รายงานโครงงานวิศวกรรมไฟฟ้า วิชา 2102499

การเรียนรู้เชิงลึกสำหรับการพยากรณ์พลังงานแสงอาทิตย์ผ่านข้อมูลอนุกรมเวลา

Deep Learning for Solar Energy Forecasting using Time-series Data

นายธนัญชย์ บุญยศิริกุล เลขประจำตัวนิสิต 6330228221

อาจารย์ที่ปรึกษา ดร. สุวิชญา สุวรรรวิมลกุล

ภาควิชาวิศวกรรมไฟฟ้า คณะวิศวกรรมศาสตร์

จุฬาลงกรณ์มหาวิทยาลัย

ปีการศึกษา 2566

ลงชื่ออาจารย์ที่ปรึกษาหลัก . . . . . .

(. . . . .)

วันที่ 26 เมษายน 2566

บทคัดย่อ

ความเข้มแสงอาทิตย์เป็นส่วนหนึ่งที่สามารถนำมาคำนวณหากำลังในการผลิตไฟฟ้าของเซลล์แสงอาทิตย์ซึ่งสามารถวัดได้จากอุปกรณ์แต่เนื่องจากไม่สามารถติดตั้งอุปกรณ์เซ็นเซอร์ได้ทุกพื้นที่ ดังนั้นการพยากรณ์ค่าความเข้มแสงอาทิตย์จึงเป็นหัวข้อที่น่าสนใจโดยจุดประสงค์ของโครงงานคือการศึกษาแบบจำลองในการทำนายข้อมูลอนุกรมเวลาด้วยการเรียนรู้เชิงลึกมีทั้งหมด 8 แบบจำลองคือ Regression LSTM, Linear, NLinear, DLinear, PatchTST, Transformer, Autoformer และ Informer และวิเคราะห์ผลการความผิดพลาดจากการทำนายความเข้มแสงอาทิตย์จากแบบจำลองที่ได้ศึกษาโดยจะใช้ข้อมูลจาก Solar CUEE Station ทำการเก็บข้อมูลตั้งแต่วันที่ 1 มกราคม พ.ศ. 2565 ถึงวันที่ 22 มิถุนายน พ.ศ. 2566 ทำการเก็บข้อมูลตั้งแต่เวลา 07:00 น. ถึง 17:00 น. มีความละเอียดทุก 15 นาทีและมีจำนวนสถานีที่วัดค่าจำนวน 56 สถานี โดยโครงงานนี้ศึกษาการทำนายระยะสั้นด้วยและการทำนายระยะยาว และมีการพิจารณาผลอันเกิดจากใช้ข้อมูลอินพุด เช่น การใช้ความเข้มแสงอาทิตย์ในอดีตเป็นอินพุตอย่างเดียว และการใช้ข้อมูลอื่นๆ เช่น ดัชนีฟ้าใส เวลา ละติจูด ลองจิจูด และการปรับพารามิเตอร์เพื่อให้ผลการทำนายดีที่สุด ซึ่งพบว่า ในการทำนายระยะสั้น แบบจำลอง Informer เป็นเวลา 1 ชั่วโมงในอนาคตมีค่าผิดพลาดเฉลี่ยสมบูรณ์ น้อยที่สุดเมื่อเทียบกับวิธีอื่นๆ คือ เท่ากับ 85.73 W/m^2 และ ในการทำนายระยะยาวด้วยแบบจำลอง PatchTST เป็นเวลา 9 ชั่วโมงในอนาคตมีค่าผิดพลาดเฉลี่ยสมบูรณ์ น้อยที่สุดเมื่อเทียบกับวิธีอื่นๆ คือเท่ากับ 128.23 W/m^2

คำสำคัญ: การทำนายค่าความเข้มแสงอาทิตย์, แบบจำลองการเรียนรู้เชิงลึก, การทำนายความเข้มแสงอาทิตย์ระยะสั่น, การทำนายความเข้มแสงอาทิตย์ระยะยาว

Abstract

Solar irradiance is a component that can be used to calculate the power production of solar cells, which can be measured by devices such as sensors. However, we can't install sensors to many area. So, forecasting solar irradiance values is an interesting topic that can be useful for predicting which areas are suitable for installing solar cells. The objective of this project is to study time-series forecasting using deep learning models: Regression LSTM, Linear, NLinear, DLinear, PatchTST, Transformer, Autoformer, and Informer. The data used in this study is obtained from the Solar CUEE station, collected from January 1 2023 to June 22 2024 recorded from 7:00 a.m. to 5:00 p.m. with a 15-minute resolution with 56 measurement stations installed nationwide. This study provides the impact of the input features, that is, the univariate inputs—where only the historical data of the target solar irradiance are used in forecasting — and the multivariate inputs—where other variables, *e.g.* clear sky index, time stamp, latitude, longitude, are considered, as well as the impact of parameter tuning. Then, the performance of these deep learning models is evaluated for short-term and long-term forecasting. Our study shows that Informer provides the best performance for the short-term forecasting (1 hour ahead) with the mean absolute error of 85.73 W/m2. Meanwhile, for the long-term forecasting, the PatchTST offers the best performance for 9 hours ahead into the future with the mean absolute error of 128.23 W/m2.

Keywords: Irradiance forecasting, Deep learning model, Short-term forecasting, Long-term forecasting

สารบัญ

1 บทนำ **5**

1.1 ที่มาและความสำคัญ 5

1.2 วัตถุประสงค์ 5

1.3 ขอบเขตของโครงงาน 6

1.4 ผลที่คาดว่าจะได้รับ 6

2 หลักการและทฤษฎีที่เกี่ยวข้อง **7**

2.1 การพยากรณ์ข้อมูลอนุกรมเวลาแบบเบื้องต้น 7

2.1.1 Autoregressive 7

2.1.2 Moving Average 8

2.1.3 Autoregressive Integrated Moving Average 8

2.2 แบบจำลองการพยากรณ์ข้อมูลอนุกรมเวลาด้วยโครงข่ายประสาท (Neural Network) 9

2.2.1 Recurrent Neural Network (RNN) 10

2.2.2 Long Short-Term Memory (LSTM) 11

2.2.3 Transformer 12

2.2.4 Autoformer 16

2.2.5 Informer 20

2.2.6 Linear, DLinear และ NLinear 22

2.2.7 PatchTST 24

2.3 การวัดประสิทธิภาพ 27

2.3.1 Mean Absolute Error (MAE) 27

2.3.2 Mean Square Error (MSE) 27

2.3.3 Normalized Root Mean Square Error (NRMSE) 27

3 ผลลัพธ์จากการดำเนินการ **28**

3.1 ชุดข้อมูลจาก Solar CUEE Station 28

3.2 ชุดข้อมูลทั่วทั้งประเทศ 28

3.3 การติดตั้งพารามิเตอร์สำหรับการฝึกโมเดล 28

3.4 การพยากรณ์ข้อมูลอนุกรมเวลา 29

3.4.1 การพยากรณ์ข้อมูลอนุกรมเวลาระยะสั้นแบบตัวแปรเดียว 29

3.4.2 การพยากรณ์ข้อมูลอนุกรมเวลาระยะยาวแบบตัวแปรเดียว 31

3.4.3 การพยากรณ์ข้อมูลอนุกรมเวลาระยะสั้นแบบหลายตัวแปรที่ไม่รวมค่าความเข้มแสง . . . 33

3.4.4 การพยากรณ์ข้อมูลอนุกรมเวลาระยะสั้นแบบหลายตัวแปรที่รวมค่าความเข้มแสง 35

3.4.5 การพยากรณ์ข้อมูลอนุกรมเวลาระยะยาวแบบหลายตัวแปรที่รวมค่าความเข้มแสง . . . 37

3.5 ผลการปรับแต่งค่าพารามิเตอร์ของแบบจำลอง 40

3.5.1 กรณีทำนายข้อมูลอนุกรมเวลาในระยะสั้นหลายตัวแปรที่รวมค่าความเข้มแสง 40
3.5.2 กรณีทำนายข้อมูลอนุกรมเวลาในระยะยาวแบบหลายตัวแปรที่รวมค่าความเข้มแสง . . . 45

4 บทสรุป **50**
4.1 สรุปผลการดำเนินการ 50
4.2 ปัญหา อุปสรรค และแนวทางแก้ไข 51

5 กิตติกรรมประกาศ **52**

A ภาคผนวก **55**
A.1 การตั้งค่า Hyperparameters ในการทดลองที่ 3.4.1 และ 3.4.2 55
A.2 การตั้งค่า Hyperparameters ในการทดลองที่ 3.4.3 56
A.3 การตั้งค่า Hyperparameters ในการทดลองที่ 3.4.4 และ 3.4.5 57
A.4 ตัวอย่างการใช้งาน 58

1 บทนำ

1.1 ที่มาและความสำคัญ

ค่าพลังงานแสงอาทิตย์ (Solar irradiance) เป็นกำลังงานของพลังงานแสงอาทิตย์ที่ฉายบนพื้นโลกต่อหนึ่งหน่วยพื้น ข้อมูลชนิดนี้สามารถเก็บในลักษณะของอนุกรมเวลา และการทำนายค่าพลังงานอนาคต นั้นอาจเป็นการทำนายระยะสั้น ภายในระยะ 1 ชั่วโมง (intra-hour) หรือการทำนายระยะยาวเช่น ระหว่างวัน (intra-day) ซึ่งในงานวิจัยด้านหลังงานส่วนใหญ่จะทำการออกแบบโครงข่ายประสาท (neural network) โดยใช้สถาปัตยกรรมพื้นฐานเช่น การใช้ multilayer perceptron (MLP), convolution (CONV), และ Long short-term memory (LSTM) ซึ่งได้รวบรวมอย่าสมบูรณ์ไว้ใน [**?**] นอกจากนี้ วิธีการที่รวบรวมไว้นี้รับค่าอินพุตเป็นค่าที่วัดและค่าทางสถิติในอดีต เพื่อใช้เป็น features ซึ่งไม่มีความเกี่ยวข้องกับค่าพลังงานแสงอาทิตย์ หรือกล่าวคือ เป็นค่าที่เกี่ยวของกับสภาวะภายนอก (exogenous features) โดยให้เอาท์พุตเป็น การทำนายค่าพลังงานแสงอาทิตย์ ณ เวลาหนึ่งๆ (output length $= 1$)

อย่างไรก็ดีเมื่อไม่นานมานี้ มีงานวิจัยจำนวนกำเนิดมา เสนอการพัฒนาการพยากรณ์อนุกรมเวลาระยะยาว (long-term time-series forecasting) ด้วยวิถีทางการเรียนรู้เชิงลึก (deep learning) ซึ่งไม่ได้ออกแบบมาเพื่อการทำนายค่าพลังงานแสงอาทิตย์โดยเฉพาะ งานวิจัยเหล่านี้ มุ่งเน้นไปที่การพยากรณ์ระยะยาวและให้ผลการทำนายค่ามากกว่า 1 ค่า (output length > 1) ซึ่งการพยากรณ์ข้อมูลอนุกรมโดยการออกแบบโครงข่ายประสาท สามารถทำในสองลักษณะ คือ Iterated Multi-step Forecasting (IMF) และ Direct Multi-step Forecasting (DMF) โดยที่วิธีการของ (IMF) จะสร้างแบบจำลองทำนายข้อมูลที่เวลาถัดไป โดยอ้างอิงจากข้อมูลการทำนายก่อนหน้า เช่น RNN [1], LSTM [2], AutoFormer [3], ในขณะที่ DMP จะสร้างแบบจำลองทำนายข้อมูลที่เวลาถัดไป โดยไม่มีการอ้างอิงข้อมูลการทำนายก่อนหน้า เช่น Informer [4], Linear [5] และ PatchTST [6] วิธีการเหล่านี้ได้ถูกศึกษา ทั้งการใช้ ค่าอินพุตฟีเจอร์ มีความเกี่ยวข้องกับสภาวะภายใน (endogenous features) และมีความเกี่ยวข้องกับสภาวะภายนอก (exogenous features)

ดังนั้นโครงงานฉบับนี้จึงได้รวบรวมการศึกษาการประยุกต์วิธีการพยากรณ์ข้อมูลอนุกรมโดยการออกแบบโครงข่ายประสาท (neural network) และทำการเปรียบเทียบสมรรถนะของงานวิจัยที่พึ่งกำเนิดมานี้ในการทำนายค่าพลังงานแสงอาทิตย์โดยเฉพาะ โดยศึกษาตามกรณีต่อไปนี้

- Iterated Multi-step Forecasting (IMF): LSTM, AutoFormer
- Direct Multi-step Forecasting (DMF): Transformer, Linear/DLinear/NLinear, Informer, PatchTST
- ค่าอินพุตฟีเจอร์เพียงตัวแปรเดียว (unvariate) และ หลายตัวแปร (multi-variate)
- การตั้งค่าค่าข้อมูลขาเข้าในกรณีที่ 1. มีค่าความเข้มแสงในอดีตเป็นส่วนหนึ่งของการเรียนรู้แบบจำลองและ 2. กรณีที่ไม่มีความเข้มแสงในอดีตสำหรับการเรียนรู้แบบจำลอง

โครงงานฉบับนี้จึงเน้นศึกษาคุณสมบัติของโครงข่ายประสาท (neural network) ในช่วงการพยากรณ์เอาท์พุตที่สั้นลง คือ ความยาวระหว่างวัน (Intraday) เพื่อหาเข้าใจถึงประสิทธิภาพและเพื่อวิเคราะห์หาการต่อยอดวิธีการเหล่านี้ให้มีประสิทธิภาพที่ดีขึ้น

1.2 วัตถุประสงค์

- เพื่อศึกษาและออกแบบโมเดลสำหรับการพยากรณ์ทางอนุกรมเวลาด้วยโครงข่ายประสาท

- เพื่อหาแนวทางในการพัฒนาและปรับปรุงโมเดลสำหรับการพยากรณ์ทางอนุกรมเวลาสำหรับการใช้งานแบบ Intra-day forecasting

1.3 ขอบเขตของโครงงาน

- โครงงานชิ้นนี้จะมุ่งเน้นการพยากรณ์ทางอนุกรมเวลาในระยะยาว (Long Term Time-series Forecasting)
- โครงงานชิ้นนี้จะมุ่งเน้นการหาผลลัพธ์ดังกล่าวด้วยวิธีของการเรียนรู้เชิงลึก (Deep Learning)
- ข้อมูลที่ใช้เป็นข้อมูลที่ถูกวัดที่สถานีภาคพื้นดิน (Ground Station) โดยจำกัดระบบการประมวณข้อมูลเป็นชนิด 1 มิติ (1 Dimensional Data) เท่านั้น

1.4 ผลที่คาดว่าจะได้รับ

1. การเปรียบเทียบประสิทธิภาพระหว่าง LSTM, AutoFormer, Transformer, Linear/DLinear/NLinear, Informer, PatchTST และการตั้งค่าต่อไปนี้
    - ค่าอินพุตฟีเจอร์เพียงตัวแปรเดียว (unvariate) และ หลายตัวแปร (multi-variate)
    - การใช้ค่าอินพุตมีความเกี่ยวข้องกับสภาวะภายใน (endogenous features) และมีความเกี่ยวข้องกับสภาวะภายนอก (exogenous features)
    - การ forecast ที่ predict length = 4 (intra-hour) และ 36 (intra-day)
2. วิธีการของเทคนิคที่ใช้ในการเรียนรู้เชิงลึกและการเตรียมข้อมูล

2 หลักการและทฤษฎีที่เกี่ยวข้อง

การเลือกวิธีการพยากรณ์ที่จำเป็นจะต้องหาวิธีที่เหมาะสมกับลักษณะของข้อมูลอนุกรมเวลา วิธีการพยากรณ์ข้อมูลอนุกรมเวลาเบื้องต้นมักจะมีสมมติฐานว่า ข้อมูลอนุกรมเวลามีคุณลักษณะนิ่ง (Stationary) คุณลักษณะนิ่ง เป็นคุณลักษณะพื้นฐานที่ระบุว่า หาก ข้อมูลอนุกรมเวลามีคุณลักษณะนิ่ง แล้วนั้น ค่าทางสถิติไม่เปลี่ยนแปลงตามการเปลี่ยนของเวลา วิธีการพยากรณ์ที่อาศัยสมมติฐานนี้ ได้แก่ Autoregressive และ Moving Average แต่โดยทั่วไปแล้วข้อมูลที่มีอยู่ในชีวิตจริงจะมีคุณสมบัติ Non-stationary ทำให้ต้องใช้องค์ประกอบเสริมขึ้นมาบางส่วนเพื่อทำให้มีคุณสมบัติใกล้เคียงกับ Stationary ซึ่งกลายเป็นวิธี Autoregressive Integrated Moving Average

ข้อมูลที่มีอยู่ในชีวิตจริงที่มีคุณสมบัติ Non-stationary อาจจะมีคุณลักษณะทางเวลา (Time Series Pattern) อื่น ซึ่งสามารถจำแนกได้เป็น Trend, Seasonal Pattern, และ Cyclic Pattern โดยข้อมูลอนุกรมเวลาส่วนใหญ่จะมีคุณลักษณะเหล่านี้ผสมกันอยู่ [7] โดยการพยากรณ์ด้วยโครงข่ายประสาทส่วนใหญ่จะถูกออกแบบให้รองรับข้อมูลอนุกรมเวลาที่มีลักษณะเหล่านี้

- Trend คือการเพิ่มหรือลดอย่างต่อเนื่องและแสดงทิศทางการเพิ่มหรือลดของอนุกรมเวลาในระยะยาว
- Seasonal Pattern คือการเปลี่ยนแปลงของอนุกรมเวลาที่ถูกกระทบด้วยฤดูกาล ซึ่งมีความคงที่ในการซ้ำคาบและอาจจะมีรอบของความถี่ที่สูงและต่ำ
- Cyclic Pattern คือการเปลี่ยนแปลงของอนุกรมขึ้นลงแบบไม่ซ้ำคาบแต่มักจะเกิดจากปัจจัยภายนอกเช่น สภาวะเศษฐกิจ

โดยในต่อไปนี้ข้าพเจ้าจะกล่าถึงวิธีการพยากรณ์ข้อมูลอนุกรมเวลาแบบเบื้องต้นในบทที่ 2.1 และการพยากรณ์ด้วยโครงข่ายประสาท (Neural Network) ในบทที่ 2.2

2.1 การพยากรณ์ข้อมูลอนุกรมเวลาแบบเบื้องต้น

2.1.1 Autoregressive

Autoregressive [7] เป็นการพยากรณ์ข้อมูลอนุกรมเวลา ด้วย Linear combination ของข้อมูลที่วัดได้ในอดีต (y) ซึ่งสามารถแสดงได้ด้วยสมการ

$$y_t(p) = c + \sum_{i=1}^{p} \phi_i y_{t-i} + \epsilon_t \tag{1}$$

โดยที่

- $\epsilon_{(.)}$ คือสัญญาณรบกวนแบบสุ่ม (White noise) ซึ่งมีความถี่สูง มีค่าเฉลี่ยเป็นศูนย์และความแปรปรวนเป็น σ^2
- $\phi_{(.)}$ คือค่าสัมประสิทธ์ของแบบจำลอง Autoregressive
- p คือพารามิเตอร์ที่ระบุตำแหน่งของเวลาก่อนหน้าปัจจุบันหรือเรียกว่า Lag ที่เวลา p ยกตัวอย่างเช่น $y(t-p)$ แสดงถึงค่าที่วัดได้ ณ Lag ที่เวลา p หากสัญญาณที่เวลาปัจจุบันอยู่ที่เวลา t

2.1.2 Moving Average

แบบจำลอง Moving Average [7] เป็นการพยากรณ์ข้อมูลอนุกรมเวลา ด้วย Linear Combination ของค่าความผิดพลาดในอดีต (Past Forecast Error) ซึ่งสามารถแสดงได้ด้วยสมการ

$$y_t(q) = c + \sum_{i=1}^{q} \theta_i \epsilon_{t-i} \tag{2}$$

โดยที่

- c คือค่าคงที่
- q คือพารามิเตอร์ที่ระบุตำแหน่งของเวลาก่อนหน้าปัจจุบันหรือเรียกว่า Lag ที่เวลา q ยกตัวอย่างเช่น $y(t-q)$ แสดงถึงค่าที่วัดได้ ณ Lag ที่เวลา q หากสัญญาณที่เวลาปัจจุบันอยู่ที่เวลา t
- $\theta_{(\cdot)}$ คือค่าสัมประสิทธ์ของแบบจำลอง Moving Average
- $\epsilon_{(\cdot)}$ คือค่าความผิดพลาดในอดีต ซึ่งมีความถี่สูง และมีการกระจายตัวแบบสุ่ม (White Noise) ที่มีค่าเฉลี่ยเป็นศูนย์และความแปรปรวนเป็น σ^2

2.1.3 Autoregressive Integrated Moving Average

Autoregressive Integrated Moving Average [7] เป็นแบบจำลองที่ใช้ในการพยากรณ์ข้อมูลอนุกรมเวลาด้วยวิธีทางสถิติโดยอ้างอิงจากข้อมูลในอดีตโดยที่ข้อมูลที่ใช้ในแบบจำลองจะต้องมีคุณสมบัติ Stationary คือไม่ขึ้นอยู่กับเวลา รวมถึงไม่มีแนวโน้มของข้อมูล (Trend) และการแปรผันตามฤดูกาล (Seasonal) แต่โดยทั่วไปแล้วข้อมูลที่มีอยู่ในชีวิตจริงจะมีคุณสมบัติ Non-stationary ทำให้ต้องใช้องค์ประกอบเสริมขึ้นมาบางส่วนเพื่อทำให้มีคุณสมบัติใกล้เคียงกับ Stationary

แบบจำลองของ ARIMA จะเป็นการรวมเอา Autoregressive และ Moving Average ในการสร้างโมเดลเพื่อพยากรณ์ นอกจากนี้ยังมีองค์ประกอบที่เสริมขึ้นมาเพื่อรองรับความเป็น Non-stationary ดังนั้นจำนวนสัมประสิทธิ์ที่ใช้ในแต่ละโมเดลจะเป็นส่วนสำคัญที่ประกอบกันเป็น ARIMA ดังนั้น การอ้างอิงโมเดล ARIMA จะต้องระบุจำนวนสัมประสิทธิ์นี้ เช่น หากให้ p, q, และ d แทนจำนวนสัมประสิทธิ์ที่ใช้ในแต่ละส่วนของ Autoregressive, Moving Average, และ โมเดลองค์ประกอบที่เสริมขึ้น เราจะให้ตัวแทน ARIMA นี้คือ ARIMA(p, q, d)

องค์ประกอบสามส่วนมีดังนี้

1. แบบจำลอง Autoregressive เป็น Linear combination ของข้อมูลที่วัดได้ในอดีต (y) ดังสมการ (1)
2. แบบจำลอง Moving Average เป็นการคำนวณค่าเฉลี่ยค่าความผิดพลาดในอดีต ดังสมการ (2)
3. กระบวนการ Integrated เป็นการหาผลต่างของข้อมูลอนุกรมเวลาระหว่างข้อมูลปัจจุบันและเวลาในอดีต ซึ่งองค์ประกอบนี้เป็นส่วนสำคัญที่ทำให้ข้อมูลมีคุณสมบัติ Stationary สามารถแสดงได้ด้วยสมการ

$$y_d(t) = y_{d-1}(t) - y_{d-1}(t-1) \tag{3}$$

โดยที่ d คือจำนวนของการหาผลต่างของข้อมูลปัจจุบันและข้อมูลในอดีตในกระบวนการ Integrated ยกตัวอย่างเช่น $y_1(t) = y_0(t) - y_0(t-1)$ เป็นการนำข้อมูล ณ เวลา t=0 ลบด้วยข้อมูล ณ เวลา t=t-1

จำนวนสัมประสิทธิ์ที่ใช้ในแบบจำลอง Autoregressive และ จำนวนสัมประสิทธิ์ Moving Average สามารถหาจากการวิเคราะห์จากการพล็อต Autocorrelation function (ACF) และ Partial-autocorrelation function (PACF) โดยสามารถพล็อตโดยใช้คอมพิวเตอร์ในการจำลอง

ทั้งนี้ สมการ (3) สามารถนำมาแสดงผลใหม่โดยใช้ Lag operator ซึ่งเป็นตัวดำเนินการที่ทำการเลื่อนทางเวลาในข้อมูลอนุกรมเวลา ยกตัวอย่างเช่น $Ly(t) = y(t-1)$ โดยสามารถแสดงแบบจำลองในรูปพหุนามที่ใช้ Lag operatorดังนี้

$$A(L)(I-L)^d y(t) = C(L)\epsilon(t) \tag{4}$$

โดยที่

- $A(L) = 1 - (a_1 L + \ldots + a_p L^p)$: เป็นแบบจำลอง Autoregressive polynomial ที่สนใจข้อมูลในอดีตที่เวลา p
- $C(L) = 1 + (c_1 L + \ldots + a_q L^q)$: เป็นแบบจำลอง Moving Average polynomial ที่สนใจหาค่าเฉลี่ยของข้อมูลสัญญาณรบกวนแบบสุ่มในอดีตที่เวลา q
- $(I-L)^d$ แสดงถึงการหาผลต่างระหว่างข้อมูลปัจจุบันและข้อมูลในอดีตเป็นจำนวน d ครั้ง

