การปรับปรุงโมเดลออนไลน์ซีเควนเชียลเอกซ์ทรีมเลิร์นนิงแมชชีน สำหรับพยากรณ์โหลดไฟฟ้าในระยะสั้น

IMPROVEMENT OF ONLINE SEQUENTIAL EXTREME LEARNING MACHINE FOR SHORT TERM ELECTRICITY LOAD FORECASTING

ชานนท์ ชูพงษ์

ดุษฎีนิพนธ์ นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตร
ปริญญาวิศวกรรมศาสตรดุษฎีบัณฑิต สาขาวิชาวิศวกรรมไฟฟ้า
คณะวิศวกรรมศาสตร์
มหาวิทยาลัยเทคโนโลยีราชมงคลธัญบุรี
ปีการศึกษา 2566

การปรับปรุงโมเดลออนไลน์ซีเควนเชียลเอกซ์ทรีมเลิร์นนิงแมชชีน สำหรับพยากรณ์โหลดไฟฟ้าในระยะสั้น

ชานนท์ ชูพงษ์

ดุษฎีนิพนธ์ นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตร
ปริญญาวิศวกรรมศาสตรดุษฎีบัณฑิต สาขาวิชาวิศวกรรมไฟฟ้า
คณะวิศวกรรมศาสตร์
มหาวิทยาลัยเทคโนโลยีราชมงคลธัญบุรี
ปีการศึกษา 2566

หัวข้อดุษฎีนิพนธ์	การปรับปรุงโมเดลออนไลน์ซีเควนเชียลเอกซ์ทรีมเลิร์นนิงแมชชีนสำหรับพยากรณ์โหลดไฟฟ้าในระยะสั้น Improvement of Online Sequential Extreme Learning Machine for Short Term Electricity Load Forecasting
ชื่อ – นามสกุล	นายชานนท์ ชูพงษ์
สาขาวิชา	วิศวกรรมไฟฟ้า
อาจารย์ที่ปรึกษา	รองศาสตราจารย์บุญยัง ปลั่งกลาง, Dr.-Ing.
ปีการศึกษา	2566

คณะกรรมการสอบดุษฎีนิพนธ์

..... ประธานกรรมการ
(รองศาสตราจารย์ประมุข อุณหเลขกะ, วศ.ด.)

..... กรรมการ
(รองศาสตราจารย์พฤศยน นินทนาวงศา, Ph.D.)

..... กรรมการ
(รองศาสตราจารย์ณฐภัทร พันธ์คง, Ph.D.)

..... กรรมการ
(ผู้ช่วยศาสตราจารย์มนฑล นาวงษ์, วศ.ด.)

..... กรรมการ
(รองศาสตราจารย์บุญยัง ปลั่งกลาง, Dr.-Ing.)

คณะวิศวกรรมศาสตร์ มหาวิทยาลัยเทคโนโลยีราชมงคลธัญบุรี อนุมัติดุษฎีนิพนธ์ฉบับนี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาดุษฎีบัณฑิต

..... คณบดีคณะวิศวกรรมศาสตร์
(รองศาสตราจารย์สรพงษ์ ภวสุปรีย์, Ph.D.)
วันที่ 12 เดือน มีนาคม พ.ศ. 2567

หัวข้อดุษฎีนิพนธ์	การปรับปรุงโมเดลออนไลน์ซีเควนเชียลเอกซ์ทรีมเลิร์นนิงแมชชีนสำหรับพยากรณ์โหลดไฟฟ้าในระยะสั้น
ชื่อ – นามสกุล	นายชานนท์ ชูพงษ์
สาขาวิชา	วิศวกรรมไฟฟ้า
อาจารย์ที่ปรึกษา	รองศาสตราจารย์บุญยัง ปลั่งกลาง, Dr.-Ing.
ปีการศึกษา	2566

บทคัดย่อ

ในปัจจุบันรูปแบบการใช้พลังงานไฟฟ้าแตกต่างจากอดีตอย่างมากเนื่องมาจากการใช้งานระบบเซลล์แสงอาทิตย์และสถานีอัดประจุยานยนต์ไฟฟ้าที่มีจำนวนเพิ่มขึ้นอย่างรวดเร็ว จนส่งผลให้การพยากรณ์โหลดด้วยวิธีการเดิมมีความผิดพลาดสูงขึ้น เพื่อแก้ปัญหาดังกล่าวจึงได้นำเสนอโมเดลที่สามารถเรียนรู้เพิ่มขึ้นจากข้อมูลที่ได้รับระหว่างที่ทำงาน คือออนไลน์ซีเควนเชียลเอกซ์ทรีมเลิร์นนิงแมชชีน (OS-ELM) โดยจุดเด่นของโมเดลนี้คือสามารถทำงานโดยใช้ทรัพยากรต่ำแต่มีความเร็วสูง แต่โมเดล OS-ELM ก็ยังมีจุดที่ต้องปรับปรุงอยู่ คือความแม่นยำในการพยากรณ์ และความต้องการชุดข้อมูลที่มีจำนวนมากพอสำหรับฝึกสอนเริ่มต้นก่อนการใช้งาน ดังนั้นดุษฎีนิพนธ์นี้จึงได้นำเสนอวิธีการปรับปรุงโมเดล OS-ELM ให้สามารถใช้งานได้สะดวกและมีความแม่นยำในการพยากรณ์โหลดมากขึ้น

ดุษฎีนิพนธ์นี้ได้นำเสนอวิธีการปรับปรุงโมเดล OS-ELM ไว้ 4 วิธีคือ 1) การใช้โมเดลชุด 2) การสังเคราะห์ข้อมูลเพื่อฝึกสอนเริ่มต้น 3) การจัดกลุ่มข้อมูลให้โมเดลทำการพยากรณ์ และ 4) การให้โมเดลได้เรียนรู้ซ้ำ โดยทำการทดลองด้วยชุดข้อมูลการใช้ไฟฟ้าในรัฐทางชายฝั่งตะวันออกสหรัฐอเมริกาเพื่อให้โมเดลทำการพยากรณ์โหลดไฟฟ้ารายชั่วโมง และเปรียบเทียบค่าความผิดพลาดในการพยากรณ์ของวิธีการที่นำเสนอกับวิธีการที่เป็นเส้นฐานของแต่ละการทดลอง

ผลการศึกษาพบว่า 1) การใช้โมเดลชุดช่วยความผิดพลาดในการพยากรณ์ลดลงได้ 5-8% 2) การสังเคราะห์ข้อมูลช่วยให้การใช้งานโมเดล OS-ELM มีความสะดวกมากยิ่งขึ้นและช่วยลดค่าความผิดพลาดในการพยากรณ์ลงได้มากกว่า 10% 3) การจัดกลุ่มข้อมูลให้โมเดลทำการพยากรณ์ให้ค่าความผิดพลาดในการพยากรณ์ไม่แตกต่างจากการใช้โมเดลชุด และ 4) การเรียนรู้ซ้ำช่วยลดความผิดพลาดในการพยากรณ์ได้ 3-9%

คำสำคัญ: ออนไลน์ซีเควนเชียลเอกซ์ทรีมเลิร์นนิงแมชชีน การเรียนรู้แบบเพิ่มขึ้น การพยากรณ์โหลดไฟฟ้า

Dissertation Title	Improvement of Online Sequential Extreme Learning Machine for Short Term Electricity Load Forecasting
Name-Surname	Mr. Charnon Chupong
Program	Electrical Engineering
Dissertation Advisor	Associate Professor Boonyang Plangklang, Dr.-Ing.
Academic Year	2023

ABSTRACT

Nowadays, electricity consumption patterns are totally different from the past due to the rapid increase of PV systems and EV charging stations. As this result, the load forecasting using the traditional method tends to have higher errors. To solve this problem, the Online Sequential Extreme Learning Machine (OS-ELM) was proposed in order to incrementally learn from received data while working. The feature of this model is that it can work with low resources but at a high speed. However, the OS-ELM model has some points that need to be improved such as forecasting accuracy and requiring datasets for initial training. Therefore, this dissertation proposed methods for improving the OS-ELM model to be more convenient to use and more accurate in load forecasting.

This dissertation proposed 4 methods for improving the OS-ELM model: 1) using an ensemble model, 2) synthesizing data for initial training, 3) clustering data for forecasting, and 4) using re-learning. The experiment was conducted with a dataset of electricity usage on east coast of the United States to forecast hourly electricity loads. The forecasting error was compared with the proposed method and the baseline method of each experiment.

The study results revealed that: 1) the ensemble model could reduce the forecast error by 5-8% , 2) the synthesis of data made the use of the OS-ELM model more convenient and the forecasting error could be reduced by more than 10% , 3) clustering the data for each model could not reduce the forecasting error compared to the ensemble model, and 4) the re-learning method could reduce the forecasting errors by 3-9%.

Keywords: online sequential extreme learning machine (OS-ELM), incremental learning, load forecasting

กิตติกรรมประกาศ

ดุษฎีนิพนธ์นี้สำเร็จลุล่วงตามวัตถุประสงค์ไปได้ด้วยดีเพราะได้รับความอนุเคราะห์จากท่านอาจารย์ที่ปรึกษาคือ รองศาสตราจารย์ ดร.บุญยัง ปลั่งกลาง ที่กรุณาเสียสละเวลาให้คำปรึกษาแนะนำตลอดจนชี้แนะแนวทางในการทำดุษฎีนิพนธ์ให้สำเร็จลุล่วงไปได้ด้วยดี ผู้วิจัยจึงขอกราบขอบพระคุณท่านอาจารย์เป็นอย่างสูงกราบขอบพระคุณ รองศาตราจารย์ ดร.ประมุข อุณหเลขกะ ผู้ทรงคุณวุฒิจากภายนอก รองศาสตราจารย์ ดร.พฤศยน นินทนาวงศา, รองศาสตราจารย์ ดร.ณฐภัทร พันธ์คง และผู้ช่วยศาสตราจารย์ ดร.มนฑล นาวงษ์ กรรมการสอบดุษฎีนิพนธ์ ที่กรุณาให้คำแนะนำและแก้ไขดุษฎีนิพนธ์นี้ให้สมบูรณ์ ขอขอบพระคุณอาจารย์และเจ้าหน้าที่ของมหาวิทยาลัยเทคโนโลยีราชมงคลธัญบุรีทุกท่านที่ได้ดำเนินการในเรื่องการขอทุนการศึกษาระดับดุษฎีบัณฑิตแก่ผู้วิจัย และขอขอบคุณอาจารย์และเจ้าหน้าที่ภาควิชาวิศวกรรมไฟฟ้าที่ต้องรับภาระงานหลายส่วนแทนผู้วิจัยระหว่างที่ผู้วิจัยดำเนินการทำดุษฎีนิพนธ์ฉบับนี้

สุดท้ายนี้ขอขอบพระคุณบิดา, มารดาและครอบครัวที่ให้การสนับสนุนและเป็นกำลังแก่ผู้วิจัยมาโดยตลอดจนสำเร็จการศึกษา

ชานนท์ ชูพงษ์

สารบัญ

สารบัญ (ต่อ)

หน้า

สารบัญตาราง

สารบัญรูป

หน้า

บทที่ 1
บทนำ

1.1 ความเป็นมาและความสำคัญของปัญหา

โดยทั่วไประบบไฟฟ้าถูกแบ่งส่วนออกเป็น ส่วนการผลิตไฟฟ้า (Power Generation), ส่วนการส่งกำลังไฟฟ้า (Power Transmission) , ส่วนการจ่ายกำลังไฟฟ้า (Power Distribution) และส่วนผู้ใช้ไฟฟ้า (Power Consumption) ในช่วงแรกเริ่มการบริหารจัดการระบบไฟฟ้ามักจะเน้นไปที่ การผลิตไฟฟ้า, การส่งกำลังไฟฟ้า และการจ่ายกำลังไฟฟ้า จนถึงช่วงประมาณปี ค.ศ. 1970 เริ่มมีความสนใจในการบริหารจัดการด้านโหลด (Demand Side Management, DSM) [1] เพื่อเพิ่มประสิทธิภาพให้ระบบไฟฟ้า ซึ่งการพยากรณ์โหลดไฟฟ้าเป็นส่วนสำคัญส่วนหนึ่งในการบริหารจัดการด้านโหลดรวมถึงการบริหารจัดการการผลิตไฟฟ้า และการบริหารจัดการด้านการส่งและจ่ายไฟฟ้าด้วย

มีการศึกษาวิจัยเรื่องการพยากรณ์โหลดไฟฟ้าอยู่เป็นจำนวนมาก [2,3] ซึ่งการพยากรณ์ถูกแบ่งตามระยะเวลาในการพยากรณ์ล่วงหน้า 4 ระยะได้แก่

1) การพยากรณ์ในระยะสั้นมาก (Very Short Term Load Forecasting) เป็นการพยากรณ์ล่วงหน้าไม่เกิน 1 ชั่วโมง โดยข้อมูลที่ได้จากการพยากรณ์มักนำไปใช้ในการควบคุมคุณภาพของระบบไฟฟ้า

2) การพยากรณ์ในระยะสั้น (Short Term Load Forecasting) เป็นการพยากรณ์ล่วงหน้า 1 ชั่วโมง แต่ไม่เกิน 1 สัปดาห์ โดยข้อมูลที่ได้จากการพยากรณ์มักนำไปใช้ในการปรับสมดุลระหว่างการใช้ไฟฟ้าและการผลิตไฟฟ้า

3) การพยากรณ์ในระยะกลาง (Medium Term Load Forecasting) เป็นการพยากรณ์ล่วงหน้า 1 เดือนแต่ไม่เกิน 1 ปี โดยข้อมูลที่ได้จากการพยากรณ์มักนำไปใช้ในการวางแผนจัดหาเชื้อเพลิงในการผลิตไฟฟ้า

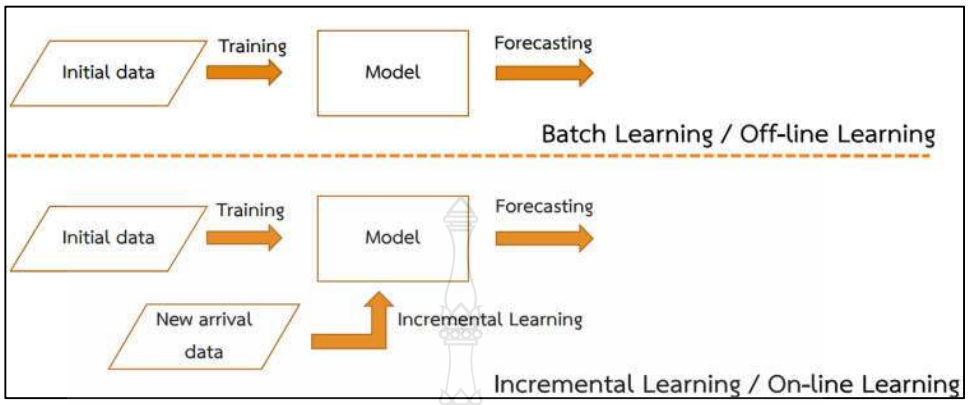
4) การพยากรณ์ในระยะยาว (Long Term Load Forecasting) เป็นการพยากรณ์ล่วงหน้าในหน่วยปี โดยข้อมูลที่ได้จากการพยากรณ์มักนำไปใช้ในการวางแผนลงทุนสร้างโรงไฟฟ้าหรือโครงสร้างพื้นฐานในระบบไฟฟ้า

โมเดลที่ใช้ในการพยากรณ์โหลดไฟฟ้าที่พบได้บ่อยในงานวิจัยมีอยู่สองรูปแบบหลักคือ

1) โมเดล Autoregressive Integrated Moving Average (ARIMA) เป็นโมเดลที่นิยมใช้ในการวิเคราะห์ข้อมูลอนุกรมเวลา (Time Series Analysis) โดยโมเดลจะประกอบด้วยส่วน Autoregressive (AR) ซึ่งเป็นโมเดลที่อธิบายการถดถอยเชิงเส้น (Linear Regression) ระหว่างข้อมูลปัจจุบันกับข้อมูลในอดีต และส่วน Moving Average (MA) ซึ่งเป็นโมเดลที่อธิบายการถดถอยเชิงเส้น (Linear Regression) ระหว่างข้อมูลปัจจุบันกับค่าความผิดพลาดจากการพยากรณ์ในอดีตซึ่งเกิดจากสัญญาณรบกวนไวต์ (White Noise) ของข้อมูล โดยปกติโมเดล ARและ MA นี้จะใช้ได้กับข้อมูลนิ่ง (Stationary Data) เท่านั้น และเพื่อเป็นการปรับปรุงให้โมเดล AR และ MA สามารถใช้งานได้กับข้อมูลที่ไม่นิ่ง (Non-Stationary Data) ได้ จึงได้มีการเพิ่มส่วนที่เรียกว่า Integrated ในโมเดลและเมื่อนำทั้งหมดมารวมกันจะเรียกโมเดลดังกล่าวว่า Autoregressive Integrated Moving Average (ARIMA)

2) โมเดลโครงข่ายประสาทเทียม (Artificial Neural Network, ANN) ในการพยากรณ์อนุกรมเวลานั้นโมเดลโครงข่ายประสาทเทียมมีข้อแตกต่างจากโมเดล ARIMA ตรงที่สามารถใช้ฟังก์ชันกระตุ้นที่ไม่เป็นเชิงเส้น (Non-linear Activation Function) และโครงสร้างที่มีชั้นซ่อน (Hidden Layer) ทำให้โครงข่ายประสาทเทียมสามารถพยากรณ์ข้อมูลที่มีความสัมพันธ์แบบไม่เป็นเชิงเส้น (Non-Linear) ได้ดีกว่าโมเดล ARIMA [2,3]

การสร้างโมเดลสำหรับพยากรณ์โหลดไฟฟ้าข้างต้นจะต้องใช้ข้อมูลในอดีตมาคำนวณหาค่าพารามิเตอร์ต่างๆ ในโมเดล ซึ่งในโมเดลโครงข่ายประสาทเทียมที่ใช้เทคนิคต่างๆ ทางด้านการเรียนรู้ของเครื่อง (Machine Learning) เราจะเรียกการหาค่าพารามิเตอร์นี้ว่าเป็นการฝึกสอน (Training) โดยข้อมูลที่ใช้ฝึกสอนนี้จะต้องมีจำนวนมากเพียงพอ และควรมีแบบรูป (Pattern) เหมือนกับข้อมูลที่โมเดลจะต้องพยากรณ์ การฝึกสอนดังกล่าวจะกระทำครั้งเดียวตอนเริ่มต้นด้วยข้อมูลทั้งหมดเรียกว่าเป็นการเรียนข้อมูลทั้งกลุ่ม (Batch Learning) แต่ในปัจจุบันแบบรูป (Pattern) การใช้พลังงานไฟฟ้าของผู้ใช้ไฟฟ้ามีการเปลี่ยนแปลงตลอดเวลาเนื่องมาจาก ระบบผลิตไฟฟ้าจากเซลล์แสงอาทิตย์ (PV System) [4], ยานยนต์ไฟฟ้า (Electric Vehicle) [5], ระบบกักเก็บพลังงาน (Energy Storage) [6] และการตอบสนองด้านโหลด (Demand Response) [7] ดังนั้นโมเดลที่ถูกสร้างด้วยวิธี Batch Learning จำเป็นจะต้องมีการฝึกสอนใหม่ (Retraining) ตลอดเวลาเพื่อความถูกต้องในการพยากรณ์ ซึ่งการฝึกสอนใหม่ในกรณีดังกล่าวนี้สร้างความไม่สะดวกในทางปฏิบัติ ดังนั้นนักวิจัยจึงได้นำเสนอโมเดลที่สามารถเรียนรู้เพิ่มขึ้นได้ในขณะที่กำลังทำงานอยู่ หรือเรียกได้ว่าโมเดลที่เรียนรู้แบบออนไลน์ (Online Learning) บางงานวิจัยจะเรียกโมเดลลักษณะนี้ว่าเป็นโมเดลที่เรียนรู้แบบเพิ่มขึ้น (Incremental Learning)



รูปที่ 1.1 เปรียบเทียบการเรียนรู้ของโมเดลแบบเรียนรู้ข้อมูลทั้งกลุ่ม (Batch Learning) กับแบบเรียนรู้เพิ่มขึ้น (Incremental Learning)

โมเดลเรียนรู้แบบเพิ่มขึ้น (Incremental Learning) ได้มีการศึกษาวิจัยและนำเสนอไว้จำนวนหนึ่ง [8] เช่น Incremental Support Vector Machine [9], Online Random Forecast [10], Learn++ [11], Online Sequential Extreme Learning Machine [12] โดยในดุษฎีนิพนธ์นี้สนใจศึกษาโมเดล Online Sequential Extreme Learning Machine (OS-ELM) เนื่องจากมีความเร็วสูงในการเรียนรู้จากข้อมูล (Learning Speed) ทำให้สามารถทำงานบนอุปกรณ์ที่มีทรัพยากรในการคำนวณน้อยได้ดี เหมาะกับงานพยากรณ์กำลังไฟฟ้าระยะสั้นที่สามารถเรียนรู้เพิ่มขึ้นได้ แต่โมเดล OS-ELM ยังมีข้อด้อยที่ต้องปรับปรุงอยู่บ้าง เช่น ต้องใช้ข้อมูลในการเริ่มต้นฝึกสอนโมเดลซึ่งอาจไม่เหมาะกับอาคารที่สร้างใหม่หรือพื้นที่ที่ยังไม่มีการเก็บข้อมูลการใช้ไฟฟ้า และค่าพารามิเตอร์ในชั้นซ่อนของโมเดลเกิดจากการสุ่มตัวเลขซึ่งอาจไม่ใช่วิธีการที่ดีในการสกัดคุณลักษณะ (Feature extraction) ของข้อมูลอินพุตจนส่งผลให้การพยากรณ์ไม่แม่นยำเท่าที่ควร ดังนั้นในงานวิจัยนี้จึงนำเสนอวิธีการปรับปรุงโมเดล OS-ELM ให้สามารถทำงานได้โดยไม่ต้องใช้ข้อมูลตัวอย่างสำหรับฝึกสอนเริ่มต้น และปรับปรุงให้มีความแม่นยำในการพยากรณ์สูงขึ้นโดยใช้วิธีที่มีต้นทุนในการคำนวณต่ำเหมาะสมที่จะนำไปใช้งานบนอุปกรณ์ที่มีทรัพยากรต่ำเช่นไมโครคอนโทรลเลอร์หรือคอมพิวเตอร์บอร์ดเดี่ยว (Single Board Computer) ได้

1.2 วัตถุประสงค์

1.2.1 เพื่อศึกษาแนวทางต่างๆ ในการปรับปรุงโมเดล OS-ELM

1.2.2 นำเสนอวิธีการใช้งานโมเดล OS-ELM สำหรับพยากรณ์โหลดไฟฟ้าระยะสั้น โดยไม่ต้องใช้ข้อมูลตัวอย่างสำหรับฝึกสอนเริ่มต้น

1.2.3 นำเสนอวิธีการปรับปรุงความแม่นยำของโมเดล OS-ELM ในการพยากรณ์โหลดไฟฟ้าระยะสั้น

1.3 สมมติฐานของการวิจัย

1.3.1 ข้อมูลตัวอย่างที่ได้จากการสังเคราะห์สามารถใช้แทนข้อมูลโหลดไฟฟ้าจริงในอดีต เพื่อฝึกสอนเริ่มต้นให้กับโมเดล OS-ELM ได้โดยไม่ทำให้ค่าความแม่นยำในการพยากรณ์โหลดไฟฟ้าลดลง

1.3.2 โมเดลแบบกลุ่ม (Ensemble Model) จะช่วยเพิ่มความแม่นยำในการพยากรณ์ได้

1.3.3 การจัดกลุ่มข้อมูล (Clustering) ตามลักษณะการใช้ไฟฟ้า แล้วแยกใช้โมเดลแต่ละชุดเพื่อพยากรณ์ลักษณะการใช้ไฟฟ้าแต่ละแบบ จะช่วยเพิ่มความแม่นยำในการพยากรณ์ได้

1.3.4 การเรียนซ้ำ (Re-learn) กับข้อมูลตัวอย่างชุดเดิมจะช่วยให้โมเดล OS-ELM ปรับตัวตามข้อมูลที่เปลี่ยนแปลงได้เร็วขึ้น ทำให้ความแม่นยำในการพยากรณ์สูงขึ้น.

1.4 ขอบเขตของการวิจัย

1.4.1 ศึกษารวบรวมโมเดล Online Sequential Extreme Learning Machine (OS-ELM) และแนวทางต่างๆ ในการปรับปรุงความแม่นยำในการพยากรณ์

1.4.2 นำเสนอวิธีการที่สามารถเพิ่มความแม่นยำในการพยากรณ์ให้กับโมเดล OS-ELM

1.4.3 โมเดล OS-ELM ที่นำเสนอนี้สร้างในคอมพิวเตอร์ด้วยภาษาไพทอน (Python)

1.4.4 ทดสอบความถูกต้องแม่นยำในการพยากรณ์ของโมเดล OS-ELM ที่นำเสนอ ด้วยข้อมูลโหลดไฟฟ้าจริง Hourly energy consumption จากเว็บไซต์ www.kaggle.com

1.4.5 เปรียบเทียบความแม่นยำในการพยากรณ์ของโมเดล OS-ELM ที่นำเสนอกับโมเดล OS-ELM ปกติ

1.5 ขั้นตอนการวิจัย

1.5.1 ศึกษารายละเอียดของดุษฎีนิพนธ์จากเอกสารตำรา และงานวิจัยที่เกี่ยวข้อง เพื่อกำหนดปัญหา, วัตถุประสงค์, สมมติฐานและขอบเขตของงานวิจัย

1.5.2 ศึกษารายละเอียดจากเอกสารและงานวิจัยที่เกี่ยวข้องกับ การใช้โมเดล Online Sequential Extreme Learning Machine (OS-ELM) ในรูปแบบต่างๆ เพื่อการพยากรณ์อนุกรมเวลา

1.5.3 สร้างโมเดล OS-ELM ด้วยภาษาคอมพิวเตอร์ Python

1.5.4 ศึกษารายละเอียดจากเอกสารและงานวิจัยที่เกี่ยวข้องกับ การปรับปรุงโมเดล OS-ELM

1.5.5 ออกแบบและสร้างวิธีการปรับปรุงโมเดล OS-ELM ให้มีความแม่นยำในการพยากรณ์มากขึ้น

1.5.6 ทดสอบโมเดล OS-ELM ด้วยวิธีการที่สร้างขึ้น โดยทดลองพยากรณ์โหลดไฟฟ้าระยะสั้นจากข้อมูลจริง “Hourly Load Data” จากเว็บไซต์ www.kaggle.com

1.5.7 เปรียบเทียบความแม่นยำในการพยากรณ์ของโมเดล OS-ELM ที่สร้างขึ้นกับโมเดล OS-ELM ปกติ

1.5.8 เขียนบทความเพื่อเผยแพร่ผลงานวิจัย

1.5.9 จัดทำดุษฎีนิพนธ์ฉบับสมบูรณ์

1.6 ประโยชน์ที่คาดว่าจะได้รับ

1.6.1 ช่วยเพิ่มพูนความรู้ในการสร้างโมเดลพยากรณ์โหลดไฟฟ้าระยะสั้นแบบเรียนรู้เพิ่มขึ้นได้

1.6.2 ได้แนวคิดในการพัฒนาระบบมิเตอร์อัจฉริยะ (Advance Metering Infrastructure, AMI) สำหรับงานบริหารจัดการพลังงาน

1.6.3 ได้แนวทางในการนำโมเดลแบบเรียนรู้เพิ่มขึ้นไปใช้งานในอุปกรณ์ที่มีทรัพยากรในการคำนวณต่ำ

บทที่ 2
วรรณกรรมหรืองานวิจัยที่เกี่ยวข้อง

การปรับปรุงโมเดล Online Sequential Extreme Learning Machine (OS-ELM) จำเป็นต้องเข้าใจพื้นฐานการเรียนรู้ของเครื่อง (Machine learning) และหลักการทำงานของโมเดล Extreme Learning Machine (ELM) และ Online Sequential Extreme Learning Machine (OS-ELM) ดังนี้

2.1 พื้นฐานทางด้านการเรียนรู้ของเครื่อง (Machine Learning)

การเรียนรู้ของเครื่องเป็นสาขาหนึ่งในเทคโนโลยีทางด้านปัญญาประดิษฐ์ (Artificial intelligence, AI) โดยมีนิยามคือ “ขั้นตอนวิธี (Algorithm) ที่ใช้สร้างแบบจำลองทางคณิตศาสตร์ (Mathematical model) ของข้อมูล เพื่อที่จะใช้ในงานทำนายหรือตัดสินใจบางอย่าง โดยไม่ได้มีการตั้งโปรแกรมให้ทำงานดังกล่าวอย่างชัดเจน” จุดเด่นของการเรียนรู้ของเครื่องเมื่อเทียบกับโปรแกรมคอมพิวเตอร์โดยทั่วไปคือ ไม่มีการกำหนดตรรกะในการทำงานไว้ในตัวโปรแกรม ตัวโปรแกรมจะต้องสร้างตรรกะในการทำงานขึ้นมาเองจากข้อมูลที่ป้อนเข้าไป ดังรูปที่ 2.1 โปรแกรมคอมพิวเตอร์ที่ได้รับการยอมรับว่าเป็นการเรียนรู้ของเครื่องโปรแกรมแรกถูกนำเสนอโดย Arthur L. Samuel [13] เป็นโปรแกรมการเล่นหมากฮอส โดยผู้ใช้งานป้อนข้อมูลกติกาการเล่นหมากฮอสและคอมพิวเตอร์ประมวลผลการเล่นที่เหมาะสมเอง

รูปที่ 2.1 เปรียบเทียบโปรแกรมคอมพิวเตอร์ทั่วไปกับการเรียนรู้ของเครื่อง

รูปแบบของงานที่ใช้การเรียนรู้ของเครื่องเข้าไปแก้ปัญหาโดยทั่วไปมีอยู่ 3 รูปแบบคือ

2.1.1 การวิเคราะห์การถดถอย (Regression analysis)

เป็นการหาฟังก์ชันความสัมพันธ์ระหว่างข้อมูลอินพุตและข้อมูลเป้าหมายเพื่อนำฟังก์ชันดังกล่าวไปใช้พยากรณ์ค่าเป้าหมายสำหรับข้อมูลอินพุตใดๆ โดยแนวทางที่นิยมใช้คือการกำหนดสมมติฐาน (hypothesis) หรือแบบจำลองทางคณิตศาสตร์ (Mathematical model) หรือโมเดล ของฟังก์ชันความสัมพันธ์ระหว่างข้อมูลอินพุตและข้อมูลเป้าหมาย ดังนี้

$$Y' = h_\theta(X) \tag{2.1}$$

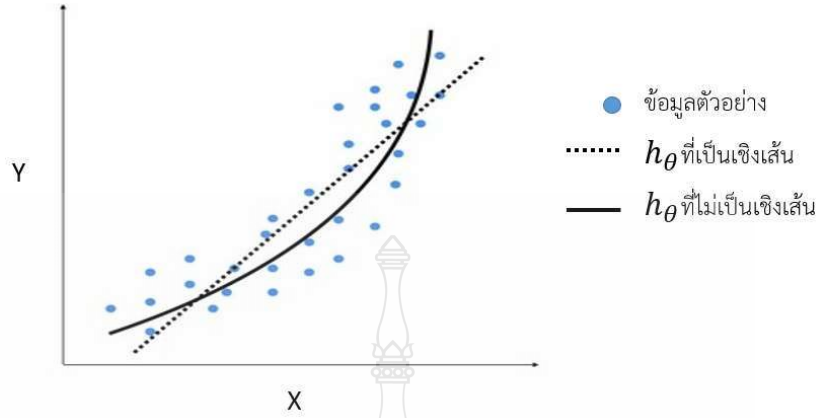
โดย

Y' คือ ค่าพยากรณ์ที่ได้จากสมมติฐานหรือโมเดล

h_θ คือ ฟังก์ชันสมมติฐานหรือโมเดล ที่มีค่าพารามิเตอร์เป็น θ

X คือ ค่าอินพุตที่ป้อนเข้าโมเดล

โดยฟังกัน h_θ มีรูปแบบการกำหนดได้หลากหลาย เช่นกำหนดให้เป็นการถดถอยเชิงเส้น (Linear regression) ค่าพยากรณ์จะคำนวณโดยนำค่าอินพุตแต่ละตัวมาคูณค่าน้ำหนักและรวมกัน เช่น $y = \theta_1 x_1 + \theta_2 x_2 + \cdots$ หากกำหนด h_θ เป็นโครงข่ายประสาทเทียม (Artificial neural network) ค่าพยากรณ์จะคำนวณตามโครงสร้างของโครงข่ายประสาทเทียมนั้นๆเช่น $y = g_1(\theta_{11}x_1 + \theta_{12}x_2 + \cdots) + g_2(\theta_{21}x_1 + \theta_{22}x_2 + \cdots) + \cdots$ โดย $g(.)$ คือฟังก์ชันกระตุ้น (Activation function) ของโครงข่ายประสาทเทียม



รูปที่ 2.2 การวิเคราะห์การถดถอย (Regression analysis)

ขั้นตอนต่อไปคือการหาค่าพารามิเตอร์ของโมเดลซึ่งเริ่มจากการกำหนดฟังก์ชันต้นทุน (Cost function) $\mathbf{J(\theta)}$ ซึ่งมักนิยมใช้ฟังก์ชันชนิดค่าผิดพลาดยกกำลังสองเฉลี่ย (Mean square error) ดังนี้

$$J(\theta) = \frac{1}{2m}\sum_{i=1}^{m}(Y'^{(i)} - Y^{(i)})^2 \tag{2.2}$$

โดย

m คือ จำนวนข้อมูลตัวอย่าง

$Y'^{(i)}$ คือ ค่าพยากรณ์ของข้อมูลตัวอย่างที่ i

$Y^{(i)}$ คือ ค่าจริงของข้อมูลตัวอย่างที่ i

ขั้นตอนต่อไปคือการใช้เทคนิคในการหาค่าที่เหมาะสมที่สุด (Optimization) หาค่าพารามิเตอร์ θ ที่ทำให้ค่าฟังก์ชันต้นทุนมีค่าต่ำที่สุด โดยวิธีการที่นิยมใช้มากที่สุดคือการเคลื่อนลงตามความชัน (Gradient descent) ซึ่งไม่ขอกล่าวถึงในที่นี้

นอกจากการใช้ Gradient descent ในการหาค่าพารามิเตอร์ที่เหมาะสมของโมเดลแล้ว ยังมีอีกวิธีหนึ่งที่นิยมใช้โดยเฉพาะกับโมเดลที่เป็น Linear regression ได้แก่วิธีการสมการปกติ (Normal equation) ซึ่งเป็นการคำนวณหาค่าพารามิเตอร์ของโมเดลด้วยวิธีการทางพีชคณิต (Algebra) โดยเริ่มจากกำหนด h_θ ให้อยู่ในรูปของ Linear regression ซึ่งสามารถเขียนสมการในรูปผลคูณของเมทริกซ์และเวกเตอร์ได้ดังนี้

$$\mathbf{Y \approx Y' = X\theta} \tag{2.3}$$

โดย

$\boldsymbol{Y}$ คือ ค่าจริงจากข้อมูลตัวอย่าง

$\boldsymbol{Y'}$ คือ ค่าพยากรณ์ที่ได้จากโมเดล

$\boldsymbol{X}$ คือ ค่าอินพุตที่ป้อนเข้าโมเดล

$\boldsymbol{\theta}$ คือ พารามิเตอร์ของโมเดล

จากสมการที่ 2.3 ซึ่งทราบค่า Y และค่า X ซึ่งเป็นเมทริกซ์และโดยมากจะไม่ใช่เมทริกซ์จัตุรัส (Square matrix) ค่าพารามิเตอร์ θ สามารถคำนวณได้ดังนี้

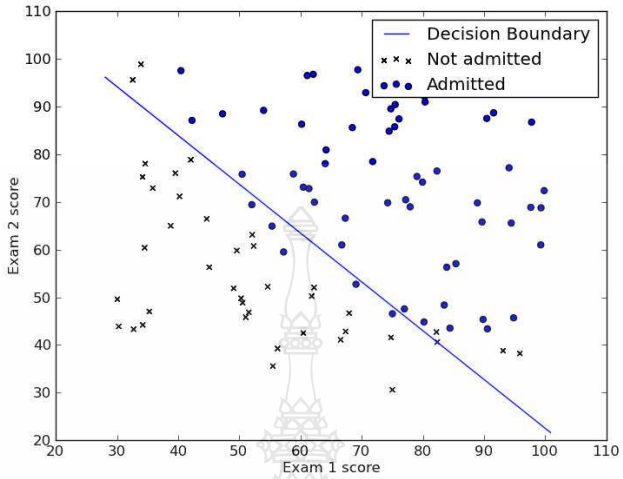
$$\mathbf{Y} \approx \mathbf{X\theta}$$
$$\mathbf{X^T Y} \approx \mathbf{X^T X\theta}$$
$$\mathbf{(X^T X)^{-1} X^T Y} \approx \mathbf{(X^T X)^{-1} X^T X\theta}$$

$$\mathbf{(X^T X)^{-1} X^T Y} \approx \boldsymbol{\theta} \tag{2.4}$$

การหาค่าพารามิเตอร์ของโมเดลด้วยวิธี Normal equation นี้จุดเด่นเมื่อเทียบกับ Gradient descent คือ ไม่จำเป็นต้องทำการปรับค่าให้เป็นบรรทัดฐาน (Normalization), ไม่จำเป็นต้องกำหนดค่าอัตราการเรียนรู้ (Learning rate) และโปรแกรมไม่ต้องทำงานแบบวนซ้ำ (Iteration) แต่มีข้อด้อยคือ เมื่อทำงานกับข้อมูลที่มีมิติมากและ/หรือ มีจำนวนมากโปรแกรมจะทำงานได้ช้า (เกิน 10,000 ตัวอย่าง [14]) และอาจพบเมทริกซ์ที่ไม่สามารถหาค่าผกผันได้ (non-invertible matrix)

2.1.2 การจำแนกประเภท (Classification)

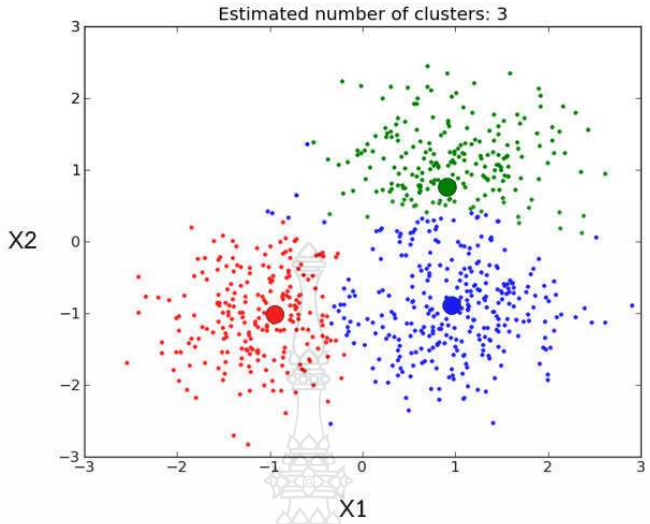
เป็นการหาฟังก์ชันความสัมพันธ์ระหว่างข้อมูลอินพุตกับข้อมูลเป้าหมายเช่นเดียวกับ Regression analysis เพียงแต่ค่าเป้าหมายในกรณีของ Classification นั้นจะมีลักษณะเป็นเซตวิยุต (Discrete set) เช่น การจำแนกรูปภาพว่าเป็นใบหน้าคนใช่หรือไม่, การจำแนกกรุ๊ปเลือดว่าเป็น A, B, AB หรือ O เป็นต้น สมมติฐานหรือโมเดลหนึ่งที่นิยมใช้ในการทำ Classification คือ การถดถอยแบบโลจิสติกส์ (Logistic regression) ซึ่งเป็นการจำแนกข้อมูลออกเป็น 2 ประเภท (Binary classification) โดยนำผลรวมเชิงเส้น (Linear combination) ของอินพุตในแต่ละมิติมาป้อนเข้าฟังก์ชัน sigmoid หรือใช้โมเดลโครงข่ายประสาทเทียมสำหรับการจำแนกข้อมูลออกเป็น หลายประเภท (Multi-class classification) ส่วน cost function หนึ่งที่นิยมใช้ในการทำ classification คือ Cross entropy function



รูปที่ 2.3 ตัวอย่างการจำแนกประเภท (Classification) ข้อมูล

2.1.3 การจับกลุ่ม (Clustering)

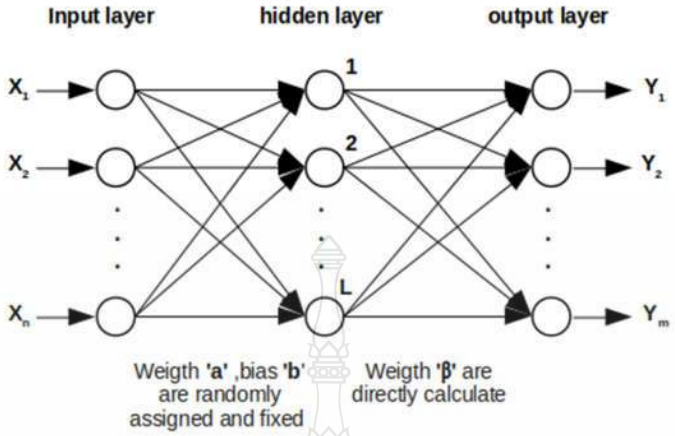
การจับกลุ่มข้อมูล (Data clustering) มีความแตกต่างจากการทำ regression และ classification ค่อนข้างมาก เนื่องจากข้อมูลที่ป้อนเข้าให้โมเดลเรียนรู้เพื่อปรับค่าพารามิเตอร์นั้นจะไม่มีค่าเป้าหมายซึ่งเรียกว่าเป็นการเรียนรู้แบบไม่มีผู้ฝึกสอน (Unsupervised learning) ส่วนการเรียนรู้เพื่อปรับค่าพารามิเตอร์ในงาน Regression และงาน Classification นั้นเรียกว่าเป็นการเรียนรู้แบบมีผู้ฝึกสอน (Supervised learning) โดยวัตถุประสงค์ของการทำ Clustering คือการจับเอาข้อมูลที่มีความ “คล้ายกัน” ไว้ในกลุ่มเดียวกัน โมเดลหรือสมมติฐานของการทำ Clustering คือการวัดความคล้ายกันของข้อมูลซึ่งมักจะวัดจากระยะห่างของข้อมูลแต่ละตัวในปริภูมิและจัดให้ข้อมูลที่อยู่ใกล้เคียงกันอยู่ในกลุ่มเดียวกัน ขั้นตอนวิธี (algorithm) ที่นิยมใช้มากในงาน Clustering คือการทำ K-Means Clustering ตัวอย่างงานที่ต้องใช้การทำ Clustering เช่น การแบ่งกลุ่มลูกค้า, การแบ่งประเภทผู้ใช้ไฟฟ้า, การตรวจสอบผู้บุกรุกในเครือข่ายคอมพิวเตอร์ เป็นต้น



รูปที่ 2.4 ตัวอย่างการจับกลุ่ม (Clustering) ข้อมูล

2.2 โมเดล Extreme Learning Machine (ELM)

