CPE378 Machine Learning: Final Project Report

Thai Traffic Signs Recognition

เสนอ

รศ.ดร.พีรพล ศิริพงศ์วุฒิกร

ผศ.ดร.สันติธรรม พรหมอ่อน

ดร.อัญชลิสา แต้ตระกูล

จัดทำโดย

กัญจน์ชยาภรณ์ แซ่จุง	62070505201
ณิชาพัชร์ นบนอบ	62070505203
นัทธ์ชนภัทร พรหมณะ	62070505208
วายุ รักษ์วงศ์สิริ	62070505215
ฐานิตาพัชร ทวีชลพิสิฐ	62070505230

รายงานฉบับนี้เป็นส่วนหนึ่งของรายวิชา CPE 378 การเรียนรู้ของเครื่อง (Machine Learning)

สาขาวิชาวิทยาศาสตร์ข้อมูลสุขภาพ มหาวิทยาลัยเทคโนโลยีพระจอมเกล้าธนบุรี

ภาคเรียนที่ 2 ปีการศึกษา 2564

สารบัญ

บทนำ

รถยนต์ขับเคลื่อนอัตโนมัติ (Self-Driving Cars) เป็นเทคโนโลยีที่มีความน่าสนใจอย่างมากในปัจจุบัน เพราะเทคโนโลยีนี้ทำให้เกิดความสะดวกสบายต่อตัวผู้ขับขี่เมื่อต้องมีการเดินทางที่ต้องใช้ระยะเวลายาวนานบนท้องถนน รวมถึงสามารถช่วยลดอุบัติเหตุบนท้องถนนได้ และใช้เวลาได้อย่างมีประโยชน์มากยิ่งขึ้น เนื่องจากไม่ต้องขับรถด้วยตัวเอง ทำให้สามารถใช้เวลาบนรถทำสิ่งที่ต้องการได้ ซึ่งรถยนต์ขับเคลื่อนอัตโนมัติจะสามารถขับเคลื่อนได้อย่างปลอดภัยและไม่เกิดอุบัติเหตุได้นั้น โดยรถยนต์จะต้องสามารถปฏิบัติตามกฎจราจรได้อย่างถูกต้องและสามารถตัดสินใจได้อย่างมีเหตุและผลตามสภาพแวดล้อมที่พบเจอ ณ ขณะนั้น เช่น สามารถจำแนกตีความหมายของสัญลักษณ์ป้ายจราจรและปฏิบัติตามได้อย่างถูกต้อง เป็นต้น

ปัจจุบันมีงานวิจัยจำนวนมากพยายามศึกษาวิจัยการตีความหมายสัญลักษณ์ป้ายจราจรจากการประมวลผลภาพเป็นจำนวนมาก โดยมีวัตถุประสงค์เพื่อวิจัยและพัฒนาขั้นตอนวิธีการเรียนรู้และจดจำสัญลักษณ์ป้ายจราจรให้มีความแม่นยำ เพื่อให้สามารถนำไปประยุกต์ใช้ในรถยนต์ขับเคลื่อนอัตโนมัติได้อย่างมีประสิทธิภาพ ดังนั้นงานวิจัยนี้จึงต้องการศึกษาและเปรียบเทียบแบบจำลองที่มีความแตกต่างกันคือ Convolutional Neural

Network, Support Vector Machine และ Random Forest สำหรับทำการจำแนกสัญลักษณ์ป้ายจราจรและทำการเปรียบเทียบว่แต่ละาแต่ละแบำลองมีประสิทธิภาพแตกต่างกันอย่างไรและแบบจำลองใดมีประสิทธิภาพดีที่สุด

ชุดข้อมูลรูปภาพ (Dataset)

ชุดข้อมูลรูปภาพที่นำมาใช้ร่วมกับแบบจำลองในงานวิจัยนี้จะใช้รูปสัญลักษณ์ป้ายจราจรของประเทศไทยทั้งหมด 10 ประเภท ที่พบเห็นได้บ่อยตามท้องถนนและทางผู้จัดทำมีความเห็นว่ามีความสำคัญต่อการขับเคลื่อนของรถยนต์ขับเคลื่อนอัตโนมัติ ได้แก่ ป้ายเตือนทางโทตัดเอก ป้ายคนข้ามถนน ป้ายเตือนวงเวียนข้างหน้า ป้ายจุดกลับรถ ป้ายทางม้าลาย ป้ายเขตโรงเรียน ป้ายจำกัดความเร็ว 30 ป้ายจำกัดความเร็ว 80 ป้ายซ้ายผ่านตลอด และป้ายสัญญาณจราจร เป็นจำนวนทั้งหมด 462 รูปภาพ โดยแต่ละสัญลักษณ์ป้ายจราจรจะมีจำนวนประมาณ 40 - 50 รูปภาพ

รูปที่ 1 สัญลักษณ์ป้ายจราจรทั้ง 10 ประเภท

รูปที่ 2 ตัวอย่างชุดข้อมูลรูปภาพสัญลักษณ์ป้ายจราจรของประเทศไทยทั้ง 10 ประเภท

แบบจำลอง (Model)

1. Convolutional Neural Network (CNN)

Convolutional Neural Network (CNN) หรือ โครงข่ายประสาทแบบคอนโวลูชัน เป็นโครงข่ายประสาทเทียมหนึ่งในกลุ่ม bio-inspired โดยที่ CNN มีความสามารถในการจำแนกข้อมูลประเภทรูปภาพได้ดีกว่า Neural network โดยการเรียนรู้ Feature ต่าง ๆ ของรูปภาพเหล่านั้น ในแต่ละ Layer ต่อยอดขึ้นไปเรื่อยๆ

CNN จะใช้ Convolution Layer มาประกอบกับ Layer ชนิดอื่น เช่น Pooling Layer แล้วนำกลุ่ม Layer ดังกล่าวมาซ้อนต่อๆ กัน โดยอาจเปลี่ยน Hyperparameter บางอย่าง เช่นขนาดของ Filter Layer (ซึ่งเป็นส่วนหนึ่งของ Convolution Layer) และจำนวน Channel ของ Layer วิธีการนำเอาส่วนต่างๆ มาประกอบกันนี้ เรียกว่าเป็นโครงสร้าง (Architecture) ของ CNN ซึ่งมีหลายแบบ เช่น LeNet, AlexNet, VGG, ResNet, Inception Network เป็นต้น

การคำนวณตามสถาปัตยกรรรมของ CNN มีขั้นตอนการคำนวณที่แบ่งออกได้ 3 ขั้นตอนคือ ขั้นตอนการคอนโวลูชัน (Convolution stage) ขั้นตอนการตรวจจับ (Detector stage) และขั้นตอนการพูลลิ่ง (Pooling stage)

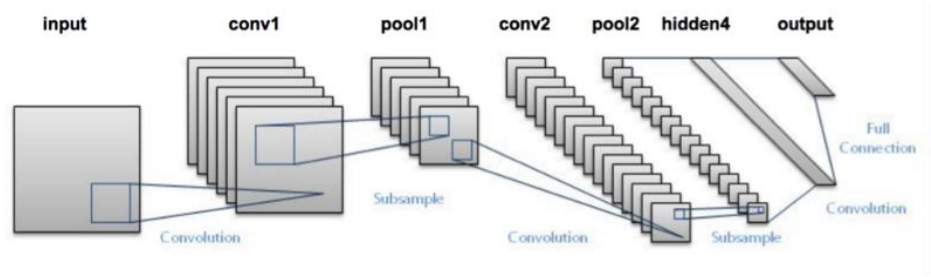
1.) ขั้นตอนการคอนโวลูชัน (Convolution Stage) ใช้หลักการเดียวกันกับการคำนวณคอนโวลูชันเชิงพื้นที่ (Spatial Convolution) ที่ใช้ในงานในด้านการประมวลผลภาพดิจิทัล (Digital Image Processing) จุดมุ่งหมายของการคำนวณคอนโวลูชันเชิงพื้นที่กับรูปภาพดิจิทัลก็คือ การสกัดลักษณะเด่นจากรูปภาพอินพุตแบบดิจิทัล โดยการคำนวณคอนโวลูชันทำให้เกิดการแปลงเชิงเส้น (Linear Transformation) ของรูปภาพอินพุตที่สอดคล้องกับข้อมูลเชิงพื้นที่จากตัวกรอง (Filter) โดยมีค่าถ่วงน้ำหนัก (Weight) ของแต่ละ layer จะเป็นตัวกำหนดรายละเอียดของคอนโวลูชันเคอร์นอล (Convolution Kernel) ดังนั้น Convolution Kernel สามารถทำการเทรน (Training) หรือทำการฝึกสอนได้และขึ้นอยู่กับอินพุตของเครือข่ายประสาทเทียบแบบ CNN

