การพัฒนาระบบจดจำภาพสำหรับการตรวจจับอะไหล่ โดยใช้โครงข่ายประสาทเทียมแบบคอนโวลูชัน

นายอุเทน ว่องไว

วิทยานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาวิทยาศาสตรมหาบัณฑิต
แขนงวิชาเทคโนโลยีสารสนเทศและการสื่อสาร สาขาวิชาวิทยาศาสตร์และเทคโนโลยี มหาวิทยาลัยสุโขทัยธรรมาธิราช
พ.ศ. 2561

Development of an Image Recognition System for Spare Parts Detection Using Convolutional Neural Network

Mr. Utain Vongvai

A Thesis Submitted in Partial Fulfillment of the Requirements for

the Degree of Master of Science in Information and Communication Technology

School of Science and Technology

Sukhothai Thammathirat Open University

2018

หัวข้อวิทยานิพนธ์ การพัฒนาระบบจดจำภาพสำหรับการตรวจจับอะไหล่
โดยใช้โครงข่ายประสาทเทียมแบบคอนโวลูชัน

ชื่อและนามสกุล นายอุเทน ว่องไว

แขนงวิชา เทคโนโลยีสารสนเทศและการสื่อสาร

สาขาวิชา วิทยาศาสตร์และเทคโนโลยี มหาวิทยาลัยสุโขทัยธรรมาธิราช

อาจารย์ที่ปรึกษา 1. ผู้ช่วยศาสตราจารย์ ดร.วฤษาย์ ร่มสายหยุด
2. อาจารย์ ดร.ธงชัย แก้วกิริยา

วิทยานิพนธ์นี้ ได้รับความเห็นชอบให้เป็นส่วนหนึ่งของการศึกษา
ตามหลักสูตรระดับปริญญาโท เมื่อวันที่ 7 พฤศจิกายน 2561

คณะกรรมการสอบวิทยานิพนธ์

.....ประธานกรรมการ
(ผู้ช่วยศาสตราจารย์ ดร.วรลักษณ์ วงศ์โดยหวัง ศิริเจริญ)

.....กรรมการ
(ผู้ช่วยศาสตราจารย์ ดร.วฤษาย์ ร่มสายหยุด)

.....กรรมการ
(อาจารย์ ดร.ธงชัย แก้วกิริยา)

.....ประธานกรรมการบัณฑิตศึกษา
(รองศาสตราจารย์ ดร.กฤษณา รุ่งโรจน์วณิชย์)

ชื่อวิทยานิพนธ์ การพัฒนาระบบจดจำภาพสำหรับการตรวจจับอะไหล่โดยใช้โครงข่ายประสาทเทียมแบบคอนโวลูชัน

ผู้วิจัย นายอุเทน ว่องไว **รหัสนักศึกษา** 2599600364

ปริญญา วิทยาศาสตรมหาบัณฑิต (เทคโนโลยีสารสนเทศและการสื่อสาร)

อาจารย์ที่ปรึกษา (1) ผู้ช่วยศาสตราจารย์ ดร.วฤษาย์ ร่มสายหยุด (2) อาจารย์ ดร.ธงชัย แก้วกิริยา

ปีการศึกษา 2561

บทคัดย่อ

การวิจัยครั้งนี้มีวัตถุประสงค์ 1) เพื่อพัฒนาระบบจดจำภาพสำหรับการตรวจจับอะไหล่โดยใช้โครงข่ายประสาทเทียมแบบคอนโวลูชัน และ 2) เพื่อประเมินประสิทธิภาพของระบบจดจำภาพสำหรับตรวจจับอะไหล่โดยใช้โครงข่ายประสาทเทียมแบบคอนโวลูชัน

การวิจัยครั้งนี้ประกอบด้วยสองวิธีหลัก วิธีที่หนึ่งขยายความสามารถอัลกอริทึม จากหลักการของการจดจำรูปภาพตามวิธีการเรียนรู้การถ่ายโอน จากความรู้เกี่ยวกับแบบจำลองการถ่ายโอน นำข้อมูลจากแหล่งข้อมูลไปที่แบบจำลองเป้าหมาย วิธีนี้ทำการฝึกฝนข้อมูลบนชุดข้อมูลขนาดใหญ่เพื่อตรวจจับชิ้นส่วนอะไหล่ โดยการฝึกฝนจากภาพชิ้นส่วนอะไหล่จำนวน 500 – 1,200 ภาพ และวิธีที่สองดำเนินการโดยการสร้างและการทดสอบแบบจำลอง ซึ่งแบ่งกลุ่มอะไหล่ออกเป็น 5 หมวดหมู่ 1) อะไหล่ขนาดเล็ก 2) อะไหล่พลาสติก 3) อะไหล่โลหะ 4) อะไหล่แผ่นโลหะ และ 5) อะไหล่แท่งเฟอร์ไรต์ การดำเนินการนี้ประยุกต์ใช้โครงข่ายประสาทเทียมตามวิธีการเรียนรู้การถ่ายโอนเพื่อเพิ่มประสิทธิภาพของระบบตรวจจับภาพอะไหล่ และแสดงคำอธิบายชิ้นส่วนอะไหล่แบบเรียลไทม์

ผลการวิจัยนี้มีค่าความแม่นยำร้อยละ 87.44 ค่าเรียกคืนร้อยละ 86.76 ค่าความถูกต้องร้อยละ 86.76 และการวัดประสิทธิภาพโดยรวมร้อยละ 86.64 ตามลำดับ

คำสำคัญ การจดจำภาพ อัลกอริทึมการจดจำภาพ โครงข่ายประสาทเทียมแบบคอนโวลูชัน การเรียนรู้เชิงถ่ายโอนข้อมูล

Thesis title: Development of an Image Recognition System for Spare Parts Detection Using Convolutional Neural Network
Researcher: Mr. Utain Vongvai; **ID:** 2599600364;
Degree: Master of Science (Information and Communication Technology);
Thesis advisors: (1) Dr. Walisa Romsaiyud, Assistant Professor;
(2) Dr. Thongchai Kaewkiriya; **Academic year**: 2018

Abstract

The purposes of this research were as follows: 1) to develop an image recognition system for spare parts detection using convolutional neural network, and 2) to evaluate the effectiveness of the image recognition system for spare parts detection using convolution neural network.

This research consisted of two main methods. The first method extended an algorithm from principle approach of an image recognition based on the transfer learning method by the knowledge of transfer model from source to target task model. This method was trained on a large dataset to detect spare parts by training 500 – 1,200 photos. The second method was carried out by generating and testing the model that was operated by categorizing groups of spares into 5 categories as 1) small spares 2) plastic spares 3) metal spares 4) metal sheet spares, and 5) ferrite bar spares. This method applied the convolutional neural network based on transfer learning method for increasing the performance of image detection system and displaying spare part description in real-time.

The measurement results were 87.44% for precision, 86.76% for recall, 86.76% for accuracy, and 86.64 for F-measure respectively.

Keywords: Image recognition, Image recognition algorithm, Convolution neural network, Transfer learning

กิตติกรรมประกาศ

วิทยานิพนธ์เล่มนี้ ผู้วิจัยได้รับความอนุเคราะห์อย่างดียิ่งจากผู้ช่วยศาสตราจารย์ ดร. วฤษาย์ ร่มสายหยุด และอาจารย์ ดร.ธงชัย แก้วกิริยา อาจารย์ที่ปรึกษาวิทยานิพนธ์ ที่ได้กรุณาให้คำปรึกษา ติดตาม ให้คำแนะนำ และตรวจสอบการทำวิทยานิพนธ์ครั้งนี้อย่างใกล้ชิดตลอดมา นับตั้งแต่เริ่มต้นจนกระทั่งสำเร็จเรียบร้อยสมบูรณ์ ผู้วิจัยรู้สึกซาบซึ้งในความกรุณาของท่านเป็นอย่างยิ่ง

ผู้วิจัยขอขอบคุณเพื่อนที่ทำงานและเพื่อนนักศึกษาแขนงวิชาเทคโนโลยีสารสนเทศและการสื่อสาร ที่ให้ความช่วยเหลือและเป็นกำลังใจที่ดีเสมอมา

สุดท้ายนี้ขอกราบขอบพระคุณ คุณย่า บิดามารดา ที่ให้การสนับสนุน อบรมสั่งสอน และเป็นกำลังใจในการศึกษา ตลอดจนเป็นผู้ที่คอยให้โอกาสที่ดีในชีวิตแก่ผู้วิจัย ประโยชน์ของวิทยานิพนธ์เล่มนี้ผู้วิจัยขอมอบบูชาพระคุณของคุณย่า บิดามารดา และคณาจารย์ทุกท่าน ที่ได้อบรมชี้แนะแนวทางที่ดีให้ผู้วิจัยในการดำเนินชีวิตตลอดมา

อุเทน ว่องไว

พฤศจิกายน 2561

สารบัญ

หน้า

บทคัดย่อภาษาไทย ..... ง
บทคัดย่อภาษาอังกฤษ ..... จ
กิตติกรรมประกาศ ..... ฉ
สารบัญตาราง ..... ฌ
สารบัญภาพ ..... ญ
บทที่ 1 บทนำ 1
ความเป็นมาและความสำคัญของปัญหา 1
วัตถุประสงค์การวิจัย 3
กรอบแนวคิดงานวิจัย 3
ขอบเขตของการวิจัย 4
นิยามศัพท์เฉพาะ 5
ประโยชน์ที่จะได้รับ 5
บทที่ 2 วรรณกรรมที่เกี่ยวข้อง 6
ทฤษฎีที่เกี่ยวข้อง 6
ภาพและพิกเซล 6
โครงข่ายประสาทเทียม 9
โครงข่ายประสาทเทียมแบบคอนโวลูชัน 13
การเรียนรู้เชิงถ่ายโอนข้อมูล 18
งานวิจัยที่เกี่ยวข้อง 21
บทที่ 3 วิธีดำเนินการวิจัย 27
การศึกษาและวิเคราะห์ปัญหา 27
การวิเคราะห์และขั้นตอนการทำงานของระบบ 33
ข้อมูลและกลุ่มตัวอย่างที่ใช้ในงานวิจัย 42
การพัฒนาระบบรู้จำอะไหล่ 51
การทดสอบประสิทธิภาพของระบบ 55
บทที่ 4 ผลการดำเนินงาน 58
การเก็บและบันทึกข้อมูลรูปภาพเพื่อใช้ฝึกฝนตัวแบบ 58

สารบัญ (ต่อ)

หน้า

ส่วนที่ใช้ในการสร้างตัวแบบ 61
ส่วนที่ใช้ในการแสดงผลลัพธ์ 64
ผลการประเมินประสิทธิภาพของระบบ 68
บทที่ 5 สรุปการวิจัย อภิปรายผล และข้อเสนอแนะ 78
สรุปการวิจัย 78
ข้อเสนอแนะ 81
ปัญหาและอุปสรรค 82
บรรณานุกรม 83
ภาคผนวก 87
ประวัติผู้วิจัย 122

สารบัญตาราง

หน้า

ตารางที่ 3.1 System flow ระบบการเบิกจ่ายอะไหล่ปัจจุบันกรณีทราบหมายเลขอะไหล่ 28
ตารางที่ 3.2 System flow ของระบบการเบิกจ่ายอะไหล่ปัจจุบันกรณีไม่ทราบหมายเลขอะไหล่ 29
ตารางที่ 3.3 System flow ของระบบการเบิกจ่ายอะไหล่ใหม่กรณีทราบหมายเลขอะไหล่ 30
ตารางที่ 3.4 System flow ของระบบการเบิกจ่ายอะไหล่ใหม่กรณีไม่ทราบหมายเลขอะไหล่ .. 31
ตารางที่ 3.5 แสดงคอนฟิวชันเมทริกซ์ของการจำแนกแบบ 3 คลาส 55
ตารางที่ 4.1 การเปรียบเทียบค่าความแม่นยำ (Precision) และค่าเรียกคืน (Recall) ของจำนวนรอบของการหาเอาต์พุต 77

สารบัญภาพ

หน้า

ภาพที่ 1.1 แสดงกรอบแนวคิดภาพรวมของงานวิจัย 3
ภาพที่ 2.1 แสดงความแตกต่างระหว่างสิ่งที่มนุษย์เห็นและคอมพิวเตอร์เห็น 7
ภาพที่ 2.2 แสดงหนึ่งหน่วยประสาท (neural) 10
ภาพที่ 2.3 ดัดแปลงมาจากสถาปัตยกรรมของโครงข่ายประสาทเทียม 11
ภาพที่ 2.4 ตัวอย่างการทำคอนโวลูชัน โดยมีขนาดของข้อมูลรับเข้าขนาด 7 x 7 และเมทริกซ์ตัวกรองขนาด 3 x 3 13
ภาพที่ 2.5 การทำคอนโวลูชันแบบกว้างและการเสริมเติม 14
ภาพที่ 2.6 การทำคอนโวลูชันโดยมีข้อมูลรับเข้าขนาด 5 x 5 ตัวกรอง 3 x 3 และมีขนาดของการก้าวข้ามเป็น 2 15
ภาพที่ 2.7 การทำคอนโวลูชันโดยมีจำนวนตัวกรองเท่ากับ 3 15
ภาพที่ 2.8 ชั้นการรวมโดยใช้ค่ามากสุดใน 2 มิติ 17
ภาพที่ 2.9 ชั้นการรวมโดยใช้เคค่ามากสุดใน 1 มิติ โดยกาหนดให้ k = 2 17
ภาพที่ 2.10 การเรียนรู้เชิงถ่ายโอนข้อมูล 19
ภาพที่ 2.11 แสดงชั้นการเรียนรู้เชิงถ่ายโอนข้อมูล 21
ภาพที่ 3.1 แสดงโครงสร้างของระบบ 33
ภาพที่ 3.2 แสดงตัวอย่างการทำงานของระบบในส่วนของผู้ใช้งาน 37
ภาพที่ 3.3 โครงสร้างของตัวแบบ Inception V3 38
ภาพที่ 3.4 แสดงการทำงานของชั้นคอนโวลูชัน 39
ภาพที่ 3.5 แสดงการทำงานของชั้น AvgPool (Average pooling) และ MaxPool (Max Pooling) 39
ภาพที่ 3.6 แสดงการทำงานของชั้น Concat (concatenate) 40
ภาพที่ 3.7 แสดงการเปรียบเทียบการทำงานของโครงข่ายก่อนและหลังการใส่ชั้น Dropout 41
ภาพที่ 3.8 แสดงตัวอย่างของข้อมูลภาพอะไหล่ที่มีขนาดเล็ก 43
ภาพที่ 3.9 แสดงตัวอย่างของข้อมูลภาพอะไหล่ที่เป็นพลาสติก 44
ภาพที่ 3.10 แสดงตัวอย่างของข้อมูลภาพอะไหล่ที่เป็นแท่งโลหะ 45
ภาพที่ 3.11 แสดงตัวอย่างของข้อมูลภาพอะไหล่ที่เป็นแผ่นโลหะ 46
ภาพที่ 3.12 แสดงตัวอย่างของข้อมูลภาพอะไหล่ที่เป็นแท่งเฟอร์ไรท์ 47

สารบัญภาพ (ต่อ)

หน้า

ภาพที่ 3.13 แสดงตัวอย่างของข้อมูลอะไหล่แท่งโลหะ 48
ภาพที่ 3.14 แสดงตัวอย่างของข้อมูลอะไหล่ขนาดเล็ก 48
ภาพที่ 3.15 แสดงตัวอย่างของข้อมูลอะไหล่พลาสติก 49
ภาพที่ 3.16 แสดงตัวอย่างของข้อมูลอะไหล่แผ่นโลหะ 50
ภาพที่ 3.17 แสดงตัวอย่างของข้อมูลอะไหล่แท่งเฟอร์ไรท์ 50
ภาพที่ 3.18 ยูสเคสไดอะแกรมของระบบจดจำภาพสำหรับการตรวจจับอะไหล่ 54
ภาพที่ 4.1 แสดงขั้นตอนการเตรียมเก็บและบันทึกภาพอะไหล่ 58
ภาพที่ 4.2 แสดงอะไหล่ที่ถูกเก็บไว้ในแฟ้มข้อมูล 59
ภาพที่ 4.3 แสดงภาพอะไหล่ที่ถูกเก็บไว้ในแฟ้มข้อมูล 60
ภาพที่ 4.4 แสดงภาพการตั้งชื่อแฟ้มข้อมูล 60
ภาพที่ 4.5 แสดงภาพอะไหล่ที่อยู่ในแฟ้มที่เป็นข้อมูลนำเข้าเพื่อฝึกฝน 61
ภาพที่ 4.6 แสดงภาพที่โปรแกรมสร้างเพื่อเก็บข้อมูล image feature vector 62
ภาพที่ 4.7 แสดงภาพที่โปรแกรมสร้างไฟล์ Text เพื่อเก็บข้อมูล image feature vector 63
ภาพที่ 4.8 แสดงภาพตัวอย่างข้อมูล image feature vector ซึ่งเก็บไว้ในไฟล์ Text 63
ภาพที่ 4.9 แสดงภาพการเตรียมอะไหล่ที่ไม่รู้จักหมายเลขหรือรหัสและชื่อเพื่อทำนาย 65
ภาพที่ 4.10 แสดงการเรียงลำดับข้อมูลที่คำนวณได้จากชั้น Softmax 65
ภาพที่ 4.11 แสดงผลลัพธ์การกำหนดเงื่อนไขก่อนการแสดงผลการพยากรณ์ 66
ภาพที่ 4.12 แสดงภาพที่ใช้อะไหล่หมายเลข 603409 มาให้โปรแกรมวิเคราะห์ และผลการวิเคราะห์ตรงกัน 67
ภาพที่ 4.13 แสดงภาพที่ใช้อะไหล่หมายเลข 603382 มาให้โปรแกรมวิเคราะห์ และผลการวิเคราะห์ตรงกัน 68
ภาพที่ 4.14 แสดงภาพรวมของขั้นตอนการทดสอบระบบจดจำภาพอะไหล่ 70
ภาพที่ 4.15 แสดงการคำนวณหาค่าความแม่นยำ (Precision) และค่าเรียกคืน (Recall) ของจำนวนรอบของการหาเอาท์พุตเท่ากับ 4000 71
ภาพที่ 4.16 แสดงการคำนวณหาค่าความแม่นยำ (Precision) และค่าเรียกคืน (Recall) ของจำนวนรอบของการหาเอาท์พุตเท่ากับ 10000 73

สารบัญภาพ (ต่อ)

หน้า

ภาพที่ 4.17 แสดงการคำนวณหาค่าความแม่นยำ (Precision) และค่าเรียกคืน (Recall) ของจำนวนรอบของการหาเอาท์พุตเท่ากับ 20000 75

บทที่ 1
บทนำ

1. ความเป็นมาและความสำคัญของปัญหา

การทำธุรกิจในปัจจุบันมีการแข่งขันที่สูง การส่งสินค้าหรือบริการให้ทันต่อความต้องการของลูกค้าถือว่าเป็นปัจจัยหนึ่งที่สำคัญต่อการอยู่รอดขององค์กร การนำเทคโนโลยีสารสนเทศเข้ามาใช้ทำให้เพิ่มขีดความสามารถให้กับองค์กรธุรกิจเกี่ยวกับการผลิตอุปกรณ์รับส่งสัญญาณความถี่ต่ำในรถยนต์ถือเป็นการสร้างความได้เปรียบในการแข่งขัน ในสายพานการผลิตจะมีเครื่องจักรเพื่อช่วยในการผลิตเป็นจำนวนมาก แต่ละเครื่องทำหน้าที่แตกต่างกันไป ดังนั้นสิ่งที่มีความเกี่ยวข้องโดยตรงกับชิ้นงานการผลิตคืออะไหล่ อะไหล่บางชิ้นสัมผัสกับชิ้นงานโดยตรงซึ่งจะต้องถูกเปลี่ยนหลังจากครบกำหนดที่ต้องเปลี่ยน เช่น ถูกเปลี่ยนเมื่อผลิตชิ้นงานครบ 50,000 ชิ้น เป็นต้น และสาเหตุอีกประการที่จะต้องเปลี่ยนอะไหล่คือการชำรุด แตก หัก อะไหล่แต่ละชิ้นมีลักษณะที่หลากหลาย ทั้งขนาด สี รูปทรงที่แตกต่างกันไปทำให้ยากในการค้นหารายละเอียดของอะไหล่เพื่อทำการเบิกและเปลี่ยนได้ทันที ซึ่งการทำให้การผลิต ผลิตชิ้นงานออกมาได้อย่างต่อเนื่องโดยไม่มีปัญหาใด ๆ ย่อมเป็นสิ่งที่ทุกองค์กรปรารถนา แต่ในความเป็นจริงแล้วอุปสรรคและปัญหาอาจเกิดขึ้นได้ทุกเวลา ทั้งที่โดยตั้งใจและไม่ได้ตั้งใจจากพนักงาน ทั้งที่สามารถควบคุมได้และควบคุมไม่ได้จากสถานการณ์ต่าง ๆ เครื่องจักรก็เป็นปัจจัยหนึ่งของกระบวนการผลิตซึ่งมีการใช้งานทุกวัน โอกาสที่ชิ้นส่วนหรืออะไหล่เกิดการชำรุดเสียหายจึงมีสูง เมื่ออะไหล่เกิดชำรุดเสียหายกระบวนการผลิตจึงหยุดลงเพื่อรอการเปลี่ยนอะไหล่ แต่ในการเบิกอะไหล่จะต้องใช้หมายเลขอะไหล่ที่ลงทะเบียนไว้ในระบบการวางแผนทรัพยากรองค์กร (Enterprise resource planning : ERP) ของบริษัทที่นำมาใช้ในการจัดการทรัพยากรขององค์กร ทุกครั้งที่พนักงานจะเบิกแต่ไม่สามารถบอกได้ว่าอะไหล่ที่จะเบิกนั้นหมายเลขอะไร เพราะไม่ได้ระบุไว้ในตัวอะไหล่ จำเป็นจะต้องให้พนักงานหรือเจ้าหน้าที่สโตร์ค้นหาโดยใช้คำค้นในส่วนรายละเอียดของอะไหล่ในการค้นหาในระบบซึ่งบางครั้งอาจตรงกันหลายตัว จำเป็นจะต้องไปหาทีละชิ้นในชั้นเก็บที่ระบุไว้ในระบบ และบางครั้งในการลงทะเบียนอะไหล่เข้าระบบ การป้อนข้อมูลรายละเอียดของอะไหล่ไม่ได้สื่อถึงรายละเอียดของอะไหล่ชิ้นนั้นมีผลทำให้การค้นหาเสียเวลามาก กรณีจะค้นหาที่คู่มือของเครื่องจักรก็ไม่สามารถทำได้เพราะหมายเลขอะไหล่ในคู่มือไม่ได้สอดคล้องกับหมายเลขที่ใช้การ

จัดเก็บในระบบของบริษัท ผลกระทบทำให้การผลิตหยุดชงัก หรือเกิดความล่าช้า อาจส่งสินค้าไม่ได้ตรงตามเวลาที่ลูกค้าต้องการได้ ทำให้เสียโอกาสทางธุรกิจ

จากปัญหาดังกล่าวผู้วิจัยจึงได้ประยุกต์ใช้เทคโนโลยีจดจำภาพ (Image recognition) ซึ่งเป็นประเภทหนึ่งของการเรียนรู้ของเครื่องจักร (Machine learning) มาทำการแก้ไขปัญหานี้ การใช้เทคโนโลยีนี้สิ่งสำคัญคือการสกัดคุณลักษณะสำคัญของภาพก่อนเข้าสู่กระบวนการจำแนกประเภทเพื่อใช้แยกประเภทหรือใช้ในการรู้จำรูปภาพ ซึ่งงานวิจัยส่วนใหญ่มุ่งเน้นพัฒนาในส่วนการสกัดลักษณะเด่นของภาพเพื่อให้ความแม่นยำสูง บางครั้งมีการผสมผสานหลากหลายวิธีในการสกัดคุณลักษณะเด่น และใช้จำนวนภาพที่นำมาฝึกฝนเป็นจำนวนมาก มีผลทำให้การฝึกฝนใช้เวลานาน เช่น วิธีการสกัดคุณลักษณะเด่นของภาพโดยวิธีการ Scale Invariant Feature Transformer : SIFT (ชัชวาลย์ วรวิทย์รัตนกุล, 2557) เมื่อทำการหาความสอดคล้องจนได้ครบทุกจุดในภาพแล้ว ก็อาจพบว่าจุดบางจุดที่เป็นจุดที่อาจอยู่นอกความสนใจในการพิจารณา ซึ่งมีผลทำให้เกิดความผิดพลาดในกระบวนการประมาณความถูกต้องของภาพ จึงต้องทำการวิเคราะห์เพื่อคัดจุดที่อยู่นอกความสนใจออก วิธีการ SIFT จะไม่สามารถสกัดคุณลักษณะเด่นได้อย่างมีประสิทธิภาพ สำหรับงานวิจัยที่สกัดคุณลักษณะเด่นด้วยวิธี ใช้ Histogram of Oriented Gradients : HOG ในการค้นคืนภาพพบว่าการเปรียบเทียบสีในฮิสโตแกรมแบบสีต่อสี โดยไม่คำนึงถึงข้อมูลสีข้างเคียงในฮิสโตแกรมทำให้เกิดความผิดพลาด (ประภาพร กุลลิ้มรัตน์ชัย, 2551) การใช้วิธี Pyramid Histogram of Oriented Gradients : PHOG สามารถช่วยเพิ่มความแม่นยำแต่มีค่าการเรียกคืนไม่สูงมากนัก วิธีรู้จำสองชั้นโดยอาศัยวิธีการ Convolutional neural network : CNN ซึ่งเป็นโครงข่ายประสาทเทียมประเภทหนึ่งที่สามารถใช้สกัดคุณลักษณะสำคัญของภาพและคุณลักษณะเด่นพิรามิดฮิสโตแกรม (วศิน ทับแสง, 2557) ซึ่งให้ความแม่นยำที่สูงและใช้ภาพในการฝึกฝนจำนวน 3,000 ถึง 7,000 ภาพ โดยค่าความแม่นยำจะลดลงเมื่อจำนวนคลาสเพิ่มขึ้น และระบบจะทำงานได้อย่างไม่มีประสิทธิภาพหากมีการถ่ายภาพในลักษณะก้มลงด้านล่างหรือมุมเงยขึ้นด้านบนหรือถ่ายจากด้านหลัง อีกทั้งยังไม่สามารถพัฒนาในรูปแบบเรียลไทม์ได้

ดังนั้นงานวิจัยนี้นำเสนอการพัฒนาระบบจดจำภาพสำหรับการตรวจจับอะไหล่โดยใช้โครงข่ายประสาทเทียมแบบคอนโวลูชันด้วยวิธีการที่เรียกว่า การเรียนรู้เชิงถ่ายโอนข้อมูล (Transfer Learning) กล่าวคือเป็นการใช้ตัวแบบของโครงข่ายประสาทเทียมแบบคอนโวลูชันที่ได้รับการฝึกฝนมาแล้ว ตัวแบบที่ใช้คือ Inception V3 เป็นตัวแบบหรือแบบจำลองของโครงข่ายประสาทเทียมแบบคอนโวลูชันซึ่งถูกพัฒนาโดยกูเกิลซึ่งได้จากการสอนโดยรูปภาพจำนวน 1.2 ล้านภาพใช้ฝึกฝนบนหน่วยประมวลผลด้านกราฟฟิก เพื่อใช้ในการจำแนกประเภทรูปภาพทั้งหมด 1,000 ภาพ ดังนั้นด้วยวิธีการนี้จะใช้ส่วนที่สกัดคุณลักษณะของอะไหล่ด้วยตัวแบบนี้พร้อมทั้ง

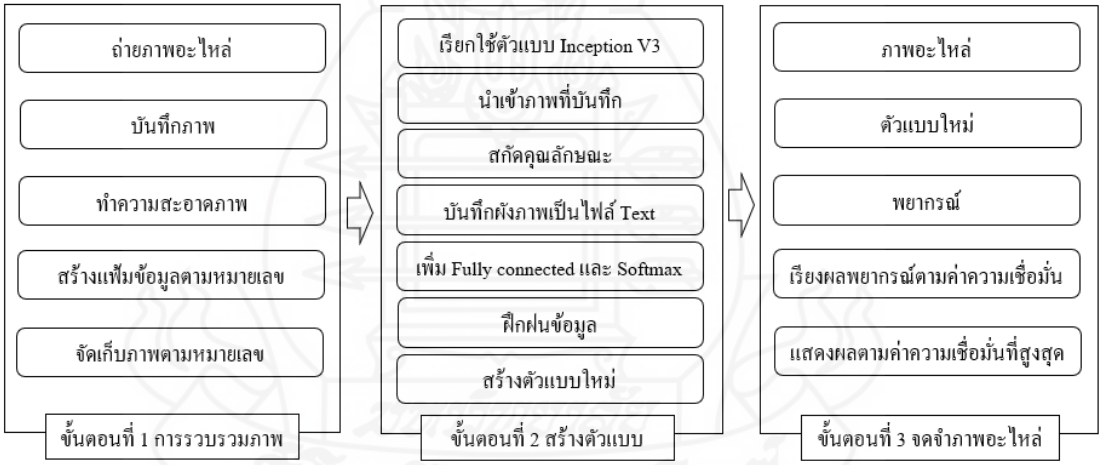
ฝึกฝนส่วนรู้จำด้วยข้อมูลอะไหล่ด้วยส่วนการจำแนกประเภท โดยใช้รูปภาพอะไหล่ 500 ภาพเพื่อทำการฝึกฝน และพัฒนาในรูปแบบเรียลไทม์ ซึ่งจะเหมาะสมกับสถานการณ์ปัจจุบันที่ต้องการความรวดเร็ว เพื่อความได้เปรียบเชิงการแข่งขัน

2. วัตถุประสงค์การวิจัย

2.1 เพื่อพัฒนาระบบจดจำภาพสำหรับการตรวจจับอะไหล่โดยใช้โครงข่ายประสาทเทียมแบบคอนโวลูชัน

2.2 เพื่อประเมินความถูกต้องของค่าความแม่นยำและค่าเรียกคืนของระบบจดจำภาพสำหรับตรวจจับอะไหล่โดยใช้โครงข่ายประสาทเทียมแบบคอนโวลูชัน

3. กรอบแนวคิดงานวิจัย



ภาพที่ 1.1 แสดงกรอบแนวคิดภาพรวมของงานวิจัย

จากภาพที่ 1.1 กรอบแนวความคิดของงานวิจัยแบ่งออกเป็น 3 ขั้นตอนดังนี้

3.1 ขั้นตอนที่ 1 การรวบรวมภาพ เป็นขั้นตอนที่นำอะไหล่มาถ่ายภาพและบันทึกไว้ในฮาร์ดดิสก์ เพื่อทำความสะอาดข้อมูลภาพโดยการคัดเลือกภาพที่มีภาพอะไหล่อยู่ในส่วนของภาพทั้งหมด และนำภาพที่ได้ทำความสะอาดแล้วมาจัดเก็บแยกตามคลาสหรือตามหมายเลขของอะไหล่ในแฟ้มข้อมูล

3.2 ขั้นตอนที่ 2 ส่วนของการสร้างตัวแบบ โดยการเรียกใช้ตัวแบบ Inception V3 ที่ได้รับการฝึกฝนมาแล้วมาทำการสกัดคุณลักษณะของภาพและจัดเก็บในรูปแบบของไฟล์ Text เพื่อจะใช้ในการฝึกฝน ทำการเพิ่มชั้นของ Fully connected และ Softmax เพื่อใช้ไฟล์ที่ผ่านการสกัดคุณลักษณะแล้วมาทำการฝึกฝนเป็นข้อมูลอะไหล่ และทำการบันทึกเป็นไฟล์ตัวแบบเพื่อนำไปใช้งานจดจำอะไหล่ต่อไป

3.3 ขั้นตอนที่ 3 จดจำภาพอะไหล่ เป็นการเรียกใช้ตัวแบบที่ได้จากขั้นตอนที่ 2 เพื่อใช้ในการจดจำอะไหล่ โดยการทำงานจะพยากรณ์อะไหล่ตัวที่นำมาทดสอบ ผลที่ได้จะเป็นหมายเลข ชื่ออะไหล่ และค่าความเชื่อมั่น จากนั้นให้ระบบเรียงลำดับจากความเชื่อมั่นสูง และแสดงผลของอะไหล่ที่พยากรณ์ได้จากค่าความเชื่อมั่นที่สูงนั้น

4. ขอบเขตของการวิจัย

4.1 ขอบเขตด้านเทคนิควิธีการ

4.1.1 ขอบเขตด้านตัวแบบหลัก โดยการใช้ตัวแบบที่ได้รับการฝึกฝนมาแล้วชื่อ Inception v3 มาทำการถ่ายโอนเชิงการเรียนรู้

4.1.2 ขอบเขตด้านการหาคุณลักษณะสำคัญ เป็นการถ่ายความรู้จากตัวแบบหลักที่ได้จากการฝึกฝนข้อมูลขนาดใหญ่ของโครงข่ายประสาทเทียมเพื่อนำมาสกัดคุณลักษณะสำคัญของภาพของข้อมูลใหม่

4.1.3 ขอบเขตด้านการสร้างตัวแบบใหม่ โดยการเพิ่มชั้น Fully connected กับ Softmax เพื่อฝึกฝนข้อมูลใหม่ โดยใช้ค่าน้ำหนักและไบแอสของโครงข่ายประสาทเทียมในตัวแบบหลักเป็นค่าเริ่มต้นในการฝึกฝน

4.2 ขอบเขตด้านข้อมูล

4.2.1 ขอบเขตของกลุ่มตัวอย่าง การวิจัยครั้งนี้เป็นการศึกษากับกลุ่มตัวอย่างของอะไหล่ที่แบ่งกลุ่มออกเป็นจำนวน 5 กลุ่ม ๆ ละ 5 ชิ้น รวมทั้งหมด 25 ชิ้น ตามลักษณะรูปทรงของอะไหล่ และ/หรือวัสดุที่ใช้ทำอะไหล่ และถ่ายภาพอะไหล่เพื่อนำมาวิจัยอะไหล่ชิ้นละ 500 ภาพ รวมจำนวนรูปภาพอะไหล่ทั้งหมด 12,500 ภาพ

4.2.2 ขอบเขตของพื้นหลังของภาพ (Background) เป็นการศึกษาข้อมูลโดยใช้การทดลองกับภาพที่มีพื้นหลังเป็นสีขาว

4.2.3 ขอบเขตของระยะห่างจากกล้องถึงอะไหล่ เป็นการศึกษาข้อมูลกับการทดลองที่ระยะห่างจากกล้องถึงอะไหล่ 5 – 15 เซนติเมตร

4.3 ขอบเขตด้านระบบงาน

4.3.1 ส่วนของผู้ใช้งาน ใช้แสดงหมายเลขและชื่อของอะไหล่พร้อมทั้งเปอร์เซ็นต์ค่าความเชื่อมั่นของการพยากรณ์

4.3.2 ส่วนของผู้ดูแลระบบ

1) ปรับปรุงข้อมูลรูปภาพอะไหล่ โดยการลบแฟ้มข้อมูลสำหรับอะไหล่เก่าไม่ใช้งานในระบบ และเพิ่มแฟ้มข้อมูลสำหรับอะไหล่ที่เข้ามาใหม่

2) ปรับปรุงตัวแบบสำหรับส่วนของผู้ใช้งานหลังการเพิ่มข้อมูลอะไหล่ใหม่หรือลบข้อมูลรูปภาพของอะไหล่เก่าที่เลิกการใช้งานไปแล้ว

5. นิยามศัพท์เฉพาะ

5.1 โครงข่ายประสาทเทียมแบบคอนโวลูชัน (Convolutional Neural Network : CNN) คือเทคนิคการสกัดคุณลักษณะสำคัญและใช้ในการจำแนกรูปภาพ

5.2 การเรียนรู้เชิงถ่ายโอนข้อมูล (Transfer learning) เป็นเทคนิคที่ใช้ความรู้จากการฝึกฝนข้อมูลจำนวนมากก่อนหน้า มาทำการเปลี่ยนค่าเป้าหมายและทำการฝึกฝนซ้ำด้วยข้อมูลอีกชุด

5.3 การรู้จำรูปภาพ (Image recognition) คือความสามารถในการเก็บข้อมูลรูปภาพและใช้ข้อมูลรูปภาพในการค้นหาและเรียกข้อมูลออกมาใช้

5.4 อะไหล่ (Spare part) หรืออุปกรณ์หรือส่วนประกอบของเครื่องจักรที่จัดเตรียมไว้ใช้เมื่อจำเป็นหรือเมื่อของเก่าชำรุดเสียหายไป ซึ่งอะไหล่ที่ใช้ในงานวิจัยนี้จะแบ่งออกเป็น 5 กลุ่ม คืออะไหล่ขนาดเล็ก อะไหล่พลาสติก อะไหล่แท่งโลหะ อะไหล่แผ่นโลหะ และอะไหล่แท่งเฟอร์ไรท์ ซึ่งเป็นกลุ่มตัวอย่างของอะไหล่ทั้งหมดของบริษัท

6. ประโยชน์ที่จะได้รับ

6.1 ได้ระบบจดจำภาพสำหรับการตรวจจับอะไหล่โดยใช้เทคนิคโครงข่ายประสาทเทียมแบบคอนโวลูชัน

6.2 ได้ระบบจดจำภาพสำหรับการตรวจจับอะไหล่โดยใช้เทคนิคโครงข่ายประสาทเทียมแบบคอนโวลูชันที่มีการทดสอบค่าความถูกต้องของความถูกต้องและค่าเรียกคืน

บทที่ 2
วรรณกรรมที่เกี่ยวข้อง

เนื้อหาส่วนนี้แบ่งออกเป็น 2 ส่วนหลัก ส่วนที่หนึ่งเป็นส่วนที่รวบรวมทฤษฎี หลักการและข้อมูลต่าง ๆ ที่เกี่ยวข้องกับงานวิจัย โดยครอบคลุมเจาะลึกเรื่องที่จะศึกษาเพื่อให้เกิดความเข้าใจ ส่วนที่สองเป็นงานวิจัยที่เกี่ยวข้องเป็นส่วนที่นำเสนอผลงานวิจัยที่มีผู้ทำมาก่อนเพื่อใช้เป็นแนวทางในการปรับปรุงและพัฒนางานวิจัย ดังนี้

1. ทฤษฎีที่เกี่ยวข้อง ประกอบด้วยเนื้อหา 1) ภาพและพิกเซล 2) โครงข่ายประสาทเทียม 3) โครงข่ายประสาทเทียมแบบคอนโวลูชัน และ 4) การเรียนรู้เชิงถ่ายโอนข้อมูล

2. งานวิจัยที่เกี่ยวข้อง

1. ทฤษฎีที่เกี่ยวข้อง

1.1 ภาพและพิกเซล

พิกเซล (Pixel) (อุษนีย์ สังฆธรรม, 2556) คือจุดหรือเม็ดสีที่ประกอบกันทำให้เกิดเป็นภาพ ภาพหนึ่ง ๆ จะประกอบด้วยพิกเซลมากมาย ซึ่งภาพแต่ละภาพที่สร้างขึ้นจะมีความหนาแน่นของพิกเซลเหล่านี้แตกต่างกันออกไป ความหนาแน่นนี้เป็นตัวบอกถึงความละเอียด (Resolution) ของภาพ ซึ่งมีหน่วยเป็น ppi (Pixel Per Inch) คือจำนวนพิกเซลต่อนิ้ว ถ้าค่า ppi ยิ่งสูงภาพก็จะมีความละเอียดและคมชัดมากขึ้น และในหนึ่งพิกเซลจะเป็นจุดสีหนึ่งสีใดเท่านั้นจะมีสีอื่นไม่ได้ ซึ่งแต่ละภาพที่ถูกสร้างขึ้นจากความหนาแน่นของจุดภาพดังกล่าวบางครั้งเรียกว่า ความคมชัด โดยมากจะระบุเป็นจำนวนพิกเซลด้านแนวนอนคูณด้านแนวตั้ง และใช้เป็นการบอกถึงคุณสมบัติของจอภาพหรืออุปกรณ์แสดงผลของภาพ

ภาพที่ 2.1 แสดงความแตกต่างระหว่างสิ่งที่มนุษย์เห็นและคอมพิวเตอร์เห็น

ที่มา: https://adeshpande3.github.io/A-Beginner%27s-Guide-To-Understanding-Convolutional-Neural-Networks/

จากภาพที่ 2.1 แสดงตัวอย่างของความแตกต่างระหว่างสิ่งที่มนุษย์เห็นกับสิ่งที่คอมพิวเตอร์เห็น โดยภาพทางด้านซ้ายจะเป็นสิ่งที่มนุษย์เห็นจะอาศัยหลักการของการตกกระทบของแสงผ่านดวงตาและประมวลผลด้วยสมองเพื่อให้สามารถระบุว่าวัตถุนั้นคืออะไร ส่วนภาพทางด้านขวาจะเป็นสิ่งที่คอมพิวเตอร์เห็นโดยจะเห็นเป็นค่าตัวเลข ตัวเลขที่เก็บนั้นอยู่ในรูปแบบของอาร์เรย์ของจุดสีและค่านั้นจะขึ้นอยู่กับความละเอียดและขนาดของรูปภาพ และคอมพิวเตอร์ไม่สามารถระบุได้ว่าวัตถุหรือภาพนั้นคืออะไร

การหาลักษณะสำคัญของภาพมีหลากหลายวิธีดังนี้

1. ลักษณะสำคัญของลวดลาย (Texture Feature)

ลวดลายภายในภาพเป็นหนึ่งในลักษณะสำคัญที่ใช้ในการระบุวัตถุ หรือขอบเขตที่น่าสนใจในภาพ ไม่ว่าจะเป็นภาพถ่ายทางอากาศ (Aerial Photograph) ภาพถ่ายจากดาวเทียม (Satellite Images) หรือภาพถ่ายทางการแพทย์ (Medical Images) ซึ่งลวดลายในภาพเหล่านี้สามารถอธิบายถึงคุณสมบัติของภาพ และสามารถตีความหรือบอกความแตกต่างของภาพได้ คุณสมบัติของลวดลายนั้นยังสามารถนำไปใช้ในการจำแนกภาพทางการแพทย์เช่น ภาพแมมโมแกรม และ ภาพอัลตราซาวด์ ลักษณะสำคัญหรือคุณลักษณะของลวดลายนั้นจะเป็นข้อมูลที่เกี่ยวกับการกระจายของรูปแบบโทนสี(Tone) ภายในภาพ ดังนั้นลักษณะการเปลี่ยนแปลงของโทนสีภายในภาพจึงเป็นข้อมูลสำคัญซึ่งนามาใช้ในการจำแนกภาพได้ ลักษณะสำคัญของลวดลายนั้นสามารถหาได้จากเมตริกซ์ของระดับสีเทาที่เกิดขึ้นร่วมกัน (Grey-level Co-occurrence Matrix) หรือ GLCM ฟังก์ชัน

ใน GLCM นั้นจะทำการคำนวณและเปรียบเทียบการเกิดขึ้นของระดับสีเทาในภาพหรือรูปแบบ (Pattern) ของระดับสีเทาระหว่างพิกเซลในภาพ โดยใช้ความน่าจะเป็นในการแสดงผลของความชัดเจนของลวดลาย (Contrast) การเกิดขึ้นร่วมกันของลวดลาย (Correlation) และการเป็นเนื้อเดียวกันของลวดลาย (Homogeneity) ซึ่งลักษณะสำคัญของลวดลายเหล่านี้จะถูกนำไปใช้ในกระบวนการจำแนกภาพ

2. ลักษณะสำคัญของฮิสโตแกรม (Histogram Based Feature)

รูปร่างลักษณะและคุณสมบัติของฮิสโตแกรม เป็นลักษะสำคัญอีกประเภทหนึ่งที่นิยมนำมาใช้เป็นลักษณะสำคัญในการจำแนกภาพ ซึ่งกราฟฮิสโตแกรมจะให้ข้อมูลสถิติของความเข้มสีที่เกิดขึ้นในภาพ และสามารถหาความน่าจะเป็นของความเข้มสีระดับสีเทาที่เกิดขึ้นในภาพ รูปร่างลักษณะและคุณสมบัติของฮิสโตแกรม เป็นลักษณะสำคัญอีกประเภทหนึ่งที่นิยมนามาใช้เป็นลักษณะสำคัญในการจำแนกภาพ ซึ่งกราฟฮิสโตแกรมจะให้ข้อมูลสถิติของความเข้มสีที่เกิดขึ้นในภาพ และสามารถหาความน่าจะเป็นของความเข้มสีระดับสีเทาที่เกิดขึ้นในภาพ

1) ค่าเฉลี่ย (Mean) คือ ค่าเฉลี่ยความเข้มสี

2) ค่าความแปรปรวน (Variance) คือ การเปลี่ยนแปลงความเข้มสีรอบ

3) ความเบ้ (Skewness) คือ ค่าที่แสดงถึงความสมมาตรของกราฟฮิสโตแกรม หากฮิสโตแกรมมีความสมมาตรแล้ว ความเบ้จะมีค่าเป็น 0

4) ความโด่ง (Kurtosis) คือ ค่าที่วัดจุดสูงสุดและต่ำสุดภายในกราฟในฮิสโตแกรมซึ่งมีความสัมพันธ์กับการกระจายข้อมูลแบบปกติ (Normal Distribution)

3. ลักษณะสำคัญของรูปร่าง (Shape Feature)

การหาลักษณะสำคัญของรูปร่างนั้นเป็นองค์ประกอบที่สำคัญอย่างหนึ่งในการจำแนกภาพ สมมติว่าต้องการที่จะจำแนกวัตถุสองชนิดซึ่งมีรูปร่างต่างกันจากภาพ ลักษณะสำคัญของรูปร่างของวัตถุจะเป็นตัวระบุความแตกต่างของวัตถุแต่ละชนิด ลักษณะสำคัญของรูปร่างที่สำคัญได้แก่ พื้นที่ (Area) เส้นผ่าศูนย์กลาง (Diameter) ส่วนนูนของรูปร่าง (Convex Area) โครงร่าง (Skeleton) เส้นรอบรูป (Perimeter) ระยะทางจากจุดศูนย์กลางไปยังเส้นขอบ (Centroid to Distance)

