NOVA IMS Information Management School

MDSAA

Master Degree Program in

Data Science and Advanced Analytics

Transformers in Time Series Forecasting

A Systematic Literature Review

Diogo Miguel Goulart Ferreira Pereira da Silva

Master Thesis

presented as partial requirement for obtaining a Master's Degree in Data Science and Advanced Analytics

NOVA Information Management School
Instituto Superior de Estatística e Gestão de Informação
Universidade Nova de Lisboa

NOVA Information Management School
Instituto Superior de Estatística e Gestão de Informação
Universidade Nova de Lisboa

Transformers in Time Series Forecasting

A Systematic Literature Review

by

Diogo Miguel Goulart Ferreira Pereira da Silva

Master Thesis presented as partial requirement for obtaining the Master's degree in Data Science and Advanced Analytics, with a specialization in Data Science.

Supervised by

Roberto Henriques, PhD, Nova IMS

July, 2024

STATEMENT OF INTEGRITY

I hereby declare having conducted this academic work with integrity. I confirm that I have not used plagiarism or any form of undue use of information or falsification of results along the process leading to its elaboration. I further declare that I have fully acknowledged the Rules of Conduct and Code of Honor from the NOVA Information Management School.

[Lisbon, 11/07/2024]

DEDICATION

Para o meu avô Antônio, que brilha sempre, mesmo na adversidade. Tentarei sempre seguir os teus passos.

ACKNOWLEDGEMENTS

Agradeço à minha família, o meu primeiro pilar, o meu abrigo e a minha casa, independentemente de tudo.

Agradeço à minha mãe, que sempre fez os impossíveis por mim, e por a minha educação ter sido sempre uma das suas grandes prioridades. Obrigado por todas as portas que me abriste, a teu custo. Onde quer que chegue só vai ser graças a teres me levantado nos teus ombros.

Agradeço aos meus avôs que foram incansáveis em nunca me faltar nada, quer em grande plano quer por de trás das cenas, em gestos pequenos e grandes, sempre consistentes e presentes. E que, apesar de não perceberem nada deste assunto, foram incansáveis a acompanhar a escrita desta tese, sempre preocupados e a querer saber como estava a correr, e até a tentaram ler.

Agradeço à Ana por, como boa irmã, ser sempre direta e honesta e me trazer de volta à realidade quando preciso.

Agradeço ao Sérgio, que é sempre fonte de calma e segurança, e arranja sempre tempo e boa disposição quando necessário, sempre disposto a ajudar.

Agradeço aos meus amigos que me mantiveram são neste ano difícil, vocês sabem quem são.

Agradeço ao meu orientador, Professor Roberto, por ter tido sempre disponibilidade quando foi necessário, principalmente na altura stressante em que tive que mudar de tema de tese.

E agradeço à Nova IMS por me ter dado a oportunidade de neste mestrado descobrir a minha verdadeira vocação.

ABSTRACT

The growing interest in transformers for time series forecasting has triggered an exponential surge in publications, making it increasingly challenging to organize and understand emerging trends and discern future directions. This thesis is a systematic literature review of the subject, that aims to study how the different components and aspects of the transformers evolved, finding trends, and highlighting gaps and future directions.

The review focuses on papers in English, peer reviewed, published in reputable sources that focus exclusively on transformer models for time series forecasting. It spans sources such as ACM, ARXiv, Elsevier, MDPI, OpenReview, AAAI, IJCAI, IEEEExplore, NeurIPS, Springer, epubs, and Jstage, from December of 2023 to April of 2024 – selecting a total of 99 papers.

To assess the risk of bias, statements regarding conflicts of interest and sources of funding were examined. Publication bias was evaluated by comparing transformer results over time. The reviewed papers reveal a preference for nine datasets in specific but a lot of papers still use unique private datasets, with some variation of the time-horizons used as well as the metrics.

Recent advancements showcase innovations in Input Representation (Positional Encoding, Segmentation, Decomposition and covariates), Modelling (Attention Mechanism, Feature Selection, Channel Dependence or Independence, Multi-scale and Hierarchical approaches, and Hybridization), Model Optimization (Regularization, Loss function, and Normalization), and Types of Learning. These advancements aim to enhance model adaptability, accuracy, robustness, and computational efficiency, as well as improve the capture of local context and temporal relationships at different granularities.

However, there is a limited understanding of transformer performance across diverse domains and datasets, as most studies focus on tailored models rather than cross-domain evaluations. The lack of standardized benchmarks complicates model assessment, necessitating more transparent and explainable models. Scalability and computational efficiency remain major concerns, especially for large-scale and real-time implementations.

KEYWORDS

Transformer; Time Series; Forecast; Attention Mechanism; Deep Learning

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF ABBREVIATIONS AND ACRONYMS

ARIMA	AutoRegressive Integrated Moving Average
RNN	Recurrent Neural Network
LSTM	Long Short-Term Memory
GRU	Gated Recurrent Unit
TCN	Temporal Convolutional Network
CNN	Convolutional Neural Network
NLP	Natural Language Processing
LTSF	Long Time Series Forecasting
STF	Spatio-Temporal Forecasting
MAE	Mean Absolute Error
MAPE	Mean Absolute Percentage Error
RMSE	Root Mean Squared Error
CRPS	Continuous Ranked Probability Score
FFT	Fast Fourier Transform
STL	Seasonal-Trend Decomposition
ODE	Ordinary Differential Equation

1. INTRODUCTION

Predicting the future can significantly influence decision-making processes across various sectors. In retail, accurate demand forecasting aids in inventory management, reducing overstock costs and minimizing waste (Honjo et al., 2022). In healthcare, predicting disease spread can optimize resource allocation and preventive strategies (Garg et al., 2012), as evidenced during the Covid-19 pandemic (Chimmula & Zhang, 2020). Similarly, in the stock market, accurate stock predictions assist investors in deciding on whether to buy, hold, or sell (Kim & Kim, 2019; C.-H. Su & Cheng, 2016). Traffic flow forecasts can optimize travel routes and times (Abadi et al., 2015), while precise electricity production and consumption predictions are crucial for infrastructure planning (Ghimire et al., 2023). In agriculture, weather forecasting plays a pivotal role (Hachimi et al., 2023; Fernandes et al., 2020).

This thesis exclusively focuses on time series forecasting. Initially, classical statistical models like ARIMA were used for non-stationary data by differencing it (Ham et al., 2017). As datasets grew and got more complex, these methods were outgrown due to their limited scalability and linear assumptions. They also required prior knowledge about time series structures, such as trend and seasonality. They could not utilize time-related features, and information not being sharable between different variables (S. Li et al., 2019).

The need to overcome these obstacles and the increased computational power led to the adoption of deep-learning neural networks. Recurrent Neural Networks (RNNs) were proposed to model temporal dependencies but suffered from vanishing and exploding gradients (Ribeiro et al., 2020). Variants like LSTM and GRU were introduced to mitigate these issues but still struggled with long-term forecasting. (S. Li et al., 2019) Later, state-of-the-art models like Temporal Convolutional Networks (TCNs) and attention mechanisms applied to existing models emerged, but they often captured either short-term or long-term dependencies, not both (Cirstea et al., 2022).

Transformers were conceived for machine translation, and were based solely on attention mechanisms, which disposed entirely of recurrence and convolutions (Vaswani et al., 2023). Their main advantages include parallel processing of the data, which is increasing more relevant as the data grows more extensive and more complex, scalability, and good at handling long sequences. After achieving outstanding results in Natural Language Processing and Computer Vision, they were then applied in time series forecasting with promising results.

However, adaptations were necessary since transformers weren't originally designed for this end. This led to the creation of specialized transformers such as Informer (H. Zhou et al., 2020), Autoformer (H. Wu et al., 2021), and FEDformer (T. Zhou et al., 2022). Yet, these were only some of the first transformers that would be created for this end.

Due to the growing interest in the area as well as the success of the proposed models, there has been an exponential increase of published papers, focusing on different aspects and making it increasingly challenging to organize and understand emerging trends, as well as to discern future directions. Additionally, existing surveys and reviews quickly became outdated. This poses a significant challenge for those entering the field and potentially overwhelming even experts.

This thesis aims to address this gap by posing the **research question**: "How have transformer models evolved to address the specific challenges of time series forecasting, and what are the emerging trends and future directions in this area?", with the **research objectives** being to:

- Identify and analyse the emerging trends, innovations, and methodological advancements in transformer models tailored for time series forecasting.
- Pinpoint existing gaps, challenges, and limitations in the scientific knowledge and of state-of-the-art transformer models
- Propose future research directions and potential technological advancements.

The **organization** of this thesis will be as follows:

Chapter 2. Background Theory of Transformers: Explains the theory behind transformers; **Chapter 3. Methodology**: Explaining the search strategy used to find the papers, the inclusion and exclusion criteria, the data collection process, state the primary and secondary outcomes and the method to assess the risk of bias; **Chapter 4. Papers' Analysis**: Focuses on publication dates, sources, code availability, and biases, as well as specific information like datasets, metrics, and loss functions used. Interesting but less relevant information is in Appendix B (Papers' Study Extra); **Chapter 5. Evolution and Discussion**: Shows the evolution of transformers, divided by component of the transformer architecture. This approach provides a clear understanding of how different components change across studies; **Chapter 6. Conclusion and Future Work**.

Appendix A (Complexity Equations): Shows the different complexity equations shown in the papers and the mechanisms to reach them; **Appendix B (Papers' Study Extra)** – Studies on interesting but not the most relevant characteristics – number of pages, number of paged of the annex, number of references used, number of authors and where the authors are from - were moved to this appendix; **Appendix C (Table of Summary of Changes)**: A table summarizing the changes made to each transformer, whether the code is publicly available, the motivation/ issue addressed, and the reference; **Appendix D (Aggregation of Results)**: Section focused on agglomerating the final metrics of different papers.

2. BACKGROUND THEORY OF TRANSFORMERS

The transformer was proposed by Vaswani et al. (2023), designed for Machine Translation. This section focuses on explaining the inner mechanics of the architecture, with the main contributions for the section being the previous paper, L. Su et al. (2023), Tay et al. (2022), and T. Lin et al. (2021).

Figure 2.1 shows the architecture of the transformer, that uses self-attention and point-wise fully connected layers for both the encoder and the decoder.

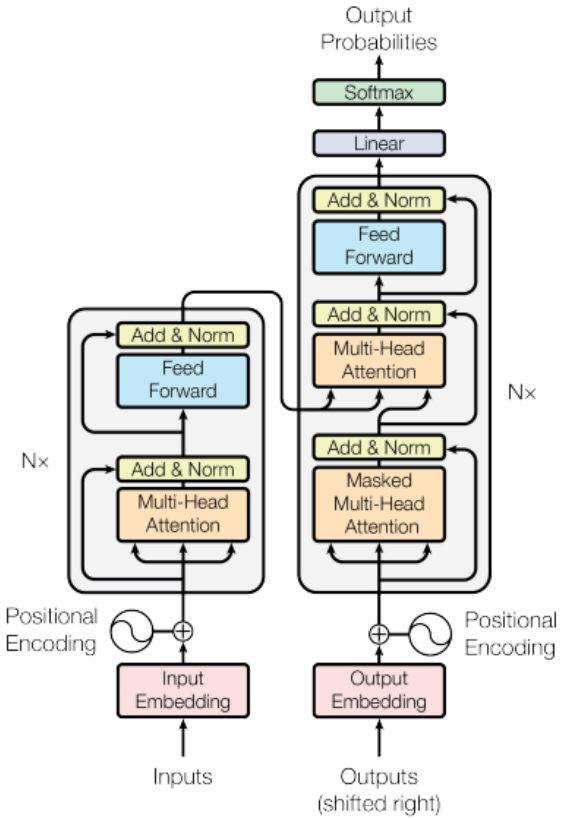

Figure 2.1 – Architecture of the Transformer model (Vaswani et al., 2023)

2.1. ENCODER AND DECODER

The **encoder** is depicted on the left-hand side of Figure 2.1. The encoder component comprises two primary networks: the multi-head attention mechanism and the two-layer feed-forward neural network. Each of these sub-networks benefits from the addition of residual connections, which help mitigate the vanishing gradient problem and enable the network to train deeper layers more effectively, as shown by He et al. (2015), Layer normalization follows these residual connections, ensuring that the outputs are stable and normalized, represented as $LayerNorm(x + Sublayer(x))$. Additionally, dropout is applied to prevent overfitting.

The two-layer feed-forward network (FFN) in the second part of the encoder processes the input feature representation x through the following sequence of operations:

$$FFN(x) = \max(0, xW_1 + b)\, W_2 + \ b_2 \qquad (1)$$

Where W_1 and W_2 denote the weight matrices, while b_1 and b_2 represent the bias vectors. The function $\max(0, \dots)$ signifies the utilization of ReLU as the activation function.

The **decoder** is similar to the encoder, albeit with the inclusion of an extra multi-head attention mechanism that interacts with the encoder output. The decoder consists of three primary sub-networks: two similar to the encoder's sub-networks and a third that performs encoder-decoder attention.

In the encoder-decoder attention component, the input query Q is derived from the output of the preceding multi-head attention mechanism, while the keys K and values V are linearly transformed outputs (Memory) from the encoder. This structure ensures that the decoder can effectively leverage the encoded input information to produce accurate and contextually relevant outputs.

2.2. POSITIONAL ENCODING

In the realm of modelling text-related data, the initial step involves vectorizing the text. Traditional methods such as one-hot encoding, bag-of-words models, and TF-IDF are frequently used for text representation. However, in deep learning, it is more common to associate individual words or tokens with a low-dimensional, dense vector space using an embedding layer. This process is referred to as token embedding or word embedding.

In architectures like Convolutional Neural Networks (CNNs) or Recurrent Neural Networks (RNNs), the text vectorization phase is often sufficient as these networks inherently capture temporal features. CNNs capture local features while RNNs capture temporal sequences. However, this does not apply to Transformer networks, which lack recursion and convolution. Transformers rely solely on a self-attention mechanism, which operates through linear transformations by multiplying multiple matrices. This means that without additional mechanisms, Transformers are agnostic to the order of the input sequence, leading to potential loss of the original text sequence structure.

To address the lack of inherent sequence order in Transformers, positional encoding is added to the input embeddings. Positional encoding provides the model with information about the position of each token in the sequence, thus preserving the sequence order. The original Transformer employs sinusoidal positional encodings. These encodings are designed to inject information about the relative and absolute positions of tokens in the sequence. The encoding vectors are of the same dimensionality (d_{model}) as the input embeddings, enabling element-wise addition. The sinusoidal positional encodings are defined as:

$$PosEmb(pos, 2i) = \sin\left(\frac{pos}{10000^{\frac{2i}{d_{model}}}}\right) \quad (2)$$

$$PosEmb(pos, 2i+1) = cos\left(\frac{pos}{10000^{\frac{2i}{d_{model}}}}\right) \quad (3)$$

Here, pos represents the position, and i represents the dimension. Each dimension of the positional encoding corresponds to a sinusoid, with wavelengths forming a geometric progression from 2π to $1000 \cdot 2\pi$. The 2 equations represent the positional encoding for position pos at the $2i$ -th dimension (even index) and the $(2i+1)$-th dimension (odd index).

This design allows the model to attend to tokens by relative positions, as any fixed offset K can be represented as a linear function of the positional encoding.

2.3. SELF-ATTENTION

The self-attention mechanism is a fundamental component of the Transformer architecture, enabling the model to capture dependencies between different positions in a sequence. Unlike recurrent and convolutional neural networks, self-attention allows for direct connections between all tokens in a sequence, providing a powerful method to model long-range dependencies efficiently.

An attention function can be described as mapping a query and a set of key-value pairs to an output, where the query, keys, values, and output are all vectors. The output is computed as a weighted sum of the values, where the weight assigned to each value is determined by a compatibility function of the query with the corresponding key.

The specific attention mechanism used in Transformers is called "Scaled Dot-Product Attention." The input consists of queries (Q), keys (K), and values (V) matrices. The dimensions of the queries and keys are d_k , and the dimension of the values is d_v. The process can be summarized as follows:

1. **Dot Products**: Compute the dot products of the query with all keys.
2. **Scaling**: Divide each dot product by the square root of the dimension of the keys.
3. **Softmax**: Apply the softmax function to obtain the weights for the values.
4. **Weighted Sum**: Compute the output as the weighted sum of the values.

The mathematical formula is:

$$Attention(Q, K, V) = softmax\left(\frac{QK^T}{\sqrt{d_k}}\right)V \quad (4)$$

There are two commonly used attention mechanisms: additive attention and dot-product (multiplicative) attention. Additive attention computes the compatibility function using a

feed-forward network with a single hidden layer, while dot-product attention directly uses the dot products of queries and keys. Dot-product attention, especially when scaled, is faster and more space-efficient as it leverages optimized matrix multiplication code. For small values of d_k, both mechanisms perform similarly. However, for larger values of d_k, scaling the dot products by $\sqrt{d_k}$ is necessary to keep the softmax gradients manageable and the model trainable.

2.4. MULTI-HEAD ATTENTION

Multi-head attention is a fundamental component in transformer models. This mechanism extends the concept of single-head attention by employing multiple attention heads to process the input data in parallel, each head attending to different aspects of the input simultaneously.

The primary idea is to project the queries (Q), keys (K), and values (V) linearly into multiple subspaces, each of reduced dimensionality. This is achieved through different learned linear transformations, creating multiple sets of Q, K, and V for each head. Specifically, the queries, keys, and values are projected into d_q, d_k, and d_v dimensions, respectively. These projections are then processed through the attention function concurrently. The output from each head is a d_v - dimensional vector, and these outputs are concatenated and linearly transformed to produce the final output. The formula for this operation is as follows:

$$Multhead(Q,K,V)Concat = (head_1, \dots, head_h)W_O \qquad (5)$$, where:

$$head_i = Attention(QW_{Qi}, kW_{Ki}, VW_{Vi}) \qquad (6)$$

Multi-head attention allows the model to attend to information from different representation subspaces at various positions. This is particularly beneficial as it enables the model to capture richer relationships within the data compared to single-head attention, which may average out subtle but crucial details.

For example, in a model with $h = 8$ heads, if the model dimensionality (d_{model}) is 512, each head operates in a subspace where $d_q = d_k = d_v = \frac{d_{model}}{h} = 64$. Despite the increase in the number of attention heads, the computational cost remains manageable due to the reduced dimensionality per head.

Types of Multi-Head Attention:

- In **self-attention**, the queries, keys, and values are derived from the same input sequence, $Q = K = V = X$. This allows the model to capture dependencies within the same sequence.
- Used in the Transformer decoder, **masked self-attention** ensures that the model does not attend to future positions in the sequence. This is achieved by applying a mask to the attention matrix, setting illegal positions to $-\infty$.

- In **cross-attention**, the queries come from the previous decoder layer, while the keys and values come from the encoder's output. This mechanism enables the decoder to attend to different parts of the input sequence, facilitating more effective information integration from the encoder.

3. METHODOLOGY

This paper will follow the PRISMA (Preferred Reporting Items for Systematic Reviews and Meta-Analyses) guidelines to ensure a comprehensive and transparent literature review process.

A comprehensive search strategy was implemented across several academic databases and repositories to identify relevant publications. The searched databases included IEEE Xplore, Scopus, Web of Science, ScienceDirect, Semantic Scholar, and Google Scholar for broader reach. Relevant GitHub repositories were also reviewed for additional resources.

The search was conducted by searching the following keywords in the following sources:

- **IEEE Xplore, Web of Science, and Scopus**: "transformer" AND "time series" AND "forecast" in the search fields: Title, Abstract, or Keywords.
- **Semantic Scholar and Google Scholar**: "transformer", "time series", "forecast"
- **GitHub Repositories**: Key repositories searched included:
    - Awesome Time Series Forecasting/Prediction Papers (ddz16, 2022/2024)
    - Transformers in Time Series Review (Wen, 2022/2024b)
    - AI for Time Series (AI4TS) Papers, Tutorials, and Surveys (Wen, 2022/2024a)
    - Time Series AI Papers (xiyuanzh, 2021/2024)

Inclusion and Exclusion Criteria - Publications were selected based on the following criteria:

- **Language**: Only papers written in English were considered.
- **Content**: The paper must specifically address time series forecasting - not any other kind of data nor other area related to time series.
- **Publication Status**: Only peer-reviewed articles published in reputable sources were included.
- **Accessibility**: Documents must be accessible with a Nova IMS student ID.

A minimum number of citations was not used as a criterion since this area is very recent and most of the papers have been published in the last 2 years, so a small number of citations does not have a lot of meaning. How the papers were selected is shown in Figure 3.1.