2.)ขั้นตอนการตรวจจับ (Detector Stage) ในขั้นตอนนี้จะทำหน้าที่รับข้อมูลที่ได้จากขั้นตอน Convolution Stage มาแปลงให้อยู่ในรูปแบบที่ไม่เป็นเชิงเส้น (Noninear) โดยใช้ฟังก์ชันการกระตุ้น (Activation function) เช่น Rectifed Linear Units (ReLU) โดยผลลัพธ์ที่ได้จากการทำคอนโวลูชั่นในแต่ละตำแหน่งจะผ่านการแปลงค่าด้วยฟังก์ชัน ReLU ที่เป็นการแปลงแบบไม่เป็นเชิงเส้น เพื่อความง่ายในการคำนวณและประสิทธิภาพของผลลัพธ์

3.) ขั้นตอนการพูลลิ่ง (Pooling Stage) การคำนวณ Pooling เป็นการประมวลผลที่ทำให้เกิดการลดขนาดข้อมูลหรือการสุ่มต่ำข้อมูล (Subsampling) โดยทำให้ข้อมูลที่ได้ทางด้านเอาต์พุตมีขนาดเล็กลงโดยที่รายละเอียดของข้อมูลที่ป้อนเข้ามายังคงครบถ้วนเหมือนเดิม การพูลลิ่งแบบค่าสูงสุด (Max Pooling) เป็นตัวกรองแบบหนึ่งที่ค้นหาค่าสูงสุด (Maximum) ในบริเวณที่ตัวกรองทาบอยู่แล้วนำมาเป็นผลลัพธ์ โดยจะเตรียมตัวกรองใน

ลักษณะเดียวกับขั้นตอนการทำ Feature extraction ของ CNN มาทาบบนข้อมูลแล้วเลือกค่าสูงสุดบนตัวกรองนั้นมาเป็นผลลัพธ์ใหม่จากนั้นทำการเลื่อนตัวกรองไป

ตาม Stride ที่กำหนดไว้ การ Pooling มีประโยชน์ในเรื่องของการเพิ่มความไวในการคำนวณและยังช่วยในการแก้ปัญหาการเกิด Overfitting ในขั้นตอนของการเรียนรู้วิธีการ Pooling ที่นิยมใช้ทั่วไปคือ วิธีการ MAX Pooling กับวิธีการ L2 โดยในสถาปัตยกรรมของ CNN โดยทั่วไปแล้ว การคำนวณ Pooling จะใช้ขนาดของหน้าต่างในการคำนวณเท่ากับ 2x2 และใช้ขนาดของ Stride เท่ากับ 2 โดยที่ไม่ต้องมีการเพิ่มพิกเซลภาพ (Paddine) ที่บริเวณขอบของภาพ ซึ่งเป็นวิธีการคำนวณที่แตกต่างไปจากการคำนวณ Convolution โดยทั่วไป



รูปที่ 3 โครงสร้างของแบบจำลอง CNN

(ที่มา : https://datawow.io/blogs/cnn-models)

2. Support Vector Machine (SVM)

โดยซัพพอร์ตเวคเตอร์แมชชีน เป็นวิธีที่ใช้ในการจำแนกข้อมูล (Classification) และการถดถอย (regression) โดยอาศัยหลักการของการหาสัมประสิทธิ์ของสมการและการจำแนกหมวดหมู่ที่ใช้หลักการสร้างสมการเส้นตรงเพื่อสร้างเส้นแบ่งแยกกลุ่มข้อมูลให้ได้เส้นเส้นแบ่งแยกกลุ่มข้อมูลได้ดีที่สุด (optimal separating hyperplane) โดยซัพพอร์ตเวคเตอร์แมชชีน สามารถแบ่งออกเป็น 2 ประเภทคือ Linear SVM ใช้สำหรับข้อมูลที่แยกได้เชิงเส้นคือ ชุดข้อมูลสามารถจำแนกออกเป็นสองคลาสโดยใช้เส้นตรงเส้นเดียว ข้อมูลดังกล่าวจะเรียกว่าข้อมูลที่แยกได้แบบเชิงเส้น และ Non-linear SVM ใช้สำหรับข้อมูลที่แยกจากกันแบบไม่เชิงเส้น คือ ชุดข้อมูลไม่สามารถจัดประเภทโดยใช้เส้นตรงได้ และโครงสร้างของซัพพอร์ตเวคเตอร์แมชชีน ประกอบด้วย 2 ส่วนหลักคือการเพิ่มระยะการจำแนกมากที่สุด และการลดข้อผิดพลาดให้ต่ำที่สุด

หลักการของแบบจำลองคือพยายามที่จะทำการลดความผิดพลาดจากการ ทำนาย (Minimize error) พร้อมกับเพิ่มระยะจำแนกให้มากที่สุด (Maximized Margin) โดยจะใช้ฟังก์ชันแม็ปข้อมูลจาก Input Space ไปยัง Feature Space และสร้างฟังก์ชั่นวัดความคล้ายที่เรียกว่าเคอร์เนลฟังก์ชั่น (Kernel Function) บน Feature Space ซึ่งการทำจะเหมาะสำหรับการใช้ข้อมูลที่มีลักษณะมิติของข้อมูลที่มีปริมาณมาก ตัวอย่างสมการ มีดังนี้คือ แบบเชิงเส้น แบบพหุนาม แบบเกาส์เซียน และแบบกำลังสอง โดยจะแสดงสมการตามลำดับ

แนวคิดพื้นฐานของทฤษฎีสามารถอธิบายได้โดยนำข้อมูลที่ต้องการจำแนกป้อนเข้าเพื่อให้ SVM คัดแยกกลุ่มข้อมูลออกมาโครงสร้างข้อมูลสำหรับสอนหรือป้อนเข้า และผลลัพธ์ด้านออกของ SVM ที่จะทำให้ระบบเกิดการรู้จำ ดังสมการที่ 1

$$(x_i y_i), \dots, (x_n, y_n) \text{ เมื่อ } x \epsilon R^m, y \epsilon \{+1, -1\} \quad (1)$$

เมื่อ $(x_i, y_i), \dots, (x_n, y_n)$ เป็นคุณลักษณะเด่นสำหรับใช้ในการสอน n คือ จำนวนข้อมูลตัวอย่าง m คือจำนวนมิติข้อมูลด้านเข้า และ y คือ ผลลัพธ์มีค่า +1 หรือ -1 ดังนั้นข้อมูลจะถูกจำแนกเป็นสองกลุ่มดังสมการที่ 2

$$(w \cdot x) + b > 0 \text{ ถ้า } y_i = +1$$

$$\text{และ} \quad (w \cdot x) + b < 0 \text{ ถ้า } y_i = -1 \quad (2)$$

เมื่อ w คือ ค่าน้ำหนัก และ b คือค่า bias โดยมีเส้นแบ่งหรือระนาบตัดสินใจที่คำนวณได้จากสมการที่ 3 เวกเตอร์ของข้อมูลที่ถูกป้อนเข้าสู่ระบบการสอน เพื่อให้ระบบเรียนรู้แทนด้วยสมการ และข้อมูลทั้งสองด้านแบ่งเป็นบวกและลบ สถานะของข้อมูลจึงแทนด้วย y ซึ่งมีสองค่า คือ y = 1 และ y = -1 นั้น แต่ยังตัดสินไม่ได้ว่าว่าเส้นแบ่งใดจึงจะดีที่สุด วิธีการ หาเส้นแบ่งที่ดีที่สุดคือการเพิ่งขอบให้กับเส้นแบ่งทั้ง 2 ข้าง ทำให้ได้เส้นขอบ (Margin) ใหม่ที่จะถือเป็นขอบของข้อมูลแต่ละฝั่งอีกด้วย เส้นขอบที่เป็นเส้นแบ่งนั้นจะเป็นเส้นที่สัมผัสกับค่าข้อมูลใน feature space ที่ใกล้ที่สุด เส้นขอบทั้ง 2 เส้นถูกแทนด้วยสมการ $w \cdot x^+ + b \geq y \geq 1$ ถ้าอยู่ด้าน y =1 และ $w \cdot x^- b \leq y >\leq -1$ ถ้า y = -1 หากเส้นขอบของเส้นแบ่งใด ๆ ที่มีความกว้างมากที่สุด แสดงว่าข้อมูลทั้ง 2 ชุดมีการแยกกันชัดเจน ดังนั้นเส้นแบ่งที่มีขอบกว้างที่สุดจึงเป็นเส้นแบ่งที่ดีที่สุด เราสามารถหาความกว้างของเส้นขอบ (maximization of margin) Y ได้จากสมการที่ 4 ค่าของ w และ b หาได้จาก สมการที่ 5 และ สมการที่ 6 ตามลำดับ

$$w \cdot x^{\mp} b \leq y \leq -1 \quad (3)$$

$$(w \cdot x) + b = 0 \quad (4)$$

$$w = \sum_{i=1}^{N} \alpha_i y_i x_i \quad (5)$$

เมื่อ α คือสัมประสิทธิ์คงที่ $\alpha_i \geq 0;\ i = 1,2,3, \dots, N$

$$b = -\frac{\max\limits_{y_i=-1}(w \cdot x_i) + \min\limits_{y_i=1}(w \cdot x_i)}{2} \quad (6)$$

