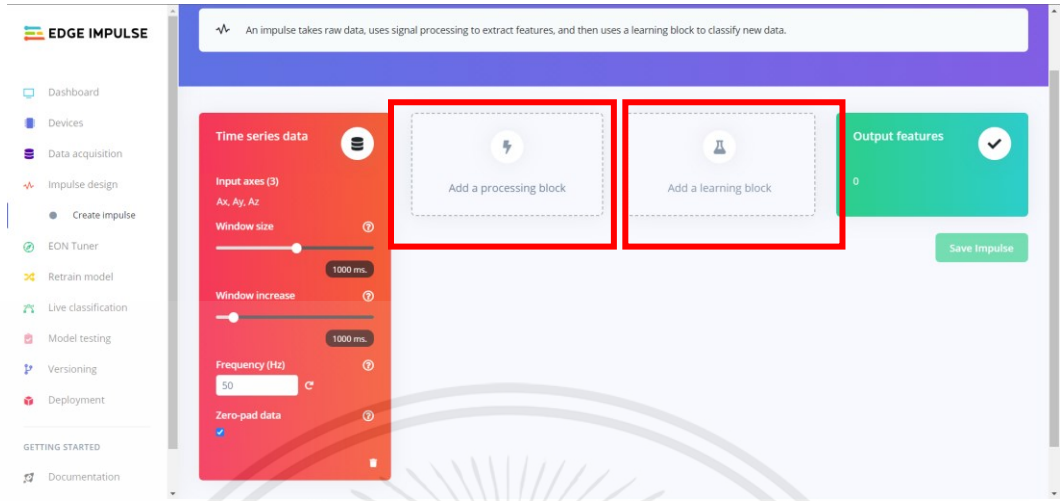
ให้ทำการเก็บข้อมูลอีก 1 label โดยตั้งชื่อ label ว่า 1 โดยครั้งนี้ให้วาดบอร์ดเป็นเลข 1 โดยเก็บข้อมูลประมาณ 20-30 ครั้ง



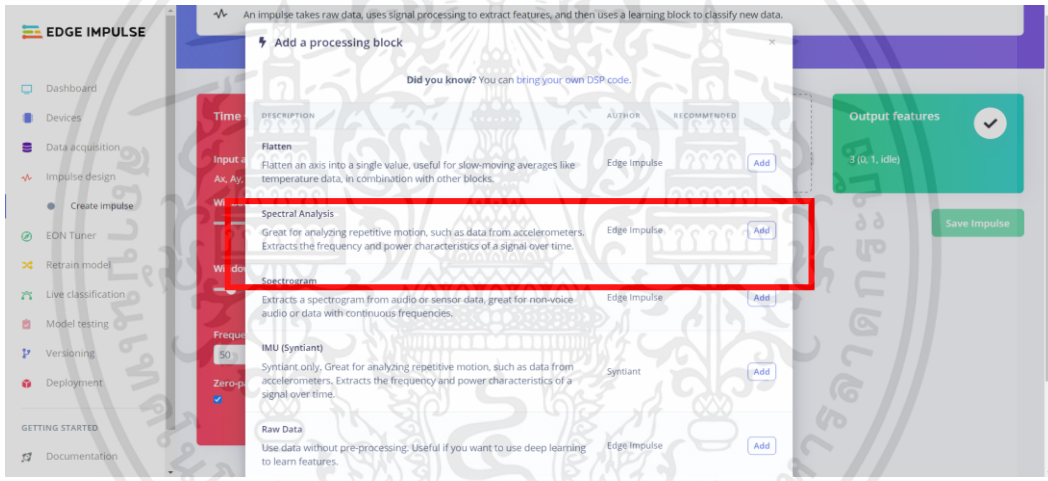
ภาพที่ 4.194 ตัวอย่างการตั้งค่า label ข้อมูล IMU (3)

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

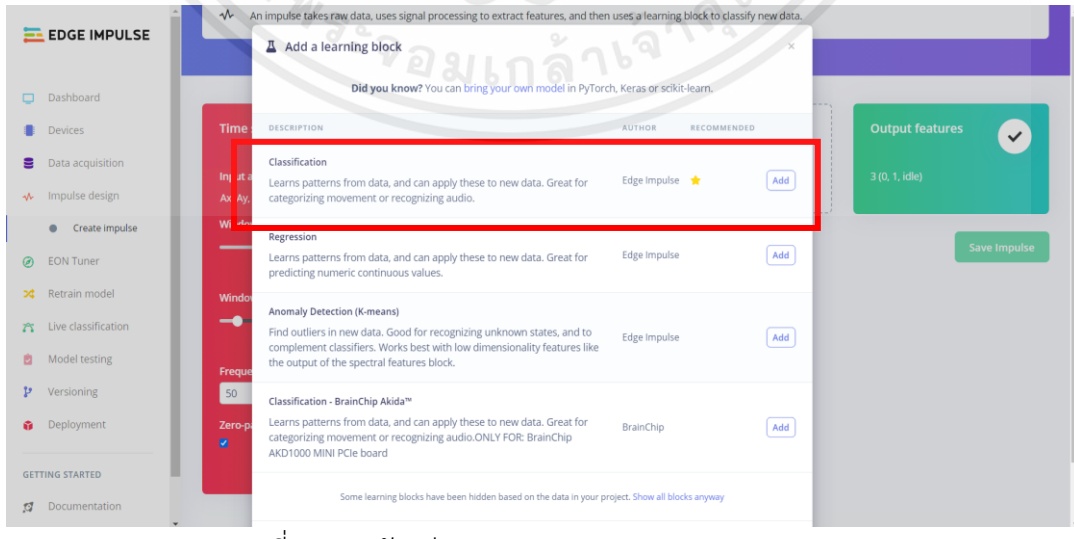
ในเมนู Create impulse ในส่วน Add a processing block ให้เลือก Spectral Analysis ในส่วน Add a learning block ให้เลือก Classification เมื่อตั้งค่าเสร็จแล้วให้กด Save impulse



ภาพที่ 4.195 ตัวอย่างการ Create impulse (1)