Extreme Learning Machine (ELM) เป็นโมเดลที่มีโครงสร้างเหมือนโมเดลโครงข่ายประสาทเทียมชนิดชั้นซ่อนเดียวป้อนข้อมูลไปข้างหน้า (Single hidden layer feed-forward neural network, SLFN) ELM ถูกนำเสนอครั้งแรกโดย Guang Bin Hong ในปี ค.ศ. 2004 [15] จุดเด่นที่สุดของ ELM คือมีความเร็วสูงมากในการเรียนรู้ข้อมูลเพื่อปรับค่าพารามิเตอร์ของโมเดล เนื่องจาก ELM ไม่ใช้วิธีการวนซ้ำเช่น Gradient descent ในการคำนวณค่าพารามิเตอร์แต่ใช้วิธี Normal equation ในการคำนวณ โดยค่า bias และ weight ที่เชื่อมต่อระหว่างอินพุตและชั้นซ่อนจะถูกสร้างแบบสุ่ม (Random Generated) และจะมีค่าคงที่ตลอดการเรียนรู้ข้อมูล ส่วนค่า weight ระหว่างชั้นซ่อนกับเอาต์พุตจะคำนวณโดยใช้เทคนิค Normal equation อธิบายได้ดังนี้



รูปที่ 2.5 โครงสร้างของโมเดล Extreme Learning Machine (ELM)

ถ้าข้อมูลตัวอย่างที่ต้องการเรียนรู้มีจำนวน N ตัวอย่าง (x_j, y_j) โดยที่ $x_j \in \mathbb{R}^n$ และ $y_j \in \mathbb{R}^m$ จำนวนโหนดในชั้นซ่อนของโครงข่ายเท่ากับ L ความสัมพันธ์ระหว่าง x_j และ y_j จะเป็นดังนี้

$$\mathbf{y_j} = \sum_{\mathbf{i=1}}^{\mathbf{L}} \boldsymbol{\beta}_\mathbf{i} g(\mathbf{a_i x_j} + \mathbf{b_i}) \qquad j = 1,2,..n \tag{2.5}$$

โดย $\boldsymbol{\beta}_\mathbf{i} \in \mathbb{R}^m$ เป็นค่าน้ำหนักในชั้นซ่อน, $g(.) : \mathbb{R} \to \mathbb{R}$ เป็นฟังก์ชันกระตุ้นในชั้นซ่อน, $\mathbf{a}_i \in \mathbb{R}^n$ เป็นค่าน้ำหนักในชั้นอินพุต และ $\mathbf{b}_i \in \mathbb{R}$ เป็นค่าไบแอสในชั้นอินพุต สมการที่ 2.5 สามารถเขียนให้อยู่ในรูปแบบที่ง่ายขึ้นได้ดังนี้

$$\mathbf{Y} = \mathbf{H}\boldsymbol{\beta} \tag{2.6}$$

โดย

$$\mathbf{H} = \begin{bmatrix} g(a_1 x_1 + b_1) & \cdots & g(a_L x_1 + b_L) \\ \vdots & \ddots & \vdots \\ g(a_1 x_N + b_1) & \cdots & g(a_L x_N + b_L) \end{bmatrix}_{N \times L}$$

$$\boldsymbol{\beta} = \begin{bmatrix} \beta_1^T \\ \vdots \\ \beta_L^T \end{bmatrix}_{L \times m} \quad \text{และ} \quad \mathbf{Y} = \begin{bmatrix} y_1^T \\ \vdots \\ y_N^T \end{bmatrix}_{N \times m}$$

เมทริกซ์ $\boldsymbol{H}$ เรียกว่า The Hidden Layer Output Matrix ซึ่งค่า a_i และค่า b_i เป็นค่าคงที่ที่ได้จากการสุ่ม ทำให้ค่าเมทริกซ์ $\boldsymbol{H}$ เป็นค่าคงที่เช่นกัน ดังนั้นการเรียนรู้ข้อมูลของ ELM จึงเป็นการคำนวณค่า $\boldsymbol{\beta}$ ในสมการที่ 2.6 ซึ่งหากจำนวนข้อมูลหรือมิติของข้อมูลไม่มีค่ามากเกินไปเราสามารถใช้วิธี Normal equation คำนวณได้โดยตรงอย่างรวดเร็วเมื่อเทียบกับวิธี Gradient Descent โดยค่า $\boldsymbol{\beta}$ สามารถคำนวณได้ดังนี้

$$\boldsymbol{\beta} = \mathbf{H}^{\dagger}\mathbf{Y} \quad \text{โดย} \quad \mathbf{H}^{\dagger} = (\mathbf{H}^{\mathbf{T}}\mathbf{H})^{-1}\mathbf{H}^{\mathbf{T}} \tag{2.7}$$

ถึงแม้ว่า ELM จะใช้การสุ่มค่าน้ำหนักและค่าไบแอสในชั้นอินพุตที่ แต่จากงานวิจัยจำนวนมากพบว่า ELMสามารถในการเรียนรู้ข้อมูลได้เป็นอย่างดีเช่นเดียวกับ Learning Algorithm อื่น ๆ เช่นกัน [16], [17], [18] โมเดล ELM อาจพบปัญหาทางด้านเสถียรภาพเชิงตัวเลข (Numerical stability) เนื่องจากในบางครั้งจะพบว่าไม่สามารถหาค่าเมทริกซ์ผกผัน (inverse matrix) ของ $\mathbf{H}^{\mathbf{T}}\mathbf{H}$ ได้ ซึ่งจากการทบทวนวรรณกรรมพบว่ามีวิธีการแก้ปัญหาดังกล่าวอยู่ 2 วิธีคือ

- ใช้แฟคเตอร์ปรับค่า (Regularization factor, λ) [12] โดยเป็นจำนวนที่มีค่าต่ำๆ บวกกับค่า $\mathbf{H}^{\mathbf{T}}\mathbf{H}$ ก่อนที่จะทำการแปลงผกผันดังนี้ $(\mathbf{H}^{\mathbf{T}}\mathbf{H}+\boldsymbol{\lambda}\mathbf{I})^{-1}$ ซึ่งช่วยให้เป็นเมทริกซ์แบบเต็มแรงก์ที่สามารถแปลงผกผันได้
- ใช้วิธี Singular Value Decomposition (SVD) ในการประมาณค่าเมทริกซ์ผกผัน [12]

2.3 โมเดล Online Sequential Extreme Learning Machine (OS-ELM)

โมเดล ELM เป็นการเรียนรู้แบบ Batch learning หรือ Offline learning คือมีข้อมูลที่มีทั้งหมดในการฝึกสอนโมเดลและให้โมเดลทำงาน โดยระหว่างที่โมเดลทำงานจะไม่สามารถเรียนรู้เพิ่มเติมจากข้อมูลที่เข้ามาใหม่ได้ ซึ่งในการใช้งานจริงบางครั้งข้อมูลที่เข้ามาใหม่นี้อาจมีคุณลักษณะที่แตกต่างข้อมูลที่ใช้ฝึกสอนตอนเริ่มต้น โมเดลจึงจำเป็นจะต้องใช้ข้อมูลที่เข้ามาใหม่ในการฝึกสอนโมเดล เพื่อให้โมเดลสามารถปรับเปลี่ยนค่าพารามิเตอร์ โดยการที่โมเดลสามารถเรียนรู้จากข้อมูลที่รับเข้ามาใหม่ดังกล่าวจะเรียกว่าเป็นการเรียนรู้แบบออนไลน์ (Online learning) หรือเป็นการเรียนรู้แบบเพิ่มขึ้น (Incremental learning)

มีงานวิจัยที่นำเสนอการปรับปรุงโมเดล ELM ให้สามารถทำงานแบบ Incremental learning ได้ และเรียกโมเดลดังกล่าวว่า Online Sequential Extreme Learning Machine (OS-ELM) [12] โดยใช้

หลักการเรียกซ้ำ (recursive) จากสมการที่ 2.7 ในการทำงานรอบแรกของโมเดลจะเป็นดังนี้ $\beta_0 = K_0^{-1} H_0^T Y_0$ โดย $K_0 = H_0^T H_0$ โดยตัวห้อย 0 หมายถึง พารามิเตอร์ในการทำงานเริ่มต้น (ครั้งที่ 0) เมื่อมีข้อมูลชุดใหม่เพิ่มเข้ามาทำให้เมทริกซ์ $\mathbf{H}$ และเมทริกซ์ $\mathbf{Y}$ เปลี่ยนแปลง ซึ่งสามารถคำนวณค่า β_1 ได้ดังนี้

$$\boldsymbol{\beta}_1 = \mathbf{K}_1^{-1}\begin{bmatrix}\mathbf{H}_0\\ \mathbf{H}_1\end{bmatrix}^{\mathbf{T}}\begin{bmatrix}\mathbf{Y}_0\\ \mathbf{Y}_1\end{bmatrix} \tag{2.8}$$

โดย $\mathbf{K}_1 = \begin{bmatrix}\mathbf{H}_0\\ \mathbf{H}_1\end{bmatrix}^{\mathbf{T}}\begin{bmatrix}\mathbf{H}_0\\ \mathbf{H}_1\end{bmatrix}$

$$\mathbf{K}_1 = \begin{bmatrix}\mathbf{H}_0^{\mathbf{T}} & \mathbf{H}_1^{\mathbf{T}}\end{bmatrix}\begin{bmatrix}\mathbf{H}_0\\ \mathbf{H}_1\end{bmatrix}$$

$$\mathbf{K}_1 = \mathbf{K}_0 + \mathbf{H}_1^{\mathbf{T}}\mathbf{H}_1 \tag{2.9}$$

และ $\begin{bmatrix}\mathbf{H}_0\\ \mathbf{H}_1\end{bmatrix}^{\mathbf{T}}\begin{bmatrix}\mathbf{Y}_0\\ \mathbf{Y}_1\end{bmatrix} = \mathbf{H}_0^{\mathbf{T}}\mathbf{Y}_0 + \mathbf{H}_1^{\mathbf{T}}\mathbf{Y}_1$

$$\begin{bmatrix}\mathbf{H}_0\\ \mathbf{H}_1\end{bmatrix}^{\mathbf{T}}\begin{bmatrix}\mathbf{Y}_0\\ \mathbf{Y}_1\end{bmatrix} = \mathbf{K}_0\mathbf{K}_0^{-1}\mathbf{H}_0^{\mathbf{T}}\mathbf{Y}_0 + \mathbf{H}_1^{\mathbf{T}}\mathbf{Y}_1$$

$$\begin{bmatrix}\mathbf{H}_0\\ \mathbf{H}_1\end{bmatrix}^{\mathbf{T}}\begin{bmatrix}\mathbf{Y}_0\\ \mathbf{Y}_1\end{bmatrix} = \mathbf{K}_0\boldsymbol{\beta}_0 + \mathbf{H}_1^{\mathbf{T}}\mathbf{Y}_1$$

$$\begin{bmatrix}\mathbf{H}_0\\ \mathbf{H}_1\end{bmatrix}^{\mathbf{T}}\begin{bmatrix}\mathbf{Y}_0\\ \mathbf{Y}_1\end{bmatrix} = (\mathbf{K}_1 - \mathbf{H}_1^{\mathbf{T}}\mathbf{H}_1)\boldsymbol{\beta}_0 + \mathbf{H}_1^{\mathbf{T}}\mathbf{Y}_1$$

$$\begin{bmatrix}\mathbf{H}_0\\ \mathbf{H}_1\end{bmatrix}^{\mathbf{T}}\begin{bmatrix}\mathbf{Y}_0\\ \mathbf{Y}_1\end{bmatrix} = \mathbf{K}_1\boldsymbol{\beta}_0 - \mathbf{H}_1^{\mathbf{T}}\mathbf{H}_1\boldsymbol{\beta}_0 + \mathbf{H}_1^{\mathbf{T}}\mathbf{Y}_1 \tag{2.10}$$

แทนค่าสมการที่ 2.9 และ 2.10 ลงในสมการที่ 2.8 จะได้ว่า

$$\boldsymbol{\beta}_1 = \mathbf{K}_1^{-1}(\mathbf{K}_1\boldsymbol{\beta}_0 - \mathbf{H}_1^{\mathbf{T}}\mathbf{H}_1\boldsymbol{\beta}_0 + \mathbf{H}_1^{\mathbf{T}}\mathbf{Y}_1)$$

$$\boldsymbol{\beta}_1 = \boldsymbol{\beta}_0 - \mathbf{K}_1^{-1}\mathbf{H}_1^{\mathbf{T}}\mathbf{H}_1\boldsymbol{\beta}_0 + \mathbf{K}_1^{-1}\mathbf{H}_1^{\mathbf{T}}\mathbf{Y}_1$$

$$\boldsymbol{\beta}_1 = \boldsymbol{\beta}_0 + \mathbf{K}_1^{-1}\mathbf{H}_1^{\mathrm{T}}(\mathbf{Y}_1 - \mathbf{H}_1\boldsymbol{\beta}_0) \tag{2.11}$$

$$\text{โดย} \quad \mathbf{K}_1 = \mathbf{K}_0 + \mathbf{H}_1^{\mathrm{T}}\mathbf{H}_1$$

จากสมการที่ 2.11 สามารถจัดให้เป็นรูปแบบทั่วไปได้ดังนี้

$$\boldsymbol{\beta}_{k+1} = \boldsymbol{\beta}_k + \mathbf{K}_{k+1}^{-1}\mathbf{H}_{k+1}^{\mathrm{T}}(\mathbf{Y}_{k+1} - \mathbf{H}_{k+1}\boldsymbol{\beta}_k) \tag{2.12}$$

$$\text{โดย} \quad \mathbf{K}_{k+1} = \mathbf{K}_k + \mathbf{H}_{k+1}^{\mathrm{T}}\mathbf{H}_{k+1}$$

โดย k คือ รอบที่หรือครั้งที่ของข้อมูลที่โมเดลรับเข้ามาโดยขนาดของข้อมูลที่รับเข้ามาแต่และครั้งไม่จำเป็นต้องมีขนาดเท่ากันก็ได้ ซึ่ง $k = 0$ จะหมายถึงการทำงานเริ่มต้นในรอบแรก

สรุปได้ว่าโมเดล OS-ELM มีวิธีการทำงานที่แบ่งออกเป็นสองขั้นตอนคือ

ขั้นตอนที่ 1 ตั้งค่าพารามิเตอร์เริ่มต้น ขั้นตอนนี้จะเหมือนกันกับโมเดล ELM ปกติ คือใช้ข้อมูลฝึกสอนเริ่มต้นเพื่อคำนวณค่า β_0 โดยใช้สมการที่ 2.7

ขั้นตอนที่ 2 ใช้ข้อมูลที่รับเพิ่มมาและข้อมูลพารามิเตอร์ปัจจุบันของโมเดลมาคำนวณค่า β_{k+1} ตามสมการที่ 2.12

2.4 งานวิจัยที่เกี่ยวข้อง

บทความและงานวิจัยที่เกี่ยวข้องกับการพัฒนาปรับปรุงโมเดล OS-ELM สามารถรวบรวมและสรุปได้ดังตารางที่ 2.1 ดังนี้

ตารางที่ 2.1 งานวิจัยที่เกี่ยวข้อง (ต่อ)

ปี	นักวิจัย	สาระสำคัญ
2009	van Heeswijk Mark, Miche Yoan, Lindh-Knuutila T., Hilbers Peter, Honkela Timo, Oja E. and Lendasse Amaury [19]	คณะนักวิจัยได้นำเสนอชุดของโมเดล ELM (Ensemble of ELM) ที่ใช้สำหรับพยากรณ์อนุกรมเวลา โดยชุดของโมเดลดังกล่าวประกอบด้วยโมเดล ELM จำนวนหลายตัวซึ่งในบทความมีการทดสอบผลของจำนวนของโมเดลที่มีต่อความแม่นยำในการพยากรณ์ ค่าที่แต่ละโมเดลพยากรณ์ได้จะนำมาคำนวณผ่านวิธีเฉลี่ยถ่วงน้ำหนัก (Weighted

ตารางที่ 2.1 งานวิจัยที่เกี่ยวข้อง (ต่อ)

ปี	นักวิจัย	สาระสำคัญ
		average) เป็นค่าพยากรณ์ของชุดโมเดล โดยค่าถ่วงน้ำหนักนี้จะปรับเปลี่ยนไปตามค่าความผิดพลาดในการพยากรณ์ของรอบที่ผ่านมา คณะนักวิจัยได้ทดสอบชุดของโมเดลที่นำเสนอกับข้อมูลอนุกรมเวลาที่นิ่ง (Stationary) และข้อมูลอนุกรมเวลาที่ไม่นิ่ง (Non-stationary) ซึ่งได้ผลที่ดี
2010	S. J. Pan and Q. Yang [20]	บทความนี้เป็นบทความทางด้านการเรียนรู้แบบถ่ายโอน (Transfer learning) ที่ถูกอ้างถึงมากที่สุด โดยได้ให้คำนิยามศัพท์ที่เป็นพื้นฐาน Source, Target, Domain, และ Task อธิบายรูปแบบการเรียนรู้แบบถ่ายโอนที่เป็น Inductive transfer learning และ Transductive transfer learning อธิบายแนวทางในการเรียนรู้แบบถ่ายโอน 4 แนวทางได้แก่ 1) Transferring of Instances เป็นการเลือกข้อมูลบางตัวจาก Source domain ไปฝึกสอนที่ Target domain โดยผ่านการกำหนดค่าน้ำหนัก 2) Transferring of feature representation เป็นการนำจุดเด่นของของข้อมูลจาก Source domain ไปฝึกสอนที่ Target domain โดยใช้ Feature function บางอย่าง 3) Transferring parameter เป็นการนำค่าพารามิเตอร์ของโมเดลใน Source domain ไปปรับใช้กับโมเดลใน Target domain และ 4) Transferring relational เป็นการหาความสัมพันธ์ระหว่าข้อมูลใน Source domain และ Target domain

ตารางที่ 2.1 งานวิจัยที่เกี่ยวข้อง (ต่อ)

ปี	นักวิจัย	สาระสำคัญ
2010	N. Liu and H. Wang [21]	คณะนักวิจัยได้นำเสนอการนำโมเดล ELM หลายตัวมาทำงานร่วมกันเป็นชุด (Ensemble) โดยได้รวมขั้นตอนการเลือกค่าน้ำหนักของชั้นอินพุต (Input weight) และการตรวจสอบความสมเหตุสมผล (Validation) ไว้ในขั้นตอนการฝึกสอนโมเดล และได้นำโมเดลไปทดสอบด้วยการจำแนกข้อมูล (Classification) ด้วยชุดข้อมูลที่หลากหลายพบว่าโมเดลสามารถทำงานได้เป็นอย่างดี
2013	Kasun Liyanaarachchi, Zhou Hongming, Huang Guang-Bin, Vong Chi-Man [22]	คณะนักวิจัยได้นำเสนอเทคนิคการเข้ารหัสอัตโนมัติ (Autoencoder) โดยใช้โมเดล ELM และใช้โมเดลดังกล่าว ในชั้นซ่อนของ ELM แทนการสุ่มค่าพารามิเตอร์ ซึ่งพบว่าโมเดล ELM ที่ใช้การเข้ารหัสอัตโนมัติ (Autoencoder) ในชั้นซ่อนแทนการสุ่มค่าพารามิเตอร์ สามารถทำงานบนชุดข้อมูล (Data set) MNIST ได้อย่างรวดเร็วและแม่นยำ
2013	Y. Simmhan and M. U. Noor [23]	บทความนี้ได้นำเสนอการใช้เทคนิคการจัดกลุ่มข้อมูลแบบเรียนรู้เพิ่มขึ้น (Incremental clustering) ในการจัดกลุ่มผู้ใช้ไฟฟ้า และนำข้อมูลการใช้ไฟฟ้าของแต่ละกลุ่มมาสร้างโมเดลพยากรณ์การใช้ไฟฟ้าเฉพาะกลุ่ม นอกจากนี้ในบทความยังได้นำเสนอวิธีการวัดความสัมพันธ์ของข้อมูลเพื่อให้จัดกลุ่มเรียกว่า Affinity score
2014	Pak Kin Wong, Chi Man Vong, Xiang Hui Gao and Ka In Wong [24]	โมเดล OS-ELM จำเป็นต้องใช้ข้อมูลในการฝึกสอนเริ่มต้น ซึ่งทางคณะนักวิจัยมองว่าในการประยุกต์ใช้งานจริงบางครั้งไม่สามารถหาข้อมูลฝึกสอนเริ่มต้นได้

ตารางที่ 2.1 งานวิจัยที่เกี่ยวข้อง (ต่อ)

ปี	นักวิจัย	สาระสำคัญ
		จึงได้พัฒนาโมเดล Fully Online Sequential Extreme Learning Machine (FOS-ELM) ที่ไม่จำเป็นต้องใช้ข้อมูลในการฝึกสอนเริ่มต้น โดยการตั้งค่าพารามิเตอร์ตั้งต้นของโมเดลให้เท่ากับศูนย์ และทางคณะนักวิจัยได้นำโมเดล FOS-ELM นี้ไปทดสอบโดยการจำลองการทำงานของชุดควบคุมอัตราส่วนของอากาศและเชื้อเพลิงของเครื่องยนต์โดยเป็นการควบคุมแบบปรับตัวได้ (Adaptive Control)
2015	S. Scardapane,n D. Comminiello, M. Scarpiniti and A. Uncini [25]	คณะนักวิจัยได้นำเสนอการปรับปรุงโมเดล ELM โดยการใช้ Kernel recursive least square และเรียกว่า Kernel Online Sequential ELM (KOS-ELM) ซึ่งทำให้โมเดล ELM ทำงานคล้ายกับโมเดล OS-ELM ต่างกันเพียง อินพุตของ KOS-ELM จะผ่าน kernel function ก่อน ส่วนอินพุตของ OS-ELM จะคูณกับค่าน้ำหนัก (weight) ซึ่งเป็นฟังก์ชันเชิงเส้นก่อน จากการทดลองโมเดล KOS-ELM สามารถทำงานจำแนกข้อมูล (Classification) และวิเคราะห์การถดถอย (Regression) ของข้อมูลได้ดีเช่นเดียวกับ OS-ELM
2016	Guo, W., Xu, T. and Tang, K [26]	คณะนักวิจัยได้นำเสนอข้อด้อยของโมเดล OS-ELM ที่ใช้วิธีการของกำลังสองต่ำสุด (Least square) ในการคำนวณค่าพารามิเตอร์ ว่าไม่มีความทนทาน (Robustness) ต่อข้อมูลที่มีค่าผิดปกติ (outlier data) และได้นำเสนอวิธีการคำนวณค่าพารามิเตอร์โดยใช้ M-estimator ซึ่งในการคำนวณจะมีการพิจารณาค่าความผิดพลาดกับค่าขีดเริ่มเปลี่ยน

ตารางที่ 2.1 งานวิจัยที่เกี่ยวข้อง (ต่อ)

ปี	นักวิจัย	สาระสำคัญ
		(Threshold) ที่ตั้งไว้ หากค่าความผิดพลาดต่ำกว่าค่า Threshold โมเดลจะคำนวณค่าพารามิเตอร์ตามปกติ หากค่าความผิดพลาดสูงกว่าค่า Threshold จะถือว่าเป็นข้อมูลที่มีค่าผิดปกติและคำนวณพารามิเตอร์ด้วยวิธีที่แตกต่างออกไป ทำให้โมเดลมีความทานต่อข้อมูลที่มีค่าผิดปกติ
2016	J. Tang, C. Deng and G. Huang [27]	คณะนักวิจัยได้นำเสนอการพัฒนาโมเดล ELM ที่ปกติจะมีเพียงชั้นซ่อนเดียวให้มีจำนวนชั้นซ่อนเพิ่มมากขึ้นสำหรับทำงานกับข้อมูลที่มีความไม่เป็นเชิงเส้นเช่นข้อมูลรูปภาพหรือข้อมูลเสียงและเรียกโมเดลนี้ว่า Hierarchical ELM (H-ELM) โดยในชั้นซ่อนจะใช้เทคนิคการเข้ารหัสอัตโนมัติ (Autoencoder) สำหรับโมเดล ELM ซึ่งสามารถทำได้หลายชั้นและจัดเป็นการเรียนรู้แบบไม่มีผู้ฝึกสอน (Unsupervised learning) ส่วนในชั้นเอาต์พุตจะเป็นโมเดล ELM แบบปกติซึ่งเป็นการเรียนรู้แบบมีผู้ฝึกสอน (Supervised learning) ทำให้โมเดล H-ELM สามารถทำงานกับข้อมูลที่ไม่เป็นเชิงเส้นเช่นเดียวกับโมเดลเรียนรู้เชิงลึกทั่วไป (Deep Learning) และสามารถเรียนรู้ได้เร็วแบบโมเดล ELM
2016	K. Phurattanaprapin and P. Horata [28]	โมเดล Hierarchical ELM (H-ELM) ประกอบด้วยชั้นซ่อนจำนวนหลายชั้นที่ใช้เทคนิคการเข้ารหัสอัตโนมัติ (Autoencoder) สำหรับโมเดล ELM ที่เรียนรู้แบบไม่มีผู้ฝึกสอน (Unsupervised learning) และชั้นเอาต์พุตที่เป็นโมเดล ELM ซึ่งปกติจะมีเพียงชั้นเดียว

ตารางที่ 2.1 งานวิจัยที่เกี่ยวข้อง (ต่อ)

ปี	นักวิจัย	สาระสำคัญ
		เรียนรู้แบบมีผู้ฝึกสอน (Supervised learning) ทางคณะนักวิจัยได้นำเสนอวิธีการปรับปรุงโมเดล H-ELM ดังกล่าว โดยการเพิ่มจำนวนชั้น (layer) ของส่วนเอาต์พุตในโมเดล H-ELM เดิม และเรียกโมเดลใหม่นี้ว่า Extended Hierarchical Extreme Learning Machine (EH-ELM) โมเดล EH-ELM ถูกทดสอบกับงานจำแนกข้อมูล (Classification) ด้วยชุดข้อมูลต่างๆ พบว่ามีความแม่นยำสูงกว่า H-ELM ปกติ
2017	Jin-Man Park , Jong-Hwan Kim [29]	คณะนักวิจัยได้ปรับปรุงโมเดล OS-ELM ให้มีโครงสร้างเหมือนโครงข่ายประสาทเทียมแบบวนซ้ำ (Recurrent artificial neural network) และเรียกว่า Online Recurrent Extreme Learning Machine (OR-ELM) กล่าวคือ ใช้ข้อมูลเอาต์พุตจากการพยากรณ์ในรอบที่ผ่านมาเป็นข้อมูลอินพุตสำหรับพยากรณ์ในรอบปัจจุบันด้วยนอกจากนี้ยังได้ใช้เทคนิคการเข้ารหัสอัตโนมัติ (Autoencoder) ในการตั้งค่าพารามิเตอร์ในชั้นซ่อนอีกด้วย โมเดล OR-ELM ถูกทดสอบโดยนำไปพยากรณ์อนุกรมเวลาของชุดข้อมูล New-York city passenger และพบว่าสามารถพยากรณ์ได้แม่นยำกว่าโมเดล OS-ELM ปกติ
2017	Zhang, H., Zhang, S. and Yin, Y. [30]	การเรียนรู้เพิ่มขึ้นของโมเดล OS-ELM จะทำการปรับค่าพารามิเตอร์จากข้อมูลที่ได้รับ โดยข้อมูลสะสมในอดีตและข้อมูลใหม่จะมีผลต่อการปรับค่าพารามิเตอร์เท่าเทียมกัน แต่เพื่อให้การปรับตัวทำได้รวดเร็วยิ่งขึ้น ทางคณะวิจัยจึงได้นำเสนอพารามิเตอร์มาตัวหนึ่งซึ่ง

ตารางที่ 2.1 งานวิจัยที่เกี่ยวข้อง (ต่อ)

ปี	นักวิจัย	สาระสำคัญ
		เรียกว่า Forgetting factor โดยพารามิเตอร์ดังกล่าวจะเป็นตัวช่วยลดผลของข้อมูลสะสมในอดีตต่อการปรับค่าพารามิเตอร์ของโมเดลทำให้โมเดลปรับตัวตามข้อมูลใหม่ได้รวดเร็วมากขึ้น
2017	Salaken, S., Khosravi, A., Nguyen, T. and Nahavandi, S [31]	บทความนี้ได้รวบรวมข้อมูลของงานที่ใช้การเรียนรู้แบบถ่ายโอน (Transfer learning) กับโมเดล ELM ได้แก่ การชดเชยการเลื่อนค่า (Drift) ของเซนเซอร์จมูกอิเล็กทรอนิกส์, การจำแนกข้อมูลที่มีการเปลี่ยนแปลง (Concept drift), การระบุตำแหน่งด้วยสัญญาณ Wi-Fi, การรู้จำกิจกรรม (Activity recognition) ของบุคคล, การทำคอมพิวเตอร์วิทัศน์ เป็นต้น โดยในแต่ละงานมีสิ่งหนึ่งที่คล้ายคลึงกันคือข้อมูลจะมีการเปลี่ยนแปลงเลื่อนค่า ดังนั้นการเรียนรู้แบบถ่ายโอน ความรู้จากแหล่งอื่นจะแก้ปัญหาดังกล่าวได้
2017	W. Zhu, W. Yu, B. Kan and G. Liu [32]	บทความนี้นำเสนอการจัดกลุ่มข้อมูลที่ได้จากมาตรวัดไฟฟ้าอัจฉริยะ (Smart meter) เพื่อแบ่งลักษณะของผู้ใช้ไฟฟ้าที่ใช้ขั้นตอนวิธีที่เรียกว่า Modified streaming k-means ซึ่งเป็นวิธีจัดกลุ่มข้อมูลที่สามารถเรียนรู้เพิ่มขึ้นได้ (Incremental clustering) พัฒนาต่อยอดมาจากขั้นตอนวิธี Streaming k-means โดยจุดที่เพิ่มเติมคือ การปรับปรุงค่ากลุ่มของข้อมูล (Clustering index) ในอดีตที่เคยจัดกลุ่มไว้แล้วในทุกครั้งที่มีข้อมูลชุดใหม่เข้ามาจัดกลุ่ม โดยข้อดีที่ได้รับคือการจัดกลุ่มจะมีประสิทธิผลดีกว่าแบบ

ตารางที่ 2.1 งานวิจัยที่เกี่ยวข้อง (ต่อ)

ปี	นักวิจัย	สาระสำคัญ
		Streaming k-means ปกติ แต่ข้อเสียคือมีต้นทุนในการประมวลผลที่สูงขึ้น
2018	M. Roy, S. K. Bose, B. Kar, P. K. Gopalakrishnan and A. Basu [33]	คณะนักวิจัยได้นำเสนอการสกัดคุณลักษณะ (Feature extraction) ของข้อมูลเพื่อตรวจสอบความผิดปกติ (Anomaly detection) โดยการนำโมเดลเข้ารหัสอัตโนมัติ (Autoencoder) เรียงซ้อน (stacked) เข้ากับโมเดล OS-ELM โดยโมเดล Autoencoder นี้เป็นโมเดลแบบดั้งเดิมทำการฝึกสอนแบบออฟไลน์ก่อนนำไปใช้งาน โดยนำเอาต์พุตของชั้นซ่อนของ Autoencoder ป้อนเป็นอินพุตให้ OS-ELM คณะนักวิจัยได้ทดสอบระบบดังกล่าวกับชุดข้อมูล NASA Bearing Dataset พบว่าสามารถตรวจสอบความผิดปกติของตลับลูกปืนในชุดข้อมูลได้เป็นอย่างดี
2018	Ribeiro, M., Grolinger, K., ElYamany, H., Higashino, W. and Capretz, M. [34]	คณะนักวิจัยได้นำเสนอขั้นตอนวิธีที่ชื่อว่า Hephaestus ซึ่งเป็นการเรียนรู้แบบถ่ายโอนในรูปแบบ Inductive transfer learning เหมาะสำหรับข้อมูลอนุกรมเวลา ที่นำค่าตามฤดูกาล (Seasonal) และค่าแนวโน้ม (Trend) ของข้อมูลมาพิจารณา โดยขั้นตอนวิธี Hephaestus นี้ถูกใช้ในการทำ pre-processing และ post-processing ข้อมูล ร่วมกับโมเดลการเรียนรู้ของเครื่องแบบใดก็ได้ โดยคณะนักวิจัยได้ทดสอบวิธีการที่นำเสนอนี้กับการพยากรณ์การใช้พลังงานของอาคารโรงเรียนพบว่าการใช้ Hephaestus สามารถเพิ่มความแม่นยำในการพยากรณ์ได้ถึง 11.2%

ตารางที่ 2.1 งานวิจัยที่เกี่ยวข้อง (ต่อ)

ปี	นักวิจัย	สาระสำคัญ
2018	C. Chen, B. Jiang and X. Jin [35]	คณะนักวิจัยได้นำเสนอการเรียนรู้แบบถ่ายโอน (Transfer learning) ด้วยแนวทาง Transferring parameter โดยใช้โมเดล ELM จุดเด่นของงานชิ้นนี้คือการใช้ Projection matrix เชื่อมโยงค่าพารามิเตอร์ของโมเดลใน Source domain ไปยังค่าพารามิเตอร์ของโมเดลใน Target domain โดยจากการทดลองให้โมเดลจำแนกข้อมูล (Classification) พบว่าโมเดลที่นำเสนอมีความแม่นยำมากกว่าโมเดล ELM, Support vector machine (SVM) ที่ไม่ได้ทำการเรียนรู้แบบถ่ายโอน และยังสูงกว่าโมเดลเรียนรู้แบบถ่ายโอน Geodesic Flow Kernel (GFK), Max-Margin Domain Transform (MMDT) และ Cross-Domain Landmark Selection (CDLS)
2019	W. Cao, Z. Ming, Z. Xu, J. Zhang and Q. Wang [36]	โดยปกติค่า Forgetting factor ที่ใช้กับโมเดล OS-ELM จะถูกตั้งให้เป็นค่าคงที่ ซึ่งหากปรับตั้งค่าไม่เหมาะสมจะทำให้โมเดลมีความผิดพลาดในการพยากรณ์ค่าได้ ในบทความนี้คณะนักวิจัยจึงได้นำเสนอวิธีการปรับค่า Forgetting factor โดยอัตโนมัติตามค่าความผิดพลาดในการพยากรณ์ของโมเดล กล่าวคือหากโมเดลมีความผิดพลาดในการพยากรณ์มากขึ้นค่า Forgetting factor จะลดลงทำให้โมเดลสามารถปรับตัวตามข้อมูลชุดใหม่ได้รวดเร็วขึ้น ในทางตรงกันข้าม หากโมเดลมีความผิดพลาดในการพยากรณ์ลดลงค่า Forgetting factor จะสูงขึ้น

ตารางที่ 2.1 งานวิจัยที่เกี่ยวข้อง (ต่อ)

ปี	นักวิจัย	สาระสำคัญ
2019	A. Hammoudeh, M. Fraihat and M. Almomani [37]	บทความนี้ได้นำสนอชุดของโมเดล (Ensemble model) ที่ทำงานแบบเลือกค่าของโมเดลแค่บางตัวในชุดโมเดลมาพิจารณา โดยเลือกโมเดลแต่ละตัวในชุดโมเดลเป็นคนละชนิดได้แก่ Artificial neural network (ANN), Convolution neural network (CNN), Support vector machine (SVM) และ Random forecast (RF) ให้ทำงานร่วมกัน โดยมีตัว Selector ทำหน้าที่เลือกเอาต์พุตของแต่ละโมเดลว่าจะนำมาพิจารณาเป็นค่าพยากรณ์หรือไม่ โดยตัว Selector นี้จะมีผ่านการฝึกสอนแบบมีผู้ฝึกสอน (Supervised learning) คณะนักวิจัยได้ทดสอบโมเดลนี้กับงานจำแนกข้อมูล (Classification) และพบว่าโมเดลสามารถทำงานได้เป็นอย่างดี
2019	H. T. Thu Thuy, D. T. Anh and V. T. Ngoc Chau [38]	บทความนี้นำเสนอการปรับปรุงขั้นตอนวิธีจัดกลุ่มข้อมูล (Clustering) Leader algorithm และตั้งชื่อใหม่ว่า Improved leader algorithms (I-Leader) เพื่อใช้สำหรับข้อมูลอนุกรมเวลา Leader algorithm เป็นการจัดกลุ่มข้อมูลที่สามารถเรียนรู้แบบเพิ่มขึ้น (Incremental clustering) ซึ่งจะประมวลผลข้อมูลแต่ละตัวเพียงแค่ครั้งเดียว (Single pass) ไม่จำเป็นต้องกำหนดจำนวนกลุ่มไว้ล่วงหน้า แต่ข้อด้อยของวิธีนี้คือข้อมูลอาจถูกแบ่งกลุ่มออกเป็นจำนวนมาก วิธีการ กลุ่มนักวิจัยจึงพัฒนา I-Leader ขึ้นมาเพื่อแก้ไขปัญหาดังกล่าวทำให้การจัดกลุ่มข้อมูลมีความเหมาะสมยิ่งขึ้น

ตารางที่ 2.1 งานวิจัยที่เกี่ยวข้อง (ต่อ)

ปี	นักวิจัย	สาระสำคัญ
2020	Fan C., Sun Y., Xiao F., Ma J., Lee D., Wang J. and Tseng Y. [39]	ในบทความนี้ได้นำเสนอการทดลองพยากรณ์การใช้พลังงานของอาคาร โดยเป็นอาคารเก่าที่ไม่ได้ติดตั้งระบบตรวจวัดและบันทึกการใช้พลังงานทำให้มีข้อมูลบางช่วงขาดหายไป และอาคารใหม่ที่มีระบบตรวจวัดและบันทึกการใช้พลังงานแต่มีข้อมูลที่บันทึกอยู่เป็นจำนวนไม่มาก โดยแนวทางการทำการเรียนรู้แบบถ่ายโอนในงานวิจัยนี้ใช้เป็น Transferring parameter ซึ่งจะเรียกว่า Network-based transfer ซึ่งใช้งานในสองลักษณะคือ 1) ใช้โมเดลที่ฝึกสอนจาก Source domain ซึ่งเป็นโมเดลที่มีหลายชั้น (Multi-layer) โดยให้ค่าพารามิเตอร์ในชั้นแรกๆ คงที่ทำหน้าที่เป็นตัวสกัดคุณลักษณะ (Feature extraction) ของข้อมูลส่วนชั้นสุดท้ายนำมาทำการฝึกสอนและปรับค่าพารามิเตอร์ตามข้อมูลใน Target domain 2) ใช้พารามิเตอร์ของโมเดลที่ฝึกสอนจาก Source domain เป็นค่าเริ่มต้นในการฝึกสอนและปรับค่าพารามิเตอร์ตามข้อมูลใน Target domain จากผลการทดลองพบว่าทั้งสองลักษณะให้ค่าความแม่นยำใกล้เคียงกัน

จากข้อมูลในการทบทวนวรรณกรรมข้างต้นสามารถจัดกลุ่มของแนวทางในการปรับปรุงโมเดล OS-ELM ได้ 4 แนวทางดังตารางที่ 2.2

ตารางที่ 2.2 สรุปแนวทางการปรับปรุงโมเดล OS-ELM

ลำดับ	แนวทางปรับปรุงโมเดล OS-ELM	บทความ
1	การทำชุดโมเดล (Ensemble model)	[19], [21], [37]
2	การปรับปรุงการฝึกสอนเริ่มต้น (Initial training)	
	- การเรียนรู้แบบถ่ายโอน (Transfer learning)	[20], [31], [34], [35], [39]
	- การทำ Fully Online	[24]
3	การสกัดคุณลักษณะ (Feature extraction)	
	- การเข้ารหัสอัตโนมัติ (Autoencoder)	[22], [27], [28], [33]
	- การนำข้อมูลอินพุตผ่านฟังก์ชันเคอร์เนล (Kernel)	[25]
	- การแบ่งกลุ่มข้อมูล (Clustering)	[23], [32], [38]
4	การปรับฟังก์ชันต้นทุน (Cost function)	
	- การใช้ Forgetting factor	[30], [36]
	- ใช้ขั้นตอนวิธี M-estimator	[26]
	- การปรับโมเดลให้เป็นแบบเวียนซ้ำ (Recurrent)	[29]

การนำแนวทางทั้ง 4 ข้อข้างต้นมาใช้ปรับปรุงโมเดล OS-ELM เพื่อพยากรณ์โหลดไฟฟ้าระยะสั้นนั้น มีประเด็นที่ต้องกล่าวถึงดังต่อนี้

1) การทำชุดโมเดล (Ensemble model) จะช่วยลดข้อผิดพลาดจาการใช้โมเดลเพียงแค่ตัวเดียวในพยากรณ์ โดยจะต้องมีขั้นตอนวิธีเพื่อประมวลผลข้อมูลจากโมเดลแต่ละตัวให้เป็นค่าพยากรณ์รวมของโมเดลซึ่งขั้นตอนวิธีนี้จะต้องสามารถเรียนรู้แบบเพิ่มขึ้นได้ และการใช้โมเดลหลายตัวในการพยากรณ์อาจจะเป็นการเพิ่มระยะเวลาในการประมวลผลซึ่งอาจทำให้จุดเด่นของ OS-ELM หายไป

2) การเรียนรู้แบบถ่ายโอน (Transfer Learning) จะเป็นตัวช่วยให้การใช้งานโมเดลพยากรณ์โหลดไฟฟ้ามีความสะดวกมากยิ่งขึ้น เนื่องจากผู้ใช้งานไม่จำเป็นต้องมีข้อมูลโหลดไฟฟ้าของสถานที่นั้นๆ โดยโมเดลจะถูกฝึกสอนเบื้องต้น (Pre-trained) ด้วยข้อมูลอื่นที่คล้ายกันก่อนถูกติดตั้งใช้งาน แต่ผู้ใช้งานต้องเลือกข้อมูลตัวอย่างที่เหมาะสมใกล้เคียงกับการใช้งานจริงการใช้งานจึงจะได้ผลดี

3) การทำ Fully online โดยการกำหนดค่าเริ่มต้นของพารามิเตอร์บางตัวเป็นศูนย์ทำให้ไม่จำเป็นต้องมีการฝึกสอนเริ่มต้น ทำให้ช่วงเริ่มต้นทำงานโมเดลอาจมีความผิดพลาดในการพยากรณ์สูงมาก โดยเฉพาะข้อมูลมีความซับซ้อนมาก

4) การเข้ารหัสอัตโนมัติ (Autoencoder) เป็นตัวช่วยในการสกัดคุณลักษณะของข้อมูล (Feature extraction) และเพิ่มจำนวนชั้นให้กับโมเดลในการประมวลผลของมูลซึ่งจะให้ผลลัพธ์ที่ดีกว่าการสุ่มค่าพารามิเตอร์ของโมเดล ELM แบบปกติ และทำให้โมเดลสามารถพยากรณ์ข้อมูลที่มีความสัมพันธ์กันแบบไม่เป็นเชิงเส้นได้ดีขึ้น โดยวิธีการเข้ารหัสอัตโนมัตินี้ต้องสามารถเรียนรู้แบบเพิ่มขึ้นได้

5) การนำข้อมูลอินพุตผ่านฟังก์ชันเคอร์เนล (Kernel function) จะช่วยลดมิติ (Dimension) ของข้อมูลอินพุตทำให้โมเดลสามารถประมวลผลข้อมูลได้ง่ายขึ้น โดยเฉพาะข้อมูลที่มีความสัมพันธ์แบบไม่เป็นเชิงเส้น