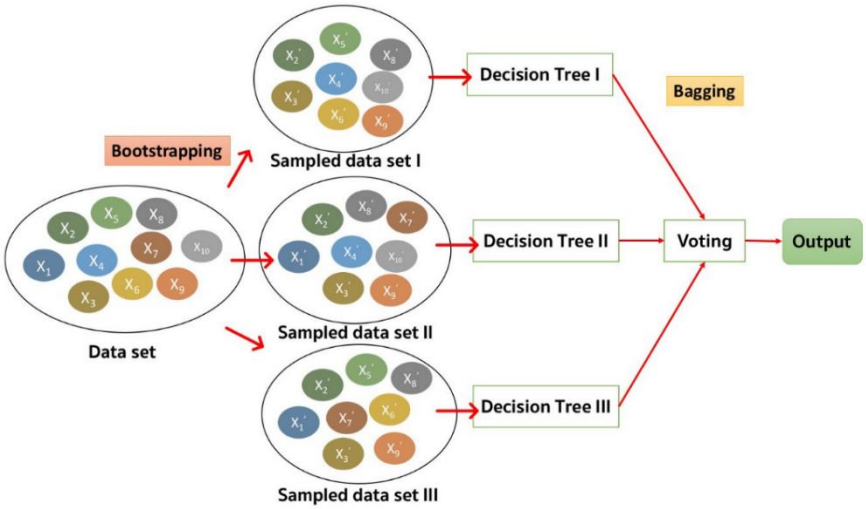
3. Random Forest (RF)

แนวคิดของ Random Forest นี้คือ การสร้างแบบจำลองด้วยวิธีการ Decision Tree ขึ้นมาหลายๆ แบบจำลอง โดยวิธีการสุ่มตัวแปร แล้วนำผลที่ได้แต่ละแบบจำลองมารวมกันพร้อมนับจำนวนผลที่มีจำนวนซ้ำกันมากที่สุด สกัดออกมาเป็นผลลัพธ์สุดท้าย วิธีการของ Decision Tree คือเทคนิคที่ให้ผลลัพธ์ในลักษณะเป็นโครงสร้างของต้นไม้ ภายในต้นไม้จะประกอบไปด้วยโหนด (node) ซึ่งแต่ละ โหนดจะมีเงื่อนไขของคุณลักษณะเป็นตัวทดสอบกิ่งของต้นไม้ (branch) แสดงถึงค่าที่เป็นไปได้ของคุณลักษณะที่ถูกเลือกทดสอบ และ ใบ (leaf) เป็นสิ่ง

ที่อยู่ล่างสุดของต้นไม้แสดงถึงกลุ่มของข้อมูล (class) ก็คือผลลัพธ์ที่ได้จากการพยากรณ์ ซึ่งข้อดีของวิธีการนี้คือให้ผลการพยากรณ์ที่แม่นยำและเกิดปัญหา overfitting น้อย โดยแบบจำลองนี้เป็นการทำนายแบบชุดของ Decision Tree หลายๆ ต้น (Ensemble of Decision Trees) โดยสร้างจากการสุ่มข้อมูลตัวอย่างแบบเลือกแล้วใส่กลับ (Random Sampling with Replacement) เพื่อนำมาสร้างเป็นแบบจำลองต้นไม้โดยแต่ละต้นมีลักษณะที่ไม่ซ้ำกัน โดยแต่ละแบบจำลองจะมีการทำนายผล ซึ่งผลจากการทำนายของต้นไม้แต่ละต้นจะทำการโหวตเลือกผลการทำนายที่ได้รับการโหวตมากที่สุดซึ่งวิธีการนี้เรียกว่า Bagging หรือ Boostraping

หลักการทำ Random Forest

1. sample ข้อมูล (Bootstrapping) จาก dataset ทั้งหมด ให้ได้ข้อมูลออกมา n ชุด ที่ไม่เหมือนกัน ตามจำนวน Decision Tree ใน Random Forest เช่น dataset ตั้งต้นมีอยู่ 10 feature (X1,X2,...,X10) แต่ละ Decision Tree จะได้ feature ไปไม่เหมือนกัน และจะได้ข้อมูลไม่ครบทุก row ด้วยจาก dataset ทั้งหมดด้วย (X1 -> X1',X2->X2',...)
2. สร้าง model Decision Tree สำหรับแต่ละชุดข้อมูล
3. ทำ aggregation ผลลัพธ์จากแต่ละ model (Bagging) เช่น voting ในกรณี classification หรือ หาค่า mean ในกรณี regression



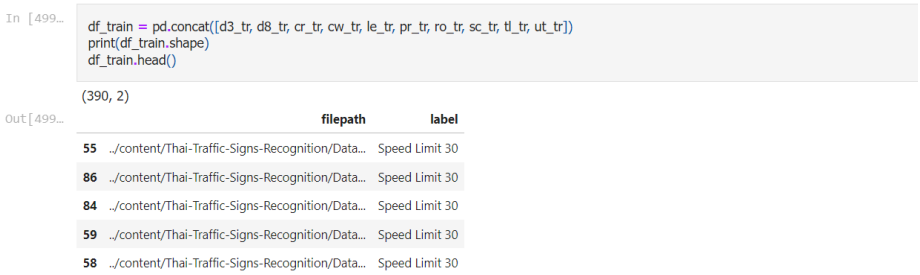
รูปที่ 4 หลักการทำ Random Forest

(ที่มา : https://shorturl.asia/Tl5uS)

การจัดเตรียมข้อมูล (Data Preparation)

การเตรียมข้อมูลของงานวิจัยนี้คือการสร้าง Pandas Dataframe ที่คอลัมน์แรกเป็นที่อยู่ของไฟล์รูปภาพทุกรูป (Filepaths) และ คอลัมน์ที่สองเป็นประเภทของป้ายจราจรนั้นๆ (Label) เพื่อสอดคล้องกับ library flow_from_dataframe ของ keras ในการสร้างรูปภาพเพื่อใช้เป็นข้อมูลนำเข้าของแบบจำลอง โดยเราจะใช้ library Path ในการดึงค่าที่อยู่ไฟล์รูปภาพทั้งหมด จากนั้นใช้สร้างฟังก์ชั่นในการ Map ที่อยู่ของไฟล์นั้นๆให้เข้ากับ

ประเภทของป้ายนั้นๆ ในตอนนี้ จะได้ Dataframe ที่มีที่อยู่ไฟล์และประเภทของป้ายจราจรเรียบร้อยแล้ว จากนั้นทำการแยก Dataframe ออกเป็นแต่ละประเภทของป้ายจราจร เพื่อทำการแยกข้อมูลที่จะนำไปใช้ Train, Validation และ Test ในแต่ละประเภทก่อน โดยมีสัดส่วนคือ Train 90%, Test 10% และแบ่ง Validation ออกจาก Train อีก 5% เมื่อแบ่ง Dataframe ออกเป็น 3 ส่วนครบแล้ว จึงค่อยนำ Dataframe ป้ายจราจรทุกประเภทมารวมกันเป็น Dataframe เดียว ซึ่งจะเก็บในตัวแปร df_train, df_val และ df_test ตามลำดับ



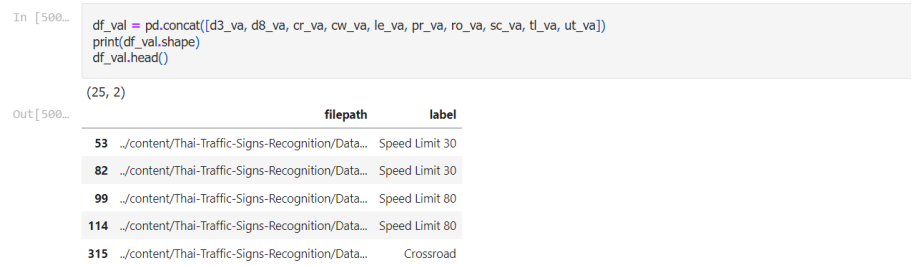
```
In [499…
df_train = pd.concat([d3_tr, d8_tr, cr_tr, cw_tr, le_tr, pr_tr, ro_tr, sc_tr, tl_tr, ut_tr])
print(df_train.shape)
df_train.head()
```

