บทที่ 2
ทฤษฎีและงานวิจัยที่เกี่ยวข้อง

ในการทำปริญญานิพนธ์นี้ คณะผู้จัดทำได้ศึกษาค้นคว้างานวิจัยและทฤษฎีต่าง ๆ ที่เกี่ยวข้อง เพื่อให้การทำปริญญานิพนธ์มีความสมบูรณ์ครบถ้วน เป็นไปตามขอบเขตและวัตถุประสงค์ โดยทฤษฎีและงานวิจัยที่เกี่ยวข้องที่คณะผู้จัดทำได้ศึกษามีตามลำดับ ดังนี้

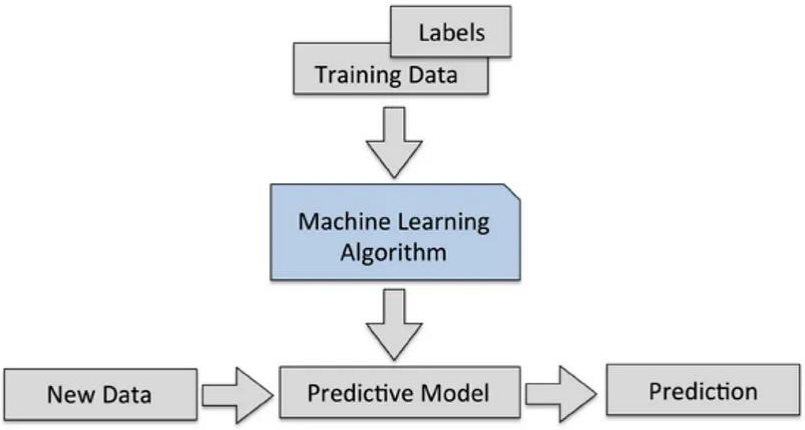
2.1 การเรียนรู้ของเครื่อง (Machine Learning) [1]

การเรียนรู้ของเครื่อง (Machine Learning) คือ รูปแบบหนึ่งของการวิเคราะห์ข้อมูล ที่ดำเนินการวิเคราะห์ด้วยแบบจำลองอย่างเป็นอัตโนมัติ ซึ่งตั้งอยู่บนรากฐานแนวคิดว่า ระบบต่าง ๆ นั้น สามารถที่จะเรียนรู้และมีปฏิสัมพันธ์กับชุดข้อมูลต่าง ๆ รวมถึงสามารถระบุ และทราบรูปแบบต่าง ๆ ที่เกิดขึ้น นำไปสู่การตัดสินใจได้เองโดยไม่จำเป็นต้องพึ่งพามนุษย์ โดยมีรูปแบบการเรียนรู้ 3 ประเภท ดังนี้

2.1.1 การเรียนรู้แบบมีผู้สอน (Supervised Learning)

รูปแบบการเรียนรู้แบบมีผู้สอน จะอยู่ในลักษณะการทำนายผลลัพธ์ การทำให้คอมพิวเตอร์สามารถหาคำตอบของปัญหาได้ด้วยตัวเองหลังจากที่ได้เรียนรู้จากชุดข้อมูลตัวอย่างที่ได้ฝึกฝน

โดยในชุดข้อมูลตัวอย่างที่ใช้ฝึก (Training Data) จะมีมนุษย์คอยทำป้ายกำกับแยกประเภทหรือบอกผลลัพธ์ที่ถูกต้อง (Label) จากนั้นจะนำข้อมูลที่ใช้ฝึกที่ผ่านการแยกประเภทแล้ว ไปฝึกฝน (Train) ผ่านอัลกอริทึมสำหรับสร้างโมเดล (Machine Learning Algorithm) เพื่อทำนายผลลัพธ์ จากนั้นเมื่อได้โมเดลสำหรับทำนายผลลัพธ์มาแล้ว (Predictive Model) มนุษย์จะนำข้อมูลใหม่ (New Data) ที่คอมพิวเตอร์ไม่เคยเห็นมาให้เครื่องทำนาย (Predict) ว่าคำตอบที่ได้ควรจะเป็นอย่างไร ดังรูปที่ 2.1

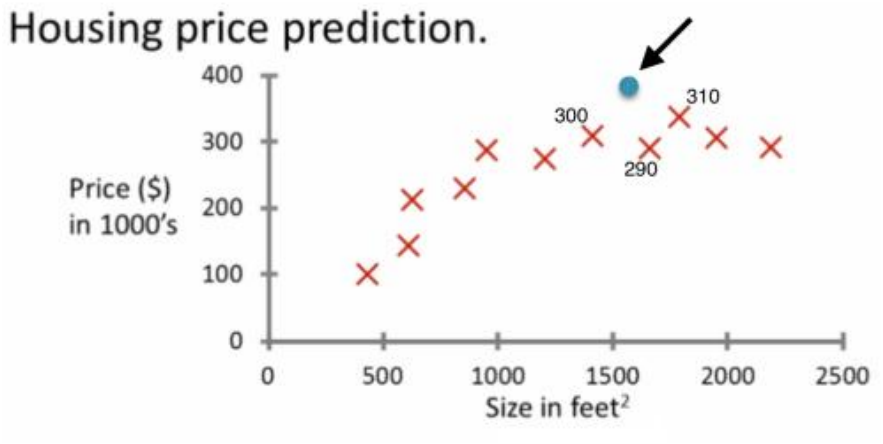


รูปที่ 2.1 การทำงานของการเรียนรู้แบบมีผู้สอน (Supervised Learning) [1]

โดยการเรียนรู้แบบมีผู้สอนแบ่งออกเป็น 2 ประเภท ดังนี้

1) การถดถอย (Regression)

เป็นการนำเอาข้อมูลที่เก็บไว้ในอดีต มาทำนายแนวโน้มของข้อมูลที่จะเกิดขึ้นในอนาคต โดยใช้รูปแบบสมการเชิงเส้น (Linear) โดยใช้วิธีการหาความสัมพันธ์ของตัวแปร 2 ตัวขึ้นไป และจะต้องระบุว่าตัวแปรใด คือ ตัวแปรต้น และตัวแปรใด คือ ตัวแปรตาม

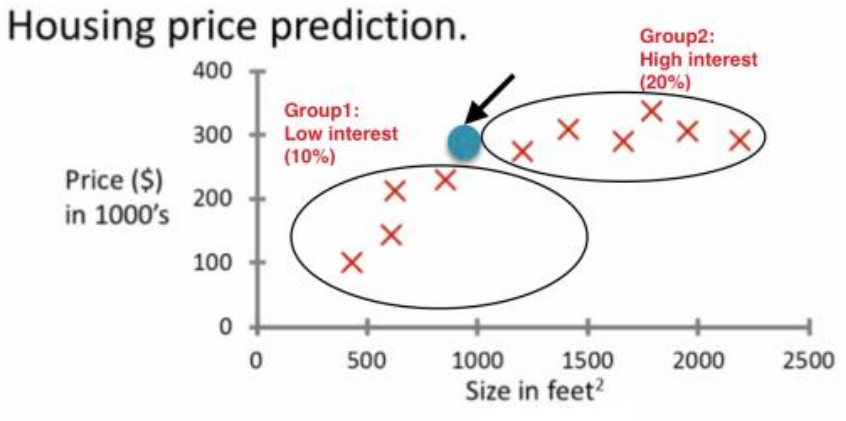


รูปที่ 2.2 การหาความสัมพันธ์ระหว่างราคาบ้านกับขนาดของบ้านด้วยการถดถอย [2]

จากรูปที่ 2.2 [2] มีข้อมูลสอนที่เป็นขนาดขอองบ้านและราคาของบ้าน 11 หลัง ซึ่งแทนด้วยเครื่องหมายกากบาทสีแดง และแกน X คือ ขนาดของบ้าน แกน Y คือ ราคาของบ้านหลังนั้น ๆ ข้อมูลเหล่านี้เป็นข้อมูลที่มีอยู่ก่อนแล้ว ซึ่งจะถือเป็นข้อมูลสอน ถ้ามีบ้านหลังที่ 12 เพิ่มเข้ามา มีขนาด 1600 ตารางฟุต แสดงด้วยจุดสีฟ้า สิ่งที่การถดถอย (Regression) ต้องทำก็คือ จะสามารถหาได้อย่างไรว่าราคาที่เหมาะสมของบ้านหลังนี้ควรจะเป็นเท่าไหร่

2) การจำแนกข้อมูล (Classification)

การจำแนกหรือการแบ่งประเภทข้อมูล โดยหาต้นแบบหรือสำรวจจุดเด่นจุดด้อยที่ปรากฏอยู่ภายในชุดข้อมูล ด้วยการใช้ข้อมูลจำนวนหนึ่งในการสร้างต้นแบบ ซึ่งต้นแบบนั้นจะสามารถนำไปใช้ในการกำหนดประเภทของชุดข้อมูลว่าควรมีกี่ประเภท



รูปที่ 2.3 การหาความสัมพันธ์ระหว่างราคาบ้านกับขนาดของบ้านด้วยการจำแนกข้อมูล [2]

จากรูปที่ 2.3 [2] เป็นข้อมูลเดียวกับที่ยกตัวอย่างไปในการถดถอย (Regression) ซึ่งก่อนที่จะทำงาน ต้องมีการจัดกลุ่มข้อมูลสอนก่อน เช่น การจัดกลุ่มของบ้านที่เสียภาษีต่ำกับเสียภาษีสูงก่อน ดังรูป จากนั้นสิ่งที่การจำแนกข้อมูล (Classification) จะต้องทำคือ เมื่อมีข้อมูลของบ้านหลังที่ 12 เข้ามา ซึ่งเป็นจุดสีฟ้า โมเดลจะต้องจัดให้ได้ว่าบ้านหลังใหม่นี้ต้องเสียภาษีในอัตราที่ต่ำหรือสูง

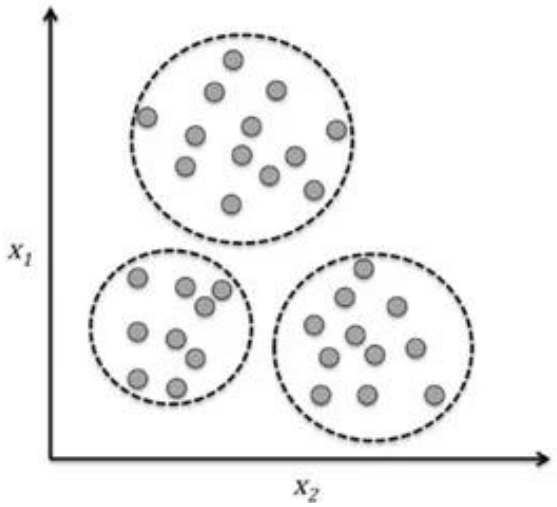
2.1.2 การเรียนรู้แบบไม่มีผู้สอน (Unsupervised Learning)

รูปแบบการเรียนรู้แบบไม่มีผู้สอน คือ การที่คอมพิวเตอร์จะทำการวิเคราะห์และเรียนรู้จากข้อมูลโดยข้อมูลนั้นไม่ป้ายกำกับ (Label) หรือคำตอบที่ชัดเจน แต่คอมพิวเตอร์จะค้นหารูปแบบ โครงสร้าง หรือความสัมพันธ์ในข้อมูลด้วยตัวเอง ใช้กรณีที่ไม่มีข้อมูลที่ถูกแบ่งประเภทไว้

โดยการเรียนรู้แบบไม่มีผู้สอนแบ่งออกเป็น 2 ประเภท ดังนี้

1) การจับกลุ่มของข้อมูล (Clustering)

ในชุดข้อมูลที่มีข้อมูลจำนวนมากเกินกว่าที่จะทำป้ายข้อมูล (Label) แต่ทราบว่าในชุดข้อมูลนั้นประกอบด้วยกลุ่มจำนวนเท่าใด แล้วสามารถระบุให้คอมพิวเตอร์แบ่งกลุ่มออกได้ตามจำนวนที่ต้องการ จากนั้นมาระบุเองว่าแต่ละกลุ่มที่คอมพิวเตอร์พบคืออะไร โดยให้ข้อมูลในกลุ่มเดียวกันมีความคล้ายคลึงกันมากที่สุด และข้อมูลในกลุ่มที่ต่างมีความแตกต่างกันมากที่สุด

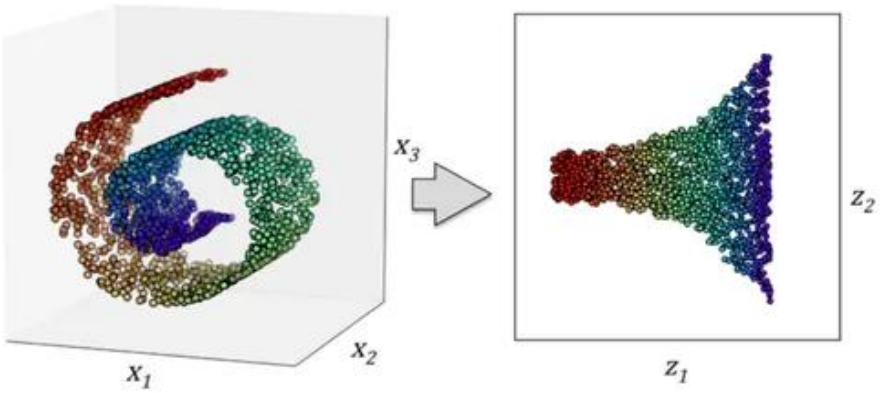


รูปที่ 2.4 การจับกลุ่มข้อมูลจากข้อมูลเข้าสองมิติ [1]

จากรูปที่ 2.4 เป็นภาพตัวอย่างการจับกลุ่มข้อมูลจากข้อมูลเข้าสองมิติ โดยที่ไม่ทราบว่าข้อมูล (จุดสีเทา) ตัวไหนอยู่ในกลุ่มใด แต่สามารถกำหนดให้คอมพิวเตอร์หาวิธีแบ่งกลุ่มข้อมูลออกเป็น 3 กลุ่ม และได้ผลออกมาเป็นขอบเขตของแต่ละกลุ่ม

2) การลดมิติข้อมูล (Dimensionality Reduction)

เป็นการลดจำนวนมิติ (Features) ในข้อมูลขนาดใหญ่ลง เพื่อทำให้การวิเคราะห์หรือการประมวลผลข้อมูลมีประสิทธิภาพมากขึ้น ในขณะเดียวกันก็รักษาข้อมูลสำคัญหรือโครงสร้างหลักของข้อมูลไว้ให้มากที่สุด เป็นกลไกที่ไม่จำเป็นต้องเก็บข้อมูลไว้ครบแต่ก็ยังสามารถจำแนกข้อมูลได้



รูปที่ 2.5 การลดข้อมูลจากสามมิติเหลือสองมิติ [1]

จากรูปที่ 2.5 เป็นตัวอย่างการลดข้อมูลสามมิติของหลาย ๆ คลาสให้เหลือสองมิติ อีกทั้งยังสามารถแยกประเภทคลาสได้ดีเท่าเดิม โดยที่ไม่ทราบว่าจะแปลงข้อมูลสามมิติให้เป็นสองมิติที่ดีได้อย่างไร โดยการจัดเรียงข้อมูลนั้นมีโครงสร้างแบบ Unknown Structure อาจกำหนดเพียงแค่ว่าโมเดลของการเปลี่ยนแปลงข้อมูลเป็นอย่างไร เช่น

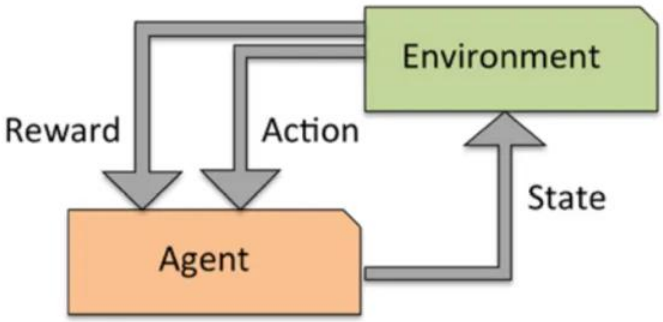
$$z_1 = k_1x_1 + k_2x_2 + k_3x_3 \tag{2.1}$$

จากสมการที่ 2.1 หมายความว่าจะสร้างค่า z ใด ๆ มาจากการรวมกันแบบเชิงเส้นของค่า x ทั้งสามแกน แต่ไม่ทราบว่าค่า k แต่ละตัวควรเป็นเท่าใด และต้องการให้คอมพิวเตอร์หาให้

2.1.3 การเรียนรู้แบบเสริมแรง (Reinforcement Learning)

รูปแบบการเรียนรู้แบบเสริมแรง คือ การฝึกฝนซอฟต์แวร์ (Software) ให้ทำการตัดสินใจเพื่อให้ได้ผลลัพธ์ที่เหมาะสมที่สุด โดยเลียนแบบกระบวนการเรียนรู้แบบลองผิดลองถูกที่มนุษย์ใช้เพื่อบรรลุเป้าหมาย โดยที่คอมพิวเตอร์หรือเอเจนท์ (Agent) ทำการตัดสินใจและดำเนินการ (Action) ใน

สภาพแวดล้อม (Environment) ที่สถานะ (State) ต่างๆ โดยมีเป้าหมายเพื่อเพิ่มผลตอบแทน (Reward) ผ่านการทดลองและปรับปรุงการกระทำตามผลลัพธ์ที่ได้ ดังรูปที่ 2.6



รูปที่ 2.6 การทำงานของการเรียนรู้แบบเสริมแรง [1]

2.2 การเรียนรู้เชิงลึก (Deep Learning) [3]

การเรียนรู้เชิงลึก (Deep Learning) คือ วิธีการเลียนแบบการทำงานของโครงข่ายประสาทมนุษย์ (Neurons) โดยนำระบบโครงข่ายประสาท (Neural Network) มาซ้อนกันหลายชั้น (Layer) และทำการเรียนรู้ข้อมูลตัวอย่างที่ป้อนเข้ามา และทำการประมวลผลอัตโนมัติเพื่อหาข้อมูลตัวอย่างที่จำเป็นในการตรวจจับรูปแบบหรือจัดหมวดหมู่ข้อมูล ความสามารถในการเรียนรู้คุณลักษณะอัตโนมัติ จากนั้นข้อมูลจะถูกนำไปใช้ในการตรวจจับรูปแบบ (Pattern) หรือจัดหมวดหมู่ข้อมูล

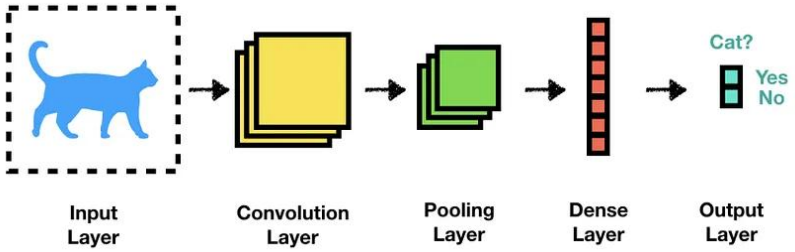
การเรียนรู้เชิงลึกในปริมาณข้อมูลเพิ่มขึ้นที่เหมาะสม ข้อมูลทั้งหมดที่รวบรวมใช้เพื่อให้ได้ผลลัพธ์ที่ถูกต้องผ่านรูปแบบการเรียนรู้แบบวนซ้ำ การวิเคราะห์ข้อมูลขนาดใหญ่ซ้ำ ๆ ช่วยลดข้อผิดพลาดและความคลาดเคลื่อนในการค้นพบและได้ผลลัพธ์ที่น่าเชื่อถือได้ในที่สุด รูปแบบการเรียนรู้นี้มีการนำมาใช้งานอย่างแพร่หลายในทางคอมพิวเตอร์ การรู้จำภาพทั่วไป การประมวลผลภาษาธรรมชาติ (NLP) และการรู้จำเสียงพูด

