Introduction

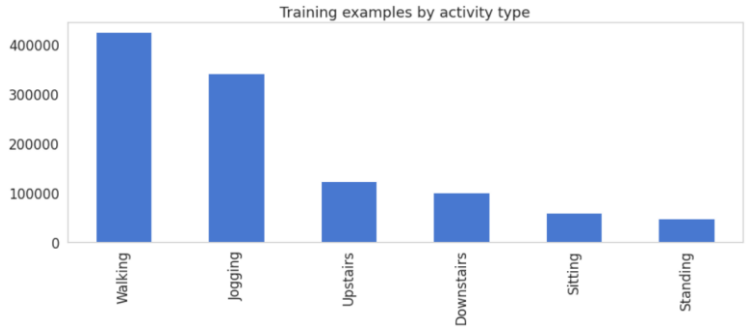
งานนี้เกี่ยวข้องกับ Human Activity Recognition (HAR) ที่รวบรวมจากเซ็นเซอร์ของ Smartphone เพื่อแบ่งแยกประเภทของกิจกรรมทั้งหมด 6 ประเภท โดยได้เปรียบเทียบประสิทธิภาพของ Model ระหว่าง การใช้ Traditional ML และ การใช้ Deep Learning ประกอบด้วย LSTM, GRU และ LSTM+CNN ซึ่งได้นำ Architecture จาก [1] มาปรับค่าพารามิเตอร์

Data

ชุดข้อมูลที่ใช้ในการทดลองมาจาก WISDM Lab โดยมีตัวอย่างทั้งหมด 1,098,207 ตัวอย่าง รวบรวมจากกลุ่มตัวอย่างทั้งหมด 36 คน เพื่อที่จะจัดเก็บท่าทางต่างๆเช่น การเดิน (Walking), การวิ่งเหยาะๆ (Jogging), การเดินขึ้นบันได (Upstairs), การเดินลงบันได (Downstairs), การนั่ง (Sitting) และการยืน (Standing) โดยแต่ละท่าทางมีจำนวนดังรูปที่ 1 และมีจำนวน Attributes เท่ากับ 6 ประกอบด้วย User ID, Activity, Timestamp, X-acceleration, Y-acceleration และ Z-acceleration ดังตารางที่ 1

user	activity	timestamp	x-axis	y-axis	z-axis
33	Jogging	49105962326000	-0.694638	12.680544	0.503953
33	Jogging	49106062271000	5.012288	11.264028	0.953424
33	Jogging	49106112167000	4.903325	10.882658	-0.081722
33	Jogging	49106222305000	-0.612916	18.496431	3.023717
33	Jogging	49106332290000	-1.184970	12.108489	7.205164

ตารางที่ 1 แสดงตัวอย่างของชุดข้อมูลแต่ละ Attribute

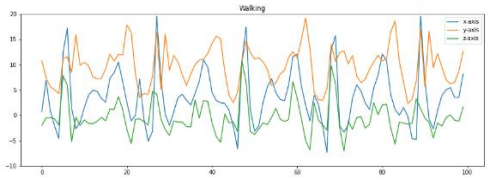


รูปที่ 1 แสดงจำนวนแต่ละท่าทางของชุดข้อมูล

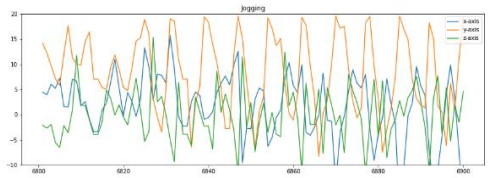
ชุดข้อมูลที่นำมาสร้าง model จะทำการลบข้อมูลที่ไม่ถูกต้องออกเช่น Null value และ timestamp ที่มีค่าเป็นศูนย์ ดังนั้นจะเหลือข้อมูลสำหรับการใช้งานทั้งหมด 1,085,360 ตัวอย่าง จากนั้นทำการแบ่งข้อมูลออกเป็น Train, Validation และ Test โดย user ที่ 1-25 เป็นชุดข้อมูล Train, User ที่ 26-32 เป็นข้อมูล Validation และ User ที่ 32-36 เป็นข้อมูล Test ซึ่งมีรายละเอียดดังตารางที่ 2