6) การจัดกลุ่มข้อมูล (Clustering) เป็นวิธีการที่นิยมใช้ในการวิเคราะห์ข้อมูลเพื่อหารูปแบบและจัดกลุ่มข้อมูลที่มีรูปแบบเหมือนกัน โดยขั้นตอนวิธีการจัดกลุ่มข้อมูลนี้สามารถเรียนรู้แบบเพิ่มขึ้นได้ [5], [14], [20] ซึ่งอาจใช้เป็นตัวสกัดคุณลักษณะของข้อมูล (Feature extraction) ก่อนป้อนข้อมูลให้กับชุดโมเดล OS-ELM (OS-ELM Ensemble model) ที่ทำงานแบบเลือกค่า (Selection) ซึ่งจากการทบทวนวรรณกรรมยังไม่พบบทความที่ทำการศึกษาโมเดลในลักษณะดังกล่าวมากนัก

7) การใช้ค่า Forgetting factor จะช่วยให้โมเดลปรับตัวได้เร็วขึ้นเมื่อคุณสมบัติทางสถิติของข้อมูลมีการเปลี่ยนแปลง ซึ่งการเลือกค่าพารามิเตอร์หรือฟังก์ชันที่ใช้ปรับค่าพารามิเตอร์ดังกล่าวหากทำอย่างไม่เหมาะสมจะส่งผลต่อค่าความแม่นยำในการพยากรณ์อย่างมาก

8) การใช้ขั้นตอนวิธี M-estimator เป็นการปรับปรุงวิธีการคำนวณค่าพารามิเตอร์ของโมเดลให้มีความทนทาน (Robustness) ต่อข้อมูลที่มีค่าผิดปกติ (Outlier data)

9) การปรับโมเดลให้มีโครงสร้างเป็น Recurrent เป็นการนำข้อมูลจากการพยากรณ์ในรอบที่แล้วกลับเข้ามาเป็นอินพุต ซึ่งช่วยให้โมเดลสามารถจดจำรูปแบบของข้อมูลได้ในช่วงเวลาที่ยาวนานขึ้น

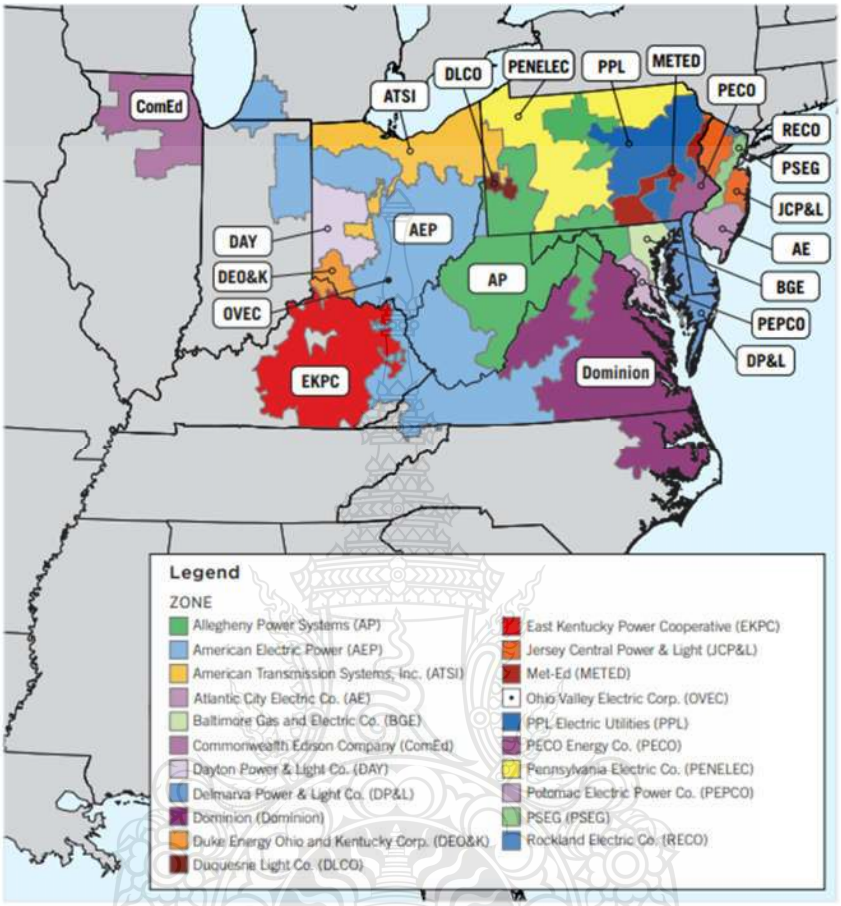
บทที่ 3
วิธีดำเนินการ

ดุษฎีนิพนธ์นี้นำเสนอวิธีการปรับปรุงโมเดล Online Sequential Extreme Learning Machine (OS-ELM) ให้มีความแม่นยำมากขึ้นในการพยากรณ์โหลดโดยใช้แนวทางในการปรับปรุง 4 แนวทางดังที่ได้นำเสนอในบทที่ 2 ซึ่งสามารถสรุปภาพรวมของการทดลองได้ดังตารางที่ 3.1 ดังนี้

ตารางที่ 3.1 ภาพรวมของการทดลองในงานวิจัยนี้

แนวทางการปรับปรุง	หัวข้อที่ทำการทดลอง
การทำโมเดลชุด (Ensemble Model)	เปรียบเทียบความแม่นยำในการพยากรณ์โหลดของโมเดลชุดกับโมเดลเดี่ยว
การปรับปรุงการฝึกสอนเริ่มต้น (Initial training)	หารูปแบบที่เหมาะสมของสัญญาณรบกวนแบบสุ่ม (random noise) เพื่อใช้ในการการสังเคราะห์ข้อมูลตัวอย่างเพื่อฝึกสอนเริ่มต้นให้กับโมเดล OS-ELM
การสกัดคุณลักษณะ (Feature extraction) ของข้อมูลอินพุต	ทดลองใช้วิธีแบ่งกลุ่มข้อมูลอินพุต (Clustering) แล้วส่งข้อมูลอินพุตแต่ละกลุ่มไปให้โมเดล OS-ELM ที่ถูกฝึกสอนให้พยากรณ์เฉพาะข้อมูลกลุ่มนั้นๆ
การปรับฟังก์ชันต้นทุน (Cost function) ของการฝึกสอนโมเดล	ใช้การเรียนรู้ซ้ำบนข้อมูลชุดเดิม (Re-learning) เพื่อให้โมเดล OS-ELM ปรับเข้ากับข้อมูลใหม่ได้ดียิ่งขึ้น

โดยข้อมูลตัวอย่างที่ใช้ในการทดลองนี้มีชื่อว่า Hourly Energy Consumption จาก www.kaggle.com [40] ซึ่งเป็นข้อมูลการใช้ไฟฟ้ารายชั่วโมงในช่วงเดือนตุลาคม พ.ศ. 2547 ถึง กรกฎาคม พ.ศ. 2561 ของรัฐทางชายฝั่งตะวันออกสหรัฐอเมริกา ที่มี PJM Interconnection LLC เป็นผู้ให้บริการส่งจ่ายพลังงานไฟฟ้าให้ภูมิภาค (Regional Transmission Operator, RTO) ดังกล่าว โดยในข้อมูลตัวอย่างนี้มีชุดข้อมูลย่อยอยู่ 10 ชุด ซึ่งแบ่งตามบริษัทที่รับไฟฟ้าไปจำหน่ายให้กับผู้ใช้ไฟคือ AEP, COMD, DAYTON, DEOK, DOM, DUQ, EKPC, FE, NI และ PJM โดย PJM เป็นข้อมูลการใช้ไฟฟ้ารวมทั้งหมดของ PJM Interconnection LLC



รูปที่ 3.1 พื้นที่ให้บริการส่งจ่ายพลังงานไฟฟ้าของ PJM Interconnection

ที่มา: https://www.pjm.com/library/~/media/about-pjm/pjm-zones.ashx

การพยากรณ์โหลดทั้งหมดในงานวิจัยนี้เป็นการพยากรณ์โหลดหนึ่งชั่วโมงถัดไปโดยใช้ข้อมูลอินพุตเป็นการใช้ไฟฟ้ารายชั่วโมงก่อนหน้าที่จะพยากรณ์เป็นเวลา 24 ชั่วโมง ดังนั้นโมเดลทุกตัวในการทดลองจะมีโครงสร้างแบบ 24 อินพุต 1 เอาต์พุต โดยข้อมูลอินพุตจะผ่านการประมวลผลล่วงหน้า (Pre-processing) ดังสมการที่ 3.1 และเอาต์พุตที่ได้จากโมเดลจะผ่านการประมวลผลภายหลัง (Post-processing) ดังสมการที่ 3.2 จึงได้เป็นค่าพยากรณ์โหลดไฟฟ้า ข้อมูลที่นำมาใช้ฝึกสอนและทดสอบการของโมเดลทำงานดังที่กล่าวข้างต้นจะถูกจัดเรียงและแบ่งกลุ่มใหม่เป็นค่าอินพุตและค่าเป้าหมาย (Target) โดยค่าเป้าหมายคือค่าการใช้

ไฟฟ้าในแต่ละชั่วโมงเริ่มตั้งแต่ชั่วโมงที่ 25 และค่าอินพุตของแต่ละเป้าหมายคือค่าการใช้ไฟฟ้ารายชั่วโมงก่อนหน้าค่าเป้าหมายเป็นเวลา 24 ค่าดังแสดงในรูปที่ 3.2

$$x_{i,scaled} = \frac{x_i}{\max(x_1, \cdots, x_{24})}, \qquad i = 1, 2, \cdots 24 \tag{3.1}$$

โดย

$x_{i,scaled}$ คือ ค่าอินพุตที่ผ่านการ Pre-processing แล้ว

x_i คือ ค่าอินพุตจากชุดข้อมูลที่ยังไม่ผ่านการ Pre-processing

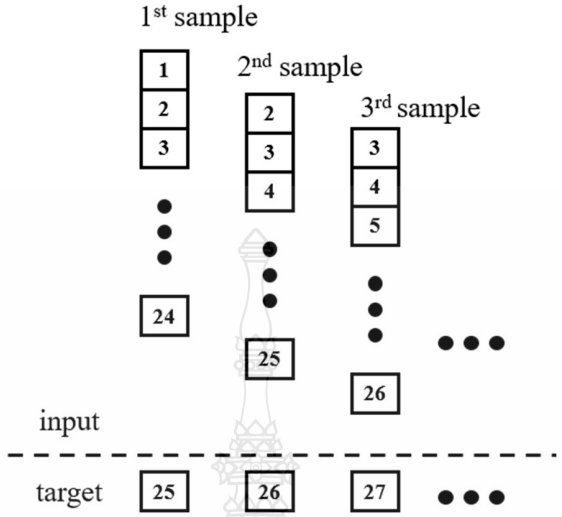
$$y_i = y_{i,scaled} \times \max(x_1, \cdots x_{24}) \tag{3.2}$$

โดย

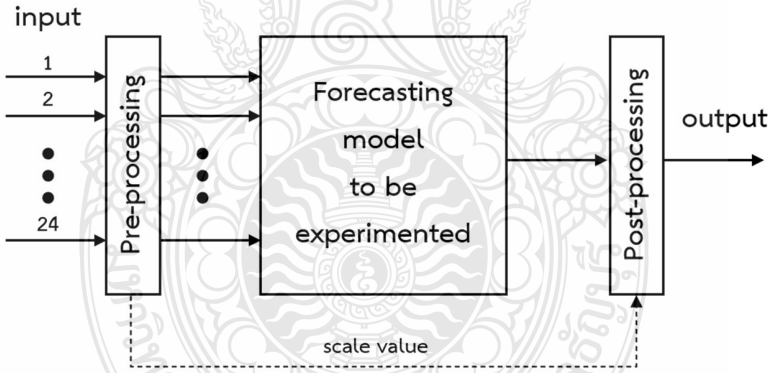
$y_{i,scaled}$ คือ ค่าเอาต์พุตจากโมเดลที่ยังไม่ผ่านการ Post-processing

y_i คือ ค่าเอาต์พุตจากโมเดล

จากสมการที่ 3.1 และ 3.2 ค่าที่ใช้ปรับทั้งข้อมูลอินพุตที่ป้อนเข้าโมเดลและเอาต์พุตที่ออกมาจากโมเดลคือตัวเดียวกันซึ่งได้แก่ค่าที่สูงที่สุดของข้อมูลอินพุตทั้ง 24 ค่าซึ่งในที่นี้จะเรียกว่า scale value โดยทำการคำนวณที่ขั้นตอน Pre-processing และนำไปใช้งานที่ขั้นตอน Pre-processing และ Post-processing ของแต่ละชุดตัวอย่าง ดังรูปที่ 3.3



รูปที่ 3.2 การจัดเรียงข้อมูลให้เป็นค่าอินพุตและค่าเป้าหมายเพื่อใช้ฝึกสอนโมเดล



รูปที่ 3.3 รูปแบบของอินพุตและเอาต์พุตสำหรับโมเดลต่างๆในงานวิจัยนี้

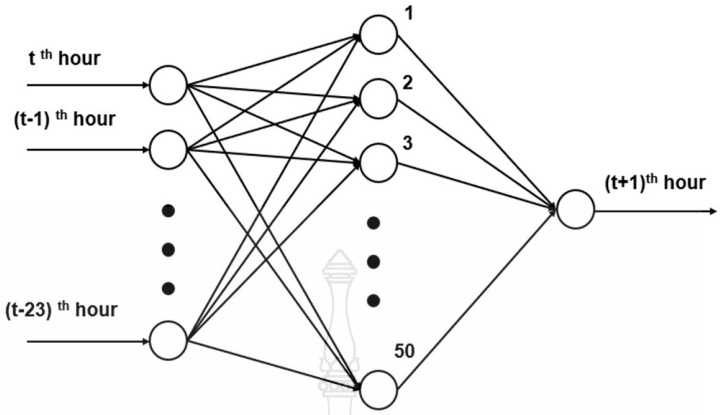
การสร้างโมเดลเพื่อพยากรณ์โหลดและการทดลองต่างๆ ในงานวิจัยนี้ถูกเขียนขึ้นโดยใช้ภาษาไพทอนเวอร์ชั่น 3.8 โดยใช้โปรแกรม Spyder เป็นสภาพแวดล้อมในการพัฒนาแบบเบ็ดเสร็จ (Integrated Development Environment: IDE) โดยทำงานบนเครื่องคอมพิวเตอร์ที่มีหน่วยประมวลผลกลาง Intel Core i5 ,หน่วยความจำขนาด 8 กิกะไบต์ บนระบบปฏิบัติการ Windows 11

3.1 การปรับปรุงโมเดล OS-ELM ด้วยการทำโมเดลชุด (Ensemble Model)

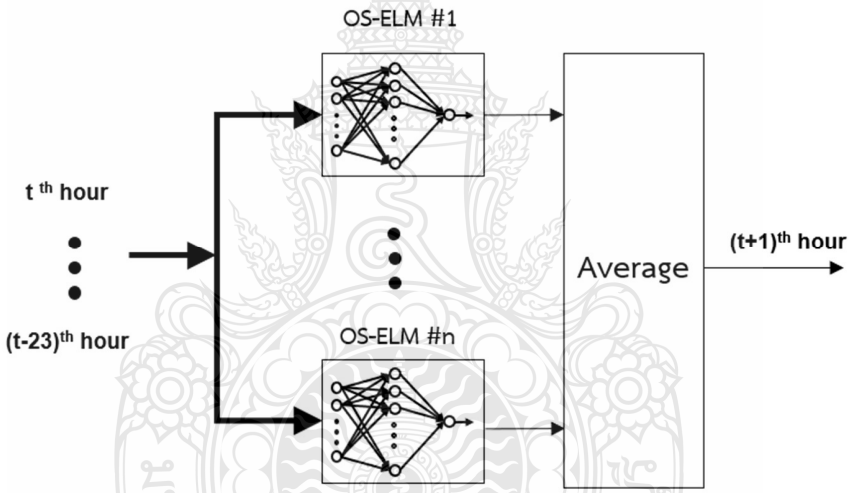
โมเดลชุด (Ensemble Model) เป็นเทคนิคการเพิ่มความแม่นยำในการพยากรณ์ให้กับโมเดลการเรียนรู้ของเครื่อง โดยใช้โมเดลหลายตัวทำงานร่วมกันซึ่งโมเดลเหล่านี้อาจเป็นโมเดลชนิดเดียวกันหรือต่างชนิดกันก็ได้ และโมเดลเหล่านี้จะได้รับการฝึกสอนด้วยข้อมูลตัวอย่างที่แตกต่างกันเพื่อให้ค่าเอาต์พุตของแต่ละโมเดลมีความหลากหลาย ค่าเอาต์พุตสุดท้ายของโมเดลชุดเกิดจากการนำเอาต์พุตของแต่ละโมเดลไปประมวลผลซึ่งอาจใช้วิธีการดูค่าคำตอบส่วนใหญ่ (majority voting) ในกรณีของการจำแนกประเภท (Classification) หรือค่าวิธีเฉลี่ยของทุกโมเดล ในกรณีของการวิเคราะห์การถดถอย (Regression) ซึ่งส่งจะผลให้เอาต์พุตรวมของโมเดลชุดมีความแม่นยำมากกว่าโมเดลเดี่ยว

แต่ในงานวิจัยนี้เป็นการศึกษาพยากรณ์โหลดด้วยโมเดล OS-ELM ที่ทำงานแบบเรียนรู้เพิ่มขึ้น ซึ่งข้อมูลตัวอย่างที่ใช้ในการฝึกสอนโมเดลจะทยอยเข้ามาครั้งละ 1 ตัวอย่าง ดังนั้นหากใช้โมเดลชุดในการพยากรณ์โมเดลย่อยทุกตัวในโมเดลชุดจะได้ถูกฝึกสอนด้วยข้อมูลชุดเดียวกันซึ่งอาจส่งผลให้เอาต์พุตของโมเดลย่อยแต่ละตัวไม่มีความหลากหลายจนทำให้ความแม่นยำในการพยากรณ์ของโมเดลชุดอาจจะไม่แตกต่างจากโมเดลเดี่ยวมากนัก ดังนั้นการทดลองในหัวข้อนี้จึงมีเป้าหมายเพื่อทดสอบว่าการใช้โมเดลชุดในการพยากรณ์โหลดแบบเรียนรู้แบบเพิ่มขึ้นด้วยโมเดล OS-ELM จะมีความแม่นยำมากกว่ากรใช้โมเดลเดี่ยว OS-ELM อย่างไร จำนวนโมเดลย่อยในโมเดลชุดที่เหมาะสมควรมีจำนวนเท่าไร

การทดลองในหัวข้อนี้ใช้โมเดล OS-ELM เป็นโมเดลเดี่ยวที่มีจำนวน 24 โหนดในชั้นอินพุต, 50 โหนดในชั้นซ่อนและ 1 โหนดในชั้นเอาต์พุตดังรูปที่ 3.4 ส่วนโมเดลชุดจะใช้โมเดล OS-ELM ที่มีโครงสร้างเหมือนโมเดลเดี่ยวดังที่กล่าวมาแล้วจำนวน n ตัวมาทำงานร่วมกันโดยในการให้ค่าพยากรณ์ของโมเดลชุดจะใช้การนำค่าเอาต์พุตของโมเดลย่อยทั้ง n ตัวมาหาค่าเฉลี่ยดังรูปที่ 3.5 ซึ่งในการทดลองนี้จะทดลองหาค่า n ที่เหมาะสมโดยพิจารณาจากความแม่นยำในการพยากรณ์โหลดและเวลาที่ใช้ในการคำนวณค่าเอาต์พุต



รูปที่ 3.4 โครงสร้างของโมเดลเดี่ยว OS-ELM ที่ใช้ในการพยากรณ์โหลด



รูปที่ 3.5 โครงสร้างของโมเดลชุด OS-ELM ที่ใช้ในการพยากรณ์โหลด

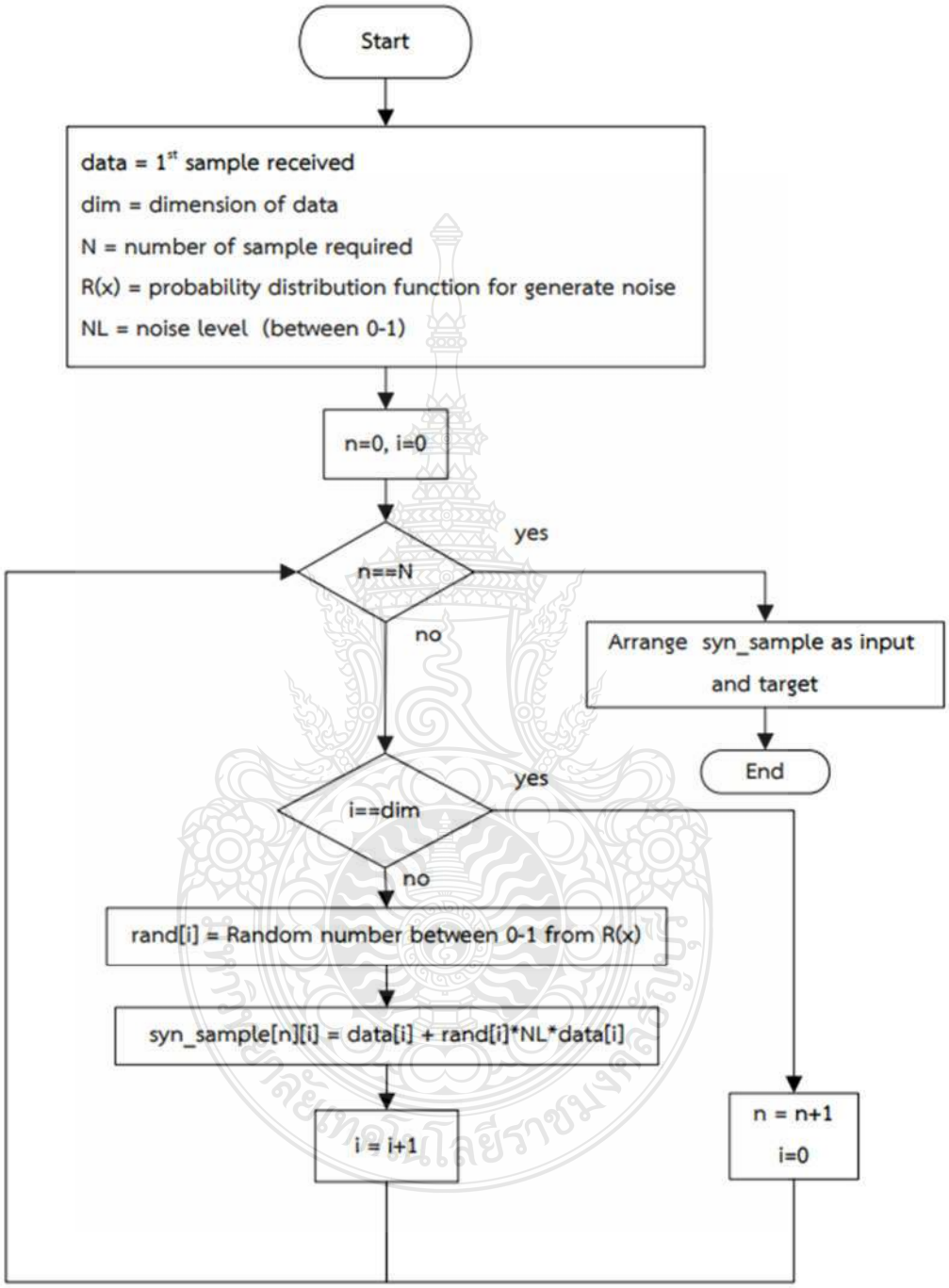
3.2 การปรับปรุงโมเดล OS-ELM ด้วยการปรับปรุงการฝึกสอนเริ่มต้น (Initial Training)

ดังที่กล่าวถึงในบทที่ 2 โมเดล OS-ELM ต้องการข้อมูลตัวอย่างเพื่อทำการฝึกสอนเริ่มต้นก่อนที่โมเดลจะสามารถเรียนรู้เพิ่มขึ้นจากข้อมูลตัวอย่างใหม่ที่รับเข้ามา โดยจำนวนของข้อมูลตัวอย่างที่ใช้สำหรับฝึกสอนเริ่มต้นนี้จะต้องมีไม่ต่ำกว่าจำนวนโหนดในชั้นซ่อนของโมเดล OS-ELM [15] และจำนวนโหนดในชั้นซ่อนของโมเดล OS-ELM ส่งผลต่อความแม่นยำในการพยากรณ์ของโมเดล กล่าวคือหากโมเดลมีจำนวนโหนดในชั้นซ่อนมากขึ้นโมเดลก็จะมีแนวโน้มที่จะพยากรณ์ได้แม่นยำขึ้น [15] จากเหตุผลข้างต้นทำให้

สามารถกล่าวได้ว่า หากต้องการให้โมเดลมีความแม่นยำในการพยากรณ์มากขึ้นจำเป็นจะต้องมีข้อมูลตัวอย่างที่ใช้ฝึกสอนเริ่มต้นมากขึ้น แต่มีบางสถานการณ์ที่ไม่สามารถหาข้อมูลตัวอย่างที่เหมาะสมและเพียงพอเพื่อมาใช้ฝึกสอนเริ่มต้นให้กับโมเดล OS-ELM ได้ เช่น ในพื้นที่ที่ไม่เคยมีการบันทึกข้อมูลการใช้พลังงานไฟฟ้า, พื้นที่ที่มีการขยายเขตไฟฟ้าเข้าไปใหม่ หรืออาคารสร้างใหม่ ซึ่งเป็นข้อจำกัดในการใช้งานโมเดล OS-ELM ในการพยากรณ์โหลด

เพื่อแก้ปัญหาดังกล่าวนักวิจัยจึงได้นำเสนอวิธีการต่างๆ เช่นการเรียนรู้แบบถ่ายโอน (Transfer Learning) [20] ซึ่งใช้ข้อมูลตัวอย่างจากแหล่งอื่นมาฝึกสอนเริ่มต้นให้โมเดล OS-ELM แต่การเลือกข้อมูลเริ่มต้นที่มีลักษณะเหมือนกับจุดที่จะใช้งานยังเป็นเรื่องที่ยาก นักวิจัยบางท่านได้นำเสนอโมเดล Fully Online Sequential Extreme Learning Machine (FOS-ELM) [24] ซึ่งเป็นการปรับโมเดล OS-ELM โดยทำให้ไม่จำเป็นต้องใช้ข้อมูลตัวอย่างสำหรับเรียนรู้เริ่มต้น แต่โมเดลดังกล่าวยังให้ค่าความผิดพลาดในการพยากรณ์สูงในช่วงเริ่มต้นเนื่องจากการทำงานในช่วงเริ่มต้นเสมือนกับการใช้ข้อมูลตัวอย่างที่มีค่าอินพุตและเป้าหมายเท่ากับศูนย์มาใช้ฝึกสอน

ดังนั้นงานวิจัยนี้จึงได้นำเสนออีกวิธีหนึ่งในการใช้งานโมเดล OS-ELM โดยไม่ต้องจัดหาข้อมูลตัวอย่างในการฝึกสอนเริ่มต้น ซึ่งวิธีที่จะนำเสนอนี้ได้รับแนวคิดมาจากการสร้างข้อมูลตัวอย่างเสริม (Data Augmentation) ในการฝึกสอนโมเดลเรียนรู้เชิงลึก (Deep Learning Model) โดยการรับข้อมูลตัวอย่างจากการทำงานจริงมา 1 ชุดแล้วใช้ค่าสัญญาณรบกวนแบบสุ่ม (Random Noise) บวกเข้าไปกับข้อมูลตัวอย่างดังกล่าวเพื่อเพิ่มจำนวนข้อมูลตัวอย่างให้มีมากพอที่จะใช้ฝึกสอนเริ่มต้นให้โมเดล OS-ELM ขั้นตอนดังกล่าวแสดงด้วยผังงาน (Flowchart) ดังรูปที่ 3.6



รูปที่ 3.6 ผังงานของการสร้างข้อมูลตัวอย่างสำหรับฝึกสอนเริ่มต้นให้โมเดล OS-ELM

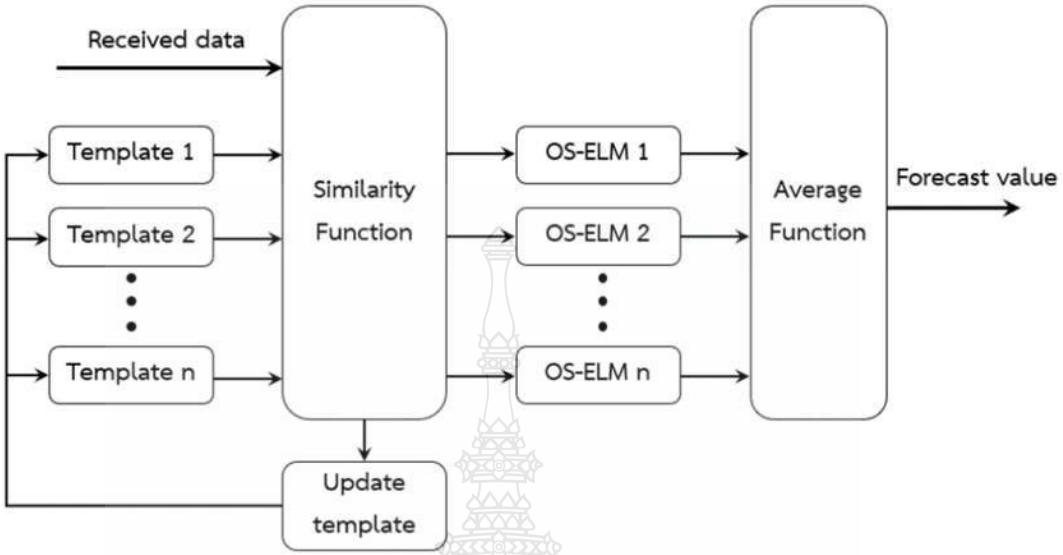
การสร้างข้อมูลตัวอย่างตามผังงานในรูปที่ 3.6 สามารถอธิบายได้ดังนี้

- รับข้อมูลตัวอย่างจากสถานการณ์จริงจำนวน 1 ตัวอย่าง (ทั้งค่าอินพุตและค่าเป้าหมาย), กำหนดมิติของข้อมูลตัวอย่างซึ่งในที่นี้เท่ากับ 25 (อินพุต 24 ค่าและเป้าหมาย 1 ค่า), กำหนดจำนวนข้อมูลตัวอย่างที่ต้องการซึ่งต้องมากกว่าหรือเท่ากับจำนวนโหนดในชั้นซ่อนของโมเดล OS-ELM ซึ่งในที่เลือกเป็น 50 ตัวอย่าง, กำหนดฟังก์ชันแจกแจงความน่าจะเป็น (Probability density function, R(x)) ที่ใช้สำหรับสร้างค่าสัญญาณรบกวน, กำหนดค่าระดับของสัญญาณรบกวนที่จะใช้งาน (มีค่าระหว่าง 0-1)
- วนลูปเพื่อสร้างข้อมูลตัวอย่างตามขนาดมิติที่ต้องการและตามจำนวนที่ต้องการ โดย
- สุ่มค่าสัญญาณรบกวนที่มีขนาดระหว่าง 0-1
- สร้างข้อมูลตัวอย่างแต่ละมิติของแต่ละตัวอย่างโดย นำค่าสัญญาณรบกวนที่สุ่มได้คูณกับขนาดสัญญาณรบกวนที่ต้องการและคูณกับข้อมูลจริงแล้วนำค่าที่ได้บวกเข้ากับข้อมูลจริงอีกครั้ง
- หลังจากสร้างข้อมูลตัวอย่างครบแล้วจึงจัดรูปให้เป็นค่าอินพุตและค่าเป้าหมายเพื่อใช้ฝึกสอนโมเดล OS-ELM ต่อไป

โดยในการทดลองนี้จะทำการศึกษารูปแบบของฟังก์ชันแจกแจงความน่าจะเป็น (Probability density function) และค่าระดับของสัญญาณรบกวนที่เหมาะสมในการสร้างข้อมูลตัวอย่างเพื่อใช้ฝึกสอนโมเดล OS-ELM เพื่อลดค่าความผิดพลาดในการพยากรณ์โหลด

3.3 การปรับปรุงโมเดล OS-ELM ด้วยการสกัดคุณลักษณะ (Features Extraction) ของข้อมูลอินพุต

ในหัวข้อนี้จะประยุกต์ใช้แนวคิดการสกัดคุณลักษณะและแบ่งกลุ่มข้อมูลอินพุต โดยโมเดล OS-ELM แต่ละตัวจะ”รับผิดชอบ”การพยากรณ์ข้อมูลอินพุตแต่ละรูปแบบซึ่งเรียกว่า Template ตามที่ได้แบ่งกลุ่มไว้ ซึ่งโมเดล OS-ELM แต่ละตัวจะมีความแม่นยำในการพยากรณ์ข้อมูลในแต่ละ template ที่ได้เรียนรู้ และเมื่อนำโมเดล OS-ELM แต่ละตัวมาทำงานร่วมกันก็จะสามารถพยากรณ์ข้อมูลได้อย่างแม่นยำในหลากหลายรูปแบบ ขั้นตอนวิธีที่นำเสนอในหัวข้อนี้ชื่อว่า Similarity Based Ensemble OS-ELM โดยหลักการทำงานสามารถอธิบายได้ด้วยไดอะแกรมดังรูปที่ 3.7 และขั้นตอนวิธี (algorithm) ในรูปที่ 3.8



รูปที่ 3.7 ไดอะแกรมแสดงการทำงานของ Similarity Based Ensemble OS-ELM

ดังรูปที่ 3.7 ข้อมูลอินพุตที่รับเข้ามาจะทำถูกเปรียบเทียบกับแต่ละ template โดยใช้ Similarity Function หากข้อมูลอินพุตมีลักษณะคล้ายกับ template ใดโมเดล OS-ELM ที่รับผิดชอบ template นั้นก็จะรับข้อมูลอินพุตเข้ามาเพื่อพยากรณ์โดยจำนวนโมเดลที่ทำการพยากรณ์ในแต่ละครั้งอาจจะมีมากกว่า 1 ตัว ดังนั้นค่าพยากรณ์จากทุกโมเดลจึงถูกส่งไปคำนวณค่าเฉลี่ยเพื่อเป็นค่าพยากรณ์สุดท้ายของโมเดลรวม ซึ่งในทุกรอบที่ทำงานจะมีการปรับค่า template เพื่อให้โมเดลสามารถปรับตัวได้ตามข้อมูลอินพุตที่อาจมีการเปลี่ยนแปลง และโมเดล OS-ELM แต่ละตัวยังเรียนรู้เพิ่มขึ้นจากข้อมูลตัวอย่างที่รับเข้ามาได้อีก

Similarity Based Ensemble OS-ELM Algorithm

```
## Create load profile templates from samples data template ##
## Train each OS-ELM model by each template ##
for i in range number-of-template:
    template[i] = sample [ i: i + number-of-template]
    train FOS-ELM[i] by template[i]
end for
## Compare incoming data with each template ##
## Use model(s) which high similarity to forecast and update template ##
while receiving data:
    k=0, max-similar-value=0, max-similar-template=0
  for j in range number-of-template:
     similarity = similarity-function (data, template[j])
    if similarity > threshold:
       forecast [k] = output-from-FOS-ELM[j]
      if similarity > max-similar-value:
         max-similar-value = similarity
         max-similar-template = j
      end if
       k = k+1
    end if
  end for
    template[max-similar-template] = received data
    return mean(forecast)
end while
```

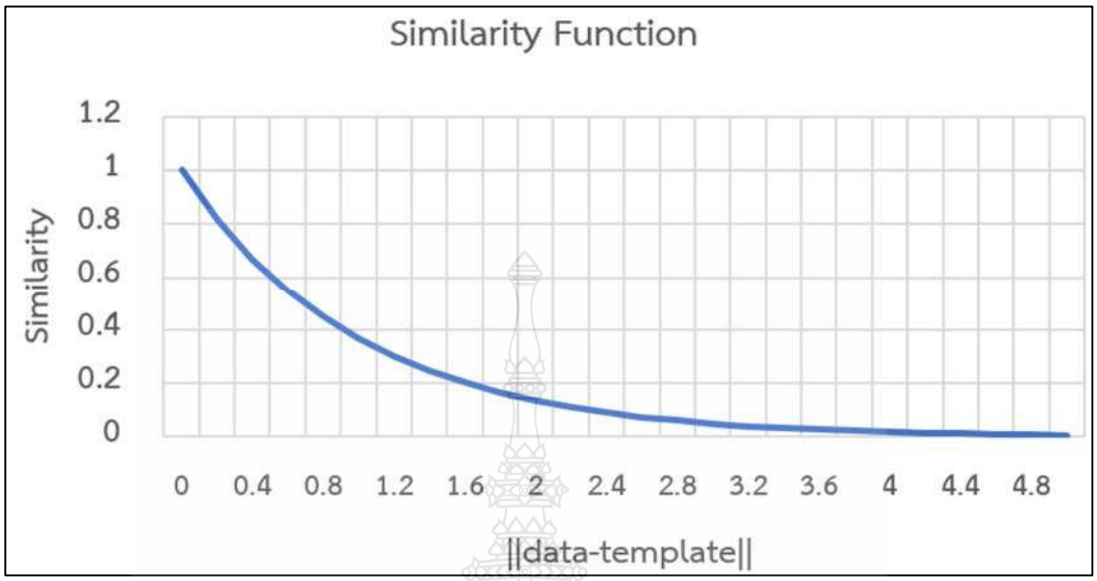
รูปที่ 3.8 ขั้นตอนวิธี (Algorithm) ของ Similarity Based Ensemble OS-ELM

รูปที่ 3.8 เป็นขั้นตอนวิธีของการทำ Similarity Based Ensemble OS-ELM ที่นำเสนอซึ่งสามารถอธิบายได้ดังนี้

- ในบรรทัดที่ 3-6 เป็นการสร้างข้อมูลโหลดที่เป็น Template เป็นจำนวนตามที่ผู้ใช้งานกำหนด โดยข้อมูลแต่ละ Template จะนำไปฝึกสอนโมเดล FOS-ELM แต่ละตัวที่จะทำหน้าที่พยากรณ์เมื่อข้อมูลอินพุตมีลักษณะใกล้เคียงกับ Template ของตน
- บรรทัดที่ 10 เป็นการกำหนดค่าตั้งต้นให้ตัวแปรต่างๆ ที่จะใช้งานในแต่ละข้อมูลอินพุต
- บรรทัดที่ 11-12 เป็นการนำค่าอินพุตมาเปรียบเทียบกับข้อมูลในแต่ละ template โดยใช้ similarity-function ที่ให้ค่าเอาต์พุตเป็น similarity ที่มีค่าระหว่าง 0 ถึง ซึ่ง 1 หมายถึงอินพุตและ template เหมือนกัน 100% โดย similarity-function ที่ใช้ในที่นี้คือฟังก์ชันเรเดียลเบสิสดังสมการที่ 3.3 ซึ่งค่า similarity จะมีความสัมพันธ์กับระยะห่างบนปริภูมิระหว่างข้อมูลอินพุตกับ template ดังรูปที่ 3.9

$$\text{similarity} = e^{\|\text{data-template}\|} \tag{3.3}$$

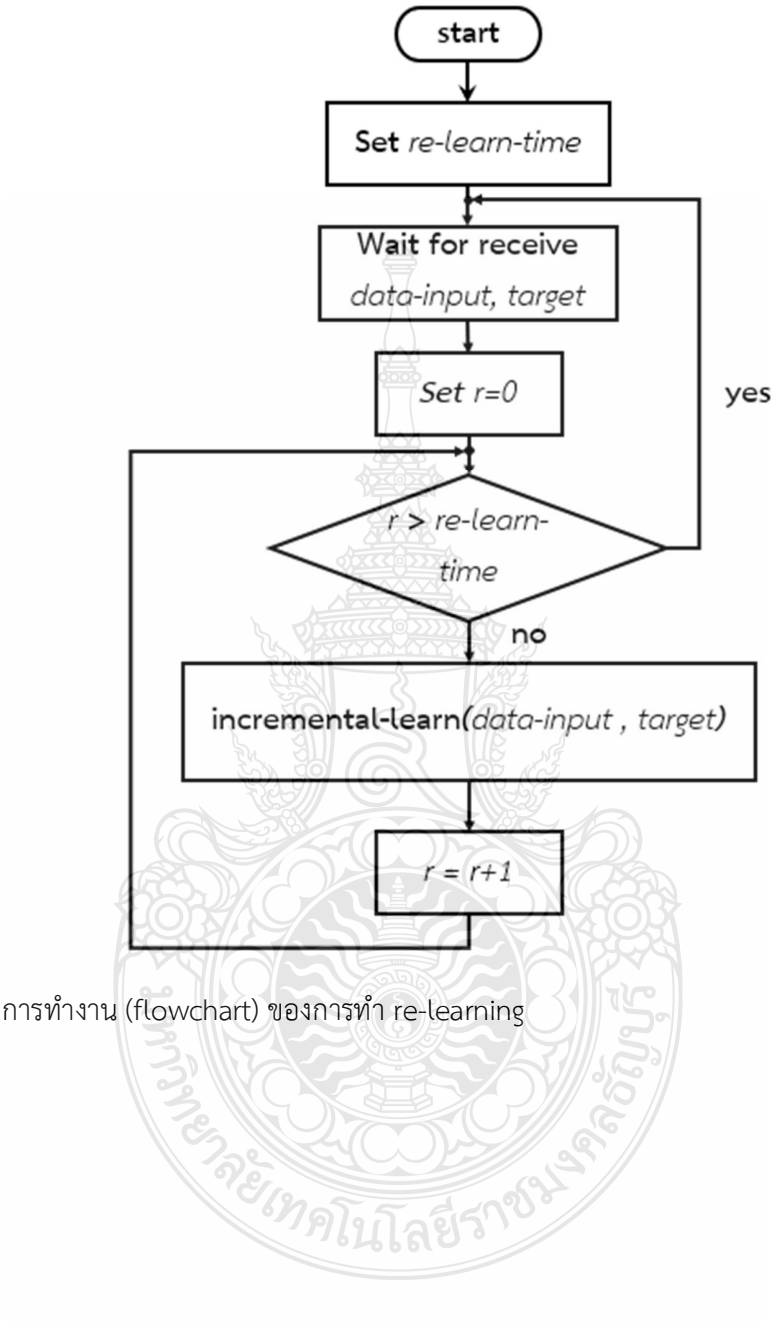
- บรรทัดที่ 13-14 เป็นการเลือกโมเดลที่จะพยากรณ์ตามค่า similarity กล่าวคือโมเดลที่รับผิดชอบ template ที่มีค่า similarity สูงว่าค่า threshold จะถูกนำมาใช้พยากรณ์ โดยค่า threshold นี้ผู้ใช้งานจะเป็นคนกำหนด โดยการทดลองในหัวข้อนี้เป็นการหาค่า threshold ที่เหมาะสมในการใช้งาน
- บรรทัดที่ 15-18 เป็นค้นหา template ที่มีค่า similarity สูงที่สุด
- บรรทัดที่ 22 เป็นการปรับค่า template ที่มีค่า similarity สูงที่สุด โดยปรับให้เท่ากับอินพุตนั้นๆ เพื่อให้โมเดลมีการปรับตัวได้ตามค่าอินพุตใหม่ๆ ที่รับเข้ามา
- บรรทัดที่ 23 เป็นการให้ค่าเอาต์พุตสุดท้ายของการพยากรณ์ซึ่งจะเท่ากับค่าเฉลี่ยของเอาต์พุตของทุกโมเดลที่ถูกใช้พยากรณ์