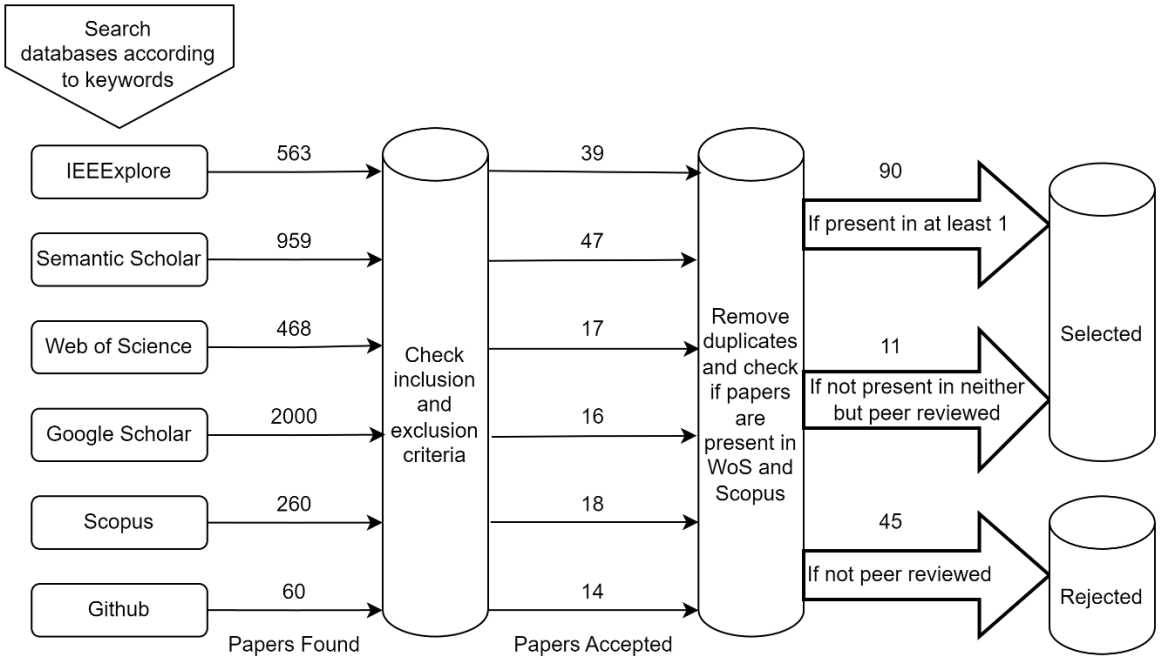

Figure 3.1 – Schematic Representation of the Paper's Selection Process

Of the 11 papers not present in Web of Science nor Scopus but peer-reviewed – eight were accepted as conference posters and the other three were published in journals. These were accepted as they all are peer-reviewed, and one of these cases is the paper on PatchTST, which is cited frequently in later papers.

There were three main reasons why that initially passed the selection phase were subsequently rejected upon closer inspection: Only the preprint was available as seen with papers by Eisenach et al. (2022), and Nivron et al. (2023); The focus was on spatio-temporal forecasting rather than time-series forecasting, as in the work by Liang et al. (2022); The model proposed was not a transformer, exemplified by the study from Z. Gong et al. (2023).

For the **Data Collection Process**, data from each paper were meticulously catalogued manually into an Excel sheet. This allowed for verification and summarization while reading and annotating the documents. Recorded data fields include:

- In the excel sheet "info" - Transformer name, paper title, URL, publication date, search engine, hosting website, document type, event name, event date, keywords, if peer review is public, authors and affiliations, country of origin of funding, funding type (public, private, or both), supplementary material existence, number of citations in Web of Science and Scopus, page count, annex page count, reference count, code availability, Univariate or Multivariate forecasting, Iterative vs direct multi-step forecasting, datasets used, dataset links, dataset descriptions, forecasting horizons, metrics, loss functions, positional encoding, preprocessing steps, training and tuning

details, attention mechanisms, critique of prior work, contributions, future research directions, and additional notes. Empty cells «were filled with "-".

- In the excel sheet “agglomeration” – Agglomeration of the results of different papers by metrics and forecasting horizons.

Primary outcomes: To identify and analyse the emerging trends, innovations, and methodological advancements in transformer models tailored for time series forecasting; To pinpoint existing gaps, challenges, and limitations in the scientific knowledge and of state-of-the-art transformer models; To propose future research directions and potential technological advancements.

Secondary outcomes: Compile of the most commonly used datasets and evaluation metrics in the literature, providing a standardised reference for future research. Analyse publication trends, including the number of papers published over time, the most common publication venues, and the geographical distribution of research efforts. Asses the existence of bias. To determine the extent of the impact of the innovations. Providing a timeline outlining the summarized key changes proposed in each paper as well as their motivation.

To assess the risk of **bias**, the funding sources for each paper were reviewed, noting whether they were disclosed and identifying the nature (public, private, or mixed) and origin (country) of the funding. Next, the stated biases were extracted from the papers, specifically looking for declarations of potential conflicts of interest by the authors. To assess publication bias, the performance of the transformers was analysed, comparing the results of the papers that used MSE as the metric in chronological order, to see if there was a tendency to only improving models. Additionally, the use of datasets was evaluated, noting the diversity, accessibility, and geographic origin of the datasets used in the studies.

ChatGPT was used as an editor to detect typos and improve the English used, ensuring clarity and suitability for a thesis.

4. PAPERS' ANALYSIS

This section consists of studying the 99 Papers' characteristics.

4.1. DATES, DATABASES AND CONFERENCES

The date the papers are made available, and the dates they are published are often different. Sometimes they are made public before publishing through Arxiv or Openreview, which are pre-prints before peer-review. Sometimes, they are only made public after being presented at conferences. The dates are the same if they are published as part of a book. The dates of publication post-peer-review were used for the plots.

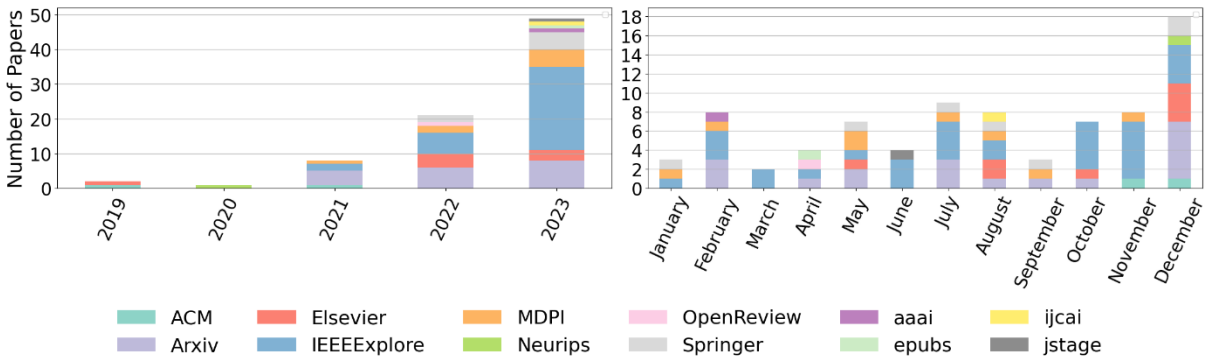

Figure 4.1 – Number of Papers Published per Year and per Month by Publisher - Excluding 2024

The yearly distribution shows that the number of papers per year has increased exponentially. The most common source is IEEEExplore, with Arxiv, Elsevier, MDPI and Springer also having a significant number of papers. From the monthly distribution, it is clear that some websites have only accepted one or two papers, such as Jstage, epubs, NeurIPS, AAAI, and ACM. It is also shown that some websites publish papers throughout the year, such as IEEEExplore, Arxiv, MDPI, and Springer. December is also the month with the most papers published and the months with the least publications are January, March, and September.

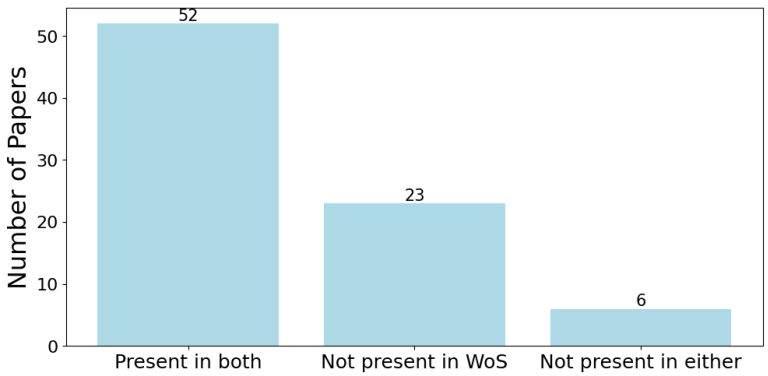

Figure 4.2 – Presence of papers in Web of Science and Scopus

Although some papers are only present in Scopus, none are only on Web of Science.

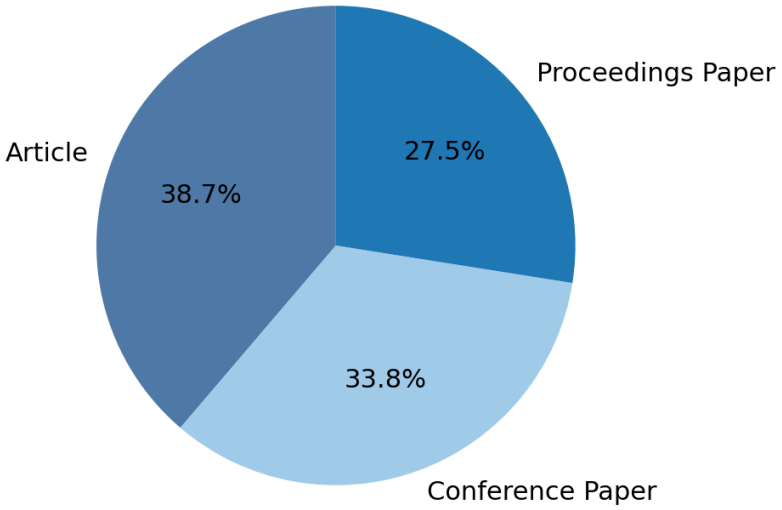

Figure 4.3 – Type of papers

Articles, which are peer-reviewed pieces published in journals, constitute the most significant portion at 38.7%. Conference papers, which are presented at conferences, make up 33.8%. Proceedings papers, which are included in the official record of the conference, account for 27.5%. This indicates a well-rounded representation of academic research dissemination methods.

4.2. CITATIONS

There are 10 papers with no citations on either Web of Science nor Scopus, and another 13 papers that aren't present in Web of Science and have 0 citations on Scopus. This is to be expected due to the recency of a lot of the papers.

Figure 4.1 shows the number of papers per number of binned citations.

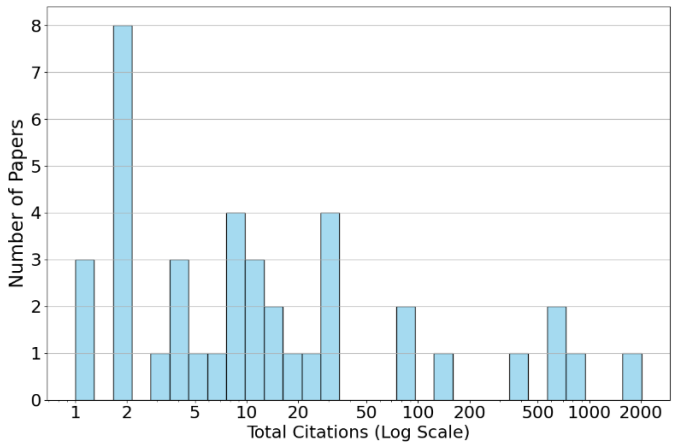

Figure 4.4 - Distribution of Total Citation Counts

It shows that there a small number of papers are cited a lot of times, while the rest have fewer citations.

4.3. Code Availability and Peer Review

Thirty five of the papers have the code available and the other 67 do not - the presence of code allows for independent verification of results, fostering transparency and facilitating further research advancements.

Of the papers 87 do not have a public peer review, and 12 do. The peer review being public allows the academic community to scrutinize the critique and validation experts offered before publication. It also directs future papers on the rigour of the reviews and what to avoid.

4.4. Datasets

In order to study the datasets used, the number of times each was utilized was counted, and the ones that were only used in one paper were aggregated into a single count called "unique datasets" – this is shown in Figure 4.5.

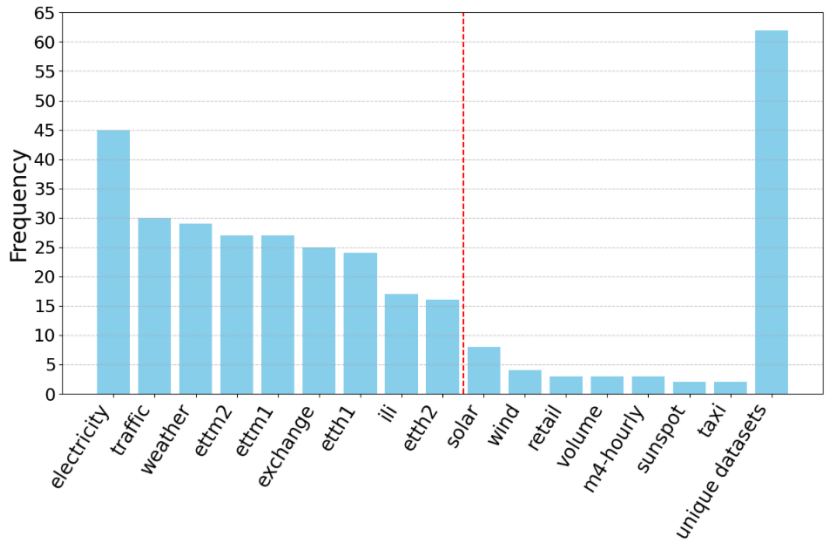

Figure 4.5 - Frequency of Datasets, grouping Unique Datasets

Most of the papers use all or a subset of the nine datasets to the left of the red line.

Of the 62 unique datasets 14 came from papers that used a transformer for a very specific purpose, with the datasets not being available, but some of them explain how the data was retrieved. 19 papers used a combination of unique datasets, usually in a specific area, and

also usually not provided a link to the data. And finally, 29 unique datasets were used after the authors benchmarked the results of the proposed transformed against previous papers.

Table 4.1 - Datasets characteristics, L and # Col. symbolize total time elapsed and number of Columns, Respectively

Datasets	Start Date	End Date	L	Periodicity	# Points	# Col.	URL
Electricity/ ECL	01/07/2016	17/11/2017	2y	1h	26304	321	link
Weather / WTH	01/01/2020	01/01/2021	1y	10m	52696	21	link
Traffic / Pems	01/07/2016	02/07/2018	2y	1h	17544	862	link
ETTh1 / ETTh2	01/07/2016	26/06/2018	2y	1h	17420	7	link
ETTm1/ ETTm2	01/07/2016	26/06/2018	2y	15m	69680	7	link
Exchange-Rate/ Exchange	1990	2016	26y	1d	7587	7	link
ILI	2021	2022	1y	1w	966	8	link
Solar	01/01/2016	31/12/2016	1y	10m	7009	137	link
Wind	1986	2015	29y	1h	354 040	28	link

Take note that the same dataset may be called differently in different papers. The following pairs refer to the same datasets: ECL and Electricity, Exchange-rate and Exchange, Traffic and Pems, Weather and WTH.

Descriptions of the eleven most used datasets are:

- **Electricity** - Electricity Load Diagrams Dataset (ECL), containing the electricity consumption of 321 customers;

- **WTH** - contains hourly data collected at nearly 1600 locations in the United States;
- **Traffic** - the road occupancy rate measured by 862 sensors collected on the San Francisco Bay Area highways;
- **ETT** (Electricity Transformer Temperature) - a crucial indicator in the electric power long-term deployment. Data was collected from two separate counties in China. To explore the granularity on the LSTF problem, separate datasets were created for the two counties as ETTh1 and ETTh2 for 1-hour level - and ETTm1 and ETTm2 for 15-minute-level;
- **Exchange** - is the collection of the daily exchange rates of eight foreign countries including Australia, British, Canada, Switzerland, China, Japan, New Zealand and Singapore;
- **ILI** - patients data from the U.S. Centers for Disease Control and Prevention between 2002 to 2021, containing the ratio of Influenza patients and the total number of patients;
- **Solar** - records the solar power production of 137 PV plants;
- **Wind** - contains hourly estimates of an area's energy potential.

4.5. METRICS

The following bar plot, Figure 4.6, show the number of times each metric was used in the papers and reveals distinct patterns in model performance evaluation.

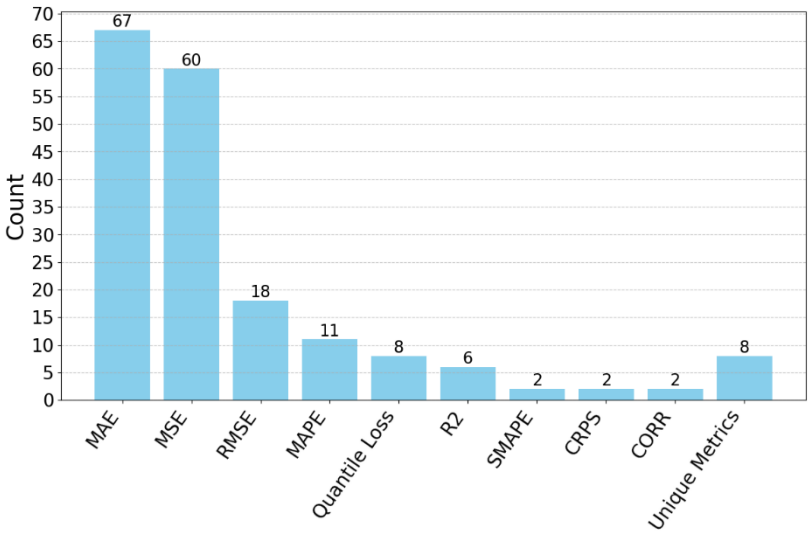

Figure 4.6 – Bar plot of the metric used

MAE and MSE are the most commonly used metrics, appearing in 67 and 60 studies respectively, indicating their widespread acceptance for assessing prediction accuracy. RMSE, cited in 18 studies, is also frequently used, highlighting its utility in penalizing larger errors. MAPE and Quantile Loss, used in 11 and 8 studies respectively, demonstrate their relevance in contexts where relative error and quantile-specific predictions are important. Less frequently used metrics such as R-squared, CRPS, CORR, and SMAPE appear in 6, 3, 2,

and 2 studies respectively, suggesting a more specialized application for these measures. This distribution shows a clear preference for error magnitude metrics, with occasional use of other metrics tailored to specific forecasting needs.

The Mean Absolute Error (MAE) is a measure of the average magnitude of the errors between predicted values ($\hat{y}_i$) and actual values (y_i), providing a straightforward interpretation of the average error magnitude in the same units as the data, without considering the direction of the errors. It is calculated as:

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

The Mean Squared Error (MSE) is a measure of the average squared differences between predicted values and actual values, it penalizes larger errors more than smaller ones due to the squaring of the error term, which can be useful for emphasizing significant deviations. It is calculated as:

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

The Root Mean Squared Error (RMSE) is the square root of the MSE, providing an error metric that is in the same units as the original data, penalizing larger errors more heavily and gives a measure of the spread of the errors. It is calculated as:

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}$$

The Mean Absolute Percentage Error (MAPE) measures the average absolute percentage difference between predicted and actual values, being useful for understanding the error in relative terms, making it easier to compare errors across different scales. It is calculated as:

$$MAPE = \frac{100\%}{n}\sum_{i=1}^{n}\left|\frac{y_i - \hat{y}_i}{y_i}\right|$$

Quantile Loss is used to estimate the conditional quantiles of a response variable. For a given quantile q, being particularly useful in scenarios where the goal is to predict a specific quantile rather than the mean. For a given quantile q, where $0 < q < 1$, the quantile loss is calculated as:

$$Quantile\ loss = \frac{2}{n}\sum_{i=1}^{n}(q - 1\ (y_i < \hat{y}_i))(y_i - \hat{y}_i)$$

where $1\ (y_i < \hat{y}_i)$ is the indicator function.

The R-squared (R^2) metric measures the proportion of the variance in the dependent variable that is predictable from the independent variables. It is calculated as:

$$R^2 = 1 - \frac{\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}{\sum_{i=1}^{n}(y_i - \bar{y}_i)^2}$$

where $\bar{y}_i$ is the mean of the actual values. R^2 ranges from 0 to 1, with 1 indicating perfect prediction and 0 indicating no predictive power.

The Continuous Ranked Probability Score (CRPS) measures the accuracy of probabilistic forecasts. It is defined as:

$$CRPS(F, y) = \int_{-\infty}^{\infty} (F(x) - 1(y \leq x))^2 dx$$

where F is the cumulative distribution function (CDF) of the forecasts and y is the actual value. CRPS is useful for evaluating the quality of probabilistic predictions, accounting for both the predicted distribution and the observed outcome.

4.6. BIAS

After retrieving the funding from the papers, it was found that a considerate number of papers did not state any funding. Among the funded papers, 62 received grants from governmental or public entities, two were funded solely by private entities, and 5 had a mix of both private and public funding.

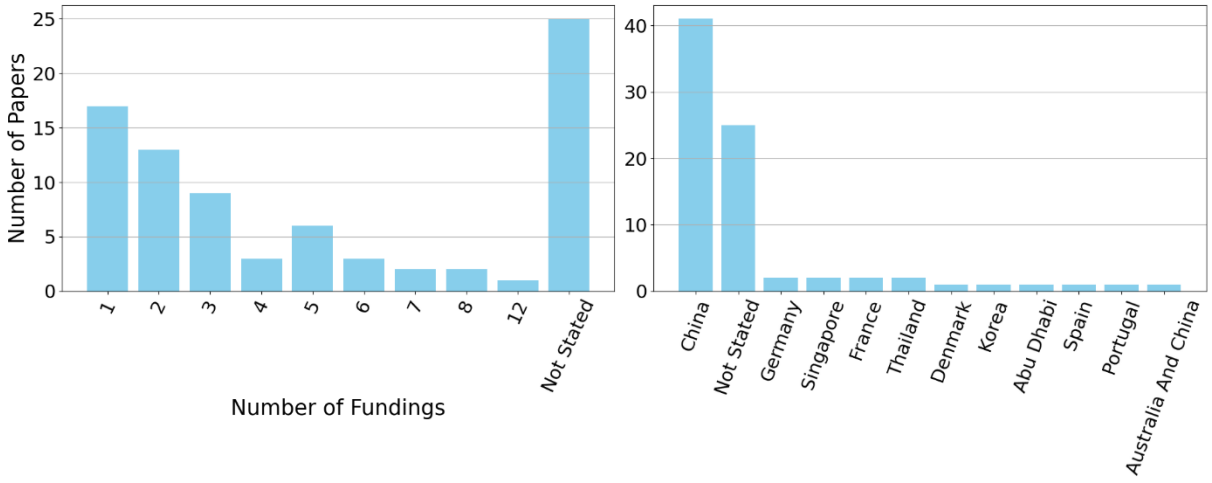

Figure 4.7 - Number of Papers by Grants Count; Number of Papers Funded per Country

The many papers that do not state their funding show a lack of transparency that can obscure potential conflicts of interest. Most the papers state they receive between 1 and 3 grants. The low number of grants might suggest that many research projects are likely constrained by limited financial resources, depending on the values of the grants, as paid GPUs are necessary to run the code.