2.3 โครงข่ายประสาทเทียมแบบคอนโวลูชัน (Convolutional Neural Network : CNN) [3]

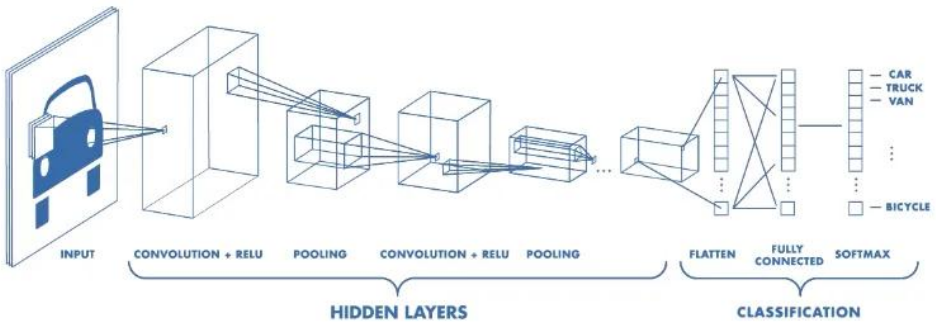
โครงข่ายประสาทเทียมแบบคอนโวลูชัน จัดเป็นการเรียนรู้เชิงลึก (Deep Learning) มีการทำงานโดยเริ่มจากการรับข้อมูลเข้าผ่านชั้นนำข้อมูล (Input Layer) เพื่อส่งเข้าไปทำการประมวลผลโดยหน่วยย่อยของแต่ละชั้น ที่ชั้นประมวลผล (Hidden Layers) และเมื่อประมวลผลเสร็จจนถึงชั้นสุดท้าย จะส่ง

ข้อมูลไปยังชั้นแสดงผล (Output Layer) ซึ่ง CNN สามารถจับคุณสมบัติเด่นของข้อมูล โดยใช้ตัวกรองที่เกี่ยวข้องในการลดจำนวนพารามิเตอร์ของข้อมูล โดยไม่ลดประสิทธิภาพของคุณสมบัติลง

แต่ถ้าหากว่า Deep Learning จะรับข้อมูลดิบเข้าทันที และทำการประมวลผลอัตโนมัติเพื่อหาข้อมูลตัวอย่างที่จำเป็นในการตรวจจับรูปแบบ หรือจัดหมวดหมู่ข้อมูล ความสามารถในการเรียนรู้คุณลักษณะอัตโนมัติทำให้ Deep Learning เป็นประโยชน์อย่างยิ่งสำหรับการใช้งานในสถานการณ์ต่าง ๆ สิ่งที่ท้าทาย คือการหาโครงข่ายระบบประสาทที่เหมาะสม และการค้นหาตัวแปรที่มีผลต่อสมรรถนะในการสอน (Training Performance) ของโครงข่าย ยังคงเป็นเรื่องยากที่จะรู้ได้ว่า Deep Learning สามารถเรียนรู้คุณลักษณะใดบ้าง นอกจากนี้ Deep Learning ยังมีลักษณะไม่ต่างจาก Machine Learning เนื่องจากยังไม่สามารถจัดการข้อมูลรับเข้าที่มีความละเอียดเฉพาะทาง (Carefully Crafted Input) จึงอาจทำให้โมเดลเกิดการอนุมานผิดพลาด (Wrong Interfaces)



รูปที่ 2.7 การทำงานของ Convolutional Neural Network [3]



รูปที่ 2.8 สถาปัตยกรรม Convolutional Neural Network [3]

จากรูปที่ 2.7 และ 2.8 ขั้นตอนการหลักของ Convolutional Neural Network มีทั้งหมด 3 ชั้น ดังนี้

1) Convolutional Layer

เป็นการค้นหาคุณลักษณะที่สำคัญของภาพ โดยขั้นตอนนี้จะใช้ตัวกรอง (Filter หรือ Kernel) เพื่อแยกองค์ประกอบต่าง ๆ ของภาพ เช่น ขอบ สี เป็นต้น โดยปกติภาพจะมีสีหลัก 3 สี (RGB) คือ สีแดง สีน้ำเงิน และสีเขียว แบ่งเป็น 3 Channel และภาพแสดงเป็นเมทริกซ์ 3 มิติ คือ ความกว้าง (Width) ความสูง (Height) และความลึก (Depth) โดยความลึกสอดคล้องกับช่องสี (RGB) ซึ่งแต่ละพิกเซลบอกความเข้มของสี ตั้งแต่ 0 ถึง 255 และทำการ Convolution เพื่อเก็บค่าไว้ในเมทริกซ์ชุดใหม่ที่เรียกว่า Feature Map

Stride เป็นตัวกำหนดว่าเราจะเลื่อนตัวกรอง (Filter) ไปด้วย Step เท่าไร สามารถกำหนดค่าของ Stride ให้มากขึ้นได้ ถ้าต้องการให้การคำนวณหาคุณลักษณะมีพื้นที่ทับซ้อนกันน้อยลง เมื่อทำการ Convolution จะทำให้ภาพมีขนาดเล็กลง จึงทำการ Padding ให้ Output มีขนาดใหญ่ขึ้นและทำให้เล็กลงในขั้นตอนของ Pooling Layer แทน

การ Convolution ในการส่งข้อมูล โดยแต่ละครั้งใช้ตัวกรองที่แตกต่างกัน ทำให้มีการรวม Feature Map ที่ได้ทั้งหมดเข้าด้วยกัน และแสดงผลข้อมูลสุดท้ายใน Convolution Layer ด้วยการค้นหาองค์ประกอบของภาพจากการทำงานของ CNN ทำได้ด้วยสมการ ดังนี้

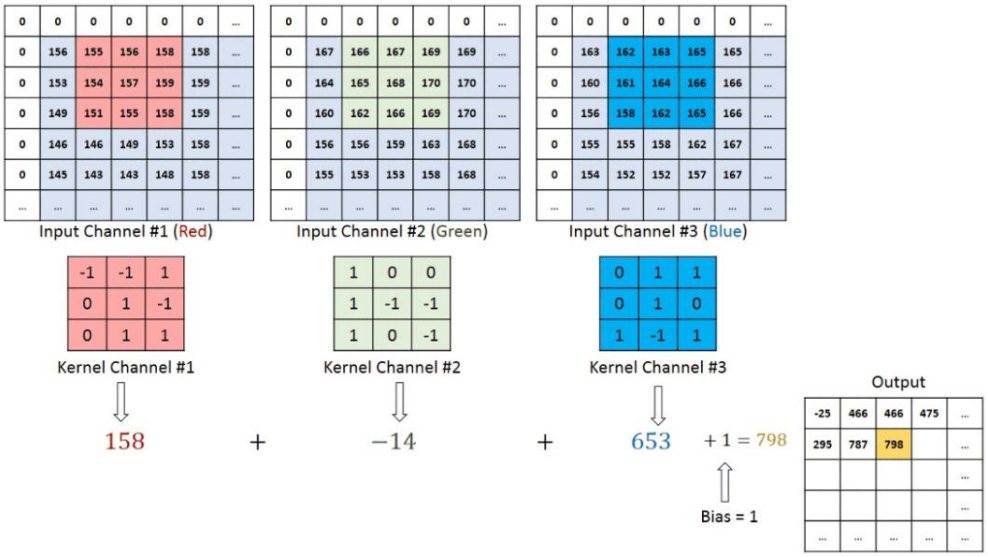
$$Output\ of\ size = Output\ Feature\ Map = \frac{W-F+2P}{S} + 1 \qquad (2.2)$$

โดยที่ W คือ ขนาดของภาพ

F คือ ขนาดของ Filter

P คือ ขนาดของ Padding

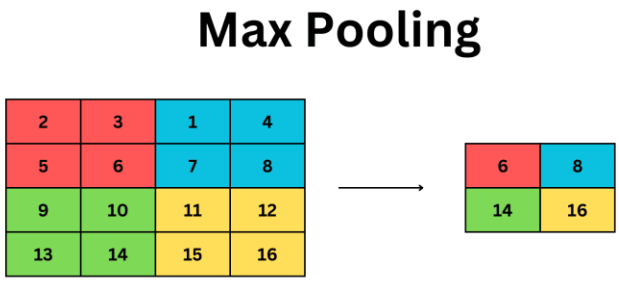
S คือ จำนวนการ Stride



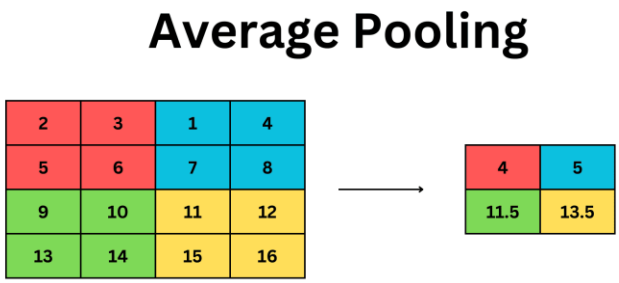
รูปที่ 2.9 การ Convolution บนเมทริกซ์ MxNx3 ที่มีเคอร์เนล 3x3x3

2) Pooling Layer

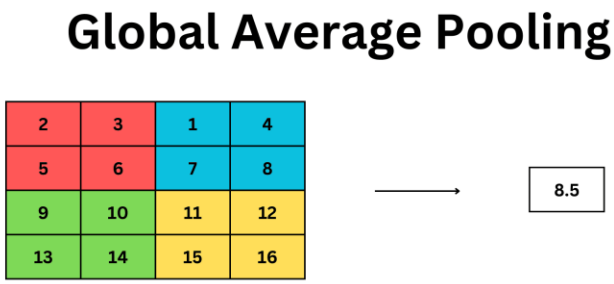
เป็นการลดขนาดของ Feature Map ให้มีขนาดเล็กลงแต่ยังคงรักษาคุณสมบัติของข้อมูลที่สำคัญไว้ โดยมีการหาส่วนที่สำคัญที่สุดของข้อมูล และเพิ่มประสิทธิภาพการประมวลผลรวดเร็วมากขึ้น ด้วยการหาค่าที่สูงที่สุด (Max Pooling) และการหาค่าเฉลี่ย (Average Pooling) ของส่วนที่ครอบคลุมด้วยตัวกรองเก็บไว้ หรือการหาค่าเฉลี่ยรวมของ Feature Map 1 รายการ (Global Average Pooling : GAP)



รูปที่ 2.10 การทำ Max Pooling



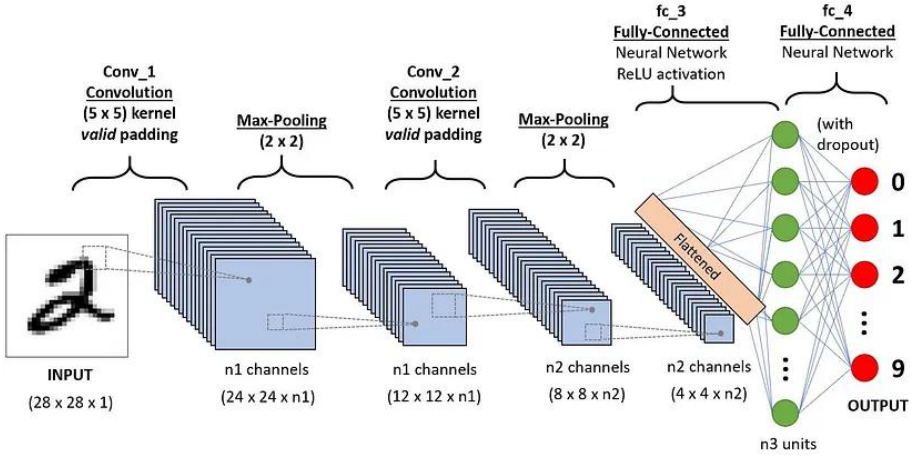
รูปที่ 2.11 การทำ Average Pooling



รูปที่ 2.12 การทำ Global Average Pooling

3) Fully Connected Layer

เป็นการเชื่อมต่อกันของแต่ละชั้นอย่างสมบูรณ์ จากกระบวนการ 2 ขั้นตอน คือ Convolution Layer ทำการสกัดคุณลักษณะที่สำคัญของภาพ และ Pooling Layer ทำการลดมิติของ Feature Map ให้มีขนาดเล็กลงแต่ยังคงรักษาคุณสมบัติของข้อมูลที่สำคัญไว้ ซึ่งในขั้นตอนนี้จะเป็นการทำซ้ำตั้งแต่ Convolution Layer ไปจนถึง Pooling Layer จนกว่าจะเกิดการเชื่อมต่อกันอย่างสมบูรณ์ และทำกระบวนการ 2 ขั้นตอน ด้วย Activation Functions และ Loss Function เพื่อนำลักษณะเด่นที่สำคัญของภาพที่ได้มาทำการสร้างเป็น Neural Network สำหรับการเรียนรู้และทำนายประเภทของภาพ



รูปที่ 2.13 CNN เพื่อจำแนกตัวเลขที่เขียนด้วยลายมือ [3]

ก่อนจะได้ผลลัพธ์การทำนาย ต้องนำค่าตัวเลขผ่านขั้นตอนรับผลรวมการประมวลผลทั้งหมดออกมาเป็นค่าความน่าจะเป็นด้วยฟังก์ชัน Activation Functions ด้วยสมการ ดังนี้ [4]

$$Softmax(x) = S(x) = \frac{e^{x}i}{\sum_{i=1}^{k} e^{x_k}} \ , i = 1, \dots , k \tag{2.3}$$

โดยที่ x_i คือ ค่าที่ Input

เมื่อคำนวณครบทุกค่า ผลลัพธ์จะได้ตั้งแต่ 0 ถึง 1 ถ้าทั้งหมดรวมกันจะเท่ากับ 1 เสมอ

$$Sigmoid = S(x) = \frac{1}{1+ e^{-x}} \tag{2.4}$$

โดยที่ x คือ ค่าที่ Input

เมื่อคำนวณครบทุกค่า ผลลัพธ์จะได้ตั้งแต่ 0 ถึง 1

$$Tanh = S(x) = \frac{e^{x} - e^{-x}}{e^{x} + e^{-x}} \quad (2.5)$$

โดยที่ x คือ ค่าที่ Input
เมื่อคำนวณครบทุกค่า ผลลัพธ์จะได้ตั้งแต่ -1 ถึง 1

$$ReLU = S(x) = \max(o, x) \quad (2.6)$$

โดยที่ x คือ ค่าที่ Input
เมื่อคำนวณครบทุกค่า ผลลัพธ์จะได้ตั้งแต่ 0 ถึง ∞
โดยที่ถ้าค่าต่ำกว่า 0 จะได้ Output เป็น 0 ทันที
หากค่ามากกว่าหรือเท่ากับ 0 จะได้ Output ที่มากกว่า 0

Cross Entropy Loss เป็น Loss Function สำหรับทดสอบ Model แบบ Classification เป็นการนำความน่าจะเป็นของผลลัพธ์ (Output probabilities : P) และประมาณค่าจากค่าความจริง (Actual) เป็นการเรียนรู้และทำนายผลลัพธ์ที่ได้ออกมา ค่าที่คำนวณจะแทนค่าดังสมการ

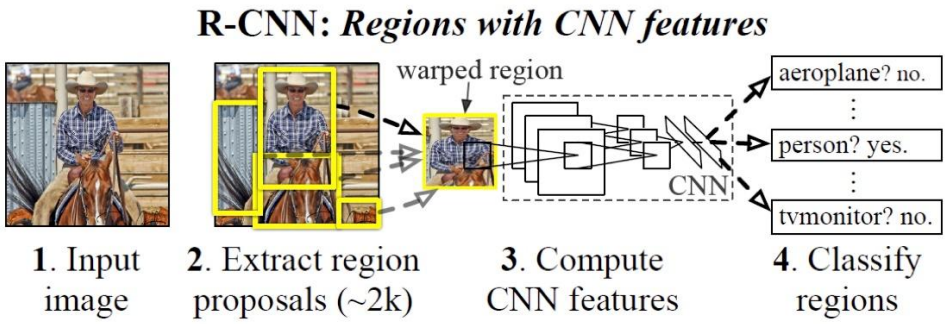
$$Cross\ Entropy\ Loss = CE = -\sum_{i=1}^{N} y_i \cdot log\hat{y}_i \quad (2.7)$$

โดยที่ N คือ จำนวนคลาสทั้งหมด
y_i คือ ค่าความน่าจะเป็นสำหรับคลาสที่ i ในค่าความจริง (Actual)
$\hat{y}_i$ คือ ค่าความน่าจะเป็นสำหรับคลาสที่ i ในค่าที่ทำนาย (Prediction)
log คือ ค่า log ฐาน 2

2.4 โครงข่ายประสาทเทียมแบบคอนโวลูชันแบบเสนอพื้นที่ (Region-based Convolutional Neural Networks : R-CNN) [5]

โครงข่ายประสาทเทียมแบบคอนโวลูชันเสนอพื้นที่ เป็นอัลกอริทึมในการตรวจจับวัตถุ (Object Detection) โดยใช้โครงข่ายประสาทเทียมแบบคอนโวลูชัน (Convolution Neural Network : CNN) เป็นหลักในการสกัดลักษณะของภาพ (Feature Extraction) และใช้เทคนิคการคัดเลือก (Selection Search) เพื่อหาพื้นที่ที่มีวัตถุภายในภาพ ซึ่งจะถูกนำมาสร้างเป็นการเสนอพื้นที่ (Region Proposals) (พื้นที่ที่จะถูก

ตรวจสอบว่ามีวัตถุหรือไม่) หลังจากนั้นโครงข่ายประสาทเทียมแบบคอนโวลูชันพื้นที่จะนำพื้นที่ที่ถูกนำเสนอไปทำการทำนายว่าพื้นที่นั้นมีวัตถุปรากฏอยู่หรือไม่ โดยใช้โครงข่ายประสาทเทียมแบบคอนโวลูชันที่ถูกฝึกสอนมาล่วงหน้าด้วยภาพที่ถูกระบุไว้แล้วว่าภาพที่ถูกนำมาฝึกสอนคือภาพอะไรซึ่งเป็น Positive Examples และ Negative Examples เพื่อเรียนรู้วิเคราะห์ภาพว่ามีวัตถุปรากฏอยู่หรือไม่ และแนวโน้มของขนาดและตำแหน่งของวัตถุในพื้นที่ที่กำหนด จากนั้นจะนำพื้นที่ที่ถูกนำเสนอ ที่ถูกตรวจจับวัตถุไปทำการหา Bounding Box Classification ที่ให้ความแม่นยำสูงสุดสำหรับวัตถุนั้น ๆ และใช้ Intersection Over Union : IoU ดังรูปที่ 2.14



รูปที่ 2.14 การทำงานของ Region-based Convolutional Neural Networks

2.4.1 การเสนอพื้นที่ (Region Proposals)

วิธีการเสนอพื้นที่ คือ การหาพื้นที่ที่มีสิ่งที่น่าสนใจภายในภาพ โดยใช้วิธีการต่าง ๆ เช่น Selective Search และ Edge Boxes ซึ่งเป็นเทคนิคในการตรวจหาวัตถุในภาพ โดยที่วิธีการทำงานของวิธีเสนอพื้นที่มีขั้นตอน ดังนี้

1) การสุ่ม (Sampling) โดยใช้วิธีการต่าง ๆ เช่น Selective Search, Edge Boxes เพื่อหาพื้นที่ที่มีสิ่งที่น่าสนใจภายในภาพ

2) คำนวณคุณลักษณะ (Features) ของพื้นที่ที่สุ่มมาโดยใช้โมเดลการเรียนรู้เชิงลึก (Deep Learning) เพื่อนำไปใช้ในการตรวจหาวัตถุ