รูปที่ 3.9 ความสัมพันธ์ระหว่างค่า similarity กับระยะห่างระยะห่างบนปริภูมิระหว่างข้อมูลอินพุตกับ template

3.4 การปรับปรุงโมเดล OS-ELM ด้วยการปรับฟังก์ชันต้นทุน (Cost Function)

โมเดล OS-ELM ใช้ฟังก์ชันต้นทุนเป็นการคำนวณลักษณะเดียวกับการวนซ้ำของวิธีกำลังสองต่ำที่สุด (Recursive Least Square) เพื่อเรียนรู้แบบเพิ่มขึ้นดังที่กล่าวแล้วในบทที่ 2 ซึ่งในหัวข้อนี้จะนำเสนอวิธีการปรับปรุงการใช้ฟังก์ชันต้นทุนดังกล่าวเพื่อให้โมเดลสามารถพยากรณ์ได้แม่นยำมากขึ้น โดยการทำให้โมเดลเรียนรู้เพิ่มขึ้นจากข้อมูลใหม่ที่รับเข้ามามากกว่าหนึ่งครั้ง ซึ่งเรียกวิธีดังกล่าวว่าการเรียนซ้ำ (Re-learning) ด้วยวิธีการดังกล่าวเสมือนว่าโมเดลได้ให้ “น้ำหนัก” กับข้อมูลตัวอย่างที่รับเข้ามาใหม่มากขึ้นจึงสามารถปรับค่าพารามิเตอร์ของโมเดลให้เข้ากับแนวโน้มของข้อมูลใหม่ๆ ได้รวดเร็วยิ่งขึ้นทำให้ค่าความผิดพลาดในการพยากรณ์ลดลงได้ รายละเอียดของวิธีการเรียนซ้ำดังกล่าวสามารถอธิบายได้ดังผังงานในรูปที่ 3.10 โดยเริ่มจากการกำหนดค่า re-learn-time ซึ่งเป็นจำนวนรอบที่โมเดลจะเรียนรู้ซ้ำจากชุดข้อมูลเดิม จากนั้นโมเดลจะรอรับค่าข้อมูลตัวอย่างถัดไป ซึ่งจะประกอบด้วยข้อมูลอินพุตและค่าเป้าหมายที่ต้องการพยากรณ์ และเรียนรู้ซ้ำตามจำนวนรอบที่กำหนดไว้ ซึ่งในหัวข้อนี้จะทดลองเพื่อหาค่าจำนวนรอบการเรียนรู้ซ้ำที่เหมาะสม



รูปที่ 3.10 ผังการทำงาน (flowchart) ของการทำ re-learning

บทที่ 4
วิเคราะห์ผลการทดลอง

ในบทนี้จะกล่าวถึงผลการทดลองและการวิเคราะห์ผลการทดลองทั้ง 4 การทดลอง เพื่อปรับปรุงความแม่นยำในการพยากรณ์โหลดด้วยโมเดล OS-ELM ซึ่งรายละเอียดของการทดลองได้อธิบายไว้ในบทที่ 3 โดยการทดลองทั้ง 4 การทดลองมีดังนี้

1) การทำโมเดลชุด (Ensemble Model)
2) การปรับปรุงการฝึกสอนเริ่มต้น (Initial Training)
3) การพยากรณ์ตามลักษณะของข้อมูลอินพุตด้วยวิธี Similarity Based Ensemble OS-ELM
4) การเรียนรู้ซ้ำ (Re-Learning)

ชุดข้อมูลที่ใช้ในการทดลองคือ Hourly Energy Consumption จาก www.kaggle.com ซึ่งได้กล่าวไว้แล้วในบทที่ 3 เช่นกัน โดยทุกการทดลองจะให้โมเดล OS-ELM ทำการพยากรณ์โหลดของชั่วโมงถัดไปโดยใช้อินพุตเป็นข้อมูลโหลดย้อนหลัง 24 ชั่วโมง ผลการทดลองและการวิเคราะห์ผลการทดลองทั้ง 4 การทดลองมีรายละเอียดดังนี้

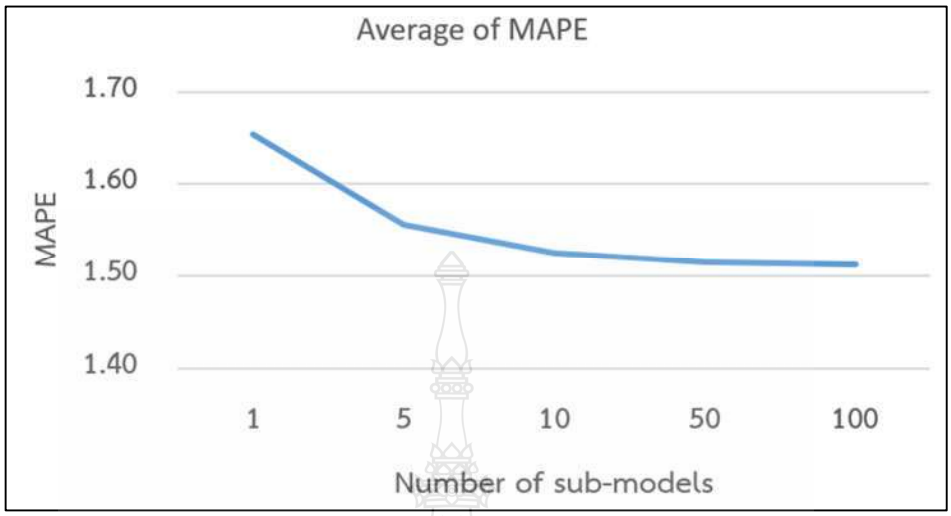
4.1 การทำโมเดลชุด (Ensemble Model)

การทดลองนี้เป็นการเปรียบเทียบการใช้โมเดล OS-ELM แบบชุด (Ensemble OS-ELM) กับโมเดล OS-ELM แบบเดี่ยวในการพยากรณ์โหลด โดยโมเดลแบบชุดได้ใช้โมเดลย่อยจำนวน 5, 10, 50 และ 100 โมเดล เอาต์พุตที่ได้จากแต่ละโมเดลย่อยจะนำมาหาค่าเฉลี่ยเพื่อเป็นค่าพยากรณ์โหลด ข้อมูลที่ใช้ในการทดลองเป็นชุดข้อมูลโหลดทั้ง 9 แห่งใน Hourly Energy Consumption โดยฝึกสอนเริ่มต้นให้โมเดลด้วยข้อมูลตัวอย่าง 50 ชุดแรกของแต่ละชุดข้อมูลและให้โมเดลพยากรณ์โหลดรายชั่วโมงจำนวน 8,760 ชั่วโมง หรือคิดเป็นเวลา 1 ปี พร้อมทั้งเรียนรู้เพิ่มขึ้นทุกครั้งที่พยากรณ์ ดัชนีที่ใช้ในการประเมินค่าความผิดพลาดในการพยากรณ์คือ ค่าร้อยละความคลาดเคลื่อนสัมบูรณ์เฉลี่ย (Mean Absolute Percentage Error, MAPE) ดังสมการที่ 4.1 และผลการทดลองดังตารางที่ 4.1 ดังนี้

$$\text{MAPE}=\frac{\left|\text{actual-forecast}\right|}{\text{actual}}\times 100 \tag{4.1}$$

ตารางที่ 4.1 ค่าความผิดพลาดในการพยารกรณ์โหลดของโมเดลเดี่ยวและโมเดลชุด

Dataset	MAPE of Load forecasting				
	Single model	Number of sub-models in ensemble model			
		5	10	50	100
AEP	1.39	1.27	1.24	1.24	1.23
COMED	1.23	1.10	1.07	1.06	1.06
DAYTON	1.64	1.56	1.52	1.50	1.50
DEOK	1.38	1.34	1.31	1.30	1.30
DOM	1.70	1.60	1.58	1.57	1.56
DUQ	1.88	1.84	1.8	1.80	1.80
EKPC	2.54	2.42	2.4	2.38	2.38
FE	1.51	1.46	1.43	1.42	1.42
NI	1.61	1.42	1.38	1.37	1.36
Average	1.65	1.56	1.53	1.52	1.51



รูปที่ 4.1 ค่า MAPE เฉลี่ยในการพยากรณ์โหลดของโมเดลชุด

จากผลการทดลองข้างต้นพบได้ว่าการใช้ OS-ELM แบบ ensemble สามารถลดความผิดพลาดในการพยากรณ์ลงได้ 5-8% เมื่อเทียบกับการใช้ OS-ELM แบบ single model เนื่องจากเอาต์พุตของโมเดล OS-ELM เป็นผลรวมเชิงเส้นของค่าคุณลักษณะแบบสุ่มที่ไม่เป็นเชิงเส้น (Linear combination of non-linear random features) กล่าวคือ เป็นโมเดลที่ใช้ชั้นซ่อนเดียวและค่าน้ำหนักในชั้นซ่อนยังถูกกำหนดโดยการสุ่มค่า ส่วนชั้นเอาต์พุตใช้วิธีการเดียวกันกับการถดถอยเชิงเส้น (Linear Regression) ซึ่งค่าน้ำหนักที่สุ่มมานี้อาจส่งผลให้ค่าพยากรณ์บางจุดไม่แม่นยำได้ แต่เมื่อใช้หลายๆ โมเดลทำงานร่วมกันจะส่งผลให้ค่าความผิดพลาดของแต่ละโมเดลชดเชยกันจนความแม่นยำในการพยากรณ์สูงขึ้นได้ สามารถอธิบายได้ดังสมการต่อไปนี้

ในกรณีใช้โมเดลเดี่ยวค่าการพยากรณ์โหลดแต่ละค่าจะเป็นดังนี้

$$\text{forecast} = \text{true} + \varepsilon \tag{4.2}$$

โดย

forecast	คือ	ค่าพยากรณ์ (เอาต์พุตจากโมเดล)
true	คือ	ค่าโหลดจริงที่โมเดลต้องการพยากรณ์ให้ถูกต้อง
ε	คือ	ความผิดพลาดที่คงเหลือ (residual)

โดยปกติค่าความผิดพลาดคงเหลือ (Residual) นี้ได้มาจากขั้นตอนการฝึกสอนโมเดลด้วยชุดข้อมูลตัวอย่าง ซึ่งจะไม่เท่ากับศูนย์และจะมีการกระจายตัวแบบปกติ (Normal distribution) ส่วนในกรณีใช้โมเดลชุด (ensemble model) ค่าการพยากรณ์โหลดแต่ละค่าจะเป็นดังนี้

$$\text{forecast} = \sum_{i=1}^{n} \frac{\text{true} + \varepsilon_n}{n}$$

$$\text{forecast} = \text{true} + \sum_{i=1}^{n} \frac{\varepsilon_n}{n} \tag{4.3}$$

โดย

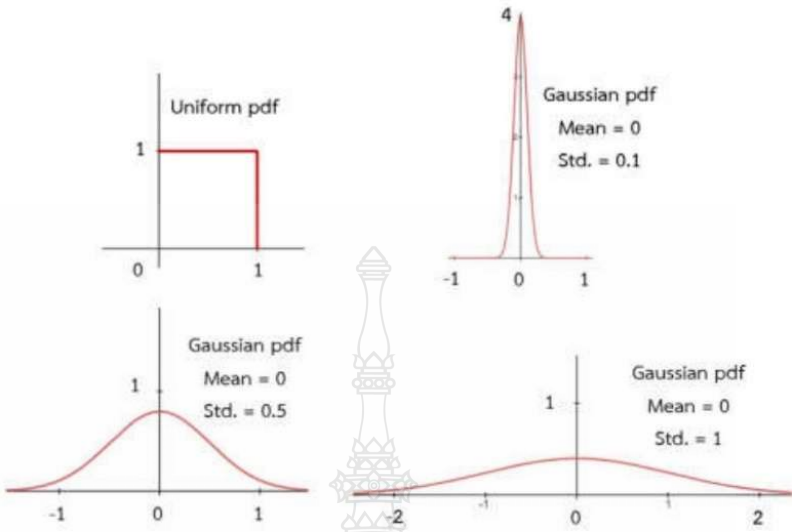
n คือ จำนวนโมเดลในโมเดลชุด

จากสมการที่ 4.3 พจน์ $\sum_{i=1}^{n} \frac{\varepsilon_n}{n}$ หมายถึงค่าเฉลี่ยของความผิดพลาดคงเหลือในการพยากรณ์ ซึ่งค่าดังกล่าวมีการกระจายตัวแบบปกติ (Normal distribution) ดังนั้นค่าเฉลี่ยของความผิดพลาดคงเหลือจะมีค่าใกล้เคียงศูนย์ ทำให้ค่าการพยากรณ์ของโมเดลชุด (ensemble model) ตามสมการที่ 4.3 ใกล้เคียงค่าจริงมากกว่าค่าการพยากรณ์ของโมเดลเดี่ยวตามสมการที่ 4.2

4.2 การปรับปรุงการฝึกสอนเริ่มต้น

การทดลองนี้เป็นการศึกษารูปแบบที่เหมาะสมของสัญญาณรบกวนแบบสุ่ม (Random noise) ที่ใช้เพื่อสร้างข้อมูลตัวอย่างสำหรับฝึกสอนโมเดล OS-ELM ตามที่ได้กล่าวถึงแล้วในหัวข้อ 3.2 โดยรูปแบบของสัญญาณรบกวนที่ทำการศึกษาทดลองมีสองประเด็นได้แก่

1) ฟังก์ชันการกระจายตัวของความน่าจะเป็น (Probability density function) ที่ใช้ในการสร้างสัญญาณรบกวน ซึ่งในการทดลองนี้ใช้การกระจ่ายตัวแบบสม่ำเสมอ (Uniform distribution) และการกระจายตัวแบบเกาส์เซียน (Gaussian distribution) และปรับค่าเบี่ยงเบนมาตรฐานเป็น 1, 0.5 และ 0.1 ดังรูปที่ 4.2 โดยเลือกค่าขนาดสูงสุดของสัญญาณรบกวนที่มีโอกาสเกิดขึ้น (noise level) เป็น 10%
2) ขนาดสูงสุดของสัญญาณรบกวนที่ใช้ โดยเลือกใช้ฟังก์ชันการกระจายตัวแบบสม่ำเสมอ (Uniform distribution) และปรับค่าขนาดสูงสุดของสัญญาณรบกวนที่มีโอกาสเกิดขึ้น (noise level) เป็น 1%, 5%, 10% และ 20% ของข้อมูลจริงที่นำมาสร้างเป็นข้อมูลใหม่



รูปที่ 4.2 ฟังก์ชันการกระจายของความน่าจะเป็น (Probability density function) ที่ใช้ในการทดลอง

การทดลองทั้งสองหัวข้อใช้ชุดข้อมูลที่ใช้ในการทดลองเป็นชุดข้อมูลโหลดทั้ง 9 แห่งใน Hourly Energy Consumption โดยใช้วิธีการดังรูปที่ 3.6 สร้างข้อมูลตัวอย่างจำนวน 50 ชุดมาฝึกสอนเริ่มต้นให้กับโมเดล OS-ELM และให้โมเดลทำการพยากรณ์โหลดและเรียนรู้เพิ่มขึ้นเป็นเวลาเพียง 72 ชั่วโมงเพื่อดูความแม่นยำในการพยากรณ์เพียงแค่ช่วงเริ่มต้นทำงานและเปรียบเทียบกับการใช้โมเดล FOS-ELM ซึ่งเป็นโมเดลที่ไม่ต้องการข้อมูลตัวอย่างในการฝึกสอนเริ่มต้น ดัชนีที่ใช้ในการประเมินค่าความผิดพลาดในการพยากรณ์คือ ค่าร้อยละความคลาดเคลื่อนสัมบูรณ์เฉลี่ย (Mean Absolute Percentage Error, MAPE) ดังสมการที่ 4.1 และผลการทดลองดังตารางที่ 4.2 และ 4.3 ดังนี้

ตารางที่ 4.2 ผลการทดลองใช้ฟังก์ชันการกระจายตัวของความน่าจะเป็น (Probability density function) แบบต่างๆ

Dataset	MAPE เมื่อใช้ Probability density function แบบต่างๆ ในการสร้าง noise			
	Uniform	Gaussian with std.=0.1	Gaussian with std.=0.5	Gaussian with std.=1
AEP	2.03	1.95	2.11	2.44
COMED	1.61	1.47	1.80	2.19
DAYTON	2.73	2.80	2.65	2.89
DEOK	1.99	1.83	2.21	2.52
DOM	1.99	2.07	1.95	2.36
DUQ	2.53	3.10	2.44	2.44
EKPC	3.67	4.37	3.13	2.93
FE	2.31	2.67	2.38	2.57
NI	1.63	1.70	1.61	1.92
Average	2.28	2.44	2.25	2.47

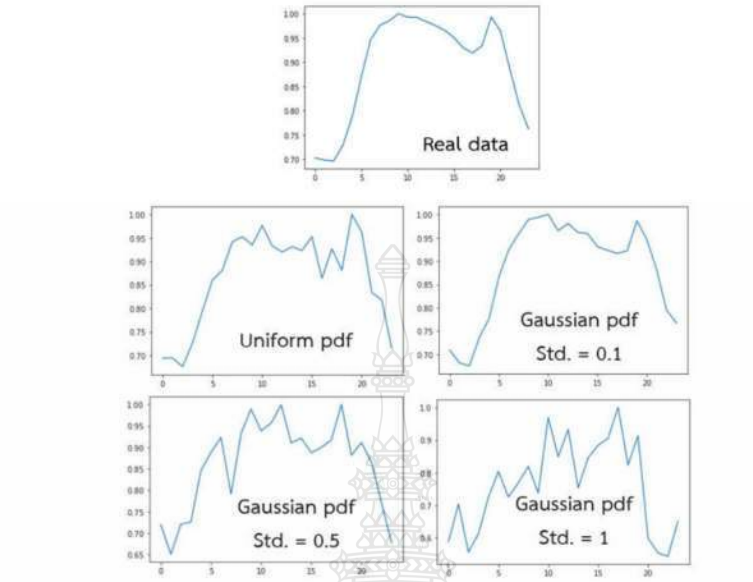
จากผลการทดลองในตารางที่ 4.2 พบว่าการสุ่มสัญญาณรบกวนโดยใช้ฟังก์ชันการกระจายตัวของความน่าจะเป็น (Probability density function, PDF) แบบเกาส์เซียน (Gaussian) มาสร้างข้อมูลตัวอย่างเพื่อฝึกสอนเริ่มต้นให้กับโมเดล OS-ELM จะทำให้โมเดลมีความผิดพลาดในการพยากรณ์ต่ำที่สุด แต่การใช้ PDF แบบ Gaussian นั้นต้องเลือกค่าส่วนเบี่ยงเบนมาตรฐาน (Standard deviation, std.) ที่เหมาะสม ซึ่งในการทดลองนี้พบว่าหากเลือกใช้ค่า std. เท่ากับ 0.5 จะให้ค่าความผิดพลาดในการพยากรณ์ต่ำที่สุด และหากเลือกใช้ค่า std. สูงเกินไปหรือต่ำเกินไปซึ่งในการทดลองนี้มีค่าเท่ากับ 1 และ 0.1 ตามลำดับจะพบว่าค่าความผิดพลาดในการพยากรณ์จะสูงขึ้น เนื่องจากการใช้ค่า std. สูงเกินไปจะทำให้ข้อมูลตัวอย่างที่ใช้ฝึกสอนเริ่มต้นมีค่าผิดไปจากข้อมูลเดิมอย่างมากจนโมเดลไม่สามารถเรียนรู้ได้อย่างถูกต้องซึ่งเป็นปัญหา under-fit ส่วนการเลือกค่า std. ต่ำเกินไปจะทำให้ข้อมูลตัวอย่างที่ใช้ฝึกสอนเริ่มต้นมีลักษณะคล้ายกันทั้งหมดจนส่งผลให้โมเดลเกิดปัญหา over-fit ในการเรียนรู้ ส่วนการใช้ PDF แบบ uniform นั้นให้ค่าความผิดพลาดสูงกว่าการใช้ Gaussian ที่มีค่า std. 0.5 เพียงเล็กน้อยและจุดเด่นของการใช้ PDF แบบ uniform คือไม่ต้องกังวลในการเลือกค่า std. ที่เหมาะสม

ตารางที่ 4.3 ผลการทดลองปรับขนาดสูงสุดของสัญญาณรบกวน (Noise level) ที่ใช้

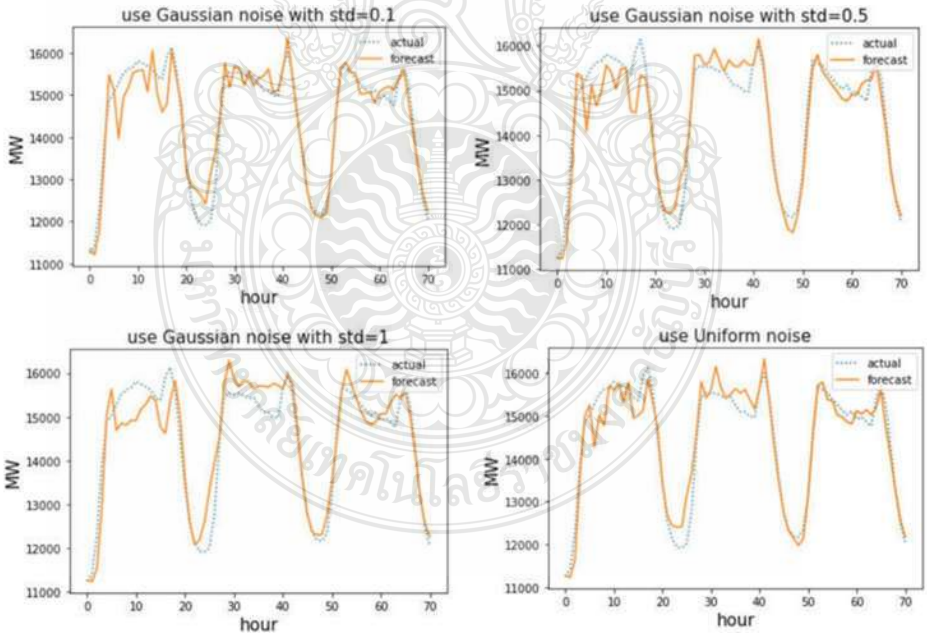
Dataset	Method	% Noise	MAPE	Dataset	Method	% Noise	MAPE
AEP	FOS-ELM	N/A	2.18	DUQ	FOS-ELM	N/A	3.46
	OS-ELM (Proposed)	1	2.04		OS-ELM (Proposed)	1	3.43
		5	1.95			5	2.75
		10	2.03			10	2.53
		20	2.13			20	2.45
COMED	FOS-ELM	N/A	1.86	EKPC	FOS-ELM	N/A	4.83
	OS-ELM (Proposed)	1	1.49		OS-ELM (Proposed)	1	3.85
		5	1.48			5	3.68
		10	1.61			10	3.67
		20	1.87			20	3.58
DAYTON	FOS-ELM	N/A	2.96	FE	FOS-ELM	N/A	2.77
	OS-ELM (Proposed)	1	2.85		OS-ELM (Proposed)	1	2.78
		5	2.76			5	2.54
		10	2.73			10	2.31
		20	2.7			20	2.43
DEOK	FOS-ELM	N/A	2.08	NI	FOS-ELM	N/A	1.60
	OS-ELM (Proposed)	1	1.84		OS-ELM (Proposed)	1	1.58
		5	1.77			5	1.56
		10	1.99			10	1.63
		20	2.18			20	1.69
DOM	FOS-ELM	N/A	2.43	Average	FOS-ELM	N/A	2.69
	OS-ELM (Proposed)	1	2.08		OS-ELM (Proposed)	1	2.44
		5	2.01			5	2.28
		10	1.99			10	2.28
		20	2.06			20	2.34

จากตารางที่ 4.3 การสุ่มสัญญาณรบกวนโดยใช้ฟังก์ชันการกระจายตัวของความน่าจะเป็น (Probability density function, PDF) แบบสม่ำเสมอ (Uniform) มาสร้างข้อมูลตัวอย่างเพื่อฝึกสอนเริ่มต้นให้กับโมเดล OS-ELM โดยเลือกขนาดสูงสุดของสัญญาณรบกวน (Noise level) 1% ถึง 20% ค่าความผิดพลาดในการพยากรณ์ก็ยังคงต่ำกว่าการใช้โมเดล FOS-ELM ซึ่งเป็นโมเดลที่ทำงานได้โดยไม่ต้องใช้ข้อมูลตัวอย่างสำหรับฝึกสอนเริ่มต้น จากการทดลองพบว่าค่า Noise level ที่ทำให้ค่าความผิดพลาดในการพยากรณ์ต่ำที่สุดคือ 5% และ 10% ซึ่งการเลือกค่า Noise level ที่ต่ำเกินไปจะทำให้ข้อมูลตัวอย่างที่ได้มีลักษณะคล้ายกันทำให้โมเดลเกิดปัญหา over-fit และหากเลือกค่า Nosie level ที่สูงเกินไปจะทำให้ข้อมูลตัวอย่างที่ได้ผิดไปจากข้อมูลเดิมอย่างมากจนโมเดลไม่สามารถเรียนรู้ได้อย่างถูกต้อง

จากผลการทดลองปรับรูปแบบของฟังก์ชันการกระจายตัวของความน่าจะเป็น (Probability density function, PDF) ที่ใช้สร้างข้อมูลตัวอย่างสำหรับฝึกสอนเริ่มต้น ทั้งในตารางที่ 4.2 และ 4.3 พบว่าการใช้ข้อมูลตัวอย่างสังเคราะห์ช่วยลดความผิดพลาดในการพยากรณ์ได้มากกว่า 10% เมื่อเทียบกับการใช้โมเดล FOS-ELM และพบว่าการใช้ PDF แบบเกาส์เซียน (Gaussian) ที่มีค่าส่วนเบี่ยงเบนมาตรฐานเท่ากับ 0.5 ให้ค่าความผิดพลาดในการพยากรณ์ต่ำที่สุด รองลงมาคือการเลือกใช้ PDF แบบสม่ำเสมอ (uniform) ที่มีขนาดสูงสุดของสัญญาณรบกวน (Noise level) 5% -10% ซึ่งมีค่าความผิดพลาดในการพยากรณ์ไม่แตกต่างกันมากนัก หากสังเกตข้อมูลในผลการทดลองจะพบการเลือกใช้ PDF แบบสม่ำเสมอ (uniform) จะสามารถปรับค่า noise level ได้กว้างโดยส่งผลกระทบต่อค่าความแม่นยำในการพยารกณ์ไม่มาก หรือกล่าวได้ว่าการใช้ PDF แบบสม่ำเสมอ (uniform) จะมีความทนทาน (Robust) ต่อการปรับค่าพารามิเตอร์มากกว่าการใช้ PDF แบบเกาส์เซียน (Gaussian)



รูปที่ 4.3 Load profile ที่สร้างขึ้นเพื่อใช้เป็นข้อมูลตัวอย่างสำหรับฝึกสอนเริ่มต้น



รูปที่ 4.4 ค่าจริงและค่าพยากรณ์โหลดในกรณีที่ใช้ฟังก์ชันการกระจายตัวของความน่าจะเป็นแบบต่างๆ สร้างข้อมูลตัวอย่าง

4.3 การพยากรณ์ตามลักษณะของข้อมูลอินพุตด้วยวิธี Similarity Based Ensemble OS-ELM

การทดลองนี้เป็นการศึกษาการทำงานของอัลกอริทึม Similarity Based Ensemble OS-ELM ที่ได้นำเสนอไว้ในหัวข้อที่ 3.3 ซึ่งเป็นการใช้โมเดล OS-ELM หลายโมเดลแบ่งหน้าที่กันรับผิดชอบ โดยแต่ละโมเดลจะรับผิดชอบเฉพาะรูปแบบอินพุตหนึ่งๆที่เรียกว่า Template เท่านั้น เมื่อได้รับอินพุตเข้ามาอัลกอริทึมจะทำการเปรียบเทียบความคล้าย (Similarity) กับ Template ทุกตัวและหาก Template ใดมีความคล้ายมากกว่า Threshold ที่กำหนดไว้ อัลกอริทึมก็จะส่งอินพุตให้โมเดลที่รับผิดชอบ Template นั้นทำการพยารกรณ์ค่าและเรียนรู้เพิ่มขึ้น

ชุดข้อมูลที่ใช้ในการทดลองนี้คือชุดข้อมูลทั้ง 9 ชุดใน Hourly Energy Consumption ส่งให้อัลกอริทึมทำการพยากรณ์โหลดรายชั่วโมงและเรียนรู้เพิ่มขึ้นกับข้อมูลทั้งสิ้น 8,760 ข้อมูลหรือคิดเป็นระยะเวลา 1 ปี โดยอัลกอริทึมจะใช้โมเดล OS-LEM จำนวน 50 ตัวรับผิดชอบ 50 Template ที่มาจากข้อมูลตัวอย่าง 50 รายการแรกในแต่ละชุดข้อมูลและทดลองเปรียบเทียบค่าความผิดพลาดในการพยากรณ์โดยใช้ Threshold 0, 0.2, 0.4 และ 0.8 เพื่อศึกษาแนวโน้มของการเลือกใช้ค่า Threshold ที่เหมาะสมในการพยากรณ์โหลดด้วยอัลกอริทึมที่นำเสนอ ผลการทดลองดังตารางที่ 4.4

โมเดล OS-ELM ทั้ง 50 ตัวที่ใช้ในการทดลองนี้ได้ผ่านการฝึกสอนเริ่มต้น (Initial training) ด้วยข้อมูลตัวอย่างที่สังเคราะห์ขึ้นตามวิธีการที่นำเสนอในหัวข้อ 3.2 โดยใช้สัญญาณรบกวนที่สุ่มจากฟังก์ชันการกระจายตัวของความน่าจะเป็น (Probability Density Function: PDF) แบบสม่ำเสมอ (Uniform) และกำหนดขนาดสูงสุดของสัญญาณรบกวน(Noise Level) ไว้ที่ 10%

ตารางที่ 4.4 ผลการทดลองการใช้ Similarity Based Ensemble OS-ELM เพื่อพยากรณ์โหลด

Dataset	MAPE for each Threshold value				
	0	0.2	0.4	0.6	0.8
AEP	1.2	1.24	1.23	1.27	1.36
COMED	1.09	1.07	1.09	1.10	1.20
DAYTON	1.49	1.5	1.47	1.52	1.56
DEOK	1.34	1.35	1.34	1.35	1.36
DOM	1.56	1.57	1.58	1.58	1.62
DUQ	1.90	1.89	1.88	1.89	1.87
EKPC	2.43	2.42	2.42	2.45	2.25
FE	1.44	1.44	1.43	1.45	1.49
NI	1.39	1.40	1.39	1.42	1.52
Average	1.54	1.54	1.54	1.56	1.61

จากผลการทดลองในตารางที่ 4.4 จะพบว่าการเลือกใช้ค่า Threshold เท่ากับ 0, 0.2, 0.4 และ 0.6 จะให้ค่าความผิดพลาดในการพยากรณ์ไม่แตกต่างกัน ส่วนการเลือกใช้ค่า Threshold เท่ากับ 0.8 จะให้ค่าความผิดพลาดในการพยากรณ์สูงกว่าการใช้ค่าอื่น และการเพิ่มค่า Threshold มีแนวโน้มที่ค่าความผิดพลาดในการพยากรณ์สูงมากขึ้น สาเหตุเนื่องจากที่ค่า Threshold สูงๆ นั้นจะมีจำนวนโมเดล OS-ELM ที่ถูกเลือกมาใช้ในการพยากรณ์เป็นจำนวนน้อยซึ่งในบางครั้งอาจใช้แค่โมเดลเดียวในการพยากรณ์โหลดทำให้เกิดความผิดพลาดในการพยากรณ์สูงดังที่ได้อธิบายไว้ในหัวข้อ 4.1

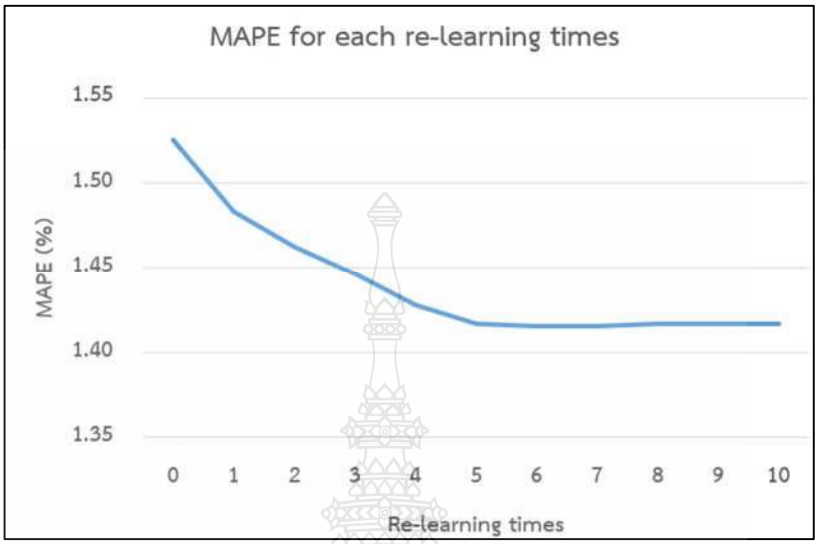
การเลือกใช้ค่า Threshold เท่ากับ 0 ในการทดลองนี้มีความหมายเท่ากับการใช้โมเดลกลุ่ม (ensemble model) ที่มีจำนวนโมเดลย่อย 50 ตัวกับทุกข้อมูลอินพุต ซึ่งจากผลการทดลองจะพบว่าการใช้อัลกอริทึ่ม Similarity Based Ensemble OS-ELM ให้ค่าความแม่นยำในการพยากรณ์ไม่แตกต่างจากการใช้วิธีโมเดลกลุ่ม แต่การใช้โมเดลกลุ่มไม่ใช้ต้องคำนวณค่า Similarity ดังนั้นการใช้งานโมเดลกลุ่มจึงเหมาะสมมากกว่า

4.4 การเรียนรู้ซ้ำ (Re-learning)

ในหัวข้อนี้เป็นการทดสอบวิธีการเรียนรู้ซ้ำที่ได้อธิบายไว้ในหัวข้อที่ 3.4 ซึ่งเป็นการปรับวิธีเรียนรู้ของโมเดล OS-ELM โดยให้โมเดลทำการเรียนรู้เพิ่มขึ้น (Incremental learning) ซ้ำที่ข้อมูลชุดเดิมมากกว่า 1 ครั้ง โดยการทดลองนี้ได้ใช้ชุดข้อมูลทั้ง 9 ชุดใน Hourly Energy Consumption ให้อัลกอริที่มทำการพยากรณ์โหลดรายชั่วโมงและเรียนรู้เพิ่มขึ้นกับข้อมูลทั้งสิ้น 8,760 ข้อมูลหรือคิดเป็นระยะเวลา 1 ปี โดยเปรียบเทียบความผิดพลาดในการพยากรณ์โหลดเมื่อใช้จำนวนครั้งของการเรียนรู้ซ้ำ (Re-learning) ตั้งแต่ 0-10 ครั้ง โดยการเรียนรู้ซ้ำ 0 ครั้งหมายถึงการไม่ต้องเรียนรู้ซ้ำจากข้อมูลชุดเดิม (เรียนรู้แค่ครั้งเดียวไม่ต้องเรียนรู้ซ้ำ) ซึ่งเป็นการทำงานโดยปกติของโมเดล OS-ELM โมเดล OS-ELM ที่ใช้ในการทดลองนี้เป็นการใช้โมเดลชุด (Ensemble model) ที่ประกอบด้วยโมเดลย่อยจำนวน 10 โมเดลและฝึกสอนเริ่มต้น (Initial training) ด้วยข้อมูลตัวอย่างที่สังเคราะห์ขึ้นตามวิธีการที่นำเสนอในหัวข้อ 3.2 โดยใช้สัญญาณรบกวนที่สุ่มจากฟังก์ชันการกระจายตัวของความน่าจะเป็น (Probability Density Function: PDF) แบบสม่ำเสมอ (Uniform) และกำหนดขนาดสูงสุดของสัญญาณรบกวน(Noise Level) ไว้ที่ 10% ผลการทดลองเป็นดังตารางที่ 4.5

ตารางที่ 4.5 ผลการทดลองการใช้วิธีเรียนรู้ซ้ำ (Re-learning) กับโมเดล OS-ELM เพื่อพยากรณ์โหลด

Dataset	MAPE for each re-learning times										
	0	1	2	3	4	5	6	7	8	9	10
AEP	1.24	1.19	1.17	1.15	1.14	1.14	1.14	1.14	1.14	1.14	1.14
COMED	1.07	1.03	1.01	1.01	1	0.9	0.9	0.9	0.9	0.9	0.9
DAYTON	1.52	1.47	1.46	1.44	1.41	1.41	1.4	1.4	1.41	1.41	1.41
DEOK	1.31	1.27	1.25	1.23	1.21	1.21	1.2	1.2	1.2	1.2	1.2
DOM	1.58	1.55	1.51	1.5	1.48	1.48	1.48	1.48	1.48	1.48	1.48
DUQ	1.8	1.75	1.73	1.72	1.7	1.7	1.6	1.6	1.6	1.6	1.6
EKPC	2.4	2.35	2.34	2.32	2.3	2.3	2.2	2.2	2.2	2.2	2.2
FE	1.43	1.39	1.37	1.35	1.33	1.33	1.33	1.33	1.33	1.33	1.33
NI	1.38	1.35	1.32	1.3	1.28	1.28	1.28	1.28	1.28	1.28	1.28
Average	1.53	1.48	1.46	1.45	1.43	1.42	1.39	1.39	1.39	1.39	1.39



รูปที่ 4.5 ค่า MAPE เฉลี่ยของการพยากรณ์โหลดเมื่อใช้จำนวนครั้งในการเรียนรู้ซ้ำต่างๆ

จากผลการทดลองในตารางที่ 4.5 และรูปที่ 4.5 จะพบว่าการใช้วิธีเรียนรู้ซ้ำ (Re-learning) สามารถลดค่าความผิดพลาดในการพยากรณ์โหลดของโมเดล OS-ELM ลงได้ 3-9% เมื่อเทียบกับโมเดล OS-ELM ที่ไม่มีการเรียนรู้ซ้ำ (โดยการเรียนรู้ซ้ำ 0 ครั้งในการทดลองหมายถึงไม่มีการเรียนรู้ซ้ำ) ซึ่งคือการใช้งานโมเดล OS-ELM แบบปกติ และเมื่อเพิ่มจำนวนครั้งในการเรียนรู้ซ้ำจะทำให้ค่าความผิดพลาดในการพยากรณ์ลดลงด้วย ซึ่งการเพิ่มจำนวนครั้งในการเรียนรู้ซ้ำจะเป็นเหมือนการเพิ่มจำนวนข้อมูลชุดล่าสุดให้โมเดลได้เรียนรู้ หรือเป็นการเพิ่มน้ำหนักในการเรียนรู้ให้กับข้อมูลชุดล่าสุด ซึ่งโมเดลจะเรียนรู้เพิ่มขึ้นโดยให้ความสำคัญกับข้อมูลชุดล่าสุดมากกว่าข้อมูลเดิมส่งผลให้โมเดลสามารถปรับตัวตามการเปลี่ยนแปลงของข้อมูลหรือสถานการณ์ได้ดีขึ้น แต่หากเพิ่มจำนวนครั้งของการเรียนรู้ซ้ำมากเกินไปเช่นในการทดลองนี้พบว่าหากเรียนรู้ซ้ำมากกว่า 4 ครั้งค่าความผิดพลาดในการพยากรณ์จะไม่ลดลงแล้ว เนื่องจากโมเดลเกิดปัญหา over-fitting ที่ข้อมูลล่าสุดมากเกินไป

บทที่ 5
สรุปผลและข้อเสนอแนะ

5.1 สรุปผล

จากแนวทางการปรับปรุงโมเดล OS-ELM ทั้ง 4 แนวทางที่นำเสนอในบทที่ 3 และผลการทดลองของทั้ง 4 แนวทางที่นำเสนอในบทที่ 4 สามารถสรุปได้ดังนี้

- การใช้โมเดลชุด (Ensemble Model) ของ OS-ELM ช่วยลดความผิดพลาดในการพยากรณ์โหลดได้ 5-8% เมื่อเทียบกับการใช้โมเดล OS-ELM แบบเดี่ยว เนื่องมาจากคามผิดพลาดในการพยากรณ์โหลดของโมเดลย่อยแต่ละตัว หรือที่เรียกว่าความผิดพลาดที่คงเหลือ (residual error) จะทำการหักล้างกันเองจนทำให้ค่าพยากรณ์เข้าใกล้ค่าจริงมากยิ่งขึ้น
- การปรับปรุงการฝึกสอนเริ่มต้น (Initial training) โดยการใช้ข้อมูลจริงเพียงชุดเดียวมาสร้างชุดข้อมูลตัวอย่างที่จำนวนเพียงพอที่จะฝึกสอนเริ่มต้น การสร้างข้อมูลตัวอย่างนี้จะนำข้อมูลจริงมาบวกเพิ่มด้วยค่าสัญญาณรบกวนแบบสุ่ม พบว่าการใช้ข้อมูลสังเคราะห์ในการฝึกสอนเริ่มต้นจะช่วยลดค่าความผิดพลาดในการพยากรณ์ในช่วงเริ่มต้นลงได้มากกว่า 10% เมื่อเปรียบเทียบกับการใช้โมเดล OS-ELM โดยการทดลองนี้ยังได้ศึกษารูปแบบของสัญญาณรบกวนที่เหมาะสมในการใช้งาน ซึ่งจากการทดลองพบว่า การใช้สัญญาณรบกวนที่สุ่มจากฟังก์ชันแจกแจงความน่าจะเป็นแบบเกาส์เซียน(Gaussian) มาสร้างชุดข้อมูลตัวอย่างเพื่อฝึกสอนเริ่มต้นให้กับโมเดลจะทำให้โมเดลสามารถพยากรณ์โหลดโดยมีค่าความผิดพลาดต่ำที่สุด แต่ต้องทำการเลือกค่าส่วนเบี่ยงเบนมาตรฐานของฟังก์ชันเกาส์เซียน(Gaussian) ให้เหมาะสมซึ่งในการทดลองพบว่าค่า 0.5 เป็นค่าที่เหมาะสมที่สุด ส่วนการใช้ฟังก์ชันแจกแจงความน่าจะเป็นแบบสม่ำเสมอ (Uniform) สร้างชุดข้อมูลตัวอย่างเพื่อฝึกสอนเริ่มต้นให้กับโมเดลจะทำให้โมเดลสามารถพยากรณ์โหลดโดยมีค่าความผิดพลาดสูงกว่าการใช้ฟังก์ชันเกาส์เซียน (Gaussian) เพียงเล็กน้อย แต่การปรับค่าพารามิเตอร์ของฟังก์ชันแจกแจงความน่าจะเป็นแบบสม่ำเสมอซึ่งในที่นี้คือขนาดของสัญญาณรบกวน (Noise level) ส่งผลต่อความแม่นยำในการพยากรณ์น้อยมาก ซึ่งอาจกล่าวได้ว่าการใช้ฟังก์ชันแจกแจงความน่าจะเป็นแบบสม่ำเสมอ (Uniform) สร้างชุดข้อมูลตัวอย่าง จะทำให้โมเดลมีความทนทาน (Robust) มากกว่าการ

ใช้ฟังก์ชันเกาส์เซียน และมีความแม่นยำแตกต่างกันไม่มากเนื่องจากช่วงของค่าสัญญาณรบกวนกว้างกว่าทำให้ข้อมูลตัวอย่างที่ได้กระจายตัวมากกว่าส่งผลให้โมเดลเกิดปัญหา Over-fitting น้อยกว่า

- การใช้อัลกอริทึ่ม Similarity Based Ensemble OS-ELM เป็นการเลือกใช้โมเดลย่อยในโมเดลชุดแค่บางตัวมาพยากรณ์โหลด โดยโมเดลแต่ละตัวจะได้ได้รับการฝึกสอนให้พยากรณ์ข้อมูลที่มีลักษณะหนึ่งๆ ซึ่งเรียกว่า Template โมดลตัวที่ถูกเลือกมาพยากรณ์จะเป็นโมเดลที่มีค่า Template ใกล้เคียงกับข้อมูลอินพุตที่รับเข้ามา โดยในการทดลองได้ปรับค่า Threshold ซึ่งเป็นค่าที่วัดความใกล้เคียงกันระหว่างข้อมูลอินพุตและ Template เพื่อเลือกค่า Threshold ที่เหมาะสม ซึ่งจาการทดลองพบว่าการใช้ค่า Threshold ต่ำจะให้ค่าความผิดพลาดในการพยากรณ์ต่ำกว่าการใช้ค่า Threshold สูง ซึ่งการใช้ค่า Threshold ต่ำคือการใช้โมเดลหลายตัวมาพยากรณ์โดยให้ไม่ให้ความสำคัญกับความคล้ายกันระหว่างอินพุตและ template ซึ่งคล้ายกับการใช้โมเดลชุด ส่วนการใช้ค่า Threshold สูงคือการเลือกใช้เฉพาะโมเดลจำนวนน้อยที่ได้รับการฝึกสอนมาสำหรับรูปแบบอินพุตนั้นๆ ดังนั้นจึงสามารถตีความได้ว่า การใช้ OS-ELM แบบโมเดลชุดโดยให้โมเดลทุกตัวช่วยกันพยากรณ์โหลดจะให้ค่าความผิดพลาดน้อยกว่าการเลือกใช้โมเดลที่ฝึกสอนมาโดยเฉพาะเพียงไม่กี่ตัว ดังนั้นอัลกอริทึ่ม Similarity Based Ensemble OS-ELM นี้ให้ผลลัพธ์ไม่แตกต่างจากการใช้โมเดลชุด (Ensemble model)
- การเรียนรู้ซ้ำ (Re-learning) เป็นการให้โมเดลชุด OS-ELM เรียนรู้เพิ่มขึ้น (Increment learning) จากข้อมูลล่าสุดมากกว่า 1 ครั้งซึ่งเป็นเหมือนการเพิ่มน้ำหนักในการเรียนรู้ให้กับข้อมูลล่าสุดทำให้โมเดลสามารถปรับตัวเข้ากับแนวโน้มของข้อมูลใหม่ได้เร็วขึ้น จากการทดลองพบว่าเมื่อให้โมเดลเรียนรู้ซ้ำจากข้อมูลชุดล่าสุดจะทำให้ความผิดพลาดในการพยากรณ์โหลดลดลงได้ 3-9% แต่เมื่อเพิ่มจำนวนครั้งในการเรียนรู้ซ้ำมากกว่า 4 ครั้งพบว่าค่าความผิดพลาดในการพยากรณ์จะไม่สามารถลดลงได้อีก เนื่องจากโมเดลเกิดปัญหา over-fitting ที่ข้อมูลล่าสุดมากเกินไป

5.2 ข้อเสนอแนะ

ในการวิจัยการปรับปรุงโมเดล OS-ELM นี้ ผู้วิจัยมีข้อเสนอแนะอยู่ 2 ประเด็นเกี่ยวกับแนวทางการวิจัยเพิ่มเติมและการนำไปประยุกต์ใช้งานดังนี้

- การวิจัยครั้งนี้ไม่ได้นำตัวแปรที่เป็นปัจจัยภายนอกเช่น อุณหภูมิ มาเป็นข้อมูลอินพุตในการพยากรณ์โหลด ดังนั้นหากสามารถหาชุดข้อมูลที่เหมาะสมก็สามารถทำการวิจัยในหัวข้อนี้เพิ่มเติมได้
- ควรมีการศึกษาทดลอง การประยุกต์ใช้งานโมเดล OS-ELM บนไมโครคอนโทรลเลอร์หรือคอมพิวเตอร์บอร์ดเดี่ยวร่สมกับสมาร์ทมิเตอร์เพื่อทำการพยากรณ์โหลดในระบบไฟฟ้าจริง ตามแนวทางการประมวลผลแบบเอดจ์คอมพิวติง (Edge-computering)

บรรณานุกรม

[1] I. Lampropoulos, W. L. Kling, P. F. Ribeiro and J. van den Berg, "History of demand side management and classification of demand response control schemes," *2013 IEEE Power & Energy Society General Meeting*, Vancouver, BC, 2013, pp. 1-5, doi: 10.1109/PESMG.2013.6672715.