โดยทั่วไปวิธีการประมวลผลภาพเชิงตัวเลขที่ทำให้คอมพิวเตอร์สามารถรู้จักวัตถุในภาพได้นั้นแบ่งออกได้เป็นสองระดับด้วยกันคือ การประมวลผลภาพในระดับต่ำ (Low-Level Image Processing) และการประมวลผลภาพในระดับสูง (High-Level Image Processing)

การประมวลผลภาพในระดับต่ำ คือ การประมวลผลเชิงตัวเลขเกือบทั้งหมด เพื่อหาตัวแปรต่าง ๆ มาอธิบายข้อมูลของภาพ เพื่อต้องการนำตัวแปรต่าง ๆ เหล่านั้นไปใช้สำหรับการประมวลผลภาพระดับสูงต่อไปโดยทั่วไป แล้วการประมวลผลภาพระดับต่ำจะประกอบด้วย การประมวลผล

ภาพก่อน (Pre-Processing) เช่น การกำจัดสัญญาณรบกวน หรือการทำให้ภาพคมชัดและการหาขอบภาพ เป็นต้น

การประมวลผลระดับสูงเป็นการนำผลลัพธ์ หรือสัญลักษณ์ที่ได้จากการประมวลผลภาพระดับต่ำ มาตีความหรือประมวลผลเพื่อให้คอมพิวเตอร์สามารถรู้จักและเข้าใจภาพได้ สำหรับความแตกต่างของการประมวลผลภาพ ทั้ง 2 ประเภท นั้นคือ การประมวลผลภาพระดับต่ำ จะใช้ค่าพิกเซลของจุดภาพ ส่วนการประมวลภาพระดับสูงนั้นจะนำ ข้อมูลภาพที่ได้มา ประมวลผลและถูกแสดงในลักษณะของสัญลักษณ์ของสิ่งต่าง ๆ ที่ปรากฏในภาพ เช่น ขนาดของวัตถุ รูปร่าง และความสัมพันธ์กันระหว่างวัตถุในภาพ

1.2 โครงข่ายประสาทเทียม (Neural Network)

โครงข่ายประสาทเทียม (กชกร ณ นครพนม, 2558) เป็นเทคนิคที่ได้รับความนิยมเลือกใช้ในการพยากรณ์ เนื่องจากสามารถรองรับการทำงานที่มีความซับซ้อนได้ดี และให้ผลการทำนายได้อย่างถูกต้องแม่นยำ ส่วนมากการพยากรณ์โดยโครงข่ายประสาทเทียมมักถูกจัดทำเป็นชุดเครื่องมือสำเร็จรูปที่มีให้เลือกใช้ในซอฟต์แวร์ของการทำเหมืองข้อมูล ซึ่งผู้ใช้ส่วนใหญ่เลือกใช้มักไม่ทราบที่มาและหลักการทำงานภายใน โดยมักเลือกใช้เนื่องจากเป็นเทคนิคที่มีความฉลาดสามารถเรียนรู้เพื่อหาความสัมพันธ์ของข้อมูลได้ด้วยตนเอง จนสามารถนำแบบจำลองหรือโมเดลที่ได้มาใช้ในการพยากรณ์ผลลัพธ์ตามที่ผู้ใช้ต้องการ นอกจากกนี้โครงข่ายประสาทเทียมนิยมนำมาใช้ในรูปแบบของการทำงานเหมืองข้อมูลได้หลากหลาย เช่นการจำแนกประเภท การจัดกลุ่ม หรือการทำเหมืองข้อมูลด้วยอนุกรมเวลา เป็นต้น

รูปแบบการเรียนรู้ของโครงข่ายประสาทเทียมสำหรับการพยากรณ์ เป็นการเรียนรู้แบบมีผู้สอน (supervised learning) คือการเรียนรู้ที่ต้องทราบหรือสามารถระบุค่าเป้าหมาย (target) ที่ต้องการทำนายไว้ก่อนล่วงหน้า นิวรัลจะทำการสร้างความสัมพันธ์จากสัญญาณค่าเป้าหมาย (target signal) ที่สอดคล้องกับอินพุตการเรียนรู้แบบมีผู้สอนนี้มักนำมาปรับใช้ในการพยากรณ์ เช่น การทำนายอายุลูกค้าที่สนใจของสินเชื่อจากธนาคาร โดยอินพุตประกอบด้วย รายละเอียดต่าง ๆ ของลูกค้า เช่น เพศ อาชีพ รายได้ และค่าเป้าหมายคืออายุลูกค้าที่สนใจขอสินเชื่อกับธนาคาร สำหรับใช้เป็นข้อมูลให้โครงข่ายประสาทเทียมเรียนรู้ การเรียนรู้แบบมีผู้สอนมีหลายอัลกอริทึมด้วยกัน เช่นการเรียนรู้แบบย้อนกลับ (backpropagation) เป็นต้น

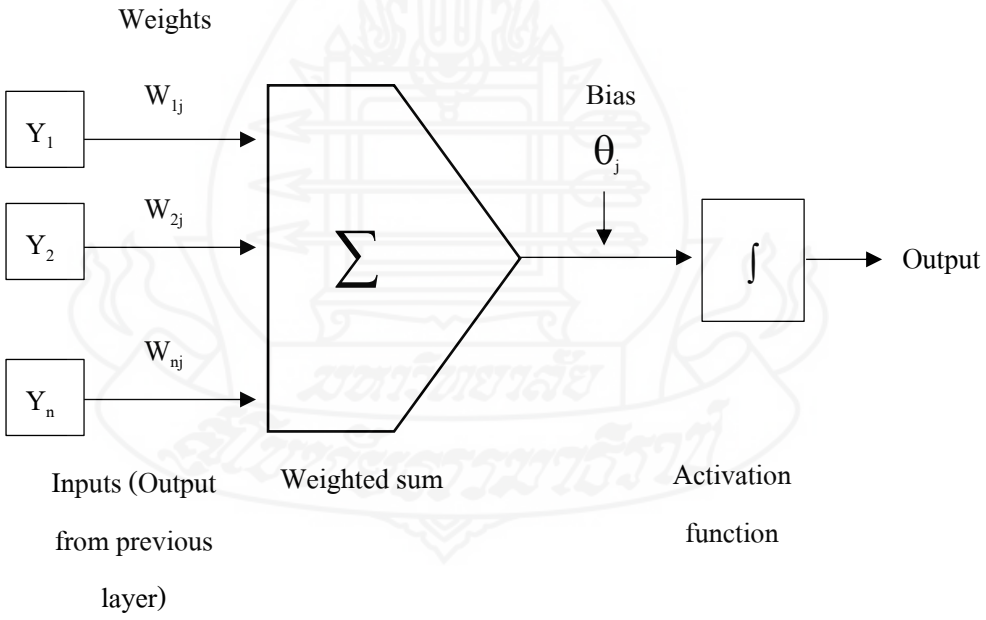
โครงสร้างและขั้นตอนวิธีการเรียนรู้และย้อนกลับ นิยมนำมาใช้เพื่อเรียนรู้ความสัมพันธ์ของข้อมูลจดจำรูปแบบเพื่อสกัดองค์ความรู้ใหม่โดยอาศัยของการปรับค่าถ่วงน้ำหนักจนกรั่งได้ผลลัพธ์ที่ใกล้เคียงกับค่าเป้าหมายที่ต้องการทำนายมากที่สุด

ประเภทของข้อมูลในโครงข่ายประสาทเทียม เนื่องจากการพยากรณ์ถือเป็นการเรียนรู้แบบมีผู้สอนหรือมีเป้าหมาย ชุดข้อมูลที่ใช้จำเป็นต้องทราบค่าเป้าหมาย โดยแบ่งชนิดของข้อมูลเป็น 2 ประเภทด้วยกัน

1. อินพุต (input) คือข้อมูลหรือแอตทริบิวต์ที่ถูกเลือกเพื่อใช้เป็นอินพุตในการเรียนรู้ของโครงข่ายประสาทเทียม ซึ่งโดยทั่วไปผู้ใช้มักเลือกข้อมูลที่คิดว่าสำคัญและมีผลต่อค่าเป้าหมายที่ต้องกาทำนายและมักไม่นำค่าแอตทริบิวต์ที่คาดว่าไม่มีส่วนเกี่ยวข้องกับค่าเป้าหมายมาเป็นอินพุต

2. เอาท์พุต (output) คือค่าหรือข้อมูลที่โครงข่ายประสาทเทียมสร้างความสัมพันธ์เพื่อทำนายผลลัพธ์ ในการเรียนเรียนรู้ของโครงข่ายจำนำเอาท์พุตที่ได้มาเปรียบเทียบกับค่าเป้าหมาย (target value) หากเอาท์พุตที่ทำนายได้ยังมีค่าไม่ใกล้เคียงกับค่าเป้าหมายที่ยอมรับได้ โครงข่ายประสาทเทียมจะทำการเรียนรู้ซ้ำจนกระทั่งเอาท์พุตมีค่าใกล้กับค่าเป้าหมาย

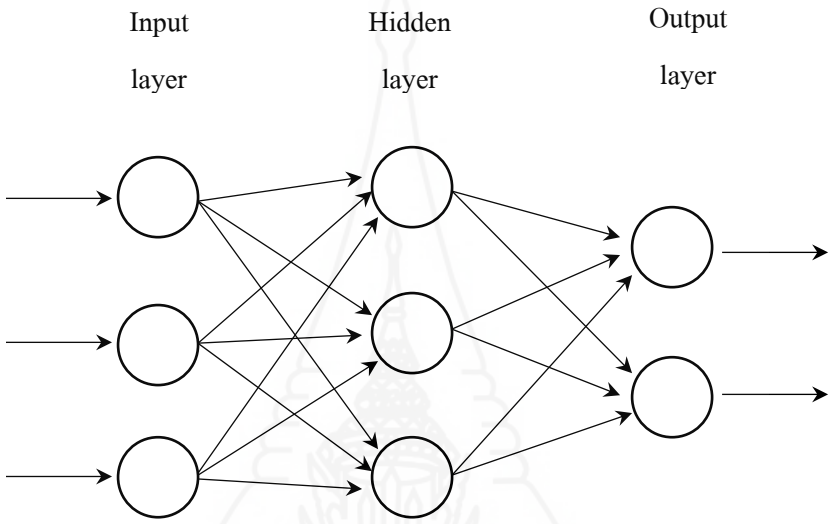
หนึ่งหน่วยประสาท ในการทำงานของโครงข่ายประสาทเทียมมักประกอบไปด้วยนิวรัลโหนดหลายโหนดด้วยกัน โดยแต่ละนิวรัลโหนดมีการทำงานถือเป็นหนึ่งหน่วยประสาท (neural) ซึ่งมีรูปแบบการทำงานที่เหมือนกันดังแสดงในภาพที่ 2.2



ภาพที่ 2.2 แสดงหนึ่งหน่วยประสาท (neural)

ที่มา: กชกร ณ นครพนม (2558)

โดยในหนึ่งหน่วยที่เชื่อมติดกันจะทำการรับข้อมูลจากนิวรัลโหนดก่อนหน้า หรือถือได้ว่าเป็นเอาท์พุตจากลำดับชั้น (layer) ก่อนหน้า พร้อมคำนวณผลรวมของค่าถ่วงน้ำหนัก (weight sum) จากนำค่าถ่วงน้ำหนักคูณกับค่าอินพุตที่เข้ามายังนิวรัลโหนด เมื่อได้ผลรวมดังกล่าวแล้ว จึงมีการบวกด้วยค่า ไบแอส (bias) ซึ่งใช้เป็นค่าที่ช่วยในการขยับช่วงของข้อมูลและผ่านฟังก์ชันกระตุ้น (activation function) ก่อนแสดงค่าเอาท์พุตต่อไป



ภาพที่ 2.3 ดัดแปลงมาจากสถาปัตยกรรมของโครงข่ายประสาทเทียม

ที่มา: กชกร ณ นครพนม (2558)

จากภาพที่ 2.3 แสดงชั้นของโครงข่ายประสาทเทียมโดยทั่วไป จะประกอบด้วยชั้นทั้งหมด 3 ชั้น คือ input layer เป็นชั้นที่รับข้อมูลเข้า hidden layer ชั้นนี้เป็นชั้นที่ใช้สำหรับเรียนรู้ของโครงข่าย และ output layer เป็นชั้นที่รวมผลลัพธ์ที่ได้จาก hidden layer และแปลงข้อมูลเพื่อส่งออก

1) ชั้นอินพุต (input layer) เป็นชั้นที่ติดต่อกับข้อมูลอินพุตโดยจำนวนตัวรับข้อมูลหรืออินพุตโหนด (input node) จะเท่ากับจำนวนแอตทริบิวต์ที่เลือกมาเป็นอินพุตเช่นการทำนายรายได้ของลูกค้าโดยอาศัยข้อมูลอินพุต 3 ค่า คือระดับการศึกษา ตำแหน่ง และอายุ ใช้จำนวนอินพุตโหนดเท่ากับ 3 โหนด เป็นต้น

2) ชั้นซ่อน (hidden layer) หรือบางครั้งเรียกว่านิวโรด (neurodes) เป็นชั้นที่อยู่กึ่งหลางเชื่อมต่อระหว่างชั้นอินพุตและชั้นเอาท์พุต ทำหน้าที่หลักในการสร้างโมเดลจากชั้นอินพุตเพื่อทำการสร้างและปรับค่าถ่วงน้ำหนัก โดยเรียนรู้ความสัมพันธ์ของข้อมูลดังกล่าวและทำการส่งข้อมูล

ต่อเพื่อเป็นอินพุตต่อไป ชั้นซ่อนนี้อาจจะประประกอบด้วยชั้นย่อยมากกว่าหนึ่งชั้นด้วยกัน ทั้งนี้ผู้ใช้มักกำหนดจำนวนชั้นซ่อนจากความซับซ้อนของข้อมูลหลัก

3) ชั้นเอาท์พุตเป็นชั้นสุดท้ายที่รับข้อมูลจากชั้นซ่อนเพื่อใช้คำนวณค่าเอาท์พุต

องค์ประกอบของ Multilayer Feed Forward Neural Network นอกจากอินพุตและค่าเป้าหมายซึ่งเป็นข้อมูลสำคัญพื้นฐานของโครงข่ายประสาทเทียมแบบ Multilayer Feed Forward Neural Network แล้ว ภายในสถาปัตยกรรมของโครงข่ายมีองค์ประกอบสำคัญดังนี้

1) ค่าถ่วงน้ำหนัก (weight) ใช้ในการปรับค่าตามการเรียนรู้จากอินพุตที่นำเข้าเพื่อสร้างความสัมพันธ์ระหว่างอินพุตและค่าเป้าหมายที่ต้องการ โดยทั่วไปค่าเริ่มต้นของค่าถ่วงน้ำหนักถูกกำหนดโดยการสุ่ม มักใช้ค่าน้อย ๆ เช่นมีค่าอยู่ระหว่าง -0.5 ถึง 0.5 เป็นต้นสำหรับกรณีทั่วไป โปรแกรมสำเร็จรูปทางด้านเหมืองข้อมูลจะทำการสุ่มให้โดยอัตโนมัติ

2) ไบแอส (bias) เป็นค่าที่เชื่อมติดกับนิวรัลโหนดในชั้นซ่อนและชั้นเอาท์พุต ใช้ในการขยับช่วงของข้อมูลให้ขยับช่วงเพื่อเข้าใกล้ค่าเป้าหมายมากขึ้นมีการกำหนดค่าเริ่มต้นเช่นเดียวกับค่าถ่วงน้ำหนักคือเป็นการสุ่มค่าน้อย ๆ

3) ฟังก์ชันกระตุ้น (activation function) เป็นฟังก์ชันที่ใช้ในการควบคุมค่าและช่วงของเอาท์พุตจากโครงข่าย ซึ่งผู้ใช้สามารถกำหนดฟังก์ชันที่เลือกมาเป็นฟังก์ชันกระต้นได้ เช่น โลจิสติกซิกมอยด์ (logistic sigmoid) ซึ่งให้ค่าเอาท์พุตอยู่ระหว่าง 0 ถึง 1 หรือไฮเพอร์บอลิกแทนเจนต์ (hyperbolic tangent) หรือเพียวลิเนียร์ (pure linear) เป็นต้น ทั้งนี้อยู่กับค่าเอาท์พุตที่ผู้ใช้ต้องการ โดยทั่วไปมักใช้โลจิสติกสซิกมอยด์เป็นฟังก์ชันกระตุ้นโดยโปรแกรมสำเร็จรูปจะมีการกำหนดค่าปริยาย (default value) ไว้ให้ ทั้งนี้ผู้ใช้สามารถเลือกใช้ฟังก์ชันอื่นแทนได้

4) อัตราการเรียนรู้ (learning rate) เป็นค่าที่ใช้ในการกำหนดอัตราการปรับค่าถ่วงน้ำหนักของโครงข่ายประสาทเทียมเพื่อให้เข้าใกล้ค่าเป้าหมาย โดยทั่วไปมักกำหนดอัตราการเรียนรู้ (learning rate) ให้มีค่าอยู่ระหว่าง 0 ถึง 1เช่นหากต้องการให้มีการปรับค่าถ่วงน้ำหนักในอัตราที่เร็ว โดยกำหนดให้อัตราการเรียนรู้มีค่า 0.9 เป็นต้น

5) ตัววัดระดับความถูกต้องหรือเทรดโฮลด์ (threshold) หรือตัววัดความคลาดเคลื่อน (error) มักถูกกำหนดค่าเพื่อให้โครงข่ายทราบค่าความผิดพลาดหรือค่าความคลาดเคลื่อนที่สามารถยอมรับได้เพื่อยุติการเรียนรู้ (training)

อัลกอริทึมของการเรียนรู้แบบย้อนกลับ (backpropagation algorithm) เป็นอัลกอริทึมที่มีจุดประสงค์หลักเพื่อหาค่าถ่วงน้ำหนักที่เหมาะสมกับข้อมูลมากที่สุด สำหรับใช้เป็นตัวแทนของโมเดลจากชุดข้อมูลนำเข้าในการพยากรณ์ข้อมูลที่เป็นชุดทดสอบที่ไม่เคยเห็นมาก่อน

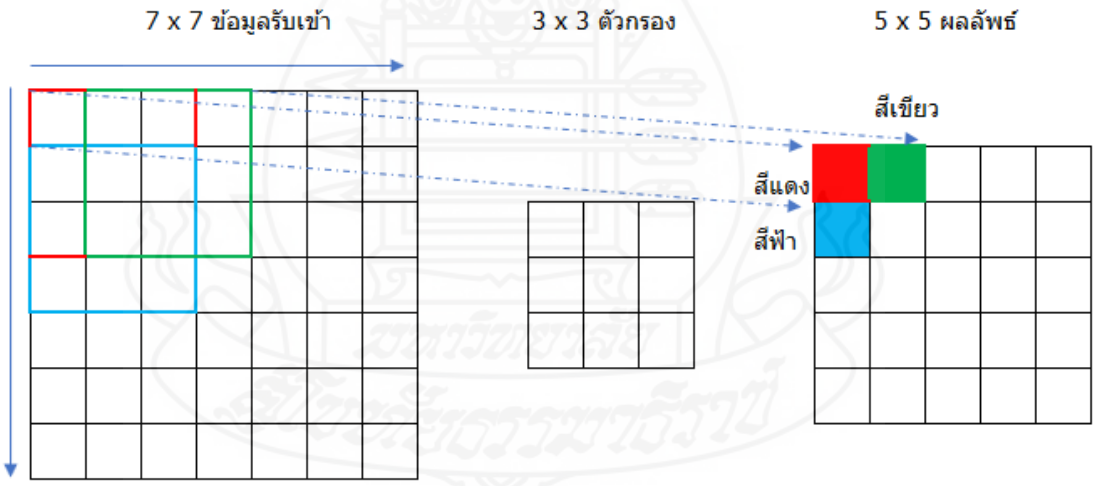
หลักการทำงานสำคัญของอัลกอริทึมของการเรียนรู้แบบย้อนกลับคือการทำการปรับค่าถ่วงน้ำหนักซ้ำจนกระทั่งค่าที่ทำนายได้นั้นเข้าใกล้เคียงกับค่าเป้าหมายภายใต้ค่าความคลาดเคลื่อนที่ยอมรับได้เพื่อให้ยุติการเรียนรู้ แต่หากในแต่ละรอบการเรียนรู้เพื่อปรับค่าถ่วงน้ำหนักยังไม่เข้าใกล้ค่าเป้าหมายเพียงพอ จะทำการปรับค่าถ่วงน้ำหนักและค่าไบแอสย้อนกลับมาที่โครงข่าย โดยทำการปรับค่าถ่วงน้ำหนักย้อนกลับจากด้านเอาท์พุตมาจนกระทั่งอินพุต ก็คือปรับจากชั้นเอาท์พุตมาจนกระทั่งถึงชั้นซ่อน

1.3 โครงข่ายประสาทเทียมแบบคอนโวลูชัน (Convolutional Neural Network)

โครงข่ายประสาทเทียมแบบคอนโวลูชัน (ธนภัทร์ คุ้มสุภา, 2559) เป็นโครงข่ายประสาทเทียมเชิงลึกรูปแบบหนึ่ง มีจุดเริ่มต้นมาจากการงานวิจัยทางด้านการรู้จำภาพตัวอักษร โดยมักจะใช้ข้อมูลรับเข้าเป็นเมทริกซ์จากการแปลงมาจากรูปภาพ ซึ่งประกอบด้วยชั้นต่าง ๆ ดังต่อไปนี้

1.3.1 ชั้นคอนโวลูชัน (Convolutional Layer)

เป็นชั้นที่ทำการหาฟีเจอร์จากกลุ่มของข้อมูลรับเข้าที่อยู่ใกล้ๆ กันโดยใช้วิธีดอทเมทริกซ์กับตัวกรอง (filter) โดยน้ำหนักของตัวกรองนั้น จะเป็นน้ำหนักที่มีการใช้ร่วมกันทุก ๆ การทำคอนโวลูชันของข้อมูลรับเข้า ดังภาพที่ 2.4



ภาพที่ 2.4 ตัวอย่างการทำคอนโวลูชัน โดยมีขนาดของข้อมูลรับเข้าขนาด 7 x 7 และเมทริกซ์ตัวกรองขนาด 3 x 3

จากภาพที่ 2.4 เป็นทำคอนโวลูชันโดยมีข้อมูลรับเข้า หรือรูปภาพขนาด 7 x 7 และผ่านตัวกรองขนาด 3 x 3 การทำงานเริ่มจากนำตัวกรองมาทาบกับข้อมูลรับเข้าโดยเริ่มจากซ้ายมือบนสุด

(ช่องสีแดง) จากนั้นนำค่าของข้อมูลรับเข้าคูณกับค่าของตัวกรองที่มีช่องตรงกัน จะได้ผลคูณ 9 ค่า นำผลคูณทั้ง 9 ค่ามารวมกันจะเป็นผลลัพธ์ตรงช่องสีแดงทางด้านผลลัพธ์ จากนั้นเลื่อนมาอีก 1 ช่อง (ช่องสีเขียว) หรือขึ้นอยู่กับขนาดของการก้าวข้าม ทำในลักษณะเดียวกันเหมือนกับการทำช่องสีแดง จะได้ผลลัพธ์ออกเป็นช่องสีเขียว การเลื่อนจะทำการเลื่อนจากซ้ายบนสุดไปยังขวาสุด ถ้าสิ้นสุดทางด้านขวาแล้วจะเป็นการเลื่อนลงมาอีก 1 ช่องโดยเริ่มจากซ้ายไปขวาเช่นกัน จากนั้นเลื่อนไปเรื่อย ๆ จนกว่าจะครบทุกช่องในชั้นคอนโวลูชัน มีองค์ประกอบที่ต้องคำนึงถึงดังต่อไปนี้

1) ขนาดของตัวกรอง (Filter Size)

คือความกว้างและความสูงของตัวกรองที่นำมาใช้ในการทำคอนโวลูชัน

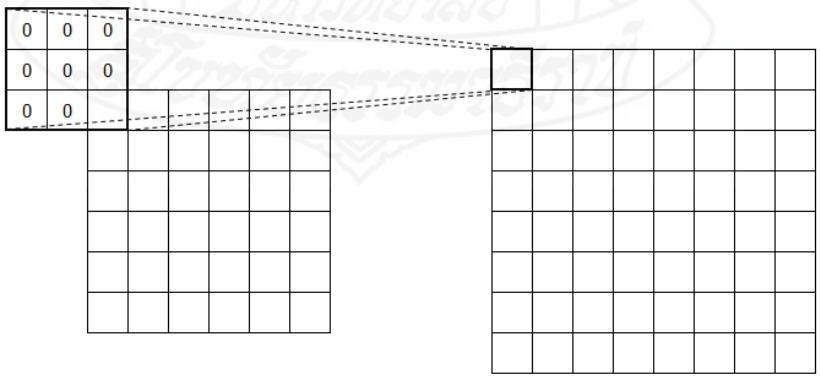
2) ชนิดของการทำคอนโวลูชัน (Convolution type)

(1) คอนโวลูชันแบบแคบ (Narrow Convolution)

การทำคอนโวลูชันโดยทั่วไปมักจะเป็นการทำคอนโวลูชันแบบแคบ กล่าวคือการทำคอนโวลูชัน ตัวกรองที่นำไปทำการดอทเมทริกซ์นั้นจะไม่มีการกระทำเลยขอบของเมทริกซ์รับเข้า ส่งผลให้ผลลัพธ์ของการทำคอนโวลูชันที่มีข้อมูลรับเข้าขนาด N x N กับตัวกรองขนาด m x m จะได้เมทริกซ์ขนาด (N-m+1) x (N-m+1)

(2) คอนโวลูชันแบบกว้าง (Wide Convolution)

เป็นการทำคอนโวลูชันที่มีการกระทำเลยขอบของเมทริกซ์รับเข้าออกไป โดยพื้นที่ที่เกินออกไปนั้น จะมีการแทนค่าของข้อมูลช่องนั้น ๆ ด้วย 0 เรียกว่าการเสริมเติม (padding) ผลลัพธ์ของการทำคอนโวลูชันแบบกว้างที่มีข้อมูลรับเข้าขนาด N x N กับตัวกรองขนาด m x m จะได้เมทริกซ์ขนาด (N + m - 1) x (N + m - 1) ทั้งนี้การทำคอนโวลูชันแบบกว้างและการเสริมเติม

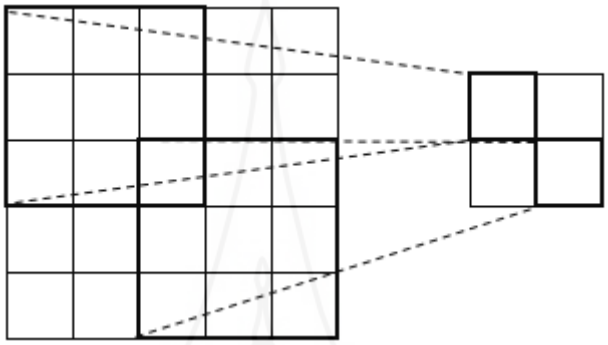


ภาพที่ 2.5 การทำคอนโวลูชันแบบกว้างและการเสริมเติม

ที่มา: ธนภัทร์ คุ้มสุภา (2559)

3) ขนาดของการก้าวข้าม (Stride Size)

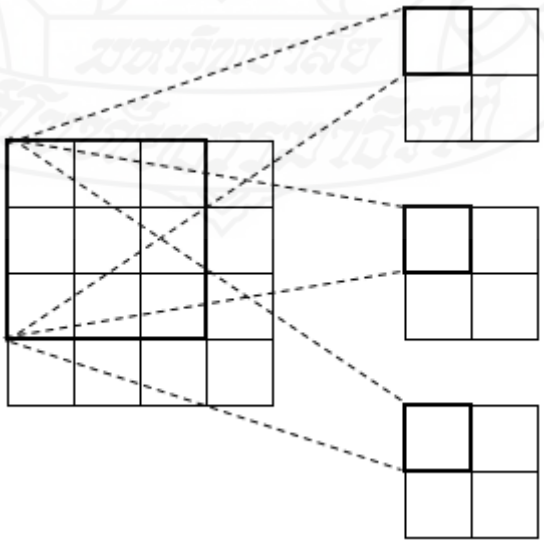
ขนาดของการก้าวข้ามคือจำนวนช่องของข้อมูลรับเข้า ที่จะทำการเลื่อนไปเมื่อทำการหาผลลัพธ์ของคอนโวลูชันในแต่ละช่อง โดยทั่วไปมักจะใช้ขนาดของการก้าวข้ามเป็น 1 ภาพที่ 2.4 แสดงลักษณะของการทำคอนโวลูชันที่มีขนาดของการก้าวข้ามเป็น 2



ภาพที่ 2.6 การทำคอนโวลูชันโดยมีข้อมูลรับเข้าขนาด 5 x 5 ตัวกรอง 3 x 3 และมีขนาดของการก้าวข้ามเป็น 2

4) จำนวนตัวกรอง (Number of Filters)

ในการทำแต่ละชั้นคอนโวลูชันนั้น สามารถมีตัวกรองได้มากกว่าหนึ่ง โดยน้ำหนักของตัวกรองแต่ละตัวจะใช้แยกกัน โดยจำนวนตัวกรองในชั้นคอนโวลูชันใด ๆ จะเป็นการกำหนดจำนวนช่องสัญญาณ (Channel) ของข้อมูลรับเข้าในชั้นถัดไป ภาพที่ 2.5 แสดงตัวอย่างการทำคอนโวลูชันโดยมีจำนวนตัวกรองเป็น 3



ภาพที่ 2.7 การทำคอนโวลูชันโดยมีจำนวนตัวกรองเท่ากับ 3

5) จำนวนช่องสัญญาณ (Channel)

จำนวนช่องสัญญาณหรือเรียกได้อีกอย่างหนึ่งว่าความลึกของข้อมูลรับเข้า อาจจะมีค่ามากกว่าหนึ่งได้ เช่นในงานวิจัยทางด้านการภาพ มีการใช้ช่องสัญญาณทั้งหมด 3 ช่องสัญญาณแทนค่าของแม่สี หรือสามารถเกิดจากจำนวนของตัวกรองในชั้นคอนโวลูชันก่อนหน้า กำหนดให้จำนวนช่องสัญญาณมีค่าเป็น k จะคำนวณผลลัพธ์ของชั้นคอนโวลูชัน

6) การแพร่กระจายย้อนกลับและการเรียนรู้ (backpropagation and Training)

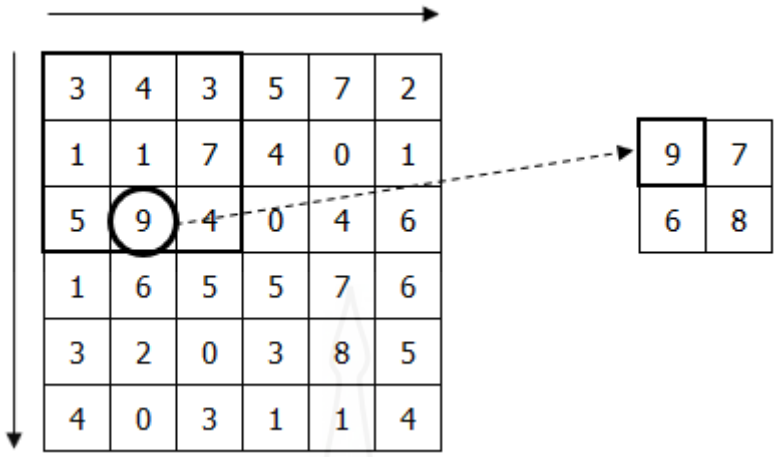
หลักการทำงานสำคัญของของการเรียนรู้แบบย้อนกลับคือการสุ่มค่าน้ำหนักให้กับส่วนนำเข้าของแต่ละเซลล์ ซึ่งค่าน้ำหนักที่สุ่มได้อาจมีค่าอยู่ช่วง -1.0 ถึง 1.0 หรือ -0.5 ถึง 0.5 สามารถเปลี่ยนแปลงได้หลังจากการคำนวณหรือการฝึกฝนแล้ว กำหนดค่าไบแอสซึ่งเป็นค่าเริ่มต้นในขั้นตอนแรก อาจมีค่าเท่ากับ 1 หรือสุ่มขึ้นมา นำข้อมูลนำเข้าไปในโครงข่ายผ่านทางชั้นอินพุต จากนั้นคำนวณค่าเอาท์พุตในแต่ละชั้น คำนวณหาค่าความคลาดเคลื่อนในแต่ละชั้น โดยคำนวณชั้นเอาท์พุตก่อน จากนั้นจึงคำนวณที่ชั้นซ่อน ซึ่งค่าความคลาดเคลื่อนสามารถคำนวณได้จากการเปรียบเทียบผลลัพธ์ที่คำนวณได้กับค่าเป้าหมายที่ตั้งไว้ ทำการปรับค่าน้ำหนักและค่าไบแอสจากค่าความคลาดเคลื่อนที่คำนวณได้ การทำการปรับค่าถ่วงน้ำหนักซ้ำจนกว่าจะป้อนข้อมูลนำเข้าของการฝึกฝนหมด หรือกระทั่งค่าผลลัพธ์นั้นเข้าใกล้เคียงกับค่าเป้าหมายภายใต้ค่าความคลาดเคลื่อนที่ยอมรับได้เพื่อให้ยุติการเรียนรู้ กล่าวคือหากในแต่ละรอบการเรียนรู้เพื่อปรับค่าถ่วงน้ำหนักยังไม่เข้าใกล้ค่าเป้าหมายเพียงพอ จะทำการปรับค่าถ่วงน้ำหนักและค่าไบแอสย้อนกลับมาที่โครงข่าย โดยทำการปรับค่าถ่วงน้ำหนักย้อนกลับจากด้านเอาท์พุตมาจนกระทั่งอินพุต ก็คือปรับจากชั้นเอาท์พุตมาจนกระทั่งถึงชั้นซ่อน

1.3.2 ชั้นการรวม (Pooling Layer)

ชั้นการรวมเป็นชั้นที่ทำหน้าที่ลดขนาดของข้อมูลลง เพื่อให้เหลือแค่เพียงข้อมูลที่สำคัญ ๆ เท่านั้น โดยทั่วไป มักจะทำการเลือกข้อมูลที่มีค่ามากที่สุดจากช่วงของเมทริกซ์เพื่อสร้างเป็นเมทริกซ์ขนาดเล็กลง

1) ชั้นการรวมโดยใช้ค่ามากสุด (Max Pooling)

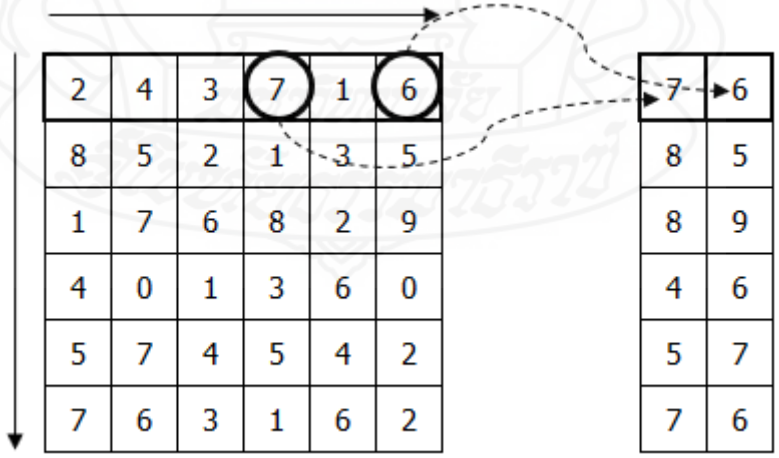
ชั้นการรวมโดยใช้ค่ามากสุดจะทำการเลือกเฉพาะค่ามากสุดจากกลุ่มของข้อมูลที่สนใจและนำไปใช้งานต่อไปในชั้นถัดไป จากภาพ 7 ที่ เป็นการรวมโดยค่ามากสุดบนเมทริกซ์ขนาด 6 x 6 โดยกลุ่มที่สนใจจะมีขนาด 3 x 3 ซึ่งขอบเขตของกลุ่มที่สนใจจะมีการเลื่อนไปจนครอบคลุมเมทริกซ์ต้นฉบับทั้งหมด



ภาพที่ 2.8 ชั้นการรวมโดยใช้ค่ามากสุดใน 2 มิติ

2) ชั้นการรวมโดยใช้เคค่ามากสุด (K-max pooling)

ชั้นการรวมโดยใช้เคมากสุดจะทำการเลือก k ฟีเจอร์ที่มีค่ามากที่สุดจากข้อมูลทั้งหมดที่มีมักจะกระทำบนเวกเตอร์และนำไปใช้ในเน็ตเวิร์กที่กระทำกับข้อมูลที่เป็นข้อความ จากภาพที่ 2.7 เป็นการทำการรวมโดยใช้เคค่ามากสุดซึ่งกำหนดให้ k = 2 ทั้งนี้ในการทำการรวมแต่ละครั้ง จะสนใจในแนวเวกเตอร์ในแนวแกน X และการเลื่อนขอบเขตที่สนใจจะทำการเลื่อนเฉพาะแนวแกน Y โดยจะสังเกตได้ว่าผลลัพธ์จากการทำการรวมโดยค่าเคมากสุดจะเรียงลำดับแบบเดิมเช่นเดียวกับเมทริกซ์ในชั้นก่อนหน้าในแนวแกน X



ภาพที่ 2.9 ชั้นการรวมโดยใช้เคค่ามากสุดใน 1 มิติ โดยกำหนดให้ k = 2

3) ชั้นการรวมโดยใช้เคค่ามากสุดแบบพลวัต (Dynamic k-max pooling)

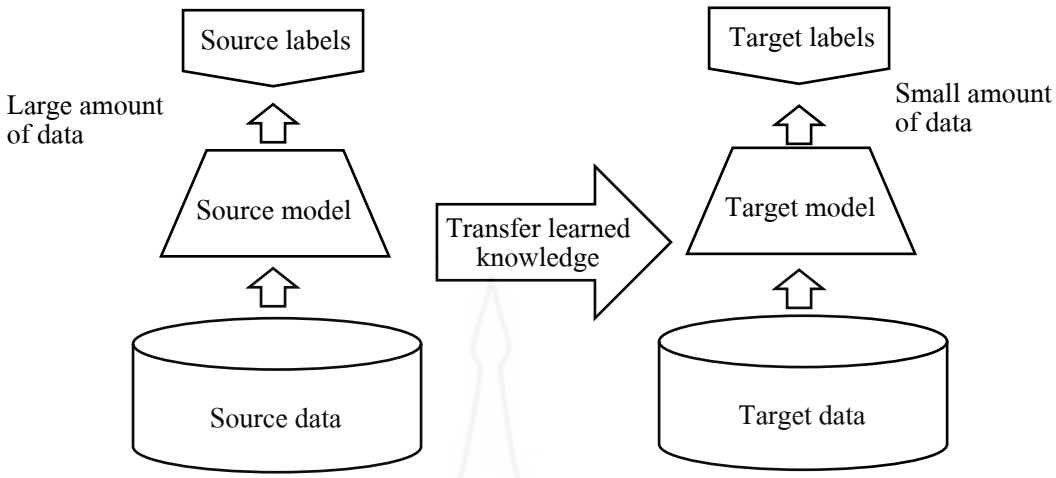
เป็นชั้นการรวมที่พัฒนาต่อจากชั้นการรวมโดยใช้เคค่ามากสุด โดยจะแตกต่างอยู่ที่จะไม่มีกำหนดค่า k ไว้ก่อน แต่จะทำการคำนวณค่า k ใหม่ให้เหมาะสมกับทุกขนาดของข้อมูลในชั้นก่อนหน้า

1.3.3 ชั้นการเชื่อมโยงเต็มรูปแบบ (Fully Connected Layer)

หลังจากการประกอบกันของชั้นคอนโวลูชันและชั้นการรวมจำนวนหนึ่งแล้ว ในชั้นสุดท้ายของโครงข่ายประสาทเทียมแบบคอนโวลูชัน จะเป็นการเชื่อมโยงเต็มรูปแบบ นั่นคือ ในชั้นนี้ประกอบด้วยชั้นย่อย ๆ ที่มีเพอร์เซ็ปตรอนอยู่จำนวนหนึ่ง โดยที่เพอร์เซ็ปตรอนแต่ละตัว จะมีเส้นเชื่อมกับเพอร์เซ็ปตรอนทุกตัวก่อนหน้า และเพอร์เซ็ปตรอนทุกตัวในชั้นถัดไป ทั้งนี้การคำนวณการป้อนไปข้างหน้าและการแพร่กระจายย้อนกลับสามารถทำได้ด้วยวิธีปกติ

1.4 การเรียนรู้เชิงถ่ายโอนข้อมูล (Transfer Learning)

การเรียนรู้เชิงถ่ายโอนข้อมูล (Lisa Torrey and Jude Shavlik, n.d.) คือการเรียนรู้ของเครื่องที่มีข้อมูลหรือความรู้ที่ได้จากการฝึกฝนข้อมูลเดิมมาใช้กับการฝึกฝนข้อมูลใหม่หรือความรู้จากงานเดิมอย่างน้อยหนึ่งงานที่เกี่ยวข้องกัน โดยธรรมชาติการเรียนรู้ของมนุษย์จะมีวิธีการถ่ายทอดโดยอาศัยประสบการณ์ในชีวิตประจำวัน เช่น เมื่อต้องพบกับงานใหม่ๆ หรือไปสมัครงานในตำแหน่ง หรือสาขางานที่แตกต่างแต่อาจเกี่ยวข้องกัน จำเป็นจะใช้ความรู้และประสบการณ์ก่อนหน้านี้มาประยุกต์ใช้ ในการทำงานงานใหม่นั้นและใช้เวลาปรับตัวไม่นาน ก็สามารถที่จะทำงานใหม่นั้นได้ดีและแก้ปัญหาที่เกี่ยวข้องกับงานนั้นได้ไม่ต่างกับงานเดิมที่ทำอยู่แต่ก่อน หรือเมื่อต้องการให้เด็กจดจำตัวอักษรใหม่ จำเป็นต้องเขียนตัวอักษรด้วยปากกา ดินสอ ที่มีสี ความหนาแตกต่างกันออกไปและเด็กจะต้องพิจารณาคุณลักษณะที่สำคัญของตัวอักษรเหล่านั้น อาจจะต้องใช้เวลานานและจะต้องมีข้อมูลตัวอย่างจำนวนมากเพื่อให้เด็กได้เรียนรู้ตัวอักษรตัวใหม่ แต่สำหรับเด็กที่มีการเรียนรู้มาก่อน เพียงแค่จดจำเฉพาะเส้นหรือส่วนโค้งออกจากสิ่งที่ไม่เกี่ยวข้องเช่นสี ขนาดของเส้น ของตัวอักษร และใช้ตัวอย่างเพียงแค่เล็กน้อยก็สามารถที่จดจำตัวอักษรใหม่ ๆ ได้ด้วยแนวคิดนี้



ภาพที่ 2.10 การเรียนรู้เชิงถ่ายโอนข้อมูล

ที่มา: https://www.slideshare.net/xavigiro/transfer-learning-and-domain-adaptation-dlai-d5l2-2017-upc-deep-learning-for-artificial-intelligence

จากภาพที่ 2.10 การเรียนรู้เชิงถ่ายโอนข้อมูลมีการที่จะพยายามถ่ายทอดความรู้จากงานหนึ่งทำการปรับปรุงพัฒนาเพื่อไปใช้กับอีกงานหนึ่ง หรือหลายงานที่เกี่ยวข้องกันได้ โดยเทคนิคนี้เริ่มจากการพัฒนาตัวแบบที่ได้จากแหล่งข้อมูลขนาดใหญ่ (Source data) ฝึกฝนเพื่อให้ได้ตัวแบบในการรู้จำ (Source model) เพื่อใช้จำแนกหรือระบุค่าเป้าหมาย (Source label) ซึ่งจากการฝึกฝนด้วยปริมาณข้อมูลที่มากทำให้ค่าความถูกต้องและแม่นยำที่สูงในการพยากรณ์ จากการค่าความถูกต้องและแม่นยำที่สูงนี้เนื่องมากจากความรู้ที่ได้จากการเรียนรู้ในกระบวนการฝึกฝนสามารถนำความรู้ดังกล่าวมาใช้กับงานที่มีข้อมูลขนาดเล็ก (Target data) อาจเนื่องมาจากไม่มีข้อมูลมากพอที่จะใช้ในการฝึกฝนเพื่อให้ตัวแบบที่ต้องการได้เกิดจากเรียนรู้ที่มากพอที่จะทำให้ค่าความถูกต้องและความแม่นยำที่สูงในการพยากรณ์ การใช้ความรู้ที่ได้จากการฝึกฝนมาแล้วมาทำการฝึกฝนตัวแบบที่ต้องการ (Target model) เพื่อใช้ระบุค่าเป้าหมายที่ต้องการ (Target label) ซึ่งเทคนิคนี้มีความพยายามที่จะทำให้เครื่องมีการเรียนรู้ที่มีประสิทธิภาพได้ดีเท่ากับการเรียนรู้ของมนุษย์เป้าหมายของการเรียนรู้เชิงการถ่ายโอนคือการปรับปรุงการเรียนรู้ของงาน เป้าหมาย (Target task)โดยใช้ความรู้จากแหล่งความรู้ที่ได้จากการฝึกฝนมาแล้ว (Source Task Knowledge)

ในการเรียนรู้ของเครื่องจักรแบบดั้งเดิมกระทำโดยการเริ่มจากการจัดเตรียมข้อมูลนำเข้า จากนั้นออกแบบและสร้างตัวแบบ ทำการฝึกฝนข้อมูลโดยการสุ่มค่าต่าง ๆ ภายใน โดยค่าที่สุ่มจะสุ่มค่าตั้งแต่ค่าน้อย ๆ และขยับค่าในการสุ่มไปเรื่อย ๆ ทีละนิด เพื่อปรับค่าให้ได้ผลลัพธ์

