Machine Learning Techniques for Prediction of Time Series Data (SCSE23-0396)

Submitted by: Andrew Ng Yong Kuan

Supervisor: Associate Professor Yeo Chai Kiat

Examiner: Dr Liu Siyuan

School of Computer Science and Engineering

A final year project report presented to the Nanyang Technological University in partial fulfilment of the requirements of the degree of Bachelor of Engineering (Computer Science)

Academic Year: 2023/2024

Abstract

Forecasting time series data accurately is paramount across a spectrum of disciplines ranging from finance to environmental science. In recent years, the application of advanced machine learning techniques, particularly deep learning models, has shown promising results in this field. This report delves into the innovative integration of stationary wavelet transformation (SWT) and transformer-based models to enhance time series data prediction accuracy.

The study systematically evaluates the performance of a hybrid model that combines SWT and transformers. Firstly, SWT is employed as a preprocessing step to decompose the time series data into different frequency components. Subsequently, these components serve as input to a transformer-based model designed to capture complex temporal dependencies. Through empirical analysis of several benchmark time series datasets, this report aims to demonstrate that the hybrid approach outperforms traditional methods and standalone transformer models in terms of prediction accuracy.

Furthermore, the adaptability of the hybrid model opens avenues for the exploration of other wavelet transformations. This study proposes the Maximal Overlap Discrete Wavelet Transform (MODWT) as a viable alternative to address the limitations encountered with the Stationary Wavelet Transformation (SWT).

In summation, the synergy between wavelet transformations and transformer models presents a promising avenue for advancing time series prediction, offering insights that could facilitate more informed decision-making across multiple domains.

Acknowledgements

I would like to express my sincere gratitude to my supervisor, Professor Yeo Chai Kiat, for her invaluable guidance, support, and mentorship throughout this project. Her expertise, insightful feedback, and unwavering encouragement have been instrumental in shaping the direction and quality of this report. I am deeply grateful for the opportunity to learn from her wealth of knowledge and experience in Machine Learning.

I would also like to extend my heartfelt thanks to my supervisor’s graduate student, Chan Jun Wei, for his assistance and dedication during this academic journey. His assistance in providing clarification, reviewing drafts, and offering constructive suggestions has been immensely beneficial in refining the content and presentation of this report.

Lastly, I extend my appreciation to my family and friends for their unwavering encouragement and understanding throughout this endeavor.

Table of Contents

List of Figures

List of Tables

1 Introduction

1.1 Background

Time series forecasting plays a crucial role in numerous fields, such as finance [1], weather prediction [2], inventory management [3], and more, by providing a systematic way to project future values based on historical data. This predictive capability is invaluable for planning, decision-making, and strategizing in both business and scientific contexts. By analyzing trends, seasonal patterns, and cyclic behavior in past data, businesses and researchers alike can stay one step ahead, optimizing processes and minimizing potential setbacks.

The toolkit for forecasting these time series is diverse, spanning from classical statistical approaches to cutting-edge machine learning algorithms. Statistical models include ARIMA (Autoregressive Integrated Moving Average), which captures various temporal structures in the data, and Exponential Smoothing, which weights past observations differently to forecast future values. While statistical models are still relevant in several applied domains due to their simplicity and straightforward interpretability, they tend to fail when faced with highly intricate scenarios, such as modeling nonlinear and nonstationary time series. To address such situations, machine learning techniques, such as Transformers, employ complex algorithms to model time series data by capturing nonlinear patterns and interactions that traditional methods might miss.

1.2 Motivation

In the rapidly advancing field of deep learning, the Transformer model has emerged as a revolutionary architecture, particularly within the domain of natural language processing (NLP) and related areas. Unveiled by Vaswani and colleagues in 2017 [4], the Transformer model introduced the concept of self-attention mechanisms, enabling the parallel processing of data sequences. This marked a departure from the conventional sequential processing techniques employed by Recurrent Neural Networks (RNNs) and their derivatives, offering a leap in both efficiency and performance. The Transformer architecture has catalyzed unprecedented progress in various tasks, including machine translation [5], text summarization [6], and language comprehension [7], thereby redefining standards across diverse applications.

Despite its groundbreaking performance, the Transformer architecture is not devoid of challenges. One of the key challenges [8] lies in its handling of long sequence data, where the self-attention mechanism can struggle to capture and model long-term dependencies efficiently. This issue becomes particularly pronounced in scenarios demanding the analysis of intricate sequences over prolonged durations, a common requirement in areas such as time series forecasting.

To address this shortcoming, the integration of advanced signal processing methods like Wavelet decomposition with Transformer architectures emerges as a promising strategy. Wavelet decomposition [9] is a powerful tool for signal processing, enabling the decomposition of a time series into components at different frequencies, providing a multi-resolution analysis that captures both the temporal and frequency characteristics of the signal. By employing Wavelet decomposition for preliminary data processing, long-term dependencies and patterns within time series sequences are more effectively highlighted and represented in a form that enhances the Transformer model's ability to learn from such data.

1.3 Project Objectives

This project aims to explore the innovative integration of Wavelet Decomposition techniques with Transformer models to enhance the prediction accuracy of future time steps in time series data. The project will specifically leverage these wave transformation algorithms, Stationary Wavelet Transform (SWT) and Maximal Overlap Discrete Wavelet Transform (MODWT), for data preprocessing. The project will also explore and evaluate variations of Transformer models in processing wavelet-transformed inputs, assessing their impact on forecasting accuracy.

The research is expected to demonstrate the potential of Wavelet Decomposition as a preprocessing step to improve the performance of Transformer models in time series forecasting. By providing detailed analysis and comparative data, the project aims to contribute to the advancement of forecasting methodologies and offer insights into the synergistic potential of combining advanced signal processing techniques with deep learning models.

1.4 Scope of Project

This project will evaluate different variations of Transformer Models made by past contributors. These models will be compared to classical statistical models and other machine learning models such as Auto Regressive Integrated Moving Average (ARIMA) and Time Series Recurrent Neural Network (tsRNN).

This project will use 4 different datasets as experiments:

1. London residential load data – London [10], collected from London smart meter data store
2. PJM system load data – AEP [11], electricity consumption in Michigan and a part of Ohio recorded by the company AEP under PJM Interconnection LLC
3. PJM system load data – DAYTON [11], electricity consumption in Michigan and a part of Ohio recorded by the company AEP under PJM Interconnection LLC
4. Spain – REE [12], Energy consumption generation prices and weather by Spain RED Electrica de Espana

1.5 Project Schedule

The Gantt chart in Figure 1.1 details the project progress over time.

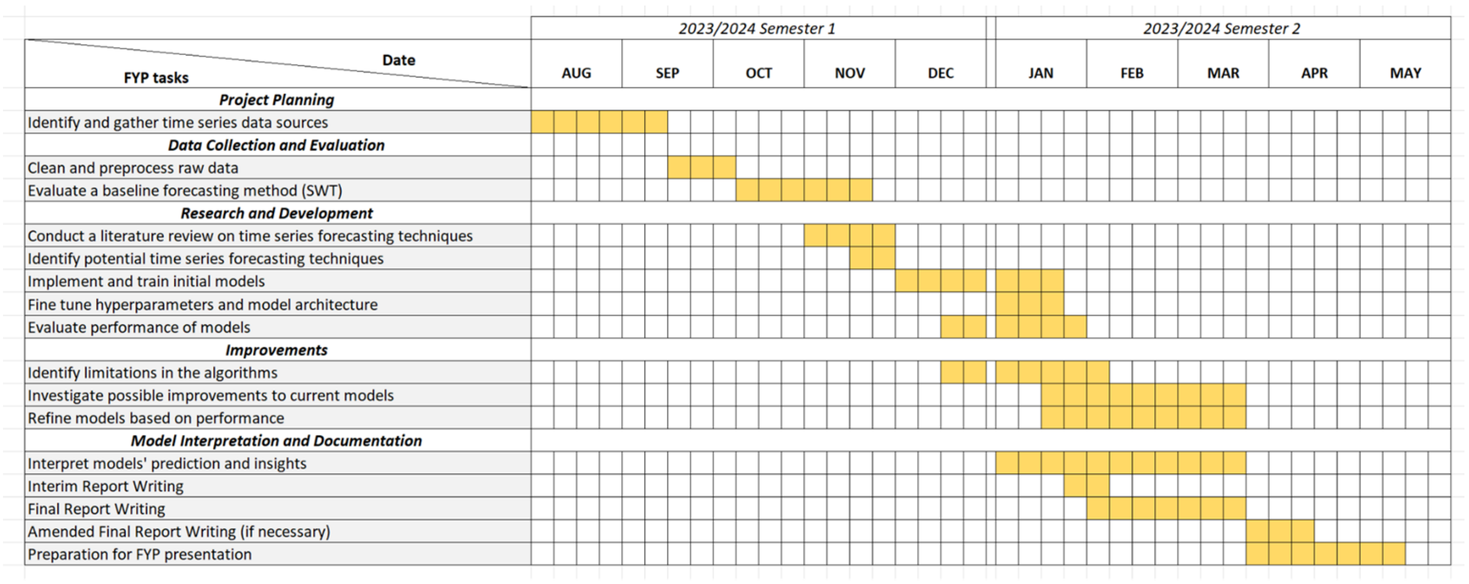


Figure 1.1 Project Progress Gantt Chart

1.6 Organization of Report

The rest of this report is organized as follows: section 2 discusses the literature review and related works, section 3 elaborates the initial proposed method, and section 4 details the experiments conducted, presents the results and shares the limitations and solutions. Finally, section 5 concludes this paper.

2 Literature Review

This section discusses the mathematical foundation of the wavelet decomposition and transformer architecture to be used in our proposed hybrid Wavelet Transformer model.

2.1 Wavelet Decomposition

Wavelet decomposition [13], [14] is a mathematical technique used for signal processing and data analysis, which involves breaking down a signal into components that vary over different scales or frequencies. Unlike Fourier transform [15], which represents a signal as a sum of sinusoids, wavelet transform uses wavelets that are localized in both time and frequency. This process provides a way for analyzing the signal at various levels of detail, making it particularly suitable for analyzing signals with transient or non-stationary characteristics [16].

2.2 Discrete Wavelet Transform (DWT)

There are two main types of wavelet transform [17]: Continuous Wavelet Transform (CWT) and Discrete Wavelet Transform (DWT). CWT provides a continuous representation of the signal and is useful for detailed analysis, while DWT is more computationally efficient and is often used in applications like signal compression and noise reduction. This project will primarily focus on DWT since it is frequently used in energy sectors like Seismic [18], Hydrologic [19], Wind power [20], precipitation [21] and many other sectors.

The essence of DWT lies in its ability to capture both high-frequency and low-frequency components of a signal at different scales, through the process of decomposition [22]. The signal is passed through a series of high-pass and low-pass filters, resulting in approximation (low-frequency) and detail (high-frequency) components.

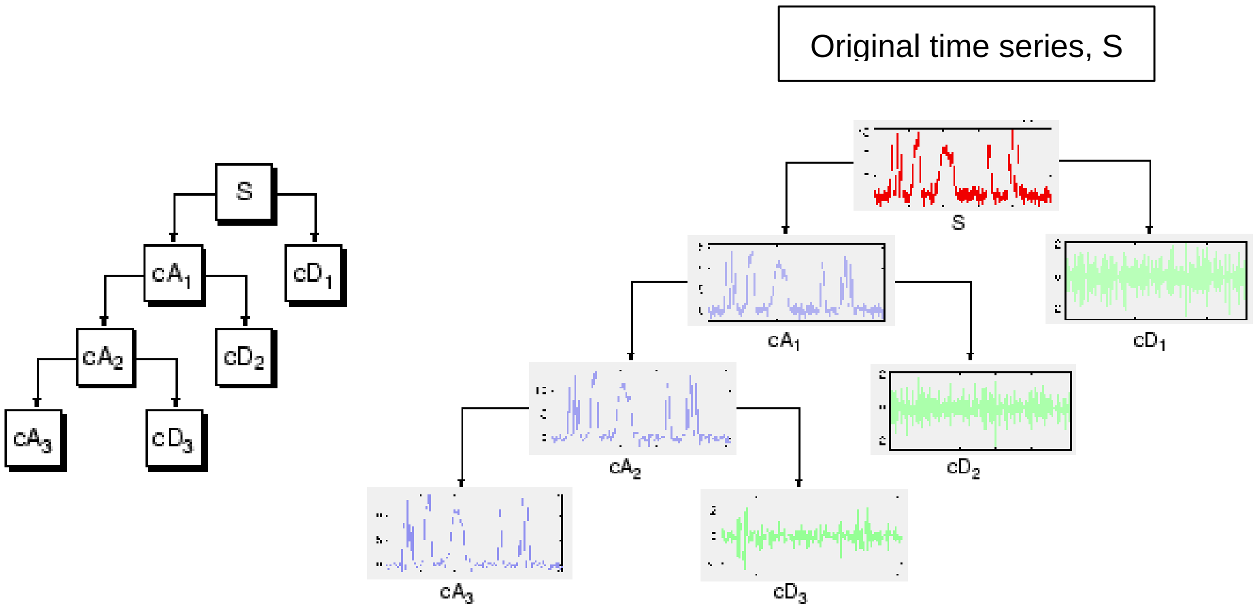


Figure 2.1 Original signal, S, multi-level decomposition process [22]

The DWT process [23] can be iterated multiple times, with each level of decomposition providing finer resolution and detail as illustrated in Figure. 2.1. Similarly, this process can be reversed to reconstruct a signal from its components as illustrated in Figure 2.2. The mathematical equations can be found in [24].