ลำดับการทำงาน

- พิจารณาข้อมูลอนุกรมเวลาก่อนโดยวิเคราะห์คุณสมบัติ Stationary ซึ่งหากพบว่ายังไม่มีคุณสมบัตินี้จึงใช้กระบวนการ Integrated จากสมการสมการ (3) โดยจำนวนครั้งที่ทำการหาผลต่างจะเป็นพารามิเตอร์ d สำหรับแบบจำลองนี้
- พิจารณากราฟ ACF และ PACF ซึ่งพารามิเตอร์ p จากแบบจำลอง Autoregressive จะพิจารณาที่กราฟ PACF มีค่าลู่เข้าสู่ศูนย์ ณ ตำแหน่งของเวลาในอดีต และพารามิเตอร์ q จากแบบจำลอง Moving Average จะพิจารณาที่กราฟ ACF มีค่าลู่เข้าสู่ศูนย์ ณ ตำแหน่งของเวลาในอดีต

เมื่อกำหนดพารามิเตอร์ดังกล่าวเสร็จสิ้น แบบจำลองนี้สามารถทำนายข้อมูลในอนาคตได้ แต่ข้อจำกัดของแบบจำลองนี้ คือประสิทธิภาพในการทำนายข้อมูลในอนาคตที่ระยะยาวลดลง

2.2 แบบจำลองการพยากรณ์ข้อมูลอนุกรมเวลาด้วยโครงข่ายประสาท (Neural Network)

การพยากรณ์ข้อมูลอนุกรมโดยการออกแบบโครงข่ายประสาท สามารถทำในสองลักษณะ คือ (i) Iterated Multi-step Forecasting (IMF) และ (ii) Direct Multi-step Forecasting (DMF):

(i) Iterated Multi-step Forecasting จะสร้างแบบจำลองทำนายข้อมูลที่เวลาถัดไป โดยอ้างอิงจากข้อมูลการทำนายก่อนหน้า หลังจากทำนายเสร็จก็จะเลื่อนลำดับข้อมูลก่อนหน้าในการสร้างแบบจำลองและทำนายข้อมูลในลำดับถัด ตัวอย่างของวิธีเหล่านี้ เช่น Autoregressive, Moving Average เฉกเช่นกับวิธีการด้วยโครงข่ายประสาท (Neural Network) เช่น RNN [1] และ LSTM [2] แม้วิธีเหล่านี้นั้นมีประสิทธิภาพสูงในการทำงาน อย่างไรก็ตามงานวิจัยของ [5] ได้ออกมาตั้งคำถามถึงความเป็นไปได้วิธีเหล่านี้อาจจะมีการสะสมค่าผิดพลาด เมื่อถูกใช้ในในการทำนายข้อมูลอนุกรมเวลาระยะยาวจึงอาจส่งผลให้แบบจำลองไม่มีประสิทธิภาพ

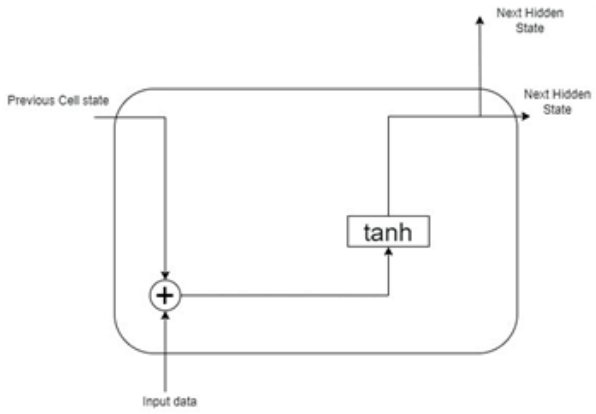
ในขณะที่ (ii) Direct Multi-step Forecasting จะสร้างแบบจำลองทำนายข้อมูลที่เวลาถัดไป โดยไม่มีการอ้างอิงข้อมูลการทำนายก่อนหน้าหรือก็คือเป็นการทำนายข้อมูลในอนาคตด้วยแบบจำลองเดียวโดยอ้างอิงกับข้อมูลอนุกรมเวลาที่ได้รับมาทั้งหมด ซึ่งพบว่าวิธีนี้เหมาะสมสำหรับทำนายข้อมูลอนุกรมเวลาในระยะยาวเนื่องจากไม่มีการสะสมค่าผิดพลาดในทุกการเลื่อนของเวลา

นอกจากนี้โครงข่ายประสาทอาจถูกออกแบบให้สกัดคุณลักษณะในข้อมูลอนุกรมเวลา (Time Series Pattern) ซึ่งก็คือ Trend, Seasonal, และ Cyclic [7] ตัวอย่างได้แก่ตระกูล Transformers และ Prophet เช่น Autoformer [3], NeuralProphet [8], NLinear และ DLinear [5] อย่างไรก็ตาม พบว่าวิธี NLinear และ DLinear ให้ค่าผิดพลาดที่น้อยกว่า Transformer โดยประมาณ 20% เนื่องจากวิธีของ Transformer จะมีกระบวนการที่ใช้ประโยชน์จากข้อมูลที่มีความหมายเชิงสัญลักษณะ (Semantic Information) ดังนั้นจึงมีประสิทธิภาพถดถอยลงหากข้อมูลที่ใช้ขาดความหมายเชิงสัญลักษณะ นอกจากนี้ NLinear และ DLinear จะทำการแยกเพียงแค่ Seasonal และ Trend Features ออกจากกันเท่านั้นและโมเดลจะกระกอบด้วย Linear Layer เป็นหลัก เมื่อเทียบกับ NeuralProphet พยายามสกัดหาคุณลักษณะอื่นๆ โดยการประกอบโครงข่ายประสาทอื่นๆ เข้าด้วยกัน ดังนั้น NeuralProphet จึงมีความซับซ้อนของโมลเดลมากกว่า NLinear และ DLinear

โครงสร้างแบบจำลองที่ใช้ในการทำนายข้อมูลอนุกรมเวลาโดยแบบจำลองโครงข่ายประสาทมีดังนี้

2.2.1 Recurrent Neural Network (RNN)

Recurrent Neural Network (RNN) [1] เป็นโครงสร้างของโครงข่ายประสาทเทียม (Neural Network) ที่ถูกออกแบบมาเพื่อการจัดการกับข้อมูลแบบลำดับ (Sequential Data) ที่เปลี่ยนไปตามเวลาหรือลำดับของข้อมูล เช่น ข้อมูลเสียง ข้อมูลราคาหุ้น เป็นต้น การที่ RNN มีความแตกต่างจากโครงข่ายประสาททั่วไปคือการมีการรับข้อมูลจากเลเยอร์ก่อนหน้า (Hidden Layer) เพื่อนำไปประมวลผลในเลเยอร์ปัจจุบันและส่งข้อมูลไปยังเลเยอร์ถัดไป ซึ่งกระบวนการนี้ทำให้เมื่อฝึกสอนแบบจำลอง RNN จะสามารถเรียนรู้และจดจำข้อมูลทำให้โครงสร้างนี้เหมาะสมสำหรับการทำนายข้อมูลชนิดที่เป็นลำดับ (Sequential Data) รูปที่ 1. แสดงองค์ประกอบและลำดับการทำงานของ RNN



รูปที่ 1: โครงสร้างการทำงานของ RNN (Recurrent Neural Network)

$$h_t = \tanh\ (W_x \times x_t + W_h \times h_{t-1} + b) \tag{5}$$

โดยที่

- x_t คือ ข้อมูลขาเข้า (Input Data) ที่เป็นข้อมูลชนิดลำดับ

- h_t คือ ข้อมูลขาออกสำหรับสถานะ ณ ปัจจุบัน ของแบบจำลอง ซึ่งอาจในไปใช้เป็นสถานะของแบบจำลอง ซึ่งจะเป็นข้อมูลขาเข้าสำหรับ Hidden Layer
- W_x คือ เมทริกซ์ถ่วงน้ำหนักที่ใช้สำหรับขารับข้อมูลเข้า
- W_h คือ เมทริกซ์ถ่วงน้ำหนักที่ใช้กับสถานะของแบบจำลอง
- b คือ ไบแอส (Bias) ช่วยปรับแบบจำลองให้มีประสิทธิภาพดีขึ้น

โครงสร้าง ประกอบด้วย ขารับข้อมูลเข้า (Input Data) (x_t) และ ขารับข้อมูลจาก (Hidden Layer (h_t)) โดยที่ข้อมูลทั้งสองจะถูกประมวลผล (Process) ดัง สมการ (5) ดังนี้ ข้อมูลขาเข้าจะถูกฉายลงบนเมทริกซ์ถ่วงน้ำหนักด้วยเมทริกซ์ W_x และ ข้อมูลจากเลเยอร์ก่อนหน้า (h_t) จะถูกฉายลงบนเมทริกซ์ถ่วงน้ำหนักด้วยเมทริกซ์ W_h หลังจากนั้นนำผลลัพธ์ที่ได้มาบวกและคำนวณผ่านฟังก์ชัน $tanh$ ซึ่งผลลัพธ์เป็นข้อมูลขาออกสำหรับสถานะ ณ ปัจจุบัน และจะเป็นขารับข้อมูลเข้าในสถานะของแบบจำลองถัดไป

ลำดับการทำงาน ข้อมูลจากเลเยอร์ก่อนหน้า (h_t) ของ RNN จะถูกนำไปใช้ในการส่งผ่านข้อมูลที่ถูกจำมาจากข้อมูลในอดีตของสถานะของแบบจำลองในขณะที่ขา x_t จะรับข้อมูลที่ฉับพลันกว่า (Instantaneous data) ซึ่งเป็นข้อมูลขาเข้า ซึ่งผลลัทธ์จากการประมวลดังสมการ (5) จะมีการรวมกันระหว่างข้อมูลที่ถูกจดจำและข้อมูลที่ฉับพลัน

2.2.2 Long Short-Term Memory (LSTM)

Long Short-Term Memory (LSTM) [2, 9] เป็นแบบจำลองที่ได้รับการออกแบบมาเพื่อแก้ปัญหาของ RNN ทั่วไปที่มีปัญหาในการเรียนรู้ให้จำข้อมูลที่มีระยะเวลายาวและสามารถเลือกที่จะรับและลืมข้อมูลในอดีตที่ไม่จำเป็นได้และเก็บในรูปแบบของความทรงจำระยะสั้น (Hidden State) และความทรงจำระยะยาว (Cell State) โดย LSTM จะช่วยพัฒนาแบบจำลอง RNN ให้ไม่เกิดปัญหาเกรเดียนส์เข้าใกล้ศูนย์ขณะฝึกสอนแบบจำลอง (Vanishing gradient) และ เกรเดียนส์มีค่าเข้าใกล้อนันต์ขณะฝึกสอนแบบจำลอง (Exploding gradient) ในโครงสร้างของ RNN ที่ข้อมูลลำดับมีความยาวมากกว่าปกติการทำงานของ LSTM สามารถอธิบายได้ด้วยสมการ (6) - สมการ (10) ดังนี้

$$f_t = \sigma((W_h \times h_{t-1}) + (W_x \times x_t) + b_f) \tag{6}$$

$$i_t = \sigma((W_i \times x_t) + (W_h \times h_{t-1}) + b_i) \tag{7}$$

$$O_t = \sigma((W_x \times x_t) + (W_h \times h_{t-1}) + b_o) \tag{8}$$

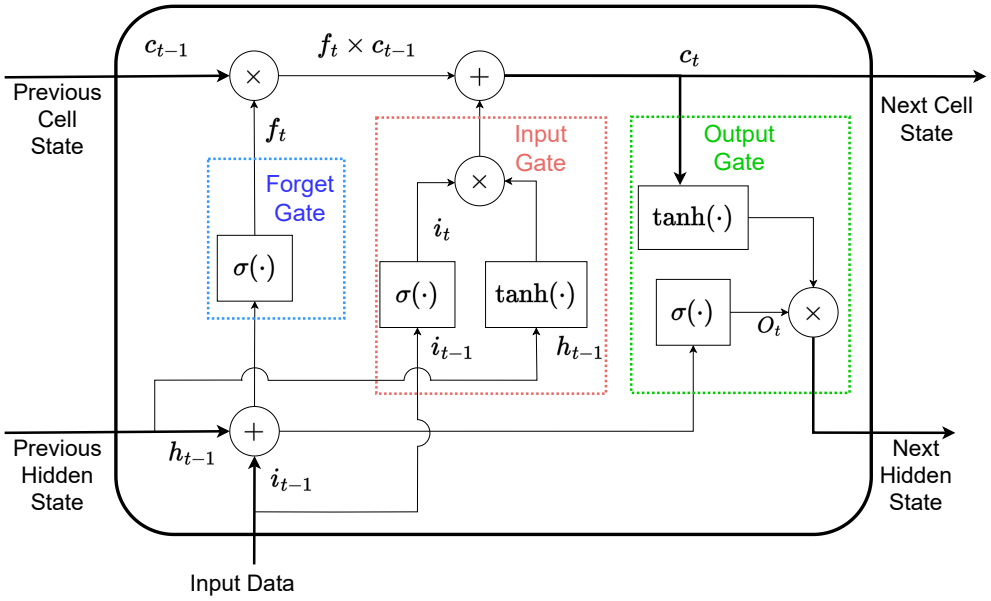
$$c_t = (f_t \times c_{t-1} + i_t \times tanh\ (h_{t-1})) \tag{9}$$

$$h_t = O_t \times tanh\ (c_t) \tag{10}$$

โดยที่ $\sigma(\cdot)$ คือฟังก์ชัน Sigmoid มีนิยามคือ $\sigma(x) = \frac{1}{(1+e^{-x})}$ ซึ่งมีค่าระหว่างศูนย์ถึงหนึ่ง

โครงสร้างและลำดับการทำงาน ของ LSTM จะอธิบายตามขั้นตอนการทำงานของประตูรับสัญญาณ (Gate) ดังนี้ :

- Forget Gate: มีหน้าที่ตัดสินใจว่าจะเก็บหรือลบข้อมูลในความทรงจำระยะยาวซึ่งเป็นขาข้อมูลความจำระยะยาว(Long-term memory)โดยอ้างอิงผลลัพธ์จากสมการ (6) ซึ่งใช้ข้อมูลจากความทรงจำระยะสั้นก่อนหน้านี้และ



รูปที่ 2: โครงสร้างการทำงานของ Long Short-Term Memory (LSTM)

อินพุทปัจจุบันและถ่วงน้ำหนักกับ f_t ซึ่งมีค่าระหว่างศูนย์ถึงหนึ่ง เนื่องจากผ่านฟังก์ชัน Sigmoid เมทริกซ์ถ่วงน้ำหนัก f_t นี้จะใช้ลดทอนหรือเสริมข้อมูลในความทรงจำระยะยาวดั้งเดิม (Previous ความทรงจำระยะยาว) เพื่อพิจารณาว่าข้อมูลระยะยาวที่มีอยู่นี้ควรถูกลบหรือบันทึกไว้

- Input Gate: มีหน้าที่ตัดสินใจว่าจะปรับข้อมูลใหม่สู่ข้อมูลระยะยาวที่เก็บในความทรงจำระยะยาวอย่างไรโดยอ้างอิงผลลัพธ์จากสมการ (7) ซึ่งรับข้อมูลจากอินพุทปัจจุบันที่ถูกถ่วงน้ำหนักด้วยข้อมูลจากความทรงจำระยะสั้นก่อนหน้าโดย $\tanh(\cdot)$ นี้มีค่าตั้งแต่ $-\infty$ ถึง ∞ เพื่อคำนวณว่าข้อมูลที่ถูกรับเข้ามาควรส่งผลอย่างไรกับข้อมูลในความทรงจำระยะยาว
- ข้อมูลขาออก Gate: มีหน้าที่ตัดสินใจว่าข้อมูลในความทรงจำระยะสั้นและข้อมูลขาเข้าปัจจุบันควรส่งออกไปเพื่อเป็นผลลัพธ์อย่างไรเพื่อสรุปผลลัพธ์เป็นข้อมูลขาออกซึ่งมีค่าในช่วงศูนย์ถึงหนึ่งโดยอ้างอิงตามสมการ (9)
- ความทรงจำระยะสั้น: ปรับข้อมูลความทรงจำระยะสั้นเมทริกซ์ถ่วงน้ำหนักซึ่งจะเป็นส่วนที่เก็บในความทรงจำระยะสั้นโดยข้อมูลในความทรงจำระยะสั้นจะเป็นผลลัพทย์ของ ข้อมูลขาออก Gate ที่ถูกถ่วงน้ำหนักด้วยข้อมูลความจำระยะยาวที่ผ่าน $\tanh(\cdot)$ ซึ่งต่าถ่วงน้ำหนักนี้มีค่าตั้งแต่ $-\infty$ ถึง ∞ ทั้งนี้อ้างอิงผลลัพธ์จากสมการที่สมการ (10)

2.2.3 Transformer

Transformer [10] เป็นโครงสร้างของโครงข่ายประสาทเทียมที่ถูกออกแบบมาเพื่อจัดการกับข้อมูลที่มีลำดับ Transformer ประกอบด้วย 2 ส่วนหลัก คือ ตัวเข้ารหัส (Encoder) และ ตัวถอดรหัส (Decoder) โดย ตัวเข้ารหัส (Encoder) จะรับข้อมูลไปสกัดคุณลักษณะขณะที่ตัวถอดรหัส (Decoder) จะเป็นส่วนที่ให้การทำนายข้อมูลขาออกโดยรับ Input ส่วนหนึ่งซึ่งเป็นคุณลักษณะมาจากตัวเข้ารหัส (Encoder) และอีกส่วนหนึ่งเป็น Input ที่ใช้ในการเริ่มต้นสร้างข้อมูลขาออกเช่นค่าสำหรับ Initialization หรืออาจจะเป็นค่าข้อมูลขาออกจาก Iteration ก่อนหน้า (Previous Predicted ข้อมูลขาออก)

Self-Attention. สิ่งที่ทำให้ Transformer มีความแตกต่างจาก RNN และ LSTM คือกระบวนการ Self-Attention ที่ถูกนำมาใช้เพื่อสกัด คุณลักษณะของ Deep Learning ในตัวเข้ารหัส (Encoder) และตัวถอดรหัส (Decoder) Self-Attention เป็นการทำ Matrix-Vector Product คล้ายกับการสกัดหาเบสิสที่สำคัญจากค่าความสัมพันธ์ภายในข้อมูล (Correlation matrix) ทั้งนี้ Self-Attention รับข้อมูลมาสามรูปแบบคือ Key, Query, และ Value โดยที่ข้อมูลทั้งสามมี Dimension คือ $\mathcal{R}^{N\times M}$ โดย N คือความยาวของข้อมูลลำดับ M คือจำนวนคุณลักษณะ Dimension ซึ่งการประมวณ Key, Query, และ Value ใน Self-Attention มี Concept ดังนี้

- Query เป็นข้อมูลที่เรารับมาเป็นอินพุตและเราต้องการหาว่า Query นั้น Match กับข้อมูลที่อ้างอิงคุณลักษณะที่ถูกเก็บไว้ใน Key
- Key และ Value เป็นข้อมูลอ้างอิงคุณลักษณะที่จะเป็นตัวเทียบกับ Query
- Key จะถูกใช้ร่วมกับ Query ในการคำนวน Softmax attention ซึ่งเป็น Matrix ในสองมิติ
- Value จะถูกใช้ในขั้นตอนสุดท้ายเพื่อหา Self-attention ซึ่งเป็น Matrix-Vector Product ระหว่าง Softmax attention

ซึ่งข้อมูลดังกล่าวสามารถคำนวณด้านล่าง

$$\text{Self-Attention}(Q,K,V) = \text{Softmax}\left(\frac{QK^T}{\sqrt{d}}\right)V \tag{11}$$

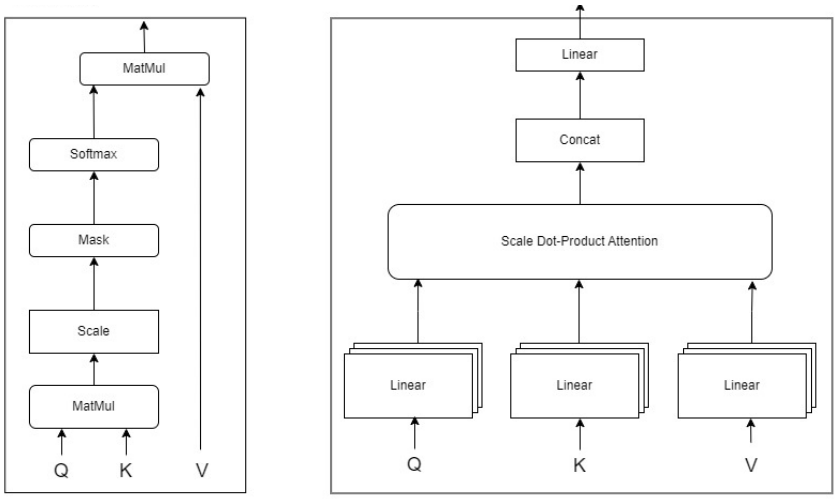
โดยที่ Q คือ Query; K คือ Key และ V คือ Value และ ให้ $\mathbf{x} \in \mathcal{R}^N$ ฟังก์ชัน Softmax นั้นนิยามดังนี้

$$\text{Softmax}(\mathbf{x}) = \frac{e^{x_i}}{\sum_i e^{x_i}} \tag{12}$$

Self-Attention Module มีโครงสร้าง ดังรูปที่ 3(a). ซึ่ง Self-Attention สามารถถูกปรับให้สามารถแบ่งข้อมูลเพื่อเพิ่ม Efficiency โดยการแบ่งข้อมูล 1 Batch ย่อยลงมาเป็นหลาย Heads และคำนวน Self-Attention ในทุกๆ Head แบบขนาน ดังนั้น สมการ (11) จะถูกปรับให้เป็นการคำนวนแบบ Multiple-Heads สามารถแสดงในสมการต่อไปนี้

$$\text{Multi-Head Attention} = \text{Concat}(\text{Self-Attention}_1(Q_1,K_1,V_1),\ldots,\text{Self-Attention}_N(Q_1,K_N,V_N)) \tag{13}$$

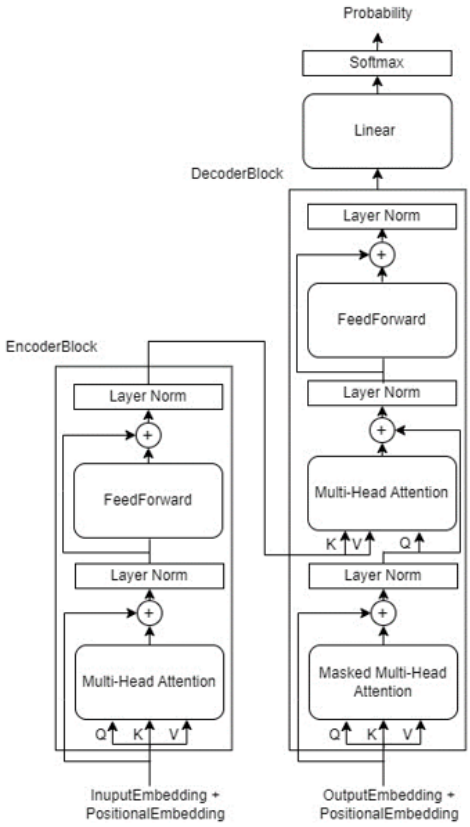
โครงสร้างของ Multi-Head Attention Module แสดงดัง รูปที่ 3(b).



(a) Self-Attention (b) Multi-Head Attention

รูปที่ 3: โครงสร้างของบล็อก (a) Self-Attention และ (b) Multi-Head Attention โดยที่ Self-Attention จะถูกใช้ในการทำ Scale Dot-Product Attention ใน Multi-Head Attention

จากโครงสร้างของ Transformer ซึ่งประกอบด้วยองค์ประกอบต่างๆ ภายใน Transformer ได้แก่ตัวเข้ารหัส (Encoder), ตัวถอดรหัส (Decoder), และ Multi-Head Attention Module ดังรูปที่ 4. จะเห็นว่า ตัวเข้ารหัส (Encoder) จะใช้ Multi-Head Attention Module ที่ขาเข้าของข้อมูลเพื่อสกัดคุณลักษณะที่ส่งให้กับตัวถอดรหัส (Decoder); ในขณะที่ตัวถอดรหัส (Decoder) จะใช้ Multi-Head Attention Module ในการคำนวณหาค่าความสัมพันธ์คุณลักษณะระหว่าง Input จากตัวเข้ารหัส (Encoder) และ Input ที่ใช้ในการสร้างข้อมูล ข้อมูลขาออกที่เป็นการทำนายข้อมูลในอนาคต



รูปที่ 4: โครงสร้างการทำงานของ Transformer

นอกจากนี้แต่ละส่วนภายใน Transformer มีลำดับการทำงานดังนี้

การทำงานในส่วนของ ตัวเข้ารหัส (Encoder):

- Input Embedding: ข้อมูลขาเข้าที่เป็นลำดับจะถูกแปลงไปเป็นรูปแบบเวกเตอร์โดยใช้การ Word Embedding เช่น หากข้อมูลขาเข้าอาจจะเป็นข้อความ (Text) โดย Word Embedding จะแปลงหน่วยย่อยของข้อความไปเป็นค่าใน คุณลักษณะ Space ซึ่งทำหน้าที่คล้าย Look-up table หากแต่ว่าการประมวลผลของ Input Embedding จะเป็นการเข้าคู่เป็นเวกเตอร์ด้วย Mapping Function

- Positional Embedding: เป็นการระบุตำแหน่งของข้อมูล (Position) จากข้อมูลลำดับให้อยู่ในรูปแบบเวกเตอร์ซึ่งผลลัพท์ของเวกเตอร์ที่ได้ จะถูกนำไปผสมกับคุณลักษณะของเวกเตอร์นั้นๆ ของข้อมูลจาก Input Embedding ด้วยการบวกเวกเตอร์ทั้งสองเข้าด้วยกัน

- Multi-Head Self-Attention: เป็นการเรียนรู้และหาความสัมพันธ์ของข้อมูลลำดับทั้งหมดโดยคำนวณจากสมการที่สมการ (11) มาเชื่อมต่อกันเป็นเมทริกผ่านสมการที่สมการ (13) และโครงสร้างของ Multi-Head Attention Module เป็นดังรูปที่ 3.