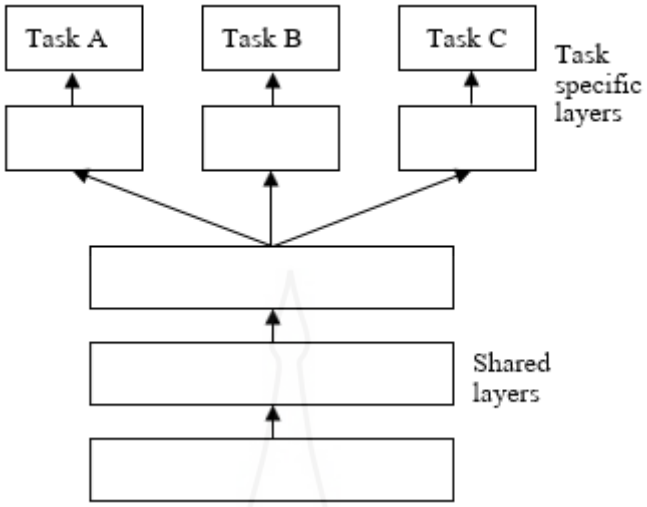
เท่ากับหรือใกล้เคียงกับค่าเป้าหมาย แต่การเรียนรู้เชิงถ่ายโอนข้อมูลสามารถกระทำได้โดยเริ่มต้นจากการนำค่าที่ได้จากการสุ่มของตัวแบบที่ได้จากการฝึกฝนไว้แล้ว มาเป็นการตั้งต้นของตัวแบบตัวที่สอง เพื่อใช้เป็นตัวแบบกับงานอื่น แทนที่จะทำการฝึกฝนโดยการสุ่มค่าใหม่ตั้งแต่ต้น และมีอยู่ 3 เหตุผลที่เชื่อว่าการเรียนรู้เชิงถ่ายโอนข้อมูลเป็นทักษะที่สำคัญในฐานะที่เป็นนักวิทยาศาสตร์ด้านข้อมูล

1) การเรียนรู้เชิงถ่ายโอนข้อมูลเป็นสิ่งสำคัญอย่างหนึ่งของการเรียนรู้ของเครื่อง โดยปกติการจะแก้ปัญหาใดปัญหาหนึ่งจำเป็นต้องใช้ข้อมูล ฝึกฝนตัวแบบเพื่อนำตัวแบบไปใช้ได้กับงานใดงานหนึ่ง เมื่อมีปัญหาที่ไม่เคยเจอจำเป็นจะต้องสร้างตัวแบบและทำการฝึกฝนตัวแบบใหม่ไปเรื่อย ๆ ดังนั้นความสามารถในการใช้ความรู้ที่ได้จากการฝึกฝนข้อมูลปริมาณมาก ๆ และนำมาใช้กับปัญหาใหม่ ภายใต้สภาพแวดล้อมใหม่ สามารถแก้ปัญหาใหม่ได้โดยไม่ต้องทำกระบวนการตั้งแต่ต้นใหม่ทั้งหมด เพียงแค่ใช้ความรู้เดิมมาเป็นตัวตั้งต้น

2) การเรียนรู้เชิงถ่ายโอนข้อมูลเป็นหัวใจสำคัญในการพัฒนาเทคนิคการเรียนรู้เชิงลึก อีกทั้งยังมีงานวิจัยหลากหลายงานที่เกี่ยวข้องกับเรื่องนี้ ในความเป็นจริงการใช้เทคนิคการเรียนรู้เชิงลึกต้องใช้ข้อมูลที่ระบุป้ายจำนวนนับล้านเพื่อใช้ในการฝึกฝนตัวแบบ การใช้ข้อมูลจำนวนมากนี้ก็เพื่อปรับแต่งพารามิเตอร์นับล้าน ๆ ในโครงข่ายประสาท โดยเฉพาะอย่างยิ่งกรณีของการเรียนรู้แบบมีผู้สอน นั่นหมายความว่าจะต้องใช้ข้อมูลที่มีป้ายหรือที่สามารถระบุชนิดข้อมูลได้จำนวนมากมาทำการฝึกฝนตัวแบบ การถ่ายโอนข้อมูลเป็นวิธีการหนึ่งใช้ในการลดขนาดความต้องการของชุดข้อมูลเพื่อให้โครงข่ายประสาทเทียมสามารถทำงานได้

3) การเรียนรู้เชิงถ่ายโอนข้อมูลเป็นการโอนทั้งข้อดีและข้อเสียของตัวแบบหนึ่งไปยังอีกตัวแบบหนึ่งเพื่อใช้ในการแก้ปัญหาใหม่ การทำความเข้าใจข้อเสีย และการดูความเหมาะสมนั้นเป็นสิ่งสำคัญในการใช้เทคนิคนี้ จึงจำเป็นจะต้องมีการทดลองและทดสอบกับตัวแบบใหม่ที่ได้จากการเรียนรู้จากตัวแบบเดิม เช่นเดียวกันกับการไม่ไว้ใจจากการที่เด็กขัดขับรถของเล่น และจะไปขับรถของจริง ดังนั้นการเรียนรู้เชิงถ่ายโอนข้อมูลอาจจะไม่สามารถแก้ปัญหาได้ทุกกรณี

แนวคิดทั่วไปในเทคนิคการเรียนรู้เชิงถ่ายโอนข้อมูล โดยมีข้อกำหนดในการเรียนรู้เชิงถ่ายโอนข้อมูลคือ การเรียนรู้เชิงถ่ายโอนข้อมูลจำเป็นต้องมีความสามารถในการถ่ายทอดความรู้จากโดเมนหนึ่งไปยังอีกโดนเมนหนึ่ง



ภาพที่ 2.11 แสดงชั้นการเรียนรู้เชิงถ่ายโอนข้อมูล

ที่มา: https://www.datacamp.com/community/tutorials/transfer-learning

จากภาพที่ 2.11 แสดงให้เห็นถึงการปรับเปลี่ยนและใช้พารามิเตอร์ร่วมกันในชั้นแรก ๆ โดยที่ชั้นผลลัพธ์เป็นงานที่แตกต่างกันโดย Task A และ Task B และ Task C นั้นเป็นงานที่แตกต่างกันออกไปเช่น Task A อาจใช้ในการจดจำภาพชนิดของดอกไม้ และ Task B ใช้ในการจดจำภาพชนิดของรถ และ Task C เป็นการจดจำภาพของใบหน้าคน เป็นต้น จากหลักการการเรียนรู้เชิงถ่ายโอนข้อมูลจะนำโครงข่ายที่มีอยู่มาใช้ซ้ำโดยใช้ค่าพารามิเตอร์ต่าง ๆ เช่นค่าน้ำหนัก หรือค่าไบแอส ของโครงข่ายที่เคยได้เรียนรู้ด้วยข้อมูลที่มีปริมาณมาก โดยนำชั้นดังกล่าว (Shared layers) นำมาใช้กับงานที่ต้องการโดยไม่ต้องทำการฝึกฝนทั้งระบบใหม่ทั้งหมด เพียงแต่ฝึกฝนในส่วนของการรู้จำเพื่อใช้ในการจำแนกประเภทหรือส่วนที่ใช้ในการพยากรณ์เท่านั้น

2. งานวิจัยที่เกี่ยวข้อง

วศิน ทับแสง (2557) ได้นำเสนองานวิจัยเรื่อง การตรวจจับและรู้จำตราสัญลักษณ์ของรถยนต์โดยใช้วิธีการโครงข่ายประสาทแบบคอนโวลูชันและพีระมิด ฮิสโตแกรมของทิศทางเกรเดียนต์ โดยศึกษาเกี่ยวกับการตรวจจับและรู้จำตราสัญลักษณ์ของรถยนต์โดยได้ทดลองวิธีการ CNN แต่พบว่าอัตราความแม่นยำค่อนข้างต่ำ จึงได้นำเสนอคุณลักษณะเด่นของ PHOG เพื่อนำมาช่วยใน

การกำจัดส่วนที่ไม่ใช่ตราสัญลักษณ์ออกไปจากภาพเพื่อความถูกต้องและแม่นยำมากยิ่งขึ้น โดยใช้คุณลักษณะเด่นของ PHOG ซึ่งจะแบ่งภาพออกเป็นส่วนเล็ก ๆตามสมการ และกำหนดระดับของพิรามิดเป็น 1 แล้วนำมาสกัดคุณลักษณะเด่นด้วย HOG เมื่อได้คุณลักษณะเด่นแล้วนำมาต่อกันเป็นคุณสมบัติเด่นโดยรวมของภาพและทำการจำแนกโดยใช้วิธีการ SVM ซึ่งฝึกสอนโดยใช้ข้อมูลเดียวกันกับการฝึกสอน CNN แต่แบ่งข้อมูลคนละชุด ชุดข้อมูลที่นำมาฝึกสอนมีจำนวน 3,000 ถึง 7,000 ภาพ ในส่วนของ SVM จะเลือกจำแนกประเภทเพียงสองคลาสเท่านั้นคือ คลาสที่เป็นตราสัญลักษณ์กับคลาสที่ไม่ใช่ตราสัญลักษณ์ จากการทดสอบพบว่าการใช้คุณลักษณะเด่นของ PHOG ร่วมกับการจำแนกประเภท SVM ทำงานร่วมกับ CNN สามารถลดข้อผิดพลาดเป็นอย่างมากทำให้ค่าความแม่นยำที่สูง แต่งานวิจัยมีข้อจำกัดคือประสิทธิภาพจะลดลงเมื่อมีการถ่ายภาพในลักษณะก้มลงด้านล่างหรือเงยขึ้นด้านบนหรือถ้าถ่ายจากด้านข้างเพราะวิธีการนี้ไม่ทนทานต่อภาพวัตถุที่มีการหมุนหรือบิด หดเข้าหรือยืดออก และระบบยังไม่พัฒนาเป็นแบบเรียลไทม์

อรรณพ ขันธิกุล (2544) ได้เสนองานวิจัยเรื่องการจำแนกใบไม้โดยใช้เครือข่ายประสาทเทียมแบบคอนโวลูชันนอล ซึ่งได้ศึกษาเกี่ยวกับการจำแนกใบไม้ในพืชในกลุ่มของสมุนไพร ก่อนที่จะนำพืชไปใช้ประโยชน์จำเป็นต้องจำแนกว่าพืชเป็นชนิดใดและใบไม้ก็เป็นอวัยวะของพืชที่เหมาะสมสำหรับใช้ในการจำแนก เนื่องจากมีคุณลักษณะสำคัญที่โดดเด่นและง่ายต่อการเก็บตัวอย่างมาวิเคราะห์ ใบไม้แต่ละใบมีคุณลักษณะสำคัญที่ต่างกันทำให้การสร้างโมเดลทางคณิตศาสตร์ ของรูปใบไม้มีความแตกต่างกันไป การศึกษารูปใบไม้ทุกชนิดจึงทำได้ยาก จึงโดยเสนอแนวคิดการใช้เครือข่ายประสาทเทียมแบบคอนโวลูชัน เพื่อจำแนก รูปใบไม้ ตามชนิดของพืชโดยอาศัยคุณลักษณะสำคัญของใบไม้แต่ละชนิด นำไปฝึกสอนในโครงข่ายประสาทเทียมแบบคอนโวลูชัน ในวิทยานิพนธ์มีการสร้างระบบจำแนกใบไม้ที่อยู่ในกลุ่มของสมุนไพร ทั้งที่มีรูปร่างเหมือนกันและแตกต่างกันจำนวน 15 ชนิด ผลการทดลองกับใบไม้ แสดงให้เห็นว่าโครงข่ายประสาทเทียบแบบคอนโวลูชันมีข้อดีกว่าแบบเครือข่ายประสาทเทียมแบบมัลติเลเยอร์เพอร์เซ็พตรอน และระบบสามารถจำแนกใบไม้ที่มีคุณลักษณะ สำคัญได้อย่างแม่นยำ แต่พบว่าประสิทธิภาพการจำแนกประเภทจะลดลงเมื่อนำใบไม้ที่มีลักษณะคล้ายกันมาทำการทดสอบ โดยระบบการจำแนกประเภทนี้มีค่าเฉลี่ยความผิดพลาดเท่ากับ 17.78 เปอร์เซ็นต์

สุทิพา เผือกผ่อง (2559) ได้เสนองานวิจัยเรื่องการเรียนรู้เชิงลึกโดยใช้เครือข่ายประสาทเทียมแบบคอนโวลูชันสำหรับจำภาพใบหน้าบุคคล โดยในปัจจุบันโครงข่ายประสาทเทียมแบบคอนโวลูชันเป็นระบบรู้จำที่มีประสิทธิภาพใกล้เคียงมนุษย์ แต่ต้องใช้ข้อมูลจำนวนมากในการฝึกฝน ซึ่งเป็นเรื่องยากเพราะต้องประยุกต์เข้ากับงานระบบความปลอดภัย จึงนำเสนอระบบรู้จำภาพใบหน้าบุคคลโดยใช้โครงข่ายประสาทเทียมแบบคอนโวลูชันด้วยวิธีการถ่ายโอนเชิงการเรียนรู้ ใน

ตอนแรก CNN จะถูกฝึกฝนด้วยข้อมูลขนาดใหญ่ จากนั้นชั้นจำแนกจะถูกแทนที่ด้วยชั้นใหม่และฝึกฝนด้วยภาพซึ่งเป็นฐานข้อมูลขนาดเล็ก 2 ฐานข้อมูลคือ ฐานข้อมูล 9 คน และ 109 คนถูกนำมาใช้ในการศึกษาประสิทธิภาพการรู้จำภาพใบหน้าบุคคล โดยใช้โครงสร้างแบบ LeNet ผลการทดลองพบว่าการใช้ CNN แบบถ่ายโอนการเรียนรู้ให้ความถูกต้องที่สามารถยอมรับได้ร้อยละ 80 เนื่องจากค่าน้ำหนักที่ได้จากฐานข้อมูลขนาดใหญ่ซึ่งมีภาพในฐานข้อมูลจำนวนมากทำให้ประสิทธิภาพการเรียนรู้คุณลักษณะของ CNN เพิ่มขึ้น และเมื่อนำค่าน้ำหนักเริ่มต้นไปกำหนดเป็นค่าเริ่มต้นของฐานข้อมูลขนาดเล็กช่วยเพิ่มประสิทธิภาพการรู้จำใบหน้าบุคคลได้

Ariadna Quattoni (2009) ได้เสนองานวิจัยเรื่อง Transfer Learning Algorithms for Image Classification โดยได้ทำการศึกษาการจำแนกรูปภาพโดยใช้ประโยชน์คุณสมบัติที่มีความซับซ้อนของโครงข่ายประสาทเทียมและ SIFT Scale-Invariant Feature Transform) ในการสกัดคุณลักษณะสำคัญ ทำการทดลองพัฒนาอัลกอริทึมการเรียนรู้เชิงถ่ายโอนข้อมูล โดยใช้ข้อมูล 2 ชุด คือชุดข้อมูลที่ไม่มีการรระบุค่าเป้าหมายหรือไม่ระบุ Label และข้อมูลชุดที่ 2 มีการกำหนดค่าลาเบลจากหมวดหมู่ที่เกี่ยวข้อง จากนั้นทำการออกแบบโครงสร้างเครือข่ายและทำการฝึกฝนชุดข้อมูลที่ไม่ได้กำหนดค่าลาเบลหรือค่าเป้าหมายไว้เมื่อทำการฝึกฝนเสร็จ ได้ออกแบบอัลกอริทึมอีกตัวเพื่อโอนข้อมูลการเรียนรู้จากการฝึกฝนแล้ว เพื่อนำมาฝึกฝนซ้ำด้วยชุดข้อมูลอีกชุดที่มีการกำหนดค่าเป้าหมายหรือลาเบล ในการทดลองนี้ได้ทดลองเกี่ยวกับการจำแนกประเภทรูปภาพของการพาดหัวข่าวว่าเป็นข่าวประเภทใด เช่นฟุตบอล นักแสดง หรือการประกาศผลรางวัล เป็นต้น และจากการพัฒนาสามารถที่จะถ่ายโอนความรู้จากงานหนึ่งไปแก้ไขปัญหาของอีกงานหนึ่งที่มีรูปแบบเกี่ยวข้องกันได้

Muhind Salim Hmoud Alradad (2015) ได้เสนองานวิจัยเรื่อง Robust classification with convolutional neural network ได้นำเสนอหลักการจำแนกรูปภาพโดยใช้โครงข่ายประสาทเทียมแบบคอนโวลูชัน โดยวิธีการรวมเอา Max Pooling และ Average Pooling เข้าด้วยกันก่อนที่จะเป็นข้อมูลนำเข้าในส่วนของการจำแนกประเภท สำหรับการแยกประเภทของ เพศและอายุของบุคคลที่อยู่ในรูปภาพ โดยใช้รูปภาพจากใบหน้าโดยการรวบรวมชุดข้อมูล LFW (Labeled Faces in the Wild Dataset) เพื่อใช้ในการฝึกฝนมาจากเว็บไซต์โดยมีรูปใบหน้าเพศชายจำนวน 4,264 รูป และเพศหญิง 1,486 รูป และให้ผลสรุปของค่าความถูกต้องอยู่ที่ 95.5 เปอร์เซ็นต์ ในส่วนของการพยากรณ์อายุได้จัดกลุ่มออกเป็น 7 กลุ่มคือช่วงอายุ 0-2, 3-7, 8-12, 13-19, 20-36, 37-65 และ 65 ขึ้นไป โดยใช้ชุดข้อมูลเดียวกันกับการแยกเพศ โดยให้ค่าความถูกต้องโดยเฉลี่ยเท่ากับ 69 เปอร์เซ็นต์

Jacopo Credi (n.d.) ได้เสนองานวิจัยเรื่อง Traffic sign classification with deep convolutional neural network โดยในงานวิจัยนี้เกี่ยวกับการเรียนรู้เชิงลึก (Deep learning) ที่กำลัง

เติบโตเป็นอย่างมากทั้งในแวดวงทางวิชาการและอุตสาหกรรม ทางวิศวกรจำเป็นจะต้องพัฒนาเพื่อให้ทันต่อความต้องการ โดยครอบคลุมได้หลายหลากแพล็ตฟอร์ม หนึ่งในนั้นคือโครงข่ายประสาทเทียมแบบคอนโวลูชัน ซึ่งได้มีการพัฒนาโครงสร้างเพื่อใช้งานได้หลากหลาย งานวิจัยนี้เริ่มจากแนวคิดที่จะสร้างเครื่องมือภายใต้เฟรมเวิร์ค .NET มาช่วยในการสร้างและฝึกฝนเครือข่ายเชิงลึก โดยโปรแกรมภาษา C# และผสมผสานกับโปรแกรม .NET ทั้งนี้การฝึกฝนต้องอาศัยการประมวลผล GPU (Graphic Processing Unit) หรืออุปกรณ์อื่น ๆที่สนับสนุน OpenCL ซึ่งเป็นกระบวนการประมวลผลที่ทำคู่ขนานกันไป ในงานวิจัยนี้ใช้ไลบรารี่ Conv.NET เพื่อฝึกฝนเพื่อหาเครื่องหมายการจราจร โดยใช้ชุดข้อมูล GTSRB (German Traffic Sign Recognition Benchmark data set) โดยใช้สถาปัตยกรรมแบบ LeNet (LeNet เป็นโครงข่ายประสาทเทียมแบบคอนโวลูชันยุคแรก และมีโครงสร้างไม่ต่างจากโครงสร้างเครือข่ายประสาทเทียมในปัจจุบัน แต่เนื่องจากคอมพิวเตอร์ตอนนั้นมีประสิทธิภาพไม่สูงมากและมีข้อมูลจำกัดจึงทำให้ไม่ได้เป็นที่รู้จักมากนัก) โดยมีความถูกต้องของการจำแนกประเภทเท่ากับ 96.2 เปอร์เซ็นต์ โดยหวังว่าจะเป็นไลบรารี่ที่เป็นจุดเริ่มต้นสำหรับโครงการในอนาคตในการพัฒนาที่เป็นแบบโอเพ่นซอร์สที่กลุ่มนักวิจัยสามารถมาปรับระบบเพื่อให้สามารถใช้งานได้หลายด้าน

Ernest Bofill Ylla (2017) เสนองานวิจัยเรื่อง Synthesizing Images to Recognize Natural Images with Transfer Learning in Convolutional Neural Network โดยได้ใช้โครงข่ายประสาทเทียมแบบคอนโวลูชันด้วยวิธีการเรียนรู้เชิงถ่ายโอนข้อมูล เพื่อทดลองหาค่าความถูกต้องและแม่นยำในการรู้จำวัตถุ ซึ่งวัตถุที่ใช้ในการทดลองคือตัวต่อ Lego จำนวน 10 ชิ้นหรือ 10 คลาส ซึ่งมีลักษณะที่แตกต่างกัน การทดลองเพื่อเปรียบเทียบข้อมูลที่ใช้ในการฝึกฝนเป็นแบบภาพที่เกิดจากการสังเคราะห์หรือภาพที่ถูกตัดต่อจากคอมพิวเตอร์เปรียบเทียบกับภาพที่ถ่ายจากภาพจริงจากกล้อง โดยการทดลองแบ่งกลุ่มตัวอย่างในการใช้เป็นข้อมูลฝึกฝนเป็น 3 กรณีคือ กรณีที่ 1 ใช้ภาพที่ถ่ายจากภาพจริงโดยกล้องจำนวนคลาสละ 30 ภาพ กรณีที่ 2 ใช้ภาพถ่ายจากภาพจริงจากกล้องจำนวนคลาสละ 30 ภาพและใช้ภาพที่ได้จากการตัดต่อจากคอมพิวเตอร์จำนวน 1,800 ภาพต่อคลาส กรณีที่ 3 ใช้ภาพที่ได้จากการตัดต่อจากคอมพิวเตอร์ทั้งหมดจำนวนคลาสละ 5,400 ภาพ จากนั้นทำการทดสอบโดยใช้ภาพที่ได้จากการถ่ายภาพด้วยกล้องเพื่อให้ตัวแบบทำการพยากรณ์ ผลการทดลองพบว่า กรณีที่ 1 ให้ค่าความถูกต้องเท่ากับ 82 เปอร์เซ็นต์ กรณีที่ 2 ให้ค่าความถูกต้องเท่ากับ 68 เปอร์เซ็นต์ และกรณีที่ 3 ให้ค่าความถูกต้องเท่ากับ 60.67 เปอร์เซ็นต์ ผลการทดลองชี้ให้เห็นว่าโครงข่ายประสาทเทียมแบบคอนโวลูชันความสามารถในการจดจำวัตถุลดลงเมื่อได้ฝึกฝนจากภาพที่ถูกสร้างขึ้นมาเมื่อเปรียบเทียบกับการฝึกฝนจากภาพถ่ายจริง ถึงแม้ว่าภาพถ่ายจริงจะมีปริมาณข้อมูลการฝึกฝนน้อยกว่าแต่สามารถให้ค่าความถูกต้องที่สูงกว่า

Qinghui Liu (2017) ได้เสนองานวิจัยเรื่อง Deep Learning Applied to Automatic Polyp Detection in Colonoscopy Images ซึ่งทำการศึกษาเทคนิคการตรวจจับหาเนื้องอกลำไส้ใหญ่โดยผ่านกล้องส่องภาพและระบุว่าภาพที่จับได้เป็นเนื้องอกหรือไม่เป็นเนื้องอก และเพื่อเรียนรู้อัลกอริทึมที่จะสามารถนำมาใช้งานในการวินิจฉัยโรคแบบอัตโนมัติและแบ่งคลาสการพยากรณ์ไว้ 2 คลาสคือเป็นและไม่เป็นเนื้องอก โดยได้เปรียบเทียบวิธีการ 3 แบบคือ 1) การใช้เทคนิควิธีการเรียนรู้ของเครื่องแบบดั้งเดิม (ML: Machine Learning) โดยการใช้วิธีการประมวลผลภาพในระดับต่ำ เช่น ฮิสโตแกรม เพื่อหาคุณลักษณะสำคัญของภาพก่อนนำมาจำแนกประเภท 2) ใช้เทคนิคการเรียนรู้ของเครื่องระดับลึก (DL : Deep Learning) ซึ่งจะใช้โครงข่ายประสาทเทียมแบบคอนโวลูชันมาทำการออกแบบเพื่อสกัดคุณลักษณะสำคัญของภาพพร้อมทั้งจำแนกประเภท 3) ใช้เทคนิคการเรียนรู้เชิงถ่ายโอนข้อมูล (TL : Transfer Learning)โดยใช้สถาปัตยกรรมของโครงข่ายประสาทเทียบแบบคอนโวลูชัน ที่มีการฝึกฝนข้อมูลมาแล้ว (Pre-trained model) มาทำการฝึกฝนข้อมูลในชั้นสุดท้าย (Final layer) เพื่อจำแนกประเภท ผลการทดลองพบว่าการใช้เทคนิคการเรียนรู้เชิงถ่ายโอนข้อมูลให้ค่าความถูกต้องเท่ากับ 96.00 เปอร์เซ็นต์ ค่าความแม่นยำ 96.00 เปอร์เซ็นต์ และค่าประสิทธิภาพโดยรวมของระบบเท่ากับ 96.00 เปอร์เซ็นต์ซึ่งสูงกว่าอีก 2 วิธีที่ได้นำมาทำการทดลองเปรียบเทียบ และวิธีการนี้มีความยืดหยุ่นสามารถที่จะปรับขยายเพื่อใช้ในการตรวจจับสิ่งผิดปกติอื่น ๆ ได้ในอนาคต

Hanna Bjorgvinsdottir and Robin Seibold (2016) ได้นำเสนองานวิจัยเรื่อง Face Recognition Based on Embedded Systems โดยใช้โครงข่ายประสาทเทียมแบบคอนโวลูชัน และใช้เทคนิคการเรียนรู้เชิงถ่ายโอนข้อมูล โดยการนำตัวแบบที่มีการฝึกฝนแล้วด้วยข้อมูลขนาดใหญ่ชื่อ VGG-16 หรือ VGG-face model (The Oxford Visual Geometry Group) ซึ่งถูกฝึกฝนด้วยชุดข้อมูลที่เป็นใบหน้าคน 2.6 ล้านรูป เพื่อจำแนกหน้าคนทั้งหมด 2,622 คน มีการใช้ Dropout เพื่อป้องกันการเกิด Overfitting แต่งานวิจัยมุ่งเน้นการใช้งานที่อุปกรณ์แบบฝังตัวทำให้เกิดข้อจำกัดทั้งด้านตัวฮาร์ดแวร์และเวลาในการฝึกฝนการที่จะฝึกฝนชุดข้อมูลทั้งหมดจึงไม่ใช่ทางเลือกที่จะใช้กับงานวิจัยดังกล่าว ดังนั้นจึงใช้เทคนิคการเรียนรู้เชิงถ่ายโอนข้อมูลโดยการโอนความรู้ที่ได้จากการฝึกฝนจากตัวแบบดังกล่าวมาใช้กับงานวิจัยโดยการฝึกฝนข้อมูลเฉพาะชั้นสุดท้ายของตัวแบบเท่านั้น ในงานวิจัยได้ใช้เทคนิคการเรียนเชิงถ่ายโอนข้อมูลโดยการทดลองใช้รูปภาพในการฝึกฝนข้อมูลคลาสละ 80 ภาพจำนวน 2,904 คลาส ซึ่งให้ค่าความถูกต้องเท่ากับ 91.66 เปอร์เซ็นต์ในการจำแนกหน้าคนจำนวน 2,904 คน

Ondrej Zapletal (2017) ได้นำเสนองานวิจัยเรื่อง Image recognition by convolutional neural networks – basic concepts ซึ่งทำการศึกษาและรวบรวมแนวคิดและทฤษฎีพื้นฐานของ

โครงข่ายประสาทเทียแบบคอนโวลูชัน ศึกษาถึงโครงสร้างต่าง ๆ ของโครงข่ายที่มีผลต่อความสามารถการทำงานของโครงข่ายในการฝึกฝนข้อมูล ผลลัพธ์จะถูกนำมาเปรียบเทียบกับตัวแบบหรือโครงข่ายที่มีการพัฒนาและออกแบบจากการแข่งขันของ ILSVRC (The ImageNet Large Scale Visual Recognition Challenge) โดยการทดลองได้นำชุดข้อมูลจากการแข่งขันของ ILSVRC ซึ่งสุ่มมา 100 ชุดข้อมูลหรือ 100 คลาส และการเลือกแบบสุ่มนี้จะทำให้จำนวนของข้อมูลที่ใช้ในการฝึกฝนข้อมูลไม่เท่ากัน แต่จำนวนภาพโดยเฉลี่ยจะอยู่ที่ 500 ภาพ บางคลาสมีภาพถึง 1,000 ภาพ เพื่อใช้ในการทดลองตัวแบบที่สร้างขึ้นมานี้ได้ทำการปรับพารามิเตอร์ทีละตัวเช่นอัตราการเรียนรู้ (Learning Rate) และทำการฝึกฝนข้อมูลเพื่อหาความถูกต้องที่สูงที่สุด จากนั้นสร้างตัวแบบใหม่โดยการใช้พารามิเตอร์ที่ได้จากการทดลองดังกล่าวมาทำการทดลองกับข้อมูลชุดเดิมอีก 1 รอบ โดยผลจากการทดลองและเปรียบเทียบกับตัวแบบที่ได้จาการแข่งขันพบว่าตัวแบบที่ออกแบบเองให้ค่าความถูกต้อง (Top-5 Accuracy) เท่ากับ 84 เปอร์เซ็นต์ ซึ่งต่ำกว่าค่าความถูกต้องของตัวแบบที่ได้จากการแข่งขัน โครงข่ายแบบ GoogLeNet ซึ่งให้ค่าความถูกต้องเท่ากับ 93 เปอร์เซ็นต์

จากงานวิจัยที่เกี่ยวข้องแสดงให้เห็นการพัฒนาเพื่อใช้โครงข่ายประสาทเทียมแบบคอนโวลูชันเพื่อใช้ในการจำแนกประเภทของรูปภาพ รวมถึงใช้ในการอธิบายรูปภาพเนื่องมาจากเทคโนโลยีของทั้งฮาร์ดแวร์และซอฟต์แวร์มีการพัฒนามากในด้านความเร็วของการประมวลผล ทำให้การพัฒนาตัวแบบเป็นเรื่องที่สามารถที่ทำได้เพราะลดเวลาการฝึกฝนลงได้มากเนื่องจากความเร็วของอุปกรณ์ ในเรื่องความถูกต้องของการพยากรณ์ก็ให้ค่าความถูกต้องที่สามารถยอมรับได้ งานวิจัยชิ้นนี้ได้นำข้อดีของแต่ละงานวิจัยมาปรับปรุงและพัฒนาต่อเพื่อเพิ่มขีดความสามารถของโครงข่ายประสาทเทียมมแบบคอนโวลูชันและปรับเปลี่ยนเทคนิคเพื่อให้สามารถใช้ได้กับอุปกรณ์ฮาร์ดแวร์และซอฟต์แวร์ที่ใช้กันอยู่ทั่วไปในปัจจุบัน เช่นคอมพิวเตอร์ส่วนบุคคล หรือคอมพิวเตอร์โน้ตบุค เป็นต้น โดยในการพัฒนาระบบจดจำภาพสำหรับการตรวจจับอะไหล่โดยใช้โครงข่ายประสาทเทียมแบบคอนโวลูชัน จะใช้เทคนิคโครงข่ายประสาทเทียแบบคอนโวลูชันเนื่องจากเป็นการประมวลผลภาพขั้นสูง และเป็นการเรียนรู้ของเครื่องแบบเชิงลึก และนำวิธีการถ่ายโอนเชิงการเรียนรู้เพื่อใช้ตัวแบบที่ได้รับการฝึกฝนการข้อมูลที่มากเพราะค่าน้ำหนักและค่าไบแอสได้ทำการปรับให้เข้ากับการพยากรณ์ที่มีความถูกต้องที่สูง และได้ทำการปรับแต่งค่าพารามิเตอร์เพื่อเพิ่มประสิทธิภาพในการพยากรณ์ในเรื่องค่าความถูกต้อง และค่าความแม่นยำรวมถึงค่าประสิทธิภาพโดยรวมของระบบให้กับงานวิจัยชิ้นนี้ด้วย

บทที่ 3
วิธีดำเนินการวิจัย

วิธีการดำเนินงานวิจัยในการพัฒนาระบบจดจำภาพสำหรับการตรวจจับอะไหล่โดยใช้โครงข่ายประสาทเทียมแบบคอนโวลูชัน สามารถแบ่งขั้นตอนการดำเนินการดังนี้

1. การศึกษาและวิเคราะห์ปัญหา
2. การวิเคราะห์และขั้นตอนการทำงานของระบบ
3. ข้อมูลและกลุ่มตัวอย่างที่ใช้ในงานวิจัย
4. การพัฒนาระบบรู้จำอะไหล่
5. การทดสอบประสิทธิภาพของระบบ

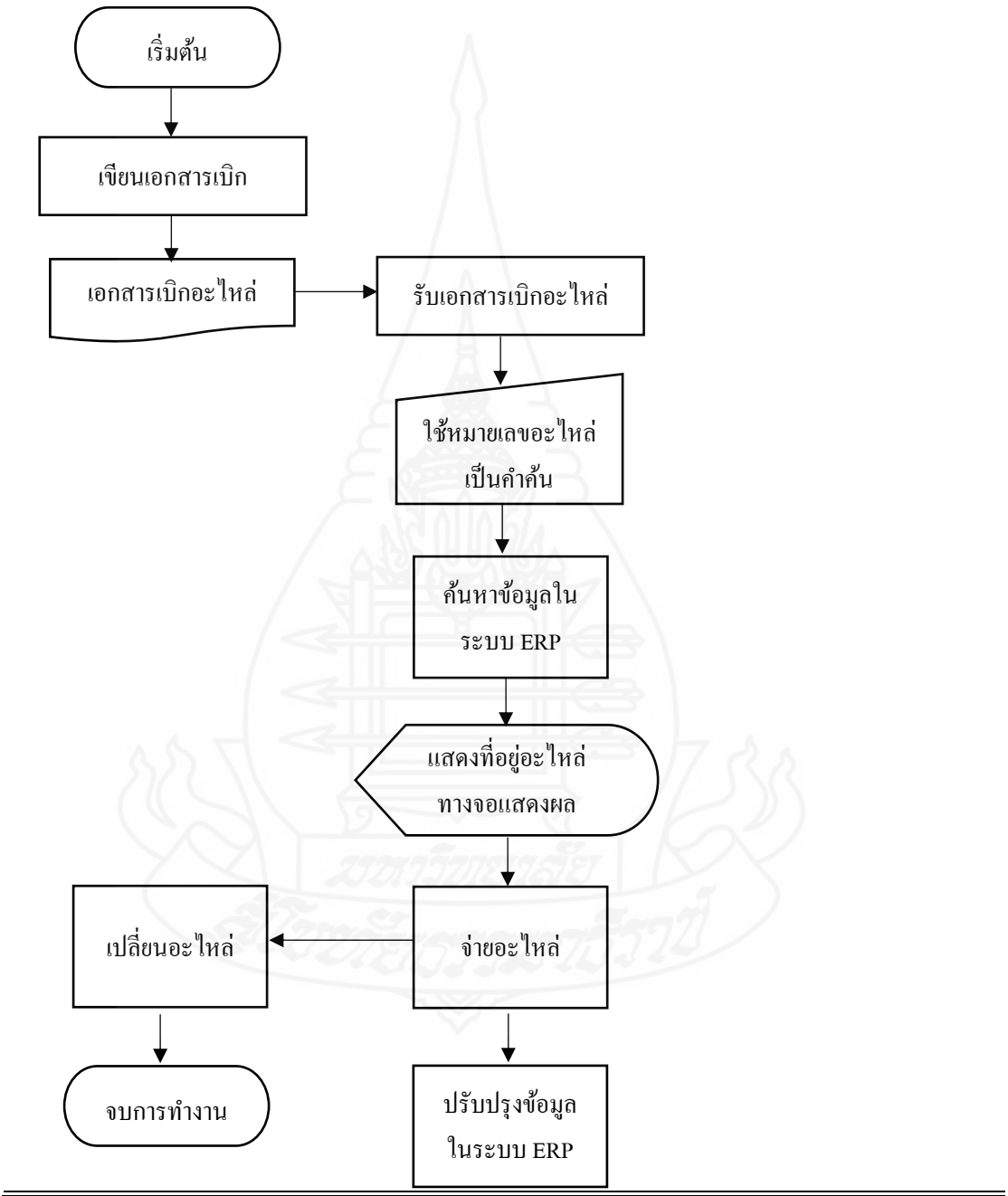
1. การศึกษาและวิเคราะห์ปัญหา

จากสถานการณ์ปัจจุบันการแข่งขันทางธุรกิจนั้นสูงมากหลาย ๆ บริษัทมุ่งเน้นพัฒนาในส่วนต่าง ๆ ในองค์กรเพื่อให้ได้เปรียบกว่าคู่แข่งขัน สิ่งสำคัญประการหนึ่งคือการส่งมอบสินค้าให้ลูกค้าที่ทันตามเวลาที่กำหนด ดังนั้นกระบวนการผลิตจึงเป็นกระบวนการที่สำคัญและมีผลต่อการส่งมอบสินค้ามาก การทำให้การผลิต ผลิตชิ้นงานออกมาได้อย่างต่อเนื่องโดยไม่มีปัญหาใด ๆ ย่อมเป็นสิ่งที่ทุกองค์กรปรารถนา แต่ในความเป็นจริงแล้วอุปสรรคและปัญหาอาจเกิดขึ้นได้ทุกเวลา ทั้งที่โดยตั้งใจและไม่ได้ตั้งใจจากพนักงาน ทั้งที่สามารถควบคุมได้และควบคุมไม่ได้จากสถานการณ์ต่าง ๆ เครื่องจักรก็เป็นปัจจัยหนึ่งของกระบวนการผลิตซึ่งมีการใช้งานทุกวัน โอกาสที่ชิ้นส่วนหรืออะไหล่เกิดการชำรุดเสียหายจึงมีสูง เมื่ออะไหล่เกิดชำรุดเสียหายกระบวนการผลิตจึงหยุดลงเพื่อรอการเปลี่ยนอะไหล่ แต่ในการเบิกอะไหล่จะต้องใช้หมายเลขอะไหล่ที่ลงทะเบียนไว้ในระบบ โดยบริษัทจะมีระบบการวางแผนทรัพยากรองค์กร ซึ่งเป็นระบบที่บูรณาการระบบภายในองค์กรทุกแผนกเข้าด้วยกันรวมถึงการจัดการกับอะไหล่ก็รวมอยู่ในระบบนี้ด้วย ทุกครั้งที่พนักงานจะเบิกแต่ไม่สามารถบอกได้ว่าอะไหล่ที่จะเบิกนั้นหมายเลขอะไร เพราะไม่ได้ระบุไว้ในตัวอะไหล่ ทำให้การค้นหาเป็นเรื่องที่ยุ่งยาก

1.1 System flow ของระบบงานปัจจุบัน

1.1.1 System flow ระบบการเบิกจ่ายอะไหล่ปัจจุบัน กรณีทราบรหัสหรือหมายเลขอะไหล่

ตารางที่ 3.1 System flow ระบบการเบิกจ่ายอะไหล่ปัจจุบันกรณีทราบหมายเลขอะไหล่

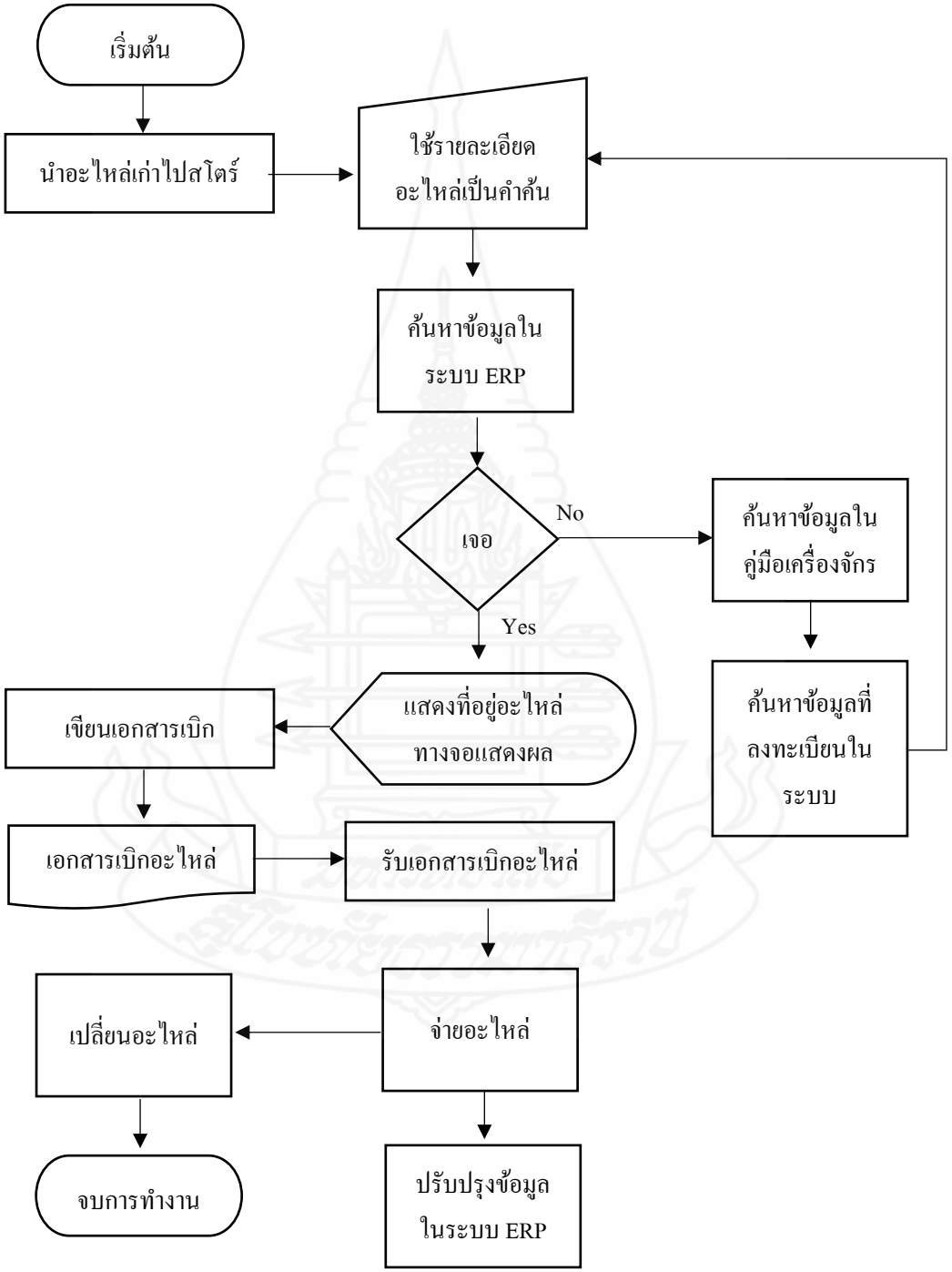


1.1.2 System flow ระบบการเบิกจ่ายอะไหล่ปัจจุบัน กรณีไม่ทราบรหัสหรือหมายเลขอะไหล่

ตารางที่ 3.2 System flow ของระบบการเบิกจ่ายอะไหล่ปัจจุบันกรณีไม่ทราบหมายเลขอะไหล่

ระบบการเบิกจ่ายอะไหล่ปัจจุบัน : กรณีไม่ทราบรหัสหรือหมายเลขของอะไหล่

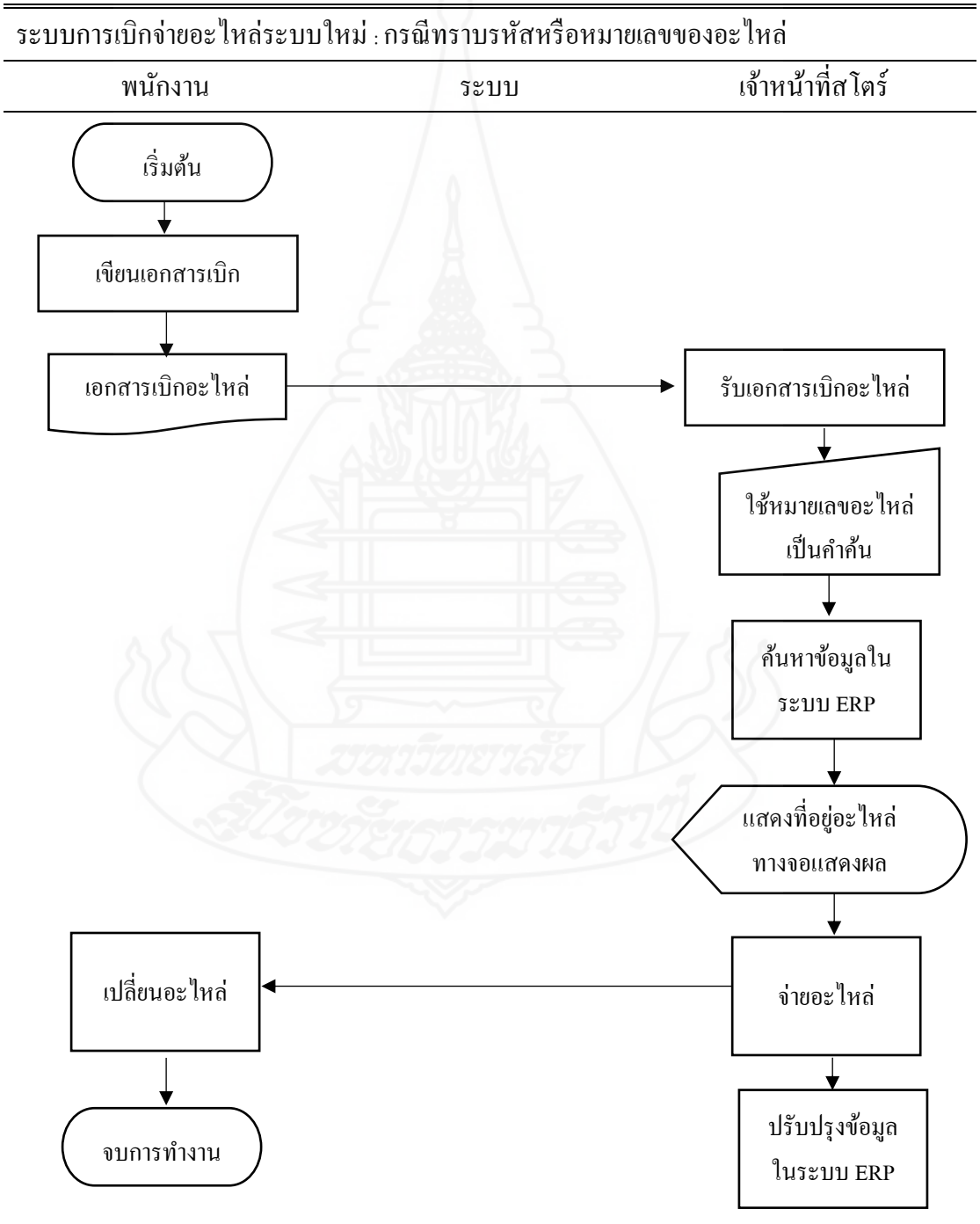
พนักงาน	เจ้าหน้าที่สโตร์	เจ้าหน้าที่ฝ่ายซ่อมบำรุง