More than half the papers are funded by entities from China, potentially leading to regional/ geographical bias as many papers use private datasets from China, and the research priorities and perspectives could be too similar.

Regarding stated bias, 20 of the papers state to not have any, and only one paper (J. Zhu et al., 2023) states that part of the authors might have a potential conflict of interest as they are employed by SPIC.

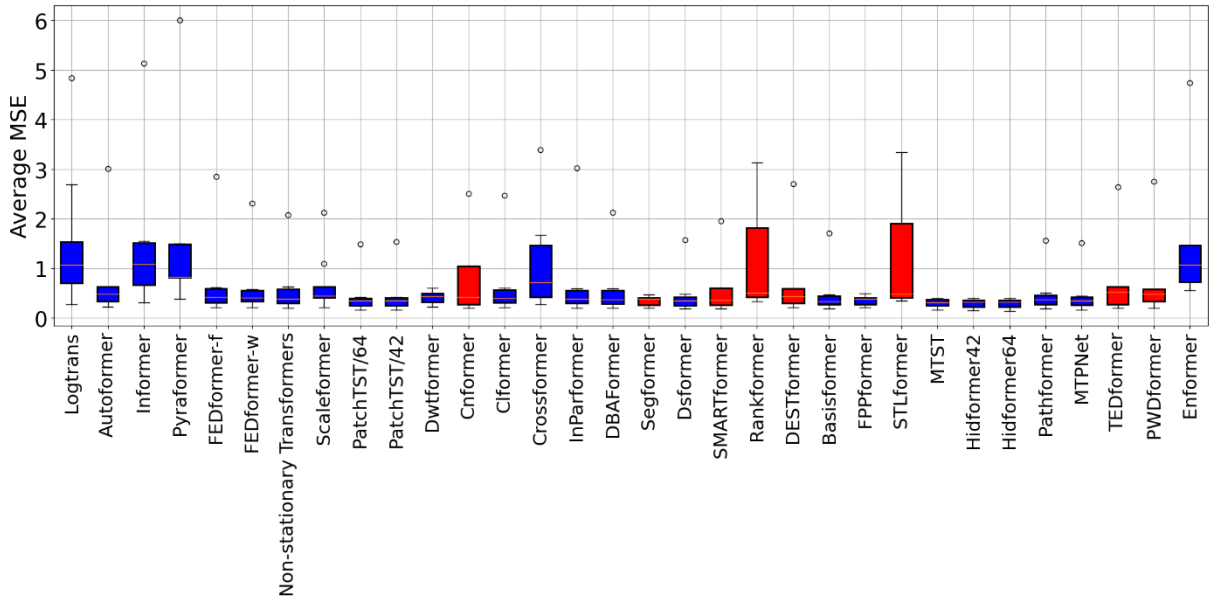

Figure 4.8 – Average MSE per transformer, ordered by publication date

The boxplots in blue mean that the papers use all of the 9 most common datasets, while in red they only use a subset of them. It is visible that there are transformers that achieve worse results than previous models, suggesting there is a relatively low publication bias, which could mean that the main goal of some papers is to explore how well different strategies work, and not necessarily on only improving accuracy.

However, the presence of bias cannot be entirely dismissed. Of the 100 papers analysed, there are 62 unique datasets, many of which are not publicly available. This lack of accessibility and the weak comparisons made - often using only a single dataset or comparing to a limited number of transformers - suggest that some degree of bias may exist as the performance of those models isn't robustly assessed. Furthermore, some papers employ metrics that are not widely used, complicating comparisons with other studies in the literature.

Additionally, the results of the transformers have some inherit randomness, as seen in **Table 0.1 – Evolution of Informer metrics throughout different papers**, but only three papers show the standard deviation of the results. This could mean that there is some skewness in the results as the results shown are presumably the best results obtained by the authors.

4.7. FORECASTING TRENDS

This section focuses on finding forecasting trends by plotting the evolution of the number of different types of forecasting strategies.

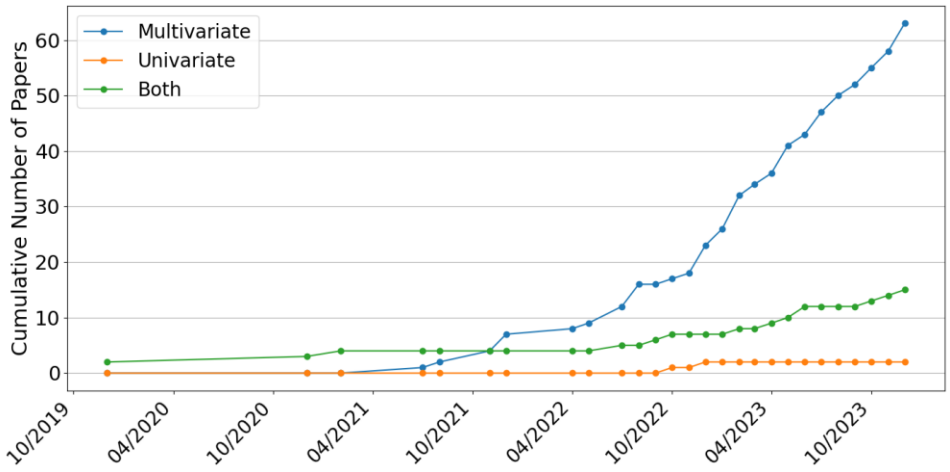

Figure 4.9 – Cumulative number of papers by forecasting type

Initially, research on transformers in time series forecasting explored both multivariate and univariate conditions. Multivariate forecasting involves predicting multiple time series simultaneously, capturing their interdependencies, while univariate forecasting deals with predicting a single time series based solely on its past values.

Over time, however, there has been a noticeable shift toward primarily focusing on multivariate scenarios. Despite some papers claiming that the proposed transformers are suitable for both settings, demonstrations are predominantly confined to multivariate data. This trend highlights a growing interest in leveraging the complex interdependencies within multivariate datasets, which often encapsulate richer information and more realistic conditions for forecasting.

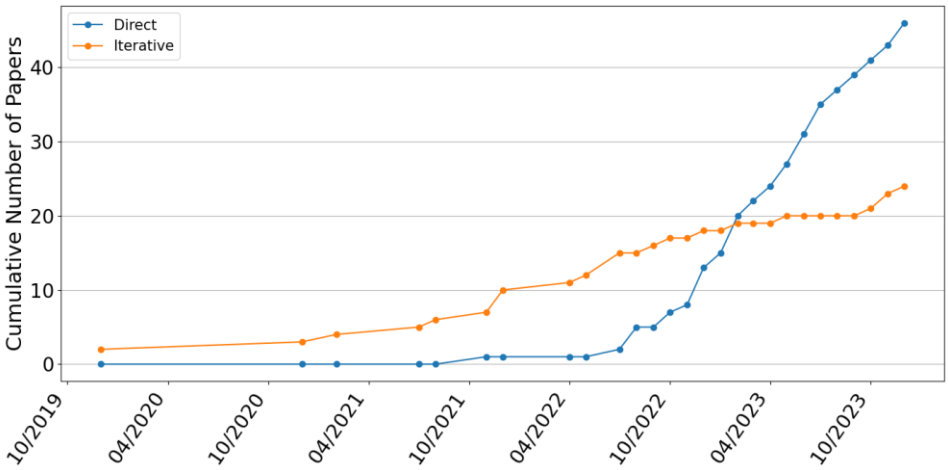

Figure 4.10 – Cumulative number of papers by forecasting type

In the early stages, transformers in time series forecasting primarily employed an iterative approach, where predictions were made one step at a time, using the output of the previous step as input for the next prediction. However, by the end of 2022, there was a significant shift towards using direct multi-step forecasting, wherein models predict multiple steps ahead in a single forward pass. This change was driven by the need to address error accumulation inherent in the iterative approach.

4.8. PREPROCESSING, SPLITS, AND TUNING

The systematic literature review reveals consistent trends in **preprocessing techniques** across various datasets, ensuring data quality and model compatibility. Normalization, particularly z-score standardization, is prominently employed to achieve zero mean and unit variance. This method is applied across diverse datasets, including electricity, traffic, retail, and volatility data, stabilizing the data and enhancing model performance. Log transformation is another common practice, particularly for datasets with skewed distributions such as retail sales and volatility, helping to stabilize variance and normalize data distribution. These preprocessing steps are critical for maintaining model stability and mitigating issues arising from differing data scales.

Handling **missing values** is a significant aspect of preprocessing, with various methods employed to address gaps in the data. The retail dataset, for instance, resamples data at regular daily intervals and imputes missing days using the last available observation. Similarly, Bitcoin data undergoes rigorous data cleaning to remove missing values and inconsistent data formats. Other techniques include K-Nearest Neighbour (KNN) data imputation and mean filling algorithms. These methods ensure that the datasets are complete and reliable, which is crucial for effective model training and accurate predictions.

Feature transformation and extraction are also prevalent preprocessing steps, enhancing the model's ability to capture patterns and trends in the data. Datasets such as electricity and traffic include day-of-week, hour-of-day, and time index as real-valued inputs, treating

categorical variables like entity identifiers separately. Advanced techniques like Fast Fourier Transform (FFT) and seasonal-trend decomposition are used to extract informative features and smooth the data. Additionally, segmentation into patches, data chunking, and sampling-type coding are applied to stabilize time series data and facilitate effective model training. These preprocessing techniques collectively contribute to the robustness and accuracy of the models across various datasets. Decomposition techniques integrated in the transformer architecture are studied in section **5.1.4 Decomposition**.

The reviewed papers consistently **split** datasets into training, validation, and testing sets, although the specific ratios vary. A common approach is to use a 70/15/15 or 70/10/20 split, ensuring that a substantial portion of the data is allocated for training while reserving smaller parts for validation and testing to evaluate model performance. For instance, the ETT dataset frequently uses a 6:2:2 ratio for training, validation, and testing, whereas other datasets like traffic and electricity often employ a 7:1:2 split. Some studies also follow a time-based splitting method, such as dividing data into specific months for training, validation, and testing, exemplified by the ECL dataset's 15/3/4 month split and the weather dataset's 28/10/10 month configuration. These consistent yet flexible strategies across different datasets highlight the emphasis on robust model validation and testing to ensure generalizability and accuracy.

The **tuning** of parameters in the reviewed papers involves a combination of random search, grid search, and automated hyperparameter optimization techniques. Grid search and random search are commonly used to explore various combinations of hyperparameters such as learning rate, dropout rate, batch size, number of layers, and attention heads. For instance, some studies conduct grid searches over specific learning rates, α-values, encoder and decoder blocks, while others use random search to identify the best settings across multiple iterations, such as 200 for the Exchange Rate and Electricity datasets.

More advanced methods, like those implemented using the Optuna framework, involve automated hyperparameter optimization. Optuna uses efficient sampling and pruning methods, like the tree-structured Parzen estimator and the asynchronous successive halving algorithm, to iteratively refine the search space based on historical performance data. This approach significantly reduces the manual effort and improves the efficiency of finding optimal hyperparameter values. These tuning practices ensure robust and well-optimized models for various forecasting tasks.

5. EVOLUTION AND DISCUSSION

Transformers have significantly evolved to address the unique challenges of time series forecasting, with various modifications enhancing their capability to capture temporal dynamics and improve predictive accuracy. This section delves into key areas of these advancements, focusing on each different component and aspect of the transformers.

The organization of this section is shown in Figure 5.1. It divides the transformer into “Input Representation”, “Modelling”, “Model Optimization”, “Types of Learning” and “Repeated Transformers”. The first three are further sub-divided into their composing elements, and each of these subsections groups the mechanisms introduced by their similarity.

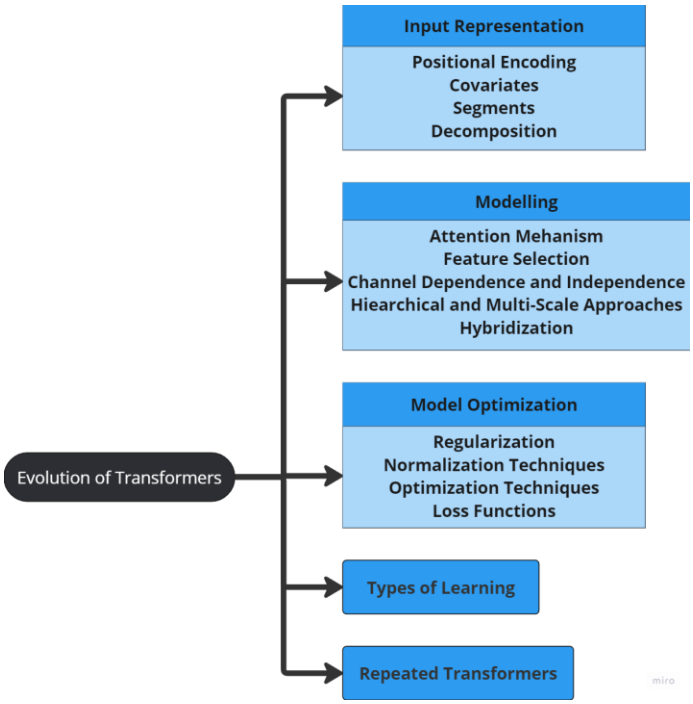

Figure 5.1 – Organization of the “Evolution and Discussion” Section

This framework was chosen to better showcase how each aspect of the transformer model has evolved, making it easier to find patterns, and allowing the reader to focus on one aspect at a time. There is some overlap, or repetition, of some innovations in the different sections as some fall under more than one category.

In order to evaluate the Certainty of Evidence and impact for each outcome, the ablation studies present in the papers, where the authors remove one of the introduced components one by one to see how each affects the performance of the model, were analysed. The last paragraph of each outcome is a study of the certainty of evidence by giving an overall summary of the relevant ablation studies.

The summary of the innovations introduced by each paper is chronologically ordered in **Appendix C (Table of Summary of Changes).**The evolution of the complexity equations and the different approaches of each paper are in **Appendix A (Complexity Equations)**.

5.1. Input Representation

A. Zeng et al. (2022) critically examine the effectiveness of Transformer-based models for long-term time series forecasting (LTSF). Despite their popularity, the authors argue that Transformers suffer from temporal information loss due to their permutation-invariant self-attention mechanism. They introduce a simple one-layer linear model, LTSF-Linear, for comparison and demonstrate through extensive experiments on nine real-world datasets that this model consistently outperforms Transformer-based solutions. The study reveals that the temporal modelling capabilities of Transformers until then were overstated for current benchmarks, suggesting the need for new research directions and a re-evaluation of Transformer-based approaches for time series analysis tasks.

Following this study, subsequent research has increasingly focused on enhancing input representation to better preserve the order of inputs and more effectively capture context and temporal dependencies. This sub-section covers various methods to achieve this, as illustrated in Figure 5.2. These methods include improving positional encoding, utilizing covariates, segmenting the data, and decomposing the data.

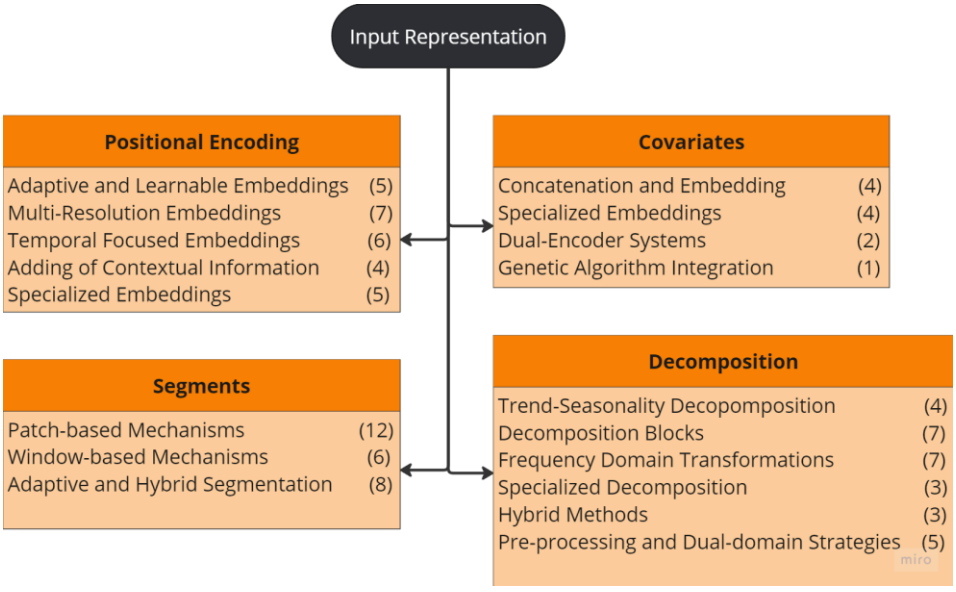

Figure 5.2 – Division and Sub-division of the Input Representation, Showing the Number of Models that fall under each Sub-division

5.1.1. Positional Embedding

Traditional Transformers use sinusoidal positional encodings to retain sequence order, as self-attention mechanisms are inherently order-agnostic. Various studies have modified these methods to enhance performance in time series forecasting by capturing complex patterns and addressing unique temporal challenges.

5.1.1.1. Adaptive and Learnable Embeddings

Some models integrate adaptive and learnable embeddings, allowing the model to dynamically adjust to different datasets, reducing manual tuning and optimizing temporal feature representation during training. Zhong et al. (2023), in **NTformer**, utilize learnable timestamp encoding to effectively handle the temporal features. This method involves converting temporal features - seconds, minutes, and hours - into dense vectors through an embedding layer that is learned during training, improving sequence understanding and prediction based on historical data. Y. Chen et al. (2023), in **JTFT**, use a learnable position embedding that is added to the joint time-frequency domain representation, ensuring the model captures the sequential order of the data in the combined Time Domain and Frequency Domain space.

Yang & Lu (2023), in **Foreformer**, introduce Adaptive Positional Encoding, combining linear $\alpha it + i\beta$ and sinusoidal $\sin(\alpha + i\beta)$ components to capture non-periodic and periodic patterns, respectively. The learnable parameters αi and βi allow adaptation to specific temporal dynamics. Similarly, Ni et al. (2024), in **Basisformer**, introduce learnable and interpretable basis vectors through adaptive self-supervised learning. This approach treats the historical and future sections of a time series as two distinct views, using contrastive learning to establish and refine these basis vectors. The learned bases replace the traditional positional encoding mechanism, embedding both the position and intrinsic features of the time series within the model's architecture

Further extending these ideas, Z. Zhang et al. (2024), in **SageFormer**, add learnable tokens to the input sequence of each series. These tokens capture overarching patterns and dependencies across different series. During the encoding process, they interact with the series data through iterative message passing, involving intra-series interactions via standard transformer mechanisms and inter-series interactions via graph neural networks (GNNs).

5.1.1.2. Multi-Resolution Embeddings

Incorporating multi-resolution embeddings in time series forecasting enhances the ability to capture patterns at different temporal scales. Y. Cao & Zhao (2023), in **DWTformer**, use 2D-FeaturesBlock that enhances the input representation of time series data by leveraging convolutional neural networks. After the wavelet decomposition transforms the 1D time series into 2D tensors, the 2D-FeaturesBlock applies an Inception-like structure with multi-scale convolutional filters to capture both local and global patterns. This multi-scale

convolutional approach effectively extracts complex intra-period and inter-period variations. Similarly, G. Tong et al. (2024), in **RSMformer**, alter the input representation by using a multiscale approach. It employs up-and-down sampling techniques to transform the input data into multiple resolutions, capturing temporal dependencies at different granularities. This enables the model to effectively manage long-range dependencies and combine information from various scales.

In another approach, X. Wang, Liu, Du, et al. (2023), in **CLformer**, alter the input representation by first applying dilated causal convolutions to capture multi-scale temporal patterns, followed by sequence decomposition to separate the time series into seasonal and trend components. This preprocessing refines the input data, emphasizing regular and predictable patterns. The input representation is further refined by segmenting the time series and applying using the locally grouped autocorrelation (LGAC) mechanism that calculates autocorrelation within these segments, enhancing the focus on local temporal dynamics. Adding to these methodologies, Y. Li, Lu, et al. (2023), in **Conformer**, utilize an advanced input representation. It employs Fast Fourier Transform (FFT) to compute correlations among variables, emphasizing relevant features with a softmax function. Additionally, the model represents the data at various temporal resolutions, embedding these multi-scale representations into a latent space. These embeddings are concatenated and weighted to form a comprehensive input that captures intricate temporal patterns.

Y. Zhang, Wu, et al. (2024), in **MTPNet**, introduce a dimension-invariant (DI) embedding technique, which, utilizing a 1-layer CNN, projects multivariate time series data into a higher-dimensional space while preserving both spatial and temporal dimensions. The data is then decomposed into seasonal and trend components. The seasonal component is segmented into non-overlapping patches of varying sizes corresponding to different temporal scales. Each level of the hierarchical pyramid structure processes these patches, allowing the model to capture and learn from fine to coarse temporal dependencies. The trend component is handled separately using a linear layer.