3) วิเคราะห์คุณลักษณะของพื้นที่ที่สุ่มมาว่ามีลักษณะเป็นวัตถุอย่างไร โดยใช้โมเดลการเรียนรู้เชิงลึก

4) ประมวลผลเพื่อหาว่าพื้นที่นั้นมีวัตถุหรือไม่ โดยใช้โมเดลการเรียนรู้เชิงลึก โดยจะใช้วิธีการตรวจหาวัตถุในภาพ

5) สร้าง Bounding Box หรือสี่เหลี่ยมที่ครอบคลุมพื้นที่ที่มีวัตถุ โดยค่าความแม่นยำของ Bounding Box จะขึ้นอยู่กับค่าความแม่นยำของโมเดลการเรียนรู้เชิงลึก

6) ได้ตำแหน่งสี่เหลี่ยมที่ครอบคลุมพื้นที่ที่มีวัตถุ และจะถูกนำไปใช้ในการตรวจหาวัตถุและทำการจำแนกวัตถุต่อไป

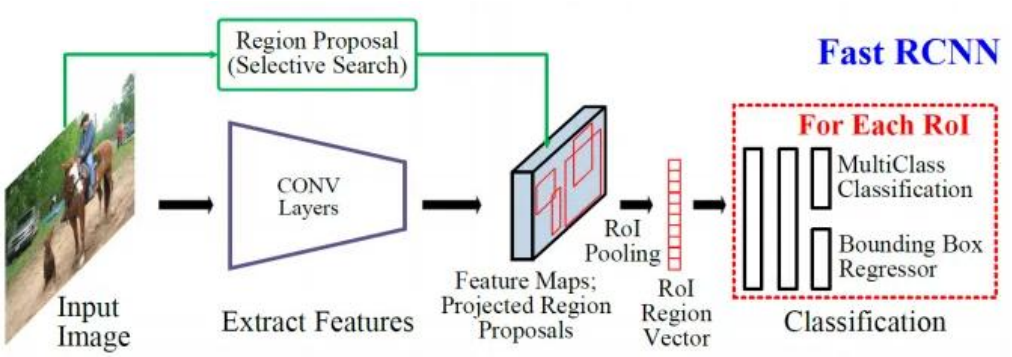
วิธีการเสนอพื้นที่ เป็นเทคนิคที่มีประสิทธิภาพสูงในการหาพื้นที่ที่มีวัตถุอยู่ในภาพและมักถูกนำมาใช้ร่วมกับการตรวจหาวัตถุเพื่อปรับปรุงประสิทธิภาพในการตรวจหาวัตถุในภาพ โดยเฉพาะในงานด้านวิทยาศาสตร์การคอมพิวเตอร์และการมองเห็น (Computer Vision) ซึ่งเป็นที่นิยมใช้ในการตรวจจับวัตถุและวิเคราะห์ภาพในแวดวงต่าง ๆ เช่น การตรวจจับใบหน้า (Face Detection), การตรวจจับสิ่งของ (Object Detection), การติดตามวัตถุ (Object Tracking) และอื่น ๆ อีกมากมาย

ในทางปฏิบัติ การใช้วิธีการเสนอพื้นที่มักถูกนำมาใช้ร่วมกับโมเดลการเรียนรู้เชิงลึก (Deep Learning) เพื่อเพิ่มประสิทธิภาพในการตรวจหาวัตถุในภาพ โดยที่โมเดลการเรียนรู้เชิงลึกจะใช้โครงข่ายประสาทเทียมแบบคอนโวลูชันในการสกัดคุณลักษณะของภาพ และทำงานร่วมกับ Region Proposals เพื่อช่วยในการหาพื้นที่ที่วัตถุอยู่ในภาพ

2.5 Fast R-CNN [6]

Fast R-CNN คือ โครงข่ายประสาทเทียมแบบคอนโวลูชันแบบเสนอพื้นที่ (Region-based Convolutional Neural Networks : R-CNN) ที่มีความรวดเร็วขึ้น เนื่องจากตัวของ R-CNN มีจุดอ่อนหลัก คือการต้องรู้จำพื้นที่ย่อยทุกพื้นที่ในลักษณะเหมือนเริ่มใหม่ทุกครั้ง เช่น ถ้าพื้นที่ถูกเสนอมา 2,000 พื้นที่ ก็ต้องทำการรู้จำภาพที่เป็นอิสระจากกัน 2,000 ครั้ง นั่นก็คือการที่ต้องคำนวณ Convolutional Neural Network (CNN) ใหม่ 2,000 รอบ ส่งผลให้กระบวนการมีความช้า

ในขณะที่ Fast R-CNN ใช้ Convolutional Neural Network (CNN) ในการคำนวณเพียงครั้งเดียว แล้วนำข้อมูลที่ได้จากผลลัพธ์ที่ผ่านการทำ CNN (Feature Map) มาประมวลผลต่อที่ระดับการเสนอพื้นที่ (Region Proposals) ทำให้ลดการทำงานที่ซ้ำซ้อน จึงมีกระบวนการที่เร็วยิ่งขึ้น



รูปที่ 2.15 การทำงานของ Fast R-CNN [6]

จากรูปที่ 2.15 การทำงานของ Fast R-CNN มีดังนี้

1) Extract Features ใช้ Convolutional Neural Network (CNN) เพียงครั้งเดียวเพื่อสร้าง Feature Map จากทั้งภาพแทนที่การประมวลผลภาพแต่ละพื้นที่

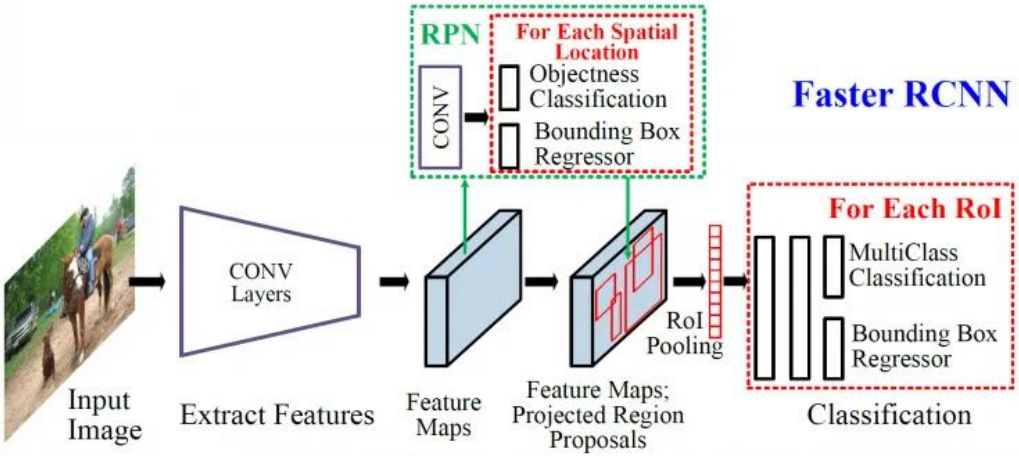
2) Region Proposal ใช้เทคนิคการคัดเลือก (Selective Search) เพื่อสร้างชุดของพื้นที่ที่น่าสนใจ (Region Proposals) โดยแต่ละพื้นที่จะถูกแปลงตำแหน่งภาพ (Map) ไปยัง Feature Map

3) RoI Pooling (Region of Interest Pooling) เป็นกระบวนการเพื่อปรับขนาดของ Feature Map ของแต่ละพื้นที่ให้มีขนาดคงที่ ซึ่งช่วยลดภาระการประมวลผลและทำให้การตรวจจับเร็วขึ้น

4) Classification เป็นการผ่าน Fully Connected Layers เพื่อทำการจำแนกประเภทของวัตถุ (Classification) และทำนายตำแหน่งของ Bounding Box

2.6 Faster R-CNN [6]

Faster R-CNN ถูกพัฒนาเพิ่มเติมจาก Fast R-CNN โดยปรับปรุงขั้น Region Proposal เพื่อให้ทำงานเร็วและมีประสิทธิภาพมากยิ่งขึ้น ด้วยการใช้โครงข่ายเสนอพื้นที่ (Region Proposal Network : RPN) ซึ่งเป็นโครงข่ายที่เรียนรู้เพื่อสร้างพื้นที่ที่น่าสนใจ (Region Proposals) โดยตรง แทนการใช้เทคนิคการคัดเลือก (Selective Search)



รูปที่ 2.16 การทำงานของ Faster R-CNN [6]

2.6.1 Region Proposal Network (RPN) [7]

โครงข่ายเสนอพื้นที่ (Region Proposal Network) เป็นส่วนสำคัญใน Faster R-CNN ที่ทำหน้าที่สร้างพื้นที่ที่น่าสนใจ (Region Proposals) ในภาพที่อาจมีวัตถุอยู่ โดยทำงานร่วมกับโครงข่ายประสาทเทียมแบบคอนโวลูชัน (Convolutional Neural Network : CNN) เพื่อทำให้กระบวนการตรวจจับวัตถุรวดเร็วและแม่นยำขึ้นเมื่อเทียบกับเทคนิคการคัดเลือก (Selective Search) โดยการทำงานของโครงข่ายเสนอพื้นที่ มีดังนี้

1) Input เป็นการที่ RPN รับข้อมูล Feature Map จาก CNN ซึ่งได้จากการประมวลผลภาพต้นฉบับ

2) Sliding Window เป็นการที่ RPN ใช้ Sliding Window ขนาดเล็ก เช่น 3x3 สแกนทั่วทั้ง Feature Map ในแต่ละตำแหน่งของ Sliding Window จะสร้าง Anchor Boxes หลายขนาดและอัตราส่วนต่าง ๆ

3) Anchor Boxes คือ กรอบที่เสนอว่าบริเวณนั้นอาจมีวัตถุ

4) Prediction คือ การทำนาย ใน RPN มีเลเยอร์ที่ทำงาน 2 ส่วน

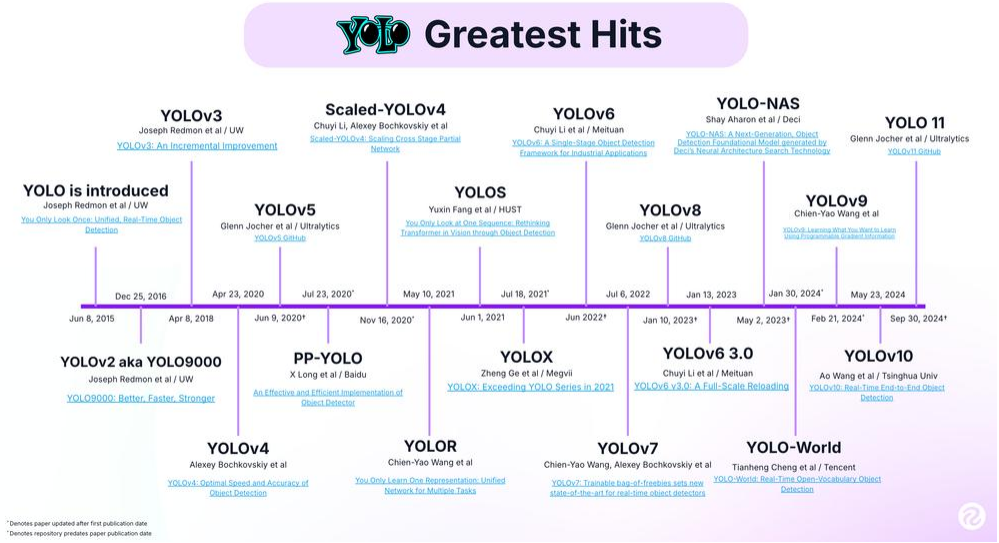
- Objectness Score : ทำนายว่า Anchor Boxes มีวัตถุหรือไม่
- Bounding Box Regression : ปรับตำแหน่งและขนาดของ Anchor Box ให้แม่นยำยิ่งขึ้น

5) Non-Maximum Suppression (NMS) เป็นการกำจัด Anchor Boxes ที่ซ้อนทับกันมากเกินไป และเลือกเฉพาะ Anchor Boxes ที่มีคะแนนสูงสุด

6) Output เป็นการที่ RPN จะส่ง Regression Proposals ที่ดีที่สุด ไปยังขั้นตอน RoI Pooling

2.7 YOLO [8]

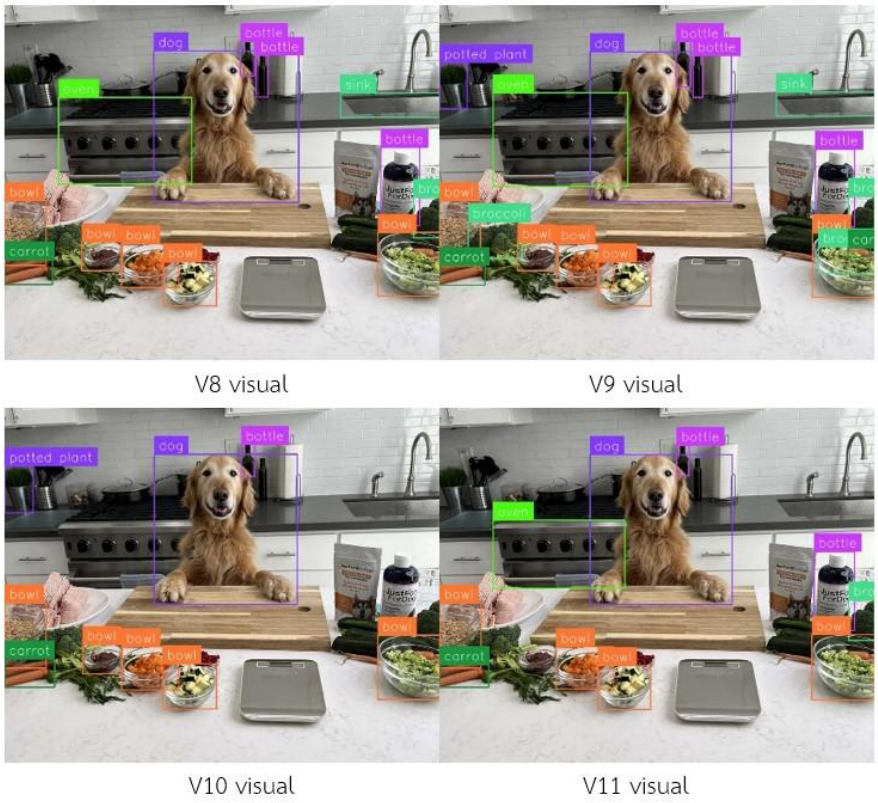
YOLO (You Only Look Once) เป็นกลุ่มโมเดลคอมพิวเตอร์วิชัน (Computer Vision) ที่ได้รับความสนใจเป็นอย่างมาก ถูกพัฒนาขึ้นโดย Joseph Redmon บนเฟรมเวิร์ค (Framework) ที่เขาออกแบบเองชื่อว่า Darknet ซึ่งเป็นเฟรมเวิร์คสำหรับงานวิจัยที่ยืดหยุ่นและพัฒนาด้วยภาษาโปรแกรมระดับต่ำ Darknet ได้รับการพัฒนามาอย่างต่อเนื่องและได้สร้างโมเดลตรวจจับวัตถุแบบเรียลไทม์ที่ดีที่สุดในสาย Computer Vision นั่นก็คือ YOLO



รูปที่ 2.17 ตระกูลโมเดล YOLO ในเวอร์ชันต่าง ๆ [8]

ตระกูลโมเดล YOLO ได้มีการพัฒนามาอย่างต่อเนื่องตั้งแต่เปิดตัวครั้งแรก โดยเฉพาะ YOLOv2 และ YOLOv3 ซึ่งพัฒนาโดย Joseph Redmon ส่วนโมเดล YOLO ที่ออกมาหลังจาก YOLOv3 ได้รับการพัฒนาโดยผู้เขียนคนใหม่ แต่ละเวอร์ชันมีเป้าหมายที่แตกต่างกันตามแนวคิดของผู้พัฒนา โมเดล YOLO ดั้งเดิมถือเป็นเครือข่ายตรวจจับวัตถุ (Object Detection) ตัวแรกที่สามารถรวมปัญหาการระบุบริเวณที่

น่าสนใจ (Bounding Box) และการระบุป้ายกำกับคลาส (Class Labels) เข้าไว้ในเครือข่ายแบบ End-to-End Differentiable ได้สำเร็จ ในขณะที่บางโมเดลตรวจจับวัตถุจะแบ่งการทำงานออกเป็นสองส่วน ได้แก่ การระบุบริเวณที่น่าสนใจ (Bounding Box) ซึ่งเป็นที่ตั้งของวัตถุ และการจัดประเภท (Classify) บริเวณที่ระบุไว้นั้น วิธีนี้เรียกว่า Two-Stage Detector โดยโมเดลยอดนิยม เช่น Faster R-CNN ใช้วิธีการนี้ในการตรวจจับวัตถุ



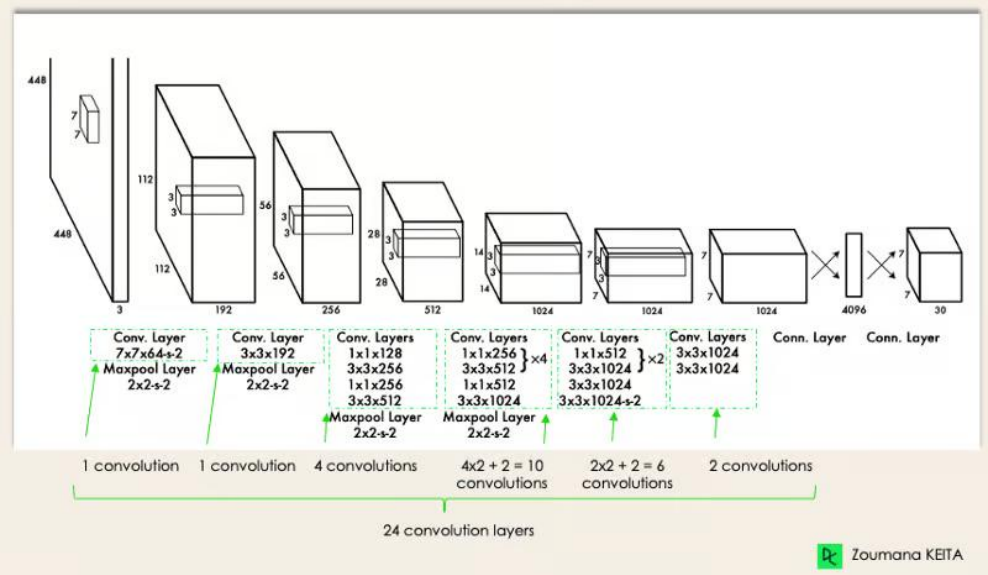
รูปที่ 2.18 การเปรียบเทียบการตรวจจับของโมเดล YOLOv8 – YOLOv11 [8]

2.7.1 สถาปัตยกรรม YOLO [8] [9]

YOLO เป็นตัวตรวจจับแบบขั้นตอนเดียว (single-stage detector) ซึ่งจัดการทั้งการระบุวัตถุ (Object Detection) และการจัดประเภท (Classification) ในครั้งเดียว และมีประสิทธิภาพในด้านความเร็วและความแม่นยำ เนื่องจากมีขนาดเล็กมาก ซึ่งช่วยให้ฝึกได้เร็วขึ้นและนำไปใช้งานได้ง่ายขึ้น โดยเฉพาะอย่างยิ่งบนอุปกรณ์ที่มีทรัพยากรจำกัด

อัลกอริทึม YOLO ใช้สำหรับการตรวจจับวัตถุแบบเรียลไทม์ ก่อนหน้าที่จะมี YOLO นั้น ตัวโมเดล R-CNN เป็นหนึ่งในวิธีการที่ใช้กันทั่วไปในการตรวจจับวัตถุ แต่มีความล่าช้าและไม่เหมาะสำหรับการใช้งานแบบเรียลไทม์ โมเดล YOLO ให้ความเร็วที่จำเป็นสำหรับกรณีการใช้งานที่ต้องการการวิเคราะห์ที่รวดเร็ว เช่น การตรวจจับรถยนต์ การระบุสัตว์ และการตรวจสอบการละเมิดความปลอดภัย

