Time Series Estimation Methods

İsmail Enes Akkal

Advisor: Berk Gökberk

December 25, 2024

TABLE OF CONTENTS

1. Plan

Task	Deadline
Submit timetable	25 October 2024 23:59
Literature Review and Project Planning	8 November 2024
Literature Review / Setup Environment and Tools	15 November 2024
Literature Review	20 November 2024
Submit Progress Report with Preliminary Results	24 November 2024 23:59
Software Development and Implementation	1 December 2024
Software Development and Implementation	8 December 2024
Test the Software Project and Prepare the Final Report	15 December 2024
Submit final report and demo	29 December 2024 23:59

I am planning to do the literature review untill the progress report then I will continue with the software development after the progress report.During the literature review process, I am planning to read atleast 10 article on my project topic. I will test my project by 15 December 2024 at the latest.Then I will take one week for submit my final report and do the demo.

2. Introduction

Time series forecasting has become an essential field of study across various domains, offering critical insights that shape strategic decision-making in areas such as finance, healthcare, and environmental monitoring. As we delve into the time series forecasting, it is essential to first understand the foundational elements that compose time-series data. These elements are critical for accurately modeling and predicting future values. We classify these into three main components: trend, seasonality, and residuals. Each plays a pivotal role in shaping the data's characteristics and influencing the forecasting outcomes. Let's take a closer look at these components to understand their functions and implications in time series analysis.

Trend: The trend represents the long-term direction of the data, illustrating its overall movement over time while excluding seasonal influences and irregular fluctuations. Trends can take on various forms, such as linear, exponential, or parabolic patterns.

Seasonality: Patterns that occur at regular intervals are categorized as seasonality. Examples can include weather changes, economic cycles, and recurring holidays.

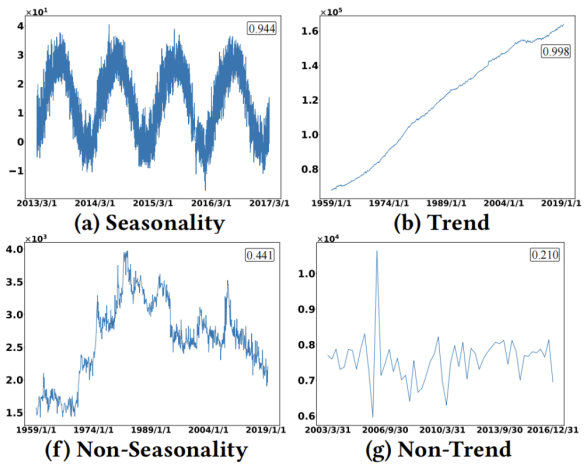

Figure 2.1. Visualization of Seasonality and Trend [1]

Residuals: Once trend and seasonality are accounted for, the remaining variations are referred to as residuals. If these residuals are substantial, they may dominate the observed data, obscuring the underlying trend and seasonal patterns. Identifying the sources of these fluctuations can provide valuable insights, potentially indicating future shifts in the trend. [2]

Traditionally, classical models have focused on parametric approaches that incorporate domain knowledge, including methods like autoregressive models (ARIMA), exponential smoothing, and structural time series models. These classical models, while interpretable and effective in identifying linear relationships and stationary trends, often struggle to capture the intricate nonlinear dependencies and high-dimensional dynamics characteristic of real-world data. [3] In contrast, advanced techniques, such as machine learning, provide a data-driven approach to capturing temporal dynamics, offering flexibility and adaptability across diverse applications [4].

Moreover, recent advancements in machine learning, particularly in deep learning, have spurred the development of sophisticated models adept at identifying complex patterns and representing features and long-term dependencies within data. Variants of Recurrent Neural Networks (RNNs), such as Long Short-Term Memory (LSTM) and Gated Recurrent Units (GRU), along with Convolutional Neural Networks (CNNs) and Transformer models, have been specifically tailored to handle multivariate time series

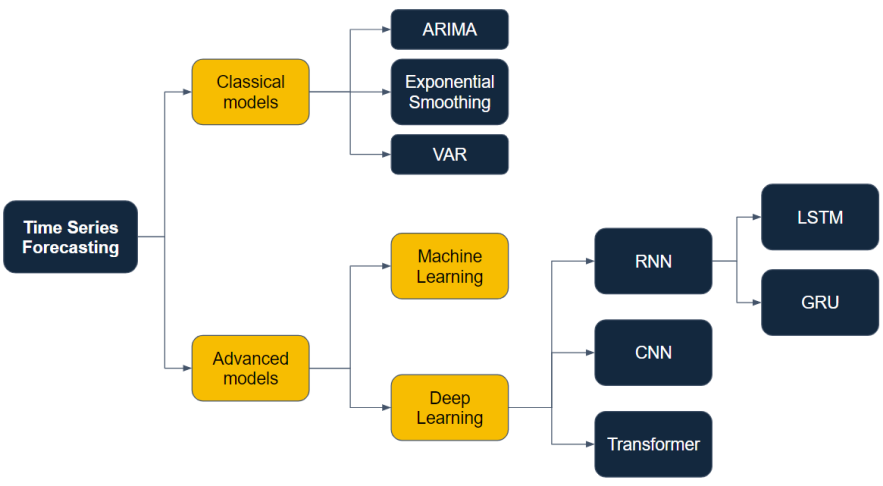

Figure 2.2. Time Series Estimation Methods

data, further refining and enhancing traditional models. Notably, whether applied to univariate or multivariate datasets, as well as in single-step or multi-step forecasting contexts, deep learning models have shown substantial improvements in accuracy [5].

In this paper, we will explore these time series forecasting methods in greater depth, analyzing their capabilities, limitations, and suitability for various types of data. The following sections will provide a comprehensive overview of classical and advanced model, highlighting recent advancements and their impact on predictive accuracy in time series estimation.

3. Literature Review

This section introduces a range of state-of-the-art models that represent significant advancements in time series forecasting.Models such as CARD, iTransofrmer, ARMA employ diverse techniques and architectures to address challenges such as capturing temporal dependencies, modeling variable interactions, and enhancing computational efficiency. We will examine each of these models in detail, evaluating their design, capabilities, and performance across key metrics. From this analysis, we will identify the three most promising models based on their effectiveness and versatility in addressing diverse forecasting scenarios. This approach will provide a comprehensive understanding of current advancements in the field and highlight the models that offer the greatest potential for further development and application.

