การวินิจฉัยโรคพาร์กินสันโดยใช้การเรียนรู้ของเครื่อง

นายหัสพล ธัมมิกรัตน์

วิทยานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาวิศวกรรมศาสตรมหาบัณฑิต
สาขาวิชาวิศวกรรมคอมพิวเตอร์ ภาควิชาวิศวกรรมคอมพิวเตอร์
คณะวิศวกรรมศาสตร์ จุฬาลงกรณ์มหาวิทยาลัย
ปีการศึกษา 2563

Parkinson's Disease Diagnosis Using Machine Learning

Mr. Hassapol Thummikarat

A Thesis Submitted in Partial Fulfillment of the Requirements
for the Degree of Master of Engineering in Computer Engineering
Department of Computer Engineering
FACULTY OF ENGINEERING
Chulalongkorn University
Academic Year 2020

หัวข้อวิทยานิพนธ์	การวินิจฉัยโรคพาร์กินสันโดยใช้การเรียนรู้ของเครื่อง
โดย	นายหัสพล ธัมมิกรัตน์
สาขาวิชา	วิศวกรรมคอมพิวเตอร์
อาจารย์ที่ปรึกษาวิทยานิพนธ์หลัก	ศาสตราจารย์ ดร.ประภาส จงสถิตย์วัฒนา

คณะวิศวกรรมศาสตร์ จุฬาลงกรณ์มหาวิทยาลัย อนุมัติให้นับวิทยานิพนธ์ฉบับนี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาวิศวกรรมศาสตรมหาบัณฑิต

..... คณบดีคณะวิศวกรรมศาสตร์

(ศาสตราจารย์ ดร.สุพจน์ เตชวรสินสกุล)

คณะกรรมการสอบวิทยานิพนธ์

..... ประธานกรรมการ

(ผู้ช่วยศาสตราจารย์ ดร.สุกรี สินธุภิญโญ)

ประภาส จงสถิตย์วัฒนา ..... อาจารย์ที่ปรึกษาวิทยานิพนธ์หลัก

(ศาสตราจารย์ ดร.ประภาส จงสถิตย์วัฒนา)

..... กรรมการ

(อาจารย์ ดร.ดวงดาว วิชาดากุล)

..... กรรมการภายนอกมหาวิทยาลัย

(ศาสตราจารย์ ดร.ณชล ไชยรัตนะ)

หัสพล ธัมมิกรัตน์ : การวินิจฉัยโรคพาร์กินสันโดยใช้การเรียนรู้ของเครื่อง. (Parkinson's Disease Diagnosis Using Machine Learning) อ.ที่ปรึกษาหลัก : ศ. ดร.ประภาส จงสถิตย์วัฒนา

วิทยานิพนธ์นี้นำเสนอวิธีการวินิจฉัยโรคพาร์กินสันด้วยการใช้การเรียนรู้ของเครื่องสำหรับการตรวจพบโรคพาร์กินสันในระยะเริ่มต้น โดยใช้โครงข่ายประสาทเทียมแบบวนซ้ำชนิดพิเศษ Long Short-Term Memory กับข้อมูลโรคพาร์กินสันที่ได้รับจากผู้เชี่ยวชาญของโรงพยาบาลจุฬาลงกรณ์ โดยข้อมูลที่ใช้ประกอบไปด้วยข้อมูลจากเซ็นเซอร์และคีย์บอร์ดจากการเก็บข้อมูลจากผู้ร่วมทดสอบซึ่งมีทั้งกลุ่มควบคุมและผู้ป่วยจำนวนหนึ่งผ่านตัวควบคุมที่เก็บข้อมูลคีย์บอร์ดและเซ็นเซอร์ ซึ่งข้อมูลเซ็นเซอร์มีค่าตัวแปรความเร่งและมุม ข้อมูลคีย์บอร์ดคือการกดคีย์บอร์ดเป็นตัวอักษรพร้อมทั้งเวลาการกดคีย์บอร์ด การวิจัยนี้ทำเพื่อช่วยการวินิจฉัยแยกแยะระหว่างอาการสั่นหรือมีปัญหาทางการควบคุมการเครื่องไหวของผู้ป่วยโรคอื่นและผู้ป่วยโรคพาร์กินสัน การวิจัยนี้ได้ใช้การเรียนรู้ของเครื่องเพื่อคัดกรองผู้ป่วยเบื้องต้นแทนการใช้แพทย์ผู้เชี่ยวชาญทางโรคพาร์กินสันสำหรับแพทย์แผนกผู้ป่วยนอกในวินิจฉัยการคัดกรองผู้ป่วยที่มีอาการใกล้เคียงอย่างการเคลื่อนไหว และความผิดปกติของระบบประสาทและสมอง ผลการวินิจฉัยพบว่าการเรียนรู้เครื่องสามารถตรวจพบการวินิจฉัยโรคพาร์กินสัน ได้ร้อยละความถูกต้องที่ 88.78 เปอร์เซ็นต์

สาขาวิชา วิศวกรรมคอมพิวเตอร์ ลายมือชื่อนิสิต .....

ปีการศึกษา 2563 ลายมือชื่อ อ.ที่ปรึกษาหลัก .....

6070398021 : MAJOR COMPUTER ENGINEERING

KEYWORD: Parkinson’s disease diagnosis, Machine Learning, Neural Network, Long Short-Term Memory, Binary Classification

Hassapol Thummikarat : Parkinson's Disease Diagnosis Using Machine Learning. Advisor: Prof. PRABHAS CHONGSTITVATANA, Ph.D.

This thesis proposes a machine learning method for early detection of Parkinson's Disease. Long Short-Term Memory approach is applied to patients’ data collected from Chulalongkorn Hospital by the experts. The data used in this research consists of the data of sensors and keyboard collected from both control and PD groups. The data from sensors measure acceleration and angle from accelerometer and gyroscope sensors respectively, and the data from keyboard measures value of the key button pressing and the time of the button pressing. The participants perform the prescribed tasks in the data collecting process. The result from this research can be used to assist the out-patience department (OPD) doctors in diagnose the symptoms which are tremor or neuro disorder. The result from the experiments shows the detection accuracy of 88.78%.

Field of Study: Computer Engineering Student's Signature .....

Academic Year: 2020 Advisor's Signature .....

กิตติกรรมประกาศ

ขอขอบคุณศาสตราจารย์ ดร.ประภาส จงสถิตย์วัฒนา อาจารย์ที่ปรึกษาที่มีความเมตตากรุณาให้คำแนะนำทั้งในด้านวิชาการและการใช้ชีวิตจนทำให้วิทยานิพนธ์สำเร็จลุล่วงได้ด้วยดี

ขอขอบคุณผู้ช่วยศาสตราจารย์ แพทย์หญิง พัทธมน ปัญญาแก้ว แพทย์ผู้เชี่ยวชาญโรคพาร์กินสันและกลุ่มโรคการเคลื่อนไหวผิดปกติ โรงพยาบาลจุฬาลงกรณ์ สภากาชาดไทย ผู้เชี่ยวชาญที่ให้ข้อมูลและคำปรึกษาทางการแพทย์

ขอขอบคุณคณะกรรมการสอบวิทยานิพนธ์ทุกท่านที่ให้ความอนุเคราะห์เป็นกรรมการสอบและให้คำแนะนำการชี้แนะในการศึกษาวิจัยจนสำเร็จเป็นวิทยานิพนธ์นี้

ขอขอบคุณคณาจารย์และเจ้าหน้าที่ทุกท่านในภาควิชาวิศวกรรมคอมพิวเตอร์ คณะวิศวกรรมศาสตร์ จุฬาลงกรณ์มหาวิทยาลัย ที่ให้คำแนะนำและความรู้แก่ข้าพเจ้า

ขอบคุณทุกคนศิษย์ปัจจุบันและศิษย์เก่าในภาควิชาและห้องปฏิบัติการ ISL (Intelligent Systems Laboratory) ที่เป็นกำลังใจและช่วยชี้แนะให้แก่ข้าพเจ้าตลอดมา

ขอขอบคุณครอบครัว ญาติ มิตรสหาย และตัวข้าพเจ้าเอง ที่ส่งเสริมข้าพเจ้าจนมาถึงจุดนี้

หัสพล ธัมมิกรัตน์

สารบัญ

หน้า

สารบัญตาราง

หน้า

สารบัญภาพ

หน้า

บทที่ 1
ที่มาและความสำคัญของปัญหา

การวินิจฉัยโรคพาร์กินสันโดยแพทย์ทางประสาทวิทยาจะทำการวินิจฉัยบนพื้นฐานของประวัติการรักษา พิจารณาจากสัญญาณและอาการต่างๆ และการทดสอบทางประสาทวิทยาและทางกายภาพ แพทย์อาจจะแนะนำการใช้การตรวจ specific-photon emission computerized tomography (SPECT) ที่เรียกว่าการตรวจ dopamine transporter (DAT) เป็นการตรวจที่ดีที่สุด ซึ่งโดยทั่วไปผู้ป่วยไม่จำเป็นต้องใช้การตรวจแบบนี้ บางครั้งอาจใช้ค่าที่ได้จากการวิเคราะห์ (lab tests) อย่างเช่น การตรวจเลือด ซึ่งอาจจะพบเหตุปัจจัยของการเกิดอาการต่างๆ การตรวจด้วยภาพอย่างเช่น MRI, CT, ultrasound และ PET อาจช่วยสำหรับความผิดปกติอื่นๆ แต่ไม่ได้ช่วยสำหรับโรคพาร์กินสันโดยตรง นอกจากนี้ การทดสอบด้วยยาอย่าง carbidopa-levodopa (Rytary, Sinemet และอื่นๆ) ซึ่งเป็นยารักษาโรคพาร์กินสัน ด้วยการใช้ยากในปริมาณที่เพียงพอจะแสดงผลประโยชน์จากการรับยา เพราะถ้าปริมาณยาต่อวันที่ได้น้อยหรือ 1-2 ครั้งต่อวันนั้นไม่น่าเชื่อถือการที่ใช้ยาแล้วได้ผลที่ดีขึ้นอย่างมีนัยสำคัญมักจะเป็นการยืนยันการวินิจฉัยโรคพาร์กินสัน

บางครั้งการวินิจฉัยโรคพาร์กินสันอาจใช้เวลานาน แพทย์จึงแนะนำให้ติดตามการผลรักษาด้วยแพทย์เฉพาะทางหรือนักประสาทวิทยาที่จะประเมินเหตุปัจจัยและอาการต่างๆผ่านเวลาและการวินิจฉัยโรคพาร์กินสัน

- Tremor การสั่นหรือเขย่า
- Bradykinesia ความช้าของหารเคลื่อนไหว
- Rigidity ความล้าของแขน ขา หรือลำตัว
- Postural instability ความสามารถในการทรงตัวและความเสี่ยงในการล้ม

Motor symptoms ลำดับขั้นอาการของโรคพาร์กินสันแบ่งได้ด้านล่าง การสั่นของแขนขวาอาจจะดูเบาๆ แต่ถ้าคนนั้นถนัดมือขวาและอาการรุนแรง อาจมีผลกระทบต่อชีวิตประจำวันได้

- Mild stage อาการที่เป็นการรบกวน แต่ไม่ขัดขวางการทำงานต่างๆ และยามักสามารถควบคุมการพัฒนาอาการของโรค โดยสังเกตจาก
  - แขนไม่สามารถแกว่งได้อย่างอิสระเวลาเดิน
  - ใบหน้าไม่สามารถแสดงอารมณ์ได้

- ขารู้สึกหนัก
- ท่าทางเริ่มเอียงเล็กน้อย
- ลายมือขนาดเล็กลง
- แขนหรือขาแข็งทื่อ
- มีอาการแค่ข้างเดียว เหมือนกับมีอาการสั่นหรือเขย่าที่แขนข้างเดียว

- Moderate stage บ่อยครั้งใน 3-7 ปี จะเห็นถึงความเปลี่ยนแปลงต่างๆมากขึ้น ตอนแรกอาจจะมีปัญหาตอนติดกระดุมเสื้อแต่เมื่อถึงระยะนี้อาจจะติดกระดุมไม่ได้เลย และอาจจะพบว่าฤทธิ์ยาเริ่มหายไปในระหว่างรอบการใช้ยา ซึ่งสามารถคาดเดาอาการได้จาก
  - การพูดมีการเปลี่ยนแปลง เหมือนกับว่าเสียงเบาลง หรือต้นเสียงชัดแต่ปลายเสียงหาย
  - การกลืนลำบาก
  - การล้ม
  - การเคลื่อนไหวช้า
  - การก้าวขา ระยะเท้าสั้นและลากเท้า
- Advanced stage ผู้ป่วยบางคนไม่เคยมาถึงระยะนี้ เกิดเมื่อการรักษาไม่ได้ผลได้มากเท่ากับความพิการอย่างรุนแรง
  - ใช้ชีวิตจำกัดอยู่แค่กับเตียงหรือรถเข็น
  - ไม่สามารถอยู่ได้ด้วยตัวเอง
  - มีท่าทางผิดปกติรุนแรงที่คอ หลัง และสะโพก
  - ต้องการความช่วยเหลือในการดำรงชีวิตประจำวัน

Non-Motor symptoms เกือบทุกคนที่เป็นโรคพาร์กินสันมีอาการหนึ่งในอาการเหล่านี้ เมื่ออาการรุนแรงเหมือนจะมีมากกว่าประเด็นของการสั่งการที่จะนำปสู่ความพิการต่างๆหรือทำให้ต้องใช้ชีวิตในสถานพยาบาล อาการเหล่านี้จะแสดงออกมาแทบตลอดเวลาแต่จะเป็นไปตาม general trend

สิ่งที่อาจจะปรากฏก่อน อาจจะมีประเด็นปัญหาเหล่านี้ก่อนที่จะมีอาการของการสั่งการเหมือนกับการสั่นต่างๆ

- อาการท้องผูก
- อาการซึมเศร้า
- การรับกลิ่นได้แย่ลง

- ความดันโลหิตต่ำเมื่อยืนขึ้น
- เจ็บปวด
- ปัญหาการนอน

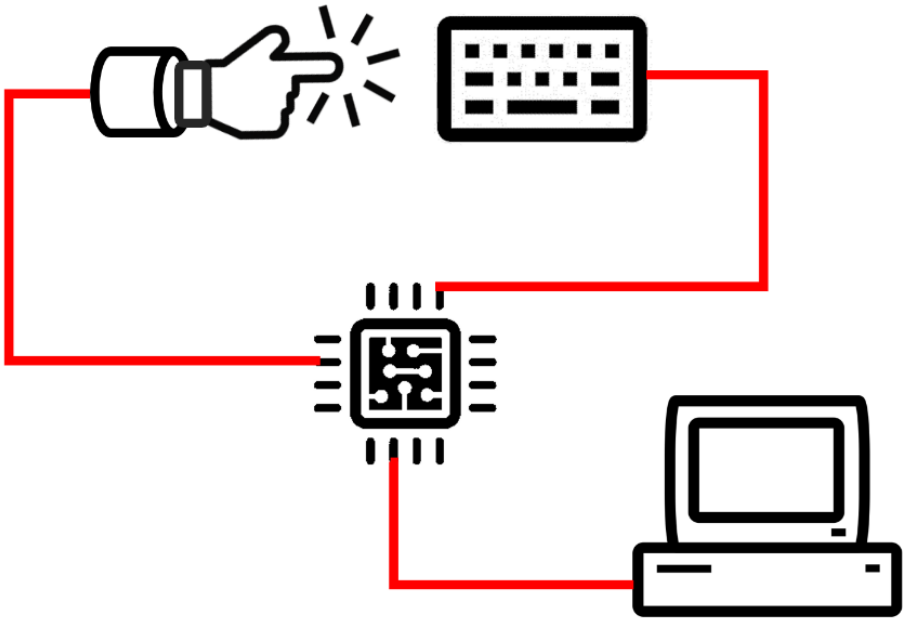
อาการเหล่านี้อาจจะเกิดหลังจากโรคพาร์กินสัน แม้ว่าการมีอาการเหล่านี้ก็ไม่ได้หมายความว่าเป็นโรคพาร์กินสัน นักวิทยาศาสตร์ยังคงกำลังหาความเชื่อมโยงอยู่ อาจจะมีอาการช่วง mild เมื่อทำการคิดหรือวางแผน เหมือนกับวับว่า การลืมบ่อยครั้ง สมาธิสั้น และประสบความยากในการบริหารจัดการ น้ำลายไหลย้อยและจำเป็นต้องปัสสาวะบ่อยเป็นปกติ

สิ่งที่อาจจะปรากฏหลัง โรคสมองเสื่อม (Dementia) และโรคจิต (Psychosis) เป็นปัญหาสุขภาพทางจิตวิทยาหลักที่จะเกิดขึ้นหลังการเป็นโรคสักระยะหนึ่ง โรคจิตเป็นสภาวะร้ายแรงเมื่อผู้ป่วยเห็นหรือได้ยินสิ่งที่ไม่มีอยู่ตรงนั้นหรือเชื่อในสิ่งไม่มีอยู่จริง โรคสมองเสื่อมหมายถึงผู้ป่วยไม่สามารถคิดได้อีกต่อไป จำได้อีกต่อไป และมีสติเหตุผลดีพอที่จะใช้ชีวิตปกติ

การวินิจฉัยโรคพาร์กินสันโดยมาตรฐานแล้วในปัจจุบันทางการแพทย์ แพทย์จะใช้การสังเกตจากการขยับร่างกายของผู้ป่วยเป็นหลัก โดยทำกันหลายแบบและช่วงเวลาต่างกัน หากมีเครื่องมือเข้ามาช่วย เครื่องมือนั้นก็มีราคาสูงมาก ทั้งยังยังมีค่าการบำรุงดูแลรักษาสูงทั้งเครื่องมือและโปรแกรมหากจะทำการอัพเกรดจะได้จากค่าลิขสิทธิ์ที่มีราคาสูง

ด้วยเหตุนี้จึงได้มีการพัฒนาอุปกรณ์และพัฒนาโปรแกรมที่จะช่วยให้การวินิจฉัยและรักษาโรคพาร์กิน-สันนี้ขึ้นมาเพื่อใช้งานในราคาที่ถูกกว่าและประสิทธิภาพใกล้เคียง โดยที่สามารถเลือกการทำงานได้ตามความต้องการของแพทย์ที่ตรวจใช้งานจริงให้สะดวก ใช้ง่ายและราคาถูก

เครื่องมือที่ใช้ในการเก็บข้อมูลหลักๆนั้นมาจาก 2 แหล่ง ประกอบไปด้วยคีย์บอร์ดและอุปกรณ์วัดที่ข้อมือผู้ป่วยซึ่งประกอบไปด้วยเซ็นเซอร์วัดมุมและวัดความเร่ง ผ่านตัวควบคุมที่จะรวมข้อมูลจากทั้งคีย์บอร์ดและเซ็นเซอร์ที่ข้อมือ หลังจากนั้นบันทึกข้อมูลลงบนเครื่องคอมพิวเตอร์ซึ่งมีโปรแกรมที่ทำหน้าที่จับเวลา เลือกรายการและรายละเอียดของผู้ที่ถูกเก็บข้อมูล



ภาพที่ 1.1 แผนภาพการเก็บข้อมูลโรคพาร์กินสัน

ภาพที่ 1.2 เซ็นเซอร์นิ้วมือและตัวควบคุม

ภาพที่ 1.3 อุปกรณ์ที่สำหรับเก็บข้อมูล

1.1 จุดประสงค์และขอบเขตการวิจัย

งานวิจัยนี้ทำขึ้นเพื่อการนำการเรียนรู้เครื่องมาประยุกต์ใช้กับข้อมูลทางการแพทย์ โดยการนำการเรียนรู้ของเครื่องมาเป็นตัวช่วยในการตัดสินใจการวินิจฉัยอาการของโรคพาร์กินสันและโรคใกล้เคียง เพื่อเป็นตัวช่วยในการตัดสินใจสำหรับแพทย์แผนกผู้ป่วยนอกในการคัดกรองผู้ป่วยเบื้องต้น มีจุดประสงค์เพื่อลดภาระงานของแพทย์ผู้เชี่ยวชาญในการตรวจวินิจฉัยแยกโรคที่เกี่ยวกับการเคลื่อนไหวผิดปกติและโรคพาร์กินสันออกจากกัน ซึ่งในงานวิจัยนี้ใช้ข้อมูลของโรคพาร์กินสันจากแพทย์ผู้เชี่ยวชาญด้านโรคพาร์กินสันเป็นผู้รวบรวมและทำแบบทดสอบสำหรับข้อมูลที่ใช้สำหรับงานวิจัยนี้

งานวิจัยนี้มีขอบเขตการดำเนินงานวิจัยที่การใช้การเรียนรู้ของเครื่องเพื่อให้ได้ผลการดำเนินการเรียนรู้ของเครื่องนั้น ดังเช่นจุดประสงค์เพื่อเป็นตัวช่วยตัดสินใจในการตรวจวินิจฉัยโรคเป็นอีกหนึ่งวิธีการประยุกต์เทคนิคทางวิศวกรรมกับทางการแพทย์

1.2 ผลที่ได้รับจาการวิจัย

ผลการดำเนินงานวิจัยของการนำการเรียนรู้ของเครื่องมาใช้กับข้อมูลทางการแพทย์ พบว่าประสิทธิภาพของการทำงานของการเรียนรู้ของเครื่องนั้น มีประสิทธิภาพระดับเพียงพอสำหรับการเป็นตัวช่วยในการตัดสินใจในการตรวจวินิจฉัยข้อมูลโรคพาร์กินสัน ซึ่งแสดงให้เห็นว่าสามารถประยุกต์ใช้การเรียนรู้ของเครื่องกับข้อมูลทางการแพทย์ในฐานะเป็นตัวช่วยการตัดสินใจได้

บทที่ 2
ทฤษฎีที่เกี่ยวข้อง

2.1 Neural Networks

โครงข่ายงานประสาท (Neural Networks) ในหนึ่งโครงข่ายประสาทจะประกอบด้วยข้อมูลขาเข้า vector x ซึ่งสามารถพิจารณาเป็นเซ็ตของหลายเซลล์ประสาท(neurons) แต่ละเซลล์ประสาท (neuron)จะต่อกับชั้นซ่อนตัว (hidden layer) ของหลายเซลล์ ผ่านเซ็ตของน้ำหนักที่ถูกเรียนรู้แล้ว ข้อมูลขาออกของเซลล์ซ่อนตัว (hidden neuron outputs) ณ j^{th} คือ

$$h_j = \phi(\sum_i w_{ij} x_i)$$

เมื่อ ϕ คือ activation function ชั้นซ่อนตัวนั้นเชื่อมต่อกันอย่างทั่วถึงกันทั้งหมด (fully connected) กับชั้นข้อมูลขาออก (output layer) จะได้ข้อมูลขาออกของเซลล์ประสาทขาออก (output neuron outputs) ณ j^{th} คือ

$$y_j = \phi(\sum_i v_{ij} h_i)$$

ถ้าต้องการความน่าจะเป็นเหล่านั้น สามารถแปลงชั้นข้อมูลขาออก (output layer) ด้วย softmax function

Matrix notation:

$$h = \phi(Wx)$$

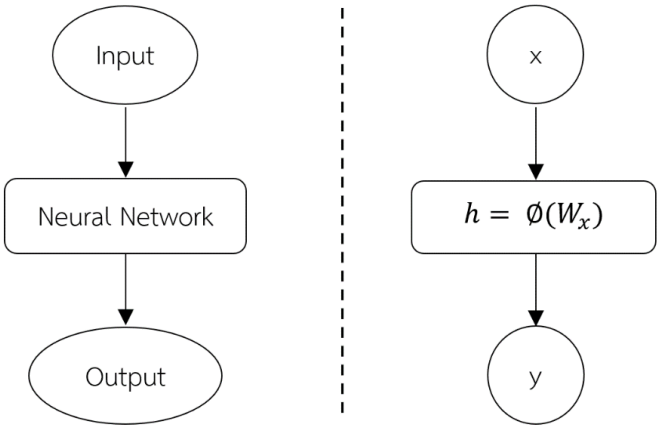
$$y = Vh$$

เมื่อ x = input vector

W = a weight matrix connecting the input and hidden layers

V = a weight matrix connecting the hidden and output layers

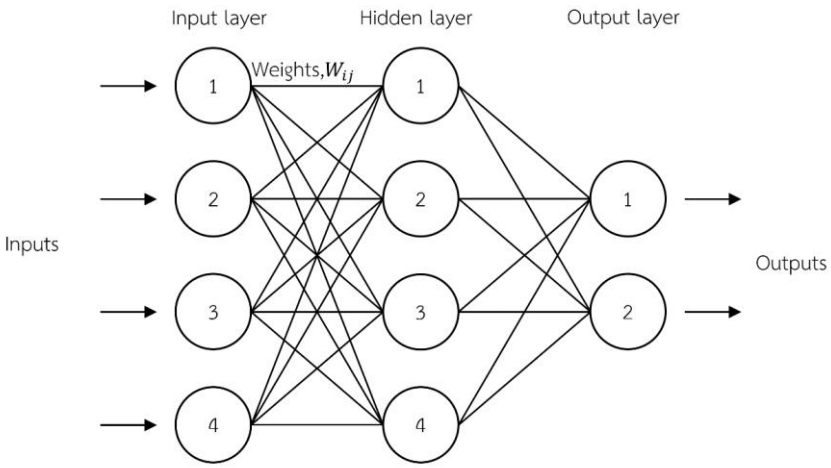
Common activation functions สำหรับ ϕ คือ sigmoid function, $\sigma(x)$ ซึ่งคือการการเอาเลขมาให้อยู่ในช่วง 0 ถึง 1 หรือ hyperbolic tangent, $tanh(x)$ การการเอาเลขมาให้อยู่ในช่วง -1 ถึง 1 หรือ rectified linear unit, $ReLU(x) = \max(0, x)$



ภาพที่ 2.1 โครงข่ายประสาทเทียม

โครงข่ายงานประสาทแบบป้อนไปหน้า (Feedforward Networks) มีลักษณะคือ การป้อนข้อมูลขาเข้า(input) เข้าสู่โครงข่ายประสาทแล้วถูกเปลี่ยนได้เป็นข้อมูลขาออก(output) ด้วยการเรียนรู้แบบมีผู้สอน (supervised Learning) คือการทำให้คอมพิวเตอร์สามารถหาคำตอบของปัญหาได้ด้วยตัวเอง หลังจากเรียนรู้จากชุดข้อมูลตัวอย่างไปแล้วระยะหนึ่ง ข้อมูลขาออกที่ผ่านการกระทำทางคณิตศาสตร์จะถูกติดฉลากแล้วถูกประยุกต์ให้กลายเป็นข้อมูลขาเข้า โดยจะผ่านจุด(node)ในโครงข่ายเพียงครั้งเดียว