- Feed-Forward Neural Networks: นำเมทริกซ์ที่ได้จาก Multi-Head Self-Attention มาทำกระบวนการ Fully Connected Layer ด้วย RELU Activation

- Residual Connections เป็นการนำข้อมูลขาเข้าก่อนผ่านกระบวนการนำมาบวกเพิ่มเข้ากับข้อมูลขาออกที่ผ่านกระบวนการมาแล้วซึ่งจะช่วยลดปัญหาเกรเดียนส์เข้าใกล้ศูนย์ขณะฝึกสอนแบบจำลอง (Gradient Vanishing)
- Layer Norm เป็นการทำ Normalization ในแต่ละข้อมูลลำดับช่วยแก้ปัญหา Covariate Shift ซึ่งจะทำให้แบบจำลองไม่มีประสิทธิภาพหากข้อมูลมีการเปลี่ยนไปเพียงเล็กน้อยเช่นจากการทำ Data Augmentation

การทำงานในส่วนของตัวถอดรหัส (Decoder):

- ข้อมูลขาออก Embedding: ข้อมูลที่ ตัวถอดรหัส (Decoder) จะทำนายโดยแปลงเป็นเวกเตอร์โดยใช้ Word Embedding
- Masked Multi-Head Attention: เป็นกระบวนการ Multi-Head Attention
- ตัวเข้ารหัส (Encoder)-ตัวถอดรหัส (Decoder) Multi-Head Attention: เป็นกระบวนการ Multi-Head Attention โดยใช้ข้อมูล Keys และ Values ในส่วนของ ตัวเข้ารหัส (Encoder) และข้อมูล Queries ในตัวถอดรหัส (Decoder) และนำผลลัพธ์ที่ได้มารวมกัน
- ข้อมูลขาออก: ทำนายผลลัพธ์ของลำดับข้อมูลที่เป็นเวกเตอร์โดยใช้ Linear Transformation เพื่อพิจารณาผลลัพธ์ที่เป็นไปได้มากที่สุดโดยดูจากความน่าจะเป็นที่คำนวณผ่านฟังก์ชัน Softmax

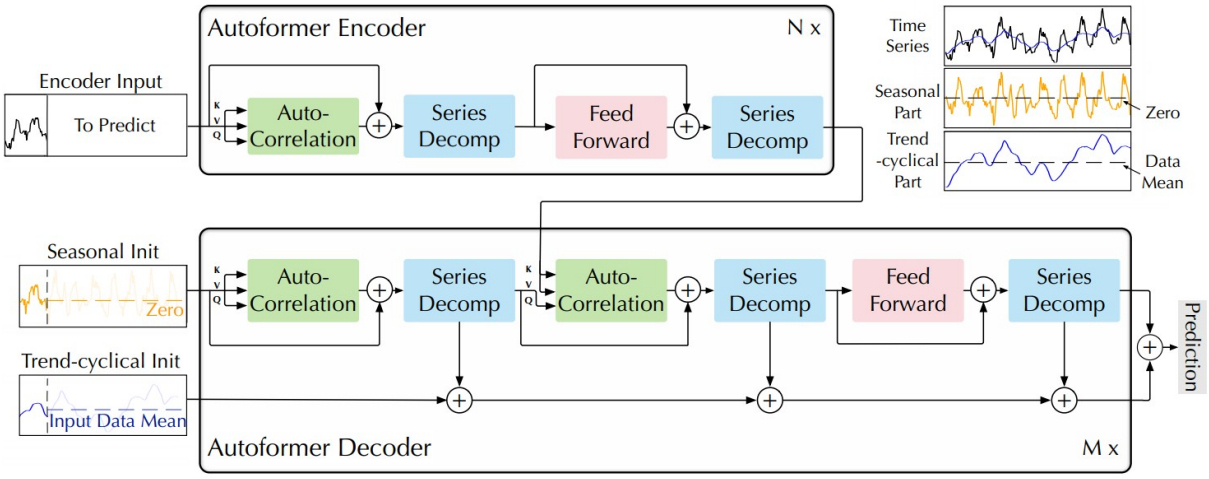
ลำดับการทำงาน

- ข้อมูลขาเข้าในตัวเข้ารหัส (Encoder Block) ถูกแปลงข้อมูลให้อยู่ในรูปเวกเตอร์และถูกเพิ่มข้อมูลระบุตำแหน่ง (Position embedding) แบ่งข้อมูลเป็น Queries, Keys และ Values โดยข้อมูลทั้งหมดจะถูกประมวลผลในบล็อก Multihead Attention นำไปคำนวณผ่านสมการ (11) และ สมการ (13) นำผลลัพธ์ไปคำนวณผ่าน Feed Forward รวมถึงถูก Normalization และใช้ Residual Connection ซึ่งผลลัพธ์ที่ได้จากตัวเข้ารหัส (Encoder) Block จะนำข้อมูล Keys และ Values ไปใช้ต่อไปในตัวถอดรหัส (Decoder Block)
- ข้อมูลขาเข้าในตัวถอดรหัส (Decoder Block) ถูกแปลงข้อมูลให้อยู่ในรูปแบบเวกเตอร์และถูกเพิ่มข้อมูลระบุตำแหน่งจากนั้นข้อมูลจะถูกประมวลผลในบล็อก Multihead Attention นำไปคำนวณผ่านสมการ (11) และ สมการ (13) แต่มีความแตกต่างจากตัวเข้ารหัส (Encoder) คือมีการปิดบังข้อมูลในอนาคตเพื่อให้แบบจำลองทำนายข้อมูลในอนาคตจากนั้นจึงไปประมวลผลต่อในตัวเข้ารหัส (Encoder) ตัวถอดรหัส (Decoder) Multi Head Attention ซึ่งเป็นการนำข้อมูลลำดับจากตัวเข้ารหัส (Encoder) Block มาคำนวณแล้วนำผลลัธ์ไปประมวลผลผ่าน Feed Forward รวมถึงถูก Normalization และ ใช้ Residual Connection
- นำข้อมูลทั้งหมดไปผ่าน Linear Block และคำนวณความน่าจะเป็นของลำดับข้อมูลในอนาคตด้วยฟังก์ชัน Softmax

2.2.4 Autoformer

Autoformer [3] เป็นการประยุกต์แบบจำลอง Transformer โดยที่ Autoformer มีขั้นตอนการทำงานคล้ายกับ Transformer คือแบบจำลองประกอบด้วยตัวเข้ารหัส (Encoder) และตัวถอดรหัส (Decoder) โดยที่ตัวเข้ารหัส (Encoder) จะทำการสกัดคุณลักษณะของข้อมูลอินพุตขาเข้าขณะที่ตัวถอดรหัส (Decoder) จะใช้เพื่อทำนายข้อมูลโดยที่ใน Autoformer จะใช้ Auto-Correlation Module แทน Multi-Head Attention Model ในการสกัดคุณลักษณะ ทั้งนี้ผลการ

สกัดคุณลักษณะของ Auto-Correlation Module จะเป็นลักษณะเป็นฤดูกาล (Seasonal) และลักษณะ Trend-Cyclical โดยที่ Trend-Cyclical จะเป็นสัญญาณที่มีความเป็นคาบแต่ Seasonalจะเป็นข้อมูลอนุกรมเวลาที่ถูกแยก Trend-Cyclical ออก รูปที่ 5. แสดงโครงสร้างของ Autoformer

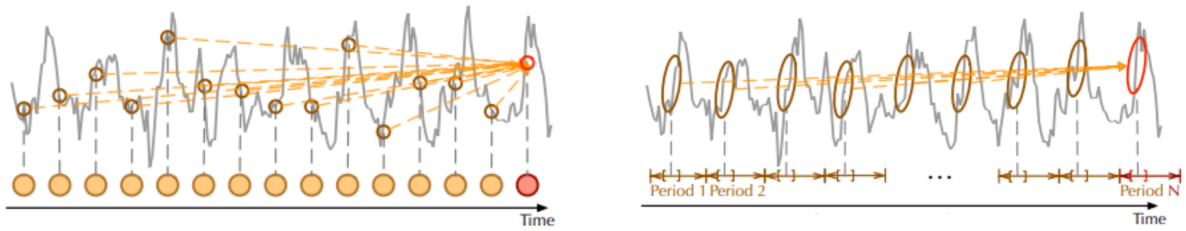


รูปที่ 5: โครงสร้างแบบจำลอง Autoformer[1]

Auto-Correlation Module มีลักษณะการทำงานตามหลักการของ Stochastic Process ที่ Auto-Correlation จะเป็น function ที่ขึ้นกับช่วงความต่างทางเวลา τ ซึ่งการคำนวณ Attention ใน Auto-Correlation Module จะเป็นผลของการทำ Dot Product ที่เป็นการบวกของข้อมูลทั้งหมดที่ถูกเลือกเข้ามาในกรอบเวลาของข้อมูล ซึ่งเป็นการคำนวณแบบ Series-wise mechanism ในทางตรงกันข้าม Self-Attention ถูกออกแบบเพื่อใช้กับเวกเตอร์ลำดับของข้อมูลใดๆ เป็นการคำนวณ Correlation ระหว่างแต่ละส่วนในข้อมูลเวกเตอร์ซึ่งเป็นการคำนวณแบบ Point-wise mechanism โดยจะทำการบวกแต่ละจุดข้อมูล ความแตกต่างระหว่าง Self-Attention และ Auto-Correlation แสดงได้ดัง รูปที่ 6(a). และรูปที่ 6(b).

ดังนั้นหากนำ Transformer มาประยุกต์โดยตรงในการทำ Auto-Correlation ทางอนุกรมเวลาในลักษณะนี้จะให้ Time Complexity ซึ่งก็คือ Quadratic Complexity กับของความยาวของข้อมูลลำดับ L (Sequence Length) หรือ $O(L^2)$ โดยที่ Time Complexity นี้คือค่าระยะเวลาที่แย่ที่สุดที่คอมพิวเตอร์ต้องจ่ายให้กับความซับซ้อนของขั้นตอนวิธี (Algorithm) ซึ่งทำให้ไม่เหมาะสมสำหรับการทำนายข้อมูลอนุกรมเวลาระยะยาว (Long-term time series) แต่ใน Autoformer นี้ผู้คิดค้นมาสามารถจัดสรรค์ข้อมูลโดยลด Time complexity เป็น $O(L \log L)$ ได้โดยการจำกัดจำนวน Auto-Correlation Coefficients และการใช้ประโยชน์จาก Fourier Transformation ของสัญญาณคาบในการคำนวณ Auto-Correlation ซึ่งสามารถเอาท์พุตครั้งเดียวเป็น Series-wise โดยไม่ต้องทำการวนลูปคำนวน Auto-Correlation

[1]นำรูปมากจาก [3]



(a) Self-Attention (b) Multi-Head Attention

รูปที่ 6: ความแตกต่างระหว่าง (a) Self-Attention ซึ่งอยู่ในลักษณะ Point-wise และ (b) Auto-Correlation ซึ่งอยู่ในลักษณะ Series-wise[2]

โครงสร้างของบล็อก Auto-Correlation Module ประกอบด้วย Auto-Correlation และ Time delay aggregation ดังรูปที่ 7(a). และรูปที่ 7(b). มีขั้นตอนการทำงานดังนี้

1. **Auto-correlation** โดยทั่วไปการคำนวณ Auto-correlation จะทำโดยการคำนวณดังสมการด้านล่างซึ่งต้องทำการวนลูปคำนวน

$$R_{Q,K}(\tau) = \lim_{L\to\infty} \sum_{t=1}^{L} Q_t K_{t-\tau} \tag{14}$$

แต่เนื่องจากมีการใช้กรอบเวลา (Window) ซึ่งมองสัญญาณเป็นสัญญาณคาบ ดังนั้นจึงสามารถหา Auto-Correlation ระหว่างสัญญาณ Query (Q) และ Key (K) โดยใช้ Fourier Transform:

$$R_{Q,K}(\tau) = \mathcal{F}^{-1}\left(S_{Q,K}\right) \tag{15}$$

$$S_{Q,K} = \mathcal{F}\left(Q\right)\overline{\mathcal{F}\left(K\right)} \tag{16}$$

ในกรณีนี้ $Q, K \in R^{L\times d}$ เมื่อ L คือความยาวของข้อมูลลำดับและ d คือคุณลักษณะ Dimension $\mathcal{F}(\cdot)$ คือ Fourier Transformation ของสัญญาณคาบที่ถูกกำหนดโดยกรอบเวลา (Window Size) L และ $\overline{\mathcal{F}(\cdot)}$ คือสังยุคของ Fourier ซึ่งการตั้งต้นการคำนวณแบบนี้จะทำให้ Time Complexity ของการคำนวณโดยรวมลดลงเป็น $O(L\log L)$ หลังจากนั้น $R_{Q,K}(\tau)$ และ Value (หรือ V) จะถูกนำไปใช้ในการกระบวนการ Time-Delay Aggregation

2. **Time-Delay Aggregation** มีขั้นตอนการทำงานดังนี้

   - ทำการหาช่วง Lagging ซึ่งเป็น Period Length ที่ให้ Auto-Correlation ที่สูงที่สุดมา k ช่วงโดยใช้คำสั่ง Topk ดังสมการ (17)

$$\tau_1, ..., \tau_k = \underset{\tau\in\{1,...,L\}}{\arg\,\text{Topk}}(R_{Q,K}(\tau)), \tag{17}$$

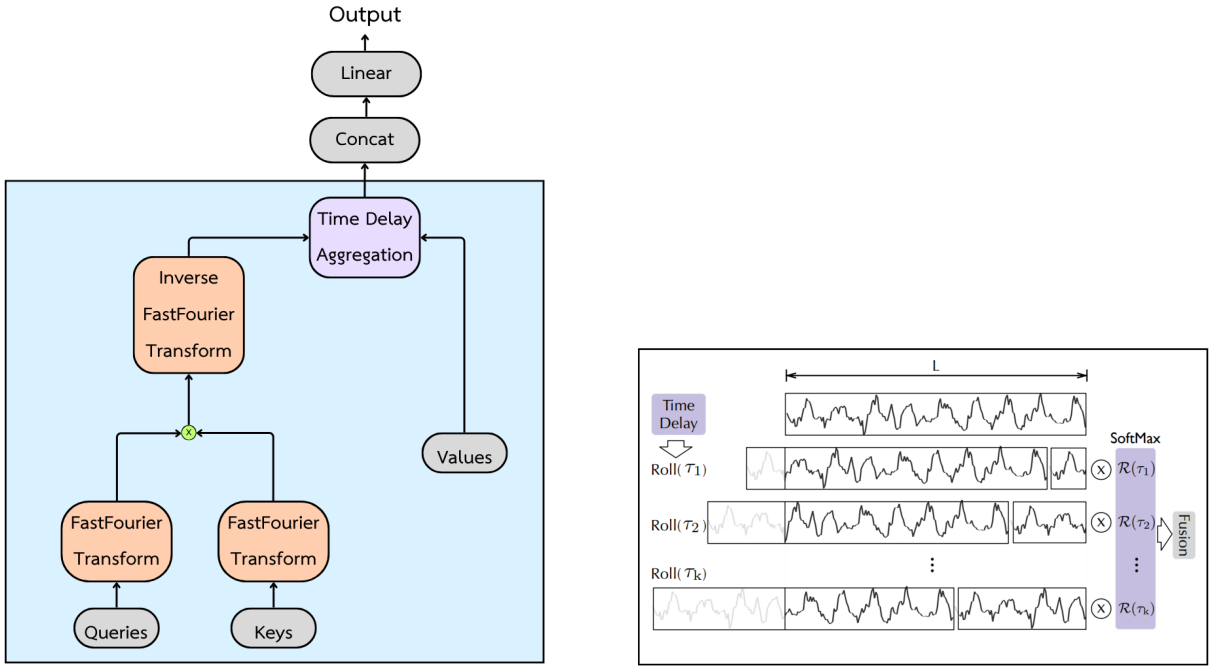
[2]นำรูปมากจาก [3]

- ส่วนที่สองคือคำนวณค่า Weight ของ Auto-Correlation ช่วงระหว่างข้อมูล Queries และ Keys ที่ถูกเลือกมา k ช่วงโดยใช้ ฟังก์ชัน Softmax ดังสมการ (18)

$$\hat{R}_{Q,K}(\tau_1), \ldots, \hat{R}_{Q,K}(\tau_k) = \text{Softmax}(R_{Q,K}(\tau_1), \ldots, R_{Q,K}(\tau_k)) \tag{18}$$

- และสุดท้ายเป็นการหา Time-Delay Aggregation Score ดังสมการ (19) ซึ่งเป็นค่าที่คำนวณระหว่าง Value ที่ถูกทำ Shift-Delay ทาง Time และ Weight ของ Auto-Correlation ($\hat{R}_{Q,K}$) โดยที่ Value ที่ถูกทำ Shift-Delay นี้จะเป็นข้อมูลลำดับที่ถูกทำการ Rolling ซึ่งจะเป็นการ Shift สัญญาณทาง Time ไปทางด้านซ้ายไปตามการ Period length ที่เลือกมา k พจน์ (จากสมการ (17)) ดังรูปที่ 5.

$$\text{Time-Delay Aggregation } (Q, K, V) = \sum_{i=1}^{k} \text{Roll } (V, \tau_i)\hat{R}_{Q,K}(\tau_i) \tag{19}$$



(a) Auto-Correlation (b) Time Delay Aggregation

รูปที่ 7: ความแตกต่างระหว่าง (a) Self-Attention ซึ่งอยู่ในลักษณะ Point-wise และ (b) Auto-Correlation ซึ่งอยู่ในลักษณะ Series-wise[3]

องค์ประกอบที่สำคัญในตัวเข้ารหัส (Encoder) และตัวถอดรหัส (Decoder):

- ตัวเข้ารหัส (Encoder): อินพุตเป็นข้อมูลลำดับขาเข้าที่เป็นลำดับถูกแปลงจากรูปแบบข้อความเป็นรูปแบบเวกเตอร์โดยใช้ Word Embedding รวมถึงแปลงข้อมูลเวลาด้วย Temporal Embedding

- Auto-Correlation: เป็นกระบวนการที่คำนวณความสัมพันธ์ระหว่างข้อมูล

[3]นำรูปมากจาก [3]

- Series Decomposition: เป็นการแยกส่วนที่เป็นข้อมูลแนวโน้ม (Trend) และความแปรผันตามฤดูกาล (Seasonality) โดยใช้ขั้นตอนในการแยกให้น้อยที่สุดเนื่องจากแบบจำลองใช้ลำดับของข้อมูลในการคำนวณความสัมพันธ์จึงต้องการข้อมูลที่มีการเปลี่ยนแปลงจากเดิมน้อยที่สุด

- Feed-Forward Neural Networks: นำข้อมูลที่เป็น Trend และ Seasonality มาผ่านกระบวนการ Fully connected layer ด้วยฟังก์ชัน RELU ($f(x) = \max(x, 0)$)

- ตัวถอดรหัส (Decoder): ใช้อินพุตข้อมูลจากตัวเข้ารหัส (Encoder)ที่ถูกแยกข้อมูลที่มีแนวโน้มและมีความแปรผันตามฤดูกาลโดยใช้ข้อมูลส่วนที่สองในการประมวลผลผ่านบล็อก Autocorrelation, Series Decomposition และ Feedforward neural network โดยข้อมูลในส่วนแรกจะใช้ในการทำ Residual connection เพื่อแก้ปัญหาเกรเดียนส์เข้าใกล้ศูนย์ขณะฝึกสอนแบบจำลอง (Gradient Vanishing)

ลำดับการทำงาน

- ข้อมูลขาเข้าในตัวเข้ารหัส (Encoder) ถูกแปลงข้อมูลลำดับให้อยู่ในรูปเวกเตอร์และถูกเพิ่มข้อมูลระบุเวลา (Temporal Embedding) แบ่งข้อมูลเป็น Queries, Keys และ Values โดยข้อมูลทั้งหมดจะถูกประมวลผลในบล็อก Autocorrelation นำไปคำนวณผ่านสมการ (18) และ สมการ (19) แยกข้อมูลที่มีแนวโน้มและมีความแปรผันตามฤดูกาลผ่านบล็อก Series decomposition นำผลลัพธ์ไปผ่าน Feedforward Neural Network และหลังจากประมวลผลเสร็จให้นำข้อมูล Keys และ Values จากตัวเข้ารหัส (Encoder) ไปใช้ต่อไปในตัวถอดรหัส (Decoder)

- ข้อมูลขาเข้าในตัวถอดรหัส (Decoder) จะเป็นข้อมูลเดียวกับตัวเข้ารหัส (Encoder) เพียงแต่ถูกใช้คำนวณเฉพาะส่วน Seasonality มาประมวลผลผ่านบล็อก Autocorrelation และ Series decomposition โดยจะนำข้อมูล Keys และ Values จากกระบวนการในตัวเข้ารหัส (Encoder) และ Queries จากตัวถอดรหัส (Decoder) มาประมลผลอีกครั้งผ่าน Autocorrelation และ Series decomposition นำผลลัพธ์ไปผ่าน Feedforward Neural Network ผลลัพธ์สุดท้ายจะนำข้อมูลส่วนที่แนวโน้มจากข้อมูลขาเข้าเพิ่มเข้าไปและได้ผลลัพธ์เป็นลำดับข้อมูลในอนาคต

2.2.5 Informer

Informer [4] เป็นการประยุกต์แบบจำลอง Transformer โดยยังคงเป็นแบบจำลองแบบตัวเข้ารหัสและตัวถอดรหัสแต่สิ่งที่แตกต่างกันคือเปลี่ยนแปลงกระบวนการ Scale dot product self attention เป็น ProbSparse self-attention โดยจากเดิมจะเป็นการคำนวณ self-attention ของ q_i กับทุกๆค่าของ k_j โดยจะถูกคำนวณผ่านฟังก์ชัน Softmax ดังสมการ

$$p(k_j|q_i) = \frac{\exp\left(\frac{q_i k_j^T}{\sqrt{d}}\right)}{\sum_l \exp\left(\frac{q_i k_l^T}{\sqrt{d}}\right)} \tag{20}$$

จากนั้นจึงนำมาคูณด้วยค่า V สำหรับกรณีแบบจำลอง Transformer [11] แต่ในกรณีของ ProbSparse attention จะมีการวัดความเป็น Sparsity ของ q_i และ k_j ด้วยวิธีการลู่เข้าแบบ Kullback-Leibler ระหว่างค่า Queries และทุก ๆ ค่า Keys $p(k_j|q_i)$ โดยมีสมมติฐานว่าแต่ละ q_i มีการกระจายตัวแบบยูนิฟอร์ม (Uniform distribution) โดยผลลัพธ์ที่ได้จากวิธีนี้จะเรียกว่า Sparsity measurement ซึ่งแสดงเป็นสมการ

$$M(q_i, K) = \ln \sum_{j=1}^{L_K} \exp\left(\frac{q_i k_j^T}{\sqrt{d}}\right) - \frac{1}{L_K} \sum_{j=1}^{L_K} \frac{q_i k_j^T}{\sqrt{d}} \tag{21}$$

- q_i คือ query vector ที่ i
- k_j คือ key vector ที่ j
- d คือมิติของ query และ key
- L_K คือความยาวของข้อมูลอนุกรมเวลา

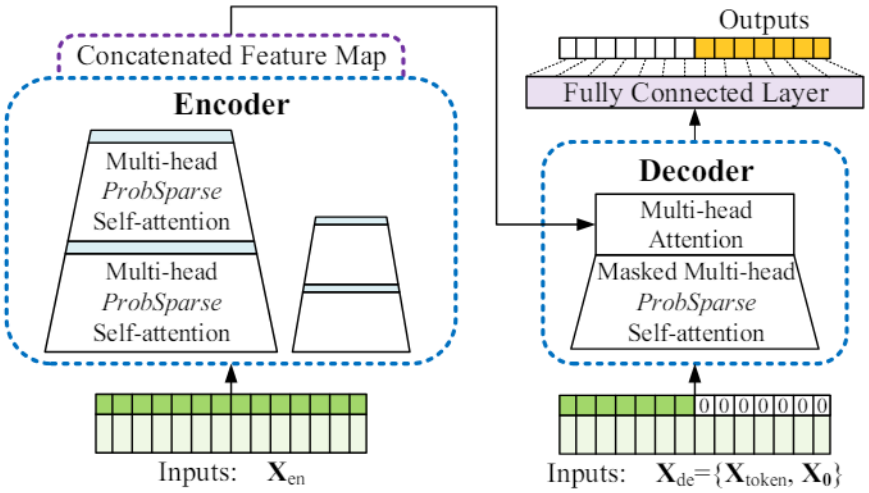
หากค่า Sparsity measurement มีค่าสูงแสดงถึงโอกาสที่ Queries ที่ตำแหน่ง i มี Keys บางตัวที่มีความเกี่ยวข้องกับเวคเตอร์ q_i แต่ในการทดลองของรายงานเล่มนี้จะใช้วิธีการประมาณค่าด้วยวิธี Max-Mean measurement ซึ่งอ้างอิงมาจากบทตั้งที่ 1 [4] ซึ่งแสดงได้ดังสมการ

$$M(q_i, K) = \max_j \frac{q_i k_j^T}{\sqrt{d}} - \frac{1}{L_K} \sum_{j=1}^{L_K} \frac{q_i k_j^T}{\sqrt{d}} \tag{22}$$