Dai et al. (2023), in **PDF**, transform the input representation from a 1D time series into a richer 2D format. This involves decoupling the series into short-term and long-term components using the Fast Fourier Transform (FFT), then reshaping these components into 2D tensors. Rows in these tensors capture short-term variations, while columns capture long-term variations. These tensors are further divided into patches and slices to separately model long-term and short-term temporal patterns. The Dual Variations Modeling Block (DVMB) then processes these patches and slices to effectively capture and model both short-term and long-term variations.

Cen & Lim (2024), in **PatchTCN-TST**, employ TCN for embedding, using causal convolutions to preserve temporal order. The TCN block ensures that the embedding does not use future data, only focusing on past and present, effectively embedding the temporal information directly into the data representation. Additionally, the model employs a patching operation

that segments the input sequence into smaller sub-series or patches. This segmentation allows the model to focus on local dependencies within each patch, improving the granularity of feature extraction and reducing computational complexity.

5.1.1.3. Temporal Focused Embeddings

Focusing on temporal-specific embeddings significantly enhance the model's ability to capture and predict time series data by refining how positional information is represented. Y. Liu et al. (2024), in **iTransformer**, replace traditional positional encoding by restructuring the data to focus on variate tokens rather than temporal tokens. In this approach, each variable in the time series is treated as a separate token, and the attention mechanism is applied across these variate tokens to capture their interdependencies. This inversion of data dimensions inherently incorporates the order and relationship within each variate's time series through the structure of the data and the model's architecture, making explicit positional encoding unnecessary.

Similarly, Y. Li, Qi, et al. (2023), in **SMARTformer**, use Time-Independent Embedding (TIE) to decouple positional encoding from value embeddings. TIE uses three separate learnable projection matrices to represent minute/hour, weekday, and month, ensuring that positional information is added without distorting the actual data values.

Domínguez-Cid et al. (2023), in **TEFNEN**, replace the standard embedding layers of the vanilla transformer with fully connected single-layer neural networks. Additionally, Instead of using the standard positional encoding to provide information about the order of the sequence, TEFNEN incorporates a Recurrent Neural Network (RNN) layer within the encoder-decoder structure. This RNN layer captures the sequential nature and temporal dependencies of time-series data.

Y. Zhang, Ma, et al. (2024), in **MTST**, employ relative positional encoding (*RPE*) that captures the relative distances between tokens, focusing on their intervals rather than absolute positions. This method enhances the model's ability to recognize and predict temporal dependencies and periodic patterns in time-series data. By encoding the differences between token positions, MTST effectively learns relationships that occur at varying intervals.

In another approach, Z. Wang et al. (2024), in **PWDformer**, incorporate position weights, which enhance the model's sensitivity to the order of time series data. These weights assign varying importance to different positions within the input sequence.

Finally, Zeng et al. (2023), in **Seformer**, introduce Binary Position Encoding, with intrablock and interblock position encodings. The intrablock position encoding handles positions segments the sequence into smaller blocks and uses a simplified, discretized ODE approach to efficiently capture local positional dynamics, reducing computational complexity. Interblock position encoding then recursively manages the relationships between these

blocks to manage long-range dependencies and prevent issues like baseline drift in longer sequences.

5.1.1.4. Adding of Contextual Information

Shabani et al. (2023), in **Scaleformer**, alter the input embedding by incorporating three components: value, temporal, and positional embeddings. The value embedding includes an additional indicator for the source of observations (look-back, zero initialization, or previous predictions). The temporal embedding maps time stamps to the hidden dimension, adjusted by the current scale factor. The positional embedding is adapted to the different scales s_i. These changes enable the model to effectively integrate and process multi-scale information, improving forecasting accuracy. Building on this idea, Qu et al. (2024), in **Forwardformer**, change the input representation by incorporating a comprehensive embedding structure that includes value embeddings for daily load data, timestamp embeddings for temporal context (day, week, and month), and information embeddings for additional contextual factors such as holidays, temperature, and pressure. This enriched input representation allows the model to capture and utilize the intricate temporal and contextual dependencies in the load data.

Similarly, Drouin et al. (2022), in **TACTiS**, incorporate timestamps, enabling the model to manage irregular sampling intervals and maintain accurate temporal order, static and time-varying covariates, and Boolean masks for missing values in the input, enriching context for forecasting. Positional encodings reflect time-specific factors like seasonality, enhancing temporal dynamics understanding. Further extending this approach, Ashok et al. (2024), in **TACTiS-2**, introduce a dual-encoder architecture, processing primary time series data and covariates with positional encodings. One encoder handles marginal distributions, while the other handles copula distributions, specializing in different aspects of the input data.

5.1.1.5. Specialized Embeddings

C. Zhang et al. (2023), in **Causalformer**, modify the input representation by combining temporal and causal features. It uses a Time Encoder with ProbSparse Self-attention to capture temporal dependencies and a Causal Encoder to identify causal relationships among variables through a Granger causality graph. This causal information, embedded as a sparse adjacency matrix, is integrated into the input data, which is then processed by a generative multi-head decoder.

Jawed & Schmidt-Thieme (2022), in **GQFormer**, introduce unique time-series ID embedding that assigns each time series a distinct identifier, which is embedded into a high-dimensional space and integrated with the input data. This allows the model to learn and differentiate patterns specific to each time series, improving its accuracy across diverse datasets. Additionally, the implicit quantile level embedding samples quantile levels from a uniform distribution and embeds these into a high-dimensional space. These embeddings are

combined with the time series data to enable the model to generate forecasts for multiple quantiles simultaneously.

Bou et al. (2022), in **InTrans**, introduce incremental embedding that is designed to enhance computational efficiency by reusing computations for overlapping segments of input data. This approach applies to positional, temporal, and value embeddings. For positional and temporal embeddings, the model divides the data into overlapping and non-overlapping segments. It reuses the embeddings from the overlapping segments of previous computations and only recalculates embeddings for the new, non-overlapping segments. For value embeddings, which use Conv1d layers to capture local patterns, the convolutional computations for the overlapping parts are reused, excluding the last few points that interact with padding values, while new computations are performed for the non-overlapping parts. This incremental approach significantly reduces redundant calculations.

Thwal et al. (2023), in **Federated Transformer**, and Tevare & Revankar (2023), in **Stack Transformer**, employ Time2Vec embeddings, which offer a more nuanced representation by encoding both periodic and non-periodic patterns of time. This method enables the embedding of time as a feature into the model, providing a dual representation where one component handles linear time features and another handles periodic features through sinusoidal functions.

5.1.1.6. Discussion

The presented advancements reveal a clear trend toward enhancing traditional transformer models through sophisticated positional and temporal embedding techniques, multi-scale and multi-resolution representations, and the integration of additional contextual information. These approaches collectively aim to address the inherent challenges of capturing complex temporal dependencies and patterns within data. Adaptive and learnable embeddings allow models to dynamically adjust to specific temporal dynamics, while multi-scale methods ensure the capture of both local and global patterns. Additionally, incorporating comprehensive contextual information and specialized embeddings, such as unique identifiers and causal relationships, significantly enriches the input representation, enabling models to handle diverse and intricate datasets more effectively.

Regarding the certainty of evidence, several models, including JTFT, SMARTformer, PWDformer, CLformer, SageFormer, RSMformer, Conformer, MTST, MTPNet, and Seformer conducted ablation studies that overall strongly confirmed the positive impact of their innovative embeddings. The other models did not report specific ablation studies (or similar) on their embedding methods. However, given that embeddings are crucial for maintaining the order of input data in transformers, the good performance of all these models implicitly supports the effectiveness of their embedding strategies. Thus, even in the absence of specific ablation studies for some models, their overall success suggests that the embedding

techniques employed are fundamentally sound and contribute to improved time series forecasting performance.

5.1.2. Covariates

In time series forecasting, leveraging covariates — i.e. independent variables that can influence the outcome of a given statistical trial, but which are not of direct interest — can significantly enhance predictive performance. Covariates, which can be either static (unchanging over time – e.g. IDs; product categories) or dynamic (varying over time – e.g. day of the week), provide extra context that helps models understand and anticipate complex patterns.

5.1.2.1. Concatenation and Embedding

This approach embeds covariates into vectors and concatenates them with the main time series data, creating a single representation before feeding it into the model. For example, S. Wu et al. (2020), in **AST**, encode both time-independent and time-dependent covariates into vectors concatenated with target data. Similarly, Jawed & Schmidt-Thieme (2022), in **GQFormer**, embed social time covariates into high-dimensional space and concatenate them with the main time series input, processed through the encoder's sparse self-attention mechanism.

Additionally, Lin et al. (2021), in **SSDNet**, explicitly incorporate multi-dimensional covariates into the input layer alongside primary time series data, helping in directly estimating the parameters of the state space model during the encoding phase. Furthermore, Yang & Lu (2023), in **Foreformer**, introduce a Static Covariates Processing Module, embedding static covariates into higher-dimensional space, using ELU activation and CNNs in the Multi-Temporal Resolution Module to capture local temporal features, and integrating with dynamic time series data through a gating mechanism that selectively filters information, allowing the model to suppress irrelevant data and emphasize useful covariates.

5.1.2.2. Specialized Embeddings

This approach involves creating specialized embeddings for covariates, integrated into the model in a tailored manner. For instance, Lim et al. (2020), in **TFT**, integrate static features using GRNs that create multiple context vectors, which condition the variable selection networks, initialize hidden states in sequence-to-sequence layers, and enrich temporal features during processing. This ensures that static covariates dynamically influence the selection and processing of time-varying features. Similarly, S. Zhu et al. (2023), in **MR-Transformer**, integrate covariate vectors through specialized embeddings summed with positional and value embeddings.

In another example, Drouin et al. (2022), in **TACTiS**, handle static and time-varying covariates, using them as conditioning variables that are integrated through the input embedding process, which combines each time series observation with its corresponding covariates. In addition, Ye et al. (2023), in **PDTrans**, incorporate both static and dynamic covariates into its forecasting process, alongside the historical time series data.

5.1.2.3. Dual-Encoder Systems

Drouin et al. (2022), in **TACTiS-2**, embed covariates and process them through both marginal and copula distribution encoders, to learn variable distributions and capture inter-variable dependencies, respectively. Similarly, Ye et al. (2023), in **TDT**, process historical covariates along with time series data in the encoder, and use them in two decoders: one that predicts future mean values, and the other forecasts the standard deviation.

5.1.2.4. Genetic Algorithm Integration

Pan et al. (2023), in **Autoformer-GA**, integrate a Genetic Algorithm (GA) optimizes external variable weights through selection, crossover, and mutation, guided by a fitness function. The Elite Variable Voting Operator emphasizes the most influential variables, ensuring optimal solutions. The Archive Storage Operator maintains top-performing solutions, preserving high-quality individuals and enhancing model stability by reducing random fluctuations.

5.1.2.5. Discussion

Various approaches are being tried to effectively utilize different types of external information. Concatenation and embedding methods allow covariates to be embedded into vectors and combined with the primary time series data, providing a unified representation for the model to process. Specialized embeddings offer tailored integration of covariates through dedicated network components, ensuring that these additional variables dynamically influence the model's understanding and processing of the data. Dual-encoder systems utilize separate encoders to capture the distributions and dependencies of covariates, further refining the model's predictions. Additionally, the use of genetic algorithms in Autoformer-GA demonstrates a promising future direction for integrating optimization techniques, which can enhance the effectiveness of covariate integration.

Regarding the certainty of evidence, while there is direct evidence that show clear benefits from covariate integration, from ablation studies in Foreformer, and TFT. For others, the inferred improvements and superior comparative performance suggest covariates are beneficial, though further targeted studies could strengthen this evidence.

5.1.3. Segments

The use of segments in time series forecasting with transformers has proven to be a significant strategy in enhancing model performance. By dividing time series data into meaningful segments, models can better retain local semantic information, attend to longer historical sequences, and reduce computation and memory usage. This segmentation can take various forms, each tailored to address different aspects of temporal data complexity and dependencies.

5.1.3.1. Patch-based Mechanisms

These approaches involve dividing the time series data into smaller, fixed-size patches or tokens to manage local and global temporal information effectively. Cirstea et al. (2022), in **Triformer**, introduce a novel Patch Attention mechanism. By dividing input time series into patches represented by pseudo timestamps, it efficiently computes attention scores within each patch, balancing computational complexity and temporal relevance. Similarly, Similarly, Shen et al. (2023), in **FPPformer**, employ hierarchical patch-wise and element-wise attention frameworks. This method segments input sequences into patches, adapting positional embeddings dynamically, thereby enhancing the model's ability to forecast complex multivariate and multi-scale data.

PatchTST, as described by Nie et al. (2023), divides the time series into subseries-level patches, embedding each patch as distinct input tokens. This method reduces computational complexity while capturing both local and comprehensive semantic information. From there, **PatchTCN-TST** by Cen & Lim (2024), segments input sequences into patches before applying attention mechanisms. The segment sizes are determined by the patch length and stride, which are optimized based on validation performance to balance local temporal dependency capture and efficiency.

Yu et al. (2023), in **Dsformer**, segment time series into smaller, contiguous blocks in the Piecewise Sampling of the Double Sampling Block, capturing localized information and emphasizing unique data characteristics. Similarly, Y. Zhang, Ma, et al. (2024), in **MTST**, use Patch-level Tokenization that helps capture temporal patterns within and between patches, facilitating the learning of both short-term and long-term trends.

X. Wang, Liu, Du, et al. (2023), in **CLformer**, use the locally grouped autocorrelation (LGAC) mechanism to divide time series into equal-length segments, performing autocorrelation calculations within these segments. Similarly, Zeng et al. (2023), in **Segformer**, divide time series into smaller equal-length segments for segment-based attention mechanisms.

Dai et al. (2023), in **PDF**, use patches to capture long-term temporal dynamics by dividing the 2D tensors, created from the time series data using the Multi-periodic Decoupling Block (MDB). Patch sizes are guided by periods identified through Fast Fourier Transform (FFT) analysis, revealing intrinsic data periodicity. And, Y. Chen et al. (2023), in **JTFT**, similarly

segment time series into continuous patches of a specified length with a given stride, either overlapping or non-overlapping. Each channel of the multivariate series is independently divided into patches, which are then rearranged into a three-dimensional tensor comprising channels, patches, and patch length.

5.1.3.2. Window-based Mechanisms

Window-based segmentation works by continuously dividing time series data into smaller, sequential segments that can overlap, breaking down continuous data into manageable chunks and allowing the model to update its understanding of the data in real-time. Overlapping windows capture more detailed and continuous temporal dependencies, preserving smooth transitions. (S. Li et al., 2019), in **LogTrans**, employ a rolling window approach for model evaluation, segmenting data into overlapping windows. This method forecasts future values based on a moving segment, testing under varying conditions and ensuring robustness.

(Bou et al., 2022), in **InTrans**, emphasize overlapping data segments for incremental learning, reusing previously computed values for overlapping parts. This reduces computational complexity and enables faster, real-time predictions. The sliding window technique prepares input and output sequences, allowing continuous updates without recalculating embeddings for processed data points.

Y. Li, Lu, et al. (2023), in **Conformer**, use a sliding-window attention mechanism to focus on local context within a specified window, reducing computational complexity. Similarly, (Lee et al., 2024), in **TS-Fastformer**, use a sliding window transformation (SWT) to create smaller, non-overlapping sub-windows by selecting consecutive sequences of data of a specified length, enabling faster training while maintaining effective performance.

Y. Liu et al. (2022), in **SWLHT,** divide input time series into overlapping segments to manage information over different scales and durations. Each segment, representing a subset of the time series data, slides forward by a fixed step size, capturing a wide range of temporal dependencies and dynamics. Finally, J. Ma & Dan (2023), in **FFT-Informer**, divide the time-series data into segments using sliding windows, capturing segments without overlap with predefined window lengths and sliding distances (e.g., 100 for the SMC dataset and 50 for the ASCE dataset). This ensures non-overlapping blocks, allowing the model to process each segment independently.

5.1.3.3. Adaptive and Hybrid Segmentation

Adaptive and hybrid segmentation techniques dynamically adjust segment sizes and leverage multiple strategies to capture temporal dependencies and patterns effectively. Du et al. (2023), in **Preformer**, introduce Multi-Scale Segment-Correlation (MSSC), dividing time series into discrete segments and computing correlations between them for attention

mechanisms, preserving the segments' continuity and dynamics by using multiple segment sizes starting from L_0 and increasing exponentially.

Y. Li, Qi, et al. (2023), in **SMARTformer**, employ a Semi-Autoregressive (SAR) Decoder that processes the input data to non-overlapping segments. Each segment is processed iteratively using an autoregressive (AR) approach to capture fine-grained local details, and then these segments are refined with a non-autoregressive (NAR) approach to maintain global coherence. The length of the segments is adjusted based on the dataset's characteristics. Typically, smaller segment lengths are used in the initial stages to capture detailed local patterns, and larger segments are used in later stages to capture broader, long-term dependencies.

Y. Zhang, Wu, et al. (2024), in **MTPNet**, partition time series data into patches of varying sizes, focusing on different temporal scales with a dimension-invariant embedding technique to maintain original spatial and temporal dimensions while transforming data into a higher-dimensional space. Patch-size is fixed at each level of its hierarchical pyramid structure but vary between levels. S. Zhu et al. (2023), in **MR-Transformer**, use dynamic time warping (DTW), to split the time series into non-overlapping segments dynamically, capturing local patterns and maintaining long-term dependencies.

Zeng et al. (2023), in **Seformer**, segment long sequences into smaller blocks, using intrablock position encoding to capture local dynamics and interblock encoding to preserve global context. P. Chen et al. (2023), in **Pathformer**, segment time series into patch of sizes that are adaptively selected by a multi-scale router based on the temporal characteristics of the input data, capturing local details with intra-patch attention and global dependencies with inter-patch attention, with sizes selected by a multi-scale router.

Z. Wang et al. (2024), in **PWDformer**, adjust the size of the time aggregation window based on local patterns, through the Deformable-Local (DL) Aggregation Mechanism, ensuring optimal feature capture. Yang & Lu (2023), in **Foreformer**, use segments in the Multi-Temporal Resolution (MTR) Module by downsampling the input time series into smaller subsequences. These segments are processed with convolutional layers to capture local features. Interactive learning ensures information exchange between segments, and they are finally concatenated and realigned to maintain the original sequence order, integrating multi-scale temporal patterns for enhanced forecasting.

Finally, Z. Liu et al. (2024), in **Hidformer**, a segment-and-merge architecture where input data is divided into segments to capture local features and dynamics. After processing, these segments are merged together to integrate local insights into a cohesive global understanding. And, B. Li, Cui, et al. (2023), in **Difformer**, employ Patch Merging to combine the segments to form new inputs that reduce temporal resolution but increase feature dimensionality. Then, Dynamic Ranging helps the model dynamically determine the

boundaries of these segments or chunks, adjusting the size and number of the segments based on the inherent patterns observed in the data.

5.1.3.4. Discussion

Segmentation strategies in time series forecasting have evolved from basic rolling windows and simple patches to more adaptive and dynamic methods, demonstrating versatile techniques that enhance forecasting performance. Patch-based mechanisms effectively capture local patterns, reduce computational load, and manage long time series efficiently by treating segments as distinct units. Window-based segmentation ensures models can continuously evaluate and adapt to changing conditions, while adaptive segmentation allows for detailed analysis of temporal dynamics and dependencies. These methods underscore a shift towards more context-aware and granular data processing.

Research on this has reached a point where it indicates that patch size controls the ability of transformers to learn temporal patterns at different frequencies, with shorter patches being effective for localized, high-frequency patterns and longer patches needed for mining long-term seasonalities and trends. (Y. Zhang, Ma, et al., 2024)

Regarding the certainty of evidence, while some models lack dedicated ablation studies, the consistent findings across the other models provide strong evidence that segmentation strategies are promising for enhancing transformer-based time series forecasting.

5.1.4. Decomposition

Decomposition techniques are relevant for time series forecasting, improving model accuracy by breaking down data into manageable components. These methods address complex patterns, noise, and trends by isolating trend, seasonality, and residual components, allowing for precise modelling and prediction. This enhances model robustness and reliability, leveraging both time-domain and frequency-domain information.

5.1.4.1. Trend-Seasonality Decomposition

Various Transformers decompose time series into trend and seasonal components to improve forecasting accuracy. X. Zhang et al. (2022), in **TDformer**, use multiple average filters of different sizes, combined with adaptive weights, to extract trend patterns. Similarly, X. Wang, Liu, Du, et al. (2023), in **CLformer,** apply dilated causal convolutions in the Temporal Convolution Block (TCB) and then use sequence decomposition to isolate trends by subtracting a moving average, leaving the seasonal component as the remainder. Y. Huang & Wu (2023), In **ADAMS**, uses a moving average to extract long-term trends and

subtracts this from the original series to obtain the seasonal component, further stationarizing the series by transforming it into a Gaussian distribution.

Y. Zhang, Ma, et al. (2024), in **MTPNet**, and D. Cao et al. (2024), in **TEMPO**, also utilize moving averages for trend extraction. MTPNet subtracts the trend to get the seasonal component, while TEMPO further decomposes the series into trend, seasonality, and residuals using smoothing and averaging techniques. Thwal et al. (2023), in Federated Transformer, employ moving average smoothing for preprocessing, reducing noise and enhancing stationarity.

5.1.4.2. Decomposition Blocks

Other transformers separate the input data into trend and seasonal components using specialized decomposition blocks. H. Wu et al. (2021), **in Autoformer**, employ decomposition blocks to focus on seasonal patterns and refine trend predictions. In the encoder, these blocks facilitate the detection of seasonal dependencies, while in the decoder, they progressively refine trend predictions using the trend-cyclical components. Similarly, T. Zhou et al. (2022), in **FEDformer**, use Mixture of Expert Decomposition (MOEDecomp), using a set of average pooling filters with varying sizes, combined with Fourier and wavelet transformations in Frequency Enhanced Blocks (FEBs) to process decomposed components in the frequency domain.