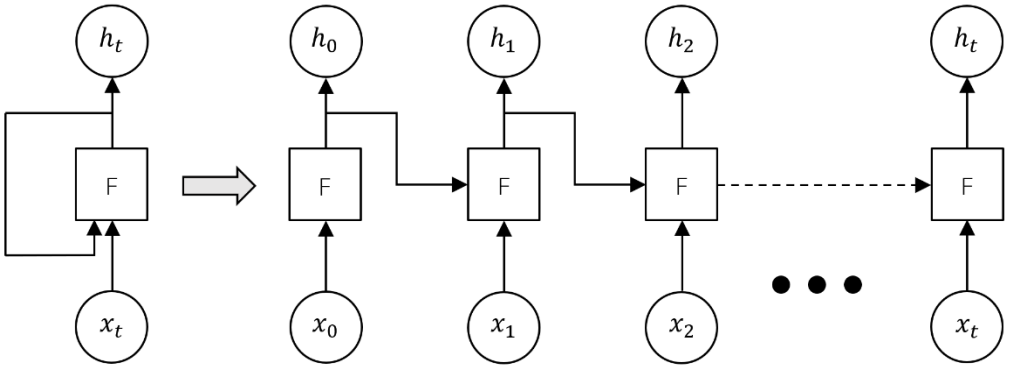
โครงข่ายงานประสาทแบบป้อนไปหน้านั้นไม่ได้สนใจลำดับเชิงเวลา สนใจเพียงแค่ข้อมูลขาเข้าที่ถูกติดฉลากแล้ว ณ สถานะปัจจุบัน



ภาพที่ 2.2 โครงข่ายประสาทเทียมแบบป้อนไปหน้า

2.2 Recurrent Neural Networks (RNNs)

โครงข่ายประสาทเทียมแบบวนซ้ำ (Recurrent Neural Network) นอกจากจะรับข้อมูลขาเข้าแล้วยังรับเวลาก่อนหน้านี้เข้ามารวมด้วย ซึ่งเป็นข้อมูลขาออก ณ สถานะก่อนหน้านี้ ด้วยธรรมชาติของนี้ ทำให้ทราบว่า โครงข่ายประสาทเทียมแบบวนซ้ำนั้นสามารถทำงานกับลำดับต่อเนื่องหรือรายการ ดังแผนภาพต่อไปนี้ ข้อมูลขาเข้า $\mathbf{X_t}$ และข้อมูลขาออก $\mathbf{h_t}$ การวนซ้ำ A ที่ให้ข้อมูลผ่านจากจุดหนึ่งไปจุดต่อไปในโครงข่าย

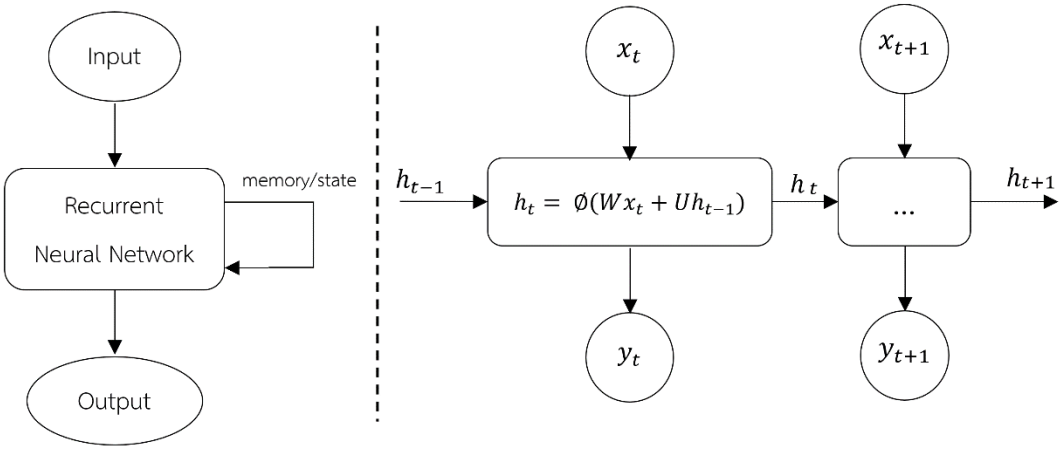


ภาพที่ 2.3 โครงข่ายประสาทเทียมแบบวนซ้ำ(1)

RNN equations:

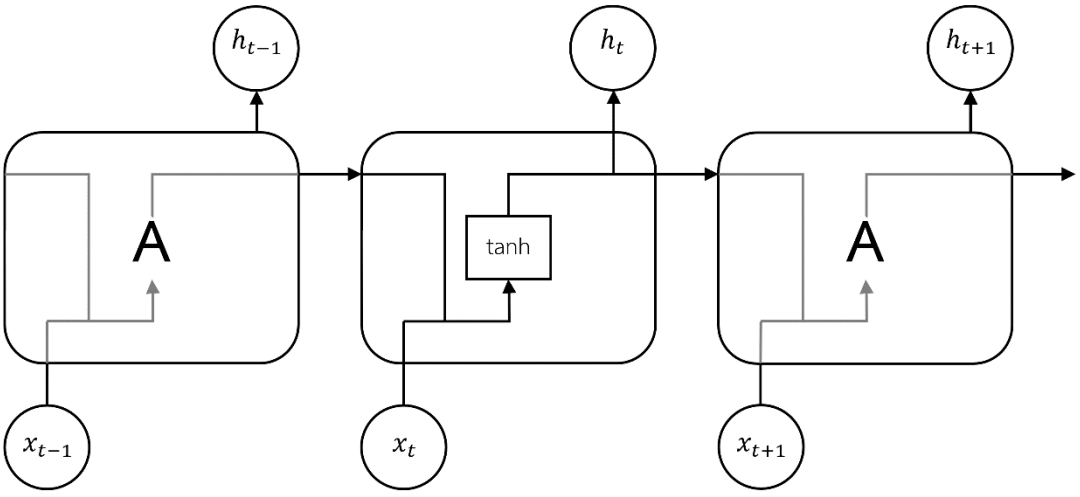
$$\mathbf{h_t} = \phi(\mathbf{W x_t} + \mathbf{U h_{t-1}})$$
$$\mathbf{y_t} = \mathbf{V h_t}$$

สถานะซ่อนตัว (hidden state) ถูกคำนวน ณ เวลา $\mathbf{t}$ ($\mathbf{h_t}$ คือ internal knowledge) จะถูกป้อนกลับ ณ เวลาต่อไป (hidden state, knowledge, memories และ beliefs จะใช้ในการอธิบาย $\mathbf{h_t}$)



ภาพที่ 2.4 โครงข่ายประสาทเทียมแบบวนซ้ำ(2)

เมื่อเป็นเรื่องยากที่จะจะเก็บรักษาความจำระยะยาว (long-term memory) จึงอยากให้โครงข่ายประสาทสามารถเรียนรู้ที่จะปรับ (update) ตัวกับความเชื่อของ (beliefs) ในทางที่ความรู้ (knowledge) ของทั้งหมดนั้นวิวัฒนาการ (evolve) ไปอย่างค่อยเป็นค่อยไป

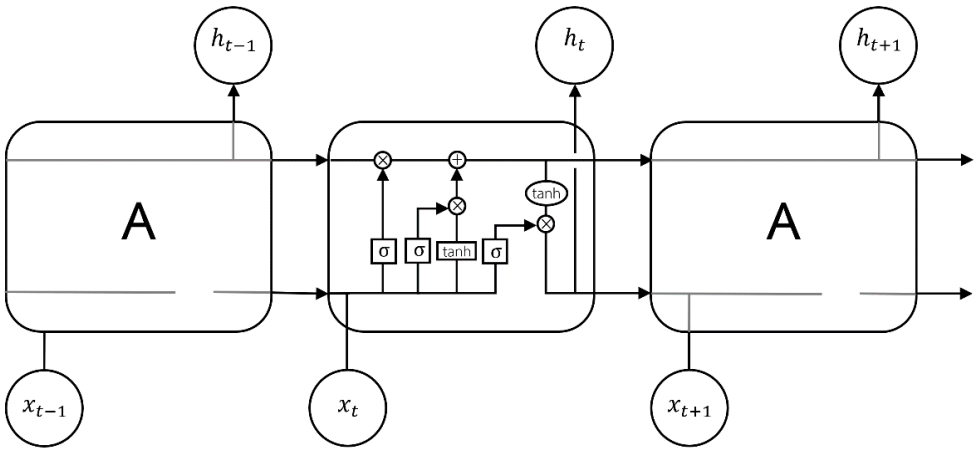


ภาพที่ 2.5 โครงข่ายประสาทเทียมแบบวนซ้ำ(3)

2.3 Long Short-term Memory networks (LSTMs)

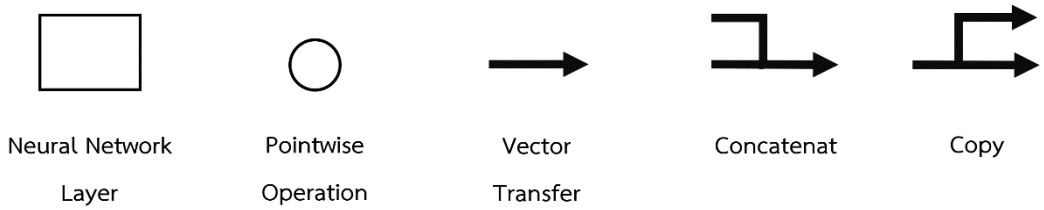
Long Short-term Memory networks (LSTMs) แปลงความจำของตัวเองไปในทางที่ค่อนข้างแม่นยำถูกต้องด้วยการ specific learning mechanisms สำหรับทุกเศษเสี้ยวข้อมูลที่จะจำ ที่จะปรับตัว ที่จะสนใจ ซึ่งจะช่วยให้สามารถติดตามข้อมูลได้ในระยะเวลาที่นานกว่า

1. เพิ่ม Forgetting mechanism เข้าไป ในท้ายที่สุดเมื่อ inputs เข้ามาจะเลือกว่า เก็บหรือปล่อย beliefs นั้นไปเรียกว่า“forgetting/remembering” mechanism
2. เพิ่ม Saving mechanism เมื่อตัวแบบ (model) ได้พบกับข้อมูลใหม่เข้ามา ประเมินว่าจะบันทึกข้อมูลนั้นหรือไม่
3. เมื่อเวลาข้อมูลใหม่เข้ามา ตัวแบบจะเริ่มจากการลืม long-term information ก่อน หลังจากนั้นจะเรียนรู้ว่าส่วนไหนของข้อมูลใหม่นั้นควรค่าแก่การให้งานและบันทึกสู่ long-term memory
4. Focusing long-term memory into working memory ในท้ายที่สุดตัวแบบจะต้องสามารถเรียนรู้ว่าส่วนไหนของ long-term memory ของ LSTM เป็นประโยชน์ ณ ขณะนั้น ดังนั้นนอกจากใช้ full long-term memory ตลอดเวลา จึงเรียนรู้ว่าส่วนไหนควรให้ความสนใจแทน



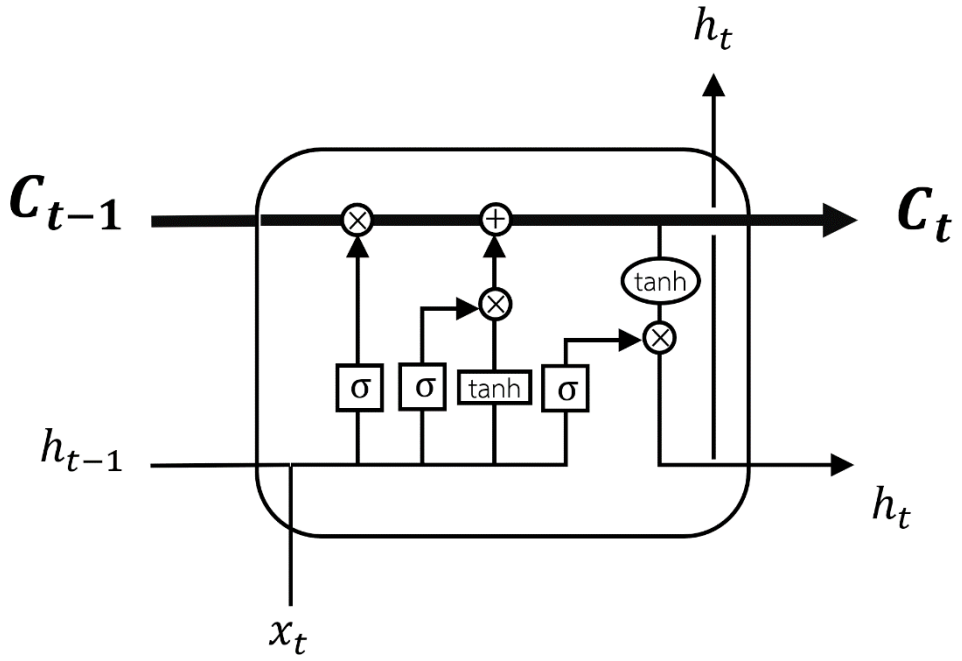
ภาพที่ 2.6 LSTMs

สัญลักษณ์ต่างๆมีความหมายดังนี้



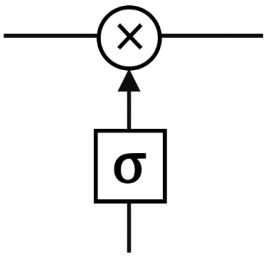
ภาพที่ 2.7 สัญลักษณ์สำหรับแผนภาพ

แนวคิดหลักของ LSTMs อยู่ที่เส้นแนวนอนด้านบนที่ทอดผ่านทั้งแผนภาพ สถาณะของเซลล์ (cell state) เป็นเหมือนสายพาน (conveyer belt) ที่จะปฏิสัมพันธ์เชิงเส้นเพียงเล็กน้อยเท่านั้น จึงง่ายสำหรับข้อมูลที่แค่ไหลผ่านอย่างไม่เปลี่ยนแปลง



ภาพที่ 2.8 การทำงานหลักของ LSTM

LSTM มีความสามารถที่จะเพิ่มหรือลดข้อมูลของสถานะเซลล์ด้วยการควบคุมผ่านโครงสร้างที่เรียกว่า gates ซึ่งเป็นตัวเลือกสำหรับจะให้ข้อมูลผ่าน อย่างเช่น sigmoid neural net layer และ pointwise multiplication operation ด้านล่างนี้



ภาพที่ 2.9 Gates

Sigmoid layer outputs มีค่าระหว่าง 0 ถึง 1 หมายความว่าจะให้ข้อมูลผ่านไปมากเท่าไร ถ้า 0 แปลว่าไม่ให้อะไรผ่าน และ 1 หมายถึงผ่านไปได้ทั้งหมด

Mathematically:

ณ เวลา $\mathbf{t}$ ได้รับข้อมูลขาเข้าใหม่ $\mathbf{x_t}$ ยังมี long-term memory และ working memory ที่ถูกส่งผ่านมาจากช่วงเวลาก่อน $\mathbf{ltm_{t-1}}$ และ $\mathbf{wm_{t-1}}$ ทั้งสองเป็น n-length vectors ที่ต้องการจะอัพเดต

การรู้ว่าชิ้นไหนของ long-term memory ที่จะบันทึกต่อและอันไหนที่จะทิ้งไป ดังนั้นจึงใช้ new input และ working memory ที่จะเรียนรู้ remember gate ของจำนวน n ระหว่าง 0 ถึง 1 ซึ่งแต่ละการตัดสินใจว่า long-term memory จำนวนเท่าไรที่จะเก็บไว้ (1 คือเก็บไว้ 0 คือลืมให้หมด) จะได้ remember gate ดังนี้

$$\mathbf{remember_t} = \sigma(\mathbf{W_r x_t} + \mathbf{U_r wm_{t-1}})$$

(ใช้ sigmoid function ซึ่งมีค่าระหว่าง 0 ถึง 1)

ต่อจากนี้คำนวนข้อมูลที่สามารถเรียนรู้จาก $\mathbf{x_t}$ อย่าง candidate addition to long-term memory

$$\mathbf{ltm'_t} = \phi(\mathbf{W_l x_t} + \mathbf{U_l wm_{t-1}})$$

(ϕ คือ activation function โดยปกติจะเป็น $\mathbf{tanh}$)

ก่อนที่จะเพิ่ม candidate ไปที่ความจำต้องเรียนรู้ว่าส่วนไหนที่ควรค่าแก่การบันทึก

$$\mathbf{save'_t} = \sigma(\mathbf{W_s x_t} + \mathbf{U_s wm_{t-1}})$$

รวมทุกขั้นตอน หลังจากลืมทุกความจำที่คิดว่าคงไม่ต้องการใช้อีกและบันทึกส่วนที่มีประโยชน์ของข้อมูลที่กำลังเข้ามา จะได้ updated long-term memory

$$\mathrm{ltm}_t = \mathrm{remember}_t \circ \mathrm{ltm}_{t-1} + \mathrm{save}_t \circ \mathrm{ltm'}_t$$

(∘ แสดงถึง element-wise multiplication)

ได้เวลาที่จะปรับแก้ working memory ต้องการที่จะเพ่งความสนใจไปที่ long-term memory ไปที่ข้อมูลที่จะมีประโยชน์ทันที ดังนั้นจะเรียนรู้ focus/attention vector

$$\mathrm{focus}_t = \sigma(\mathrm{W_f x}_t + \mathrm{U_f wm}_{t-1})$$

จะได้ working memory

$$\mathrm{wm}_t = \mathrm{focus}_t \circ \phi(\mathrm{ltm}_t)$$

ในอีกนัยหนึ่ง ให้ความสนใจกับสมาชิกที่มีค่าเป็น 1 และไม่สนใจสมาชิกที่เป็น 0

สรุปได้ว่า RNN ใช้สมการที่จะปรับ hidden state หรือ hidden memory แค่

$$\mathrm{h}_t = \phi(\mathrm{W x}_t + \mathrm{U h}_{t-1})$$

แต่ LSTM ใช้หลายสมการ

$$\mathrm{ltm}_t = \mathrm{remember}_t \circ \mathrm{ltm}_{t-1} + \mathrm{save}_t \circ \mathrm{ltm'}_t$$

$$\mathrm{wm}_t = \mathrm{focus}_t \circ \phi(\mathrm{ltm}_t)$$

ขณะที่แต่ละ memory หรือ attention sub-mechanism ก็แค่ส่วนย่อย

$$\mathrm{remember}_t = \sigma(\mathrm{W_r x}_t + \mathrm{U_r wm}_{t-1})$$

$$\mathrm{save'}_t = \sigma(\mathrm{W_s x}_t + \mathrm{U_s wm}_{t-1})$$

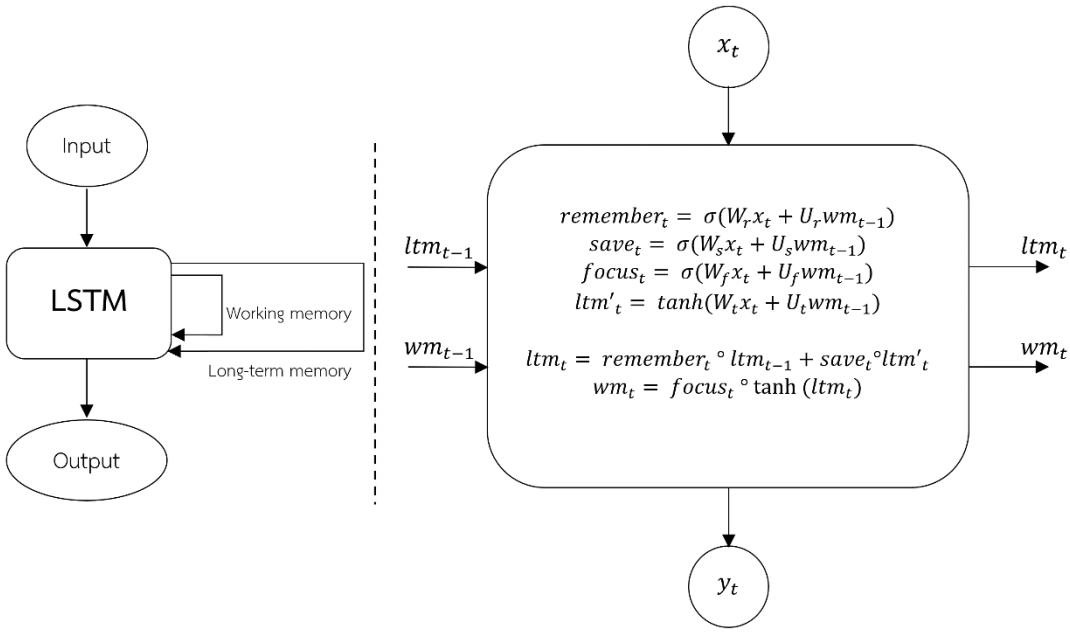
$$\mathrm{focus}_t = \sigma(\mathrm{W_f x}_t + \mathrm{U_f wm}_{t-1})$$

$$\mathrm{ltm'}_t = \phi(\mathrm{W_l x}_t + \mathrm{U_l wm}_{t-1})$$

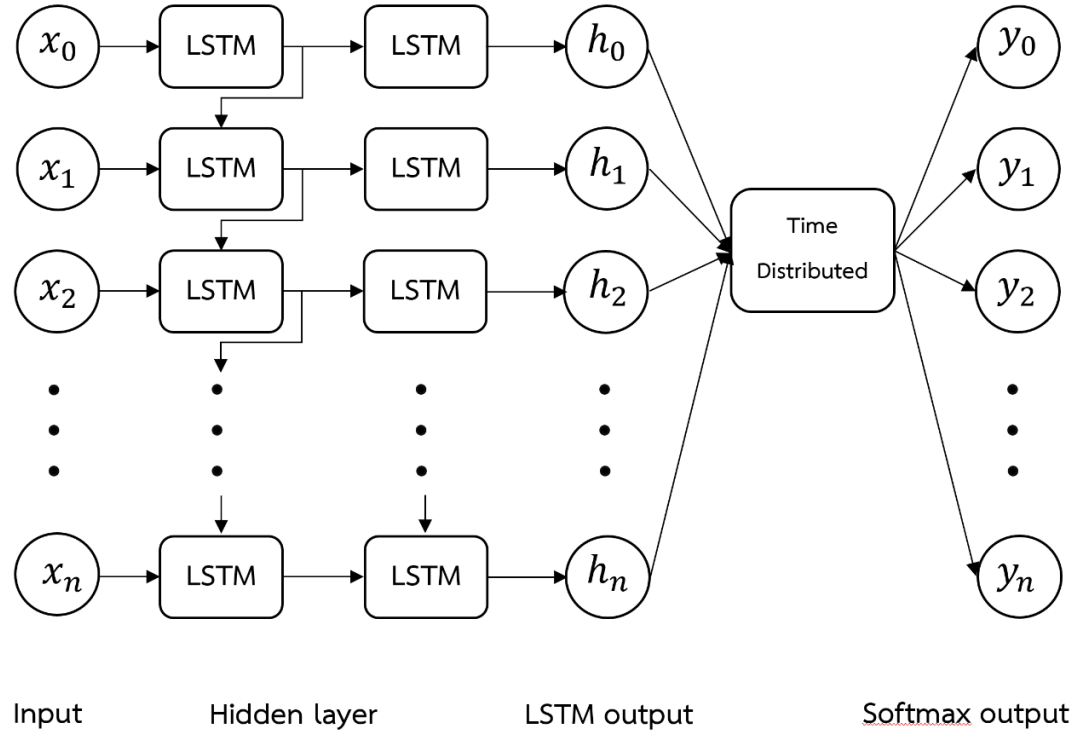
จากที่ตัวแปลงและนิยามต่างๆที่กล่าวมาที่ใช้ต่างกันซึ่งสามารถสลับสับเปลี่ยนกันได้ดังนี้

- long-term memory, ltm_t, เรียกว่า cell state , แสดงถึง c_t
- working memory, wm_t, เรียกว่า hidden state , แสดงถึง h_t
- remember vector, $\mathrm{remember}_t$, เรียกว่า forget gate (ถึงแม้ว่า 1 ใน forget gate ยังมีความหมายว่าเก็บรักษาความจำนั้น และ 0 ยังมีความหมายว่าลืม) แสดงถึง f_t
- save vector, save_t, เรียกว่า input gate (เพราะจากการที่ตัดสินว่าข้อมูลจำนวนเท่าใดที่จะถูกปล่อยเข้า cell state), แสดงถึง i_t

- focus vector, $\mathbf{focus_t}$, เรียกว่า output gate, แสดงถึง $\mathbf{o_t}$



ภาพที่ 2.10 LSTMs

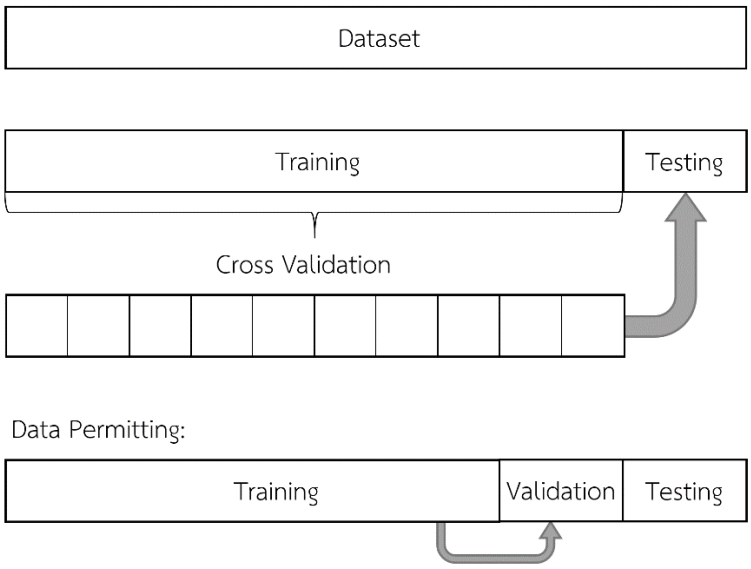


ภาพที่ 2.11 LSTM architecture

2.4 Cross Validation

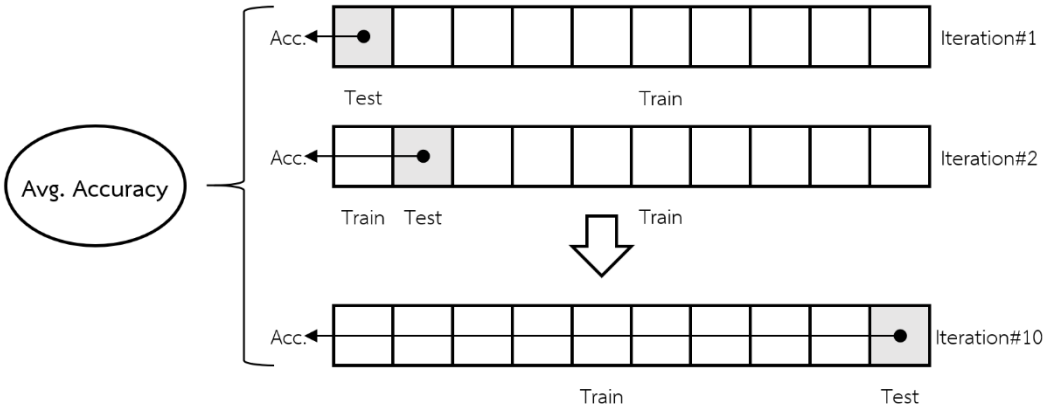
เป็นวิธีการเชิงสถิติที่ถูกใช้ประมาณทักษะความสามารถของโมเดลการเรียนรู้เครื่อง ซึ่งเป็นที่ใช้กันอย่างแพร่หลายในแวดวงของการเรียนรู้เครื่องเพื่อเปรียบเทียบและเลือกโมเดลสำหรับปัญหานั้นๆ เพราะ ความสะดวกในการเข้าใจ การดำเนินงาน และผลลัทธ์ ที่เที่ยงตรงกว่าวิธีอื่นๆ