- **CARD (Channel Aligned Robust Blend Transformer)**: CARD is a transformer-based model designed to improve multivariate time series forecasting by addressing the challenge of channel (variable) dependencies. Traditional transformer architectures often treat each channel independently, which can overlook important relationships across variables. CARD introduces a channel-aligned attention mechanism, which explicitly captures correlations across different variables, enabling it to model interdependencies effectively. Additionally, it incorporates a Token Blend Module, which aggregates tokens at multiple resolutions to capture both fine-grained patterns and long-term trends, enhancing its ability to analyze multi-scale temporal relationships. CARD employs a robust signal decay-based loss function, which prioritizes near-future predictions while down-weighting errors in far-future predictions to counteract overfitting from noise. [6]

- **FITS (Frequency Interpolation Time Series)**: Based on the Multi-Layer Perceptron (MLP) framework,FITS employs the real Fast Fourier Transform (rFFT) to transform time-domain data into its frequency domain equivalent, which consists of complex numbers encoding amplitude and phase information for each frequency component. FITS is designed with a lightweight architecture, containing fewer than 10,000 parameters, making it highly suitable for deployment on edge devices with limited computational resources. To enhance its robustness, FITS employs a Low-Pass Filter (LPF), which eliminates high-frequency noise, focusing instead on the dominant frequency components. This allows the model to capture the essential patterns of the data while avoiding overfitting. [7]
- **ModernTCN (Modernized Temporal Convolutional Network)**: ModernTCN is a modernized version of Temporal Convolutional Networks (TCNs), designed to compete with transformer-based and MLP-based approaches in time series forecasting. Built on the principles of Convolutional Neural Networks (CNNs), ModernTCN incorporates enhancements such as depth-wise and point-wise convolutions, which expand the model's effective receptive field (ERF) to better capture long-term dependencies in time series data. Unlike traditional TCNs, ModernTCN is specifically designed to model both temporal and variable-level dependencies in multivariate datasets, making it highly versatile for tasks such as forecasting, classification, and anomaly detection. The DWConv module captures temporal dependencies for each variable independently, while the ConvFFN modules are divided into two parts: one focuses on learning feature representations within each variable, and the other captures dependencies between variables. This decoupling makes the learning process more efficient and effective for ModernTCN. [8]
- **TimeXer(Empowering Transformers for Time Series Forecasting with Exogenous Variables)**:TimeXer is a Transformer-based model tailored to handle time series forecasting that incorporate both endogenous variables (those being forecasted) and exogenous variables (external factors influencing the forecast). Unlike conventional models that either ignore exogenous data or treat all variables equally, TimeXer bridges endogenous and exogenous data through advanced

embedding and attention mechanisms. The model uses two different levels of representations for the variables. For endogenous variables, TimeXer employs patch-level tokens, where the time series is divided into non-overlapping patches, each representing a segment of the series. For exogenous variables, TimeXer adopts a variate-level tokenization strategy. Each exogenous variable is represented by a single token that encapsulates its overall behavior, avoiding excessive computational complexity. By combining patch-level, variate-level it provides a unified framework for forecasting with exogenous variables, showing good performance in both univariate and multivariate settings. [9]

- **MOMENT(A Family of Open Time-Series Foundation Models)**:MOMENT represents a family of large pre-trained foundation models designed for diverse time series analysis tasks, including forecasting, classification, anomaly detection, and imputation.During pre-training, MOMENT uses a self-supervised approach called masked time series modeling. In this method, parts of the input time series are randomly masked, and the model learns to reconstruct the missing segments. This process helps MOMENT develop a deep understanding of temporal patterns without relying on labeled data. By leveraging the "Time Series Pile" a diverse dataset compilation covering domains like healthcare, engineering, and finance. MOMENT achieves strong performance in zero-shot and few-shot settings. Its patch-based input structure reduces computational complexity and facilitates scalability, making MOMENT an efficient and versatile model [10]
- **SOFTS (Series-Core Fused Time Series Forecaster)** SOFTS is an MLP-based model designed for efficient multivariate time series forecasting. Its core innovation lies in the STAR Aggregate-Redistribute (STAR) module, which aggregates data from multiple channels into a global representation (the "core") and redistributes this information to each channel. This centralized interaction method reduces computational complexity and enhances robustness, outperforming traditional distributed interaction mechanisms like attention. SOFTS ensures linear scalability concerning both the number of channels and the length of the input window. [11]

- **xLSTMTime (Long-Term Time Series Forecasting with Extended LSTM)**: The xLSTMTime model is a specialized architecture for long-term time series forecasting (LTSF), developed by adapting the xLSTM framework originally designed for natural language processing. It addresses key challenges in time series forecasting, such as capturing long-term dependencies, handling nonlinear temporal dynamics, and maintaining computational efficiency. The model begins by decomposing the input time series into trend and seasonal components using learnable moving averages. This decomposition isolates periodic patterns from overarching trends, simplifying the forecasting process.Batch normalization is also applied to stabilize the learning process by standardizing the input data across mini-batches. This technique accelerates training and reduces sensitivity to parameter initialization [12]
- **iTransformer (Inverted Transformer for Time Series Forecasting)**: The iTransformer is a Transformer-based model tailored for time series forecasting, addressing the limitations of traditional Transformers in capturing multivariate correlations. Unlike conventional approaches, which treat temporal tokens (data at each timestamp) as the basic unit for analysis, iTransformer inverts the process by embedding each time series (variable) as an independent token which means that the model treats each variable's entire time series as an independent token, enabling the model to preserve the unique characteristics of each variable while learning complex relationships between them.The model leverages a self-attention mechanism across the variate tokens to capture meaningful correlations between variables. This attention framework identifies which variables influence one another, providing interpretability and improving the accuracy of predictions. Additionally, a feed-forward network (FFN) is employed to model the nonlinear representations of individual variables. [13]

- **ARMA (Autoregressive Moving-Average Attention Mechanism)**:The ARMA model integrates concepts from traditional statistical methods, specifically the Autoregressive Moving Average (ARMA) framework, into modern attention mechanisms. At its foundation, the ARMA Transformer builds on the autoregressive (AR) structure of decoder-only Transformers, commonly used in sequence modeling tasks. Traditional AR mechanisms focus on historical data to make future predictions, but they often fail to account for short-term fluctuations effectively. The model processes time series data by dividing it into patches, applying linear or gated linear attention for the AR term, and dynamically adjusting predictions using the Moving-Average (MA) term. The combination of AR and MA terms improves both local and global pattern modeling, addressing limitations of traditional autoregressive Transformers. [14]
- **TIME-LLM (Reprogramming Large Language Models for Time Series Forecasting)**: TIME-LLM repurposes large language models (LLMs), like GPT, for time series forecasting without modifying their core architecture. Unlike traditional approaches that require task-specific models or fine-tuning for each forecasting scenario, TIME-LLM reprograms time series data into a format that LLMs can interpret as natural language, enabling the use of these pre-trained models without altering their backbone architecture. The method works by first transforming time series data into "patches" and embedding them as text prototypes that align with the input space of LLMs. To further enhance the reasoning capabilities of LLMs for time series tasks, TIME-LLM uses a technique called Prompt-as-Prefix (PaP). This involves adding natural language prompts to guide the model in interpreting the data and generating accurate forecasts. This approach eliminates the need for fine-tuning the LLM itself, making it efficient and resource-friendly. TIME-LLM also excels in few-shot and zero-shot learning scenarios, meaning it can perform well with minimal training data or even without prior task-specific training. [15]

3.1. Detailed Explanation of Used Models

3.1.1. iTransformer

The iTransformer model is an innovative enhancement of the standard Transformer model, specifically designed for forecasting time series data, utilizing a distinct methodology by rearranging how its fundamental components are applied without changing their essential structure. Unlike conventional Transformers, which treat multiple variables recorded at the same timestamp as a single temporal token, iTransformer processes each variable independently across time, embedding them as variate tokens to better capture multivariate correlations. Its encoder-only design includes embedding layers, self-attention mechanisms, feed-forward networks (FFNs), and layer normalization, all fine-tuned to tackle the specific challenges presented by multivariate time series data. The self-attention mechanism focuses on learning dependencies between variables rather than modeling temporal relationships, enabling the model to uncover meaningful interactions among variates.