[2] L. Hernandez et al., "A Survey on Electric Power Demand Forecasting: Future Trends in Smart Grids, Microgrids and Smart Buildings," in IEEE Communications Surveys & Tutorials, vol. 16, no. 3, pp. 1460-1495, Third Quarter 2014, doi: 10.1109/SURV.2014.032014.00094.

[3] Corentin Kuster, Yacine Rezgui, Monjur Mourshed, “Electrical load forecasting models: A critical systematic review,” Sustainable Cities and Society, vol.35, pp. 257-270, 2017, doi: 10.1016/j.scs.2017.08.009

[4] Y. Wang, N. Zhang, Q. Chen, D. S. Kirschen, P. Li and Q. Xia, "Data-Driven Probabilistic Net Load Forecasting with High Penetration of Behind-the-Meter PV," in IEEE Transactions on Power Systems, vol. 33, no. 3, pp. 3255-3264, May 2018, doi: 10.1109/TPWRS.2017.2762599.

[5] Y. Kongjeen and K. Bhumkittipich, "Modeling of electric vehicle loads for power flow analysis based on PSAT," 2016 13th International Conference on Electrical Engineering/Electronics, Computer, Telecommunications and Information Technology (ECTI-CON), Chiang Mai, 2016, pp. 1-6, doi: 10.1109/ECTICon.2016.7561430.

[6] W. Lee, J. Jung and M. Lee, "Development of 24-hour optimal scheduling algorithm for energy storage system using load forecasting and renewable energy forecasting," 2017 IEEE Power & Energy Society General Meeting, Chicago, IL, 2017, pp. 1-5, doi: 10.1109/PESGM.2017.8273907.

[7] A. Garulli, S. Paoletti and A. Vicino, "Models and Techniques for Electric Load Forecasting in the Presence of Demand Response," in IEEE Transactions on Control

บรรณานุกรม (ต่อ)

Systems Technology, vol.23, no. 3, pp. 1087-1097, May 2015, doi: 10.1109 TCST. 2014. 2361807.

[8] Losing, V., Hammer, B. and Wersing, H., 2018. Incremental on-line learning: A review and comparison of state of the art algorithms. *Neurocomputing*, 275, pp.1261-1274.

[9] G.Cauwenberghs, T.Poggio, "Incremental and decremental support vector machine learning," Proceedings of the 2001 Neural Information Processing Systems (NIPS), 2001

[10] A.Saffari, C.Leistner, J.Santner, M.Godec, H.Bischof, "On-line random forests," Proceedings of the 2009 IEEE Twelfth International Conference on Computer Vision Workshops, 2009

[11] R.Polikar, L.Upda, S.Upda, V.Honavar, "Learn++: an incremental learning algorithm for supervised neural networks," IEEE Trans. Syst. Man Cybern. 31 (4), pp.497-508, 2001

[12] Nan-ying Liang, Guang-Bin Huang, P.Saratchandran and N.Sundarrajan, "A Fast and Accurate Online Sequential Learning Algorithm for Feedforward Networks." IEEE Tran. On Neural Networks, Vol.17, No.6, pp.1411-1423, November 2006

[13] A. L. Samuel, "Some Studies in Machine Learning Using the Game of Checkers," in IBM Journal of Research and Development, vol. 3, no. 3, pp. 210-229, July 1959, doi: 10.1147/rd.33.0210.

[14] Ng, A., 2020. Coursera | Online Courses & Credentials From Top Educators Coursera. [online] Coursera. Available at: <https://www.coursera.org/learn/machine-learning/supplement/bjjZW/normal-equation> [Accessed 6 April 2020].

[15] Guang-Bin Huang, Qin-Yu Zhu and Chee-Kheong Siew, "Extreme learning machine: a new learning scheme of feedforward neural networks," 2004 IEEE International Joint Conference on Neural Networks (IEEE Cat. No.04CH37541), Budapest, 2004, pp. 985-990 vol.2, doi: 10.1109/IJCNN.2004.1380068.

บรรณานุกรม (ต่อ)

[16] X. Liu, S. Lin, J. Fang, Z. Xu, "Is extreme learning machine feasible? a theoretical assessment (part I)," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 26, no. 1, pp. 7-20, 2015.

[17] S. Lin, X. Liu, J. Fang, Z. Xu, "Is extreme learning machine feasible? A theoretical assessment (Part II)", *IEEE Trans. Neural Networks Learn. Syst.*, vol. 26, no. 1, pp. 21-34, 2015.

[18] Gao Huang, Guang-Bin Huang, Shiji Song, Keyou You, "Trend in extreme learning machines: A review.", Neural Networks 61, pp.32-48, 2015.

[19] Van Heeswijk, Mark & Miche, Yoan & Lindh-Knuutila, T. & Hilbers, Peter & Honkela, Timo & Oja, E. & Lendasse, Amaury. (2009). "Adaptive ensemble models of extreme learning machines for time series prediction", Proceedings of the 19th International Conference on Artificial Neural Networks: Part II, ICANN '09.

[20] S. J. Pan and Q. Yang, "A Survey on Transfer Learning," in IEEE Transactions on Knowledge and Data Engineering, vol. 22, no. 10, pp. 1345-1359, Oct. 2010, doi: 10.1109/TKDE.2009.191.

[21] N. Liu and H. Wang, "Ensemble Based Extreme Learning Machine," in IEEE Signal Processing Letters, vol. 17, no. 8, pp. 754-757, Aug. 2010, doi: 10.1109/LSP.2010.2053356.

[22] Kasun, Liyanaarachchi & Zhou, Hongming & Huang, Guang-Bin & Vong, Chi-Man. (2013). "Representational Learning with ELMs for Big Data," IEEE Intelligent Systems. 28. 31-34.

[23] Y. Simmhan and M. U. Noor, "Scalable prediction of energy consumption using incremental time series clustering," 2013 IEEE International Conference on Big Data, Silicon Valley, CA, 2013, pp. 29-36, doi: 10.1109/BigData.2013.6691774.

บรรณานุกรม (ต่อ)

[24] Wong, P., Vong, C., Gao, X. and Wong, K., 2014. Adaptive Control Using Fully Online Sequential-Extreme Learning Machine and a Case Study on Engine Air-Fuel Ratio Regulation. Mathematical Problems in Engineering, 2014, pp.1-11.

[25] S. Scardapane, D. Comminiello, M. Scarpiniti and A. Uncini, "Online Sequential Extreme Learning Machine With Kernels," in *IEEE Transactions on Neural Networks and Learning Systems*, vol. 26, no. 9, pp. 2214-2220, Sept. 2015, doi: 10.1109/TNNLS.2014.2382094.

[26] Guo, W., Xu, T. and Tang, K., 2016. M-estimator-based online sequential extreme learning machine for predicting chaotic time series with outliers. Neural Computing and Applications, 28(12), pp.4093-4110.

[27] J. Tang, C. Deng and G. Huang, "Extreme Learning Machine for Multilayer Perceptron," in IEEE Transactions on Neural Networks and Learning Systems, vol. 27, no. 4, pp. 809-821, April 2016, doi: 10.1109/TNNLS.2015.2424995.

[28] K. Phurattanaprapin and P. Horata, "Extended hierarchical extreme learning machine with multilayer perceptron," 2016 13th International Joint Conference on Computer Science and Software Engineering (JCSSE), Khon Kaen, 2016, pp. 1-5, doi: 10.1109/JCSSE.2016.7748874.

[29] J. Park and J. Kim, "Online recurrent extreme learning machine and its application to time-series prediction," 2017 International Joint Conference on Neural Networks (IJCNN), Anchorage, AK, 2017, pp. 1983-1990, doi: 10.1109/IJCNN.2017.7966094.

[30] Zhang, H., Zhang, S. and Yin, Y., 2017. Online sequential ELM algorithm with forgetting factor for real applications. Neurocomputing, 261, pp.144-152.

[31] Salaken, S., Khosravi, A., Nguyen, T. and Nahavandi, S., 2017. Extreme learning machine based transfer learning algorithms: A survey. Neurocomputing, 267, pp.516-524.

[32] W. Zhu, W. Yu, B. Kan and G. Liu, "Smart Meter Data Analytics Based on Modified Streaming k-Means," 2017 3rd International Conference on Big Data Computing and

บรรณานุกรม (ต่อ)

Communications (BIGCOM), Chengdu, 2017, pp. 328-333, doi: 10.1109/BIGCOM.2017.49.

[33] M. Roy, S. K. Bose, B. Kar, P. K. Gopalakrishnan and A. Basu, "A Stacked Autoencoder Neural Network based Automated Feature Extraction Method for Anomaly detection in On-line Condition Monitoring," 2018 IEEE Symposium Series on Computational Intelligence (SSCI), Bangalore, India, 2018, pp. 1501-1507, doi: 10.1109/SSCI.2018.8628810.

[34] Ribeiro, M., Grolinger, K., ElYamany, H., Higashino, W. and Capretz, M., 2018. Transfer learning with seasonal and trend adjustment for cross-building energy forecasting. Energy and Buildings, 165, pp.352-363.

[35] C. Chen, B. Jiang and X. Jin, "Parameter Transfer Extreme Learning Machine based on Projective Model," 2018 International Joint Conference on Neural Networks (IJCNN), Rio de Janeiro, 2018, pp. 1-8, doi: 10.1109/IJCNN.2018.8489244.

[36] W. Cao, Z. Ming, Z. Xu, J. Zhang and Q. Wang, "Online Sequential Extreme Learning Machine With Dynamic Forgetting Factor," in IEEE Access, vol. 7, pp. 179746-179757, 2019.

[37] A. Hammoudeh, M. Fraihat and M. Almomani, "Selective Ensemble Model for Telecom Churn Prediction," 2019 IEEE Jordan International Joint Conference on Electrical Engineering and Information Technology (JEEIT), Amman, Jordan, 2019, pp. 485-487, doi: 10.1109/JEEIT.2019.8717406.

[38] H. T. Thu Thuy, D. T. Anh and V. T. Ngoc Chau, "Incremental Clustering for Time Series Data Based on an Improved Leader Algorithm," 2019 IEEE-RIVF International Conference on Computing and Communication Technologies (RIVF), Danang, Vietnam, 2019, pp. 1-6, doi: 10.1109/RIVF.2019.8713702.

บรรณานุกรม (ต่อ)

[39] Fan C., Sun Y., Xiao F., Ma J., Lee D., Wang J. and Tseng Y., 2020. Statistical investigations of transfer learning-based methodology for short-term building energy predictions. Applied Energy, 262, p.114499.

[40] Mulla, R., 2020. *Hourly Energy Consumption*. [online] Kaggle.com. Available at: <https://www.kaggle.com/robikscube/hourly-energy-consumption> [Accessed 6 October 2020].

ภาคผนวก ก
ผลงานวิจัยเผยแพร่


iEECON 2019
The 2019 International Electrical Engineering Congress
March 6-8, 2019 Hua Hin, Thailand
The Future of Life
Power & Energy
Computer & IT
Electronics & Controls
Communications
Digital Signal Processing
Important Dates
Papers submission deadline: October 8, 2018
Paper acceptance notification: December 11, 2018
Camera-ready submission deadline: January 14, 2019
http://www.ieecon2019.mut.ac.th
EEAAT
IEEE
THAILAND SECTION

Comparison Study on Artificial Neural Network and Online Sequential Extreme Learning Machine in Regression Problem

Charnon Chupong
Electrical Engineering Department
Rajamangala University of Technology Thanyaburi
Pathumtani, Thailand
charnon.c@en.rmutt.ac.th

Boonyang Plangklang
Electrical Engineering Department
Rajamangala University of Technology Thanyaburi
Pathumtani, Thailand
boonyang.p@en.rmutt.ac.th

***Abstract*— At present, machine learning techniques is frequently used to extraction and/or prediction some information from enormous collected data. Traditional machine learning is performed batch learning like a popular one, Artificial Neural Network (ANN). That means it can't integrate new information into already trained model but it is retrained from scratch. Online Sequential Extreme Learning Machine (OS-ELM) is a one of online incremental machine learning techniques that can learn and update model from new receive data without to retrain the model. This paper shows the result of comparison experiment between ANN and OS-ELM and found that, the OS-ELM has acceptable performance in situation of few initial data for training and/or statistic properties of data is changing while it working.**

Keywords—online incremental learning, extreme learning machine, artificial neural network

I. Introduction

According to the growth of Internet of Things (IoT) technology and Digital Economy, we have enormous data increasing daily. There is an article said "90% of all the data in the world has been generated over the last two years" [1]. Machine Learning is commonly used to extract useful information from collected data and/or predict some interested information. Artificial Neural Network (ANN) is a popular machine learning model because it is easy to implement and it can good represent the nonlinear relationship between data. But for update model, the traditional ANN can't integrate new information into already trained model but it instead retrained new model from scratch [1]. This method is time and cost consuming, then the model which can perform online learning from incremental data is solution for this problem. Online incremental machine learning is the algorithm for update the model by learning from new receive data without forgetting past data and not to retrain the model from scratch. Online incremental machine learning is enable the devices to adapt to individual user and environment especially in smart home system [2,3]. There are many algorithms developed for online incremental machine learning e.g. Incremental Support Vector Machine [4], Online Random Forest [5], Learn++ [6], Stochastic Gradient Descent [7] and Online Sequential Extreme Learning Machine [8].

There are some of papers present the performance evaluation of OS-ELM but almost of them focus on classification problem [8,9,11,15]. Then this paper will focus on performance evaluation of OS-ELM in regression problem. Experiments in this paper are for compare the accuracy of prediction between ANN and OS-ELM when following situations occur 1) device is installed to new site or assigned to work with new task with have only few initial data to training and let OSELM learn from receive data while working and 2) there are some changing in data e.g. user behavior is changing and/or working environment is changing how the OSELM can adapt itself to that change. We simulate that change by change the statistic properties of data.

This paper is organized as follow. Section II gives some review of ELM and OS-ELM. Section III present the regression experiment and result comparison between ANN and OS-ELM in situation of few initial data for training the model and statistic properties of data is changing. And section IV is conclusion and future work.

II. Related Works

A. Extreme Learning Machine (ELM)

ELM was proposed by Guang-Bin Huang [9]. The main advantage of ELM is training speed, it extreme faster than conventional ANN because it does not implement the slow gradient-based learning algorithms. ELM is a learning algorithm for single hidden layer feed-forward neural network which weights and bias values connecting between input layer and hidden layer are randomly generated and fixed. The weights connecting between hidden layer and output layer can be computed directly, then learning process is extremely fast. The structure and detail of ELM can describe as follow.

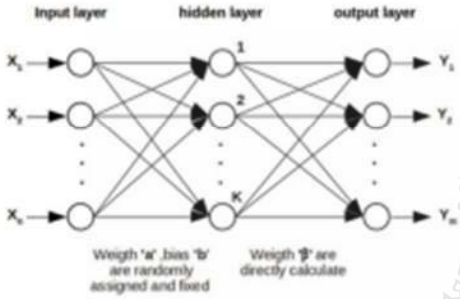


Fig. 1. Structure of ELM

For K training samples (x_j, y_j) where $x_j \in \mathbb{R}^n$ and $y_j \in \mathbb{R}^m$,the model have L hidden nodes relation between x_j and y_j can be expressed as follow.

$$y_j = \sum_{i=1}^{L} \beta_i \, G(a_i \cdot x_j + b_i), \qquad j = 1,2,\ldots,K \quad (1)$$

where $\beta_i \in \mathbb{R}^m$ is the weight connection between hidden layer and output layer, $a_i \in \mathbb{R}^n$ and $b_i \in \mathbb{R}$ are weight and bias connection between input layer and hidden layer, and$G: \mathbb{R} \to \mathbb{R}$ is activation function in hidden layer. Equation 1 can be simplifying as.

$$Y = H\beta \quad (2)$$

where

$$H = \begin{bmatrix} G(a_1 x_1 + b_1) & \cdots & G(a_L x_1 + b_L) \\ \vdots & & \vdots \\ G(a_1 x_n + b_1) & \cdots & G(a_L x_n + b_L) \end{bmatrix}_{K \times L}$$

$$\beta = \begin{bmatrix} \beta_1^T \\ \vdots \\ \beta_L^T \end{bmatrix}_{L \times m} \text{ and } Y = \begin{bmatrix} y_1^T \\ \vdots \\ y_K^T \end{bmatrix}_{K \times m}$$

$\boldsymbol{H}$ is called the hidden layer output matrix of neural network [9]. In the ELM a_i, b_i is randomly generated and fixed then $\boldsymbol{H}$ is fixed, that mean we can train the neural network by calculate the β that satisfy (2). If $\boldsymbol{H}$ is square matrix (number of training samples equal to number of hidden nodes) then weight β can be calculated as.

$$\beta = H^{-1}Y \quad (3)$$

In general case the number of training samples is greater than the number of node in hidden layer, then weight β can not be calculated as in (3). In this case the weight β can be calculated by using Moore-Penrose generalized inverse [10] as follow.

$$\beta = H^{\dagger}Y$$
$$\text{where } H^{\dagger} = (H^T H)^{-1} H^T \quad (4)$$

Although ELM uses randomly generated weights, it still maintains the universal approximation capability of neural networks [11, 12, 13, 14].

B. Online Sequential Extreme Learning Machine (OS-ELM)

ELM have fast learning speed and easy to implement, then it has a potential for online incremental learning application. Online Sequential Extreme Learning Machine (OS-ELM) was also proposed by Guang-Bin Huang [15]. OS-ELM is an ELM which able to learn incremental data online one-by-one or chunk-by-chunk by applying recursive least squares method [16] to ELM. OS-ELM algorithms consist of two phases: boosting phase and sequential learning phase as follow.

Phase 1, boosting phase is a phase for training OS-ELM with initial data. It's start with random generate weight and bias connection between input layer and hidden layer and calculate matrix $\boldsymbol{H}$ in (2). Equation (3) or (4) used to calculate initial weight β.

Phase 2, sequential learning phase is phase for incremental learning from each further coming data. Suppose now we are given another data (x_j, y_j) where $j = K+1, K+2 \ldots$ do

a) calculate the hidden layer output matrix for new coming data $\boldsymbol{H}_{(K+1)}$

b) calculate latest weight $\beta_{(K+1)}$ based on recursive least square algorithm as follow.

$$\beta_{(K+1)} = \beta + M_{K+1} H_{K+1} (y_{K+1}^T - H_{K+1}^T \beta)$$
$$\text{where } M_{K+1} = M - \frac{M H_{K+1} H_{K+1}^T M}{1 + H_{K+1}^T M H_{K+1}}$$
$$\text{and } M = (H^T H)^{-1} \quad (5)$$

From (5) we can see $\beta_{(K+1)}$ is a function of β that mean OS-ELM can learn new data without complete forgetting previous learned data.

III. Experiment and Result

We setup experiment for comparison study between ANN and OS-ELM in regression problem with two situations 1) few initial data for training, and 2) statistic properties of data is changing. In this experiment sci-kit learn on python3.6.7 is used as tool to create ANN and OS-ELM model as describe above. The software is run on Linux Ubuntu 18.01.1 LTS with memory 4GB and Intel core i3 2.7GHz CPU, detail in experimentation and results as follow.

A. Few initial data for training

We use well know dataset, "Diabetes" from scikit-learn tool. The dataset has 442 samples which target value is blood sugar's level and 10 dimension of features. We choose only 3 features age, BMI and blood's pressure to make a blood sugar's level prediction. We compare 3 types of model 1) ANN which trained by 5 samples, 2) OS-ELM which initial trained by 5 samples and incremental trained by one by one sample while prediction and, 3) ANN which trained by whole samples in data set. All models have similar structure with 1 hidden layer 5 hidden node, sigmoid function is used as activation function and in output layer identity function is used as activation function. Mean Absolute Percentage Error (MAPE) is used as a performance metrics and results as shown in table 1 and figure 2.

TABLE 1. RESULT OF PREDICTION WITH FEW INITIAL DATA FOR TRAINING

Model/ Metrics	ANN trained by 5 samples	OS-ELM	ANN trained by whole samples
MAPE	52.91	47.30	43.34
Training execution time [ms]	28	0.4	48

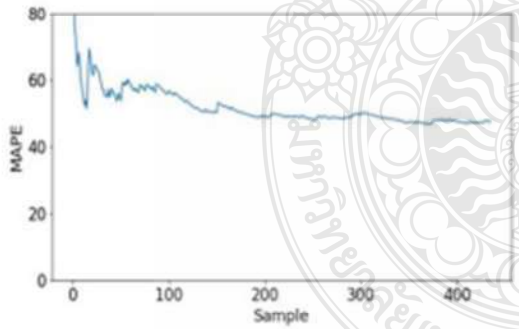


Fig. 2. Trend of MAPE value of OS-ELM while incremental learning

As result show that, the OS-ELM model with incremental learning have MAPE significant lower than ANN which trained by 5 samples. That mean OS-ELM can be used to replace the conventional ANN in situation which we have only few initial data for training the model and we can let it incremental learning while it working. MAPE of ANN which trained by whole dataset is an idea for minimum MAPE for prediction this dataset. The trend of MAPE value of OS-ELM is decrease while incremental learning and stay still after pass the 200th sample, that describe how fast of the incremental learning process for this dataset.

B. Statistic properties of data is changing

For simulate the changing of statistic properties of data we generate 2 difference data set for regression by sci-kit learn tool called data set A and data set B. Data set A have 400 samples and data set B have 600 samples both have 3 features variable and 1 target variable created by difference multivariate linear equations.

Fig. 3. Generated data set for simulate the changing of statistic properties of data

We compare 3 types of model 1) ANN which trained by 200 samples from data set A and let it prediction data set A and B 2) OS-ELM which initial trained by 200 samples from data set A and incremental learning while prediction data set A and B 3) ANN which trained by whole samples in data set A and B. All models have similar structure with 1 hidden layer 20 hidden node, sigmoid function is used as activation function and in output layer identity function is used as activation function. Mean Absolute Percentage Error (MAPE) is used as a performance metrics and results as shown in table 2 and figure 4.

TABLE 2. RESULT OF PREDICTION WITH STATISTIC PROPERTIES OF DATA IS CHANGING

Model/ Metrics	ANN trained by data set A only	OS-ELM	ANN trained by whole samples
MAPE	11.01	7.88	7.30
Training execution time [ms]	65	1	165

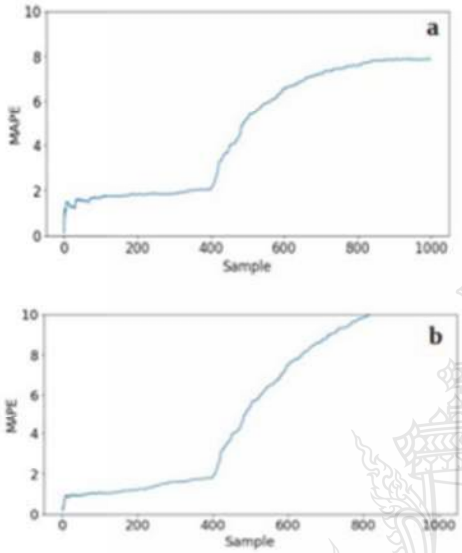


Fig. 4. Trend of MAPE value of a) OS-ELM
b) ANN trained by data set A only

As result show that, the OS-ELM with incremental learning have MAPE significant lower than ANN which trained by data set A only. That mean OS-ELM is more robustness to the changing of data than ANN. MAPE of ANN which trained by whole dataset is an idea for minimum MAPE for prediction in this dataset. The figure 4 show trend of MAPE value compare between OS-ELM and ANN, we can see after switch to data set B the MAPE of both are increase but it more convergence in OS-ELM model.

IV. Conclusion and Future Work

From the results of experiments show that, 1) in the situation of few initial training data OSELM can work better than ANN model trained with few data and performance closed to ideal case that ANN model which trained by whole data and 2) in the situation of have some changing statistic properties of data OSELM can work better than ANN model and performance closed to ideal case that ANN model which trained by whole data. Our future work is use OSELM for prediction time series data in smart grid application and improve it online accuracy by introduce the forgetting factor.

References

[1] Viktor Losing, Barbara Hammer and Heiko Wersing, "Incremental on-line learning: A review and comparison of state of the art algorithms," Neurocomputing, vol. 275, pp.1261-1274, 2018.

[2] R.Yang, M.W. Newman, "Learning from a learning thermostat: lesson for intelligent system for the home," Proceeding of the UbiComp 13, pp.93-102, 2013.

[3] B.D. Carolis, S. Ferilli, D.Redavid, "Incremental learning of daily routine as workflows in a smart home environment, ACM Trans. Interact. Intell, Syst. 4, pp.20:1-20:23, 2015.

[4] G.Cauwenberghs, T.Poggio, "Incremental and decremental support vector machine learning," Proceedings of the 2001 Neural Information Processing Systems (NIPS), 2001.

[5] A.Saffari, C.Leistner, J.Santner, M.Godec, H.Bischof, "On-line random forests," Proceedings of the 2009 IEEE Twelfth International Conference on Computer Vision Workshops, 2009

[6] R.Polikar, L.Upda, S.Upda, V.Honavar, "Learn++: an incremental learning algorithm for supervised neural networks," IEEE Trans. Syst. Man Cybern. 31 (4), pp.497-508, 2001

[7] L.Bottou, "Large-scale machine learning with stochastic gradient descent," Proceedings of the COMPSTAT 2010, Springer, pp.177–186, 2010.

[8] Nan-ying Liang, Guang-Bin Huang, P.Saratchandran and N.Sundarrajan, "A Fast and Accurate Online Sequential Learning Algorithm for Feedforward Networks." IEEE Tran. On Neural Networks, Vol.17, No.6, pp.1411-1423, November 2006

[9] Guang-Bin Huang, Qin-Yu Zhu, Chee-Kheong Siew, "Extreme learning machine: Theory and applications." Neurocomputing 70, pp.489-501, 2006.

[10] C.R. Rao, S.K. Mitra, "Generalized Inverse of Matrices and its Applications," Wiley, New York, 1971.

[11] Gao Huang, Guang-Bin Huang, Shiji Song, Keyou You, "Trend in extreme learning machines: A review.", Neural Networks 61, pp.32-48, 2015.

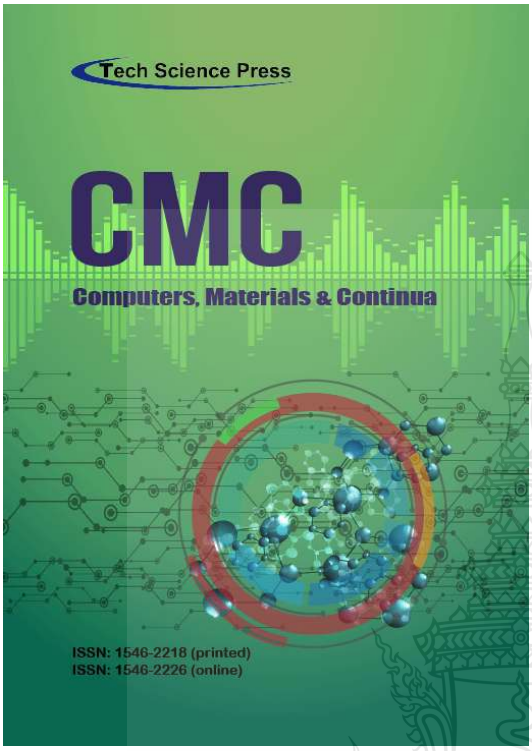
[12] C. Li, Z. Ding, G. Zhang, L.Xu. "Prediction of building energy consumption a comparative study," IEEE Chinese Automation Congress (CAC), pp.1691-1697, 2017.

[13] X. Liu, S. Lin, J. Fang, Z. Xu, "Is extreme learning machine feasible? a theoretical assessment (part I)," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 26, no. 1, pp. 7-20, 2015.

[14] S. Lin, X. Liu, J. Fang, Z. Xu, "Is extreme learning machine feasible? A theoretical assessment (Part II)", IEEE Trans. Neural Networks Learn. Syst., vol. 26, no. 1, pp. 21-34, 2015.

[15] Guang-Bin Huang, Nan-Ying Liang, Hai-Jun Rong, P.Saratchandran and N.Sundararajan, "On-Line Sequential Extreme Learning Machine," The IASTED International Conference on Computational Intelligence (CI2005), Calgary, Canada, July4-6, 2005.

[16] E.K.P.Chong and S.H.Zak, "An introduction to optimization." New York, Wiley, 2001.



Vol.72, No.3, 2022

Computer Materials & Continua (CMC)
อยู่ในฐานข้อมูล Web of Science ระดับ Q2
ในปี ค.ศ.2022

Computers, Materials & Continua
DOI: 10.32604/cmc.2022.028416
Article

Tech Science Press

Incremental Learning Model for Load Forecasting without Training Sample

Charnon Chupong and Boonyang Plangklang*

Faculty of Engineering, Rajamangala University of Technology Thanyaburi, Pathum Thani, 12110, Thailand
*Corresponding Author: Boonyang Plangklang. Email: boonyang.p@en.rmutt.ac.th
Received: 09 February 2022; Accepted: 14 March 2022

Abstract: This article presents hourly load forecasting by using an incremental learning model called Online Sequential Extreme Learning Machine (OS-ELM), which can learn and adapt automatically according to new arrival input. However, the use of OS-ELM requires a sufficient amount of initial training sample data, which makes OS-ELM inoperable if sufficiently accurate sample data cannot be obtained. To solve this problem, a synthesis of the initial training sample is proposed. The synthesis of the initial sample is achieved by taking the first data received at the start of working and adding random noises to that data to create new and sufficient samples. Then the synthesis samples are used to initial train the OS-ELM. This proposed method is compared with Fully Online Extreme Learning Machine (FOS-ELM), which is an incremental learning model that also does not require the initial training samples. Both the proposed method and FOS-ELM are used for hourly load forecasting from the Hourly Energy Consumption dataset. Experiments have shown that the proposed method with a wide range of noise levels, can forecast hourly load more accurately than the FOS-ELM.

Keywords: Incremental learning; load forecasting; Synthesis data; OS-ELM

1 Introduction

Global photovoltaic system deployments from 2017 to 2020 increased from 384.45 to 707.50 GW, representing an approximately 84% increase in three years [1]. By 2020, global electric vehicle registrations increased by 41%, bringing the total number of electric vehicles to about 10 million [2]. By the end of 2020, 5 GW of grid-sized energy storage systems has been installed globally, a 50% increase from mid-2019 and a steady increase [3]. All of the above events have resulted in a dramatic change in electricity usage patterns from the past. Therefore, the models used for electrical load forecasting need to be constantly updated according to the changes in the patterns of electricity consumption. But in general, those conventional models cannot learn from newly received data to adapt themselves during operation. Improving the models require new and sufficient data to re-train the model from scratch, which time consuming and inconvenient.

To solve this problem, researchers have developed a model that can learn from newly received data without forgetting what was previously learned, called incremental learning. The examples

of incremental learning include Incremental Support Vector Machine (ISVM) [4], Online Random Forecast (ORF) [5], Incremental Learning Vector Quantization (ILVQ) [6], Learn++ [7], Stochastic Gradient Descent (SGD) [8], and Online Sequential Extreme Learning Machine (OS-ELM) [9]. In this article, OS-ELM was chosen because it is a fast-learning model suitable for short-term load forecasting and easy to implement in hardware with low computational power. OS-ELM is a single hidden layer feed-forward neural network model where the weights in the input layer are randomly generated and retained over time, while hidden layer weights are computed based on a recursive least square method. This structure allows OS-ELM to function similarly to conventional neural networks but learning speed is more quickly. The OS-ELM requires initial samples for initial training, the amount of this initial sample must be more or equal to the number of nodes in the OS-ELM hidden layer. Usually, the number of nodes in this hidden layer affects forecast accuracy, implying that the amount of initial training data also affects the forecast accuracy.

In the implementation of OS-ELM, there are cases where a sufficient initial training sample cannot be obtained, such as a new building or building that never records the electricity usage data. Such a problem may be solved by the Transfer Learning technique [10], where data from other similar tasks is used as the source for initial training. But users cannot be sure whether the source data is similar to the real data or not. The researchers later proposed improvements to the OS-ELM model so that it can be used without an initial training sample called Fully Online Sequential Extreme Learning Machine (FOSELM) [11].

In the image classification and object detection research field, there is a method to solve the insufficient training samples problem by creating augmented samples from real samples [12]. For example in [13], the authors use the Mosaic data augmentation method to create more training samples. But in the load forecasting research field, there are few studies on using the augmented sample to train the forecasting models.

To the convenient use of OS-ELM especially in the case of new buildings or buildings without historical data of energy usage, this article proposed a method that allows OS-ELM to be used without the need for initial training samples. The proposed method takes a single sample through the synthesis process by adding random noise. The synthesis process makes enough new samples for the initial training of the OS-ELM. The proposed method was compared to the FOSELM in short-term load forecasting using a dataset called Hourly Energy Consumption [14]. The dataset was based on the hourly electricity consumption of cities in the Eastern United States from nine utility companies. The rest of this article contains related research in part 2, the proposed method is presented in part 3, the experiment and results in part 4, analysis of the experimental results in part 5, and conclusion in part 6.

2 Related Research

2.1 Load Forecasting

Load forecasting is an integral part of the smart grid. Numerous studies have been conducted on load forecasting [15], load forecasting is divided into four forecasting periods.

1. Very short-term load forecasting is a forecast up to 1 h in advance where the forecast result is often used to control the power system quality.
2. Short-term load forecasting is a forecast one hour but not more than a week in advance. The forecast result is often used to balance the supply and demand use of electricity.
3. Medium-term load forecasting is a forecast one month but not more than 1 year in advance. The forecast result is often used in planning the fuel supply in the electricity generation.

4. Long-term load forecasting is a forecast more than one year in advance. The forecasting result is often used in investment planning in power plants or power system infrastructure.

There are two main models used for load forecasting that are commonly found in research.

1. Auto Regressive Integrated Moving Average (ARIMA) is popular for time series data analysis. ARIMA consists of:
   a) The Autoregressive (AR) part describes the linear regression between current data and past data, and
   b) The Moving Average (MA) part describes linear regression between current data and past forecast errors caused by white noise in the data.
   AR and MA are only applicable for stationary data. To improve AR and MA model to be applicable with non-stationary data "Integrated" part has been added and called ARIMA.
2. Artificial Neural Network (ANN) in time series forecasting the ANN model differs from the ARIMA model in that it can use non-linear activation functions and hidden layer structures. That allows the ANN can predict the non-linear relationship between input and output better than ARIMA models [16].

2.2 *Incremental Learning*

Modeling for load forecasting requires historical sample data to calculate parameters in the model. In ANN, various machine learning techniques are used to calculate ANN's parameters, this process is called "training". The sample data used for training must be sufficiently large and should have a pattern similar to the data that the model must forecast. The training is done once at the beginning of the operation, known as batch learning. But at present, the pattern of electricity load has changed all the time due to photovoltaic systems [17–19] electric vehicles [20,21], and energy storage systems [22,23]. Therefore models created by batch training method require constantly re-training the model from scratch to be able to forecast accurately. The re-training in such cases creates inconvenience in practice. Therefore, researchers have introduced a model that can learn from new incoming data and still remember the past data, known as online learning or incremental learning. There are some examples of incremental learning such as.

- Incremental Support Vector Machine (ISVM) [4] incremental version of Support Vector Machine (SVM) working by storage some data as "candidate vector". This candidate vector may be promoted as "support vector" according to newly receive data during operation.
- Online Random Forest (ORF) [5] works like Random Forecast, but the number of trees (the number of sub-models) increases if the new data received changes from the past data.
- Incremental Learning Vector Quantization (ILVQ) [6] a Learning Vector Quantization (LVQ) that can be expanded by increasing the number of prototypes in the model when the new data received changes from the past data.
- Learn++ [7] uses the same principle as the ensemble model like AdaBoost [24]. Sub-model is added and trained with new data that is randomly selected from past data. Data samples with high forecasting errors are more likely to be selected, therefore Learn++ can adjust according to changes in data.
- Stochastic Gradient Descent (SGD) [8] is an optimization method for adjusting the model's parameters without needing to use the entire batch of data at once. The model's parameter can be adjusted according to the change of newly received data.

- Online Sequential Extreme Learning Machine (OS-ELM) [9] is an incremental model that is characterized by learning speed and low computation cost allowing it to run on the machine with low computational power. The details of the OS-ELM are presented in the next sections.

2.3 *Extreme Learning Machine (ELM)*

ELM structure is like a single hidden layer feed-forward neural network (SLFN) model. The ELM was first introduced by Guang et al. in 2004 [25]. The most outstanding feature of ELM is its extremely high learning speed since ELM does not use an iterative method such as gradient descent to calculate the parameters but uses the normal equation instead. The structure of the ELM is shown in Fig. 1. The bias and weight values that are connected between the input and the hidden layer are randomly generated and remain constant. While the weight between the hidden layer and the output layer is calculated by the normal equation as follows.