1.2 System flow ของระบบงานใหม่

1.2.1 System flow ระบบการเบิกจ่ายอะไหล่ปัจจุบัน กรณีทราบรหัสหรือหมายเลขอะไหล่

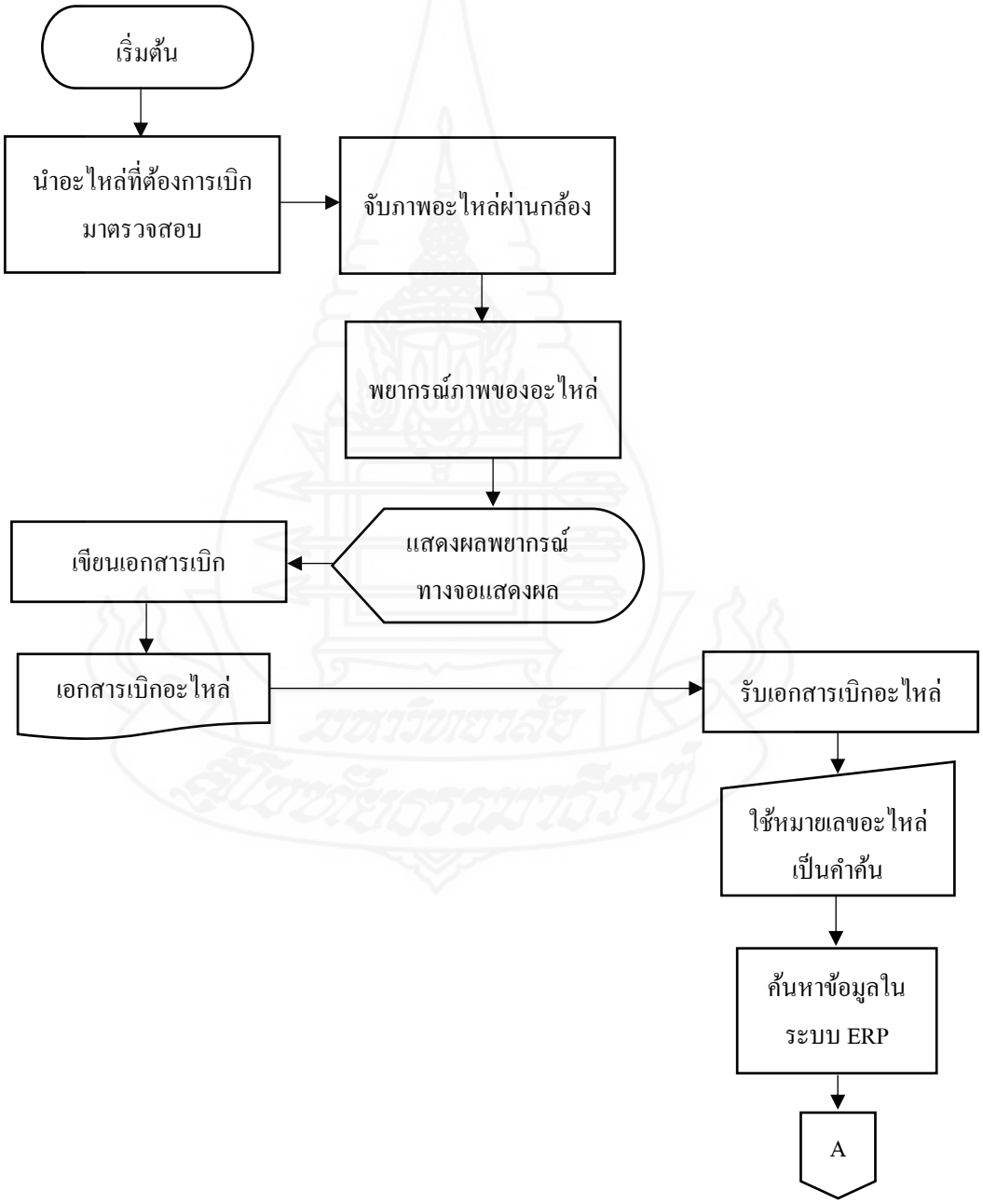
ตารางที่ 3.3 System flow ของระบบการเบิกจ่ายอะไหล่ใหม่กรณีทราบหมายเลขอะไหล่



1.2.2 System flow ระบบการเบิกจ่ายอะไหล่ปัจจุบัน กรณีไม่ทราบรหัสหรือหมายเลขอะไหล่

ตารางที่ 3.4 System flow ของระบบการเบิกจ่ายอะไหล่ใหม่กรณีไม่ทราบหมายเลขอะไหล่

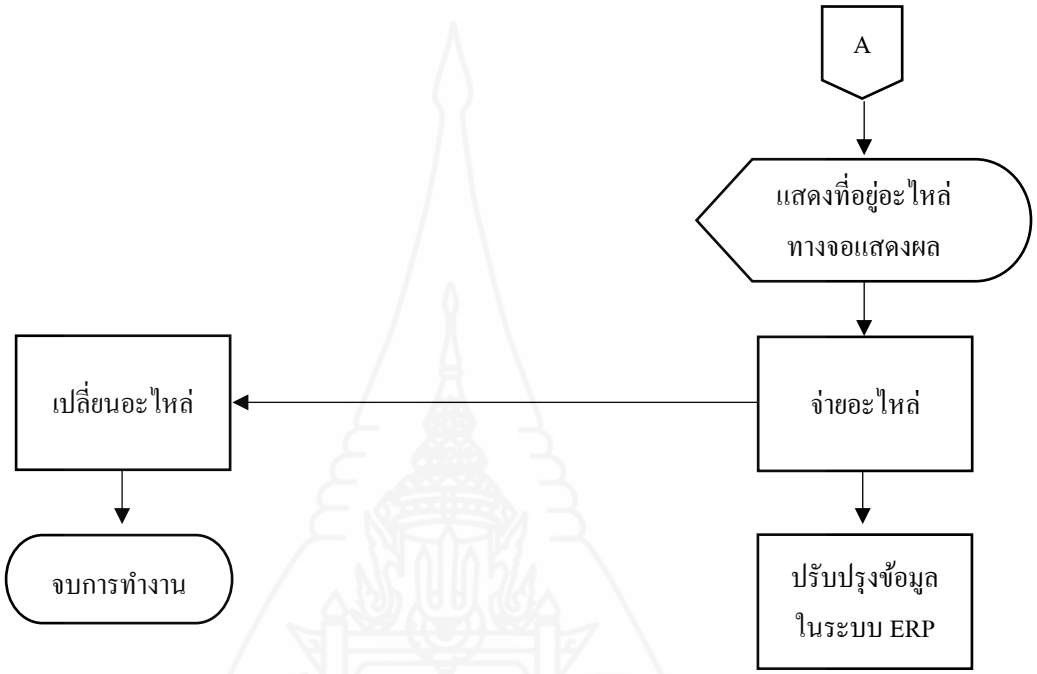
ระบบการเบิกจ่ายอะไหล่ระบบงานใหม่ : กรณีไม่ทราบรหัสหรือหมายเลขของอะไหล่		
พนักงาน	ระบบ	เจ้าหน้าที่สโตร์



ตารางที่ 3.4 (ต่อ)

ระบบการเบิกจ่ายอะไหล่ระบบงานใหม่ : กรณีไม่ทราบรหัสหรือหมายเลขของอะไหล่

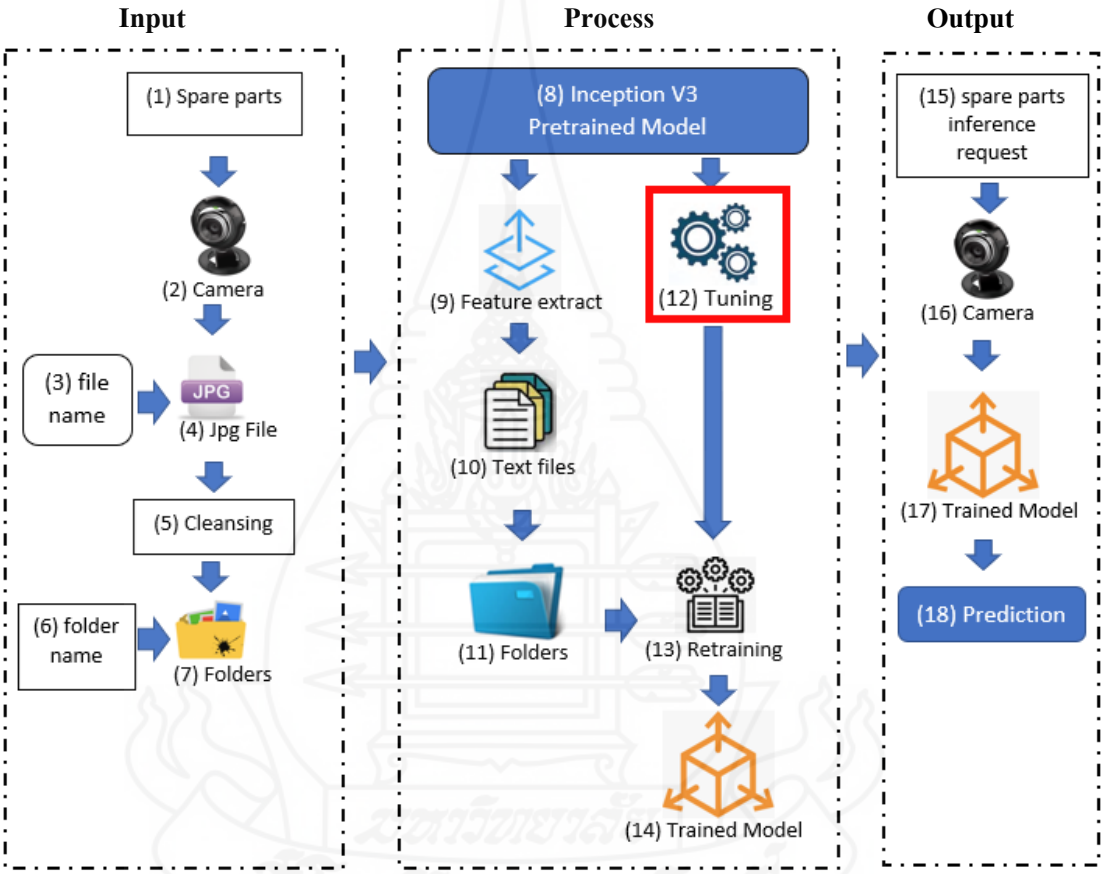
พนักงาน	ระบบ	เจ้าหน้าที่สโตร์



การค้นคืนด้วยภาพหรือการจดจำภาพจึงเป็นทางเลือกที่ผู้วิจัยต้องการที่จะศึกษาเพื่อใช้ในการแก้ไขปัญหานี้ โดยพนักงานสามารถนำอะไหล่ที่ต้องการจะเปลี่ยนมาทำการค้นหาแทนการจดจำหมายเลขของอะไหล่ เพื่อใช้ในการเบิกจ่ายในระบบของบริษัทต่อไป จะเป็นการลดเวลาในการค้นหาอันเนื่องมาจากการไม่รู้หมายเลขหรือรหัสของอะไหล่

2. การวิเคราะห์และออกแบบขั้นตอนการทำงานของระบบ

โครงสร้างของระบบ สามารถแบ่งกระบวนการทำงานย่อยออกเป็น 3 ส่วน คือ ส่วนการนำข้อมูลนำเข้า (Input) ส่วนประมวลผลข้อมูล (Process) และส่วนผลลัพธ์ที่ได้จากการประมวลผลข้อมูล (Output) ดังแสดงรายละเอียดในภาพที่ 3.1



ภาพที่ 3.1 แสดงโครงสร้างของระบบ

จากภาพที่ 3.1 แสดงโครงสร้างของระบบโดยเริ่มจากข้อมูลนำเข้าซึ่งเป็นอะไหล่ของจริงนำมาเก็บภาพผ่านทางกล้องและบันทึกไฟล์พร้อมกำหนดชื่อไฟล์ จากนั้นนำรูปภาพมาทำความสะอาดโดยการคัดเลือกรูปที่มีความคมชัดและมีภาพอะไหล่อยู่ในภาพ ทำการย้ายรูปภาพที่ได้ทำความสะอาดแล้วมาเก็บไว้ในแฟ้มข้อมูลพร้อมกำหนดชื่อโดยการกำหนดเป็นหมายเลขและชื่อของอะไหล่ผลลัพธ์ของกระบวนการนี้คือได้แฟ้มข้อมูลที่มีไฟล์อะไหล่ที่ถูกทำความสะอาดแล้ว จากนั้นเข้าสู่กระบวนการสร้างตัวแบบโดยเริ่มจากการเรียกใช้ตัวแบบที่ได้รับการฝึกฝนแล้วนำมาสกัด

คุณลักษณะเด่นของภาพของไฟล์ที่ได้จากกระบวนการเตรียมข้อมูล และกระบวนการนี้จะทำการฝึกฝนข้อมูลโดยใช้การเพิ่มชั้นของ Fully connected และ Softmax ซึ่งจะใช้ตัวน้ำหนักและไบแอสของตัวแบบที่ได้รับการฝึกฝนมาแล้วเป็นค่าตั้งต้นและทำการปรับให้เข้ากับข้อมูลอะไหล่ที่นำมาฝึกฝน กระบวนการนี้จะได้ตัวแบบที่จะใช้กับระบบจดจำภาพสำหรับการตรวจจับอะไหล่ กระบวนการสุดท้ายเป็นกระบวนการของการนำไปใช้งานโดยผู้ใช้จะนำอะไหล่ที่ต้องการให้ระบบพยากรณ์มาให้กล้องจับภาพระบบจะส่งภาพให้ตัวแบบทำการพยากรณ์และส่งผลลัพธ์ที่เป็นหมายเลข ชื่อของอะไหล่รวมถึงเปอร์เซ็นต์ค่าความเชื่อมั่นว่าเป็นอะไหล่ชิ้นนั้น ซึ่งแสดงผลในลักษณะแบบเรียลไทม์

2.1 อินพุต (Input)

2.1.1 ลำดับที่ (1) คือ Spare parts เป็นชิ้นส่วนอะไหล่ที่ต้องการให้ระบบจดจำ เป็นการเตรียมการนำเข้าของข้อมูลอะไหล่โดยนำอะไหล่ที่ต้องการให้ระบบจดจำทั้งหมดมาเตรียมเพื่อรวบรวมข้อมูลภาพของอะไหล่

2.1.2 ลำดับที่ (2) คือ Camera เป็นกล้องสำหรับจับหรือถ่ายภาพ เป็นการสร้างภาพจากอะไหล่จริงมาเป็นภาพที่ใช้กับคอมพิวเตอร์โดยการแปลงจากภาพจริงมาเป็นรหัสคอมพิวเตอร์เพื่อสามารถเก็บและอ่านได้จากคอมพิวเตอร์ระบบทำการเก็บภาพผ่านกล้องจับภาพโดยผู้ดูแลระบบมีหน้าที่หมุนกล้อง หรือหมุนอะไหล่ เพื่อให้กล้องจับภาพอะไหล่ได้ทุกมุมมองเพื่อเก็บรายละเอียดของอะไหล่ที่ต้องการให้ระบบจดจำมากที่สุด

2.1.3 ลำดับที่ (3) คือ file name เป็นการตั้งชื่อไฟล์ เป็นการใช้รหัสคอมพิวเตอร์ที่ถูกสร้างขึ้นมาเพื่อให้สามารถกำหนดชื่อของไฟล์ที่ได้จากการสร้างเพื่อให้สามารถใช้งานได้ตรงความต้องการของผู้ออกแบบระบบ ระบบจะทำการบันทึกภาพแต่ละภาพโดยกำหนดชื่อให้ภาพโดยอัตโนมัติเพื่อป้องกันการซ้ำกันของชื่อของภาพ ในงานวิจัยนี้ได้กำหนดให้ระบบกำหนดชื่อเป็นปีเดือนวันชั่วโมงนาทีวินาทีและตัวเลขอัตโนมัติ และจัดเก็บไว้ในฮาร์ดดิสก์ภายใต้ชื่อแฟ้มข้อมูล .../images ของเครื่องที่ระบบได้ถูกติดตั้งไว้

2.1.4 ลำดับที่ (4) คือ jpg file เป็นชนิดของไฟล์รูปภาพที่ได้จากการสร้างโดยรหัสคอมพิวเตอร์จะทำการกำหนดชื่อไฟล์ไว้แล้วจากกระบวนการก่อนหน้านี้ ซึ่งไฟล์นี้จะสนับสนุนการทำงานของกระบวนการสร้างการเรียนรู้จากการฝึกฝนตัวแบบของโครงข่าย

2.1.5 ลำดับที่ (5) คือ Cleansing เป็นกระบวนการทำความสะอาดภาพโดยผู้ดูแลระบบ โดยผู้ดูแลระบบจะทำการเลือกภาพที่ได้จากการสร้างและตั้งชื่อโดยคัดเลือกภาพที่มีความคมชัดและมีชิ้นส่วนอะไหล่อยู่ในภาพ มีภาพของสิ่งที่ไม่เกี่ยวข้องกับภาพอะไหล่น้อยที่สุด และที่สำคัญให้มีรูปอะไหล่อยู่ภายในภาพอย่างน้อย 60 เปอร์เซ็นต์ของชิ้นส่วนอะไหล่

2.1.6 ลำดับที่ (6) คือ folder name เป็นการตั้งชื่อแฟ้มข้อมูลโดยผู้ดูแลระบบ โดยใช้หลักการคือใช้ตัวเลข 6 หลัก เว้นวรรคและตามด้วยชื่อของอะไหล่ที่ต้องการให้ระบบจดจำ (ตัวเลข 6 หลัก อ้างอิงจากระบบการวางแผนทรัพยากรองค์กรของบริษัทที่ลงทะเบียนอะไหล่ด้วยตัวเลข 6 หลัก) ซึ่งหนึ่งแฟ้มข้อมูลจะเท่ากับข้อมูลอะไหล่หนึ่งชิ้น โดยมีข้อมูลรูปภาพที่ได้จากการบันทึกภาพหลากหลายมุมมองของอะไหล่ชิ้นนั้นบรรจุอยู่

2.1.7 ลำดับที่ (7) คือ folders เป็นแฟ้มข้อมูลชื่อ training_images เป็นการรวบรวมแฟ้มข้อมูลอะไหล่ทั้งหมดที่ต้องการให้ระบบรู้จำเข้ามาเก็บไว้ในแฟ้มข้อมูลนี้ ซึ่งเป็นขั้นตอนสุดท้ายของการจัดเตรียมข้อมูลเพื่อเป็นข้อมูลนำเข้าสำหรับการฝึกฝนเพื่อใช้สร้างในส่วนรู้จำอะไหล่ โดยผู้ดูแลระบบจะคัดลอกภาพที่ได้จากการทำความสะอาดแล้วมาเก็บไว้ในแฟ้มข้อมูลที่ชื่อ …/ training_images โดยสร้างแฟ้มข้อมูลใหม่และกำหนดชื่อเป็นหมายเลขของอะไหล่เว้นวรรคตามด้วยชื่ออะไหล่ เช่น …/ training_images / 602964 NOZZLE DISPENSING TIP

2.1.8 ทำซ้ำในขั้นตอนที่ 2.1.1 ถึง 2.1.7 สำหรับอะไหล่ชิ้นต่อไป ทำจนกว่าจะครบอะไหล่ทุกชิ้นที่ต้องการให้ระบบรู้จำ

2.2 กระบวนการ (Process)

2.2.1 ลำดับที่ (8) คือ Inception V3 Pretrained model เป็นกระบวนการเรียกใช้ตัวแบบที่ได้จากการฝึกฝนด้วยข้อมูลปริมาณมากไว้แล้วจากกูเกิล เพื่อใช้ในการถ่ายโอนความรู้ที่ได้จากการฝึกฝนมาใช้กับการรู้จำอะไหล่

2.2.2 ลำดับที่ (9) คือ feature extract เป็นกระบวนการทำงานของตัวแบบ Inception V3 ที่ฝึกฝนมาแล้วเพื่อสกัดคุณลักษณะของภาพอะไหล่ โดยเรียกใช้ไฟล์รูปภาพที่ได้จากกระบวนการเตรียมข้อมูล มาเข้าสู่ชั้น Convolution เพื่อสกัดคุณลักษณะเด่นของภาพ จากนั้นส่งข้อมูลมาชั้น AvgPool และ MaxPool เพื่อลดขนาดปริมาณข้อมูลนำเข้าโครงข่ายแต่ยังคงคุณลักษณะเด่นครบ ก่อนที่จะรวมข้อมูลทั้งหมดที่ชั้น Concat เพื่อจัดรูปแบบก่อนที่จะเข้าสู่ขั้นตอนต่อไป

2.2.3 ลำดับที่ (10) คือ Text file เป็นรูปแบบไฟล์คอมพิวเตอร์ชนิ่งหนึ่งที่รหัสโปรแกรมสร้างขึ้นเพื่อเตรียมสำหรับกระบวนการถัดไป เป็นการอ่านค่าที่ได้จากการสกัดคุณลักษณะมาบันทึกไว้ในไฟล์ชนิดนี้เพื่อเตรียมเป็นไฟล์รูปแบบการนำเข้าในกระบวนการฝึกฝนข้อมูล

2.2.4 ลำดับที่ (11) คือ Folders เป็นแฟ้มข้อมูลชื่อ bottleneck_data ที่รวบรวมไฟล์เก็บไฟ Textของแต่ละชิ้นส่วนอะไหล่ที่ได้จากการสกัดคุณลักษณะแล้วจัดเก็บไฟล์คุณลักษณะให้อยู่ในแฟ้มข้อมูลชื่อ …/ bottleneck_data โดยระบบจะสร้างแฟ้มข้อมูลใหม่เหมือนกับแฟ้มข้อมูล

ต้นทาง คือกำหนดชื่อเป็นหมายเลขของอะไหล่เว้นวรรคตามด้วยชื่อของอะไหล่ เช่น …/ bottleneck_data / 602964 NOZZLE DISPENSING TIP

2.2.5 ระบบจะทำซ้ำตั้งแต่ขั้นตอนที่ 2.2.1 ถึง 2.2.4 สำหรับข้อมูลอะไหล่ชิ้นต่อไป ทำเรื่อย ๆ จนครบอะไหล่ทุกชิ้นที่ได้จัดเก็บภาพไว้ในแฟ้มข้อมูลที่ชื่อ …/ training_images

2.2.6 ลำดับที่ (12) คือ Tuning เป็นขั้นตอนการปรับแต่งระบบทำได้โดยการปรับค่าตัวแปรชื่อ HOW_MANY_TRAINING_STEPS ในไฟล์ชื่อ retrain.py จากการตั้งค่าโดยปริยาย (Default value) เป็น 4000 ให้เปลี่ยนเป็น 10000 ค่านี้มีผลให้ผลลัพธ์ของค่าความถูกต้อง และค่าความแม่นยำสูงขึ้น จากนั้นบันทึกไฟล์ ขั้นตอนนี้จะทำเพียงครั้งเดียวและไฟล์สามารถใช้ได้ตลอดไปกรณีที่ไม่ต้องการปรับแต่งใด ๆ

2.2.7 ลำดับที่ (13) คือ Retraining เป็นขั้นตอนการฝึกฝนข้อมูลเพื่อใช้ในการรู้จำอะไหล่ ซึ่งทำการฝึกฝนข้อมูลโดยเริ่มจากเพิ่มชั้น Fully connected และ ชั้น Softmax ซึ่งจะนำข้อมูลคุณลักษณะของภาพในแฟ้มข้อมูลชื่อ …/ bottleneck_data มาเข้าโครงข่าย และใช้ค่าน้ำหนัก และค่าไบแอสของตัวแบบเดิมเป็นค่าตั้งต้น เพื่อทำการฝึกฝนข้อมูล

2.2.8 ลำดับที่ (14) คือ Trained model เป็นผลของการฝึกฝนข้อมูลจากกระบวนการก่อนหน้าเมื่อทำการฝึกฝนข้อมูลทั้งหมดจะได้ตัวแบบที่ใช้ในการตรวจจับอะไหล่ ตัวแบบที่จะนำไปใช้งานชื่อ retrained_graph.pb

2.3 เอาท์พุต (Output)

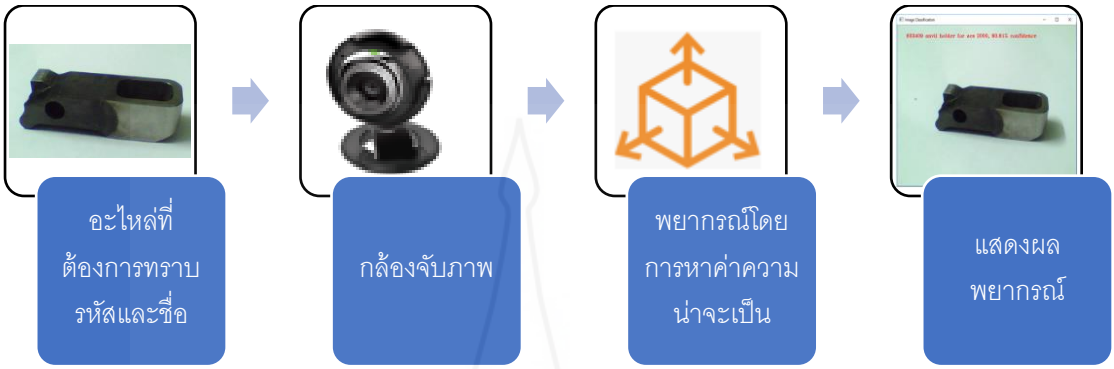
2.3.1 ลำดับที่ (15) คือ spare parts inference request เป็นส่วนที่ผู้ใช้งานจะนำอะไหล่ที่ไม่รู้จักหมายเลขหรือรหัสของอะไหล่ หรืออะไหล่ที่ต้องการนำมาพยากรณ์มาให้กล้องจับภาพ

2.3.2 ลำดับที่ (16) คือ Camera เป็นกล้องสำหรับจับหรือถ่ายภาพ เป็นการสร้างภาพจากอะไหล่จริงมาเป็นภาพที่ใช้กับคอมพิวเตอร์โดยการแปลงจากภาพจริงมาเป็นรหัสคอมพิวเตอร์เพื่อสามารถเก็บและอ่านได้จากคอมพิวเตอร์ระบบทำการเก็บภาพผ่านกล้องจับภาพ โดยผู้ใช้งานมีหน้าที่นำอะไหล่ที่ต้องการมาให้ระบบพยากรณ์หมายเลขหรือชื่อมาทำการให้กล้องจับภาพ

2.3.3 ลำดับที่ (17) คือ Trained Model เป็นตัวแบบที่ได้จากการฝึกฝนข้อมูลอะไหล่ชื่อ retrained_graph.pb โดยระบบจะเรียกใช้ตัวแบบเพื่อรับข้อมูลภาพที่นำเข้าจากกล้องจับภาพมาทำการประมวลผลเพื่อพยากรณ์ผลลัพธ์คือรหัสหรือชื่อของอะไหล่

2.3.4 ลำดับที่ (18) คือ Prediction เป็นการแสดงผลลัพธ์ของการพยากรณ์มาทางจอภาพ โดยการแสดงผลจะแสดงภาพของอะไหล่ที่ตรวจจับในขณะนั้น ระบบจะแสดงหมายเลข

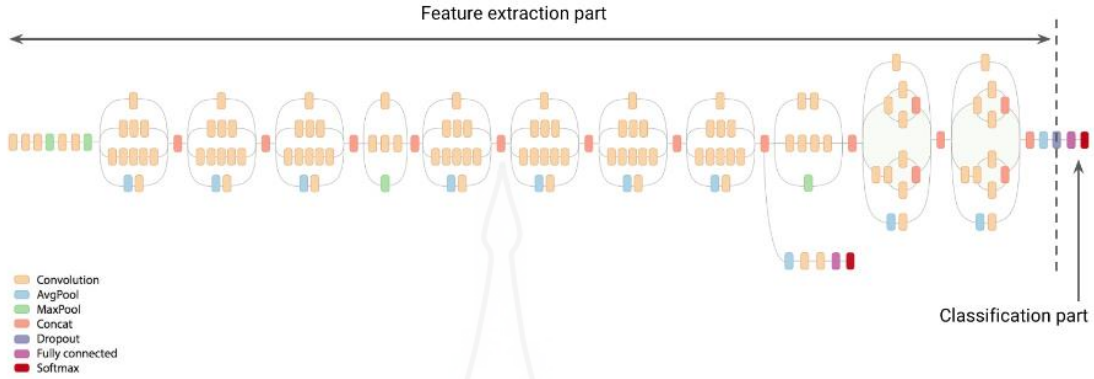
หรือรหัสของอะไหล่ ชื่อของอะไหล่พร้อมทั้งค่าเปอร์เซ็นต์ความเชื่อมั่นหรือค่าความน่าจะเป็นมาแสดง ในรูปแบบเรียลไทม์



ภาพที่ 3.2 แสดงตัวอย่างการทำงานของระบบในส่วนของผู้ใช้งาน

จากภาพที่ 3.2 เมื่อตัวแบบได้ทำการเรียนรู้ในส่วนของการฝึกฝนข้อมูลและนำมาใช้งานแล้ว ขั้นตอนนี้จะเป็นขั้นตอนการทำงานของระบบในส่วนผู้ใช้งาน โดยระบบจะเริ่มจากการจับภาพผ่านกล้องโดยผู้ใช้งานจะนำอะไหล่ที่ต้องการทราบรหัสและชื่อของอะไหล่และต้องการให้ระบบพยากรณ์ จากนั้นระบบทำการสกัดคุณลักษณะของภาพและทำการจำแนกประเภท ในขั้นตอนนี้จะเรียกตัวแบบที่ได้รับการฝึกฝนด้วยข้อมูลอะไหล่แล้ว นำข้อมูลที่ได้จากการสกัดคุณลักษณะทั้งหมดของมาทำการพยากรณ์ด้วยตัวแบบ โดยการหาค่าความน่าจะเป็นของภาพที่จับในเวลานั้น ระบบคำนวณหาค่าความน่าจะเป็นและเรียงลำดับจากข้อมูลทั้งหมด ค่าความน่าจะเป็นจะถูกเรียงลำดับจากค่าที่มากไปหาค่าที่น้อย และนำผลลัพธ์ที่ได้จากค่าความน่าจะเป็นสูงสุดมาทำการแสดงผล และกระบวนการนี้จะวนซ้ำไปเรื่อย ๆ จนกว่าจะสั่งให้โปรแกรมจบการทำงาน โดยการแสดงผลนั้นจะแสดงผลภาพของอะไหล่ที่กำลังจับภาพอยู่ หมายเลขและชื่ออะไหล่ที่โปรแกรมคำนวณได้ และค่าความน่าจะเป็นหรือค่าความเชื่อมั่นที่เป็นอะไหล่นั้นแสดงผลเป็นค่าเปอร์เซ็นต์ ขั้นตอนแต่ละขั้นตอนจะทำงานอย่างรวดเร็วขึ้นอยู่กับประสิทธิภาพของเครื่องคอมพิวเตอร์ กรณีเครื่องคอมพิวเตอร์ประสิทธิภาพต่ำอาจทำให้การแสดงผลภาพมีลักษณะภาพไม่ต่อเนื่อง กระตุก ซึ่งเป็นจังหวะของการทำงานของระบบการตรวจจับภาพ

การเรียกใช้ตัวแบบ Inception V3 ซึ่งเป็นตัวแบบของโครงข่ายประสาทเทียบแบบคอนโวลูชันดังแสดงในภาพที่ 3.3

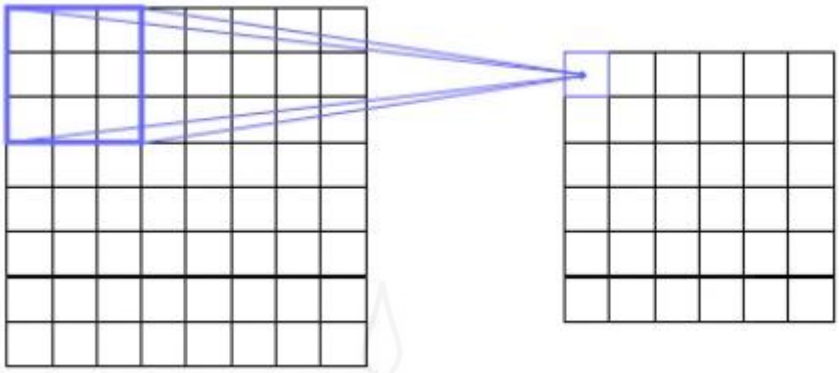


ภาพที่ 3.3 โครงสร้างของตัวแบบ Inception V3

ที่มา: https://hackathonprojects.files.wordpress.com/2016/09/74911-image03.png

จากภาพที่ 3.3 โครงสร้างของตัวแบบ Inception V3 เมื่อต้องการสร้างตัวแบบที่เป็นชุดข้อมูลที่ต้องการจำแนกประเภทหรือรู้จำทำได้โดยทำการฝึกฝนซ้ำในส่วนของการจำแนกประเภทด้วยชุดข้อมูลที่ต้องการจำแนกประเภทหรือส่วนการรู้จำ และใช้ส่วนสกัดคุณลักษณะเดิมของตัวแบบด้วยพารามิเตอร์เดิมที่ได้รับการฝึกฝนด้วยชุดข้อมูลจำนวนมากก่อนหน้าแล้ว ซึ่งประกอบไปด้วยชั้นต่าง ๆ ดังนี้

Convolution เป็นการหาคุณลักษณะเด่นของภาพโดยการกระทำกันระหว่างตัวฟิลเตอร์กับรูปภาพเพื่อทำการสกัดคุณลักษณะเอง และทำซ้ำไปเรื่อย ๆ จนกว่าจะได้คุณลักษณะที่มีความสัมพันธ์กับผลลัพธ์มากที่สุด ซึ่งการหาคุณลักษณะเด่นของภาพนั้นจะทำโดยการดอทเมทริกซ์กับตัวกรองที่มีน้ำหนัก (weight) เลื่อนผ่านจุดสีของภาพโดยเลื่อนจากด้านซ้ายบนไปด้านขวาบนทีละขั้นขึ้นอยู่กับขนาดตัวกรองจนครบทุกจุดสีภายในภาพ ดังภาพ 3.4 ซึ่งกระบวนการนี้จะอยู่ในขั้นตอนของการสกัดคุณลักษณะของภาพในงานวิจัย

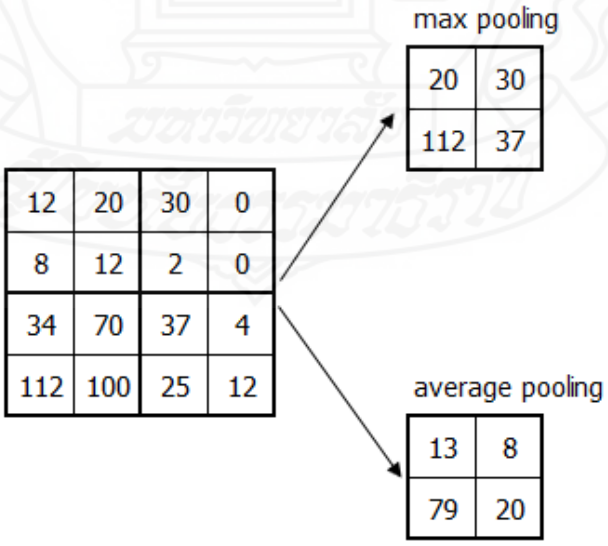


ภาพที่ 3.4 แสดงการทำงานของชั้นคอนโวลูชัน

ที่มา: http://socs.binus.ac.id/2017/02/27/convolutional-neural-network/

AvgPool (average pooling) ทำหน้าที่ลดขนาดให้เล็กลงโดยที่รายละเอียดของข้อมูลนำเข้ายังเท่าเดิมโดยการหาค่าเฉลี่ยของค่าจากการดอทเมทริกซ์กับตัวกรองเพื่อสร้างผังภาพ (feature map) ใหม่ซึ่งกระบวนการนี้จะอยู่ในขั้นตอนของการสกัดคุณลักษณะของภาพในงานวิจัย

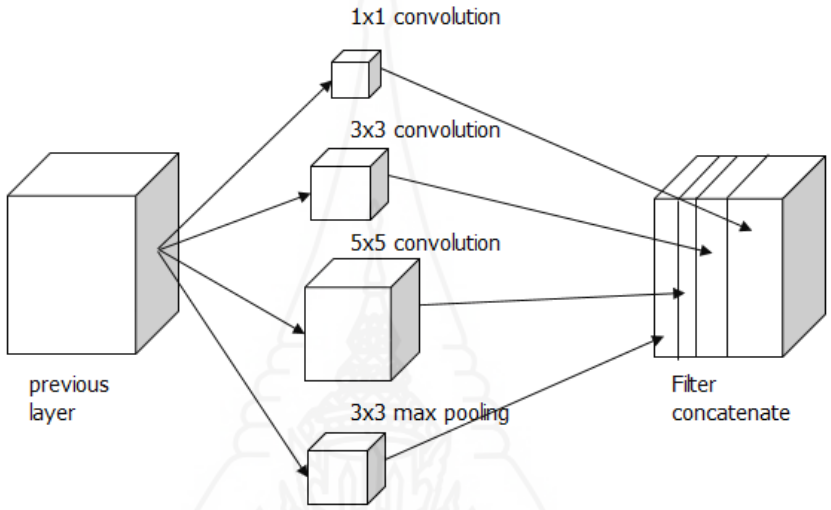
MaxPool (max pooling) ทำหน้าที่ลดขนาดให้เล็กลงโดยที่รายละเอียดของข้อมูลนำเข้ายังเท่าเดิมโดยการหาค่าสูงสุดของค่าที่ได้จากการทำดอทเมทริกซ์กับตัวกรองเพื่อสร้างแผนภาพใหม่ซึ่งกระบวนการนี้จะอยู่ในขั้นตอนของการสกัดคุณลักษณะของภาพในงานวิจัย



ภาพที่ 3.5 แสดงการทำงานของชั้น AvgPool (Average pooling) และ MaxPool (Max Pooling)

จากภาพที่ 3.5 เป็นการแสดงความแตกต่างของการทำงานของชั้น AvgPool และ ชั้น MaxPool โดยการทำงานของชั้น AvgPool เป็นการลดขนาดของข้อมูลโดยการหาค่าเฉลี่ยของข้อมูลชั้นก่อนหน้า ส่วนชั้น MaxPool เป็นการลดขนาดของข้อมูลโดยการหาค่าสูงสุดของชั้นก่อนหน้า

Concat (concatenate) เป็นการรวมผังภาพที่ได้จากการคำนวณชั้นก่อนหน้า เพื่อจัดรูปแบบให้เป็นข้อมูลนำเข้าของชั้นถัดไป การทำงานของกระบวนการนี้ ดังภาพที่ 3.6

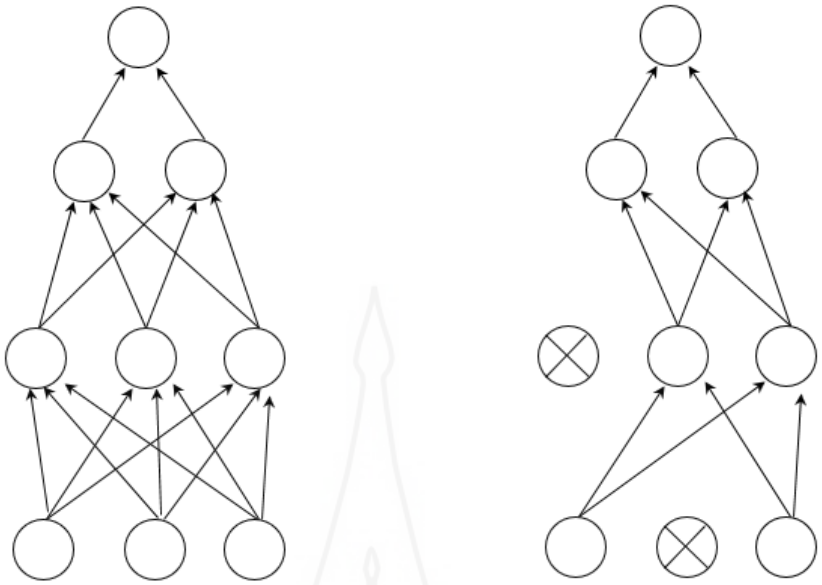


ภาพที่ 3.6 แสดงการทำงานของชั้น Concat (concatenate)

ที่มา: https://www.slideshare.net/ckmarkohchang/applied-deep-learning-1103-convolutional-neural-networks

ในกระบวนการ Concat จะอยู่ในขั้นตอนของการสกัดคุณลักษณะของภาพในงานวิจัย

Dropout เป็นการเพิ่มประสิทธิภาพในการฝึกฝนโดยการป้องกันการเกิด overfitting หรือปัญหาที่เครื่องทำงานได้ดีกับชุดข้อมูลตัวอย่างที่ใช้ในการฝึกฝนแต่จะทำงานได้ไม่ดีกับชุดข้อมูลอื่นที่ไม่ใช่ชุดข้อมูลฝึกฝน เมื่อมีข้อมูลการฝึกฝนในโครงข่ายประสาทเทียมจะทำการจัดโครงสร้างในโครงข่ายประสาทให้เป็นระเบียบลดการเรียนรู้ระหว่างเซลล์ประสาทและปรับค่าน้ำหนักให้เหมาะสม โดยทำการละการคำนวณของโหนดบางโหนดออกไปแบบสุ่มในแต่ละรอบของการฝึกฝนทำให้ค่าน้ำหนักที่เชื่อมกับโหนดเหล่านั้นไม่ถูกปรับในรอบการคำนวณนั้น ดังภาพที่ 3.7 ซึ่งกระบวนการนี้จะอยู่ในขั้นตอนของการฝึกฝนข้อมูลภาพในงานวิจัย



Standard Neural Net After apply dropout

ภาพที่ 3.7 แสดงการเปรียบเทียบการทำงานของโครงข่ายก่อนและหลังการใส่ชั้น Dropout

ที่มา: https://mohitatgithub.github.io/2018-03-28-MNIST-Image-Classification-with-CNN-&-Keras/

Fully connected ในชั้นนี้เป็นการเชื่อมต่อของชั้นที่มีการลดขนาด ชั้นการรวมผังภาพของก่อนหน้านี้ทั้งหมดเพื่อใช้ในการคำนวณหาผลลัพธ์ ซึ่งกระบวนการนี้จะอยู่ในขั้นตอนของการฝึกฝนข้อมูลภาพในงานวิจัย

Softmax จะเป็นชั้นสุดท้ายที่จะคำนวณหาค่าความน่าจะเป็นของเป้าหมายทั้งหมดออกมาหรือเป็นผลการจำแนก กล่าวคือเป็นการจำแนกประเภทแบบหลายคลาส (Multiclass classifier) ซึ่งใช้ความน่าจะเป็นเพื่อแสดงผลลัพธ์ และถูกใช้เป็นฟังก์ชันกระตุ้น (Activation function) ของโครงข่ายประสาทเทียมที่หาระยะห่างระหว่างค่าที่คำนวณได้จริงกับค่าเป้าหมายซึ่งกระบวนการนี้จะอยู่ในขั้นตอนของการฝึกฝนข้อมูลภาพในงานวิจัย และหาได้จากสมการ [1] (ปริญญา สงวนสัตย์, 2560) ดังนี้

$$P(c_i|x) = \frac{P(x|c_i)P(c_i)}{\sum_{j=1}^{K} P(x|c_j)P(c_j)} = \frac{e^{o_i}}{\sum_{j=1}^{K} e^{o_j}}$$

โดยกำหนดให้

$P(c_i)$ คือ ค่าความน่าจะเป็นที่เกิดขึ้นกับคลาส i

$P(x|c_i)$ คือ ค่าความน่าจะเป็นที่จะเป็นตัวอย่าง x เมื่อเป็นคลาสที่ i

K คือ จำนวนของคลาสทั้งหมด

o_i คือ ค่าป้อนออกของโหนดที่ i

e คือ ฟังก์ชันเลขชี้กำลังหรือฟังก์ชันเอกซ์โพเนนเชียล (exponential function)