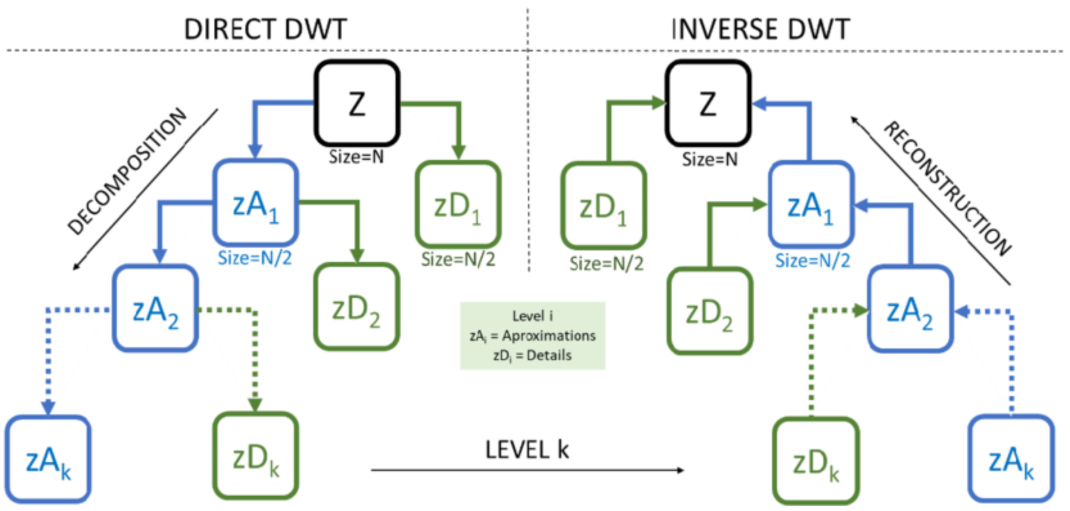


Figure 2.2 Decomposition and Reconstruction process with DWT (Signal Z)

DWT is computationally efficient and suitable for applications where signal discontinuities and sharp spikes are of interest. However, DWT has several limitations, such as shift sensitivity, poor temporal localization in higher frequency bands and the requirement for signal length to be a power of two, which may necessitate padding in practical applications [25].

2.3 Stationary Wavelet Transform (SWT)

The Stationary Wavelet Transform [26], [27], also known as the "Undecimated Wavelet Transform" or "Translation-Invariant Wavelet Transform," is a variation of the DWT designed to overcome one of its main limitations: the lack of translation invariance. The translation invariance issue with DWT means that if the input signal is shifted in time, its wavelet transform does not merely shift but can also change in more complex ways, which might affect the analysis.

SWT addresses this by omitting the down-sampling step that is part of the DWT process [28]. Instead of discarding samples after each filtering operation (as DWT does), SWT retains all samples, leading to a redundant representation of the signal. This redundancy ensures that the SWT is translation-invariant, making it more robust for applications where the exact positioning of features within a signal is crucial.

2.4 Maximally Overlapping Discrete Wavelet Transform (MODWT)

MODWT [29], [30], [31], [32], also known as the non-decimated wavelet transform, addresses some of the limitations of DWT. It is fully shift-invariant and does not require the signal to be of any specific length (not power of two). MODWT achieves this by retaining all scales of the wavelet coefficients at each decomposition level, leading to redundant representations. This redundancy is particularly advantageous in analyzing non-stationary signals and provides better frequency localization. However, these advantages come at the cost of higher computational costs and increased memory requirements due to redundancy. This redundant representation might not be ideal for all applications, especially those with stringent memory or computational constraints.

2.5 Transformers in time series forecasting

Transformers, initially designed for tasks in natural language processing (NLP), have found a groundbreaking application in the field of time series forecasting [33]. The core innovation of Transformers, the self-attention mechanism [34], enables the model to weigh the importance of different parts of the input data dynamically. This feature is especially beneficial for time series forecasting, where the relevance of past observations can vary significantly over time.

The adaptation of Transformers for time series forecasting involves several key modifications [35], [36]:

- **Input Representation**
    - Unlike textual data, time series data is numeric and continuous. To adapt Transformers for time series forecasting, the input data needs to be encoded into a format that the model can process. This often involves embedding time series data into a higher-dimensional space, similar to word embeddings in NLP, to capture the nuances of the data.
- **Positional Encoding**
    - In NLP, positional encodings are added to input embeddings to provide information about the position of words in a sequence. For time series forecasting, positional encodings are adapted to convey the temporal order of observations. This includes linear time and cyclic representations to capture patterns like seasonality and trends.
- **Output Layer Adaptation**
    - The original Transformer uses a final output layer suited for classification tasks (e.g., predicting the next word in a sentence). For time series forecasting, the output layer is often modified to produce continuous values that predict future points in the series.

In addition to the change in architecture to cater for time series data inputs, numerous research focus on introducing new time-aware self-attention mechanisms. These mechanisms are designed to give the model a better understanding of temporal dynamics.

- **Time-aware Attention** [37]
    - This variant of the self-attention mechanism allows the model to factor in the elapsed time between observations. This is crucial for handling irregularly spaced time series or understanding the significance of time gaps in data.
- **Dilated Attention** [38], [39]
    - Also known as **Sparse Attention** or **Fixed Pattern Attention**.
    - Inspired by dilated convolutions in CNNs, dilated attention allows the Transformer to efficiently handle long sequences by skipping over certain intervals in the input data. This approach helps in capturing long-range dependencies without overwhelming the model with too much information at once.

- **Causal Attention** [40]
    - For forecasting, it is essential that the model only uses information from the past to predict the future, avoiding any "look-ahead" bias. Causal attention ensures that the self-attention mechanism only considers previous and current time steps, not future ones.

Adapting Transformers for time series also involves addressing seasonality and trends, which are common in time series data but not in textual data. This can involve incorporating additional layers or mechanisms specifically designed to model and predict these patterns. For instance, incorporating Wavelet decomposition as part of the inputs can help the model capture and utilize cyclical patterns in the data.

2.6 Related Works

Since the proposal of Transformers in a 2017 paper "Attention Is All You Need" by Google [4], there has been a rise in the popularity of transformers on all fronts of machine learning challenges such as natural language processing, image processing, and many more. As research in transformers continues to grow, we can observe the integration of other theories and concepts into this architecture. Here is some research that is closely related to our project.

FEDformer [41], which uses Fourier transform in conjunction with Transformers, was able to reduce prediction error in 6 benchmark datasets across multiple domains (energy, traffic, economics, weather and disease) by 14.8% and 22.6% for multivariate and univariate time series, respectively. Fourier transformation was more commonly used as a data preprocessing step prior to the evolution of wavelet decomposition.

Wavelet decomposition was integrated with ARIMA, a statistical model, in a one month forecast of the casualties cases of COVID-19 [42]. Based on their analysis of 5 different countries (Italy, France, Spain, USA, UK), the results clearly show that the integration of wavelet decomposition improves ARIMA’s predictive capabilities in all metrics (MAE, RMSE, MAPE, R-square).

A Continuous Wavelet Sliding Vision Transformer [43] was proposed to tackle image denoising for image restoration tasks. DWT is first performed along a sliding window to extract deep features of an image. The wavelet values are then used as inputs to a Vision Transformer where a CNN decoder aims to reconstruct the deep features into denoised images. The sliding window methodology of this paper closely relates to one of the methods used in this project.

Most research on transformers in time series forecasting claims a significant improvement in predictive capabilities than older or simpler machine learning models. Transformers have indeed proven their revolutionizing capabilities as a core architecture in Large Learning Models (LLMs) [44], [45], such as ChatGPT and Google Gemini. However, its capabilities in time series forecasting are still under question. Researchers at the Chinese University of Hong Kong have questioned the validity of the research on transformers in time series forecasting [46]. In their paper, they compared LTSF-Linear models, a set of simple one-layer linear models, with existing Transformer-based LTSF models [41]. Their results over 9 popular datasets consistently show that simple models perform better than Transformers. The conclusion to question the forecasting results of transformers aided this project in debunking some suspicious results faced later in this report.

3 Methodology

3.1 Outline

The project will be carried out in stages, beginning with the establishment of a baseline using a conventional Transformer model applied to various datasets. Following this, the datasets will undergo preprocessing using Stationary Wavelet Transform (SWT) for Wavelet Decomposition. The decomposed datasets will then be fed into Transformer models, including standard and modified versions, to evaluate the impact of wavelet values on forecasting performance. Comparative analysis will be conducted to assess the effectiveness of different approaches, with a focus on accuracy metrics and computational efficiency.

3.2 Data Description

As per the project's requirements, the proposed model and all subsequent models will be evaluated on 4 different electrical consumption datasets. Additional information on the details and distribution of these datasets are as follows:

- London residential load data – London [10]
    a. London dataset is collected from the London smart meter data store. It contains the energy consumption readings of 5,567 households in 112 blocks between November 2011 and February 2014 in kilowatt-hours (kWh).
    b. London dataset consists of 5,567 half-hourly series, all of which have differing starting and ending dates. The min and max consumption values in each series are different from one another, and all consumption values range between 0 and 10 kWh.
- PJM system load data – AEP [11]
    a. PJM system load data is collected from the website of PJM Interconnection LLC, which is a regional transmission organization in the United States operating an electricity transmission system serving 14 states of the US.
    b. The sub dataset, AEP, records the electricity consumption in Michigan and it is recorded by the company AEP. It consists of one hourly series from 2 Oct 2004 00:00 to 2 Aug 2018 23:00 with 121,272 time steps in megawatts (MW). The min and max consumption values in the AEP series are 9581.0 and 25695.0 megawatts respectively.
- PJM system load data – DAYTON [11]

a. Similarly, this dataset is part of the PJM system load data that is collected from the website of PJM Interconnection LLC

b. The sub dataset, DAYTON, records the electricity consumption in a part of Ohio and it is recorded by the company DAYTON. Similarly to AEP, it consists of one hourly series from 2 Oct 2004 00:00 to 2 Aug 2018 23:00 with 121,272 time steps in megawatts (MW). The min and max consumption values in the DAYTON series are 982.0 and 3746.0 megawatts respectively.

- Spain – REE [12]

  a. This dataset records energy consumption generation prices and weather by Spain RED Electrica de Espana. it consists of one hourly series from 1 Jan 2015 00:00 to 31 Dec 2018 23:00 with 35,040 time steps in megawatts (MW). The consumption values range from 18,041 MW to 41,015 MW.

3.3 Transformer Architecture

The transformer architecture used in this project follows closely to the original papers [4]. The core components are stacks of encoders and decoders that are based on self-attention mechanisms, which allows them to weigh the importance of different parts of the input data differently. See Figure 3.1.

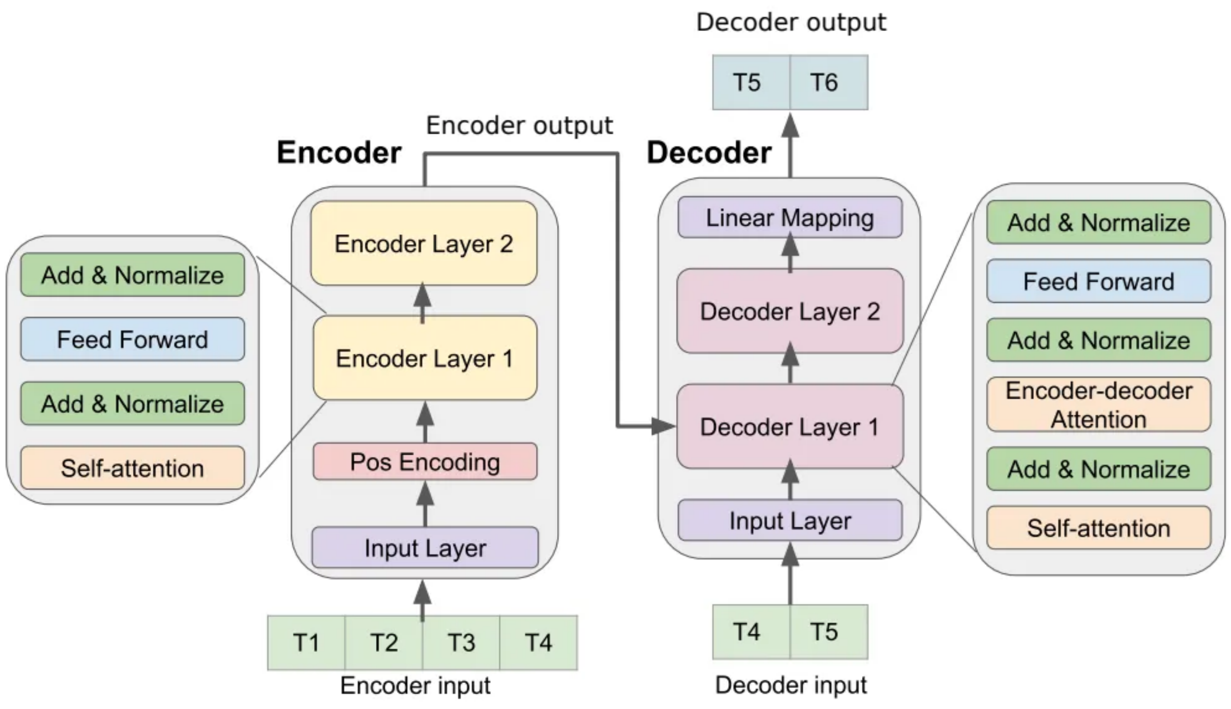


Figure 3.1 Common Transformer Architecture for Time Series prediction [47]

3.3.1 Input Embedding

Input tokens are converted into vectors of fixed dimensionality. This embedding layer is the initial step that maps each token to a high-dimensional space.

3.3.2 Positional Encoding

Transformers lack the capability to process sequential data as sequences. Positional encodings are added to the input embeddings to provide some information about the position of the tokens in the sequence. This helps the model to understand the order of tokens.

3.3.3 Encoder

The encoder is comprised of a stack of identical encoder layers, each containing two main sub-layers: a multi-head self-attention mechanism and a position-wise fully connected feed-forward network. The multi-head self-attention mechanism allows the model to focus on different parts of the input sequence when predicting a part of the sequence, effectively capturing the dependencies between tokens regardless of their positional distance from each other. The Feed-Forward Network is applied to each position separately and identically. This layer helps in further refining the representations after considering the entire context through attention mechanisms. Finally, each sub-layer in an encoder layer is usually followed by normalization and includes a residual connection. This helps in stabilizing the training process and allows for deeper models by mitigating the vanishing gradient problem.

3.3.4 Decoder

The decoder is similar in structure to the encoder, but with an additional sub-layer that performs multi-head attention over the encoder's output. This structure enables the decoder to focus on appropriate parts of the input sequence when producing the output. Most decoders also include masking in the self-attention layers to prevent positions from attending to subsequent positions. With relation to time series prediction, this masking, combined with the output from the encoder, allows the decoder to consider the entire input sequence and the previously generated outputs when generating the next token.