Meanwhile, the feed-forward networks operate independently on each variate token, extracting complex, non-linear patterns that are essential for accurate forecasting, while layer normalization reduces discrepancies caused by variations in scale and measurement units across variables, further improving stability and generalization. By adopting this inverted perspective, the iTransformer avoids the pitfalls of conventional Transformers, such as performance degradation with longer lookback windows, and effectively leverages larger historical contexts without sacrificing efficiency. Its ability to generalize to unseen variables and datasets highlights its flexibility, making it suitable for a wide range of forecasting tasks. Moreover, the interpretability of its attention mechanism facilitates clear visualization of relationships among variables, which enhances the accuracy of its predictions. [13]

3.1.2. TimeXer

TimeXer is a Transformer-based model developed for time series forecasting that effectively integrates both endogenous variables (the main variables to be predicted) and exogenous variables (external factors influencing the predictions). To handle these two types of data, TimeXer employs a dual embedding strategy, ensuring each is processed differently to capture their distinct roles in forecasting.

For the endogenous series, TimeXer divides the time-series data into small, non-overlapping segments known as patches, with each patch converted into a token to capture local patterns. Given that endogenous variables (internal data) and exogenous variables (external influences) function at distinct levels of detail, their direct combination can lead to misalignment. To address this issue, a global token is incorporated as a summary of the entire series, aiding in the connection and alignment of information between the patches and external variables.

In contrast, the exogenous series is processed differently. Instead of using fine-grained representations like patches, exogenous variables are embedded as variates to handle irregularities such as missing values or different timestamps more efficiently. Each exogenous series is encoded into a single variate token that summarizes its information. This approach minimizes complexity and noise while allowing the model to efficiently incorporate external factors during forecasting.

To integrate information from exogenous variables into the predictions, TimeXer employs a global token as a bridge between the two embeddings. This token serves two purposes. Firstly, it aggregates patterns from the endogenous patches to create a high-level summary. Secondly, it selectively incorporates information from the exogenous variables through a cross-attention mechanism.

Attention Mechanisms in TimeXer Self-Attention for Endogenous Series

- **Patch-to-Patch:** Learns relationships between different patches within the endogenous series, capturing local dependencies.

- **Global-to-Patch:** The global token interacts with each patch, providing contextual information about the overall series behavior.

- **Patch-to-Global:** The global token acts as a summary representation of the entire series and interacts with each patch (temporal token) to exchange information.

- **Cross-Attention (between Endogenous and Exogenous Series):** The global token of the endogenous series interacts with the variate tokens of the exogenous variables, allowing the model to learn how external factors influence the overall behavior of the endogenous series. This cross-attention mechanism uses the endogenous variable as the query and the exogenous variable as the key and value to establish connections between these two variable types. [9]

3.1.3. TimeMixer

The TimeMixer is a multiscale mixing framework designed for time series forecasting, effectively addressing intricate temporal variations through a hierarchical approach. It operates by decomposing input data into multiple scales and processing them with two key components: Past-Decomposable-Mixing (PDM) and Future-Multipredictor-Mixing (FMM). At the core of its architecture, TimeMixer first downsamples the input time series into multiple scales using average pooling, creating representations where finer scales capture detailed patterns and coarser scales emphasize broader trends.This multiscale representation enables the model to disentangle complex patterns into simpler components for more accurate forecasting.

The PDM module focuses on extracting information from past data by decomposing it into seasonal and trend components, treating these separately to capture short-term fluctuations and long-term patterns. It employs two complementary mechanisms

- **Fine-to-Coarse:** Data from finer scales is combined into coarser scales, allowing detailed insights to be understood within wider patterns.
- **Coarse-to-Fine:** Information from coarser scales is incorporated into finer scales, providing a global context for local variations.

This dual-directional approach ensures that both micro-level details and macro-level patterns are well-captured, addressing the complexity of temporal variations. The FMM module takes the processed information from the PDM and generates future forecasts. It employs a set of multiple predictors, each trained on a specific multiscale representation. This strategy leverages the complementary information captured at different scales, as each predictor specializes in forecasting based on the characteristics of its respective scale. By combining the forecasts from these multiple predictors, FMM produces a more robust and accurate prediction than would be possible with a single predictor operating on a single scale. [16]

3.2. Dataset Explanation

3.2.1. ETT Dataset

The ETT dataset contains electricity load and transformer oil temperature data collected from electricity transformers in China over a period of time. It is designed for short-term and long-term forecasting tasks, targeting variables like oil temperature (OT) and other operational features. We've used Etth2 dataset for this dataset. This dataset have a total of 17421 rows.

ETT dataset has four primary variants:

(i) 1. ETTh1 (Hourly):
Contains 7 features including transformer oil temperature (OT).

(ii) 2. ETTh2 (Hourly):
Contains the same features as ETTh1.

(iii) 3. ETTm1 (Minute-level):
Data sampled at 15-minute intervals for higher granularity.
Contains the same 7 features as ETTh1 but with shorter time intervals

(iv) 4. ETTm2 (Minute-level):
Similar to ETTm1 but collected from another transformer system.

The dataset has 8 multivariate time series features, including different parameters and a target variable (OT). The features are:

- DATE
- HUFL (High Useful Load)
- HULL (High Useless Load)
- MUFL (Middle Useful Load)
- MULL (Middle Useless Load)
- LUFL (Low Useful Load)
- LULL (Low Useless Load)
- OT (Oil Temperature)

3.2.2. Jena Climate dataset

We will utilize the Jena Climate dataset, collected by the Max Planck Institute for Biogeochemistry. The data is collected at 10-minute intervals throughout the entire year of 2020, encompassing 22 meteorological variables, including air temperature, humidity, and other atmospheric parameters and Target variable. This dataset have a total of 52697 rows.