สถาปัตยกรรมของ YOLO มีความคล้ายคลึงกับ GoogleNet โดยประกอบไปด้วยเลเยอร์คอนโวลูชัน 24 ชั้น (Convolution Layer) เลเยอร์ Max Pooling 4 ชั้น (Max Pooling Layer) และเลเยอร์ Fully Connected 2 ชั้น (Fully Connected Layer)



รูปที่ 2.17 สถาปัตยกรรมของ YOLO [9]

การทำงานของสถาปัตยกรรม YOLO มี 4 ขั้นตอน ดังนี้

1) การปรับขนาดภาพ ภาพอินพุต (Input Image) จะถูกปรับขนาดเป็น 448x448 ก่อนที่จะผ่านเข้าสู่เครือข่ายคอนโวลูชัน

2) การคอนโวลูชัน เริ่มด้วยการใช้คอนโวลูชันขนาด 1x1 เพื่อลดจำนวนช่องสัญญาณ (Channels) ตามด้วยคอนโวลูชันขนาด 3x3 เพื่อสร้างเอาต์พุต (Output) ที่เป็นรูปทรงสี่เหลี่ยม

3) ฟังก์ชันการเปิดใช้งาน ที่ใช้คือ ReLU ยกเว้นสำหรับเลเยอร์สุดท้ายที่ใช้ฟังก์ชันเปิดใช้งานแบบเชิงเส้น (Linear Activation Function)

4) การใช้เทคนิคเพิ่มเติม มีการใช้เทคนิคต่าง ๆ เช่น Batch Normalization และ Dropout เพื่อทำให้โมเดลไม่ Overfitting

2.7.2 YOLOv11 [10]

YOLOv11 เป็นเวอร์ชันล่าสุดในตระกูล YOLO ที่นำเสนอการพัฒนาอย่างมากในด้านความเร็ว ความแม่นยำ และการดึงคุณลักษณะ (Feature Extraction) โครงสร้างของ YOLOv11 เน้นส่วนประกอบหลักของโมเดล โดยทั่วไปแล้วประกอบด้วย 3 ส่วนหลัก คือ

1) Backbone

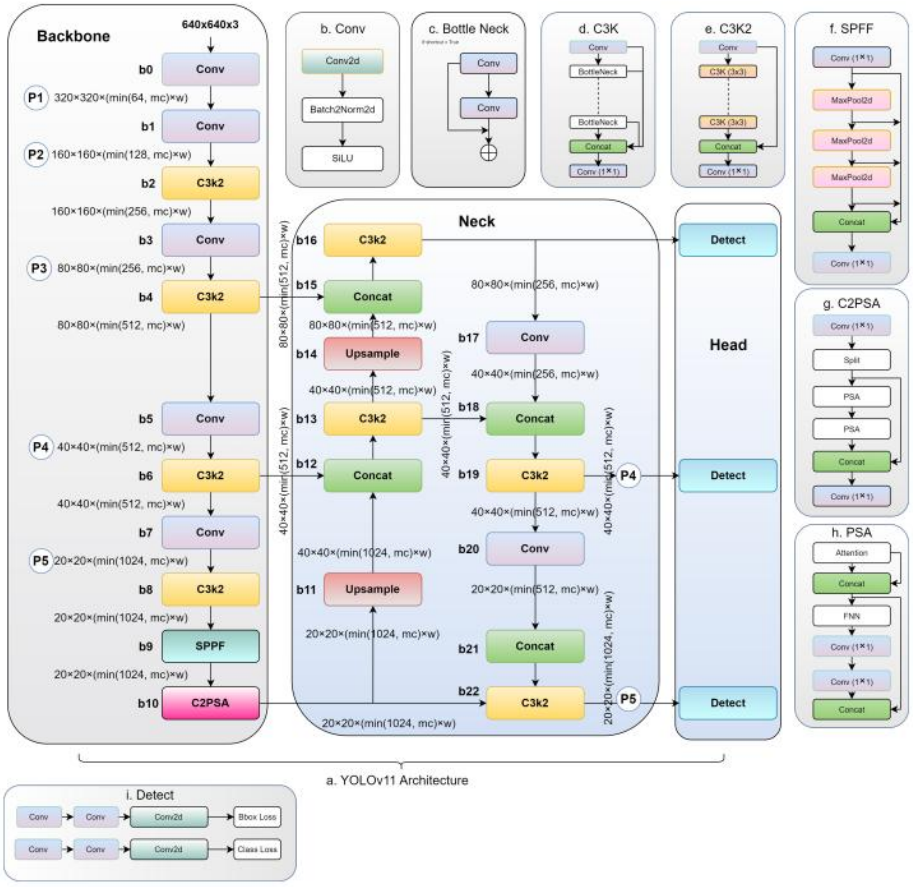
เป็นส่วนที่รับผิดชอบในการดึงคุณลักษณะสำคัญจากภาพอินพุต (Input Image) ในหลายระดับ (scales) ส่วนนี้ประกอบด้วยหลาย ๆ Conv block โดยแต่ละบล็อกประกอบด้วยสามส่วนย่อย คือ Conv2D, BatchNorm2D และฟังก์ชันการเปิดใช้งาน SiLU รวมไปถึง C3K2 Blocks ช่วยในการทำงานที่มีประสิทธิภาพมากขึ้นในการประมวลผล Cross-Stage Partial (CSP) และในสองบล็อคสุดท้ายของ Backbone คือ Spatial Pyramid Pooling Fast (SPPF) และ Cross-Stage Partial with Spatial Attention (C2PSA) โดยบล็อก SPPF ใช้หลายชั้น Max Pooling เพื่อดึงคุณลักษณะหลายระดับจากภาพอินพุตอย่างมีประสิทธิภาพ ส่วน C2PSA block จะนำกลไกการให้ความสนใจ (Attention Mechanism) มาใช้เพื่อเสริมความแม่นยำของโมเดล

2) Neck

บทบาทหลักของ Neck คือ การรวมคุณลักษณะจากหลายระดับ (Scales) และส่งผ่านไปยัง Head Blocks ประกอบไปด้วยหลาย Conv layer, C3K2 blocks, การเชื่อมต่อ (Concat Operations) และบล็อก Upsample พร้อมกับข้อได้เปรียบของกลไก C2PSA

3) Head

เป็นโมดูลสำคัญที่รับผิดชอบในการสร้างการทำนาย เป็นการกำหนดคลาสของวัตถุ คำนวณคะแนน Objectness และทำนายกรอบขอบเขต (Bounding Boxes) ของวัตถุที่ตรวจพบได้อย่างแม่นยำ



รูปที่ 2.19 สถาปัตยกรรมของ YOLOv11 [10]

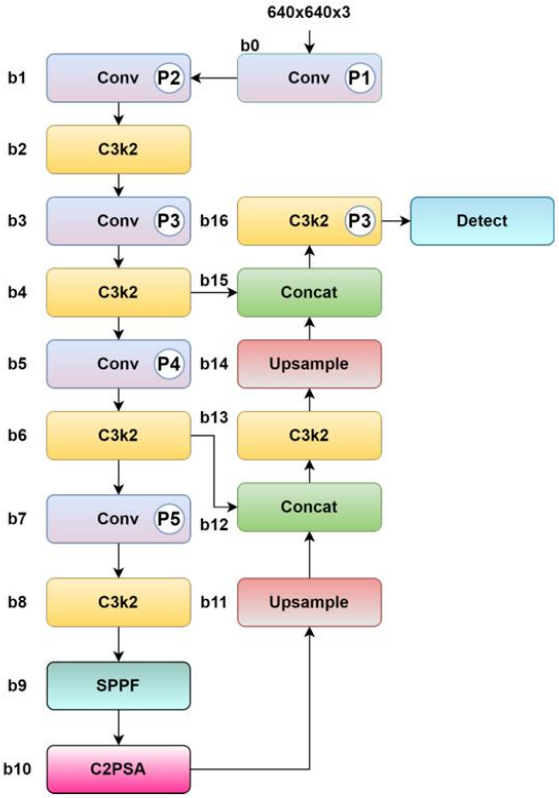
Backbone ของสถาปัตยกรรม YOLOv11 ทำการลดขนาดของภาพอินพุตหลายรอบจนกลายเป็นหลายระดับ เช่น 2x, 4x, 8x, 16x, และ 32x ซึ่งกระบวนการนี้จะสร้างชุดคุณลักษณะ 5 ชุด (320x320, 160x160, 80x80, 40x40 และ 20x20) ชุดคุณลักษณะเหล่านี้ที่เรียกว่า (P1, P2, P3, P4, P5) ตามที่แสดงในรูปที่ 2.18 จะถูกรวมกับส่วนประกอบอื่น ๆ ของโมเดล เช่น SPPF และ C2PSA แล้วส่งผ่านไปยัง Head Blocks ชุดคุณลักษณะที่มีขนาดใหญ่จะรับผิดชอบในการตรวจจับวัตถุขนาดใหญ่ ในขณะที่ชุดคุณลักษณะที่มีขนาดกลาง เช่น 40x40 จะใช้ในการตรวจจับวัตถุขนาดกลาง และชุดคุณลักษณะที่มีขนาดเล็ก เช่น 20x20 จะมุ่งเน้นไปที่การตรวจจับวัตถุขนาดเล็ก

โมเดล YOLOv11 จะมี Head ที่ประกอบไปด้วยสามบล็อกการตรวจจับ ซึ่งแต่ละบล็อกจะรับผิดชอบในการตรวจจับวัตถุในขนาดที่แตกต่างกัน เช่น วัตถุขนาดเล็กมักจะมีขนาดน้อยกว่า 32^2 พิกเซล

วัตถุขนาดกลางมีขนาดมากกว่า 32^2 พิกเซล แต่ต่ำกว่า 96^2 พิกเซล และวัตถุขนาดใหญ่มีขนาดมากกว่า 96^2 พิกเซล และในบางกรณีแอปพลิเคชันการตรวจจับวัตถุอาจถูกออกแบบมาเพื่อมุ่งเน้นไปที่ขนาดวัตถุเฉพาะ ตัวอย่างเช่น แอปพลิเคชันทางอากาศมักจะเกี่ยวข้องกับการตรวจจับวัตถุขนาดเล็กในภาพ เพื่อเพิ่มประสิทธิภาพในการใช้ทรัพยากร แทนที่จะใช้สถาปัตยกรรม YOLOv11 มาตรฐาน โมเดล YOLOv11 จึงได้พัฒนา 6 เวอร์ชันที่ปรับแต่งและออกแบบมาเพื่อตรวจจับขนาดของวัตถุที่เฉพาะเจาะจง ดังนี้

1) YOLOv11-small

เวอร์ชันที่ปรับแต่งแรก คือเวอร์ชันขนาดเล็ก ซึ่งถูกออกแบบมาเพื่อตรวจจับวัตถุที่มีขนาดพื้นที่น้อยกว่าหรือเท่ากับ 32^2 พิกเซล เพื่อปรับแต่ง YOLOv11

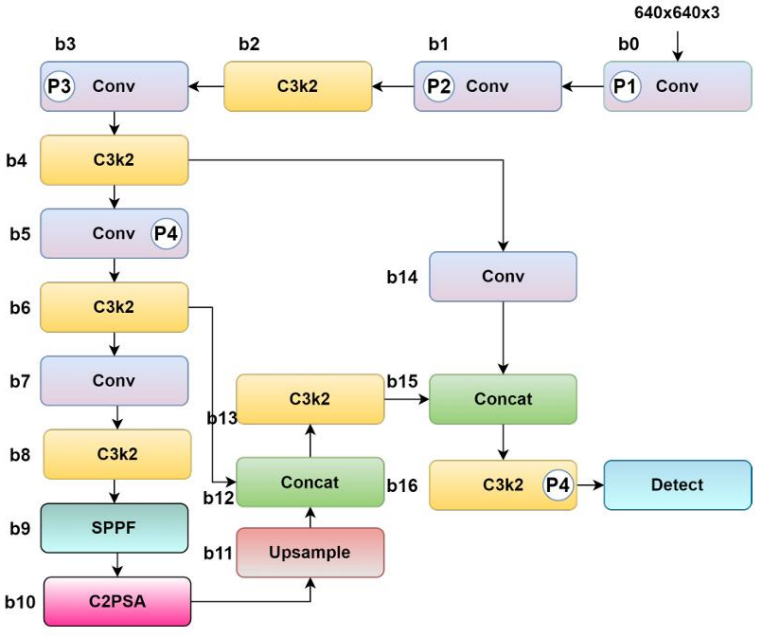


รูปที่ 2.20 สถาปัตยกรรมของ YOLOv11-small [10]

จากรูปที่ 2.20 การทำงานของ YOLOv11-small เริ่มตั้งแต่ทำการติดป้ายแต่ละบล็อกในสถาปัตยกรรมดั้งเดิม โดยเริ่มต้นด้วย "b" ตั้งแต่ b0 ถึง b22 เพื่อความสะดวก ตามที่กล่าวถึงในส่วนก่อนหน้า บล็อกการตรวจจับตัวแรกถูกใช้สำหรับตรวจจับขนาดของวัตถุขนาดเล็ก สำหรับสิ่งนี้จึงลบบล็อกการตรวจจับที่สองและสามออก เป็นการได้ลบบล็อกที่ให้คุณลักษณะสำหรับการตรวจจับขนาดที่ใหญ่ขึ้นออกไปด้วย ผลลัพธ์คือบล็อกตั้งแต่ b17 ถึง b22 ซึ่งเกี่ยวข้องกับวัตถุขนาดกลางและใหญ่ถูกลบออก ทำให้ตรวจจับได้เฉพาะวัตถุขนาดเล็กเท่านั้น

2) YOLOv11-medium

เวอร์ชันที่ปรับแต่งที่สอง ถูกออกแบบมาเพื่อตรวจจับวัตถุขนาดกลางโดยเฉพาะ ซึ่งนิยามว่ามีขนาดใหญ่กว่า 32^2 พิกเซล แต่เล็กกว่า 96^2 พิกเซล ในสถาปัตยกรรมของ YOLOv11-medium ได้ทำการลบบล็อกทั้งหมดที่เกี่ยวข้องกับการตรวจจับวัตถุขนาดเล็กและขนาดใหญ่ จากในสถาปัตยกรรม YOLOv11-medium โดยลบบล็อกที่รับผิดชอบการประมวลผลคุณลักษณะที่เกี่ยวข้องกับการตรวจจับวัตถุขนาดเล็กและขนาดใหญ่โดยเฉพาะ ทำให้ตรวจจับได้เฉพาะวัตถุขนาดกลางเท่านั้น

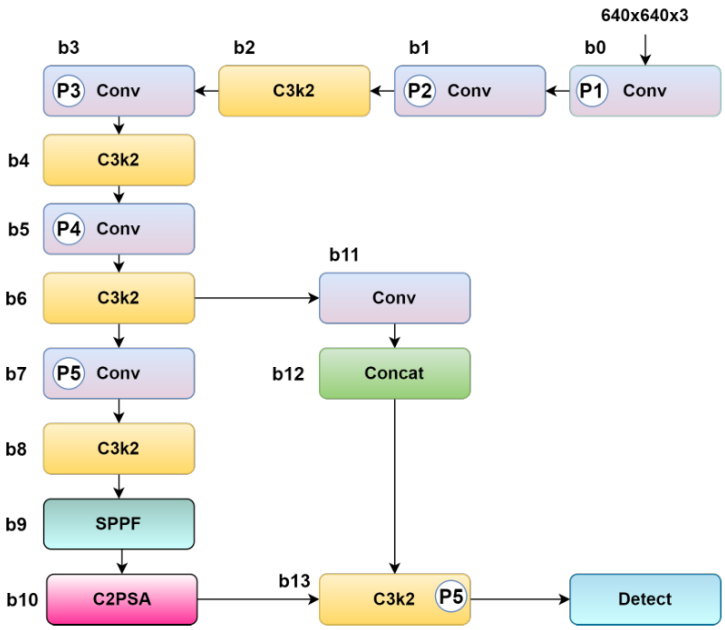


รูปที่ 2.21 สถาปัตยกรรมของ YOLOv11-medium [10]

จากรูปที่ 2.21 สำหรับวัตถุขนาดเล็ก บล็อก b14, b15, และ b16 ถูกลบออก เนื่องจากบล็อกเหล่านี้ให้ข้อมูลกับหัวตรวจจับสำหรับวัตถุขนาดเล็ก ในทำนองเดียวกัน บล็อก b20, b21, และ b22 ถูกลบออก เนื่องจากบล็อกเหล่านี้ให้ข้อมูลกับหัวการตรวจจับสำหรับวัตถุขนาดใหญ่ หลังจากลบบล็อกเหล่านี้ออกแล้ว ได้เปลี่ยนชื่อบล็อก YOLOv11 เดิมที่เกี่ยวข้องกับวัตถุขนาดกลางที่ก่อนหน้านี้ คือ b17, b18, และ b19 เป็น b14, b15, และ b16 ตามลำดับ

3) YOLOv11-large

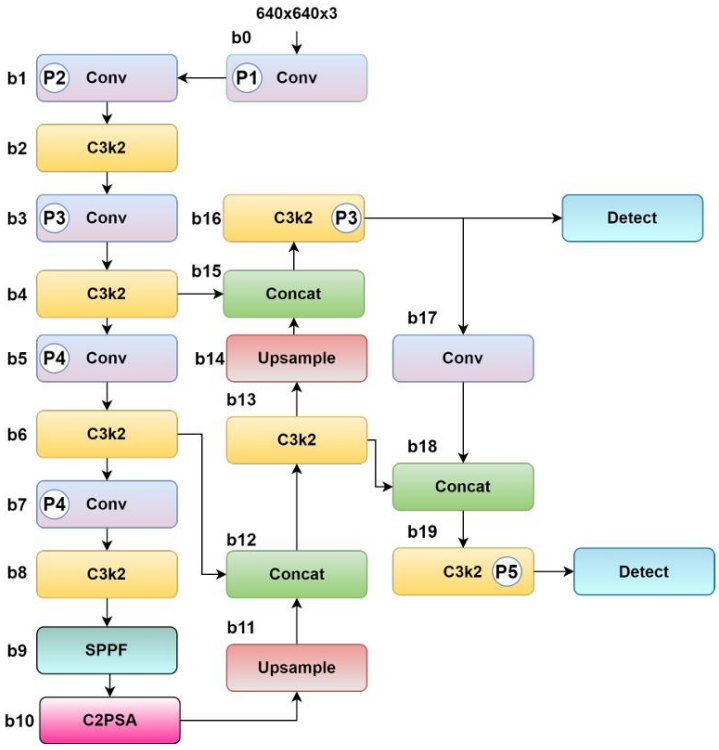
เวอร์ชันที่สามที่ปรับปรุงและได้รับการออกแบบมาเพื่อมุ่งเน้นไปที่การตรวจจับวัตถุขนาดใหญ่ โดยวัตถุที่มีพื้นที่ขนาดมากกว่า 96^2 พิกเซล ในการสร้างโมเดล YOLOv11-large ได้มีการปรับแต่งสถาปัตยกรรมดั้งเดิมโดยการลบส่วนประกอบที่ไม่เกี่ยวข้องกับการตรวจจับวัตถุขนาดใหญ่ออก และทำการเชื่อมต่อบล็อกที่ไม่ได้เชื่อมต่อก่อนหน้านี้อีกครั้ง โดยเฉพาะบล็อก b11 ถึง b19 ได้ถูกลบออก เนื่องจากเกี่ยวข้องกับการให้ฟีเจอร์สำหรับการตรวจจับวัตถุขนาดเล็กและขนาดกลาง เพื่อให้เครือข่ายยังคงต่อเนื่อง บล็อก b19 ได้รับการเชื่อมต่อเพื่อรับฟีเจอร์จากบล็อก b6 เนื่องจากทั้งสองใช้แผนที่ฟีเจอร์เดียวกันเป็นอินพุต นอกจากนี้ บล็อก b19, b21 และ b22 ได้เปลี่ยนชื่อเป็น b11, b12 และ b13 ตามลำดับ ดังรูปที่ 2.22