การประเมินโมเดลด้วยการใช้ Dataset เพื่อปรับเข้ากับโมเดลและประมาณสมรรถภาพของโมเดล ปัญหาเกิดขึ้นเมื่อข้อมูลมีขนาดไม่ใหญ่มากแล้วต้องแบ่งข้อมูลเป็น Training dataset ปรับเข้ากับโมเดล และTesting dataset เพื่อประเมินโมเดล ข้อมูลต้องใหญ่และเป็นเอกลักษณ์พอที่จะบ่งชี้ปัญหานั้นได้



ภาพที่ 2.12 Cross Validation

K-Fold Cross-Validation คือ การที่ dataset ถูกแบ่งเป็นจำนวน K ส่วน และแต่ละส่วนนั้นจะถูกนำมาใช้เป็น testing set ณ จุดหนึ่ง อย่างเช่น จำนวน K=10 ข้อมูลถูกแบ่งเป็น 10 ส่วน ในการทำงานรอบแรก ส่วนแรกจะถูกใช้ทดสอบโมเดลเป็น testing set และส่วนที่เหลือเป็นส่วนที่ใช้ฝึกฝนโมเดลเป็น training set ในการทำงานรอบต่อมา ส่วนที่สองจะถูกน้ำมาเป็นตัวทดสอบโมเดล ส่วนแรกและส่วนที่เหลือถูกนำมาฝึกฝนโมเดล กระบวนการนี้จะถูกทำซ้ำจนกว่าแต่ละส่วนข้อมูลถูกใช้เป็น testing set



ภาพที่ 2.13 K-Fold Cross Validation

Random Seed

Random seed (seed state หรือ seed) เป็นจำนวนหรือเวกเตอร์ที่ถูกใช้สำหรับการเริ่มต้นการสุ่มตัวเลขเทียม (pseudorandom number generator) ซึ่งไม่จำเป็นต้องสุ่มเสมอไป เพราะ ธรรมชาติของขั้นตอนการสร้างจำนวนดั้งเดิมก็ไม่ได้สนใจ seed ค่าที่ได้จากอัลกอริทึมนั้นเป็นไปตามการการกระจายของความน่าจะเป็น (probability distribution) ในลักษณะการสุ่มเทียม ชุดตัวเลขจากการสุ่มจะถูกกำหนดโดย seed ดังนั้น การสุ่มเทียมจะถูกเริ่มอีกครั้งด้วย seed เดียวกัน จึงได้ชุดตัวเลขเดียวกัน

ในการวิจัยนี้ได้ใช้ฟังก์ชัน seed ในการกำหนดการสุ่ม เพื่อให้ผลการทดลองแต่ละครั้งสามารถวิเคราะห์และคาดเดา ความสำคัญของตัวแปรต่างๆในโมเดลได้

2.5 งานวิจัยที่เกี่ยวข้อง

งานวิจัยที่เกี่ยวกับโรคพาร์กินสันนั้นเกิดขึ้นมาสักพักแล้ว การเอาข้อมูลที่มีมาวิเคราะห์และวิธีที่ใช้นั้นแตกต่างกันอยู่ตามข้อมูลที่ได้มาใช้กัน การเลือกใช้การเรียนรู้เครื่องในการแก้ปัญหาจึงมีมาอย่างต่อเนื่อง เพื่อพัฒนาทั้งวงการแพทย์และวิศวกรรม

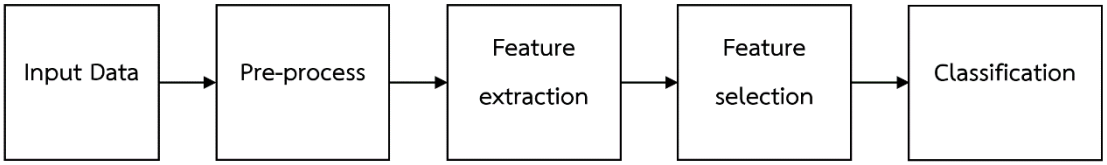
งานวิจัย [1] เป็นการจำแนกการวินิจฉัยโรคด้วย Computer-Aided Diagnosis (CADx) โดยการใช้ตัวจำแนก Rotation Forest (RF) กับการวินิจฉัยโรคพาร์กินสัน โรคทางเดินอาหาร และโรคหัวใจ งานวิจัย [2] ได้ทำการใช้การเรียนรู้เครื่องโดยเลือกใช้ LSTM ในการตรวจสอบพฤติกรรมหนึ่งของโรคหัวใจ Microsleep กับข้อมูล EEG, วิดิโอของหน้าผู้เข้าการรักษา และการติดตามการรักษา งานวิจัย [3] ใช้ RNNs และ LSTMs ในการศึกษาและปรับปรุงกระบวนการทำงานในส่วนงานทางการแพทย์ของ Intensive care unit (ICU) กับข้อมูลการเข้ารับการรักษา ข้อมูลเซ็นเซอร์ต่างๆ และผลการทดสอบจากห้องทดลองที่ถูกบันทึกไว้ในเครื่อง Electronic Health Record (EHR) งานวิจัย [4] การใช้ LSTM กับอุปกรณ์ที่สามารถส่วนใส่ในการวิเคราะห์คลื่น EEG กับการวิเคราะห์ข้อมูลแบบ real-time ด้วยการใช้ Multi-level binarized LSTM ซึ่งได้ผลการคาดเคลื่อน 0.01% งานวิจัย [5] ใช้ Time-aware LSTM (T-LSTM) ในการจัดการกับข้อมูลบันทึกการรักษาของผู้ป่วยเพื่อให้ผู้ป่วยสามารถทำการรักษาได้ตรงตามอาการในคลินิกการแพทย์เฉพาะทาง

LSTMs คือโครงข่ายประสาทเทียมที่พัฒนาขึ้นเพื่อแก้ไขปัญหา vanishing gradient ของ RNNs ซึ่งมีความยากในการเจอกับการทำงานของข้อมูลที่ขึ้นกับเวลา เมื่อการทำงานรอบลึกขึ้น ซึ่งในงานวิจัย [6] ได้แสดงถึงการแก้ปัญหานี้ กับปัญหาข้อมูลการโทรซึ่งเป็นปัญหา Language Modeling โดยมีการเปรียบเทียบให้เห็นว่า LSTMs นั้นทำได้ดีกว่า RNNs การใช้ข้อมูลแบบ Time-series เมื่อต้องทำการวิเคราะห์ข้อมูลประเภทนี้ การใช้โครงข่ายประสาทเทียมในการวิเคราะห์ข้อมูลมักเป็นทางเลือกลำดับต้นๆในการเลือกวิธีการแก้ปัญหาดังเช่นในงานวิจัย [7] Speech recognition หรือ Sleep stage classification การใช้ Deep learning สามารถให้ประสิทธิภาพสูงในการจำแนก งานวิจัย [8] ใช้ LSTM-RNN ในการจัดการกับข้อมูลเชิงภาษาบนเว็ป ในการจัดการลำดับคำและประโยคซึ่งจะถูกฝังอยู่ในรูปแบบ Semantic Vector งานวิจัย [9] ใช้ LSTM ในการวิเคราะห์ข้อมูลคลื่นเสียงจนาดใหญ่ Acoustic Modelling ซึ่งดีกว่าการใช้ DNNs ในการวิจัยนั้นการใช้ข้อมูลแบบ Time-series จำเป็นต้องทำให้ข้อมูลที่มีนั้นมีความสอดคล้องกันหากมีลำดับข้อมูลที่ไม่ตรงกัน งานวิจัย [10] ได้ใช้ Dynamic Time Warping (DTW) ในการจัดการข้อมูลให้สอดคล้องกันระหว่างแต่ละลำดับข้อมูล

การนำการเรียนรู้เครื่องมาใช้กับปัญหาการจำแนกประเภท หนึ่งในนั้นคือการใช้ LSTM มาใช้กับปัญหาการจำแนกเรียกว่า LSTM classification ใช้กับข้อมูลที่เป็นลักษณะเป็นลำดับต่อเนื่องกัน เป็น Sequence งานวิจัย [11] ใช้ LSTM กับปัญหา Pixel-level segmentation และ Classification กับข้อมูลรูปภาพที่เป็นลำดับรูปถาพเรียงลำดับกัน งานวิจัย [12] LSTM Fully Convolutional Networks กับการจำแนกข้อมูลลำดับเชิงเวลา โดยใช้ Attention LSTM Fully Convolutional Networks (ALSTM FCNs) งานวิจัย [13] ใช้จำแนกสเปกตรัมของความถี่ของข้อมูลสัญญาณจากเซ็นเซอร์ด้วยการแปลงข้อมูลจากอนาล็อกเป็นดิจิทัล Analog to Digital Converter (ADC) ด้วย LSTM modulation จึงสามารถจำแนกสัญญาณได้ งานวิจัย [14] การสกัดแบบแผนของข้อมูลสัญญาณด้วยการใช้การเรียนรู้เครื่องอย่าง Decision Tree และ Regression Tree ซึ่งเป็นวิธีที่ไม่ซับซ้อนและได้ผลที่ดี

การประยุกต์ใช้ข้อมูลของโรคพาร์กินสันสำหรับการวินิจฉัยโรคกับการใช้การเรียนรู้เครื่องเป็นที่สนใจมาระยะหนึ่งแล้ว งานวิจัย [15] การใช้ Fuzzy k-nearest neighbor (FKNN) ซึ่งเป็นการเรียนรู้เครื่องแบบหนึ่งการข้อมูลของโรคพาร์กินสัน โดยทำการเปรียบเทียบผลกับวิธีการ Support Vector Machines (SVM) โดยผลการทดลองได้ความแม่นยำถึง 96.07% โดยเป็นการใช้ FKNN ร่วมกับ K-fold cross validation งานวิจัย [16] เป็นการวินิจฉัยโรคพาร์กินสันด้วย Extreme Learning Machine (ELM) และ Kernel ELM (KELM) ร่วมกับ K-fold cross validation ด้วยการเลือกคีย์พารามิเตอร์เพื่อทำการปรับแก้โครงข่ายประสาทเทียม ด้วยการปรับจำนวนตัวประสาทเทียมและ Activation function ใน Hidden layer ของ ELM และ KELM งานวิจัย [17] ใช้สถาปัตยกรรมของโครงข่ายประสาทเทียมแบบวนซ้ำ ในการกำหนดยุทธศาสตร์ในการดำเนินการรักษาโรคพาร์กินสัน ในการวิจัยพัฒนาขั้นการดำเนินการรักษาด้วยการใช้ Dynamic Temporal Matching (DTM) ซึ่งได้รับแรงบันดาลใจจาก DTW ในงานวิจัยนี้ใช้ DTM กับโมเดลของ Personalized Linear Regression (LR), Personalized SVM, Multiclass LR, Multiclass SVM, K-nearest Neighbors (KNN) และ LSTM ผลการเปรียบเทียบพบว่า KNN ทำได้ดีที่สุดถึง 95% งานวิจัย [18] ใช้ข้อมูลของ Parkinson's Progression Markers Initiative (PPMI) dataset ซึ่งประกอบไปด้วย การใช้รูปภาพจากเครื่อง MRI เป็นการใช้ Convolutional Neural Networks (CNNs) ในการจำแนกรูปภาพจาก MRI ระหว่างภาพการสแกนสมองของผู้ที่ป่วยเป็นโรคพาร์กินสันและคนธรรมดา การใช้ LSTM สำหรับข้อมูล Unified Parkinson Disease Rating Scale (UPDRS) ซึ่งมีหลาย Features ในการ

วิจัยนี้ได้ทำการรวมทั้ง 2 อย่างเข้าด้วยกันเป็นโมเดล CNN/LSTM classification งานวิจัย [19] Deep Neural Networks ถูกนำมาใช้กับข้อมูลรูปภาพของภาพการแสกนสมองจากเครื่อง MRI และ DaT ด้วยการใช้ DNNs ของโมเดล CNN, CNN-RNN และ Bidirectional LSTM (B-LSTM) งานวิจัย [20] ใช้ข้อมูลจาก PPMI กับ LSTM เพื่อจำแนกอาการของโรคพาร์กินสัน ซึ่งโรคนี้นั้นมีจำเป็นต้องทำการจำแนกกลุ่มอาการย่อยเพื่อทำการรักษา งานวิจัย [21] ใช้ Artificial Neural Networks (ANNs) และ SVMs ในการพัฒนาระบบช่วยตัดสิน (Decision Support System) สำหรับโรคพาร์กินสัน งานวิจัย [22] ได้ทำการนิยามการวิเคราะห์ข้อมูลโรคพาร์กินสันของ Neuroimaging Modalities ซึ่งเป็นการศึกษางานวิจัยจำนวนมาก เพื่อนำเสนอรูปแบบกระบวนการทำงานของการวิจัยสำหรับการจำแนกของโรคพาร์กินสันที่มีทั้งของมูล EEG, MRI, SPECT, Gait Analysis, Motion และ Speech ขึ้นมาได้เป็นดังรูป



ภาพที่ 2.14 การทำงานของ Neuroimaging modalities เพื่อตรวจสอบ PD

งานวิจัย [23] การใช้สมาร์ทโฟนเข้ามาใช้ในการศึกษาโรคพาร์กินสันสามารถช่วยให้ข้อมูลสามารถเก็บได้เยอะและเข้าถึงได้สะดวกขึ้น ด้วยการศึกษาการนอนหลับของคนธรรมดาและผู้ป่วยโรคพาร์กินสัน Idiopathic REM sleep behavior disorder (iRBD) ด้วย 7 แบบทดสอบ เพื่อทำการประเมิน เสียง, การทรงตัว, การสัมผัสของนิ้วมือ, การตอบสนอง, การสั่นขณะพัก และ การสั่นขณะออกท่าทาง จากแอปพลิเคชันที่ถูกทำมาโดยเฉพาะ งานวิจัย [24] การใช้สมาร์ทโฟนและการเรียนรู้เครื่องเพื่อบ่งชี้ความรุนแรงของโรคพาร์กินสัน การใช้สมาร์ทโฟนในการเก็บข้อมูลกว่า 6 เดือนในการสกัด Features สำคัญกับเพื่อ mPDS จากวิธี Rank-based machine learning งานวิจัย [25] ได้ทำการเปรียบเทียบการเรียนรู้เครื่องแบบต่างๆประกอบไปด้วย Neural Networks, DMNeural, Regression และ Decision tree กับข้อมูลโรคพาร์กินสันที่เป็นข้อมูลเสียงของผู้ป่วยเพื่อทำการศึกษาและผลการศึกษาพบว่าโครงข่ายประสาทเทียมสามารถทำได้ดีที่สุด

บทที่ 3
วิธีการดำเนินงานวิจัย

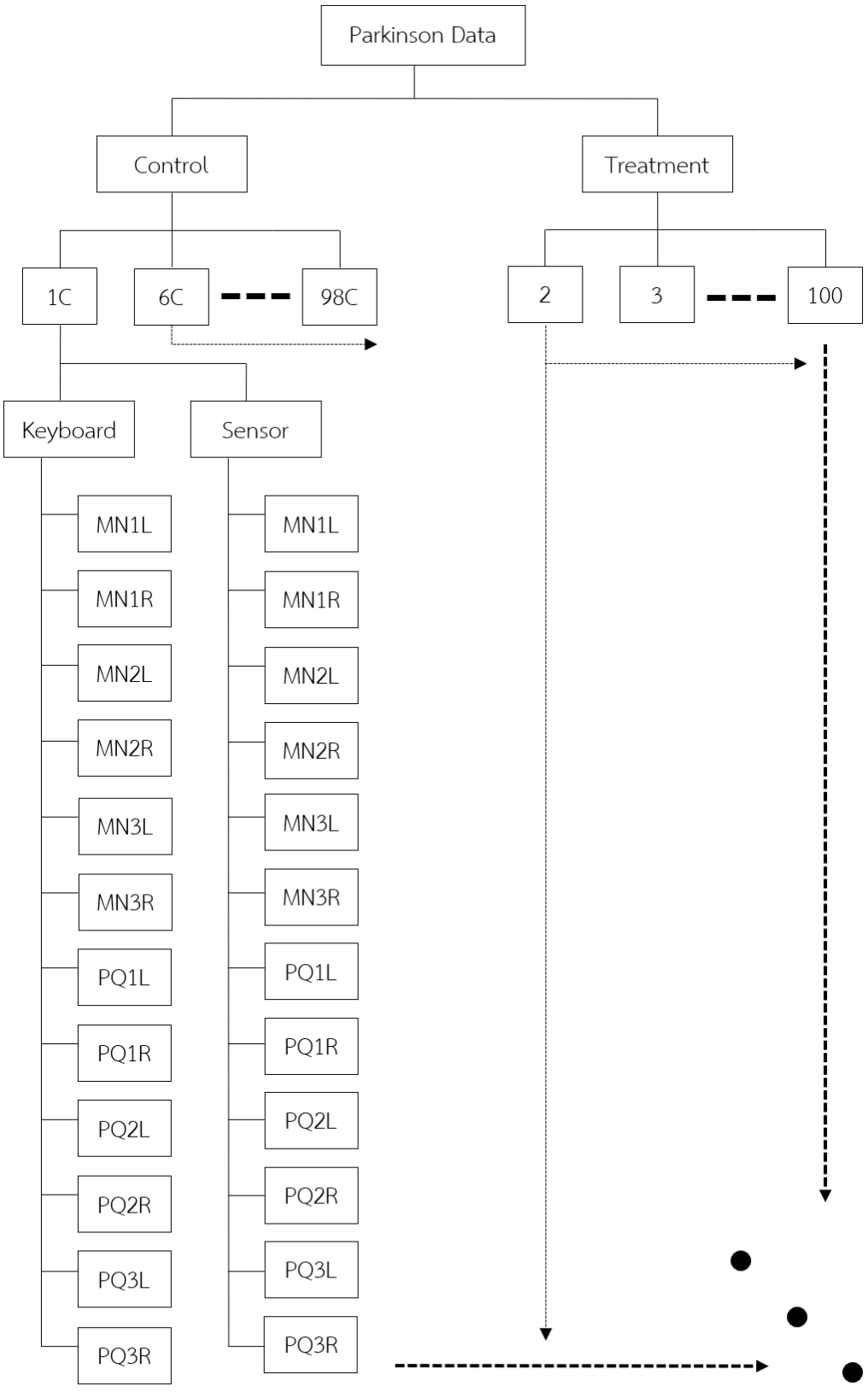
3.1 ข้อมูลที่ใช้ในการวิจัย (Data Overviewing)

ข้อมูลที่ใช้ในการวิจัยนี้มีอยู่ด้วยกัน 2 แบบคือ ข้อมูลจากคีย์บอร์ด และข้อมูลจากเซ็นเซอร์ ซึ่งได้มาจากการทำการทดสอบผู้ร่วมทดสอบ 100 คน แบ่งเป็นผู้ที่ป่วยเป็นโรคพาร์กินสัน 50 คน และผู้ที่ไม่ป่วยเป็นโรคพาร์กินสันหรือคนปกติ 50 คน โดยแบบทดสอบจะได้ข้อมูลมา 100 ชุด ของแต่ละคนและแยกจากกัน การทดสอบการเคลื่อนที่ข้อนิ้วมือผู้ร่วมทดสอบที่ละข้าง และทำการทดสอบทั้งนิ้วมือข้างซ้ายและข้างขวา

การทดสอบแบบใกล้ คือ การทดสอบการเคลื่อนที่ของนิ้วมือข้างเดียวใช้เวลาทดสอบ 15 วินาที กดคีย์บอร์ด 2 ตัวอักษร ในระยะใกล้ในที่นี้คือ ตัวอักษร “B”และ”M” โดยสวมเซ็นเซอร์ไว้ที่นิ้วที่กด 1 ตัวและนิ้วที่ไม่ได้กดอีก 1 ตัว และทำการทดสอบแบบนี้ 3 ครั้ง

การทดสอบแบบไกล คือ การทดสอบการเคลื่อนที่ขอนิ้วมือข้างเดียวใช้เวลาทดสอบ 30 วินาที กดคีย์บอร์ด 2 ตัวอักษร ในระยะไกลในที่นี้คือ ตัวอักษร “z”และ”/” โดยสวมเซ็นเซอร์ไว้ที่นิ้วที่กด 1 ตัวและนิ้วที่ไม่ได้กดอีก 1 ตัว และทำการทดสอบแบบนี้ อีก 3 ครั้ง

การทำการเก็บข้อมูลจากผู้ร่วมทดสอบนี้ได้ด้วยการเก็บข้อมูลลงบนคอมพิวเตอร์ผ่านตัวควบคุม ซึ่งมีหน้าที่เก็บข้อมูลทั้งจากคีย์บอร์ดและเซ็นเซอร์ในเวลาเดียวกัน แล้วทำการเก็บข้อมูลสำหรับแต่ละคน ดังนั้นในหนึ่งโฟลเดอร์จะประกอบไปด้วยข้อมูลจากคีย์บอร์ดและเซ็นเซอร์ อุปกรณ์เป็นไปดังรูปที่ 3 ในโฟลเดอร์จะได้ข้อมูลดิบเก็บไว้ในนั้น และเมื่อผู้ทดสอบทำแบบทดสอบครบทุกคนจะได้ข้อมูลทั้งหมดเก็บในโฟลเดอร์เดียวกันอย่างอัตโนมัติ ซึ่งได้แสดงรูปแบบข้อมูลที่ได้มาในในรูปที่ 12 โดยข้อมูลนั้นจะเก็บในรูปแบบไฟล์นามสกุล .csv ซึ่งเป็นที่นิยมใช้กันอย่างแพร่หลายอยู่แล้ว ทั้งยังสะดวกในการนำไปประยุกต์ใช้งานอื่นๆต่อไปได้อีก



ภาพที่ 3.1 ข้อมูลในโฟล์เดอร์

ตารางที่ 3.1 ตัวอย่างข้อมูลของคีย์บอร์ด

Time	key
0.851	m
1.243	b
1. 602	m
1.945	b
2.304	m
2.647	b
2.99	m
3.318	b
3.661	m
4.4	b
4.348	m
4.66	b
5.3	m
5.377	b
5.72	m
6.64	b
6.438	m
6.781	b
7.14	m
7.514	b

ตารางที่ 3.2 ตัวอย่างข้อมูลของเซ็นเซอร์วัดความเร่งและมุม

Time	Ax	Ay	Az	Gx	Gy	Gz	Ax2	Ay2	Az2	Gx2	Gy2	Gz2
0.05	-3540	4056	14024	177	-20	131	-3268	7432	16188	500	-220	9
0.08	-3624	4116	13888	164	2	115	-3432	7264	16244	482	-357	8
0.11	-3476	4124	13776	250	371	35	-3284	7576	16548	381	-872	2
0.14	-3548	4124	14072	161	-211	126	-3360	7404	16400	451	-226	7
0.18	-3596	4232	13884	166	35	133	-3240	7504	16348	496	-254	9
0.21	-3480	4172	14048	188	177	41	-3256	7432	16304	488	-106	-3
0.24	-3724	4060	13864	159	127	79	-3440	7340	16376	500	-147	2
0.28	-3516	4160	13940	172	237	58	-3396	7268	16204	500	-85	1
0.31	-3736	4116	13988	221	122	66	-3424	7396	16176	486	-170	4
0.34	-3412	4100	13964	179	-151	178	-3148	7384	16288	507	-440	17
0.38	-3540	4240	14028	238	333	29	-3296	7416	16204	580	-2	-1
0.41	-3628	4068	13876	141	-9	129	-3460	7360	16272	449	-232	7
0.45	-3388	4136	13984	183	-142	240	-3196	7428	16348	531	-249	9
0.48	-3556	4164	14072	282	265	40	-3316	7484	16392	547	-89	-2
0.51	-3480	4008	13988	240	61	82	-2776	7704	16720	586	-27	-2
0.55	-3600	4304	14264	289	321	133	-4012	6996	16192	639	-314	15
0.58	-3444	4232	14248	341	312	260	-3216	4004	12876	-2444	-124	22
0.62	-3816	4152	14152	454	1080	-84	-3456	7736	16432	479	391	25
0.65	-3304	4284	13880	814	1324	-36	-2916	7556	16484	-40	409	48
0.68	-3904	4364	14208	918	1674	103	-3944	7276	16476	-1102	-13	32

3.2 การจัดการข้อมูล (Data processing)

การทำความสะอาดข้อมูล (Data cleansing) ข้อมูลที่ได้มามีจำนวนไม่มากนัก แต่มีความผิดพลาดอยู่จึงจำเป็นต้องทำการจัดการให้ถูกต้องจึงจะนำไปใช้ประโยชน์ต่อได้อย่างไม่เกิดความผิดพลาดสับสน หากข้อมูลที่มีความผิดพลาดนำไปใช้ในโครงข่ายประสาทเทียม จะทำให้ผลลัพธ์การดำเนินงานแย่กว่าที่ควรจะเป็น ซึ่งความผิดปกติข้อข้อมูลมีดังนี้

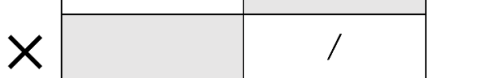
1) ลำดับไม่ถูกต้อง คือลำดับเวลาต้องเพิ่มขึ้นตลอด เมื่อลำดับเวลาผิดจึงต้องจัดเรียงใหม่

Time	Key
0.153	b
0.953	m
1.725	m
0.573	b
0.573	m
0.79	b
2.469	b

ภาพที่ 3.2 ตัวอย่างของข้อมูลลำดับไม่ถูกต้อง

2) ข้อมูลหาย คือข้อมูลลำดับเวลาหายไป และข้อมูลคีย์บอร์ดหายไปแต่ในกรณีข้อมูลคีย์นั้นพบว่าเป็นการกดเว้นวรรค(space bar)

Time	Key
0.59	/
1.885	Z
2.354	/
3.106	
	/
5.693	/



ภาพที่ 3.3 ตัวอย่างของข้อมูลที่หาย

3) ข้อมูลทับซ้อนกัน คือข้อมูลของการทดสอบแบบใก้ลและไกลมาต่อกันหรือแทรกขึ้นมา

Time	Key	
12.006	m	
12.98	b	
13.453	m	
14.4	b	
10.54	z	✕
11.301	/	✕
12.068	/	✕

ภาพที่ 3.4 ตัวอย่างของข้อมูลทับซ้อนกัน

4) ข้อมูลไม่ต่อเนื่องกัน ข้อมูลระหว่างคีย์บอร์ดและเซ็นเซอร์ ลำดับไม่เท่ากันและเวลาไม่ตรงกัน จึงจับมาต่อกันเลยไม่ได้ ต้องทำการแก้ไขก่อน

1	Time	key		Time	Ax	Ay	Az	Gx	Gy	...
2	0.851	m		0.05	-3540	4056	14024	177	-20	...
3	1.243	b		0.08	-3624	4116	13888	164	2	...
4	1.602	m		0.11	-3476	4124	13776	250	371	...
5	1.945	b		0.14	-3548	4124	14072	161	-211	...
6	2.304	m		0.18	-3596	4232	13884	166	35	...
7	2.647	b		0.21	-3480	4172	14048	188	177	...
8	2.99	m		0.24	-3724	4060	13864	159	127	...
9	3.318	b		0.28	-3516	4160	13940	172	237	...
10	3.661	m		0.31	-3736	4116	13988	221	122	...
...	...	...		...	...	...	...	...	...	...