(390, 2)

Out[499…

	filepath	label
55	../content/Thai-Traffic-Signs-Recognition/Data...	Speed Limit 30
86	../content/Thai-Traffic-Signs-Recognition/Data...	Speed Limit 30
84	../content/Thai-Traffic-Signs-Recognition/Data...	Speed Limit 30
59	../content/Thai-Traffic-Signs-Recognition/Data...	Speed Limit 30
58	../content/Thai-Traffic-Signs-Recognition/Data...	Speed Limit 30

รูปที่ 5 คำสั่งการสร้าง train set ในรูปแบบ dataframe



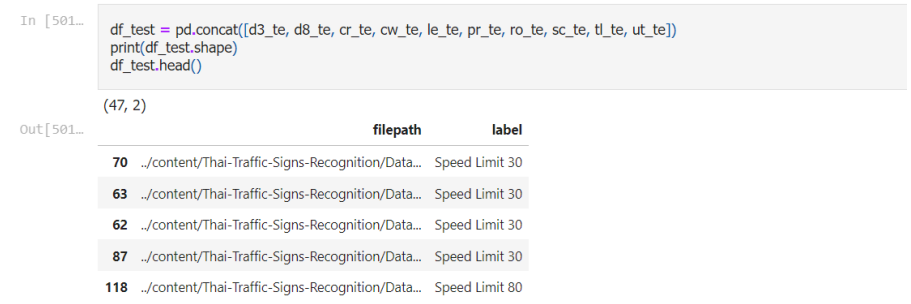
```
In [500…
df_val = pd.concat([d3_va, d8_va, cr_va, cw_va, le_va, pr_va, ro_va, sc_va, tl_va, ut_va])
print(df_val.shape)
df_val.head()
```

(25, 2)

Out[500…

	filepath	label
53	../content/Thai-Traffic-Signs-Recognition/Data...	Speed Limit 30
82	../content/Thai-Traffic-Signs-Recognition/Data...	Speed Limit 30
99	../content/Thai-Traffic-Signs-Recognition/Data...	Speed Limit 80
114	../content/Thai-Traffic-Signs-Recognition/Data...	Speed Limit 80
315	../content/Thai-Traffic-Signs-Recognition/Data...	Crossroad

รูปที่ 6 คำสั่งการสร้าง validation set ในรูปแบบ dataframe



```
In [501…
df_test = pd.concat([d3_te, d8_te, cr_te, cw_te, le_te, pr_te, ro_te, sc_te, tl_te, ut_te])
print(df_test.shape)
df_test.head()
```

(47, 2)

Out[501…

	filepath	label
70	../content/Thai-Traffic-Signs-Recognition/Data...	Speed Limit 30
63	../content/Thai-Traffic-Signs-Recognition/Data...	Speed Limit 30
62	../content/Thai-Traffic-Signs-Recognition/Data...	Speed Limit 30
87	../content/Thai-Traffic-Signs-Recognition/Data...	Speed Limit 30
118	../content/Thai-Traffic-Signs-Recognition/Data...	Speed Limit 80

รูปที่ 7 คำสั่งการสร้าง test set ในรูปแบบ dataframe

การนำแบบจำลองไปใช้งานกับชุดข้อมูลรูปภาพ

1. **Convolutional Neural Network (CNN)**

ในการสร้างแบบจำลองนี้จะใช้ library keras ซึ่งต้องทำการ import keras เข้ามาก่อน และก่อนที่จะนำข้อมูลเข้าแบบจำลองจะต้องทำการ Preprocessing Data โดยการทำ Data Augmentation เพื่อเพิ่มจำนวนของชุดข้อมูลรูปภาพที่นำมาใช้ และ rescale ด้วย ImageDataGenerator ซึ่งประกอบด้วยการปรับค่า rotation_range, width_shift_range, height_shift_range, brightness_range, zoom_range และ fill_mode = 'reflect'

```
In [ ]:
#create preset of image generator
train_generator = keras.preprocessing.image.ImageDataGenerator(
    rescale = 1./255,
    rotation_range = 30,
    width_shift_range = 0.2,
    height_shift_range = 0.2,
    brightness_range = [0.4,1.5],
    zoom_range = 0.3,
    fill_mode = 'reflect'
)

validate_generator = keras.preprocessing.image.ImageDataGenerator(
    rescale = 1./255
)


test_generator = keras.preprocessing.image.ImageDataGenerator(
    rescale = 1./255
)
```

Parameter setup

```
In [ ]:
X_COL = 'filepath'
Y_COL = 'label'
MODE = 'rgb'
BATCH_SIZE = 16
```

รูปที่ 8 คำสั่งการทำ Data Augmentation ด้วย ImageDataGenerator

ขั้นตอนนี้จะทำการแบ่งข้อมูลออกเป็น 90% โดยประกอบด้วย train set จำนวน 390 รูปภาพและ validation set จำนวน 25 รูปภาพ (ซึ่งคิดเป็น 5% ของ train set) และข้อมูลใน testing set จะเหลือ 10% ซึ่งเป็นจำนวน 47 รูปภาพ และเปลี่ยนขนาดของรูปภาพเป็นขนาด 224 x 224 pixels

```
In [ ]:
#load data by flow_from_dataframe
train_ds = train_generator.flow_from_dataframe(
    dataframe = df_train,
    x_col = X_COL,
    y_col = Y_COL,
    target_size = (224, 224),
    color_mode = MODE,
    class_mode = 'categorical',
    batch_size = BATCH_SIZE,
    shuffle = False
)

val_ds = validate_generator.flow_from_dataframe(
    dataframe = df_val,
    x_col = X_COL,
    y_col = Y_COL,
    target_size = (224, 224),
    color_mode = MODE,
    class_mode = 'categorical',
    batch_size = BATCH_SIZE,
    shuffle = False
)

test_ds = test_generator.flow_from_dataframe(
    dataframe = df_test,
    x_col = X_COL,
    y_col = Y_COL,
    target_size = (224, 224),
    color_mode = MODE,
    class_mode = 'categorical',
    batch_size = BATCH_SIZE,
    shuffle = False
)
```

```
Found 390 validated image filenames belonging to 10 classes.
Found 25 validated image filenames belonging to 10 classes.
Found 47 validated image filenames belonging to 10 classes.
```

รูปที่ 9 คำสั่งการแบ่งข้อมูลเป็น train set, validation set และ test set

ทำการสร้างแบบจำลอง Convolutional Neural Network (CNN) โดยการใช้ Transfer Learning ที่ชื่อว่า “InceptionResNetV2”

```
In [502…
from tensorflow.keras.applications import InceptionResNetV2
from keras.models import Model
from keras.layers import Dense
from keras.layers import Flatten

In [ ]:
def def_model():
  model = InceptionResNetV2(
      include_top=False,
      input_shape=(224, 224, 3)
  )
  for layer in model.layers:
          layer.trainable = False
  flat1 = Flatten()(model.layers[-1].output)
  full_conn = Dense(128, activation='relu', kernel_initializer='he_uniform')(flat1)
  output = Dense(10, activation='sigmoid')(full_conn)
      # define new model
  model = Model(inputs=model.inputs, outputs=output)
      # compile model
  model.compile(optimizer='adam', loss='categorical_crossentropy', metrics=['acc'])
  model.save('/content/model/InceptionResNetV2.h5')
  model.summary()
  return model

In [ ]:
irnv2 = def_model()
```