เริ่มจากการใช้ชั้นสกัดคุณลักษณะของตัวแบบมาทำการสกัดคุณลักษณะของอะไหล่ที่ได้จากขั้นตอนกระบวนการนำเข้า ทำการบันทึกเป็นไฟล์และจัดเก็บอยู่ในแฟ้มข้อมูลโดยแยกตามหมายเลขอะไหล่ จากนั้นทำการฝึกฝนข้อโดยการเรียกใช้พารามิเตอร์ที่ได้รับการฝึกฝนมาแล้วจากตัวแบบ Inception V3 มาทำการฝึกฝนข้อมูลอะไหล่ ผลลัพธ์ที่ได้จากกระบวนการนี้คือตัวแบบที่จะนำไปพยากรณ์อะไหล่ที่ไม่รู้จัก กระบวนการสุดท้ายคือผลลัพธ์ที่ได้จากการประมวลผลนั้นคือการใช้ตัวแบบที่ได้จากการฝึกฝนมาแล้วนำมาใช้งาน โดยเริ่มจากนำอะไหล่ที่ไม่รู้จัก ให้กล้องจับภาพและส่งให้ตัวแบบพยากรณ์ผลลัพธ์ โดยจะแสดงความน่าจะเป็นของอะไหล่ชิ้นนั้นและแสดงหมายเลข ชื่ออะไหล่พร้อมทั้ง แสดงเปอร์เซ็นต์ค่าความเชื่อมั่นของอะไหล่ชิ้นนั้นด้วย

3. ข้อมูลและกลุ่มตัวอย่างที่ใช้ในงานวิจัย

ข้อมูลและกลุ่มตัวอย่างที่ใช้ในงานวิจัยนี้เป็นรูปภาพของอะไหล่สำหรับเครื่องจักรและอุปกรณ์ช่วยในการผลิตโดยจะแบ่งกลุ่มอะไหล่ออกเป็น 5 กลุ่มหลัก กลุ่มละ 5 ชิ้น รวมทั้งหมด 25 ชิ้น ซึ่งเป็นตัวแทนของอะไหล่ทั้งหมดของบริษัทโดยพิจารณาให้ครอบคลุมถึงชนิดของอะไหล่จากรูปทรง วัสดุที่ใช้ทำ และสีของอะไหล่ เพื่อต้องการนำมาทดสอบความแม่นยำของระบบ ซึ่งภาพจะถูกบันทึกไว้หลากหลายมุมมองบนภาพพื้นหลังเป็นสีขาว ซึ่งรายละเอียดของกลุ่มอะไหล่สามารถอธิบายได้ดังนี้

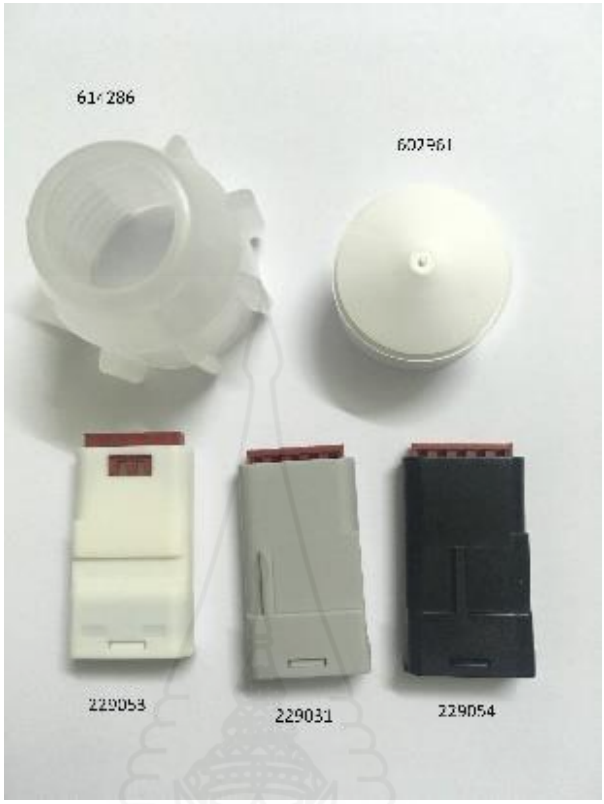
1. ภาพอะไหล่ที่มีขนาดเล็ก เป็นอะไหล่ที่มีขนาดความยาวไม่เกิน 50 มิลลิเมตรและมีรูปร่างเป็นทรงกระบอกหรือคล้ายทรงกระบอกที่เส้นผ่านศูนย์กลางของอะไหล่ไม่เกิน 10 มิลลิเมตร ดังภาพที่ 3.8

ภาพที่ 3.8 แสดงตัวอย่างของข้อมูลภาพอะไหล่ที่มีขนาดเล็ก

จากภาพที่ 3.8 แสดงภาพของอะไหล่พร้อมรหัสและชื่อของอะไหล่ที่มีขนาดเล็กดังนี้

1. 606174 SODERMATE SOLDER TIP
2. 601356 DISPENSING NEEDLE(BERNSTEIN)
3. 601482 NEEDLE, SYRINGE, GREEN COLOUR
4. 602964 NOZZLE DISPENSING TIP
5. 602102 PROBE, D3.00, H9.00-ROUGH

2. ภาพอะไหล่ที่เป็นพลาสติก เป็นอะไหล่ที่ผลิตมาจากวัสดุที่เป็นพลาสติกซึ่งมีหลากหลายรูปทรง ถ้าเป็นทรงกระบอกให้เส้นผ่านศูนย์กลางมากกว่า 10 มิลลิเมตร ดังภาพที่ 3.9

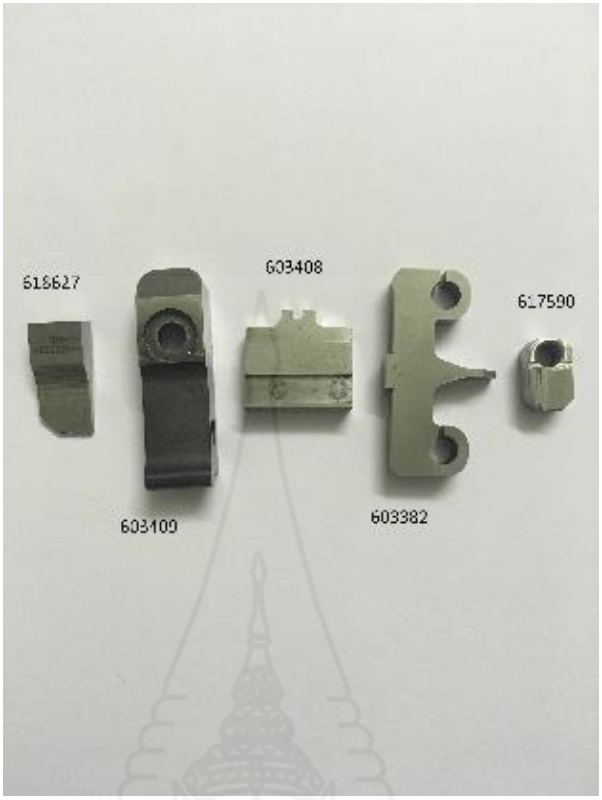


ภาพที่ 3.9 แสดงตัวอย่างของข้อมูลภาพอะไหล่ที่เป็นพลาสติก

จากภาพที่ 3.9 แสดงภาพของอะไหล่พร้อมรหัสและชื่อของอะไหล่ที่เป็นพลาสติกดังนี้

1. 614286 PLASTIC CONNECTING NUT
2. 602961 PISTION 30CC/55CC FLOW PISTON
3. 229053 CONNECTOR JST SRVPB-B04-S
4. 229031 CONNECTOR JST SRVPB-A05-H
5. 229054 CONNECTOR JST SRVPB-D02-K

3. ภาพอะไหล่ที่เป็นชิ้นโลหะ เป็นอะไหล่ที่ผลิตมากจากโลหะหรือวัสดุอื่นใดที่คล้ายโลหะ มีรูปทรงเป็นแท่งสี่เหลี่ยมหรือคล้ายสี่เหลี่ยม ที่มีด้านกว้าง ด้านยาว และความสูง ดังภาพที่ 3.10



ภาพที่ 3.10 แสดงตัวอย่างของข้อมูลภาพอะไหล่ที่เป็นแท่งโลหะ

จากภาพที่ 3.10 แสดงภาพของอะไหล่พร้อมรหัสและชื่อของอะไหล่ที่เป็นแท่งโลหะ ดังนี้

1. 618627 ANVIL
2. 603408 Clincer 4mm-92 for ACS2000
3. 603409 Anvil holder for ACS 2000
4. 603382 ANVIL # 99127 1357
5. 617590 ANVIL 2017

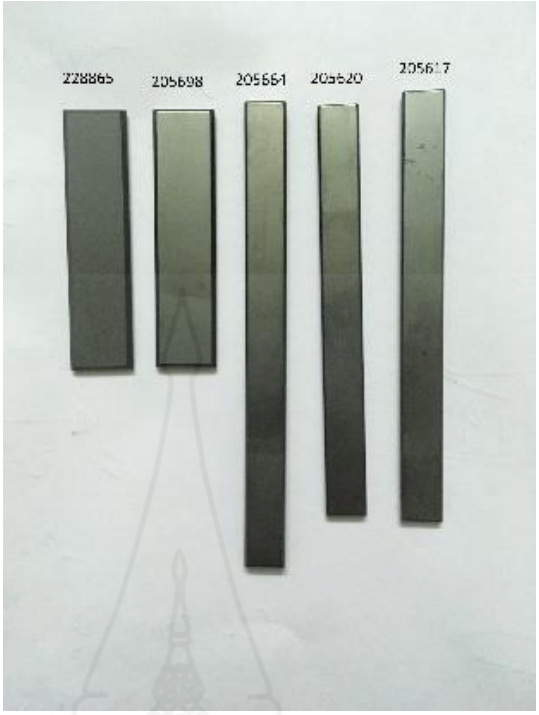
4. ภาพอะไหล่ที่เป็นแผ่นโลหะเป็นอะไหล่ที่ทำมาจากโลหะหรือวัสดุอื่นใดที่คล้ายโลหะมีรูปทรงเป็นแผ่น โดยมีความหนาไม่เกิน 5 มิลลิเมตร ดังภาพที่ 3.11

ภาพที่ 3.11 แสดงตัวอย่างของข้อมูลภาพอะไหล่ที่เป็นแผ่นโลหะ

จากภาพที่ 3.11 แสดงภาพของอะไหล่พร้อมรหัสและชื่อของอะไหล่ที่เป็นแผ่นโลหะ ดังนี้

1. 603383 Driver blade, 4mm-922 for ACS2000
2. 603387 Spring,short,for ACS2000
3. 603380 WIRE CRIMPER
4. 618609 INSULATION CRIMPER
5. 618628 WIRE CRIMPER

5. ภาพอะไหล่ที่เป็นแท่งเฟอร์ไรท์ เป็นอะไหล่ที่ทำมาจากส่วนผสมที่มีคาร์บอนเป็นส่วนประกอบและอัดเป็นแท่ง ดังภาพที่ 3.12



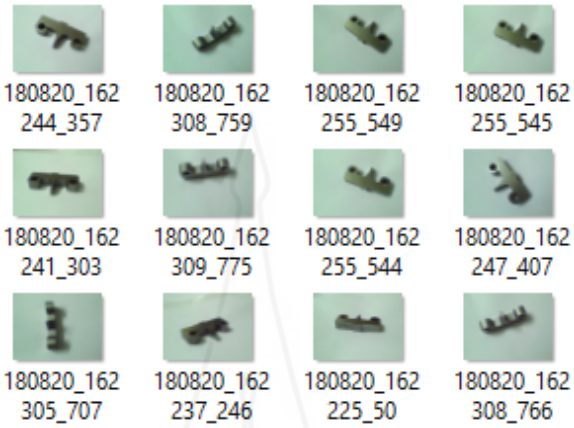
ภาพที่ 3.12 แสดงตัวอย่างของข้อมูลภาพอะไหล่ที่เป็นแท่งเฟอร์ไรท์

จากภาพที่ 3.12 แสดงภาพของอะไหล่พร้อมรหัสและชื่อของอะไหล่ที่เป็นแท่งเฟอร์ไรท์ดังนี้

1. 228865 CORE I MNZN 50.2MMX12.1MMX2.6MM
2. 205698 FERRITE BAR 2,5X 12X 50
3. 205664 FERRITE BAR 90X8X4 UNCOATED
4. 205620 FERRITE BAR 3X 8X 80 UNCOATED
5. 205617 FERRITE BAR 3X 8X 83 UNCOATED

การรวบรวมข้อมูลโดยจะรวบรวมข้อมูลรูปภาพอะไหล่ในลักษณะ มุมมอง และทิศทางต่าง ๆ กัน เพื่อให้ภาพมีความหลากหลายเพิ่มประสิทธิภาพในการรู้จำ บนพื้นหลัง (Background) สีขาว เพื่อขจัดสิ่งที่ไม่เกี่ยวข้องกับอะไหล่ออกไปจากภาพ และให้ระยะห่างจากกล้องถึงตัวอะไหล่ประมาณ 5 – 15 เซนติเมตร โดยเป็นการจำลองสถานการณ์เมื่อนำระบบไปใช้งานจริง และบันทึกไว้ในฮาร์ดดิสก์เพื่อใช้เป็นข้อมูลนำเข้า และทำการตรวจสอบคุณภาพของภาพและเลือกภาพที่ดีที่สุดที่มีอะไหล่อยู่ภายในภาพเพื่อใช้ในการฝึกฝน จากจำนวนอะไหล่ทั้งหมดทำให้ได้จำนวนตัวอย่างหรือ

จำนวนรูปภาพรวมทั้งหมด 12,500 ภาพ เพื่อใช้ในการพัฒนาและทดสอบระบบแต่ละภาพมีความละเอียด 640 x 480 พิกเซลโดยได้จากคุณสมบัติของกล้องที่ใช้ในการทดลอง

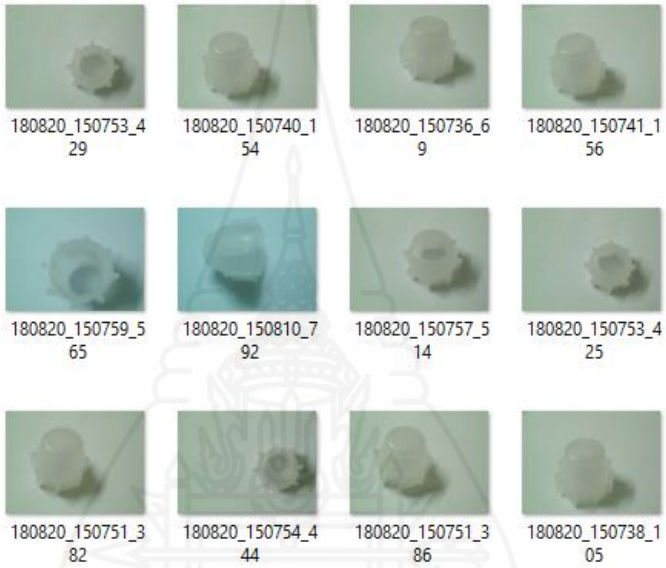


ภาพที่ 3.13 แสดงตัวอย่างของข้อมูลอะไหล่แท่งโลหะ

ดังภาพที่ 3.13 แสดงตัวอย่างภาพอะไหล่แท่งโลหะ ในภาพคือหมายเลขอะไหล่ 603382 ชื่ออะไหล่ ANVIL # 99127 1357 ที่ถูกจับภาพด้วยมุมมองภาพต่าง ๆ ทั้งหมด 700 ภาพ และคัดเลือกหรือเข้าสู่กระบวนการทำความสะอาดข้อมูลจนได้ภาพที่จะใช้สำหรับฝึกฝนข้อมูลจำนวน 500 ภาพ อะไหล่แท่งโลหะประกอบด้วยอะไหล่จำนวน 5 ชิ้น ดังนั้นภาพที่ใช้สำหรับฝึกฝนข้อมูลอะไหล่แท่งโลหะรวมทั้งหมด 2,500 ภาพ โดยแยกออกเป็นแฟ้มข้อมูลแต่ละหมายเลขของอะไหล่แต่ละชิ้น

ภาพที่ 3.14 แสดงตัวอย่างของข้อมูลอะไหล่ขนาดเล็ก

จากภาพที่ 3.14 แสดงตัวอย่างภาพอะไหล่ที่มีขนาดเล็ก ในภาพคือหมายเลขอะไหล่ 606174 ชื่ออะไหล่ SODERMATE SOLDER TIP ที่ถูกจับภาพด้วยมุมมองภาพต่าง ๆ ทั้งหมด 700 ภาพ และคัดเลือกหรือเข้าสู่กระบวนการทำความสะอาดข้อมูลจนได้ภาพที่จะใช้สำหรับฝึกฝนข้อมูลจำนวน 500 ภาพ อะไหล่ที่มีขนาดเล็กประกอบด้วยอะไหล่จำนวน 5 ชิ้น ดังนั้นภาพที่ใช้สำหรับฝึกฝนข้อมูลอะไหล่ขนาดเล็กรวมทั้งหมด 2,500 ภาพ โดยแยกออกเป็นแฟ้มข้อมูลแต่ละหมายเลขของอะไหล่แต่ละชิ้น



ภาพที่ 3.15 แสดงตัวอย่างของข้อมูลอะไหล่พลาสติก

จากภาพที่ 3.15 แสดงตัวอย่างภาพอะไหล่พลาสติก ในภาพคือหมายเลขอะไหล่ 614286 ชื่ออะไหล่ PLASTIC CONNECTING NUT ที่ถูกจับภาพด้วยมุมมองภาพต่าง ๆ ทั้งหมด 700 ภาพ และคัดเลือกหรือเข้าสู่กระบวนการทำความสะอาดข้อมูลจนได้ภาพที่จะใช้สำหรับฝึกฝนข้อมูลจำนวน 500 ภาพ อะไหล่พลาสติก ประกอบด้วยอะไหล่จำนวน 5 ชิ้น ดังนั้นภาพที่ใช้สำหรับฝึกฝนข้อมูลอะไหล่พลาสติก รวมทั้งหมด 2,500 ภาพ โดยแยกออกเป็นแฟ้มข้อมูลแต่ละหมายเลขของอะไหล่แต่ละชิ้น

ภาพที่ 3.16 แสดงตัวอย่างของข้อมูลอะไหล่แผ่นโลหะ

จากภาพที่ 3.16 แสดงตัวอย่างภาพอะไหล่แผ่นโลหะ ในภาพคือหมายเลขอะไหล่ 603380 ชื่ออะไหล่ WIRE CRIMPER ที่ถูกจับภาพด้วยมุมมองภาพต่าง ๆ ทั้งหมด 700 ภาพ และคัดเลือกหรือเข้าสู่กระบวนการทำความสะอาดข้อมูลจนได้ภาพที่จะใช้สำหรับฝึกฝนข้อมูลจำนวน 500 ภาพ อะไหล่แผ่นโลหะประกอบด้วยอะไหล่จำนวน 5 ชิ้น ดังนั้นภาพที่ใช้สำหรับฝึกฝนข้อมูลอะไหล่แผ่นโลหะรวมทั้งหมด 2,500 ภาพ โดยแยกออกเป็นแฟ้มข้อมูลแต่ละหมายเลขของอะไหล่แต่ละชิ้น

ภาพที่ 3.17 แสดงตัวอย่างของข้อมูลอะไหล่แท่งเฟอร์ไรท์

จากภาพที่ 3.17 แสดงตัวอย่างภาพอะไหล่แท่งเฟอร์ไรท์ ในภาพคือหมายเลขอะไหล่ 205698 ชื่ออะไหล่ FERRITE BAR 2,5X 12X 50 ที่ถูกจับภาพด้วยมุมมองภาพต่าง ๆ ทั้งหมด 700 ภาพ และคัดเลือกหรือเข้าสู่กระบวนการทำความสะอาดข้อมูลจนได้ภาพที่จะใช้สำหรับฝึกฝนข้อมูลจำนวน 500 ภาพ อะไหล่แท่งเฟอร์ไรท์ประกอบด้วยอะไหล่จำนวน 5 ชิ้น ดังนั้นภาพที่ใช้สำหรับฝึกฝนข้อมูลอะไหล่แท่งเฟอร์ไรท์รวมทั้งหมด 2,500 ภาพ โดยแยกออกเป็นแฟ้มข้อมูลแต่ละหมายเลขของอะไหล่แต่ละชิ้น

4. การพัฒนาระบบรู้จำอะไหล่

การพัฒนาระบบจดจำภาพสำหรับการตรวจจับอะไหล่โดยใช้โครงข่ายประสาทเทียมแบบคอนโวลูชัน จากขั้นตอนการออกแบบและวิเคราะห์ระบบ ทำให้ทราบขั้นตอนการทำงานของระบบ เนื่องจากระบบนี้เกี่ยวข้องกับผู้ใช้เฉพาะแผนก ดังนั้นผู้วิจัยจึงพัฒนาเป็นโปรแกรมระบบซึ่งติดตั้งบนเครื่องคอมพิวเตอร์ของแผนกที่เกี่ยวข้องเท่านั้น โดยพัฒนาด้วยภาษาไพธอนซึ่งเป็นภาษาที่สนับสนุนการทำการเรียนรู้ของเครื่อง ใช้ไลบรารี่ของ OpenCV เพื่อใช้ในการจัดรับภาพและบันทึกผ่านกล้อง ซึ่งเปรียบเสมือนดวงตาของระบบ ใช้ TensorFlow เฟรมเวิร์คซึ่งสนับสนุนการทำการเรียนรู้เชิงถ่ายโอนข้อมูล และใช้ตัวแบบโครงข่ายประสาทเทียมแบบคอนโวลูชัน Inception V3 เพื่อใช้ค่าน้ำหนักและค่าไบแอสของตัวแบบต้นทางนำมาใช้กับตัวแบบปลายทางหรือตัวแบบที่ต้องการจะพัฒนาให้ระบบรู้จำอะไหล่ ซึ่งข้อมูลรูปภาพและไฟล์ที่เกี่ยวข้องจะถูกบันทึกไว้ในฮาร์ดดิสก์ของเครื่องที่ติดตั้งทั้งหมด

OpenCV หรือ Open Source Computer Vision Library ซึ่งเป็นซอฟต์แวร์คอมพิวเตอร์วิทัศน์แบบโอเพนซอร์สและซอฟต์แวร์ของการเรียนรู้ของเครื่อง (machine learning) OpenCV ถูกสร้างขึ้นมาให้มีโครงสร้างพื้นฐานที่ทำงานร่วมกันสำหรับแอปพลิเคชันการมองเห็นและการรับรู้ของเครื่องคอมพิวเตอร์ และถูกสร้างมาให้ง่ายต่อการใช้งานและการแก้ไขดัดแปลงโค้ดของโปรแกรม ไลบรารี่ของ OpenCV มีอัลกอริทึมที่มีการปรับปรุงมากกว่า 2,500 รายการ ซึ่งรายการดังกล่าวครอบคลุมถึงการทำงานของคอมพิวเตอร์วิทัศน์ (Computer vision) ที่มีทั้งอัลกอริทึมแบบดั้งเดิมและแบบที่ถูกปรับปรุงให้ทันสมัย อัลกอริทึมเหล่านี้มีสามารถใช้ในการตรวจจับและจดจำใบหน้าของคน ระบุชนิดวัตถุ แยกประเภทการกระทำของมนุษย์ในวีดีโอ สามารถที่จะติดตามการเคลื่อนไหวของวัตถุ ค้นหาภาพที่คล้ายกันจากฐานข้อมูล ลบตาแดงจากภาพที่ถูกถ่ายด้วยกล้องที่ใช้แฟลช จดจำทิวทัศน์และสร้างเครื่องหมายเพื่อซ้อนทับกับภาพได้ OpenCV มีผู้ใช้เป็นจำนวนมาก ไลบรารีมีการใช้อย่างกว้างขวางทั้งในส่วนของบริษัทเอกชน กลุ่มนักวิจัยและหน่วยงานของรัฐ

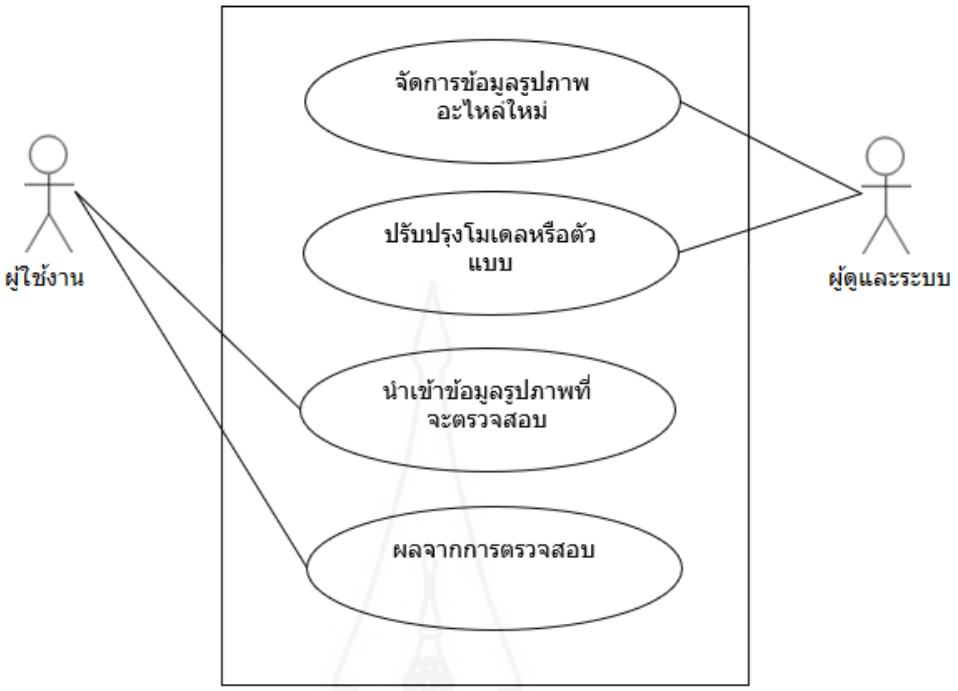
นอกเหนือจากบริษัทที่มีชื่อเสียงเช่น Google, Yahoo, Microsoft, Intel, IBM, Sony, Honda, Toyota ที่มีการใช้ไลบรารีตัวนี้แล้ว ยังมีกลุ่มสตาร์ทอัพ (Startups) ที่มีการใช้อย่างมากมายเช่น VideoSerf การใช้งานของ OpenCV มีการใช้งานหลากหลากครอบคลุมไปถึงการผสานรูปภาพของ Street View ใช้ในการเฝ้าระวังเพื่อตรวจจับการบุกรุกในอิสราเอล การตรวจสอบอุปกรณ์ขุดในประเทศจีน ช่วยนำทางและหยิบจับวัตถุในหุ่นยนต์ที่ Willow Garage ซึ่งเป็นบริษัทที่วิจัยหุ่นยนต์ในสหรัฐอเมริกา การตรวจจับอุบัติเหตุจากการจมน้ำในสระในประเทศยุโรป ใช้งานเกี่ยวกับเรื่องศิลปะในสเปนและนิวยอร์ก การตรวจสอบรันเวย์ในตุรกี การตรวจสอบฉลากบนผลิตภัณฑ์ในโรงงานทั่วโลก และการตรวจจับใบหน้าในประเทศญี่ปุ่น เป็นต้น OpenCV มีการเชื่อมต่อกับโปรแกรมภาษา C++, Python, Java และ MATLAB อีกทั้งยังสนับสนุนการทำงานบน Windows, Linux, Android และ Mac OS ซึ่งไลบรารีของ OpenCV มักมุ่งเน้นการพัฒนาแอปพลิเคชันเกี่ยวกับการมองเห็นแบบเรียลไทม์

TensorFlow เป็นไลบรารีซอฟต์แวร์แบบโอเพนซอร์ส สำหรับการคำนวณตัวเลขที่มีประสิทธิภาพสูง โครงสร้างหรือสถาปัตยกรรมพัฒนาให้มีความยืดหยุ่นในการทำงาน ทำให้การนำมาปรับใช้ในเรื่องการคำนวณได้อย่างง่าย บนแพลตฟอร์มที่หลากหลาย (CPU, GPUs, TPUs) และนำไปใช้ได้ตั้งแต่เครื่องคอมพิวเตอร์แบบตั้งโต๊ะไปจนถึงการทำงานบนเซิร์ฟเวอร์ รวมถึงสามารถพัฒนาใช้งานกับอุปกรณ์มือถือ โดยการพัฒนาจากนักวิจัยและทีมวิศวกรจากทีม Google Brain ของกูเกิล ซึ่งความสามารถของ TensorFlow มาพร้อมกับการสนับสนุนการทำเกี่ยวกับการเรียนรู้ของเครื่องและการเรียนรู้เชิงลึก ซึ่งในการที่จะพัฒนาหรือทำระบบเกี่ยวกับการเรียนรู้ของเครื่องนั้นเป็นเรื่องที่ยุ่งยากซับซ้อนและยากที่จะเขียนโค้ดหรือโปรแกรม เพราะมีกระบวนการที่ซับซ้อน TensorFlow จึงเป็นไลบรารีที่มีความสามารถลดความซับซ้อนดังกล่าวทำให้การพัฒนาเป็นเรื่องที่ง่ายและรวดเร็ว TensorFlow สามารถเขียนและบูรณาการได้ในหลากหลายโปรแกรมภาษาเช่น Python JavaScript, C++, Java, Go, Swift แต่ที่ใช้ได้ง่ายสุดคือภาษา Python และสามารถทำงานได้ทั้งระบบปฏิบัติการวินโดว์ ลีนุกซ์ และแอนดรอยด์ และยังมีการพัฒนา TensorFlow Lite ที่ทางกูเกิลได้พัฒนาเพื่อให้ใช้งานกับระบบปฏิบัติการแอนดรอยด์ เพื่อให้ไฟล์มีขนาดเล็กลงสามารถรันได้เร็วในอุปกรณ์มือถือ TensorFlow ถูกนำไปใช้ในการค้นหาดาวเคราะห์ดวงใหม่ การช่วยสนับการวินิจฉัยของแพทย์ในการตรวจโรคเบาหวานที่ประสาทตา การแจ้งเตือนให้เจ้าหน้าที่ทราบเกี่ยวกับการตัดไม้ทำลายป่าที่ผิดกฎหมาย เป็นต้น

ภาษาไพธอน (Python Language) เป็นโปรแกรมภาษาที่ถูกออกแบบและพัฒนามาเพื่อให้ง่ายต่อการเขียน เพราะลักษณะการเขียนจะสั้น กะทัดรัด แต่มีประสิทธิภาพสูง สามารถเรียนรู้ได้ง่าย รวดเร็ว ภาษาไพธอนเป็นภาษาที่มีการพัฒนาจากนักพัฒนาโปรแกรมทั่วโลกจึงส่งผล

ทำให้โปรแกรมภาษาดังกล่าวมีความสามารถสูงและสามารถรองรับงานต่าง ๆ อย่างหลากหลาย เช่น งานด้านวิศวกรรม เกมส์และแอปพลิเคชัน งานด้านวิทยาศาสตร์ งานด้านระบบฐานข้อมูล งานด้านเครือข่ายคอมพิวเตอร์ และเว็บแอปพลิเคชัน รวมถึงใช้ในการเรียนการสอนในต่างประเทศตั้งแต่ระดับมัธยมศึกษาจนถึงระดับมหาวิทยาลัย เช่น NASA, กูเกิล, หน่วยงานของสหรัฐ, ไอบีเอ็ม และอื่น ๆ ส่วนประเทศไทยกำลังเพิ่มมากขึ้นเรื่อย ๆ เช่น มหาวิทยาลัยมหิดล เกษตรศาสตร์ จุฬาลงกรณ์ และพระจอมเกล้าเจ้าคุณทหารลาดกระบัง เป็นต้น ภาษาไพธอนถูกออกแบบโดยการนำลักษณะเด่นของโปรแกรมภาษาอื่น ๆ มาเป็นพื้นฐานในการพัฒนาต่อยอด เช่น ภาษา C, C++, Java, Matlab เป็นต้น ภาษาไพธอนถูกเรียกว่าเป็น ภาษาที่มีหลายกระบวนทัศน์ (สุชาติ คุ้มมะณี, 2561) หรือ หลายมุมมอง (multi-paradigm languages) ซึ่งเกิดจากการผสมผสานรวมเอาแนวความคิดในการ พัฒนาซอฟต์แวร์แบบต่าง ๆ เข้าไว้ด้วยกันให้อยู่ในตัวของไพธอน คือ การเขียนโปรแกรมเชิงวัตถุ (Object-oriented programming), การเขียนโปรแกรมเชิงโครงสร้าง (Structured programming), การโปรแกรมเชิงฟังก์ชัน (Functional programming) และการเขียนโปรแกรมเชิงลักษณะ (Aspect-oriented programming) ซึ่งทาง IEEE Spectrum ได้ออกมาเผยถึงผลการจัดอันดับ Programming Language สำหรับปี 2018 (Stephen Cass, 2018) ซึ่งได้จัดอันดับของความนิยมของภาษาเขียนโปรแกรมด้วยกันทั้งหมด 47 ภาษา ครอบคลุมการใช้งานทั้งสำหรับ Web, Mobile, Enterprise และ Embedded โดย 10 อันดับแรกโดยเรียงจากค่าความนิยมจากมากไปหาน้อยที่ได้รับความนิยมสูงสุดคือ Python, C++, Java, C, C#, PHP, R, JavaScript, Go, และ Assembly ตามลำดับ

ระบบการจดจำภาพสำหรับการตรวจจับอะไหล่ได้แบ่งการทำงานออกเป็น 2 ส่วนหลัก คือผู้ดูแลระบบและผู้ใช้งานซึ่งสามารถอธิบายรายละเอียดดังแสดงในรูปยูสเคสไดอะแกรม (วฤษาย์ ร่มสายหยุด, 2561) ซึ่งประกอบด้วยผู้ใช้ระบบ (actor) ดังนี้ คือ ผู้ดูแลระบบ (admin) และผู้ใช้งาน (user) ดังแสดงในภาพที่ 3.18



ภาพที่ 3.18 ยูสเคสไดอะแกรมของระบบจดจำภาพสำหรับการตรวจจับอะไหล่

1. ส่วนผู้ดูแลระบบ (admin) มีหน้าที่ในการจัดการกับข้อมูลรูปภาพอะไหล่ คือทำหน้าที่เพิ่มรูปภาพอะไหล่เมื่อมีอะไหล่หมายเลขใหม่เข้ามาโดยการทำการรันโปรแกรมเพื่อถ่ายภาพอะไหล่ในมุมมองต่าง ๆ จากนั้นทำการคัดเลือกรูปภาพที่มีรายละเอียดของอะไหล่อยู่ในรูปภาพอย่างน้อย 60 เปอร์เซ็นต์ของอะไหล่และมีสิ่งที่ไม่เกี่ยวข้องกับอะไหล่น้อยที่สุด จากการทดลองจะใช้พื้นหลังเป็นสีขาวเพื่อขจัดส่วนที่ไม่เกี่ยวข้องออกเพราะจะมีผลต่อการรู้จำระบบ ผู้ดูแลระบบทำหน้าที่ลบรูปภาพอะไหล่ในกรณีที่อะไหล่นั้นเลิกใช้งานไปแล้ว และทำหน้าที่ปรับปรุงโมเดลหรือตัวแบบคือทำฝึกฝนตัวแบบเมื่อมีการเปลี่ยนแปลงเกี่ยวกับข้อมูลอะไหล่ เช่นการลบ หรือการเพิ่มอะไหล่เข้ามาในระบบ และผู้ดูแลระบบจะมีหน้าที่นำตัวแบบที่ได้ทำการปรับปรุงแล้วนำไปใช้งานเพื่อให้ระบบในส่วนของผู้ใช้งานมีข้อมูลที่ทันสมัยตามที่ได้ฝึกฝนข้อมูลไว้

2. ส่วนของผู้ใช้งาน (user) มีหน้าที่ใช้งานระบบโดยการนำอะไหล่ที่ไม่รู้จักหมายเลขหรือชื่อมาทำการตรวจสอบโดยส่งผ่านข้อมูลไปยังตัวแบบที่ได้ฝึกฝนไว้แล้วซึ่งในตัวแบบจะทำการสกัดคุณลักษณะและทำการหาผลลัพธ์จากนั้นเปรียบเทียบกับค่าเป้าหมาย และแสดงผลลัพธ์ออกมาเป็นหมายเลขและชื่อของอะไหล่พร้อมทั้งค่าความเชื่อมั่นของความน่าจะเป็นผ่านทางจอภาพจากนั้นผู้ใช้งานจะใช้ข้อมูลผลลัพธ์ที่ได้จากการพยากรณ์นำไปเป็นรายละเอียดในการเบิกจ่ายอะไหล่ต่อไป

5. การทดสอบประสิทธิภาพของระบบ

การประเมินผลและการทดสอบเพื่อประเมินประสิทธิภาพการจำแนกข้อมูลในงานวิจัยนี้ จะใช้การวัด ค่าความแม่นยำ (Precision) เป็นเกณฑ์การวัดค่าความถูกต้องในการจำแนกประเภท มีมุมมองในการพิจารณาว่าหากตัวแบบทำนายประเภทเป็น A แล้วมีกี่เปอร์เซ็นต์ที่ทำนายถูกต้อง ค่าเรียกคืน (Recall) เป็นเกณฑ์ที่บอกว่าอัตราส่วนของโปรแกรมที่ทำนายได้ถูกต้องกับของที่ทำนายได้ถูกต้องทั้งหมด ค่าความถูกต้อง (Accuracy) เป็นเกณฑ์การวัดค่าความถูกต้องโดยรวมในการจำแนกประเภทเพื่อพิจารณาการทำนายของตัวแบบว่าถูกต้องมากน้อยเพียงใด โดยนับจำนวนที่ทำนายถูกทั้งหมดในทุกประเภทเทียบกับจำนวนตัวอย่างทั้งหมด และการวัดประสิทธิภาพโดยรวม (F-measure) เป็นเกณฑ์การวัดค่าความถูกต้องในการจำแนกประเภทที่ใช้ค่าเฉลี่ยระหว่าง True positive rate หรือค่า Recall และค่า Precision โดยทุกค่าจะพิจารณาค่าความถูกต้องจากตารางคอนฟิวชันเมทริกซ์ (Confusion Matrix)

คอนฟิวชันเมทริกซ์ (Confusion Matrix) (ธนภัทร์ คุ้มสุภา, 2559) คือเมทริกซ์ที่แสดงผลของการจำแนกโดยแสดงรายละเอียดแบ่งตามคลาส ดังตารางที่ 3.5

ตารางที่ 3.5 แสดงคอนฟิวชันเมทริกซ์ของการจำแนกแบบ 3 คลาส

		คลาสที่ทำนายได้		
		A	B	C
คลาสจริง	A	$M_{1,1}$ (TP_A)	$M_{1,2}$	$M_{1,3}$
	B	$M_{2,1}$	$M_{2,2}$ (TP_B)	$M_{2,3}$
	C	$M_{3,1}$	$M_{3,2}$	$M_{3,3}$ (TP_C)

ตารางที่ 3.5 เป็นการจำแนกข้อมูลแบบหลายคลาสที่ใช้ในการประเมินผลของการทดสอบระบบรู้จำอะไหล่โดยใช้โครงข่ายประสาทเทียมแบบคอนโวลูชัน ได้ประยุกต์ใช้ตารางคอนฟิวชันเมทริกซ์แบบ 3 คลาส มาประเมินผลการทดสอบประสิทธิภาพการจำแนกข้อมูลของอะไหล่ทั้งหมด 25 ชิ้น หรือ 25 คลาส การประยุกต์ใช้จะอ้างถึงคลาสที่ทำนายได้เทียบกับคลาสจริง โดยกำหนดให้ c เป็นจำนวนคลาสทั้งหมด ซึ่งงานวิจัยนี้คลาสทั้งหมดจะเท่ากับ 25 คลาส ค่าในแต่ละแถวจะแสดงถึงจำนวนข้อมูลที่อยู่ในคลาสนั้นจริง ๆ ส่วนค่าในแต่ละสดมภ์จะหมายถึงจำนวน

ข้อมูลที่ทำนายได้คลาสนั้น ตัวอย่าง M1,1 หมายถึงจำนวนข้อมูลที่อยู่ในคลาสที่ทำนายได้สดมภ์ที่ 1 กับคลาสจริงในแถวที่ 1 นั่นคือข้อมูลของคลาสที่ทำนายได้คลาส A กับข้อมูลคลาสจริงของคลาส A ค่าที่ได้จะเป็นค่าของจำนวนข้อมูลที่ทำนายได้คลาส A และผลลัพธ์คือคลาส A (True Positive) หรือค่า TP_A

โดยกำหนดให้

M คือ จำนวนข้อมูลที่อยู่ในตำแหน่งของคลาสที่ทำนายได้กับคลาสจริง

c คือ จำนวนคลาสทั้งหมด

i คือ ตำแหน่งข้อมูลคลาสที่ทำนายได้

j คือ ตำแหน่งข้อมูลคลาสจริง

ดังสมการที่ (1) ถึง (3)

- TP คือจำนวนข้อมูลที่ทำนายได้คลาสนั้นและผลลัพธ์คือคลาสนั้น (True Positive)

$$TP_i = M_{i,j} \quad (j = i) \qquad (1)$$

- FP คือจำนวนที่ทำนายได้ในคลาสนั้นแต่ผลลัพธ์คือคลาสอื่น (False Positive)

$$FP_i = \sum_{j=1}^{c} M_{j,i} \quad (\text{ยกเว้น } j = i) \qquad (2)$$

- FN คือจำนวนข้อมูลที่ทำนายได้คลาสอื่นแต่ผลลัพธ์คือคลาสนั้น (False Negative)

$$FN_i = \sum_{j=1}^{c} M_{i,j} \quad (\text{ยกเว้น } j = i) \qquad (3)$$

ตัววัดประสิทธิภาพแยกตามคลาส การคำนวณหา Precision, Recall และ F-measure สามารถคำนวณได้จากสมการที่ (4) ถึง (6) ตามลำดับ

$$Pr_i = \frac{TP_i}{TP_i + FP_i} \qquad (4)$$

$$Re_i = \frac{TP_i}{TP_i + FN_i} \qquad (5)$$

$$F1_i = \frac{2\,x\,TP_i\,xRe_i}{Pr_i + Re_i} \quad (6)$$

ตัววัดประสิทธิภาพโดยรวม เป็นการคำนวณประสิทธิภาพของการจำแนกโดยรวมจะใช้ค่าเฉลี่ยของตัววัดประสิทธิภาพในแต่ละคลาสมาคำนวณดังสมการที่ (7) ถึง (9) ตามลำดับ

$$Pr = \frac{\sum_{i=1}^{c} Pr_i}{c} \quad (7)$$

$$Re = \frac{\sum_{i=1}^{c} Re_i}{c} \quad (8)$$

$$F1 = \frac{\sum_{i=1}^{c} F1_i}{c} \quad (9)$$

และในการหาค่าความแม่นยำ (Accuracy) กำหนดให้ N คือจำนวนข้อมูลทั้งหมด ดังสมการที่ (10)

$$Accuracy = \frac{\sum_{i=1}^{c} TP_i}{N} \quad (10)$$

บทที่ 4
ผลการการดำเนินงาน

ผู้วิจัยได้แบ่งผลการดำเนินงานออกเป็น 4 ตอน ได้แก่ ตอนที่ 1 การเก็บและบันทึกข้อมูลรูปภาพเพื่อใช้ฝึกฝนตัวแบบ ตอนที่ 2 ส่วนที่ใช้ในการสร้างตัวแบบ ตอนที่ 3 ส่วนที่ใช้ในการแสดงผลลัพธ์ และตอนที่ 4 การประเมินประสิทธิภาพของระบบ ดังต่อไปนี้

ตอนที่ 1 การเก็บและบันทึกข้อมูลรูปภาพเพื่อใช้ฝึกฝนตัวแบบ

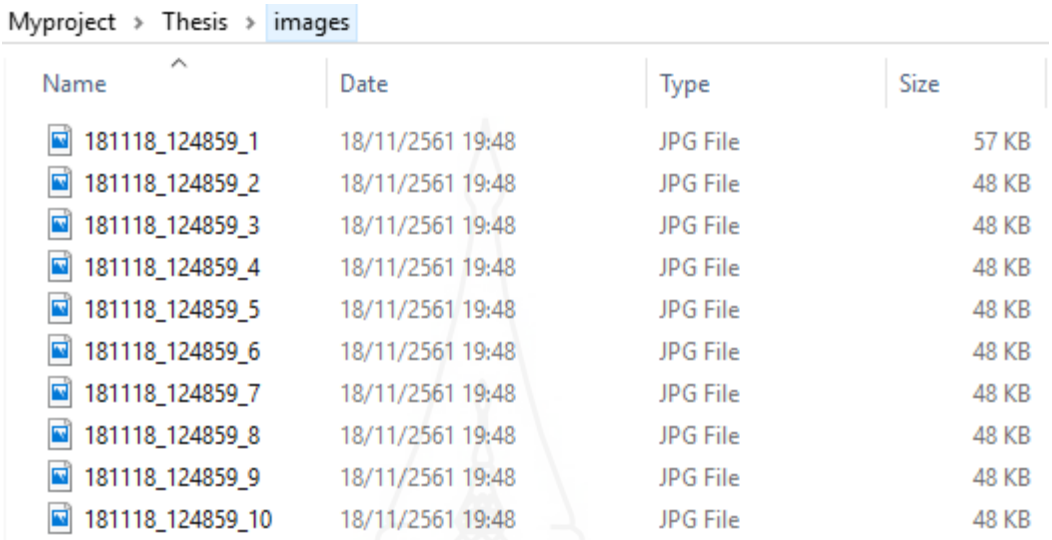
ส่วนการเก็บและบันทึกข้อมูลรูปภาพเพื่อใช้สำหรับฝึกฝนตัวแบบ

1. นำอะไหล่ที่ต้องการให้ระบบรู้จำมาเตรียมเพื่อทำการเก็บภาพโดย การเก็บภาพเพื่อเป็นข้อมูลนำเข้าในการฝึกฝนข้อมูลนั้นได้จำลองจากสถานการณ์การใช้งานจริงที่แผนกสโตร์ โดยการติดตั้งอุปกรณ์ใช้งานจริงจะกำหนดให้กล้องอยู่ห่างจากอะไหล่ที่ระยะระหว่าง 5 – 15 เซนติเมตร บนพื้นหลัง (Background) สีขาว ดังภาพที่ 4.1