Activities	Train		Validation		Test	
	Samples	Percentage	Samples	Percentage	Samples	Percentage
Walking	288,339	39.05%	93,929	41.35%	41,637	34.73%
Jogging	231,513	31.36%	58,551	25.78%	40,266	33.58%
Upstairs	83,989	11.38%	27,045	11.91%	11,564	9.64%
Downstairs	64,123	8.69%	24,513	10.79%	11,558	9.64%
Sitting	38,833	5.40%	11,913	5.24%	8,922	7.44%
Standing	30,526	4.12%	11,184	4.93%	5,955	4.97%
Total	738,323	100.00%	227,135	100.00%	119,902	100.00%

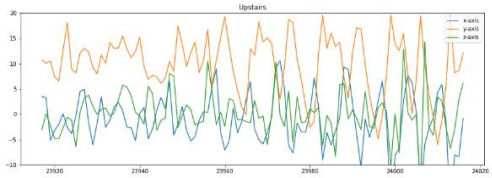
ตารางที่ 2 แสดงจำนวนข้อมูลของแต่ละ Activities ในแต่ละชุดข้อมูล



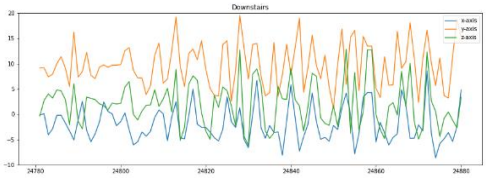
(a) ตัวอย่างข้อมูล Walking



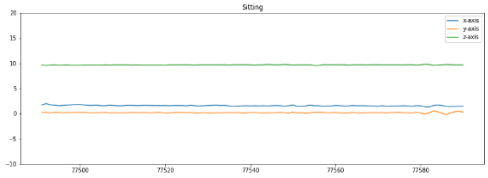
(b) ตัวอย่างข้อมูล Jogging



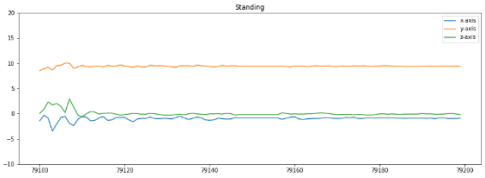
(c) ตัวอย่างข้อมูล Upstairs



(d) ตัวอย่างข้อมูล Downstairs



(e) ตัวอย่างข้อมูล Sitting



(f) ตัวอย่างข้อมูล Standing

รูปที่ 2 ภาพแสดงตัวอย่างข้อมูลแต่ละ Activities

Traditional ML

- **Data Preparation**

ในส่วนของ Traditional ML ทางกลุ่มสร้าง Model จากข้อมูลโดยแบ่งออกเป็น 2 ส่วนคือ ส่วนที่ Train Model โดยใช้ข้อมูลดิบโดยไม่มีการทำ Feature Engineering แต่มีการทำในส่วนของ Normalize โดยใช้ Standard Scaler โดยคำนวณตามสมการที่ 1 และ ส่วนที่มีการทำ Feature Engineering โดยทางกลุ่มได้แบ่งข้อมูลออกมาเป็นชุดสำหรับค่า x, y และ z ชุดละ 100 record (Window size) และมี Step size เท่ากับ 50 และใช้ Statistic Function ของค่า x, y, และ z ได้แก่ aad, aad_fft, above_mean, above_mean_fft, avg_result_accl, avg_result_accl_fft, energfft, energy, IQR, IQR_fft, kurtosis, kurtosis_fft, mad, mad_fft, mafft, max, maxmin_diff, maxmin_diff_fft, mean, mean_fft, median, median_fft, min, min_fft, neg_count, peak_count, peak_count_fft, pos_count, skewness, skewness_fft, sma, sma_fft, std, std_ff และมีการทำในส่วนของ Normalize โดยใช้ Standard Scaler

$$z = \frac{x_i - \mu}{\sigma} \qquad (1)$$

- **Model**

ในการ Train Model เนื่องจากเป็นปัญหา Classification ทางกลุ่มจึงเลือกใช้ Logistic Regression ของ Scikit-Learn โดยไม่มีการำ Hyperparameter Tuning