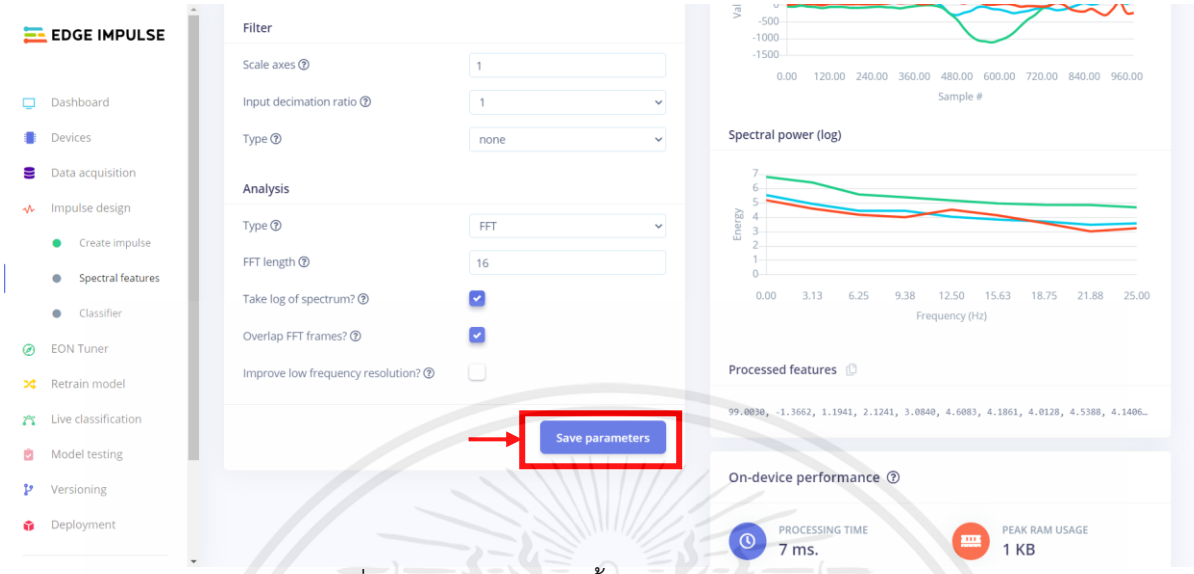
ภาพที่ 4.196 ตัวอย่างการการ Create impulse (2)



ภาพที่ 4.197 ตัวอย่างการการ Create impulse (3)

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ในเมนู Spectral Features ทำการกด Save parameters โดยจะแสดงหน้าต่าง Generate features ขึ้นมา ให้กด Generate features

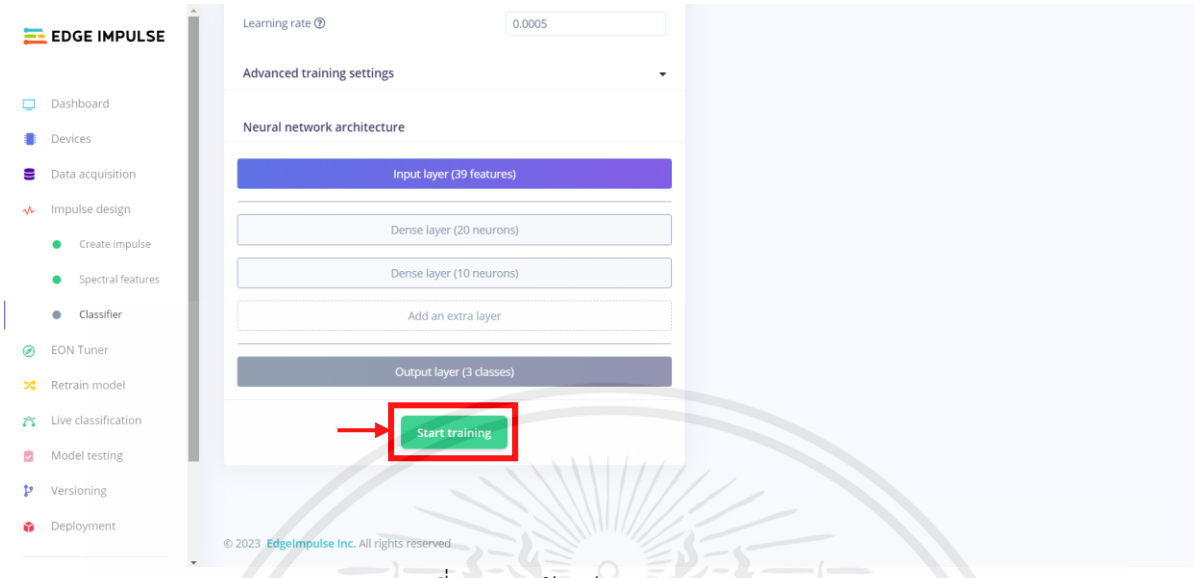


ภาพที่ 4.198 ตัวอย่างการตั้งค่า Spectral features (1)

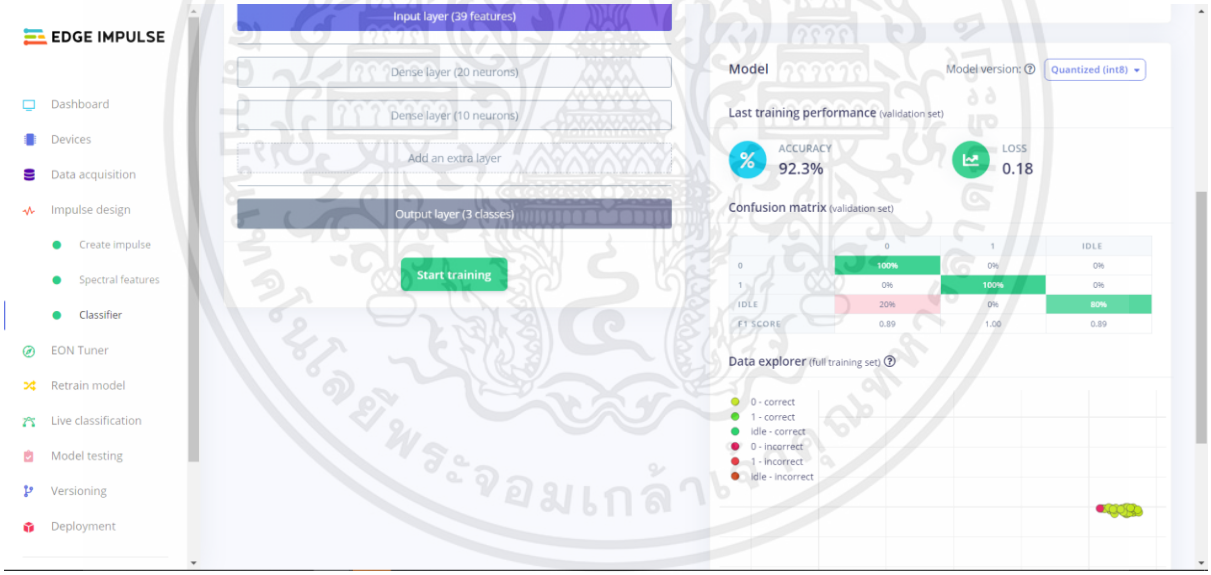
ภาพที่ 4.199 ตัวอย่างการตั้งค่า Spectral features (2)