และจะมีเกณฑ์ที่ใช้เลือกจำนวนของ q_i ที่มีค่า Sparsity measurement สูงที่สุดจำนวนนึงซึ่งกำหนดได้เองทำให้แบบจำลองใช้การประมวลผลที่น้อยกว่าแบบจำลองดั้งเดิมเหลือเพียง $O(L \times \ln L)$ แสดง ProbSparse self-attention ได้ดังสมการ

$$A(\bar{Q}, K, V) = \text{Softmax}\left(\frac{\bar{Q}K^T}{\sqrt{d}}\right) V \tag{23}$$

เมื่อ $\bar{Q}$ คือเซตของ Queries ที่ถูกเลือกมาจำนวน $u = c \cdot lnL_Q$ ที่คัดเลือกตามค่า Sparsity measurement โดยค่า c เป็นค่าคงที่สามารถกำหนดได้เอง



รูปที่ 8: ลำดับการทำงานของแบบจำลอง Informer [4]

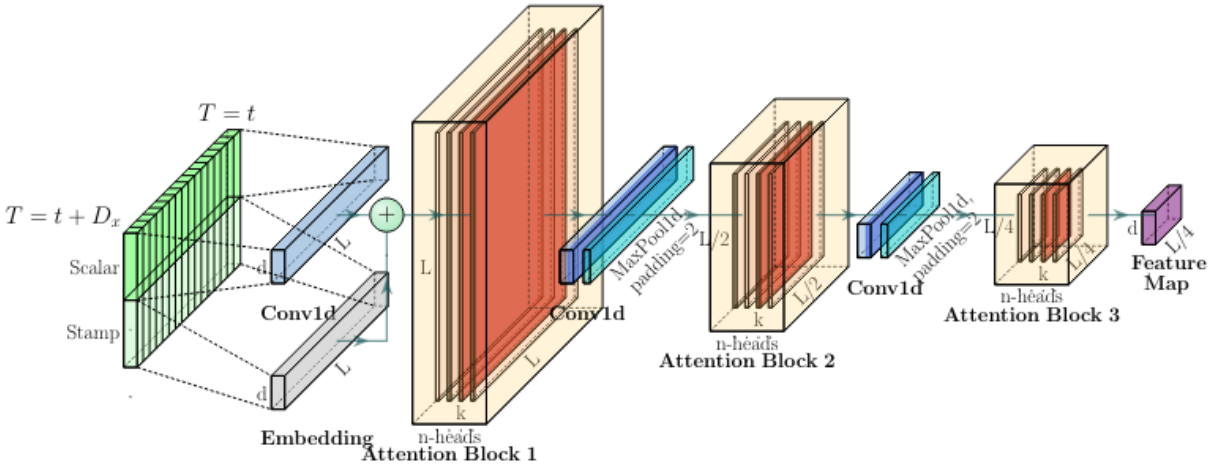
ลำดับการทำงาน

- ในส่วนของ Encoder : นำข้อมูลขาเข้าผ่านไปยัง Multi-head ProbSparse self-attention จากนั้นจึงไปผ่านกระบวนการ Self-attention distilling ระหว่าง Self-attention blocks ตามสมการ

$$X_t^{j+1} = \text{MaxPool}(\text{ELU}(\text{Conv1d}([X_t^j]_{AB}))) \tag{24}$$

โดย j หมายถึงลำดับของเลเยอร์และ $[\cdot]_{AB}$ หมายถึง attention block โดยเพื่อทำการเลือก Feature map ไปใช้ต่อในส่วนของ Decoder

- ในส่วนของ Decoder : ลักษณะเด่นของตัวถอดรหัสนี้คือจะทำนายข้อมูลลำดับเวลาทั้งหมดไม่ใช่ทีละตำแหน่งโดยข้อมูลขาเข้าจากฝั่งตัวถอดรหัสจะนำข้อมูลในอดีตย้อนหลังมาทำการ Tokenization นำมารวมเข้ากับเวคเตอร์ศูนย์ขนาดตามความยาวที่ต้องการทำนาย ซึ่งเพิ่มประสิทธิภาพการทำนายลำดับข้อมูลที่ยาวมากขึ้น



รูปที่ 9: กระบวนการ Self-attention distilling ใน Encoder block [4]

2.2.6 Linear, DLinear และ NLinear

แม้ Transformer จะเป็นโมเดลที่ได้รับความยอมรับในด้านประสิทธิภาพอย่างยิ่งในช่วงปี 2017-2023 ในการทำนายอนุกรมเวลางานวิจัยล่าสุด [5] ส่วนสำคัญที่ทำให้ Transformer ยังคงให้ผลลัพท์ได้ดีคือ

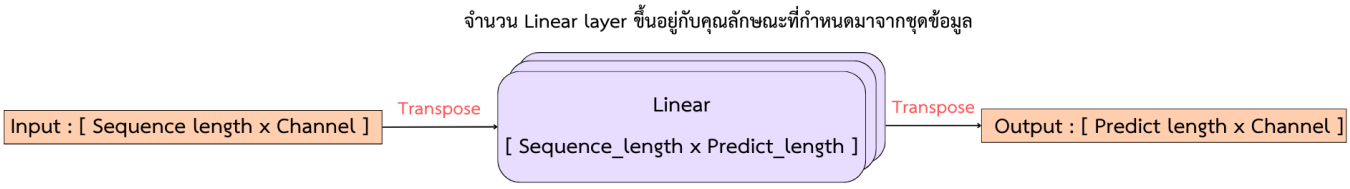
- การประมวณผลในลักษณะ Direct Multi-step Forecasting
- การเก็บข้อมูล (Aggregate) ใน Self-Attention ของ Transformer เป็นแบบไม่มีลำดับ (Anti-Order) คือสาเหตุที่ทำให้ Transformer สามารถประมวลข้อมูลเชิงสัญลักษณ์ได้ดีแต่ประมวลข้อมูลที่มีการเปลี่ยนแปลงตามเวลา (Temporal Change) ได้แย่เนื่องจากสูญเสียจากจัดลำดับของข้อมูลไป
- นอกจากนี้โดยทั่วไปบทความที่ตีพิมพ์จะเทียบ Transformer กับวิธีทางสถิติแบบพื้นฐานที่ทำการทำนายในลักษณะ Iterated Multi-step Forecastiong เท่านั้น

[3]นำรูปมากจาก [4]

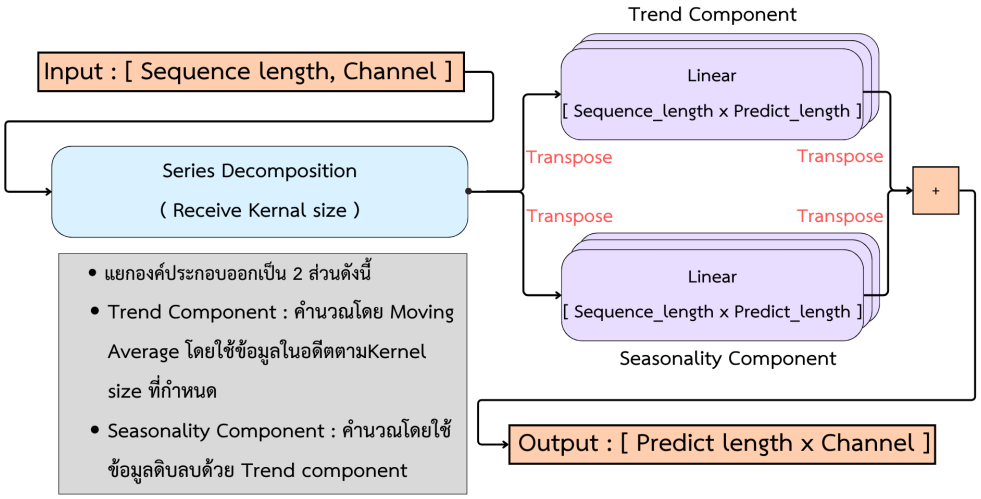
ดังนั้นจึงเสนอให้แก้ไขโดยใช้เพียง Linear Layer เป็น Layer สุดท้ายเพื่อใช้ในการประมวณผลในลักษณะ Direct Multi-step Forecasting ในการทำนายข้อมูลอนุกรมเวลาและแทน Self-Attention Model ใน Encoder หรือ Decoder ของ Transformer ด้วย Linear Layer

ในการศึกษานี้ ผู้เขียน [5] ได้เสนอโมเดลที่มีความซับซ้อนน้อยโดยผสม Linear Layer กับเทคนิคการแยกคุณลักษณะของ AutoFormer [3] โดยที่ยังคงส่วนของการแยก Trend และ Seasonal เนื่องจากผู้เขียน [5] สันนิฐานว่าเป็นส่วนสำคัญในการเพิ่มศักยภาพของแบบจำลองทั้งนี้คุณลักษณะของแต่ละแบบจำลองมีความแตกต่างดังนี้คือ

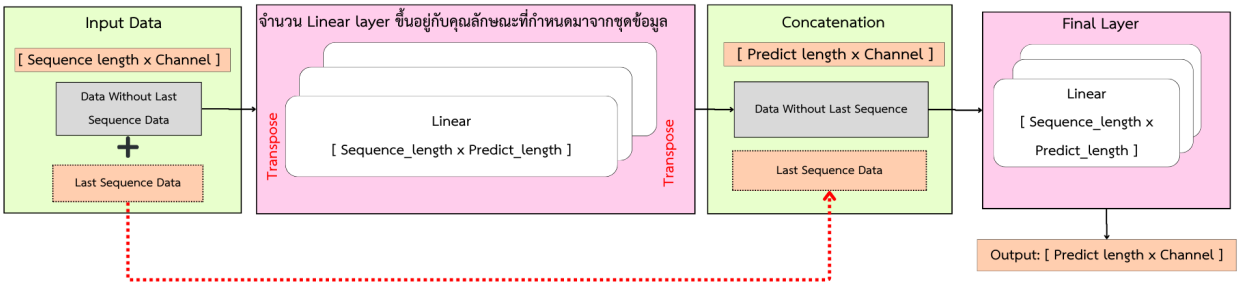
- **Linear Layer** เป็นดังสมการสมการ (25) จะทำนายข้อมูลในอนาคตโดยระยะเวลาที่ทำนายตามจำนวน Predict length โดยค่า Predict length เป็นค่าที่กำหนดเอง (Hyper-parameter)
- **DLinear Layer** : เป็นการสร้างแบบจำลองโดยการทำ Time Series decomposition เพื่อแยก Trend และ Seasonal Features ออกจากกัน (แบบเดียวกับของ Autoformer) โดยแยก Trend Features ด้วยการใช้ Moving Average และ Seasonal Features โดยการนำข้อมูลอนุกรมเวลามาหักเอา Trend Features ออกหลังจากนั้น Linear Layer สองชุดจะถูกนำมาใช้ในการทำนายข้อมูลอนาคต ดังรูปที่ 13. โดยชุดแรก (บน) จะถอดรหัส (Decode) ข้อมูลจาก Trend Features และโดยชุดที่สอง (ล่าง) จะถอดรหัสคุณลักษณะข้อมูลจาก Seasonality และสุดท้ายจะผลการถอดรหัสมารวมกันเป็นผลลัพธ์โดยจำนวนชั้นของ Linear ยังคงอ้างอิงจากจำนวน Channel
- **NLinear Layer** : เป็นการสร้างแบบจำลองโดยการหั่นสัญญาณเป็นสองส่วน โดยส่วนแรกจะถูกนำไปใช้เป็นข้อมูลขาเข้า Linear Layer เพื่อสร้างการทำนาย โดยผลการทำนายนี้จะเอาไปต่อกับข้อมูลอีกส่วนที่ถูกแยกออกมา แล้วนำสัญญาณทั้งหมดไปผ่าน Layer ชั้นสุดท้ายเพื่อทำนายข้อมูลอนาคตดังรูปที่ 12.



รูปที่ 10: การทำงานของแบบจำลอง Linear



รูปที่ 11: การทำงานของแบบจำลอง DLinear



รูปที่ 12: การทำงานของแบบจำลอง NLinear

$$Output = W_{X_i} \times X_i \tag{25}$$

โดยที่ $W_{X(\cdot)}$ และ $X(\cdot)$ คือเมทริกซ์ถ่วงน้ำหนักและข้อมูลขาเข้าของแต่ละ Channel ตามลำดับ

- Sequence length หมายถึงระยะเวลาของข้อมูลที่เป็นส่วนเริ่มต้นที่อ้างอิงกับชุดข้อมูล
- Predict length หมายถึงระยะเวลาของข้อมูลที่ต้องการทำนายโดยจะเป็นส่วนที่ทำนายหลังจาก Sequence length อ้างอิงจากชุดข้อมูล
- Channel หมายถึงจำนวนคุณลักษณะที่กำหนดจากชุดข้อมูลที่ต้องการทำนาย

2.2.7 PatchTST

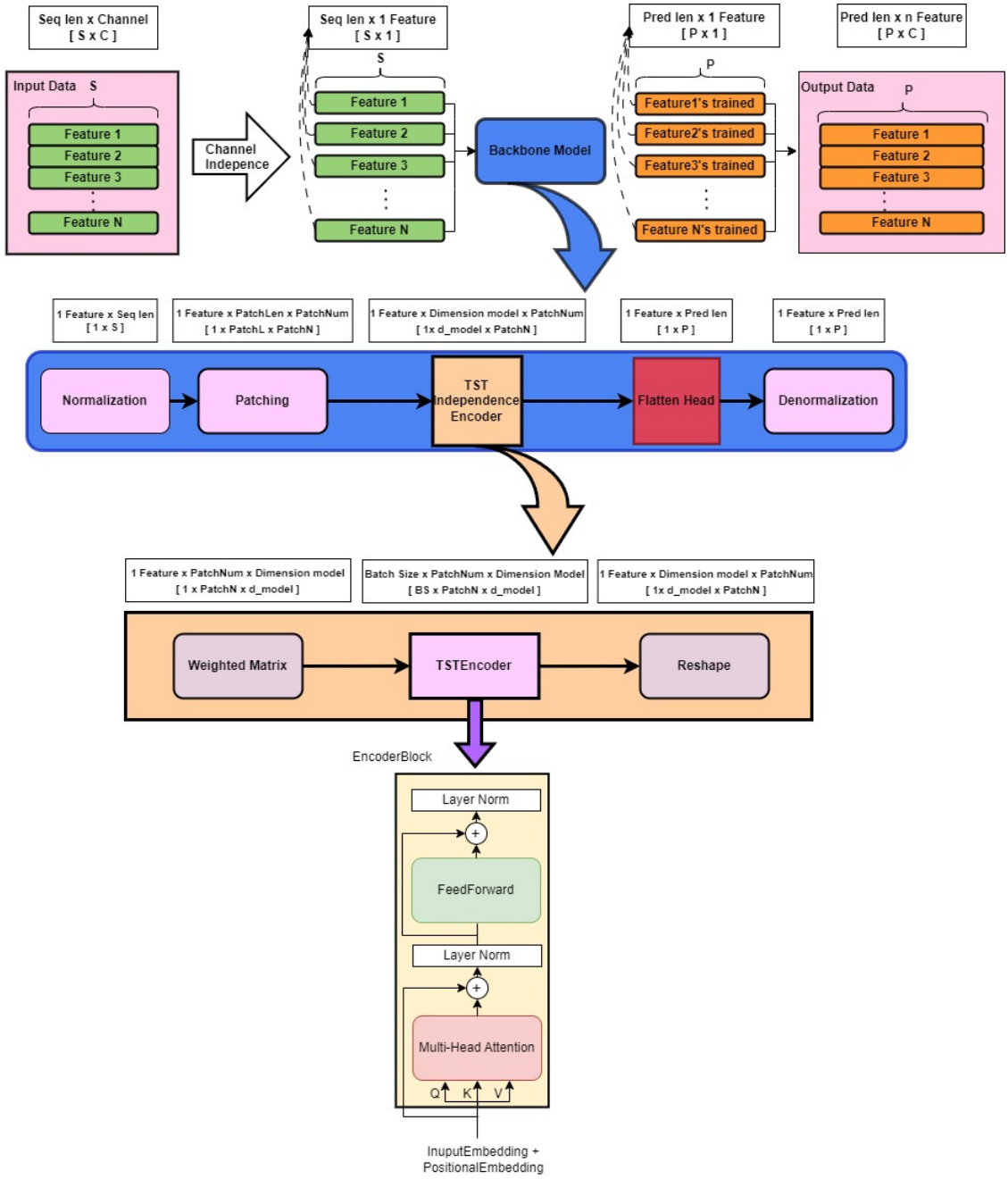
PatchTST [6] ใช้แนวคิดที่ต่อยอดมาจากโมเดล Linear, DLinear และ NLinear [5] บนสมมุติฐานที่ว่าหากต้องการทำนายข้อมูลอนุกรมเวลาที่ iterations ห่างจากเวลาปัจจุบันมาก ๆ จะมีความสัมพันธ์ของข้อมูลอนุกรมเวลาระหว่างข้อมูลอนุกรมเวลาอื่นน้อยซึ่งส่งผลให้แต่ละ feature มีความเป็นอิสระต่อกันทำให้ใช้แต่ละ feature สำหรับการเรียนรู้ของแบบจำลอง

ใช้ features แยกกันแต่ยังคงเรียนรู้แบบจำลองด้วยแบบจำลองเดียวกันซึ่งเรียกว่า Backbone model โดยในรายงานเล่มนี้จะใช้แบบจำลอง Encoder เพียงอย่างเดียวจากแบบจำลอง Transformer ซึ่งสามารถทดลองกับ Encoder block ของแบบจำลอง Autoformer [3] และ Informer [4] เพิ่มเติมได้และในส่วนของการทำ Embedding ข้อมูลอนุกรมเวลาจะใช้เทคนิคที่เรียกว่า Patching โดยการนำข้อมูลอนุกรมเวลามาทำการ Tokenization จากนั้นจึงนำไปเรียนรู้กับแบบจำลอง Backbone แบบจำลองนี้เป็นการพัฒนาต่อยอดจากแบบจำลอง [5] ที่ไม่ได้นำความสัมพันธ์ระหว่างข้อมูลใน Time-steps ใกล้เคียงกันและยังส่งผลให้ความซับซ้อนของโมเดล (Complexity) ของแบบจำลอง Transformer มีขนาดลดลงเหลือ $O(L^2/P^2)$ โดย L คือความยาวของข้อมูลอนุกรมเวลาขาเข้าและ P คือขนาดของการ Patch องค์ประกอบที่สำคัญของแบบจำลองนี้คือ

- **Channel independence** เป็นสมมติฐานที่ว่าแต่ละข้อมูลอนุกรมเวลาที่ต่างกันนำมาแยกองค์ประกอบเป็นข้อมูลลำดับอนุกรมเวลาย่อย (Sub sequence time series data) และนำมาเรียนรู้ผ่านแบบจำลองเดียวกัน เนื่องจาก features แต่ละชนิดจะมี Attention maps ที่ต่างกันทำให้เสมือนมองว่าเป็นการทำ Univariate time series forecasting บนแต่ละ features รวมถึงสามารถออกแบบ Spatio-temporal correlation ระหว่างข้อมูลอนุกรมเวลา
- **Patching** เป็นการทำ Tokenization บนข้อมูลอนุกรมเวลาซึ่งจะแปลง เป็นการแปลงจำนวน Sequence Length ให้ขยายออกเป็นเมทริกซ์ขนาด [$PatchLength\ x\ PatchNumber$] มีจุดประสงค์คือการแบ่งสัดส่วนของข้อมูลอนุกรมเวลาให้ยังคงสามารถตรวจจับความสัมพันธ์ของข้อมูลช่วง Time-steps ในลำดับก่อนหน้าได้

ลำดับการทำงาน

- แบ่งข้อมูลขาเข้าออกเป็นข้อมูลอนุกรมเวลาย่อยตามจำนวน features นำข้อมูลแต่ละส่วนมาทำการ Reversible Instance Normalization เพื่อลดปัญหา Distribution Shift [12] และนำไปทำการ Patching โดยจะแปลงข้อมูลจากความยาวของข้อมูลขาเข้าเป็นข้อมูล 2 มิติระหว่างความยาวของ Patch และจำนวน Patch
- นำข้อมูลที่ถูก Patching แล้วนำไปเรียนรู้ใน Backbone ใน TSTEncoder ซึ่งเป็น Encoder block จาก Transformer
- ปรับขนาดของข้อมูลจนได้เป็นผลลัพธ์จากการทำนายข้อมูลและทำการ Denormalize กลับไปยังข้อมูลการกระจายแบบเดิมและรวบรวมข้อมูลแต่ละ features ให้กลับเป็นรูปแบบเดิมกับข้อมูลขาเข้า



รูปที่ 13: โครงสร้างและลำดับการทำงานของแบบจำลอง PatchTST

2.3 การวัดประสิทธิภาพ

การวัดมีหลากหลายวิธีโดยในที่นี้จะขอใช้สัญลักษณ์ y เป็นค่าที่วัดได้จริง, $\hat{y}$ เป็นค่าที่ได้จากการทำนายด้วยแบบจำลองและ N เป็นจำนวนข้อมูลทั้งหมดซึ่งใช้วิธีการวัดประสิทธิภาพดังนี้

2.3.1 Mean Absolute Error (MAE)

$$MAE = \frac{1}{N}\sum_{i=1}^{N}|\hat{y}_i - y_i| \tag{26}$$

เป็นการหาผลรวมของค่าสัมบูรณ์ของผลต่างระหว่างค่าที่แท้จริงและค่าที่ทำนายมาคำนวณค่าเฉลี่ย

2.3.2 Mean Square Error (MSE)

$$MSE = \frac{1}{N}\sum_{i=1}^{N}(\hat{y}_i - y_i)^2 \tag{27}$$

เป็นการหาค่าเฉลียของผลต่างระหว่างค่าที่แท้จริงและค่าที่ทำนายทั้งหมดยกกำลัง 2

2.3.3 Normalized Root Mean Square Error (NRMSE)

$$NRMSE = \frac{\sqrt{\frac{1}{N}\sum_{i=1}^{n}(\hat{y}_i - y_i)^2}}{\bar{x}} \times 100\% \tag{28}$$

เป็นการคำนวณรากของ MSE ที่ถูก Normalized โดยในที่นี้จะอ้างอิงกับค่าเฉลี่ยของข้อมูล

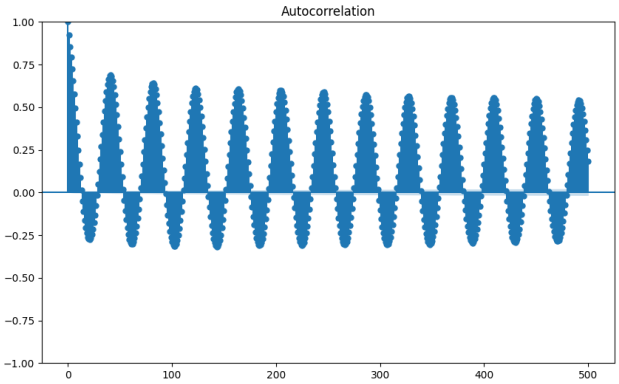
3 ผลลัพธ์จากการดำเนินการ

3.1 ชุดข้อมูลจาก Solar CUEE Station

[13] เป็นชุดข้อมูลที่เก็บมาจากเซ็นเซอร์จากภาควิชาวิศวกรรมไฟฟ้า คณะวิศวกรรมศาสตร์ จุฬาลงกรณ์มหาวิทยาลัย โดยจำนวนข้อมูลทั้งหมดคือ 462,674 ชุด แต่ละชุดประกอบด้วย ค่าความเข้มแสง (Irradiance), ความเข้มแสงในสภาวะฟ้าใส (Irradiance clear sky), ดัชนีฟ้าใส (Clear-sky index), ค่าสัมประสิทธิ์มวลอากาศ (Airmass coefficient), ละติจูด, ลองจิจูด และ มุมของดวงอาทิตย์เทียบกับแนวตั้งฉากของโลก (Zenith angle) โดยในการทำนายข้อมูลความเข้มแสงสำหรับการทดลองในรายงานเล่มนี้จะใช้ค่าความเข้มแสง ความเข้มแสงในสภาวะฟ้าใส ละติจูด ลองจิจูด วัน เดือน เวลาในหน่วยชั่วโมงและนาที โดยข้อมูลที่ได้เก็บตั้งแต่วันที่ 1/1/2023 ถึงวันที่ 30/6/2023 โดยหากดูจากกราฟชุดข้อมูลและกราฟฟังก์ชันสหสัมพันธ์พบว่าค่าความเข้มแสงยังมีความไม่คงที่ (Non-Stationary)

3.2 ชุดข้อมูลทั่วทั้งประเทศ

[13] เป็นชุดข้อมูลที่เก็บมาจาก 56 สถานี ทั่วประเทศ โดยจำนวนข้อมูลทั้งหมดคือ 1,181,852 ชุด แต่ละชุดประกอบด้วยข้อมูล ค่าความเข้มแสง (Irradiance), ความเข้มแสงในสภาวะฟ้าใส (Irradiance clear sky), ดัชนีฟ้าใส (Clear-sky index), ละติจูด, ลองจิจูด และ อุณหภูมิ โดยในการทำนายข้อมูลความเข้มแสงสำหรับการทดลองในรายงานเล่มนี้จะใช้ ข้อมูลจาก 51 สถานี ค่าความเข้มแสง ความเข้มแสงในสภาวะฟ้าใส ละติจูด ลองจิจูด วัน เดือน เวลาในหน่วยชั่วโมงและนาที โดยข้อมูลที่ได้เก็บตั้งแต่วันที่ 1/1/2022 ถึงวันที่ 30/6/2023