Ban et al. (2024), in **MOEA**, enhance the Autoformer with MOEDecomp and Wavelet Soft Threshold Denoising (WSTD) to better extract complex trends and reduce noise. (Du et al., 2023), in **Preformer**, use decomposition blocks with a Multi-Scale Segment-Correlation (MSSC) mechanism in the encoder and recombination in the decoder. Ouyang et al. (2023a), in **Rankformer**, Multi-Level Decomposition (MLDecomp) with Multiple Moving Average (MMA) filters to capture trends at different scales, using trainable weights for refinement.

Dai et al. (2023), in **PDF**, incorporate a multi-periodic decoupling block (MDB) that models short-term and long-term variations using transformer encoder layers and convolutional layers. Lastly, X. Wang, Liu, Yang, et al. (2023), in **CNformer**, use a series decomposition block with averaging and differencing techniques, refining seasonal components with stacked dilated convolutional blocks (SDCBs) and using convolutional attention in the decoder for improved forecasting accuracy.

5.1.4.3. Frequency Domain Transformations

Several transformers utilize frequency domain transformations to enhance forecasting by capturing significant periodic components. Y. Wang, Zhu, & Kang (2023), in **DESTformer**, apply Fast Fourier Transform (FFT) to separate high-frequency and low-frequency elements, processing the seasonal component with a Multi-View Attention mechanism that considers amplitude and phase for capturing complex periodic changes. The trend component is managed with a Multi-Scale Attention mechanism using convolutional filters of various

lengths to capture trends at different scales. Similarly, J. Ma & Dan (2023), in **FFT-informer**, segment data, apply FFT to each segment, and extract frequency components to summarize trends and periodicities.

Y. Cao & Zhao (2023), in **DWTformer**, implement wavelet decomposition to create multi-resolution representations, separating high-frequency details and low-frequency trends. Convolution Transformer introduces a Frequency Enhanced Block that maps time-series data to the frequency domain using Discrete Fourier Transform (DFT), selects significant frequencies, refines them through a learnable kernel, and converts the data back to the time domain enriched with global features. P. Chen et al. (2023), in **Pathformer**, apply DFT to extract significant periodic components and reconstructing seasonality with Inverse DFT. The residual data is processed with average pooling to extract the trend, and these components are combined and fed into a multi-scale router to capture both local and global dependencies.

S. Huang et al. (2022), in **CEEMDAN**, use Complete Ensemble Empirical Mode Decomposition with Adaptive Noise (CEEMDAN) to decompose data into multiple intrinsic mode functions (IMFs), addressing mode mixing with adaptive noise and categorizing segments by complexity using Sample Entropy (SE). High-complexity segments are processed by a transformer model, while low-complexity segments are handled by a Back Propagation Neural Network (BPNN). Sasal et al. (2022), in **W-transformers,** utilize the Maximal Overlap Discrete Wavelet Transform (MODWT) to decompose time series into components at different scales, applicable to data of any length and producing coefficients representing various frequency bands and trends.

J. Chen et al. (2023), in **Segformer**, employ a multi-components decomposition approach using FFT to capture different frequency patterns, decomposing time series into an overall trend, odd frequency components, and even frequency components to effectively capture both local and global temporal patterns.

5.1.4.4. Specialized Decomposition

Some transformers perform Seasonal-Trend decomposition with specialized components. Fan et al. (2023), in **TEDformer**, do it using Loess (STL), and then applying Temporal Convolutional Networks (TCNs) for the trend and an autocorrelation mechanism along with STL for the seasonal components. The decoder processes these decomposed components using TCN and feature fusion strategies.

Similarly, H. Cao et al. (2023), in **InParformer**, employ the Evolutionary Seasonal-Trend Decomposition (EvoSTD) algorithm for dynamic binary decomposition, iteratively splitting time series into seasonal and trend components to refine the model's understanding of underlying patterns. Ouyang et al. (2023b), in **STLformer**, also use STL Decomposition with LOESS (Locally Estimated Scatterplot Smoothing) to separate data into seasonal and trend-cyclical components. Additionally, STLformer incorporates a Rank Correlation Block using

Spearman's Rank Correlation to capture nonlinear dependencies and relationships between ranked data points.

5.1.4.5. Hybrid Methods

Some transformers integrate other models for decomposition and preprocessing. Lin et al. (2021), in **SSDNet**, embed a fixed-form state space model (SSM) within the Transformer architecture to directly decompose time series data into trend and seasonality components. Similarly, M. Li et al. (2022), in **Umformer**, employ feature decomposition and dataset preprocessing using the Prophet algorithm, which systematically decomposes univariate time series data into trend, seasonality, holiday effects, and residual noise. The decomposed data is then normalized and scaled to improve the learning efficiency.

J. Tong et al. (2023), in **PDTrans**, enhance the model's predictive ability by decomposing time series into trend and seasonality components using convolution operations and multi-layer perceptrons (MLPs), respectively. This approach allows the model to separately model and predict distinct patterns.

5.1.4.6. Pre-processing and Dual-Domain Strategies

Some transformers focus on advanced preprocessing techniques to enhance model performance. Y. Liu et al. (2023), in **Non-stationary Transformer**, employ employ series stationarization, normalizing time series data to zero mean and unit variance before processing, and then de-normalizing the predictions to maintain interpretability in their original context. This approach helps manage the non-stationary nature of real-world data. Similarly, Zhong et al. (2023), in **NTformer**, include both non-temporal and temporal feature processing. Non-temporal processing removes irrelevant data, fills missing values, and normalizes numerical data, while temporal processing aligns time characteristics across different stages, reducing feature complexity and temporal misalignment.

J. Zhu et al. (2023), in **SL-transformer**, use the Savitzky–Golay (SG) filter to smooth wind speed data and the Local Outlier Factor (LOF) filter to remove outliers in solar power generation data, enhancing input quality. Yu et al. (2023), in **Dsformer**, employ a Double Sampling (DS) Block that combines downsampling to emphasize long-term patterns and piecewise sampling to retain detailed short-term patterns, creating feature vectors that capture both macro and micro-level information.

Y. Chen et al. (2023), in **JTFT**, use a joint time-frequency domain approach, transforming time series into both time-domain (TD) and frequency-domain (FD) components. They apply a customized Discrete Cosine Transform (CDCT) to obtain FD components, capturing essential periodic and trend elements while filtering out noise. These FD components are then combined with recent TD data points, allowing the model to leverage both temporal and frequency information.

5.1.4.7. Discussion

Decomposition techniques in time series forecasting are increasingly sophisticated, focusing on isolating trend, seasonality, and residual components to enhance model accuracy and robustness. Recent trends include integrating moving averages and filters for fundamental pattern extraction, using specialized decomposition blocks combined with frequency domain techniques for refined component separation, and applying advanced transformations like Fourier and wavelet for capturing complex periodic components. Dynamic and adaptive methods address intricate variations in data, while hybrid approaches incorporate diverse models and preprocessing algorithms for comprehensive decomposition. Enhanced preprocessing and dual-domain strategies emphasize meticulous data preparation, leveraging stationarization, noise reduction, and combined time-frequency domain analysis to manage the complexities of real-world data effectively.

Regarding the certainty of evidence, only 6 of the papers lack any kind of ablation study with the other strongly supporting the impact of decomposition, demonstrating their effectiveness in enhancing model performance. These studies consistently show that advanced decomposition methods, whether through specialized blocks, frequency transformations, or combined domain approaches, significantly improve predictive accuracy. The evidence underscores that these techniques refine models' abilities to capture and predict complex temporal patterns, validating their necessity and effectiveness in the field.

5.2. MODELING

Recent advancements in time series forecasting have heavily relied on Transformer-based models, primarily due to their powerful attention mechanisms. However, challenges such as computational inefficiency and the difficulty in capturing long-term dependencies persist. This section delves into various modelling strategies designed to address these challenges and others, exploring innovations in attention mechanisms, hierarchical structures, and hybrid approaches – as shown in Figure 5.3.

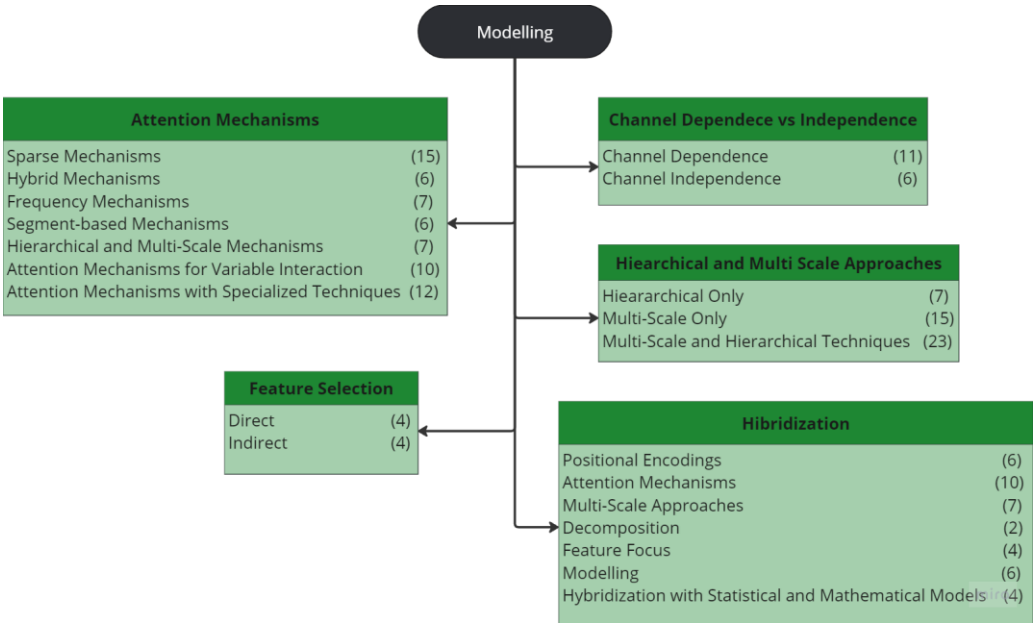

Figure 5.3 – Division and Sub-division of Modelling Components, Showing the Number of Models that fall under each Sub-division

5.2.1. Attention Mechanism

Time series forecasting has seen significant advancements with the introduction of Transformer models. The attention mechanism, a core component of Transformers, has been adapted and evolved to better handle the unique challenges presented by time series data, such as long-range dependencies, temporal dynamics, and computational efficiency.

5.2.1.1. Sparse Mechanisms

Sparse mechanisms have been increasingly utilized in time series forecasting models to improve efficiency and interpretability by focusing on the most relevant parts of the input sequence. H. Zhou et al. (2020), in **Informer**, ProbSparse self-attention uses a sparsity measurement based on the Kullback-Leibler (KL) divergence to identify the top-u most important queries, proportional to the logarithm of the sequence length. This mechanism is adapted in **Yformer** (Madhusudhanan et al., 2022) within a U-Net inspired multi-resolution framework, applied in both contracting (downscaling) and expanding (upscaling) paths, and used in skip connections between corresponding encoder and decoder layers, allowing the model to leverage multi-resolution feature maps, maintaining consistency in predictions. Similarly, X. Wang, Xia, & Deng (2023), in **MSRN-Informer**, employ the ProbSparse Self-attention from Informer, while **FFT–Informer** (J. Ma & Dan, 2023) combines it with a Hampel filter to identify and filters out outliers, enhancing robustness.

S. Wu et al. (2020), in **AST**, replace the standard softmax function with α-entmax, introducing sparsity into the attention weights, assigning zero weights to irrelevant time

steps, thus improving efficiency. Yang & Lu (2023), in **Foreformer**, implement explicit sparse multi-head attention, retaining only the top attention scores and normalizing them, ensuring focus on the most relevant parts of the sequence. Jawed & Schmidt-Thieme (2022), in **GQFormer**, employ sparse self-attention over a logarithmically spaced subset of positions, capturing long-range dependencies and introducing quantile transposed attention to capture interactions among different quantile forecasts.

N. Wang & Zhao (2023a), in **Enformer**, introduce Sparse Periodic Self-Attention, which identifies key queries with concentrated attention distributions using KL divergence, capturing periodic context through auto-correlation and softmax weighting. G. Tong et al. (2024), in **RSMformer**, introduce Residual Sparse Attention (RSA) that focuses on computing attention scores for only the most relevant queries, using sparse matrix operations and residual connections to enhance efficiency and stability. W. Li, Meng, et al. (2023), in **Bidformer**, introduce bidirectional sparse self-attention (Bis-Attention) and the shared-QK method to reduce computational complexity and redundancy. Guo et al. (2021), in **Stacked-Informer Network**, alters attention mechanisms by combining probabilistic sparse attention, that selectively computes attention scores for a subset of important tokens, and self-attention distilling, that halves the number of elements considered in successive layers, significantly improving efficiency. M. Li et al. (2022), in **Umformer**, enhance efficiency and accuracy with Shared Double-heads ProbSparse Attention (SDHPA), combining probabilistic sparsity and shared multi-head attention that uses two attention heads to capture a broader range of patterns while sharing the same values across different heads, ensuring that different heads attend to the same set of values but from different perspectives.

Additionally, graph-based mechanisms incorporate sparse attention to handle long sequence time-series forecasting efficiently. Su et al. (2021), in **AGCNT**, use ProbSparse Adaptive Graph Self-Attention, integrating ProbSparse self-attention to select a sparse subset of key elements, constructing an adaptive graph to represent dependencies, and applying graph convolution to aggregate information from neighbouring nodes. Similarly, Y. Wang et al. (2024), in **Graphformer**, replace traditional self-attention with graph self-attention, working with the Temporal Inertia Module to capture spatial dependencies and recent data trends, ensuring the model remains responsive to changes. C. Zhang et al. (2023), in **Causalformer**, employ ProbSparse Self-attention in the Time Encoder and adapts the Causal Encoder's attention to create a Granger causality graph, integrating causal relationships.

5.2.1.2. Hybrid Mechanisms

Convolutional mechanisms leverage convolutional layers to enhance the capture of local dependencies and improve computational efficiency. S. Li et al. (2019), in **Logtrans**, replace traditional dot-product self-attention with sparse causal convolutions to generate queries and keys, preserving temporal order and capturing local context like shape. Shen & Wang (2022), in **TCCT**, introduce CSPAttention, splitting the input into two parts: one processed by a lightweight 1×1 convolution to maintain overall structure, and the other by traditional self-

attention to capture global dependencies. The outputs are concatenated, reducing computational and memory costs significantly while improving gradient flow and capturing both local and global dependencies.

Y. Li et al. (2022), in **ACT**, incorporate a convolutional attention block with a causal convolutional layer before self-attention, effectively capturing local dependencies and addressing traditional self-attention's local context insensitivity. Similarly, X. Wang, Liu, Yang, et al. (2023), in **CNformer**, employ stacked dilated convolutional blocks (SDCBs) to refine seasonal components and capture local and long-term dependencies through convolutions in temporal and feature dimensions.

Z. Liu et al. (2024), in **Hidformer**, replace traditional multi-head attention with a combination of recurrence and linear attention. Recurrence captures temporal dependencies by processing data sequentially, leveraging information from previous time steps, while linear attention reduces computational complexity, scaling linearly with sequence length instead of quadratically. H. Cao et al. (2023), in **InParformer**, introduce Interactive Parallel Attention (InPar Attention), operating in both frequency and time domains. It includes query selection to reduce computational load, key-value pair compression through Interactive Partitioned Convolution (IPConv), and a dual attention structure processing time-aware and frequency-aware attention in parallel.

5.2.1.3. Frequency Mechanisms

H. Wu et al. (2021), in **Autoformer**, replace traditional self-attention with an Auto-Correlation mechanism, capturing period-based dependencies by aligning and aggregating similar sub-series from different periods, enhancing long-term forecasting efficiency. Similarly, Z. Wang, Chen, Yang, et al. (2023), in **HAFF**, refine this Auto-Correlation mechanism to better capture historical and nonlinear characteristics of electricity demand, recognizing long-range dependencies and complex temporal patterns, and considers external factors like working days and seasonal changes to dynamically adjust forecasts.

FEDformer, introduced by T. Zhou et al. (2022), uses Frequency Enhanced Attention (FEA) blocks that operate in the frequency domain, focusing on significant frequency components through Fourier and wavelet transforms, selectively emphasizing important frequencies before transforming the data back to the time domain, using the Inverse Fourier Transform. Similarly, **TDformer** by X. Zhang et al. (2022) employs Fourier attention to handle seasonal components, transforming data using the Fourier Transform, computing attention scores among frequency components, and normalizing these scores with softmax to emphasize significant periodic patterns before converting the data back to the time domain.

X. Wang, Liu, Du, et al. (2023), in **CLformer**, replace standard global self-attention with a locally grouped autocorrelation mechanism within equal-length segments, improving the capture of local temporal dynamics and multi-scale dependencies. (Y. Cao & Zhao, 2023), in **DWTformer**, incorporate auto-correlation to identify and emphasize significant time lags and

periodic patterns, rolling the time series to align similar phases and using Softmax-normalized auto-correlation values to focus on key periods, enhancing long-term dependencies capture.

Ouyang et al. (2023a), in **Rankformer**, replace Pearson correlation with Spearman's Rank Correlation (RankCorr) in self-attention mechanisms, capturing nonlinear dependencies by ranking input data and calculating autocorrelation on these ranks using FFT and the Wiener-Khinchin theorem, significantly reducing computational complexity and making the model more robust to outliers and better at identifying monotonic relationships.

5.2.1.4. Segment-based Mechanisms

Y. Li, Lu, et al. (2023), in **Conformer**, introduce a sliding-window attention mechanism that restricts each point to attend to a predefined number of nearby points, reducing computational complexity and efficiently capturing relevant temporal dependencies for long-term forecasting. Similarly, J. Chen et al. (2023), in **Segformer**, introduce SegAttention, which divides the input sequence into smaller segments and applies two-dimensional convolution operations to the query and key matrices within each segment, capturing fine-grained temporal patterns more effectively.

Cirstea et al. (2022), in **Triformer**, introduces a "Patch Attention" mechanism, breaking the input time series into smaller segments (patches), each associated with a pseudo timestamp representing the entire patch. This mechanism processes each patch by computing attention scores between the pseudo timestamp and the real timestamps within the patch, forming a triangular structure where the input size for each subsequent layer decreases, maintaining computational efficiency while effectively capturing relationships among different patches and long-term dependencies.

Shen et al. (2023), in **FPPformer**, introduce Diagonal-Masked (DM) Self-Attention, combining patch-wise and element-wise attention to capture local and global dependencies. This mechanism masks diagonal elements in the attention score matrix to prevent self-focus, reducing the impact of outliers. Y. Li, Qi, et al. (2023), in **SMARTformer**, introduce Window Attention (IWA), splitting attention into two branches: one for local self-attention within non-overlapping multi-scale windows and another for global attention across these windows, with the local branch focusing on detailed dependencies within windows and the global branch maintaining broader context across the sequence.

Z. Wang et al. (2024), in **PWDformer**, introduce the Deformable-Local (DL) Aggregation Mechanism, which enhances the attention mechanism by adaptively adjusting the size of the time aggregation window, allowing the transformer to focus more precisely on relevant local temporal patterns, improving its ability to capture both short-term and long-term dependencies.

5.2.1.5. Hierarchical and Multi-Scale Mechanisms

S. Liu et al. (2022), in **Pyraformer**, introduce the Pyramidal Attention Module (PAM), organizing data into a pyramidal graph with layers representing different resolutions, from fine to coarse. PAM is sparse and employs a dual attention mechanism operating both intra-scale, allowing nodes within the same layer to focus on local dependencies, and inter-scale, connecting nodes across layers for efficient data integration. Similarly, Du et al. (2023), in **Preformer**, use Multi-Scale Segment-Correlation (MSSC) to calculate segment-wise correlations, capturing local continuity and reducing complexity, with the decoder leveraging Predictive MSSC (PreMSSC).

Qu et al. (2024), in **Forwardformer**, introduce Multi-Scale Forward Self-Attention (MSFSA), capturing dependencies at multiple scales by integrating forward sliding window attention for local dependencies by focusing on a fixed number of immediate future tokens, forward dilated sliding window attention for medium-range dependencies by attending to non-contiguous future tokens, and forward global sliding window attention for long-range dependencies and significant but infrequent tokens like holidays. Y. Zhang, Ma, et al. (2024), in **MTST**, adapt the attention mechanism to use relative positional encoding and patch tokenization, performing self-attention on patch-level tokens in each branch to capture periodic and seasonal patterns across multiple resolutions.

P. Chen et al. (2023), in **Pathformer**, introduce a dual attention system operating on multi-scale divisions of time series data, using intra-patch attention for local dependencies within patches and inter-patch attention for global dependencies across patches. Similarly, Y. Wang, Zhu, & Kang (2023), in **DESTformer**, introduce Multi-View Attention (MVI-Attention) for capturing seasonal changes from multiple perspectives in the frequency domain and Multi-Scale Attention (MSC-Attention) for trend components, using convolutional filters to analyze sub-trends at various scales.

Zeng et al. (2022), in **Muformer**, introduce a multi-granularity attention head mechanism combined with multiple perceptual domain (MPD) processing. This approach processes input data into different perceptual domains, which are fed into various attention heads, each focusing on distinct information. Additionally, Muformer implements an attention head pruning mechanism using the Kullback-Leibler (KL) divergence measure to prune similar redundant information among the heads.