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ในเมนู Classifier กด Start training เพื่อเริ่ม train model เมื่อ train เสร็จจะแสดงผลเป็น Accuracy ขึ้นมา



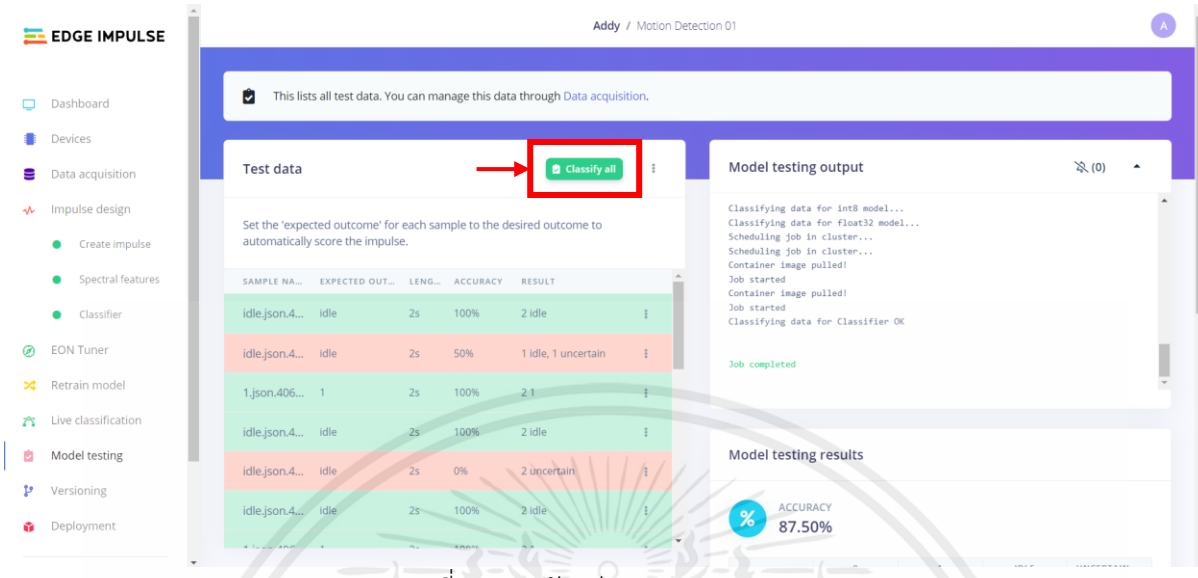
ภาพที่ 4.200 ตัวอย่างการ Classifier (1)



ภาพที่ 4.201 ตัวอย่างการ Classifier (2)

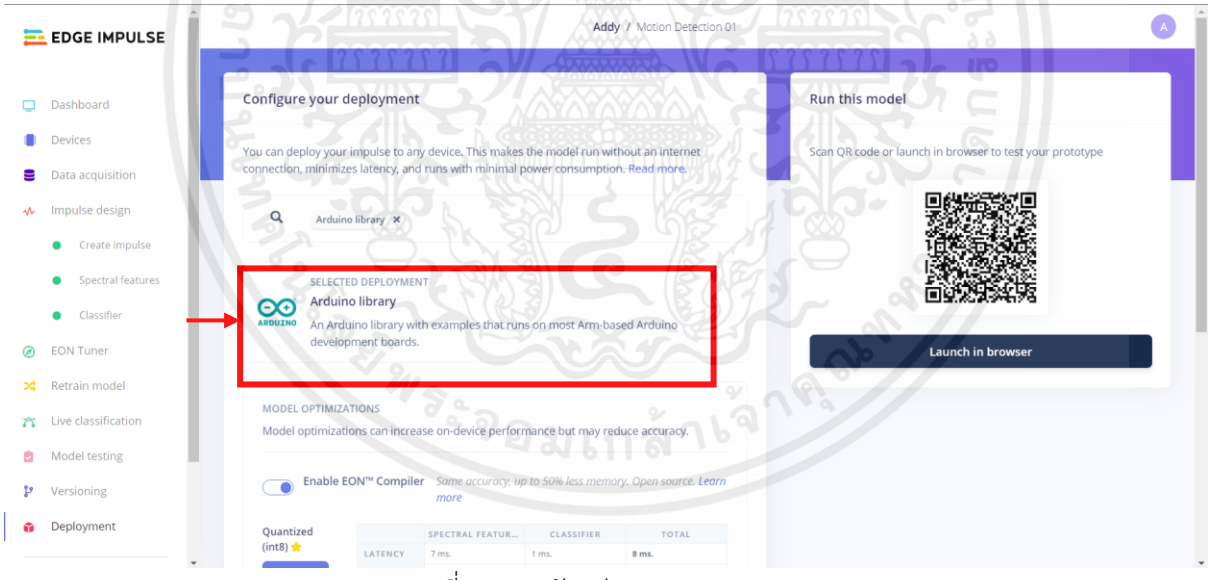
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ในเมนู Model testing ให้ทำการกด Classify all เพื่อทดสอบ Model โดยจะได้ผลลัพธ์มาเป็น Accuracy ของ Model