ภาพที่ 3.5 ตัวอย่างของข้อมูลไม่ต่อเนื่องกัน

เมื่อทำการแก้ปัญหาทั้งหมดแล้ว ก็นำข้อมูลมาวิเคราะห์เบื้องต้นเป็นข้อมูลเชิงสถิติ การนำข้อมูลมาวิเคราะห์เบื้องต้นเชิงสถิติทำให้สามารถเข้าใจลักษณะ แนวโน้ม และสามารถกำหนดทิศทางในการนำข้อมูลไปใช้ได้สะดวกขึ้น

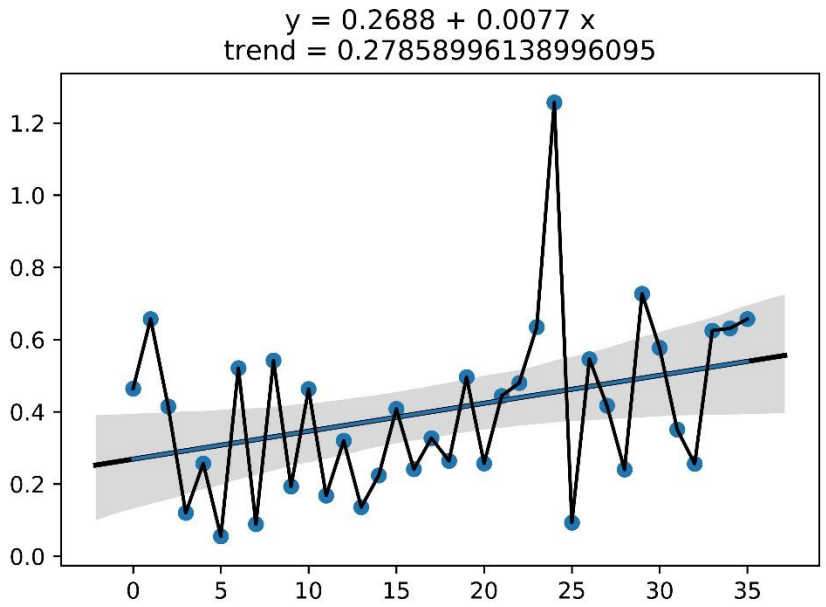
ตารางที่ 3.3 ข้อมูลสถิติจากข้อมูลดิบ

Result	far_ctrl	far_treat	far_treat_hdom	near_ctrl	near_treat	near_treat_hdom
mean difference	0.559464274	0.73818	0.750641	0.396426	0.445987	0.44779581
std difference	0.131547289	0.199998	0.202275	0.084454	0.190223	0.217287216
Var difference	0.036547837	0.063239	0.060914	0.010302	0.057279	0.0707939
mean trend	-0.004580016	0.008825	0.012798	0.006592	0.028921	0.020493553
sum trend	-1.181644025	2.603239	1.881256	1.700642	8.763194	3.135513625
burst	0.027131783	0.091525	0.156463	0.224806	0.864686	1.366013072
std burst	0.162783157	0.629569	0.873512	0.651133	2.669085	3.535071714
repeat	0.798449612	1.054237	1.170068	2.414729	3.980198	5.137254902
std repeat	1.121976057	1.824933	2.133793	4.359624	5.514545	6.664970114
2keys at once	0	0	0	0.089147	0.415842	0.529411765
std 2k	0	0	0	0.335625	1.03515	1.147415996
key1	29.00775194	22.47119	22.06122	23.18605	22.51815	23.1503268
key2	27.80620155	21.54915	21.17007	22.09302	19.36634	19.02614379
key3	0.879844961	0.711864	0.802721	0.217054	0.825083	1.052287582
key4	0.166666667	0.213559	0.258503	0.011628	0.09571	0.14379085
error	1.808681672	2.058979	2.395945	0.502513	2.151118	2.757685353
std err	2.973807776	4.431773	4.274251	1.880007	5.665992	5.675309985
velocity	1.89379845	1.467345	1.441043	3.018605	2.792299	2.811764706
std velocity	0.448564189	0.37087	0.349411	1.301374	1.265705	1.448579153

เพื่อทำความเข้าใจตารางที่ 3.3 ตัวแปลที่มีชื่อ “ctrl” หมายถึง คนปกติที่เข้าร่วมทดสอบ เรียกว่า “control” และชื่อ “treat” หมายถึง ผู้ทดสอบที่เป็นพาร์กินสัน เรียกว่า “treat”

มีสิ่งหนึ่งที่ต้องทราบ คือ ทุกคนที่ป่วยจะมีมือข้างที่มีอาการมากกว่าอีกข้างเรียกมือนั้นว่า “dominant” ซึ่งในที่นี้ใช้ “hdom” คือ dominant hand นั่นเอง ดังนั้นจากตารางที่ 3 จะพบว่าข้อมูลสถิติของ far_treat_hdom และ near_treat_hdom จะพบว่ามีค่าสูงกว่าตัวอื่นๆ

วิเคราะห์ตารางที่ 3 พบว่าผู้ป่วยจะมีค่าต่างๆสูงกว่ากลุ่มควบคุม ยกเว้นจำนวนครั้งการกดคีย์บอร์ด ซึ่งก็คือค่าต่างๆตั้งแต่ key1 เรื่อยลงมาจนถึงความเร็วในการกดคีย์บอร์ดอย่าง velocity ซึ่งผู้ป่วยจะประสบปัญหาในการเคลื่นที่ของมือและนิ้วมืออยู่แล้วจึงไม่สามารถทำได้มากเท่ากลุ่มควบคุม ซึ่งเป็นปกติ



ภาพที่ 3.6 ตัวอย่างของกราฟของข้อมูลผู้ทดสอบที่มีอาการของโรค

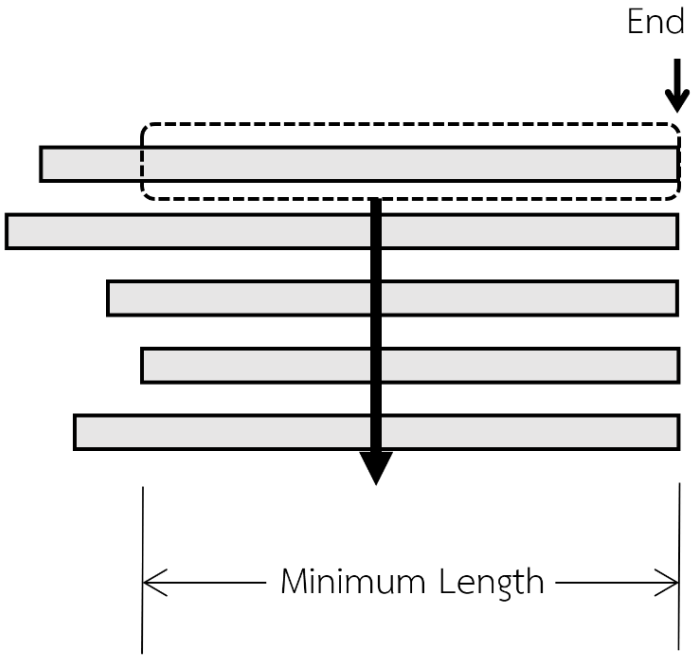
เมื่อพอเข้าใจความหมายของตัวแปรและค่าต่างๆในตารางที่ 3.3 จะพบว่าผู้ป่วยจะมีแนวโน้มในการทำแบบทดสอบได้แย่กว่าคนปกติ เหตุการณ์ที่มักจะเกิดขึ้นกับผู้ป่วยที่เข้าร่วมทดสอบนี้ คือ การกดซ้ำ (repeat) การกดค้าง (burst) และการ 2 คีย์พร้อมกัน (2k at once) จึงปรากฏให้เห็นมากกว่าในผู้ป่วย ทั้งนี้เมื่อสังเกตแนวโน้นการทำแบบทดสอบจากที่มีอยู่ตารางที่ 3.3 ตัวแปรแนวโน้ม (trend) พบว่าของกลุ่มควบคุมนั้นกลับมีการติดลบ เนื่องจากว่าคนปกติทั่วไปเมื่อทำแบบซ้ำๆเดิมๆที่

ไม่ซับซ้อนอะไรแบบนี้จะเกิดการพัฒนา คือ ทำได้เร็วขึ้นและดีขึ้น แต่ในทางผู้ป่วยจะมีแนวโน้มที่แย่สวนทางกัน และจากภาพที่ 3.6 ซึ่งเป็นการแสดงให้เห็นว่าผู้ป่วยคนหนึ่งทำแบบทดสอบได้แย่ลงอย่างชัดเจน และอาการเหล่านี้จะปรากฏมากขึ้นเมื่อเวลาผ่านไป หรือช่วงท้ายๆของแบบทดสอบ

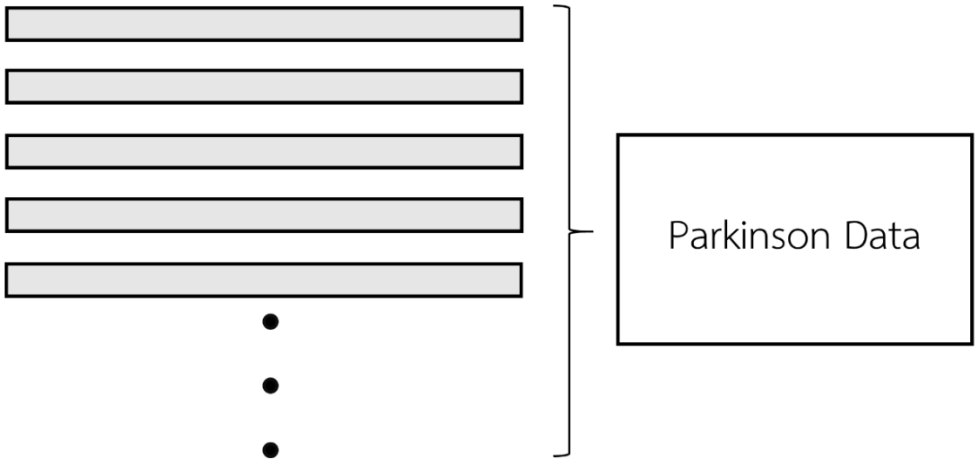
หลังจากทำความเข้าใจลักษณะต่างๆของข้อมูลนี้แล้ว จึงสามารถหาแนวทางต่างๆไปใช้ประโยชน์ต่อได้สะดวกขึ้น ซึ่งในการวิจัยนี้คือ การนำข้อมูลชุดนี้ไปใช้กับโครงข่ายประสาทเทียมแบบ LSTMs นั่นเอง

การแปลงข้อมูล (Data Transforming) หลังจากจัดการกับข้อมูล (Data Cleansing) เสร็จแล้ว จึงนำข้อมูลมาจัดระเบียบให้อยู่ในรูปแบบเดียวกัน เนื่องจากข้อมูลที่ได้มามีความยาวไม่เท่ากัน เพื่อทำให้สามารถนำไปใช้และประสิทธิภาพในการทำงานของโครงข่ายประสาทเทียบแบบ LSTMs จึ่งจำเป็นต้องทำให้ข้อมูลมีความยาวเท่ากัน

เมื่อทราบอยู่แล้วว่าข้อมูลที่มีนัยสำคัญอยู่ในช่วงท้ายของแต่ละการทดสอบ และเพื่อให้ได้ข้อมูลสมบูรณ์ที่สุดจึงหาความยาวต่ำสุดของข้อมูลทั้งหมด แล้วใช้ส่วนนั้นเป็นความยาวของข้อมูลที่ใช้กับโครงข่ายประสาทเทียมแบบ LSTMs



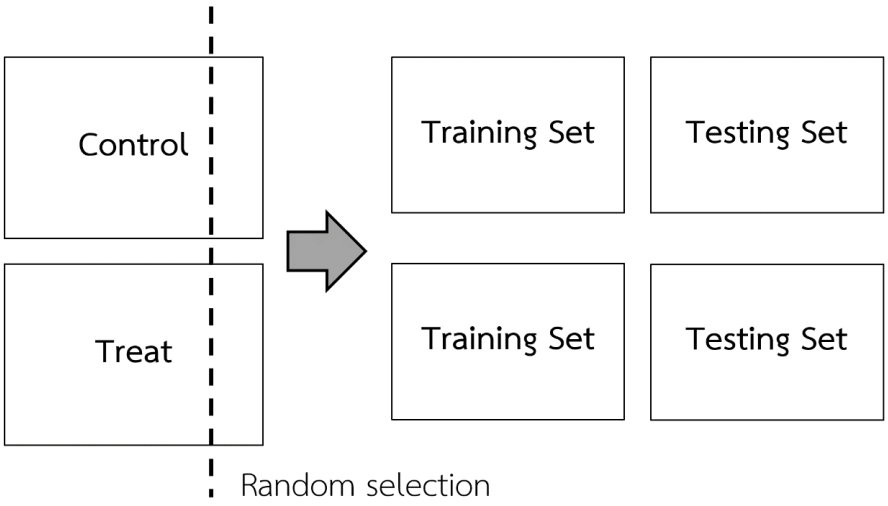
ภาพที่ 3.7 การกำหนดความยาวของข้อมูล



ภาพที่ 3.8 ข้อมูลที่ถูกกำหนด

จากภาพที่ 3.7 ข้อมูลของทุกคนและทุกแบบทดสอบจะถูกแปลงให้เหมือนกัน จนได้เป็นข้อมูลในภาพที่ 3.8 โดยในการวิจัยนี้ได้ทำการแปลงเป็นหลายแบบเพื่อการใช้งานที่หลากหลายโดยจะแสดงให้ทราบเพิ่มเติมในขั้นตอนต่อไปเพื่อความเข้าใจในแต่ละการทดสอบ

การแบ่งข้อมูล (Data splitting) ในงานวิจัยนี้ทำการแบ่งข้อมูลแบบสุ่ม สำหรับการฝึกโครงข่ายประสาทเทียม (Training set) และสำหรับการทดสอบ (Testing set) ไว้ที่
Train set 80% : Test set 20% หรือจากการข้อมูลที่ได้มา 100 คน คือสำหรับกลุ่มควบคุม (Control) ที่ 40:10 และผู้ป่วย (Treat) ที่ 40:10



ภาพที่ 3.9 Data Splitting

ข้อมูลในการวิจัยมีจำกัด จึงได้ใช้วิธี K-Fold Cross Validation เพื่อประสิทธิภาพในการเรียนรู้ของโมเดลสูงที่สุดเท่าที่จะเป็นไปได้ เนื่องจากการแบ่งข้อมูลแบบทั่วไปจะพบปัญหาในการเรียนรู้ของเครื่องที่ไม่ทั่วถึงทุกตัวอย่างการเรียนรู้จึงไม่ได้ประสิทธิภาพเท่าที่ควร

การวิจัยนี้จะดำเนินการทั้ง 2 แบบการแบ่งข้อมูลเพื่อแสดงให้เห็นความแตกต่างของผลลัพธ์ โดยจะใช้โมเดลเดียวกันและข้อมูลเดียวกัน หลังจากนั้นจะทำการหาค่าประสิทธิภาพสูงสุดจากการปรับค่าตัวแปรต่างๆของโมเดล ดังนั้นจึ่งแบ่งได้เป็น 3 ช่วงหลักๆคือ

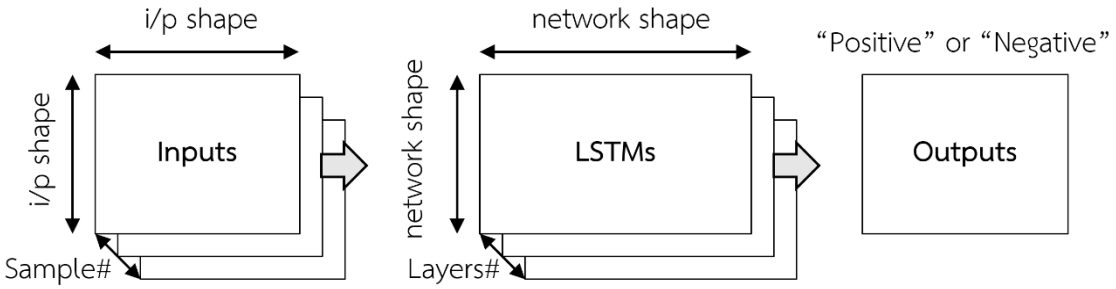
- การดำเนินการกับการแบ่งข้อมูลพื้นฐาน
- การดำเนินการกับ K-Fold Cross Validation
- การปรับเพื่อหาประสิทธิภาพสูงสุดกับ K-Fold Cross Validation

3.3 กำหนดโมเดล (Model Creating)

ข้อมูลที่ใช้กับระบบโครงข่ายโครงข่ายประสาทเทียมแบบ LSTMs มีหลายแบบเพื่อแสดงให้เห็นการประยุกต์ใช้การเรียนรู้ของเครื่องกับทางการแพทย์ ว่าสามารถทำได้และทำได้หลายแบบ เนื่องจากข้อมูลนั้นมีความแตกต่างกันระหว่างข้อมูลจากคีย์บอร์ดและข้อมูลจากเซ็นเซอร์และที่สำคัญความยาวไม่เท่ากัน แต่ละวิธีจึงมีมาตรฐานความยาวต่างกัน

การวิจัยนี้เป็น Binary Sequence Classification โดยใช้ LSTMs

โครงข่ายระบบประสาทเทียม LSTMs ในงานวิจัยนี้จำเป็นต้องกำหนดขนาดรูปร่างและจำนวนการวนซ้ำ และทราบอยู่แล้วว่าต้องการข้อมูลขาออกเป็นการทำนายผลที่เป็น Positive หรือ Negative



ภาพที่ 3.10 ลักษณะของโครงข่ายระบบประสาทเทียมในการวิจัย

การวิจัยนี้ใช้ Multivariate multi-step LSTM model จากข้อมูลขาออกจึงลงลึกจำแนกได้เป็นแบบ Multiple input ซึ่งตัวแปรต่างๆมีความสำคัญในการตั้งค่าโครงข่ายประสาทเทียม LSTM ในงานวิจัยนี้ที่ซึ่งใช้เครื่องมือของ Keras เป็นที่นิยมใช้กันอย่างแพร่หลาย

ตัวแปร epochs, batch size และ neurons จะมีผลต่อประสิทธิภาพและประสิทธิผลของการวิจัยนี้อยู่พอสมควร ปรากฏการณ์ overfitting สามารถปรับปรุงแก้ไขได้จากพารามิเตอร์เหล่านี้ ซึ่งเป็นตัวกำหนดโครงสร้างของโมเดล

- Epochs จำนวนรอบที่ป้อนเข้า
- Batch size ขนาดของ batch ในการป้อนแต่ละรอบ
- Neurons จำนวน neurons ของแต่ละ hidden layer

ตัวแปรหลักๆที่สนใจสำหรับการปรับปรุงเพื่อหาประสิทธิภาพของโมเดล

- Activation Function

- Optimizer
- Regression Loss function
- Dropout และ/หรือ Recurrent dropout

สามารถกำหนดจำนวนชั้นของ Hidden layers ตามความเหมาะสม ทั้งยังกำหนดรูปแบบได้เช่นกัน เมื่อได้ model แล้ว จึงเหลือเพียงการกำหนดรูปแบบข้อมูลขาเข้า สามารถนำข้อมูลที่จัดการเสร็จแล้วมาปรับใช้

กำหนด model โดยเริ่มต้นได้จากปัญหาและข้อมูล ข้อมูลขาออกจะถูกกำหนดโดยปัญหาคือ n_step_out จากปัญหา Binary Classification นี้ได้เท่ากับ 1 และใช้ activation function คือ sigmoid ลำดับขั้นตอนการดำเนินการสำหรับการเรียนรู้เครื่อง หลักๆประกอบไปด้วย 5 ขั้นตอนคือ

1. Define Network :

Neural Network ใน Keras จะถูกกำหนดเป็นอนุกรมของ layers ด้วยการกำหนด sequential class ใน layer#1 จะกำหนด input shape 3D หากจะซ้อน LSTM layer จะต้องให้ layer ก่อนหน้า output เป็น sequence การเลือก activation function สำหรับ output มีความสำคัญที่สุดสำหรับแต่ละปัญหาซึ่งโดยทั่วไปแต่ละปัญหาใช้ activation function

- Regression ใช้ linear และจำนวน output neurons เท่ากับ outputs ที่ต้องการ
- Binary Classification (2 class) ใช้ sigmoid และจำนวน 1 neuron output
- Multiclass Classification (>2 class) ใช้ softmax และจำนวน output neurons เท่ากับจำนวน class

2. Compile Network :

Compilation เป็นการแปลง layer sequence ที่ถูกกำหนดก่อนหน้านี้เป็นอนุกรมของ matrix transform ในรูปแบบสำหรับการจะถูกดำเนินการใน GPU หรือ CPU

Optimization function ยอดนิยม

- Stochastic Gradient Descent ซึ่งจะต้องการ tuning ของ learning rate และ momentum
- ADAM ซึ่งต้องการ tuning ของ learning rate
- RMSprop ซึ่งต้องการ tuning ของ learning rate

Loss function มาตรฐาน

- Regression ใช้ mean square error
- Binary Classification (2 class) ใช้ logarithmic loss ถูกเรียกว่า cross entropy หรือ binary crossentropy
- Multiclass Classification (>2 class) ใช้ multiclass logarithmic loss หรือ categorical crossentropy

3. Fit Network :

หลังจากถูก complied สามารถปรับ weight บน training dataset บนรูปแบบ input, output = X, y และสามารถปรับ epoch และ batch ได้ ใช้ข้อมูล training X และ train y

4. Evaluate Network :

หลังจากถูกเรียนรู้แล้วจะทำการประเมินโดยทดสอบด้วยข้อมูลที่ถูกแยกไว้คือ testing X และ testing y

5. Make Predictions :

ทำการทดสอบทำนายผลกับข้อมูลได้เป็นรูปแบบ format ของ output layer

- Regression อาจจะได้เป็น format ตรงจาก linear activation
- Binary Classification (2 class) จะได้ array ของความน่าจะเป็น ซึ่งถูกปัดเป็น 0 หรือ 1
- Multiclass Classification (>2 class) จะได้ array ของความน่าจะเป็น

ต่อจากนี้จะแสดงอัลกอริทึมสำหรับการใช้งานทั่วไปของการเรียนรู้ของเครื่องและตัวอย่างคำสั่งเบื้องต้นของภาษา Python เพื่อให้เข้าใจโครงสร้างและรูปแบบการทำงานของการเรียนรู้ของเครื่องสำหรับงานวิจัยนี้

```
# 1. define model
model = Sequential()
model.add(LSTM(n_neurons, input_shape= (n_steps,n_features) ,
      return_sequences =True))
model.add(LSTM(n_neurons), activation= 'relu', return_sequences =True)
model.add(Dense(n_step_out, activation='sigmoid'))
# 2. compile model
model.compile(optimizer='sgd', loss='binarry_crossentropy')
# 3. fit model
model.fit(X_train, y_train, epochs=n_epochs, batch=n_batch, verbose=0)
# 4. evaluate  model
loss, accuracy = model.evaluate(X_test, y_test, verbose=0)
# 5. make predictions
yhat = model.predict(X_test)
# Save model to single file
```

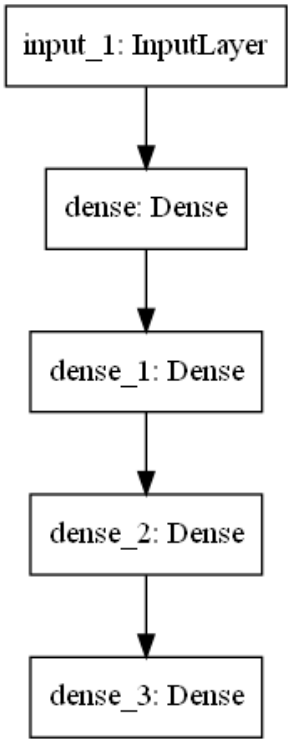
สามารถทดสอบการทำนายผลของ model ได้ด้วยการนำข้อมูลมาทำสอบความสามารถในการจำแนกได้ ด้วยการนำ input ใหม่มาแปลงเป็น 3 มิติ แล้วลองดูผลของการทำนายใหม่

```
# demonstrate prediction
x_input = array(...example...)
x_input = x_input.reshape((n_samples, n_steps, n_features))
yhat = model.predict(x_input, verbose=0)
print(yhat)
```

สามารถทำการบันทึก model สำหรับนำไปเรียกใช้งานอีกครั้ง ใน script ใหม่สามารถดาวน์โหลดโมเดลที่ถูกฝึกมาแล้ว มาทดสอบได้อีก กับข้อมูลใหม่ที่มีรูปแบบ X, y (แบบของ supervised learning)

```
from keras.models import load_model
# load model from single file
model = load_model('lstm_model.h5')
# make predictions
ypredict = model.predict(X, verbose=0)
print(ypredict)
```

โมเดลสำหรับโครงข่ายประสาทเทียมแบบโหนดประสาทแบบทั่วไป ที่เชื่อมต่อกันครบทุกโหนดทั้ง Hidden layers และ Output layer ซึ่งภายในโหนดไม่มีได้มีการทำงานซับซ้อน อย่างเช่นโหนดของ LSTM ที่จะทำงานซับซ้อนกว่าแม้จะมีจำนวนชั้นเท่ากัน



ภาพที่ 3.11 Simple fully-connected neuron model

ขั้นตอนวิธีสำหรับ K-Fold Cross-Validation

ขั้นตอนวิธีนี้สามารถทำให้การเรียนรู้ของเครื่องสามารถทำได้หลายโมเดลพร้อมกันในการใช้ข้อมูลขาเข้าของโมเดลนั้นเป็นแบบเดี่ยวกันได้ และสามารถเปรียบเทียบผลของโมเดลได้

```
Fix random seed for reproducibility
Load dataset
Split into input (X) and output (Y) variables
Reshape input into 3D format : (samples, steps, features)
Define Stratified K-Fold cross validation test harness : K=10
Create variables for collecting accuracy data from models
For train, test in kfold.split(X, Y):
    Create a model#1
    Compile model#1
    Fit the model#1
    Evaluate the model#1
    Print the accuracy percentage of the model
    Add the performance data into the variable

    Create a model#2
    Compile model#2
    Fit the model#2
    Evaluate the model#2
    Print the accuracy percentage of the model
    Add the performance data into the variable
Print the overall performance from the variables
```

3.4 ดำเนินการใช้ข้อมูลกับโครงข่ายประสาทแบบเทียม (Data Using)

ข้อมูลขาเข้าของ LSTM input layer ต้องเป็นสามมิติ (3D) คือ Samples, time steps และ features ซึ่งจะถูกกำหนดไว้ที่ input shape ไม่ว่าข้อมูลขาเข้าจะเป็นอย่างไรสามารใช้ reshape() ในการทำให้เป็นข้อมูลสามมิติได้เสมอ

ตารางที่ 3.4 Input shape

Feature 1	Feature 2	...	Feature n
Time_step 1	...	...	...
Time_step 2	...	...	...
...	...	...	...
Time_step n	...	...	...