5.2.1.6. Attention Mechanisms for Variable Interaction

Lim et al. (2020), in **TFT**, introduce Interpretable Multi-Head Attention that enhances interpretability by sharing value vectors across all heads and using additive aggregation to create a unified attention matrix. This approach clarifies which parts of the input sequence are most influential, making temporal relationships and feature importance easier to understand. Similarly, Tang & Matteson (2021), in **ProTran**, integrates stochastic latent variables in its attention mechanism to model non-Markovian dependencies, enabling the

model to handle inherent noise and complex dynamics in time series data, resulting in more accurate probabilistic forecasts.

Yu et al. (2023), in **Dsformer**, introduce the Temporal Variable Attention (TVA) block, combining temporal attention for contextual relationships and variable attention for inter-variable dependencies within a parallel structure. In a different approach, Y. Liu et al. (2024), in **iTransformer**, focus on variate tokens instead of temporal tokens, calculating the importance of each variate through self-attention. This enhances the model's ability to learn complex inter-variable relationships and provides insights into variable influence, increasing interpretability.

T. Li, Liu, et al. (2023), in **MASTER**, use a flexible, data-driven approach to compute attention scores for stock correlations, capturing temporal details within individual stocks and dynamic correlations between stocks. J. Huang et al. (2023), in **DBAFormer**, introduce a double-branch attention mechanism with temporal and variable attention branches to capture dependencies across time steps and features, respectively, reducing computational redundancy with query-independent attention.

Y. Chen et al. (2023), in **JTFT**, utilize low-rank attention layers to efficiently capture cross-channel dependencies, reducing interaction dimensionality through a lightweight multi-head self-attention mechanism and integrating outputs with channel-independent representation. Similarly, S. Zhu et al. (2023), in **MR-Transformer**, enhance multivariate time series forecasting with Variable-Specific Attention, using temporal convolution filters for unique feature extraction and self-attention for variable interactions. The variable-specific features are then integrated with the Long Short-Term Attention (LSA), which captures both short-term and long-term temporal patterns through adaptive segmentation and global attention..

Zhong et al. (2023), in **NTformer**, employ an enhanced attention mechanism with a feature conversion mechanism, dividing features into domain and wave features, and using scaled dot-product attention to strengthen temporal correlations. Shen et al. (2023), in **TACTiS-2**, apply a two-stage optimization, starting with independent modelling of marginal distributions followed by optimizing copula parameters using Causal Attention Mechanism, effectively capturing dependencies between variables for a valid joint distribution of time series data.

5.2.1.7. Attention Mechanisms with Specialized Techniques

Y. Liu et al. (2023), in **Non-stationary Transformer**, introduce "De-stationary Attention" to reintroduce non-stationary characteristics initially removed during normalization. This mechanism recalibrates attention weights to reflect original data dynamics, enhancing predictions by leveraging inherent temporal dependencies. Similarly, Y. Huang & Wu (2023), in **ADAMS**, employs a De-Stationary Attention Module alongside an Auto-Correlation Mechanism to manage non-stationary time series data, normalizing input data into a stable

Gaussian distribution and reintroducing specific time dependencies to enhance long-term relationships.

Bou et al. (2022), in **InTrans**, introduce incremental self-attention to optimize Query, Key, and Value computations for overlapping segments, reducing redundant calculations. Y. Liu et al. (2022), in **SWLHT**, integrate a memory-efficient module for longer prediction horizons and complex data dependencies, enabling selective attention over extended periods. Tevare & Revankar (2023), in **Stack Transformer**, combine single and multi-headed attention with temporal attention, enhancing stock price prediction by considering adjacent observations in both forward and backward directions.

Ni et al. (2024), in **BasisFormer**, introduce a bidirectional cross-attention mechanism in the Coef Module to measure similarity between time series and learned basis vectors, enhancing relevant data associations. Thwal et al. (2023), in **Federated Transformer**, propose Federated Attention (FedAtt) to enhance federated learning by weighting updates based on similarity between local and global model parameters, improving performance and maintaining data privacy in decentralized environments.

Dai et al. (2023), in **PDF**, introduces Dual Variations Modeling Block (DVMB) and Variations Aggregation Block (VAB) to process 2D tensors created by period patching and frequency slicing, combining self-attention for long-term dependencies and convolutional layers for local patterns. **DDPformer** by Xie & He (2022) presents Deconstruction and Dot Product (DDP) attention, replacing matrix multiplication with element-wise dot products and non-linearity, simplifying computation and enhancing long-term dependency capture with channel-dependent attention.

Convolution Transformer by N. Wang & Zhao (2023b) introduces ES Attention, combining self-attention with an external attention mechanism in a dual-branch setup to capture dependencies within sequences and model relationships between different samples. B. Li, Cui, et al. (2023), in **Difformer**, use tanh to control addition/subtraction of scores and sigmoid functions to filter less relevant elements, highlighting significant changes and ensuring stability.

Lee et al. (2024), in **TS-Fastformer**, introduce the Past Attention Decoder to effectively capture both long-term and short-term dependencies in the data. It achieves this through a dual attention mechanism: the long-term multi-head attention focuses on broad, historical patterns, while short-term multi-head attention concentrates on recent data points. This approach allows the decoder to simultaneously consider distant and immediate past information, leading to more accurate and robust predictions.

5.2.1.8. Discussion

The evolution of attention mechanisms demonstrates a clear trend towards enhancing efficiency and interpretability while effectively capturing complex temporal dependencies.

Sparse mechanisms, such as ProbSparse self-attention, have been widely adopted to focus on the most relevant parts of the input sequence, improving computational efficiency and robustness. Hybrid approaches, integrating convolutional layers or combining recurrence with linear attention, further enhance local dependency capture and scalability. The introduction of frequency-based mechanisms, such as those using Fourier and wavelet transforms, allows models to emphasize significant periodic patterns, addressing the unique challenges of time series data. Segment-based and hierarchical mechanisms, which break data into manageable parts or multi-scale layers, improve the capture of both local and long-term dependencies. Attention mechanisms tailored for variable interaction and specialized techniques like de-stationary attention illustrate a focus on modelling complex inter-variable relationships and handling non-stationary data. Overall, the trend is towards creating more sophisticated, adaptable models that can efficiently handle the intricacies of time series data, providing more accurate and interpretable forecasts while improving, or at least maintaining, efficiency.

Regarding the certainty of evidence, ten of the models show strong evidence, thirteen show weak evidence, and the remaining ones have no dedicated ablation studies or similar. However, the overall trend is promising, as most models demonstrate improvements in forecasting accuracy and efficiency. This indicates that the innovative attention mechanisms are generally beneficial, even when the direct evidence may vary in strength. The continuous advancements and empirical validations in time series forecasting suggest a positive impact of these mechanisms on enhancing model performance and adaptability to complex temporal patterns.

5.2.2. Feature Selection

Effective feature selection is beneficial in time series forecasting, as it enhances model performance by focusing on the most relevant data while reducing noise. This section examines direct and indirect methods of feature selection used in various transformer models, highlighting their strategies to optimize forecasting accuracy.

5.2.2.1. Direct

Lim et al. (2020), in **TFT**, perform feature selection through variable selection networks that dynamically determine the relevance of each input feature at every time step. These networks generate selection weights for both static and time-varying inputs by processing them through a Gated Residual Network (GRN) followed by a softmax layer. The resulting weights scale the feature vectors, highlighting the most informative features and removing unnecessary noisy inputs.

M. Li et al. (2022), in **Umformer**, employ a Feature Selection Network using GRNs to evaluate and optimize the selection of features based on their predictive importance, managing

nonlinear relationships within the data. This process is enhanced by GLUV3, which refines the gating mechanisms within GRNs, allowing for nuanced control over feature retention through advanced gating operations that modulate the flow of information precisely.

T. Li, Liu, et al. (2023), in **MASTER**, implement a market-guided gating mechanism that dynamically scales input features based on current market conditions, allowing the model to emphasize or de-emphasize features according to their current relevance. Additionally, Z. Wang et al. (2024), in **PWDformer**, create the Frequency Selection Module that emphasizes significant features and reduces noise, optimizing the input for better prediction accuracy in complex time series data.

5.2.2.2. Indirect

Xie & He (2022), In **DDPformer**, use channel-dependent attention mechanisms, which dynamically adjust the importance or weights of features processed by each head in the multi-head attention framework. This selectively emphasizes more informative features and diminishes less relevant ones during the learning process.

C. Zhang et al. (2023), in **Causalformer**, integrate a form of causal feature selection through a Granger causality graph. The model identifies and encodes causal relationships among variables in multivariate time series data. This matrix effectively acts as a selection mechanism by highlighting the causal influences among the variables, focusing the model's attention on significant relationships for forecasting.

Ni et al. (2024), in **Basisformer**, performs implicit variable selection through the learning of basis vectors and the calculation of similarity coefficients in the Coef Module. These coefficients highlight the most relevant basis vectors for each time series, effectively selecting the variables that are most informative for forecasting. Finally, Tevare & Revankar (2023), in **Stack Transformer**, use attention mechanisms, both single and multi-headed, to weight input features based on their relevance to the prediction task, allowing the model to dynamically determine which features of the input data are most predictive during the training process, enhancing model performance without a pre-modelling feature selection.

5.2.2.3. Discussion

Feature selection strategies are gaining more and more attention as they allow the model to reduce complexity by focusing on the relevant features, and also allows for better predictions since it can discard unhelpful variables. Some transformers do this directly, while others do it indirectly or implicitly.

Regarding the certainty of evidence for feature selection effectiveness varies across the models. TFT and PWDformer provide strong evidence through ablation studies showing significant performance impacts when key feature selection components are removed,

confirming their importance. In contrast, the other models lack dedicated ablation studies, but their good results implicitly suggest that the alterations done were relevant in that.

5.2.3. Channel Dependence and Independence

In the realm of time series forecasting using transformers, the nuanced handling of channel dependence and independence has emerged as a pivotal aspect of model design. Models leveraging channel dependence emphasize capturing inter-variable relationships to enhance forecasting accuracy, while those prioritizing channel independence focus on preventing cross-channel interference to improve robustness and efficiency.

5.2.3.1. Channel Dependence

Transformers emphasizing channel dependence capture correlations between variables to improve forecasting accuracy. The **TACTiS** model (Drouin et al., 2022) sets a foundation by utilizing an attentional copula mechanism to effectively model intricate variable correlations effectively. Building on this, **Muformer** (P. Zeng et al., 2022) introduces a Similar Pruning Heads (SPH) mechanism to reduce redundancy by dynamically pruning less informative heads based on KL divergence, emphasizing channel-specific contributions.

Further extending this concept, **DDPformer** (Xie & He, 2022) adapts channel attention mechanisms inspired by Squeeze-and-Excitation networks to dynamically weigh the importance of each channel, enhancing feature-specific focus within the data. Similarly, **Dsformer** (Yu et al., 2023) employs a Temporal Variable Attention block to scrutinize interdependencies among variables, facilitating a deeper understanding of multivariate interactions.

iTransformer (Y. Liu et al., 2024) captures dependencies and correlations between variables by applying self-attention across variate tokens, allowing the model to leverage inter-variable relationships. Meanwhile, **MR-transformer** (S. Zhu et al., 2023) takes a dual approach by capturing both consistent and specific variable features through temporal convolution and variable attention mechanisms, enhancing the model's capacity to understand and model dependencies among different time series variables.

In financial forecasting, **MASTER** (T. Li, Liu, et al., 2023) exemplifies the strategic use of channel dependence through its inter-stock aggregation phase, processing correlations among stocks to reflect the complex dynamics of financial markets. Similarly, **RSMformer** (G. Tong et al., 2024) employs mechanisms like Residual Sparse Attention and cross-scale feature relationships to enhance forecasting by exploiting channel interdependencies at various scales. **SageFormer** (Z. Zhang et al., 2024) integrates graph neural networks and global tokens to exploit inter-series relationships, showcasing the trend towards utilizing network structures to enhance channel dependence.

Additionally, **MSRN-Informer** (X. Wang, Xia, et al., 2023) employs Residual Networks with skip connections to bypass one or more layers, allowing the network to learn residual functions instead of direct mappings. This design helps maintain information flow, mitigates the vanishing gradient problem, and facilitates the training of deeper networks, ultimately improving convergence speed and accuracy. Finally, **JTFT** (Y. Chen et al., 2023) incorporates channel dependence through low-rank attention (LRA) layers, efficiently capturing cross-channel dependencies after initial channel-independent processing.

5.2.3.2. Channel Independence

On the flip side, some models prioritize channel independence to prevent cross-channel interference and enhance model robustness. **PatchTST** (Nie et al. 2023) processes each feature of a multivariate time series separately, segmenting each feature into patches that are independently embedded and processed through the Transformer encoder using shared weights across all channels. **PatchTCN-TST**, (Cen & Lim, 2024) also adopts a channel-independent strategy, avoiding cross-channel noise while sharing parameters across channels.

FPPformer (Shen et al., 2023) extends this approach by processing each variable's data in parallel paths during initial stages, avoiding inductive biases and ensuring features from one variable are not influenced by others. After independent feature extraction, these features are integrated for final prediction, improving robustness and accuracy.

JTFT, (Y. Chen et al., 2023) employs a Channel-Independent (CI) setting, treating each channel as an independent univariate series, reducing the risk of overfitting and improving robustness by sharing network structures across all channels. A low-rank attention layer (LRA) integrates cross-channel dependencies without entangling them with temporal dependencies. **Triformer** (Cirstea et al., 2022) uniquely addresses temporal patterns of each variable in multivariate data by assigning distinct parameters to each variable. This is achieved by factorizing projection matrices into shared and unique parts, with each variable having its own small, unique matrix generated from a learnable memory vector through a neural network generator. This design captures the individual temporal characteristics of each variable without averaging them across variables, maintaining channel independence.

5.2.3.3. Discussion

In time series forecasting using transformers, balancing channel dependence and independence is crucial for optimizing model performance. While some models emphasize capturing inter-variable relationships to enhance accuracy, others prioritize channel independence to prevent cross-channel interference and improve robustness. Notably, JTFT exemplifies a hybrid approach by employing a channel-independent setting initially and later integrating cross-channel dependencies through a low-rank attention layer, combining the strengths of both strategies for robust and accurate forecasting.

The certainty of evidence regarding channel dependence and independence mechanisms is substantiated through various dedicated ablation studies, that provide robust evidence supporting the design choices of these models, confirming the pivotal roles of channel dependence and independence in optimizing time series forecasting performance.

5.2.4. Hierarchical and Multi-Scale Approaches

As research advances, several models have emerged that incorporate hierarchical structures and multi-scale methods to improve predictive accuracy and efficiency. These approaches vary in implementation, focusing on different aspects of temporal data to handle complexity and scale effectively.

5.2.4.1. Hierarchical Only

Several models combine hierarchical structures with other techniques to enhance their predictive capabilities. H. Su et al. (2021), in **AGCNT**, use a hierarchical encoder with stacked Probsparse Graph Self-Attention Blocks (PGAB) to capture complex dependencies at different abstraction levels, enhanced by convolution and pooling operations, compressing and refining the data for effective local and global pattern recognition. Similarly, Zeng et al. (2023), in **Seformer**, employ hierarchical positional encoding to process long sequences by segmenting them into blocks, managing local dynamics within each block (intrablock) and relationships between blocks (interblock) to capture both local and global dependencies effectively, while maintaining the integrity of positional information across different scales. T. Li, Liu, et al. (2023), in **MASTER**, use intra-stock and inter-stock hierarchical aggregation to model stock correlations. It employs self-attention for fine-grained temporal details within individual stocks and multi-head attention to understand dynamic relationships between multiple stocks, enhancing stock price prediction accuracy.

Jawed & Schmidt-Thieme (2022), in **GQFormer**, utilize quantile transposed attention to capture interactions among different quantile forecasts, focusing on pairwise interactions across quantile levels for accurate modelling of probabilistic forecast distributions. Similarly, Ye et al. (2023), in **TDT**, implement a hierarchical method for forecasting with two decoders: one predicting the mean to capture central tendency and regular patterns, and another predicting standard deviation to quantify uncertainty around the mean, providing structured probabilistic forecasts.

W. Li et al. (2023), in **Bidformer**, integrate hierarchical self-attention distillation, downsampling sequences through max-pooling to focus on dominant features and reduce memory usage progressively. This approach enhances scalability and performance in long sequence time-series forecasting. J. Tong et al. (2023), in **PDTrans**, use a two-layer approach, where Transformer extracts temporal features and makes primary forecasts. Then a

conditional generative model refines these forecasts using variational inference. This allows the model to handle complex dependencies and interactions more effectively

5.2.4.2. Multi-Scale Only

Other models focus solely on multi-scale or multi-resolution techniques to capture data at different levels of granularity. Shen & Wang (2022), in **TCCT**, utilize the Passthrough mechanism to combine outputs from all encoder self-attention blocks, creating a multi-scale feature map that integrates both local details and broader contextual information. Similarly, Madhusudhanan et al. (2022), in **Yformer**, inspired by the U-Net architecture, employ an encoder-decoder structure with contracting and expanding paths, downsampling input data to capture multi-level granularity and upscaling it to refine predictions, handling both high-level trends and fine-grained details.

Y. Li, Lu, et al. (2023), in **Conformer**, represent time-series data at various temporal resolutions (seconds, minutes, and hours) by embedding multi-scale representations into a latent space. The model concatenates and weights these embeddings to capture diverse temporal patterns across different granularities for accurate long-term forecasting. Sasal et al. (2022), in **W-Transformers**, utilize Maximal Overlap Discrete Wavelet Transform (MODWT) for multi-resolution analysis, decomposing time series data into multiple levels to capture different frequency components. This approach identifies both high-frequency details and low-frequency trends, enhancing the model's understanding of non-stationary data for more accurate forecasts.

X. Wang, Liu, Yang, et al. (2023), in **CNformer**, implement stacked dilated convolutional blocks (SDCBs) to expand the receptive field and capture patterns at varying scales. This multi-scale method identifies short-term fluctuations and long-term trends within time series data, integrating information across temporal resolutions to improve prediction accuracy. X. Wang, Liu, Du, et al. (2023), in **CLformer**, integrate dilated causal convolutions to expand the receptive field exponentially and capture temporal patterns at multiple scales. This approach aggregates information over different temporal resolutions,. Additionally, the locally grouped autocorrelation mechanism further supports multi-scale analysis by segmenting the time series and calculating autocorrelation within these segments

Qu et al. (2024), in **Forwardformer**, employ Multi-Scale Forward Self-Attention (MSFSA) with forward sliding window attention for capturing local dependencies, forward dilated sliding window attention for medium-range dependencies by introducing gaps, and forward global sliding window attention for long-range dependencies. This multi-scale approach enables efficient processing of data at different temporal resolutions. H. Cao et al. (2023), in **InParformer**, use Interactive Partitioned Convolution (IPConv) with multiple parallel convolution branches to capture features at various resolutions. This multi-resolution processing extracts information from fine details to broader trends, enhancing the model's ability to predict complex temporal patterns. Yu et al. (2023), in **Dsformer**, implement

Double Sampling (DS) Blocks for multi-resolution processing, combining down-sampling to capture global patterns and piecewise sampling for fine-grained details. This approach ensures effective capture and utilization of both broad trends and detailed variations in time series data for accurate long-term predictions.

Du et al. (2023), in **Preformer**, use the Multi-Scale Segment-Correlation (MSSC) mechanism to segment time series into exponentially increasing lengths, capturing dependencies at different temporal resolutions. This multi-scale segmentation enables the model to analyze both fine-grained and coarse-grained patterns, enhancing forecasting by aggregating outputs across scales. Ouyang et al. (2023a), in **Rankformer**, use Multi-Level Decomposition (MLDecomp) blocks, which employ multiple moving average (MMA) filters with different kernel sizes. This multi-scale approach decomposes time series into components such as short-term seasonal variations and long-term trends, improving forecasting accuracy by understanding complex patterns in the data.

Y. Wang, Zhu, & Kang (2023), in **DESTformer**, implement Multi-Scale Attention (MSC-Attention) to capture sub-trends at various scales using convolutional filters with different receptive fields. This approach adaptively models long-term trends and dependencies at multiple resolutions, providing a nuanced understanding of trend patterns for improved forecasting. N. Wang & Zhao (2023a), in **Enformer**, utilize the Coarse Matching Module (CMM) with convolutional layers of varying kernel sizes to extract features at different temporal resolutions. This multi-scale approach captures dependencies at multiple time scales, enhancing the model's ability to capture short-term and long-term patterns in time-series data.

Lee et al. (2024), in **TS-Fastformer**, incorporate the Sub Window Tokenizer (SWT) and Past Attention Decoder for multi-resolution processing. The SWT segments input data into smaller windows to capture patterns at different granularities, improving forecasting accuracy over varying time horizons. Chen et al. (2023), in **Pathformer**, divide time series data into patches of various sizes to capture different temporal resolutions simultaneously. The multi-scale router dynamically selects the most relevant patch sizes based on the input data’s characteristics, and the dual attention mechanism processes these patches to capture both local and global dependencies.

5.2.4.3. Multi-Scale and Hierarchical Techniques

Several models combine multi-scale analysis with hierarchical techniques to capture complex temporal patterns effectively. H. Zhou et al. (2020), in **Informer**, use self-attention distilling, where each layer condenses and abstracts the information from the previous layer, to progressively reduce sequence length via max-pooling, capturing multi-scale information efficiently. H. Wu et al. (2022), in **Autoformer**, utilize multi-resolution decomposition to analyze time series data at different scales, enhancing prediction accuracy by focusing on various granularities and integrating period-based dependencies.