ภาพที่ 4.202 ตัวอย่างการ test model

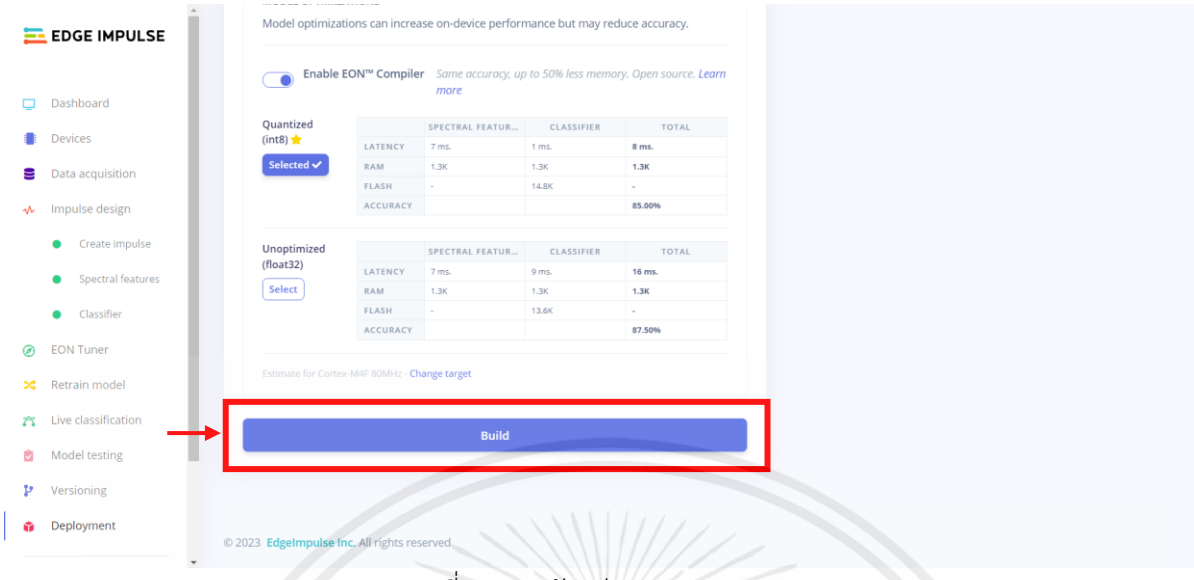
เพื่อ Deploy ลงบอร์ด เข้ามาในเมนู Deployment ค้นหา Arduino library



ภาพที่ 4.203 ตัวอย่างการ Deploy (1)

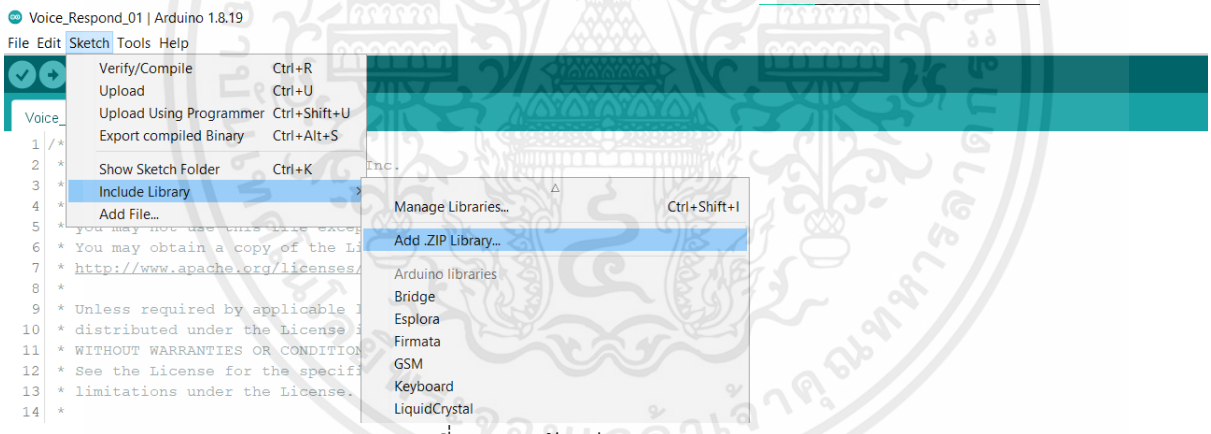
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

เมื่อเลือก Arduino library แล้ว ให้กด build เพื่อ download ไฟล์ .ZIP



ภาพที่ 4.204 ตัวอย่างการ Deploy (2)

เปิดโปรแกรม Arduino เพื่อเพิ่ม library ทำการเลือกเมนู Sketch และทำการ Include Library ด้วยการ Add .ZIP Library แล้วเลือกไฟล์ที่โหลดมา