รูปที่ 2.22 สถาปัตยกรรมของ YOLOv11-large [10]

4) YOLOv11-sm

เวอร์ชันที่ถูกปรับปรุงลำดับที่สี่ ได้รับการออกแบบมาเพื่อเพิ่มความยืดหยุ่นในการตรวจจับวัตถุขนาดเล็กและขนาดกลาง โดยวัตถุที่ตรวจจับจะมีพื้นที่ขนาดน้อยกว่า 96^2 พิกเซล สำหรับการปรับปรุงในเวอร์ชันนี้ ส่วนที่เกี่ยวข้องกับการตรวจจับวัตถุขนาดใหญ่ถูกตัดออกไป ขณะที่บล็อกสำหรับการตรวจจับวัตถุขนาดเล็กและขนาดกลางยังคงอยู่ การปรับปรุงนี้เกี่ยวข้องกับการลบส่วนหัวการตรวจจับที่สามและบล็อกทั้งหมดที่ป้อนข้อมูลเข้าไปในส่วนนี้ บล็อก b20, b21 และ b22 ถูกลบออก ในขณะที่บล็อกตั้งแต่ b0 ถึง b19 ยังคงไม่เปลี่ยนแปลง ดังรูปที่ 2.23

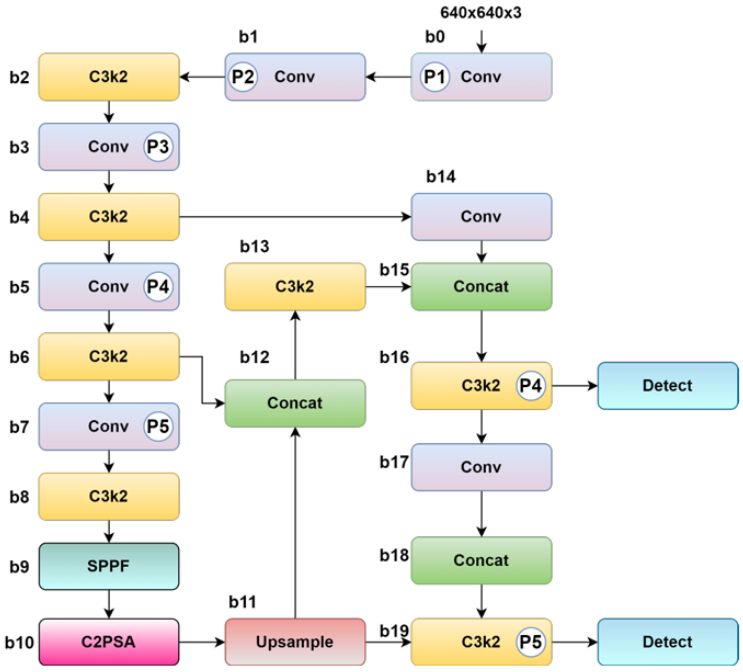


รูปที่ 2.23 สถาปัตยกรรมของ YOLOv11-sm [10]

5) YOLOv11-ml

เป็นเวอร์ชันที่ปรับปรุงลำดับที่ห้า ได้รับการออกแบบมาเพื่อมุ่งเน้นการตรวจจับวัตถุขนาดกลางและขนาดใหญ่ โดยวัตถุที่มีขนาดใหญ่กว่า 32^2 พิกเซล จะสามารถประมวลผลได้อย่างมีประสิทธิภาพในโมเดลนี้ การปรับปรุง YOLOv11-ml เกิดจากการปรับแต่งส่วนที่เกี่ยวข้องกับการตรวจจับ

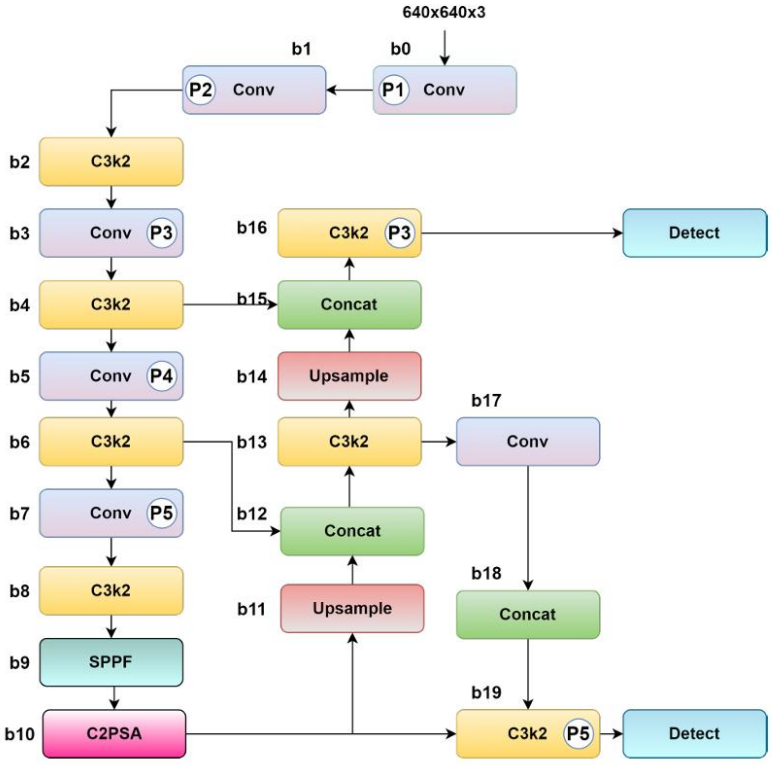
วัตถุขนาดเล็กโดยเฉพาะ บล็อก b14, b15 และ b16 ถูกตัดออกไป หลังจากนั้นบล็อกที่เหลืออยู่ b17 ถึง b22 ได้รับการเปลี่ยนหมายเลขใหม่เป็น b14 ถึง b19 สุดท้ายบล็อกหมายเลข b4 ถูกเชื่อมโยงกับบล็อก b14 ดังรูปที่ 2.24



รูปที่ 2.24 สถาปัตยกรรมของ YOLOv11-ml [10]

6) YOLOv11-sl

เป็นเวอร์ชันสุดท้าย ได้รับการออกแบบมาเพื่อมุ่งเน้นการตรวจจับวัตถุขนาดเล็กและขนาดใหญ่ โดยวัตถุต้องมีพื้นที่ ≤ 32^2 หรือ > 96^2 พิกเซล สำหรับการพัฒนาโมเดลนี้จาก YOLOv11 ส่วนที่เกี่ยวข้องกับการตรวจจับวัตถุขนาดกลางถูกตัดออกไป โดยเฉพาะบล็อก b17, b18 และ b19 ถูกลบออก และบล็อก b20 ถูกเชื่อมโยงใหม่เพื่อแทนที่บล็อก b17 หลังจากนั้นหมายเลขของบล็อกถูกปรับเปลี่ยนใหม่ โดยบล็อก b0 ถึง b16 ยังคงเดิม ขณะที่บล็อก b20, b21 และ b22 ได้รับหมายเลขใหม่เป็น b17, b18 และ b19 ตามลำดับ ดังรูปที่ 2.25



รูปที่ 2.25 สถาปัตยกรรมของ YOLOv11-sl [10]

ตารางที่ 2.1 ขนาดของ YOLOv11 ในเวอร์ชันต่าง ๆ

Model Name	Object Size Range
YOLOv11-small	area ≤ 32^2
YOLOv11-medium	32^2 < area ≤ 96^2
YOLOv11-large	area > 96^2
YOLOv11-sm	area ≤ 96^2
YOLOv11-ml	32^2 < area
YOLOv11-sl	area ≤ 32^2 or area > 96^2

2.8 Grounding DINO [11]

Grounding DINO คือ โมเดลตรวจจับวัตถุแบบเปิดเซ็ต โดยการผสานโมเดลตรวจจับแบบ Transformer ที่ชื่อ DINO เข้ากับการฝึกฝนโมเดลแบบ Grounded ซึ่งสามารถตรวจจับวัตถุใด ๆ ก็ได้โดยใช้ข้อมูลนำเข้าจากมนุษย์ เช่น ชื่อหมวดหมู่หรือคำอธิบายอ้างอิง วิธีแก้ปัญหาหลักของการตรวจจับวัตถุแบบเปิดเซ็ต คือการนำภาษาเข้ามาใช้ร่วมกับโมเดลตรวจจับแบบปิดเซ็ต เพื่อให้เกิดการสรุปแนวคิดแบบเปิดเซ็ตได้ ในการผสานรวมข้อมูลจากภาษาและภาพเข้าด้วยกันอย่างมีประสิทธิภาพ ได้มีการแบ่งเครื่องตรวจจับแบบปิดเซ็ตออกเป็น 3 เฟสในเชิงแนวคิด และเสนอวิธีการผสานรวมที่มีประสิทธิภาพ ซึ่งประกอบด้วย ตัวเพิ่มประสิทธิภาพฟีเจอร์ (Feature Enhancer) การเลือกคิวรีโดยใช้ภาษานำทาง (Language-Guided Query Selection) และตัวถอดรหัสข้ามโมดาลิตี (Cross-Modality Decoder) สำหรับการรวมข้อมูลขนาดใหญ่ รวมถึงข้อมูลการตรวจจับวัตถุ ข้อมูลการจับคู่ ข้อมูลคำบรรยาย และการประเมินผลโมเดลในทั้งสองตัวชี้วัด ได้แก่ การตรวจจับวัตถุแบบเปิดเซ็ต และการตรวจจับวัตถุจากการอ้างอิง (Referring Object Detection Benchmarks) โดย Grounding DINO สามารถทำผลงานได้ดีในทั้งสามการตั้งค่า รวมถึงตัวชี้วัดบน COCO, LVIS, ODinW และ RefCOCO ซึ่ง Grounding DINO ได้คะแนน 52.5 AP บนตัวชี้วัด COCO Zero-Shot นอกจากนี้ยังทำลายสถิติใหม่บนตัวชี้วัด ODinW Zero-Shot ด้วยคะแนนเฉลี่ย 26.1 AP

การตรวจจับวัตถุแบบเปิดเซ็ต (Open-Set) ถูกฝึกด้วย Annotation ของ Bounding Box ที่มีอยู่และมุ่งเน้นไปที่การตรวจจับคลาสที่ไม่จำกัดด้วยการใช้ความสามารถของภาษาทั่วไป การตรวจจับแบบเปิดเซ็ตมี ดังนี้

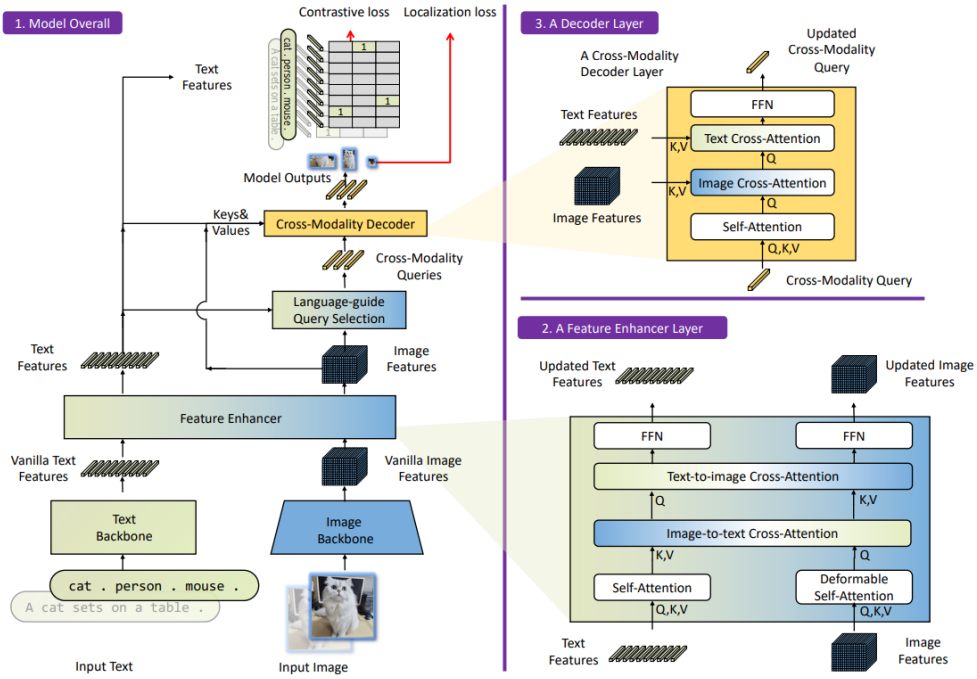
1) โมเดลตรวจจับวัตถุที่รองรับคำศัพท์แบบเปิด (Open-Vocabulary : OV-DETR) ใช้การฝังภาพและข้อความ (Images and Text Embedding) ที่เข้ารหัสโดยโมเดลฝึกภาษาและภาพด้วยการเรียนรู้เชิงเปรียบเทียบ (Contrastive Language Pre-Training : CLIP) เป็น Query เพื่อถอดรหัสกล่องที่ระบุประเภทภายในกรอบ

2) โมเดลตรวจจับวัตถุด้วยภาพและภาษา (DETR Vision and Language knowledge Distillation : DETR ViLD) ใช้วิธีถ่ายทอดความรู้จากโมเดล CLIP ไปยังตัวตรวจจับแบบ R-CNN เพื่อให้ฝัง (Embedding) ของพื้นที่ที่เรียนรู้มีความหมายทางภาษา

3) โมเดลเชื่อมโยงข้อความกับภาพ (Grounded Language-Image Pre-training GLIP) กำหนดปัญหาการตรวจจับวัตถุให้เป็นปัญหาการระบุตำแหน่ง (Grounding) และใช้ข้อมูลการระบุตำแหน่งเพิ่มเติมเพื่อช่วยให้สามารถเรียนรู้ความหมายที่สอดคล้องกันทั้งในระดับวลีและระดับพื้นที่

ผลลัพธ์แสดงให้เห็นว่าวิธีนี้สามารถให้ประสิทธิภาพที่ดียิ่งขึ้น แม้แต่ในชุดข้อมูลที่มีการกำกับดูแลเต็มรูปแบบ

4) โมเดลตรวจจับวัตถุแบบเปิดโดยใช้แนวคิดจากพจนานุกรม (Dictionary-Enriched Visual-Concept Paralleled Pre-training for Open-world Detection : DetCLIP) ใช้ชุดข้อมูลการอธิบายภาพขนาดใหญ่และใช้ Pseudo Labels ที่สร้างขึ้นเพื่อขยายฐานความรู้ ซึ่งช่วยเพิ่มความสามารถในการสรุปผลได้อย่างมีประสิทธิภาพ



รูปที่ 2.26 โครงสร้างโดยรวมของ Grounding DINO, Feature Enhancer Layer และ Decoder Layer ใน Block 1, Block 2 และ Block 3 ตามลำดับ [11]

Grounding DINO ทำหน้าที่จับคู่กล่องวัตถุและคำนามหลายคู่จากคู่ข้อมูลรูปภาพและข้อความ (Images and Text) ที่กำหนด ตัวอย่างตามที่แสดงในรูปที่ 2.26 โมเดลสามารถระบุตำแหน่งของแมวและโต๊ะจากภาพ และจับคู่กับคำว่า “cat” และ “Table” จากข้อความอินพุต ทั้งงานตรวจจับวัตถุและงานระบุตำแหน่ง (REC) สามารถทำงานร่วมกันได้ โดยอ้างอิงจาก CLIP นำชื่อหมวดหมู่ทั้งหมดมารวมกันเป็น

ข้อความอินพุตสำหรับงานตรวจจับวัตถุ ส่วนงาน REC จะต้องใช้กล่องระบุตำแหน่ง (Bounding Box) สำหรับแต่ละข้อความอินพุต และจะเลือกวัตถุที่มีคะแนนสูงสุดเป็นผลลัพธ์

Grounding DINO ใช้โครงสร้างแบบ Dual-Encoder-Single-Decoder โดยมีส่วนประกอบหลักดังนี้

1) Image Backbone สำหรับดึงคุณลักษณะของภาพ
2) Text Backbone สำหรับดึงคุณลักษณะของข้อความ
3) Feature Enhancer สำหรับรวมข้อมูลจากภาพและข้อความเข้าด้วยกัน
4) Language-Guided Query Selection Module สำหรับเลือกข้อมูลที่เกี่ยวข้องกับข้อความ
5) Cross-Modality Decoder สำหรับปรับปรุงตำแหน่งของกล่องวัตถุ

2.8.1 Feature Extraction and Enhancer

เมื่อได้รับคู่ข้อมูล (Images and Text) จะทำการดึงคุณลักษณะของภาพหลายระดับโดยใช้ Images Backbone เช่น Swin Transformer และดึงคุณลักษณะของข้อความโดยใช้ Text Backbone เช่น BERT โดยอ้างอิงจากตัวตรวจจับแบบ DETR คุณลักษณะหลายระดับจะถูกดึงมาจากผลลัพธ์ของบล็อกต่าง ๆ หลังจากดึงคุณลักษณะพื้นฐานของภาพและข้อความแล้ว จะนำข้อมูลเหล่านี้เข้าสู่ Feature Enhancer เพื่อรวมคุณลักษณะข้ามโมดอล

Feature Enhancer ประกอบด้วยหลายเลเยอร์ ดังตัวอย่างของ Feature Enhancer ในรูปที่ 2.26 Block 2 และได้มีการนำ Deformable Self-Attention มาใช้เพื่อเสริมสร้างคุณลักษณะของภาพ และใช้ Vanilla Self-Attention สำหรับการเสริมสร้างคุณลักษณะของข้อความ โดยได้รับแรงบันดาลใจจาก GLIP ในการเพิ่มโมดูล Image-to-Text Cross-Attention และ Text-to-Image Cross-Attention เพื่อผสานคุณลักษณะของภาพและข้อความเข้าด้วยกัน โมดูลเหล่านี้ช่วยให้คุณลักษณะของภาพและข้อความสามารถจับคู่และสอดคล้องกันได้อย่างมีประสิทธิภาพ

2.8.2 Language-Guided Query Selection

Grounding DINO มีเป้าหมายเพื่อตรวจจับวัตถุจากภาพที่กำหนดโดยข้อความอินพุต เพื่อใช้ข้อความอินพุตนำทางการตรวจจับวัตถุได้อย่างมีประสิทธิภาพ ได้มีการออกแบบ Language-Guided Query Selection Module เพื่อเลือกคุณลักษณะที่เกี่ยวข้องกับข้อความอินพุตมากขึ้น มาใช้เป็น Query ของ Decoder กำหนดให้คุณลักษณะของภาพเป็น $X_I \in R^{NI \times d}$ และคุณลักษณะของข้อความเป็น $X_T \in R^{NT \times d}$ โดยที่ N_I จำนวน Token ของภาพ, N_T แทนจำนวนโทเคนของข้อความ และ d คือมิติของคุณลักษณะ ในการทดลอง ได้กำหนด d = 256 โดยทั่วไป ค่า N_I ในโมเดลของเรามักจะมากกว่า 10,000

ขณะที่ N_T ต่ำกว่า 256 เป้าหมายของเราคือการดึง Nq queries จากคุณลักษณะของภาพของ encoder มาใช้เป็นอินพุตของ decoder ตามแนวทางของ DINO เรากำหนดให้ N_q = 900 ดัชนี query อันดับสูงสุด N_q ของคุณลักษณะของภาพ ซึ่งแทนด้วย I_{Nq} ถูกเลือกโดยใช้สมการ

$$\mathrm{I}_{N_q} = \mathrm{Top}_{N_q}\left(\mathrm{Max}^{(-1)}(\mathrm{X}_I\mathrm{X}_T^{\mathrm{T}})\right) \tag{2.8}$$