- DATE - Date and time of the data record (end)
- p - Air pressure
- T - Air temperature
- Tpot - Potential temperature
- Tdew - Dew point temperature
- rh - Relative humidity
- VPmax - Saturation water vapor pressure
- VPact - Actual water vapor pressure
- VPdef - Water vapor pressure deficit
- sh - Specific humidity
- H2OC - Water vapor concentration
- rho - Air density
- wv - Wind velocity
- max. wv - Maximum wind velocity
- wd - Wind direction
- rain - Precipitation
- raining - Duration of precipitation
- SWDR - Short wave downward radiation
- PAR - Photosynthetically active radiation
- max. PAR - Maximum photosynthetically active radiation
- Tlog - Internal logger temperature
- CO2 - CO_2 concentration of ambient air
- OT - Target Variable

3.3. Performance Evaluation

3.3.1. The Approach for Evaluation Model's Performance

I've set Input Sequence set at 96 for all the case scenarios which means that the model will look at the last 96 time steps to make predictions.Then I picked different prediction lengths for each model so this will be the number of future time steps the

models are going to predict.

List of Prediction Lenths

- 96
- 192
- 336
- 720

Therefore we'll have 4 different cases for each model in which we'll end up with 24 different results for all datasets. What we are expecting is that the models will have lower MSE and MAE results if our prediction length is small like 96 and 192. To demonstrate how does the models perform on the datasets, I will include ground truth vs prediction graphs to have a better idea of how our models work. However, I will only include the ones with prediction length 96 because the file size will be bigger and I might not be able to upload my report due to the file size so you can see the other graphs in the zip folder. The results will be under test results folder in the project.

You can see the results below

3.3.2. iTransformer Model Results on ETTh2 Dataset

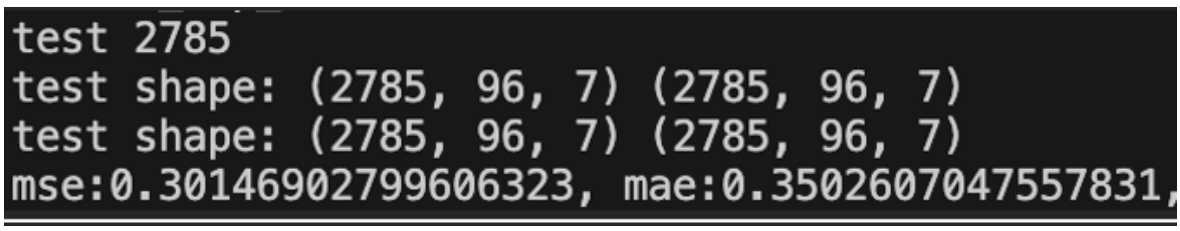

Figure 3.1. Results for iTransformer - CASE 1 (Prediction Length = 96)

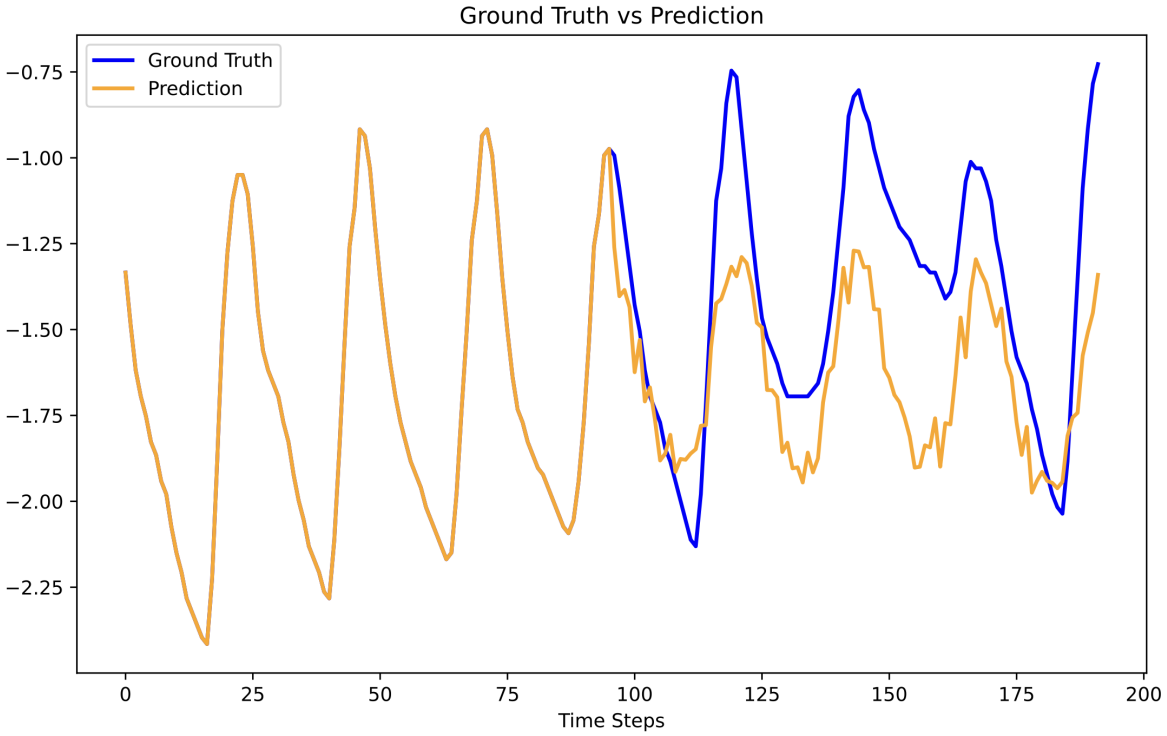

Figure 3.2. Comparison of truth vs prediction with iTransformer model (Prediction Length = 96)

```
test 2689
test shape: (2689, 192, 7) (2689, 192, 7)
test shape: (2689, 192, 7) (2689, 192, 7)
mse:0.3917694091796875, mae:0.40607237815856934
```

Figure 3.3. Results for iTransformer - CASE 2 (Prediction Length = 192)

```
test 2545
test shape: (2545, 336, 7) (2545, 336, 7)
test shape: (2545, 336, 7) (2545, 336, 7)
mse:0.4244594871997833, mae:0.4331867694854736
```

Figure 3.4. Results for iTransformer - CASE 3 (Prediction Length = 336)

```
test 2161
test shape: (2161, 720, 7) (2161, 720, 7)
test shape: (2161, 720, 7) (2161, 720, 7)
mse:0.4274697899818420 4, mae:0.4466357529163360 6
```

Figure 3.5. Results for iTransformer - CASE 4 (Prediction Length = 720)

3.3.3. TimeXer Model Results on ETTh2 Dataset

```
test 2785
test shape: (2785, 96, 7) (2785, 96, 7)
test shape: (2785, 96, 7) (2785, 96, 7)
mse:0.29145270586013794, mae:0.3441474735736847
```

Figure 3.6. Results for TimeXer - CASE 1 (Prediction Length = 96)