Tang & Matteson (2021), in **ProTran**, implement a hierarchical structure with layers of stochastic latent variables, each layer capturing different levels of abstraction and temporal patterns, improving forecasting robustness. S. Liu et al. (2022), in **Pyraformer**, employ a hierarchical, multi-resolution approach through Coarser-Scale Construction Module (CSCM), that creates a pyramidal graph structure by downsampling the input data to form layers that represent different resolutions, from fine to coarse. Followed by PAM that then applies a tailored attention mechanism both within each layer (intra-scale) and across different layers (inter-scale). Cirstea et al. (2022), in **Triformer**, utilize a triangular layering design, where the input size to each subsequent layer is reduced exponentially by using only the pseudo timestamps from the previous layer, to capture features at multiple temporal scales, aggregating outputs from all layers for comprehensive understanding.

T. Zhou et al. (2022), in **FEDformer**, integrate Discrete Wavelet Transform (DWT) for multi-resolution analysis and Mixture Of Expert Decomposition (MOEDecomp) for hierarchical processing, effectively managing complex temporal information to enhance forecasting accuracy. S. Huang et al. (2022), in **CEEMDAN**, combine multi-resolution decomposition using CEEMDAN to decompose the time series data into IMFs at different scales. Subsequently, the hierarchical approach is applied by analysing the complexity of these IMFs with Sample Entropy and then selectively forecasting them with either a Transformer model for high-complexity components or a BPNN for low-complexity ones, improving accuracy across varying data complexities.

Shabani et al. (2023), in **Scaleformer**, implement a hierarchical, multi-scale approach by iteratively refining forecasts at progressively finer resolutions, starting with a coarse scale. At each scale, the input to the encoder is downsampled, and the decoder uses the upsampled output from the previous scale, ensuring detailed and refined predictions. Cross-scale normalization is applied to maintain consistency and reduce errors. Yang & Lu (2023), in **Foreformer**, use Multi-Temporal Resolution (MTR) Module that operates by downsampling the input sequence into smaller subsequences at each level of a binary tree-like hierarchy. Convolutional layers are then applied to these subsequences to extract local temporal features at various scales. This hierarchical structure allows the model to capture temporal patterns at multiple resolutions. Zhong et al. (2023), in **NTformer**, deploy hierarchical attention mechanisms for multi-resolution/multi-scale analysis, capturing short-term to long-term patterns simultaneously. This hierarchical approach improves model robustness by organizing temporal dependencies effectively.

Zeng et al. (2022), in **Muformer**, a multi-granularity attention head mechanism combined with multiple perceptual domain (MPD) processing. The hierarchical framework of Muformer prunes redundant information using the Kullback-Leibler (KL) divergence measure, ensuring that the attention heads capture unique and relevant features across different scales, ensuring detailed pattern recognition. S. Ma et al. (2023), in **TCLN**, use a multi-kernel CNN module for multi-scale spatial feature extraction and hierarchical stacking

of encoder layers (LSTM and Transformer) for comprehensive temporal abstraction. This combined approach ensures robust modelling of complex multivariate time series data. Y. Li, Qi, et al. (2023), in **SMARTformer**, combine Integrated Window Attention (IWA) for multi-scale dependency handling and a Semi-Autoregressive (SAR) Decoder for hierarchical sequence processing. This integration enables effective capture of both local and global temporal patterns, enhancing forecasting accuracy.

Y. Cao & Zhao (2023), in **DWTformer**, utilize wavelet decomposition, that transforms the 1D time series into 2D tensors that capture different frequency components, and Inception-like convolutional layers within the 2D-FeaturesBlock that then apply convolutional filters of various sizes in parallel, capturing features at multiple scales simultaneously. N. Wang & Zhao (2023b), in **Convolution Transformer**, integrate multi-resolution techniques through a Multi-layer Feature Fusion component, combining feature maps from different encoder layers to capture detailed local features and global patterns. Hierarchical processing in the encoder builds upon successive layers for comprehensive pattern extraction. B. Li, Cui, et al. (2023), in **DifFormer**, employ multi-resolutional differencing that inherently creates a hierarchical structure by analysing the data at different scales. Patch Merging and Dynamic Ranging further support this hierarchy by segmenting the time series into patches of varying sizes and adjusting the attention spans dynamically based on the data's patterns.

D. Cao et al. (2024), in **TEMPO**, implement multi-scale analysis by decomposing time series data into various resolutions to capture different patterns and periodicities. Its hierarchical methods organize data into levels of abstraction, enhancing forecasting accuracy by integrating broad trends with specific patterns. Ni et al. (2024), in **BasisFormer**, integrate adaptive self-supervised learning to capture basis vectors representing multi-resolution temporal patterns. Hierarchical Coef Module selects and weighs relevant patterns across different scales, enhancing forecasting accuracy through structured multi-scale integration. Shen et al. (2023), in **FPPformer**, uses hierarchical encoder-decoder structures to process data from fine to coarse resolutions, capturing diverse temporal dynamics. This approach ensures comprehensive feature integration across multiple scales, improving forecasting accuracy for complex time-series data.

G. Tong et al. (2024), in **RSMformer**, implement a multiscale forecasting strategy, using up-and-down sampling techniques, with hierarchical refinement, transforming data across resolutions to capture temporal dependencies comprehensively. This approach ensures accurate modelling of detailed and broad temporal information in time-series data. Y. Zhang, Wu, et al. (2024), in **MTPNet**, Utilizes a hierarchical pyramid structure to partition time series data into patches at different scales, preserving spatial and temporal dimensions. This multi-scale and hierarchical integration improves the model's ability to learn complex temporal patterns and enhance forecasting accuracy. Y. Wang et al. (2024), in **Graphformer**, integrate Multi-Scale Feature Fusion for capturing temporal correlations at various granularities and hierarchical graph self-attention for processing inter-variable dependencies. This combined

approach ensures robust handling of temporal and spatial dynamics in multivariate time series forecasting. Z. Liu et al. (2024), in **Hidformer**, use a segment-and-merge architecture for multi-resolution/multi-scale processing, capturing local patterns at different temporal scales. Hierarchical dual-tower architecture integrates information from time and frequency domains, enhancing accuracy and robustness in long-term time series forecasting.

5.2.4.4. Discussion

The trend in time-series forecasting models shows a clear shift towards incorporating hierarchical and multi-scale techniques to handle the complexity and scale of temporal data. These models use hierarchical structures to manage dependencies at various abstraction levels, enhancing the ability to capture intricate patterns and interactions within the data. Multi-scale approaches are equally prevalent, enabling models to analyze data at different granularities, from fine details to broad trends. By combining these methods, researchers can leverage the strengths of both hierarchical organization and multi-resolution analysis, leading to more comprehensive and reliable forecasting models.

Regarding the certainty of evidence, only the transformers with dedicated components for multi-scale techniques were considered, as the ones that do it indirectly, for example through embedding, have their dedicated studies regarding certainty of evidence. the ablation studies of transformers that directly incorporate multi-scale components demonstrate their significant impact on time-series forecasting accuracy. Muformer's multi-granularity attention mechanism, based on Multiple Perceptual Domains (MPD), improves MSE and MAE by reducing redundancy in attention heads. Forwardformer's Multi-Scale Forward Self-Attention (MSFSA) mechanism captures short-term and long-term dependencies effectively through optimized parameter tuning. Preformer's multi-scale structure extracts dependencies at various temporal resolutions, leading to better performance on complex datasets. Scaleformer combines a multi-scale framework with adaptive loss for enhanced results. Foreformer's Multi-Temporal Resolution (MTR) module is captures detailed temporal features, significantly improving forecasting accuracy. RSMformer’s multiscale forecasting and residual sparse attention refine predictions iteratively, focusing on relevant information. Lastly, MTPNet's multi-scale transformer pyramid outperforms single-scale models, highlighting the importance of multi-scale temporal dependency learning. These studies confirm the value of direct multi-scale approaches in developing robust and precise time-series forecasting models.

5.2.5. Hybridization

In the realm of time series forecasting, hybridization stands as a pivotal strategy, combining the strengths of transformer with the strengths of different models.

5.2.5.1. Positional Encodings

As covered the **section 5.1.1 Positional Embedding**, various hybrid approaches have been employed to enhance time series forecasting. For instance, Y. Cao & Zhao (2023), in **DWTformer**, use 2D-FeaturesBlock with multi-scale convolutional filters to capture complex patterns in time series data. Similarly, X. Wang, Liu, Du, et al. (2023), in **CLformer**, apply dilated causal convolutions to capture multi-scale temporal patterns, refining input data by emphasizing regular and predictable patterns.

Y. Zhang, Wu, et al. (2024), in **MTPNet**, take a different approach by using a 1-layer CNN to project time series data into a higher-dimensional space, aiding in capturing fine to coarse temporal dependencies. Building on this idea, Domínguez-Cid et al. (2023), in **TEFNEN**, employ fully connected single-layer neural networks for embedding and an RNN layer for capturing sequential and temporal dependencies.

Additionally, Cen & Lim (2024), in **PatchTCN-TST**, use TCN with causal convolutions to embed temporal information, preserving temporal order and focusing on local dependencies. Finally, (J. Zhu et al., 2023), in **SL-Transformer**, integrate transformers with LSTM networks for feature extraction and temporal sequence processing, enhancing power forecasting accuracy.

5.2.5.2. Attention Mechanisms

As covered in the **section 5.2.1.2 Hybrid Mechanisms**, various approaches have been adopted to enhance the attention mechanisms. Shen & Wang (2022), in **TCCT**, incorporate a lightweight 1×1 convolution to maintain structure alongside self-attention for global dependencies, reducing computational costs. Similarly, Y. Li et al. (2022), in **ACT**, utilize a convolutional attention block with a causal convolutional layer to capture local context before self-attention.

Another approach is seen in Z. Liu et al. (2024), in **Hidformer**, use recurrence to captures temporal dependencies by processing data sequentially, leveraging information from previous time steps, while linear attention reduces computational complexity, scaling linearly with sequence length. H. Cao et al. (2023), in **InParformer**, utilize Interactive Partitioned Convolution (IPConv) for key-value pair compression in its dual attention structure.

Moreover, not explicitly covered in that section, Dai et al. (2023), in **PDF**, combine self-attention with convolutional layers in the Dual Variations Modeling Block to process 2D tensors and capture local patterns. Expanding on convolutional integration, (B. Li, Cui, et al., 2023), in **Difformer**, integrate CNN elements via Δ-Lagging to capture local patterns and adjust focus dynamically in time series.

Further enhancing long-term dependency capture, N. Wang & Zhao (2023b), in **Convolution Transformer,** use causal dilated convolutions between self-attention modules to enhance long-term dependency capture. In contrast, Cirstea et al. (2022), in **Triformer**, employ recurrency because Patch Attention makes it harder to capture relationships among different patches and long-term dependencies. It connects pseudo timestamps of consecutive patches to maintain temporal information flow, using a gating mechanism to control the flow of information from one pseudo timestamp to the next. Lastly, (H. Su et al., 2021), in **AGCNT**, employ 1-D convolutional layers to process outputs from adaptive graph self-attention layers.

5.2.5.3. Multi-Scale Approaches

Various models leverage multi-scale and multi-resolution techniques to enhance their performance, as covered in **section 5.2.4 Hierarchical and Multi-Scale Approaches**. N. Wang & Zhao (2023a), in **Enformer**, utilize the Coarse Matching Module with convolutional layers of varying kernel sizes to capture dependencies at multiple time scales. Similarly, H. Cao et al. (2023), in **InParformer**, use Interactive Partitioned Convolution with multiple parallel convolution branches for multi-resolution processing of temporal patterns.

Building on these approaches, Y. Wang, Zhu, & Kang (2023), in **DESTformer**, implement Multi-Scale Attention with convolutional filters of different receptive fields to adaptively model trends at various scales. Likewise, Yang & Lu (2023), in **Foreformer**, use a Multi-Temporal Resolution Module with convolutional layers to extract features from downsampled subsequences in a binary tree-like hierarchy.

Additionally, X. Wang, Xia, & Deng (2023), in **MSRN-Informer**, use 1D CNNs with multi-scale convolutional kernels to capture features at various scales and local dependencies. Further enhancing multi-resolution processing, N. Wang & Zhao (2023b), in **Convolution Transformer**, integrate multi-resolution techniques through Multi-layer Feature Fusion, combining feature maps from different encoder layers for detailed local and global pattern capture. Lastly, S. Ma et al. (2023), in **TCLN**, use a multi-kernel CNN module for multi-scale spatial feature extraction and hierarchical stacking of encoder layers for comprehensive temporal abstraction.

5.2.5.4. Decomposition

In the realm of decomposition for time series forecasting, different approaches have been utilized to break down data into its fundamental components. J. Tong et al. (2023), in **PDTrans**, use convolution operations and MLP to decompose time series into trend and seasonality components, respectively. Similarly, X. Wang, Liu, Yang, et al. (2023), in **CNformer**, employ stacked dilated convolutional blocks to refine seasonal components and capture both local and long-term dependencies.

5.2.5.5. Feature Focus

Feature selection and extraction play an important role in improving the accuracy and efficiency. Lim et al. (2020), in **TFT**, perform feature selection through variable selection networks using Gated Residual Networks (GRNs) to dynamically determine the relevance of input features. Building on this, M. Li et al. (2022), in **Umformer**, employ a Feature Selection Network with GRNs and GLUV3 to optimize feature selection and manage nonlinear relationships with advanced gating mechanisms.

A similar approach is taken by S. Zhu et al. (2023), in **MR-transformer**, which utilize temporal convolution and variable attention mechanisms to capture both consistent and specific variable features. Enhancing temporal feature extraction further, (Fan et al., 2023), in **TEDformer**, enhance temporal feature extraction using Temporal Convolutional Networks (TCN) to capture local patterns and trends. The TCN processes trend components within the encoder to effectively capture long-range dependencies and local features, integrating TCN into both the encoder and decoder.

5.2.5.6. Modeling

In the modelling of time series forecasting, several innovative hybrid approaches have been developed. For instance, Zeng et al. (2023), in **Seformer**, use recursive interblock position encoding to manage relationships between data blocks, maintaining global sequence context and long-range dependencies. Similarly, S. Huang et al. (2022), in **CEEMDAN**, combine Back Propagation Neural Network (BPNN) for forecasting low-complexity subsequences with transformers for high-complexity segments.

Taking a different approach, X. Zhang et al. (2022), in **TDformer**, utilize a Multi-Layer Perceptron (MLP) to handle the trend component of time-series data. The time series is decomposed into trend and seasonal components, with the MLP predicting future values by learning complex, non-linear relationships in the trend data. Building on the concept of decomposition, Y. Li, Lu, et al. (2023), in **Conformer**, use RNNs and GRUs within the Stationary and Instant Recurrent Network (SIRN) to capture both long-term and short-term temporal patterns. The trend component is processed by a GRU to refine long-term patterns, while the seasonal component undergoes further decomposition through convolution and GRU blocks to capture short-term variations.

Moreover, Z. Zhang et al. (2024), in **SageFormer**, integrate Graph Neural Networks (GNNs) with the traditional transformer encoder to capture inter-series dependencies in multivariate time series data. It uses global tokens and iterative message passing to facilitate inter-series interactions, with GNNs handling these interactions and the transformer managing intra-series interactions, ultimately simplifying the decoding process with a linear ForecastingHead.

Enhancing deep information extraction, Jiang et al. (2022), in **Deep Autoformer**, enhance the basic Autoformer framework by strategically integrating Multi-Layer Perceptrons (MLPs) to improve deep information extraction. This integration enhances feature extraction efficiency, contributing to superior performance in Very Short-Term Load Forecasting (VSTLF) and Short-Term Load Forecasting (STLF).

5.2.5.7. Hybridization with Statistical and Mathematical Models

(Tang & Matteson, 2021), in **ProTran**, integrate transformers with state-space models (SSMs), combining attention mechanisms with the structured approach of SSMs to capture long-term dependencies and manage uncertainties uncertainties through latent variables. Similarly, (Y. Lin et al., 2021), in **SSDNet**, merge transformers with SSMs, where the transformer extracts temporal patterns and generates latent representations, which are then fed into the SSM for trend, seasonality, and residual decomposition. This integration leverages both complex feature extraction and structured, interpretable decomposition for accurate forecasts.

Taking a different approach, H. Wu et al. (2023), in **D-Transformer**, integrate the Duffing equation into the transformer's initialization, using this nonlinear differential equation to set initial weights for the encoder, decoder, and fully connected layers. This approach captures complex dynamical behaviours, enhancing the model's ability to handle nonlinearities in time series data, leading to faster training convergence and improved prediction accuracy.

Moreover, C. Zhang et al. (2023), in **Causalformer**, combine temporal and causal features using a Time Encoder with ProbSparse Self-attention for temporal dependencies and a Causal Encoder with a Granger causality graph to identify and integrate causal relationships.

5.2.5.8. Discussion

Hybridization strategies in transformer models for time series forecasting enhance performance by integrating various neural network architectures, statistical models, and mathematical equations. Combining convolutional layers with transformers, as seen in DWTformer and CLformer, captures both local details and broader temporal dependencies. Similarly, integrating RNNs and GRUs in models like TEFNEN and Conformer improves the capture of sequential dependencies and long-term patterns, essential for accurate forecasting.

Additionally, hybridization with statistical and mathematical models adds structure and interpretability. Models such as ProTran and SSDNet combine transformers with state-space models (SSMs) for advanced feature extraction and structured decomposition, leading to more accurate forecasts. The inclusion of the Duffing equation in D-Transformer highlights how mathematical models enhance the handling of nonlinearities in time series data. These diverse methodologies enable hybrid models to make use of the advantages of different models.

Regarding the certainty of evidence, in models like PDF and Difformer, the necessity of convolutional structures is highlighted, showing significant impacts on performance when altered or removed. TCLN's results emphasize the effectiveness of combining LSTM, Transformer encoders, and Multi-kernel CNNs for superior feature extraction. Similarly, the Convolution Transformer and Triformer underline the importance of maintaining expansive receptive fields and continuous temporal information flows for accuracy. Studies like those for TDformer and Conformer further reveal how specific elements like MLP and SIRN critically support the handling of trend data and the integration of local-global patterns, respectively. These findings demonstrate a strong certainty.

5.3. Model Optimization

Recent advancements have highlighted the importance of robust optimization techniques to enhance model performance. This section delves into various model optimization strategies, focusing on regularization, normalization, advanced optimization techniques, and diverse loss functions. These methods collectively aim to improve the generalization, stability, and accuracy of transformer models. Figure 5.4 illustrates the division and sub-division of these optimization techniques, providing a comprehensive overview of this section.

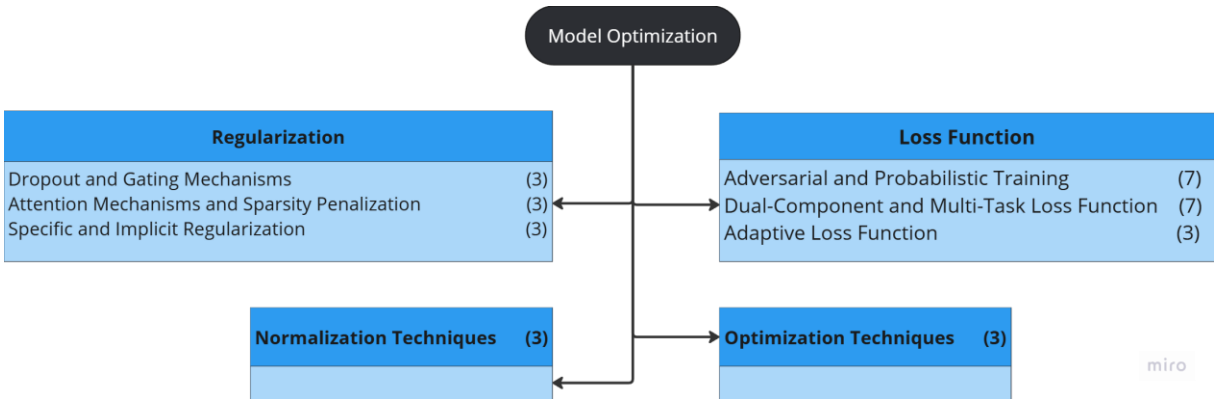

Figure 5.4 – Division and Sub-division of Model Optimization Techniques, Showing the Number of Models that fall under each Sub-division

5.3.1. Regularization

Incorporating regularization in transformers mitigates issues like overfitting and enhances the model's robustness and predictive accuracy.

5.3.1.1. Dropout and Gating Mechanisms

Lim et al. (2020), in **TFT**, employ dropout within Gated Residual Network (GRN) blocks to prevent overfitting, gating mechanisms to skip unnecessary parts providing adaptive depth,

and layer normalization to stabilize training. Domínguez-Cid et al. (2023), in **TEFNEN**, and H. Wu et al (2023), in **D-Transformer**, both use dropout regularization, with specific dropout rates of 0.2 and 0.05, respectively, to prevent overfitting by randomly omitting neurons during training.

5.3.1.2. Attention Mechanisms and Sparsity Penalization

Zeng et al. (2022), in **Muformer**, employ attention head pruning to eliminate redundant information and reduce overfitting. H. Zhou et al. (2020), in **Informer**, propose self-attention distilling that progressively reduces input sequence length at each encoder layer using self-attention and max-pooling. After self-attention, a convolution, ELU activation, and max-pooling halve the sequence length. C. Zhang et al. (2023), in **Causalformer**, use sparsity penalties to the weight matrix during decoding, focusing on the most significant causal relationships, improving interpretability and reducing complexity.

5.3.1.3. Specific and Implicit Regularization

Zeng et al. (2023), in **Seformer**, use a Conditional Variational Autoencoder (CVAE) with a KL-cross mechanism for information transfer regularization, preserving positional and sequential information by optimizing the Evidence Lower Bound (ELBO) and minimizing KL divergence between the approximated posterior and true prior distributions. Ni et al. (2024), in **BasisFormer**, use a smoothness regularization term in the loss function to ensure that the learned basis vectors transition smoothly over time, preventing overfitting and improving the interpretability of the basis vectors. Zhong et al. (2023), in **NTformer**, employ L2 regularization by adding a penalty term to the loss function, which is proportional to the sum of the squares of the model parameters.