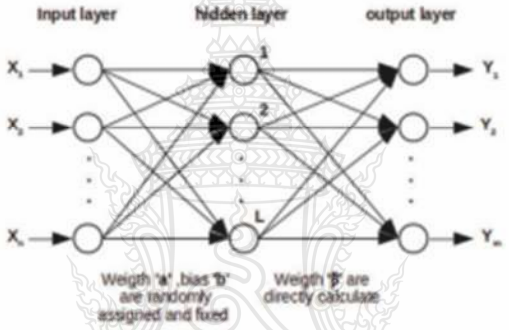


Figure 1: ELM Structure

If the dataset has N samples called $(\mathbf{x}_j, \mathbf{y}_j)$ where $\mathbf{x}_j \in \mathbb{R}^n$ and $\mathbf{y}_j \in \mathbb{R}^m$ numbers of nodes in the hidden layer equal to L relationship between $\mathbf{x}_j$ and $\mathbf{y}_j$ can be defined as follows.

$$\mathbf{y}_j = \sum_{i=1}^{L} \beta_i g\left(\mathbf{a}_i \mathbf{x}_j + b_i\right) \quad , j = 1, 2, \ldots, N \tag{1}$$

where $\beta_i \in \mathbb{R}^m$ are weights in the hidden layer, $g(\ldots) : \mathbb{R} \to \mathbb{R}$ is an activation function in the hidden layer, $\mathbf{a}_i \in \mathbb{R}^n$ are weights in the input layer, and $b_i \in \mathbb{R}$ are bias in input layer, Eq. (1) can be re-write in more simple form as follows:

$$\mathbf{Y} = \mathbf{H}\beta \tag{2}$$

where

$$\mathbf{H} = \begin{bmatrix} g(\mathbf{a}_1 \mathbf{x}_1 + b_1) & \cdots & g(\mathbf{a}_L \mathbf{x}_1 + b_L) \\ \vdots & \ddots & \vdots \\ g(\mathbf{a}_1 \mathbf{x}_N + b_1) & \cdots & g(\mathbf{a}_L \mathbf{x}_N + b_L) \end{bmatrix}_{N \times L}$$

$$\beta = \begin{bmatrix} \beta_1^T \\ \vdots \\ \beta_L^T \end{bmatrix}_{L \times M} \quad \text{and} \quad \mathbf{Y} = \begin{bmatrix} \mathbf{y}_1^T \\ \vdots \\ \mathbf{y}_N^T \end{bmatrix}_{N \times m}$$

Matrix $\mathbf{H}$ is called "Hidden layer output matrix" $\mathbf{a}_i$ and $\mathbf{b}_i$ are constant from random generated, causing $\mathbf{H}$ is also constant, therefore training ELM is calculating β as in Eq. (2). If the number of data or dimensions is not too large, the normal equation can be directly calculated more quickly as compared to the Gradient Descent method. β can be calculated as follows:

$$\beta = \left(\mathbf{H}^T\mathbf{H}\right)^{-1}\mathbf{H}^T\mathbf{Y} \tag{3}$$

or

$$\beta = \mathbf{K}^{-1}\mathbf{H}^T\mathbf{Y} \text{ where } \mathbf{K}=\mathbf{H}^T\mathbf{H} \tag{4}$$

Although ELM uses random weights and bias values in the input layer, many studies have shown that ELM has good performance as other learning algorithms [26,27]. The ELM model may encounter problems with numerical stability since sometimes inversion of the matrix $\mathbf{H}^T\mathbf{H}$ is not possible, the Singular Value Decomposition (SVD) method has been proposed [28] to find the inverse of $\mathbf{H}^T\mathbf{H}$.

2.4 Regularization Extreme Learning Machine (Re-ELM)

Numerical stability problems of the ELM model can be solved by adding the regularization factor [29] for calculating matrix K as shown in Eq. (5) which lowers the cost of computational than SVD.

$$\mathbf{K}=\mathbf{H}^T\mathbf{H} + \lambda\mathbf{I} \tag{5}$$

where λ is a very small value called the regularization factor and $\mathbf{I}$ is the identity matrix.

2.5 Online Sequential Extreme Learning Machine (OS-ELM)

The ELM model is a batch learning method, where it uses all available data to train the model and make the model work. While the model is running, it cannot learn more from the new incoming data. Therefore, a research paper proposed an improvement of the ELM model to be able to perform incremental learning called the Online Sequential Extreme Learning Machine (OS-ELM) [9] using the recursive principle of Eq. (4). At the beginning of work, the OS-ELM will calculate $\beta_0 = \mathbf{K}_0^{-1}\mathbf{H}_0^T\mathbf{Y}_0$ where $\mathbf{K}_0 = \mathbf{H}_0^T\mathbf{H}_0$ which subscript 0 mean round 0 of work or initial of work. When OS-ELM receives new sample(s) data causing the value of matrix $\mathbf{H}$ and $\mathbf{Y}$ change, therefore β_1 can be calculated as follows.

$$\beta_1 = \mathbf{K}_1^{-1}\begin{bmatrix}\mathbf{H}_0\\ \mathbf{H}_1\end{bmatrix}^T\begin{bmatrix}\mathbf{Y}_0\\ \mathbf{Y}_1\end{bmatrix} \tag{6}$$

$$\mathbf{K}_1 = \begin{bmatrix}\mathbf{H}_0\\ \mathbf{H}_1\end{bmatrix}^T\begin{bmatrix}\mathbf{H}_0\\ \mathbf{H}_1\end{bmatrix}$$

$$\mathbf{K}_1 = \mathbf{H}_0^T\mathbf{H}_0 + \mathbf{H}_1^T\mathbf{H}_1$$

$$\mathbf{K}_1 = \mathbf{K}_0 + \mathbf{H}_1^T\mathbf{H}_1 \tag{7}$$

and

$$\begin{bmatrix}\mathbf{H}_0\\ \mathbf{H}_1\end{bmatrix}^T\begin{bmatrix}\mathbf{Y}_0\\ \mathbf{Y}_1\end{bmatrix} = \mathbf{H}_0^T\mathbf{Y}_0 + \mathbf{H}_1^T\mathbf{Y}_1$$

$$\begin{bmatrix}\mathbf{H}_0\\ \mathbf{H}_1\end{bmatrix}^T\begin{bmatrix}\mathbf{Y}_0\\ \mathbf{Y}_1\end{bmatrix} = \mathbf{K}_0\mathbf{K}_0^{-1}\mathbf{H}_0^T\mathbf{Y}_0 + \mathbf{H}_1^T\mathbf{Y}_1$$

$$\begin{bmatrix}\mathbf{H}_0\\ \mathbf{H}_1\end{bmatrix}^T\begin{bmatrix}\mathbf{Y}_0\\ \mathbf{Y}_1\end{bmatrix}=\mathbf{K}_0\beta_0+\mathbf{H}_1^T\mathbf{Y}_1$$

Substitute the $\mathbf{K}_0$ value from Eq. (7) into the above equation.

$$\begin{bmatrix}\mathbf{H}_0\\ \mathbf{H}_1\end{bmatrix}^T\begin{bmatrix}\mathbf{Y}_0\\ \mathbf{Y}_1\end{bmatrix}=\left(\mathbf{K}_1-\mathbf{H}_1^T\mathbf{H}_1\right)\beta_0+\mathbf{H}_1^T\mathbf{Y}_1$$

$$\begin{bmatrix}\mathbf{H}_0\\ \mathbf{H}_1\end{bmatrix}^T\begin{bmatrix}\mathbf{Y}_0\\ \mathbf{Y}_1\end{bmatrix}=\mathbf{K}_1\beta_0-\mathbf{H}_1^T\mathbf{H}_1\beta_0+\mathbf{H}_1^T\mathbf{Y}_1 \tag{8}$$

Substitute Eq. (8) into Eq. (6).

$$\beta_1=\mathbf{K}_1^{-1}\left(\mathbf{K}_1\beta_0-\mathbf{H}_1^T\mathbf{H}_1\beta_0+\mathbf{H}_1^T\mathbf{Y}_1\right)$$

$$\beta_1=\beta_0-\mathbf{K}_1^{-1}\mathbf{H}_1^T\mathbf{H}_1\beta_0+\mathbf{K}_1^{-1}\mathbf{H}_1^T\mathbf{Y}_1$$

$$\beta_1=\beta_0+\mathbf{K}_1^{-1}\mathbf{H}_1^T\left(\mathbf{Y}_1-\mathbf{H}_1\beta_0\right) \text{ where}$$

$$\mathbf{K}_1=\mathbf{K}_0+\mathbf{H}_1^T\mathbf{H}_1 \tag{9}$$

Eq. (9) can be arranged in a general recursive form as:

$$\beta_{k+1}=\beta_k+\mathbf{K}_{k+1}^{-1}\mathbf{H}_{k+1}^T\left(\mathbf{Y}_{k+1}-\mathbf{H}_{k+1}\beta_k\right) \text{ where } \mathbf{K}_{k+1}=\mathbf{K}_k+\mathbf{H}_{k+1}^T\mathbf{H}_{k+1} \tag{10}$$

The subscription k means the k^{th} samples and the subscription $k+1$ means the $(k+1)^{th}$ samples. Each incoming sample does not need to be equal in numbers. In short, the OS-ELM works in two steps as follows.

- The initial step is like the ELM, the initial samples are used to calculate β_0 by Eq. (4). The number of initial samples needs to be more than or equal to the number of nodes in a hidden layer of the model.
- The incremental learning step uses the newly receives sample to adjust the model parameters as in Eq. (10).

2.6 *Regularization Online Sequential Extreme Learning Machine (ReOS-ELM)*

Like the ELM, if k is a non-invertible matrix then the OS-ELM has a numerical stability problem that can be solved by adding a regularization factor [30] as shown in Eq. (11).

$$\mathbf{K}_{k+1}=\mathbf{K}_k+\mathbf{H}_{k+1}^T\mathbf{H}_{k+1}+\lambda\mathbf{I} \tag{11}$$

where λ is a very small value called the regularization factor and $\mathbf{I}$ is the identity matrix.

2.7 *Fully Online Sequential Extreme Learning Machine (FOS-ELM)*

The OS-ELM and ReOS-ELM require samples for initial training, but in some situations, we are unable to obtain enough appropriate samples for initial training. Therefore, the FOS-ELM model is presented, which is a model that does not need the initial data for initial training [11] by setting initial parameters $\beta_0=0$, $\mathbf{K}_0=\lambda\mathbf{I}$. In other words, the FOS-ELM is initial training by the initial sample ($\mathbf{X}=\mathbf{0}$, $\mathbf{Y}=\mathbf{0}$).

3 Proposed Method

As mentioned above, the implementation of OS-ELM require sufficient and appropriate initial training samples. But in some situations, those samples cannot be obtained. Therefore, in this article, we present a method to solve that problem. The method presented in this article uses a single sample received at the start of working to synthesize sufficiently the initial training data. The data synthesis method is done by adding random noise to all features of the sample to create new samples. The noise is randomly generated according to uniform distribution and the user can adjust the level of noise that to be added to the sample called "percent noise". This proposed method can be described as a diagram in Fig. 2 and an algorithm 1. Please note that this article uses ReOS-ELM instead of OS-ELM to avoid the numerical stability problem.

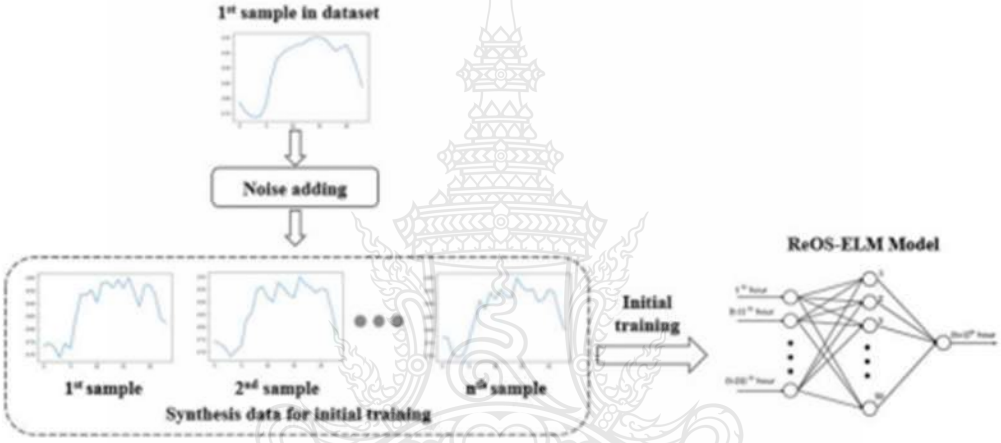


Figure 2: Diagram of the proposed method

Algorithm 1: Synthesize data by noise adding

INPUT: *data* = 1st sample in dataset
N = numbers of output sample required
percent_noise = noise level (set by user)
OUTPUT: *synt_data*
STEP:
1: for *n=0* **to** *n=N:*
2: **for** *i*=0 **to** *i*=length(*data*):
3: *rand[i]* = random value between [0,1) in uniform distribution
4: *synt_data[n][i] = data[i] + (data[i]*percent_noise/100)*rand[i]*
5: **end for**
6: synt_data[n]/max(synt_data[n])
7: end for
8: return *synt_data*
9: end

To improve forecasting accuracy, ten ReOS-ELM models were combined to form an ensemble model and the mean function is used as an ensemble function. In other words, the forecast values are obtained by averaging the 10 forecast values from all ReOS-ELM models. All 10 models are learning from the same sample both the initial training phase and the incremental learning phase as shown in Fig. 3.

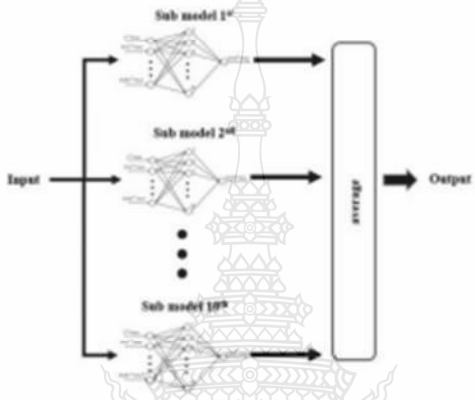


Figure 3: The proposed ensemble model

4 Experiment Results and Discussion

This experiment uses a dataset from www.kaggle.com named "Hourly Energy Consumption" [14], which is the hourly electricity usage (hourly load profile) of cities in the eastern United States that gathered from nine utility companies. In the experiment, data are divided into two parts: (1) target is the 1-hour load each and (2) input is the 24-h load before the target as shown in Fig. 4. Inputs are fed to the model so that the model forecasts the next hour's load. The forecast values obtained from the model are compared to the target and forecasting errors can be determined. Both the input and the target are used in the model for incremental learning. The model uses an ensemble model as shown in Fig. 3, where the sub-model is ReOS-ELM with 24 nodes in input layers, 50 nodes in hidden layers, and 1 node in the output layer. The hidden layer uses Sigmoid as the activation function.

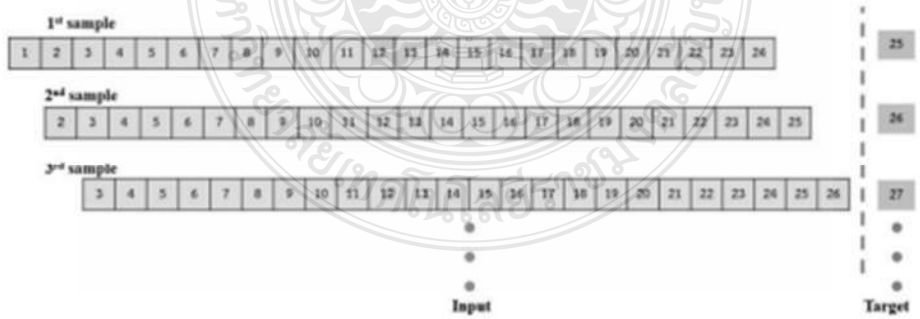


Figure 4: Dataset is divided into Input and Target

The experiment in this article uses two incremental learning models: (1) the FOS-ELM model is the baseline for comparison, and (2) the ReOS-ELM model trained with 50 synthesized samples by the method presented in Section 3. In addition, various percent noises for synthesizing the sample are also tested to determine the appropriate values. The experimental flowchart is shown in Fig. 5, using the first 72 h of data in the dataset to study the early phase of the forecasting operation.

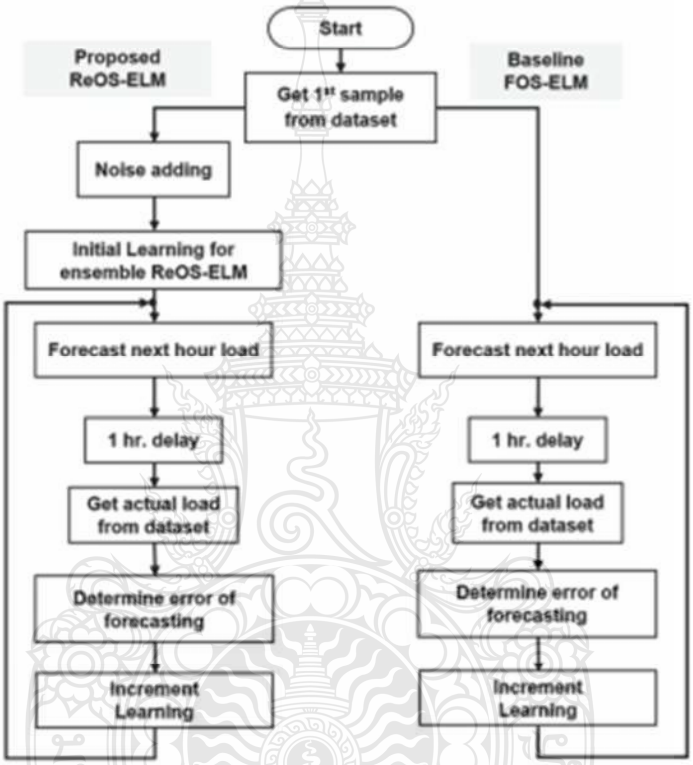


Figure 5: Flow chart of the experiment

The performance metrics used to compare the accuracy of the forecasting is Mean Absolute Percentage Error (MAPE) and Mean Absolute Error (MAE). The experimental results are shown in Tab. 1.

Table 1: Experimental results

Dataset	Method	Percent noise	MAPE (%)	MAE (MW)
AEP	FOS-ELM (baseline)	n/a	2.18	316.74
	Proposed method	1	2.04	292.79
		5	**1.95**	**280.55**
		10	1.97	282.54

(Continued)

Table 1: Continued

Dataset	Method	Percent noise	MAPE (%)	MAE (MW)
		20	2.13	303.47
COMED	FOS-ELM (baseline)	n/a	1.86	231.64
	Proposed method	1	1.49	190.18
		5	**1.48**	**189.79**
		10	1.59	202.36
		20	1.87	237.19
DAYTON	FOS-ELM (baseline)	n/a	2.96	57.56
	Proposed method	1	2.85	55.34
		5	2.76	53.59
		10	**2.56**	**49.67**
		20	2.7	52.54
DEOK	FOS-ELM (baseline)	n/a	2.08	65.88
	Proposed method	1	1.84	59.02
		5	**1.77**	**57.19**
		10	1.96	62.08
		20	2.18	68.4
DOM	FOS-ELM (baseline)	n/a	2.43	202.02
	Proposed method	1	2.08	182.15
		5	2.01	176.08
		10	**1.98**	**173.02**
		20	2.06	179.42
DUQ	FOS-ELM (baseline)	n/a	3.46	57.22
	Proposed method	1	3.43	56.92
		5	2.75	45.33
		10	**2.37**	**38.79**
		20	2.45	40.21
EKPC	FOS-ELM (baseline)	n/a	4.83	60.17
	Proposed method	1	3.85	55.22
		5	3.66	49.29
		10	**3.08**	**39.02**
		20	3.58	44.82
FE	FOS-ELM (baseline)	n/a	2.77	223.51
	Proposed method	1	2.78	223.52
		5	2.54	205.94
		10	**2.25**	**182.58**
		20	2.43	197.85
NI	FOS-ELM (baseline)	n/a	1.60	169.84
	Proposed method	1	1.58	168.54
		5	**1.56**	**166.11**
		10	1.61	171.79
		20	1.69	178.72

From the results in Tab. 1, it was found that using the ReOS-ELM trained with samples synthesized by the proposed method can achieve lower forecasting error than the FOS-ELM. This is because FOS-ELM is like using a single sample for the initial training, resulting in the model facing the over-fit problem. While the ReOS-ELM model uses synthesized samples with sufficient numbers for the initial training, so the over-fit problems can be avoided.

From the experiment, it was found that the appropriate percent noise is flexible. That is to say, the forecasting error of the ReOS-ELM still lower than the FOS-ELM regardless of the percent noise value (except in the EKPC and FE dataset, when 1% percent noise is used, forecasting error of the ReOS-ELM is slightly higher than the FOS-ELM).

Different forecasting errors that occur when using different percent noise may be caused by the following reasons. At low percent noise, the synthesized samples for initial training are similar to each other, making the model more prone to over-fit problems. At high percent noise, the synthesized samples for initial training may differ from the real data, this makes the model prone to under-fit problems. The experiment found that approximately 5%–10% noise levels resulted in the lowest forecasting error.

5 Conclusion

The proposed method can allow ReOS-ELM or OS-ELM to run without an example for initial training. By adding noise to a single sample received at the start of the operation to increase the number of samples to be sufficient for initial training. The experiment found that whether using noise levels 1%, 5%, 10%, or 20% the ReOS-ELM can forecast loads more accurately than the FOS-ELM. The noise level for synthesis initial samples that allow the ReOS-ELM model to the most accurate forecasting is about 5%–10%.

Funding Statement: The authors received no specific funding for this study.

Conflicts of Interest: The authors declare that they have no conflicts of interest to report regarding the present study.

References

[1] Statistical Review of World Energy-BP, "Installed solar energy capacity," 2021. [Online]. Available: https://ourworldindata.org/grapher/installed-solar-pv-capacity.

[2] IEA, *Global EV Outlook*. Paris: IEA, 2021. [Online]. Available at: https://www.iea.org/reports/global-ev-outlook-2021.

[3] IEA, *Energy Storage*. Paris: IEA, 2021. [Online]. Available at: https://www.iea.org/reports/energy-storage.

[4] G. Cauwenberghs and T. Poggio, "Incremental and decremental support vector machine learning," *Advance Neural Information Processing Systems*, vol. 13, pp. 388–394, 2001.

[5] A. Saffari, C. Leistner, J. Santner, M. Godec and H. Bischof, "On-line random forests," in *Proc. of the 2009 IEEE Twelfth Int. Conf. on Computer Vision Workshops*, Kyoto, Japan, pp. 1393–1400, 2009.

[6] A. Sato and K. Yamada, "Generalized learning vector quantization," in *Proc. of the 1995 Neural Information Processing Systems*, Denver, Colorado, USA, pp. 423–429, 1995.

[7] R. Polikar, L. Upda, S. Upda and V. Honavar, "Learn++: An incremental learning algorithm for supervised neural networks," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 31, no. 4, pp. 497–508, 2001.

[8] T. Zhang, "Solving large scale linear prediction problems using stochastic gradient descent algorithms," in *Proc. of the Twenty-First Int Conf. on Machine Learning, ACM*, Banff, Alberto, Canada, pp. 116–123, 2004.

[9] N. Y. Liang, G. B. Huang, P. Saratchandran and N. Sundarrajan, "A fast and accurate online sequential learning algorithm for feedforward networks," *IEEE Transaction on Neural Networks*, vol. 17, no. 6, pp. 1411–1423, 2006.

[10] S. M. Salaken, A. Khosravi, T. Nguyen and S. Nahavandi, "Extreme learning machine based transfer learning algorithms: A survey," *Neurocomputing*, vol. 267, no. 10, pp. 516–524, 2017.

[11] P. K. Wong, C. M. Vong, X. H. Gao and K. I. Wong, "Adaptive control using fully online sequential extreme learning machine and a case study on engine air-fuel ratio regulation," *Mathematical Problems in Engineering*, vol. 2014, no. 3, pp. 1–11, 2014.

[12] P. Kaur, B. S. Khehra and E. B. S. Mavi, "Data augmentation for object detection: A review," in *2021 IEEE Int. Midwest Symp. on Circuits and Systems*, East Lansing, Michigan, USA, Virtual & Online Conference, pp. 537–543, 2021.

[13] W. Sun, L. Dai, X. R. Zhang, P. S. Chang and X. Z. He, "RSOD: Real-time small object detection algorithm in UAV-based traffic monitoring," *Applied Intelligence*, vol. 92, no. 6, pp. 1–16, 2021.

[14] R. Mulla, "Hourly Energy Consumption., Kaggle, 2018. https://kaggle.com/robikscue/hourly-energy-consumption.

[15] L. Hernandez, C. Baladron, J. M. Aguiar, B. Carro, A. J. Sanchez-Esguevillas *et al.*, "A survey on electric power demand forecasting: Future trends in smart grids, microgrids and smart buildings," *IEEE Communications Surveys & Tutorials*, vol. 16, no. 3, pp. 1460–1495, 2014.

[16] C. Kuster, Y. Rezgui and M. Mourshed, "Electrical load forecasting models: A critical systematic review," *Sustainable Cities and Society*, vol. 35, no. 11, pp. 257–270, 2017.

[17] Y. Wang, N. Zhang, Q. Chen, D. S. Kirschen, P. Li *et al.*, "Data-driven probabilistic net load forecasting with high penetration of behind-the-meter PV," *IEEE Transactions on Power Systems*, vol. 33, no. 3, pp. 3255–3264, 2018.

[18] S. Rahman, S. Saha, S. N. Islam, M. T. Arif, M. Mosadeghy *et al.*, "Analysis of power grid voltage stability with high penetration of solar PV systems," *IEEE Transactions on Industry Applications*, vol. 57, no. 3, pp. 2245–2257, 2021.

[19] I. Calero, C. A. Canizares, K. Bhattacharya and R. Baldick, "Duck-Curve mitigation in power grids with high penetration of PV generation," *IEEE Transactions on Smart Grid*, vol. 13, no. 1, pp. 314–329, 2022.

[20] Y. Kongjeen and K. Bhumkittipich, "Modeling of electric vehicle loads for power flow analysis based on PSAT," in *13th Int. Conf. on Electrical Engineering/Electronics, Computer, Telecommunications and Information Technology (ECTI-CON)*, Chiang Mai, pp. 1–6, 2016.

[21] B. Al-Hanahi, I. Ahmad, D. Habibi and M. A. S. Masoum, "Charging infrastructure for commercial electric vehicles: Challenges and future works," *IEEE Access*, vol. 9, pp. 121476–121492, 2021.

[22] W. Lee, J. Jung and M. Lee, "Development of 24-hour optimal scheduling algorithm for energy storage system using load forecasting and renewable energy forecasting," in *2017 IEEE Power & Energy Society General Meeting*, Chicago, pp. 1–5, 2017.

[23] O. H. Abdalla, G. Abdel-Salam and A. A. A. Mostafa, "Multifunction battery energy storage system for distribution networks," *Energy Engineering*, vol. 119, no. 2, pp. 569–589, 2022.

[24] Y. Freund and R. E. Schapire, "A decision-theoretic generalization of on-line learning and an application to boosting," *Journal of Computer and System Sciences*, vol. 55, no. 1, pp. 119–139, 1997.

[25] G. B. Huang, Q. Y. Zhu and C. K. Siew, "Extreme learning machine: A new learning scheme of feedforward neural networks," in *2004 IEEE International Joint Conference on Neural Networks (IEEE Cat. No.04CH37541)*, Budapest, vol. 2, pp. 985–990, 2004.

[26] X. Liu, S. Lin, J. Fang and Z. Xu, "Is extreme learning machine feasible? A theoretical assessment (part I)," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 26, no. 1, pp. 7–20, 2015.

[27] S. Lin, X. Liu, J. Fang and Z. Xu, "Is extreme learning machine feasible? A theoretical assessment (Part II)," *IEEE Trans. Neural Networks Learn. Syst*, vol. 26, no. 1, pp. 21–34, 2015.

[28] V. Klema and A. Laub, "The singular value decomposition: Its computation and some applications," *IEEE Transactions on Automatic Control*, vol. 25, no. 2, pp. 164–176, 1980.
[29] W. Deng, Q. Zheng and L. Chen, "Regularized extreme learning machine," in *2009 IEEE Symp. on Computational Intelligence and Data Mining*, Nashville, TN, USA, pp. 389–395, 2009.
[30] W. Guo, T. Xu, K. Tang, J. Yu and S. Chen, "Online sequential extreme learning machine with generalized regularization and adaptive forgetting factor for time-varying system prediction," *Hindawi Mathematical Problems in Engineering*, vol. 2018, no. 7, pp. 1–22, 2018.


Call for Papers
ICPEI 2022
INTERNATIONAL CONFERENCE ON POWER, ENERGY AND INNOVATIONS
19 - 21 October 2022 I Amari Hotel, Pattaya
The 2022 International Conference on Power, Energy and Innovations (ICPEI 2022) organized by the Electrical Engineering Academic Association (Thailand), EEAAT and The Sirindhorn International Thai-German Graduate School of Engineering (TGGS), KMUTNB, technical sponsored by IEEE Thailand Section and IEEE PES- Thailand Chapter. ICPEI is the conference of the Power, Energy and Innovations providing a forum for researchers and engineers involved in electric power and energy engineering to share ideas and result.
Topics of interest for submission including, but are not limited to:
Power
Smart grid technologies
Power system management
Modeling, analysis and operation
Planning and operation of power systems
Power system dynamics, stability and control
High voltage engineering
Condition monitoring and diagnostics
Lightning protection of power systems
Switching-mode power supplies and UPS
Power quality
Applications of power electronics in power system and generation/FACTS
Motor drives and motion control
Power transmission and distribution technology
Other related topics
Energy
Renewable energy technologies
Wind energy
Biogas and biomass
Solar cell technology
Hydrogen and fuel cells
Hybrid energy systems
Solar thermal applications
Solar electricity and PV application
Energy policy, planning and management
Electric Vehicle and charging system
Power electronics in traction and automotive
Analysis and design of electrical machines
Advances in sustainable buildings and cities
Energy storage and battery technology
Transportation electrification
Other related topics
For more information, please visit http://www.ICPEI.NET/2022/
Innovation
IT innovation
Nanotechnology
Information engineering
Technology for big data
Material engineering
Technological innovation
Innovations in power system
Technology-enhanced learning
Digital signal and image processing
Innovations for renewable energy
Other related topics
Important Dates
Extended Submission Deadline for Full Manuscript: August 10, 2022
Notification of Acceptance: August 15, 2022
Camera-ready Submission Deadline (Full Manuscript Only): August 31, 2022
Early Bird Registration Deadline: September 9, 2022
Conference Dates: October 19-21, 2022
SPONSORED BY
IEEE THAILAND SECTION
PES Power & Energy Society, Thailand Chapter
PEA
การไฟฟ้านครหลวง
Schneider Electric
Hitachi Energy
TGM
APE GROUP 20
GPSC
UNIPOWER ENGINEERING
PTS Construction
Organized by
KMUTNB
TGGS
EEAAT
Contact:
Assoc. Prof. Dr. Soamsiri Chataraskul
ICPEI2022 Secretariat
Assoc. Prof. Dr. -Ing. Thanapong Suwanasri
ICPEI2022 TPC
Email: icpei2022@gmail.com

2022 International Conference on Power, Energy and Innovations (ICPEI 2022)
October 19-21, 2022, Pattaya, THAILAND

Short-Term Load Forecasting by FOS-ELM with Re-Learning Method

Charnon Chupong
Department of Electrical Engineering
Rajamangala University of Technology Thanyaburi
Pathum Thani, Thailand
charnon.c@en.rmutt.ac.th

Nitikorn Junhuathon
Department of Electrical Engineering
Rajamangala University of Technology Thanyaburi
Pathum Thani, Thailand
nitikorn.j@en.rmutt.ac.th

Boonyang Plangklang
Department of Electrical Engineering
Rajamangala University of Technology Thanyaburi
Pathum Thani, Thailand
boonyang.p@en.rmutt.ac.th

***Abstract*— This article presents a method for short-term load forecasting by Fully Online Sequential Extreme Learning Machine (FOS-ELM) model. The model can learn incrementally without the need for initial training samples. The model is suitable for use in locations where historical load data is not available to train the model. This article also presents the improvement of the forecasting accuracy of the FOS-ELM model by letting the FOS-ELM model learn from each newly received sample more than once, called "re-learning". A comparative experiment between the FOS-ELM model without re-learning and the FOS-ELM model with re-learning found that the re-learning method can reduce the error in load forecasting. This is because the model can quickly adapt to the trends of new data. But after the fifth round of re-learning, the forecasting error value could no longer decrease. This is because the model has an over-fit with the latest learned data. In other words, re-learning allows the model to have lower forecasting errors, but the number of re-learning times must be chosen appropriately.**

Keywords*— *FOS-ELM, incremental learning, load forecasting, re-learning

I. Introduction

Load forecasting plays an important role in electricity grid management. Numerous investigations on the statistical method for time series forecasting, such as the Auto Regressive Moving Average (ARIMA) [1] have been conducted in the past. Following that, some progress was made in the use of a nonlinear feature to tackle the time series forecasting problem. The Artificial Neural Network (ANN) is a well-known model for non-linear features time series forecasting [2-3]. At the present day, the electricity grid has changed dramatically with high penetration of renewable energy, storage system, electric vehicle, and demand response program [4-6]. For this reason, the statistical properties of load data have significantly changed over time in an unforeseen way called 'concept drift' [7]. Concept drift must be addressed by retraining the model from scratch, which is an expensive and time-consuming operation.

To forecast the data with concept drift, researchers have proposed the incremental learning algorithm that can incrementally learn from new data without forgetting what has already been learned, some examples are Stochastic Gradient Descent [8], Learn++[9], Online Random Forest [10], Incremental Support Vector Machine [11], and Online Sequential Extreme Learning Machine [12]. In this article, we focus on the Online Sequential Extreme Learning Machine (OS-ELM) because it learns very fast and it is easy to implement. The OS-ELM requires sufficient data for offline initial training. In that case, the history data for initial training does not available, then some researchers have proposed the Fully Online Sequential Extreme Learning Machine (FOS-ELM) to solve this problem [13]. FOS-ELM doesn't require initial data for training it starts by setting some parameters in the model equal to zero and can be adjusted while increment learns from incoming data.

Some FOS-ELM parameters are generated randomly, which can cause the model to highly error in load forecasting due to under-fit problems. Therefore, this article presents an accuracy improvement of FOS-ELM in short-term load forecasting by re-learning from the sample more than one time. The proposed method is an analogy to adding weight to the latest sample for learning, allowing the model to adapt more quickly to new input. The rest of this article consists of related research in part 2, the proposed method in part 3, the experiment and result in part 4, and the conclusion in part 5.

II. Related Research

A. Extreme Learning Machine (ELM)

ELM is a single hidden layer feed-forward neural network (SLFN) learning technique. ELM was proposed by Guang-Bin Huang [14]. The most defining characteristic of ELM is learning speed, it can learn a hundred times faster than the ordinary back-propagation algorithm [14]. Since the sluggish gradient-based learning algorithm is not used, its learning speed is extremely fast. The model of ELM is the same as SLFN but the bias value and weights value of the link between the input layer and the hidden layer are randomly generated and fixed. The weight value of the link between the hidden layer and the output layer can be directly calculated. The

978-1-6654-6019-4/22/$31.00 ©2022 IEEE

working principle of ELM can be described as follows.

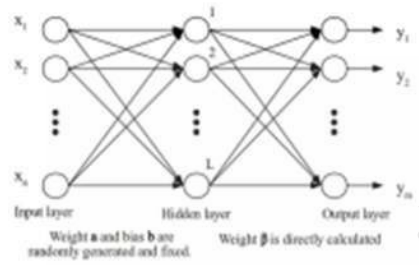


Figure 1 ELM Model

For any K training samples (x_j, y_j), where $x_j \in \mathbb{R}^n$ and $y_j \in \mathbb{R}^m$ the model has L nodes in a hidden layer. The following equations describe how input and output are related.

$$y_j = \sum_{i=1}^{L} \beta_i G(a_i x_j + b_i), \quad j = 1, 2, 3, \ldots K \qquad (1)$$

Where the $\beta_i \in \mathbb{R}^m$ is the weights of the link between the hidden layer and output layer, $G: \mathbb{R} \to \mathbb{R}$ is the activation function of the hidden layer, $a_i \in \mathbb{R}^n$ and $b_i \in \mathbb{R}$ are weights and biases of the link between the input and hidden layer. Equation (1) can be simplified as follow.

$$\mathbf{Y} = \mathbf{H}\boldsymbol{\beta} \qquad (2)$$

where

$$\mathbf{H} = \begin{bmatrix} G(a_1 x_1 + b_1) & \cdots & G(a_L x_1 + b_L) \\ \vdots & \ddots & \vdots \\ G(a_1 x_K + b_1) & \cdots & G(a_L x_K + b_L) \end{bmatrix}_{K \times L}$$

$$\mathbf{Y} = \begin{bmatrix} \mathbf{y}_1 \\ \vdots \\ \mathbf{y}_K \end{bmatrix}_{K \times m}, \text{ and } \quad \boldsymbol{\beta} = \begin{bmatrix} \boldsymbol{\beta}_1 \\ \vdots \\ \boldsymbol{\beta}_L \end{bmatrix}_{L \times m}$$

Matrix $\mathbf{H}$ is called "hidden layer output matrix" [14], while the learning process $\mathbf{H}$ and $\mathbf{Y}$ are already known, then $\boldsymbol{\beta}$ can be directly calculated as follow:

$$\boldsymbol{\beta} = \mathbf{H}^{\dagger}\mathbf{Y}, \quad \mathbf{H}^{\dagger} = (\mathbf{H}^{\mathrm{T}}\mathbf{H})^{-1}\mathbf{H}^{\mathrm{T}} \qquad (3)$$

where the $\mathbf{H}^{\dagger}$ is the pseudo-inverse of a matrix $\mathbf{H}$. There are several methods to calculate the $\mathbf{H}^{\dagger}$. The method shown in (3) tends to become singular in some applications [15], therefore we use the singular value decomposition (SVD) [16] to calculate the $\mathbf{H}^{\dagger}$.

B. Online Sequential Extreme Learning Machine (OS-ELM)

Guang-Bin Huang was proposed the OS-ELM in 2015 [12]. OS-ELM is based on ELM with the capacity to continuously learn from new arrival data without losing track of the past learned data. Applying the recursive least square method to ELM allows it to learn from new data one at a time or in chunks. OS-ELM has two-phase of working: 1) initialization phase and 2) sequential learning phase detailed as follows.

In the initialization phase, initial samples were used to create the hidden layer output matrix $\mathbf{H}_0$ and then calculate the $\boldsymbol{\beta}_0 = \mathbf{P}_0\mathbf{H}_0^{\mathrm{T}}\mathbf{Y}_0$ where $\mathbf{P}_0 = (\mathbf{H}_0^{\mathrm{T}}\mathbf{H}_0)^{-1}$ is the same as in ELM. Set a variable $c = 0$ as a sequence number of the incoming data. The sequential learning phase, when the $(c+1)^{th}$ data arrived, creates the new hidden layer output matrix $\mathbf{H}_{c+1}$ and calculates the new output weight $\boldsymbol{\beta}_{c+1}$ as follows.

$$\boldsymbol{\beta}_{c+1} = \boldsymbol{\beta}_c + \mathbf{P}_{c+1}\mathbf{H}_{c+1}^{\mathrm{T}}(\mathbf{Y}_{c+1} - \mathbf{H}_{c+1}\boldsymbol{\beta}_c) \qquad (4)$$

Where

$$\mathbf{P}_{c+1} = \mathbf{P}_c - \mathbf{P}_c\mathbf{H}_{c+1}^{\mathrm{T}}(\mathbf{I} + \mathbf{H}_{c+1}\mathbf{P}_c\mathbf{H}_{c+1}^{\mathrm{T}})^{-1}\mathbf{H}_{c+1}\mathbf{P}_c$$

From (4), notice that $\boldsymbol{\beta}_{c+1}$ is a result of $\boldsymbol{\beta}_c$, it means that OS-ELM can incrementally learn from newly arrived samples without forgetting past samples.

C. Fully Online Sequential Extreme Learning Machine (FOS-ELM)

The OS-ELM can not be used in the case of the $\mathbf{H}_0^{\mathrm{T}}\mathbf{H}_0$ is singular. To prevent this singular problem OS-ELM requires the number of initial training samples more than the number of nodes in the hidden layer [13]. Another singularity prevention method is to add some small value $\boldsymbol{\lambda}$ to $\mathbf{P}_0 = (\mathbf{H}_0^{\mathrm{T}}\mathbf{H}_0 + \boldsymbol{\lambda}\mathbf{I})^{-1}$ [15]. In case of use without initial samples, some researchers have proposed the Fully Online Sequential Extreme Learning Machine (FOS-ELM) [13]. The FOS-ELM is the same as OS-ELM but in the initial phase, it

starts with $\boldsymbol{\beta}_0 = \mathbf{0}$, $\mathbf{P}_0 = (\lambda \mathbf{I})^{-1}$. Then, FOS-ELM can be used without the initial samples for training.

III. Proposed Method

Some parameters of FOS-ELM are randomly generated, in some cases, this can cause an under-fit problem for load forecasting. To reduce the under-fit problem, this article proposed "the re-learning method" by letting the FOS-ELM learn from each new arrival sample more than once, as the algorithm shown in Fig.2.

Algorithm: FOS-ELM with Re-learning

INPUT: X = load 24 hour ago
Y_target = load next hour
R = re-learn times

OUTPUT: Y_forecast = forecast value of load

STEP:
1: **repeat**
2: Receive X
3: Calculate Y_forecast by FOSELM
4: Wait for Y_target
5: Calculate $MAPE = \frac{|Y\_forecast - Y\_target|}{Y\_target} \times 100$
6: **for** r = 0 **to** r = R:
7: increment learn (X, Y_target)
8: r = r+1
9: **end for**
10: **until** end of data

Figure 2 FOS-ELM with Re-Learning Algorithm

In addition, FOSELM ensemble modeling was also used to improve the accuracy of load forecasting by using 10 FOS-ELM models working together as shown in Fig.3, where the forecast value is the average of the outputs from all models.

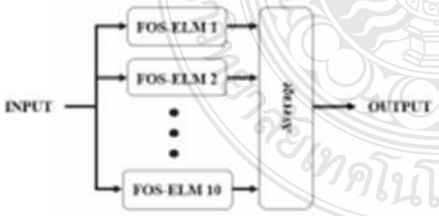


Figure 3 Ensemble FOS-ELM

IV. Experiment and Results

The experiment used a dataset from Hourly Energy Consumption from kaggle.com, which is the hourly electricity consumption data of a regional transmission organization called PJM in the USA. The dataset was divided into "target" and "input" as shown in Fig.4. Where the "target" is the hourly load that the model has to forecast and the "input" is the load 24 hours ago before the "target", that the model used as input to forecast. This experiment will forecast load 1 hour in advance. The total number of trials is 720 times, i.e., 720 hours of data were used.

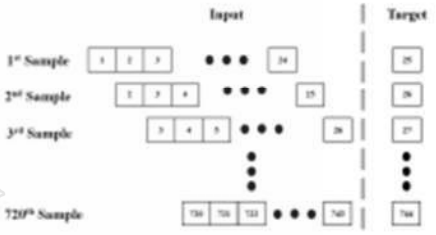


Figure 4 Division of the dataset

The experiment in this article uses ensemble FOS-ELM with 2 learning methods for comparative studies: 1) single learning from the sample data, and 2) re-learning from the sample data according to the algorithm as shown in Fig.2. The structure of the FOS-ELM is 24 nodes in the input layer, 50 nodes in the hidden layer with sigmoid as the activation function, and 1 node in the output layer. From the experiment, the mean absolute percentage error (MAPE) in the load forecasting from each model is shown in Table 1. In addition to the MAPE values, this experiment also measures the average time it took to calculate the forecast values for each input as shown in Table 1. The time is shown in Table 1 based on a computer with an Intel Core i5 processor with 8GB of memory.