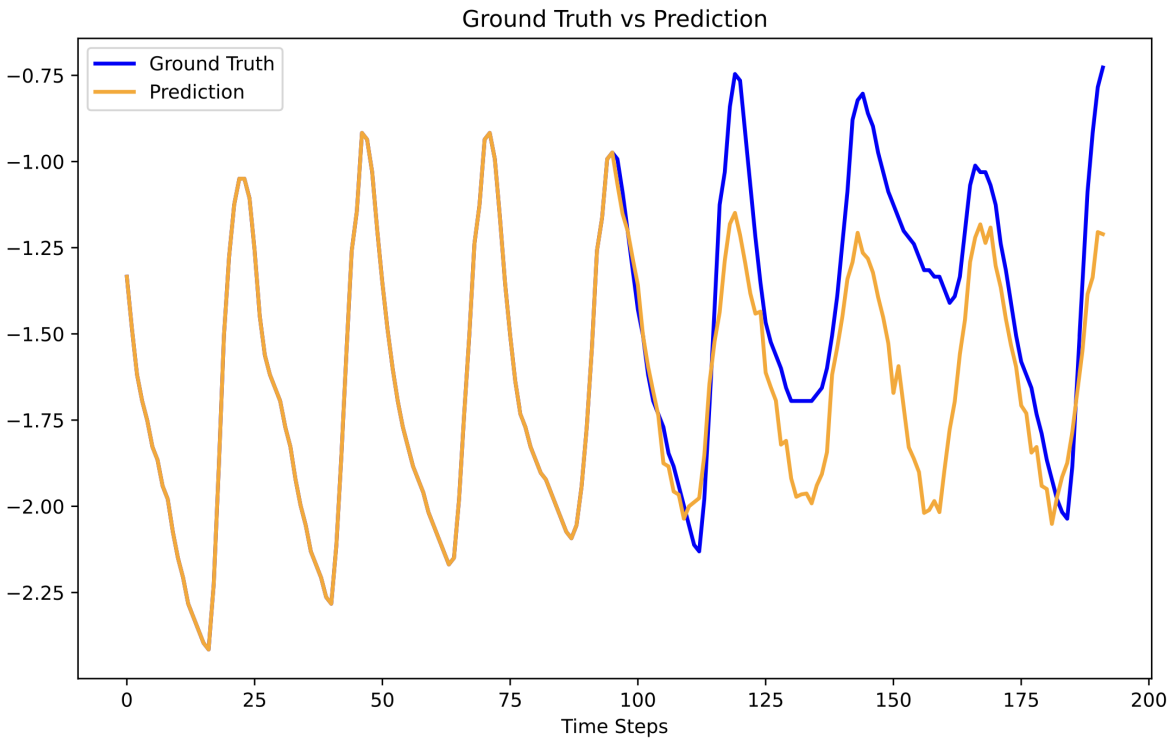

Figure 3.7. Comparison of truth vs prediction with TimeXer model (Prediction Length = 96)

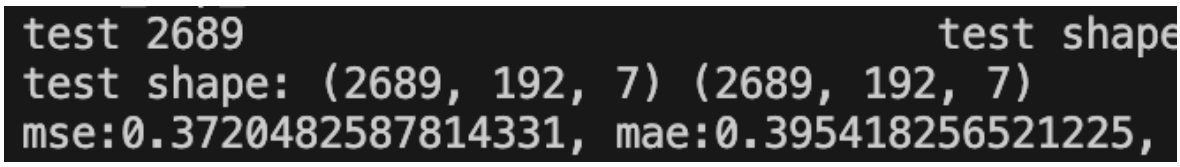

Figure 3.8. Results for TimeXer - CASE 2 (Prediction Length = 192)

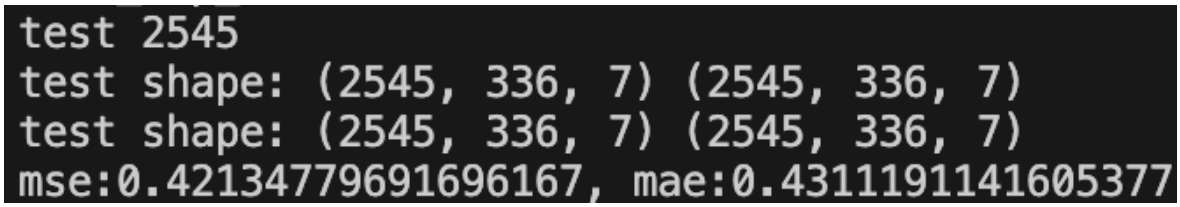

Figure 3.9. Results for TimeXer - CASE 3 (Prediction Length = 336)

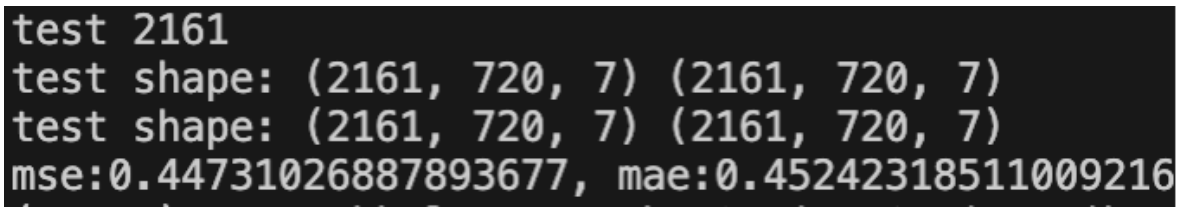

Figure 3.10. Results for TimeXer - CASE 4 (Prediction Length = 720)