ภาพที่ 4.1 แสดงขั้นตอนการเตรียมเก็บและบันทึกภาพอะไหล่

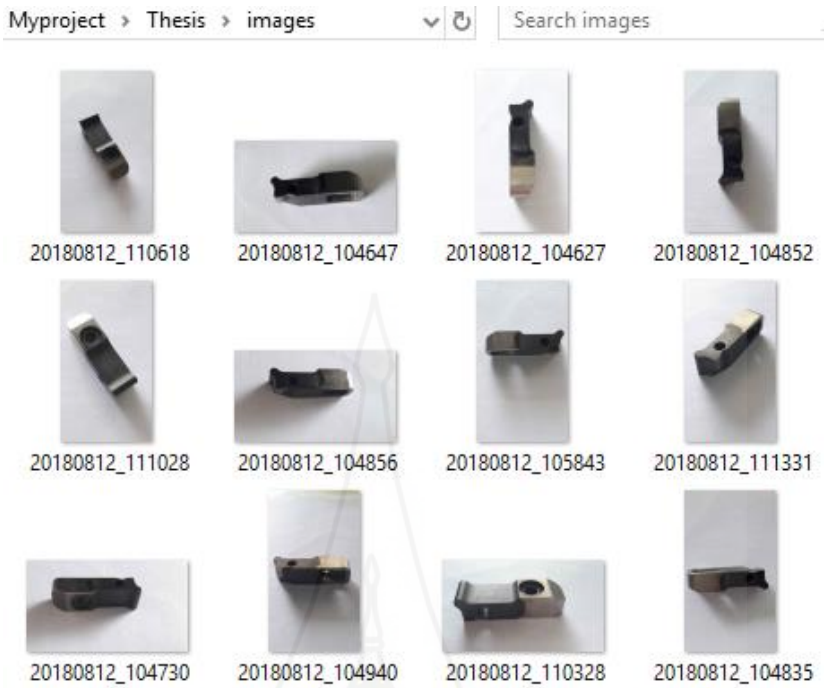
2. ทำการรันไฟล์ Create_image.py เพื่อให้โปรแกรมทำงาน โดยโปรแกรมจะทำการการเปิดกล้องเพื่อทำการจับภาพ ในขณะที่โปรแกรมจับภาพให้ทำการหมุนอะไหล่ หรือหมุนกล้องเพื่อให้จับภาพได้หลากหลายลักษณะหลากหลายมุมมอง และกำหนดชื่อไฟล์โดยไม่ให้ชื่อไฟล์ซ้ำกันเพราะจะมีผลทำให้โปรแกรมเกิดข้อผิดพลาดในการบันทึกข้อมูลเข้าเครื่องคอมพิวเตอร์ ผู้วิจัยได้กำหนดเป็นการต่อตัวเลขของปีเดือนวันชั่วโมงนาทีวินาทีและตัวแปรที่เพิ่มค่า โปรแกรมทำการ

บันทึกลงในฮาร์ดดิสก์ ในแฟ้มชื่อ .../image โดยโปรแกรมจะจับภาพจำนวน 700 ภาพจากนั้นหยุดการทำงาน ไฟล์ที่ถูกสร้างและเก็บไว้ในแฟ้มข้อมูลดังภาพที่ 4.2



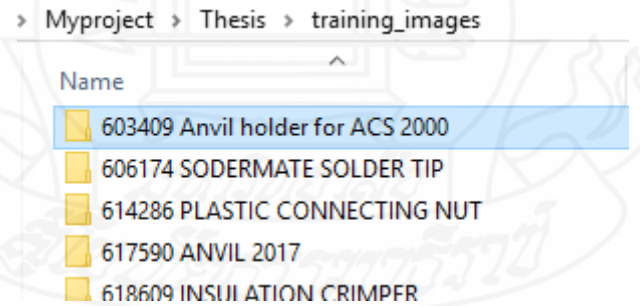
ภาพที่ 4.2 แสดงอะไหล่ที่ถูกเก็บไว้ในแฟ้มข้อมูล

3. เปิดแฟ้มข้อมูลชื่อ .../image เพื่อตรวจสอบและทำความสะอาดไฟล์ภาพที่ได้โดยเลือกภาพที่มีความชัดเจนและมีภาพอะไหล่อยู่ในรูปภาพจำนวน 500 ภาพ จากนั้นทำการคัดลอกไฟล์ภาพทั้งหมดไว้ กรณีที่รูปภาพคัดเลือกแล้วมีคุณภาพไม่พอหรือไม่ได้จำนวน 500 ภาพเนื่องจากมีข้อผิดพลาดระหว่างการจับภาพและบันทึกภาพหรือมีส่วนที่ไม่เกี่ยวข้องกับภาพที่กำลังพิจารณามากภายในภาพ สามารถที่จะสั่งโปรแกรมเพื่อทำการจับภาพและบันทึกภาพเพิ่มเติม หรือสามารถที่จะลบข้อมูลไฟล์ภาพทั้งหมดเพื่อทำการจับภาพและบันทึกใหม่ได้เช่นกัน ตัวอย่างภาพที่ทำความสะอาดและถูกคัดเลือกเพื่อใช้เป็นข้อมูลนำเข้าในกระบวนการฝึกฝน ดังภาพที่ 4.3



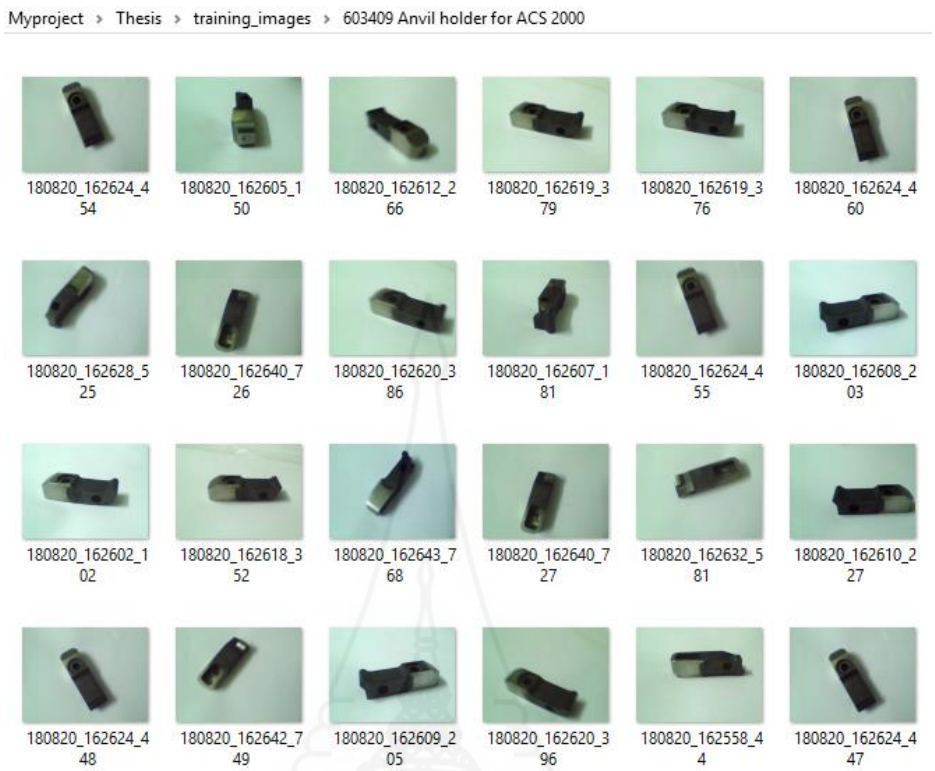
ภาพที่ 4.3 แสดงภาพอะไหล่ที่ถูกเก็บไว้ในแฟ้มข้อมูล

4. ทำการสร้างแฟ้มข้อมูลย่อยในแฟ้มข้อมูลหลักที่ชื่อ …/training_images โดยตั้งชื่อเลขหกหลักเป็นหมายเลขหรือรหัสอะไหล่และตามด้วยชื่อของอะไหล่ ดังภาพที่ 4.4



ภาพที่ 4.4 แสดงภาพการตั้งชื่อแฟ้มข้อมูล

5. ทำการวางไฟล์ที่ได้ทำการคัดลอกไว้ในขั้นตอนที่ 3 ไว้ในแฟ้มข้อมูลที่ได้ทำการสร้างไว้ในขั้นตอนที่ 4 จากนั้นทำซ้ำตั้งแต่ข้อที่ 1 – 5 สำหรับอะไหล่ชิ้นถัดไป จนครบหมดทุกชิ้นเพื่อเป็นข้อมูลนำเข้าในการฝึกฝนข้อมูล ดังภาพที่ 4.5



ภาพที่ 4.5 แสดงภาพอะไหล่ที่อยู่ในแฟ้มที่เป็นข้อมูลนำเข้าเพื่อฝึกฝน

ตอนที่ 2 ส่วนที่ใช้ในการสร้างตัวแบบ (Model)

เป็นส่วนที่ใช้ในการสร้างตัวแบบ (Model) โดยการนำภาพที่ได้ทำการจัดเก็บไว้และทำความสะอาดข้อมูลโดยการเลือกภาพที่มีคุณภาพมาใช้สำหรับฝึกฝนตัวแบบ

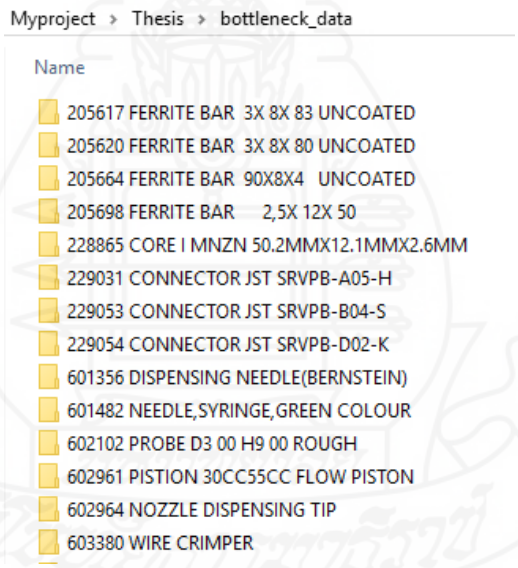
1. เปิดไฟล์ชื่อ retrain.py จากนั้นทำการแก้ไขค่าของตัวแปรโดยการเปลี่ยนค่าของตัวแปรชื่อ HOW_MANY_TRAINING_STEPS ซึ่งเป็นตัวแปรที่ใช้ในการทดลองเพื่อแสดงความแตกต่างของผลทดลองมาเปรียบเทียบค่าความถูกต้องและแม่นยำในการพยากรณ์ จากนั้นจะใช้ค่าที่ให้ผลลัพธ์ดีที่สุดมาใช้ในการทำงานจริง โดยเริ่มต้นจากการทดลองค่าตั้งต้นเป็น 4000 และบันทึกไฟล์ จากนั้นทำการทดลองและบันทึกผล จากนั้นให้เปลี่ยนค่าเป็น 10000 และ 20000 ตามลำดับ ทั้งนี้การปรับค่าตัวแปรดังกล่าวเพื่อเพิ่มจำนวนรอบของการฝึกฝนให้มากขึ้น เป็นการเพิ่มค่าความถูกต้องและค่าความแม่นยำในการพยากรณ์ให้สูงขึ้น

2. ผู้ดูแลระบบทำการรันไฟล์ retrain.py โปรแกรมจะเริ่มจากการเรียกใช้ตัวแบบ Inception v3 ที่ได้รับการฝึกฝนมาแล้ว การเรียกใช้ตัวแบบดังกล่าวจะทำการสกัดคุณลักษณะของภาพโดยใช้ค่าน้ำหนักที่ได้จากการฝึกฝนจากข้อมูลก่อนหน้ามาใช้ในการฝึกฝนข้อมูลอะไหล่

3. โปรแกรมจะทำการสร้างรายการของรูปภาพในแฟ้มข้อมูลที่ชื่อ …/training_images ซึ่งเป็นข้อมูลที่ได้จากการเตรียมเพื่อเป็นข้อมูลนำเข้าในการฝึกฝน โดยในแฟ้มข้อมูลดังกล่าวจะประกอบด้วยแฟ้มข้อมูลย่อยซึ่งบรรจุไฟล์ภาพของอะไหล่แต่ละชิ้น

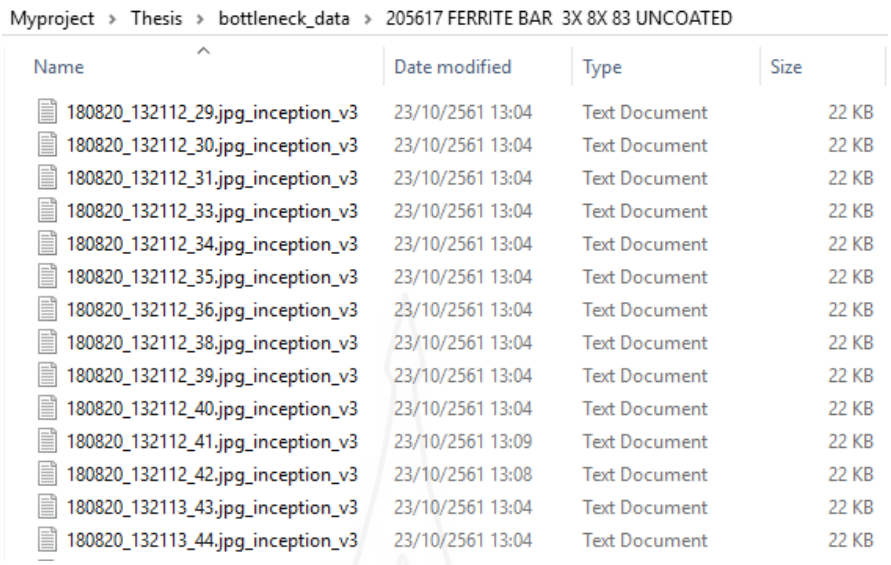
4. โปรแกรมจะแปลงรูปภาพให้อยู่ในรูปแบบที่สามารถทำงานร่วมกับ Tensorflow ได้ กล่าวคือเรียกใช้ข้อมูลรูปภาพที่ใช้ในการฝึกฝนที่ถูกเตรียมไว้ในขั้นตอนก่อนหน้านี้เพื่อมาแปลงให้อยู่ในรูปแบบที่จะใช้ทำงานร่วมกับ Tensorflow

5. วิเคราะห์และคำนวณแต่ละรูปภาพให้อยู่ในรูปของ image feature vector มาเก็บไว้ในไฟล์ Text และบันทึกไว้ในแฟ้มข้อมูล …/bottleneck_data ซึ่ง bottleneck_data เป็นแฟ้มข้อมูลที่ใช้เก็บข้อมูลของชั้น (layer) ก่อนหน้าจะถึงชั้นแสดงผลลัพธ์ (final output) เนื่องจากในระหว่างการฝึกฝนรูปที่มีจะถูกเรียกใช้ซ้ำ ๆ หลายครั้ง การคำนวณชั้นหลัง bottleneck_data แต่ละรูปจะใช้เวลามากและชั้นดังกล่าวไม่มีการเปลี่ยนแปลงจึงทำการบันทึกเป็นไฟล์ text ของแต่ละรูปไว้ก่อนฝึกฝนเปรียบเสมือน cache ดังภาพที่ 4.6 ภาพที่ 4.7 และภาพที่ 4.8



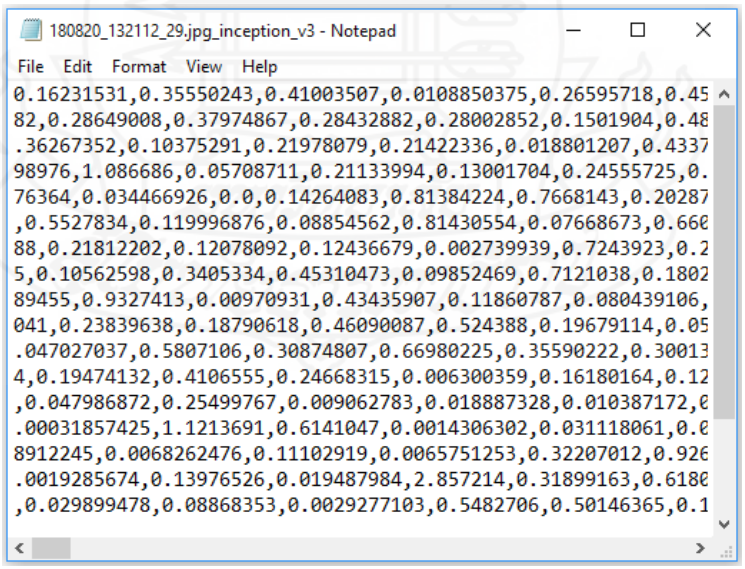
ภาพที่ 4.6 แสดงภาพที่โปรแกรมสร้างแฟ้มข้อมูลเพื่อเก็บไฟล์ image feature vector

จากภาพที่ 4.6 เป็นภาพที่แสดงแฟ้มข้อมูลของอะไหล่ที่ถูกสร้างโดยโปรแกรม โดยจะแยกเก็บไฟล์ที่ได้จากการสกัดคุณลักษณะของอะไหล่แต่ละชิ้น เพื่อเตรียมไว้ในขั้นตอนการฝึกฝนข้อมูลเพื่อทำการแยกประเภทหรือในส่วนของการรู้จำ



ภาพที่ 4.7 แสดงภาพที่โปรแกรมสร้างไฟล์ Text เพื่อเก็บข้อมูล image feature vector

จากภาพที่ 4.7 แสดงตัวอย่างผลลัพธ์ของอะไหล่ 205617 FERRITE BAR 3X 8X 83 UNCOATED ที่ได้จากโปรแกรมที่ทำงานโดยจะสร้างไฟล์ text ข้อมูลที่เก็บข้อมูล image feature vector หรือไฟล์คุณลักษณะของภาพที่ถูกสกัดมาเพื่อเตรียมเป็นข้อมูลนำเข้าในการฝึกฝนข้อมูลเพื่อจำแนกประเภทหรือส่วนการรู้จำ



ภาพที่ 4.8 แสดงภาพตัวอย่างข้อมูล image feature vector ซึ่งเก็บไว้ในไฟล์ Text

จากภาพที่ 4.8 แสดงภาพของตัวอย่างข้อมูลที่เก็บไว้ในไฟล์ text ซึ่งถูกเรียกว่า image feature vector เป็นข้อมูลที่ได้จากการสกัดคุณลักษณะและถูกแปลงให้อยู่ในรูปแบบของเวกเตอร์ เพื่อเป็นข้อมูลนำเข้าในการฝึกฝนข้อมูลในชั้นสุดท้ายสำหรับการจำแนกประเภทหรือส่วนของการรู้จำอะไหล่

7. เพิ่มชั้น fully-connected layer และ softmax ใหม่สำหรับการฝึกฝนข้อมูลอะไหล่ซึ่งเป็นชั้นสุดท้ายอยู่ในส่วนของการรู้จำ ในตัวแบบของ Inception V3 เป็นการฝึกฝนข้อมูลด้วยฐานข้อมูลขนาดใหญ่รวมถึงชั้นของ fully-connected layer และชั้นของ softmax ซึ่งใช้จำแนกประเภทข้อมูล 1,000 คลาส ดังนั้นจำเป็นต้องเพิ่มชั้นดังกล่าวใหม่เพื่อให้ระบบจำแนกประเภทตามจำนวนคลาสที่ต้องการให้ระบบรู้จำ ในงานวิจัยจะฝึกฝนข้อมูลอะไหล่เพื่อการทดลองจะจำแนกประเภทเป็นข้อมูลอะไหล่ 25 ชิ้น หรือ 25 คลาส

8. ฝึกฝนตัวแบบโดยทำการเลือกสุ่มจากไฟล์ที่อยู่ในแฟ้มข้อมูล …/bottleneck_data ป้อนเข้าสู่เน็ตเวิร์คเพื่อใช้ในการพยากรณ์ เป็นการฝึกฝนข้อมูลซึ่งอยู่ในชั้นของ fully-connected layer และ softmax โดยการใช้ค่าน้ำหนักและค่าไบแอสของตัวแบบเดิมเพื่อเป็นค่าตั้งต้น จากนั้นทำการฝึกฝนข้อมูลโดยการปรับค่าดังกล่าวเพื่อให้ได้ผลลัพธ์ใกล้เคียงกับค่าเป้าหมาย

9. เมื่อกระบวนการฝึกฝนสิ้นสุดลงจะบันทึกข้อมูลการฝึกฝนข้อมูลอะไหล่ไว้ในไฟล์ retrained_graph.pb และเป็นไฟล์ที่จะนำไปใช้งานเพื่อใช้ในการตรวจจับอะไหล่ในส่วนโปรแกรมของผู้ใช้งาน

ตอนที่ 3 ส่วนที่ใช้ในการแสดงผลลัพธ์

ส่วนที่ใช้ในการแสดงผลหรือส่วนที่ผู้ใช้นำไปใช้งานโดยการนำไฟล์ผลลัพธ์ที่ได้จากการฝึกฝนข้อมูลจะได้ตัวแบบเพื่อนำมาใช้งาน

1. ผู้ใช้งานนำอะไหล่ที่ไม่ทราบหมายเลขและชื่อมาแสดงที่หน้ากล้องเพื่อจับภาพ ผู้วิจัยได้นำอะไหล่หมายเลข 603409 มาให้โปรแกรมวิเคราะห์ ดังภาพที่ 4.9

ภาพที่ 4.9 แสดงภาพการเตรียมอะไหล่ที่ไม่รู้จักหมายเลขหรือรหัสและชื่อเพื่อทำนาย

2. ผู้ใช้งานทำการรันไฟล์ test – camera.py โปรแกรมจะทำการเปิดกล้องเพื่อรับภาพอะไหล่ที่ไม่รู้จักหมายเลขหรือรหัสและชื่ออะไหล่

3. โปรแกรมจะทำการเก็บรูปภาพจากกล้อง

4. แปลงรูปที่ได้ให้อยู่ในรูปแบบของ Tensorflow image

5. จากนั้นทำการรันเน็ตเวิร์คโดยเรียกไฟล์จากตัวแบบ retrained_graph.pb ที่ผ่านการฝึกฝนมาแล้วเพื่อทำนายหมายเลขและชื่ออะไหล่มาและเก็บข้อมูลไว้

6. เรียงลำดับข้อมูลที่คำนวณได้จากชั้น Softmax โดยเรียงจากค่าจากสูงไปหาต่ำและเก็บข้อมูลไว้

```
the object appears to be a 603409 anvil holder for acs 2000, 94.22% confidence
603409 anvil holder for acs 2000 (0.94221)
603382 anvil 99127 1357 (0.01445)
618627 anvil (0.01377)
603380 wire crimper (0.01155)
205698 ferrite bar 2 5x 12x 50 (0.00494)
614286 plastic connecting nut (0.00209)
617590 anvil 2017 (0.00207)
229054 connector jst srvpb d02 k (0.00191)
602964 nozzle dispensing tip (0.00151)
618609 insulation crimper (0.00127)
601356 dispensing needle bernstein (0.00074)
602102 probe d3 00 h9 00 rough (0.00071)
229031 connector jst srvpb a05 h (0.00068)
603383 driver blade 4mm 922 for acs2000 (0.00065)
601482 needle syringe green colour (0.00048)
618628 wire crimper (0.00040)
606174 sodermate solder tip (0.00014)
603408 clincer 4mm 92 for acs2000 (0.00012)
228865 core i mnzn 50 2mmx12 1mmx2 6mm (0.00010)
205620 ferrite bar 3x 8x 80 uncoated (0.00005)
602961 pistion 30cc55cc flow piston (0.00005)
229053 connector jst srvpb b04  (0.00005)
603387 spring short for acs2000 (0.00004)
205664 ferrite bar 90x8x4 uncoated (0.00002)
205617 ferrite bar 3x 8x 83 uncoated (0.00001)
```

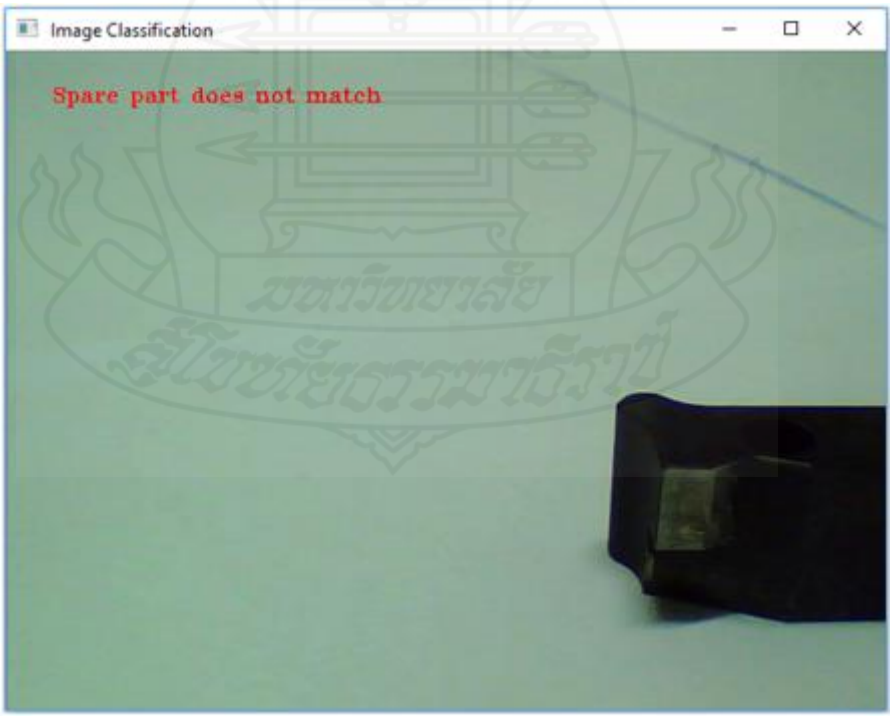
ภาพที่ 4.10 แสดงการเรียงลำดับข้อมูลที่คำนวณได้จากชั้น Softmax

จากภาพที่ 4.10 เป็นภาพแสดงผลของการตำนวณที่ได้จากชั้น Softmax และถูกเรียงลำดับจากค่าความเชื่อมั่นค่าสูงไปหาค่าต่ำ โดยตัวเลข 6 หลักแสดงถึงรหัสของอะไหล่ และตามด้วยชื่อหรือรายละเอียดของอะไหล่ ในส่วนที่อยู่ในวงเล็บจะเป็นค่าที่คำนวณได้จากชั้น Softmax ดังนั้นจากภาพจะเห็นว่าอะไหล่หมายเลข 603409 อะไหล่ชื่อ anvil holder for acs 2000 เป็นอะไหล่ที่คำนวณได้ค่าที่สูงที่สุดของคลาสทั้งหมดคือ 94.22 เปอร์เซ็นต์ ระบบจึงพยากรณ์ว่าภาพอะไหล่ที่นำมาทดสอบนั้นเป็นอะไหล่รหัส 603409 และนำค่าที่ได้จากการคำนวณของรหัสอะไหล่นี้เก็บเพื่อเป็นข้อมูลนำเข้าเพื่อเตรียมนำไปแสดงผลในกระบวนการถัดไป

7. เก็บค่าของหมายเลขและชื่อของอะไหล่โดยอ้างอิงจากข้อมูลที่ทำนายมาได้

8. แปลงค่าที่ได้จากการคำนวณในชั้น Softmax ทำให้เป็นค่าเปอร์เซ็นต์ความเชื่อมั่นซึ่งเป็นการนำค่าสูงสุดจากการพยากรณ์ได้มาทำการแปลงค่าให้อยู่ในรูปของเปอร์เซ็นต์

9. แทรกข้อความที่ได้จากข้อ 8 โดยก่อนทำการแทรกทำการปรับขนาด และสีให้เหมาะสมไว้ในภาพที่แสดงผ่านทางจอแสดงผล ในงานวิจัยชิ้นนี้จะใช้อักษรสีแดงและปรับขนาดตัวอักษรให้เหมาะสมเพื่อแสดงผลลัพธ์ของการพยากรณ์ทางจอแสดงผล แต่ก่อนจะส่งค่าไปยังขั้นตอนต่อไปจะมีการกำหนดเงื่อนไขก่อนโดยกำหนดให้ถ้าค่าที่คำนวณได้ต่ำกว่า 60 เปอร์เซ็นต์ของค่าสูงสุดที่คำนวณได้จะแสดงข้อความ “Spare part does not match”

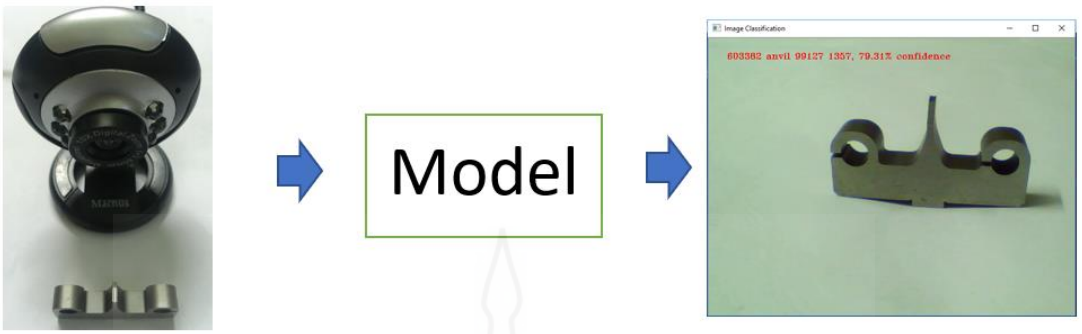


ภาพที่ 4.11 แสดงผลลัพธ์การกำหนดเงื่อนไขก่อนการแสดงผลการพยากรณ์

จากภาพที่ 4.11 แสดงผลลัพธ์ของการกำหนดเงื่อนไขก่อนการแสดงผลของการพยากรณ์ ขั้นตอนนี้จะกำหนดเมื่อนำตัวแบบไปใช้งานจริงเพื่อเพิ่มความมั่นใจในการพยากรณ์ของระบบ โดยค่าที่กำหนดเป็นเงื่อนไขสามารถที่จะปรับเปลี่ยนได้แล้วแต่งานที่จะนำไปใช้ว่าต้องการความถูกต้องแม่นยำมากน้อยเพียงใด ในงานวิจัยนี้กำหนดเงื่อนไขผลของการพยากรณ์ไว้ที่ค่า 60 เปอร์เซ็นต์ หมายความว่าถ้านำอะไหล่มาทำการทดสอบกับระบบและระบบคำนวณแล้วทุกคลาสและคลาสที่คำนวณได้ค่าสูงสุดแล้วต่ำกว่า 60 เปอร์เซ็นต์ จะแสดงข้อความ “Spare part does not match” แต่ถ้าค่าของคลาสที่คำนวณได้ค่าสูงสุดแล้วเกิน 60 เปอร์เซ็นต์ระบบจะแสดงรหัสและชื่อของอะไหล่มาแสดงผลทางจอภาพ กรณีที่จะเกิดข้อความ “Spare part does not match” เนื่องมาจากกล้องจับภาพได้ไม่ชัดเจน หรือมุมมองภาพไม่ถูกต้อง หรือกล้องจับภาพแล้วไม่มีอะไหล่อยู่ในภาพหรืออยู่ในภาพน้อย และโอกาสที่จะไม่มีอะไหล่ตัวนั้นอยู่ในระบบ กล่าวคืออะไหล่ที่นำมาทดสอบไม่ได้รับการถูกฝึกฝนตั้งแต่แรก เป็นต้น

10. แสดงผลลัพธ์ทางหน้าจอ โดยจะแสดงภาพที่กำลังจับอยู่ พร้อมแสดงข้อความที่ได้จากการทำนายได้ ดังภาพที่ 4.12 และภาพที่ 4.13

ภาพที่ 4.12 แสดงภาพที่ใช้อะไหล่หมายเลข 603409 มาให้โปรแกรมวิเคราะห์ และผลการวิเคราะห์ตรงกัน



ภาพที่ 4.13 แสดงการทำงานของระบบเมื่อใช้อะไหล่หมายเลข 603382 มาให้โปรแกรมวิเคราะห์และผลการวิเคราะห์ตรงกัน

ตอนที่ 4 การประเมินประสิทธิภาพของระบบ

ผลการประเมินประสิทธิภาพของระบบจดจำภาพสำหรับการตรวจจับอะไหล่ผู้วิจัยได้ทำการทดสอบโดยการปรับแต่งค่าตัวแปรของโปรแกรมเพื่อเปรียบเทียบค่าความแม่นยำของระบบและเลือกค่าที่มีความแม่นยำสูงสุดเพื่อนำไปใช้งาน ค่าตัวแปรที่ทำการปรับแต่งและทำให้มีผลต่อค่าความแม่นยำคือ HOW_MANY_TRAINING_STEPS โดยเป็นจำนวนรอบการหาค่าเอาท์พุต จากค่าอินพุตที่เข้าไปในโครงข่ายประสาทเทียม และทำการเปรียบเทียบค่าของเอาท์พุตที่ได้ทำการหาไว้แล้ว ถ้าเปรียบเทียบแล้วเป็นค่าที่ยอมรับได้ก็ให้ทำการรับข้อมูลชุดถัดไป แต่ถ้าหากค่ามากกว่าค่าที่ยอมรับได้ ให้ทำการปรับค่าถ่วงน้ำหนักซึ่งเป็นค่าที่ใช้ในการปรับค่าตามการเรียนรู้จากอินพุตที่นำเข้าเพื่อสร้างฟังก์ชันแสดงความสัมพันธ์ระหว่างอินพุตและค่าเป้าหมายที่ต้องการ และค่าไบแอสเป็นค่าที่เชื่อมติดระหว่างชั้นซ่อนและชั้นเอาท์พุตเพื่อใช้ในการขยับช่วงของข้อมูลให้ขยับช่วงเพื่อเข้าใกล้ค่าเป้าหมายมากขึ้น มีการกำหนดค่าเริ่มต้นเช่นเดียวกับค่าถ่วงน้ำหนักคือเป็นการสุ่มค่าน้อย ๆ เมื่อทำการปรับเรียบร้อยแล้วให้ทำการรับข้อมูลชุดถัดไปและทำตามขั้นตอนซ้ำตามจำนวนที่ได้ปรับแต่งไว้ ผู้วิจัยได้ทำการทดสอบโดยปรับแต่งจำนวนรอบการหาค่าเอาท์พุตเป็น 4000, 10000 และ 20000 เพื่อทำการเปรียบเทียบค่าความแม่นยำในการพยากรณ์ โดยค่า 4000 เป็นค่าที่ทางกูเกิลได้กำหนดเป็นค่าตั้งต้นไว้

ขั้นตอนในการทดสอบเพื่อประเมินประสิทธิภาพของระบบจดจำภาพ ผู้วิจัยทำการเตรียมการทดสอบโดยการทำการจับภาพและบันทึกภาพใหม่ของอะไหล่แต่ละชิ้นทั้งหมด 25 ชิ้น ชิ้นละ 100 ภาพ ในการทดสอบจะทำการทดสอบกับภาพนิ่งโดยให้กล้องจับภาพและบันทึกไว้และ

จะใช้ภาพที่บันทึกดังกล่าวใช้ในการทดสอบทั้งระบบเพื่อเปรียบเทียบผลที่ได้ ผู้วิจัยได้ทำการสร้างไฟล์เพื่อใช้ในการพยากรณ์ภาพนิ่งชื่อไฟล์ test-image.py โดยการปรับแต่งโปรแกรมจากการให้โปรแกรมอ่านค่าจากกล้องเปลี่ยนเป็นให้อ่านค่าจากไฟล์ภาพแทน การปรับเปลี่ยนดังกล่าวเป็นคุณลักษณะการทำงานของฟังก์ชันใน OpenCV เนื่องจากผู้วิจัยต้องการได้ค่าพยากรณ์จากโปรแกรมเป็นค่าเดียวที่ไม่ปรับเปลี่ยนไปตามมุมมองหรือระยะเวลาที่จับภาพที่ต่างกันซึ่งอาจจะมีผลต่อการพยากรณ์ที่ทำให้ค่าที่ได้ไม่นิ่งยากต่อการบันทึกค่าที่ได้จากการพยากรณ์ จากนั้นทำการทดสอบตามขั้นตอนดังต่อไปนี้

1. ทำการปรับแต่งโดยกำหนดค่าพารามิเตอร์จำนวนรอบในการหาค่าเอาท์พุตในโปรแกรมในไฟล์ retrain.py เป็นค่า 4000

2. ทำการฝึกฝนข้อมูลใหม่โดยการรันไฟล์ retrain.py

3. เมื่อการฝึกฝนข้อมูลเสร็จสิ้นจะได้ไฟล์ retrained_graph.pb ที่มีการปรับปรุงค่าแล้ว

4. ทำการรันไฟล์ชื่อ test-image.py เพื่อใช้ตัวแบบที่ได้ฝึกฝนแล้วทำการพยากรณ์ไฟล์ภาพที่ได้จากขั้นตอนการเตรียมไว้แล้ว ไฟล์ดังกล่าวจะถูกเขียนรหัสโปรแกรมคอมพิวเตอร์เพื่อให้อ่านไฟล์ภาพจากฮาร์ดดิสก์แทนการจับภาพผ่านกล้อง

5. โปรแกรมจะทำการอ่านไฟล์ภาพจากแฟ้มข้อมูลและทำการพยากรณ์ทีละภาพ ของอะไหล่แต่ละชิ้น

5.1 อะไหล่ชิ้นที่ 1 จะทำการพยากรณ์ภาพที่ 1 และแสดงผลทางจอภาพเช่นเดียวกับการแสดงผลแบบเรียลไทม์ เพียงแต่การทดสอบจะทดสอบกับภาพนิ่ง

5.2 บันทึกผลการพยากรณ์ในตารางคอนฟิวชันเมทริกซ์

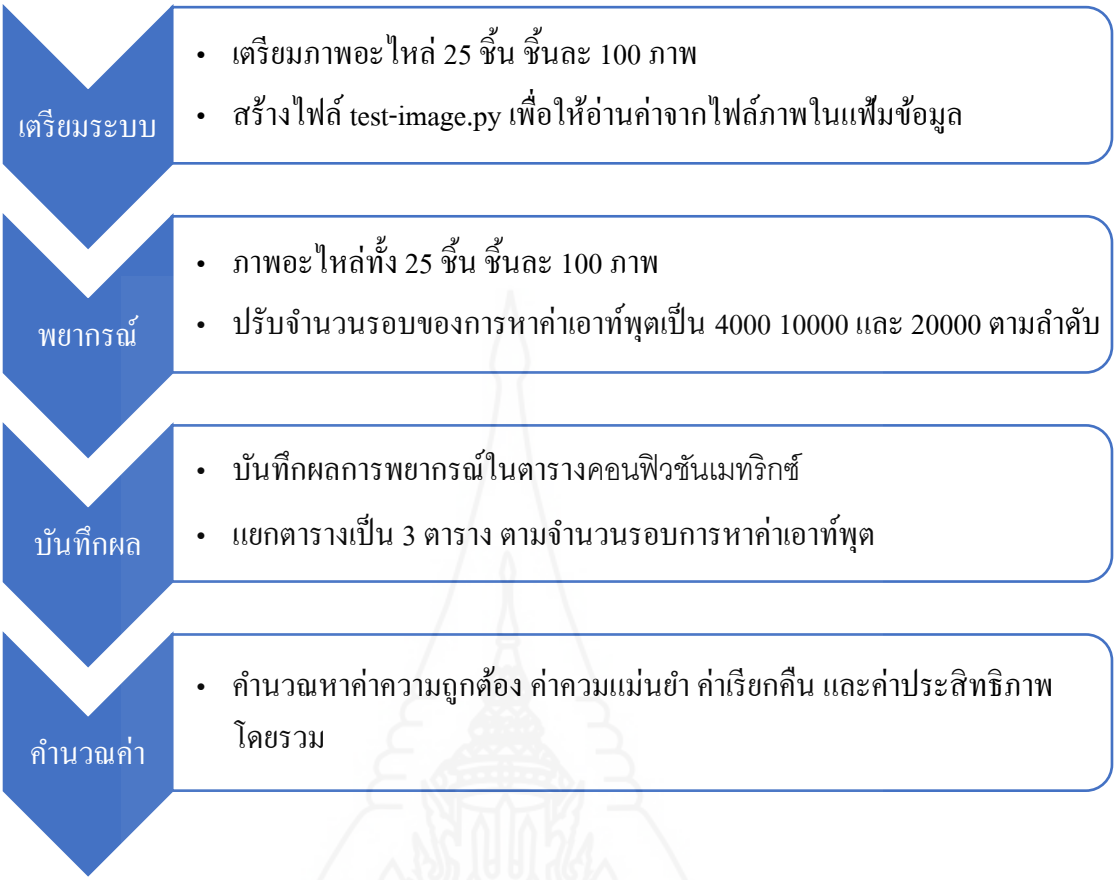
5.3 เมื่อกดปิดหน้าจอโปรแกรมจะทำการพยากรณ์ภาพถัดไป

5.4 ทำซ้ำขั้นตอนที่ 5.1 ถึงขั้นตอนที่ 5.3 สำหรับการบันทึกผลการพยากรณ์ของภาพที่ 2 ถึงภาพที่ 100 หรือจนครบทุกภาพ โปรแกรมจะหยุดการทำงานเมื่ออ่านไฟล์ทั้งหมดแล้ว

6. ทำซ้ำในขั้นตอนที่ 5 สำหรับอะไหล่ชิ้นต่อไป ทดสอบจนครบอะไหล่ทั้ง 25 ชิ้น

7. ทำซ้ำขั้นตอนที่ 2 ถึงขั้นตอนที่ 6 สำหรับจำนวนรอบในการหาค่าเอาท์พุตในไฟล์ retrain.py เป็น 10000 และ 20000 ตามลำดับ

8. ทำการคำนวณหาค่าความถูกต้อง ค่าเรียกคืน ค่าความแม่นยำ และค่าประสิทธิภาพโดยรวมของระบบจากตารางคอนฟิวชันเมทริกซ์



ภาพที่ 4.14 แสดงภาพรวมของขั้นตอนการทดสอบระบบจดจำภาพอะไหล่

จากภาพที่ 4.14 เป็นภาพที่แสดงขั้นตอนการทดสอบระบบจดจำภาพสำหรับการตรวจจับอะไหล่โดยใช้โครงข่ายประสาทเทียมแบบคอนโวลูชันแบ่งเป็น 4 ขั้นตอน คือขั้นตอนแรกเป็นการเตรียมระบบ โดยการเตรียมภาพสำหรับการทดสอบโดยอะไหล่ 1 ชิ้นจะมีภาพเพื่อนำมาทดสอบจำนวน 100 ภาพ ขั้นตอนที่สองเป็นกระบวนการนำภาพที่ได้จากการเตรียมมาทำการทดสอบกับระบบ โดยการทดสอบจะทดลองปรับเปลี่ยนจำนวนรอบของการหาค่าเอาท์พุตเป็น 4000 ซึ่งเป็นค่าตั้งต้น ค่า 10000 และค่า 20000 เป็นค่าที่ต้องการทดสอบหาค่าความถูกต้องแม่นยำของระบบเมื่อค่าพารามิเตอร์เปลี่ยนไป จากนั้นบันทึกผลลงในตารางคอนฟิวชันเมทริกซ์โดยแยกเป็น 3 ตารางตามค่าจำนวนรอบของการหาค่าเอาท์พุต และนำค่าจากตารางคอนฟิวชันเมทริกซ์มาคำนวณหาค่าความถูกต้อง ค่าความแม่นยำ ค่าเรียกคืนและค่าประสิทธิภาพโดยรวมของระบบ เพื่อนำมาเปรียบเทียบว่าตัวแบบใดให้ค่าที่ดีกว่า และจะใช้ตัวแบบนั้นเพื่อนำมาใช้งานตรวจจับอะไหล่ในการใช้งานจริง

Class	Data Set	205617	205620	205664	205698	228865	229031	229053	229054	601356	601482	602102	602961	602964	603380	603382	603383	603387	603408	603409	606174	614286	617590	618609	618627	618628	TP	FN	Re
205617	100	61	13	26																							61	39	0.61
205620	100	26	33	29	6	5												1									33	67	0.33
205664	100	42	6	52																							52	48	0.52
205698	100	2			84	14																					84	16	0.84
228865	100				44	56																					56	44	0.56
229031	100						81	8	11																		81	19	0.81
229053	100		1				12	85	1														1				85	15	0.85
229054	100						16		84																		84	16	0.84
601356	100									79	21																79	21	0.79
601482	100									9	91																91	9	0.91
602102	100									14		85		1													85	15	0.85
602961	100												98									2					98	2	0.98
602964	100										1	2		95								2					95	5	0.95
603380	100														95				1					2		2	95	5	0.95
603382	100															99				1							99	1	0.99
603383	100	2	1	2													95										95	5	0.95
603387	100	1																99									99	1	0.99
603408	100																		97						3		97	3	0.97
603409	100																			100							100	0	1.00
606174	100																				100						100	0	1.00
614286	100																					100					100	0	1.00
617590	100																			1			99				99	1	0.99
618609	100														2									96		2	96	4	0.96
618627	100															1							14		85		85	15	0.85
618628	100																							2		98	98	2	0.98
	TP	61	33	52	84	56	81	85	84	79	91	85	98	95	95	99	95	99	97	100	100	100	99	96	85	98	2147		21.47
	FP	73	21	57	50	19	28	8	12	23	22	2	0	1	2	1	0	1	1	2	0	4	15	4	3	4			
	Pr	0.46	0.61	0.48	0.63	0.75	0.74	0.91	0.88	0.77	0.81	0.98	1.00	0.99	0.98	0.99	1.00	0.99	0.99	0.98	1.00	0.96	0.87	0.96	0.97	0.96	21.64		
	F1	0.52	0.43	0.50	0.72	0.64	0.78	0.88	0.86	0.78	0.85	0.91	0.99	0.97	0.96	0.99	0.97	0.99	0.98	0.99	1.00	0.98	0.93	0.96	0.90	0.97	21.45		

ภาพที่ 4.15 แสดงการคำนวณหาค่าความแม่นยำ (Precision) และค่าเรียกคืน (Recall) ของจำนวนรอบของการหาเอาท์พุตเท่ากับ 4000