รูปที่ 14: ฟังก์ชันสหสัมพันธ์ของข้อมูลความเข้มแสงจำนวน 500 Lags

3.3 การติดตั้งพารามิเตอร์สำหรับการฝึกโมเดล

ในการดำเนินงานจะใช้แบบจำลอง Linear, DLinear, NLinear, PatchTST, Regression LSTM, Transformer, Autoformer และ Informer ที่ใช้ชุดข้อมูล Solar CUEE Station ในการฝึกสอนแบบจำลองและนำไปเปรียบเทียบกับผลการทดสอบจริงซึ่งจะกำหนดพารามิเตอร์เริ่มต้นดังนี้

- Input/Output: จำนวนตัวแปรในชุดข้อมูล ซึ่งขณะนี้ต้องการศึกษาการทำนายจากข้อมูลตัวแปรเดียว (Univariate Forecasting) และหลายตัวแปรสำหรับการทำนายค่าความเข้มแสง (Multivariate Forecasting)
- Sequence Length: ความยาวช่วงเวลาของข้อมูลอนุกรมเวลาที่ใช้ในการฝึกสอนแบบจำลองโดยกำหนดความยาวไว้ที่ 24, 37, 74, 101 สำหรับการทำนายข้อมูลอนุกรมเวลาตัวแปรเดียวและหลายตัวแปร

- Predict Length: ความยาวช่วงเวลาของข้อมูลอนุกรมเวลาที่ใช้ในการทำนายข้อมูลจากแบบจำลองโดยกำหนดความยาวไว้ที่ 4 และ 36 สำหรับการทำนายข้อมูลอนุกรมเวลาตัวแปรเดียวและหลายตัวแปร
- Output Target: ข้อมูลอนุกรมเวลาที่ต้องการทำนายในชุดข้อมูลนี้คือค่าความเข้มแสง (Irradiance)
- Moving Average: จำนวนข้อมูลย้อนหลังในอดีตเพื่อทำการหาค่าเฉลี่ยในการเคลื่อนที่ของข้อมูลอนุกรมเวลาโดยกำหนดค่าไว้ที่ 37 และ 25
- Batch Size: ขนาดของข้อมูลที่จะใช้ในการฝึกสอนแบบจำลองโดยจะปรับพารามิเตอร์ของแบบจำลองหนึ่งครั้งต่อหนึ่งขนาดข้อมูลโดยกำหนดไว้ที่ 128
- Learning Rate: พารามิเตอร์ที่ใช้กำหนดว่าในหนึ่งการวนซ้ำของรอบการฝึกสอนแบบจำลองต้องการปรับเมทริกซ์ถ่วงน้ำหนักของโครงข่ายประสาทโดยกำหนดไว้ที่ 0.005
- Loss Function: ฟังก์ชันที่คิดคำนวณค่าผิดพลาดโดยในแบบจำลองจะพยายามปรับแต่งให้มีค่าน้อยที่สุดในทุกการทำซ้ำของการฝึกสอนแบบจำลองโดยในที่นี้จะใช้ค่าคลาดเคลื่อนกำลังสองเฉลี่ย (Mean Square Error)
- Stride : Hyperparameter สำหรับการทำ Patching ซึ่งเป็นตัวกำหนดไม่ให้ข้อมูลระหว่าง Patch มีการซ้อนทับกัน
- Patch length : ความยาวของขนาดข้อมูลที่ถูกนำมา Patching
- Dimension model : มิติของแบบจำลอง
- Embed : นำข้อมูลที่เกี่ยวข้องกับเวลามาทำการ Embedding
- Encoder Input : จำนวนของตัวแปรที่นำเข้าไปเรียนรู้ในแบบจำลองสำหรับบล็อค Encoder และ แบบจำลองอื่น ๆ ที่ไม่ใช่แบบจำลองแบบตัวแปรรหัสและถอดรหัส
- Decoder Input : จำนวนของตัวแปรที่นำเข้าไปเรียนรู้ในบล็อค Decoder
- Factor (c) : กำหนดความยาวข้อมูลที่ใช้ในการเลื่อนข้อมูลอนุกรมเวลาตามสมการ (17) ของแบบจำลอง Autoformer และกำหนดจำนวนของ Attention pairs ที่สูงที่สุดจำนวน $c \cdot lnL_Q$ โดย L_Q คือความยาวของข้อมูลอนุกรมเวลาของ Queries vector จากแบบจำลอง Informer

3.4 การพยากรณ์ข้อมูลอนุกรมเวลา

3.4.1 การพยากรณ์ข้อมูลอนุกรมเวลาระยะสั้นแบบตัวแปรเดียว

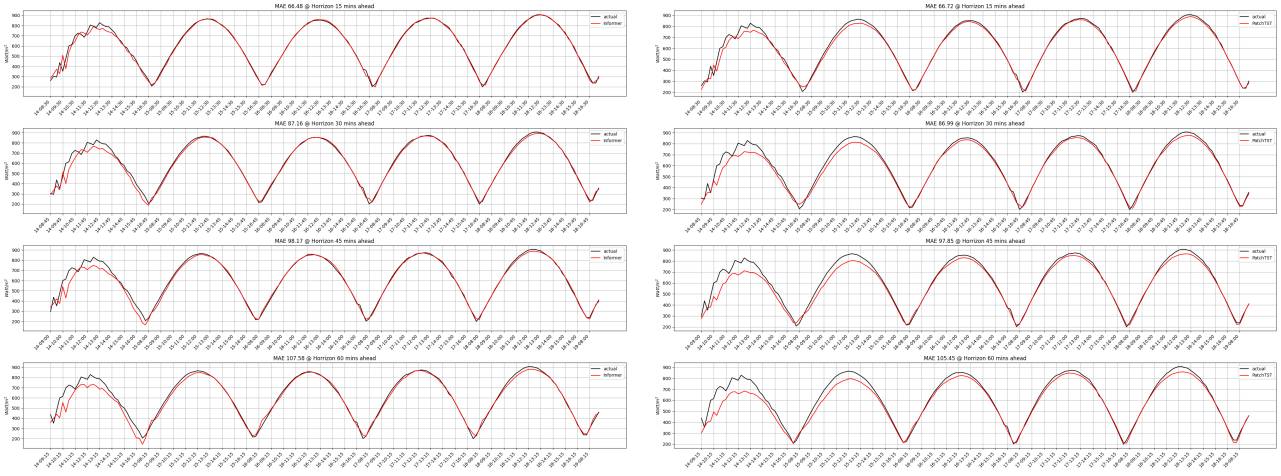
ทำการทดลองเพื่อสังเกตลักษณะการพยากรณ์อนาคตใน 4 ลำดับเวลาถัดไปของข้อมูลซึ่งคิดเป็น 1 ชั่วโมงในอนาคตโดยจะเปลี่ยนความยาวของข้อมูลอนุกรมเวลาขาเข้าเป็น 24, 37, 74, 101 สำหรับแบบจำลอง Regression LSTM, DLinear, NLinear, Linear, PatchTST, Transformer, Informer และ Autoformer

สำหรับการทดสอบครั้งที่หนึ่ง โดยหลังจากสร้างแบบจำลองเสร็จสิ้นและนำไปเรียนรู้กับข้อมูลเรียบร้อยจะนำไปทดสอบในการทำนายข้อมูลอนุกรมเวลาซึ่งแสดงผลดังนี้ จากผลการทดสอบ Informer, PatchTST, Regression LSTM และ Autoformer มีค่า Mean absolute error ที่น้อยที่สุดตามลำดับ ซึ่งสังเกตได้ว่าในการทำนายข้อมูลอนุกรมเวลาระยะสั้นแบบจำลองชนิด Transformer-base จะมีประสิทธิภาพมากกว่าแบบจำลองเชิงเส้นแบบต่าง ๆ แต่พบว่าแบบจำลอง Transformer ดั้งเดิมไม่สามารถทำนายข้อมูลลำดับเวลาได้ดี

MovingAV	Seq len	NLinear		DLinear		Linear		PatchTST	
		MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
25	37	21082.892	99.499	20984.822	104.92	21784.47	106.84	18741.42	92.183
37	37	21065.898	99.383	20974.418	104.517	21791.08	104.42	18651.691	91.35
37	74	20005.877	96.321	20103.664	101.883	21206.39	104.58	**17851.508***	**89.254***
37	101	**19841.623**	**95.923**	**19903.281**	**101.397**	**21102.97**	**102.45**	17855.152	89.744
37	25	24188.076	109.397	23841.588	117.568	22724.36	114.41	20885.303	97.685
		rLSTM		Transformer		Autoformer		Informer	
		MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
25	37	19009.09	96.418	50676.578	185.241	**21052.346**	**100.082**	17679.119	90.95
37	37	18511.551	93.364	50480.547	184.469	21184.518	102.163	**17566.8203***	**89.8475***
37	74	**17739.205**	**91.665**	53248.406	190.525	21179.452	101.546	17688.502	90.903
37	101	18427.805	94.601	50370.508	177.895	22422.576	105.933	18293.303	93.749
37	25	19355.459	97.252	**32849.188**	**133.928**	21571.836	110.882	18844.699	94.125

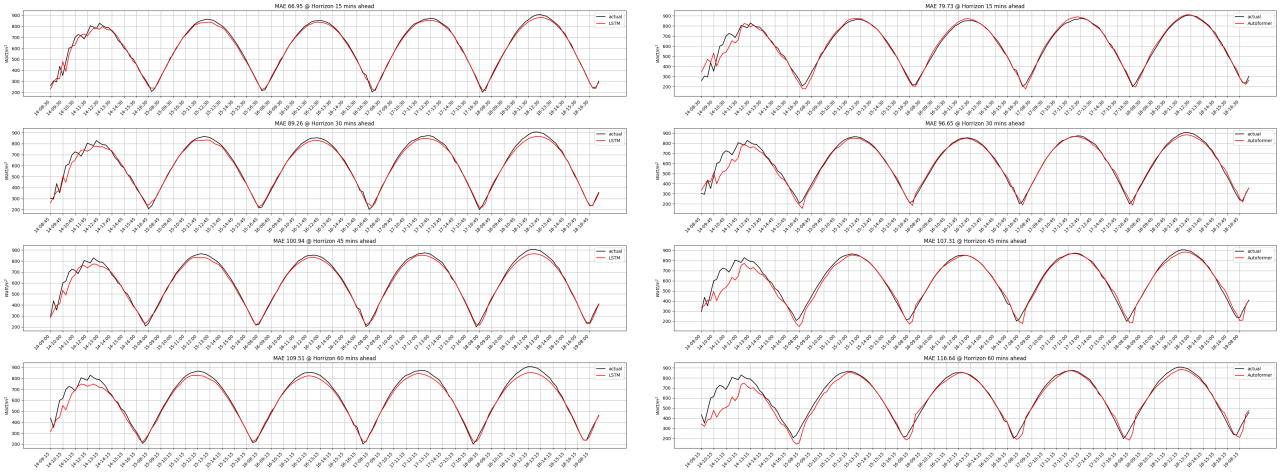
*เป็นค่าที่มีความผิดพลาดต่ำที่สุด

ตารางที่ 1: ผลการทดสอบการทำนายอนุกรมเวลาตัวแปรเดียว



(a) Informer (b) PatchTST

รูปที่ 15: ผลการทดสอบการทำนายค่าความเข้มแสงเทียบกับค่าจริงด้วยแบบจำลอง (a) Informer และ (b) PatchTST



(c) Regression LSTM (d) Autoformer

รูปที่ 16: ผลการทดสอบการทำนายค่าความเข้มแสงเทียบกับค่าจริงด้วยแบบจำลอง (c) Regression LSTM และ (d) Autoformer

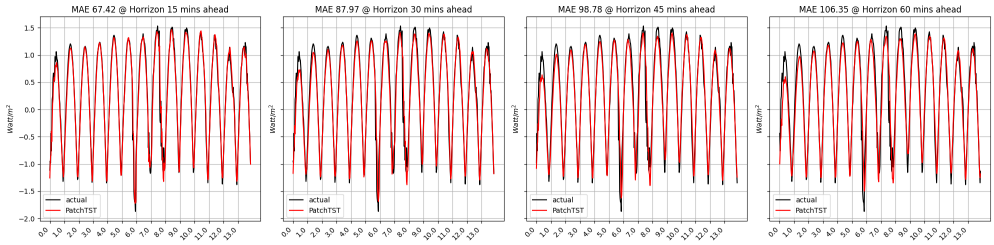
3.4.2 การพยากรณ์ข้อมูลอนุกรมเวลาระยะยาวแบบตัวแปรเดียว

สำหรับการทดสอบครั้งที่สองเพื่อสังเกตลักษณะการพยากรณ์อนาคตใน 36 ลำดับเวลาถัดไปของข้อมูลซึ่งคิดเป็น 9 ชั่วโมงในอนาคตโดยจะเปลี่ยนความยาวของข้อมูลอนุกรมเวลาขาเข้าเป็น 24, 37, 74, 101 สำหรับแบบจำลอง Regression LSTM, PatchTST, Informer และ Autoformer เนื่องจากเป็นแบบจำลองที่มีประสิทธิภาพ 4 อันดับแรกจากการพยากรณ์ข้อมูลอนุกรมเวลาระยะสั้นแบบตัวแปรเดียว หลังจากสร้างแบบจำลองเสร็จสิ้นและนำไปเรียนรู้กับข้อมูลเรียบร้อยจะนำไปทดสอบในการทำนายข้อมูลอนุกรมเวลาจากผลการทดสอบ PatchTST, Informer Regression LSTM และ Autoformer มีค่า Mean absolute error ที่น้อยที่สุดตามลำดับ ซึ่งสังเกตได้ว่าในการทำนายข้อมูลอนุกรมเวลาระยะยาวแบบจำลองชนิด Transformer-base จะมีประสิทธิภาพน้อยกว่าแบบจำลอง PatchTST

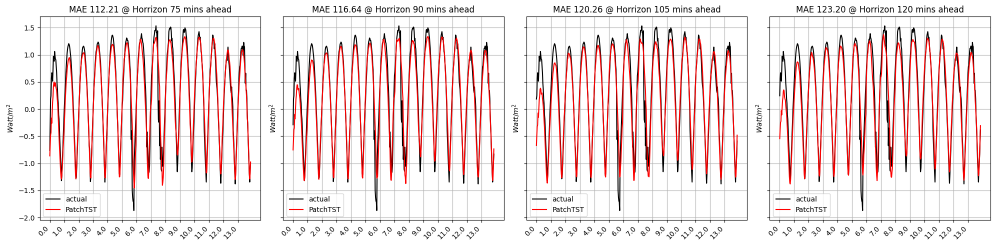
MovingAV	Seq len	NLinear		DLinear		Linear		PatchTST	
		MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
25	37	40096.82	156.415	39989.83	155.989	40109.48	157.155	35359.90	134.726
37	37	40094.17	156.225	40117.14	156.295	40108.60	156.263	35155.86	134.330
37	74	37386.52	150.297	37386.41	150.298	37404.38	150.386	32525.53	129.908
37	101	**36677.83**	**148.353**	**36630.61**	**148.177**	**36587.90**	**147.956**	**32519.26***	**130.402***
37	25	42385.61	162.699	42322.53	162.513	42435.30	162.860	38041.51	139.088
		rLSTM		Transformer		Autoformer		Informer	
		MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
25	37	36767.26	145.86	61462.84	208.55	46476.61	167.12	34733.92	140.81
37	37	36760.16	145.64	61460.27	208.54	43363.05	157.53	34729.84	141.56
37	74	35139.75	141.53	62069.19	209.76	39482.07	148.40	**34247.54**	**141.50**
37	101	**34553.88**	**141.43**	63002.97	211.02	**37898.15**	**146.96**	34394.82	141.83
37	25	38883.04	150.33	**61117.04**	**207.34**	45710.12	158.88	35260.21	141.95

*เป็นค่าที่มีความผิดพลาดต่ำที่สุด

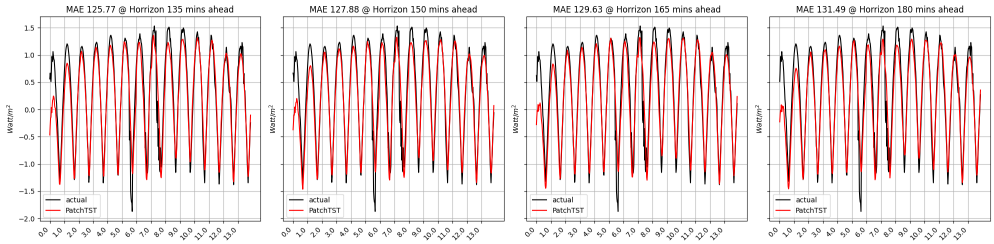
ตารางที่ 2: ผลการทดสอบการทำนายอนุกรมเวลาตัวแปรเดียว



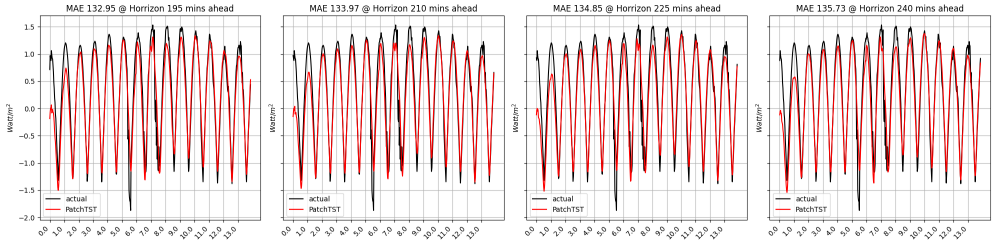
ความเข้มแสงในอนาคต 15 - 60 นาที



ความเข้มแสงในอนาคต 75 - 120 นาที

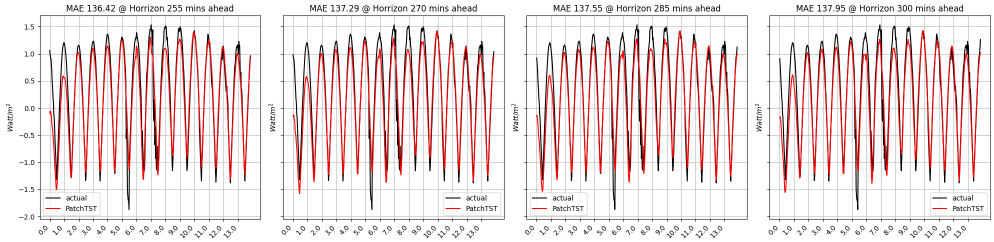


ความเข้มแสงในอนาคต 135 - 180 นาที

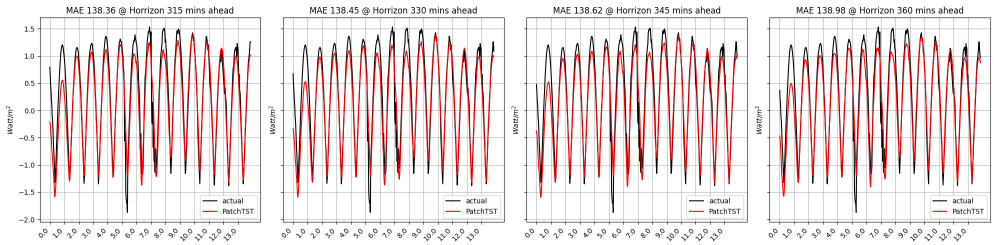


ความเข้มแสงในอนาคต 195 - 240 นาที

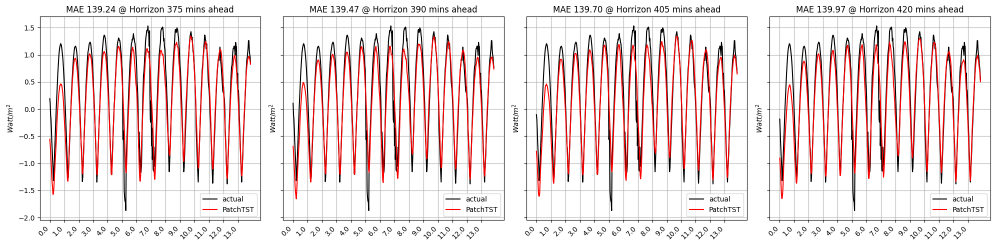
รูปที่ 17: ผลการทดสอบการทำนายค่าความเข้มแสงเทียบกับค่าจริงด้วยแบบจำลอง PatchTST



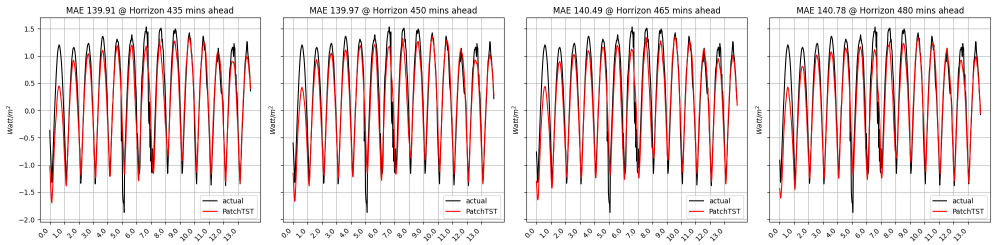
ความเข้มแสงในอนาคต 255 - 300 นาที



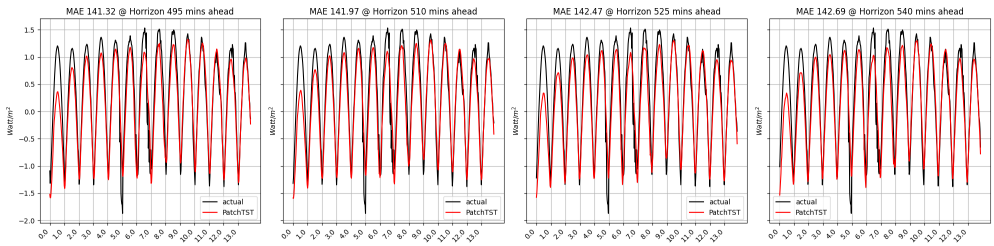
ความเข้มแสงในอนาคต 315 - 360 นาที



ความเข้มแสงในอนาคต 375 - 420 นาที



ความเข้มแสงในอนาคต 435 - 480 นาที



ความเข้มแสงในอนาคต 495 - 540 นาที

รูปที่ 18: ผลการทดสอบการทำนายค่าความเข้มแสงเทียบกับค่าจริงด้วยแบบจำลอง PatchTST (ต่อ)

3.4.3 การพยากรณ์ข้อมูลอนุกรมเวลาระยะสั้นแบบหลายตัวแปรที่ไม่รวมค่าความเข้มแสง

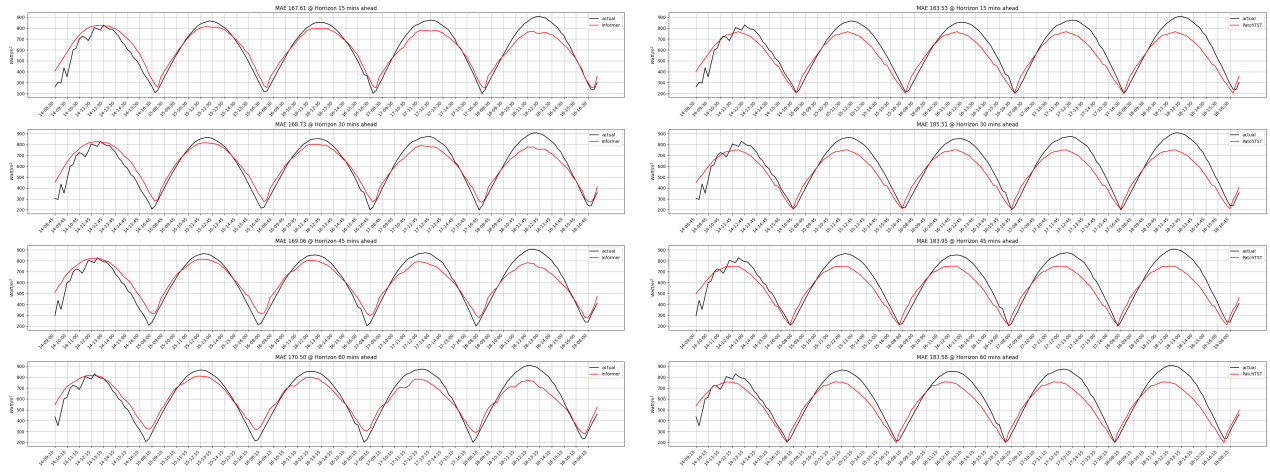
สำหรับการทดสอบครั้งที่สามในการทำนายข้อมูลอนาคตในอนาคต 1 ชั่วโมงถัดไปโดยใช้ข้อมูลตัวแปรความเข้มแสงในสภาวะฟ้าใส ละติจูด ลองจิจูด วันที่ เดือน เวลาในหน่วยชั่วโมง นาทีและความเข้มแสงในสภาวะฟ้าใสในอนาคตลำดับถัดไปเป็นตัวแปรสำหรับการทำนายค่าความเข้มแสง โดยหลังจากสร้างแบบจำลองและเรียนรู้แบบจำลองเสร็จสิ้นจะนำไป