- **Result (Accuracy, Confusion matrix)**

ในส่วนแรกที่ใช้ข้อมูลดิบโดยที่ไม่มีการทำ Feature Engineering ซึ่ง Validation Set จะได้ค่า F1 Score อยู่ที่ 0.3790 เมื่อนำไปตรวจสอบกับ Test Set จะให้ค่า F1 Score เท่ากับ 0.3709 ซึ่งเป็นค่าเฉลี่ยแบบ Harmonic mean ระหว่าง Precision และ Recall ตามสมการที่ 2,3,4 และแสดงค่าของ Model กับ Test Set ดังตารางที่ 3 และ Confusion Matrix ในรูปที่ 3

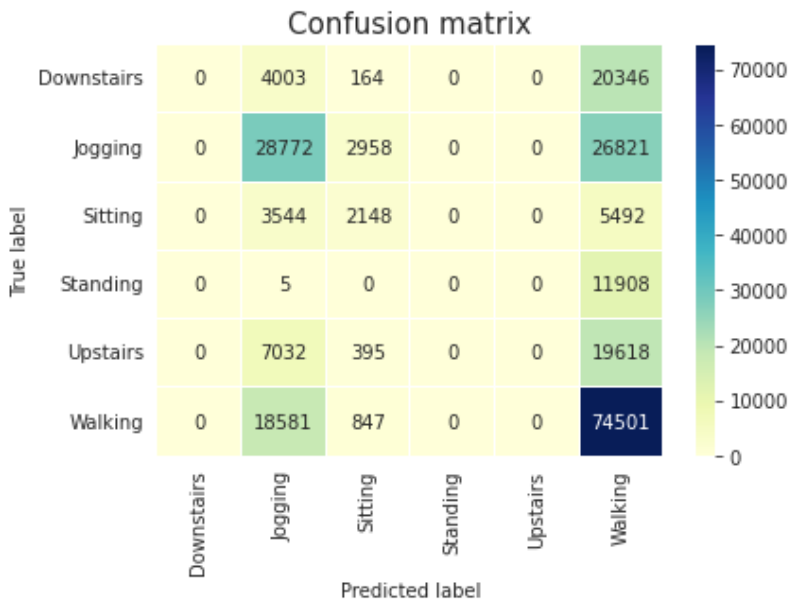
$$\text{Precision} = \frac{\text{TP}}{\text{TP+FP}} \qquad (2)$$

$$\text{Recall} = \frac{\text{TP}}{\text{TP+FN}} \qquad (3)$$

$$\text{F1 score} = 2 \cdot \frac{\text{Precision} \cdot \text{Recall}}{\text{Precision+Recall}} \qquad (4)$$

Class	Precision	Recall	F1 score
Downstairs	0.0000	0.0000	0.0000
Jogging	0.5807	0.3524	0.4171
Sitting	0.9103	0.4561	0.6077
Standing	0.0000	0.0000	0.0000
Upstairs	0.0000	0.0000	0.0000
Walking	0.3871	0.8635	0.5898

ตารางที่ 3 แสดงค่า Precision, Recall **และ** F1 score **ของ** Logistic Regression Model **กับข้อมูลดิบ**