3.3.5 Linear Mapping

The final linear layer in a transformer model applied to time series data is responsible for transforming the high-dimensional representations learned by the transformer decoder into a specific output format required by the task. This is done by applying a learned linear transformation to the high-dimensional vectors. This transformation is typically a matrix multiplication followed by a bias addition (i.e., $y = Wx + b$), where W is the weight matrix, x is the input vector, and b is the bias vector. The dimensions of W and b are chosen based on the desired output shape for the task. The output is a sequence of vectors where each vector has been transformed to have the appropriate dimensionality for the task. In time series forecasting, each vector could represent the predicted value(s) for future time steps.

3.4 Stationary Wavelet Transform Integration

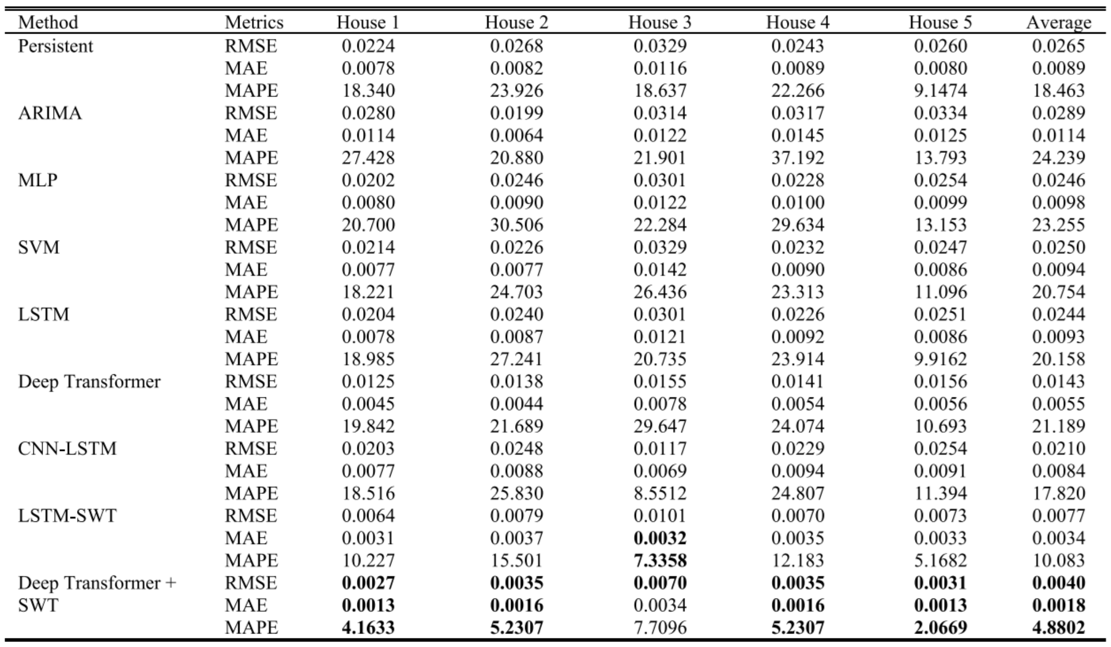
SWT provides a way to analyze time series data at multiple levels of detail or scales while preserving the time-invariance property. Given a time series sequence, there is a general guideline on how many decomposition levels can and should be made:

- Minimum Length Requirement
    - For a time series to be decomposed at level 1, it must be long enough to accommodate the chosen wavelet filter. Each subsequent level of decomposition requires the signal to be twice as long, due to the dyadic scaling (downsampling) in DWT. However, since SWT does not downsample, the length requirement is more about having enough data to meaningfully compute the approximation and detail coefficients at each level without the boundary effects that can affect DWT.
- Practical Rule of Thumb
    - A common rule of thumb is that the length of the time series should be at least 2^N, where N is the number of decomposition levels. This ensures that there are sufficient data for each level of decomposition. If the decomposed time series does not reach the desired level of decomposition due to the criteria of length 2^N, padding could be added to increase the length of the time series for higher-level decomposition.
- Analysis of Objectives and Level Selection
    - The choice of how many levels to use also depends on the analysis goals. For instance, if the aim is to capture very fine details in the time series, more levels might be necessary. Conversely, for broad trend analysis, fewer levels may suffice. The level of decomposition should be chosen based on the specific frequencies of interest and the nature of the signal.

3.5 Hybrid SWT Transformer

3.5.1 Outline

The project begins by first adopting the methodology and transformer architecture, proposed by [46]. In their study, they adopt the original transformer model [4], and utilize the Python library, PyWavelets, to convert the data to its corresponding wavelet values using SWT [48]. A small modification was made to the positional encoding layer where a Time2Vec block was used. The Time2Vec block is available on Pytorch and is proposed by [49], to provide time encodings for time series sequences. Their project [46] was studied as their results, reproduced in Figure 3.2, show a significant improvement in prediction performance when SWT is utilized with LSTM or Transformer. Therefore, this project aims to recreate their success on other datasets.



Method	Metrics	House 1	House 2	House 3	House 4	House 5	Average
Persistent	RMSE	0.0224	0.0268	0.0329	0.0243	0.0260	0.0265
	MAE	0.0078	0.0082	0.0116	0.0089	0.0080	0.0089
	MAPE	18.340	23.926	18.637	22.266	9.1474	18.463
ARIMA	RMSE	0.0280	0.0199	0.0314	0.0317	0.0334	0.0289
	MAE	0.0114	0.0064	0.0122	0.0145	0.0125	0.0114
	MAPE	27.428	20.880	21.901	37.192	13.793	24.239
MLP	RMSE	0.0202	0.0246	0.0301	0.0228	0.0254	0.0246
	MAE	0.0080	0.0090	0.0122	0.0100	0.0099	0.0098
	MAPE	20.700	30.506	22.284	29.634	13.153	23.255
SVM	RMSE	0.0214	0.0226	0.0329	0.0232	0.0247	0.0250
	MAE	0.0077	0.0077	0.0142	0.0090	0.0086	0.0094
	MAPE	18.221	24.703	26.436	23.313	11.096	20.754
LSTM	RMSE	0.0204	0.0240	0.0301	0.0226	0.0251	0.0244
	MAE	0.0078	0.0087	0.0121	0.0092	0.0086	0.0093
	MAPE	18.985	27.241	20.735	23.914	9.9162	20.158
Deep Transformer	RMSE	0.0125	0.0138	0.0155	0.0141	0.0156	0.0143
	MAE	0.0045	0.0044	0.0078	0.0054	0.0056	0.0055
	MAPE	19.842	21.689	29.647	24.074	10.693	21.189
CNN-LSTM	RMSE	0.0203	0.0248	0.0117	0.0229	0.0254	0.0210
	MAE	0.0077	0.0088	0.0069	0.0094	0.0091	0.0084
	MAPE	18.516	25.830	8.5512	24.807	11.394	17.820
LSTM-SWT	RMSE	0.0064	0.0079	0.0101	0.0070	0.0073	0.0077
	MAE	0.0031	0.0037	**0.0032**	0.0035	0.0033	0.0034
	MAPE	10.227	15.501	**7.3358**	12.183	5.1682	10.083
Deep Transformer +	RMSE	**0.0027**	**0.0035**	**0.0070**	**0.0035**	**0.0031**	**0.0040**
SWT	MAE	**0.0013**	**0.0016**	0.0034	**0.0016**	**0.0013**	**0.0018**
	MAPE	**4.1633**	**5.2307**	7.7096	**5.2307**	**2.0669**	**4.8802**

Figure 3.2 Results of the hybrid transformer model with other models for forecasting household energy consumption in five-minute intervals [46]

3.5.2 Hybrid Model Structure

Figure 3.3 shows an accurate representation of the hybrid model proposed by [46].

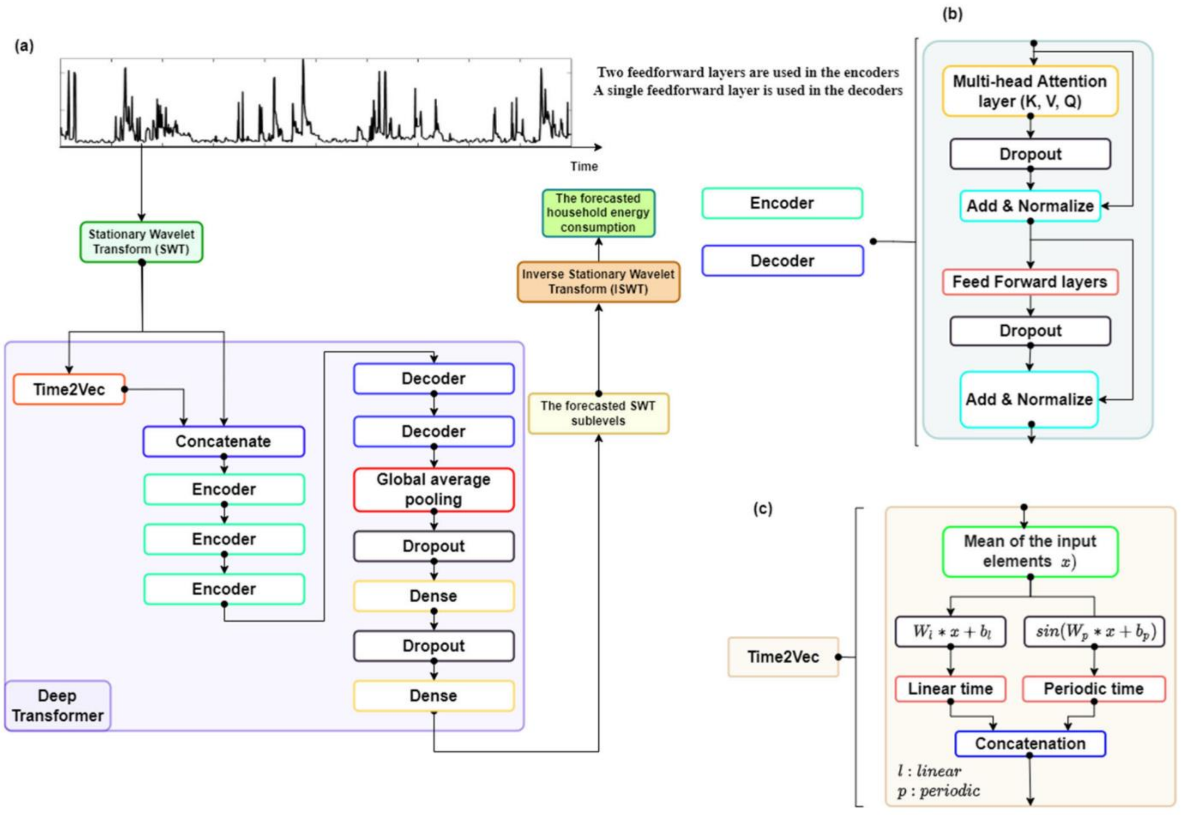

(a)
Time
Two feedforward layers are used in the encoders
A single feedforward layer is used in the decoders
Stationary Wavelet Transform (SWT)
The forecasted household energy consumption
Inverse Stationary Wavelet Transform (ISWT)
The forecasted SWT sublevels
Time2Vec
Concatenate
Encoder
Encoder
Encoder
Decoder
Decoder
Global average pooling
Dropout
Dense
Dropout
Dense
Deep Transformer
Encoder
Decoder
(b)
Multi-head Attention layer (K, V, Q)
Dropout
Add & Normalize
Feed Forward layers
Dropout
Add & Normalize
(c)
Time2Vec
Mean of the input elements x)
$W_l * x + b_l$
$sin(W_p * x + b_p)$
Linear time
Periodic time
Concatenation
l : linear
p : periodic

Figure 3.3 The proposed deep transformer SWT model for the household power consumption forecasting, (b) the transformer encoder/decoder, (c) the Time2Vec block [46]

3.5.3 Implementation

The following section notes how the experiments were done on a step-by-step basis. An example will be provided after all the steps are explained.

3.5.3.1 Model Reimplementation

The PyTorch framework is used for these experiments. The hybrid SWT Transformer model was originally implemented in TensorFlow. To compare against past research done on the same electricity consumption datasets, the model has been reimplemented in PyTorch. We can verify the validity of the reimplemented model by yielding the same results recorded in [34].

3.5.3.2 Preparing the Data

The data preparation process starts by dividing the continuous time sequences present in each dataset into two distinct portions: a historical segment dedicated to training and a future-oriented segment for testing purposes. This division is based on an 80:20 split, allocating the larger share for training and the remainder for testing.

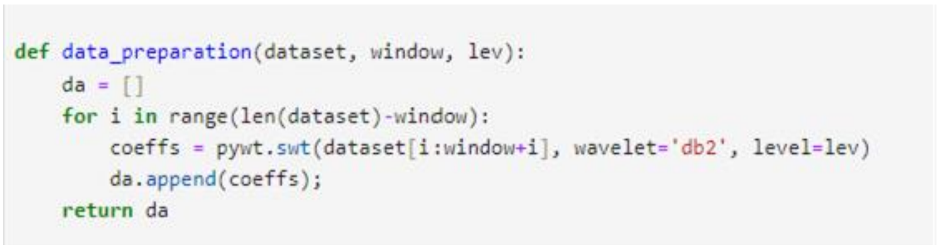
Next, we employ a lookback period of either 5 or 9 weeks in duration, to further divide these segments into more manageable time series sequences. The sliding window technique [50] uses a parameter “stride” to determine a fixed length of time series values to skip when constructing either a training or testing set. By dividing the data into these smaller sequence chunks, it facilitates the training of complex models, like hybrid SWT Transformers, on sequence batches. This approach not only streamlines the training process but also mitigates the risk of model overfitting.

Then, we normalize the training and testing segments to uniform value scales across the datasets. This normalization is achieved through the formula $\frac{\mathrm{yi-min}}{\mathrm{max-min}}$, where 'min' and 'max' represent the minimum and maximum values within the training segments, respectively. The predictive analysis is conducted on this normalized data.