ทดสอบในการทำนายข้อมูลอนุกรมเวลาซึ่งแสดงผลดังนี้ จากผลการทดสอบจากช่วงที่ทำนายระยะสั้นพบว่าแบบจำลอง Informer, Regression LSTM, Autoformer และ PatchTST มีค่าความผิดพลาดน้อยที่สุดตามลำดับ พบว่ามีค่าความผิดพลาดมากกว่าการทดลองแรกเนื่องจากไม่ได้ใช้ค่าความเข้มแสงที่วัดได้จริงในการเรียนรู้แต่ใช้ค่าความเข้มแสงในสภาวะฟ้าใสในปัจจุบันและอนาคตมาใช้ทำนายโดยจุดประสงค์ที่ทำการทดสอบคือตรวจสอบว่าแบบจำลองใดที่สามารถหาค่าความเข้มแสงที่แท้จริงกับความเข้มแสงในสภาวะฟ้าใสและตัวแปรอื่น ๆ ที่คาดว่าจะมีผลต่อการทำนายค่าความเข้มแสง

MovingAV	Seq len	NLinear		DLinear		Linear		PatchTST	
		MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
25	37	**50456.2695**	180.0738	50648.8047	181.0444	50769.1016	181.47	52505.2969	187.3819
37	37	50616.5273	183.1161	50595.6523	181.0492	50713.4531	181.2875	52691.5898	188.1265
37	74	51099.1211	182.5715	**50334.0625**	**179.6222**	50561.375	180.8857	52029.0781	187.3306
37	101	50946.7344	**179.5076**	50485.3203	180.3462	**50567.8125**	**180.5949**	**51398.1875**	**184.1407**
37	25	51882.9219	185.1432	52151.5586	187.7488	51285.668	182.5375	61768.082	203.4764
		rLSTM		Transformer		Autoformer		Informer	
		MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
25	37	**51442.9258**	**174.9538**	53340.9415	184.9826	53014.4205	184.863	52643.1523	178.5628
37	37	50276.4843	179.0837	53309.3125	184.4884	52448.5195	184.574	52074.7617	172.8711
37	74	52589.3828	178.6214	**53963.7617**	**178.9651**	**51650.6992**	**182.8291**	52157.1289	175.174
37	101	52723.9101	176.7456	56007.7266	178.9894	51812.7578	182.8628	**51766.0586***	**168.978***
37	25	51169.3476	176.6233	53218.0273	181.4876	51733.8125	183.9024	51723.4258	174.4894

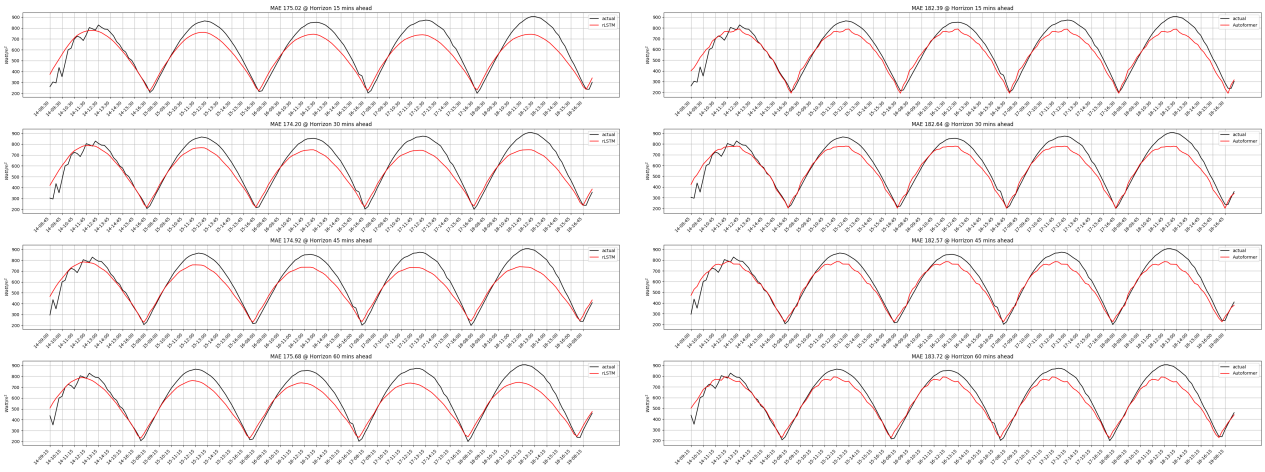
*เป็นค่าที่มีความผิดพลาดต่ำที่สุด

ตารางที่ 3: ผลการทดสอบการทดสอบการทำนายอนุกรมเวลาระยะสั้นหลายตัวแปรที่ไม่รวมค่าความเข้มแสง



(a) Informer (b) PatchTST

รูปที่ 19: ผลการทดสอบการทำนายค่าความเข้มแสงเทียบกับค่าจริงด้วยแบบจำลอง (a) Informer และ (b) PatchTST



(c) Regression LSTM (d) Autoformer

รูปที่ 20: ผลการทดสอบการทำนายค่าความเข้มแสงเทียบกับค่าจริงด้วยแบบจำลอง (c) Regression LSTM และ (d) Autoformer

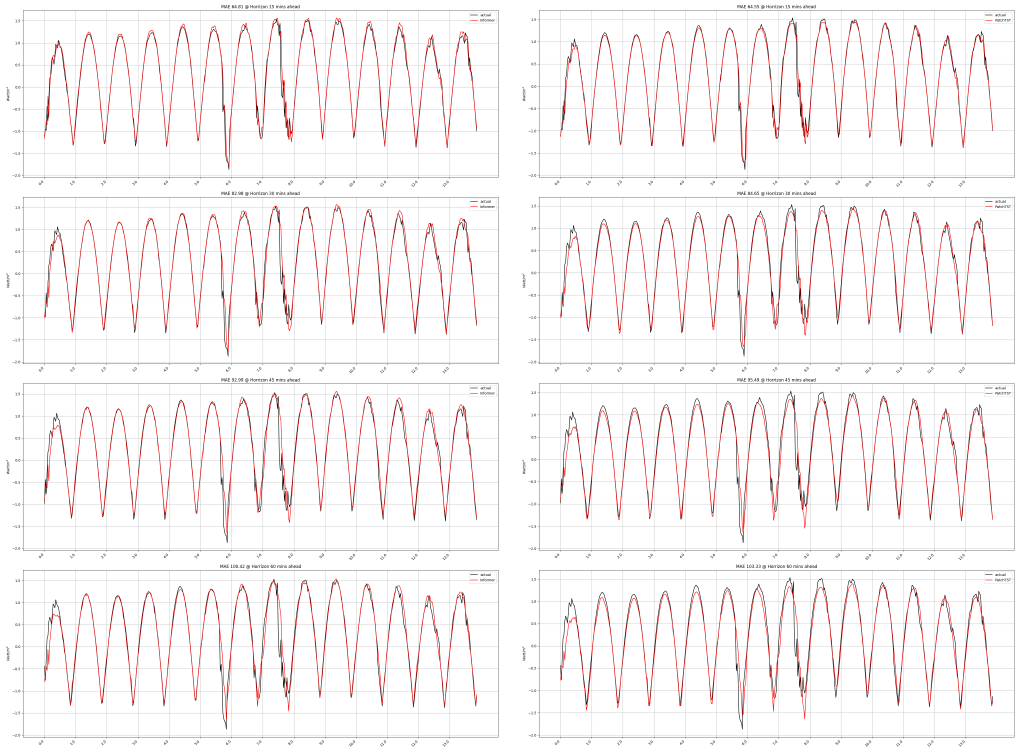
3.4.4 การพยากรณ์ข้อมูลอนุกรมเวลาระยะสั้นแบบหลายตัวแปรที่รวมค่าความเข้มแสง

สำหรับการทดสอบครั้งที่ห้าในการทำนายข้อมูลอนาคตใน 4 ลำดับเวลาถัดไปใช้ข้อมูลตัวแปรความเข้มแสง ความเข้มแสงในสภาวะฟ้าใส ละติจูด ลองจิจูด วันที่ เดือน เวลาในหน่วยชั่วโมงและนาที เป็นตัวแปรสำหรับการทำนายค่าความเข้มแสง โดยหลังจากสร้างแบบจำลองและเรียนรู้แบบจำลองเสร็จสิ้นจะนำไปทดสอบในการทำนายข้อมูลอนุกรมเวลาซึ่งแสดงผลดังนี้ จากผลการทดสอบจากช่วงที่ทำนายระยะสั้นพบว่าแบบจำลอง Informer, PatchTST, Regression LSTM และ Autoformer มีค่าความผิดพลาดน้อยที่สุดตามลำดับ

MovingAV	Seq len	NLinear		DLinear		Linear		PatchTST	
		MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
25	37	21057.23	99.32	20951.89	104.41	21037.418	105.28	18499.717	90.548
37	24	24182.52	109.37	23885.826	117.04	23828.5546	117.5093	20095.396	94.141
37	37	21072.226	99.47	20951.89	104.41	20922.8262	103.6456	18286.972	90.02
37	74	**20077.05**	**95.41**	**20072.19**	**99.06**	20049.293	100.091	**17710.504**	**88.519**
37	101	19847.82	95.72	19883.236	101.049	**19829.023**	**98.939**	17623.018	89.008
		rLSTM		Transformer		Autoformer		Informer	
		MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
25	37	6837.943	88.507	18659.86133	96.9432	17312.32	90.16	16314.86328	84.1924
37	24	17113.178	89.273	19006.1582	97.7382	17137.55	87.61	16665.33008	85.8296
37	37	16980.476	88.926	**18538.5293**	**96.6138**	17290.28	88.8	16345.2832	84.3731
37	74	**16759.104**	**88.255**	18776.48633	97.2198	**16744.71**	**87.13**	**16179.61621***	**83.6853***
37	101	16886.589	88.932	18573.59961	97.8167	16988.3	87.75	16344.70605	86.1191

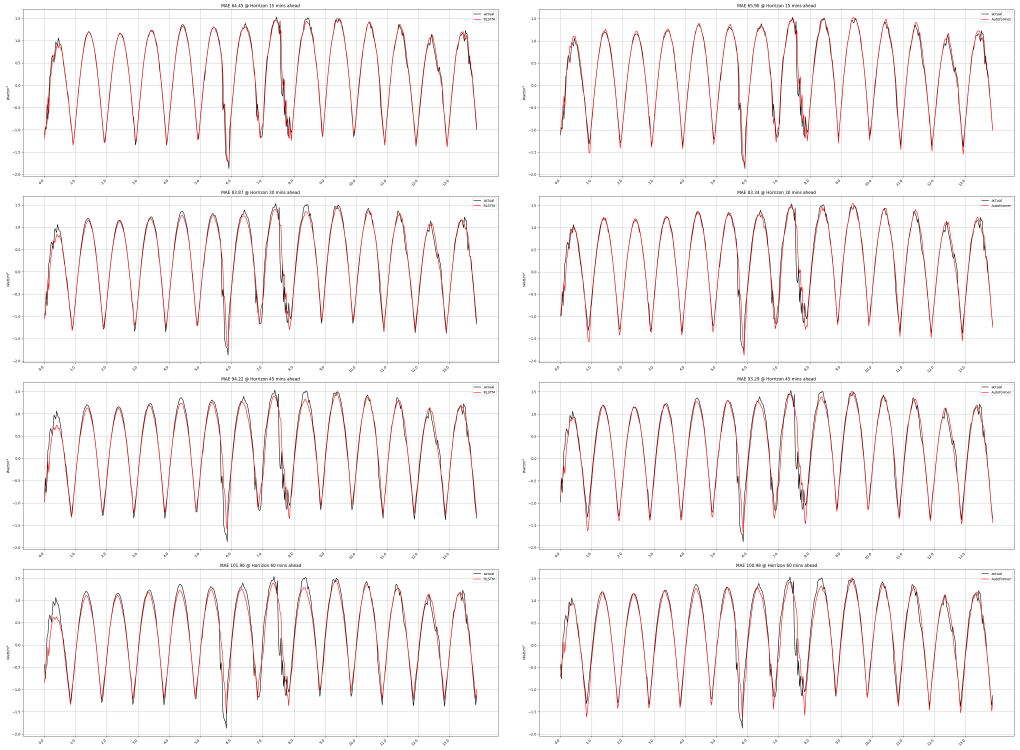
*เป็นค่าที่มีความผิดพลาดต่ำที่สุด

ตารางที่ 4: ผลการทดสอบการทำนายอนุกรมเวลาหลายตัวแปรที่รวมค่าความเข้มแสง



(a) Informer (b) PatchTST

รูปที่ 21: ผลการทดสอบการทำนายค่าความเข้มแสงเทียบกับค่าจริงด้วยแบบจำลอง (a) Informer และ (b) PatchTST



(c) Regression LSTM (d) Autoformer

รูปที่ 22: ผลการทดสอบการทำนายค่าความเข้มแสงเทียบกับค่าจริงด้วยแบบจำลอง (c) Regression LSTM และ (d) Autoformer

3.4.5 การพยากรณ์ข้อมูลอนุกรมเวลาระยะยาวแบบหลายตัวแปรที่รวมค่าความเข้มแสง

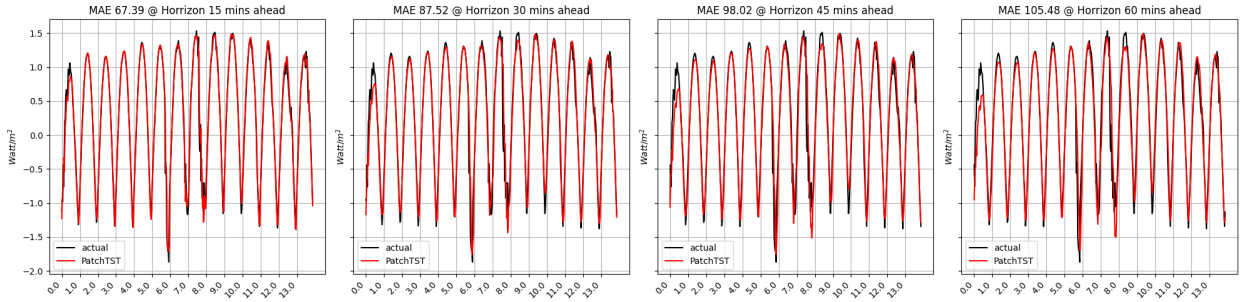
สำหรับการทดสอบครั้งที่หกในการทำนายข้อมูลอนาคต 9 ชั่วโมงโดยใช้ข้อมูลตัวแปรความเข้มแสง ความเข้มแสงในสภาวะฟ้าใส ละติจูด ลองจิจูด วันที่ เดือน เวลาในหน่วยชั่วโมงและนาที เป็นตัวแปรสำหรับการทำนายค่าความเข้มแสง โดยหลังจากสร้างแบบจำลองและเรียนรู้แบบจำลองเสร็จสิ้นจะนำไปทดสอบในการทำนายข้อมูลอนุกรมเวลาซึ่งแสดงผลดังนี้ จากผลการทดสอบจากช่วงที่ทำนายระยะสั้นพบว่าแบบจำลอง Informer, PatchTST, Regression LSTM และ Autoformer มีค่าความผิดพลาดน้อยที่สุดตามลำดับ

MovingAV	Seq len	PatchTST		Regression LSTM		Autoformer		Informer	
		MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
25	37	35537.5451	135.5458	32789.18457	136.8944	33137.48379	133.974126	31602.49245	133.18415
37	37	35559.95272	134.7346761	**32821.98313**	**136.7104588**	**32850.96696**	**132.7685649**	31601.55257	133.2944987
37	74	32916.47976	130.352594	33948.71395	140.8342124	34032.15961	139.0467171	**31607.38086**	**132.4167794**
37	101	**32688.3691**	**129.8003**	33351.4188	139.6313	35982.44694	139.97168	31787.7871	134.0804
37	24	38602.73139	139.3894747	33493.8501	140.520976	36187.14033	140.7963513	33398.55235	134.8398317

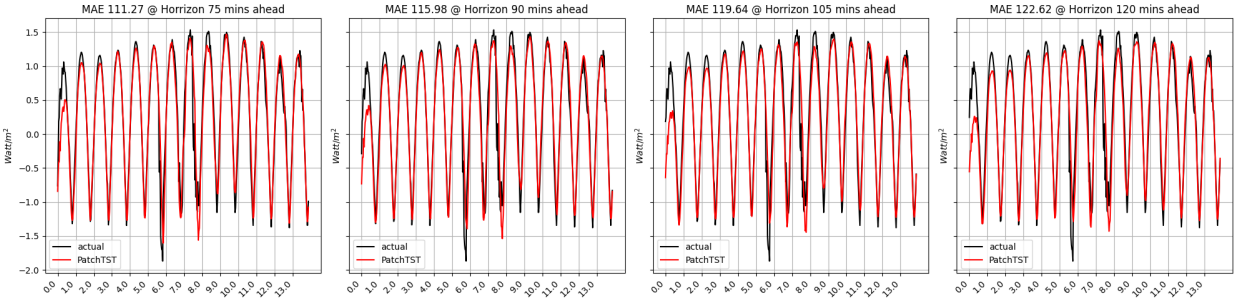
*เป็นค่าที่มีความผิดพลาดต่ำที่สุด

ตารางที่ 5: ผลการทดสอบการทำนายอนุกรมเวลาหลายตัวแปรที่รวมค่าความเข้มแสง

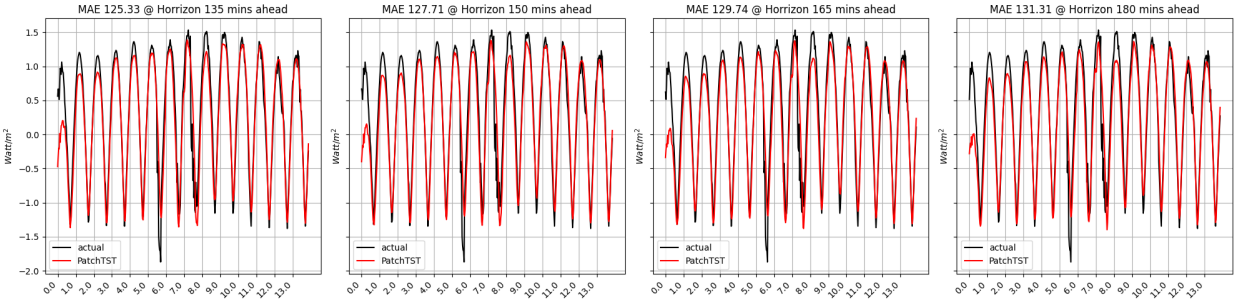
จากตาราง 5 จะเห็นว่า PatchTST สามารถให้ผลการทำนายที่มีค่าต่ำสุดในทุกช่วง Seq len และ MovingAV ดังนั้นจึงแสดงผลของค่าความเข้มแสงที่ถูกทำนายด้วย PatchTST ในช่วงเวลาในอนาคต 9 ชั่วโมง โดยแบ่งแต่ละช่วงเป็นช่วงละ 15 นาที ในอนาคต ดังรูปที่ 23-24



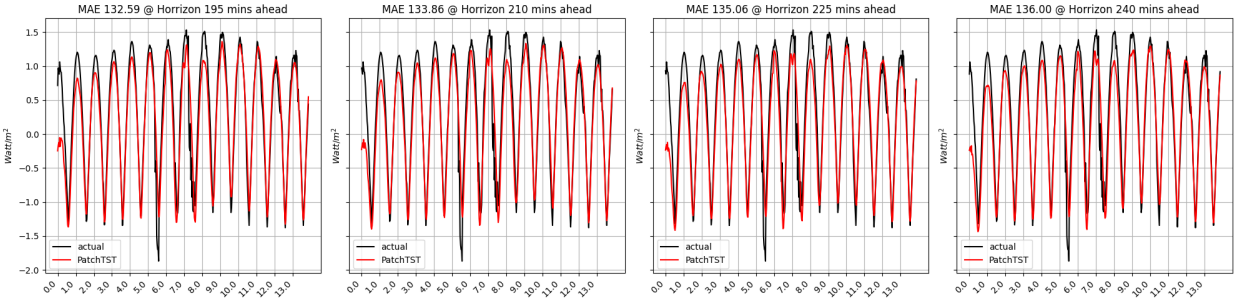
ความเข้มแสงในอนาคต 15 - 60 นาที



ความเข้มแสงในอนาคต 75 - 120 นาที

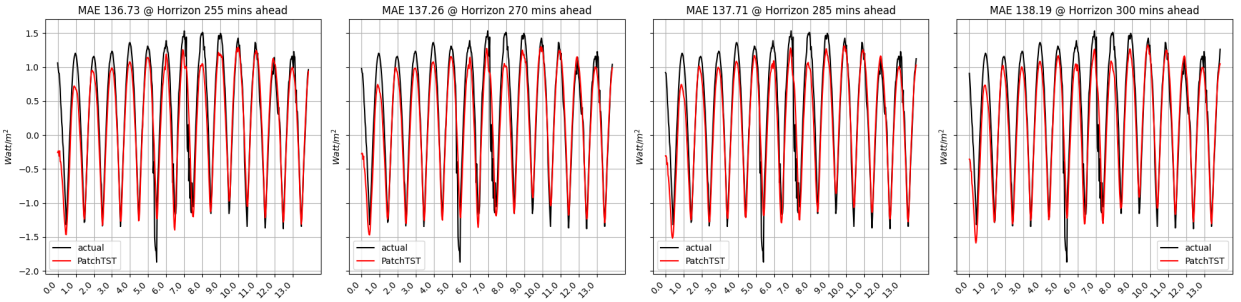


ความเข้มแสงในอนาคต 135 - 180 นาที

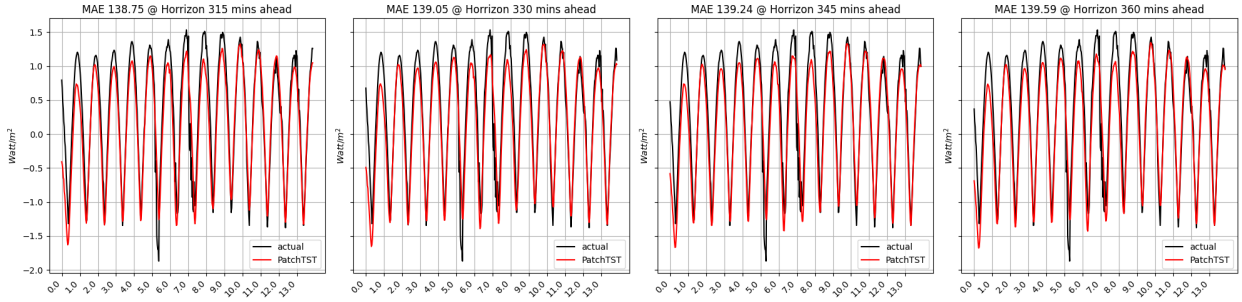


ความเข้มแสงในอนาคต 195 - 240 นาที

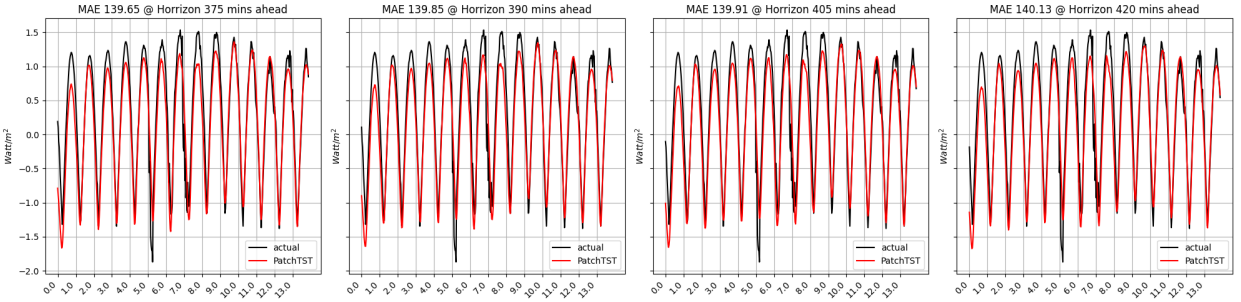
รูปที่ 23: ผลการทดสอบการทำนายค่าความเข้มแสงด้วยแบบจำลอง PatchTST ในกรณีรวมค่าความเข้มแสง



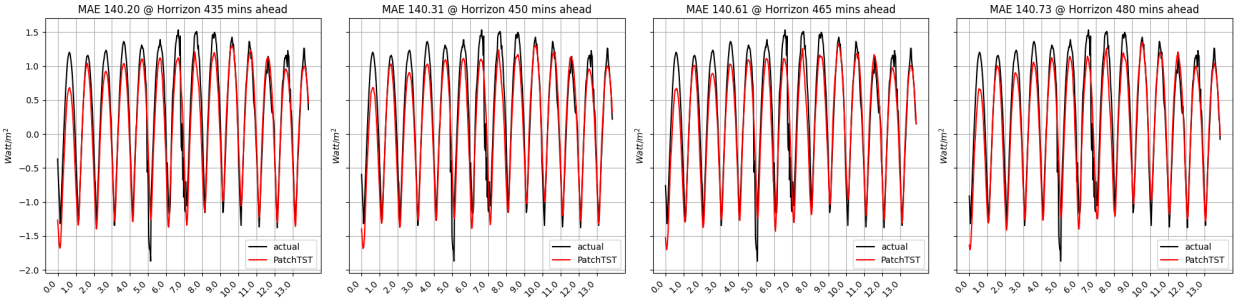
ความเข้มแสงในอนาคต 255 - 300 นาที



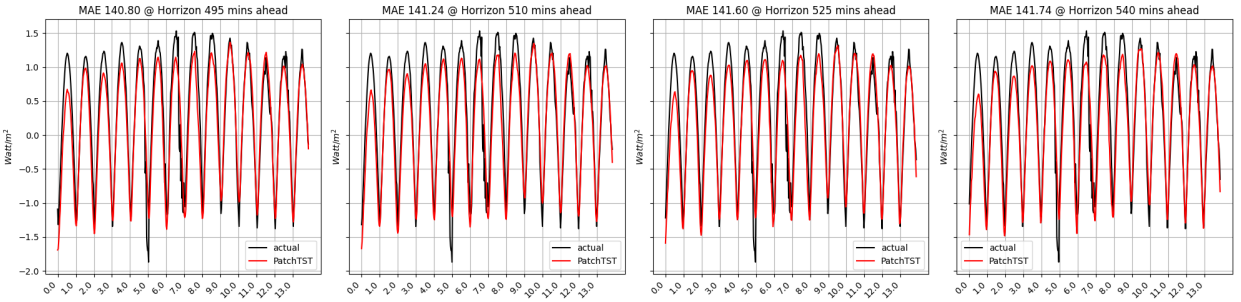
ความเข้มแสงในอนาคต 315 - 360 นาที



ความเข้มแสงในอนาคต 375 - 420 นาที



ความเข้มแสงในอนาคต 435 - 480 นาที



ความเข้มแสงในอนาคต 495 - 540 นาที

รูปที่ 24: ผลการทดสอบการทำนายค่าความเข้มแสงด้วยแบบจำลอง PatchTST ในกรณีรวมค่าความเข้มแสง (ต่อ)