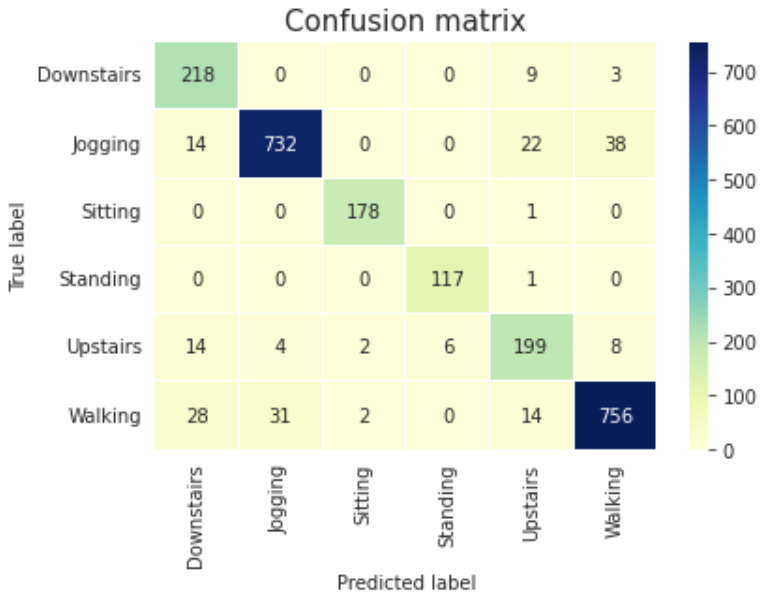


รูปที่ 3 แสดง Confusion Matrix **ของ** Logistic Regression Model **กับข้อมูลดิบ**

หลังจากที่ Train Model กับข้อมูลที่ยังไม่ทำ Feature Engineering พบว่า Model ไม่สามารถจับพฤติกรรมของบาง Class ได้เลย และในส่วนที่สามารถจับพฤติกรรมได้ ก็ได้ผลลัพธ์ที่ไม่ดีนัก ดังนั้น ทางกลุ่มจึงสร้าง Feature เพิ่มและทำการ Train Model ใหม่ พบว่าค่า F1 Score เพิ่มขึ้นอย่างมีนัยสำคัญ โดยที่ค่า F1 Score ของ Validate Set อยู่ที่ 0.8025 และ Test Set อยู่ที่ 0.9186 โดยมีค่า Precision และ Recall ของ Test Set ดังตาราง ดังตารางที่ 4 และ Confusion matrix ในรูปที่ 4

Class	Precision	Recall	F1 score
Downstairs	0.7956	0.9478	0.8651
Jogging	0.9544	0.9082	0.9307
Sitting	0.9780	0.9944	0.9861
Standing	0.9512	0.9915	0.9710
Upstairs	0.8089	0.8541	0.8309
Walking	0.9391	0.9097	0.9242

ตารางที่ 4 แสดงค่า Precision, Recall **และ** F1 score **ของ** Logistic Regression Model **กับข้อมูลที่ทำ** Feature Engineering



รูปที่ 4 แสดง Confusion Matrix **ของ** Logistic Regression Model **กับข้อมูลที่ทำ** Feature Engineering

จะเห็นได้ว่าหลังจากที่มีการทำ Feature เพิ่ม เมื่อนำข้อมูลชุดใหม่ไป Train Model ส่งผลให้ Model สามารถจับพฤติกรรมของ Class ต่างๆได้ดีขึ้น โดยที่จากเดิม F1 Score ของ Test Set อยู่ที่ 0.37 เพิ่มขึ้นมาเป็น 0.92 (+0.55)

เมื่อเปรียบเทียบ F1 Score ของ Validation Set และ Train Set พบว่า Model ตัวแรกที่ทางกลุ่มใช้ข้อมูลดิบในการ Train นั้นสามารถทำนาย Validation Set ได้ดีกว่า Test Set แต่เมื่อมีการทำ Feature เพิ่มก่อนนำไป Train Model พบว่า Model ตัวใหม่สามารถทำนาย Test Set ได้ดีกว่า Validation Set

ในส่วนของ Model ที่ Train กับข้อมูลที่มีการทำ Feature Engineering นั้นจะเห็นว่า Model สามารถทำนาย Class ส่วนใหญ่ได้ดี โดยมี F1 Score เกินกว่า 0.9 แต่จะมี Class Downstairs และ Upstairs ที่มี F1 Score ต่ำกว่า 0.9 โดยเมื่อพิจารณาร่วมกับ Confusion Matrix จะเห็นว่า 2 Class นี้น่าจะมีพฤติกรรมที่คล้ายกัน จึงส่งผลให้ Model ทำนายสลับกัน