ภาพที่ 4.205 ตัวอย่างการ Add Library

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

สร้างไฟล์ Arduino ใหม่แล้วใส่ Code ดังต่อไปนี้

```
/* Edge Impulse Arduino examples
 * Copyright (c) 2021 EdgeImpulse Inc.
 *
 * Permission is hereby granted, free of charge, to any person obtaining a copy
 * of this software and associated documentation files (the "Software"), to deal
 * in the Software without restriction, including without limitation the rights
 * to use, copy, modify, merge, publish, distribute, sublicense, and/or sell
 * copies of the Software, and to permit persons to whom the Software is
 * furnished to do so, subject to the following conditions:
 *
 * The above copyright notice and this permission notice shall be included in
 * all copies or substantial portions of the Software.
 *
 * THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR
 * IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY,
 * FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE
 * AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER
 * LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM,
 * OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN
THE
 * SOFTWARE.
 */

/* Includes ---------------------------------------------------------------- */
#include <Magic_Wand_Test_inferencing.h>
#include <LSM6DS3.h>
/*#include <U8g2lib.h>
#include <U8X8lib.h>*/
#include <Wire.h>
/* Constant defines -------------------------------------------------------- */
#define CONVERT_G_TO_MS2    9.80665f
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
#define MAX_ACCEPTED_RANGE  2.0f        // starting 03/2022, models are generated setting range to +-2, but this example use Arudino library which set range to +-4g. If you are using an older model, ignore this value and use 4.0f instead

/*
 ** NOTE: If you run into TFLite arena allocation issue.
 **
 ** This may be due to may dynamic memory fragmentation.
 ** Try defining "-DEI_CLASSIFIER_ALLOCATION_STATIC" in boards.local.txt (create
 ** if it doesn't exist) and copy this file to
 ** `<ARDUINO_CORE_INSTALL_PATH>/arduino/hardware/<mbed_core>/<core_version>/`.
 **
 ** See
 ** (https://support.arduino.cc/hc/en-us/articles/360012076960-Where-are-the-installed-cores-located-)
 ** to find where Arduino installs cores on your machine.
 **
 ** If the problem persists then there's not enough memory for this model and application.
 */

//U8X8_SSD1306_64X48_ER_HW_I2C u8x8(/* reset=*/ U8X8_PIN_NONE);

/* Private variables ------------------------------------------------------- */
static bool debug_nn = false; // Set this to true to see e.g. features generated from the raw signal
LSM6DS3 myIMU(I2C_MODE, 0x6A);
/**
* @brief      Arduino setup function
*/

const int RED_ledPin =  11;
const int BLUE_ledPin =  12;
const int GREEN_ledPin =  13;
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
void setup()
{
    // put your setup code here, to run once:
    Serial.begin(115200);
    //u8g2.begin();
    //u8x8.begin();
    Serial.println("Edge Impulse Inferencing Demo");

    //if (!IMU.begin()) {
      if (!myIMU.begin()) {
        ei_printf("Failed to initialize IMU!\r\n");
    }
    else {
        ei_printf("IMU initialized\r\n");
    }

    if (EI_CLASSIFIER_RAW_SAMPLES_PER_FRAME != 3) {
        ei_printf("ERR: EI_CLASSIFIER_RAW_SAMPLES_PER_FRAME should be equal to 3 (the 3
sensor axes)\n");
        return;
    }
}

/**
 * @brief Return the sign of the number
 *
 * @param number
 * @return int 1 if positive (or 0) -1 if negative
 */
float ei_get_sign(float number) {
    return (number >= 0.0) ? 1.0 : -1.0;
}
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
/**
* @brief      Get data and run inferencing
*
* @param[in]  debug  Get debug info if true
*/
void loop()
{
    uint8_t buf1[64]="idle";
    uint8_t buf2[64]="left&right";
    uint8_t buf3[64]="up&down";

//    u8x8.clear();
   // u8x8.setFont(u8g2_font_ncenB08_tr);

    ei_printf("\nStarting inferencing in 2 seconds...\n");

    delay(2000);

    ei_printf("Sampling...\n");

    // Allocate a buffer here for the values we'll read from the IMU
    float buffer[EI_CLASSIFIER_DSP_INPUT_FRAME_SIZE] = { 0 };

    for (size_t ix = 0; ix < EI_CLASSIFIER_DSP_INPUT_FRAME_SIZE; ix += 3) {
        // Determine the next tick (and then sleep later)
        uint64_t next_tick = micros() + (EI_CLASSIFIER_INTERVAL_MS * 1000);

//buffer[ix] = myIMU.readFloatAccelX();
//buffer[ix+1] = myIMU.readFloatAccelY();
//buffer[ix+2] = myIMU.readFloatAccelZ();

buffer[ix] = myIMU.readFloatGyroX();
buffer[ix+1] = myIMU.readFloatGyroY();
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
buffer[ix+2] = myIMU.readFloatGyroZ();

        for (int i = 0; i < 3; i++) {
            if (fabs(buffer[ix + i]) > MAX_ACCEPTED_RANGE) {
                buffer[ix + i] = ei_get_sign(buffer[ix + i]) * MAX_ACCEPTED_RANGE;
            }
        }

        buffer[ix + 0] *= CONVERT_G_TO_MS2;
        buffer[ix + 1] *= CONVERT_G_TO_MS2;
        buffer[ix + 2] *= CONVERT_G_TO_MS2;

        delayMicroseconds(next_tick - micros());
    }

    // Turn the raw buffer in a signal which we can the classify
    signal_t signal;
    int err = numpy::signal_from_buffer(buffer, EI_CLASSIFIER_DSP_INPUT_FRAME_SIZE,
&signal);
    if (err != 0) {
        ei_printf("Failed to create signal from buffer (%d)\n", err);
        return;
    }

    // Run the classifier
    ei_impulse_result_t result = { 0 };

    err = run_classifier(&signal, &result, debug_nn);
    if (err != EI_IMPULSE_OK) {
        ei_printf("ERR: Failed to run classifier (%d)\n", err);
        return;
    }
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
    // print the predictions
    ei_printf("Predictions ");
    ei_printf("(DSP: %d ms., Classification: %d ms., Anomaly: %d ms.)",
        result.timing.dsp, result.timing.classification, result.timing.anomaly);
    ei_printf(": \n");
    for (size_t ix = 0; ix < EI_CLASSIFIER_LABEL_COUNT; ix++) {
        ei_printf("    %s: %.5f\n", result.classification[ix].label, result.classification[ix].value);
    }
#if EI_CLASSIFIER_HAS_ANOMALY == 1
    ei_printf("    anomaly score: %.3f\n", result.anomaly);
#endif

  /*if (result.classification[1].value > 0.5) {
      digitalWrite(RED_ledPin, LOW);
      digitalWrite(BLUE_ledPin, HIGH);                    // idle      red
      digitalWrite(GREEN_ledPin, HIGH);
      u8x8.setFont(u8x8_font_amstrad_cpc_extended_r);
      u8x8.drawString(2,3,"idle");
      u8x8.refreshDisplay();
      delay(2000);
   }

  if (result.classification[0].value > 0.5) {
       digitalWrite(RED_ledPin, HIGH);                    //left&right          blue
       digitalWrite(BLUE_ledPin, LOW);
       digitalWrite(GREEN_ledPin, HIGH);
      u8x8.setFont(u8x8_font_amstrad_cpc_extended_r);
      u8x8.drawString(2,3,"left");
      u8x8.drawString(2,4,"right");
      u8x8.refreshDisplay();
      delay(2000);
   }
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
  if (result.classification[2].value > 0.5) {
      digitalWrite(RED_ledPin, HIGH);
      digitalWrite(BLUE_ledPin, HIGH);
      digitalWrite(GREEN_ledPin, LOW);                    //up&down          green
        u8x8.setFont(u8x8_font_amstrad_cpc_extended_r);
        u8x8.drawString(2,3,"up");
        u8x8.drawString(2,4,"down");
        u8x8.refreshDisplay();
        delay(2000);
  }*/


}
```