รูปที่ 10 คำสั่งการสร้างแบบจำลอง CNN โดยใช้ InceptionResNetV2

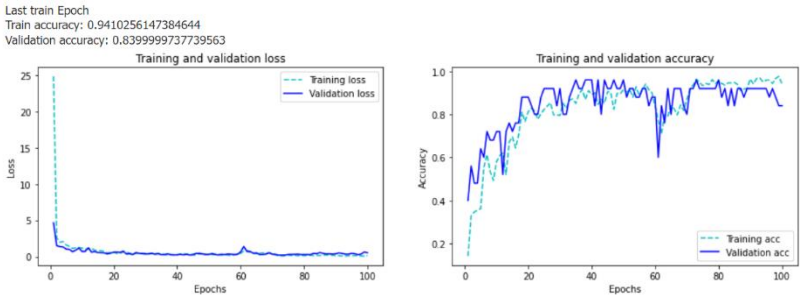
กำหนด epochs = 100 และ batch size = 16 ในการรันแบบจำลอง และใช้ ModelCheckpoint สำหรับทำการ callbacks เพื่อบันทึกค่า validation accuracy ที่มีค่าสูงที่สุด

```
In [ ]:
save_path = '/content/weight_improvement/irnv2_x224_{epoch:02d}_{val_acc:.2f}.hdf5'
checkpoint = ModelCheckpoint(save_path, monitor = 'val_acc', verbose = 1, save_best_only = True, mode = 'max')
callbacks_list = [checkpoint]

history3 = irnv2.fit(
    train_ds,
    batch_size=16,
    epochs=100,
    validation_data=(val_ds),
    callbacks = callbacks_list
)
irnv2.save('/content/model/InceptionResNetV2_x224_itr100.h5')
```

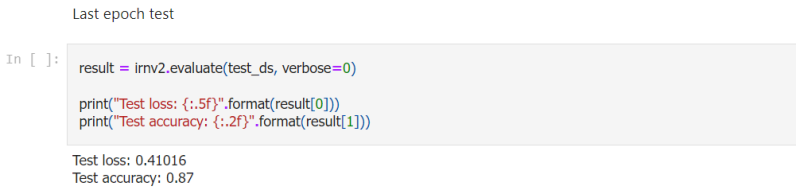
รูปที่ 11 คำสั่งการรัน epoch ของแบบจำลอง CNN

โดยมีค่า train accuracy = 0.9410256147384644 และ validation accuracy = 0.8399999737739563



รูปที่ 12 ผลลัพธ์ค่า train accuracy และ validation accuracy ที่ได้จากแบบจำลอง CNN

การประเมินผลแบบจำลองจะทำการแสดงค่า test accuracy = 0.87 โดยเป็นค่าที่ดีที่สุดจากการรัน last epoch test



```
Last epoch test

In [ ]:
result = irnv2.evaluate(test_ds, verbose=0)

print("Test loss: {:.5f}".format(result[0]))
print("Test accuracy: {:.2f}".format(result[1]))

Test loss: 0.41016
Test accuracy: 0.87
```

รูปที่ 13 คำสั่งและผลลัพธ์การประเมินผลแบบจำลอง CNN

2. Support Vector Machine (SVM)

โดยก่อนนำข้อมูลเข้าแบบจำลองจะทำการ Preprocessing Data โดยการทำ Data Augmentation ซึ่งจะทำเหมือนกับการทำ Data Augmentation จากขั้นตอน Preprocessing ของ Convolutional Neural Network (CNN) แล้วทำการแบ่งข้อมูลเป็น training set ออกจากข้อมูล 90% ซึ่งเป็นจำนวน 417 รูปภาพ และ testing set จะเหลือ 10% ซึ่งเป็นจำนวน 47 รูปภาพ และเปลี่ยนขนาดของรูปภาพเป็นขนาด 224 x 224 pixels

```
In [506…
X_COL = 'filepath'
Y_COL = 'label'
MODE = 'rgb'
BATCH_SIZE = 1

#load data by flow_from_dataframe
train_ds = train_generator.flow_from_dataframe(
    dataframe = df_train,
    x_col = X_COL,
    y_col = Y_COL,
    target_size = (224, 224),
    color_mode = MODE,
    class_mode = 'categorical',
    batch_size = BATCH_SIZE,
    shuffle = False
)

test_ds = test_generator.flow_from_dataframe(
    dataframe = df_test,
    x_col = X_COL,
    y_col = Y_COL,
    target_size = (224, 224),
    color_mode = MODE,
    class_mode = 'categorical',
    batch_size = BATCH_SIZE,
    shuffle = False
)

Found 415 validated image filenames belonging to 10 classes.
Found 47 validated image filenames belonging to 10 classes.
```

รูปที่ 14 คำสั่งการทำ Data Augmentation และการแบ่งข้อมูลเป็น train set และ test set

แล้วจึงนำรูปภาพมาทำการ convert images to array คือการเปลี่ยนรูปเป็นข้อมูลที่เป็นตัวเลข ลงในตารางอาเรย์ (Arrays) โดยจะได้ผลลัพธ์เป็น 3 มิติ ซึ่งคือ pixel แกน x , pixel แกน y , สี ซึ่งแบบจำลองสามารถนำข้อมูลเข้าได้เพียงสองมิติ เพราะฉะนั้นจึงต้องลดมิติของข้อมูลลง แล้วทำการ label ข้อมูลแต่ละประเภท ลงใน array โดยกำหนด 10 ประเภท เป็นตัวเลข

```
In [507…
train_list = []
batch_index_train = 0

while batch_index_train <= train_ds.batch_index:
    data = train_ds.next()
    train_list.append(data[0])
    batch_index_train = batch_index_train + 1

# now, data_array is the numeric data of whole images
train_array = np.asarray(train_list)

train_array.shape

Out[507…
(415, 1, 224, 224, 3)

In [508…
test_list = []
batch_index_test = 0

while batch_index_test <= test_ds.batch_index:
    data = test_ds.next()
    test_list.append(data[0])
    batch_index_test = batch_index_test + 1

# now, data_array is the numeric data of whole images
test_array = np.asarray(test_list)

print(test_array.shape)

(47, 1, 224, 224, 3)
```

รูปที่ 15 คำสั่งแปลงข้อมูลรูปภาพเป็นข้อมูลตารางอาเรย์

-->

```
Reduce dimension of batch size

In [509…
train_array = np.squeeze(train_array, axis=1)
train_array.shape

Out[509…
(415, 224, 224, 3)

In [510…
test_array = np.squeeze(test_array, axis=1)
test_array.shape

Out[510…
(47, 224, 224, 3)
```

รูปที่ 16 คำสั่งการลดมิติจาก 3 มิติเป็น 2 มิติ

การทำ feature extraction ก่อนเข้าแบบจำลอง เนื่องจาก Support Vector Machine และ Random Forest ไม่มี feature extraction layers เหมือนกับใน Convolutional Neural Network ทำให้เราจำเป็นที่จะต้องสร้าง feature extraction ที่ใช้ดึงค่าที่สำคัญเพื่อนำไปใช้เป็นข้อมูลนำเข้าสำหรับแบบจำลอง โดยมีดังนี้

1. Pixel Values คือ ค่าในแต่ละ pixel ของแต่ละรูปเอง
2. Gabor Filter คือ วิธีการแปลงข้อมูลสำหรับการทำ texture extraction
3. Sobel Filter คือ วิธีการแปลงข้อมูลสำหรับการทำ edge detection

ในการสร้างแบบจำลองข้อมูลของ Support Vector Machine จะใช้ library sklearn โดยจะใช้คำสั่ง svm.SVC (decision_function_shape='ovo') ซึ่งการใช้ decision_function_shape = ovo คือการแบ่งข้อมูลออกมากกว่า 2 กลุ่มเนื่องจากแบบจำลอง svm โดยพื้นฐานแล้วมีไว้แบ่งกลุ่มข้อมูลแค่ 2 กลุ่มออกจากกัน ซึ่งฟังก์ชั่น ovo จะเป็นฟังก์ชันตัดสินใจของกลุ่มหนึ่งเทียบกับอีกกลุ่มหนึ่ง เป็นการเทียบกันเป็นคู่ๆ

```
Modeling

In [526…
from sklearn import svm
SVM_model = svm.SVC(decision_function_shape='ovo')  #For multiclass classification
SVM_model.fit(x_train, y_train)

Out[526…
SVC(decision_function_shape='ovo')
```