จากตารางที่ 6 input shape ของ dataset นั้นจะกำหนดค่าจำนวน time steps, จำนวน features และ จำนวน samples เพื่อใช้ใน reshape(samples, time steps, features)

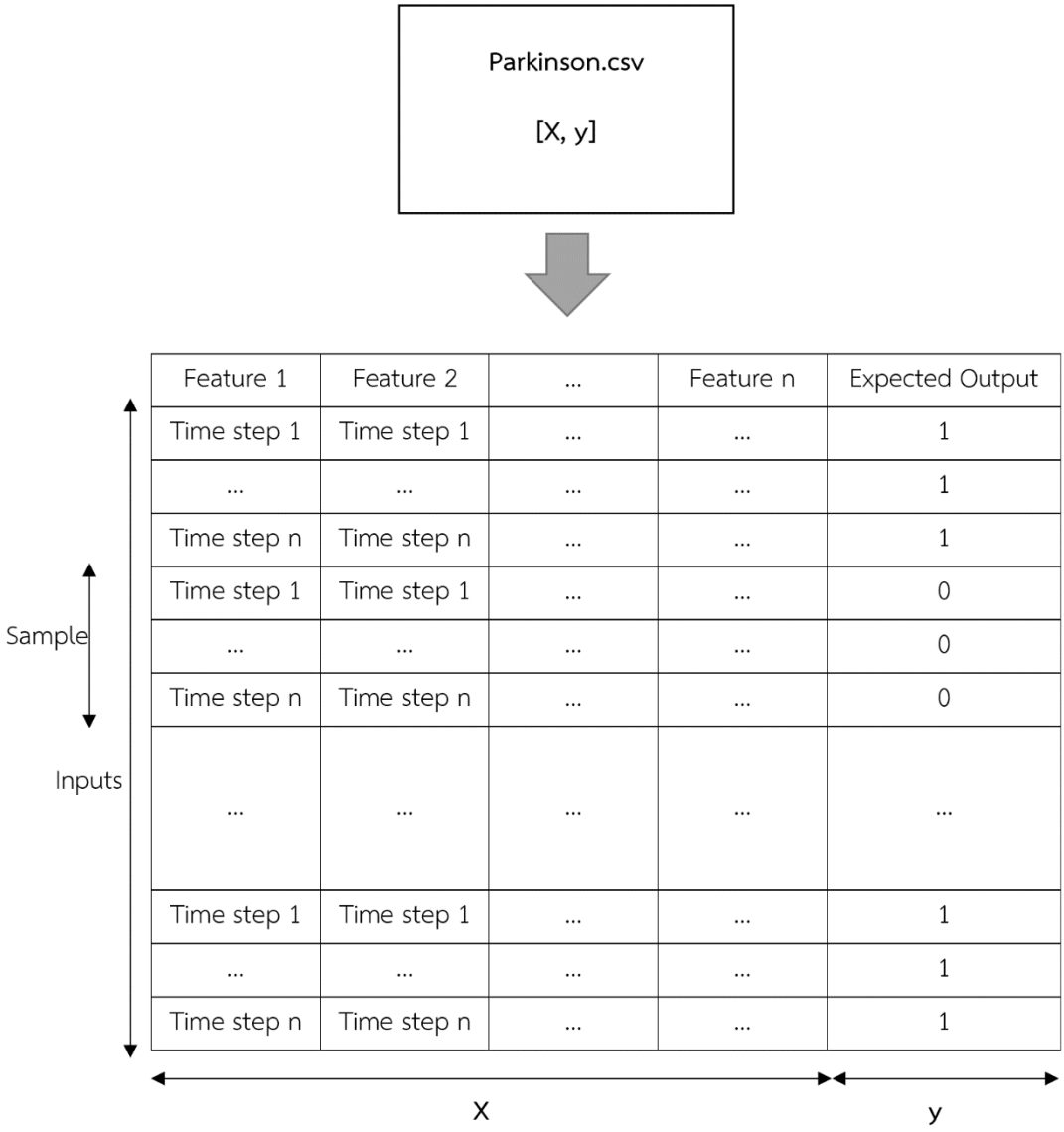
ข้อมูลส่วนแรกคือคุณลักษณ์ของสิ่งที่กำลังให้เครื่องเรียนรู้ X และข้อมูลส่วนต่อมาคือสิ่งที่เครื่องต้องจำแนก จึงแปลงข้อมูล time series ไปเป็น supervised learning และผลลัพธ์ที่คาดหวัง Expected outputs : Y

ตารางที่ 3.5 Supervised learning

X	Y
...	1
...	1
...	0
...	...

ในการวิจัยนี้ เพื่อให้สะดวกต่อการใช้งานหลายรูปแบบของข้อมูลขาเข้า จึงทำการแบ่งข้อมูลใหญ่ให้แตกย่อยออกมา จากข้อมูลใหญ่ X, y ที่ได้มาหลังจากการจัดการข้อมูล ทำการแบ่งข้อมูลแยกคุณลักษณะ (features) ออกมาเป็น feature เดียว และกำหนดค่า time steps ณ ที่นี้ใช้ค่าต่ำสุดของความยาวข้อมูลในขั้นตอนการจัดการข้อมูล คือ minimum length จากความยาวรวมของ X, y

จะได้ว่า time steps × samples = input total length ซึ่งเมื่อใช้ค่า minimum length จะทำให้ทราบค่า samples เสมอตลอดทุกการใช้งานต่อไป เพื่อไม่ให้การทดลองสับสน ในค่าพารามิเตอร์ต่างๆ เนื่องจากการวิจัยนี้จะทำหลายรูปแบบและจะทำให้เข้าใจความต่างของการใช้รูปแบบข้อมูลขาเข้าหลายรูปแบบได้ง่ายขึ้น



Feature 1	Feature 2	...	Feature n	Expected Output
Time step 1	Time step 1	...	...	1
...	...	...	...	1
Time step n	Time step n	...	...	1
Time step 1	Time step 1	...	...	0
...	...	...	...	0
Time step n	Time step n	...	...	0
...	...	...	...	...
Time step 1	Time step 1	...	...	1
...	...	...	...	1
Time step n	Time step n	...	...	1

ภาพที่ 3.12 Input shape

การทำ sequence splitting ทำให้สามารถให้ได้กับทุกรูปแบบข้อมูลขาเข้า คือข้อมูลขอเข้าจะมีได้ได้แต่ 1 feature จนไปถึง n feature ก็สามารถทำได้ ไม่จำเป็นต้องออกแบบโมเดลและ reshape ข้อมูลขาเข้าใหม่ทุกครั้งเมื่อทำการเปลี่ยนรูปแบบข้อมูลขาเข้า

3.4.1 การใช้ข้อมูลเชิงสถิติ

จากตารางที่ 3 ซึ่งเป็นข้อมูลสถิติรวมของข้อมูลคีย์บอร์ด ซึ่งแต่ละการทดสอบก็มีข้อมูลเชิงสถิติของตัวเอง จึงได้ Features หลักๆอย่าง ความแตกต่างเวลาของการกดคีย์บอร์ด (time difference), แนวโน้มของความแตกต่างของเวลา (trend) ถ้ามีค่าเป็นบวกแปลว่าการกดช้าลง หรือในทางกลับกันถ้ามีค่าเป็นลบแปลว่ากดได้ไวขึ้น, การกดค้าง (Burst), การกดซ้ำ (Repeat), การกด 2 คีย์พร้อมกัน (2 keys at once), การกดคีย์ต่างๆ (key1, key2, key3 และ key4), ความผิดพลาด (error) และอัตราเร็ว (velocity) รวมแล้วเป็น 11 features จะได้มิติของข้อมูลขาเข้าเป็น input shape เป็น 11 และตัวอย่าง (sample) เป็น 12×100=1200 samples ได้ raw sequence shape : X ดังนี้

ตารางที่ 3.6 Input สำหรับข้อมูลเชิงสถิติ

Mean	Trend	Brust	Repeat	2k	Key1	Key2	Key3	Key4	Error	Velocity	output
...	...	...	...	...	...	...	...	...	...	...	...

ข้อมูลของแต่ละ feature นั้นได้ถูกกำหนดความยาวไว้ที่ 1 time steps จึงได้มิติของ dataset shape นี้แบบ n_sample n_steps, n_features เป็น 1200×1×11 จึงได้

- Input_dim = 11 สำหรับขั้นตอนวิธีสำหรับการแบ่งข้อมูลมาตรฐาน
- Input_shape = (1, 11) สำหรับขั้นตอนวิธีสำหรับ K-Fold Cross-Validation

หลังจากได้ dataset แล้วนั้น จึงนำมาแบ่งข้อมูล (Data splitting) เป็น Training set และ Testing set ทั้ง X และ Y แล้ว ข้อมูลเป็น 3D เพื่อป้อนเข้า model หลังจากกำหนดค่าพารามิเตอร์ต่างๆเสร็จแล้ว จะได้ผลลัพธ์ออกมา โดยใช้ค่าเริ่มต้นของตัวแปรดังนี้

- neurons = n_neurons = 20
- activation = optimizer = ‘relu’
- epoch = n_epochs = 100
- loss = cost function = ‘binary_crossentropy’

3.4.2 การใช้ข้อมูลจากคีย์บอร์ด

ข้อมูลของคีย์บอร์ดถูกนำมาแปลงด้วยทำการแปลงข้อมูลจากตัวอักษรอย่าง “b”, “m”, “z”, “/” และอื่นๆ เป็นค่า -1 ถึง 1 ด้วยการกำหนดคะแนนของการกดคีย์บอร์ดเทียบค่าตัวแปรจากตารางที่3 ได้ดังนี้

- การกดค้าง (burst) กำหนดไว้ที่ -0.5
- การกดซ้ำ (repeat) กำหนดไว้ที่ 0.5
- การกดผิด (error) กำหนดไว้ที่ -1
- การกดถูก กำหนดไว้ที่ 1
- ช่วงว่าง กำหนดไว้ที่ 0

ข้อมูลคีย์บอร์ดจะถูกทำให้เป็นข้อมูล 40 Hz ให้เท่ากับข้อมูลเซ็นเซอร์ ด้วยการเปรียบเทียบเวลาของคีย์บอร์ดกับเซ็นเซอร์ ซึ่งมีขั้นตอนดังนี้

```
sensor data index = 0
keyboard data index = 0
new keyboard data = []
loop sensor index in range of sensor data length :
    check keyboard time >= sensor time:
        use keyboard feature data
        keyboard index +=1
    else : use 0 as keyboard feature data
    sensor index +=1
```

ข้อมูลจากคีย์บอร์ดจะได้ 2 features คือ เวลา (time) และ คีย์ (key) จำนวนตัวอย่าง 1200 samples จะได้ raw sequence shape : X

ตารางที่ 3.7 Input สำหรับข้อมูลคีย์บอร์ด

Time	Key	output
...	...	...

ข้อมูลของแต่ละ feature นั้นได้ถูกกำหนดความยาวไว้ที่ 500 time steps จึงได้มิติของ dataset shape นี้แบบ n_sample, n_steps, n_features เป็น 1200×500×2 จึงได้

- Input_dim = 2 สำหรับขั้นตอนวิธีสำหรับการแบ่งข้อมูลมาตรฐาน
- Input_shape = (500, 2) สำหรับขั้นตอนวิธีสำหรับ K-Fold Cross-Validation

หลังจากได้ dataset แล้วนั้น จึงนำมาแบ่งข้อมูล (Data splitting) เป็น Training set และ Testing set ทั้ง X และ Y แล้วจึงสามารถ reshape ข้อมูลเป็น 3D เพื่อป้อนเข้า LSTMs model หลังจากกำหนดค่าพารามิเตอร์ต่างๆเสร็จแล้วจะได้ผลลัพธ์ออกมา โดยใช้ค่าเริ่มต้นของตัวแปรดังนี้

- neurons = n_neurons = 100
- activation = optimizer = ‘relu’
- batch_size = n_bacth = 100
- epoch = n_epochs = 100
- dropout = 0.0
- loss = cost function = ‘binary_crossentropy’

3.4.3 การใช้ข้อมูลจากเซ็นเซอร์

พบว่าข้อมูลจากเซ็นเซอร์นั้นมีสิ่งหนึ่งที่เหมือนกันของทุกๆ sequences คือ เวลา (time) ดังนั้นจึงสามารถตัดออกได้ จึงได้ข้อมูลจากเซ็นเซอร์จะได้ 12 features ค่าจากเซ็นเซอร์ (Gyroscope sensor) ประกอบไปด้วยค่าความเร่ง (Acceleration) คือ Ax, Ay, Az, Ax2, Ay2 และ Az2 และค่ามุม (Angle)) คือ Gx, Gy, Gz, Gx2, Gy2 และ Gz2 จากทั้ง 2 ตัวที่ติดไว้ตรงข้อนิ้วปลายนิ้วของผู้ที่เข้ารับการทดสอบ จำนวนตัวอย่าง 1200 samples จะได้ raw sequence shape : X

ตารางที่ 3.8 Input ข้อมูลเซ็นเซอร์

Ax	Ay	Az	Gx	Gy	Gz	Ax2	Ay2	Az2	Gx2	Gy2	Gz2	output
...	...	...	...	...	...	...	...	...	...	...	...	...

ข้อมูลของแต่ละ feature นั้นได้ถูกกำหนดความยาวไว้ที่ 500 time steps จึงได้มิติของ dataset shape นี้แบบ n_sample, n_steps, n_features เป็น 1200×500×12 จึงได้

- Input_dim = 12 สำหรับขั้นตอนวิธีสำหรับการแบ่งข้อมูลมาตรฐาน
- Input_shape = (500, 12) สำหรับขั้นตอนวิธีสำหรับ K-Fold Cross-Validation

หลังจากได้ dataset แล้วนั้น จึงนำมาแบ่งข้อมูล (Data splitting) เป็น Training set และ Testing set ทั้ง X และ Y แล้วจึงสามารถ reshape ข้อมูลเป็น 3D เพื่อป้อนเข้า LSTMs model หลังจากกำหนดค่าพารามิเตอร์ต่างๆเสร็จแล้วจะได้ผลลัพธ์ออกมา โดยใช้ค่าเริ่มต้นของตัวแปรดังนี้

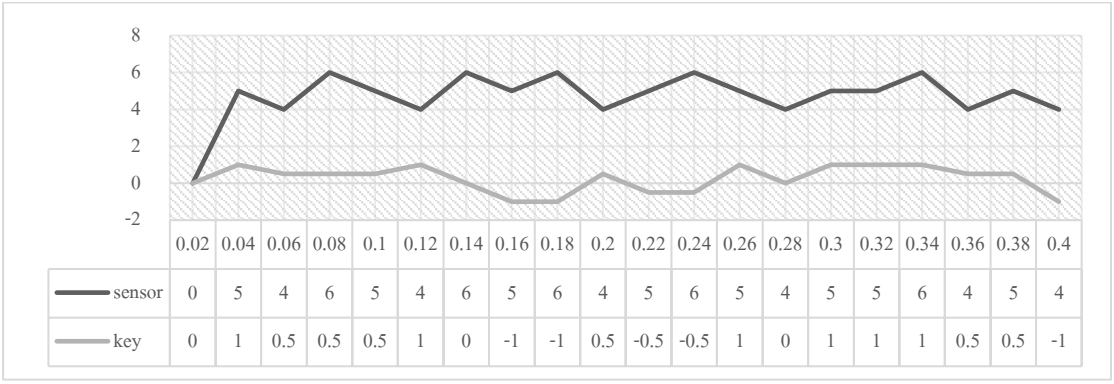
- neurons = n_neurons = 100
- activation = optimizer = ‘relu’
- batch_size = n_bacth = 100
- epoch = n_epochs = 100
- loss = cost function = ‘binary_crossentropy’

3.4.4 การใช้ข้อมูลจากทั้งคีย์บอร์ดและเซ็นเซอร์

ข้อมูลของคีย์บอร์ดและเซ็นเซอร์มีขนาดไม่เท่ากันจึงทำการขยายข้อมูลจากคีย์บอร์ด เพราะข้อมูลจากคีย์บอร์ดสั้นกว่า เมื่อยืดขยายแล้วจะมีตัวเลขค่าหนึ่งที่ต้องระวังคือค่าการกดค้าง (burst) ที่มีช่วงเวลาที่ 0.03 วินาที และทำการแปลงข้อมูลจากตัวอักษรอย่าง “b”, “m”, “z”, “/” และอื่นๆ เป็นค่า -1 ถึง 1 ด้วยการกำหนดคะแนนของการกดคีย์บอร์ดเทียบค่าตัวแปรจากตารางที่3.13 ได้ดังนี้

- การกดค้าง (burst) กำหนดไว้ที่ -0.5
- การกดซ้ำ (repeat) กำหนดไว้ที่ 0.5
- การกดผิด (error) กำหนดไว้ที่ -1
- การกดถูก กำหนดไว้ที่ 1
- ช่วงว่าง กำหนดไว้ที่ 0

การขยายข้อมูลนั้นวัดจากความถี่ โดยใช้ความถี่ (frequency, Hz) ที่ 40 Hz ด้วยเทคนิคการแปลงคลื่น (Wave Transform) จะทำให้คลื่นจากคีย์บอร์ดที่มีค่าตั้งแต่ -1 ถึง 1 ถูกแปลงเป็นคลื่นความถี่ 40 Hz เท่ากับคลื่นของเซ็นเซอร์ และจะแปลงคลื่นข้อมูลจากเซ็นเซอร์ให้เป็นความถี่ที่ 40 Hz เช่นกัน



	0.02	0.04	0.06	0.08	0.1	0.12	0.14	0.16	0.18	0.2	0.22	0.24	0.26	0.28	0.3	0.32	0.34	0.36	0.38	0.4
sensor	0	5	4	6	5	4	6	5	6	4	5	6	5	4	5	5	6	4	5	4
key	0	1	0.5	0.5	0.5	1	0	-1	-1	0.5	-0.5	-0.5	1	0	1	1	1	0.5	0.5	-1

ภาพที่ 3.13 ตัวอย่างคลื่นที่ถูกแปลง

จึงได้ข้อมูลจากเซ็นเซอร์จะได้ 13 features ค่าจากเซ็นเซอร์ (Gyroscope sensor) ประกอบไปด้วยค่าความเร่ง (Acceleration) คือ Ax, Ay, Az, Ax2, Ay2 และ Az2 และค่ามุม (Angle)) คือ Gx, Gy, Gz, Gx2, Gy2 และ Gz2 จากทั้ง 2 ตัวที่ติดไว้ตรงข้อนิ้วปลายนิ้วของผู้ที่เข้ารับ

การทดสอบ และค่าจากคีย์บอร์ด (key) จำนวนตัวอย่าง 1200 samples จะได้ raw sequence shape : X

ตารางที่ 3.9 Input ข้อมูลของเซ็นเซอร์และคีย์บอร์ด

Ax	Ay	Az	Gx	Gy	Gz	Ax2	Ay2	Az2	Gx2	Gy2	Gz2	Key	output
...	...	...	...	...	...	...	...	...	...	...	...	-1	...
...	...	...	...	...	...	...	...	...	...	...	...	0.5	...
...	...	...	...	...	...	...	...	...	...	...	...	0	...
...	...	...	...	...	...	...	...	...	...	...	...	...	...

ข้อมูลของแต่ละ feature นั้นได้ถูกกำหนดความยาวไว้ที่ 500 time steps จึงได้มิติของ dataset shape นี้แบบ n_sample, n_steps, n_features เป็น 1200×500×13 จึงได้

- Input_dim = 13 สำหรับขั้นตอนวิธีสำหรับการแบ่งข้อมูลมาตรฐาน
- Input_shape = (500, 13) สำหรับขั้นตอนวิธีสำหรับ K-Fold Cross-Validation

หลังจากได้ dataset แล้วนั้น จึงนำมาแบ่งข้อมูล (Data splitting) เป็น Training set และ Testing set ทั้ง X และ Y แล้วจึงสามารถ reshape ข้อมูลเป็น 3D เพื่อป้อนเข้า LSTMs model หลังจากกำหนดค่าพารามิเตอร์ต่างๆเสร็จแล้วจะได้ผลลัพธ์ออกมา โดยใช้ค่าเริ่มต้นของตัวแปรดังนี้

- neurons = n_neurons = 100
- activation = optimizer = ‘relu’
- batch_size = n_bacth = 100
- epoch = n_epochs = 100
- loss = cost function = ‘binary_crossentropy’

3.5 Parameter and Hyperparameter Optimization

การปรับค่าตัวแปรเพื่อหาความเหมาะสมที่สุด เพื่อให้ได้ผลลัพธ์การทำงานเป็น Optimal คือ ไม่ Overfitting หรือ Underfitting ซึ่งมี 2 วิธีที่นิยมคือ Random Search และที่ใช้ในการวิจัยนี้คือ Grid Search ด้วย scikit-learn โดยในการวิจัยนี้ใช้ KerasClassifier การใช้ Keras Models สำหรับ scikit-learn เป็นดังนี้

```
#Function to create model
def create_model():
    #Define model
    ...
    return model
#Call the model function
model = KerasClassifier(build_fn=create_model)
```

Model Parameters คือ ตัวแปรใน Neural Network ซึ่งค่าของตัวแปรนั้นสามารถประมาณการได้จากข้อมูล ซึ่งเป็นค่าที่เป็นตัวชี้วัดประสิทธิภาพของโมเดล และไม่สามารถตั้งค่าด้วยการตั้งค่าจากการโปรแกรมโดยตรง ตัวอย่างเช่น weights และ biases ดังนั้น model parameters จะถูกบันทึกรวมไปเป็นส่วนหนึ่งของโมเดลที่ได้รับการเรียนรู้

Model Hyperparameters คือ ตัวแปรภายนอก Neural Network ซึ่งค่าขอตัวแปรไม่สามารถประมาณการได้จากข้อมูล เมื่อกล่าวถึง Machine Learning Tunning ก็เป็นการหมายถึง hyperparameter tunning

ขั้นตอนวิธีการใช้ KerasClassifier คือ

- สร้างฟังก์ชันโมเดลสำหรับปรับแก้แยกไว้ก่อน
- สร้างชุดข้อมูลตัวแปร
- สร้างโมเดลของ KerasClassifier
- เรียกใช้ GridSearchCV
- ปรับ grid กับข้อมูล X, y
- แสดงผลการทำงานเพื่อหาค่าที่เหมาะสมที่สุด

```
Function to create model, required for KerasClassifier
    Create model
    Compile model
    return model
Fix random seed for reproducibility
Load dataset
Split into input (X) and output (Y) variables
Create model from the function
Define the grid search parameters
PARAMETERS = [ ..., ..., ..., ...]
param_grid = dict(PARAMETERS=PARAMETER_DESTINATION_REPLACE, ...)
grid = GridSearchCV(estimator=model, param_grid=param_grid, n_jobs=-1, cv=3)
grid_result = grid.fit(X, Y)
Summarize results (print)
```

3.5.1 Batch Size and Number of Epochs tuning

Batch size คือ จำนวนรายการที่จะใช้ optimizer ทำงานในแต่ละครั้ง ซึ่งมีผลต่อความเร็วในการทำงาน ถ้า batch ใหญ่จะทำงานเร็ว

Epochs คือ จำนวนรอบการทำการเรียนรู้

```
Function to create model, required for KerasClassifier
    Create model
    Compile model
    return model
Fix random seed for reproducibility
Load dataset
Split into input (X) and output (Y) variables
Create model
Define the grid search parameters
batch_size = [..., ..., ...]
epochs = [..., ..., ...]
param_grid = dict(batch_size=batch_size, epochs=epochs)
grid = GridSearchCV(...)
grid_result = grid.fit(X, Y)
Summarize results
```

3.5.2 Training Optimization Algorithm tuning

```
Function to create model, required for KerasClassifier
def create_model(optimizer='adam'):
        Create model
        Compile model
        model.compile(loss='binary_crossentropy', optimizer=optimizer, metrics=['accuracy'])
        return model
Fix random seed for reproducibility
Load dataset
Split into input (X) and output (Y) variables
Create model
Define the grid search parameters
optimizer = ['SGD', 'RMSprop', 'Adagrad', 'Adadelta', 'Adam', 'Adamax', 'Nadam']
param_grid = dict(optimizer=optimizer)
grid = GridSearchCV(...)
grid_result = grid.fit(X, Y)
Summarize results
```

3.5.3 Network Weight Initialization tuning

```
# Function to create model, required for KerasClassifier
def create_model(init_mode='uniform'):
    Create model
    model = Sequential()
    model.add(LSTM(..., input_shape=(...), kernel_initializer=init_mode, activation='...'))
    model.add(Dense(1, kernel_initializer=init_mode, activation='sigmoid'))
    ...
    Compile model
    return model
Fix random seed for reproducibility
Load dataset
Split into input (X) and output (Y) variables
Create model
Define the grid search parameters
init_mode = ['uniform', 'lecun_uniform', 'normal', 'zero', 'glorot_normal', 'glorot_uniform',
'he_normal', 'he_uniform']
param_grid = dict(init_mode=init_mode)
grid = GridSearchCV(...)
grid_result = grid.fit(X, Y)
Summarize results
```

3.5.4 Learning Rate and Momentum tuning

```
Function to create model, required for KerasClassifier
def create_model(learn_rate=0.01, momentum=0):
    Create model
    Compile model
    optimizer = SGD(lr=learn_rate, momentum=momentum)
    model.compile(...)
    return model
Fix random seed for reproducibility
Load dataset
Split into input (X) and output (Y) variables
Create model
Define the grid search parameters
learn_rate = [..., ..., ...]
momentum = [..., ..., ...]
param_grid = dict(learn_rate=learn_rate, momentum=momentum)
grid = GridSearchCV(...)
grid_result = grid.fit(X, Y)
Summarize results
```

3.5.5 Neuron Activation Function tuning

```
# Function to create model, required for KerasClassifier
def create_model(activation='relu'):
        Create model
        model = Sequential()
        model.add(LSTM(..., input_shape=(...), kernel_initializer='uniform',
                activation=activation))
        model.add(Dense(1, kernel_initializer='uniform', activation='sigmoid'))
        Compile model
        return model
Fix random seed for reproducibility
Load dataset
Split into input (X) and output (Y) variables
Create model
Define the grid search parameters
activation = ['softmax', 'softplus', 'softsign', 'relu', 'tanh', 'sigmoid', 'hard_sigmoid', 'linear']
param_grid = dict(activation=activation)
grid = GridSearchCV(...)
grid_result = grid.fit(X, Y)
Summarize results
```