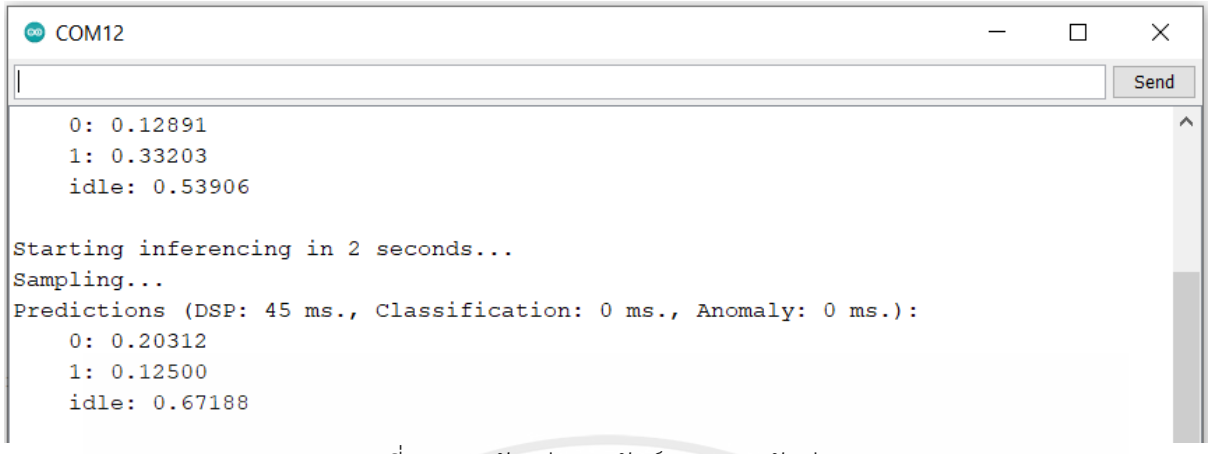
โดยให้เปลี่ยนชื่อ library ให้เป็นชื่อ library ที่โหลดมา

```
24 #include <Magic_Wand_Test_inferencing.h>
26 /*#include <U8g2lib.h>
27 #include <U8X8lib.h>*/
28 #include <Wire.h>
29 /* Constant defines ------------------------------------------------------- */
30 #define CONVERT_G_TO_MS2    9.80665f
31 #define MAX_ACCEPTED_RANGE  2.0f        // starting 03/2022, models are generated setting range to +
```

ภาพที่ 4.206 ตัวอย่างการเปลี่ยน Library

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

เมื่อ Upload code ลงบอร์ดและเปิด Serial port จะแสดงผลลัพธ์ที่ได้จากการคาดเดาทางท่า



ภาพที่ 4.207 ตัวอย่างผลลัพธ์การตรวจจับท่าทาง

คำถาม : เมื่อผู้ศึกษาได้ทดลองวาดบอร์ดเป็นเลข 1 และ 0 อย่างละ 10 ครั้งมีกี่ครั้งที่โปรแกรมสามารถทำนายผลลัพธ์ได้ถูกต้อง ผู้ศึกษาคิดว่าเป็นเพราะอะไรถึงได้ผลลัพธ์เป็นแบบนั้น

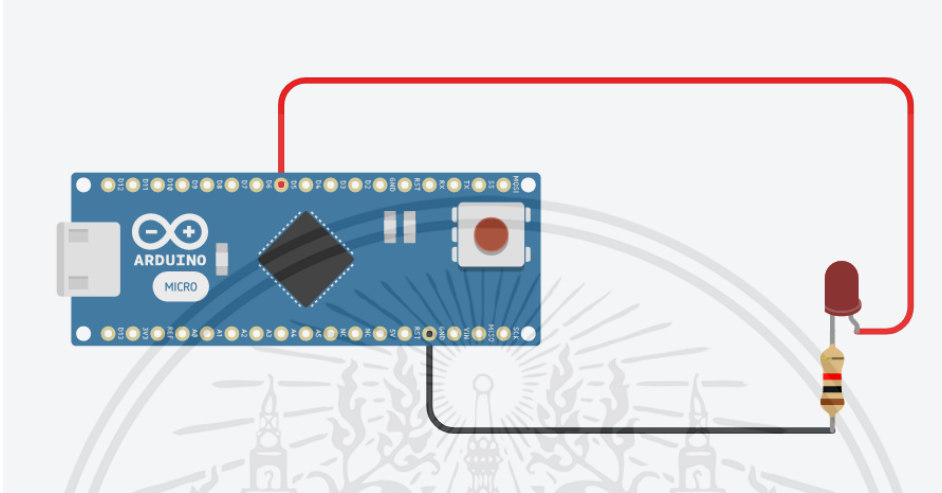
การทดลอง 1 : ให้ผู้ศึกษาเก็บข้อมูลเพิ่มขึ้นอีก 1 label เป็น label ชื่อ idle โดยเก็บข้อมูลให้บอร์ดอยู่นิ่ง ๆ ในมือ และบันทึกผลลัพธ์การทดลอง

การทดลอง 2 : ให้ผู้ศึกษาเพิ่มข้อมูลอะไรก็ได้ไปอีก 1 label และบันทึกผลลัพธ์การทดลอง

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

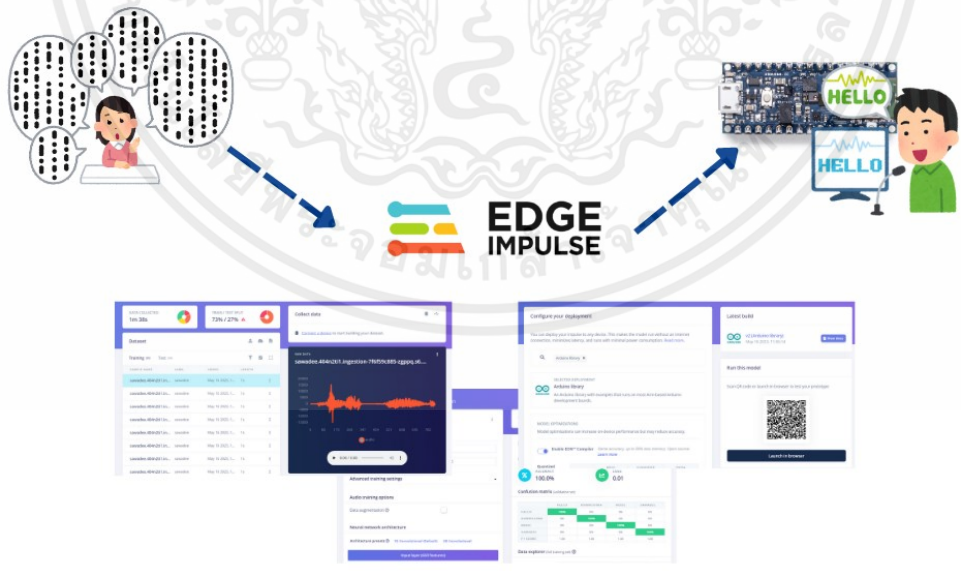
4.6 การนำไประยุกต์ใช้

ทางผู้จัดทำได้เลือกที่จะนำสิ่งที่ได้ศึกษามาสร้างเป็นแบบจำลองระบบการควบคุมหลอดไฟด้วยเสียง โดยนำบอร์ด Arduino nano 33 BLE ที่ได้ติดตั้ง code และ โมเดลที่ใช้ในการจับเสียงที่พูดเป็นเสียง เปิดและปิด