3.5 ผลการปรับแต่งค่าพารามิเตอร์ของแบบจำลอง

3.5.1 กรณีทำนายข้อมูลอนุกรมเวลาในระยะสั้นหลายตัวแปรที่รวมค่าความเข้มแสง

ในการทดสอบจะใช้แบบจำลอง 4 ประเภทคือ Informer, PatchTST, Regression LSTM และ Autoformer เนื่องจากมีประสิทธิภาพในการทำนายข้อมูลอนุกรมเวลาที่มีประสิทธิภาพมากที่สุดจากการทดสอบที่ 3.4.5 โดยจะทำการเปลี่ยนมิติของแบบจำลองเป็น 8, 16, 32, 64, 128, 256, 512 และวัดค่าผิดพลาดเฉลี่ยสมบูรณ์ของชุดข้อมูล Validation และชุดข้อมูล Testing พร้อมทั้งดูจำนวนพารามิเตอร์ของแบบจำลองด้วย

Model	d model	MAE (Validate)	Num Parameters	MAE (Testing)
RLSTM	512	50.1805	15,758,340	88.1436
Informer	**64**	**48.6563**	**877,064**	**86.3751**
PatchTST	512	56.5186	3,569,540	94.6661
Autoformer	256	51.0106	4,221,960	87.9082

a ความยาวของข้อมูลอนุกรมเวลาขาเข้า : 24

Model	d model	MAE (Validate)	Num Parameters	MAE (Testing)
RLSTM	256	49.2555	5,650,948	86.1231
Informer	**128**	**48.3011**	**1,903,624**	**85.7304**
PatchTST	512	51.9728	3,572,100	89.2437
Autoformer	512	49.8336	10,541,064	85.8790

b ความยาวของข้อมูลอนุกรมเวลาขาเข้า : 37

Model	d model	MAE (Validate)	Num Parameters	MAE (Testing)
RLSTM	256	49.5124	10,500,612	86.8404
Informer	**128**	**48.5337**	**877,064**	**85.7329**
PatchTST	512	51.0953	3,584,900	87.0067
Autoformer	64	51.3325	858,888	87.2328

c ความยาวของข้อมูลอนุกรมเวลาขาเข้า : 74

ตารางที่ 6: ผลการการปรับแต่งค่าพารามิเตอร์ (d model) กรณีทำนายข้อมูลอนุกรมเวลาในระยะสั้นหลายตัวแปรที่รวมค่าความเข้มแสง ณ ความยาวข้อมูลเข้า: (a) 24, (b) 37, และ (c) 74

(a) ความยาวของข้อมูลอนุกรมเวลาขาเข้า 24

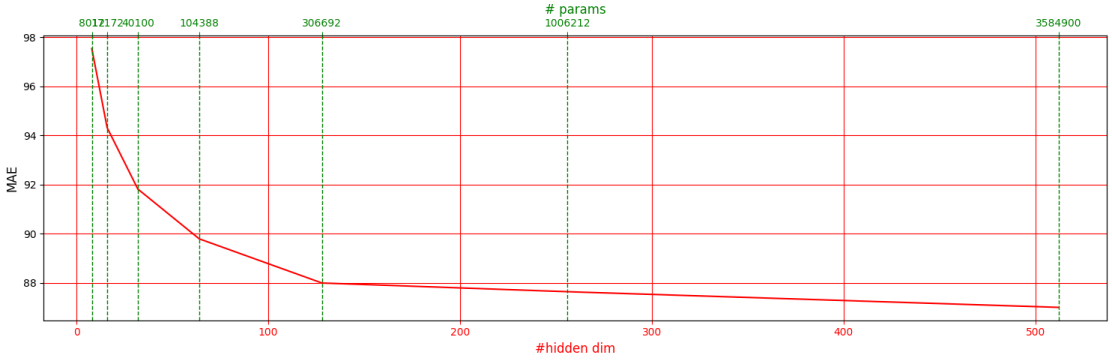
(b) ความยาวของข้อมูลอนุกรมเวลาขาเข้า 37

(c) ความยาวของข้อมูลอนุกรมเวลาขาเข้า 74

รูปที่ 25: กราฟแสดงค่าความผิดพลาดโดยการเปลี่ยนแปลงขนาดของแบบจำลอง Informer

(a) ความยาวของข้อมูลอนุกรมเวลาขาเข้า 24

(b) ความยาวของข้อมูลอนุกรมเวลาขาเข้า 37



(c) ความยาวของข้อมูลอนุกรมเวลาขาเข้า 74

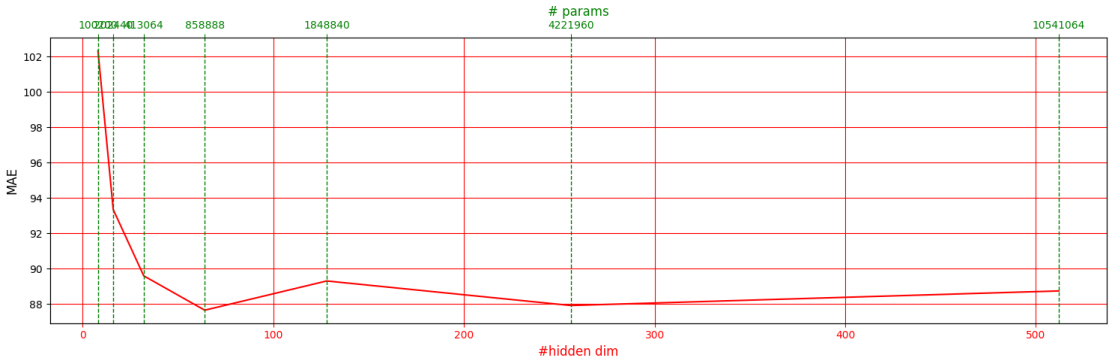
รูปที่ 26: กราฟแสดงค่าความผิดพลาดโดยการเปลี่ยนแปลงขนาดของแบบจำลอง PatchTST

(a) ความยาวของข้อมูลอนุกรมเวลาขาเข้า 24

(b) ความยาวของข้อมูลอนุกรมเวลาขาเข้า 37

(c) ความยาวของข้อมูลอนุกรมเวลาขาเข้า 74

รูปที่ 27: กราฟแสดงค่าความผิดพลาดโดยการเปลี่ยนแปลงขนาดของแบบจำลอง Regression LSTM



(a) ความยาวของข้อมูลอนุกรมเวลาขาเข้า 24

(b) ความยาวของข้อมูลอนุกรมเวลาขาเข้า 37

(c) ความยาวของข้อมูลอนุกรมเวลาขาเข้า 74

รูปที่ 28: กราฟแสดงค่าความผิดพลาดโดยการเปลี่ยนแปลงขนาดของแบบจำลอง Autoformer

3.5.2 กรณีทำนายข้อมูลอนุกรมเวลาในระยะยาวแบบหลายตัวแปรที่รวมค่าความเข้มแสง

ในการทดสอบจะใช้แบบจำลอง 4 ประเภทคือ Informer, PatchTST, Regression LSTM และ Autoformer เนื่องจากมีประสิทธิภาพในการทำนายข้อมูลอนุกรมเวลาที่มีประสิทธิภาพมากที่สุดจากการทดสอบที่ 3.4.6 โดยจะทำการเปลี่ยนมิติของแบบจำลองเป็น 8, 16, 32, 64, 128, 256, 512 และวัดค่าผิดพลาดเฉลี่ยสมบูรณ์ของชุดข้อมูล Validation และชุดข้อมูล Testing พร้อมทั้งดูจำนวนพารามิเตอร์ของแบบจำลองด้วย

แบบจำลอง RLSTM, Informer, PatchTST และ Autoformer ที่ทำนายข้อมูลอนุกรมเวลา 9 ชั่วโมงในอนาคต

Model	d model	MAE (Validate)	Num Parameters	MAE (Testing)
RLSTM	512	83.5535	15,791,140	137.2912
Informer	64	80.2952	877,064	134.8398
PatchTST	512	88.9305	3,618,724	137.0012
Autoformer	16	89.7108	202,440	141.6372

ตารางที่ 7: ความยาวของข้อมูลอนุกรมเวลาขาเข้า : 24

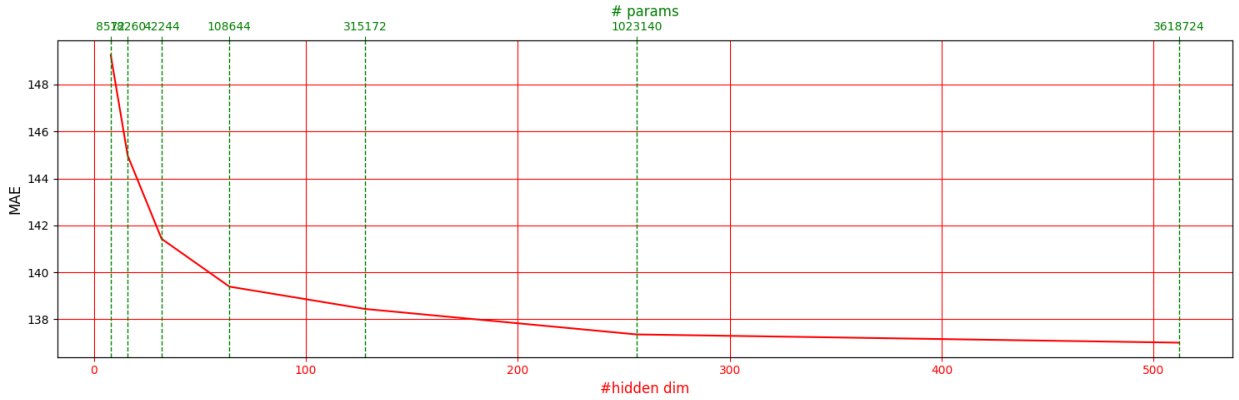
Model	d model	MAE (Validate)	Num Parameters	MAE (Testing)
RLSTM	64	82.7243	360,100	136.7104
Informer	256	79.1360	1,903,624	131.4085
PatchTST	512	82.9422	3,572,100	131.7945
Autoformer	128	81.9403	10,541,064	129.1466

ตารางที่ 8: ความยาวของข้อมูลอนุกรมเวลาขาเข้า : 37

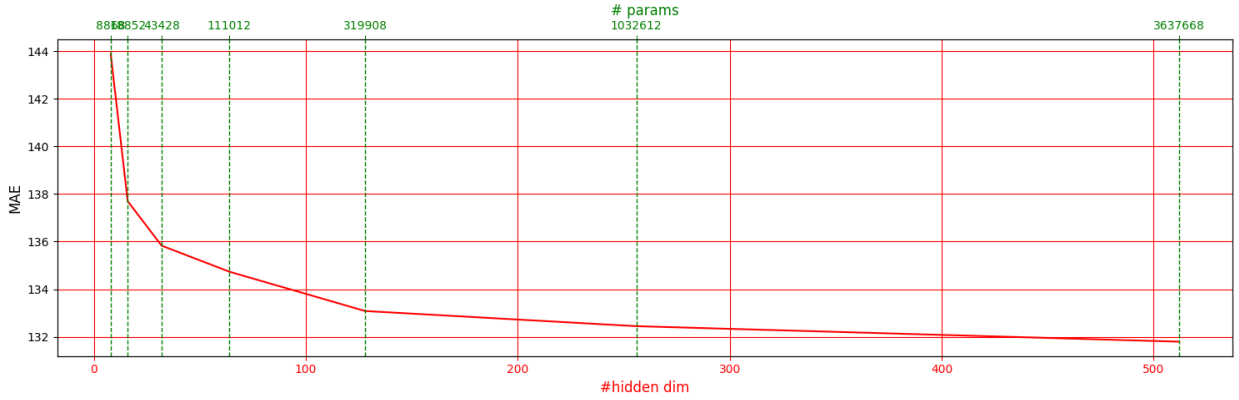
Model	d model	MAE (Validate)	Num Parameters	MAE (Testing)
RLSTM	128	84.2912	2,637,092	137.1077
Informer	64	80.5292	877,064	131.9471
PatchTST	256	80.1991	1,079,972	128.2821
Autoformer	256	82.4937	4,221,960	133.0156

ตารางที่ 9: ความยาวของข้อมูลอนุกรมเวลาขาเข้า : 74

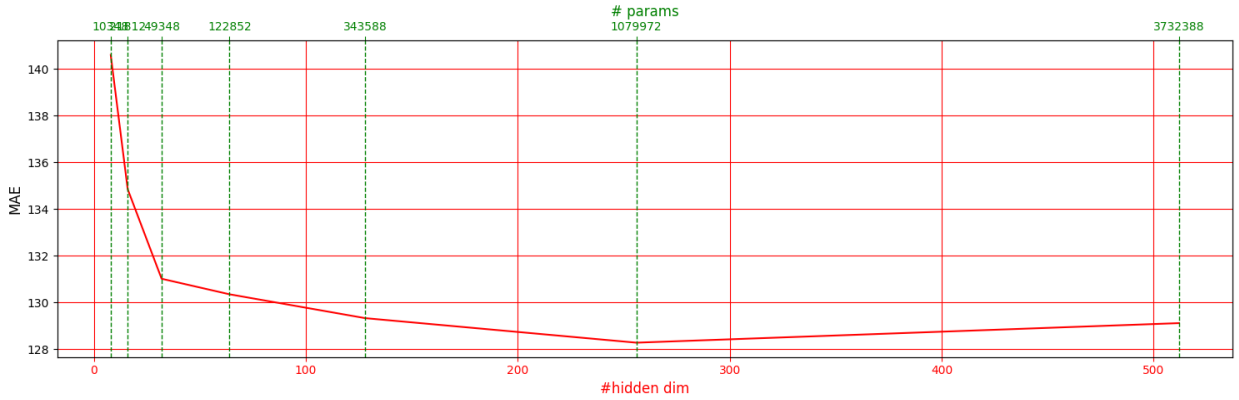
ตารางที่ 10: ผลการการปรับแต่งค่าพารามิเตอร์ (d model) กรณีทำนายข้อมูลอนุกรมเวลาในระยะสั้นหลายตัวแปรที่รวมค่าความเข้มแสง ณ ความยาวข้อมูลเข้า: 24, 37 และ 74



(a) ความยาวของข้อมูลอนุกรมเวลาขาเข้า 24

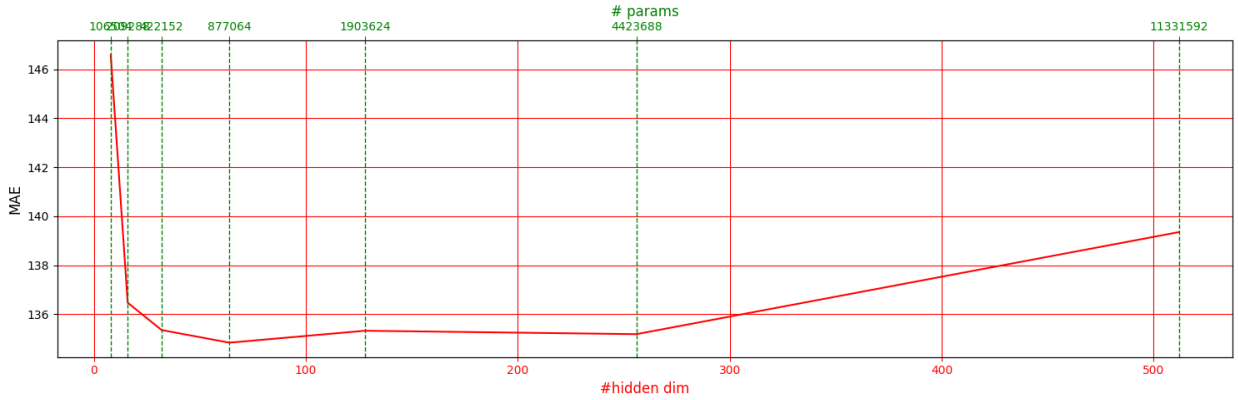


(b) ความยาวของข้อมูลอนุกรมเวลาขาเข้า 37

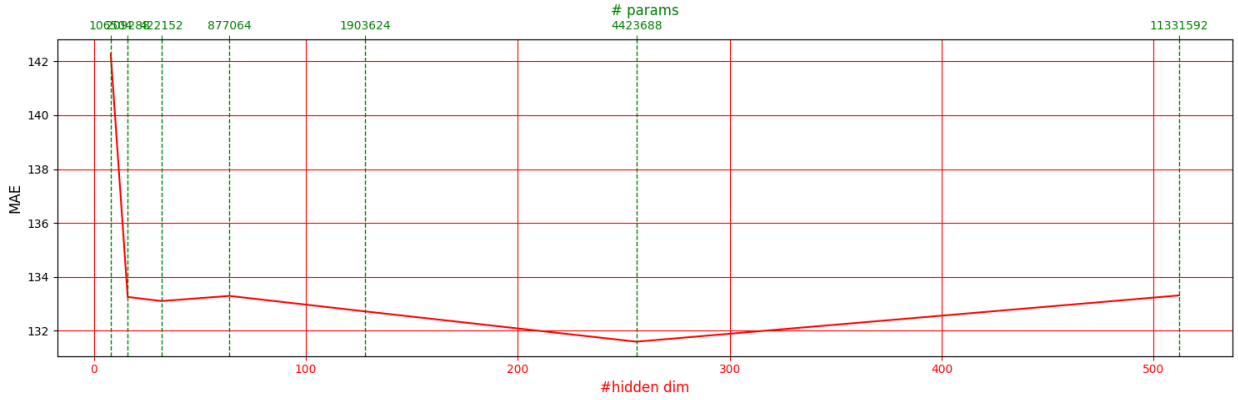


(c) ความยาวของข้อมูลอนุกรมเวลาขาเข้า 74

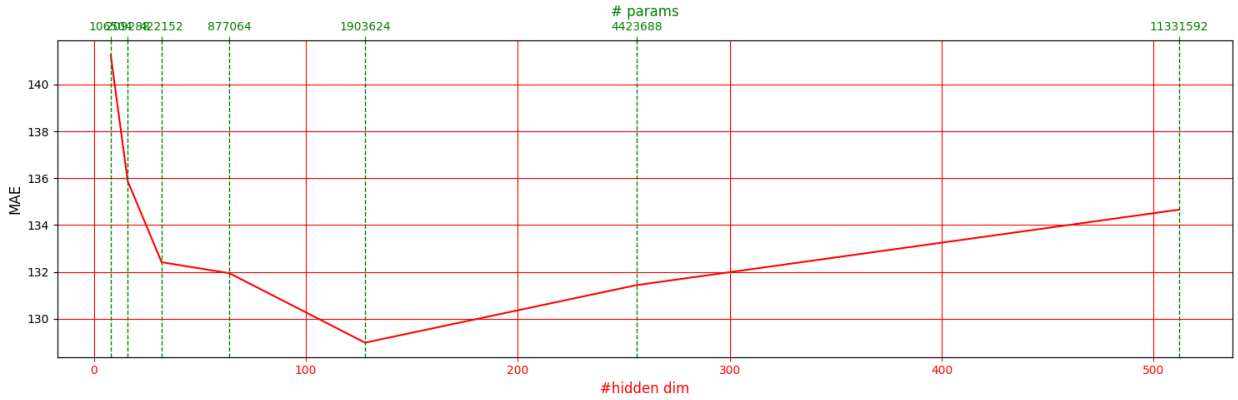
รูปที่ 29: กราฟแสดงค่าความผิดพลาดโดยการเปลี่ยนแปลงขนาดของแบบจำลอง PatchTST



(a) ความยาวของข้อมูลอนุกรมเวลาขาเข้า 24

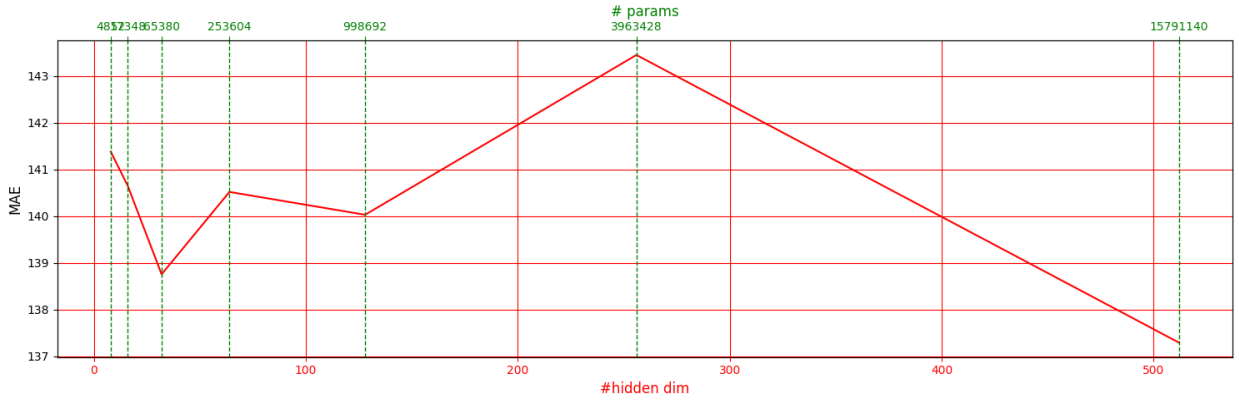


(b) ความยาวของข้อมูลอนุกรมเวลาขาเข้า 37

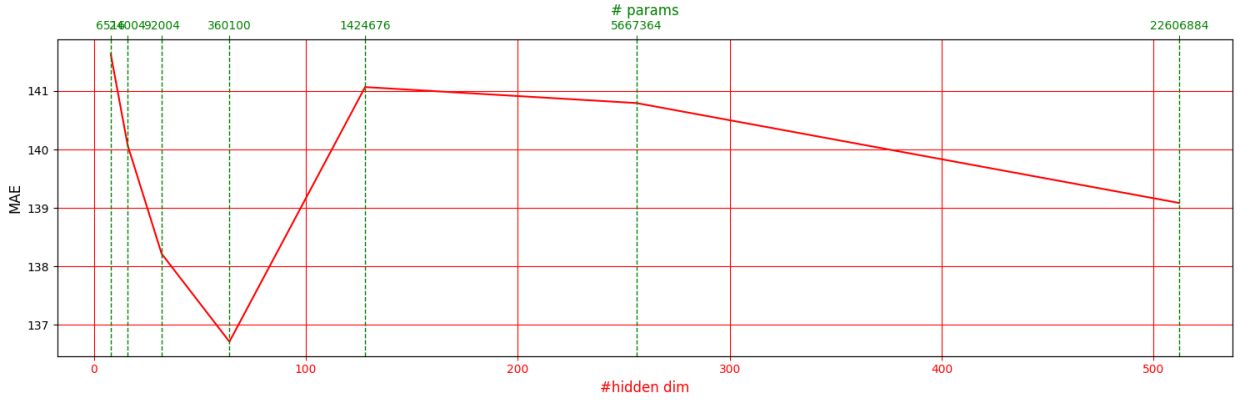


(c) ความยาวของข้อมูลอนุกรมเวลาขาเข้า 74

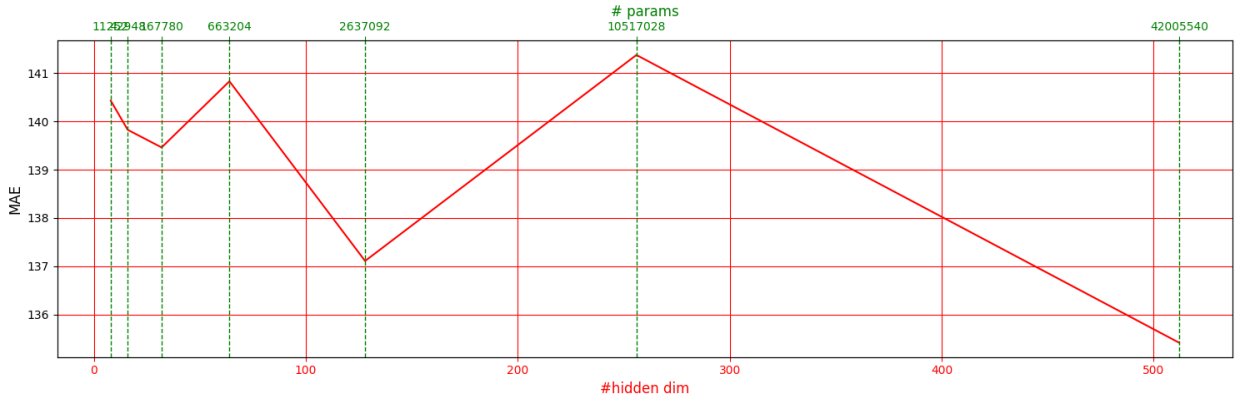
รูปที่ 30: กราฟแสดงค่าความผิดพลาดโดยการเปลี่ยนแปลงขนาดของแบบจำลอง Informer