3.5.6 Dropout Regularization tuning

```
Function to create model, required for KerasClassifier
def create_model(dropout_rate=0.0, weight_constraint=0):
        Create model
        model = Sequential()
        model.add(LSTM(..., input_shape=(...), kernel_initializer='uniform', activation='linear',
                kernel_constraint=maxnorm(weight_constraint)))
        model.add(Dropout(dropout_rate))
        model.add(Dense(1, ...))
        Compile model
        return model
Fix random seed for reproducibility
Load dataset
Split into input (X) and output (Y) variables
Create model
Define the grid search parameters
weight_constraint = [..., ..., ...]
dropout_rate = [..., ..., ...]
param_grid = dict(dropout_rate=dropout_rate, weight_constraint=weight_constraint)
grid = GridSearchCV(...)
grid_result = grid.fit(X, Y)
Summarize results
```

3.5.7 Number of Neurons in Hidden Layer tuning

```
Function to create model, required for KerasClassifier
def create_model(neurons=1):
        Create model
        model = Sequential()
        model.add(LSTM(neurons, ...)
        model.add(Dropout(0.2))
        model.add(Dense(1, ...))
        Compile model
        return model
Fix random seed for reproducibility
Load dataset
Split into input (X) and output (Y) variables
Create model
Define the grid search parameters
neurons = [..., ..., ...]
param_grid = dict(neurons=neurons)
grid = GridSearchCV(...)
grid_result = grid.fit(X, Y)
Summarize results
```

3.6 ขั้นตอนวิธีการเรียนรู้เครื่อง

```
Fix random seed for reproducibility
Load dataset
Split into input (X) and output (Y) variables
Reshape input into 3D format : (samples, steps, features)
Define Stratified K-Fold cross validation test harness : K=10
Create variables for collecting accuracy data from models
For train, test in kfold.split(X, Y):
    Create a model#1
    Compile model#1
    Fit the model#1
    Evaluate the model#1
    Export the trained model into a folder
    Print the accuracy percentage of the model
    Add the performance data into the variable

    Create a model#2
    Compile model#2
    Fit the model#2
    Evaluate the model#2
    Export the trained model into a folder
    Print the accuracy percentage of the model
    Add the performance data into the variable
Print the results from the variable

Load the exported models from a folder
Load the input for making prediction
Show the prediction result
```

โมเดลสำหรับโครงข่ายประสาทเทียมแบบ LSTM ที่มี 3 Hidden layers โดยเป็น LSTM 2 layers และต่อด้วย Dense layer และมี Output เป็นโหนดที่มีการเชื่อมต่อครบทุกโหนดก่อนหน้า

ค่าพารามิเตอร์ต่างๆ ของแต่ละส่วนคือ

- Hidden layers: activation function = ‘relu’

  Weight initialization = ‘uniform’

  Weight constraint = maxnorm(4)

  และ Dropout = 0.2

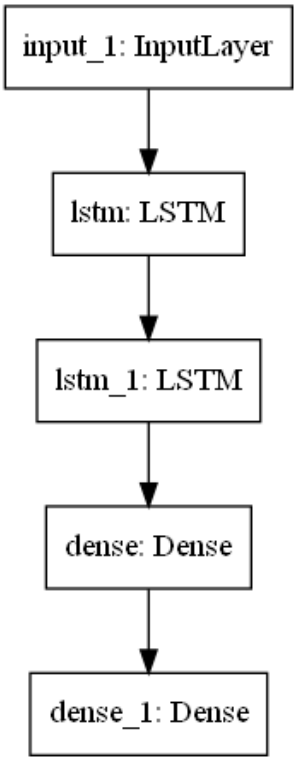
- Model Compiling: Loss function = ‘binary_crossentropy’

  Optimizer = ‘Adam’

  และ Learning rate = 0.01

- Model Fitting: Batch size = 100

  และ Epochs = 150



ภาพที่ 3.14 LSTM model

3.6.1 ขั้นตอนวิธีการเรียนรู้เครื่องแบบ Feature Extraction ที่ 1

Fix random seed for reproducibility

Load dataset

Split into input (X) and output (Y) variables

Reshape input into 3D format : (samples, steps, features)

Define Stratified K-Fold cross validation test harness : K=10

Create variables for collecting accuracy data from models

For train, test in kfold.split(X, Y):

Define input

Feature extraction

extract1 = LSTM(...)(visible)

First interpretation model

interp1 = LSTM(...)(extract1)

interp2 = Dense(...)(extract1)

Second interpretation model

interp11 = Dense(...)(extract1)

interp12 = Dense(...)(interp11)

interp13 = Dense(...)(interp12)

Merge interpretation

merge = concatenate([interp1, interp13])

Output

output = Dense(1, activation='sigmoid')(merge)

model = Model(inputs=visible, outputs=output)

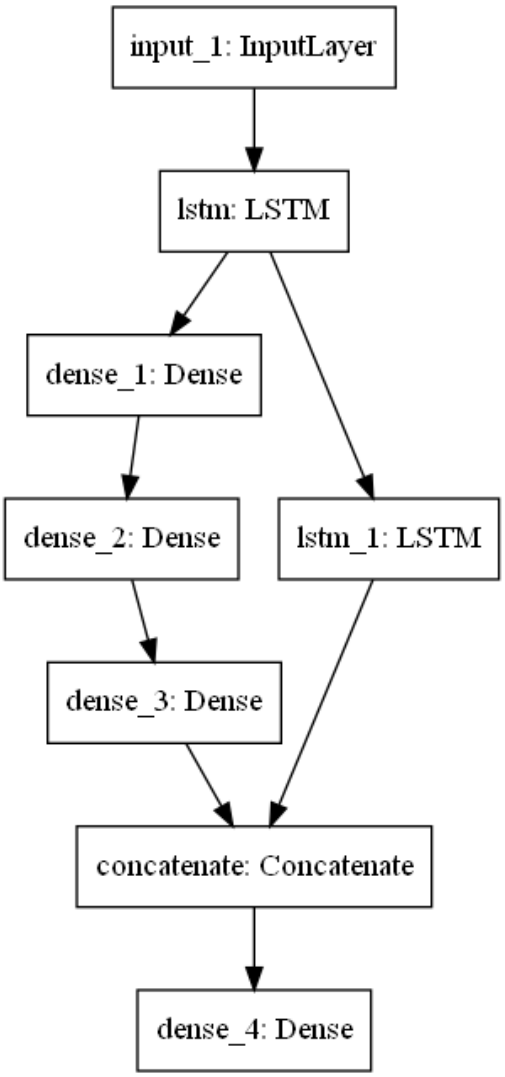
Compile model

Fit the model

Evaluate the model

Print the results from the variable

ใช้ค่าพารามิเตอร์เหมือนกับโมเดลก่อน แต่เปลี่ยนโครงสร้างโมเดลเป็นดังรูปด้านล่าง



ภาพที่ 3.15 LSTM Feature Extraction model (1)

3.6.2 ขั้นตอนวิธีการเรียนรู้เครื่องแบบ Feature Extraction ที่ 2

Fix random seed for reproducibility

Load dataset

Split into input (X) and output (Y) variables

Reshape input into 3D format : (samples, steps, features)

Define Stratified K-Fold cross validation test harness : K=10

Create variables for collecting accuracy data from models

For train, test in kfold.split(X, Y):

    Define input

    Feature extraction

    extract1 = LSTM(...)(visible)

    # 1 interpretation model

    interp1 = LSTM(...)(extract1)

    interp2 = Dense(...)(interp1)

    # 2 interpretation model

    interp11 = LSTM(...)(extract1)

    interp12 = Dense(...)(interp11)

    interp13 = Dense(...)(interp12)

    # 3 interpretation model

    interp111 = LSTM(...)(extract1)

    interp112 = Dense(...)(interp111)

    Merge interpretation

    merge = concatenate([interp2, interp13, interp112])

    Interpretation layer

    hidden1 = Dense(...)(merge)

    Output

    output = Dense(1, ...)(hidden1)

    model = Model(inputs=visible, outputs=output)

    Evaluate the model

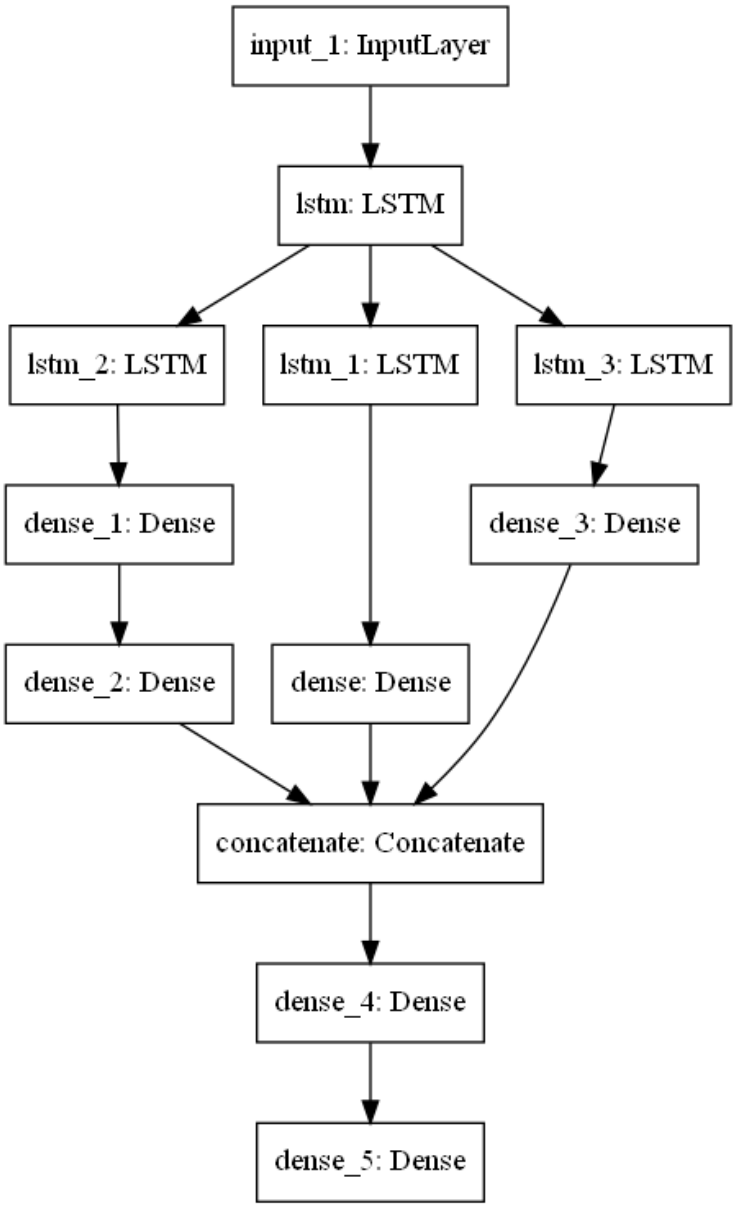
    Compile model

    Fit the model

    Evaluate the model

Print the results from the variable

เพื่อทดสอบหากว่าการเพิ่มโครงสร้างโมเดลภายในแล้วได้ผลเปรียบเทียบกับโมเดลอื่น



ภาพที่ 3.16 LSTM Feature Extraction model (2)

3.6.3 ขั้นตอนวิธีการเรียนรู้เครื่องแบบ 2-Feature Extraction

```
Fix random seed for reproducibility
Load dataset
Split into input (X) and output (Y) variables
Reshape input into 3D format : (samples, steps, features)
Define Stratified K-Fold cross validation test harness : K=10
Create variables for collecting accuracy data from models
For train, test in kfold.split(X, Y):
        Define input
        # 1 feature extraction
        extract1 = LSTM(...)(visible)
        # 2 feature extraction
        extract2 = LSTM(...)(visible)
        # 1ex first interpretation model
        interp1 = LSTM(...)(extract1)
        interp2 = Dense(...)(extract1)
        Second interpretation model
        interp11 = Dense(...)(extract1)
        interp12 = Dense(...)(interp11)
        interp13 = Dense(...)(interp12)
        # 2ex first interpretation model
        interp21 = LSTM(...)(extract2)
        interp22 = Dense(...)(interp21)
        Second interpretation model
        interp211 = Dense(...)(extract2)
        interp212 = Dense(...)(interp211)
        interp222 = Dense(...)(interp212)
        Merge interpretation
        merge = concatenate([interp13, interp222])
```

Interpretation layer

hidden1 = Dense(...)(merge)

Output

output = Dense(1, ...)(hidden1)

model = Model(inputs=visible, outputs=output)

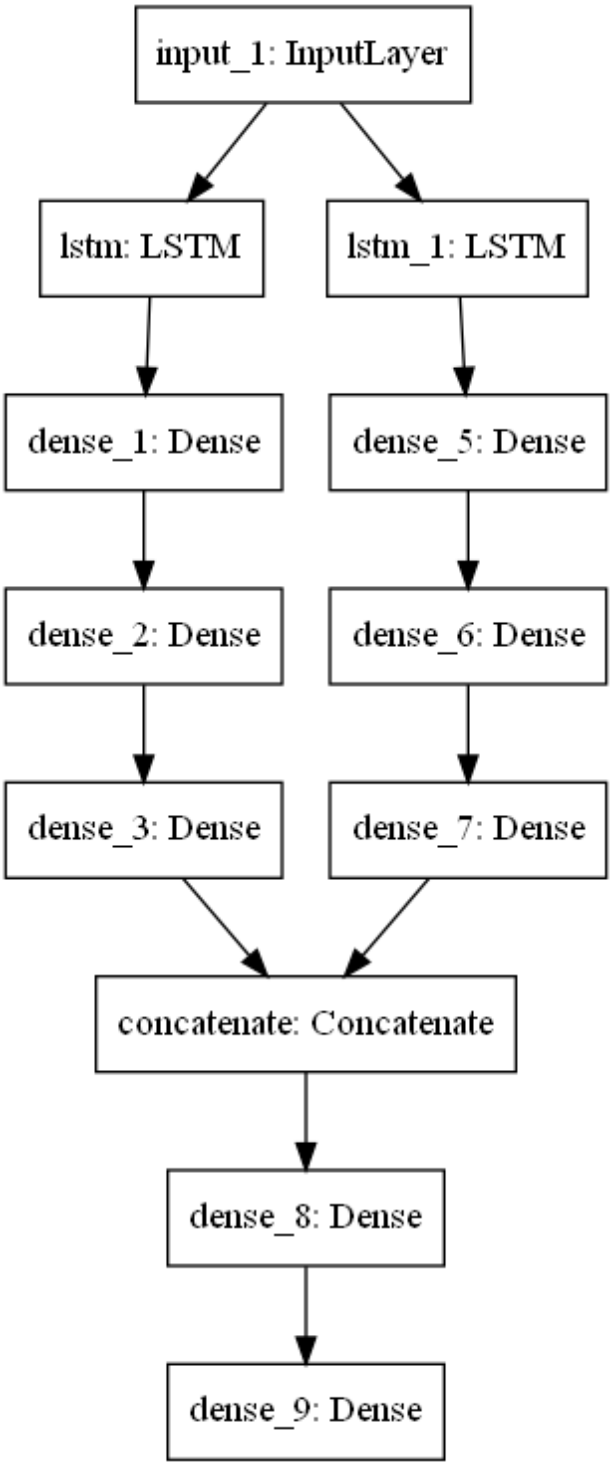
Compile model

Fit the model

Evaluate the model

Print the results from the variable

โมเดลนี้ต่างจาก 2 โมเดลก่อนหน้าตรงที่ได้ทำการทำ LSTMs เป็น 2 สายขนานกัน และแน่นอนว่าผลการทำงานย่อมดีกว่าการที่เป็นโมเดลสายเดียว โครงสร้างโมเดลเป็นดังรูปที่ 3.15



ภาพที่ 3.17 LSTM 2-Feature Extraction model

บทที่ 4
ผลการดำเนินงาน

ผลการดำเนินงานจะแบ่งเป็น 2 ส่วนหลัก คือ ผลการทดลองของการเปรียบเทียบทำงานของ Simple fully-connected neuron model ระหว่าง LSTM model ด้วยการใช้ K-Fold Cross Validation กับทั้งสองโมเดล และผลการทดลองเพื่อหา LSTM model ที่เหมาะสมที่สุด ผลการทดลองที่ได้จะเป็นตัวชี้วัดการเลือกโมเดลสำหรับงานวิจัยนี้ โดยดูจากค่าความแม่นยำสูงสุดของการทดลอง

4.1 ผลการทดลองของข้อมูลเชิงสถิติ

ผลการทดลองนี้เป็นผลการทดลองของการเปรียบเทียบทำงานของ Simple fully-connected neuron model ระหว่าง LSTM model กับข้อมูลเชิงสถิติ 30 ครั้ง ซึ่งเป็นข้อมูลที่สามารถวิเคราะห์ได้สะดวกที่สุด และสามารถเก็บรวบรวมสถิติได้เป็นดังตารางที่ 3.3

ตารางที่ 4.1 ผลการทดลองของข้อมูลเชิงสถิติ

	Standard	LSTM
ค่าเฉลี่ย	67.9684	75.3962
ส่วนเบี่ยงเบนมาตรฐาน	2.5274	1.6534
ค่าน้อยสุด	62.987	72.0779
ค่าสูงสุด	73.974	78.5714

จากตารางผลการทดลองที่ 4.1 พบว่า ค่าเฉลี่ยของวิธี Standard อยู่ที่ 67.97% และ LSTM อยู่ที่ 75.40% และจากการทดสอบความแตกต่างของค่าเฉลี่ยระหว่างวิธี Standard และ LSTM ด้วยวิธีการทดสอบที (t-Test) ด้วยความมั่นใจ 95% พบว่าวิธี LSTM มีค่าความแม่นยำสูงกว่าวิธี Standard ซึ่งได้ค่า Statistical Power ที่ 0.9319 คือ ค่าเฉลี่ยของวิธี Standard และ LSTM มีความแตกต่างกันจริง, ความผิดพลาดแบบที่ 2 ที่ 0.0681 คือ ค่าเฉลี่ยของวิธี Standard และ LSTM มีความแตกต่างกันแต่กลับสรุปว่าไม่มีความแตกต่างกัน และความผิดพลาดแบบที่ 1 ที่ 0.05 คือ ค่าเฉลี่ยของวิธี Standard และ LSTM ไม่มีความแตกต่างกันแต่กลับสรุปว่ามีความแตกต่างกัน

4.2 ผลการทดลองของข้อมูลคีย์บอร์ด

ผลการทดลองนี้เป็นผลการทดลองของการเปรียบเทียบทำงานของ Simple fully-connected neuron model ระหว่าง LSTM model กับข้อมูลคีย์บอร์ด 30 ครั้ง ข้อมูลคีย์บอร์ดเป็นข้อมูลที่สามารถนำมาใช้หาสถิติโดยรวมได้

ตารางที่ 4.2 ผลการทดลองของข้อมูลคีย์บอร์ด

	Standard	**LSTM**
ค่าเฉลี่ย	73.3319	77.7297
ส่วนเบี่ยงเบนมาตรฐาน	1.8465	2.0881
ค่าน้อยสุด	69.0652	72.1666
ค่าสูงสุด	80.3102	81.035

จากตารางผลการทดลองที่ 4.2 พบว่า ค่าเฉลี่ยของวิธี Standard อยู่ที่ 73.33% และ LSTM อยู่ที่ 77.73% และจากการทดสอบความแตกต่างของค่าเฉลี่ยระหว่างวิธี Standard และ LSTM ด้วยวิธีการทดสอบที (t-Test) ด้วยความมั่นใจ 95% พบว่าวิธี LSTM มีค่าความแม่นยำสูงกว่าวิธี Standard ซึ่งได้ค่า Statistical Power ที่ 0.9932 คือ ค่าเฉลี่ยของวิธี Standard และ LSTM มีความแตกต่างกันจริง, ความผิดพลาดแบบที่ 2 ที่ 0.0078 คือ ค่าเฉลี่ยของวิธี Standard และ LSTM มีความแตกต่างกันแต่กลับสรุปว่าไม่มีความแตกต่างกัน และความผิดพลาดแบบที่ 1 ที่ 0.05 คือ ค่าเฉลี่ยของวิธี Standard และ LSTM ไม่มีความแตกต่างกันแต่กลับสรุปว่ามีความแตกต่างกัน

4.3 ผลการทดลองของข้อมูลเซ็นเซอร์

ผลการทดลองนี้เป็นผลการทดลองของการเปรียบเทียบทำงานของ Simple fully-connected neuron model ระหว่าง LSTM model กับข้อมูลเซ็นเซอร์ 30 ครั้ง ซึ่งเป็นข้อมูลการเคลื่อนไหวของผู้ที่เข้าร่วมการทดสอบ หากใช้สายตาสังเกตแพทย์ผู้เชี่ยวชาญสามารถวินิจฉัยได้ว่าเป็นโรคพาร์กินสันหรือไม่ได้ แต่แพทย์ OPD จะเป็นการยากที่จะสามารถวินิจฉัยการอาการได้

ตารางที่ 4.3 ผลการทดลองของข้อมูลเซ็นเซอร์

	Standard	**LSTM**
ค่าเฉลี่ย	75.6524	79.0444
ส่วนเบี่ยงเบนมาตรฐาน	1.6764	0.829493
ค่าน้อยสุด	72.0779	76.6234
ค่าสูงสุด	79.8701	80.1332

จากตารางผลการทดลองที่ 4.3 พบว่า ค่าเฉลี่ยของวิธี Standard อยู่ที่ 75.65% และ LSTM อยู่ที่ 79.04% และจากการทดสอบความแตกต่างของค่าเฉลี่ยระหว่างวิธี Standard และ LSTM ด้วยวิธีการทดสอบที (t-Test) ด้วยความมั่นใจ 95% พบว่าวิธี LSTM มีค่าความแม่นยำสูงกว่าวิธี Standard ซึ่งได้ค่า Statistical Power ที่ 0.9783 คือ ค่าเฉลี่ยของวิธี Standard และ LSTM มีความแตกต่างกันจริง, ความผิดพลาดแบบที่ 2 ที่ 0.0217 คือ ค่าเฉลี่ยของวิธี Standard และ LSTM มีความแตกต่างกันแต่กลับสรุปว่าไม่มีความแตกต่างกัน และความผิดพลาดแบบที่ 1 ที่ 0.05 คือ ค่าเฉลี่ยของวิธี Standard และ LSTM ไม่มีความแตกต่างกันแต่กลับสรุปว่ามีความแตกต่างกัน

4.4 ผลการทดลองของข้อมูลเซ็นเซอร์และคีย์บอร์ด

ผลการทดลองนี้เป็นผลการทดลองของการเปรียบเทียบทำงานของ Simple fully-connected neuron model ระหว่าง LSTM model กับข้อมูลเซ็นเซอร์และคีย์บอร์ด 30 ครั้ง เมื่อข้อมูลคีย์บอร์ดถูกทำการ Normalization แล้วจะมีขนาดและมิติเท่ากับข้อมูลเซ็นเซอร์ จึงทำการรวมข้อมูลเข้าด้วยกัน เนื่องจากยิ่งข้อมูลมาก ผลการทดลองยิ่งแม่นยำสูง

ตารางที่ 4.4 ผลการทดลองของข้อมูลเซ็นเซอร์และคีย์บอร์ด

	Standard	**LSTM**
ค่าเฉลี่ย	76.3452	80.5820
ส่วนเบี่ยงเบนมาตรฐาน	1.8984	1.2463
ค่าน้อยสุด	74.026	77.5416
ค่าสูงสุด	81.016	83.0014

จากตารางผลการทดลองที่ 4.4 พบว่า ค่าเฉลี่ยของวิธี Standard อยู่ที่ 76.35% และ LSTM อยู่ที่ 80.58% และจากการทดสอบความแตกต่างของค่าเฉลี่ยระหว่างวิธี Standard และ LSTM ด้วยวิธีการทดสอบที (t-Test) ด้วยความมั่นใจ 95% พบว่าวิธี LSTM มีค่าความแม่นยำสูงกว่าวิธี Standard ซึ่งได้ค่า Statistical Power ที่ 0.9951 คือ ค่าเฉลี่ยของวิธี Standard และ LSTM มีความแตกต่างกันจริง, ความผิดพลาดแบบที่ 2 ที่ 0.0049 คือ ค่าเฉลี่ยของวิธี Standard และ LSTM มีความแตกต่างกันแต่กลับสรุปว่าไม่มีความแตกต่างกัน และความผิดพลาดแบบที่ 1 ที่ 0.05 คือ ค่าเฉลี่ยของวิธี Standard และ LSTM ไม่มีความแตกต่างกันแต่กลับสรุปว่ามีความแตกต่างกัน

การเปรียบเทียบข้อมูลทั้ง 4 แบบคือ ข้อมูลสถิติ, ข้อมูลคีย์บอร์ด, ข้อมูลเซ็นเซอร์, และข้อมูลเซ็นเซอร์และคีย์บอร์ด กับโมเดล LSTM โดยดูจากผลการทดลองที่ 4.1, 4.2, 4.3 และ 4.4 ทำการวิเคราะห์แล้วได้ผลดังนี้ คือ

การวิเคราะห์ค่าเฉลี่ยของข้อมูลสถิติ และข้อมูลเซ็นเซอร์และคีย์บอร์ด จากการทดสอบความแตกต่างของค่าเฉลี่ยระหว่างข้อมูลสถิติ และข้อมูลเซ็นเซอร์และคีย์บอร์ด ด้วยวิธีการทดสอบที (t-Test) ด้วยความมั่นใจ 95% พบว่าข้อมูลเซ็นเซอร์และคีย์บอร์ด มีค่าความแม่นยำสูงกว่าข้อมูลสถิติ ซึ่งได้ค่า Statistical Power ที่ 0.9726 คือ ค่าเฉลี่ยของข้อมูลสถิติ และข้อมูลเซ็นเซอร์และคีย์บอร์ดมีความแตกต่างกันจริง, ความผิดพลาดแบบที่ 2 ที่ 0.0274 คือ ค่าเฉลี่ยของข้อมูลสถิติ และข้อมูลเซ็นเซอร์และคีย์บอร์ด มีความแตกต่างกันแต่กลับสรุปว่าไม่มีความแตกต่างกัน และความผิดพลาด

แบบที่ 1 ที่ 0.05 คือ ค่าเฉลี่ยของข้อมูลสถิติ และข้อมูลเซ็นเซอร์และคีย์บอร์ด ไม่มีความแตกต่างกัน แต่กลับสรุปว่ามีความแตกต่างกัน

การวิเคราะห์ค่าเฉลี่ยของข้อมูลคีย์บอร์ด และข้อมูลเซ็นเซอร์และคีย์บอร์ด จากการทดสอบความแตกต่างของค่าเฉลี่ยระหว่างข้อมูลคีย์บอร์ด และข้อมูลเซ็นเซอร์และคีย์บอร์ด ด้วยวิธีการทดสอบที (t-Test) ด้วยความมั่นใจ 95% พบว่าข้อมูลเซ็นเซอร์และคีย์บอร์ด มีค่าความแม่นยำสูงกว่าข้อมูลคีย์บอร์ด ซึ่งได้ค่า Statistical Power ที่ 0.9441 คือ ค่าเฉลี่ยของข้อมูลคีย์บอร์ด และข้อมูลเซ็นเซอร์และคีย์บอร์ด มีความแตกต่างกันจริง, ความผิดพลาดแบบที่ 2 ที่ 0.0559 คือ ค่าเฉลี่ยของข้อมูลคีย์บอร์ดและข้อมูลเซ็นเซอร์และคีย์บอร์ด มีความแตกต่างกันแต่กลับสรุปว่าไม่มีความแตกต่างกัน และความผิดพลาดแบบที่ 1 ที่ 0.05 คือ ค่าเฉลี่ยของข้อมูลคีย์บอร์ด และข้อมูลเซ็นเซอร์และคีย์บอร์ด ไม่มีความแตกต่างกันแต่กลับสรุปว่ามีความแตกต่างกัน