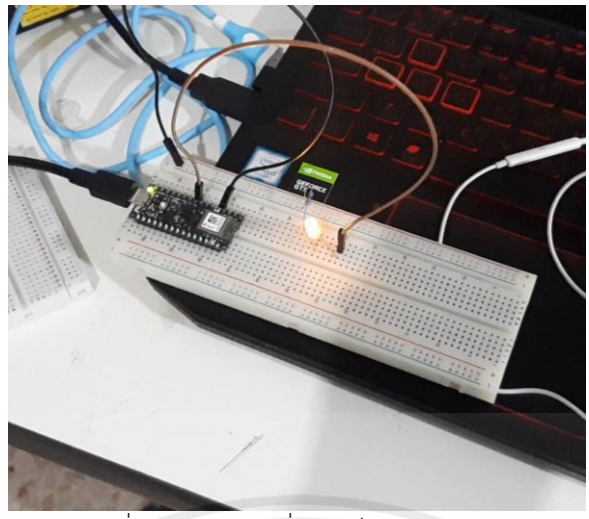
ภาพที่ 4.208 วงจรระบบการควบคุมหลอดไฟด้วยเสียง

โดยเริ่มจากการกำหนดคำที่ต้องการและอัดข้อมูลเสียงเพื่อ upload ขึ้น edge impulse นำไปผ่านการ train และ deploy ลงบอร์ด Arduino nano 33 BLE



ภาพที่ 4.209 ภาพรวมการทำงาน

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



ภาพที่ 4.210 วงจรที่ต่อเสร็จเรียบร้อยแล้ว

โดยได้ทำการทดสอบการใช้งานในสภาพวะปกติ เพิ่มระยะห่างระหว่างผู้พูดกับอุปกรณ์ และทดสอบในสภาวะที่มีเสียงรบกวนอยู่ในระยะใกล้ ได้ผลดังนี้

ตาราง 4.1 ตารางผลการทดลอง

วิธีการทดลอง	ทดลองพูดเปิด 10 ครั้ง สำเร็จกี่ครั้ง	ทดลองพูดปิด 10 ครั้ง สำเร็จกี่ครั้ง
พูดในสภาวะปกติ	8	10
พูดห่างจากอุปกรณ์ 1 เมตร	8	9
พูดในสภาวะเสียงรบกวน (เพลง , ปรบมือ)	7	8
พูดในสภาวะเสียงรบกวน (เพลง , ปรบมือ) และห่างจากอุปกรณ์ 1 เมตร	6	6
ให้บุคคลที่ไม่ใช่เจ้าของตัวอย่างเสียงพูด	3	5

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

บทที่ 5
สรุปผลและข้อเสนอแนะ

5.1 สรุปผล

ในปัจจุบันมีการนำเทคโนโลยี Machine learning มาใช้อย่างแพร่หลายในอุตสาหกรรมทุก ๆ ด้านทั้งในทางด้านการแพทย์เพื่อช่วยในการวินิจฉัยโรคต่าง ๆ หรือด้านการตลาดที่นำ Machine learning มาใช้คาดการณ์แนวโน้มของราคา หรือยอดกำไรถึงแม้ Machine learning จะมีประโยชน์และถูกนำไปประยุกต์ใช้งานกับสายงานต่าง ๆ จำนวนมาก แต่ Machine learning ก็ยังมีข้อจำกัดสำคัญบางประการอยู่ หนึ่งในนั้นคือยิ่งมีปริมาณข้อมูลที่มากเท่าไหรก็ยิ่งจำเป็นที่ต้องใช้เวลาและกำลังในการประมวลผลที่มากขึ้นตามในเวลาต่อมาจึงได้มีการพัฒนาเทคโนโลยี Machine learning ที่ไม่จำเป็นที่จะต้องใช้พลังงานและระบบประมวลผลที่กำลังสูง โดยใช้การบีบอัดข้อมูลหรือตัดบางส่วนของข้อมูลที่ไม่ได้มีความสำคัญอย่างเห็นได้ชัดออกไป โดยเน้นไปที่ความเร็วในการประมวลผล และกินพลังงานต่ำ โดยมีการนิยาม Machine learning ขนาดเล็กนี้ว่า TinyMLทั้งนี้ TinyML ยังเป็นเทคโนโลยีที่ค่อนข้างใหม่และยังไม่ได้มีการใช้อย่างแพร่หลาย ทางคณะผู้จัดทำจึงต้องการที่จะศึกษาถึงข้อเด่น ข้อด้อยของ TinyML และเปรียบเทียบการทำงานของ TinyML และ Machine learning ทั่วไป เพื่อหาถึงวิธีการใช้งานและการนำไปประยุกต์ใช้งานที่เหมาะสมในด้านต่าง ๆ

โดยผลลัพธ์ที่ได้คือผู้จัดทำสามารถเข้าใจได้ว่าถึงแม้ TinyML จะไม่สามารถแทนที่ Machine Learning ได้ทั้งหมดแต่มีงานบางประเภทที่เหมาะสมต่อการใช้งาน TinyML คืองานต้องการการเก็บและประมวลผลระยะยาว เช่น การตรวจสอบความเสียหายของอุปกรณ์ในโรงงาน การตรวจจับคำเฉพาะ การตรวจจับวัตถุอย่างง่าย เป็นต้น ซึ่งงานที่กล่าวมาทั้งหมดนี้สามารถใช้การโปรแกรมธรรมดาเพื่อให้สามารถทำงานได้ก็จริง แต่การนำ TinyML มาประยุกต์ใช้ร่วมจะทำให้การทำงานมีประสิทธิภาพมากขึ้นทั้งในด้านขอบเขตการทำงาน ไปจนถึงความคุ้มค่าในการลงทุนในระยะยาว เพียงแต่เมื่อนำ Machine Learning ฝังลงบนไมโครคอนโทรลเลอร์แล้วความเสถียรและความแม่นยำจะลดลงมาเล็กน้อยแต่ก็เพียงพอจะใช้งานแบบเฉพาะ ในท้ายสุดผู้จัดทำยังได้นำสิ่งที่ได้ศึกษามาออกแบบเป็นแบบเรียนสำหรับพัฒนาให้ผู้เริ่มต้นที่สนใจจะมาเรียนรู้เกี่ยวกับ TinyML สามารถทำความเข้าใจพื้นฐานและภาพรวมของ TinyML ในเบื้องต้นและสามารถนำไปประยุกต์ใช้ต่อกับงานของตนได้