(a) ความยาวของข้อมูลอนุกรมเวลาขาเข้า 24

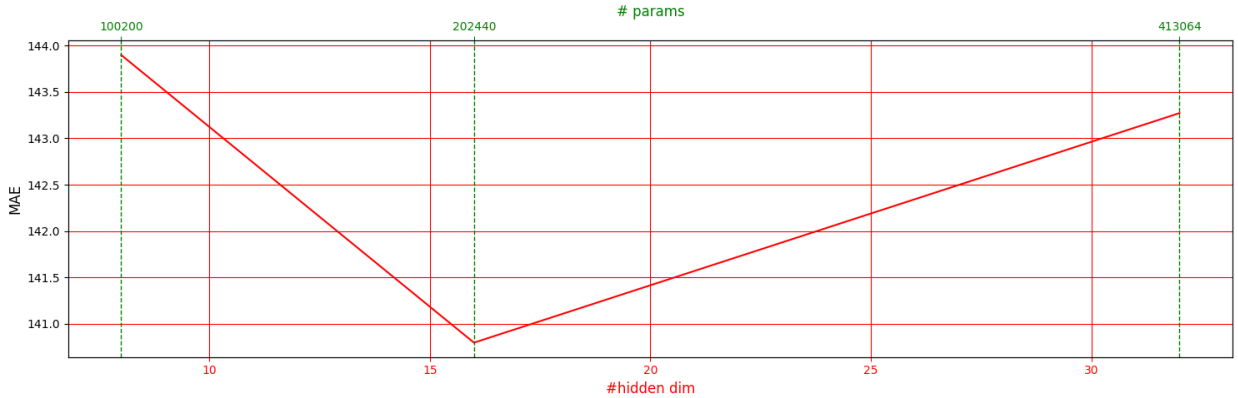


(b) ความยาวของข้อมูลอนุกรมเวลาขาเข้า 37

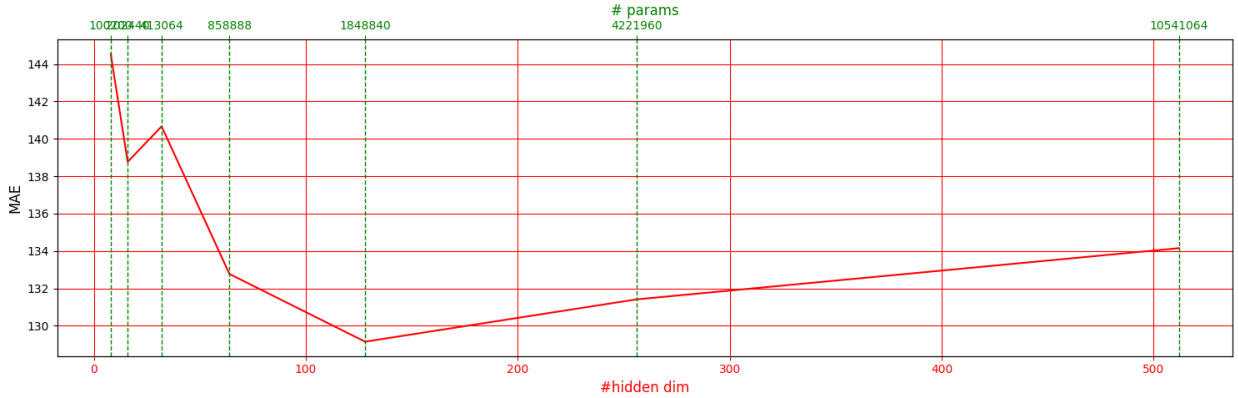


(c) ความยาวของข้อมูลอนุกรมเวลาขาเข้า 74

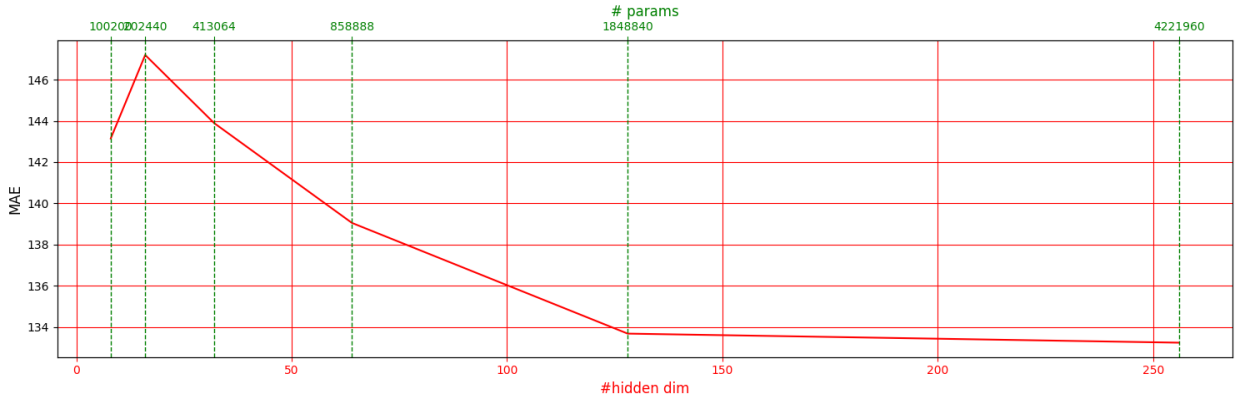
รูปที่ 31: กราฟแสดงค่าความผิดพลาดโดยการเปลี่ยนแปลงขนาดของแบบจำลอง Regression LSTM



(a) ความยาวของข้อมูลอนุกรมเวลาขาเข้า 24



(b) ความยาวของข้อมูลอนุกรมเวลาขาเข้า 37



(c) ความยาวของข้อมูลอนุกรมเวลาขาเข้า 74

รูปที่ 32: กราฟแสดงค่าความผิดพลาดโดยการเปลี่ยนแปลงขนาดของแบบจำลอง Autoformer

4 บทสรุป

4.1 สรุปผลการดำเนินการ

ได้ศึกษาโครงสร้างการทำงานของแบบจำลองการพยากรณ์ข้อมูลอนุกรมเวลาด้วยแบบโครงข่ายประสาทเชิงลึกรวมถึงแบบจำลองทางสถิติและสร้างแบบจำลองในการพยากรณ์ข้อมูลอนุกรมเวลาด้วยโครงข่ายประสาททั้งหมด 8 ประเภทคือ Linear, DLinear, NLinear, PatchTST, Regression LSTM, Transformer, Autoformer และ Informer โดยแบ่งการทดสอบออกเป็นสามส่วนหลักคือ การทำนายข้อมูลความเข้มแสงในระยะสั้นของแบบ การทำนายข้อมูลความเข้มแสงในระยะยาว และการปรับแต่งพารามิเตอร์ของแบบจำลอง โดยพบว่า

- **การทำนายข้อมูลความเข้มแสงในระยะสั้น**
  ในการทดสอบนี้จะแบ่งย่อยอีก 3 กรณีคือ

กรณีที่ 1 กรณีที่ใช้เพียงตัวแปรความเข้มแสงในการทำนายความเข้มแสงในอนาคต (ตารางที่ 1)

กรณีที่ 2 กรณีที่ไม่ใช้ความเข้มแสงในอดีตในการทำนายความเข้มแสงในอนาคต (ตารางที่ 3)

กรณีที่ 3 กรณีที่ใช้หลายตัวแปรในการทำนายความเข้มแสงในอนาคต (ตารางที่ 4)

  - จากการทดสอบจะพบว่า กรณีที่ 2 ค่าผิดพลาดเฉลี่ยสมบูรณ์ (MAE) มากกว่ากรณีที่ 1 และกรณีที่ 3 ตามลำดับ แสดงให้เห็นว่าตัวแปรอื่น ๆ มีผลต่อการทำนายข้อมูล ยกตัวอย่างเช่นละติจูดและลองจิจูดของเซ็นเซอร์ที่ทำการวัดค่าเนื่องจากชุดข้อมูลที่ใช้ประกอบด้วยเซ็นเซอร์จากหลากหลายที่ทำให้ความเข้มแสงไม่ใกล้เคียงกันกับพื้นที่ที่ต่างกัน ในทางกลับกันหากไม่มีข้อมูลความเข้มแสงในอดีตเลยมีเพียงความเข้มแสงในสภาวะฟ้าใสจะส่งผลให้แบบจำลองมีค่าผิดพลาดสมบูรณ์ที่สูงมากขึ้นอย่างเห็นได้ซึ่งแสดงให้เห็นว่าตัวแปรที่นำมาใช้ในการทำนายมีความเกี่ยวข้องกับความเข้มแสงน้อย
  - แบบจำลองที่มีประสิทธิภาพมากที่สุดสำหรับการทดลองนี้คือแบบจำลอง Informer เนื่องจากว่าการทำนายข้อมูลในระยะสั้นตัวแปรอื่น ๆ มีความสัมพันธ์กับความเข้มแสงทำให้แบบจำลองชนิด Transformer-based มีประสิทธิภาพดีกว่าเพราะแบบจำลองประเภทนี้มีความสามารถในการหาความสัมพันธ์ระหว่างข้อมูลอย่างมีประสิทธิภาพดังสมการ (23) และเป็นแบบจำลองที่ใช้รูปแบบการทำนายข้อมูลแบบ DMF (Direct Multi-step Forecasting) ที่ไม่มีการสะสมค่าความผิดพลาดระหว่างลำดับเวลาจึงเป็นเหตุผลที่แบบจำลอง Informer เป็นตัวเลือกสำหรับการทำนายข้อมูลอนุกรมเวลาในระยะสั้น

- **การทำนายข้อมูลความเข้มแสงในระยะยาว**
  ในการทดสอบนี้จะแบ่งย่อยอีก 2 กรณีคือ

กรณีที่ 1 กรณีที่ใช้เพียงตัวแปรความเข้มแสงในการทำนายความเข้มแสงในอนาคต (ตารางที่ 2)

กรณีที่ 2 กรณีที่ใช้หลายตัวแปรในการทำนายความเข้มแสงในอนาคต (ตารางที่ 5)

  - จากการทดสอบจะพบว่า กรณีที่ 2 ค่าผิดพลาดเฉลี่ยสมบูรณ์ (MAE) มากกว่ากรณีที่ 1 ตามลำดับ แสดงให้เห็นว่าตัวแปรอื่น ๆ มีผลต่อการทำนายข้อมูลความเข้มแสง
  - แบบจำลองที่มีประสิทธิภาพมากที่สุดสำหรับการทดลองนี้คือแบบจำลอง PatchTST เนื่องจากว่าการทำนายข้อมูลในระยะยาวตัวแปรอื่น ๆ จะมีความเป็นอิสระต่อกันเนื่องจากการกระจายตัวของข้อมูลของแต่ละตัวแปร

แตกต่างกันทำให้แบบจำลองชนิด PatchTST ที่นำเสนอความเป็นอิสระระหว่างตัวแปรอื่น ๆ และการเพิ่มมิติของข้อมูล (Patching) มีประสิทธิภาพดีกว่าแบบจำลองชนิด Transformer-based และเป็นแบบจำลองที่ใช้รูปแบบการทำนายข้อมูลแบบ DMF (Direct Multi-step Forecasting) ที่ไม่มีการสะสมค่าความผิดพลาดระหว่างลำดับเวลาโดยจะเห็นได้ชัดเจนหากเทียบกับแบบจำลอง Regression LSTM ซึ่งเป็นแบบจำลองที่ทำนายข้อมูลทีละลำดับเวลา (Iterative Multi-step Forecasting)จึงเป็นเหตุผลที่แบบจำลอง PatchTST เป็นตัวเลือกสำหรับการทำนายข้อมูลอนุกรมเวลาในระยะยาว

- **การปรับแต่งพารามิเตอร์ของแบบจำลอง**

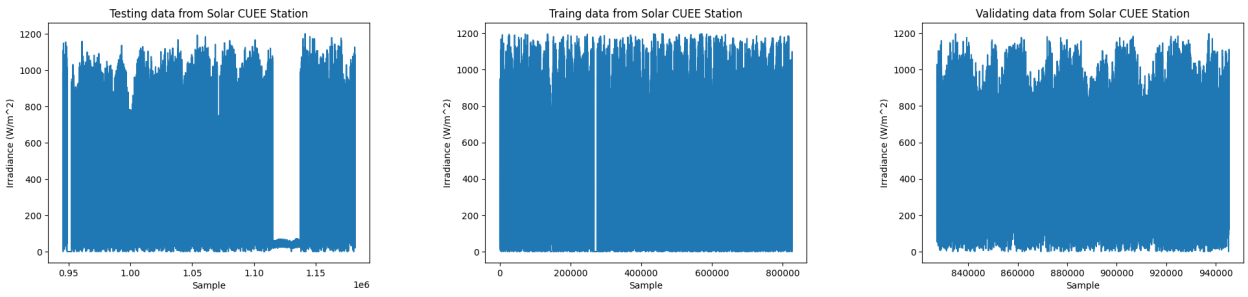
ในการทดสอบนี้จะแบ่งย่อยอีก 2 กรณีคือ

กรณีที่ 1 (ตารางที่ 6) กรณีทำนายข้อมูลอนุกรมเวลาในระยะสั้นหลายตัวแปรที่รวมค่าความเข้มแสง
เนื่องจากแบบจำลองชนิดโครงข่ายประสาทจะใช้พารามิเตอร์ในการสร้างแบบจำลองเป็นอย่างมากจึงควรทำการปรับมิติของแบบจำลอง (Dimension of model) ซึ่งอาจจะแลกมากับค่าความผิดพลาดที่มากขึ้น โดยจากผลลัพธ์จะเห็นได้ว่าเลือกใช้แบบจำลอง Informer กรณีที่ทำนายความเข้มแสงระยะสั้นที่ความยาวของข้อมูลขาเข้าเป็น 37 มีค่าผิดพลาดเฉลี่ยสมบูรณ์กับข้อมูลชุดทดสอบอยู่ที่ 85.7 และจำนวนพารามิเตอร์ที่ 1,903,624 ตัว โดยใช้มิติของแบบจำลองอยู่ที่ 128

กรณีที่ 2 (ตารางที่ 10) กรณีทำนายข้อมูลอนุกรมเวลาในระยะยาวแบบหลายตัวแปรที่รวมค่าความเข้มแสง
จากการปรับค่ามิติของแบบจำลองโดยอ้างอิงจากผลลัพธ์จะเห็นได้ว่าเลือกใช้แบบจำลอง PatchTST กรณีที่ทำนายความเข้มแสงระยะยาวที่ความยาวของข้อมูลขาเข้าเป็น 74 มีค่าผิดพลาดเฉลี่ยสมบูรณ์กับข้อมูลชุดทดสอบอยู่ที่ 128.3 และจำนวนพารามิเตอร์ที่ 1,079,972 ตัว โดยใช้มิติของแบบจำลองอยู่ที่ 256

4.2 ปัญหา อุปสรรค และแนวทางแก้ไข

มีปัญหาเนื่องจากว่าข้อมูลที่ใช้สำหรับทำการทดสอบมีอุปกรณ์ชนิดหนึ่งที่คาดว่าจะมีปัญหาโดยจากการสังเกตดูค่าความเข้มแสงจากชุดข้อมูล Solar CUEE Station พบว่ามีเซ็นเซอร์ชิ้นหนึ่งที่จังหวัดชลบุรีมีค่าความเข้มแสงที่ใกล้เคียงกันตลอดทั้งปีตั้งแต่วันที่ 1/1/2022 ถึงวันที่ 30/6/2023 ซึ่งแตกต่างจากเซ็นเซอร์อื่น ๆ อย่างชัดเจนแสดงได้ดังรูป



(a) Testing data from CUEE Station (b) Training data from CUEE Station (c) Validating data from CUEE Station

รูปที่ 33: Solar CUEE Station data

โดยหากเทียบกับข้อมูลชุดทดสอบที่มีค่าที่ผิดแปลกไปกับชุดข้อมูลทั้งหมดพบว่าเป็น 10 เปอร์เซ็นของข้อมูลชุดทดสอบทั้งหมดซึ่งทำให้คาดการณ์ได้ว่าหากตัดเซ็นเซอร์ชิ้นนี้ออกไปจะมีค่าความผิดพลาดที่น้อยลงซึ่งจะขอแสดงตัวอย่างเป็นการ

เปรียบเทียบระหว่างแบบจำลอง PatchTST และ Informer กับการทำนายข้อมูลอนุกรมเวลาแบบหลายตัวแปรที่รวมค่าความเข้มแสงในระยะยาวและระยะสั้นตามลำดับ ซึ่งผลลัพธ์จากการทดสอบเบื้องต้นพบว่าเป็นไปตามที่คาดการณ์ไว้คือ แบบจำลอง PatchTST ค่าผิดพลาดเฉลี่ยสมบูรณ์ลดลง และแบบจำลอง Informer ค่าผิดพลาดเฉลี่ยสมบูรณ์ลดลง

5 กิตติกรรมประกาศ

โครงงานฉบับนี้สำเร็จลุล่วงด้วยความกรุณาจาก ดร. สุวิชญา สุวรรรวิมลกุล ที่ได้สละเวลาแก่ผู้จัดทำ ให้คำปรึกษา แนะนำ และตรวจทานแก้ไขข้อบกพร่องต่าง ๆ ด้วยความเอาใจใส่และความละเอียดเป็นอย่างยิ่งตลอดระยะเวลาการทำโครงงานฉบับนี้ ผู้จัดทำขอขอบพระคุณเป็นอย่างสูงไว้ ณ ที่นี้ ขอขอบคุณภาควิชาวิศวกรรมไฟฟ้าที่ให้การสนับสนุนในด้านข้อมูลที่ใช้ในการจัดทำโครงงาน ฉบับนี้จากคณะวิศวกรรมไฟฟ้า จุฬาลงกรณ์มหาวิทยาลัย สุดท้ายนี้ขอขอบคุณพี่ ๆ และเพื่อน ๆ ในภาควิชาวิศวกรรมไฟฟ้า ที่ให้ความช่วยเหลือและให้คำปรึกษาปัญหาและข้อสงสัยที่พบเจอในการจัดทำโครงงานฉบับนี้

เอกสารอ้างอิง

[1] A. Sherstinsky, “Fundamentals of recurrent neural network (rnn) and long short-term memory (lstm) network,” *Physica D: Nonlinear Phenomena*, vol. 404, p. 132306, Mar. 2020. [Online]. Available: http://dx.doi.org/10.1016/j.physd.2019.132306

[2] S. Hochreiter and J. Schmidhuber, “Long short-term memory,” *Neural Comput.*, vol. 9, no. 8, p. 1735–1780, nov 1997. [Online]. Available: https://doi.org/10.1162/neco.1997.9.8.1735

[3] H. Wu, J. Xu, J. Wang, and M. Long, “Autoformer: Decomposition transformers with Auto-Correlation for long-term series forecasting,” in *Advances in Neural Information Processing Systems*, 2021.

[4] H. Zhou, S. Zhang, J. Peng, S. Zhang, J. Li, H. Xiong, and W. Zhang, “Informer: Beyond efficient transformer for long sequence time-series forecasting,” in *The Thirty-Fifth AAAI Conference on Artificial Intelligence, AAAI 2021, Virtual Conference*, vol. 35, no. 12. AAAI Press, 2021, pp. 11 106–11 115.

[5] A. Zeng, M. Chen, L. Zhang, and Q. Xu, “Are transformers effective for time series forecasting?” 2023.

[6] Y. Nie, N. H. Nguyen, P. Sinthong, and J. Kalagnanam, “A time series is worth 64 words: Long-term forecasting with transformers,” in *International Conference on Learning Representations*, 2023.

[7] R. Hyndman and G. Athanasopoulos, *Forecasting: Principles and Practice*, 3rd ed. Australia: OTexts, 2021.

[8] O. Triebe, H. Hewamalage, P. Pilyugina, N. Laptev, C. Bergmeir, and R. Rajagopal, “Neuralprophet: Explainable forecasting at scale,” 2021.

[9] R. C. Staudemeyer and E. R. Morris, “Understanding lstm – a tutorial into long short-term memory recurrent neural networks,” 2019.

[10] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. u. Kaiser, and I. Polosukhin, “Attention is all you need,” in *Advances in Neural Information Processing Systems*, I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, Eds., vol. 30. Curran Associates, Inc., 2017. [Online]. Available: https://proceedings.neurips.cc/paper_files/paper/2017/file/3f5ee243547dee91fbd053c1c4a845aa-Paper.pdf

[11] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, “Attention is all you need,” 2023.

[12] T. Kim, J. Kim, Y. Tae, C. Park, J.-H. Choi, and J. Choo, “Reversible instance normalization for accurate time-series forecasting against distribution shift,” in *International Conference on Learning Representations*, 2021. [Online]. Available: https://openreview.net/forum?id=cGDAkQo1C0p

[13] B. Nuttamon.T, “Improving cloud attenuation model for ground irradiance estimation across thailand using cloud images from himawari satellite,” Chulalongkorn university, Tech. Rep., 2023.

A ภาคผนวก

A.1 การตั้งค่า Hyperparameters ในการทดลองที่ 3.4.1 และ 3.4.2

การตั้ง Hyperparameters สำหรับการทดลองที่ 3.4.1 แสดงในตารางที่ 11

Methods	Hyper parameters
Linear/NLinear/DLinear	• `enc_in = 1`
RLSTM	• `enc_in = 8` • `d_model = 64`
Informers/Autoformer/Transformer	• `embed_type=4` • `d_model=128` • `e_layers=2` • `d_layers=1` • `factor=3` • `enc_in=1` • `dec_in=1` • `c_out=1`
PatchTST	• `e_layers=2`, • `n_heads=2`, • `d_model=128`, • `d_ff=2048`, • `dropout=0.3`, • `fc_dropout=0.3`, • `patch_len=16`, • `stride=8`, • `lradj="TST"`, • `pct_start=0.3` • `c_out=1`

ตารางที่ 11: Hyper parameter setting สำหรับแต่ละแบบจำลองที่ใช้ในการทดลองตารางที่ 1 และ 2

A.2 การตั้งค่า Hyperparameters ในการทดลองที่ 3.4.3

การตั้ง Hyperparameters สำหรับการทดลองที่ 3.4.3 แสดงในตารางที่ 3

Methods	Hyper parameters
Linear/NLinear/DLinear	• `enc_in = 8`
RLSTM	• `enc_in = 8` • `d_model = 128`
Informers/Autoformer/Transformer	• `embed_type=4` • `d_model=64` • `e_layers=2` • `d_layers=1` • `factor=3` • `enc_in=8` • `dec_in=8` • `c_out=1`
PatchTST	• `e_layers=2`, • `n_heads=2`, • `d_model=64`, • `d_ff=128`, • `dropout=0.3`, • `fc_dropout=0.3`, • `patch_len=16`, • `stride=8`, • `lradj="TST"`, • `pct_start=0.3` • `c_out=1`

ตารางที่ 12: Hyper parameter setting สำหรับแต่ละแบบจำลองที่ใช้ในการทดลองตารางที่ 3

A.3 การตั้งค่า Hyperparameters ในการทดลองที่ 3.4.4 และ 3.4.5

การตั้ง Hyperparameters สำหรับการทดลองที่ 3.4.3 และ 3.4.5 แสดงในตารางที่ 13

Methods	Hyper parameters
Linear/NLinear/DLinear	• `enc_in = 8`
RLSTM	• `enc_in = 8` • `d_model = 128`
Informers/Autoformer/Transformer	• `embed_type=4` • `d_model=64` • `e_layers=2` • `d_layers=1` • `factor=3` • `enc_in=8` • `dec_in=8` • `c_out=8`
PatchTST	• `e_layers=2`, • `n_heads=2`, • `d_model=64`, • `d_ff=128`, • `dropout=0.3`, • `fc_dropout=0.3`, • `patch_len=16`, • `stride=8`, • `lradj="TST"`, • `pct_start=0.3` • `c_out=1`

ตารางที่ 13: Hyper parameter setting สำหรับแต่ละแบบจำลองที่ใช้ในการทดลองตารางที่ 3

A.4 ตัวอย่างการใช้งาน

โดยตัวอย่างการป้อนค่าเพื่อเรียนรู้โมเดลเป็นดังรูปที่ 34

```
    seq_len=24, 37, 74, 101;
    moving_avg=25, 37;
    pred_len=4;

    label_len=0;
    batch_size=32;
    target=I

    feature_type=S
    num_features=1

    model_name=DLinear \# NLinear / Linear / AutoFormer
    python -u run_longExp.py \
        --is_training 1 \
        --model $model_name$ \
        --moving_avg $moving_avg$ \
        --data CUEE \
        --features $feature_type$ \
        --target $target$ \
        --seq_len $seq_len$ \
        --label_len $label_len$\
        --pred_len $pred_len$ \
        --enc_in $enc_in$ \
        --des 'Exp' \
        --itr 1 --batch_size $batch_size$  --learning_rate 0.005

```

รูปที่ 34: ตัวอย่างการป้อนค่าเพื่อเรียนรู้โมเดล