ตารางที่ 4.5 ผลการทดลองของการเรียนรู้เครื่องกับข้อมูลหลายแบบ

	Statistical Power	**Type II Error**
ข้อมูลสถิติ-ข้อมูลเซ็นเซอร์และคีย์บอร์ด	0.9726	0.0274
ข้อมูลคีย์บอร์ด-ข้อมูลเซ็นเซอร์และคีย์บอร์ด	0.9441	0.0559

จากการวิเคราะห์ข้องต้นจึงพอสรุปได้ว่า ข้อมูลเซ็นเซอร์และคีย์บอร์ด นั้นสามารถนำมาใช้งานได้ดีที่สุด เนื่องจากข้อมูลเซ็นเซอร์และคีย์บอร์ดเป็นข้อมูลที่มีค่าเฉลี่ยสูงสุดเมื่อเทียบกับข้อมูลอื่นๆ

จากผลการทดลองในตารางที่ 4.1, 4.2, 4.3 และ 4.4 พบว่าโมเดล LSTM สามารถทำงานได้ดีกว่าโมเดล Standard กับข้อมูลทั้ง 4 แบบ อย่างมีนัยยะสำคัญ

โมเดล LSTM กับข้อมูลการเรียนรู้แบบเซ็นเซอร์และคีย์บอร์ดร่วมกัน ให้ผลการทดลองที่ดีกว่าข้อมูลแบบสถิติและข้อมูลคีย์บอร์ดอย่างเดียว ดังที่แสดงในตารางที่ 4.5 และที่ได้ทำการอธิบายไว้ข้างต้นแล้ว

สรุปได้ว่าโมเดล LSTM กับข้อมูลเซ็นเซอร์และคีย์บอร์ดนั้นมีประสิทธิภาพดีที่สุดเมื่อเทียบกับโมเดลอื่นและข้อมูลแบบอื่น จึงได้ทำการปรับปรุงโมเดล LSTM เพื่อหาโมเดล LSTM ที่เหมาะสมมีประสิทธิภาพดีขึ้นซึ่งจะแสดงในผลการทดลองถัดไปต่อจากนี้

4.5 ผลการทดลองของขั้นตอนวิธีการเรียนรู้เครื่องแบบ Feature Extraction

ผลการทดลองนี้เป็นการนำ Feature Extraction LSTM model กับข้อมูลเซ็นเซอร์และคีย์บอร์ด 30 ครั้ง ซึ่งเป็นข้อมูลเดียวกับผลการทดลองที่ 4.4 โมเดลทั้ง 3 นี้เป็นโมเดล LSTM รูปแบบหนึ่งเพื่อเพิ่มประสิทธิภาพการทำงานกับข้อมูลเซ็นเซอร์และคีย์บอร์ด

ตารางที่ 4.6 ผลการทดลองของการเรียนรู้เครื่องแบบ Feature Extraction

	Feature Extraction#1	**Feature Extraction#2**	**2-Feature Extraction**
ค่าเฉลี่ย	88.7767	82.9891	85.6054
ส่วนเบี่ยงเบนมาตรฐาน	2.0447	1.7723	1.4390
ค่าน้อยสุด	85.1003	79.9956	83.0332
ค่าสูงสุด	93.035	86.7835	88.335

จากตารางที่ 4.5 การนำข้อมูลเชิงสถิติมาใช้แบบ K-Fold Cross Validation กับโมเดลการเรียนรู้เครื่องแบบ LSTMs ด้วยวิธีการ Feature Extraction ให้ผลการทดลองที่ดีขึ้นโดยโมเดลแบบ Feature Extraction แบบที่ 1 ได้ค่าเฉลี่ยความแม่นยำที่ 88.7767% จากวิธี Feature Extraction แบบที่ 2 ได้ค่าเฉลี่ยความแม่นยำที่ 82.9891% และจากวิธี 2-Feature Extraction ได้ค่าเฉลี่ยความแม่นยำที่ 85.6054%

4.6 ผลการทดลองเปรียบเทียบโมเดล

ผลการทดลองมีหลายโมเดลจึงเปรียบเทียบสรุปให้ทราบถึงผลการทดลองและเลือกโมเดลที่ดีที่สุด โดยเป็นการเปรียบเทียบผลของการใช้ข้อมูลเชิงสถิติซึ่งเป็นผลการทดลองการเรียนรู้ ของเครื่องในตารางที่ 4.4 และ 4.5 โดยวิธีทั้ง 5 โมเดลคือ Standard, LSTM, Feature Extraction แบบที่ 1, Feature Extraction แบบที่ 2 และ 2-Feature Extraction ทำการทดสอบพร้อมกันด้วยการนำข้อมูลเชิงสถิติมาใช้แบบ K-Fold Cross Validation กับโมเดลการเรียนรู้เครื่องเหล่านี้

ตารางที่ 4.7 ผลการทดลองของการเรียนรู้เครื่องแบบหลายโมเดล

Models	Average	Standard Deviation	Max.	Min.
Standard	76.3452	1.8984	81.016	74.026
LSTMs	80.5820	1.2463	83.0014	77.5416
Feature Extraction แบบที่ 1	88.7767	2.0447	93.035	85.1003
Feature Extraction แบบที่ 2	82.9891	1.7723	86.7835	79.9956
2-Feature Extraction	85.6054	1.4390	88.335	83.0332

จากตารางที่ 4.6 พบว่า โมเดลแบบ feature extraction แบบที่ 1 สามารถทำได้ดีที่สุด จึงเลือกเป็นโมเดลสำหรับการใช้งาน จากผลการทดลองที่ 4.4 ทราบอยู่แล้วว่า LSTM สามารถสรุปได้แล้วว่าทำงานได้ดีกว่า Standard จึงนำ LSTM และข้อมูลชุดนี้ไปใช้กับโมเดลอื่นๆ ซึ่งคือโมเดลมีลักษณะดังภาพที่ 3.15, 3.16 และ 3.17 ซึ่งจะแสดงการทดสอบที (t-Test) กับวิธี LSTM, Feature Extraction แบบที่ 1, Feature Extraction แบบที่ 2 และ 2-Feature Extraction เรียนรู้ของเครื่องทั้ง 4 นั้นเป็นโมเดลหลักในการวิเคราะห์ เพื่อทดสอบผลทางสถิติดังต่อไปนี้

4.6.1 การทดสอบการแจกแจงแบบปกติ

การทดสอบการแจกแจกแบบปกติ ด้วย The Shapiro-Wilk Test เนื่องจากข้อมูลกลุ่มที่มีนั้นมีขนาดเล็ก

- Shapiro-Wilk > 0.05 ข้อมูลมีการแจกแจงปกติ
- Shapiro-Wilk < 0.05 ข้อมูลมีการแจกแจงไม่ปกติ

จากตารางที่ 4.1 และ 4.5 ค่าผลการทดลองของ LSTM, Feature Extraction แบบที่ 1, Feature Extraction แบบที่ 2 และ 2-Feature Extraction นำข้อมูลกลุ่มนี้มาทำการทดสอบการแจกแจงแบบปกติ

Tests of Normality

	Kolmogorov-Smirnov[a]			Shapiro-Wilk		
	Statistic	df	Sig.	Statistic	df	Sig.
lstm	.138	30	.153	.960	30	.313
Fe1	.123	30	.200*	.968	30	.493
Fe2	.147	30	.097	.948	30	.146
2Fe	.139	30	.143	.956	30	.241

*. This is a lower bound of the true significance.

a. Lilliefors Significance Correction

ภาพที่ 4.1 ผลการทดสอบการแจกแจงแบบปกติ

จากภาพที่ 4.1 ดังข้างต้นพบว่า วิเคราะห์ค่า Sig. ของ Shapiro-Wilk ของทั้ง 3 โมเดล ได้ดังนี้

- 1 : LSTM ได้ 0.313 > 0.05
- 2 : Feature Extraction แบบที่ 1 ได้ 0.493 > 0.05
- 3 : Feature Extraction แบบที่ 2 ได้ 0.146 > 0.05
- 4 : 2-Feature Extraction ได้ 0.241 > 0.05

จึงสรุปได้ว่าข้อมูลของทั้ง 4 โมเดลมีการแจกแจงแบบปกติ

4.6.2 การทดสอบที

การทดสอบสมมติฐานทางสถิติที่ต้องการเปรียบเทียบค่าเฉลี่ยของตัวอย่างสมมติฐาน เมื่อทราบแล้วว่าข้อมูลทั้ง 3 กลุ่มมีการแจกแจงแบบปกติแล้ว จึงสามารถทำการทดสอบ Independent Sample t-Test โดยทำการทดสอบ 2 ครั้ง คือ ระหว่าง LSTM, Feature Extraction แบบที่ 1, Feature Extraction แบบที่ 2 และ 2-Feature Extraction โดยมีค่า α=0.05 และทำการตั้งสมมติฐานว่า

ค่าเฉลี่ยของโมเดลนั้นมีความแตกต่างกัน ดังนี้

$$H_0: \mu_1 - \mu_2 = 0$$

$$H_a: \mu_1 - \mu_2 \neq 0$$

μ_1 คือค่าเฉลี่ยประชากรของข้อมูลที่ 1

μ_2 คือค่าเฉลี่ยประชากรของข้อมูลที่ 2

ข้อมูลที่ใช้ในการทดสอบนั้นมีจำนวนเท่ากันจึงใช้ Equal Variance t-Test

การทดสอบของ LSTM กับ Feature Extraction แบบที่ 2

สมมิติฐานว่า ค่าเฉลี่ยของ LSTM ไม่เท่ากันกับ Feature Extraction แบบที่ 2

$$H_0: \mu_1 - \mu_2 = 0$$

$$H_a: \mu_1 - \mu_2 \neq 0$$

μ_1 คือค่าเฉลี่ยประชากรของข้อมูลโมเดลที่ 1 : LSTM

μ_2 คือค่าเฉลี่ยประชากรของข้อมูลโมเดลที่ 2 : LSTM แบบ Feature Extraction แบบที่ 2

Group Statistics

	define	N	Mean	Std. Deviation	Std. Error Mean
lstm-fe2	0	30	80.582013	1.2462975	.2275417
	1	30	82.989177	1.7722720	.3235711

Independent Samples Test

		Levene's Test for Equality of Variances		t-test for Equality of Means						
									95% Confidence Interval of the Difference	
		F	Sig.	t	df	Sig. (2-tailed)	Mean Difference	Std. Error Difference	Lower	Upper
lstm-fe2	Equal variances assumed	5.105	.028	-6.085	58	.000	-2.4071633	.3955673	-3.1989774	-1.6153493
	Equal variances not assumed			-6.085	52.046	.000	-2.4071633	.3955673	-3.2009105	-1.6134162

ภาพที่ 4.2 ผลการทดสอบ t-Test ระหว่าง LSTM และ Feature Extraction แบบที่ 2

จากภาพที่ 4.2 ดูค่า Sig.(2-taled) พบว่าค่า p = 0.000000049264840 แบบ 2-tailed ซึ่ง p < α จึงปฏิเสธ H_0 และยอมรับ H_a ว่าค่าเฉลี่ยของแต่ละโมเดลแตกต่างกัน จากค่าเฉลี่ยของ LSTM = 80.58% และ Feature Extraction แบบที่ 2 = 82.99% จึงสรุปได้ว่า LSTM Feature Extraction แบบที่ 2 มีค่าเฉลี่ยมากกว่า

คำนวณหาค่า Statistical Power และค่าความผิดพลาดแบบที่ 2 (Type II Error)

$$z = \frac{\bar{X} - \mu_0}{\sigma/\sqrt{n}}$$

$$\bar{X} = \mu_0 + z\,\sigma/\sqrt{n}$$

เมื่อเรารู้ว่าค่าเฉลี่ยสมมติฐานและค่าความผิดพลาดแบบที่ 1 (Type I Error)

$$H_0 : \mu = 80.5820$$

$$H_a : \mu = 82.9892$$

$$\alpha = 0.05$$

ปฏิเสธ H_0 เมื่อ

$$z \leq -1.96 : \bar{X} \leq 80.1360 \quad \text{หรือ} \quad z \geq 1.96 : \bar{X} \geq 81.028$$

Statistical Power = $P(\text{Reject } H_0, \mu_a) = 1 - \beta$

เมื่อ $\bar{X} = 81.028$ จะได้

$$z = -1.60988$$

ค่าจากตารางพื้นที่ใต้โค้งปกติจะได้

$$\text{Statistical Power} = \mathrm{P}(\text{ Reject } \mathrm{H}_0, \mu_a) = \mathrm{P}(\mathrm{z} \geq -1.61) = 0.8686$$

$$\text{Type II Error} = \beta = 0.1314$$

จึงได้ค่า Statistical Power ที่ 0.8686 คือ ค่าเฉลี่ยของวิธี LSTM และ Feature Extraction แบบที่ 2 มีความแตกต่างกันจริง, ความผิดพลาดแบบที่ 2 ที่ 0.1314 คือ ค่าเฉลี่ยของวิธี LSTM และ Feature Extraction แบบที่ 2 มีความแตกต่างกันแต่กลับสรุปว่าไม่มีความแตกต่างกัน และความผิดพลาดแบบที่ 1 ที่ 0.05 คือ ค่าเฉลี่ยของวิธี LSTM และ Feature Extraction แบบที่ 2 ไม่มีความแตกต่างกันแต่กลับสรุปว่ามีความแตกต่างกัน

การทดสอบของ Feature Extraction แบบที่ 2 กับ 2-Feature Extraction

สมมิติฐานว่า ค่าเฉลี่ยของ Feature Extraction แบบที่ 2 ไม่เท่ากันกับ 2-Feature Extraction

$$H_0: \mu_1 - \mu_2 = 0$$

$$H_a: \mu_1 - \mu_2 \neq 0$$

μ_1 คือค่าเฉลี่ยประชากรของข้อมูลโมเดลที่ 1 : LSTM แบบ Feature Extraction แบบที่ 2

μ_2 คือค่าเฉลี่ยประชากรของข้อมูลโมเดลที่ 2 : LSTM แบบ 2-Feature Extraction

Group Statistics

	define	N	Mean	Std. Deviation	Std. Error Mean
2fe-Fe2	0	30	82.989177	1.7722720	.3235711
	1	30	85.605487	1.4390056	.2627253

Independent Samples Test

		Levene's Test for Equality of Variances		t-test for Equality of Means						
									95% Confidence Interval of the Difference	
		F	Sig.	t	df	Sig. (2-tailed)	Mean Difference	Std. Error Difference	Lower	Upper
2fe-Fe2	Equal variances assumed	2.233	.140	-6.277	58	.000	-2.6163100	.4168007	-3.4506273	-1.7819927
	Equal variances not assumed			-6.277	55.653	.000	-2.6163100	.4168007	-3.4513770	-1.7812430

ภาพที่ 4.3 ผลการทดสอบ t-Test ระหว่าง Feature Extraction#2 และ 2-Feature Extraction

จากภาพที่ 4.2 ดูค่า Sig.(2-taled) พบว่าค่า p = 0.000000023695491 แบบ 2-tailed ซึ่ง p < α จึงปฏิเสธ H_0 และยอมรับ H_a ว่าค่าเฉลี่ยของแต่ละโมเดลแตกต่างกัน จากค่าเฉลี่ยของ Feature Extraction แบบที่ 2 = 82.9892 และ 2-Feature Extraction = 85.6055 จึงสรุปได้ว่า LSTM 2-Feature Extraction มีค่าเฉลี่ยมากกว่า

คำนวณหาค่า Statistical Power และค่าความผิดพลาดแบบที่ 2 (Type II Error)

$$z = \frac{\bar{X} - \mu_0}{\sigma/\sqrt{n}}$$

$$\bar{X} = \mu_0 + z\,\sigma/\sqrt{n}$$

เมื่อเรารู้ว่าค่าเฉลี่ยสมมติฐานและค่าความผิดพลาดแบบที่ 1 (Type I Error)

$$H_0 : \mu = 82.9892$$

$$H_a : \mu = 85.6055$$

$$\alpha = 0.05$$

ปฏิเสธ H_0 เมื่อ

$$z \leq -1.96 : \bar{X} \leq 82.35498 \text{ หรือ } z \geq 1.96 : \bar{X} \geq 83.62338$$

Statistical Power = $P(\text{Reject } H_0, \mu_a) = 1 - \beta$

เมื่อ $\bar{X} = 83.62338$ จะได้

$$z = -2.2589$$

ค่าจากตารางพื้นที่ใต้โค้งปกติจะได้

Statistical Power = $P(\text{Reject } H_0, \mu_a) = P(z \geq -2.589) = 0.9726$

Type II Error = $\beta = 0.0838$

จึงได้ค่า Statistical Power ที่ 0.9726 คือ ค่าเฉลี่ยของวิธี Feature Extraction แบบที่ 2 และ 2-Feature Extraction มีความแตกต่างกันจริง, ความผิดพลาดแบบที่ 2 ที่ 0.0838 คือ ค่าเฉลี่ยของวิธี Feature Extraction แบบที่ 2 และ 2-Feature Extraction มีความแตกต่างกันแต่กลับสรุปว่าไม่มีความแตกต่างกัน และความผิดพลาดแบบที่ 1 ที่ 0.05 คือ ค่าเฉลี่ยของวิธี Feature

Extraction แบบที่ 2 และ 2-Feature Extraction ไม่มีความแตกต่างกันแต่กลับสรุปว่ามีความแตกต่างกัน

การทดสอบของ 2-Feature Extraction และ Feature-Extraction แบบที่1

สมมิติฐานว่า ค่าเฉลี่ยของ LSTM แบบ 2-Feature Extraction ไม่เท่ากันกับ LSTM แบบ Feature-Extraction แบบที่ 1

$$H_0: \mu_1 - \mu_2 = 0$$

$$H_a: \mu_1 - \mu_2 \neq 0$$

μ_1 คือค่าเฉลี่ยประชากรของข้อมูลโมเดลที่ 2 : LSTM แบบ 2-Feature Extraction

μ_2 คือค่าเฉลี่ยประชากรของข้อมูลโมเดลที่ 3 : LSTM แบบ Feature-Extraction แบบที่ 1

Group Statistics

	define	N	Mean	Std. Deviation	Std. Error Mean
2Fe-Fe1	0	30	85.605487	1.4390056	.2627253
	1	30	88.776707	2.0447320	.3733153

Independent Samples Test

		Levene's Test for Equality of Variances		t-test for Equality of Means						
									95% Confidence Interval of the Difference	
		F	Sig.	t	df	Sig. (2-tailed)	Mean Difference	Std. Error Difference	Lower	Upper
2Fe-Fe1	Equal variances assumed	4.426	.040	-6.947	58	.000	-3.1712200	.4564963	-4.0849966	-2.2574434
	Equal variances not assumed			-6.947	52.068	.000	-3.1712200	.4564963	-4.0872185	-2.2552215

ภาพที่ 4.4 ผลการทดสอบ t-Test ระหว่าง 2-Feature Extraction และ Feature-Extraction#1

จากภาพที่ 4.3 ดูค่า Sig.(2-taled) พบว่าค่า p = 0.00000000179974083 แบบ2-tailed ซึ่ง p < α จึงปฏิเสธ H_0 และยอมรับ H_a ว่าค่าเฉลี่ยของแต่ละโมเดลแตกต่างกัน จากค่าเฉลี่ยของ LSTM แบบ 2-Feature Extraction = 85.6055 และ LSTM แบบ Feature-Extraction แบบที่ 1= 88.7767 จึงสรุปได้ว่า LSTM แบบ Feature-Extraction แบบที่ 1 มีค่าเฉลี่ยมากกว่า

คำนวณหาค่า Statistical Power และค่าความผิดพลาดแบบที่ 2 (Type II Error)

$$z = \frac{\overline{X} - \mu_0}{\sigma/\sqrt{n}}$$

$$\bar{X} = \mu_0 + z\,\sigma/\sqrt{n}$$

เมื่อเรารู้ว่าค่าเฉลี่ยสมมติฐานและค่าความผิดพลาดแบบที่ 1 (Type I Error)

$$H_0 : \mu = 85.6055$$

$$H_a : \mu = 88.7767$$

$$\alpha = 0.05$$

ปฏิเสธ H_0 เมื่อ

$$z \leq -1.96 : \bar{X} \leq 85.09055 \quad \text{หรือ} \quad z \geq 1.96 : \bar{X} \geq 86.12043$$

$$\text{Statistical Power} = P(\text{Reject } H_0, \mu_a) = 1 - \beta$$

เมื่อ $\bar{X} = 86.1204$ จะได้

$$z = -1.8028$$

ค่าจากตารางพื้นที่ใต้โค้งปกติจะได้

$$\text{Statistical Power} = P(\text{Reject } H_0, \mu_a) = P(z \geq -1.80) = 0.9032$$

$$\text{Type II Error} = \beta = 0.0968$$

จึงได้ค่า Statistical Power ที่ 0.9032 คือ ค่าเฉลี่ยของวิธี LSTM แบบ 2-Feature Extraction และ LSTM แบบ Feature-Extration#1 มีความแตกต่างกันจริง, ความผิดพลาดแบบที่ 2 ที่ 0.0968 คือ ค่าเฉลี่ยของวิธี LSTM แบบ 2-Feature Extraction และ LSTM แบบ Feature-Extraction แบบที่ 1 มีความแตกต่างกันแต่กลับสรุปว่าไม่มีความแตกต่างกัน และความผิดพลาดแบบที่ 1 ที่ 0.05 คือ ค่าเฉลี่ยของวิธี LSTM แบบ 2-Feature Extraction และ LSTM แบบ Feature-Extraction แบบที่ 1 ไม่มีความแตกต่างกันแต่กลับสรุปว่ามีความแตกต่างกัน

จากการทดสอบ t-Test แบบ Equal Variances และการหาค่า Statistical Power นั้นทำให้ทราบว่าโมเดล LSTM แบบ Feature-Extraction แบบที่ 1 นั้นมีค่าเฉลี่ยความแม่นยำมากกว่าของโมเดล LSTM แบบ 2-Feature Extraction, โมเดลLSTM แบบ Feature-Extraction แบบที่ 2 และโมเดล LSTM

บทที่ 5
วิเคราะห์และสรุปผลการวิจัย

เอกสารในบทนี้เป็นบทสุดท้ายประกอบด้วยสองส่วนหลัก ได้แก่ ส่วนแรกเป็นการวิเคราะห์และสรุปผลการดำเนินงานของงานวิจัยนี้และอภิปรายผลการดำเนินงานที่ได้ ในส่วนสุดท้ายเป็นข้อจำกัดและรวมถึงแนวทางวิจัยต่อไปในอนาคต ดังรายละเอียดต่อไปนี้

5.1 วิเคราะห์และสรุปผล

พบว่าการเลือกโมเดลมีผลต่อการทำงานกับข้อมูล การใช้ Feature Extraction LSTM Model ได้ผลดีกว่ากับทั้ง LSTM Model และ Simple Fully-Connected Neuron Model ตามลำดับ การเลือกโมเดลที่เหมาะสมกับการแก้ปัญหามีผลอย่างมากกับความสำเร็จในการดำเนินงาน

ผลการทำงานของโมเดลกับข้อมูลที่ใช้มีความเกี่ยวข้องกัน คือ จำนวนข้อมูลและลักษณะของข้อมูล กล่าวคือ ยิ่งข้อมูลมากยิ่งได้ผลการทำงานที่แม่นยำ ยิ่งข้อมูลมีความสมดุล หมายถึง ข้อมูลของทั้งกลุ่มผู้ป่วยโรคพาร์กินสันและผู้ที่ไม่ป่วยมีจำนวนใกล้เคียงหรือเท่ากันยิ่งดี แต่หากข้อมูลไม่เท่ากันก็มีวิธีการแก้ปัญหานี้เช่นกัน ในการวิจัยนี้มีข้อมูลจากการเก็บข้อมูลอยู่ 3 แบบ คือ ข้อมูลเชิงสถิติ ข้อมูลคีย์บอร์ด และข้อมูลเซ็นเซอร์ การจัดการและปรับใช้ข้อมูลจึงเป็นส่วนสำคัญให้การดำเนินงานสัมฤทธิ์ผล

การวิจัยนี้ได้โมเดลของการเรียนรู้เครื่องที่เป็นโครงข่ายประสาทเทียมแบบวนซ้ำแบบ LSTM ที่มีความแม่นยำอยู่ที่ 88.78 % โดยโมเดลนี้สามารถนำไปใช้ในการช่วยทำนายผลต่อไปได้ ในทุกการทดลองจะมีการเก็บโมเดลแยกไว้เมื่อการทดลองเสร็จสิ้นจึงเลือกนำโมเดลที่ได้ค่าความแม่นยำสูงสุด

การวิจัยนี้ทำให้การใช้การเรียนรู้เครื่องในการวินิจฉัยโรคพาร์กินสันสามารถทำได้เพื่อช่วยแพทย์ในการวินิจฉัยเบื้องต้นและเพื่อลดภาระการวินิจฉัยของแพทย์ผู้เชี่ยวชาญ โดยการใช้การเรียนรู้เครื่องในการเรียนรู้ข้อมูลและทำการทำนายผลของโรคพาร์กินสัน

5.2 ข้อจำกัดและแนวทางการดำเนินงานต่อไป

ข้อจำกัดในงานนี้คือข้อมูล โดยสามารถจำแนกและวิเคราะห์ข้อจำกัดของข้อมูล ในการดำเนินงานวิจัยนี้ ได้ดังนี้

จำนวนข้อมูลตัวอย่างที่มีจำนวนน้อย ทำให้การเรียนรู้เครื่องทำได้อย่างจำกัด จึงต้องให้ความสนใจกับวิธีการหาโมเดลการเรียนรู้เครื่องและการพัฒนาโมเดลการเรียนรู้เครื่องให้มีประสิทธิภาพภายใต้ข้อมูลจำกัด

ลักษณะของตัวข้อมูลก็มีความแตกต่างกันที่ทำให้เป็นอุปสรรคในการดำเนินงาน เมื่อทำการจัดการข้อมูลแล้ว พบว่า การทำความเข้าใจที่มาของข้อมูลมีส่วนสำคัญ คือ ผู้ป่วยนั่นมีอาการแตกต่างกัน ความหนักของอาการ และความเด่นของอาการ กล่าวคือ ผู้ป่วยมีอาการของมือสองข้างไม่เท่ากัน มีข้างที่เด่นและข้างที่อ่อน ข้างที่อ่อนนั้นในบางรายมีความใกล้เคียงกับโรคอื่นหรือคนธรรมดา ทำให้ข้อมูลตัวอย่างที่ได้มาจึงมีบางส่วนที่ข้อมูลคนไข้นั้นใกล้เคียงกับคนธรรมดาหรือคนที่มีอาการป่วยอย่างอื่นที่ไม่ใช่โรคพาร์กินสัน ซึ่งมีส่วนสำคัญกับประสิทธิภาพในการดำเนินงาน

แนวทางการดำเนินงานในอนาคต คือ การเก็บข้อมูลเพิ่มเติมให้ได้มากขึ้น เพื่อความสะดวกและประสิทธิภาพในการดำเนินงาน โดยจะเน้นการพัฒนาการเก็บข้อมูลเชิงสถิติ ข้อมูลคีย์บอร์ดและข้อมูลเซ็นเซอร์ เนื่องจากเป็นข้อมูลที่สามารถเก็บได้ยิ่งมีมากยิ่งเป็นผลดีการใช้เครื่องมือในการเก็บข้อมูลมีความสำคัญ ข้อมูลเซ็นเซอร์จึงเป็นส่วนสำคัญ เพราะสามารถใช้ในการวิเคราะห์ได้ดีและเป็นข้อมูลที่บ่งชี้อาการใกล้เคียงกับความรู้ของโรคพาร์กินสันที่สุด

บรรณานุกรม

1. Ozcift, A. and A. Gulten, "Classifier ensemble construction with rotation forest to improve medical diagnosis performance of machine learning algorithms," Comput Methods Programs Biomed, 2011, 104(3): p. 443-51.
2. P. R. Davidson, R.D.J., and M. T. R. Peiris, "Detecting Behavioral Microsleeps using EEG and LSTM Recurrent Neural Networks," the 2005 IEEE Engineering in Medicine and Biology 27th Annual Conference, 2005.
3. Zachary C. Lipton, D.C.K., Charles Elkan, and Randall and Wetzel, "Learning to Diagnose with LSTM Recurrent Neural Networks," arXiv:1511.03677v7, 2017.
4. Nazari, N., et al., "Multi-level Binarized LSTM in EEG Classification for Wearable Devices," in 2020 28th Euromicro International Conference on Parallel, Distributed and Network-Based Processing (PDP), 2020, p. 175-181.
5. Baytas, I.M., et al., "Patient Subtyping via Time-Aware LSTM Networks," in Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 2017, p. 65-74.
6. Daniel Soutner , a.L.M., "Application of LSTM Neural Networks in Language Modelling," Springer-Verlag Berlin Heidelberg, 2013, p. 105-112.
7. Gamboa, J., "Deep Learning for Time-Series Analysis," arXiv:1701.01887v1, 2017.
8. Palangi, H., et al., "Deep Sentence Embedding Using Long Short-Term Memory Networks," Analysis and Application to Information Retrieval, IEEE/ACM Transactions on Audio, Speech, and Language Processing, 2016. 24(4): p. 694-707.
9. Hasim Sak, A.S., and Francoise Beaufays, "Long Short-Term Memory Recurrent Neural Network Architectures for Large Scale Acoustic Modeling," Interspeech, 2014.
10. Xiaopeng Xi, E.K., Christian Shelton, Li Wei, and Chotirat Ann Ratanamahatana, "Fast Time Series Classification Using Numerosity Reduction," 23rd International Conference on Machine Learning, 2006.
11. Wonmin Byeon, T.M.B., Federico Raue, and Marcus Liwicki, "Scene Labeling with LSTM Recurrent Neural Networks," IEEE Conference on Computer Vision and Pattern Recognition, 2015.
12. Karim, F., et al., "LSTM Fully Convolutional Networks for Time Series Classification," IEEE Access, 2018. 6: p. 1662-1669.