TABLE 1. EXPERIMENTAL RESULT

Method		MAPE (%)	Time (ms)
Ensemble FOS-ELM	No re-learning	1.47	1.50
Ensemble FOS-ELM with Re-learning	Re-learning 1 time	1.42	3.07
	Re-learning 2 times	1.40	3.99
	Re-learning 3 times	1.39	5.31
	Re-learning 4 times	1.37	6.48
	Re-learning 5 times	1.37	7.78
	Re-learning 6 times	1.37	9.81
	Re-learning 7 times	1.37	10.41
	Re-learning 8 times	1.36	12.37
	Re-learning 9 times	1.36	13.29
	Re-learning 10 times	1.36	15.62

The MAPE of load forecasting in Table.1 can be plotted versus re-learning times as shown in Fig.5. In Fig.5 zero re-learning times equal no re-learning.

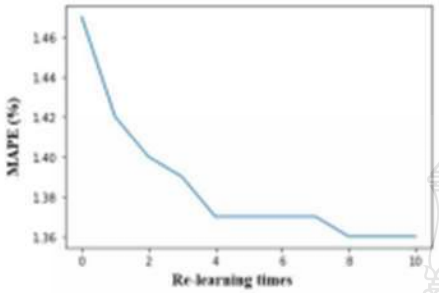


Figure 4 MAPE of load forecasting when re-learning

Some parts of the load forecast value and the actual load value are graphed in Figure 5. The results in the graph are based on the 4 times re-learning method.

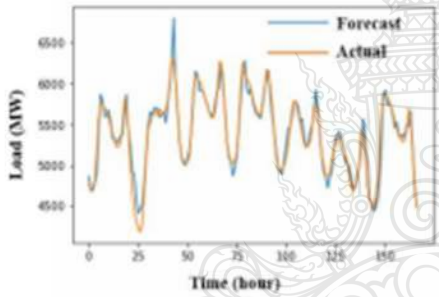


Figure 5 Some parts of the load forecast value and actual load value from the dataset

V. CONCLUSION

From the experiment, it was found that 4 times re-learning resulted in a significant reduction in load forecasting error. This is because re-learning is like adding "learning weight" to the latest sample data. The model can quickly adapt to trends in the upcoming new data. But if re-learning more than 4 times the load forecasting error will not decrease, because it is over-fitting to the latest sample too much. Re-learning can reduce load forecasting errors, but increased learning time, therefore the number of repetitions must be selected appropriately.

REFERENCES

[1] J. Contreras, R. Espinola, F. J. Nogales and A. J. Conejo, "ARIMA models to predict next-day electricity prices," in *IEEE Transactions on Power Systems*, vol. 18, no. 3, pp. 1014-1020, Aug. 2003.

[2] E. Alpaydin, Introduction to machine learning (Adaptive Computation and Machine Learning), The MIT press, Vol.5, No.8, 28, 2004.

[3] R. Adhikari, R.K. Agrawal, A homogeneous ensemble of artificial neural networks for time series forecasting, International Journal of Computer Applications, Vol.32, No.8, 1-8, 2013.

[4] X. Jin *et al.*, "Study on the driving force and challenges of developing power grid with high penetration of renewable energy,"*2017 IEEE Transportation Electrification Conference and Expo, Asia-Pacific (ITEC Asia-Pacific)*, Harbin, 2017, pp. 1-5.

[5] K. Moslehi and R. Kumar, "A Reliability Perspective of the Smart Grid," in *IEEE Transactions on Smart Grid*, vol. 1, no. 1, pp.57-64, June 2010.

[6] Neha, S. Jena and C. K. Panigrahi, "Review on Smart Power Flow Controller for Electric Vehicle Connected to Smart Grid," *2019 IEEE International Conference on Sustainable Energy Technologies and Systems (ICSETS)*, Bhubaneswar, India, 2019, pp. 93-96.

[7] Tsymbal A., "The problem of concept drift: Definitions and related work.," Technical Report. 2004, Department of Computer Science, Trinity College: Dublin, Ireland.

[8] L.Bottou, "Large-scale machine learning with stochastic gradient descent," Proceedings of the COMPSTAT 2010, Springer, pp.177–186, 2010.

[9] R.Polikar, L.Upda, S.Upda, V.Honavar, "Learn++: an incremental learning algorithm for supervised neural networks," IEEE Trans. Syst. Man Cybern.31 (4), pp.497-508, 2001

[10] A.Saffari, C.Leistner, J.Santner, M.Godec, H.Bischof, "On-line random forests," Proceedings of the 2009 IEEE Twelfth International Conference on Computer Vision Workshops, 2009

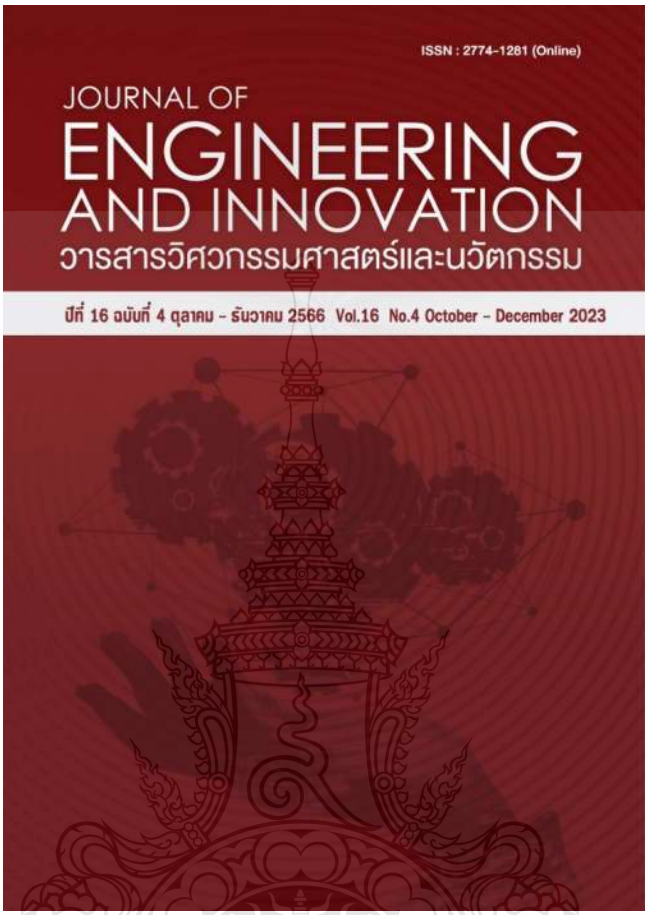
[11] G.Cauwenberghs, T.Poggio, "Incremental and decremental support vector machine learning," Proceedings of the 2001 Neural Information Processing Systems (NIPS), 2001.

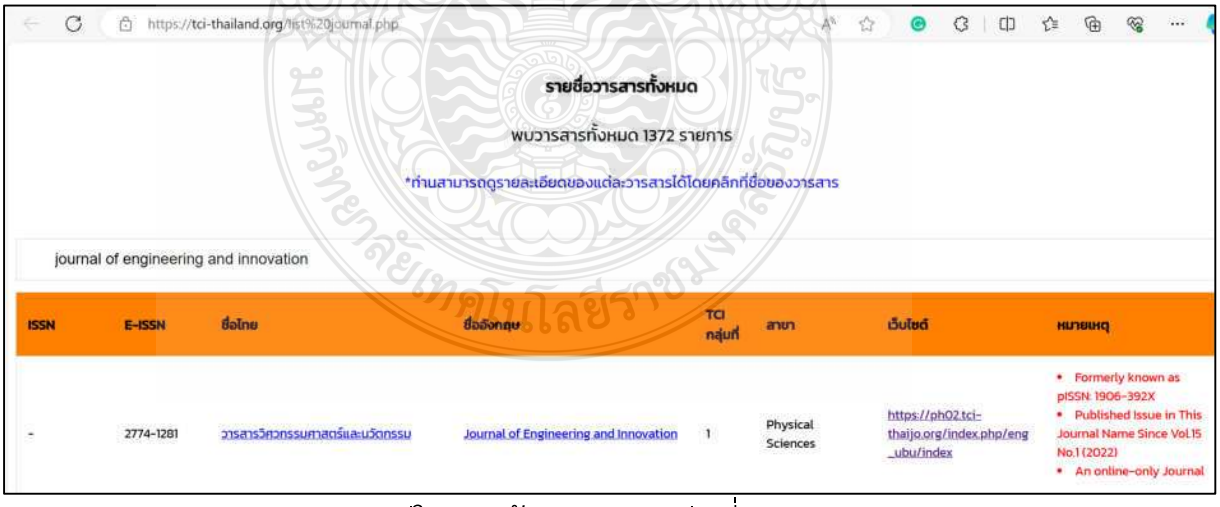
[12] Huang, G. B., Liang, N. Y., Rong, H. J., Saratchandran, P., & Sundararajan, N. (2005). On-Line Sequential Extreme Learning Machine. Computational Intelligence, 2005, 232-237.

[13] P. Wong, C. Vong, X. Gao, and K. Wong, "Adaptive control using fully online sequential-extreme learning machine and a case study on engine," in Mathematical Problems in Engineering, pp. 41-49, April 2014.

[14] G. Huang, Q. Zhu, and C. Siew, "Extreme learning machine: Theory and applications," in Neurocomputing, vol. 70, no. 1, pp. 489-501, 2006.

[15] H. T. Huynh and Y. Won, "Regularized online sequential learning algorithm for single-hidden layer feedforward neural networks," Pattern Recognition Letters, vol. 32, no. 14, pp. 1930-1935, 2011.





https://tci-thailand.org/list%20journal.php

รายชื่อวารสารทั้งหมด

พบวารสารทั้งหมด 1372 รายการ

*ท่านสามารถดูรายละเอียดของแต่ละวารสารได้โดยคลิกที่ชื่อของวารสาร

journal of engineering and innovation

ISSN	E-ISSN	ชื่อไทย	ชื่ออังกฤษ	TCI กลุ่มที่	สาขา	เว็บไซต์	หมายเหตุ
-	2774-1281	วารสารวิศวกรรมศาสตร์และนวัตกรรม	Journal of Engineering and Innovation	1	Physical Sciences	https://ph02.tci-thaijo.org/index.php/eng_ubu/index	• Formerly known as pISSN: 1906-392X • Published Issue in This Journal Name Since Vol.15 No.1 (2022) • An online-only Journal

อยู่ในฐานข้อมูล TCI กลุ่มที่ 1

วารสารวิศวกรรมศาสตร์และนวัตกรรม
Journal of Engineering and Innovation

บทความวิจัย

Synthesis of training samples for the online sequential extreme learning machine and application in load forecasting

Charnon Chupong Boonyang Plangklang*

Department of Electrical Engineering, Faculty of Engineering, Rajamangala University of Technology Thanyaburi, Pathum Thani 12110

* Corresponding author.

E-mail: boonyang.p@en.rmutt.ac.th; Telephone: 0 2549 3423

Received 18 August 2022; Revised #1 24 December 2022; Accepted 1 March 2023

Abstract

Online Sequential Extreme Learning Machine (OS-ELM) is a model capable of incremental learning from newly received samples while working, but getting start with the OS-ELM requires sufficient and appropriate sample data for initial training. In some cases, finding such sample data is not possible. To address this issue, this article presents a synthesis of sample data for the initial training of the OS-ELM model. The proposed method is to take the first sample at the time of OS-ELM initialization and then add noise to transform it to be new sufficient samples. In this article, the authors have compared different formats of the noise used in the synthesis of the sample data. It was found that the Gaussian noise with properly selected the standard deviation value gives training samples that helped the OS-ELM forecast load with the most accuracy. In addition, the use of uniform noise allows the OS-ELM to have slightly lower accuracy than Gaussian noise but can be used without worrying about selecting the appropriate standard deviation value.

Keywords

incremental learning; load forecasting; OS-ELM; training sample

1. Introduction

Sample data is very important in machine learning due to the appropriate and sufficient sample data is required to train the model. In some cases, the model has been used for a while and the environment has changed, this makes the old sample data used to train the model unsuitable for the new environment. For example, the model for load forecasting uses past sample data to train, but over time, the behavior of loads has changed dramatically, causing the sample data that was used to train the model inconsistent with the change resulting in significant error in forecasting. To solve this problem, a new set of sample data was needed to train the model again, but such an approach takes time and costs, causing practical inconvenience. The researchers developed some methods that allow the model to learn more from the newly received sample data without forgetting the already learned sample data known as "incremental learning" or "online learning".

There are some incremental models/algorithms that are frequently mentioned in research papers, such as Incremental Learning Vector Quantization (ILVQ) [1], Incremental Support Vector Machine (ISVM)

[2], Learn++ [3], Stochastic Gradient Descent (SGD) [4], Online Random Forecast (ORF) [5], and Online Sequential Extreme Learning Machine (OS-ELM) [6]. In this article, the authors focus on the OS-ELM as it is a model that can be used in both classification and regression tasks and can also learn quickly from the input with a low computational cost which can be run on low computing power devices.

The structure of OS-ELM is like an Artificial Neural Networks (ANN) with a single hidden layer, but its learning is unlike the ANN. The learning of OS-ELM does not use an iterative approach such as gradient descent, but uses random weights and calculated weights by a recursive least square method. In other words, the weights between the input layer and the hidden layer are randomly assigned and have a constant value over time and the weights between the hidden layer and the output layer use the Recursive Least Square method to calculate the value every time a new sample is received.

Although OS-ELM can incrementally learn from the newly received sample data, but getting start with OS-ELM requires a certain amount of sample data to initially train the model. The number of this initial sample data must not be less than the number of nodes in the hidden layer [6] and the number of nodes in the hidden layer also affects the model performance [7]. Therefore, it can be said that the amount of initial training sample data affects the performance of the model. This is a limitation to the use of OS-ELM if sufficient and accurate sample data cannot be obtained, e.g., load forecasting in new buildings or areas where electricity usage has never been recorded.

Several studies have proposed a solution for the problem of insufficient training samples. For example, using data pooling for individual household load forecasting [8-9], which collects datasets from other relevant sources and finds the suitable method to choose some samples for training the model. To use such an approach, one still has to collect some datasets to use for training anyway. In [10], the application of the Generative Adversarial Network (GAN) was presented to synthesize data on electric vehicle charging. GAN is a deep learning model, which is not suitable for use in systems with low computational resources. In [11], some data synthesis methods for training wind turbine power forecasting models were presented. One proposed method combines existing datasets with Gaussian distribution noise to increase the number of samples used to train the LSTM model. In [12], load forecasting was presented using the OS-ELM model. The initial training samples for the model were obtained by using the data from the first measure and added with a uniform distribution noise to increase the number of samples. In this approach, a load forecasting model can be used in a system with low computational resources and can be used online without needing any dataset.

The method presented in [12] uses uniform-distribution noise without considering other distribution forms of noise. Therefore, in this article, we present a method of sample data generation for initially training the OS-ELM by adding noise with different PDFs as shown in Fig. 1. The following contents are divided as follows: related works in part 2, methodology in part 3, experiment and results in part 4 and part 5 is discussion and conclusion.

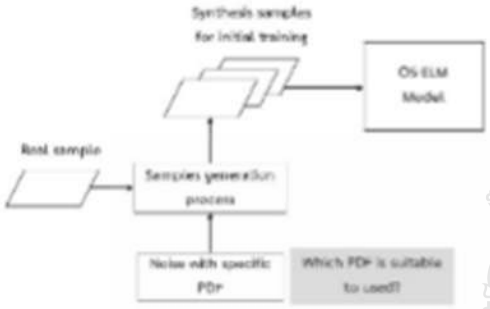


Fig.1 Main point of this article

2. Related Works

2.1 Some incremental models/algorithms

There are some incremental learning models/ algorithms often mentioned in various research studies as follows:

- Incremental Learning Vector Quantization (ILVQ) [1] is an incremental version of Learning Vector Quantization (LVQ) that uses the principle of Prototyped-based learning [13]. In the algorithm, it checks whether the received data is different from the past data that has been learned or not if significant differences are found, a new prototype will be created. This allows the model to work with changed data without forgetting the past data.
- Incremental Support Vector Machine (ISVM) [2], is like a Support Vector Machine (SVM), but some incoming data are recorded and called the Candidate Vector. The Candidate Vector may be set as Support Vector depending on the difference between the newly received data and the existing Support Vector.
- Learn++ [3] uses the ensemble model principle as Ada-boost [14], consisting of sub-models created by learning from past data. Past data where the model failed to perform is more likely to be chosen to train the sub-model. Therefore, it is said that the model can learn from mistakes to work with the changing data.
- Stochastic Gradient Descent (SGD) [4] it is an iterative method for adjusting the model parameters to optimize an objective function. Each parameter adjustment can use only a fraction of the sample data to be trained, so SGD can be applied as incremental learning.
- Online Random Forest (ORF) [5] is a Random Forecast model when it is found that the input data is different from the past data that has already been learned. The model will increase the number of trees to work with this data. Thus, ORF can work with changing data without forgetting the past data.
- Online Sequential Extreme Learning Machine (OS-ELM) [6] is an incremental version of Extreme Learning Machine (ELM) [15] which is characterized by extremely fast learning speed and low computation cost. The details of OS-LEM will be discussed in the next section.

2.2 Online Sequential Extreme Learning Machine (OS-ELM)

OS-ELM was introduced by N.Y. Liang in 2006, it is based on ELM and adds incremental learning capabilities. The structure of OS-ELM is the same as Feed-Forward Artificial Neural Networks as shown in Figure 2.

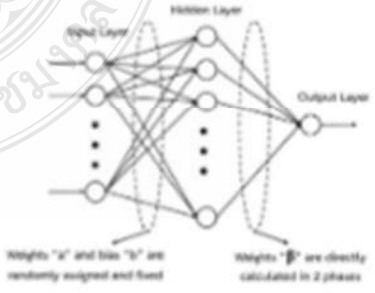


Fig.2 Structure of the OS-ELM

The weights between the input layer and the hidden layer are assigned randomly and remain constant over time. The weights between the hidden layer and the output layer are calculated in two phases as follows:

1) Initial training phase: For some N samples $(\mathbf{x}_j, \mathbf{y}_j)$ and hidden layer has L nodes. The relationship between $\mathbf{x}_j$ and $\mathbf{y}_j$ can be described as follows:

$$\mathbf{y}_j = \sum_{i=1}^{L} \boldsymbol{\beta}_i g(\mathbf{a}_i \mathbf{x}_j + b_i), \quad j = 1,2,3,\ldots N \qquad (1)$$

where $\mathbf{a}_i \in \mathbb{R}^n$ are the weights in the input layer and $b_i \in \mathbb{R}$ are the bias in the input layer, $g(\ldots): \mathbb{R} \to \mathbb{R}$ is an activation function in the hidden layer, and $\boldsymbol{\beta}_i \in \mathbb{R}^m$ are the weights in the hidden layer. Equation (1) can be simplified by writing it as follows:

$$\mathbf{Y} = \mathbf{H}\boldsymbol{\beta} \qquad (2)$$

where

$$\mathbf{H} = \begin{bmatrix} g(\mathbf{a}_1 \mathbf{x}_1 + b_1) & \cdots & g(\mathbf{a}_L \mathbf{x}_1 + b_L) \\ \vdots & \ddots & \vdots \\ g(\mathbf{a}_1 \mathbf{x}_N + b_1) & \cdots & g(\mathbf{a}_L \mathbf{x}_N + b_L) \end{bmatrix}_{N \times L}$$

$$\boldsymbol{\beta} = \begin{bmatrix} \boldsymbol{\beta}_1^T \\ \vdots \\ \boldsymbol{\beta}_L^T \end{bmatrix}_{L \times M} \quad \text{and} \quad \mathbf{Y} = \begin{bmatrix} \mathbf{y}_1^T \\ \vdots \\ \mathbf{y}_N^T \end{bmatrix}_{N \times m}$$

The initial training is to calculate the β value as in equation (3), where $\mathbf{x}$ and $\mathbf{Y}$ are known from training samples data.

$$\boldsymbol{\beta} = \mathbf{K}^{-1}\mathbf{H}^T\mathbf{Y} \quad \text{where } \mathbf{K} = \mathbf{H}^T\mathbf{H} \qquad (3)$$

If $\mathbf{x}$ and $\mathbf{Y}$ in the initial training samples have small amount or have low dimensions, the β value calculated using the normal equation as in equation (4) will be faster than using the gradient descent method [16]. Numerous studies have demonstrated that this learning method performs as well as other learning algorithms, although the input layer uses random weights and bias values [17-18].

This method may encounter the numerical instability problem when $\mathbf{H}^T\mathbf{H}$ is non-invertible. To solve this problem, two main solutions were proposed: the first solution is using the Singular Value Decomposition (SVD) method to inverse the matrix $\mathbf{H}^T\mathbf{H}$ [19]. The second solution is to use a regularization factor adding to the matrix $\mathbf{H}^T\mathbf{H}$ before inversion as shown in equation (4) [20].

$$\mathbf{K} = \mathbf{H}^T\mathbf{H} + \lambda\mathbf{I} \qquad (4)$$

where λ is a very small value called the regularization factor and $\mathbf{I}$ is the identity matrix.

2) Incremental learning phase: This is the process for adjusting the β value to be suitable to the newly received sample data as well as the previously learned data. The incremental learning phase uses the principle of the recursive least square method. The updated beta value is a function of the previous beta value recursively as in equations (5) and (6) [6].

$$\boldsymbol{\beta}_{k+1} = \boldsymbol{\beta}_k + \mathbf{K}_{k+1}^{-1}\mathbf{H}_{k+1}^T(\mathbf{Y}_{k+1} - \mathbf{H}_{k+1}\boldsymbol{\beta}_k) \qquad (5)$$

where,

$$\mathbf{K}_{k+1} = \mathbf{K}_k + \mathbf{H}_{k+1}^T\mathbf{H}_{k+1} + \lambda\mathbf{I} \qquad (6)$$

The subscription terms k and $k+1$ refer to the sample data in k^{th} order and the sample data in $(k+1)^{th}$ order, respectively. Where $k = 0$ means the value from the initial training phase.

2.3 OS-ELM without initial training sample

The use of OS-ELM requires sufficient and appropriate sample data to be used in the initial training phase. In some cases where sample data cannot be obtained, for example, load forecasting in new buildings or areas that have never collected electricity usage data, this makes the use of OSELM limited.

To solve this problem reference [12] proposed a method to use the OS-ELM without using the initial training sample data for load forecasting application. This method uses the first load profile sample obtained at startup. This sample is added with some noise value to create enough synthesis samples for initial training as shown in Fig.3.

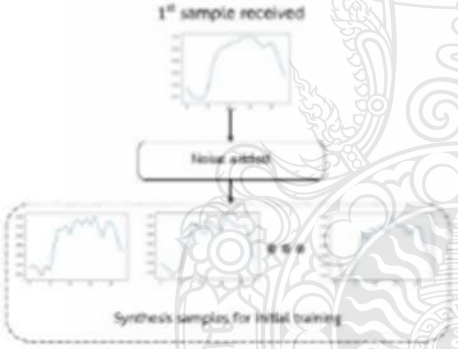


Fig.3 Synthesizing new load profile samples by adding noise value

3. Proposed Method

The synthesis load profile sample for initial training of the OS-ELM presented in section 2.3 is created by noise with a uniform probability density function (pdf). The article in section 2.3 did not experiment with other pdf of noise, such as gaussian, which is commonly used to create augmented data for training the deep learning model. Therefore, this article proposed the use of gaussian noise compared to the use of uniform noise to synthesize sample data for initial training of the OS-ELM. The algorithm for synthesizing the sample data is shown in Fig.4.

Algorithm: Synthesize data by noise adding

INPUT: *data* = 1st sample in dataset that is scaled to value between(0,1)
N = numbers of output sample required
PDF = Noise probability density function

OUTPUT: *syst_data*

STEP:
1: **for** *n*=0 **to** *n*=*N*:
2: **for** *i*=0 **to** *i*=length(*data*):
3: *rand*[*i*] = random value according to PDF
4: *syst_data*[*n*][*i*] = *data*[*i*] – (*data*[*i*]*0.1)*rand*[*i*]
5: **end for**
6: syst_data[n]=next(syst_data[n])
7: **end for**
8: **return** *syst_data*
9: **end**

Fig.4 Synthesizing data by noise adding algorithm

From the algorithm in Fig.4, it can be seen that the data used to synthesize the training samples is scaled to a value between 0 and 1 and the noise pattern is set by the user. This article uses a uniform pdf of noise with a value between 0 and 1 as in reference [12]. The synthesis process uses uniform noise compared with gaussian noise with a mean of 0 and a standard deviation (std.) of 0.1, 0.5, and 1. The noise pdfs used in this article are shown in Fig.5 and the examples of load profiles data synthesized by that noise are shown in Fig.6.

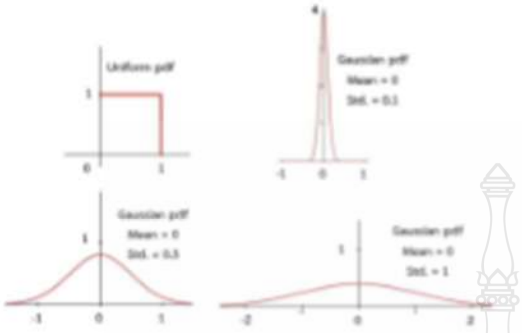


Fig.5 Probability density function (pdf) of noise used to synthesize the training samples

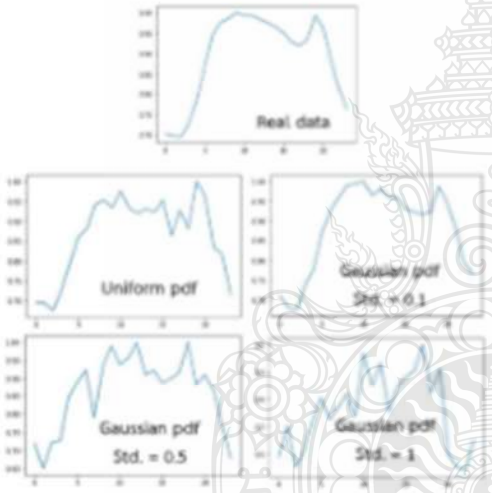


Fig.6 Example of load profile synthesized by each pdf of noise

4. Experimental

The experiment in this article used a dataset called "Hourly Energy Consumption" from Kaggle.com [21], which is an hourly load profile collected from 9 utility companies in the eastern United States. The dataset will be used as sample data for OS-ELM to learn. Each sample is divided into two parts: the input part and the target part. The target part is the hourly load and the input part is the 24-hourly load value before the target as shown in Fig.7. We chose a 24 hours time lag as input because it has a high autocorrelation value [22].

Fig.7 Input and Target from the dataset

The experimental process starts with synthesizing the sample data for initial training OS-ELM based on the algorithm in Fig.4. Then the OS-ELM forecasts the next hour's load using the previous 24 hours load as input. The forecast load values and the actual load values in the dataset are compared. The mean absolute percentage error (MAPE) is used as an indicator of OS-ELM performance. The experiment is repeated in the following sample data as shown in Fig.8. The initial training samples for OS-ELM in this experiment use noise with uniform pdf and gaussian pdf with std. equal to 0.1, 0.5, and 1. The experiment is performed on 72 samples (72 hours) to evaluate the performance of OS-ELM when startup.

This experiment uses the OS-ELM model with 24 input nodes equal to the dimension of the input data, 1 output node equal to the dimension of the output data, and a hidden layer of 50 nodes obtained from the experimental results shown in Fig 8. Fig. 8 was the result of using the OS-ELM model with different numbers of hidden layers were used to forecast the load of AEP data, and we found 50 nodes in the

hidden layer provide the appropriate accuracy and timing.

To improve the accuracy of forecasting, this experiment also uses an ensemble model of 10 OS-ELM models derived from the experimental results shown in Fig. 9. Fig. 9 was a result of using different numbers of models to forecast the load of the AEP dataset. We found that 10 models were the appropriate accurate and time- optimized results. That is, the final forecast value was derived from the mean of all 10 models of forecast values as shown in Fig. 11.

The synthesized algorithm in Fig. 4 should generate 50 sets of sample load profiles as the number of nodes in the hidden layer, that is the minimum samples for initial learning [6].

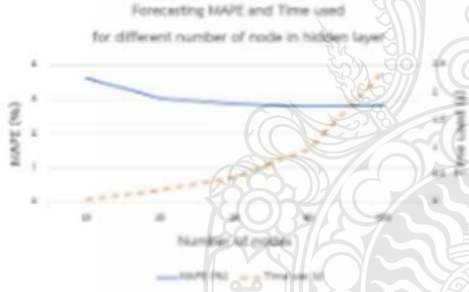


Fig. 8 MAPE and time used for different numbers of the node in a hidden layer

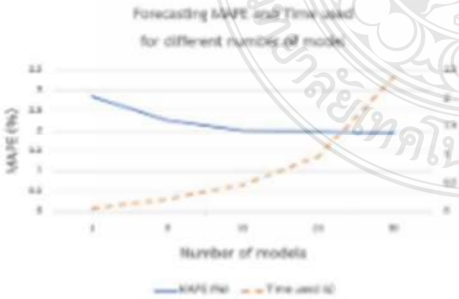


Fig. 9 Forecasting MAPE and time used for different numbers of model

Fig.10 Experiment process

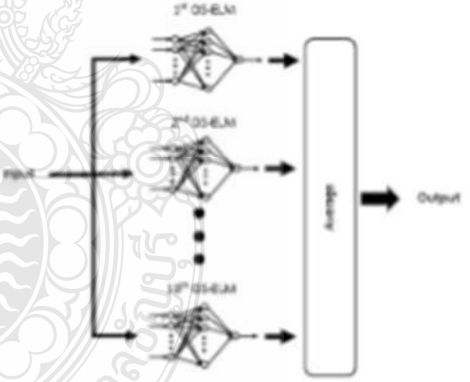


Fig.11 Ensemble of 10 OS-ELM models

5. Results and Discussion

Table 1 shows the MAPE results of load forecasting of the 9 utility datasets using different noise pdfs, and the bottom row of the table shows the average values of MAPE from all datasets.

Table 1 Experimental results

Dataset	MAPE when using each pdf of noise (%)			
	Uniform	Gaussian Std.=0.1	Gaussian Std.=0.5	Gaussian Std.=1
AEP	2.03	1.95	2.11	2.44
COMED	1.61	1.47	1.80	2.19
DAYTON	2.75	2.80	2.65	2.89
DEOK	1.99	1.83	2.21	2.52
DOM	1.99	2.07	1.95	2.56
DUQ	2.55	3.10	2.44	2.44
EKPC	3.67	4.57	3.15	2.93
PE	2.31	2.67	2.58	2.57
NI	1.65	1.70	1.61	1.92
Average	2.28	2.44	2.25	2.47

From Table 1, if considering each dataset, the most accurate is gaussian pdf noise with a standard deviation equal to 0.5 gives the most accurate forecast for the four datasets: DAYTON, DOM, DUQ, and NI. The second most accurate is gaussian pdf noise with a standard deviation equal to 0.1 which yielded the most accurate forecast for 3 datasets: AEP, COMED, and DEOK. Suppose the average MAPE of all 9 datasets is considered, the gaussian pdf noise with std. equal to 0.5 still the most accurate with an average MAPE of 2.25%, followed by the uniform pdf noise with an average MAPE of 2.28%.

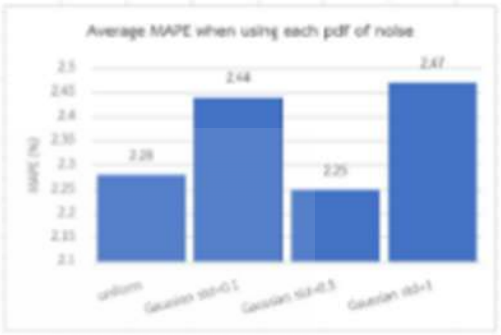


Fig.12 Average MAPE for each pdf of noise

Figure 13-16 shows some comparison graphs of forecasted load values and actual load values in the AEP data set when using different PDFs of noise.

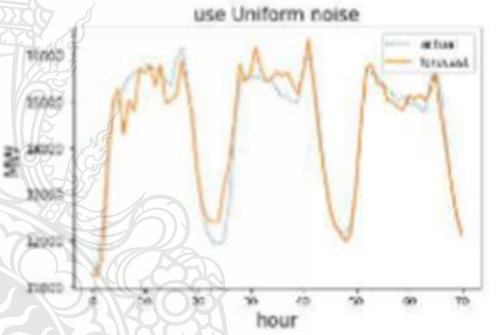


Fig.13 Forecasted load and actual load when using uniform noise

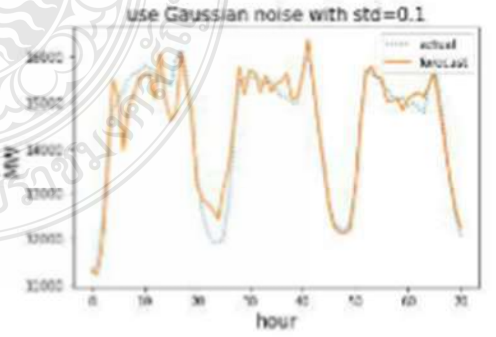


Fig.14 Forecasted load and actual load when using Gaussian noise with std = 0.1

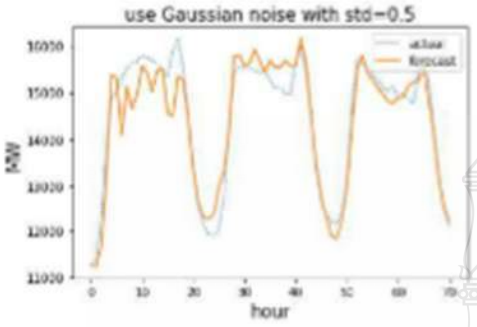


Fig.15 Forecasted load and actual load when using Gaussian noise with std = 0.5

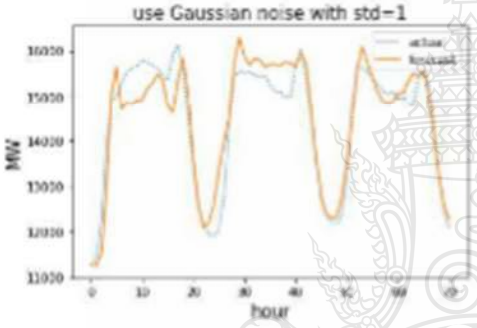


Fig.16 Forecasted load and actual load when using Gaussian noise with std = 1

This experiment found that using gaussian pdf noise to synthesize the initial training dataset for the OS-ELM would allow the most forecasting accuracy. To achieve the most forecasting accuracy, an appropriate standard deviation must be selected. In this case, the standard deviation of 0.5 is selected meaning there is a 95% probability that the noise will range between -1 and 1. In this experiment the load profile data is scaled to a range of 0 to 1, then Gaussian pdf noise with a standard deviation of 0.5 is appropriate to synthesize the training data.

The use of uniform pdf noise to synthesize the initial training dataset resulted in slightly low forecasting accuracy than using a Gaussian pdf noise with 0.5 standard deviations. However, the use of the uniform pdf noise still had higher forecasting accuracy than the gaussian pdf noise with standard deviations of 0.1 and 1. The reasons are as follows: Gaussian pdf noise with the standard deviation of 0.1 has low dispersion, making the synthesized sample data look similar, causing the model to have over-fit problems. Gaussian pdf noise with the standard deviation of 0.5 has high dispersed making synthesized sample data look different that it can't represent the real data, causing the model to have under-fit problems.

6. Conclusion

In conclusion, choosing a noise to create a dataset for the initial training of the OS-ELM requires an appropriate pdf pattern. In this experiment, the gaussian pdf noise with a standard deviation of 0.5 and the uniform pdf noise are the appropriate pdf pattern. If narrowly dispersed noise is selected, the synthesis samples will be very similar, causing the model to have an over-fit problem. If wide dispersion noise is selected, the synthesis sample will differ too much from the actual data, causing the model to have an under-fit problem.

Reference

[1] Sato A, Yamada K. Generalized learning vector quantization. In: *Proceedings of the 1995 Neural*

Information Processing Systems. Denver, Colorado, USA; 1995. p.423-429.

[2] Cauwenberghs G, Poggio T. Incremental and decremental support vector machine learning. *Advance Neural Information Processing Systems*. 2001(13): 388-394.

[3] Polikar R, Upda L, Upda S, Honavar V. Learn++: an incremental learning algorithm for supervised neural networks. *IEEE Transactions on Systems, Man, and Cybernetics: Systems*. 2001;31(4): 497-508.

[4] Zhang T. Solving large scale linear prediction problems using stochastic gradient descent algorithms. In: *Proceedings of the Twenty-First International Conference on Machine Learning*. Banff, Alberta, Canada; 2004. p.116-123.

[5] Saffari A, Leistner C, Santner J, Godec M, Bischof H. On-line random forests. In: *Proceedings of the 2009 IEEE Twelfth International Conference on Computer Vision Workshops*. Kyoto, Japan; 2009. p.1393-1400.

[6] Liang N, Huang G, Saratchandran P, Sundararajan N. A fast and accurate online sequential learning algorithm for feedforward networks. *IEEE Transaction on Neural Networks*. 2006;17(6): 1411-1423.

[7] Liu Y, Cao W, Liu Y, Zou W. A Novel Ensemble Learning Method for Online Learning Scenarios. In: *2021 IEEE 4th International Conference on Electronics Technology (ICET)*. Chengdu, China; 2012. p. 1137-1140.

[8] Shi H, Xu M, Li R. Deep Learning for Household Load Forecasting - A Novel Pooling Deep RNN. *IEEE Transactions on Smart Grid*. 2018;9(5): 5271-5280.

[9] Yang E, Youn C. Temporal Data Pooling with Meta-Initialization for Individual Short-Term Load Forecasting. *IEEE Transactions on Smart Grid*. 2022;14(4): 3246-3258.

[10] Xiaodong S, Houxiang Z, Yue X, Peng L, Junyong L. Short- term electric vehicles charging load forecasting based on deep learning in low-quality data environments. *Electric Power Systems Research*. 2022;212: 1-14.

[11] Hao C, Yngve B, Qixia Z. Data- augmented sequential deep learning for wind power forecasting. *Energy Conversion and Management*. 2021;258: 1-12.

[12] Charnon C, Boonyang P. Incremental learning model for load forecasting without training sample. *CMC- Computers Materials & Continua*. 2022;72(3): 5415-5427.

[13] Schwabacher M, Hirsh H, Ellman T. Learning prototype- selection rules for case- based iterative design. In: *Proceedings of the Tenth Conference on Artificial Intelligence for Applications*. San Antonio, TX, USA;1994. p. 56-62.

[14] Freund Y, Schapire R. A decision- theoretic generalization of on- line learning and an application to boosting. *Journal of Computer and System Sciences*. 1997;55(1): 119-139

[15] Huang G, Zhu Q, Siew C. Extreme learning machine: a new learning scheme of feedforward neural networks. In: *2004 IEEE International Joint Conference on Neural Networks*. Budapest;2004. p.985-990, 2004.

[16] Andrew ng. Machine learning specialization. Available from: https/ / coursera. org/ specializations/ machine- learning- introduction [Accessed 24th December 2022].

[17] Liu X, Lin S, Fang J, Xu Z. Is extreme learning machine feasible? a theoretical assessment (part I). *IEEE Transactions on Neural Networks and Learning Systems*. 2015;26(1): 7-20.

[18] Lin S, Liu X, Fang J, Xu Z. Is extreme learning machine feasible? A theoretical assessment (Part II). *IEEE Trans. Neural Networks Learning System*. 2015;26(1): 21-34.

[19] Klema V, Laub A. The singular value decomposition: its computation and some applications. *IEEE Transactions on Automatic Control*. 1980;25(2): 164-176.

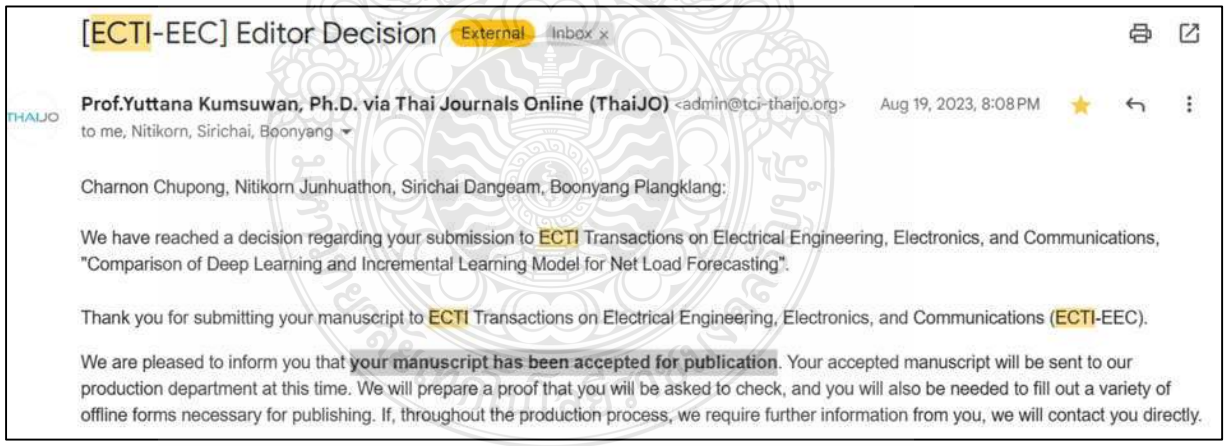
[20] Deng W, Zheng Q, Chen L. Regularized extreme learning machine. In: *2009 IEEE Symposium on Computational Intelligence and Data Mining*. Nashville, TN, USA; 2009. p. 389-395.

[21] R Mulla. Hourly Energy Consumption. Available from: https://kaggle.com/ robikscube/hourly-energy-consumption [Accessed 24th December 2022].

[22] Surakhi O, Zaidan M, Fung L, Hussein N, Serhan S, Alkhanafseh M, Ghoniem M, Hussein T. Time-Lag Selection for Time-Series Forecasting using Neural Network and Heuristic Algorithm. *Electronics*. 2021; 10(20): Available


ECTI
Association
The ECTI Transactions on Electical
Engineering, Electronics & Communications


Scopus
Search
Lists
Sources
SciVal
Create account
Sources
ISSN
Enter ISSN or ISSNs
Find sources
Title: ECTI Transactions On Electrical Engineering × Electronics × And Communications ×
ISSN: 1685-9545 ×
Filter refine list
Apply
Clear filters
Display options
Display only Open Access journals
Counts for 4-year timeframe
No minimum selected
Minimum citations
1 result
Download Scopus Source List
Learn more about Scopus Source List
View metrics for year: 2022
Source title
CiteScore
Highest percentile
Citations 2019-22
Documents 2019-22
% Cited
1
ECTI Transactions on Electrical Engineering, Electronics, and Communications
1.5
25%
547/738
Electrical and Electronic Engineering
169
116
47


[ECTI-EEC] Editor Decision
External
Inbox ×
Prof.Yuttana Kumsuwan, Ph.D. via Thai Journals Online (ThaiJO) <admin@tci-thaijo.org>
Aug 19, 2023, 8:08 PM
to me, Nitikorn, Sirichai, Boonyang
Charnon Chupong, Nitikorn Junhuathon, Sirichai Dangeam, Boonyang Plangklang:
We have reached a decision regarding your submission to ECTI Transactions on Electrical Engineering, Electronics, and Communications, "Comparison of Deep Learning and Incremental Learning Model for Net Load Forecasting".
Thank you for submitting your manuscript to ECTI Transactions on Electrical Engineering, Electronics, and Communications (ECTI-EEC).
We are pleased to inform you that your manuscript has been accepted for publication. Your accepted manuscript will be sent to our production department at this time. We will prepare a proof that you will be asked to check, and you will also be needed to fill out a variety of offline forms necessary for publishing. If, throughout the production process, we require further information from you, we will contact you directly.