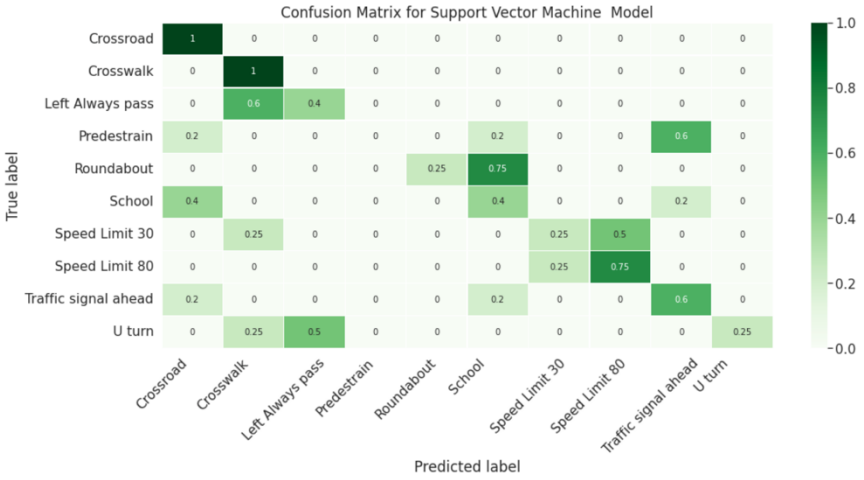
รูปที่ 17 คำสั่งการสร้างแบบจำลอง SVM

การประเมินผลแบบจำลองจะทำการแสดงค่า accuracy = 0.5106382978723404 และแสดงผลตาราง confusion matrix ซึ่งจะบอกค่าการทำนายแต่ละคลาสที่ทำนายถูก ซึ่งคลาส Crossroad หรือป้ายเตือนสี่แยก และ Crosswalk หรือป้ายทางม้าลาย ดีที่สุดเนื่องจากได้ค่า True Possitive เท่ากับ 1 และคลาส Predestrain หรือป้ายเตือยคนข้ามถนน ทำนายได้แย่ที่สุดเนื่องจากได้ค่า True Possitive เท่ากับ 0

```
In [528…  # View the classification report for test data and predictions
          print(classification_report(df_test['label'], test_pred_svm))
```

	precision	recall	f1-score	support
Crossroad	0.56	1.00	0.71	5
Crosswalk	0.55	1.00	0.71	6
Left Always pass	0.50	0.40	0.44	5
Predestrain	0.00	0.00	0.00	5
Roundabout	1.00	0.25	0.40	4
School	0.29	0.40	0.33	5
Speed Limit 30	0.50	0.25	0.33	4
Speed Limit 80	0.60	0.75	0.67	4
Traffic signal ahead	0.43	0.60	0.50	5
U turn	1.00	0.25	0.40	4
accuracy			0.51	47
macro avg	0.54	0.49	0.45	47
weighted avg	0.52	0.51	0.46	47

รูปที่ 18 ผลการประเมินผล classification report ของ SVM



รูปที่ 19 ผลการประเมินผล confusion matrix ของ SVM

3. Random Forest (RF)

โดยก่อนนำข้อมูลเข้าแบบจำลองจะทำการ preprocessing data โดยจะใช้ข้อมูลที่ preprocessing และ feature extraction จาก Support Vector Machine (SVM) มาใช้ในขั้นตอนการทำแบบจำลองได้เลย

ในการสร้างแบบจำลองข้อมูลของ Random Forest จะใช้ libary sklearn โดยจะใช้คำสั่ง RandomForestClassifier(n_estimators = 50, random_state = 42) โดย n_estimators คือจำนวนของ decision trees ใน random forest ส่วน random_state คือ random seed ที่ใช้ในการสร้างซับเซ็ทสุ่มของข้อมูลและ feature ซึ่งผลของแบบจำลองจะได้ค่า accuracy = 0.574468085106383

Modeling

```
[ ] RF_model = RandomForestClassifier(n_estimators = 50, random_state = 42)
    RF_model.fit(x_train, y_train) #For sklearn no one hot encoding

    RandomForestClassifier(n_estimators=50, random_state=42)

[ ] #Predict on test
    test_pred_rf = RF_model.predict(x_test)

    #Inverse le transform to get original label back.
    test_pred_rf = le.inverse_transform(test_pred_rf)

    #Print overall accuracy
    print ("Accuracy = ", metrics.accuracy_score(df_test['label'], test_pred_rf))

    Accuracy =  0.574468085106383
```

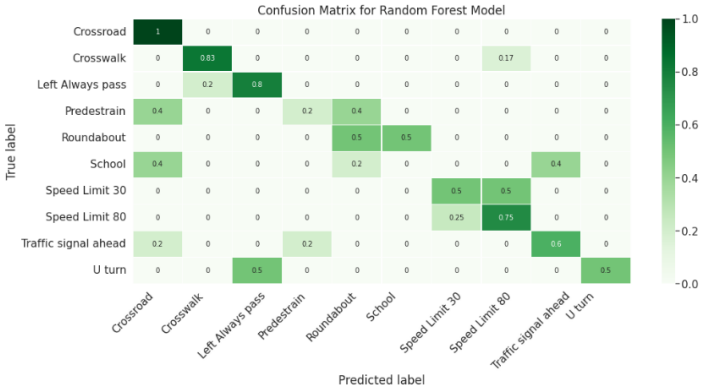
รูปที่ 20 คำสั่งการสร้างแบบจำลองแบบ RF

ส่วนขั้นตอนการประเมินผลจะแสดงค่าออกมาด้วยตาราง Confusion Matrix ซึ่งคลาส Crossroad หรือป้ายเตือนโทตัดเอก มีผลการทำนายออกมาดีที่สุดจากค่า True Possitive เท่ากับ 1 และคลาส School หรือป้ายเตือนเขตโรงเรียน ทำนายออกมาได้แย่ที่สุดจากค่า True Possitive เท่ากับ 0

```
[ ] # View the classification report for test data and predictions
    print(classification_report(df_test['label'], test_pred_rf))
```

	precision	recall	f1-score	support
Crossroad	0.50	1.00	0.67	5
Crosswalk	0.83	0.83	0.83	6
Left Always pass	0.67	0.80	0.73	5
Predestrain	0.50	0.20	0.29	5
Roundabout	0.40	0.50	0.44	4
School	0.00	0.00	0.00	5
Speed Limit 30	0.67	0.50	0.57	4
Speed Limit 80	0.50	0.75	0.60	4
Traffic signal ahead	0.60	0.60	0.60	5
U turn	1.00	0.50	0.67	4
accuracy			0.57	47
macro avg	0.57	0.57	0.54	47
weighted avg	0.57	0.57	0.54	47

รูปที่ 21 ผลการประเมินผล classification report ของ RF



รูปที่ 22 ผลการประเมินผล confusion matrix ของ RF

วิธีการประเมินแบบจำลอง

วิธีการประเมินแบบจำลอง Support Vector Machine (SVM) , Convolutional Neural Network (CNN) และ Random Forest (RF) โดยวิธีที่ใช้ในการประเมินคือ ค่าความถูกต้อง (Accuracy) หรือ ค่าที่แบบจำลองทายถูกทั้งหมด คำนวนได้จากสูตร

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN}$$

โดย TP (True Positive) คือ ข้อมูลที่ทำนายตรงกับข้อมูลจริงในคลาสที่กำลังพิจารณา

TN (True Negative) คือ ข้อมูลที่ทำนายตรงกับข้อมูลจริงในคลาสที่ไม่ได้พิจารณา

FP (False Positive) คือ ข้อมูลที่ทำนายผิดเป็นคลาสที่กำลังพิจารณา

FN (False Negative) คือ ข้อมูลที่ทำนายผิดเป็นคลาสที่ไม่ได้พิจารณา

ซึ่งค่า Accuracy มากแสดงว่าแบบจำลอง มีการเรียนรู้ที่ดี

สรุปผลการทดสอบแบบจำลอง

จากการทดสอบแบบจำลองทั้ง 3 ด้วยวิธีการประเมินแบบจำลองที่กล่าวในบทก่อนหน้า เพื่อใช้ในการแยกประเภทป้ายจราจรของประเทศไทยทั้ง 10 ประเภท มีผลลัพท์ดังนี้

Convolutional Neural Network (CNN)

Model	Training accuracy	Validation accuracy	Testing accuracy
Convolutional Neural Network (CNN)	0.941	0.840	0.870

ตารางที่ 1 ผลการประเมิน Training Accurarcy , Validation accuracy , Testing accuracy ของ CNN

Support Vector Machine and Random Forest

Model	Accuracy	Precision	Recall
Support Vector Machine (SVM)	0.511	0.520	0.510
Random Forest (RF)	0.574	0.570	0.570

ตารางที่ 2 ผลการประเมิน Accurarcy , Precision, Recall ของ SVM และ RF

อภิปรายผล

แบบจำลองที่ดีที่สุด โดยอ้างอิงจากผลลัพท์ของการทดสอบแบบจำลองเพื่อใช้ในการแยกประเภทป้ายจราจรของประเทศไทย คือ Convolutional Neural Network (CNN) โดยมีค่า Accuracy ที่ 0.870 เพราะแบบจำลองนี้เหมาะสำหรับข้อมูลประเภทรูปภาพ จึงมีการใช้งานแบบจำลองนี้ค่อนข้างมากในการจำแนกประเภท

ของข้อมูลรูปภาพอื่นๆ ดังนั้น งานวิจัยนี้จึงแนะนำให้ใช้ Convolutional Neural Network (CNN) ในการแยกประเภทป้ายจราจรของประเทศไทย