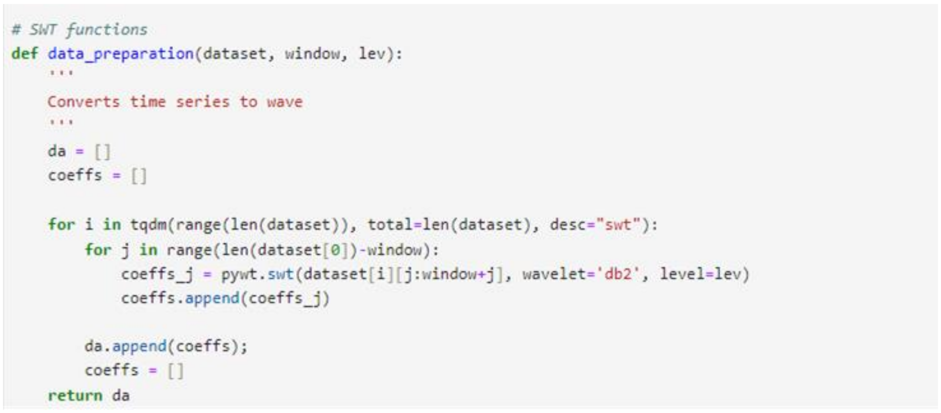
Finally, for each sequence, we choose the number of time steps we want to predict, and these are the final “x” time steps in the sequences. The sequences leading up to these predicted steps serve as the lookback period, setting the stage for the ensuing predictive modeling.

3.5.3.3 Stationary Wavelet Transform

In the SWT step, we convert each smaller chunk of time series sequences into their corresponding SWT values. This is done with the aid of the function “pywt.swt” from the PyWavelets library, which is available via imports in Python. The original paper uses a sliding window method with a stride of 1 on the full time series. However, since our data contains multiple smaller chunks of time series sequences, a small modification of the SWT conversion was made. This modification will generate fewer samples than the original process where the full time series was used. However, it will not have a large impact on the performance.



```python
def data_preparation(dataset, window, lev):
    da = []
    for i in range(len(dataset)-window):
        coeffs = pywt.swt(dataset[i:window+i], wavelet='db2', level=lev)
        da.append(coeffs);
    return da
```



```python
# SWT functions
def data_preparation(dataset, window, lev):
    '''
    Converts time series to wave
    '''
    da = []
    coeffs = []

    for i in tqdm(range(len(dataset)), total=len(dataset), desc="swt"):
        for j in range(len(dataset[0])-window):
            coeffs_j = pywt.swt(dataset[i][j:window+j], wavelet='db2', level=lev)
            coeffs.append(coeffs_j)

        da.append(coeffs);
        coeffs = []
    return da
```

Figure 3.4 SWT conversion functions, Original (Top), Modified version (Bottom)

Referencing Figure 3.4, we can observe that the SWT conversion process requires 3 arguments, Window, Level and Wavelet.

Window is the size of the sequence length we wish to convert to its SWT values. In our experiments, the window size is determined by the lookback period. This enables us to predict 1 time step per wavelet sequence based on a historical segment of lookback size.

Level, as explained under SWT integration in chapter 3.4, is the number of times SWT decomposition is recursively done on the input. The maximum level can be determined by the number of times an input can be reduced by a factor of 2 without remainders. Given a time series length of 816, the maximal decomposition level is 4. In terms of dimensions, an 816 size 1 dimensional time series will be transformed to (3,2,816) with a level 3 decomposition. In our experiments, we will use either the maximal decomposition possible or at most a level of 3.

Wavelet [51] refers to the type of wavelet transform function used to transform the data. There are two types of wavelet analysis: continuous and multiresolution. The choice of wavelets used depends on the goal. Examples of these goals include Energy Preservation, Feature Detection, Analysis of Variance, etc. In our experiments, we will stick with the wavelet function, “db2”, as it is stated in the original paper that this wavelet function produces the best performance.

3.5.3.4 Hyperparameters and loss metrics

This section details the variables used during the training process:

- Sequence Length
    - We do not require every SWT value after conversion. After applying SWT to the data, two sets of coefficients, approximation coefficients (cA) and detail coefficients (cD), are produced. These vectors are obtained by convolving the data with a low-pass filter for approximation, and with a high-pass filter for detail. The approximation coefficients (cA) typically capture the broader, more general features of a signal while the detail coefficients (cD) capture smaller details and noise. Through our experiments, we notice that an increase in SWT values used at the start does increase the performance, but it caps at a certain value. Hence, we could use a partial number of SWT values near the prediction value to reduce the training time.
- Model Parameters
    - The SWT transformer uses an embedding dimension size of 256, with 12 heads, and consists of 3 encoder layers and 2 decoder layers.
    - RELU activation is used for all activation functions present in the encoders and decoders.
    - Layer Normalization is used after multi-head self-attention layers in encoders and decoders.
    - RMSprop is used as the optimizer with a learning rate of 0.001 and a stability parameter, eps, of 1e-07.
    - A GradScaler is used to help perform the steps of gradient scaling conveniently. This improves convergence for networks with float16 gradients by minimizing gradient underflow.
    - Loss function, Mean Squared Error (MSE) loss, is used.
    - A batch size of 32 is chosen.
- Evaluation Metrics
    - Mean Squared Error (MSE) loss
        - MSE measures the average squared difference between the predicted values and the actual values. This metric heavily penalizes larger errors due to its quadratic nature, making it sensitive to outliers.
    - Root Mean Squared Error (RMSE) loss
        - RMSE is the square root of MSE, offering a metric in the same units as the target variable. It shares MSE's sensitivity to large errors but is more interpretable due to its unit compatibility with the target variable.
    - Mean absolute error (MAE) loss

        - MAE measures the average magnitude of the errors in a set of predictions, without considering their direction. It’s less sensitive to outliers than MSE or RMSE.
    - Symmetric mean absolute percentage error (sMAPE)
        - sMAPE is a variation of MAPE that addresses its asymmetry by considering both the prediction and actual values in the denominator. It expresses errors as a percentage, making it easy to interpret and compare across different datasets or models.

3.5.3.5 Transformers

The transformer, which is adapted from the original paper [46], is designed to train on wavelet data of a single time series. The input follows the following shape (number of training instances, wavelet values, levels). However, given that our data has multiple subsequences of the single time series, the data from each subsequence is individually fed into the model during both the training and testing phases. Although this approach slightly diminishes training efficiency, the duration for training still hovers between 36 seconds per epoch.

Within this transformer architecture, a significant component is the Time2Vec block, which utilizes the wavelet values to produce time encoding values. These encodings are subsequently concatenated with the input, enhancing the transformer's capability to discern the temporal sequence of the data.

The wavelet values, along with the time encoding are then passed through the Encoder and Decoder block, which uses the attention mechanism to learn different parts of a sequence that are important for prediction.

Finally, the final decoder block’s output is subjected to a pooling function and a linear layer. This sequence of operations culminates in the generation of a SWT prediction value for a single forecasted timestamp.

3.5.3.6 Inverse Stationary Wavelet Transform

The singular SWT prediction timestamp, according to the paper, replaces the original wavelet sequence at the final position. Using Inverse Stationary Wavelet Transform (iswt) on the modified wavelet sequence enables us to revert the modified wavelet sequence to its actual time series form (length of lookback period +1). This step is iteratively executed for each day forecasted. Aggregating the forecasts for the number of forecast days results in the ultimate prediction. This prediction is compared with the actual values using the evaluation metrics provided in 3.5.3.3 to evaluate the performance of this hybrid model.

3.5.3.7 Implementation Example

For clarity, we will use PJM system load data - AEP, a lookback of 5 weeks and a forecast of 24 time steps as an example. AEP consists of one hourly series from 2 Oct 2004 00:00 to 2 Aug 2018 23:00 with 121,272 time steps in megawatts (MW).

This will first be split into train and test in an 80:20 split, leaving the first 98,720 timestamps for training and the remaining 24,192 timestamps for testing. Next, the data is subdivided into smaller chunks of length 840 (5 weeks) subsequences using a sliding window technique of stride 29. A subsequence will not be made if there is not enough data to satisfy the length. This creates 1753 training sequences and 279 testing sequences.

Next, we apply SWT Transformation to each subsequence. Note from this point on, we are only talking about 1 subsequence. The window value is set at 816 (840 - 24). For each subsequence of length 840 and window of 816, it will be transformed to (24,3,2,816). The wavelet values will be coalesced into (24,6,816) and have the dimensions swap to (24,816,6) for preparation as the input to the model.

At this stage, we sample 50 SWT values to get a shape of (24,50,6). This will be the input to the Transformer model. The output of the Transformer will be a shape of (24,1,6). This output is the predicted SWT value of a single time step over a forecast history of 816. To retrieve the associated actual value of this time step, the predicted SWT value replaces the actual SWT value of the initial wavelet subsequence. This enables us to recreate a new subsequence using inverted SWT (iswt) and obtain the predicted value, which is located at the end of the new subsequence.

4 Results and Discussions

4.1 Environments

4.1.1 Hardware

Training and Evaluation of the hybrid SWT Transformer model was tested on either 1 NVIDIA GeForce RTX 2060 or Google Colab’s T4 GPU.

4.1.2 Software

The PyTorch framework is used to create the Transformer Model and input tensors. The Python PyWavelets library is used to generate SWT wavelet values from the time series subsequences. The Python NumPy library is used as a tool for large, multi-dimensional arrays as the library provides a large collection of high-level mathematical functions capable of manipulating these arrays. The Python Matplotlib is used to create quality plots for better visualization of our results.

4.2 Comparison to baseline methods

To evaluate the predictive performance of a hybrid SWT Transformer, we require several baseline methods.

For statistical time series forecasting methods, we decided to employ the help of DARTS [52]. DARTS is a Python library tailored to user-friendly forecasting and anomaly detection on time series data. It contains a variety of models from traditional methods to deep learning models. In our project, we will use DARTS to easily generate evaluations with statistical time series forecasting methods and present them according to the evaluation metrics. The statistical methods that will be compared are ARIMA, AutoARIMA, THETA and BATS.

For deep learning models, we will also evaluate some models with DARTS. The models are namely a simple recurrent neural network (RNN), a Fast Fourier Transform model (FFT) and a basic Transformer model. The hyperparameters for RNN and FFT are not optimized while the hyperparameters of the Transformer are optimized to our best abilities.

In addition, model results from research papers that precede our research will be compared. While these results are obtained from the same subset of datasets, the lookback period and forecast time steps do not match our research. Hence, we will adopt their models and change the parameters to suit our evaluation. These models include a Sparse Transformer [39] and a time-series Recurrent Neural Network (tsRNN) [53] that uses 3d convolution layers.

4.3 Results

The hybrid SWT Transformer shows promising results over the 4 different datasets. We will focus on the London dataset in this section due to its volatility. Results from the other datasets will be shown in the appendix section at the end of this report.

Due to a limited amount of time, most models will be trained on only a portion of the full dataset. This is especially true for the London dataset due to its large data size. For the London dataset:

- The hybrid SWT Transformer uses 1000 sequences split in a 60:20:20 ratio for training, validation and testing.
- The results for statistical models are produced using the mean of the first 100 test sequences.
- The results of tsRNN and Sparse Transformer use the first 200 sequences as training, the next 100 as validation and another 100 for testing.

For a more accurate representation of these models, it is recommended to check the original papers as the models are trained on a larger number of training samples and a larger epoch.

4.3.1 Result tables

Tables 4.1 and 4.2 show the results on the London dataset with lookback of 5 weeks and 9 weeks respectively.

Table 4.1 London dataset with lookback of 5 weeks

Models	Forecast time steps = 48				Forecast time = 168			
	MSE	RMSE	MAE	sMAPE	MSE	RMSE	MAE	sMAPE
ARIMA	0.027406	0.145756	0.110375	0.805650	0.025962	0.146997	0.113678	0.844652
AutoARIMA	0.017468	0.122550	0.089520	0.763886	0.018397	0.128873	0.094334	0.814561
THETA	0.029529	0.147317	0.112306	0.859390	0.025115	0.145088	0.106969	0.883139
BATS	0.016753	0.120709	0.087308	0.753121	0.019401	0.129889	0.094304	0.803567
RNN	0.027866	0.153299	0.121095	0.869817	0.033571	0.167148	0.133983	0.907172
FFT	0.013988	0.111862	0.080356	0.783498	0.024402	0.148495	0.113632	0.946309
Darts Transformer	0.015829	0.115379	0.077640	0.690609	0.017939	0.124494	0.085948	0.879493
tsRNN 3d ***	---	0.94	0.063	0.711	---	0.102	0.067	0.718
Sparse Transformer	0.012888	0.102223	0.067578	0.685793	0.012027	0.102691	0.064653	0.685936
Hybrid SWT Transformer	**0.000131**	**0.011432**	**0.010431**	**0.218091**	**0.000070**	**0.008381**	**0.006045**	**0.141917**

*** Taken from research paper [53]

Table 4.2 London dataset with lookback of 9 weeks

Models	Forecast time steps = 48				Forecast time = 168			
	MSE	RMSE	MAE	sMAPE	MSE	RMSE	MAE	sMAPE
ARIMA	0.016451	0.113670	0.077271	0.740520	0.024189	0.141457	0.116954	0.919720
AutoARIMA	0.016088	0.112561	0.079841	0.787798	0.019903	0.124250	0.097283	0.880768
THETA	0.018975	0.122047	0.087211	0.900004	0.030504	0.153729	0.127334	1.030676
BATS	0.015016	0.109753	0.080150	0.793523	0.015501	0.114639	0.086497	0.834043
RNN	0.026260	0.146610	0.113940	0.911573	0.036423	0.164656	0.133206	0.995019
FFT	0.013571	0.105230	0.077592	0.835982	0.018482	0.125021	0.095375	0.923739
Darts Transformer	0.015829	0.115379	0.077640	0.690609	0.017939	0.124494	0.085948	0.879493
tsRNN 3d ***	---	---	---	---	---	---	---	---
Sparse Transformer	0.012797	0.100596	0.066065	0.712071	0.014150	0.102192	0.065353	0.672093
Hybrid SWT Transformer	**0.000131**	**0.011432**	**0.010431**	**0.218091**	**0.000070**	**0.008381**	**0.006045**	**0.141917**
*** Taken from research paper [53]								

4.3.2 Graph Results

The plots for 1 subsequence of the London dataset are shown in Figures 4.1 to 4.3. The parameters are a lookback of 5 weeks and a forecast for 48 time steps. Note that due to how the subsequences are constructed and how many samples were taken, the ground truth of the subsequence is not the same over different models.