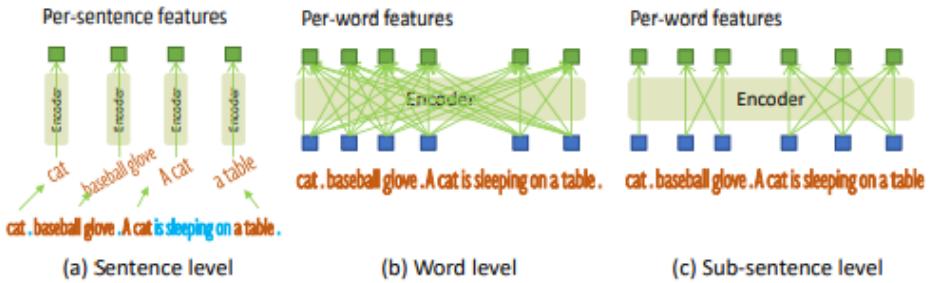
ในสมการนี้ Top_{Nq} แทนการเลือกดัชนีสูงสุด N_q ฟังก์ชัน $Max^{(-1)}$ ทำการคำนวณหาค่าสูงสุดในมิติ -1 และเครื่องหมาย ^{T} แทนการ ทรานสโพสของเมทริกซ์ ขั้นตอนการเลือก Query ใน Algorithm 1 ในรูปแบบ PyTorch Language-Guided Query Selection Module จะส่งออก N_q Indices สามารถดึงคุณลักษณะตามดัชนีที่เลือกมาเพื่อเริ่มต้นค่า Query ตามแนวทางของ DINO ใช้ Mixed Query Selection เพื่อเริ่มต้นค่า Decoder Queries แต่ละ Decoder Query ประกอบด้วยสองส่วน Content Part และ Positional Part กำหนด Positional Part เป็น Dynamic Anchor Boxes ซึ่งถูกกำหนดค่าเริ่มต้นจากผลลัพธ์ของ Encoder ส่วน Content Queries จะถูกตั้งให้สามารถเรียนรู้ได้ระหว่างการฝึกโมเดล

2.8.3 Cross-Modality Decoder

การพัฒนา Cross-Modality Decoder เพื่อรวมคุณลักษณะของภาพและข้อความเข้าด้วยกัน ตามที่แสดงในรูปที่ 2.25 Block 3 โดยแต่ละ Cross-Modality Query จะถูกป้อนเข้าสู่เลเยอร์ Self-Attention กับเลเยอร์ Image Cross-Attention เพื่อรวมคุณลักษณะของภาพ เลเยอร์ Text Cross-Attention เพื่อรวมคุณลักษณะของข้อความ และ FFN Layer ในแต่ละเลเยอร์ของ Cross-Modality Decoder แต่ละเลเยอร์ของ Decoder จะมี Text Cross-Attention Layer เพิ่มขึ้นเมื่อเทียบกับเลเยอร์ของ DINO Decoder เนื่องจากเราต้องการนำข้อมูลข้อความเข้าสู่ Query เพื่อการจัดตำแหน่งรูปแบบของข้อมูลที่ดียิ่งขึ้น

2.8.4 Sub-Sentence Level Text Feature

การสำรวจประเภทของข้อความ 2 ประเภท ที่ใช้ในงาน ซึ่งเราเรียกว่า Sentence Level Representation และ Word Level Representation ตามที่แสดงในรูปที่ 2.27



รูปที่ 2.27 การเปรียบเทียบการแสดงผลข้อความ [11]

1) Sentence Level Representation คือ การแปลงทั้งประโยคให้เป็นคุณลักษณะเดียว ซึ่งถ้าประโยคในข้อมูลมีหลายวลี ก็จะดึงแค่คำที่เป็นวลีออกมาและทิ้งคำอื่น ๆ ไป วิธีนี้จะช่วยให้ไม่มีความสัมพันธ์ระหว่างคำในประโยค แต่ก็ทำให้สูญเสียข้อมูลที่ละเอียดในประโยคไปด้วย

2) Word Level Representation คือ การแปลงคำแต่ละคำในข้อความให้เป็นคุณลักษณะ ซึ่งสามารถทำให้ข้อความหลายประเภทถูกเข้ารหัสพร้อมกันในการประมวลผลครั้งเดียว แต่จะทำให้คำที่เป็นประเภทต่าง ๆ มีความสัมพันธ์กัน ซึ่งอาจไม่จำเป็น โดยเฉพาะเมื่อข้อความนั้นเป็นการรวมชื่อประเภทต่าง ๆ เข้าด้วยกันในลำดับที่ไม่แน่นอน

3) Sub-Sentence Level Representation คือ การแสดงข้อมูลในระดับย่อยของประโยค ซึ่งจะช่วยในการแยกความสัมพันธ์ระหว่างคำที่ไม่เกี่ยวข้องออกจากกัน ในขณะที่ยังคงรักษาคุณลักษณะของแต่ละคำเพื่อให้สามารถเข้าใจรายละเอียดได้ดีขึ้น โดยการใช้ attention masks เพื่อบล็อกไม่ให้คำที่ไม่เกี่ยวข้องมีการปฏิสัมพันธ์กันระหว่างการประมวลผล

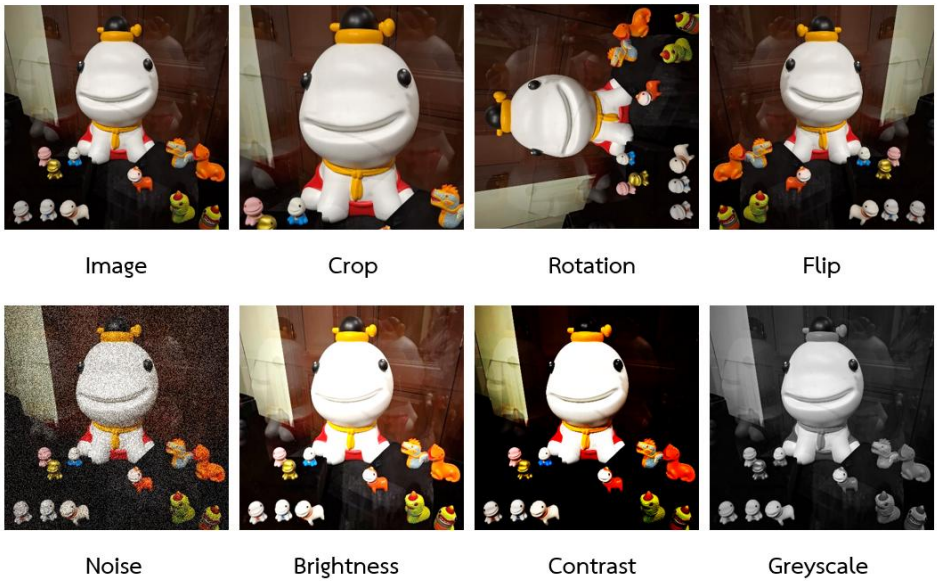
2.8.5 Loss Function

งานที่ใช้ DETR-like จะใช้ L1 Loss และ GIOU Loss สำหรับการถดถอยของกรอบ (Bounding Box) เราปฏิบัติตาม GLIP และใช้ Contrastive Loss ระหว่างวัตถุที่ทำนายและ Token ข้อความสำหรับการจำแนกประเภท (Classification) โดยเฉพาะ การคูณจุด (Dot Product) ของแต่ละ Query กับคุณลักษณะของข้อความเพื่อทำนาย Logits สำหรับแต่ละโทเคนข้อความ แล้วคำนวณ Focal Loss สำหรับแต่ละ Logit การถดถอยของกรอบและค่าใช้จ่ายในการจำแนกประเภทจะถูกใช้ในการจับคู่แบบ Bipartite ระหว่างการทำนายและค่าจริง (Ground Truths) จากนั้นเราคำนวณค่าเสียหายสุดท้ายระหว่างค่าจริงและการทำนายที่ตรงกันโดยใช้ส่วนประกอบของค่าเสียหายเหมือนกัน ตามโมเดลที่ใช้

DETR-Like จะมีการเพิ่ม Auxiliary Loss หลังจากแต่ละเลเยอร์ของ Decoder และหลังจากผลลัพธ์ของ Encoder

2.9 การเสริมข้อมูล (Data Augmentation) [3]

การเสริมข้อมูล (Data Augmentation) เป็นการเพิ่มข้อมูลรูปภาพ หรือการแปลงภาพในข้อมูลการฝึกฝน (Training Data) ให้เป็นภาพที่เปลี่ยนแปลง เช่น ครอบตัด (Crop) หมุน (Rotation) พลิก (Flip) เพิ่มนอยส์ (Noise) ปรับความสว่างของภาพ (Brightness) ปรับคอนทราส (Contrast) หรือทำภาพขาวดำ (Greyscale) จะทำให้ได้ภาพรูปแบบต่าง ๆ ไม่จำกัด เนื่องจากความแม่นยำของโมเดลขึ้นอยู่กับปริมาณข้อมูล เป็นปัจจัยสำคัญหลักเพื่อเพิ่มความน่าเชื่อถือให้กับชุดข้อมูล



รูปที่ 2.28 ตัวอย่างการเสริมข้อมูล (Data Augmentation)

2.10 JavaScript [12]

JavaScript เป็นภาษาโปรแกรมที่มีความนิยมและนักพัฒนาใช้ในการสร้างหน้าเว็บแบบอินเทอร์แอคทีฟ (Interactive) ตั้งแต่การรีเฟรช (Refresh) ฟีดสื่อโซเชียล (Feed Social) ไปจนถึงการแสดงภาพเคลื่อนไหว และแผนที่แบบอินเทอร์แอคทีฟ ฟังก์ชันของ JavaScript สามารถปรับปรุงประสบการณ์ที่ผู้ใช้จะได้รับจากการใช้งานเว็บไซต์ และในฐานะที่เป็นภาษาในการเขียนสคริปต์ฝั่งไคลเอนต์ (Client) จึง

เป็นหนึ่งในเทคโนโลยีหลักของ World Wide Web (www) ยกตัวอย่างเช่น การท่องเว็บแล้วเห็นภาพสไลด์ (Slide) เมนูดร็อปดาวน์แบบคลิกให้แสดงผล (Dropdown) หรือสีองค์ประกอบที่เปลี่ยนแบบไดนามิค (Dynamic) บนหน้าเว็บ นั่นคือเอฟเฟ็กต์ (Effect) ของ JavaScript

JavaScript เกิดขึ้นในฐานะเทคโนโลยีฝั่งเบราว์เซอร์ (Browser) เพื่อทำให้เว็บแอปพลิเคชันมีความเป็นไดนามิคมากขึ้น เมื่อใช้ JavaScript เบราว์เซอร์จะสามารถตอบสนองต่อการโต้ตอบของผู้ใช้และเปลี่ยนแปลงเค้าโครงเนื้อหาบนเว็บเพจได้ เมื่อภาษาผ่านการพัฒนาอย่างเต็มที่ นักพัฒนา JavaScript ก็สร้างไลบรารี (Library) เฟรมเวิร์ค (Framework) และแนวทางปฏิบัติในการเขียนโปรแกรม แล้วเริ่มนำ JavaScript ไปใช้นอกเว็บเบราว์เซอร์ สามารถใช้ JavaScript สำหรับทั่งการพัฒนาฝั่งไคลเอนต์และฝั่งเซิร์ฟเวอร์ (Server)

ภาษาโปรแกรมทั้งหมดทำงานด้วยการแปลไวยากรณ์ที่คล้ายภาษาอังกฤษ เป็นโค้ดสำหรับเครื่อง จากนั้นระบบปฏิบัติการจะเรียกใช้โค้ดนั้น JavaScript ได้รับการจัดประเภทอย่างกว้าง ๆ ว่าเป็นภาษาเขียนสคริปต์ หรือภาษาที่แปลผลแล้ว นั่นคือการแปลโดยตรงเป็นโค้ดภาษาสำหรับเครื่องด้วยกลไกล JavaScript ในขณะที่ในภาษาโปรแกรมอื่น ๆ คอมไพเลอร์ (Compiler) จะคอมไพล์ (Compile) โค้ดทั้งหมดเป็นโค้ดสำหรับเครื่องในขั้นตอนที่แยกต่างหาก ดังนั้น ภาษาเขียนสคริปต์ทั้งหมดจึงเป็นภาษาโปรแกรม แต่ไม่ใช่ว่าภาษาโปรแกรมทั้งหมดจะเป็นภาษาเขียนสคริปต์เสมอไป

2.10.1 กลไก JavaScript

กลไก JavaScript คือ โปรแกรมคอมพิวเตอร์ที่เรียกใช้โค้ด JavaScript และเคยเป็นเพียงตัวแปลผล แต่กลไกสมัยใหม่ทั้งหมดใช้การคอมไพล์แบบ Just-in-time หรือรันไทม์ (Run Time) เพื่อปรับปรุงประสิทธิภาพ

2.10.2 JavaScript ฝั่งไคลเอนต์ (Client)

JavaScript ฝั่งไคลเอนต์หมายถึงวิธีที่ JavaScript ทำงานในเบราว์เซอร์ ในกรณีนี้กลไก JavaScript จะอยู่ภายในโค้ดเบราว์เซอร์ เว็บเบราว์เซอร์ทั้งหมดจะมาพร้อมกับกลไก JavaScript ในตัว นักพัฒนาแอปพลิเคชันเว็บจะเขียนโค้ด JavaScript ที่มีฟังก์ชันที่แตกต่างกันสัมพันธ์กับเหตุการณ์ต่าง ๆ เช่น การคลิกเมาส์ หรือการเลื่อนเมาส์ผ่าน ฟังก์ชันเหล่านี้จะเปลี่ยนแปลง HTML และ CSS ดังนี้

1) เบราว์เซอร์โหลดเว็บเพจเมื่อมีการเยี่ยมชมเว็บเพจ

2) ระหว่างการโหลด เบราว์เซอร์แปลงหน้าและองค์ประกอบทั้งหมดของหน้า เช่น ปุ่ม ป้าย และกล่องดรอปดาวน์ เป็นโครงสร้างข้อมูลที่เรียกว่าโมเดลอ็อบเจกต์ (Model Object) เอกสาร (DOM)

3) กลไก JavaScript ของเบราว์เซอร์แปลงโค้ด JavaScript เป็นไบต์โค้ด โค้ดนี้เป็นตัวกลางระหว่างไวยากรณ์ JavaScript และเครื่อง

4) เหตุการณ์ต่าง ๆ เช่น การคลิกเมาส์บนปุ่ม จะกระตุ้นให้บล็อกโค้ด JavaScript ที่เกี่ยวข้องดำเนินการ จากนั้นกลไกจะแปลผลไบต์โค้ด และทำการเปลี่ยนแปลง DOM

5) เบราว์เซอร์แสดงผล DOM ใหม่

2.10.3 JavaScript ฝั่งเซิร์ฟเวอร์ (Server)

JavaScript ฝั่งเซิร์ฟเวอร์ หมายถึง การใช้ภาษาเขียนโค้ดในลอจิก (Logic) ของเซิร์ฟเวอร์แบ็คเอนด์ (Backend) ในกรณีนี้กลไก JavaScript จะอยู่บนเซิร์ฟเวอร์โดยตรง ฟังก์ชัน JavaScript ฝั่งเซิร์ฟเวอร์สามารถเข้าถึงฐานข้อมูล ดำเนินการทางตรรกะแบบต่าง ๆ และตอบสนองต่อเหตุการณ์ต่าง ๆ ที่ถูกกระตุ้นจากระบบปฏิบัติการของเซิร์ฟเวอร์ ข้อได้เปรียบหลักของการเขียนสคริปต์ฝั่งเซิร์ฟเวอร์ก็คือสามารถปรับแต่งการตอบสนองของเว็บไซต์โดยอ้างอิงตามข้อกำหนด สิทธิ์การเข้าถึง และคำขอข้อมูลจากระบบได้เป็นอย่างมาก

2.11 HTML [13]

HTML หรือ Hypertext Markup Language คือ ภาษาคอมพิวเตอร์ที่ใช้เขียนหน้าเว็บเพจ ถือเป็นตัวกำหนดโครงสร้างและองค์ประกอบของเว็บไซต์ผ่าน Web Browser ซึ่งมีการสร้างรูปแบบและจัดการข้อมูลให้อยู่ในรูปแบบภาษา Markup โดยรูปแบบจะประกอบไปด้วย แท็ก (Tags) หรือคำสั่งที่ใช้สำหรับกำหนดโครงสร้างและเนื้อหาภายในหน้าเว็บเพจ ส่งผลให้หน้าเว็บมีตัวหนังสือ ข้อความ สีสัน และลูกเล่นต่าง ๆ รวมไปถึงมีส่วนในการจัดทำรูปภาพเคลื่อนไหว HTML มีประโยชน์มากมาย นอกจากการสร้างเว็บไซต์ จัดการข้อมูลหรือสร้างเอกสารดิจิทัลต่าง ๆ แล้ว ยังถูกจัดให้เป็นภาษาคอมพิวเตอร์ที่ใช้เขียนโปรแกรมได้อีกด้วย ซึ่งถูกกำหนดมาตรฐานโดยองค์กร World Wide Web Consortium (W3C) และ Microsoft จึงทำให้เรียกกันติดปากว่า “ภาษา HTML”

2.12 CSS [14]

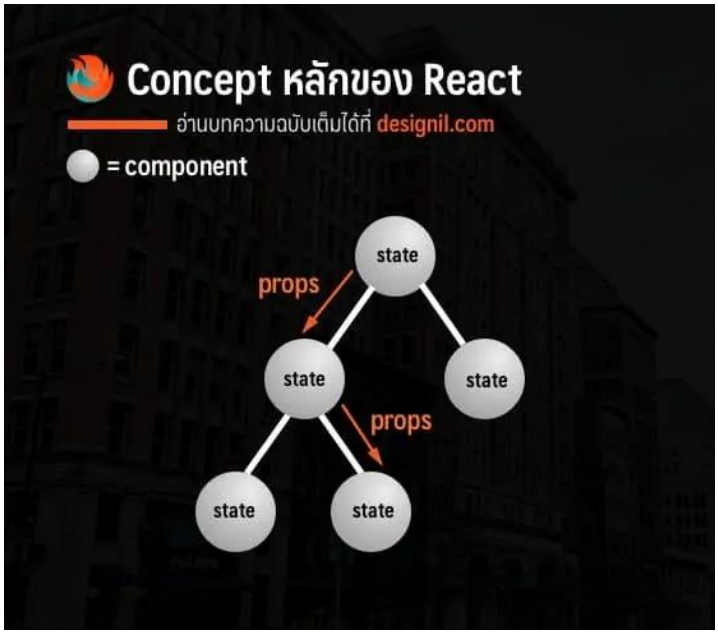
CSS หรือ Cascading Style Sheets คือ ภาษาเว็บที่ใช้สำหรับออกแบบหน้าเว็บไซต์ เพื่อกำหนดสไตล์และรูปแบบต่าง ๆ ให้กับเนื้อหาบนหน้าเว็บไซต์ได้อย่างอิสระ โดยไม่ต้องเปลี่ยนแปลงโครงสร้าง HTML ที่ใช้สร้างเนื้อหานั้น ๆ ซึ่งจะช่วยให้สามารถออกแบบเว็บไซต์ให้สวยงามและมีความสมบูรณ์มากขึ้นได้ เช่น สีพื้นหลัง ขนาดตัวอักษร รูปแบบตาราง รวมถึงการจัดตำแหน่งและการจัดรูปแบบของ