จากการทดสอบในหัวข้อที่งานวิจัยนี้เลือกมานั้น สามารถระบุข้อดีและข้อเสียในแต่ละแบบจำลองได้ดังนี้

Model	Pro	Cons
Convolutional Neural Network (CNN)	1. ได้ค่า Accuracy ที่สูง 2. ทำนายได้รวดเร็วและแม่นยำ 3. ง่ายต่อการนำไปใช้งานต่อ	1. ใช้ Computational resource เยอะมาก ในแง่ของ CPU และ GPU 2. หากว่า Computational resource ไม่มากพอ จะส่งผลโดยตรงกับเวลาที่ใช้ในการสร้างแบบจำลอง
Support Vector Machine (SVM)	1. พัฒนาได้ง่าย 2. ทำความเข้าใจการคำนวนต่างๆที่เกิดขึ้นในแบบจำลองได้ 3. เหมาะสำหรับการแยกรูปภาพ 2 ประเภท	1. ได้ค่า Accuracy ที่ต่ำ เมื่อแยกรูปภาพออกเป็นหลายประเภท 2. ใช้ Computational Resource เยอะมากในแง่ของ RAM หากมีไม่เพียงพอ จะไม่สามารถสร้างแบบจำลองได้
Random Forest (RF)	1. พัฒนาได้ง่าย ทำความเข้าใจได้ง่าย 2. ใช้ Computational resource ที่ไม่มาก	1. ได้ค่า Accuracy ที่ต่ำ เมื่อแยกรูปภาพออกเป็นหลายประเภท

ตารางที่ 3 ตารางการเปรียบเทียบข้อดีข้อเสียของแต่ละแบบจำลอง

ทั้งนี้ งานวิจัยนี้ยังไม่ครอบคลุมป้ายจราจรทั้งหมด โดยงานวิจัยนี้ให้ความสำคัญในป้ายจราจรที่มีส่วนช่วยในการขับขี่อัตโนมัติเป็นหลัก รวมถึงประเทศไทยยังไม่มีฐานข้อมูลหรือชุดข้อมูลรูปภาพเพื่อใช้ในการสร้างแบบจำลอง จึงยังขาดป้ายจราจรอื่นๆ ที่พบเห็นได้บ่อย รวมถึงสีของป้ายที่มีความหมายที่แตกต่างกัน ในอนาคตการจำแนกประเภทของป้ายจราจรที่มีจำนวนประเภทมากกว่านี้อาจจะต้องทำการจำแนกสีของป้ายจราจรก่อนในขั้นแรก จากนั้นค่อยจำแนกสัญลักษณ์ที่อยู่บนป้ายจราจรเพื่อตีความหมายของป้ายจราจรนั้นต่อไป

การทดสอบแบบจำลองเพิ่มเติม

จากผลการพัฒนาแบบจำลองพบว่า Support Vector Machine (SVM) และ Random Forest (RF) มีค่า Accuracy ที่ต่ำกว่า Convolutional Neural Network (CNN) เป็นอย่างมาก ซึ่งสาเหตุคาดว่ามาจากจำนวนของประเภท ที่มากเกินไปสำหรับแบบจำลอง งานวิจัยนี้จึงทำการทดสอบเพิ่มด้วยการลดจำนวนประเภท โดยแบ่งออกเป็น 2 การทดลองประกอบด้วย แบ่งออกเป็น 2 กลุ่ม (5 ประเภท) และ แบ่งออกเป็น 5 กลุ่ม (2 ประเภท) โดยมีผลลัพท์ดังนี้

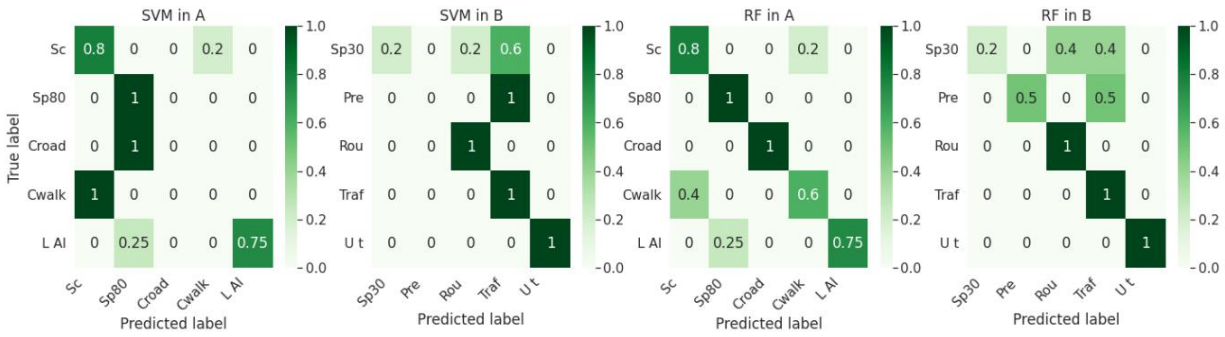
2 กลุ่ม (5 ประเภท)

กลุ่ม A ประกอบด้วย ป้ายเขตโรงเรียน, ป้ายจำกัดความเร็ว 80, ป้ายเตือนทางโทตัดเอก, ป้ายทางม้าลาย และ ป้ายซ้ายผ่านตลอด

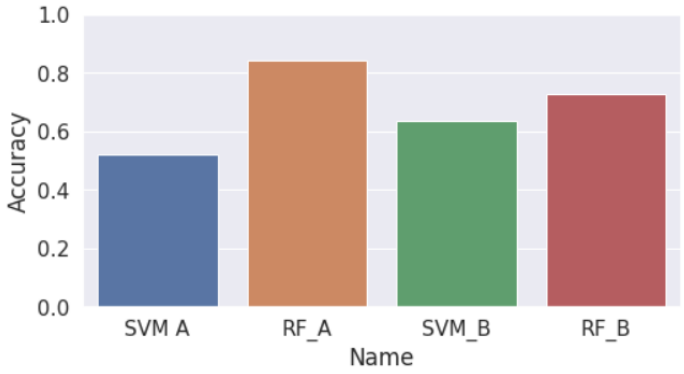
กลุ่ม B ประกอบด้วย ป้ายจำกัดความเร็ว 30, ป้ายคนข้ามถนน, ป้ายเตือนวงเวียนข้างหน้า, ป้ายสัญญาณจราจร และ ป้ายจุดกลับรถ

Group	SVM	RF	Mean of Accuracy
A	0.520	0.840	0.680
B	0.636	0.727	0.6815
Summary			0.681

ตารางที่ 4 ผลการประเมิน Accurarcy ของ SVM และ RF แบบ 2 กลุ่ม



รูปที่ 23 ผลการประเมินผล confusion matrix ของ SVM และ RF แบบ 2 กลุ่ม



รูปที่ 24 ผลการประเมินผล Accurarcy ของ SVM และ RF แบบ 2 กลุ่ม

5 กลุ่ม (2 ประเภท)

กลุ่ม A ประกอบด้วย ป้ายจำกัดความเร็ว 30 และ ป้ายจำกัดความเร็ว 80

กลุ่ม B ประกอบด้วย ป้ายคนข้ามถนน และ ป้ายเขตโรงเรียน

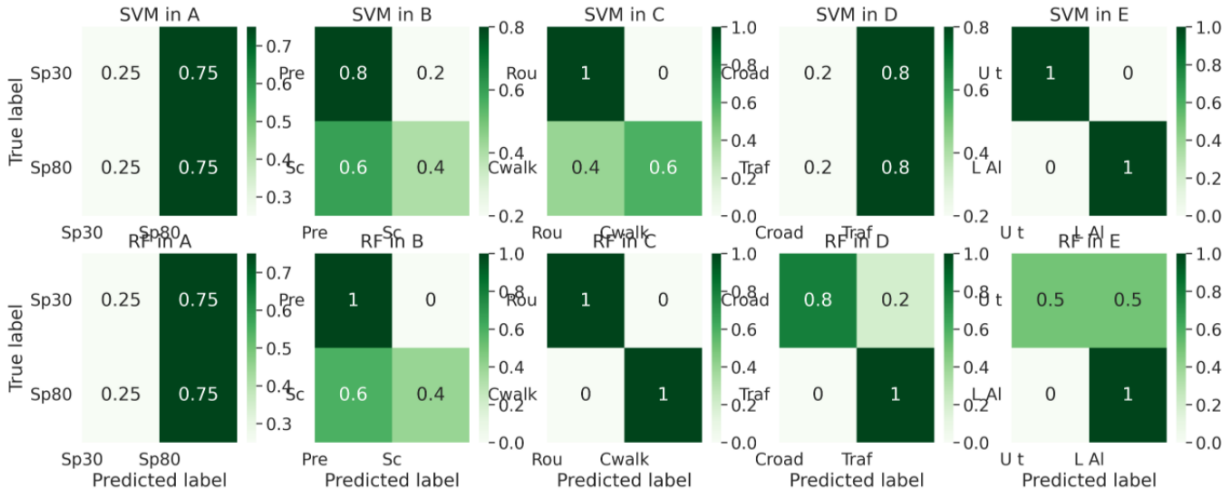
กลุ่ม C ประกอบด้วย ป้ายเตือนวงเวียนข้างหน้า และ ป้ายทางม้าลาย