การปรับจำนวนรอบการหาค่าเอาท์พุตเป็นค่า 4000 และคำนวณหาค่าประสิทธิภาพโดยรวม โดยให้ N = จำนวนข้อมูลทั้งหมด = 2,500 และ C = จำนวนคลาสทั้งหมด = 25

การคำนวณหาค่าความแม่นยำ (Precision)

$$\mathrm{Pr} = \frac{\sum_{i=1}^{c} Pr_i}{c}$$

= 21.64 / 25

= 0.8656 หรือ 86.56 %

การคำนวณหาค่าเรียกคืน (Recall)

$$\mathrm{Re} = \frac{\sum_{i=1}^{c} Re_i}{c}$$

= 21.47 / 25

= 0.8588 หรือ 85.88 %

การคำนวณหาค่าประสิทธิภาพโดยรวม (F-measure)

$$\mathrm{F1} = \frac{\sum_{i=1}^{c} F1_i}{c}$$

= 21.45 / 25

= 0.8580 หรือ 85.80 %

การคำนวณหาค่าความถูกต้อง (Accuracy)

$$Accuracy = \frac{\sum_{i=1}^{c} TP_i}{N}$$

= 2,147 / 2,500

= 0.8588 หรือ 85.88 %

การประเมินประสิทธิภาพของระบบจดจำภาพสำหรับการตรวจจับอะไหล่โดยกำหนดค่าการฝึกฝนเท่ากับ 4000 พิจารณาค่าเฉลี่ยของ Precision, Recall, Accuracy และ F-measure ได้ผลลัพธ์ค่าความแม่นยำร้อยละ 86.56 ค่าเรียกคืนร้อยละ 85.88 ค่าความถูกต้องร้อยละ 85.88 และการวัดประสิทธิภาพโดยรวมร้อยละ 85.80 ตามลำดับ

Class	Data Set	205617	205620	205664	205698	228865	229031	229053	229054	601356	601482	602102	602961	602964	603380	603382	603383	603387	603408	603409	606174	614286	617590	618609	618627	618628	TP	FN	Re
205617	100	65	16	19																							65	35	0.65
205620	100	28	33	28	7	3												1									33	67	0.33
205664	100	40	8	52																							52	48	0.52
205698	100	1			88	11																					88	12	0.88
228865	100				46	54																					54	46	0.54
229031	100						85	5	10																		85	15	0.85
229053	100						17	83																			83	17	0.83
229054	100						13		87																		87	13	0.87
601356	100									85	15																85	15	0.85
601482	100									15	85																85	15	0.85
602102	100									11		89															89	11	0.89
602961	100												99									1					99	1	0.99
602964	100										1	2		96								1					96	4	0.96
603380	100														96									2		2	96	4	0.96
603382	100															100											100	0	1.00
603383	100		3	2													95										95	5	0.95
603387	100	1																99									99	1	0.99
603408	100																		100								100	0	1.00
603409	100																			100							100	0	1.00
606174	100																				100						100	0	1.00
614286	100																					100					100	0	1.00
617590	100																						100				100	0	1.00
618609	100														2									97		1	97	3	0.97
618627	100																						17		83		83	17	0.83
618628	100																							2		98	98	2	0.98
	TP	65	33	52	88	54	85	83	87	85	85	89	99	96	96	100	95	99	100	100	100	100	100	97	83	98	**2169**		**21.69**
	FP	70	27	49	53	14	30	5	10	26	16	2	0	0	2	0	0	1	0	0	0	2	17	4	0	3			
	Pr	0.48	0.55	0.51	0.62	0.79	0.74	0.94	0.90	0.77	0.84	0.98	1.00	1.00	0.98	1.00	1.00	0.99	1.00	1.00	1.00	0.98	0.85	0.96	1.00	0.97	**21.86**		
	F1	0.55	0.41	0.52	0.73	0.64	0.79	0.88	0.88	0.81	0.85	0.93	0.99	0.98	0.97	1.00	0.97	0.99	1.00	1.00	1.00	0.99	0.92	0.97	0.91	0.98	**21.66**		

ภาพที่ 4.16 แสดงการคำนวณหาค่าความแม่นยำ (Precision) และค่าเรียกคืน (Recall) ของจำนวนรอบของการหาเอาท์พุตเท่ากับ 10000

การปรับจำนวนรอบการหาค่าเอาท์พุตเป็น 10000 และคำนวณหาค่าประสิทธิภาพโดยรวม โดยให้ N = จำนวนข้อมูลทั้งหมด = 2,500 และ C = จำนวนคลาสทั้งหมด = 25

การคำนวณหาค่าความแม่นยำ (Precision)

$$\mathrm{Pr} = \frac{\sum_{i=1}^{c} Pr_i}{c}$$

= 21.86 / 25

= 0.8744 หรือ 87.44 %

การคำนวณหาค่าเรียกคืน (Recall)

$$\mathrm{Re} = \frac{\sum_{i=1}^{c} Re_i}{c}$$

= 21.69 / 25

= 0.8676 หรือ 86.76 %

การคำนวณหาค่าประสิทธิภาพโดยรวม (F-measure)

$$\mathrm{F1} = \frac{\sum_{i=1}^{c} F1_i}{c}$$

= 21.66 / 25

= 0.8664 หรือ 86.64 %

การคำนวณหาค่าความถูกต้อง (Accuracy)

$$Accuracy = \frac{\sum_{i=1}^{c} TP_i}{N}$$

= 2,169 / 2,500

= 0.8676 หรือ 86.76 %

การประเมินประสิทธิภาพของระบบจดจำภาพสำหรับการตรวจจับอะไหล่โดยกำหนดค่าการฝึกฝนเท่ากับ 10000 พิจารณาค่าเฉลี่ยของ Precision, Recall, Accuracy และ F-measure ได้ผลลัพธ์ค่าความแม่นยำร้อยละ 87.44 ค่าเรียกคืนร้อยละ 86.76 ค่าความถูกต้องร้อยละ 86.76 และการวัดประสิทธิภาพโดยรวมร้อยละ 86.64 ตามลำดับ

Class	Data Set	205617	205620	205664	205698	228865	229031	229053	229054	601356	601482	602102	602961	602964	603380	603382	603383	603387	603408	603409	606174	614286	617590	618609	618627	618628	TP	FN	Re
205617	100	31	54	15																							31	69	0.31
205620	100	12	55	28	5																						55	45	0.55
205664	100	32	40	28																							28	72	0.28
205698	100		2		90	8																					90	10	0.90
228865	100		2		20	78																					78	22	0.78
229031	100						89	3	8																		89	11	0.89
229053	100		1				15	78	4														2				78	22	0.78
229054	100						10		90																		90	10	0.90
601356	100									93	7																93	7	0.93
601482	100									22	78																78	22	0.78
602102	100									3		96		1													96	4	0.96
602961	100												99										1				99	1	0.99
602964	100										1	2		97													97	3	0.97
603380	100														97				1					1		1	97	3	0.97
603382	100															100											100	0	1.00
603383	100	2	7	9													82										82	18	0.82
603387	100	1																99									99	1	0.99
603408	100																		97						3		97	3	0.97
603409	100																			100							100	0	1.00
606174	100																				100						100	0	1.00
614286	100																					100					100	0	1.00
617590	100																			1			99				99	1	0.99
618609	100														2									97		1	97	3	0.97
618627	100															1							19		80		80	20	0.80
618628	100																							1		99	99	1	0.99
	TP	31	55	28	90	78	89	78	90	93	78	96	99	97	97	100	82	99	97	100	100	100	99	97	80	99	2152		21.52
	FP	47	106	52	25	8	25	3	12	25	8	2	0	1	2	1	0	0	1	1	0	0	22	2	3	2			
	Pr	0.40	0.34	0.35	0.78	0.91	0.78	0.96	0.88	0.79	0.91	0.98	1.00	0.99	0.98	0.99	1.00	1.00	0.99	0.99	1.00	1.00	0.82	0.98	0.96	0.98	21.76		
	F1	0.35	0.42	0.31	0.84	0.84	0.83	0.86	0.89	0.85	0.84	0.97	0.99	0.98	0.97	1.00	0.90	0.99	0.98	1.00	1.00	1.00	0.90	0.97	0.87	0.99	21.55		

ภาพที่ 4.17 แสดงการคำนวณหาค่าความแม่นยำ (Precision) และค่าเรียกคืน (Recall) ของจำนวนรอบของการหาเอาท์พุตเท่ากับ 20000

การปรับจำนวนรอบการหาค่าเอาท์พุตเป็น 20000 และคำนวณหาค่าประสิทธิภาพโดยรวม โดยให้ N = จำนวนข้อมูลทั้งหมด = 2,500 และ C = จำนวนคลาสทั้งหมด = 25

การคำนวณหาค่าความแม่นยำ (Precision)

$$\mathrm{Pr} = \frac{\sum_{i=1}^{c} Pr_i}{c}$$

= 21.76 / 25

= 0.8704 หรือ 87.04 %

การคำนวณหาค่าเรียกคืน (Recall)

$$\mathrm{Re} = \frac{\sum_{i=1}^{c} Re_i}{c}$$

= 21.52 / 25

= 0.8608 หรือ 86.08 %

การคำนวณหาค่าประสิทธิภาพโดยรวม (F-measure)

$$\mathrm{F1} = \frac{\sum_{i=1}^{c} F1_i}{c}$$

= 21.55 / 25

= 0.8620 หรือ 86.20 %

การคำนวณหาค่าความถูกต้อง (Accuracy)

$$Accuracy = \frac{\sum_{i=1}^{c} TP_i}{N}$$

= 2,152 / 2,500

= 0.8608 หรือ 86.08 %

การประเมินประสิทธิภาพของระบบจดจำภาพสำหรับการตรวจจับอะไหล่โดยกำหนดค่าการฝึกฝนเท่ากับ 20000 พิจารณาค่าเฉลี่ยของ Precision, Recall, Accuracy และ F-measure ได้ผลลัพธ์ค่าความแม่นยำร้อยละ 87.04 ค่าเรียกคืนร้อยละ 86.08 ค่าความถูกต้องร้อยละ 86.08 และการวัดประสิทธิภาพโดยรวมร้อยละ 86.20 ตามลำดับ

จากการทดลองทดสอบระบบสำหรับการเปลี่ยนค่าจำนวนรอบของการหาเอาต์พุตเพื่อเปรียบเทียบหาค่าที่ดีที่สุดสำหรับนำไปใช้ในการพัฒนาระบบจดจำภาพสำหรับการตรวจจับอะไหล่ ดังแสดงในตารางที่ 4.1

ตารางที่ 4.1 การเปรียบเทียบค่าความแม่นยำ (Precision) และค่าเรียกคืน (Recall) ของจำนวนรอบของการหาเอาต์พุต

การวัดประสิทธิภาพ	จำนวนรอบของการหาเอาต์พุตเป็น 4000 (%)	จำนวนรอบของการหาเอาต์พุตเป็น 10000 (%)	จำนวนรอบของการหาเอาต์พุตเป็น 20000 (%)
Precision	86.56	87.44	87.04
Recall	85.88	86.76	86.08
F-measure	85.80	86.64	86.20
Accuracy	85.88	86.76	86.08

จากตารางที่ 4.1 การเปรียบเทียบค่าการวัดประสิทธิภาพของระบบเมื่อปรับจำนวนรอบของการหาเอาต์พุตในส่วนของการฝึกฝนข้อมูลยิ่งค่าสูงจะมีผลทำให้ค่าต่าง ๆ ในการวัดประสิทธิภาพเพิ่มขึ้นซึ่งหมายความว่าระบบสามารถที่จะพยากรณ์ได้แม่นยำกว่าค่าของจำนวนรอบของการหาเอาต์พุตต่ำ แต่เมื่อเพิ่มตัวเลขหรือจำนวนรอบในการฝึกฝนมากขึ้นถึงจุด ๆ หนึ่งการวัดประสิทธิภาพของระบบจะมีค่าลดลงเกิดจากตัวแบบพยายามมากเกินไปที่จะตรวจจับรูปแบบของข้อมูลที่ใช้ในการแบ่งข้อมูลของการทดสอบเพื่อให้ได้ผลลัพธ์มีความแม่นยำแต่ตัวแบบจะทำงานได้ดีกับข้อมูลที่ใช้ในการฝึกฝนเท่านั้น ข้อมูลที่ใช้ในการทดสอบบางชิ้นมีลักษณะของสี รูปทรง และขนาด ใกล้เคียงกันทำให้มีผลทำให้ค่าการวัดประสิทธิภาพลดลง

บทที่ 5
สรุปการวิจัย อภิปรายผล และข้อเสนอแนะ

ในงานวิจัยฉบับนี้ผู้วิจัยได้นำเสนอวิธีการพัฒนาระบบจดจำภาพสำหรับการตรวจจับอะไหล่โดยใช้โครงข่ายประสาทเทียมแบบคอนโวลูชันด้วยวิธีการเรียนรู้เชิงถ่ายโอนข้อมูล ซึ่งวิธีการนี้จะเป็นการใช้ประโยชน์จากผลของการฝึกฝนข้อมูลของตัวแบบที่ ใช้ข้อมูลในการฝึกฝนจากแหล่งข้อมูลที่มีปริมาณมากมาทำการถ่ายโอนการเรียนรู้ที่ได้จากการฝึกฝนตัวแบบ มาทำการฝึกฝนข้อมูลใหม่ในชั้นสุดท้ายของตัวแบบใหม่ที่ต้องการให้ระบบรู้จำ ซึ่งตัวแบบนี้จะในการรู้จำอะไหล่ ในการฝึกฝนชั้นสุดท้ายของตัวแบบใหม่จะใช้ข้อมูลในการฝึกฝนจำนวน 500 ภาพถึง 1,200 ภาพต่ออะไหล่ 1 ชิ้น เพื่อใช้ในการประกอบการค้นหาอะไหล่ในบริษัทเพื่อช่วยแก้ปัญหาเกี่ยวกับการไม่ทราบรายละเอียดของอะไหล่ที่ต้องการเบิกเช่นรหัสอะไหล่ หรือชื่ออะไหล่ ซึ่งคำค้นเหล่านี้จะนำไปใช้ค้นหาที่อยู่อะไหล่ในระบบ ERP ของบริษัทในกรณีที่ไม่ทราบก็ไม่สามารถค้นหาอะไหล่ได้จึงไม่สามารถเบิกจ่ายอะไหล่ได้ในขณะนั้นจำเป็นค้นหาตามเอกสารของเครื่องจักรซึ่งจะมีรายละเอียดของอะไหล่แต่อาจต้องใช้เวลานานและบางครั้งที่เครื่องจักรบางเครื่องจะต้องใช้อุปกรณ์ช่วยอื่นเสริมซึ่งไม่มีรายละเอียดแสดงไว้ในคู่มือกรณีนี้อาจค้นหาได้ยาก ดังนั้นงานวิจัยนี้ถูกพัฒนามาเพื่อบันทึกข้อมูลอะไหล่ของบริษัททั้งหมด ซึ่งเป็นการเพิ่มการค้นหาด้วยภาพอีกหนึ่งช่องทางจะช่วยในการลดเวลาการสืบค้นรายละเอียดของอะไหล่จากหลายช่องทางเพื่อความรวดเร็วเพราะเป็นการค้นหาอะไหล่โดยใช้ภาพ มีผลทำให้ให้สามารถทำการเบิกจ่ายอะไหล่นั้นสามารถทำสำเร็จ ได้อะไหล่ที่ถูกต้องเพื่อนำไปเปลี่ยน ลดเวลาการหยุดของกระบวนการผลิตได้และทำให้ส่งสินค้าได้ทันตามลูกค้าต้องการ ซึ่งสามารถสรุปการวิจัยได้ดังต่อไปนี้

1. สรุปผลการวิจัย
2. ข้อเสนอแนะ
3. ปัญหาและอุปสรรค

1. สรุปผลการวิจัย

งานวิจัยนี้ได้เริ่มต้นจากการนำเสนอปัญหาของการเบิกจ่ายอะไหล่ของบริษัทซึ่งเป็นปัญหาหาที่สำคัญ เพราะมีผลกระทบโดยตรงต่อกระบวนการผลิตและได้ค้นหาวิธีการที่จะแก้ปัญหาดังกล่าว ด้วยการพัฒนาระบบจดจำภาพในการตรวจจับอะไหล่ ซึ่งใช้หลักการของการจดจำภาพ

กระบวนการของการจดจำภาพสิ่งที่สำคัญคือการสกัดเพื่อหาคุณลักษณะสำคัญของภาพก่อนเข้าสู่กระบวนการจำแนกประเภทซึ่งเป็นขั้นตอนสุดท้าย แต่จากหลาย ๆ งานวิจัยที่ได้นำเสนอมีข้อจำกัดบางอย่างทำให้มีผลต่อการพัฒนาระบบเพื่อจะนำไปใช้งานจริง เช่นการใช้จำนวนรูปภาพที่มากเพื่อใช้ในการฝึกฝนตัวแบบ หรือมีการเพิ่มขั้นตอนของวิธีการสกัดคุณลักษณะขึ้นมาเพื่อให้ตัวแบบมีความถูกต้องแม่นยำสูงและยังเป็นต้นแบบเพื่อนำไปพัฒนาต่อยอดเพื่อจะนำไปใช้งานได้จริง ประสิทธิภาพของอุปกรณ์ที่ต้องมีประสิทธิภาพสูงเพราะโปรแกรมต้องการใช้ทรัพยากรระบบที่สูงเพื่อความรวดเร็วในการประมวลผล งานวิจัยดังกล่าวหากมีการเปลี่ยนมาใช้วิธีที่นำเสนออาจลดข้อจำกัดต่าง ๆ อีกทั้งยังมีผลทำให้งานวิจัยใช้งานได้จริงในการแก้ปัญหาภายในบริษัท โดยการเปลี่ยนมาใช้วิธีการจดจำรูปภาพโดยใช้โครงข่ายประสาทเทียมแบบคอนโวลูชัน ด้วยวิธีการเรียนรู้เชิงถ่ายโอนข้อมูล โดยกรใช้ตัวแบบที่มีการฝึกฝนไว้แล้ว มาทำการสกัดคุณลักษณะเด่นของภาพและฝึกฝนข้อมูลในชั้นสุดท้ายของโครงข่ายเพื่อรู้จำอะไหล่ ถ้าใช้วิธีการนี้จะสามารถลดขนาดของข้อมูลที่ใช้ในการฝึกฝนหรือใช้ภาพเพียง 500 ภาพถึง 1,200 ภาพ เพื่อใช้ในการฝึกฝน และลดเวลาในการฝึกฝนข้อมูลเพราะมิได้ฝึกฝนทั้งหมดจะทำการฝึกฝนข้อมูลเฉพาะชั้นสุดท้ายของโครงข่ายจำทำให้มีความรวดเร็วเหมาะกับสถานการณ์ภายในบริษัทและให้ค่าความถูกต้องแม่นยำที่สามารถยอมรับได้และเหมาะกับงานที่จะนำมาใช้

งานวิจัยการพัฒนาระบบจดจำภาพสำหรับการตรวจจับอะไหล่โดยใช้โครงข่ายประสาทเทียมแบบคอนโวลูชัน ประกอบด้วยวัตถุประสงค์ 2 ประการ คือ 1) เพื่อพัฒนาระบบจดจำภาพสำหรับการตรวจจับอะไหล่โดยใช้โครงข่ายประสาทเทียมแบบคอนโวลูชัน และ 2) เพื่อประเมินประสิทธิภาพของระบบจดจำภาพสำหรับตรวจจับอะไหล่โดยใช้โครงข่ายประสาทเทียมแบบคอนโวลูชัน สิ่งที่ได้คือ

1. ระบบจดจำภาพสำหรับการตรวจจับอะไหล่โดยใช้โครงข่ายประสาทเทียมแบบคอนโวลูชัน ซึ่งเป็นการประยุกต์ใช้ตัวแบบโครงข่ายประสาทเทียมที่สามารถรู้จำวัตถุ จำแนกตัวอักษรและจดจำใบหน้าของบุคคลได้อย่างมีประสิทธิภาพ ซึ่งให้ค่าความถูกต้องและค่าความแม่นยำที่สูงมาทำการถ่ายโอนความรู้ที่ได้จากการฝึกฝนเพื่อนำมาพัฒนามาเป็นระบบจดจำภาพสำหรับตรวจจับอะไหล่ แบ่งการทำงานของตัวแบบที่ได้ฝึกฝนมาออกเป็น 2 ส่วนคือกระบวนการสกัดคุณลักษณะสำคัญของภาพโดยนำเข้าภาพอะไหล่ที่ต้องการให้ระบบรู้จำมาผ่านกระบวนการสกัดคุณลักษณะซึ่งจะใช้ค่าน้ำหนักและค่าไบแอสเดิมที่ได้รับการฝึกฝนมาก่อนหน้าเพื่อเตรียมเป็นข้อมูลนำเข้าในกระบวนการจำแนกประเภท และกระบวกการจำแนกประเภทหรือส่วนรู้จำของระบบจะทำการนำเข้าข้อมูลจากกระบวนการสกัดคุณลักษณะแล้วมาทำการฝึกฝนข้อมูลโดยใช้น้ำหนักและค่าไบแอสเดิมเป็นค่าตั้งต้นจากนั้นทำการปรับค่าดังกล่าวเรื่อย ๆ จากกระบวนการฝึกฝน จนกว่าค่าที่

ได้จะมีค่าเข้าใกล้กับค่าเป้าหมาย ขั้นตอนการทำงานดังกล่าวจะถูกพัฒนาโดยใช้ภาษาไพธอนเพราะเป็นภาษาที่สนับสนุนการทำเกี่ยวกับการเรียนรู้ของเครื่องและการเรียนรู้เชิงลึก ใช้ Tensorflow ซึ่งเป็นไลบรารีที่สนับการถ่ายโอนการเรียนรู้จากตัวแบบที่ได้รับการฝึกฝนไว้แล้ว ใช้ตัวแบบ Inception v3 ซึ่งเป็นโครงข่ายประสาทเทียมแบบคอนโวลูชันที่ถูกพัฒนาโดยกูเกิลเพื่อใช้ในการรู้จำวัตถุโดยใช้ข้อมูลปริมาณที่มากและมากพอที่จะทำให้ตัวแบบมีค่าความถูกต้องแม่นยำที่สูง และใช้ OpenCV ซึ่งเป็นไลบรารีในการจัดการกับข้อมูลนำเข้าซึ่งจะทำหน้าที่จับภาและบันทึกเป็นไฟล์รูปภาพเพื่อเป็นข้อมูลในรูปแบบที่คอมพิวเตอร์ที่จะสามารถนำไปประมวลผลได้ งานวิจัยนี้ทำการทดลองกับอะไหล่ทั้งหมด 25 ชิ้น ซึ่งเป็นตัวแทนของอะไหล่ทั้งหมดของบริษัทในด้านลักษณะรูปทรง ขนาด สี และชนิดของวัสดุที่ใช้ทำ จากนั้นทำการรวบรวมข้อมูลรูปภาพจากกล้อง โดยระบบจะทำการเก็บรูปภาพจำนวน 700 รูป ซึ่งระยะห่างจากกล้องถึงอะไหล่อยู่ที่ 5 เซนติเมตรถึง 15 เซนติเมตรและจัดเก็บไว้ในแฟ้มข้อมูลซึ่งถูกแยกออกเป็น 1 แฟ้มข้อมูลต่ออะไหล่ 1 ชิ้น ผู้ดูแลระบบจะมีหน้าที่ในการคัดเลือกรูปภาพที่มีสิ่งที่ไม่เกี่ยวข้องน้อยที่สุดหรือมีอะไหล่อยู่ในภาพมากที่สุด โดยภาพจะต้องมีภาพอะไหล่อยู่ภายในรูปภาพอย่างน้อย 60 เปอร์เซ็นต์ ในลักษณะของรูปภาพที่มีหลากหลายมุมมองจำนวน 500 ภาพต่ออะไหล่ 1 ชิ้น ดังนั้นข้อมูลรูปภาพทั้งหมดที่ใช้ในการฝึกฝนมีจำนวน 12,500 ภาพ ทำการฝึกฝนข้อมูลโดยใช้ข้อมูลนำเข้าจากการสกัดคุณลักษณะของภาพและบันทึกผลการการฝึกฝนไว้ในตัวแบบเพื่อจะนำไปใช้งานหรือใช้ในการพยากรณ์อะไหล่ ขั้นตอนการสกัดคุณลักษณะและขั้นตอนของการจำแนกประเภทจะทำงานต่อเนื่องกัน ดังนั้นจึงจำเป็นต้องเตรียมข้อมูลรูปภาพและจัดเก็บไว้ในแฟ้มข้อมูลแยกเป็นส่วนของอะไหล่แต่ละชิ้น ขั้นการสกัดคุณลักษณะจะใช้เวลานานขึ้นอยู่กับประสิทธิภาพของอุปกรณ์คอมพิวเตอร์และจำนวนรูปภาพที่นำมาฝึกฝน

2. ผลการประเมินความถูกต้องของระบบ การประเมินความถูกต้องทำการทดสอบโดยใช้รูปภาพอะไหล่ที่ถ่ายและจัดเก็บใหม่ซึ่งไม่เกี่ยวข้องกับรูปภาพอะไหล่ที่ใช้ในการฝึกฝนตัวแบบ ทำการเก็บภาพ 100 ภาพต่ออะไหล่ 1 ชิ้นในลักษณะมุมมองของภาพที่หลากหลาย ผู้วิจัยได้ทำการทดลองการฝึกฝนข้อมูลและพบว่าตัวแปรหรือพารามิเตอร์ตัวหนึ่งในส่วนของโปรแกรมมีผลต่อค่าความถูกต้องแม่นยำของระบบจึงได้ทดสอบร่วมกับการปรับค่าที่มีผลต่อประสิทธิภาพของระบบจดจำภาพคือจำนวนรอบที่ใช้ในการฝึกฝนโดยปรับค่าไว้ที่ 4000, 10000 และ 20000 ตามลำดับ โดยขั้นตอนแรกจะทำการตั้งค่าไว้ที่ 4000 จากนั้นทำการฝึกฝนในส่วนของชั้นสุดท้ายของโครงข่าย เมื่อทำการฝึกฝนเสร็จให้ทำการทดสอบโดยใช้ภาพที่ได้จากการเก็บเพื่อทดสอบ โดยให้โปรแกรมเรียกไฟล์รูปภาพแต่ละภาพมาให้ตัวแบบทำนายจากนั้นทำการบันทึกข้อมูลโดยใช้ตารางคอนฟิวชันเมทริกซ์เพื่อประเมินความถูกต้องแม่นยำของระบบจดจำภาพ เมื่อทดสอบเสร็จให้ทำการฝึกฝนโดยใช้

จำนวนรอบ 10000 และ 20000 ต่อไป ซึ่งสามารถสรุปผลการวิจัยเป็นผลการทดสอบประสิทธิภาพของระบบดังนี้

ผลการทดลองปรับค่าจำนวนรอบการฝึกฝนที่ 4000 ให้ค่าความแม่นยำ (Precision) ร้อยละ 86.56 ค่าเรียกคืน (Recall) ร้อยละ 85.88 ค่าความถูกต้อง (Accuracy) ร้อยละ 85.88 และค่าการวัดประสิทธิภาพโดยรวมของระบบ (F- measure) ร้อยละ 85.80

ผลการทดลองปรับค่าจำนวนรอบการฝึกฝนที่ 10000 ให้ค่าความแม่นยำ (Precision) ร้อยละ 87.44 ค่าเรียกคืน (Recall) ร้อยละ 86.76 ค่าความถูกต้อง (Accuracy) ร้อยละ 86.76 และค่าการวัดประสิทธิภาพโดยรวมของระบบ (F- measure) ร้อยละ 86.64

ผลการทดลองปรับค่าจำนวนรอบการฝึกฝนที่ 20000 ให้ค่าความแม่นยำ (Precision) ร้อยละ 87.04 ค่าเรียกคืน (Recall) ร้อยละ 86.08 ค่าความถูกต้อง (Accuracy) ร้อยละ 86.06 และค่าการวัดประสิทธิภาพโดยรวมของระบบ (F- measure) ร้อยละ 86.20

จากผลการทดลองพบว่าจำนวนรอบการฝึกฝนที่จำนวน 10000 เป็นค่าพารามิเตอร์ที่เหมาะสมกับงานวิจัยชิ้นนี้เนื่องจากให้ค่าความแม่นยำ ค่าเรียกคืน ค่าความถูกต้องและประสิทธิภาพโดยรวมของระบบที่สูงกว่าค่าจำนวนรอบในการฝึกฝนอื่นที่ปรับแต่ง ดังนั้นจำนวนรอบของการฝึกฝนที่ 10000 จะเป็นค่าที่ใช้ในการพัฒนาระบบจดจำภาพสำหรับการตรวจจับอะไหล่โดยใช้โครงข่ายประสาทเทียมแบบคอนโวลูชัน และนำไปใช้งานจริงในการเบิกจ่ายอะไหล่โดยใช้รูปภาพของอะไหล่จริงในการค้นหาในระบบของบริษัท

2. ข้อเสนอแนะ

2.1 อะไหล่ที่เป็นแท่งเฟอร์ไรท์ซึ่งมีลักษณะคล้ายคลึงกันทั้งขนาด สี และรูปทรงทำให้ค่าของจำนวนที่ระบบสามารถตรวจพบ หรือที่โปรแกรมทำนายได้ว่าเป็นคลาสนั้นจริงมีค่าต่ำกว่าคลาสอื่น ๆ ที่มีรูป ทรง สี และขนาดที่แตกต่างกัน ดังนั้นงานวิจัยนี้สามารถใช้งานได้ดีกับการจำแนกประเภท ของกลุ่มตัวอย่างที่มีรูปทรง ขนาด และสีที่แตกต่างกันอย่างชัดเจน ดังนั้นงานวิจัยในอนาคตควรหากระบวนการอื่นเพิ่มเติมเพื่อจำแนกวัตถุที่มีความคล้ายคลึงกันดังกล่าวให้มีความถูกต้องแม่นยำมากขึ้น

2.2 เพิ่มในส่วนต่อประสานกับผู้ใช้ในรูปแบบกราฟฟิก (อังกฤษ: Graphic User Interface, GUI) มีหน้าต่างการใช้งาน เมนู หรือปุ่มเลือกของโปรแกรมเพื่อให้ง่ายต่อการใช้งานของผู้ใช้ โดยเฉพาะในส่วนของผู้ดูแลระบบเพราะมีขั้นตอนการเตรียมข้อมูล การฝึกฝนข้อมูลและการ

นำตัวแบบที่ได้จากการฝึกฝนข้อมูลไปใช้งานซึ่งขั้นตอนอาจซับซ้อนการเพิ่มส่วนต่อประสานผู้ใช้จะทำให้ใช้งานสะดวกยิ่งขึ้น

2.3 อุปกรณ์เคลื่อนหรือโทรศัพท์มือถือ ซึ่งกำลังได้รับความนิยมอยู่ในขณะนี้ การพัฒนาโปรแกรมเพื่อรันบนอุปกรณ์เคลื่อนที่ดังกล่าวจะทำให้สะดวกต่อการใช้งานมากยิ่งขึ้น

2.4 การพัฒนาระบบจดจำภาพสำหรับการตรวจจับอะไหล่นี้ใช้ตัวแบบชื่อ Inception v3 ที่ได้รับการฝึกฝนจากวัตถุต่าง ๆ มาแล้ว (Pre-Trained model) มาทำการฝึกฝนซ้ำเพื่อใช้กับการจดจำอะไหล่หรือวัตถุต่าง ๆ แต่กรณีที่จะนำไปใช้จดจำใบหน้ามีตัวแบบที่ชื่อ VGG16 นำมาทำการถ่ายโอนข้อมูลเพื่อจดจำใบหน้าที่ต้องการจะให้ค่าความถูกต้องและแม่นยำกว่าเพราะได้รับการฝึกฝนจากใบหน้าบุคคล

3. ปัญหาและอุปสรรค

เครื่องคอมพิวเตอร์ประมวลผลช้าเนื่องจากงานวิจัยนี้เป็นการทำงานเกี่ยวกับการเรียนรู้ของเครื่องและการประมวลผลภาพซึ่งมีความต้องการความเร็วในการประมวลผลสูงซึ่งเครื่องคอมพิวเตอร์ที่จะนำมาใช้งานจะต้องเพิ่มในส่วนของ GPU (Graphics Processing Unit) เพื่อช่วยในการประมวลผลภาพให้เร็วทั้งในส่วนของการฝึกฝนข้อมูลและการใช้งานของผู้ใช้ซึ่งจะทำให้ภาพและผลการพยากรณ์แสดงผลอย่างลื่นไหล ภาพไม่กระตุกเมื่อใช้งานในรูปแบบเรียลไทม์ กรณีไม่มี GPU เครื่องคอมพิวเตอร์อาจต้องทำงานหนักและจะใช้เวลานานในการฝึกฝนข้อมูลก่อนจะได้ตัวแบบที่จะนำไปใช้งาน

บรรณานุกรม

บรรณานุกรม

กชกร ณ นครพนม. (2558). *การทำเมืองข้อมูลด้วยการจำแนกประเภทและการพยากรณ์. คลังข้อมูลเหมืองข้อมูลและธุรกิจอัจฉริยะ*. นนทบุรี: มหาวิทยาลัยสุโขทัยธรรมาธิราช.

ชัชวาลย์ วรวิทย์รัตนกุล. (2557). *การหาจุดสนใจของภาพที่เหมาะสมของขั้นตอนวิธีเอฟไอเอฟทีสำหรับการยืนยันภาพลายเซนต์*. (วิทยานิพนธ์ปริญญาวิทยาศาสตรมหาบัณฑิต). มหาวิทยาลัยศรีประทุม, ปทุมธานี.

ธนภัทร์ คุ้มสุภา. (2559). *การจำแนกประเภทข้อความภาษาไทยโดยใช้นิวรอลเน็ตเวิร์กคอนโวลูชันระดับตัวอักษร*. (วิทยานิพนธ์ปริญญาวิทยาศาสตรมหาบัณฑิต). จุฬาลงกรณ์มหาวิทยาลัย, กรุงเทพฯ.

ประภาพร กุลลิ้มรัตน์ชัย. (2551). *การค้นคืนภาพโดยการพิจารณาน้ำหนักการกระจายของสีสำหรับฮิสโตแกรมสีในแบบจำลองสี HSV*. (วิทยานิพนธ์ปริญญาวิศวกรรมศาสตรมหาบัณฑิต). สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง, กรุงเทพฯ.

ปริญญา สงวนสัตย์. (2560). *การจำแนกประเภทข้อความภาษาไทยด้วยการเรียนรู้เชิงลึก*. การประชุมวิชาการระดับชาติปัญญาภิวัฒน์. 7(1). สืบค้นจาก https://conference.pim.ac.th/fileman/Uploads/ET_Presentation_1_Part1.pdf

วฤษาย์ ร่มสายหยุด. (2561). *หลักการวิศวกรรมซอฟต์แวร์*. นนทบุรี : มหาวิทยาลัยสุโขทัยธรรมาธิราช.

วศิน ทับแสง. (2557). *การตรวจจับและรู้จำตราสัญลักษณ์ของรถยนต์โดยใช้วิธีการโครงข่ายประสาทเทียมแบบคอนโวลูชันและพีรามิดฮีสโตแกรมของทิศทางเกรเดียนต์*. (วิทยานิพนธ์ปริญญาวิศวกรรมศาสตรมหาบัณฑิต). มหาวิทยาลัยเชียงใหม่, เชียงใหม่.

สุชาติ คุ้มมะณี. (2561). *Programing expert with Python เชี่ยวชาญการเขียนโปรแกรมด้วยไพธอน*. สืบค้นจาก https://isan.msu.ac.th/suchart/Python/ProgrammingExpertwithPython.pdf

สุทิพา เผือกผ่อง. (2559). *การเรียนรู้เชิงลึกโดยการใช้โครงข่ายประสาทเทียมแบบคอนโวลูชันสำหรับระบบรู้จำภาพใบหน้าบุคคล*. (ปริญญาวิศวกรรมศาสตรมหาบัณฑิต). สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง, กรุงเทพฯ.

อรรณพ ขันธิกุล, (2544). *การจำแนกใบไม้โดยใช้เครือข่ายประสาทเทียมแบบคอนโวลูชันนอล*.

(วิทยานิพนธ์ปริญญาวิศวกรรมศาสตรมหาบัณฑิต). มหาวิทยาลัยเกษตรศาสตร์, กรุงเทพฯ.

อุษนีย์ สังฆธรรม. (2556). *การรู้จำตัวอักษรพิมพ์ไทยโดยการใช้วิธีคอนดิชั่นนัลแรนดอมฟิลด์สและระยะเซนทรอยด์แบบลำดับชั้น*. (วิทยานิพนธ์ปริญญาวิทยาศาสตรมหาบัณฑิต). สถาบันบัณฑิตพัฒนบริหารศาสตร์, กรุงเทพฯ.

Adit Deshpande. (2016). *A Beginning' s Guide to Understanding Convolutional Neural Networks*. Retrieved from https://adeshpande3.github.io/A-Beginner%27s-Guide-To-Understanding-Convolutional-Neural-Networks/

Ariadna Quattoni. (2009). *Transfer Learning Algorithms for Image Classification.* (Doctor's thesis). Massachusetts Institute of Technology.

Chris Dahms. (2018). *Tensor Flow Classification Walk-through.* Retrieved from https://github.com/MicrocontrollersAndMore/TensorFlow_Tut_2_Classification_Walk-through

Ernest Bofill Ylla, (2017). *Synthesizing Images to Recognize Natural Images with Transfer Learning in Convolutional Neural Network.* (Master's thesis). Aalborg University Copenhagen.

Hanna Bjorgvinsdottir and Robin Seibold. (2016). *Face Recognition Based on Embedded Systems.* (Master's thesis). Lund University.

Jacopo Credi. (n.d.). *Traffic sign classification with deep convolutional neural networks.* (Master's thesis). Chalmers university of technology.

Lisa Torrey and Jude Shavlik. (2009). *Transfer Learning*. University of Wisconsin. USA. Retrieved from ftp://ftp.cs.wisc.edu/machine-learning/shavlik-group/torrey.handbook09.pdf

Martin Thoma. (2017). *Analysis and Optimization of Convolutional Neural Network Architectures*. (Master' s thesis). KIT University.

Muhind Salim Hmoud Alradad, (2015). *Robust classification with convolutional neural network.* (Master's thesis). Missouri-Columbia University.

Ondrej Zapletal. (2017). *Image recognition by convolutional neural networks – basic concepts.* (Master's thesis). Brno University of Technology.

Qinghui Liu. (2017). *Deep Learning Applied to Automatic Polyp Detection in Colonoscopy Images*. (Master's thesis). University College of Southeast Norway.

Stephen Cass. (2018). *The 2018 Top Programming Languages*. Retrieved from https://spectrum.ieee.org/at-work/innovation/the-2018-top-programming-languages

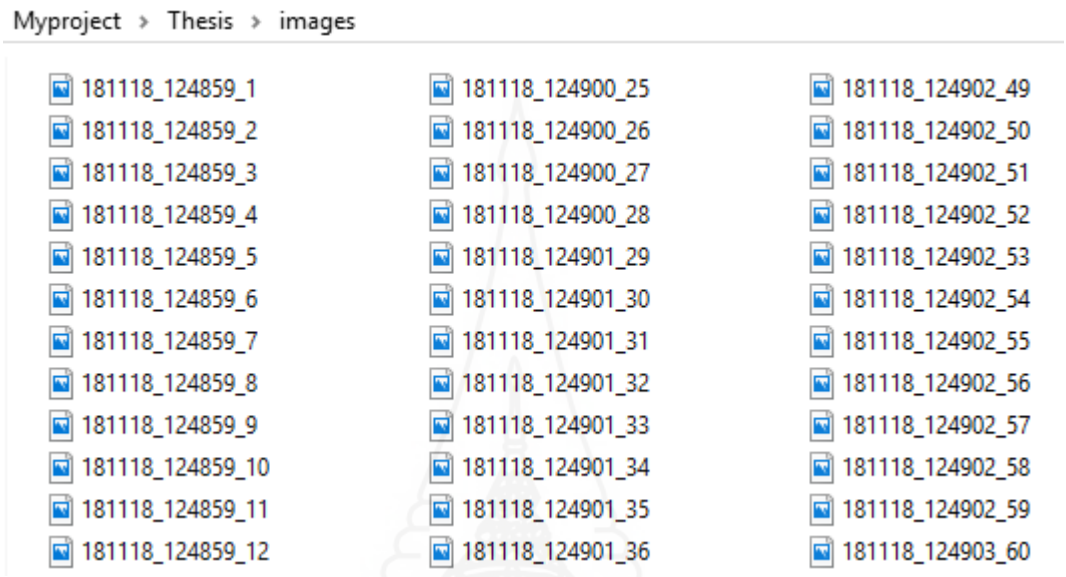
ภาคผนวก

ภาคผนวก ก

รหัสโปรแกรม

รหัสโปรแกรมของไฟล์ create_image.py

ผลลัพธ์การทำงานของรหัสโปรแกรม create_image.py



รหัสโปรแกรมการสร้างไฟล์รูปภาพอะไหล่เพื่อเก็บรูปภาพของอะไหล่แบบอัตโนมัติ ผู้ดูแลระบบมีหน้าที่หมุนชิ้นอะไหล่หรือหมุนกล้องเพื่อให้กล้องจับภาพอะไหล่ได้หลากหลายมุมมองเพื่อเป็นข้อมูลนำเข้าในการฝึกฝนข้อมูลของตัวแบบ

```
# create_image.py
import os
import cv2
import time
CRTD_IMAGES_DIR = os.getcwd() + "/images"

def main():
    print("starting program . . .")
    SampleId=0
    cam = cv2.VideoCapture(1) #0 internal webcam, 1 external webcam via usb port
    time.sleep(5)  # Wait 5 sec for camera setting up
    while (cam.isOpened()):
```

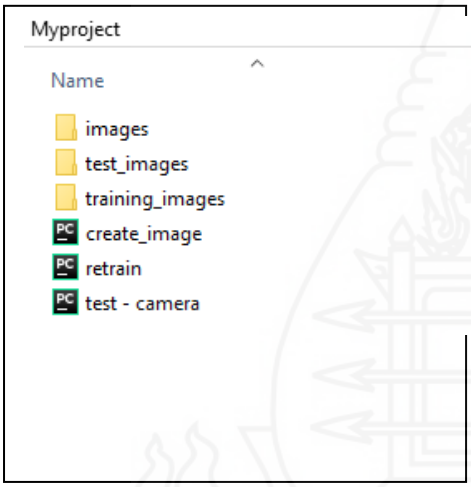
```
        ret, img = cam.read()
        SampleId=SampleId + 1
        cv2.imshow("Creating image", img)
        cv2.imwrite(CRTD_IMAGES_DIR + "/" + time.strftime("%y%m%d_%H%M%S",
time.gmtime()) + "_" + str(SampleId) + ".jpg", img)  # Update new image
        if cv2.waitKey(25) & 0xFF == ord('q') or SampleId >= 700:
            print("----- Finished -----")
            break
    cam.release()
    cv2.destroyAllWindows()

if __name__ == "__main__":
    main()
```

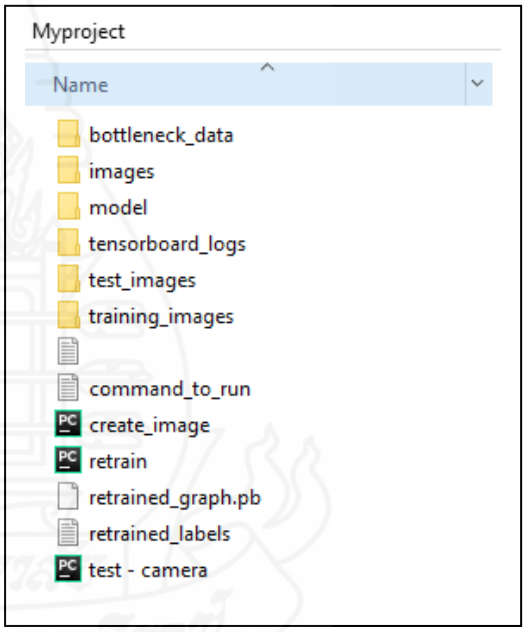
รหัสโปรแกรมของไฟล์ retrain.py

ผลลัพธ์การทำงานของรหัสโปรแกรม retrain.py โปรแกรมจะสร้างแฟ้มข้อมูลและไฟล์เพื่อสนับสนุนการทำงานของกระบวนการฝึกฝนข้อมูลและสร้างตัวแบบเพื่อนำไปใช้งานกรณีนี้จะเกิดขึ้นเมื่อรันโปรแกรมครั้งแรก สำหรับการฝึกฝนข้อมูลในครั้งต่อไปจะปรับปรุงเฉพาะในส่วนของแฟ้มข้อมูลของ training_images เพื่อทำการลบหรือเพิ่มข้อมูลภาพของอะไหล่และทำการฝึกฝนเพื่อให้ระบบรู้จำต่อไป