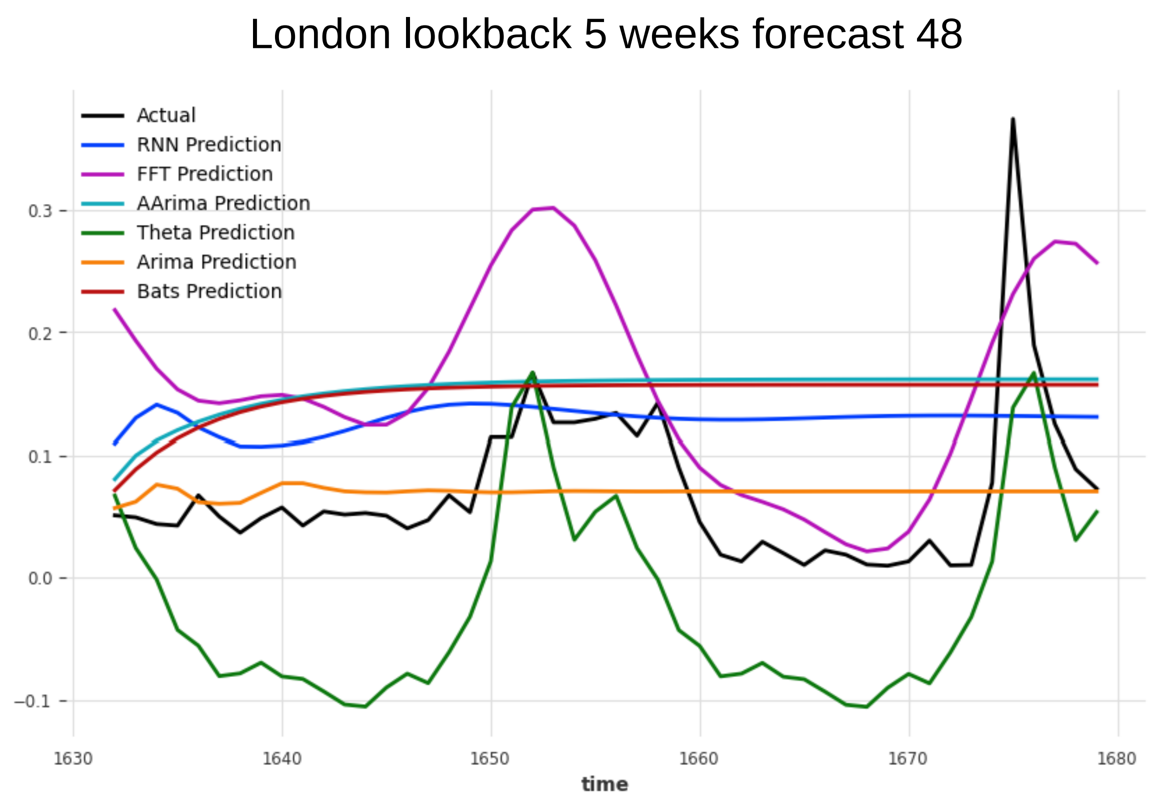


Figure 4.1 Statistical model prediction for London dataset, with 5 weeks lookback and 1 day forecast parameter

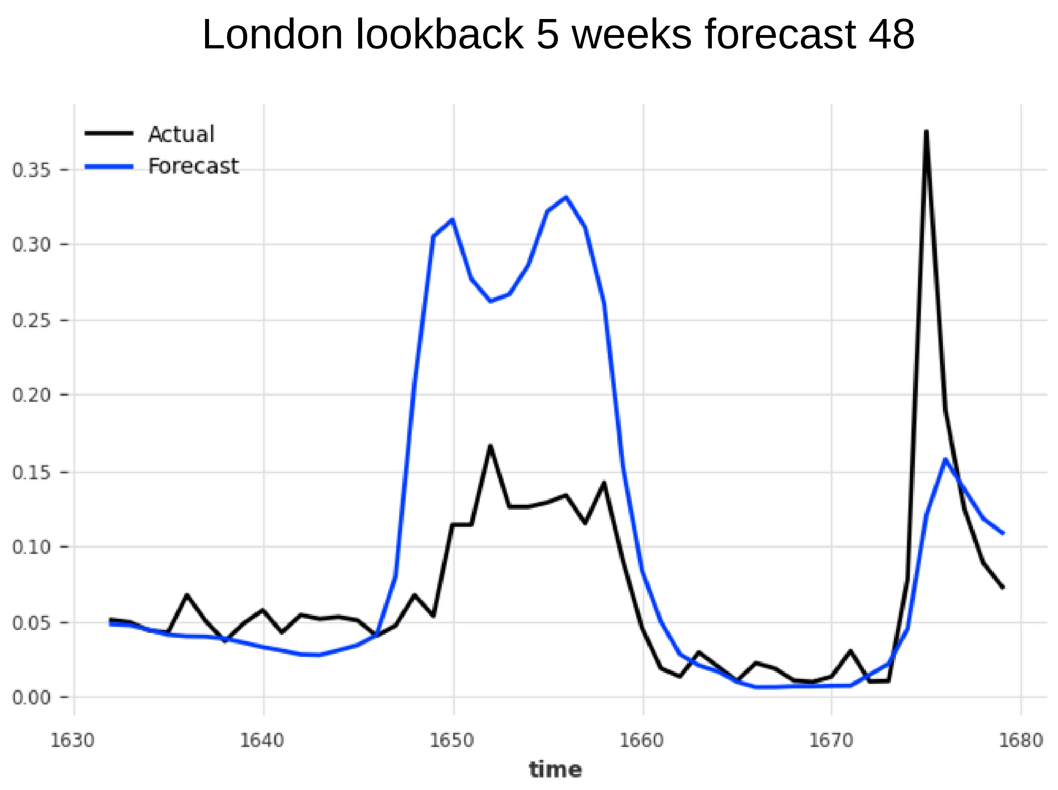


Figure 4.2 Darts Transformer prediction for London dataset, with 5 weeks lookback and 1 day forecast parameter

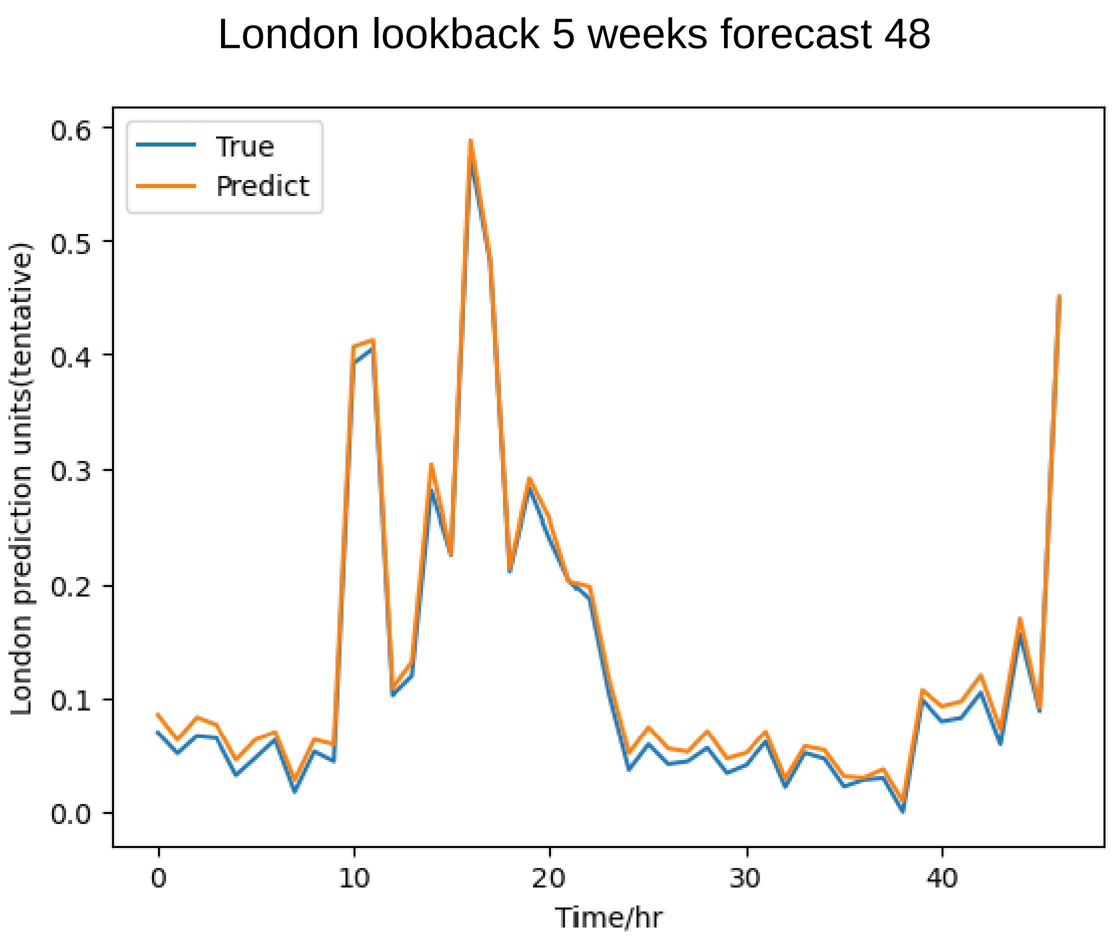


Figure 4.3 Hybrid SWT Transformer prediction for London dataset, with 5 weeks lookback and 1 day forecast parameter

4.3.3 Interpretation of Results

From the results table above and those in the appendix, we can easily see that the hybrid SWT Transformer proposed by [46] dominates in terms of performance as compared to statistical models and deep learning models that do not incorporate SWT. The performance increase after SWT is used is in line with [46], where the increase is minimally 10 times as compared to other models in terms of RMSE and MAE.

4.4 Discussions

4.4.1 Observations (Dubious results)

Looking at the results, it is evident that the hybrid SWT transformer is suspiciously accurate in its prediction. From the graph, we can see that the prediction of the model closely matches the shape of the actual result. If such results are possible, the model would be groundbreaking on many fronts as the model will be able to simulate future events without other information (univariate inputs).

Given that the transformer model in question does not represent the pinnacle of current technology, the likelihood of a basic transformer model achieving highly accurate predictions from a complex array of inputs seems impossible. Typically, a more sophisticated model must be used to cater to more complex inputs for performance increase. In this case, the original time series data, which is of 1 dimension, is expanded to 6 dimensions for a level 3 decomposition during SWT transformation. It is unheard of and incomprehensible that a basic transformer model improves this significantly after the inputs are transformed into a more complex representation.

A research paper written by individuals at The Chinese University of Hong Kong [54], [55] shares the same concern. In their paper, they too question the authenticity of the results of Transformers in time series modeling. Their investigation into the use of Transformers for forecasting purposes highlights a critical flaw: the self-attention mechanism inherent to Transformers tends to compromise temporal data, leading to partial information loss. This results in simple Long-Term Time Series Forecasting (LTSF) Linear model outperforming existing sophisticated Transformer-based LTSF models in all cases, and often by a significant margin.

4.4.2 Hypothesis

With this in mind, we hypothesize that there is a leak of the ground truth values from the prediction zone. Most often, this leakage stems from covariates. Covariates, sometimes known as external data, are independent variables which are complementary or related to the forecasting of the target time series. Unlike the primary time series data, covariates are not subject to prediction during the forecasting process and future covariates are time series variables whose future values are known at prediction time. Examples of such covariates could include weather conditions, temperature readings, etc. In cases where there is a direct correlation between covariates and the target time series, the presence of future covariates can significantly bolster the accuracy of predictions. For example, forecasting the demand for air conditioning in a region based on future weather and temperature data could yield highly accurate results due to the direct influence of these factors on air conditioning usage. However, given that our project only uses univariate data we can rule out covariates being the reason for the improved performance.

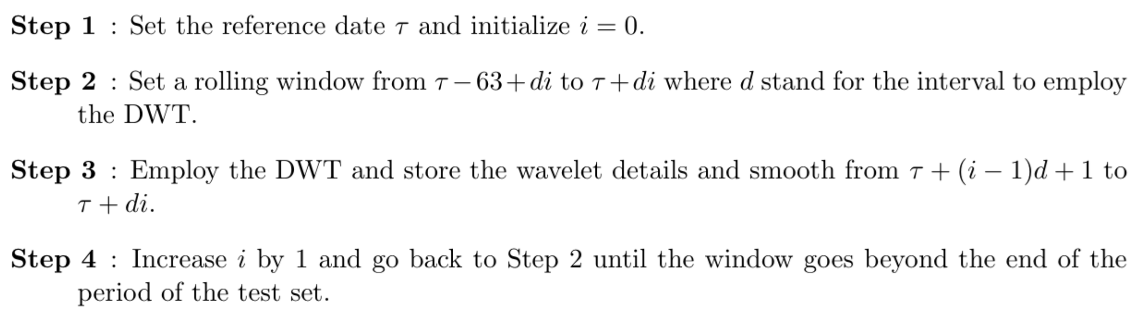
Since no covariates were used as inputs, it is likely that one or more of the steps taken during SWT transformation is the primary cause for the leak. We propose two potential scenarios that could be facilitating this leakage:

1. SWT transformation in each window could leak the actual value to other values in the wave since the waves are created based on the input. This suggests that the target future value is somehow encapsulated within the input values used for transformation, suggesting a direct data leak.
    a. This is discussed in [56] under “Why RHWT? (Redundant Haar wavelet transform)” which states that symmetric wavelets such as Daubechies, Morlet, or Symlet take into account not only previous information but also future information. This is not desirable for a forecasting problem since we can only use data obtained earlier in time.

2. During SWT transformation using a rolling window, we observe that a greater number of future target values are included in SWT transformation as the window moves over the predicted zone. This practice is inherently flawed for forecasting purposes as it violates the principle that future information should remain unknown during model training and prediction phases.
    a. Hypothesis 2 is quickly found to be supported by [57]. This is known as the Data Leakage. Data leakage in time series forecasting refers to a situation where information from outside the training dataset's time frame is inadvertently used to make predictions. This can lead to an overly optimistic appraisal of model performance during training and validation stages, a fantasy that dissipates when the model confronts genuinely new data.