กลุ่ม D ประกอบด้วย ป้ายเตือนทางโทตัดเอก และ ป้ายสัญญาณจราจร

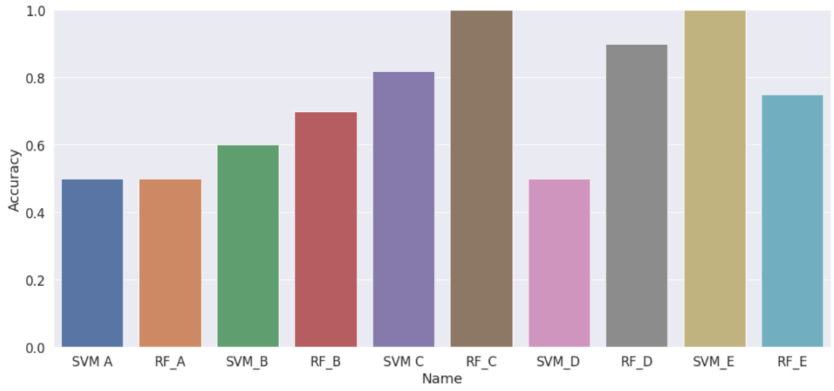
กลุ่ม E ประกอบด้วย ป้ายจุดกลับรถ และ ป้ายซ้ายผ่านตลอด

Group	SVM	RF	Mean of Accuracy
A	0.500	0.500	0.500
B	0.600	0.700	0.650
C	0.818	1.000	0.909
D	0.500	0.900	0.700
E	1.000	0.750	0.875
Summary			0.727

ตารางที่ 5 ผลการประเมิน Accurarcy ของ SVM และ RF แบบ 5 กลุ่ม



รูปที่ 25 ผลการประเมินผล confusion matrix ของ SVM และ RF แบบ 5 กลุ่ม



รูปที่ 26 ผลการประเมินผล Accurarcy ของ SVM และ RF แบบ 5 กลุ่ม

จากผลการทดสอบแบบจำลองสรุปได้ว่าจำนวนประเภท ที่ลดลงมีผลต่อค่า Accuracy ที่เพิ่มขึ้น สำหรับ 2 กลุ่ม (5 ประเภท) มีค่า Accuracy ที่ 0.681 เพิ่มขึ้นเป็น 0.727 สำหรับ 5 กลุ่ม (2 ประเภท) คิดเป็น 0.046 ที่เพิ่มขึ้น แต่หากดูย่อยในแต่ละแบบจำลองจะพบว่า Support Vector Machine มักจะมีค่า Accuracy ที่ต่ำกว่า Random Forest นั่นเพราะว่า Support Vector Machine อาจไม่เหมาะสำหรับการแยกรูปภาพหลายๆประเภท ส่วน Random Forest มีความสามาถในการแยกได้ระดับหนึ่งเท่านั้น ทั้งนี้ผลการทดสอบแบบจำลองอาจทำให้ดีขึ้นได้ด้วยการเพิ่มจำนวนรูปในแต่ละชุดข้อมูลรูปภาพและการทำ Feature Extraction เพิ่มเติม

บรรณานุกรม

ทำความเข้าใจ accuracy,precision,recall,f1-score. (2020). [ออนไลน์]. ได้จาก: http://www.ninenox.com/2020/09/24/ทำความเข้าใจ-accuracyprecisionrecallf1-score/ [สืบค้นเมื่อ วันที่ 13 เดือน พฤษภาคม ปี 2022].

การวิเคราะห์ประสิทธิภาพ Machine Learning Model ด้วย Learning Curve. (2020). [ออนไลน์]. ได้จาก: https://blog.pjjop.org/diagnose-machine-learning-model-performance-with-learning-curves/ [สืบค้นเมื่อ วันที่ 13 เดือน พฤษภาคม ปี 2022].

Evaluate Model นั้นสำคัญอย่างไร ? : Machine Learning 101. (2018). [ออนไลน์]. ได้จาก: https://medium.com/mmp-li/evaluate-model-precision-recall-f1-score-machine-learning-101-89dbbada0c96 [สืบค้นเมื่อ วันที่ 13 เดือน พฤษภาคม ปี 2022].

tutorial74_what is gabor filter.py. (2020). [ออนไลน์]. ได้จาก: https://github.com/bnsreenu/python_for_image_processing_APEER/blob/master/tutorial74_what%20is%20gabor%20filter.py [สืบค้นเมื่อ วันที่ 13 เดือน พฤษภาคม ปี 2022].

tutorial83_feature_extraction_RF_classification_V2.0.py. (2020). [ออนไลน์]. ได้จาก: https://github.com/bnsreenu/python_for_image_processing_APEER/blob/master/tutorial83_feature_extraction_RF_classification_V2.0.py [สืบค้นเมื่อ วันที่ 13 เดือน พฤษภาคม ปี 2022].

Through The Eyes of Gabor Filter. (2018). [ออนไลน์]. ได้จาก: https://medium.com/@anuj_shah/through-the-eyes-of-gabor-filter-17d1fdb3ac97 [สืบค้นเมื่อ วันที่ 13 เดือน พฤษภาคม ปี 2022].

Evaluating a Random Forest model. (2020). [ออนไลน์]. ได้จาก: https://medium.com/analytics-vidhya/evaluating-a-random-forest-model-9d165595ad56 [สืบค้นเมื่อ วันที่ 13 เดือน พฤษภาคม ปี 2022].

Understanding Edge Detection (Sobel Operator). (2018). [ออนไลน์]. ได้จาก: https://medium.datadriveninvestor.com/understanding-edge-detection-sobel-operator-2aada303b900 [สืบค้นเมื่อ วันที่ 13 เดือน พฤษภาคม ปี 2022].

Convolutional Neural Networ. (2020). [ออนไลน์]. ได้จาก:https://guopai.github.io/ml-blog19.html [สืบค้นเมื่อ วันที่ 13 เดือน พฤษภาคม ปี 2022].

Convolutional Neural Network (CNN) คืออะไร. (2018). [ออนไลน์]. ได้จาก: https://medium.com/@natthawatphongchit/มาลองดูวิธีการคิดของ-cnn-กัน-e3f5d73eebaa [สืบค้นเมื่อ วันที่ 13 เดือน พฤษภาคม ปี 2022].

Convolutional Neural Network คืออะไร ภาษาไทย ตัวอย่างการทำงาน CNN, ConvNet กับชุดข้อมูล MNIST – ConvNet ep.1. (2019). [ออนไลน์]. ได้จาก:https://www.bualabs.com/archives/2461/what-is-convolutional-neural-network-cnn-convnet-mnist-deep-learning-convnet-ep-1/ [สืบค้นเมื่อ วันที่ 13 เดือน พฤษภาคม ปี 2022].

REPEAT BUYER PREDICTION USING MACHINE LEARNING TECHNIQUE. (2018). [ออนไลน์]. ได้จาก: http://ir-ithesis.swu.ac.th/dspace/bitstream/123456789/50/1/gs581130327.pdf [สืบค้นเมื่อ วันที่ 13 เดือน พฤษภาคม ปี 2022].

เจาะลึก Random Forest !!!— Part 2 of “รู้จัก Decision Tree, Random Forest, และ XGBoost!!!” (2018). [ออนไลน์]. ได้จาก: https://medium.com/@witchapongdaroontham/เจาะลึก-random-forest-part-2-of-รู้จัก-decision-tree-random-forest-และ-xgboost-79b9f41a1c1c [สืบค้นเมื่อ วันที่ 13 เดือน พฤษภาคม ปี 2022].

การตรวจจับป้ายสัญญาณจราจรด้วยเชพคอนเท็กซ์[ออนไลน์].ได้จาก https://www.thaiscience.info/Journals/Article/TJKM/10886900.pdf [สืบค้นเมื่อ วันที่ 13 เดือนพฤษภาคม ปี2022].