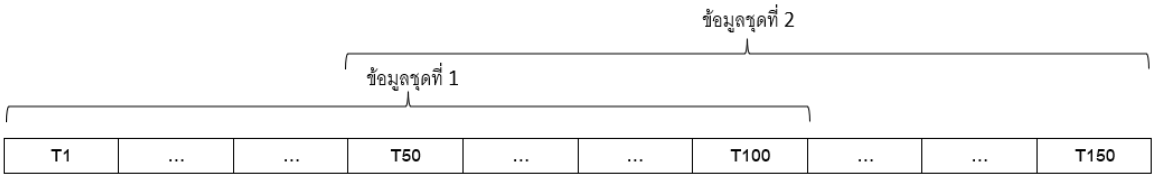
Deep Learning

- **Data Preparation**

Scaling and Normalization ในขั้นตอนนี้ทางกลุ่มมีการทำ Normalize เพื่อให้ข้อมูลอยู่ในช่วง 0-1 โดยใช้ Min-Max scaling โดยคำนวณตามสมการที่ 5

$$Xi = \frac{Xi - ximin}{ximax - ximin} \quad (i = 1,2,\ldots, n) \qquad (5)$$

เนื่องจากข้อมูลมีลักษณะเป็น sequence ในแต่ละ activities ดังนั้นจึงต้องมีการเตรียมข้อมูลสำหรับเป็น input ให้ model ใช้ในการเรียนรู้ โดยทางกลุ่มได้แบ่งข้อมูลออกมาเป็นชุดสำหรับค่า x, y และ z ชุดละ 100 record (Window size) และมี Step size เท่ากับ 50 ดังรูปที่ 5



รูปที่ 5 แสดงการเตรียมข้อมูลสำหรับ Model โดยกำหนด Window size เท่ากับ 100 และ Step size เท่ากับ 50

หลังจากทำการแบ่งข้อมูลออกเป็นชุดจะได้ข้อมูล Train, Validation และ Test มี Shape ดังตารางที่ 5

	Feature shape	Label shape
Train	(14765, 100, 3)	(14765, 6)
Validation	(4541, 100, 3)	(4541, 6)
Test	(2397, 100, 3)	(2397, 6)

ตารางที่ 5 แสดง Shape ของชุดข้อมูล Train, Validation และ Test

- **Architecture**

ในการทดลองทางกลุ่มได้เปรียบเทียบทั้งหมด 3 Model ประกอบ LSTM, GRU และ LSTM+CNN โดยมี Architecture ของแต่ละ model ดังตารางที่ 6

Layer	LSTM	GRU	LSTM+CNN
1	LSTM 128 units (return_sequences=True)	GRU 128 units (return_sequences=True)	LSTM 128 units (return_sequences=True)
2	LSTM 128 units (return_sequences=False)	GRU 128 units (return_sequences=False)	LSTM 128 units (return_sequences=True)
3	Dense units=6, activation=softmax	Dense units=6, activation=softmax	Conv1D filters=64, kernel_size=5, strides=1
4			MaxPool1D pool_size=2
5			Conv1D filters=128, kernel_size=3, strides=1
6			GlobalAveragePooling1D
7			BatchNormalization
8			Dense units=6, activation=softmax
Total Trainable parameters:	199,942	150,918	265,926

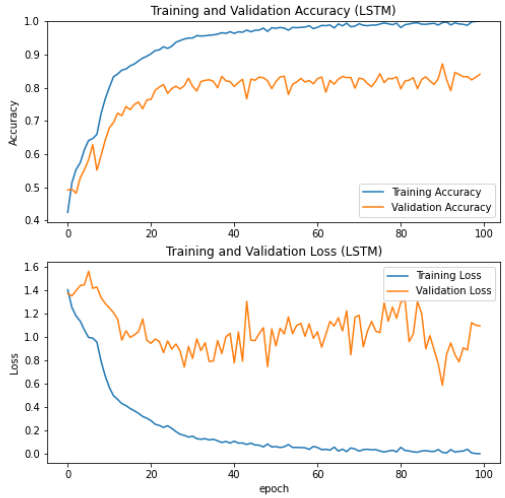
ตารางที่ 6 แสดง network structure ของ LSTM, GRU และ LSTM+CNN