4.4.3 Novel Approach

The solution proposed in [57] follows the methodology done by [58]. However, the main issue that they are tackling is the boundary problem of wavelet decomposition. Figure 4.4 shows the steps taken to mainly tackle the boundary problem.



Step 1 : Set the reference date τ and initialize $i = 0$.

Step 2 : Set a rolling window from $\tau - 63 + di$ to $\tau + di$ where d stand for the interval to employ the DWT.

Step 3 : Employ the DWT and store the wavelet details and smooth from $\tau + (i-1)d + 1$ to $\tau + di$.

Step 4 : Increase i by 1 and go back to Step 2 until the window goes beyond the end of the period of the test set.

Figure 4.4 Steps taken to avoid boundary problem and data leakage [57]

As seen in Figure 4.4, the methodology closely resembles the SWT transformation step done in hybrid SWT Transformer. Additionally, we are unable to find other sources that mention this problem, apart from only a couple of comments [59]. However, they do not suggest a solution apart from suggesting the addition of redundant information through padding as a potential workaround. Therefore, we decide to propose several approaches to tackle data leakage.

4.4.3.1 Hypothesis 1

Before that, we explore the possibility of hypothesis 1. In this experiment, we convert the subsequences to their wavelet values without using a rolling window technique. Next, we will use the historical segment to predict the actual segment directly. For example, for an 840 time steps subsequence, we use the first 816 wavelet time steps to predict the next 24 time steps directly. We compare the performance of this experiment to the Darts transformer to prove our hypothesis.

Table 4.3 London dataset, Hypothesis 1

Models	Forecast time steps = 48, lookback = 5 weeks			
	MSE	RMSE	MAE	sMAPE
Darts Transformer	**0.015829**	**0.115379**	**0.077640**	**0.690609**
Transformer (predict 48)	0.0293968	0.171455	0.106719	0.765961

Table 4.4 DAYTON dataset, Hypothesis 1

Models	Forecast time steps = 24, lookback = 5 weeks			
	MSE	RMSE	MAE	sMAPE
Darts Transformer	**0.003349**	**0.049119**	**0.040837**	**0.118589**
Transformer (predict 24)	0.025787	0.160583	0.133045	0.350473

Based on the results of the London and Dayton datasets shown in Tables 4.3 and 4.4 resepectively, we can observe that due to the large size of the lookback, it is likely that even if there is leakage due to wavelet conversion, it is negligible and does not improve the performance of the model.

4.4.3.2 Hypothesis 2

In this section, we propose a couple of solutions to tackle data leakage. These solutions avoid using the target values as inputs for wave conversion:

1. We introduce the prediction window method. This method is similar to the rolling window technique, but the predicted time step replaces the target value of the same step. This ensures that as the rolling window moves to the next predicted step, target values are not involved in SWT conversion.
2. In hypothesis 1, we observe that the basic transformer was unable to predict as well as the DARTS Transformer. In that case, we will replace the basic transformer with an existing more complex transformer, Sparse Transformer. In this case, the actual data, not the wavelet values are used as inputs.
3. Following the results in (2), we find that a more complex model architecture does improve performance. To explore if SWT does improve performance, we will experiment with wavelet values as inputs and the actual data as outputs for a more complex transformer architecture, Sparse Transformer.
4. Following the results in (4), we realize that we cannot directly use wavelet values to predict the actual values. We decided to combine wavelet values and the actual data to form a multivariate input to our Sparse Transformer. This proves to be effective as the performance improved further from (2).
5. Drawing inspiration from [60], [61], we will use multiple DARTS transformers to train on each wavelet value separately and predict them accordingly. In this experiment, Maximal overlap discrete wavelet transform (MODWT) was used as the preferred wavelet transformation function as it does not require the signal to be of any specific length (not power of two). We will perform a logarithmic number of decompositions to the number of training samples. This usually results in a higher number of decomposition levels and would result in a finer resolution or more detailed representation of the time series sequence.

For easier reference, (1) will be known as **Transformer (pred window)**, (2) as **Sparse Transformer (Predict 24)**, (3) as **Sparse Transformer (Wave to Target)**, (4) as **Sparse Transformer (WaveActual to Target)** and (5) as **multi MODWT Transformer**.

4.4.3.3 Results

The results from these experiments are done largely on the DAYTON dataset (see Table 4.5), with the London dataset (see Table 4.6) providing some insights on more complex time series fluctuations.

Table 4.5 DAYTON dataset, Data Leakage

Models	Forecast time steps = 24, lookback = 5 weeks			
	MSE	RMSE	MAE	sMAPE
DARTS Transformer	0.003349	0.049119	**0.040837**	0.118589
Transformer (predict 24)	0.025787	0.160583	0.133045	0.350473
Sparse Transformer	**0.002833**	**0.047215**	0.0409342	**0.115113**
Transformer (pred window)	0.018223	0.134993	0.107688	0.297723
Sparse Transformer (Predict 24)	0.012121	0.110098	0.089894	0.256263
Sparse Transformer (Wave to Target)	0.018295	0.13526	0.107953	0.297768
Sparse Transformer (WaveActual to Target)	0.0031148	0.055811	0.043456	0.12338
multi MODWT Transformer	0.0063847	0.0700382	0.063625	0.190151

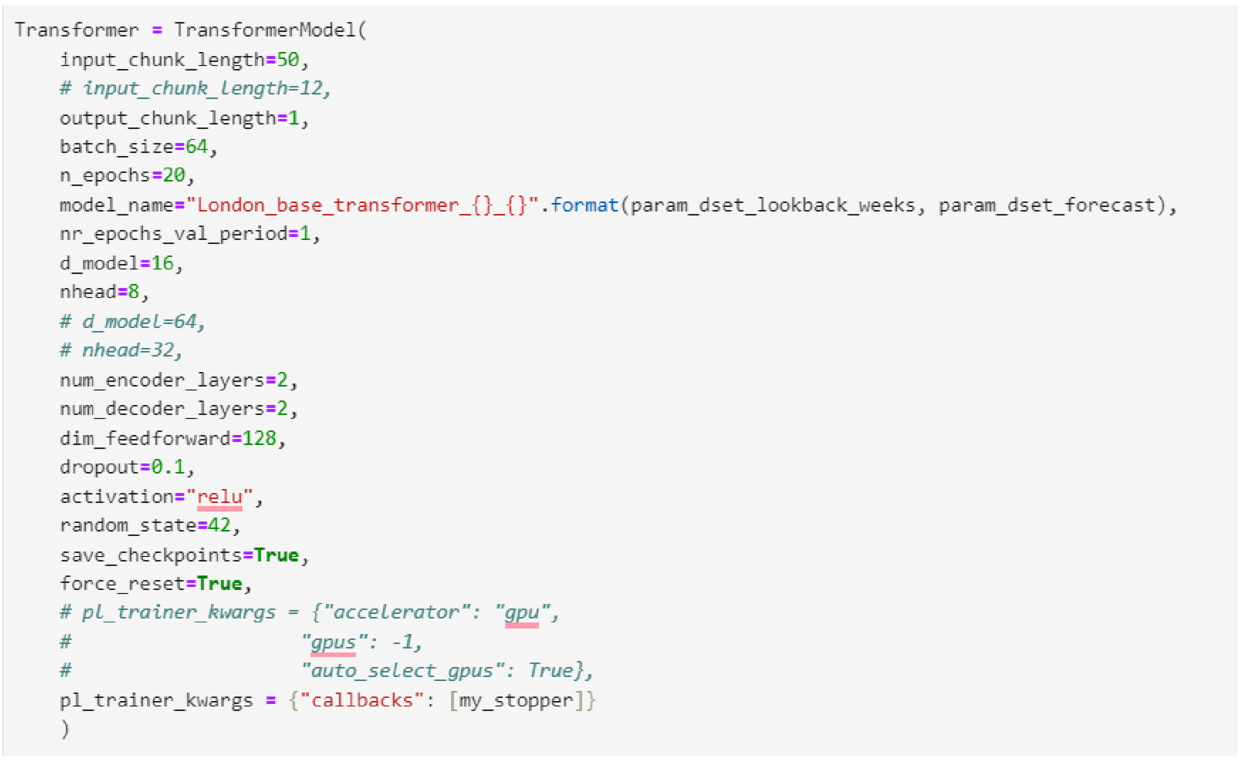
Table 4.6 London dataset, Data Leakage

Models	Forecast time steps = 48, lookback = 5 weeks			
	MSE	RMSE	MAE	sMAPE
DARTS Transformer	0.015829	0.115379	0.077640	0.690609
Transformer (predict 48)	0.0293968	0.171455	0.106719	0.765961
Sparse Transformer	**0.0128883**	**0.102223**	**0.067578**	**0.685793**
Transformer (pred window)	0.018938	0.137619	0.091194	0.824628
multi MODWT Transformer	0.021582	0.135252	0.104103	0.847849

4.4.3.4 Results Interpretation and Accuracy

From the results shown in Tables 4.5 and 4.6, it is evident that the original Sparse Transformer outshines all other models in terms of performance. Additionally, we can observe that a more complex model such as Sparse Transformer can improve performance when dealing with wavelet values. Finally, the question of whether techniques such as Wavelet Decomposition truly augment a model's predictive capabilities remains open to debate.

The DARTS transformer uses the rolling window technique too. The important parameters (see Figure 4.5) to pass into the DARTS transformer are input_chunk_length and output_chunk_length. The input_chunk_length parameter dictates the number of historical time steps considered for each model input chunk, whereas output_chunk_length specifies the number of time steps the model predicts in a single iteration. During our experiments, the model's output_chunk_length was configured to 1, a setting that may inadvertently facilitate data leakage. However, according to the guidelines provided in the official documentation [62], setting a forecast horizon that exceeds the output_chunk_length in the predict() function can avoid auto-regression. Auto-regression, akin to the rolling window method, when prevented, can ensure the elimination of data leakage by making predictions in a collective manner, rather than sequentially. This approach safeguards against the inadvertent incorporation of future information in the model's learning process, thereby maintaining the integrity of the predictive outcomes.



```python
Transformer = TransformerModel(
    input_chunk_length=50,
    # input_chunk_length=12,
    output_chunk_length=1,
    batch_size=64,
    n_epochs=20,
    model_name="London_base_transformer_{}_{}".format(param_dset_lookback_weeks, param_dset_forecast),
    nr_epochs_val_period=1,
    d_model=16,
    nhead=8,
    # d_model=64,
    # nhead=32,
    num_encoder_layers=2,
    num_decoder_layers=2,
    dim_feedforward=128,
    dropout=0.1,
    activation="relu",
    random_state=42,
    save_checkpoints=True,
    force_reset=True,
    # pl_trainer_kwargs = {"accelerator": "gpu",
    #                      "gpus": -1,
    #                      "auto_select_gpus": True},
    pl_trainer_kwargs = {"callbacks": [my_stopper]}
    )
```

Figure 4.5 DARTS Transformer model parameters

The choice of input_chunk_length plays a pivotal role in shaping the performance of our model, serving as a critical parameter that balances computational efficiency with predictive accuracy. In our experiment, we set input_chunk_length at 50 as it is a good balance between performance and computational efficiency. Subsequent experimentation revealed that increasing the input_chunk_length could further enhance model performance, as it provided a broader historical context for the model to draw from when making predictions about a single future time step. Interestingly, it should be noted that when the input_chunk_length encompasses an entire lookback period, the performance metrics should align closely with those observed in the hybrid SWT transformer model.

On another front, the multi MODWT Transformer model introduces a layer of complexity, notably when multiple transformers are employed within the model architecture. This complexity inherently translates to longer training and prediction times. Due to the constraints of the time available for this report, the multi MODWT Transformer was configured with an input_chunk_length of just 12, in contrast to the 50 used in the DARTS Transformer, to accommodate the increased computational burden. Although preliminary tests were conducted to explore the potential for performance enhancement with this model configuration, the limited scope of these tests precludes a definitive conclusion (See Table 4.7).

Table 4.7 Dayton dataset, MODWT

Models	input_chunk_length	Lookback weeks	Forecast	MSE	RMSE	MAE	sMAPE
DARTS Transformer	50	9	24	**0.007790**	**0.075381**	**0.062403**	**0.168051**
multi MODWT Transformer	12	9	24	0.009547	0.086603	0.078146	0.2261461
DARTS Transformer	50	9	84	0.026694	0.150879	0.123442	0.2938312
multi MODWT Transformer	12	9	84	**0.014283**	**0.114417**	**0.094151**	**0.2567816**

4.4.4 Possible future works

The function used for prediction in Multi MODWT Transformer is “predict”. Two particularly intriguing functions, "Historical Forecast" and "Backtrack," merit further exploration for their potential applicability and benefits in time series forecasting:

- Historical Forecast
    - This function capitalizes on the rich insights inherent in historical data to inform future predictions. In DARTS, this approach enables the utilization of accumulated historical data to train forecasting models, which are then leveraged to predict future data points. This approach relies on the assumption that historical patterns or trends will continue into the future. Given its emphasis on historical patterns as predictors of future events, "Historical Forecast" could offer a more nuanced and potentially more accurate forecasting approach than conventional "predict" functions, which prioritize current data and known variables over historical trends. Refer to Table 4.8 and Figure 4.6 for the historical forecast of the London dataset. [xxx]
- Backtrack
    - This functionality involves using historical data to validate the forecasts made by a model. Essentially, the accuracy of your forecasts is tested by comparing them against the actual historical data that was not used during training. This

helps to assess the model's generalization capabilities and its effectiveness in capturing and interpreting the underlying dynamics of the data series.

Based on the description of “Historical Forecast”, it appears to be particularly well-suited for scenarios where the primary objective is to harness historical data patterns to forecast future outcomes. Future works could, therefore, benefit from an in-depth exploration of these alternative functions, assessing their impact on model performance and their contribution to advancing the state-of-the-art in time series forecasting.