Comparison of Deep Learning and Incremental Learning Model for Net Load Forecasting

Charnon Chupong, Nitikorn Junhuathon, Sirichai Dangeam and Boonyang Plangklang*

Department of Electrical Engineering, Faculty of Engineering, Rajamangala University of Technology Thanyaburi (RMUTT), Pathum Thani, Thailand,
*Corresponding E-mail: boonyang.p@en.rmutt.ac.th

ABSTRACT

This paper presents hourly net load forecasting, which is the forecasting of the difference between the hourly power demand and the hourly power profile of the Photovoltaic (PV) system, which is the load that the utility should supply to the consumer. Three forecasting models are compared. The first model represents Long Short-Term Memory (LSTM), which is based on a deep learning model. The second model is the Fully Online Sequential Extreme Learning Machine (FOS-ELM), which is an incremental learning model that does not require initial training data. Online Sequential Extreme Learning Machine (OS-ELM) is the third model that can be learned incrementally like FOS-ELM. In addition, a method for the initial training of the OS-ELM model was proposed by taking the first sample from the studies to synthesize a sufficient number of samples for the initial training of the OS-ELM model. It was found from the experiment that in the case of fixed PV penetration rate, the LSTM model had slightly lower of error in forecasting than the other two models. In the case of increasing PV penetration rate, the FOS-ELM, and OS-ELM models had significantly lower errors in forecasting than the LSTM model. When comparing only the OS-ELM model using the proposed method with the FOS-ELM model, it was found that the OS-ELM model gave lower errors in forecasting than the FOS-ELM model because it was initially trained by the synthetic sample properly.

Keywords: Net Load Forecasting, Incremental Learning, Online Sequential Extreme Learning Machine, Data Augmentation

1. INTRODUCTION

In the past decade, the world has paid great attention to the climate change problem and one of the most popular solutions is to use electricity generated from photovoltaic (PV) systems. Global PV increased from 584 GW in the year 2019 to 843 GW in the year 2022 [1] or a 44% increase over three years. The increase in PV systems, especially those with high penetration rates, will cause instability problems in the power system [2]. The high penetration rate of PV results in changing of the load profile as shown in Fig.1. The changing of the load profile from the utility point of view causes errors in the conventional load forecasting method. Error in load forecasting can cause some mistakes in power system planning and operation that can affect power system reliability [3]. From Fig. 1, the PV power supplies the load for some periods while the rest of the load is supplied by the utility called net load profile or "net-load". To reduce the impact of PV penetration on the power system, one popular approach is PV power forecasting [4]. PV power forecasting is used in conjunction with load forecasting as net load forecasting. There are two main approaches for net load forecasting [5]: The first approach is separate forecasting, which uses 2 models separately for PV power forecast and load forecast and calculates the net load as different between load and PV power. The second approach is direct forecasting, which uses only 1 model to forecast net load directly. Subsequently, research has shown that direct net load forecasting is more accurate than separate forecasting [6].

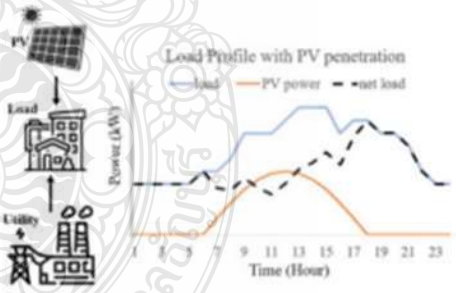


Fig.1: *Load Profile with PV Penetration*

There is some net load forecasting article presented, for example in [7], which compares net load forecasting with 7 machine learning models: Artificial Neural Network (RNN), Extreme Gradient Boosting (XGBoost), k-Nearest Neighbors (KNN), Random Forest (RF), Recurrent Neural Network (RNN), Support Vector Regression (SVR) and Naïve Persistence Models (NPM). The experimental results show that the RF model has the highest forecasting accuracy due to the ensemble structure and the second most accurate model is the RNN model,

which works well with sequential data. There are also articles using deep learning models for load forecasting, including Long Short-Term Memory (LTSM) [8] and Gate Recurrent Unit (GRU) [9], which are models that can work with time series data better than ordinary machine learning models. All of the aforementioned models must use historical data to train the model. If there are changes in the electrical system, such as the increase of PV systems and electric vehicle chargers, which cause electricity consumption patterns to differ from the historical data used in model training, some errors will occur in forecasting. As in [10], the net load forecasting is presented by using the Adaptive Neuro Fuzzy Inference System (ANFIS) model and comparing the forecasting error when the PV system penetration ratio is increased. It was found that the error in forecasting is higher when the penetration rate is changed, but the article does not present some solution.

To address this problem, the researchers proposed an incremental learning model, a model that can learn from the newly received data and adjust itself during operation, as shown in Fig. 2. Examples of incremental models such as Incremental Support Vector Machine (ISVM) [11], Online Random Forecast (ORF) [12], Incremental Learning Vector Quantization (ILVQ) [13], Learn++ [14], Stochastic Gradient Descent (SGD) [15], and Online Sequential Extreme Learning Machine (OS-ELM) [16]. This research focuses on the OS-ELM model, because it has a low computational effort, then it can be used on a low-resource device such as microcontrollers in ordinary smart meters.

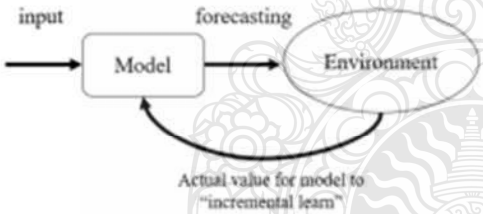


Fig.2: *Concept of Incremental Learning Model*

The OS-ELM model is a single hidden layer feed-forward neural network that does not use iterative methods like gradient descent to adjust model parameters. This allows the OS-ELM model for fast learning, low computational cost, and high accuracy for uncomplicated tasks [17]. In reference [18], researchers use OS-ELM to forecasts PV power compared to ELM which uses empirical mode decomposition (EMD) acting as feature extraction. The result shows that the OS-ELM has higher accuracy than the ELM with EDM. In reference [19], the researchers present load forecasting by using k-MEAN clustering to group load profiles and use each OS-ELM to forecasting each group of load profiles. The result shows that the grouping of data before sending it to each OS-ELM to perform is better than using OS-ELM alone. However, the OS-ELM has one disadvantage, a sufficient amount of sample data is required for initial training [20], which makes the OS-ELM unusable in some tasks if historical electricity usage or PV power data is not collected. To address this problem, researchers have proposed a Fully Online Sequential Extreme Learning Machine (FOS-ELM) model that does not require an initial training sample [20]. But the FOS-ELM has a high error at the beginning period of forecasting because it uses an initial sample with zero value. This paper presents another approach to implementing the OS-ELM model without using an initial training sample inspired by the data augmentation method [21] and compared it with FOS-ELM and LSTM in net load forecasting tasks with varying penetration rates of PV.

The main contributions of this paper are: 1) presents a method to use the OS-ELM model without the initial training sample, 2) Comparison of net load forecasting (directed approach) accuracy between the proposed methods and the FOS-ELM and LSTM models in various PV penetration rate conditions.

2. RELATED WORKS

2.1 Incremental Learning Algorithms

The incremental learning algorithm is a method that allows a model to adjust parameters based on newly received sample data without discarding what has been learned from the past sample data. The following incremental learning algorithms are frequently discussed in numerous research studies:

- Online Random Forest (ORF) [12] is a Random Forecast algorithm when the input data differs from the previously learned past data. The 'trees' of the model will be increased to work with this data. Thus, ORF can work with changing data without forgetting the past data.
- Stochastic Gradient Descent (SGD) [15] is an algorithm in which the model parameters are changed iteratively to maximize an objective function. SGD is incremental learning because the model can adjust the parameters every time it receives new sample data to learn.
- Learn++ [14] utilizes the ensemble model approach like Ada-boost [22]. It is made up of sub-models developed through learning from past data. It is more likely that the past data where the model underperformed will be used to train the sub-model. As a result, it is claimed that the model can adapt to changing input data.
- Incremental Support Vector Machine (ISVM) [11] is like a Support Vector Machine (SVM), but some incoming data are recorded and called the Candidate Vector. Depending on the discrepancy between the newly received data and the current Support Vector, the Candidate Vector may be promoted to Support Vector.

- Incremental Learning Vector Quantization (ILVQ) [13] is an incremental version of Learning Vector Quantization (LVQ) that applied the Prototyped-based learning concept [23]. A new prototype will be created if there are significant discrepancies between the received data and the previously learned data. This enables the model to operate on newly received data without losing track of learned data.
- Online Sequential Extreme Learning Machine (OS-ELM) [16] is an incremental version of Extreme Learning Machine (ELM) [24] It has a very high learning speed and a low cost of computing suitable for implementation in edge devices. The details of OS-LEM will be discussed in the next section.

2.2 Online Sequential Extreme Learning Machine (OS-ELM)

OS-ELM is an incremental learning algorithm, which was proposed by N.Y. Liang [16]. It is an incremental learning version of the Extreme Learning Machine [24]. It is applied to a single hidden layer feed-forward neural network as shown in Fig. 3.

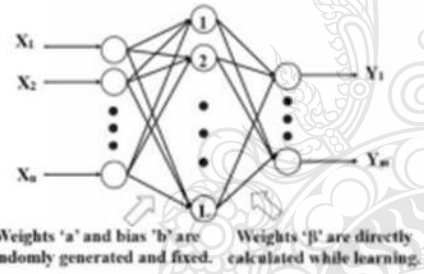


Fig.3: Structure of OS-ELM

The input layer weights and biases are randomly generated and don't change over time. The output weights are directly calculated not using an iterative approach as the gradient descent with 2 steps as follows:

1) Initial training phase: For a model with L nodes in a hidden layer and some N training samples (x_j, y_j). The connection between a and b is explained as follows:

$$\mathbf{y_j} = \sum_{i=1}^{L} \boldsymbol{\beta}_i \, g(\mathbf{a_i x_j} + \mathbf{b_i}), \quad j = 1,2,3,\dots N \qquad (1)$$

where $g(\dots): \mathbb{R} \to \mathbb{R}$ is the hidden layer activation function $\boldsymbol{\beta}$ is the hidden layer weights $\boldsymbol{a}$ and $\boldsymbol{b}$ is the input layer weights and bias respectively. Equation (1) can be written more simply as follows:

$$\mathbf{Y} = \mathbf{H}\boldsymbol{\beta} \qquad (2)$$

where

$$\mathbf{H} = \begin{bmatrix} \mathbf{g(a_1 x_1 + b_1)} & \cdots & \mathbf{g(a_L x_1 + b_L)} \\ \vdots & \ddots & \vdots \\ \mathbf{g(a_1 x_N + b_1)} & \cdots & \mathbf{g(a_L x_N + b_L)} \end{bmatrix}_{N \times L}$$

$$\boldsymbol{\beta} = \begin{bmatrix} \boldsymbol{\beta}_1^T \\ \vdots \\ \boldsymbol{\beta}_L^T \end{bmatrix}_{L \times m} \quad \text{and} \quad \mathbf{Y} = \begin{bmatrix} \mathbf{y}_1^T \\ \vdots \\ \mathbf{y}_N^T \end{bmatrix}_{N \times m}$$

The purpose of the initial training phase is to determine the value of β by using equation (3), where the x and y values are known from the initial training sample.

$$\boldsymbol{\beta} = \mathbf{K}^{-1}\mathbf{H}^T\mathbf{Y} \quad \text{where } \mathbf{K} = \mathbf{H}^T\mathbf{H} \qquad (3)$$

When the initial training samples are tiny in size or have low dimensions, the gradient descent approach will take longer to determine the value than the normal equation (4) [25]. Despite the input layer using random weights and bias values, numerous studies have shown that this learning algorithm can perform as well but faster than other learning algorithms [26-27].

When $\mathbf{H}^T\mathbf{H}$ is not invertible, this algorithm will encounter the numerical instability issue. There are at least two methods to solving this problem, the first is to invert the $\mathbf{H}^T\mathbf{H}$ using the Singular Value Decomposition (SVD) method [28]. The second method is adding a regularization factor λ to the $\mathbf{H}^T\mathbf{H}$ before inversion, as shown in equation (4) [29].

$$\mathbf{K} = \mathbf{H}^T\mathbf{H} + \lambda\mathbf{I} \qquad (4)$$

where λ is a very small value called the regularization factor and $\boldsymbol{I}$ is the identity matrix.

2) Incremental learning phase: the purpose of this phase is to adjust the $\boldsymbol{\beta}$ value according to the newly received sample without forgetting the past learned sample. The recursive concept is applied to equations (3) and (4). In the initial phase, the OS-ELM calculates the $\boldsymbol{\beta}_0 = \boldsymbol{K}_0^{-1}\boldsymbol{H}_0^T\boldsymbol{Y}_0$ where $\boldsymbol{K}_0 = \boldsymbol{H}_0^T\boldsymbol{H}_0$ which subscript 0 means round 0 of learning or the initial training. When new sample(s) data has arrived causing the value of the matrix $\boldsymbol{H}$ and $\boldsymbol{Y}$ change, the OS-ELM adjusts the $\boldsymbol{\beta}$ as follows:

$$\boldsymbol{\beta}_{k+1} = \boldsymbol{\beta}_k + \mathbf{K}_{k+1}^{-1}\mathbf{H}_{k+1}^T(\mathbf{Y}_{k+1} - \mathbf{H}_{k+1}\boldsymbol{\beta}_k)$$

$$\text{where } \mathbf{K}_{k+1} = \mathbf{K}_k + \mathbf{H}_{k+1}^T\mathbf{H}_{k+1} \qquad (5)$$

The updated $\boldsymbol{\beta}$ values can be calculated by equation (5), where it is observed that the updated $\boldsymbol{\beta}$ values are a function of the previous $\boldsymbol{\beta}$ values, which means the model

can learn and adjust its parameters from a newly received sample without forgetting the past learned sample. The subscription term k means sample in k^{th} order and the subscription term $k+1$ means sample in $(k+1)^{th}$ order. Where k=0 means the value from the initial training phase.

2.3 Fully Online Sequential Extreme Learning Machine (FOS-ELM)

The application of OS-ELM is constrained in some situations where initial sample data cannot be obtained, such as load forecasting in brand-new structures or regions that have never gathered electricity usage data. To solve this problem, FOS-ELM was introduced in reference [20] by setting $\boldsymbol{\beta_0} = 0$ and $\mathbf{K_0} = 0$, in other words, it is initial training by a sample that is equal to zero.

3. PROPOSED METHOD

Using OS-ELM without the initial training samples proposed here was inspired by data augmentation in the deep learning field [21]. This method starts by receiving the first sample from actual use (input and output to be predicted) and synthesizing additional samples by adding noise to received samples as shown in Fig. 4.

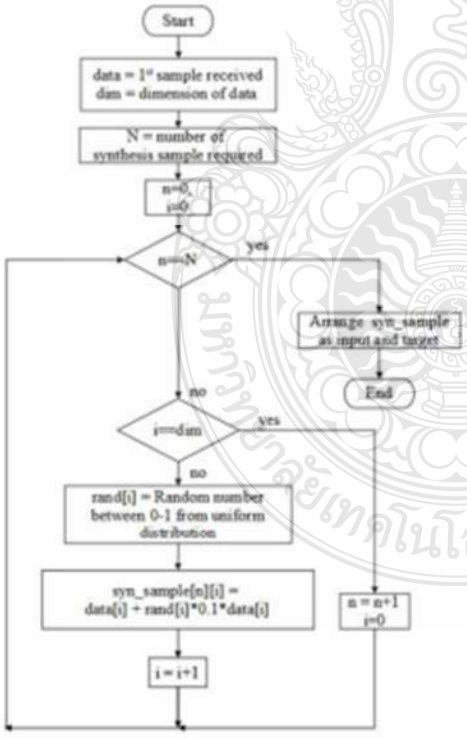


Fig.4: Flowchart of the Proposed Method

The flowchart in Fig.4 can be described as follow:

1) Receiving a sample from an actual working area and setting the dimensions of the desired synthetic samples. In this case, the dimension is set to 25 (24 input and one target value).

2) Setting the number of synthetic samples required for initial training of the OS-ELM model. In this case, 50 samples are required, which equals the number of nodes in the hidden layer of the OS-ELM model.

3) Randomizing a number between 0 and 1 using the uniform probability density function so that all numbers have an equal likelihood to occur.

4) Creating a new sample data by adding a random value obtained from step 3 to the sample data obtained from step 1. Before adding, such random values must be adjusted to not exceed 10% of the sample data. So that the new sample data is not too different from the original until the model predicts values incorrectly.

5) Repeating steps 3 and 4 until all dimensions of all synthetic samples are obtained.

6) Splitting the samples obtained from step 5 into input and target for initial training of the OS-ELM model.

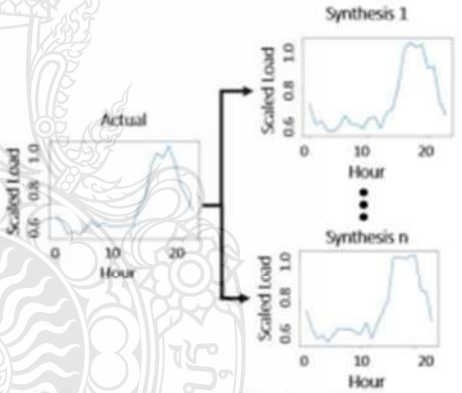


Fig.5: Example of Actual and Synthetic Samples

Fig. 5 shows some parts of the synthesized sample, it is found that the synthesized samples are different from the real sample but have the same pattern.

4. METHODOLOGY

The experiment in this article compares the net load forecasting of the OS-ELM model which does not require data in the initial training method presented in the previous topic with the FOS-ELM, which does not require the initial training sample, and the LSTM model, a deep learning model without incremental learning. In this experiment, the dataset used is described in "Solar Generation and Demand Italy 2015-2016" [30], which presents the hourly electricity consumption and generation of PV systems in

Italy in the period 2015-2016. The three models mentioned above were used to forecast the net hourly load for 2016. Data from 2015 was used to train the LSTM model, while OS-ELM and FOS-ELM were trained from 2016 data. In addition, this article also divides the experiment into 3 scenarios:

1) baseline penetration, in which the penetration rate is constant at about 20% based on the dataset,

2) low growth penetration, in which the penetration rate increases from the dataset by 5% every quarter from the 2nd quarter, and

3) high growth penetration is the scenario where the penetration rate increases from the data set by 10% every quarter from the 2nd quarter. The penetration rate is defined as in (6). In scenarios 2 and 3, we modified the PV power output in the 2016 dataset according to the penetration rate described above. The experimental methods and datasets used are summarized in Table 1.

$$Penetration\ rate = \frac{Peak\ PV\ Power\ (kW)}{Peak\ Load\ (kW)} \quad (6)$$

Table 1: *Dataset Used in Each Scenario*

Scenario /Model	LSTM	FOS-ELM	OS-ELM with the proposed method
Baseline penetration (Original dataset)	Train by the 2015 and test on the 2016 dataset	Test on the 2016 dataset	
Low growth penetration	Train by the 2015 and test on the 2016 dataset with some modification	Test on the 2016 dataset with some modification	
High growth penetration			

To forecast the hourly net load, the hourly net loads of the last 24 hours were used as input. Therefore, the data in the dataset is provided as input and target samples for training the model. The target value in each sample will be the net load of 1 hour and the input value for each sample will be the hourly net load 24 hours before the target value as shown in Figure 6.

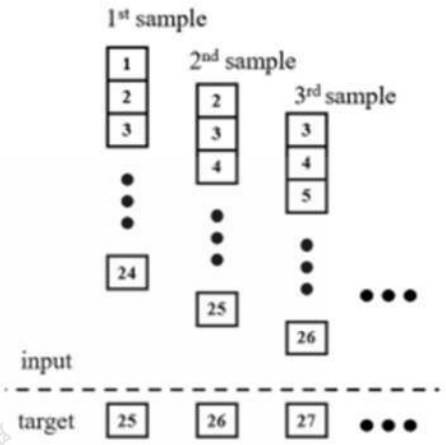


Fig.6: *Dataset Provided as Input and Target Samples*

The structure of the OS-ELM and FOS-ELM in this experiment are ensemble models consisting of sub-models with 24 nodes for the input layer and 1 node for the output layer. For the number of nodes in the hidden layer, 10, 20, 50, and 100 nodes were tested, and 10, 20, 50, and 100 sub-models were tested in the ensemble model. Therefore, a sub-model with 50 nodes in the hidden layer and 10 sub-models in the ensemble model was selected as Fig. 7, which is a compact structure with high accuracy. The OS-ELM and FOS-ELM models were implemented in Python language and run on the Spyder IDE. The LSTM model has been implemented in the *TensorFlow* platform by experimenting with the number of units 1, 5, 10, 50, and 100. The output from the LSTM passes through the 1 node in Dense Layer. The number of units was set at 50, which is a very accurate structure (see Fig. 8).

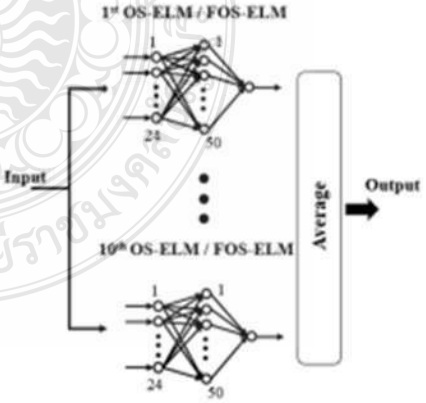


Fig.7: *Ensemble of OS-ELM/ FOS-ELM in This Experiment*

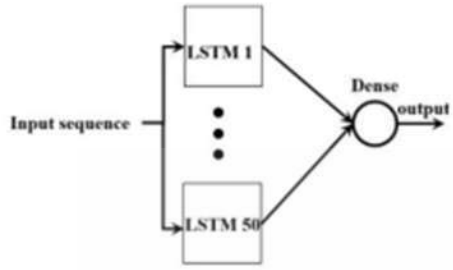


Fig.8: *LSTM in This Experiment*

5. RESULT AND DISCUSSION

The experiments described in the previous section were performed using the Mean Absolute Percentage Error (MAPE) as the performance index of each model for net load forecasting. The MAPE can be calculated by equation (7). The MAPE was measured for the whole year and quarterly in the dataset to observe the trend of the model incrementally learning and adapting itself to lower MAPE values.

$$\boldsymbol{MAPE} = \frac{1}{n}\sum_{i=1}^{n}\frac{|A_i - F_i|}{A_i} \qquad (7)$$

where:

A_i is the actual value
F_i is the forecast value
n is the total amount of data.

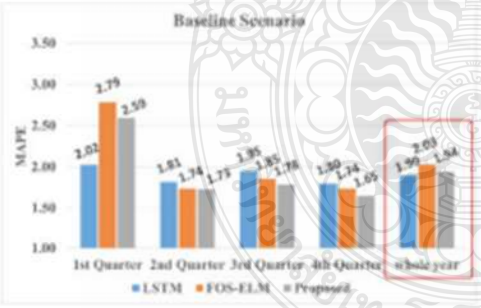


Fig.9: *Result of Baseline Scenario*

Fig. 9 shows the MAPE values of the net load forecasting in a constant penetration rate scenario. It can be seen that the MAPE of the LSTM model is nearly equal across all quarters because the model was trained from the last whole year's samples. The FOS-ELM model has a high MAPE in the 1st quarter because the model works without samples for initial training causing overfit problem. In the 2nd to 4th quarter, the FOS-ELM model has a lower MAPE than the LSTM model due to the ability of incremental learning from the newly receive samples that are closer to the real situation than the training samples in the LSTM model. When considering the whole year, FOS-ELM still has a higher MAPE than LSTM due to the high MAPE in 1st quarter. The OS-ELM model with the initial training method as proposed has MAPE similar to the FOS-ELM model i.e., high in the 1st quarter and decreasing until lower than the LSTM model in the 2nd to 4th quarters, but MAPE for the whole year is still higher than the LSTM model. However, when comparing the FOS-ELM model and the OS-ELM model, it found that the MAPE value of the OS-ELM model is lower than the FOS-ELM model in every quarter. This is because the OS-ELM model used a sufficient number of synthetic samples for the initial training which can reduce the overfit problems.

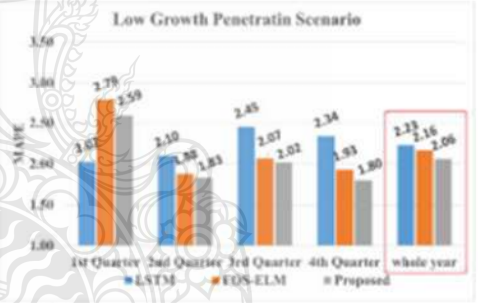


Fig.10: *Result of Low Growth Penetration Scenario*

Fig. 10 shows the MAPE values of the net load forecast in the low growth penetration scenario where the penetration rate increases by 5% every quarter (starting from the 2nd quarter). It can be seen that the LSTM model has significantly increased the MAPE in the 3rd to 4th quarter when the penetration rate increases by 10-15%. In other words, If the data changes by 10% or more, the LSTM model is highly inaccurate. This is because the training samples are different from the actual data that the model has to forecast. The FOS-ELM model has a high MAPE in the 1st quarter due to insufficient training samples. But in the 2nd to 4th quarters, the MAPE value will drop to significantly less than the LSTM model because the FOS-ELM model can adapt to situations where the environment changes. The OS-ELM model using the proposed initial training method has MAPE significantly lower than the LSTM model in the 2nd to 4th quarter,

similar to the FOS-ELM model. By comparing to the FOS-ELM model, the OS-ELM model has lower MAPE values in every quarter because the initial training by enough synthetic samples allows the model to adapt faster and reduce overfit issues. In addition, from Figure 10, it can be noticed that the MAPE of FOS-ELM and OS-ELM in the 3rd quarter is higher than in the 2nd quarter due to the summer season when PV systems have higher power output. This resulted in a greater change in net load, causing the model to have a high MAPE before adjusting to a lower MAPE in the following quarter.

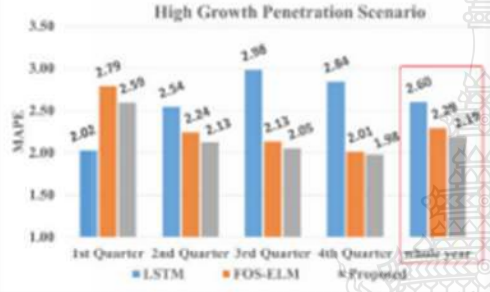


Fig.11: *Result of High Growth Penetration Scenario*

Fig. 11 shows the MAPE values of the net load forecast in the high growth penetration scenario where the penetration rate increases by 10% every quarter (starting from the 2nd quarter). It can be seen that the LSTM model significantly increases the MAPE value from the 2nd to 4th quarter, which is consistent with the results of the low growth penetration scenario. That is, if the data changes by 10% or more, the LSTM model will have a significantly high error in forecasting. As for the FOS-ELM and OS-ELM models, MAPE is higher in the 1st quarter and decreases to lower than the LSTM model, similar to the above 2 scenarios. This is because both models have incremental learning ability even in situations where the environment highly changes from the previously learned samples. When compared with the low growth penetration scenario, it found that the MAPE of both models has higher than that of the high growth penetration scenario. This is because the higher penetration rate caused a large change in the net load then the forecasting error during the model incremental learn and adjust itself are also high. From the 3 scenarios mentioned above, it is found that the FOS-ELM and OS-ELM models have a high MAPE value at the beginning of working. To mitigate this problem, there is a paper that presents a method that can help the model achieve lower forecasting error at the beginning called the Re-Learning Method [31].

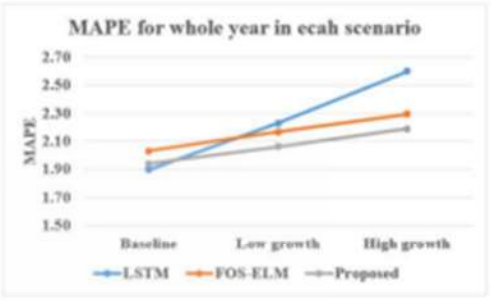


Fig.12: *MAPE for the whole year in each scenario*

Fig.12 shows the annual MAPE trend for the net load forecasting of all 3 models in the 3 scenarios. It is found that the LSTM model has a significantly higher forecasting error if the penetration rate has changed. Because the model is trained by samples that are different from the data in the real situation. In the incremental learning models such as FOS-ELM and OS-ELM, the change in penetration rate has less effect on MAPE than on the LSTM model. Because the FOS-ELM and OS-ELM models can adjust themself according to environmental changes, in other words, the FOS-ELM and OS-ELM models are more robust than the LSTM models. When comparing the FOS-ELM model and the OS-ELM with the initial training method as proposed, it is found that the OS-ELM model always has a lower forecasting error because the OS-ELM model used a sufficient amount of synthetic samples based on the actual situation for initial training, so there is less overfit problem than the FOS-ELM model that starts from zero.

6. CONCLUSION

From the experiment of the net load forecasting of LSTM, FOS-ELM, and OS-ELM models with the proposed method, it was found that in the scenario of constant PV penetration, all three models gave a nearly level of error in the annual average. Although the FOS-ELM and OS-ELM models give a higher error in the beginning, they can incrementally learn and adjust themselves to reduce the error. In the scenario of an increase in the penetration rate of PV, all three models had an increase in forecasting error. But the FOS-LEM and OS-ELM models had a much lower annual average forecasting error than the LSTM model because both models have incremental learning capacity. When comparing only the OS-ELM that uses the proposed method and the FOS-ELM model, it was found that the proposed method which is the synthesis of the initial training sample helped the OS-ELM model has a lower error of forecasting than the FOS-ELM significantly. The proposed method allows the modeling for forecasting the net load forecasting with incremental learning capability, does not require the initial training samples, and low

computational cost that is suitable for implementation on edge devices.

7. REFERENCES

[1] Statistical Review of World Energy-BP, "*Installed solar energy capacity," 2021.* [Online]. Available: https://ourworldindata.org/grapher/installed-solar-pv-capacity.

[2] M. Islam, M. Nadarajah and M. J. Hossain, "Short-Term Voltage Stability Enhancement in Residential Grid with High Penetration of Rooftop PV Units," IEEE *Transactions on Sustainable Energy*, vol. 10, no. 4, pp. 2211-2222

[3] N. Abdel-Karim, E. J. Nethercutt, J. N. Moura, T. Burgess and T. C. Ly, "Effect of load forecasting uncertainties on the reliability of North American Bulk Power System," in *2014 IEEE PES General Meeting | Conference & Exposition*, National Harbor, MD, USA, 2014, pp. 1-5

[4] K. J. Iheanetu, "Solar Photovoltaic Power Forecasting: A Review," *Sustainability*, vol. 14, no. 24, pp. 17005-17039, Dec. 2022

[5] H. Shaker, H. Chitsaz, H. Zareipour and D. Wood, "On comparison of two strategies in net demand forecasting using Wavelet Neural Network", in *2014 North American Power Symposium (NAPS)*, pullman, WA, USA, 2014, pp. 1-6.

[6] G. Aburiyana, H. Aly and T. Little, "Net load forecasting model for a power system grid with wind and solar power penetration", in *2021 Global Conference on Engineering Research (GLOBCER'21)*, Turkey, 2021, pp.1-7

[7] G. Tziolis et al., "Comparative Analysis of Machine Learning Models for Short-Term Net Load Forecasting in Renewable Integrated Microgrids," in *2022 2nd International Conference on Energy Transition in the Mediterranean Area (SyNERGY MED)*, Thessaloniki, Greece, 2022, pp. 1-5

[8] S.Muzaffar and A.Afshari,"Short-Term Load Forecasts Using LSTM Networks," *Energy Procedia*, vol. 158, pp. 2922-2927, 2019

[9] S. Kumar, L. Hussain, S. Banarjee and M. Reza, "Energy Load Forecasting using Deep Learning Approach-LSTM and GRU in Spark Cluster," in *Fifth International Conference on Emerging Applications of Information Technology (EAIT)*, Kolkata, India, pp. 1-4, 2018

[10] G. Aburiyana, H. Aly and T. Little, "Investigating the Impact of Increasing Renewable Energy Penetration Levels on the Accuracy of Net Load Forecasting," in *2022 IEEE Canadian Conference on Electrical and Computer Engineering (CCECE)*, Halifax, NS, Canada, 2022, pp. 66-71

[11] B. -f. Zhang, J. -s. Su and X. Xu, "A Class-Incremental Learning Method for Multi-Class Support Vector Machines in Text Classification," in *2006 International Conference on Machine Learning and Cybernetics*, Dalian, China, pp. 2581-2585, 2006

[12] J. Li, X. Shao and H. Zhao, "An Online Method Based on Random Forest for Air Pollutant Concentration Forecasting," in *2018 37th Chinese Control Conference (CCC)*, Wuhan, China, pp. 9641-9648, 2018

[13] Jung-Hua Wang and Wei-Der Sun, "Online learning vector quantization: a harmonic competition approach based on conservation network," *IEEE Transactions on Systems, Man, and Cybernetics Part B (Cybernetics)*, vol. 29, no. 5, pp. 642-653, Oct. 1999

[14] R. Polikar, L. Udpa, S. S. Udpa and V. Honavar, "LEARN++: an incremental learning algorithm for multilayer perceptron networks," in *2000 IEEE International Conference on Acoustics, Speech, and Signal Processing. Proceedings (Cat. No.00CH37100)*, Istanbul, Turkey, pp. 3414-3417, 2000.

[15] T. Zhang, "Solving large scale linear prediction problems using stochastic gradient descent algorithms," in *Proceedings of the Twenty-First International Conference on Machine Learning*, Banff, Alberta, Canada, ACM, pp.116-123, 2004

[16] N.Y. Liang, G.B. Huang, P.Saratchandran and N.Sundarrajan, "A fast and accurate online sequential learning algorithm for feedforward networks." *IEEE Transaction on Neural Networks*, vol.17, no.6, November 2006, pp.1411-1423

[17] S. Zhang, W. Tan and Y. Li, "A Survey of Online Sequential Extreme Learning Machine," in *2018 5th International Conference on Control, Decision and Information Technologies (CoDIT)*, Thessaloniki, Greece, pp. 45-50, 2018

[18] M. Parida, M. K. Behera and N. Nayak, "Combined EMD-ELM and OS-ELM techniques based on feed-forward networks for PV power forecasting," in *2018 Technologies for Smart-City Energy Security and Power (ICSESP)*, Bhubaneswar, India, 2018, pp. 1-6

[19] Y. Li, P. Guo, and X. Li, "Short-Term Load Forecasting Based on the Analysis of User Electricity Behavior," *Algorithms*, vol. 9, no. 4, p. 80, Nov. 2016

[20] P.K. Wong, C.M. Vong, X.H. Gao and K.I. Wong, "Adaptive control using fully online sequential extreme learning machine and a case study on engine air-fuel ratio regulation," *Mathematical Problems in Engineering*, vol. 2014, 2014

[21] M. Abufadda and K. Mansour, "A Survey of Synthetic Data Generation for Machine Learning," in *22nd International Arab Conference on Information Technology (ACIT)*, Muscat, Oman, 2021, pp. 1-7

[22] Y. Freund and R.E. Schapire, "A decision-theoretic generalization of on-line learning and an application to boosting," *Journal of Computer and System Sciences*, vol. 55, no. 1, pp.119-139, 1997

[23] M. Schwabacher, H. Hirsh and T. Ellman, "Learning prototype-selection rules for case-based iterative design," in *Proceedings of the Tenth Conference on Artificial Intelligence for Applications*, pp. 56-62, 1994.

[24] G.B. Huang, Q.Y. Zhu and C.K. Siew, "Extreme learning machine: a new learning scheme of feedforward neural networks," in *2004 IEEE International Joint Conference on Neural Networks (IEEE Cat. No.04CH37541)*, Budapest, vol.2, pp.985-990, 2004.

[25] Andrew ng, *Machine learning specialization*, Coursera, [Online] Available: https://coursera.org /specializations/machine-learning-introduction

[26] X. Liu, S. Lin, J. Fang and Z. Xu, "Is extreme learning machine feasible? a theoretical assessment (part I)," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 26, no. 1, pp. 7-20, 2015.

[27] S. Lin, X. Liu, J. Fang and Z. Xu, "Is extreme learning machine feasible? A theoretical assessment (Part II)", *IEEE Trans. Neural Networks Learn. Syst.*, vol. 26, no. 1, pp. 21-34, 2015.

[28] Y. Jaradat, M. Masoud, I. Jannoud, A. Manasrah and M. Alia, "A Tutorial on Singular Value Decomposition with Applications on Image Compression and Dimensionality Reduction," in *2021 International Conference on Information Technology (ICIT)*, Amman, Jordan, pp. 769-772, 2021

[29] W. Deng, Q. Zheng and L. Chen, "Regularized extreme learning machine," in *2009 IEEE Symposium on Computational Intelligence and Data Mining*, Nashville, TN, USA, pp. 389-395, 2009.

[30] A.Cedola, "Solar generation and power demand in Italy," Kaggle, [Online]. Available: http://www.kaggle.com/code/arielcedola/solar-generation-and-power-demand-in-italy

[31] C. Chupong, N. Junhuathon and B. Plangklang, "Short-Term Load Forecasting by FOS-ELM with Re-Learning Method," 2022 International Conference on Power, Energy and Innovations (ICPEI), Pattaya Chonburi, Thailand, 2022, pp. 1–4

Charnon Chupong received his B.Eng. degree in Electrical Engineering from King Mongkut's University of Technology North Bangkok in 2001, M.Eng. degree in Electrical Engineering from Rajamangala University of Technology Thanyaburi in 2012. He is currently working as an Assistance Professor at the Department of Electrical Engineering, Faculty of Engineering, Rajamangala University of Technology Thanyaburi (RMUTT). His research interest is on machine learning and its application in power systems and related fields.
E-mail: charnon.c@en.rmutt.ac.th

Nitikorn Junhuathon received his B. Eng., M. Eng., and Ph.D. In Electrical Engineering from Suranaree University of Technology (SUT), Thailand, in 2017, 2019 and 2023, respectively. He is currently a lecturer at Department of Electrical Engineering, Faculty of Engineering, Raja Mangala University of Technology Thanyaburi (RMUTT), Thailand. His research interest is on machine learning and application, optimization technique, power forecasting, energy, power system and related fields.
E-mail: nitikorn_j@rmutt.ac.th

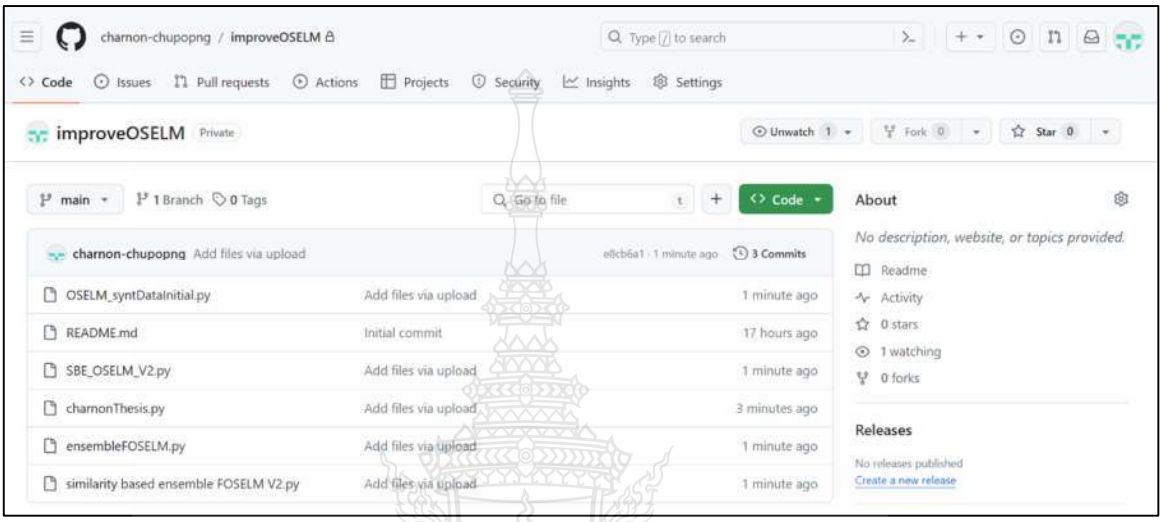
Sirichai Dangeam received his B.Eng. degree in Electrical Engineering from Rajamangala University of Technology Thanyaburi, Pathum Thani, Thailand, in 1995, M.Eng. degree in Electrical Engineering from King Mongkut's University of Technology North Bangkok, Thailand, in 2003, and D.Eng. degree in Electrical Engineering at the King Mongkut's Institute of Technology Ladkrabang, Thailand in 2020. He is currently working as an Associate Professor at the Department of Electrical Engineering, Faculty of Engineering, Rajamangala University of Technology Thanyaburi (RMUTT).
E-mail: sirichai.d@en.rmutt.ac.th

Boonyang Plangklang received his B.Eng. degree in Electrical Engineering from Rajamangala Institute of Technology, Thailand, in 1996. He received a diploma in Instrumentation at Northern Alberta Institute of Technology (NAIT), Edmonton, Alberta, Canada, in 1997. He graduated Master of Science in Electronics System and Engineering Management at University of Paderborn, division Soest, Germany, with the cooperation of Bolton Institute of Higher Education, UK, by the DAAD scholarship in 2001. He received the degree of Dr.–Ing. in Electrical Engineering from Kassel University, Germany in 2005. He is now working as an Associate Professor at the Department of Electrical Engineering, Rajamangala University of Technology Thanyaburi (RMUTT).
E-mail: boonyang.p@en.rmutt.ac.th

ภาคผนวก ข

รหัสต้นฉบับ (Source code)

https://github.com/charnon-chupopng/improveOSELM



ประวัติผู้เขียน

นายชานนท์ ชูพงษ์

จบการศึกษาระดับปริญญาตรีจากภาควิชาวิศวกรรมไฟฟ้า คณะวิศวกรรมศาตร์ มหาวิทยาลัยเทคโนโลยีพระจอมเกล้าพระนครเหนือ ในปี พ.ศ. 2544 และจบการศึกษาระดับปริญญาโทจากภาควิชาวิศวกรรมไฟฟ้า คณะวิศวกรรมศาสตร์ มหาวิทยาลัยเทคโนโลยีราชมงคลธัญบุรี ด้วยปริญญานิพนธ์ในหัวข้อ การพยากรณ์กำลังไฟฟ้าของระบบเซลล์แสงอาทิตย์โดยไม่ใช้ตัววัดความเข้มแสงอาทิตย์ “กรณีศึกษาของระบบที่ติดตั้งในมหาวิทยาลัยเทคโนโลยีราชมงคลธัญบุรี” ปัจจุบันเป็นอาจารย์ประจำภาควิชาวิศวกรรมไฟฟ้า คณะวิศวกรรมศาสตร์ มหาวิทยาลัยเทคโนโลยีราชมงคลธัญบุรี โดยมีความสนใจงานวิจัยในด้านการประยุกต์ใช้ระบบการเรียนรู้ของเครื่องในงานด้านไฟฟ้ากำลัง

E-mail: charnon.c@en.rmutt.ac.th