องค์ประกอบต่าง ๆ บนหน้าเว็บไซต์ และเป็นภาษาที่นิยมใช้กันอย่างแพร่หลายในการพัฒนาเว็บไซต์ อีกทั้งยังช่วยให้เว็บไซต์มีความสม่ำเสมอในการแสดงผลบนหลาย ๆ เบราเซอร์ และช่วยลดเวลาในการออกแบบเว็บไซต์ด้วยการใช้ไฟล์ CSS สำเร็จรูปที่สามารถเรียกใช้งานได้หลายหน้าของเว็บไซต์ได้

2.13 React [15]

React คือ JavaScript Library สำหรับการพัฒนาส่วนติดต่อกับผู้ใช้ (User Interface : UI) ที่ตอบสนองต่อการเปลี่ยนแปลงของข้อมูลได้อย่างมีประสิทธิภาพ โดย React ถูกพัฒนาโดย Facebook (Meta) ในปี 2013 และได้รับความนิยมอย่างแพร่หลายสำหรับการพัฒนาเว็บและแอปพลิเคชันแบบไดนามิค คอนเซ็ปต์หลักของ React มี 3 คอนเซ็ปต์ คือ

1) Component เป็นส่วนต่าง ๆ ในเว็บ
2) State เป็นข้อมูลที่อยู่ใน Component แต่ละชิ้น
3) Props (Properties) เป็นข้อมูลที่ถูกส่งต่อจาก Component ชั้นบนลงไปชั้นล่าง



รูปที่ 2.29 คอนเซ็ปต์หลักของ React [15]

การเขียน Component เหมือนกับการเขียน HTML แต่ใน React ใช้สิ่งที่เรียกว่า JSX ในการแสดงผลเว็บไซต์ หน้าตาจะเหมือน HTML มาก แตกต่างตรงที่เขียนเข้าไปในไฟล์ JavaScript แทนไฟล์ HTML ช่วยให้โค้ดอ่านง่ายและเข้าใจง่ายยิ่งขึ้น

2.14 Python [16]

Python คือ เป็นภาษาคอมพิวเตอร์ระดับสูง ที่ถูกปรับมาให้ใช้งานง่าย ทำงานด้วยการแปลชุดคำสั่งทีละบรรทัด (Interpreter) เพื่อป้อนภาษาเหล่านั้นเข้าสู่หน่วยประมวลผลให้คอมพิวเตอร์เข้าใจถึงความต้องการ และทำงานได้ตรงตามจุดประสงค์ ทั้งยังลดความซับซ้อนของภาษาที่ไม่จำเป็นออกไป เพื่อให้ง่ายต่อการเรียนรู้ และใกล้เคียงกับภาษาที่เราใช้ในการสื่อสารมากที่สุด ภาษา Python เองจึงกลายเป็นภาษาโปรแกรมขั้นพื้นฐาน ที่ถูกนำไปต่อยอดและใช้งานได้หลากหลาย ไม่ได้จำกัดเฉพาะทางใดทางหนึ่ง (General-purpose language) นิยมใช้ในองค์กรทั่วไป อย่างที่รู้จักกันดี คือ YouTube, Instagram, Google ฯลฯ นับเป็นภาษาที่นักโปรแกรมเมอร์นิยมมากที่สุด

Python เป็นภาษาโปรแกรมพื้นฐานที่นำไปต่อยอดได้หลายรูปแบบ เนื่องจากมีความยืดหยุ่นคล่องตัวสูง ประยุกต์ได้หลากหลายทิศทาง ตอบโจทย์การทำเว็บไซต์ เขียนโค้ด แชทบอท ทำ Data Science ไปจนถึง Machine Learning Model ทั้งยังมีฟังก์ชันในการใช้งานเยอะ เหมาะกับโปรแกรมเมอร์มือใหม่ ทั้งยังมี Tools และ Library Support ฟรีเยอะ หาข้อมูลได้ง่าย แต่ที่นิยมนำไปใช้งานอย่างแพร่หลาย มีดังต่อไปนี้

1) Python เพื่อการทำเว็บไซต์

เว็บไซต์ไม่ได้พัฒนาขึ้นจาก HTML และ JavaScript เพียงเท่านั้น แต่ปัจจุบัน Python คือภาษาหลักในการพัฒนาโปรแกรมของเว็บไซต์ดังหลากหลายแห่ง ไม่ว่าจะเป็น Spotify, Netflix, Facebook หรืออย่าง Google เอง ก็มีใช้ Python ในระบบหลังบ้าน (Backend) เช่นเดียวกัน

2) Python ในแชทบอท (Chatbot)

แชทบอท (Chatbot) ตัวช่วยอำนวยความสะดวก แบบ “ถามได้ตอบได้” ที่ถูกพัฒนาจากปัญญาประดิษฐ์ เพื่อใช้เป็นเครื่องมือในการอำนวยความสะดวก โดยมีพื้นฐานการพัฒนาระบบมาจากภาษา Python นั่นเอง

2.15 NodeJS [17]

NodeJS คือ Cross Platform Runtime Environment สำหรับฝั่ง Server เป็น Open Source และ Library ที่ใช้สำหรับพัฒนาเว็บแอปพลิเคชันต่าง ๆ ด้วยภาษา JavaScript เหมาะสำหรับการสร้างแอปพลิเคชันที่ต้องการใช้ข้อมูลจำนวนมากและนิยมใช้ในการพัฒนาแอปพลิเคชันที่ใช้ข้อมูลแบบเรียลไทม์ (Realtime) สามารถทำงานได้ทุกระบบปฏิบัติการ โดยถูกนำมาเป็น Web Server, IoT, Webkit, TVOS, OS และอื่น ๆ เป็นต้น

NodeJS ใช้ V8 Engine ที่ถูกพัฒนาโดย The Chromium Project สำหรับเพิ่มประสิทธิภาพการทำงานของภาษา JavaScript ร่วมกับ Web Browser ให้ดีมากขึ้น โดยการใช้หลักการ Compile ก่อนประมวลผล (Just-in-time Compilation) ด้วยการเป็นตัวแปลงโค้ดภาษา JavaScript หรือ JavaScript Engine ให้เป็น Machine Code ทำให้สามารถทำงานนอก Browser อื่นได้ เนื่องจากตามปกติแล้ว JavaScript สามารถรันได้บน Client เท่านั้น

NodeJS ทำงานแบบ Single Process โดยมี Event-loop เข้ามาช่วยในการทำงานแบบ Asynchronous คือ รูปแบบการทำงานของชุดคำสั่งที่เขียนขึ้นมา โดยทำงานแบบไม่เรียงขั้นตอน เนื่องจากชุดคำสั่งทำงานพร้อมกัน และเมื่อคำสั่งไหนเสร็จเรียบร้อยจะแสดงผลลัพธ์ก่อนแบบ Non-Blocking I/O สามารถส่ง Request ของ User 1 และ User 2 พร้อมกันได้เลย ทำให้ลดการใช้ Thread ได้ โดย NodeJS ไม่เหมาะสำหรับการทำงานที่เป็น CPU Intensive เพราะทำให้ถูก Block การทำงานทั้งหมด

ประโยชน์ของ NodeJS มีดังนี้

1) มีเครื่องมือที่สะดวกและรวดเร็วในการจัดการ Package อย่าง NPM (Node Package Manager) หรือ YARN (Dependency Management Tool) ช่วยลดเวลาในการเขียนโค้ดใหม่ทั้งหมด ทำให้สามารถทำงานได้อย่างมีประสิทธิภาพมากขึ้น

2) พัฒนาได้อย่างครอบคลุมทั้ง Frontend และ Backend โดยตัวอย่าง Framework และ Library ฝั่ง Frontend เช่น ReactJS, VueJS เป็นต้น และตัวอย่างฝั่ง Backend เช่น Express, NestJS, Meteor เป็นต้น โดยนักพัฒนาเรียนรู้แค่ภาษา Javascript สามารถเริ่มต้นพัฒนาแบบ Fullstack ได้แล้ว

3) NodeJS ใช้ภาษา JavaScript ซึ่งเป็นภาษายอดนิยมและเป็นที่ต้องการสูงของสายงาน Programming

2.16 PostgreSQL [18]

PostgreSQL คือ ฐานข้อมูลเชิงสัมพันธ์โอเพ่นซอร์ส (Open Source) ระดับองค์กรขั้นสูงที่รองรับการค้นหาทั้งแบบเชิงสัมพันธ์ คือ SQL และแบบไม่สัมพันธ์ คือ JSON เป็นระบบจัดการฐานข้อมูลที่มีเสถียรภาพสูงซึ่งได้รับการสนับสนุนจากการพัฒนาชุมชนมากกว่า 20 ปี แนวทางที่ละเอียดถี่ถ้วนและร่วมมือกันนี้ช่วยให้มีระดับความยืดหยุ่น ความสมบูรณ์ และความถูกต้องสูง PostgreSQL ถูกใช้เป็นที่เก็บข้อมูลหลักหรือคลังข้อมูลสำหรับแอปพลิเคชันเว็บ มือถือ ภูมิสารสนเทศ และการวิเคราะห์มากมาย

PostgreSQL มีชุดคุณลักษณะที่แข็งแกร่ง รวมถึงพื้นที่ในตาราง การจำลองแบบอะซิงโครนัส (Asynchronous) ธุรกรรมแบบซ้อน การสำรองข้อมูลออนไลน์/แบบร้อน และตัววางแผนกับตัวเพิ่มประสิทธิภาพแบบสอบถามที่ปรับแต่งแล้ว ชุมชน PostgreSQL ยังได้พัฒนาส่วนขยายที่ขยายการทำงานของฐานข้อมูล และฐานข้อมูล PostgreSQL ยังรองรับ

- ชุดอักขระสากล การเข้ารหัสอักขระหลายไบต์ (Byte) และ Unicode
- ประเภทข้อมูล SQL:2008 ส่วนใหญ่ ได้แก่ INTEGER, NUMERIC, BOOLEAN, CHAR, VARCHAR, DATE, INTER-VAL และ TIMESTAMP
- การจัดเก็บวัตถุไบนารี (Binary) ขนาดใหญ่ รวมถึงรูปภาพ เสียง วิดีโอ และแผนที่
- คีย์ต่างประเทศ การรวม มุมมอง ทริกเกอร์ และกระบวนการที่จัดเก็บไว้
- ภาษาการเขียนโปรแกรมและโปรโตคอลชั้นนำ ได้แก่ Python, Java, Perl, .Net, Go, Ruby, C/C++, Tcl และ ODBC

เซิร์ฟเวอร์ฐานข้อมูล PostgreSQL คำนึงถึงตำแหน่งสำหรับการเรียงลำดับ ความไวต่อตัวพิมพ์เล็ก-ใหญ่ และการจัดรูปแบบ เซิร์ฟเวอร์ฐานข้อมูล PostgreSQL สามารถปรับขนาดได้สูงทั้งในปริมาณข้อมูลที่สามารถจัดการได้และจำนวนผู้ใช้พร้อมกันที่สามารถรองรับได้

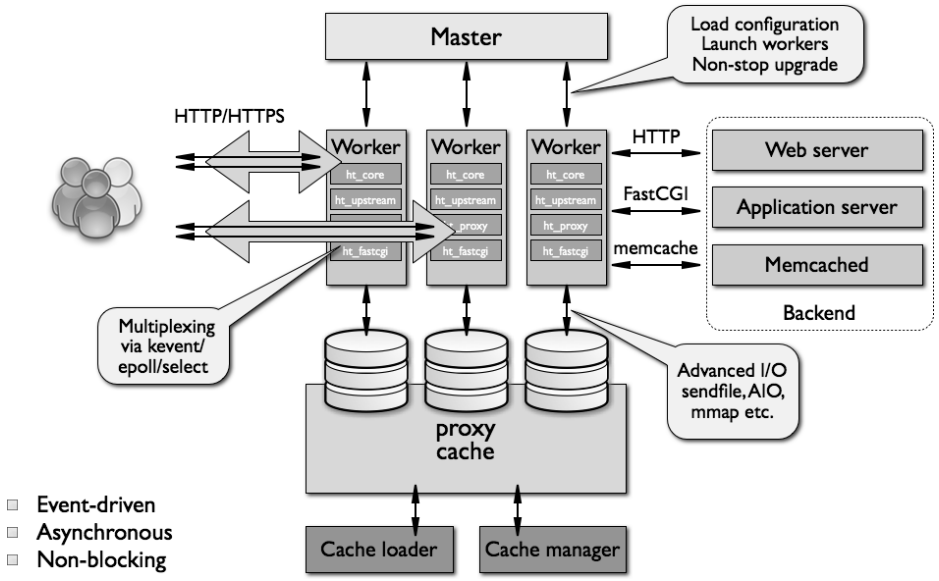
2.17 NGINX [19]

NGINX เป็นซอฟต์แวร์โอเพ่นซอร์ส (Open Source) สำหรับการให้บริการเว็บ การทำรีเวิร์สพร็อกซี (Reverse Proxy) การแคช (Cach) การปรับสมดุลโหลด การสตรีมมีเดีย และอื่น ๆ อีกมากมาย เริ่มต้นในฐานะเว็บเซิร์ฟเวอร์ที่ออกแบบมาเพื่อประสิทธิภาพและความเสถียรสูงสุด นอกเหนือจากความสามารถในการเป็นเซิร์ฟเวอร์ HTTP แล้ว NGINX ยังสามารถทำหน้าที่เป็นพร็อกซีเซิร์ฟเวอร์สำหรับอีเมล (IMAP, POP3 และ SMTP) รวมถึงเป็นรีเวิร์สพร็อกซีและตัวปรับสมดุลโหลดสำหรับเซิร์ฟเวอร์ HTTP, TCP และ UDP

NGINX พัฒนาขึ้นมาเพื่อแก้ปัญหา C10K ซึ่งเป็นคำที่เกิดขึ้นในปี 1999 เพื่ออธิบายความยากลำบากที่เว็บเซิร์ฟเวอร์ในขณะนั้นเผชิญในการจัดการกับการเชื่อมต่อพร้อมกันจำนวนมาก (10,000 การเชื่อมต่อ) ด้วยสถาปัตยกรรมที่ขับเคลื่อนด้วยเหตุการณ์ (event-driven) และการประมวลผลแบบอะซิงโครนัส (Asynchronous) NGINX ได้ปฏิวัติวิธีการทำงานของเซิร์ฟเวอร์ในบริบทที่ต้องการประสิทธิภาพสูง และกลายเป็นเว็บเซิร์ฟเวอร์ที่เร็วที่สุดในยุคนั้น ปัจจุบัน NGINX สามารถรองรับการเชื่อมต่อพร้อมกันได้นับแสนครั้ง และยังเป็นพลังขับเคลื่อนเว็บไซต์ที่มีปริมาณการใช้งานสูงที่สุดบนอินเทอร์เน็ตมากกว่าเซิร์ฟเวอร์ใด ๆ

เป้าหมายของ NGINX คือการสร้างเว็บเซิร์ฟเวอร์ที่เร็วที่สุด และการรักษามาตรฐานความยอดเยี่ยมนี้ยังคงเป็นเป้าหมายหลักของโครงการเสมอมา NGINX มีประสิทธิภาพเหนือกว่า Apache และเซิร์ฟเวอร์อื่น ๆ อย่างต่อเนื่องในการวัดผลด้านประสิทธิภาพของเว็บเซิร์ฟเวอร์ ตั้งแต่การเปิดตัวครั้งแรกของ NGINX เว็บไซต์ได้พัฒนาจากหน้า HTML แบบธรรมดาไปสู่เนื้อหาแบบไดนามิคที่มีความซับซ้อนมากขึ้น NGINX ก็ได้เติบโตควบคู่ไปกับการเปลี่ยนแปลงนี้ ปัจจุบัน NGINX รองรับทุกองค์ประกอบของเว็บสมัยใหม่ รวมถึง WebSocket, HTTP/2, gRPC และการสตรีมวิดีโอในหลายรูปแบบ (HDS, HLS, RTMP และอื่น ๆ)

นอกจากนี้ NGINX ยังถูกวางไว้ระหว่างไคลเอนต์และเว็บเซิร์ฟเวอร์ตัวที่สอง เพื่อทำหน้าที่เป็นตัวจัดการ SSL/TLS (SSL/TLS terminator) หรือเว็บแอคเซลเลอเรเตอร์ (Web Accelerators) โดยทำหน้าที่เป็นตัวกลาง NGINX สามารถจัดการงานที่อาจทำให้เว็บเซิร์ฟเวอร์ช้าลงได้อย่างมีประสิทธิภาพ เช่น การต่อรอง SSL/TLS การบีบอัด หรือการแคชเนื้อหาเพื่อปรับปรุงประสิทธิภาพ สำหรับเว็บไซต์ไดนามิคที่พัฒนาด้วยเทคโนโลยีต่าง ๆ ตั้งแต่ Node.js ไปจนถึง PHP โดยส่วนมากจะใช้ NGINX เป็นตัวแคชเนื้อหาและรีเวิร์สพร็อกซี เพื่อลดภาระของเซิร์ฟเวอร์แอปพลิเคชันและเพิ่มประสิทธิภาพการใช้งานฮาร์ดแวร์ที่มีอยู่ให้ได้มากที่สุด



รูปที่ 30 สถาปัตยกรรมของ NGINX [20]

จากรูปที่ 30 [20] ในสถาปัตยกรรมของ NGINX, Master Process เป็นกระบวนการหลักที่ดูแลการทำงานของ Worker Processes และการจัดการการตั้งค่าเซิร์ฟเวอร์ เช่น การโหลดไฟล์คอนฟิกกูเรชันใหม่ (Configuration) การควบคุมสัญญาณ (Signals) เช่น การหยุดหรือรีสตาร์ทเซิร์ฟเวอร์ และการจัดการการบันทึกข้อผิดพลาด (Error Logging) ส่วน Worker Processes ทำหน้าที่ประมวลผลคำขอที่เข้ามาจากผู้ใช้ (Clients) และเชื่อมต่อกับ Backend-Server หรือ Proxy Servers เพื่อดึงข้อมูลและส่งกลับไปยังผู้ใช้ การทำงานของ Worker ใช้การประมวลผลแบบ Event-Driven และ Asynchronous ซึ่งทำให้สามารถจัดการการเชื่อมต่อหลายพันพร้อมกันได้อย่างมีประสิทธิภาพ

1) Event-Driven Architecture ซึ่งช่วยให้สามารถจัดการกับการเชื่อมต่อจำนวนมากได้พร้อมกันโดยไม่ต้องสร้างเธรดหรือกระบวนการใหม่ทุกครั้งที่มีการเชื่อมต่อเข้ามา

2) Non-blocking I/O ช่วยให้ NGINX สามารถจัดการกับคำขอหลาย ๆ รายการในเวลาเดียวกันได้โดยไม่ต้องรอให้การประมวลผลคำขอใดคำขอหนึ่งเสร็จสิ้นก่อน ทำให้กระบวนการทำงานไม่ถูกขัดขวางและเพิ่มประสิทธิภาพการทำงาน

3) Multiplexing ทำให้สามารถรับคำขอจากหลายการเชื่อมต่อในเวลาเดียวกัน โดยไม่ต้องสร้างเธรดใหม่สำหรับการเชื่อมต่อแต่ละรายการ

การประมวลผลในรูปแบบ Asynchronous ช่วยให้ Worker สามารถจัดการกับคำขอหลายคำขอในลูป (Loop) เดียว ทำให้สามารถรองรับการจราจรที่มีปริมาณสูงได้อย่างมีประสิทธิภาพโดยไม่ต้องใช้ทรัพยากรเกินความจำเป็น