- **Training**

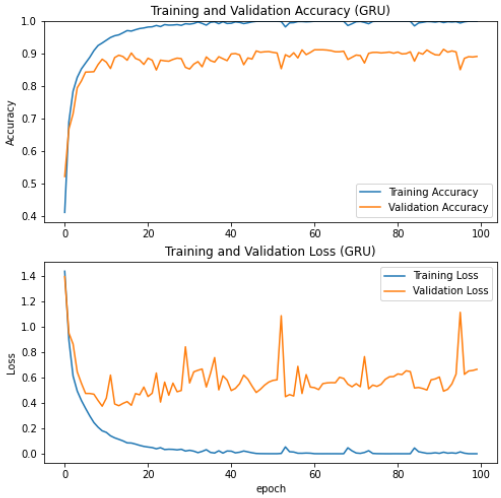
การ Train Model จะทำการทดลองโดยใช้ Google Colab โดยใช้ GPU Tesla P100 แต่ละ Network ถูก train ทั้งหมด 100 epoch โดยมี Batch size เท่ากับ 64 และ Learning rate เท่ากับ 0.001 และใช้ Loss Function เป็น Categorical Cross-Entropy จากนั้นจะทำการเก็บค่า weight ที่ให้ค่า Validation Accuracy สูงสุดไว้ ซึ่งแต่ละ Network ให้ Accuracy และ loss บน Validation set ดังตารางที่ 7 และ กราฟแสดง Accuracy และ loss ขณะการ train ดังรูปที่ 6 โดยที่ LSTM+CNN ให้ค่า Accuracy มากที่สุด

Model	Optimizer	Accuracy	Loss	Speed per Epoch
LSTM	Adam	0.8399	1.0928	~4s 17ms
GRU	Adam	0.8914	0.6637	~3s 14ms
LSTM+CNN	Adam	**0.9075**	0.6205	~5s 21ms

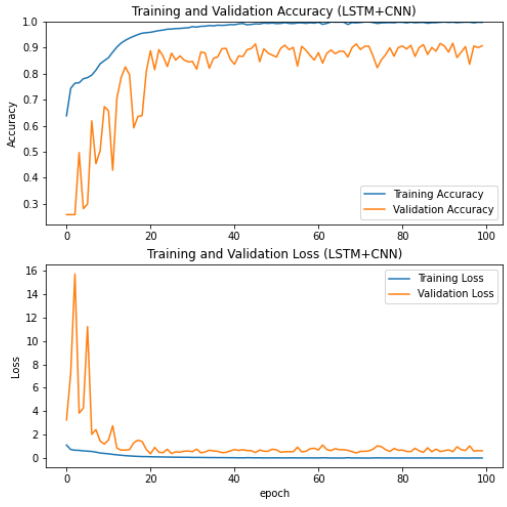
ตารางที่ 7 แสดงค่า Accuracy และ Loss บนชุดข้อมูล Validation



(a) LSTM model Accuracy 83.99%



(b) GRU model Accuracy 89.14%



(c) LSTM+CNN model Accuracy 90.75%

รูปที่ 6 กราฟแสดง Accuracy และ loss ขณะการ train

- **Result**

แต่ละ Model จะนำมาทดสอบกับ Test set โดยใช้ค่า Weight ที่ให้ค่า Validation Accuracy สูงสุด ซึ่งจะได้ค่า Accuracy, Loss และค่า F1 score ซึ่งเป็นค่าเฉลี่ยแบบ Harmonic mean ระหว่าง Precision และ Recall ตามสมการที่ 2,3,4 และแสดงค่าของแต่ละ model ดังตารางที่ 8