Table 4.8 London dataset, Historical forecast

Models	Forecast time steps = 48, lookback = 5 weeks			
	MSE	RMSE	MAE	sMAPE
DARTS Transformer	0.015829	0.115379	0.077640	0.690609
Sparse Transformer	0.0128883	0.102223	0.067578	0.685793
DARTS Transformer "Historical Forecast"	**0.010553**	**0.094407**	**0.056944**	**0.53405430**

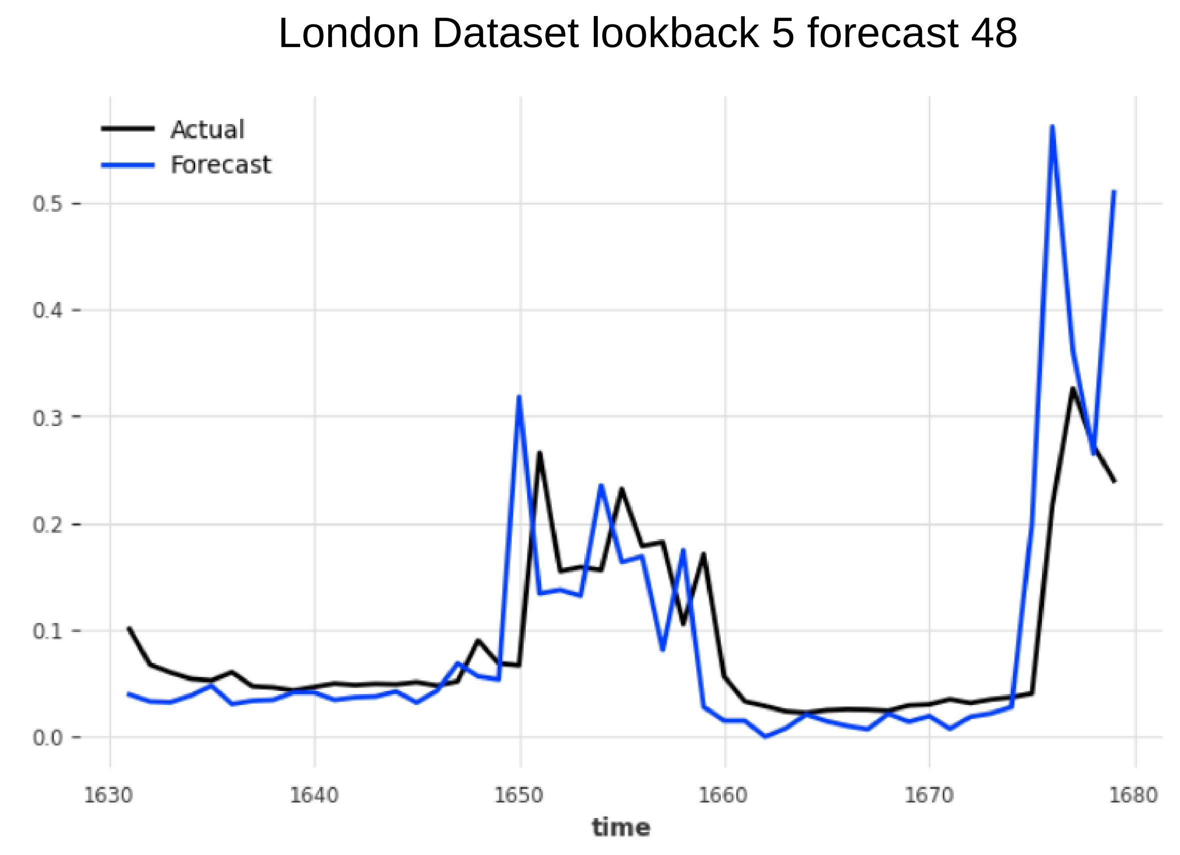


Figure 4.6 Historical forecast using DARTS Transformer

5 Conclusion

This project has explored the integration of Wavelet Decomposition techniques with Transformer models in time series forecasting. Throughout the project, a critical issue of data leakage was unearthed, inadvertently leading to an inflated perception of the model's predictive prowess. Efforts to circumvent this issue, while maintaining the utility of Wavelet Decomposition, were met with challenges, as these attempts generally fell short of the benchmarks set by established models. Despite these hurdles, with newer models such as Time-series Generative Adversarial Networks (Time GAN) [63] being introduced over recent years, Wavelet Decomposition’s potential for future applications and advancements remains promising, especially when the need for analyzing complex time series data continues to grow across various domains.

References

[1] O. B. Sezer, M. U. Gudelek, and A. M. Ozbayoglu, “Financial time series forecasting with deep learning : A systematic literature review: 2005–2019,” *Appl. Soft Comput.*, vol. 90, p. 106181, May 2020, doi: 10.1016/j.asoc.2020.106181.

[2] A. G. Salman, B. Kanigoro, and Y. Heryadi, “Weather forecasting using deep learning techniques,” in *2015 International Conference on Advanced Computer Science and Information Systems (ICACSIS)*, Oct. 2015, pp. 281–285. doi: 10.1109/ICACSIS.2015.7415154.

[3] Y. Aviv, “A Time-Series Framework for Supply-Chain Inventory Management,” *Oper. Res.*, vol. 51, no. 2, pp. 210–227, Apr. 2003, doi: 10.1287/opre.51.2.210.12780.

[4] A. Vaswani *et al.*, “Attention Is All You Need.” arXiv, Aug. 01, 2023. doi: 10.48550/arXiv.1706.03762.

[5] Q. Wang *et al.*, “Learning Deep Transformer Models for Machine Translation.” arXiv, Jun. 04, 2019. doi: 10.48550/arXiv.1906.01787.

[6] U. Khandelwal, K. Clark, D. Jurafsky, and L. Kaiser, “Sample Efficient Text Summarization Using a Single Pre-Trained Transformer.” arXiv, May 21, 2019. doi: 10.48550/arXiv.1905.08836.

[7] Z. Zhang, Y. Wu, J. Zhou, S. Duan, H. Zhao, and R. Wang, “SG-Net: Syntax Guided Transformer for Language Representation,” *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 44, no. 6, pp. 3285–3299, Jun. 2022, doi: 10.1109/TPAMI.2020.3046683.

[8] Y. Chen, S. Liu, J. Yang, H. Jing, W. Zhao, and G. Yang, “A Joint Time-frequency Domain Transformer for Multivariate Time Series Forecasting.” arXiv, Oct. 28, 2023. doi: 10.48550/arXiv.2305.14649.

[9] “Wavelet decomposition of vibrations for detection of bearing-localized defects - ScienceDirect.” Accessed: Mar. 24, 2024. [Online]. Available: https://www.sciencedirect.com/science/article/abs/pii/S0963869596000527

[10] “Smart meters in London.” Accessed: Mar. 24, 2024. [Online]. Available: https://www.kaggle.com/datasets/jeanmidev/smart-meters-in-london

[11] “Hourly Energy Consumption.” Accessed: Mar. 24, 2024. [Online]. Available: https://www.kaggle.com/datasets/robikscube/hourly-energy-consumption

[12] “vitaliy-sharandin/energy-consumption-hourly-spain · Datasets at Hugging Face.” Accessed: Mar. 24, 2024. [Online]. Available: https://huggingface.co/datasets/vitaliy-sharandin/energy-consumption-hourly-spain

[13] R. M. Alrumaih and M. A. Al-Fawzan, “Time Series Forecasting Using Wavelet Denoising an Application to Saudi Stock Index,” *J. King Saud Univ. - Eng. Sci.*, vol. 14, no. 2, pp. 221–233, Jan. 2002, doi: 10.1016/S1018-3639(18)30755-4.

[14] A. N. Akansu and R. A. Haddad, *Multiresolution Signal Decomposition: Transforms, Subbands, and Wavelets*. Academic Press, 2001.

[15] R. M. Gray and J. W. Goodman, *Fourier Transforms: An Introduction for Engineers*. Springer Science & Business Media, 2012.

[16] M. Sifuzzaman, M. R. Islam, and M. Z. Ali, “Application of Wavelet Transform and its Advantages Compared to Fourier Transform,” 2009, Accessed: Mar. 24, 2024. [Online]. Available: http://inet.vidyasagar.ac.in:8080/jspui/handle/123456789/779

[17] L. Tan and J. Jiang, “Chapter 12 - Subband and Wavelet-Based Coding,” in *Digital Signal Processing (Third Edition)*, L. Tan and J. Jiang, Eds., Academic Press, 2019, pp. 591–648. doi: 10.1016/B978-0-12-815071-9.00012-9.

[18] H. J. Grubb and A. T. Walden, “**Characterizing seismic time series using the discrete wavelet transform**,” *Geophys. Prospect.*, vol. 45, no. 2, pp. 183–205, Mar. 1997, doi: 10.1046/j.1365-2478.1997.00346.x.

[19] Y.-F. Sang, “A Practical Guide to Discrete Wavelet Decomposition of Hydrologic Time Series,” *Water Resour. Manag.*, vol. 26, no. 11, pp. 3345–3365, Sep. 2012, doi: 10.1007/s11269-012-0075-4.

[20] "Applied Sciences | Free Full-Text | Wind Power Short-Term Prediction Based on LSTM and Discrete Wavelet Transform." Accessed: Mar. 24, 2024. [Online]. Available: https://www.mdpi.com/2076-3417/9/6/1108

[21] T. Partal and M. Küçük, "Long-term trend analysis using discrete wavelet components of annual precipitations measurements in Marmara region (Turkey)," *Phys. Chem. Earth Parts ABC*, vol. 31, no. 18, pp. 1189–1200, Jan. 2006, doi: 10.1016/j.pce.2006.04.043.

[22] "Critically-Sampled Discrete Wavelet Transform - MATLAB & Simulink." Accessed: Mar. 24, 2024. [Online]. Available: https://www.mathworks.com/help/wavelet/gs/discrete-wavelet-transform.html

[23] C. Villena Munoz, D. Pazos, E. Andres, and A. Sanz-Lobera, "Influence of different wavelet filtering reconstruction techniques applied to bidimensional surface texture characterization," *IOP Conf. Ser. Mater. Sci. Eng.*, vol. 1193, p. 012056, Oct. 2021, doi: 10.1088/1757-899X/1193/1/012056.

[24] D. B. Percival and A. T. Walden, *Wavelet Methods for Time Series Analysis*. in Cambridge Series in Statistical and Probabilistic Mathematics. Cambridge: Cambridge University Press, 2000. doi: 10.1017/CBO9780511841040.

[25] F. C. A. Fernandes, R. L. C. van Spaendonck, and C. S. Burrus, "A new framework for complex wavelet transforms," *IEEE Trans. Signal Process.*, vol. 51, no. 7, pp. 1825–1837, Jul. 2003, doi: 10.1109/TSP.2003.812841.

[26] "Nondecimated Discrete Stationary Wavelet Transforms (SWTs) - MATLAB & Simulink." Accessed: Mar. 24, 2024. [Online]. Available: https://www.mathworks.com/help/wavelet/ug/discrete-stationary-wavelet-transform-swt.html

[27] "The Stationary Wavelet Transform and some Statistical Applications | SpringerLink." Accessed: Mar. 24, 2024. [Online]. Available: https://link.springer.com/chapter/10.1007/978-1-4612-2544-7_17

[28] "Wikiwand - Stationary wavelet transform," Wikiwand. Accessed: Mar. 24, 2024. [Online]. Available: https://www.wikiwand.com/en/Stationary_wavelet_transform

[29] "IoT | Free Full-Text | A MODWT-Based Algorithm for the Identification and Removal of Jumps/Short-Term Distortions in Displacement Measurements Used for Structural Health Monitoring." Accessed: Mar. 24, 2024. [Online]. Available: https://www.mdpi.com/2624-831X/3/1/3#B49-IoT-03-00003

[30] D. B. Percival, "Analysis of Geophysical Time Series Using Discrete Wavelet Transforms: An Overview," in *Nonlinear Time Series Analysis in the Geosciences*, vol. 112, R. V. Donner and S. M. Barbosa, Eds., in Lecture Notes in Earth Sciences, vol. 112. , Berlin, Heidelberg: Springer Berlin Heidelberg, 2008, pp. 61–79. doi: 10.1007/978-3-540-78938-3_4.

[31] "Sensors | Free Full-Text | The Detection of Motor Bearing Fault with Maximal Overlap Discrete Wavelet Packet Transform and Teager Energy Adaptive Spectral Kurtosis." Accessed: Mar. 24, 2024. [Online]. Available: https://www.mdpi.com/1424-8220/21/20/6895

[32] "Energies | Free Full-Text | Denoising of Heavily Contaminated Partial Discharge Signals in High-Voltage Cables Using Maximal Overlap Discrete Wavelet Transform." Accessed: Mar. 24, 2024. [Online]. Available: https://www.mdpi.com/1996-1073/14/20/6540

[33] Q. Wen *et al.*, "Transformers in Time Series: A Survey." arXiv, May 11, 2023. doi: 10.48550/arXiv.2202.07125.

[34] N. Wu, B. Green, X. Ben, and S. O'Banion, "Deep Transformer Models for Time Series Forecasting: The Influenza Prevalence Case." arXiv, Jan. 22, 2020. doi: 10.48550/arXiv.2001.08317.

[35] P. Brugiere and G. Turinici, "Transformer for Times Series: an Application to the S&P500." arXiv, Mar. 04, 2024. doi: 10.48550/arXiv.2403.02523.

[36] "A Transformer Self-attention Model for Time Series Forecasting." Accessed: Mar. 24, 2024. [Online]. Available: https://jecei.sru.ac.ir/article_1477.html

[37] Y. Chen, K. Ren, Y. Wang, Y. Fang, W. Sun, and D. Li, “ContiFormer: Continuous-Time Transformer for Irregular Time Series Modeling,” *Adv. Neural Inf. Process. Syst.*, vol. 36, pp. 47143–47175, Dec. 2023.

[38] Y. Li, C. Tian, Y. Lan, C. Yu, and K. Xie, “Transformer with Sparse Attention Mechanism for Industrial Time Series Forecasting,” *J. Phys. Conf. Ser.*, vol. 2026, no. 1, p. 012036, Sep. 2021, doi: 10.1088/1742-6596/2026/1/012036.

[39] J. W. Chan and C. K. Yeo, “Electrical Power Consumption Forecasting with Transformers,” in *2022 IEEE Electrical Power and Energy Conference (EPEC)*, Dec. 2022, pp. 255–260. doi: 10.1109/EPEC56903.2022.10000228.