3.3.4. TimeMixer Model Results on ETTh2 Dataset

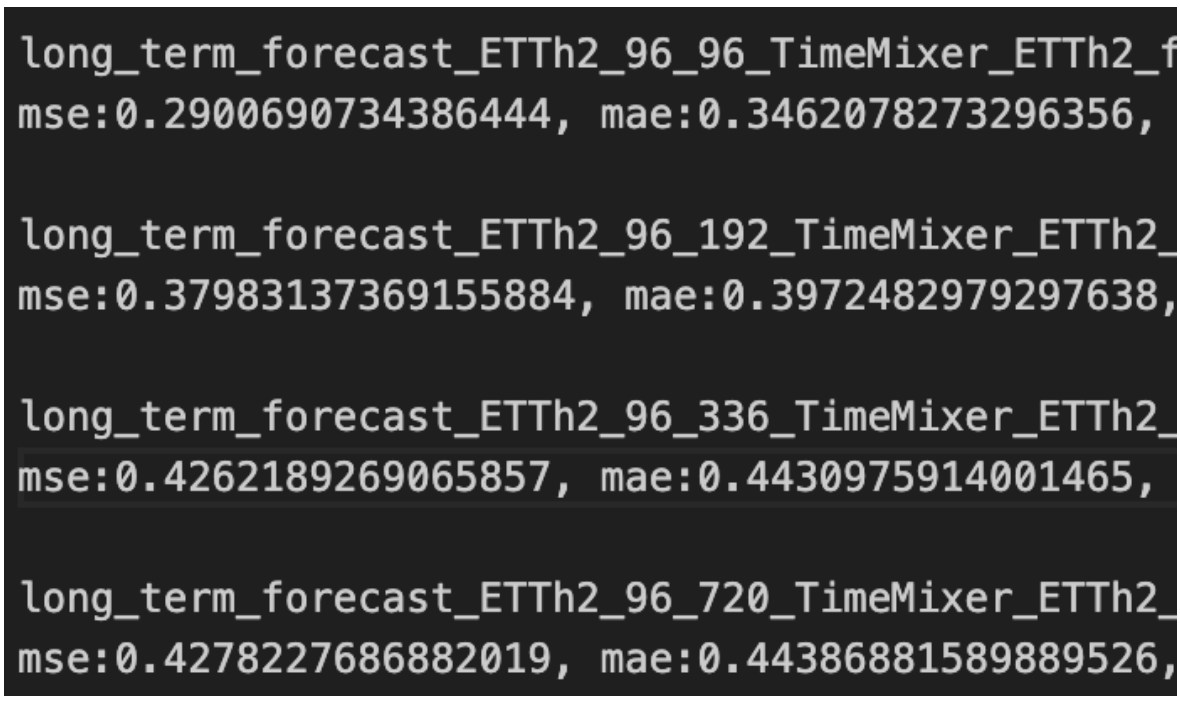

Figure 3.11. Results for TimeMixer - All cases (96,192,336,720))

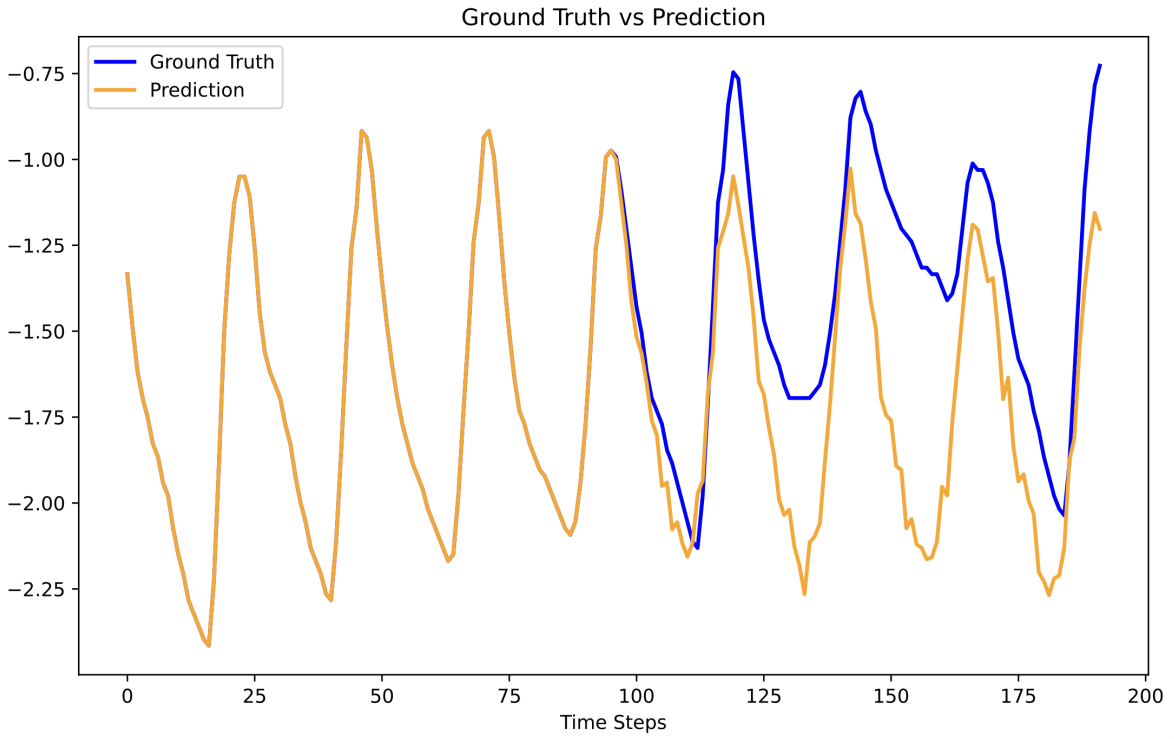

Figure 3.12. Comparison of truth vs prediction with TimeMixer model (Prediction Length = 96))

3.3.5. iTransformer Model Results on Jena Climate Dataset

```
long_term_forecast_weather_96_96_iTransformer_custom_f
mse:0.1790175884962082, mae:0.22067523002624512

long_term_forecast_weather_96_192_iTransformer_custom_
mse:0.2288939356803894, mae:0.26163849234580994

long_term_forecast_weather_96_336_iTransformer_custom_
mse:0.2820242941379547, mae:0.29859545826911926

long_term_forecast_weather_96_720_iTransformer_custom_
mse:0.3592252731323242, mae:0.3498910069465637
```

Figure 3.13. Results for iTransformer - All cases (96,192,336,720))

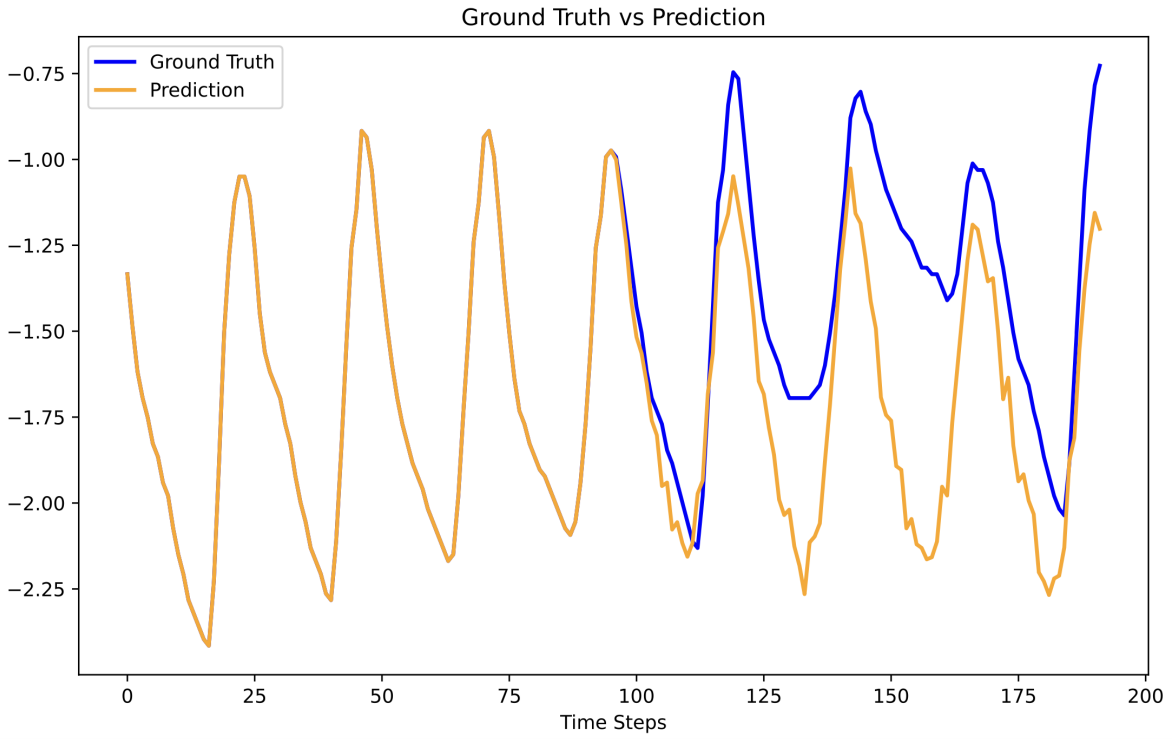

Figure 3.14. Comparison of truth vs prediction with iTransformer model (Prediction Length = 192))