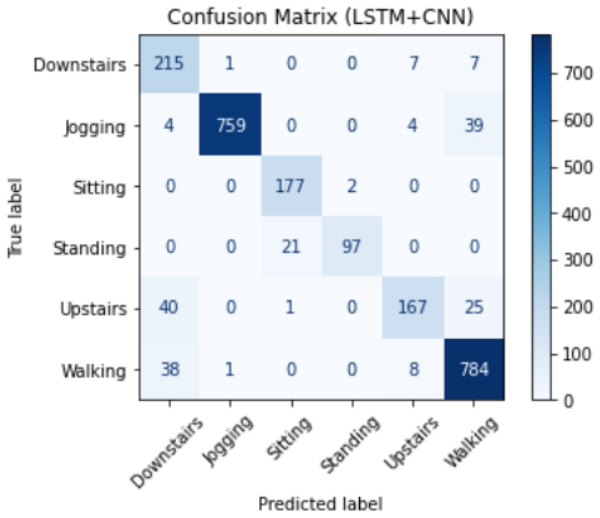
Model	Accuracy	Loss	F1 score (weighted avg)
LSTM	0.8716	0.5871	0.8380
GRU	0.9135	0.4921	0.8917
LSTM+CNN	**0.9176**	0.5459	**0.9179**

ตารางที่ 8 แสดงค่า Accuracy, Loss และ F1 score แต่ละ model

และมีค่า Precision, Recall และ F1 score ของแต่ละ Activities ของ model LSTM+CNN ดังตารางที่ 9 และ Confusion matrix ในรูปที่ 7

Class	Precision	Recall	F1 score
Downstairs	0.7239	0.9348	0.8159
Jogging	0.9974	0.9417	0.9687
Sitting	0.8894	0.9888	0.9365
Standing	0.9798	0.8220	0.8940
Upstairs	0.8978	0.7167	0.7971
Walking	0.9170	0.9434	0.9300

ตารางที่ 9 แสดงค่า Precision, Recall และ F1 score ของ model LSTM+CNN



รูปที่ 7 แสดง Confusion Matrix ของ LSTM+CNN

Discussion

เนื่องจากเป็นปัญหา Classification ทางกลุ่มจึงให้ความสำคัญกับ F1 Score มากกว่า Accuracy จากการทดสอบ Model ระหว่าง Logistic regression (Traditional Machine Learning) กับ LSTM, GRU, LSTM-CNN (Deep learning) กับ Validation Set เพื่อทำการแยกประเภทกิจกรรมทั้งหมด 6 ประเภท นั้นพบว่า F1 Score ของ Model ที่ดีที่สุดคือ LSTM-CNN รองลงมาคือ GRU, LSTM, Logistic Regression with Feature Engineering และ Logistic Regression without Feature Engineering

Model	F1 Score
Logistic Regression without Feature Engineering	0.3790
Logistic Regression with Feature Engineering	0.8025
LSTM	0.8693
GRU	0.9107
LSTM+CNN	**0.9151**

จากการทดสอบ Model ระหว่าง Logistic regression (Traditional Machine Learning) กับ LSTM, GRU, LSTM-CNN (Deep learning) กับ Test Set เพื่อทำการแยกประเภทกิจกรรมทั้งหมด 6 ประเภท นั้นพบว่า F1 Score ของ Model ที่ดีที่สุดคือ Logistic Regression รองลงมาคือ LSTM-CNN, GRU และ LSTM

Model	F1 Score
Logistic Regression without Feature Engineering	0.3709
Logistic Regression with Feature Engineering	**0.9186**
LSTM	0.8380
GRU	0.8917
LSTM+CNN	0.9179

ในส่วนของ Validation Set จะเห็นได้ว่า Model จากฝั่ง Deep Learning นั้นทำนายข้อมูลได้ดีกว่า Traditional Machine Learning โดยมี F1 Score อยู่ที่ 0.87 – 0.92 ในขณะที่ฝั่ง Traditional Machine Learning อยู่ที่ 0.80 แต่เมื่อนำ Model ไปทำนายกับ Test Set ผลลัพธ์กลายเป็นว่า Model จากฝั่ง Traditional Machine Learning สามารถให้ผลลัพธ์ที่ดีกว่า ซึ่งถือได้ว่าผิดคาดจากที่ทางกลุ่มได้ตั้งสมมติฐานไว้ในตอนแรก