[40] V. Melnychuk, D. Frauen, and S. Feuerriegel, “Causal Transformer for Estimating Counterfactual Outcomes,” in *Proceedings of the 39th International Conference on Machine Learning*, PMLR, Jun. 2022, pp. 15293–15329. Accessed: Mar. 24, 2024. [Online]. Available: https://proceedings.mlr.press/v162/melnychuk22a.html

[41] T. Zhou, Z. Ma, Q. Wen, X. Wang, L. Sun, and R. Jin, “FEDformer: Frequency Enhanced Decomposed Transformer for Long-term Series Forecasting,” in *Proceedings of the 39th International Conference on Machine Learning*, PMLR, Jun. 2022, pp. 27268–27286. Accessed: Mar. 24, 2024. [Online]. Available: https://proceedings.mlr.press/v162/zhou22g.html

[42] S. Singh, K. S. Parmar, J. Kumar, and S. J. S. Makkhan, “Development of new hybrid model of discrete wavelet decomposition and autoregressive integrated moving average (ARIMA) models in application to one month forecast the casualties cases of COVID-19,” *Chaos Solitons Fractals*, vol. 135, p. 109866, Jun. 2020, doi: 10.1016/j.chaos.2020.109866.

[43] H. Li *et al.*, “DnSwin: Toward real-world denoising via a continuous Wavelet Sliding Transformer,” *Knowl.-Based Syst.*, vol. 255, p. 109815, Nov. 2022, doi: 10.1016/j.knosys.2022.109815.

[44] A. Tam, “What are Large Language Models,” MachineLearningMastery.com. Accessed: Mar. 24, 2024. [Online]. Available: https://machinelearningmastery.com/what-are-large-language-models/

[45] “What Are Large Language Models (LLMs)? | IBM.” Accessed: Mar. 24, 2024. [Online]. Available: https://www.ibm.com/topics/large-language-models

[46] L. Saad Saoud, H. Al-Marzouqi, and R. Hussein, “Household Energy Consumption Prediction Using the Stationary Wavelet Transform and Transformers,” *IEEE Access*, vol. 10, pp. 5171–5183, 2022, doi: 10.1109/ACCESS.2022.3140818.

[47] K. G. A. Ludvigsen, “How to make a PyTorch Transformer for time series forecasting,” Medium. Accessed: Mar. 24, 2024. [Online]. Available: https://towardsdatascience.com/how-to-make-a-pytorch-transformer-for-time-series-forecasting-69e073d4061e

[48] “Stationary Wavelet Transform — PyWavelets Documentation.” Accessed: Mar. 24, 2024. [Online]. Available: https://pywavelets.readthedocs.io/en/latest/ref/swt-stationary-wavelet-transform.html

[49] S. M. Kazemi *et al.*, “Time2Vec: Learning a Vector Representation of Time.” arXiv, Jul. 11, 2019. doi: 10.48550/arXiv.1907.05321.

[50] “(PDF) Development of Deep Transformer-Based Models for Long-Term Prediction of Transient Production of Oil Wells.” Accessed: Mar. 24, 2024. [Online]. Available: https://www.researchgate.net/publication/355186549_Development_of_Deep_Transformer-Based_Models_for_Long-Term_Prediction_of_Transient_Production_of_Oil_Wells

[51] “Choose a Wavelet - MATLAB & Simulink.” Accessed: Mar. 24, 2024. [Online]. Available: https://www.mathworks.com/help/wavelet/gs/choose-a-wavelet.html

[52] “Time Series Made Easy in Python — darts documentation.” Accessed: Mar. 24, 2024. [Online]. Available: https://unit8co.github.io/darts/index.html

[53] Y. Liu, S. Dutta, A. W. K. Kong, and C. K. Yeo, “An Image Inpainting Approach to Short-Term Load Forecasting,” *IEEE Trans. Power Syst.*, vol. 38, no. 1, pp. 177–187, Jan. 2023, doi: 10.1109/TPWRS.2022.3159493.

[54] A. Zeng, M. Chen, L. Zhang, and Q. Xu, “Are Transformers Effective for Time Series Forecasting?” arXiv, Aug. 17, 2022. doi: 10.48550/arXiv.2205.13504.

[55] “cure-lab/LTSF-Linear.” cure-lab, Mar. 22, 2024. Accessed: Mar. 24, 2024. [Online]. Available: https://github.com/cure-lab/LTSF-Linear

[56] H. T. Nguyen and I. T. Nabney, “Short-term electricity demand and gas price forecasts using wavelet transform and adaptive models,” *Energy*, vol. 35, no. 9, pp. 3674–3685, Sep. 2010, doi: 10.1016/j.energy.2010.05.013.

[57] Y. Kajita and R. hasumi, *Boundary problem and data leakage: A caveat for wavelet-based forecasting*. 2018. doi: 10.13140/RG.2.2.28234.21446.

[58] A. AUSSEM and F. MURTAGH, “Combining Neural Network Forecasts on Wavelet-transformed Time Series,” *Connect. Sci.*, vol. 9, no. 1, pp. 113–122, Mar. 1997, doi: 10.1080/095400997116766.

[59] ADMIN, “A guide for using the Wavelet Transform in Machine Learning,” ML Fundamentals. Accessed: Mar. 24, 2024. [Online]. Available: https://ataspinar.com/2018/12/21/a-guide-for-using-the-wavelet-transform-in-machine-learning/

[60] L. Sasal, T. Chakraborty, and A. Hadid, “W-Transformers : A Wavelet-based Transformer Framework for Univariate Time Series Forecasting,” in *2022 21st IEEE International Conference on Machine Learning and Applications (ICMLA)*, Dec. 2022, pp. 671–676. doi: 10.1109/ICMLA55696.2022.00111.

[61] CapWidow, “CapWidow/W-Transformer.” Feb. 21, 2024. Accessed: Mar. 24, 2024. [Online]. Available: https://github.com/CapWidow/W-Transformer

[62] “Transformer Model — darts documentation.” Accessed: Mar. 24, 2024. [Online]. Available: https://unit8co.github.io/darts/generated_api/darts.models.forecasting.transformer_model.html

[63] X. Li, V. Metsis, H. Wang, and A. H. H. Ngu, “TTS-GAN: A Transformer-Based Time-Series Generative Adversarial Network,” in *Artificial Intelligence in Medicine*, M. Michalowski, S. S. R. Abidi, and S. Abidi, Eds., Cham: Springer International Publishing, 2022, pp. 133–143. doi: 10.1007/978-3-031-09342-5_13.

Appendix

Table 0.1 DAYTON dataset with lookback of 5 weeks

Models	Forecast time steps = 24				Forecast time = 84			
	MSE	RMSE	MAE	sMAPE	MSE	RMSE	MAE	sMAPE
ARIMA	0.007771	0.083266	0.067266	0.212853	0.016245	0.121236	0.098554	0.287474
AutoARIMA	0.011233	0.096365	0.078983	0.239599	0.022826	0.132445	0.111324	0.343602
THETA	0.004233	0.055464	0.046969	0.145460	0.013006	0.106071	0.088823	0.257002
BATS	0.027118	0.143873	0.116736	0.308663	0.045994	0.184200	0.159402	0.513655
RNN	0.005731	0.067990	0.056420	0.179145	0.015290	0.113586	0.094190	0.280681
FFT	0.009612	0.082702	0.073941	0.213673	0.032287	0.168193	0.142728	0.399786
Darts Transformer	0.003349	0.049119	0.040837	0.118589	0.010419	0.090818	0.072594	0.189087
tsRNN 3d	0.003278	0.049726	0.042638	0.116603	0.005016	0.064182	0.053677	0.147535
Sparse Transformer	0.002833	0.047215	0.040934	0.115113	0.005813	0.069775	0.059297	0.165900
Hybrid SWT Transformer	0.000024	0.004941	0.003983	0.011926	0.000027	0.005199	0.003957	0.012429

Table 0.2 DAYTON dataset with lookback of 9 weeks

Models	Forecast time steps = 24				Forecast time = 84			
	MSE	RMSE	MAE	sMAPE	MSE	RMSE	MAE	sMAPE
ARIMA	0.009331	0.091101	0.074599	0.228058	0.018853	0.129876	0.104633	0.311659
AutoARIMA	0.021452	0.129758	0.103905	0.282965	0.039215	0.175412	0.150184	0.501068
THETA	0.007152	0.073065	0.062405	0.186585	0.013071	0.105323	0.087881	0.268034
BATS	0.025677	0.142433	0.116146	0.309219	0.067132	0.229568	0.200825	0.690368
RNN	0.007395	0.078319	0.064331	0.197148	0.016900	0.119525	0.098154	0.282889
FFT	0.008831	0.086779	0.076602	0.217789	0.026475	0.157680	0.132763	0.382214
Darts Transformer	0.005055	0.060833	0.051242	0.162417	0.018699	0.123758	0.101481	0.294170
tsRNN 3d	---	---	---	---	---	---	---	---
Sparse Transformer	0.002839	0.047980	0.041245	0.115906	0.005045	0.065627	0.054892	0.150308
Hybrid SWT Transformer	0.000026	0.005103	0.003918	0.011794	0.000028	0.005343	0.004102	0.013102

Table 0.3 AEP dataset with lookback of 5 weeks

Models	Forecast time steps = 24				Forecast time = 84			
	MSE	RMSE	MAE	sMAPE	MSE	RMSE	MAE	sMAPE
ARIMA	0.006930	0.075170	0.063940	0.203289	0.018565	0.129176	0.108127	0.416272
AutoARIMA	0.006778	0.074168	0.062603	0.198117	0.023342	0.140183	0.119262	0.477150
THETA	0.006581	0.072560	0.061841	0.197526	0.014239	0.110805	0.094506	0.373129
BATS	0.007262	0.077979	0.065551	0.208598	0.045498	0.183891	0.157286	0.569286
RNN	0.005948	0.068487	0.058451	0.177022	0.031685	0.150678	0.126016	0.467781
FFT	0.015135	0.103526	0.095560	0.292453	0.042960	0.188577	0.160367	0.565045
Darts Transformer	0.004147	0.054294	0.046716	0.163516	0.014276	0.101756	0.087596	0.332713
tsRNN 3d	0.003341	0.050079	0.04291	0.148659	0.007861	0.079499	0.067517	0.235123
Sparse Transformer	0.004265	0.056371	0.049342	0.176805	0.009607	0.090455	0.077057	0.274690
Hybrid SWT Transformer	0.000024	0.00487	0.003755	0.01775	0.000042	0.006456	0.004965	0.024262

Table 0.4 AEP dataset with lookback of 9 weeks

Models	Forecast time steps = 24				Forecast time = 84			
	MSE	RMSE	MAE	sMAPE	MSE	RMSE	MAE	sMAPE
ARIMA	0.007316	0.076503	0.064454	0.166473	0.020437	0.136065	0.115484	0.439541
AutoARIMA	0.007056	0.074607	0.062325	0.159679	0.023784	0.142964	0.122469	0.476556
THETA	0.006443	0.069785	0.058167	0.150612	0.018294	0.127621	0.110312	0.419285
BATS	0.007152	0.075191	0.062893	0.161014	0.030790	0.155657	0.133634	0.491681
RNN	0.009873	0.086301	0.073174	0.183013	0.014043	0.103511	0.088167	0.313351
FFT	0.014890	0.105435	0.097628	0.235798	0.021431	0.134079	0.116261	0.450772
Darts Transformer	0.003686	0.052259	0.044207	0.155744	0.0111580	0.094910	0.078213	0.314904
tsRNN 3d	---	---	---	---	---	---	---	---
Sparse Transformer	0.004099	0.055853	0.048464	0.171754	0.012123	0.099704	0.082513	0.285518
Hybrid SWT Transformer	0.000019	0.004412	0.003381	0.018547	0.000039	0.006312	0.003734	0.020134

Table 7.5 Spain REE dataset with lookback of 5 weeks

Models	Forecast time steps = 24				Forecast time = 84			
	MSE	RMSE	MAE	sMAPE	MSE	RMSE	MAE	sMAPE
ARIMA	0.040969	0.193608	0.161293	0.375684	0.053211	0.224635	0.187541	0.435242
AutoARIMA	0.035619	0.181127	0.155181	0.362554	0.039126	0.193902	0.166430	0.395569
THETA	0.014530	0.100807	0.084132	0.209597	0.030110	0.156630	0.128844	0.307471
BATS	0.046731	0.206404	0.172081	0.405896	0.064659	0.241953	0.204346	0.481989
RNN	0.017714	0.116611	0.097180	0.228966	0.037752	0.184171	0.148238	0.355536
FFT	0.015860	0.113758	0.099472	0.261884	0.064251	0.250450	0.209509	0.490494
Darts Transformer	0.009851	0.087260	0.071691	0.180186	0.016061	0.120843	0.096582	0.230945
tsRNN 3d	0.009026	0.074298	0.062942	0.15351	0.010662	0.092708	0.076103	0.182457
Sparse Transformer	0.009751	0.090407	0.075050	0.188204	0.039933	0.198071	0.173116	0.400399
Hybrid SWT Transformer	0.000036	0.005965	0.004703	0.014128	0.000045	0.006707	0.005222	0.01651

Table 7.6 Spain REE dataset with lookback of 9 weeks

Models	Forecast time steps = 24				Forecast time = 84			
	MSE	RMSE	MAE	sMAPE	MSE	RMSE	MAE	sMAPE
ARIMA	0.041983	0.193393	0.159705	0.384354	0.051422	0.220348	0.184343	0.443426
AutoARIMA	0.031480	0.170529	0.143805	0.353049	0.040444	0.196312	0.167382	0.408835
THETA	0.014025	0.100326	0.083711	0.204590	0.028279	0.153058	0.128049	0.320370
BATS	0.050375	0.212212	0.175976	0.427164	0.054031	0.220847	0.187124	0.439574
RNN	0.020027	0.123868	0.098977	0.255039	0.027695	0.153355	0.124795	0.302377
FFT	0.015393	0.117626	0.101700	0.282064	0.046121	0.213211	0.179294	0.453081
Darts Transformer	0.018694	0.114177	0.091720	0.252173	0.047467	0.208333	0.161400	0.438727
tsRNN 3d	---	---	---	---	---	---	---	---
Sparse Transformer	0.009900	0.089423	0.076584	0.203271	0.011926	0.098529	0.081171	0.197058
Hybrid SWT Transformer	0.000032	0.00568	0.004464	0.014317	0.000053	0.007304	0.005635	0.017942