13. Shivam Chandhok, H.J., S. J. Darak, and A. V. and Subramanyam, "LSTM Guided Modulation Classification and Experimental Validation for Sub-Nyquist Rate Wideband Spectrum Sensing," 11th International Conference on Communication Systems and Networks, 2019.
14. Geurts, P., "Pattern Extraction for Time Series Classification," PKDD, 2019: p. 115-127.
15. Chen, H.-L., et al., "An efficient diagnosis system for detection of Parkinson's disease using fuzzy k-nearest neighbor approach," Expert Systems with Applications, 2013. 40(1): p. 263-271.
16. Chen, H.-L., et al., "An efficient hybrid kernel extreme learning machine approach for early diagnosis of Parkinson's disease," Neurocomputing, 2016. 184: p. 131-144.
17. Chao Che, C.X., Jian Liang, Bo Jin, Jiayu Zhou, and Fei Wang, "An RNN Architecture with Dynamic Temporal Matching for Personalized Predictions of Parkinson's Disease," SIAM International Conference on Data Mining, 2017.
18. Ram Deepak Gottapu, a.C.H.D., "Analysis of Parkinson's Disease Data," Procedia Computer Science 140, 2018: p. 334-341.
19. Tagaris, A., D. Kollias, and A. Stafylopatis, "Assessment of Parkinson's Disease Based on Deep Neural Networks," in Engineering Applications of Neural Networks. 2017. p. 391-403.
20. Zhang, X., et al., "Data-Driven Subtyping of Parkinson's Disease Using Longitudinal Clinical Records: A Cohort Study," Sci Rep, 2019. 9(1): p. 797.
21. David Gil A., a.M.J.B., "Diagnosing Parkinson by using Artificial Neural Networks and Support Vector Machines," Global Journal of Computer Science and Technology 9, 2009.
22. Bhat, S., et al., "Parkinson's disease: Cause factors, measurable indicators, and early diagnosis," Comput Biol Med, 2018. 102: p. 234-241.
23. Arora, S., et al., "Smartphone motor testing to distinguish idiopathic REM sleep behavior disorder, controls, and PD," Neurology, 2018. 91(16): p. 1528-1538.
24. Zhan, A., et al., "Using Smartphones and Machine Learning to Quantify Parkinson Disease Severity: The Mobile Parkinson Disease Score," JAMA Neurol, 2018. 75(7): p. 876-880.
25. Das, R., "A comparison of multiple classification methods for diagnosis of Parkinson disease," Expert Systems with Applications, 2010. 37(2): p. 1568-1572.

ภาคผนวก

บทความที่เผยแพร่ในการประชุมวิชาการ 2021 18th International Conference on Electrical Engineering/Electronics, Computer, Telecommunications and Information Technology (ECTI-CON)

- ECTI-CON 2021 – *Smart Electrical Systems & Technology*

An Implementation of Machine Learning for Parkinson's Disease Diagnosis

Hassapol Thummikarat
Department of Computer Engineering,
Chulalongkorn University,
Bangkok, Thailand.
e-mail : luping49030@hotmail.com

Prabhas Chongstitvatana
Department of Computer Engineering,
Chulalongkorn University,
Bangkok, Thailand.
e-mail : prabhas.c@chula.ac.th

***Abstract*— Machine learning is widely used in the medical applications. Parkinson's disease is a nervous system disorder which commonly causes tremors, but the disorder also commonly causes stiffness or slowing of movement. These symptoms are not only caused by Parkinson's disease but also the other movement disorder sickness. The doctors who are specialist in the Parkinson's disease can simply diagnose the tremors, which usually be hand muscle tremor of the patient. But conversely, the out-patient-department doctors find that it is difficult to diagnose those symptoms. This work proposes the use of machine learning for the Parkinson's disease data to assist the physician diagnosis. The Long Short-Term memory network is suitable for the data collected by a specialist. The result shows that the proposed method has 73% accuracy in early identifying the patient with Parkinson's disease.**

Keywords—Long Short-Term memory (LSTM), Parkinson's disease diagnosis, machine learning, neural networks , binary classification

I. Introduction

The Parkinson's disease (PD) is a brain and nervous disorder, its symptoms are obviously tremor, stiffness, and slow movement. It is vital for patients to be diagnosed as early as possible, in order that the patients are treated since the early stage of the sickness to prolong their life. Outpatient department (OPD) doctors are facing the diagnosis problem of tremors which can be causing by PD or the other diseases, on the other hand, the PD specialists simply manage to judge the diagnosis of tremors. But it is not possible for PD doctors to be at the OPD, thus the OPD doctors require a tool to improve their diagnosis before sending all those patients with tremors to PD department. In order to assist the doctor, this work proposes Machine learning method to help with the diagnosis. There are two neural network models to be compared, the simple neuron model and the LSTMs model.

Machine learning algorithms have been widely used for medical data analysis. PD and heat disease data was used to classify the difference between these two diseases, as in [1]. An application of neural network with EEG, video and tracking data chose LSTM to solve the problem, as in [2]. To establish the strategy for the intensive care unit (ICU), Recurrent Neural networks (RNNs) and Long Short-Term Memory (LSTM) are used with Electronic Health Record (EHR) in the clinic measurement and management [3]. The classification method for EEG used LSTMs to classify the data from wearable devices in real-time [4]. Various disease data is time series data and LSTMs are used to cluster patients [5].

The LSTMs are an improvement of RNNs to overcome the vanishing gradient problem. RNNs have difficulties with time dependencies. LSTMs have advantage to overcome this problem, as in [6]. Time-series analysis, using deep learning shown the application of the NNs to time dependency information, as in [7], [8] and [9]. Time-series classification required the matching data sequence, as in [10]. LSTMs were used for imaging classification, as in [11]. Time-series classification could use the LSTMs for classifying the sequences, as in [12], [13] and [14].

The applications of the Parkinson's Disease data for diagnosis are interesting and challenging in both engineering and medical development over years. Using fuzzy method was an alternative approach to the PD diagnosis, as in [15]. The Extreme Leaning Machine (ELM) was used as a hybrid kernel ELM and its potential was sufficient, as in [16]. RNNs could predict PD, as in [17]. Deep Learning (DL) could be used to obtain the classifying and predicting the PD in case of large scale data, as in [18] and [19]. LSTMs were used to receive the PD subtypes from the clinical records, as in [20]. A simple ML method called Support Vector Machine (SVM) generating a solid outcome, as in [21]. Early diagnosis for PD patients used ML to predict from various data sources, as in [22]. Smartphone applications were introduced to apply AI, as in [23] and [24]. The NNs method was the best application option for this type of data, as in [25].

This work describes an implementation of ML to improve the diagnosis of PD. The implementation consists of data preparing, neural network modelling, parameters tuning and experimenting the model.

II. Implemetation

A. Data Preparation

The raw data is contained of records from PD patients and non PD persons called control group, the records are from a keyboard and sensors collecting through a controller as shown in Fig. 1.

Figure 1. Tools for collecting PD data.

The keyboard data and the gyroscope and accelerometer sensors are collected at different frequency and have different features. Since the sensor has more sample rate than

978-1-6654-0382-5/21/$31.00 ©2021 IEEE

the keyboard, thus it is necessary to rescale and resample the keyboard data. The data is normalized as following.

- correct key : 1
- incorrect key : -1
- burst key : -0.5
- repeat key : 0.5
- double press : -0.25

These rules generate keyboard data which are convenient to analyze.

```
Algorithm for creating new keyboard data:
sensor data index = 0
keyboard data index = 0
new keyboard data = []
loop sensor index in range of sensor data length :
    check keyboard time >= sensor time:
        use the rules for keyboard feature data
        ++keyboard index
    else : use 0 as keyboard feature data
    ++sensor index
```

The above algorithm generates a new keyboard data, with frequency equal to the senser data, and the data is normalized. At 40 Hz, the both data of keyboard and sensor are equal length.

For supervised learning, features and outputs forming pairs of input and output sequences. The data are organized into three sets as follows.

- Dataset-1 : keyboard dataset
- Dataset-2 : sensor dataset
- Dataset-3 : sensor and keyboard dataset

The input shape of LSTMs network must be three dimensions meaning which the datasets from data preparation process need to be reshape as input shape of 3D data (samples , time steps , features) for the LSTMs model or input dimension (features) for the simple neuron model.

All data of the 2D datasets are reshaped into 3D datasets as the (samples × time steps × features) format The collected data consists of 100 medical records, Each record consists of 3 records for a test which there are 2 test for both hands, thus the total size is 1200 samples. The number of time steps for both the keyboard and sensor dataset is 500 time steps. The number of features depends on individual dataset's features. Dataset-1's input shape is (1200 × 500 × 2), Dataset-2's input shape is (1200 × 500 × 12), and Dataset-3's input shape is (1200 × 500 × 13).

B. Neural Networks Model

1) The simple neuron model: is created as follows.

- 3 Fully-connected neuron layers
- Output layer

The 2 hidden layer and another hidden layer has, consequently, 50 and 10 neurons dense layer with 'relu' activation function. The output layer is one fully-connected neuron with 'sigmoid' activation function. The sequential model is compiled with 'Adam' optimizer and 'binary_crossenteopy' cost function.

2) The LSTMs model: is created as follows.

- LSTMs 2 layers
- Fully-connected neuron layer
- Output layer

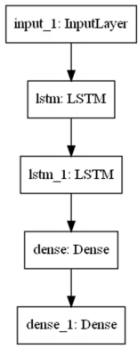


Figure 2. The LSTM model structure.

These two LSTMs hidden layers have initial 50 LSTMs each with 'tanh' activation function, and another layer is a 10 neurons fully-connected layer with 'linear' activation function. The output layer is a single fully-connected neuron (dense layer). The sequential model is comp with 'SGD' optimizer and 'binary_crossentropy' owning to the binary classification problem.

In general, the output layer is set by the problem requirement. In this application, the problem is binary classification, the model output use sigmoid function as the transfer function.

There are two methods used in this work to randomly portion out the dataset into training dataset and testing dataset. The first one is known as data splitting, and the second one is cross validation.

C. Data Splitting

The datasets have to randomly split into training dataset and testing dataset. Each dataset is [X, y] format, the dataset is, likewise, [train X, train Y, test X, test Y] format. The test ratio is 0.2 for the simple neuron model.

K-Fold cross validation is a resampling procedure for machine learning models. A parameter, called k, is assigned as the number of groups which the dataset being separated into. This k-fold cross validation splits dataset into k groups. For each group, the k-1 datasets fit the model as training dataset, then the rest evaluates the model as testing dataset. This procedure offers the opportunity for each fold dataset to train the model for k-1 times, and to test the model for once as the hold set. In this work, the K number is 10 for the LSTMs model. Stratified K-Fold cross validation is used as K-Fold CV. The Stratified K-Fold cross validation has more advantageous than the traditional one especially in case of the imbalanced data samples.

D. Parameter and Hyperparameter Optimization

The model parameters are variables internal the neural networks which theirs values are estimate from the data.

Their values define the model performance. For examples, the weights and biases are the model parameters.

The model hyperparameters are variables external the neural networks which theirs values are not able to estimate directly from the data. For example, learning rate, activation function and dropout are the hyperparameters.

In this work, the Grid Search Parameter is used as the parameter optimization method. Batch Size is the number of the size for each batch. This means the total number of training data samples in a batch. The dataset is divided into smaller batches, and those batches are fed into the neural networks. The number of Iterations is the number of batches which needs to complete one epoch. The number of Epochs is the number of times for the entire dataset passing forward and backward through the neural networks once. It means the number of batches is equal to the number of iteration for one epoch.

There are many optimizers to search for the best model of neural network; for instance, 'SGD', 'RMSprop', 'Adagrad', 'Adadelta', 'Adam', 'Adamax', 'Nadam' and so on.

The number of Learning Rate is the number which controls the amount of updating the weight at the end of each batch. The number of Momentum is the number which controls the amount of letting the previous update influence the current weight update.

The number of Dropout Regularization is important to improve the neural networks performance. In order to avoid underfitting and overfitting model, dropout regularization impact significantly on machine learning to achieve the appropriate-fitting. The number should be between [0.0, 1.0), [1] is not possible. It generally starts at 0.2 or 20% dropout rate, and the model generates usually the robust result.

The number of Neurons in the hidden layer is one of the most important parameter to be tuned for the neural networks, because it generally represents the capacity of the neural networks.

From these optimization procedures, the final LSTMs model configuration is now as follows:

- 3 hidden layers :
  - 100 LSTMs with 'relu' activation function, 'uniform' weight initializer, maxnorm(4) weight constraint and 0.2 dropout rate for 2 layers
  - 50 neurons with 'sigmoid' activation function and 'uniform' weight initializer dense layer
- Output layer : 1 neuron dense layer
- Compiling the model with 'binary_crossentropy' loss function and 'Adam' optimizer.
- Batch size : 100, 200 epochs, 0.01 learning rate.

E. Evaluating the Model

There are two methods of evaluating the models which are Hold-Out and Cross-Validation for large dataset and limited amount of dataset consequently. The hold-out method is randomly divided data into three subsets: Training set, Validation set and Testing set.

In this application, K-fold cross-validation for the LSTMs model due to amount of data availability is used to compare with traditional data splitting with 0.2 test ratio for the simple neuron model.

III. Results

There are two summarized result from initial setting and optimized setting. The records of accuracy are shown in percentage. Each result is obtained from the exported model with the Parkinson's disease data.

The first result, the simple neuron model is trained with simple training and testing dataset with 0.2 ratio test data splitting and initial configuration as shown in Table I.

TABLE I. The results from the Simple fully-connected neuron neural network with Normal data splitting and default parameter setting.

Results	Dataset1	Dataset2	Dataset3
1	53.4198	57.192	62.541
2	54.1945	60.0691	63.1512
3	52.9121	61.894	64.0891
4	56.1148	62.1905	61.261
5	54.2189	59.1892	59.198
6	54.2105	58.1984	59.0148
7	58.641	60.1444	60.4717
8	55.0001	58.176	61.2239
9	56.1894	59.6697	62.3617
10	53.4234	59.9963	63.7879

The second result, the LSTMs model is trained with K-fold cross-validation dataset with optimum parameter configuration from parameter optimization procedure as shown in Table II.

TABLE II. The results from the LSTMs neural network with K-fold cross-validation and optimized parameter setting.

Results	Dataset1	Dataset2	Dataset3
1	58.7846	67.8156	70.5491
2	61.3605	68.1602	72.911
3	62.4908	72.4869	73.6619
4	57.096	74.3201	75.6138
5	59.5541	70.7767	74.9995
6	63.0847	71.591	73.1542
7	60.3337	73.4852	75.6173
8	58.6913	69.5894	76.2189
9	62.7185	72.5507	74.0695
10	59.1515	73.0564	72.0519

The results from both configurations are distinct. The performance of the LSTMs neural network with k-fold cross validation and parameter optimization is clearly an improvement over the simple Neural Network. Using dataset 3 is the best dataset 3 for this binary classification. The comparing between simple neuron model and LSTMs model is shown in Table III.

TABLE III. Experiment result for dataset-3

	Simple Neuron (%)	LSTMs (%)
Mean	61.7100	73.8847
Max. Acc.	64.0891	76.2189
Min. Acc.	59.0148	70.5491
Std.	1.7862	1.7868

The machine learning using k-fold cross-validation and parameter optimization can achieve the best result at 76.22% with 73.88% average and 70.55% minimum. The result is better than the NN configuration for at least 10% in average. This comparison between the simple neuron and LSTMs model represent the advantage of using LSTM for time series data.

IV. Discussion

There are several limiting factors in this dataset which prevent machine learning performance.:

- Parkinson's disease patients have one dominant hand showing distinct symptom and another hand being like if normal person.
- Symptoms of some Parkinson's disease patients are almost healthy as normal, they only have slightly sickness which indicate to be diagnosed as Parkinson's disease.
- Amount of data is not adequate to train the deep neural networks so as to receive higher accuracy classification prediction.

V. Conclusion

The LSTM neural network with K-Fold cross validation is an appropriate approach for time-series binary classification. This machine learning method works not only for Parkinson's Disease data but also other time series data classification problems. The LSTM model can be improved to achieve higher accuracy using improved feature extraction.. The experiment shows promising results that the proposed method can be used for early screening of patients. This method needs further improvement before it can be used by OPD doctors for Parkinson's disease diagnosis.

References

[1] Akin Ozcift, and Arif Gulten, "Classifier ensemble construction with rotation foreast to improve medical diagnosis performance of machine learning algorithms," Computer Methods and Programs in Biomedicine 104, 2011, pp.443-451.

[2] P. R. Davidson, R. D. Jones1,and M. T. R. Peiris, "Detecting Behavioral Microsleeps using EEG and LSTM Recurrent Neural Networks," the 2005 IEEE Engineering in Medicine and Biology 27th Annual Conference, 2005.

[3] Zachary C. Lipton, David C. Kale, Charles Elkan, and Randall Wetzel, "Learning to diagnose with LSTM recurrent neural networks," arXiv:1511.03677v7, 2017.

[4] Najmeh Nazari, Seyed Ahmad Mirsalari, Sima Sinaei, Mostafa E. Salehi, and Masoud Daneshtalab, "Multi-level Binarized LSTM in EEG Classification for Wearable Devices," 28th Euromicro International Conference on Parallel, Distributed and Network-Based Processing, 2020.

[5] Inci M. Baytas, Cao Xiao, Xi Zhang, Fei Wang, Anil K. Jian, and Jiayu Zhou, "Patient Subtyping via Time-Aware LSTM Networks," KDD, 2017.

[6] Daniel Soutner , and Ludek Muller, "Application of LSTM Neural Networks in Lanfguage Modelling," Springer-Verlag Berlin Heidelberg, 2013, pp.105-112.

[7] John Gamboa, "Deep Learing for time-series analysis," arXiv:1701.01887v1, 2017.

[8] Hamid Palangi, Li Deng, Yelong Shen, Jianfeng Gao, Xiaodong He, Jianshu Chen, Xinying Song, and Rabab Ward, "Deep Sentence Embedding Using Long Short-Term Memory Networks: Analysis and Application to Information Retrieval," IEEE/ACM Transactions on Audio, Speech, and Language Processing, Vol.24, No.4, 2016.

[9] Hasim Sak, Andrew Senior, and Francoise Beaufays, "Long Short-Term Memory Recurrent Neural Network Architectures for Large Scale Acoustic Modeling," Interspeech, 2014.

[10] Xiaopeng Xi, Eamonn Keogh, Christian Shelton, and Li Wei, "Fast Time Series Classification Using Numerosity Reduction," the 23rd International Conference on Machine Learning, 2006.

[11] Wonmin Byeon, Thomas M. Breuel, Federico Raue, and Marcus Liwicki, "Scene Labeling with LSTM Recurrent Neural Networks," IEEE Conference on Computer Vision and Pattern Recognition, 2015.

[12] Fazle Karim, Somshubra Majudar, Houshang Darabi and Shen Chen, "LSTM Fully Convolutional Networks for Time Series Classification," IEEE Access, Vol.6, 2017.

[13] Shivam Chandhok, Himani Joshi, S. J. Darak, and A. V. Subramanyam, "LSTM Guided Modulation Classification and Experimental Validation for Sub-Nyquist Rate Wideband Spectrum Sensing," 11th International Conference on Communication Systems and Networks, 2019.

[14] Pierre Geurts, "Pattern Extraction for Time Series Classification," PKDD, 2001, pp.115-127.

[15] Hui-Ling Chen, Chang-Cheng Huang, Xin-Gang Yu, Xin Xu, Xin Sun, Gang Wang, and Su-Jing Wang, "An efficient diagnosis system for detection of Parkinson's disease using fuzzy k-nearest neighbor approach," Expert Systems with Application 40, 2013, pp.267-271.

[16] Hui-Ling Chen, Gang Wang, Chao Ma, Zhen-Nao Cai, and Wei-Bib Liu, "An efficient hybrid kernel extream learning machine approach for early diagnosis of Parkinson's disease," Neurocomputing 184, 2016, pp.131-144.

[17] Chao Che, Cao Xiao, Jian Liang, Bo Jin, Jiayu Zhou, and Fei Wang, "An RNN Architecture with Dynamic Temporal Matching for Personalized Predictions of Parkinson's Disease," SIAM International Conference on Data Mining, 2017.

[18] Ram Deepak Gottapu, and Cihan H. Dagli, "Analysis of Parkinson's Disease Data," Procedia Computer Science 140, 2018, pp.334-341.

[19] Athanasios Tagaris, Dimitrios Kollias, and Andreas Stafylopotis, "Assesment of Parkinson's Disease Based on Deep Neural Networks," Springer International Publicshing AG, 2017, pp.391-403.

[20] Xi Zhang, Jingyuan Chou, Jian Liang, Cao Xiao, Yize Zhao, Harini Sarva, Claire Henchcliffe, and Fei Wang, "Data-driven subtyping of Parkinson's disease using longtitudinal clinincal cecords: A Cohort study," Scientific Reports, 2019.

[21] David Gil A., and Magnus Johnson B., "Diagnosing Parkinson by using Artificial Neural Networks and Support Vector Machines," Global Journal of Computer Science and Technology 9, 2009.

[22] Shreya Bhat, U. Rajendra Acharya, Yuki Hagiwara, Nahid Dadmehr, and Hojjat Adeli, "Parkinson's disease: Cause factors, measurable indicators, and early diagnosis," Computers in Biology and Medicine 102, 2018, pp.234–241.

[23] Siddharth Arora, Fahd Baig, Christine Lo, Thomas R. Barber, Michael A. Lawton, Andong Zhan, Michal Rolinski, Claudio Ruffmann, Johannes C. Klein, D, Jane Rumbold, BSc, Amandine Louvel, Zenobia Zaiwalla, Graham Lennox, Tim Quinnell, Gary Dennis, Richard Wade-Martins, Yoav Ben-Shlomo, Max A. Little, and Michele T. Hu, "Smartphone motor testing to distinguish idiopathic REM sleep behavior disorder, controls, and PD," Neurology, 2018.

[24] Andong Zhan, Srihari Mohan, Christopher Tarolli, Ruth B. Schneider, Jamie L. Adams, Saloni Sharma, Molly J. Elson, Kelsey L. Spear, Alistair M. Glidden, Max A. Little, Andreas Terzis, E. Ray Dorsey, Suchi Saria, "Using Smartphones and Machine Learning to Quantify Parkinson Disease Severity The Mobile Parkinson Disease Score," JAMA Neurology, 2018.

[25] Resul Das, "A comparison of multiple classification methods for diagnosis of Parkinson disease," Expert Systems with Application 37, 2010, pp.1568-1572.

ประวัติผู้เขียน

ชื่อ-สกุล	หัสพล ธัมมิกรัตน์
วัน เดือน ปี เกิด	15 ธันวาคม 2537
วุฒิการศึกษา	วศ.บ. (วิศวกรรมระบบควบคุม) สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง วศ.ม. (วิศวกรรรมคอมพิวเตอร์) จุฬาลงกรณ์มหาวิทยาลัย (กำลังศึกษา)
ที่อยู่ปัจจุบัน	1046/24 สุขุมวิท101 ถนนสุขุมวิท แขวงบางจาก เขตพระโขนง กรุงเทพมหานคร 10260
ผลงานตีพิมพ์	H. Thummikarat and P. Chongstitvatana, "An Implementation of Machine Learning for Parkinson's Disease Diagnosis," 2021 18th International Conference on Electrical Engineering/Electronics, Computer, Telecommunications and Information Technology (ECTI-CON), 2021, pp. 258-261, doi: 10.1109/ECTI-CON51831.2021.9454784.