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

5.2 ปัญหาที่พบในการวิจัย

การศึกษาเกี่ยวกับการเรียนรู้ของเครื่องและการประยุกต์ใช้งานบนไมโครคอนโทรลเลอร์นั้น ปัญหาที่เจอคือการที่ไม่สามารถใช้งานไลบรารี TensorFlow บน Arduino ได้อย่างเต็มที่เนื่องจากยกเลิกไลบรารีของทาง TensorFlow กับทาง Arduino เมื่อแก้ปัญหาในการหาไลบรารีลงก็มีความแตกต่างในเรื่องเวอร์ชั่นการใช้งานจึงได้ใช้งานแพลตฟอร์ม Edge Impulse ในการฝึกโมเดลและแปลงเป็นภาษาซีเพื่อลงบน Arduino แทน

เนื่องด้วยการเรียนรู้ของเครื่องบนไมโครคอนโทรลเลอร์ นั้นเกิดจากการแปลงข้อมูลให้เล็กลงแลกกับความเสถียรของโมเดลทำให้ในบางครั้งประสิทธิภาพอาจจะต่ำลงแต่ก็สามารถใช้งานกับการตรวจจับเสียงหรือการเคลื่อนไหว แน่นอนว่าไมโครคอนโทรลเลอร์สามารถเสริม IC ที่มีความสามารถต่าง ๆ ในอีกมากมายในอนาคตจึงสามารถนำไปประยุกต์กับการต่อ IC นอกเพื่อสร้างโมเดลที่หลากหลายการใช้งานอย่างการตรวจจับสีหรือสภาพอากาศโดยใช้การเรียนรู้ของเครื่องบนไมโครคอนโทรลเลอร์หรือ TinyML ได้

5.3 ข้อเสนอแนะ

ถึงแม้การเลือกนำสิ่งที่ได้ศึกษามาทำเป็นแบบเรียนจะสามารถตอบโจทย์ผู้ที่อยากเรียนรู้ในขั้นต้นได้ แต่มันก็เป็นเพียงการเผยแพร่ความรู้ในขั้นพื้นฐานสำหรับผู้ที่สนใจเท่านั้น จึงอาจมีการนำสิ่งที่ได้ศึกษามาทำเป็นตัวต้นแบบที่เป็นลักษณะชิ้นงานที่มีการรวมผลลัพธ์จากการทดลอง เช่น อุปกรณ์ตรวจจับความผิดพลาดของอุปกรณ์เพื่อการบำรุงรักษา หรือพัฒนา Library สำหรับการเรียนและนำไปใช้เพื่อพัฒนาโปรแกรมที่เกี่ยวกับ TinyML ต่อไป

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

บรรณานุกรม

Eda Kavlakoglu. 2020, May 27. **AI vs. Machine Learning vs. Deep Learning vs. Neural Networks: What's the Difference?** [On-line]. Available: https://www.ibm.com/cloud/blog/ai-vs-machine-learning-vs-deep-learning-vs-neural-networks

Jen, Namjatturas. 4 เมษายน 2019. **ทำความรู้จัก AI, Machine learning, Deep learning ฉบับเข้าใจง่าย** [ออนไลน์]. เข้าถึงได้จาก https://techsauce.co/tech-and-biz/ai-machine-learning-deep-learning-differences

Serdar Yegulalp. 2022, Jun 3. **What is TensorFlow? The machine learning library explained** [On-line]. Available: https://www.infoworld.com/article/3278008/what-is-tensorflow-the-machine-learning-library-explained.html

What is machine learning? [On-line]. Available: https://www.ibm.com/topics/machine-learning

M. Tim Jones. 2017, December 4. **Models for machine learning** [On-line]. Available: https://developer.ibm.com/articles/cc-models-machine-learning/#reinforcement-learning

Samhitha. 2023, January 5. **The 20 Pros and Cons of Machine Learning You Must Know in 2023** [On-line]. Available: https://digitalscholar.in/pros-and-cons-of-machine-learning/

Building the Future of TensorFlow [On-line]. 2022, October 20. Available: https://blog.tensorflow.org/2022/10/building-the-future-of-tensorflow.html?_gl=1*whr9jq*_ga*MjAzMDYxMDk0OS4xNjY0Nzc1MzY5*_ga_W0YLR4190T*MTY2OTcxMDA1OC41LjEuMTY2OTcxMTA3OS4wLjAuMA..

Daniel Situnayake. 2021, June 2. **TensorFlow helps Edge Impulse make ML accessible to embedded engineers** [On-line]. Available: https://blog.tensorflow.org/2021/06/how-tensorflow-helps-edge-impulse-make-ml-accessible.html

shuxu hu. 2023, Jan 11. **Getting Started with Seeed Studio XIAO nRF52840 (Sense)** [On-line]. Available: https://wiki.seeedstudio.com/XIAO_BLE/

shuxu hu. 2023, Jan 11. **Bluetooth Usage (Seeed nrf52 mbed-enabled Boards Library)** [On-line]. Available: https://wiki.seeedstudio.com/XIAO-BLE-Sense-Bluetooth-Usage/

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

MengDu. 2023 April 11. **Seeed Studio XIAO nRF52840 Sense Edge Impulse Getting Started** [On-line]. Available: https://wiki.seeedstudio.com/XIAOEI/

Continuous motion recognition [On-line]. 2022, October. Available: https://docs.edgeimpulse.com/docs/tutorials/continuous-motion-recognition

Responding to your voice [On-line]. 2022, December. Available: https://docs.edgeimpulse.com/docs/tutorials/responding-to-your-voice

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้