จากผลลัพธ์ของ Model จะเห็นว่าในส่วนของ Logistic Regression with Feature Engineering สามารถทำนายข้อมูล Test Set ได้แม่นยำกว่า Validation Set และในส่วนของ Deep Learning เมื่อพิจารณารูปที่ 5 จะเห็นว่า Model Overfit ซึ่งทางกลุ่มคาดว่า สาเหตุอาจเกิดการแบ่ง Dataที่ไม่ดีนัก (ใช้ User ในการแบ่ง) ซึ่งอาจส่งผลกระทบต่อผลลัพธ์ของ Model

เมื่อพิจารณาในส่วนของ Confusion Matrix ของทุก Model จะพบว่ากิจกรรมที่ Model สามารถแยกประเภทได้แย่ที่สุดคือ Downstairs กับ Upstairs อาจเกิดจากข้อมูลของทั้งสองกิจกรรมมีพฤติกรรมที่คล้ายคลึงกันจึงส่งผลให้ Model ทำนายผิดในขณะที่กิจกรรมอื่นมีพฤติกรรมที่ชัดเจนเลยสามารถทำนายออกมาได้ถูกต้องแม่นยำกว่า

Conclusion

จากการทดสอบแต่ละ Model กับ Test Set พบว่า Logistic Regression เป็น Model ที่สามารถแยกประเภทกิจกรรมได้ดีที่สุด โดยมีค่า F1 Score สูงสุดอยู่ที่ 0.9186 แต่หากพิจารณาในเรื่องของการทำ Feature Engineering ประกอบไปด้วย นั้น ทางกลุ่มเห็นว่าการใช้ Model Deep Learning อย่าง LSTM+CNN ที่ให้ค่า F1 Score อยู่ที่ 0.9179 นั้นดีกว่าการใช้ Logistic Regression ด้วยเหตุผลดังนี้

1. เมื่อเปรียบเทียบในด้านเวลาที่ใช้ในการทำ Feature Engineering กับผลลัพธ์ที่ได้เพิ่มขึ้นมาเพียงแค่เล็กน้อย (F1 Score +0.0007) การใช้ LSTM+CNN ดูเป็นวิธีที่กว่า เพราะเพียงแค่จัดเตรียมข้อมูลให้อยู่ในรูปแบบที่ Model ต้องการ (Sequence Data) LSTM+CNN สามารถทำ Feature Extraction โดยที่ไม่ต้องทำในส่วนของ Feature Engineering เพิ่มเติม ซึ่งจะเห็นว่าในส่วนนี้ LSTM+CNN ประหยัดเวลากว่า
2. ในการทำ Feature Engineering หาก Feature ใหม่ที่สร้างขึ้นมาไม่สามารถบ่งบอกถึงพฤติกรรมของแต่ละกิจกรรมได้ จะส่งผลให้ประสิทธิผลของ Model แย่ลง (Garbage in Garbage out)

อย่างไรก็ตาม Deep Learning ยังมีข้อเสียอยู่ตรงที่ไม่สามารถอธิบายกระบวนการทำงาน(Black Box) ที่ใช้ในการทำนายข้อมูล ดังนั้นหากผู้ใช้งานต้องการ Model ที่สามารถอธิบายกระบวนการทำงานของ Model อาจจะต้องใช้ในส่วนของ Traditional Machine Learning แทน แต่ในการทดลองครั้งนี้ทางกลุ่มไม่ได้ต้องการที่จะใช้กระบวนการทำงานไปอธิบายต่อ แต่ให้ความสำคัญในเรื่อง Accuracy และ เวลา จึงแนะนำให้ใช้ LSTM+CNN ซึ่งเป็น Deep Learning Model ในการจำแนกประเภท Human Activity Recognition (HAR)

References

[1] K. Xia, J. Huang and H. Wang, "LSTM-CNN Architecture for Human Activity Recognition," in IEEE Access, vol. 8, pp. 56855-56866, 2020, doi: 10.1109/ACCESS.2020.2982225.