ภาพก่อนรันโปรแกรม



ภาพหลังรันโปรแกรม




```
# retrain.py
import hashlib
import os
import os.path
import random
import re
```

```
import sys
import tarfile
import numpy as np
import tensorflow as tf
from six.moves import urllib
from datetime import datetime
from tensorflow.contrib.quantize.python import quant_ops
from tensorflow.python.framework import graph_util
from tensorflow.python.framework import tensor_shape
from tensorflow.python.platform import gfile
from tensorflow.python.util import compat
MIN_NUM_IMAGES_REQUIRED_FOR_TRAINING = 10
MIN_NUM_IMAGES_SUGGESTED_FOR_TRAINING = 100
MIN_NUM_IMAGES_REQUIRED_FOR_TESTING = 3
MAX_NUM_IMAGES_PER_CLASS = 2 ** 27 - 1
TRAINING_IMAGES_DIR = os.getcwd() + '/training_images'
TEST_IMAGES_DIR = os.getcwd() + "/test_images/"
OUTPUT_GRAPH = os.getcwd() + '/' + 'retrained_graph.pb'
INTERMEDIATE_OUTPUT_GRAPHS_DIR = os.getcwd() + '/intermediate_graph'
INTERMEDIATE_STORE_FREQUENCY = 0
OUTPUT_LABELS = os.getcwd() + '/' + 'retrained_labels.txt'
TENSORBOARD_DIR = os.getcwd() + '/' + 'tensorboard_logs'
HOW_MANY_TRAINING_STEPS=10000
LEARNING_RATE = 0.01
TESTING_PERCENTAGE = 10
VALIDATION_PERCENTAGE = 10
EVAL_STEP_INTERVAL = 10
TRAIN_BATCH_SIZE = 100
TEST_BATCH_SIZE = -1
VALIDATION_BATCH_SIZE = 100
```

```
PRINT_MISCLASSIFIED_TEST_IMAGES = False
MODEL_DIR = os.getcwd() + "/" + "model"
BOTTLENECK_DIR = os.getcwd() + '/' + 'bottleneck_data'
FINAL_TENSOR_NAME = 'final_result'
FLIP_LEFT_RIGHT = False
RANDOM_CROP = 0
RANDOM_SCALE = 0
RANDOM_BRIGHTNESS = 0
ARCHITECTURE = 'inception_v3'


def main():
    print("starting program . . .")
    tf.logging.set_verbosity(tf.logging.INFO)
    if not checkIfNecessaryPathsAndFilesExist():
        return
    prepare_file_system()
    model_info = create_model_info(ARCHITECTURE)
    if not model_info:
        tf.logging.error('Did not recognize architecture flag')
        return -1
    print("downloading model (if necessary) . . .")
    downloadModelIfNotAlreadyPresent(model_info['data_url'])
    print("creating model graph . . .")
    graph, bottleneck_tensor, resized_image_tensor = (create_model_graph(model_info))
    print("creating image lists . . .")
    image_lists = create_image_lists(TRAINING_IMAGES_DIR, TESTING_PERCENTAGE,
VALIDATION_PERCENTAGE)
    class_count = len(image_lists.keys())
    if class_count == 0:
        tf.logging.error('No valid folders of images found at ' + TRAINING_IMAGES_DIR)
```

```
        return -1
    if class_count == 1:
        tf.logging.error('Only one valid folder of images found at ' + TRAINING_IMAGES_DIR + '
- multiple classes are needed for classification.')
        return -1
    doDistortImages = False
    if (FLIP_LEFT_RIGHT == True or RANDOM_CROP != 0 or RANDOM_SCALE != 0 or
RANDOM_BRIGHTNESS != 0):
        doDistortImages = True
    print("starting session . . .")
    with tf.Session(graph=graph) as sess:
        print("performing jpeg decoding . . .")
        jpeg_data_tensor, decoded_image_tensor = add_jpeg_decoding( model_info['input_width'],
model_info['input_height'], model_info['input_depth'], model_info['input_mean'],
model_info['input_std'])
        print("caching bottlenecks . . .")
        distorted_jpeg_data_tensor = None
        distorted_image_tensor = None
        if doDistortImages:
            (distorted_jpeg_data_tensor, distorted_image_tensor) =
add_input_distortions(FLIP_LEFT_RIGHT, RANDOM_CROP, RANDOM_SCALE,
RANDOM_BRIGHTNESS, model_info['input_width'],model_info['input_height'],
model_info['input_depth'],model_info['input_mean'], model_info['input_std'])
        else:
            cache_bottlenecks(sess, image_lists, TRAINING_IMAGES_DIR, BOTTLENECK_DIR,
jpeg_data_tensor, decoded_image_tensor, resized_image_tensor, bottleneck_tensor,
ARCHITECTURE)
        print("adding final training layer . . .")
        (train_step, cross_entropy, bottleneck_input, ground_truth_input, final_tensor) =
add_final_training_ops(len(image_lists.keys()),FINAL_TENSOR_NAME, bottleneck_tensor,
```

```
model_info['bottleneck_tensor_size'], model_info['quantize_layer'])
    print("adding eval ops for final training layer . . .")
    evaluation_step, prediction = add_evaluation_step(final_tensor, ground_truth_input)
    print("writing TensorBoard info . . .")
    merged = tf.summary.merge_all()
    train_writer = tf.summary.FileWriter(TENSORBOARD_DIR + '/train', sess.graph)
    validation_writer = tf.summary.FileWriter(TENSORBOARD_DIR + '/validation')
    init = tf.global_variables_initializer()
    sess.run(init)
    print("performing training . . .")
    for i in range(HOW_MANY_TRAINING_STEPS):
      if doDistortImages:
        (train_bottlenecks, train_ground_truth) = get_random_distorted_bottlenecks(sess,
image_lists, TRAIN_BATCH_SIZE, 'training', TRAINING_IMAGES_DIR,
distorted_jpeg_data_tensor, distorted_image_tensor, resized_image_tensor, bottleneck_tensor)
      else:
        (train_bottlenecks, train_ground_truth, _) = get_random_cached_bottlenecks(sess,
image_lists, TRAIN_BATCH_SIZE, 'training',
BOTTLENECK_DIR, TRAINING_IMAGES_DIR, jpeg_data_tensor,
decoded_image_tensor, resized_image_tensor, bottleneck_tensor,
ARCHITECTURE)
      train_summary, _ = sess.run([merged, train_step], feed_dict={bottleneck_input:
train_bottlenecks, ground_truth_input: train_ground_truth})
      train_writer.add_summary(train_summary, i)
      is_last_step = (i + 1 == HOW_MANY_TRAINING_STEPS)
      if (i % EVAL_STEP_INTERVAL) == 0 or is_last_step:
        train_accuracy, cross_entropy_value = sess.run([evaluation_step, cross_entropy],
feed_dict={bottleneck_input: train_bottlenecks, ground_truth_input: train_ground_truth})
        tf.logging.info('%s: Step %d: Train accuracy = %.1f%%' % (datetime.now(), i,
train_accuracy * 100))
```

```
            tf.logging.info('%s: Step %d: Cross entropy = %f' % (datetime.now(), i,
cross_entropy_value))
            validation_bottlenecks, validation_ground_truth, _ =
(get_random_cached_bottlenecks(sess, image_lists, VALIDATION_BATCH_SIZE, 'validation',
BOTTLENECK_DIR, TRAINING_IMAGES_DIR, jpeg_data_tensor,
decoded_image_tensor, resized_image_tensor, bottleneck_tensor,
ARCHITECTURE))
            validation_summary, validation_accuracy = sess.run([merged, evaluation_step],
feed_dict={bottleneck_input: validation_bottlenecks, ground_truth_input:
validation_ground_truth})
            validation_writer.add_summary(validation_summary, i)
            tf.logging.info('%s: Step %d: Validation accuracy = %.1f%% (N=%d)' %
(datetime.now(), i, validation_accuracy * 100, len(validation_bottlenecks)))
        intermediate_frequency = INTERMEDIATE_STORE_FREQUENCY
        if (intermediate_frequency > 0 and (i % intermediate_frequency == 0) and i > 0):
            intermediate_file_name = (INTERMEDIATE_OUTPUT_GRAPHS_DIR +
'intermediate_' + str(i) + '.pb')
            tf.logging.info('Save intermediate result to : ' + intermediate_file_name)
            save_graph_to_file(sess, graph, intermediate_file_name)
    print("running testing . . .")
    test_bottlenecks, test_ground_truth, test_filenames = (get_random_cached_bottlenecks(sess,
image_lists, TEST_BATCH_SIZE, 'testing', BOTTLENECK_DIR,
TRAINING_IMAGES_DIR, jpeg_data_tensor, decoded_image_tensor, resized_image_tensor,
bottleneck_tensor, ARCHITECTURE))
    test_accuracy, predictions = sess.run([evaluation_step, prediction],
feed_dict={bottleneck_input: test_bottlenecks, ground_truth_input: test_ground_truth})
    tf.logging.info('Final test accuracy = %.1f%% (N=%d)' % (test_accuracy * 100,
len(test_bottlenecks)))
    if PRINT_MISCLASSIFIED_TEST_IMAGES:
        tf.logging.info('=== MISCLASSIFIED TEST IMAGES ===')
```

```
            for i, test_filename in enumerate(test_filenames):
                if predictions[i] != test_ground_truth[i]:
                    tf.logging.info('%70s  %s' % (test_filename, list(image_lists.keys())[predictions[i]]))
        print("writing trained graph and labbels with weights")
        save_graph_to_file(sess, graph, OUTPUT_GRAPH)
        with gfile.FastGFile(OUTPUT_LABELS, 'w') as f:
            f.write('\n'.join(image_lists.keys()) + '\n')
        print("Finished")


def checkIfNecessaryPathsAndFilesExist():
    if not os.path.exists(TRAINING_IMAGES_DIR):
        print('')
        print('ERROR: TRAINING_IMAGES_DIR "' + TRAINING_IMAGES_DIR + '" does not
seem to exist')
        print('Did you set up the training images?')
        print('')
        return False
    class TrainingSubDir:


        def __init__(self):
            self.loc = ""
            self.numImages = 0
    trainingSubDirs = []
    for dirName in os.listdir(TRAINING_IMAGES_DIR):
        currentTrainingImagesSubDir = os.path.join(TRAINING_IMAGES_DIR, dirName)
        if os.path.isdir(currentTrainingImagesSubDir):
            trainingSubDir = TrainingSubDir()
            trainingSubDir.loc = currentTrainingImagesSubDir
            trainingSubDirs.append(trainingSubDir)
    if len(trainingSubDirs) == 0:
```

```
        print("ERROR: there don't seem to be any training image sub-directories in " +
TRAINING_IMAGES_DIR)
        print("Did you make a separate image sub-directory for each classification type?")
        return False
    for trainingSubDir in trainingSubDirs:
        for fileName in os.listdir(trainingSubDir.loc):
            if fileName.endswith(".jpg") or fileName.endswith(".JPG"):
                trainingSubDir.numImages += 1
    for trainingSubDir in trainingSubDirs:
        if trainingSubDir.numImages < MIN_NUM_IMAGES_REQUIRED_FOR_TRAINING:
            print("ERROR: there are less than the required " +
str(MIN_NUM_IMAGES_REQUIRED_FOR_TRAINING) + " images in " + trainingSubDir.loc)
            print("Did you populate each training sub-directory with images?")
            return False
    for trainingSubDir in trainingSubDirs:
        if trainingSubDir.numImages < MIN_NUM_IMAGES_SUGGESTED_FOR_TRAINING:
            print("WARNING: there are less than the suggested " +
str(MIN_NUM_IMAGES_SUGGESTED_FOR_TRAINING) + " images in " +
trainingSubDir.loc)
            print("More images should be added to this directory for acceptable training results")
    if not os.path.exists(TEST_IMAGES_DIR):
        print('')
        print('ERROR: TEST_IMAGES_DIR "' + TEST_IMAGES_DIR + '" does not seem to exist')
        print('Did you break out some test images?')
        print('')
        return False
    numImagesInTestDir = 0
    for fileName in os.listdir(TEST_IMAGES_DIR):
        if fileName.endswith(".jpg") or fileName.endswith(".JPG"):
            numImagesInTestDir += 1
```

```
    if numImagesInTestDir < MIN_NUM_IMAGES_REQUIRED_FOR_TESTING:
        print("ERROR: there are not at least " +
str(MIN_NUM_IMAGES_REQUIRED_FOR_TESTING) + " images in " +
TEST_IMAGES_DIR)
        print("Did you break out some test images?")
        return False
    return True


def prepare_file_system():
    if tf.gfile.Exists(TENSORBOARD_DIR):
        tf.gfile.DeleteRecursively(TENSORBOARD_DIR)
    tf.gfile.MakeDirs(TENSORBOARD_DIR)
    if INTERMEDIATE_STORE_FREQUENCY > 0:
        makeDirIfDoesNotExist(INTERMEDIATE_OUTPUT_GRAPHS_DIR)
    return


def makeDirIfDoesNotExist(dir_name):
    if not os.path.exists(dir_name):
        os.makedirs(dir_name)


def create_model_info(architecture):
    architecture = architecture.lower()
    is_quantized = False
    if architecture == 'inception_v3':
        data_url = 'http://download.tensorflow.org/models/image/imagenet/inception-2015-12-
05.tgz'
        bottleneck_tensor_name = 'pool_3/_reshape:0'
        bottleneck_tensor_size = 2048
        input_width = 299
        input_height = 299
```

```
        input_depth = 3
        resized_input_tensor_name = 'Mul:0'
        model_file_name = 'classify_image_graph_def.pb'
        input_mean = 128
        input_std = 128
    elif architecture.startswith('mobilenet_'):
        parts = architecture.split('_')
        if len(parts) != 3 and len(parts) != 4:
            tf.logging.error("Couldn't understand architecture name '%s'", architecture)
            return None
        version_string = parts[1]
        if (version_string != '1.0' and version_string != '0.75' and version_string != '0.50' and
version_string != '0.25'):
            tf.logging.error("""The Mobilenet version should be '1.0', '0.75', '0.50', or '0.25', but
found '%s' for architecture '%s'""", version_string, architecture)
            return None
        size_string = parts[2]
        if (size_string != '224' and size_string != '192' and size_string != '160' and size_string !=
'128'):
            tf.logging.error("""The Mobilenet input size should be '224', '192', '160', or '128', but
found '%s' for architecture '%s'""", size_string, architecture)
            return None
        if len(parts) == 3:
            is_quantized = False
        else:
            if parts[3] != 'quantized':
                tf.logging.error(
                    "Couldn't understand architecture suffix '%s' for '%s'", parts[3], architecture)
                return None
            is_quantized = True
```

```
        if is_quantized:
            data_url = 'http://download.tensorflow.org/models/mobilenet_v1_'
            data_url += version_string + '_' + size_string + '_quantized_frozen.tgz'
            bottleneck_tensor_name = 'MobilenetV1/Predictions/Reshape:0'
            resized_input_tensor_name = 'Placeholder:0'
            model_dir_name = ('mobilenet_v1_' + version_string + '_' + size_string +
'_quantized_frozen')
            model_base_name = 'quantized_frozen_graph.pb'
        else:
            data_url = 'http://download.tensorflow.org/models/mobilenet_v1_'
            data_url += version_string + '_' + size_string + '_frozen.tgz'
            bottleneck_tensor_name = 'MobilenetV1/Predictions/Reshape:0'
            resized_input_tensor_name = 'input:0'
            model_dir_name = 'mobilenet_v1_' + version_string + '_' + size_string
            model_base_name = 'frozen_graph.pb'
        bottleneck_tensor_size = 1001
        input_width = int(size_string)
        input_height = int(size_string)
        input_depth = 3
        model_file_name = os.path.join(model_dir_name, model_base_name)
        input_mean = 127.5
        input_std = 127.5
    else:
        tf.logging.error("Couldn't understand architecture name '%s'", architecture)
        raise ValueError('Unknown architecture', architecture)
    return {'data_url': data_url, 'bottleneck_tensor_name': bottleneck_tensor_name,
'bottleneck_tensor_size': bottleneck_tensor_size,
            'input_width': input_width, 'input_height': input_height, 'input_depth': input_depth,
'resized_input_tensor_name': resized_input_tensor_name,
```

```
        'model_file_name': model_file_name, 'input_mean': input_mean, 'input_std': input_std,
'quantize_layer': is_quantized, }

def downloadModelIfNotAlreadyPresent(data_url):
    dest_directory = MODEL_DIR
    if not os.path.exists(dest_directory):
        os.makedirs(dest_directory)
    filename = data_url.split('/')[-1]
    filepath = os.path.join(dest_directory, filename)
    if not os.path.exists(filepath):
        def _progress(count, block_size, total_size):
            sys.stdout.write('\r>> Downloading %s %.1f%%' % (filename, float(count * block_size) /
float(total_size) * 100.0))
            sys.stdout.flush()
        filepath, _ = urllib.request.urlretrieve(data_url, filepath, _progress)
        print()
        statinfo = os.stat(filepath)
        tf.logging.info('Successfully downloaded ' + str(filename) + ', statinfo.st_size = ' +
str(statinfo.st_size) + ' bytes')
        print('Extracting file from ', filepath)
        tarfile.open(filepath, 'r:gz').extractall(dest_directory)
    else:
        print('Not extracting or downloading files, model already present in disk')

def create_model_graph(model_info):
    with tf.Graph().as_default() as graph:
        model_path = os.path.join(MODEL_DIR, model_info['model_file_name'])
        print('Model path: ', model_path)
        with gfile.FastGFile(model_path, 'rb') as f:
            graph_def = tf.GraphDef()
```

```
        graph_def.ParseFromString(f.read())
        bottleneck_tensor, resized_input_tensor = (tf.import_graph_def(graph_def, name='',
return_elements=[model_info['bottleneck_tensor_name'],
model_info['resized_input_tensor_name'],]))
    return graph, bottleneck_tensor, resized_input_tensor


def create_image_lists(image_dir, testing_percentage, validation_percentage):
    if not gfile.Exists(image_dir):
        tf.logging.error("Image directory '" + image_dir + "' not found.")
        return None
    result = {}
    sub_dirs = [x[0] for x in gfile.Walk(image_dir)]
    is_root_dir = True
    for sub_dir in sub_dirs:
        if is_root_dir:
            is_root_dir = False
            continue
        dir_name = os.path.basename(sub_dir)
        if dir_name == image_dir:
            continue
        extensions = ['jpg', 'jpeg']
        file_list = []
        tf.logging.info("Looking for images in '" + dir_name + "'")
        for extension in extensions:
            file_glob = os.path.join(image_dir, dir_name, '*.' + extension)
            file_list.extend(gfile.Glob(file_glob))
        if not file_list:
            tf.logging.warning('No files found')
            continue
        if len(file_list) < 20:
```

```
            tf.logging.warning('WARNING: Folder has less than 20 images, which may cause
issues.')
        elif len(file_list) > MAX_NUM_IMAGES_PER_CLASS:
            tf.logging.warning('WARNING: Folder {} has more than {} images. Some images will
never be selected.'.format(dir_name, MAX_NUM_IMAGES_PER_CLASS))
        label_name = re.sub(r'[^a-z0-9]+', ' ', dir_name.lower())
        training_images = []
        testing_images = []
        validation_images = []
        for file_name in file_list:
            base_name = os.path.basename(file_name)
            hash_name = re.sub(r'_nohash_.*$', '', file_name)
            hash_name_hashed = hashlib.sha1(compat.as_bytes(hash_name)).hexdigest()
            percentage_hash = ((int(hash_name_hashed, 16) %
(MAX_NUM_IMAGES_PER_CLASS + 1)) * (100.0 / MAX_NUM_IMAGES_PER_CLASS))
            if percentage_hash < validation_percentage:
                validation_images.append(base_name)
            elif percentage_hash < (testing_percentage + validation_percentage):
                testing_images.append(base_name)
            else:
                training_images.append(base_name)
        result[label_name] = {'dir': dir_name, 'training': training_images, 'testing': testing_images,
'validation': validation_images,}
    return result

def add_jpeg_decoding(input_width, input_height, input_depth, input_mean, input_std):
    jpeg_data = tf.placeholder(tf.string, name='DecodeJPGInput')
    decoded_image = tf.image.decode_jpeg(jpeg_data, channels=input_depth)
    decoded_image_as_float = tf.cast(decoded_image, dtype=tf.float32)
    decoded_image_4d = tf.expand_dims(decoded_image_as_float, 0)
```

```
    resize_shape = tf.stack([input_height, input_width])
    resize_shape_as_int = tf.cast(resize_shape, dtype=tf.int32)
    resized_image = tf.image.resize_bilinear(decoded_image_4d, resize_shape_as_int)
    offset_image = tf.subtract(resized_image, input_mean)
    mul_image = tf.multiply(offset_image, 1.0 / input_std)
    return jpeg_data, mul_image


def add_input_distortions(flip_left_right, random_crop, random_scale, random_brightness,
input_width, input_height,
                          input_depth, input_mean, input_std):
    jpeg_data = tf.placeholder(tf.string, name='DistortJPGInput')
    decoded_image = tf.image.decode_jpeg(jpeg_data, channels=input_depth)
    decoded_image_as_float = tf.cast(decoded_image, dtype=tf.float32)
    decoded_image_4d = tf.expand_dims(decoded_image_as_float, 0)
    margin_scale = 1.0 + (random_crop / 100.0)
    resize_scale = 1.0 + (random_scale / 100.0)
    margin_scale_value = tf.constant(margin_scale)
    resize_scale_value = tf.random_uniform(tensor_shape.scalar(), minval=1.0,
maxval=resize_scale)
    scale_value = tf.multiply(margin_scale_value, resize_scale_value)
    precrop_width = tf.multiply(scale_value, input_width)
    precrop_height = tf.multiply(scale_value, input_height)
    precrop_shape = tf.stack([precrop_height, precrop_width])
    precrop_shape_as_int = tf.cast(precrop_shape, dtype=tf.int32)
    precropped_image = tf.image.resize_bilinear(decoded_image_4d, precrop_shape_as_int)
    precropped_image_3d = tf.squeeze(precropped_image, squeeze_dims=[0])
    cropped_image = tf.random_crop(precropped_image_3d, [input_height, input_width,
input_depth])
    if flip_left_right:
        flipped_image = tf.image.random_flip_left_right(cropped_image)
```

```
    else:
        flipped_image = cropped_image
    brightness_min = 1.0 - (random_brightness / 100.0)
    brightness_max = 1.0 + (random_brightness / 100.0)
    brightness_value = tf.random_uniform(tensor_shape.scalar(), minval=brightness_min,
maxval=brightness_max)
    brightened_image = tf.multiply(flipped_image, brightness_value)
    offset_image = tf.subtract(brightened_image, input_mean)
    mul_image = tf.multiply(offset_image, 1.0 / input_std)
    distort_result = tf.expand_dims(mul_image, 0, name='DistortResult')
    return jpeg_data, distort_result


def cache_bottlenecks(sess, image_lists, image_dir, bottleneck_dir, jpeg_data_tensor,
decoded_image_tensor,
                      resized_input_tensor, bottleneck_tensor, architecture):
    how_many_bottlenecks = 0
    makeDirIfDoesNotExist(bottleneck_dir)
    for label_name, label_lists in image_lists.items():
        for category in ['training', 'testing', 'validation']:
            category_list = label_lists[category]
            for index, unused_base_name in enumerate(category_list):
                get_or_create_bottleneck(sess, image_lists, label_name, index, image_dir, category,
bottleneck_dir, jpeg_data_tensor, decoded_image_tensor, resized_input_tensor,
bottleneck_tensor, architecture)
            how_many_bottlenecks += 1
            if how_many_bottlenecks % 100 == 0:
                tf.logging.info(str(how_many_bottlenecks) + ' bottleneck files created.')
```

```
def get_or_create_bottleneck(sess, image_lists, label_name, index, image_dir, category,
bottleneck_dir, jpeg_data_tensor, decoded_image_tensor, resized_input_tensor,
bottleneck_tensor, architecture):
    label_lists = image_lists[label_name]
    sub_dir = label_lists['dir']
    sub_dir_path = os.path.join(bottleneck_dir, sub_dir)
    makeDirIfDoesNotExist(sub_dir_path)
    bottleneck_path = get_bottleneck_path(image_lists, label_name, index, bottleneck_dir,
category, architecture)
    if not os.path.exists(bottleneck_path):
        create_bottleneck_file(bottleneck_path, image_lists, label_name, index, image_dir, category,
sess, jpeg_data_tensor, decoded_image_tensor, resized_input_tensor, bottleneck_tensor)
    with open(bottleneck_path, 'r') as bottleneck_file:
        bottleneckBigString = bottleneck_file.read()
    bottleneckValues = []
    errorOccurred = False
    try:
        bottleneckValues = [float(individualString) for individualString in
bottleneckBigString.split(',')]
    except ValueError:
        tf.logging.warning('Invalid float found, recreating bottleneck')
        errorOccurred = True
    if errorOccurred:
        create_bottleneck_file(bottleneck_path, image_lists, label_name, index, image_dir, category,
sess, jpeg_data_tensor, decoded_image_tensor, resized_input_tensor, bottleneck_tensor)
        with open(bottleneck_path, 'r') as bottleneck_file:
            bottleneckBigString = bottleneck_file.read()
        bottleneckValues = [float(individualString) for individualString in
bottleneckBigString.split(',')]
    return bottleneckValues
```

```
def get_bottleneck_path(image_lists, label_name, index, bottleneck_dir, category, architecture):
    return get_image_path(image_lists, label_name, index, bottleneck_dir, category) + '_' +
architecture + '.txt'

def create_bottleneck_file(bottleneck_path, image_lists, label_name, index,
image_dir, category, sess, jpeg_data_tensor, decoded_image_tensor, resized_input_tensor,
bottleneck_tensor):
    tf.logging.info('Creating bottleneck at ' + bottleneck_path)
    image_path = get_image_path(image_lists, label_name, index, image_dir, category)
    if not gfile.Exists(image_path):
        tf.logging.fatal('File does not exist %s', image_path)
    image_data = gfile.FastGFile(image_path, 'rb').read()
    try:
        bottleneck_values = run_bottleneck_on_image(sess, image_data, jpeg_data_tensor,
decoded_image_tensor, resized_input_tensor, bottleneck_tensor)
    except Exception as e:
        raise RuntimeError('Error during processing file %s (%s)' % (image_path, str(e)))
    bottleneck_string = ','.join(str(x) for x in bottleneck_values)
    with open(bottleneck_path, 'w') as bottleneck_file:
        bottleneck_file.write(bottleneck_string)

def run_bottleneck_on_image(sess, image_data, image_data_tensor, decoded_image_tensor,
resized_input_tensor, bottleneck_tensor):
    resized_input_values = sess.run(decoded_image_tensor, {image_data_tensor: image_data})
    bottleneck_values = sess.run(bottleneck_tensor, {resized_input_tensor: resized_input_values})
    bottleneck_values = np.squeeze(bottleneck_values)
    return bottleneck_values

def get_image_path(image_lists, label_name, index, image_dir, category):
```

```
    if label_name not in image_lists:
        tf.logging.fatal('Label does not exist %s.', label_name)
    label_lists = image_lists[label_name]
    if category not in label_lists:
        tf.logging.fatal('Category does not exist %s.', category)
    category_list = label_lists[category]
    if not category_list:
        tf.logging.fatal('Label %s has no images in the category %s.', label_name, category)
    mod_index = index % len(category_list)
    base_name = category_list[mod_index]
    sub_dir = label_lists['dir']
    full_path = os.path.join(image_dir, sub_dir, base_name)
    return full_path


def add_final_training_ops(class_count, final_tensor_name, bottleneck_tensor,
bottleneck_tensor_size, quantize_layer):
    with tf.name_scope('input'):
        bottleneck_input = tf.placeholder_with_default(bottleneck_tensor, shape=[None,
bottleneck_tensor_size], name='BottleneckInputPlaceholder')
        ground_truth_input = tf.placeholder(tf.int64, [None], name='GroundTruthInput')
    layer_name = 'final_training_ops'
    with tf.name_scope(layer_name):
        quantized_layer_weights = None
        quantized_layer_biases = None
        with tf.name_scope('weights'):
            initial_value = tf.truncated_normal([bottleneck_tensor_size, class_count], stddev=0.001)
            layer_weights = tf.Variable(initial_value, name='final_weights')
            if quantize_layer:
                quantized_layer_weights = quant_ops.MovingAvgQuantize(layer_weights,
is_training=True)
```

```
            attachTensorBoardSummaries(quantized_layer_weights)
        attachTensorBoardSummaries(layer_weights)
    with tf.name_scope('biases'):
        layer_biases = tf.Variable(tf.zeros([class_count]), name='final_biases')
        if quantize_layer:
            quantized_layer_biases = quant_ops.MovingAvgQuantize(layer_biases,
is_training=True)
            attachTensorBoardSummaries(quantized_layer_biases)
        attachTensorBoardSummaries(layer_biases)
    with tf.name_scope('Wx_plus_b'):
        if quantize_layer:
            logits = tf.matmul(bottleneck_input, quantized_layer_weights) +
quantized_layer_biases
            logits = quant_ops.MovingAvgQuantize(logits, init_min=-32.0, init_max=32.0,
is_training=True, num_bits=8, narrow_range=False, ema_decay=0.5)
tf.summary.histogram('pre_activations', logits)
        else:
            logits = tf.matmul(bottleneck_input, layer_weights) + layer_biases
            tf.summary.histogram('pre_activations', logits)
  final_tensor = tf.nn.softmax(logits, name=final_tensor_name)
  tf.summary.histogram('activations', final_tensor)
  with tf.name_scope('cross_entropy'):
    cross_entropy_mean = tf.losses.sparse_softmax_cross_entropy(labels=ground_truth_input,
logits=logits)
  tf.summary.scalar('cross_entropy', cross_entropy_mean)
  with tf.name_scope('train'):
    optimizer = tf.train.GradientDescentOptimizer(LEARNING_RATE)
    train_step = optimizer.minimize(cross_entropy_mean)
  return (train_step, cross_entropy_mean, bottleneck_input, ground_truth_input, final_tensor)
```

```
def attachTensorBoardSummaries(var):
    with tf.name_scope('summaries'):
        mean = tf.reduce_mean(var)
        tf.summary.scalar('mean', mean)
        with tf.name_scope('stddev'):
            stddev = tf.sqrt(tf.reduce_mean(tf.square(var - mean)))
        tf.summary.scalar('stddev', stddev)
        tf.summary.scalar('max', tf.reduce_max(var))
        tf.summary.scalar('min', tf.reduce_min(var))
        tf.summary.histogram('histogram', var)


def add_evaluation_step(result_tensor, ground_truth_tensor):
    with tf.name_scope('accuracy'):
        with tf.name_scope('correct_prediction'):
            prediction = tf.argmax(result_tensor, 1)
            correct_prediction = tf.equal(prediction, ground_truth_tensor)
        with tf.name_scope('accuracy'):
            evaluation_step = tf.reduce_mean(tf.cast(correct_prediction, tf.float32))
    tf.summary.scalar('accuracy', evaluation_step)
    return evaluation_step, prediction


def get_random_distorted_bottlenecks(sess, image_lists, how_many, category, image_dir,
input_jpeg_tensor, distorted_image,  resized_input_tensor, bottleneck_tensor):
    class_count = len(image_lists.keys())
    bottlenecks = []
    ground_truths = []
    for unused_i in range(how_many):
        label_index = random.randrange(class_count)
        label_name = list(image_lists.keys())[label_index]
        image_index = random.randrange(MAX_NUM_IMAGES_PER_CLASS + 1)
```

```
      image_path = get_image_path(image_lists, label_name, image_index, image_dir, category)
      if not gfile.Exists(image_path):
        tf.logging.fatal('File does not exist %s', image_path)
      jpeg_data = gfile.FastGFile(image_path, 'rb').read()
      distorted_image_data = sess.run(distorted_image, {input_jpeg_tensor: jpeg_data})
      bottleneck_values = sess.run(bottleneck_tensor, {resized_input_tensor:
distorted_image_data})
      bottleneck_values = np.squeeze(bottleneck_values)
      bottlenecks.append(bottleneck_values)
      ground_truths.append(label_index)
  return bottlenecks, ground_truths


def get_random_cached_bottlenecks(sess, image_lists, how_many, category, bottleneck_dir,
image_dir, jpeg_data_tensor,
                                  decoded_image_tensor, resized_input_tensor, bottleneck_tensor,
architecture):
  class_count = len(image_lists.keys())
  bottlenecks = []
  ground_truths = []
  filenames = []
  if how_many >= 0:
    for unused_i in range(how_many):
      label_index = random.randrange(class_count)
      label_name = list(image_lists.keys())[label_index]
      image_index = random.randrange(MAX_NUM_IMAGES_PER_CLASS + 1)
      image_name = get_image_path(image_lists, label_name, image_index, image_dir,
category)
      bottleneck = get_or_create_bottleneck(sess, image_lists, label_name, image_index,
image_dir, category, bottleneck_dir, jpeg_data_tensor, decoded_image_tensor,
resized_input_tensor, bottleneck_tensor, architecture)
```

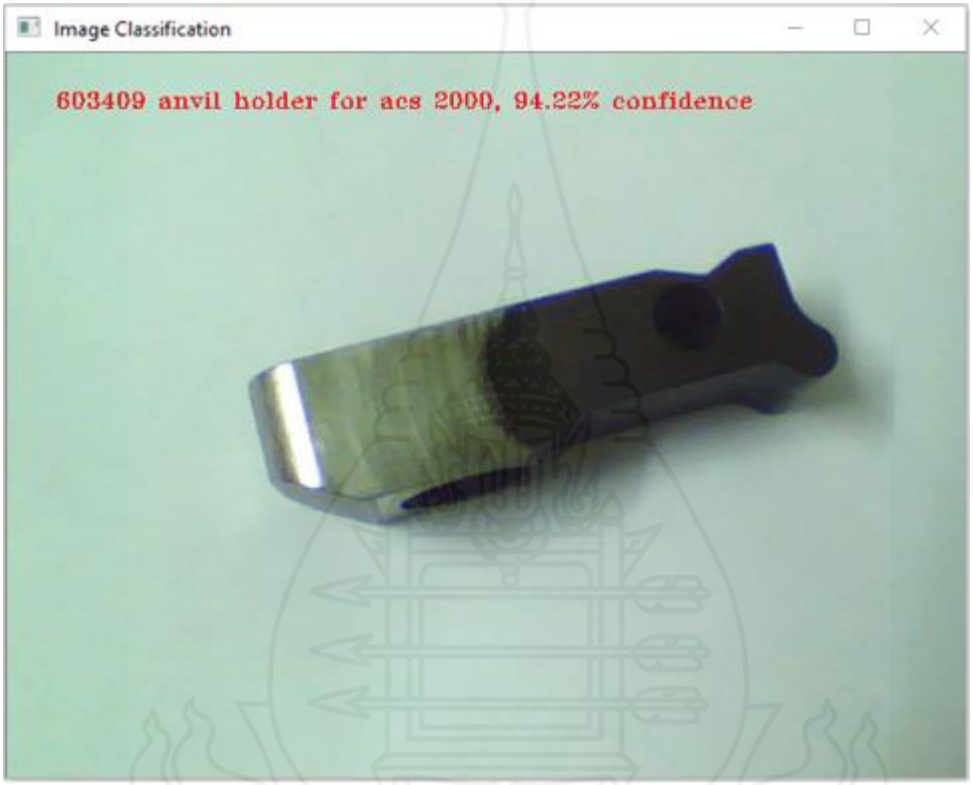
```
            bottlenecks.append(bottleneck)
            ground_truths.append(label_index)
            filenames.append(image_name)
    else:
        for label_index, label_name in enumerate(image_lists.keys()):
            for image_index, image_name in enumerate(image_lists[label_name][category]):
                image_name = get_image_path(image_lists, label_name, image_index, image_dir,
category)
                bottleneck = get_or_create_bottleneck(sess, image_lists, label_name, image_index,
image_dir, category, bottleneck_dir, jpeg_data_tensor, decoded_image_tensor,
resized_input_tensor, bottleneck_tensor, architecture)
                bottlenecks.append(bottleneck)
                ground_truths.append(label_index)
                filenames.append(image_name)
    return bottlenecks, ground_truths, filenames


def save_graph_to_file(sess, graph, graph_file_name):
    output_graph_def = graph_util.convert_variables_to_constants(sess, graph.as_graph_def(),
[FINAL_TENSOR_NAME])
    with gfile.FastGFile(graph_file_name, 'wb') as f:
        f.write(output_graph_def.SerializeToString())
    return


if __name__ == '__main__':
    main()
```

รหัสโปรแกรมของไฟล์ test – camera.py

ผลการทำงานของรหัสโปรแกรม test-camera.py เป็นนำข้อมูลภาพการนำเข้าจากกล้อง เพื่อให้ตัวแบบที่สร้างขึ้นมาทำการพยากรณ์รหัสและชื่อของอะไหล่ พร้อมแสดงค่าความน่าจะเป็นในรูปแบบเรียลไทม์




```
# test – camera.py
import os
import tensorflow as tf
import numpy as np
import cv2
RETRAINED_LABELS_TXT_FILE_LOC = os.getcwd() + "/" + "retrained_labels.txt"
RETRAINED_GRAPH_PB_FILE_LOC = os.getcwd() + "/" + "retrained_graph.pb"
TEST_IMAGES_DIR = os.getcwd() + "/test_images"
```

```
SCALAR_RED = (0.0, 0.0, 255.0)
SCALAR_BLUE = (255.0, 0.0, 0.0)
def main():
    print("starting program . . .")
    if not checkIfNecessaryPathsAndFilesExist():
        return
    classifications = []
    for currentLine in tf.gfile.GFile(RETRAINED_LABELS_TXT_FILE_LOC):
        classification = currentLine.rstrip()
        classifications.append(classification)
    print("classifications = " + str(classifications))
    with tf.gfile.FastGFile(RETRAINED_GRAPH_PB_FILE_LOC, 'rb') as retrainedGraphFile:
        graphDef = tf.GraphDef()
        graphDef.ParseFromString(retrainedGraphFile.read())
        _ = tf.import_graph_def(graphDef, name='')
    if not os.path.isdir(TEST_IMAGES_DIR):
        print("the test image directory does not seem to be a valid directory, check file / directory
paths")
        return
    cam = cv2.VideoCapture(1)
    while (cam.isOpened()):
        ret, img = cam.read()
        if ret == True:
            with tf.Session() as sess:
                openCVImage = img
                if openCVImage is None:
                    print("unable to open 'unknown.jpg' as an OpenCV image")
                    continue
                finalTensor = sess.graph.get_tensor_by_name('final_result:0')
                tfImage = np.array(openCVImage)[:, :, 0:3]
```

```
            predictions = sess.run(finalTensor, {'DecodeJpeg:0': tfImage})
            sortedPredictions = predictions[0].argsort()[-len(predictions[0]):][::-1]
            print("--------------------------------------")
            onMostLikelyPrediction = True
            for prediction in sortedPredictions:
                strClassification = classifications[prediction]
                if strClassification.endswith("s"):
                    strClassification = strClassification[:-1]
                confidence = predictions[0][prediction]
                if onMostLikelyPrediction:
                    scoreAsAPercent = confidence * 100.0
                    print("the object appears to be a " + strClassification + ", " + "{0:.2f}".format(
                        scoreAsAPercent) + "% confidence")
                    writeResultOnImage(openCVImage, strClassification + ", " + "{0:.2f}".format(
                        scoreAsAPercent) + "% confidence", scoreAsAPercent)
                    cv2.imshow("Image Classification", openCVImage)
                    onMostLikelyPrediction = False
                print(strClassification + " (" + "{0:.5f}".format(confidence) + ")")
            if cv2.waitKey(25) & 0xFF == ord('q'):
                break
        else:
            break
    cam.release()
    cv2.destroyAllWindows()

def checkIfNecessaryPathsAndFilesExist():
    if not os.path.exists(TEST_IMAGES_DIR):
        print('')
        print('ERROR: TEST_IMAGES_DIR "' + TEST_IMAGES_DIR + '" does not seem to exist')
        print('Did you set up the test images?')
```

```
        print('')
        return False
    if not os.path.exists(RETRAINED_LABELS_TXT_FILE_LOC):
        print('ERROR: RETRAINED_LABELS_TXT_FILE_LOC "' +
RETRAINED_LABELS_TXT_FILE_LOC + '" does not seem to exist')
        return False
    if not os.path.exists(RETRAINED_GRAPH_PB_FILE_LOC):
        print('ERROR: RETRAINED_GRAPH_PB_FILE_LOC "' +
RETRAINED_GRAPH_PB_FILE_LOC + '" does not seem to exist')
        return False
    return True

def writeResultOnImage(openCVImage, resultText,confidence):
        imageHeight, imageWidth, sceneNumChannels = openCVImage.shape
        fontFace = cv2.FONT_HERSHEY_TRIPLEX
        fontScale = 0.5
        fontThickness = 1
        fontThickness = int(fontThickness)
        upperLeftTextOriginX = int(imageWidth * 0.05)
        upperLeftTextOriginY = int(imageHeight * 0.05)
        textSize, baseline = cv2.getTextSize(resultText, fontFace, fontScale, fontThickness)
        textSizeWidth, textSizeHeight = textSize
        lowerLeftTextOriginX = upperLeftTextOriginX
        lowerLeftTextOriginY = upperLeftTextOriginY + textSizeHeight
        if confidence > 30:
            cv2.putText(openCVImage, resultText, (lowerLeftTextOriginX, lowerLeftTextOriginY),
fontFace, fontScale, SCALAR_RED, fontThickness)
        else:
            resultText = "Spare part no match"
```

```
        cv2.putText(openCVImage, resultText, (lowerLeftTextOriginX, lowerLeftTextOriginY),
fontFace, fontScale,
                SCALAR_RED, fontThickness)

if __name__ == "__main__":
    main()
```

รหัสโปรแกรมของไฟล์ test-image.py

```
# test-image.py
import os
import tensorflow as tf
import numpy as np
import cv2
RETRAINED_LABELS_TXT_FILE_LOC = os.getcwd() + "/" + "retrained_labels.txt"
RETRAINED_GRAPH_PB_FILE_LOC = os.getcwd() + "/" + "retrained_graph.pb"
TEST_IMAGES_DIR = os.getcwd() + "/test_images"
SCALAR_RED = (0.0, 0.0, 255.0)
SCALAR_BLUE = (255.0, 0.0, 0.0)
def main():
    print("starting program . . .")
    if not checkIfNecessaryPathsAndFilesExist():
        return
    classifications = []
    for currentLine in tf.gfile.GFile(RETRAINED_LABELS_TXT_FILE_LOC):
        classification = currentLine.rstrip()
        classifications.append(classification)
    print("classifications = " + str(classifications))
    with tf.gfile.FastGFile(RETRAINED_GRAPH_PB_FILE_LOC, 'rb') as retrainedGraphFile:
        graphDef = tf.GraphDef()
```

```
        graphDef.ParseFromString(retrainedGraphFile.read())
        _ = tf.import_graph_def(graphDef, name='')
    if not os.path.isdir(TEST_IMAGES_DIR):
        print("the test image directory does not seem to be a valid directory, check file / directory paths")
        return
    with tf.Session() as sess:
        for fileName in os.listdir(TEST_IMAGES_DIR):
            if not (fileName.lower().endswith(".jpg") or fileName.lower().endswith(".jpeg")):
                continue
            print(fileName)
            imageFileWithPath = os.path.join(TEST_IMAGES_DIR, fileName)
            openCVImage = cv2.imread(imageFileWithPath)
            if openCVImage is None:
                print("unable to open " + fileName + " as an OpenCV image")
                continue
            finalTensor = sess.graph.get_tensor_by_name('final_result:0')
            tfImage = np.array(openCVImage)[:, :, 0:3]
            predictions = sess.run(finalTensor, {'DecodeJpeg:0': tfImage})
            sortedPredictions = predictions[0].argsort()[-len(predictions[0]):][::-1]
            print("--------------------------------------")
            onMostLikelyPrediction = True
            for prediction in sortedPredictions:
                strClassification = classifications[prediction]
                if strClassification.endswith("s"):
                    strClassification = strClassification[:-1]
                confidence = predictions[0][prediction]
                if onMostLikelyPrediction:
                    scoreAsAPercent = confidence * 100.0
```

```
                print("the object appears to be a " + strClassification + ", " +
"{0:.2f}".format(scoreAsAPercent) + "% confidence")
                writeResultOnImage(openCVImage, strClassification + ", " +
"{0:.2f}".format(scoreAsAPercent) + "% confidence")
                cv2.imshow(fileName, openCVImage)
                onMostLikelyPrediction = False
            print(strClassification + " (" +  "{0:.5f}".format(confidence) + ")")
        cv2.waitKey()
        cv2.destroyAllWindows()
    tfFileWriter = tf.summary.FileWriter(os.getcwd())
    tfFileWriter.add_graph(sess.graph)
    tfFileWriter.close()

def checkIfNecessaryPathsAndFilesExist():
    if not os.path.exists(TEST_IMAGES_DIR):
        print('')
        print('ERROR: TEST_IMAGES_DIR "' + TEST_IMAGES_DIR + '" does not seem to exist')
        print('Did you set up the test images?')
        print('')
        return False
    if not os.path.exists(RETRAINED_LABELS_TXT_FILE_LOC):
        print('ERROR: RETRAINED_LABELS_TXT_FILE_LOC "' +
RETRAINED_LABELS_TXT_FILE_LOC + '" does not seem to exist')
        return False
    if not os.path.exists(RETRAINED_GRAPH_PB_FILE_LOC):
        print('ERROR: RETRAINED_GRAPH_PB_FILE_LOC "' +
RETRAINED_GRAPH_PB_FILE_LOC + '" does not seem to exist')
        return False
    return True
```

```
def writeResultOnImage(openCVImage, resultText):
    imageHeight, imageWidth, sceneNumChannels = openCVImage.shape
    fontFace = cv2.FONT_HERSHEY_TRIPLEX
    fontScale = 1.0
    fontThickness = 2
    fontThickness = int(fontThickness)
    upperLeftTextOriginX = int(imageWidth * 0.05)
    upperLeftTextOriginY = int(imageHeight * 0.05)
    textSize, baseline = cv2.getTextSize(resultText, fontFace, fontScale, fontThickness)
    textSizeWidth, textSizeHeight = textSize
    lowerLeftTextOriginX = upperLeftTextOriginX
    lowerLeftTextOriginY = upperLeftTextOriginY + textSizeHeight
    cv2.putText(openCVImage, resultText, (lowerLeftTextOriginX, lowerLeftTextOriginY),
fontFace, fontScale, SCALAR_RED, fontThickness)

if __name__ == "__main__":
    main()
```

ประวัติผู้วิจัย

ชื่อ	นายอุเทน ว่องไว
วัน เดือน ปีเกิด	2 พฤษภาคม 2522
สถานที่เกิด	อำเภอสอง จังหวัดแพร่
ประวัติการศึกษา	ปริญญาตรี บริหารธุรกิจบัณฑิต มหาวิทยาลัยธนบุรี 2557
สถานที่ทำงาน	บริษัทช้าฟเนอร์ อีเอ็มซี จำกัด นิคมอุตสาหกรรม จังหวัดลำพูน
ตำแหน่ง	เจ้าหน้าที่ฝ่ายวางแผน