2.18 Docker [21]

Docker เครื่องมือแบบ Open-source ที่ช่วยจำลองสภาพแวดล้อม (Environment) ในการรัน Service หรือ Server ตามหลักการสร้าง Container เพื่อจัดการกับ Library ต่าง ๆ อีกทั้งยังช่วยจัดการในเรื่องของ Version Control เพื่อง่ายต่อการจัดการกับปัญหาต่างๆ ที่เกิดขึ้น ซึ่งในปัจจุบันในโลกของการพัฒนา Software มีรูปแบบการทำงานแบบ Agile ที่เน้นความรวดเร็วในการส่งมอบงานในแต่ละขั้นตอน Docker จึงเป็นที่รู้จักในวงกว้างและเริ่มเข้ามามีบทบาทอย่างมากในโลกของการพัฒนา Software อีกทั้งยังเป็นเครื่องมือที่จำเป็นสำหรับการทำ DevSecOps

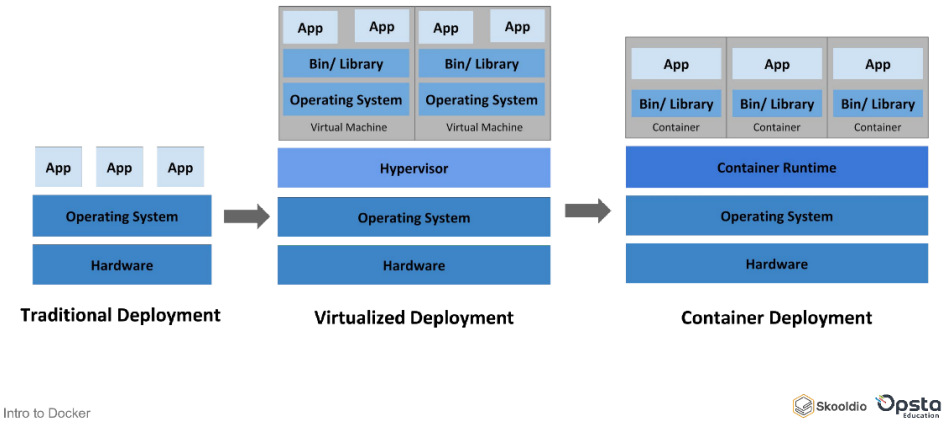
การ Deployment เป็นหนึ่งในขั้นตอนการทำงานที่นักพัฒนาทุกคนต้องเจอไม่ว่าจะเป็นองค์กรขนาดเล็กหรือขนาดใหญ่ แต่สิ่งที่แต่ละองค์กรไม่เหมือนกันคือความยุ่งยากซับซ้อน ระยะเวลาในการ Deploy ที่ต่างกัน ขึ้นอยู่กับ Process และเครื่องมือที่ใช้ ซึ่ง Docker ก็เป็นหนึ่งในเครื่องมือยอดนิยมที่จะช่วยให้การ Deployment รวดเร็วมากยิ่งขึ้น

วิวัฒนาการของการ Deploy มีดังนี้

1) Traditional Deployment เป็นยุคที่ใช้ Physical server 1 เครื่อง ในการ Deploy และเพื่อความคุ้มค่า Physical Server ที่เรามีจะถูกใช้ในการลงแอปพลิเคชันหลาย ๆ อันพร้อมกัน ซึ่งทำให้เกิดปัญหา Element ของแต่ละแอปพลิเคชันตีกัน เช่น แอปพลิเคชันแต่ละตัวมีการลง JAVA ซึ่งเป็นในเครื่องเรามี JAVA หลายเวอร์ชัน ทำให้เวลา Run มีปัญหาเกิดขึ้น ทั้งการบำรุงรักษา (Maintenance) และปัญหาในการเลือกเวอร์ชัน

2) Virtualized Deployment เป็นยุคที่มีการเกิดขึ้นของ Software Hypervisor ซึ่ง Concept คือ การจำลองเครื่อง Physical server ขึ้นมา เรียกว่า Virtual Machine (VM) ทั้ง CPU, memory, Hard Disk, Hardware ต่าง ๆ ขึ้นมาเสมือนคอมพิวเตอร์เลย ปรับสเปก (Spec) ปรับความเร็วต่าง ๆ ได้ตามงบประมาณที่มี ซึ่งส่วนใหญ่ก็จะสร้าง Virtual Machine ขึ้นมาหลาย ๆ เครื่อง ให้แต่ละเครื่องเพียงพอต่อการ Run แอปพลิเคชันแต่ละตัว แต่ปัญหาคือจะเกิดปัญหา Overhead ทั้งในเรื่องการจำลอง Hardware ต่าง ๆ ทำให้ทำงานได้ช้าลง 2-5 เท่า รวมถึง Overhead ในกรณีที่เรามีการ Run ในสภาพแวดล้อมที่ใกล้เคียงกัน จะทำให้เปลือง Resource ไปโดยใช่เหตุ

3) Container Deployment ซึ่งในยุคนี้จะพูดถึงการสร้าง Container ขึ้นมาเพื่อขังทรัพยากร (Resource) สร้างกำแพงขึ้นมาแบ่งทรัพยากรทำให้เราสามารถใช้ทรัพยากรได้อย่างมีประสิทธิภาพมากขึ้น เลือก Container ไปใช้กับแต่ละแอปพลิเคชันได้ดียิ่งขึ้น ทำให้ปัญหา Overhead ลดลง และเครื่องมือในการสร้าง Container ที่นิยมก็คือ Docker



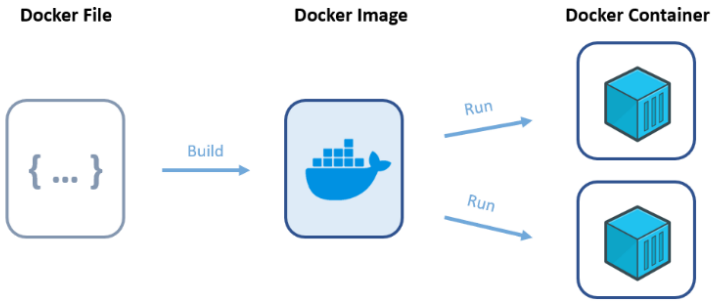
รูปที่ 2.31 วิวัฒนาการของการ Deploy [21]

องค์ประกอบพื้นฐาน Docker

1) Docker File คือ เอกสารบอกโค้ดคำสั่ง สำหรับสร้าง Docker Image นั้น ๆ

2) Docker Image เป็นแม่แบบที่ใช้ในการสร้างเป็น Docker Container ซึ่งประกอบไปด้วยแอปพลิเคชันต่างๆ ที่จะทำงานเมื่อมีการเรียกใช้งานจาก Docker Container นั้น ๆ รวมทั้งการตั้งค่าจำลองสภาพแวดล้อม (Environment) ที่จำเป็นสำหรับการทำงานของมันไว้ด้วย

3) Docker Container เป็นที่บรรจุรวมของแอปพลิเคชัน สภาพแวดล้อมที่จำเป็นต่อการทำงาน และองค์ประกอบต่าง ๆ ที่จำเป็นต่อการทำงานของมัน ซึ่งสามารถสร้างจาก Docker Image ผ่านการกำหนดโครงสร้างของมันที่ Docker File



รูปที่ 2.32 องค์ประกอบพื้นฐาน Docker [21]

ข้อดีของ Docker มีดังนี้

- Portability ทดสอบ Container ที่เดียวสามารถ Deploy ได้ทุกที่ที่มี Docker รันอยู่โดยไม่ต้องกลัวว่าจะไม่สามารถรันได้
- Performance เนื่องจาก Container ไม่ได้มีการบรรจุ OS เข้าไปด้วย นั่นหมายความว่า Docker นั่นจะมีขนาดเล็กกว่า VM ทำให้ขนาดเล็ก Build ได้เร็วกว่า รวมถึงการรันได้มีประสิทธิภาพดีกว่าด้วย
- Agility ด้วย Portability และ Performance ช่วยให้เหมาะสมกับการทำ Agile Process รวมถึงเหมาะกับการทำ CI/CD อีกด้วย ช่วยให้ Compile, Build และ Test ได้ดียิ่งขึ้น
- Scalability เราสามารถสร้าง Container ใหม่ ได้ตามความต้องการของแอปพลิเคชันที่ Scale ได้โดยใช้เวลาอันสั้น

ข้อเสียของ Docker มีดังนี้

- เนื่องจากการรัน Docker ไม่ได้รัน OS ใหม่ทั้งหมดเป็นเพียงแค่การจำลองสภาพแวดล้อม ทำให้อาจเกิดการโจมตีที่ OS หลักผ่านทาง Docker ได้และอาจกระทบกับ Container ตัวอื่น ๆ
- ตอนเริ่มแรกที่ Docker ถูกสร้าง มันถูกออกแบบมาเพื่อรองรับการรันบน Linux เท่านั้น ที่สามารถรัน Docker บน Window และ Mac ได้นั้นเพราะเมื่อลง Docker ใน Window และ Mac จะมีการสร้าง Virtual Machine ที่เป็น Linux เพื่อมารัน Docker อีกที ทำให้ประสิทธิภาพการทำงานอาจจะไม่สามารถทำได้สูงสุดเท่ารันบน Linux

- Docker ไม่เหมาะกับการจัดการทรัพยากรบนเครื่องใหญ่ ๆ หรือไม่เหมาะกับโปรแกรมที่ออกแบบมาเพื่อทำงานบน Virtual Machine
- Learn Curve ที่สูง เนื่องจากเป็นการทำงานเกี่ยวกับ OS และ Network รวมถึงการจัดการทรัพยากรต่าง ๆ ทำให้ต้องอาศัยเวลาการเรียนรู้ที่ค่อนข้างสูง แต่ทาง Docker ก็มี Tool ใหม่ ๆ ออกมาช่วยเหลือให้ใช้งานได้ง่ายยิ่งขึ้น แต่การที่จะใช้งาน Docker ได้อย่างชำชองจำเป็นต้องเรียนรู้ Tools อื่นเพื่อใช้ในการประกอบด้วย

2.19 งานวิจัยที่เกี่ยวข้อง

2.19.1 การใช้โมเดล YOLO ประเมินการตรวจจับข้อบกพร่องผสมของชุดข้อมูลในแผ่น PCB [22]

ในงานวิจัยนี้ ผู้วิจัยได้ทำการสร้างโมเดลสำหรับการตรวจจับข้อบกพร่องแบบผสมในแผ่น PCB ด้วยการสร้างชุดข้อมูลเฉพาะสำหรับการตรวจจับและจำแนกประเภทข้อบกพร่อง ซึ่งเป็นชุดข้อมูลที่สร้างข้อบกพร่องของ PCB ขึ้นมาด้วยเจตนา โดยชุดข้อมูลประกอบไปด้วยภาพขนาด 640x640 พิกเซล ที่ผ่านการประมวลผลหลายขั้นตอน ได้แก่ การติดป้ายข้อมูล การเตรียมข้อมูล การตรวจจับข้อบกพร่อง และการจำแนกประเภท ที่ได้รับการฝึกฝนบนชุดข้อมูล COCO ก่อน หลังจากนั้นจึงใช้แพลตฟอร์ม Roboflow ในการติดป้ายข้อมูล ทำให้ได้ชุดข้อมูลที่ประกอบด้วยภาพ PCB จำนวน 1,741 ภาพ ซึ่งในแต่ละภาพมีข้อบอกพร่องสองถึงสามประเภท รวมทั้งหมดมีข้อบกพร่องแตกต่างกัน 3,704 ข้อ โดยที่การจำแนกประเภทเหล่านี้เป็นข้อบกพร่องที่พบบ่อยในการผลิต PCB

เนื่องจากชุดข้อมูลที่มีอยู่เป็นภาพที่มีการติดป้ายข้อมูลบกพร่องใน PCB จำนวน 693 ภาพ ซึ่งมีความละเอียดสูงถึง 2777x2138 พิกเซล โดยครอบคลุมประเภทข้อบกพร่องต่าง ๆ เช่น รูหาย หนูกัด วงจรเปิด ลัดวงจร การปูด และทองแดงผิดปกติ ส่งผลให้มีข้อบกพร่องรวมทั้งหมด 2,953 ข้อ แต่ด้วยภาพความละเอียดที่สูงและมุ่งเน้นข้อบกพร่องที่ประเภทเดียว จึงทำให้การฝึกฝนช้าและจำกัดความหลากหลายของประเภทข้อบกพร่องใน PCB

จึงเป็นสาเหตุในการสร้างชุดข้อมูลดังกล่าวขึ้นมา และในการเตรียมข้อมูล เพื่อเพิ่มความหลากหลายของชุดข้อมูล ผู้วิจัยได้ใช้วิธีการขยายข้อมูล 6 วิธี ได้แก่ การเพิ่มนอยส์แบบเกาส์เซียน (Gaussian Noise) การปรับแสง การหมุนภาพ การพลิกภาพ การตัดภาพแบบสุ่ม และการย้ายภาพ

หลังจากการเตรียมข้อมูล จึงทำให้ชุดข้อมูลมีภาพทั้งหมด 8,705 ภาพ และมีข้อบกพร่องทั้งหมด 18,520 ข้อ

ในการฝึกฝนโมเดล ผู้วิจัยได้ใช้โมเดลของ YOLO ในการฝึกฝน และได้ทดลองในหลายเวอร์ชัน ตั้งแต่ YOLOv5 ถึง YOLOv8 ซึ่งผลจากการเปรียบเทียบของแต่ละเวอร์ชันสรุปได้ว่า YOLOv7n มีการใช้งานหน่วยความจำสูงสุดที่ 71.3 MB และ YOLOv5n มี FPS ที่เร็วที่สุดที่ 120.69 และใช้หน่วยความจำต่ำที่สุดที่ 3.87 MB ผู้วิจัยจึงเลือกใช้โมเดล YOLOv5 ซึ่งให้ผลลัพธ์ดีที่สุดในการฝึกฝน ชุดข้อมูลถูกแบ่งออกเป็นชุดการฝึกฝน การตรวจสอบ และการทดลองในอัตราส่วน 8:1:1 ตามลำดับ สำหรับการประเมินผล เพื่อให้มั่นใจในการเรียนรู้ที่มีประสิทธิภาพ และฝึกโมเดลที่ 300 epoch เพื่อรักษาความสอดคล้องและปรับแต่งประสิทธิภาพในชุดข้อมูล และมีการใช้การเรียนรู้แบบถ่ายทอด (Transfer Learning) เพื่อเร่งกระบวนการฝึกฝนและปรับปรุงความแม่นยำในการตรวจจับวัตถุ

ในการทดลองโมเดล ผู้ทดลองใช้บอร์ด Jetson Nano ในการติดตั้งโมเดล และปรับแต่งโมเดล YOLOv5n ให้เหมาะสมกับบอร์ด โดยใช้สคริปต์ Python และได้แปลงโมเดลจากการฝึกใน PyTorch เป็น TensorRT ทำให้ได้โมเดลที่มีขนาดเล็กลง ลดการใช้หน่วยความจำและความต้องการในการคำนวณได้อย่างมาก ส่งผลให้โมเดลมีขนาดเพียง 3.87 MB และเวลาในการอนุมานที่ 33.32 ms จึงช่วยให้ตรวจจับข้อบกพร่องได้อย่างรวดเร็ว ทำให้เหมาะกับสภาพแวดล้อมอุตสาหกรรมที่ต้องการการตรวจจับที่รวดเร็วและแม่นยำ

2.19.2 การตรวจจับและจัดประเภทของพื้นที่ที่มีข้อบกพร่องบนชิ้นส่วนโลหะโดยใช้การผสมผสานระหว่าง Faster R-CNN และ Shape From Shading [23]

ในงานวิจัยนี้นำเสนอการตรวจจับและจำแนกข้อบกพร่องบนพื้นผิวโลหะโดยใช้การผสมผสานระหว่าง Faster R-CNN และ Shape From Shading (SFS) เพื่อเพิ่มความแม่นยำและลดเวลาการติดฉลากข้อมูลแบบดั้งเดิมซึ่งใช้แรงงานมนุษย์ การตรวจสอบคุณภาพในอุตสาหกรรมมีความสำคัญอย่างมาก โดยเฉพาะอย่างยิ่งในอุตสาหกรรมที่เกี่ยวข้องกับโลหะที่ต้องการการควบคุมคุณภาพสูง การตรวจจับข้อบกพร่องเป็นปัญหาที่ซับซ้อนเนื่องจากพื้นผิวโลหะสามารถสะท้อนแสงและได้รับผลกระทบจากสิ่งแวดล้อมได้ง่าย วิธีแบบดั้งเดิม เช่น การใช้ตัวกรอง Gabor หรือ Fourier Transform แม้จะสามารถตรวจจับข้อบกพร่องได้ แต่กลับมีข้อจำกัดเมื่อต้องตรวจจับข้อบกพร่องที่มีลักษณะซับซ้อนและแตกต่างกัน

การผสมผสานเทคนิค SFS ช่วยให้สามารถสร้างแบบจำลองเชิงลึกของพื้นผิวและเพิ่มประสิทธิภาพการระบุข้อบกพร่องได้ดีขึ้น SFS วิเคราะห์รูปแบบแสงที่ตกกระทบและสะท้อนจากพื้นผิวเพื่อสร้างภาพสามมิติ ซึ่งทำให้สามารถตรวจจับข้อบกพร่องที่ซ่อนอยู่ได้อย่างแม่นยำมากขึ้น จากนั้น

Faster R-CNN จะถูกใช้เพื่อตรวจจับและจำแนกประเภทของข้อบกพร่อง โดย Faster R-CNN มีข้อได้เปรียบเหนือเทคนิคการตรวจจับอื่นๆ เช่น YOLO เนื่องจากสามารถระบุขอบเขตของข้อบกพร่องได้แม่นยำกว่า ผลลัพธ์ของการวิจัยพบว่าโมเดลที่พัฒนาโดยใช้ SFS และ Faster R-CNN สามารถเพิ่มความแม่นยำเฉลี่ย (mAP) ได้ถึง 0.83 ซึ่งสูงกว่าวิธีการตรวจจับข้อบกพร่องแบบอื่น

งานวิจัยนี้ใช้ชุดข้อมูลจาก Northeastern University (NEU) ซึ่งมีตัวอย่างข้อบกพร่อง 6 ประเภท ได้แก่ รอยแตกร้าว (Crazing) จุดด่าง (Patches) การปนเปื้อน (Inclusions), รอยขีดข่วน (Scratches) พื้นผิวเป็นหลุม (Pitted surface) และข้อบกพร่องที่เกิดจากกระบวนการรีด (Rolled-In Scale) โมเดลได้รับการฝึกด้วยภาพจำนวน 1440 ภาพ และทดสอบกับ 360 ภาพ ผลลัพธ์แสดงให้เห็นว่าโมเดลสามารถตรวจจับข้อบกพร่องได้อย่างแม่นยำ และลดเวลาการติดฉลากข้อมูลได้อย่างมีประสิทธิภาพ

สำหรับแนวทางในอนาคต ผู้จัดทำงานวิจัยนี้เสนอว่าการใช้แหล่งกำเนิดแสงหลายจุด (Photometric Stereo) อาจช่วยให้ได้ผลลัพธ์ที่ดียิ่งขึ้น นอกจากนี้ การพัฒนาให้รองรับการทำงานแบบออนไลน์ผ่านอินเทอร์เฟซบนเว็บอาจทำให้สามารถตรวจจับข้อบกพร่องได้แบบเรียลไทม์และสะดวกยิ่งขึ้น อีกทั้งยังสามารถขยายขีดความสามารถของ Faster R-CNN ไปสู่ Mask R-CNN เพื่อให้สามารถจำแนกประเภทของข้อบกพร่องในระดับพิกเซลได้ การศึกษานี้ถือเป็นก้าวสำคัญในการนำปัญญาประดิษฐ์มาใช้ในการตรวจสอบคุณภาพใน อุตสาหกรรม และสามารถประยุกต์ใช้กับวัสดุอื่น ๆ นอกเหนือจากโลหะ เช่น กระจกและผ้า