3.3.6. TimeXer Model Results on Jena Climate Dataset

```
long_term_forecast_ETTh2_96_96_TimeXer_ETTh2_ftM_
mse:0.29145270586013794, mae:0.3441474735736847

long_term_forecast_ETTh2_96_192_TimeXer_ETTh2_ftM
mse:0.3720482587814331, mae:0.395418256521225

long_term_forecast_ETTh2_96_336_TimeXer_ETTh2_ftM
mse:0.42134779691696167, mae:0.4311191141605377

long_term_forecast_ETTh2_96_720_TimeXer_ETTh2_ftM
mse:0.44731026887893677, mae:0.45242318511009216
```

Figure 3.15. Results for TimeXer - All cases (96,192,336,720))

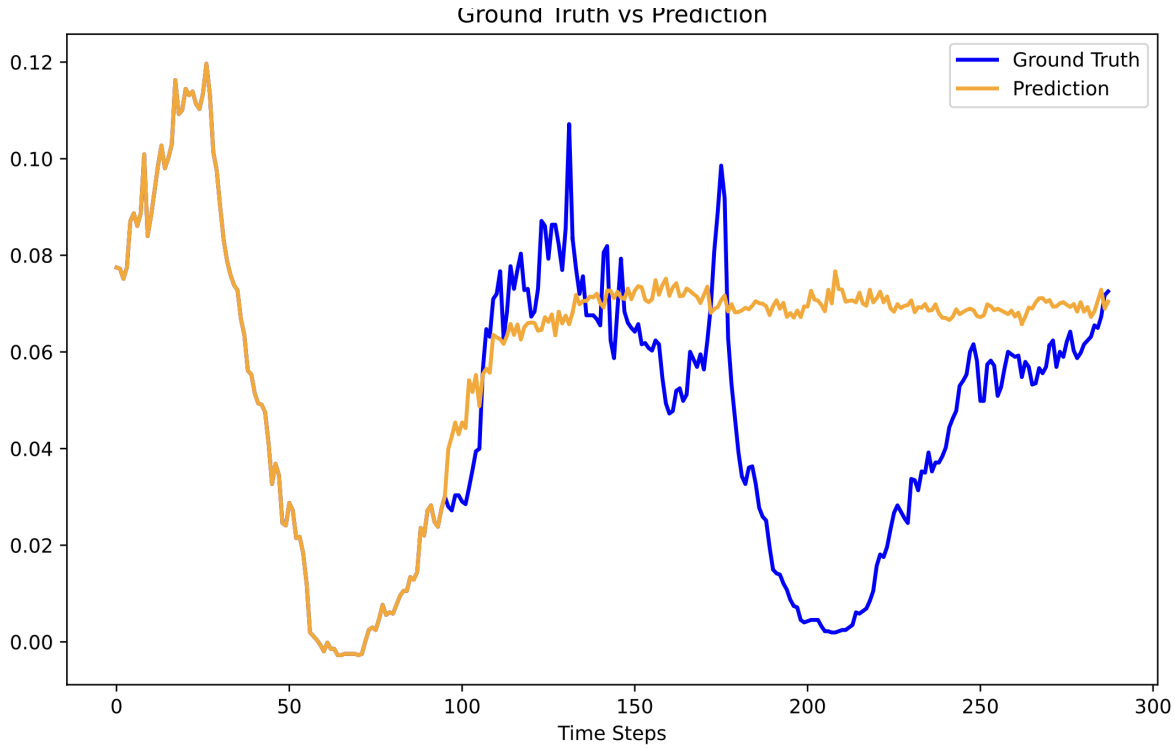

Figure 3.16. Comparison of truth vs prediction with TimeXer model (Prediction Length = 192))

3.3.7. TimeMixer Model Results on Jena Climate Dataset

```
long_term_forecast_weather_96_96_TimeMixer_custo
mse:0.16250614821910858, mae:0.2107288390398025

long_term_forecast_weather_96_192_TimeMixer_cust
mse:0.2096908986568451, mae:0.2519375681877136,

long_term_forecast_weather_96_336_TimeMixer_cust
mse:0.263397216796875, mae:0.2917398810386657,

long_term_forecast_weather_96_720_TimeMixer_cust
mse:0.3443117141723633, mae:0.3447409570217132,
```

Figure 3.17. Results for TimeMixer - All cases (96,192,336,720))

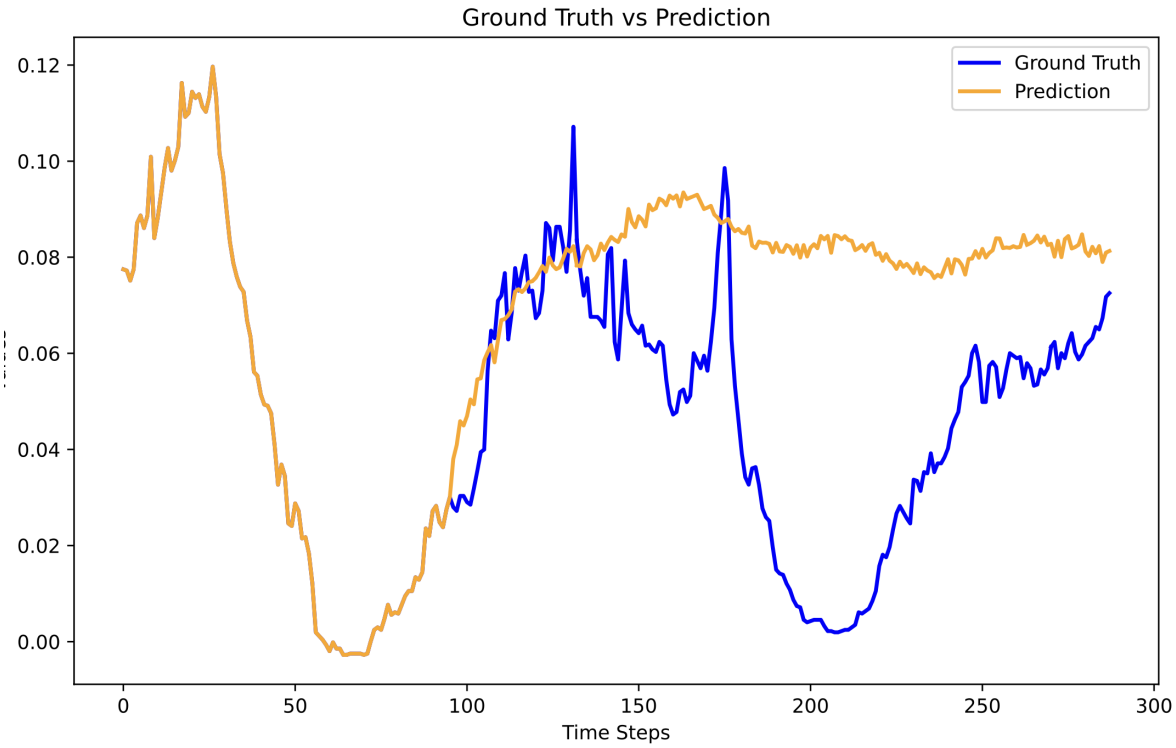

Figure 3.18. Comparison of truth vs prediction with TimeMixer model (Prediction Length = 192))

4. Conclusion

The performance evaluation of iTransformer, TimeXer, and TimeMixer across the ETTh2 and Weather datasets provides significant insights into their effectiveness in long-term time series forecasting. For the ETTh2 dataset, TimeMixer demonstrated superior performance at shorter prediction horizons, achieving the lowest MSE (0.2901) and MAE (0.3462) at 96 prediction steps.However, as the prediction length increased, iTransformer and TimeXer displayed improved scalability. At 192 prediction steps, TimeXer outperformed iTransformer with an MSE of 0.3720 and MAE of 0.3954, compared to iTransformer's MSE of 0.3918 and MAE of 0.4061. For longer horizons, such as 336 and 720 steps, both models maintained competitive results, with iTransformer slightly outperforming TimeXer at 720 steps.

For the Weather dataset, TimeMixer achieved the best results at shorter horizons, with an MSE of 0.1625 and MAE of 0.2107 at 96 steps, outperforming iTransformer, which produced an MSE of 0.1790 and MAE of 0.2207. At 192 steps, TimeXer demonstrated competitive performance, recording an MSE of 0.2026 and MAE of 0.2458, marginally outperforming iTransformer (MSE: 0.2289, MAE: 0.2616). For longer horizons, such as 336 and 720 steps, all models experienced an increase in forecasting errors. However, TimeMixer remained slightly more robust, achieving an MSE of 0.3443 and MAE of 0.3447 at 720 steps.

In conclusion, the performance of all models demonstrated a consistent decline as the forecasting horizons increased in length, irrespective of the dataset being used. This trend indicates that the models struggled with accuracy when making predictions over extended periods. Conversely, the models showed a relatively superior ability to forecast outcomes for shorter horizons, suggesting that their predictive capabilities are more robust within shorter horizons like 96 or 192 steps.

REFERENCES

1. Qiu, X., J. Hu, L. Zhou, X. Wu, J. Du, B. Zhang, C. Guo, A. Zhou, C. S. Jensen, Z. Sheng and B. Yang, "TFB: Towards Comprehensive and Fair Benchmarking of Time Series Forecasting Methods", , 2024, https://arxiv.org/abs/2403.20150.

2. Fatima, S. S. W. and A. Rahimi, "A Review of Time-Series Forecasting Algorithms for Industrial Manufacturing Systems", *Machines*, Vol. 12, No. 6, 2024, https://www.mdpi.com/2075-1702/12/6/380.

3. Wang, Y., H. Wu, J. Dong, Y. Liu, M. Long and J. Wang, "Deep Time Series Models: A Comprehensive Survey and Benchmark", , 2024, https://arxiv.org/abs/2407.13278.

4. Lim, B. and S. Zohren, "Time-series forecasting with deep learning: a survey", *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences*, Vol. 379, No. 2194, p. 20200209, Feb. 2021, http://dx.doi.org/10.1098/rsta.2020.0209.

5. Liu, X. and W. Wang, "Deep Time Series Forecasting Models: A Comprehensive Survey", *Mathematics*, Vol. 12, No. 10, 2024, https://www.mdpi.com/2227-7390/12/10/1504.

6. Xue, W., T. Zhou, Q. Wen, J. Gao, B. Ding and R. Jin, "CARD: Channel Aligned Robust Blend Transformer for Time Series Forecasting", , 2024, https://arxiv.org/abs/2305.12095.

7. Xu, Z., A. Zeng and Q. Xu, "FITS: Modeling Time Series with $10k$ Parameters", , 2024, https://arxiv.org/abs/2307.03756.

8. donghao, L. and wang xue, "ModernTCN: A Modern Pure Convolution Structure for General Time Series Analysis", *The Twelfth International Conference on Learn-*

ing Representations, 2024, https://openreview.net/forum?id=vpJMJerXHU.

9. Wang, Y., H. Wu, J. Dong, G. Qin, H. Zhang, Y. Liu, Y. Qiu, J. Wang and M. Long, "TimeXer: Empowering Transformers for Time Series Forecasting with Exogenous Variables", , 2024, https://arxiv.org/abs/2402.19072.

10. Goswami, M., K. Szafer, A. Choudhry, Y. Cai, S. Li and A. Dubrawski, "MOMENT: A Family of Open Time-series Foundation Models", , 2024, https://arxiv.org/abs/2402.03885.

11. Han, L., X.-Y. Chen, H.-J. Ye and D.-C. Zhan, "SOFTS: Efficient Multivariate Time Series Forecasting with Series-Core Fusion", , 2024, https://arxiv.org/abs/2404.14197.

12. Alharthi, M. and A. Mahmood, "xLSTMTime : Long-term Time Series Forecasting With xLSTM", , 2024, https://arxiv.org/abs/2407.10240.

13. Liu, Y., T. Hu, H. Zhang, H. Wu, S. Wang, L. Ma and M. Long, "iTransformer: Inverted Transformers Are Effective for Time Series Forecasting", , 2024, https://arxiv.org/abs/2310.06625.

14. Lu, J., X. Han, Y. Sun and S. Yang, "Autoregressive Moving-average Attention Mechanism for Time Series Forecasting", , 2024, https://arxiv.org/abs/2410.03159.

15. Jin, M., S. Wang, L. Ma, Z. Chu, J. Y. Zhang, X. Shi, P.-Y. Chen, Y. Liang, Y.-F. Li, S. Pan and Q. Wen, "Time-LLM: Time Series Forecasting by Reprogramming Large Language Models", , 2024, https://arxiv.org/abs/2310.01728.

16. Wang, S., H. Wu, X. Shi, T. Hu, H. Luo, L. Ma, J. Y. Zhang and J. Zhou, "TimeMixer: Decomposable Multiscale Mixing for Time Series Forecasting", , 2024, https://arxiv.org/abs/2405.14616.