5.3.2. Normalization Techniques

Shabani et al. (2023), in **Scaleformer**, use cross-scale normalization to address distribution shifts and error accumulation, that occur when processing data at multiple temporal scales, by normalizing the combined encoder and decoder outputs at each scale, stabilizing the learning process and enhancing robustness and accuracy in multi-scale forecasting. Nie et al. (2023), in **PatchTST**, employ instance normalization to preprocess time series data by normalizing each channel independently to have zero mean and unit variance, mitigating distribution shifts between training and testing data. Xie & He (2022), in **DDPformer**, use Layer Normalization (LN) that stabilize training by ensuring consistent mean and variance for each data sample, and Power Normalization (PN) to handle data fluctuations and outliers, improving training smoothness and performance.

5.3.3. Optimization Techniques

Y. Wang et al. (2024), in **Graphformer**, use gradient centralization (GC), by centralizing the gradient vectors to have zero mean, integrated into the Adam optimizer, enhancing training stability and efficiency. Guo et al. (2021), in **Stacked-Informer Network,** use the GC + Adam optimizer that integrates Gradient Centralization (GC) with the Adam optimizer. GC centralizes gradient vectors to zero mean, enhancing training stability and efficiency. Combined with Adam's adaptive learning rate, this approach achieves faster convergence and improved generalization.

5.3.4. Loss Functions

The loss function is crucial in machine learning as it quantifies the discrepancy between predicted outcomes and actual values, guiding the model to optimize its parameters effectively for better accuracy and performance. In the reviewed papers that use a simple Loss Function, MSE is used 45 times, MAE 17 times, RMSE 4 times, MAPE once, Quantile loss twice, and CRPS once. MSE and MAE are widely used for their effectiveness in regression tasks, with MSE being particularly sensitive to outliers and MAE offering a more robust measure by considering absolute differences. RMSE assesses the standard deviation of residuals, while MAPE focuses on percentage differences but has limitations when actual values are near zero.

5.3.4.1. Adversarial and Probabilistic Training

In order to apply adversarial training both S. Wu et al. (2020), in **AST**, and Y. Li et al. (2022), in **ACT**, use two different loss functions, one for the generator and another for the discriminator. AST uses a combination of adversarial loss and quantile loss where the discriminator uses adversarial loss, ACT uses quantile loss for the generator and cross-entropy loss for the discriminator.

Probabilistic forecasting is another important area where transformer models have made significant strides. Tang & Matteson (2021), in **ProTran**, uses a loss function combining reconstruction loss and Kullback-Leibler (KL) divergence. The reconstruction loss ensures that the generated outputs closely match the observed data, while the KL divergence regularizes the latent variables by measuring their divergence from a prior distribution, effectively capturing the uncertainties and dynamics of time series data. J. Tong et al. (2023), in **PDTrans**, combine the negative log-likelihood (NLL) loss with the KL divergence from the variational autoencoder framework, balancing reconstruction accuracy and latent space regularization for probabilistic forecasts.

Expanding on probabilistic approaches, Lin et al. (2021), in **SSDNet**, combine MAE and NLL in the loss function to simultaneously achieve accurate point and probabilistic forecasts. Y. Li,

Lu, et al. (2023), in **Conformer**, employ a dual-component loss function that integrates the MSE with the log-likelihood from the normalizing flow framework. This approach ensures the model not only aims for accurate point predictions but also models the underlying distribution of the time-series data to account for uncertainty. Ye et al. (2023), in **TDT**, utilize a NLL function assuming future values follow a Gaussian distribution, penalizing deviations from the predicted distribution, optimizing for both accuracy of the central tendency and uncertainty in predictions.

5.3.4.2. Dual-Component and Multi-Task Loss Function

Madhusudhanan et al. (2022), in **Yformer**, add a reconstruction loss to the standard forecasting loss that is used as an auxiliary task to reconstruct parts of the past input data, stabilizing training and improving generalization. Similarly, Jawed & Schmidt-Thieme (2022), in **GQFormer**, combine quantile loss functions with an Energy score-based approximation of the CRPS loss in a novel multi-task loss formulation, optimizing for sharp individual quantile estimations and ensure that these estimations are spread maximally apart to capture the various modes of the underlying distribution.

Qu et al. (2024), in **Forwardformer**, introduce a dual-component loss function that ensures that the model pays more attention to special events (holidays): the standard MSE over all days and an additional MSE component that focuses exclusively on holidays. D. Liu et al. (2024), in **Multirate-Former**, use a specialized loss function during unsupervised pretraining to minimize reconstruction errors of unobservable data, and a composite loss during supervised fine-tuning to balance prediction errors.

Shen et al. (2023), in **FPPformer**, use a composite loss function that combines MSE and MAE, where MSE penalizes larger errors and MAE provides balanced error assessment, enhancing robustness and accuracy. Ni et al. (2024), in **BasisFormer**, combine MSE loss with an alignment loss (InfoNCE loss) that ensures consistency between historical and future views of the time series, enhancing the alignment and interpretability of the learned bases.

Ashok et al. (2024), in **TACTiS-2**, use a two-stage loss function, first optimizing parameters for marginal distributions and then learning copula parameters, reducing complexity, facilitating faster convergence, and capturing dependencies among variables.

5.3.4.3. Adaptive Loss Function

Shabani et al. (2023), in **Scaleformer**, introduce an adaptive loss function, based on Barron's general and adaptive robust loss function, dynamically adjusting sensitivity to outliers to enhance performance and stability across different datasets and scales.

G. Tong et al. (2024), in **RSMformer**, and Y. Huang & Wu (2023), in **ADAMS**, employ an adaptive Huber loss function, combining the benefits of MSE and MAE, transitioning

between MSE for small errors and MAE for large errors, dynamically adjusting sensitivity through learnable parameters, effectively handling uncertainty and noise in time-series data.

5.3.5. Discussion

In optimizing transformer models for time-series forecasting, key trends include regularization, normalization, and diverse loss functions. Regularization techniques like dropout and gating prevent overfitting and enhance robustness. Normalization methods address distribution shifts, stabilizing the learning process. Optimization strategies, such as gradient centralization, improve training stability and convergence.

Loss functions are crucial: adversarial training uses distinct losses for generators and discriminators, while probabilistic approaches combine reconstruction losses with KL divergence to capture uncertainties. Dual-component and adaptive loss functions balance accuracy and robustness, handling outliers and enhancing model performance. These mechanisms collectively improve the generalization, stability, and interpretability of transformer models in time-series forecasting.

Regarding the certainty of evidence, regularization techniques show strong effectiveness in models like Muformer, NTformer, Seformer, and TFT. However, models such as TEFNEN, D-Transformer, and Causalformer lack dedicated studies. Normalization techniques present moderate evidence as Scaleformer and DDPformer demonstrate their effectiveness, while PatchTST does not. Optimization methods have less support, with strong evidence for Stacked-Informer's GC + Adam optimizer but lacking validation for Stack-transformer. Loss functions show robust evidence, particularly in Basisformer, GQFormer, Scaleformer, and RSMformer, enhancing accuracy and robustness. Although some models lack ablation studies, the overall evidence supports the effectiveness of these optimization strategies in improving transformer model performance for time-series forecasting.

5.4. TYPES OF LEARNING

Various learning strategies are employed in Transformers to enhance time series forecasting.

5.4.1. Innovative Learning Strategies

Transformers have leveraged different types of learning strategies, enabling improved accuracy and efficiency in predictions.

Thwal et al. (2023), in **Federated Transformer**, utilize federated learning, where multiple decentralized devices collaboratively train a shared model without centralizing data. This method, with an attentive aggregation mechanism (FedAtt), weights local updates based on their relevance to the global model, improving privacy, security, and model performance.

D. Cao et al. (2024), in **TEMPO**, employ zero-shot learning and transfer learning, using a pre-trained generative transformer (GPT), that has been trained on a large corpus of data to capture general patterns and structures, and adapt it to specific time series tasks via soft prompts. This enables the model to make accurate predictions on new, unseen datasets without requiring specific prior training.

Cen & Lim (2024), in **PatchTCN-TST**, employ multi-task learning (MTL), training the model to perform multiple forecasting tasks simultaneously. By sharing parameters across tasks, MTL improves model generalization and efficiency, harnessing correlations between different features to enhance prediction accuracy. Nie et al. (2023), in **PatchTST**, employs self-supervised learning, transfer learning, and unsupervised learning. It is pre-trained on large unlabelled datasets to predict masked patches, learning robust data representations. This is followed by fine-tuning on specific labelled datasets through transfer learning. This allows to identify inherent patterns, and generalize well across datasets.

Lee et al. (2024), in **TS-Fastformer**, integrate pre-trained learning through the Time-series Pre-trained Encoder (TPE), enhancing representation learning by pre-training on time-series data, capturing seasonal and trend patterns. These enriched representations feed into the transformer model, combining unsupervised pre-training with supervised learning to boost forecasting accuracy. Similarly, P. Chen et al. (2023), in **Pathformer**, demonstrate generalization and adaptability using transfer learning, pre-training on one dataset and fine-tuning on another, leveraging learned patterns to enhance performance on new data.

Ni et al. (2024), in **BasisFormer**, utilize end-to-end training, combining self-supervised and supervised learning processes within a single framework. This approach, including contrastive learning with the InfoNCE loss function, ensures consistency between historical and future segments, facilitating the learning of adaptive basis vectors

D. Liu et al. (2024), in **Multirate-Former**, employ unsupervised pretraining to estimate missing data points by minimizing reconstruction errors, followed by supervised fine-tuning on labeled data based on the prediction errors, ensuring accurate multistep predictions in industrial processes.

Ashok et al. (2024), in **TACTiS-2**, apply curriculum learning through a two-stage training process to simplify and improve the learning of complex dependencies. In the first stage, the model trains the parameters of the marginal distributions independently, ignoring the dependencies between variables. Once the marginals are accurately learned, the second stage focuses on training the copula parameters, which capture the dependencies, while keeping the marginal parameters fixed. This sequential approach helps the model converge faster and more effectively by breaking down the complex learning task into manageable steps.

5.4.2. Generative Training

Generative training has gained attention as it allows the model to predict outputs in a single forward pass, reducing inference time and mitigating error accumulation.

Some models integrate a Generative Adversarial Network (GAN) framework with the Transformer. S. Wu et al. (2020), in **AST**, and Y. Li et al. (2022), in **ACT**, implement this by having the generator forecasts future values, and the discriminator differentiate these from actual data, enhancing the model's replication of the data distribution.

Other models use a Generative Decoder. (H. Zhou et al., 2020), in **Informer**, and Guo et al. (2021), in **Stacked-Informer Network** feed the decoder with a combination of start tokens and placeholders for the target sequence, with the latter using attention mechanisms to focus on relevant parts of the input sequence. W. Li et al. (2023), in **Bidformer**, use generative inference, dividing the input sequence into a start token sequence and a placeholder for the target sequence, processed using masked multi-head attention.

Su et al. (2021), in **AGCNT**, implement generative training through its generative inference feature in the decoder, allowing the generation of future values in a single operation. Similarly, C. Zhang et al. (2023), in **Causalformer**, employ a generative approach in its decoder, where each head focuses on a specific variable, considering its causal influences, and produces an extended sequence in parallel, enhancing inference speed and accuracy by integrating causal insights and applying sparsity regularization. Yu et al. (2023), in **Dsformer**, employ a generative decoder based on MLP to transform processed feature vectors into final prediction outputs. This MLP-based decoder takes the comprehensive representations generated by the Temporal Variable Attention (TVA) blocks, which integrate both temporal and variable information, and synthesizes them into predicted values.

And finally, other models decide to incorporate Latent Variable Models. J. Tong et al. (2023), in **PDTrans**, integrate a conditional generative model with a Transformer using a variational autoencoder (VAE) framework to refine initial forecasts by modelling their predictive distribution through variational inference, producing probabilistic forecasts. Y. Li, Lu, et al. (2023), in **Conformer**, incorporate generative elements through a normalizing flow block, modelling the distribution of future series directly from latent states, capturing underlying patterns and uncertainties, enhancing robustness in long-term forecasting.

5.4.3. Probabilistic Forecasting

Probabilistic forecasting provides a comprehensive understanding of future uncertainties, enabling more informed decision-making and risk assessment. Tang & Matteson (2021), **in ProTran**, use stochastic latent variables and variational inference to capture uncertainties in time series data, generating diverse long-term forecasts and prediction intervals for enhanced interpretability. Jawed & Schmidt-Thieme (2022), in **GQFormer**, generate multiple quantile estimates across the forecast horizon using implicit quantile networks and a multi-

task loss function with Continuous Ranked Probability Score (CRPS) to estimate future values' conditional distribution.

Lin et al. (2021), in **SSDNet**, leverage the Transformer's ability to extract temporal patterns and the State Space Model's (SSM) probabilistic estimates. The Transformer processes historical data and covariates to generate latent representations, which are then used to estimate the SSM parameters. The SSM models the time series as a combination trend, seasonality, and Gaussian-distributed residuals. This enables SSDNet to predict future values and their distribution, capturing uncertainty using a combined loss function of MAE and Negative Log-Likelihood (NLL). Ye et al. (2023), in **TDT**, predict the probability distribution of future values by sequentially forecasting mean and standard deviation, effectively quantifying uncertainty with confidence intervals and risk assessments.

5.4.4. Discussion

The section on Types of Learning showcases how various innovative strategies are tailored to meet different needs. Federated learning enhances privacy by decentralizing data processing, transfer learning makes use of pre-existing knowledge, multi-task learning improves generalization, self-supervised learning derives robust data representations from unlabelled datasets, and end-to-end and curriculum learning handles complex dependencies and ensure consistent training.

Generative training, incorporating techniques like GANs and generative decoders, is gaining attention for reducing inference time and mitigating error accumulation, while probabilistic forecasting methods emphasize capturing uncertainties through advanced mechanisms like stochastic latent variables and state-space models, providing comprehensive insights into future predictions.

Regarding the certainty of evidence, the types of learning are difficult to test their impact since to test the probabilistic forecasting the ablation would lead to a different metric and standard supervised learning cannot easily replace Innovative Learning Strategies. As such most of them lack dedicated ablation studies except for AST, ACT and Informer that show that the generative training alleviates the error accumulation, as well as TEMPO and TACTiS-2 that show that the exclusion of the prompt component in ablation studies led to a deterioration in predictive accuracy and that the two-stage curriculum significantly contributes to the model's performance and efficiency, respectively. Overall, while dedicated ablation studies are scarce, the available evidence for these specific models highlights the benefits of their unique components and the good performance of the models also implicitly does the same.

5.5. REPEATED TRANSFORMERS

As previously discussed, most research in this area focuses on creating transformers tailored for specific purposes, rather than improving existing transformers. Consequently, there are relatively few studies that apply existing transformers to other domains, thus expanding the knowledge about their performance.

The Temporal Fusion Transformer (**TFT**) has been widely applied to various sectors, showcasing its utility in multiple contexts. For instance, it has been effectively utilized for forecasting Bitcoin price trends (Amadeo et al., 2023), weekly variations in hospital patient counts (Caldas & Soares, 2023), and shifts in energy demands within power distribution networks (Liao & Radhakrishnan, 2022). Similarly, it has proven valuable in healthcare for predicting vital sign trajectories (Phetrittikun et al., 2021) and in agriculture for forecasting wheat yields (Junankar et al., 2023). Additionally, its application extends to predicting the state of charge of sodium-sulphur (NaS) batteries (Almarzooqi et al., 2023) and energy consumption patterns (Jittanon et al., 2023), as well as wind power forecasting in South Africa (van Heerden et al., 2023). This diverse application spectrum illustrates the TFT's capacity to adapt to different data characteristics and forecasting requirements.

The Informer transformer, meanwhile, was used in more technical applications such as predicting motor bearing vibrations (Z. Yang et al., 2022) and heating loads (M. Gong et al., 2022). Notably, its use in heating load forecasting was enhanced by the introduction of a Relative Position Encoding algorithm, highlighting a specialized adaptation.

Lastly, the FEDformer was specifically employed to predict wind speed (Deng et al., 2022), showing its potential in environmental and weather-related forecasting.

All these show the flexibility and potential of transformer to be applied to diverse areas of real world datasets.

Also, most of the transformers use the Vanilla Transformer as their basis, only a few transformers apply their innovations to other existing transformers. This is represented in Figure 5.5, where all the transformers based on different transformers and what transformer they are based on are shown. The arrow to GPT is dotted to draw attention that GPT was not created for time series forecasting, unlike the other models.

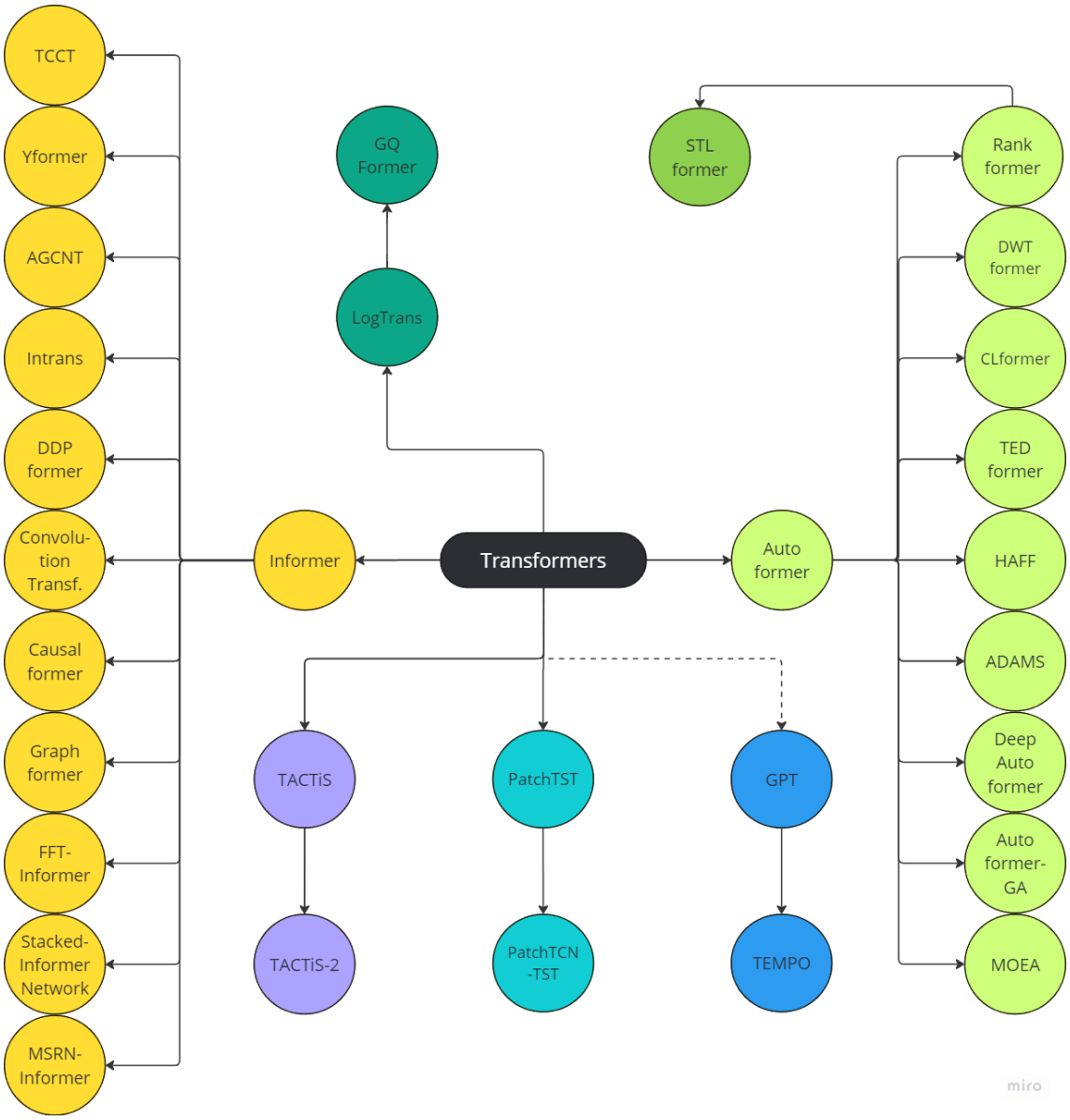

Figure 5.5 – Diagram of the Inspiration for Different Transformers

6. CONCLUSION AND FUTURE WORK

Recent advancements in transformer models for time series forecasting showcase significant trends, innovations, and methodological advancements. Enhanced input representations, such as adaptive and learnable embeddings and multi-resolution embeddings, improve model adaptability and capture temporal dependencies more effectively. Techniques like segment-based approaches and decomposition methods isolate trends and patterns, enhancing accuracy and robustness. Innovative attention mechanisms, including sparse and hybrid attention, address computational efficiency and capture both local and global dependencies. Hierarchical and multi-scale approaches decompose data into multiple resolutions, managing dependencies at various levels and improving predictive accuracy. Effective feature selection and covariate integration dynamically determine relevant inputs, reducing noise and enhancing model performance.

Additionally, advancements in handling channel dependence and independence have been made. Some models emphasize capturing inter-variable relationships for improved forecasting accuracy, while others focus on preventing cross-channel interference to enhance robustness. Hybridization strategies integrate transformers with other models like CNNs and RNNs for comprehensive feature extraction and temporal abstraction, leveraging the strengths of different architectures for improved performance. Regularization techniques, normalization methods, and advanced optimization strategies stabilize training and prevent overfitting. Diverse loss functions balance accuracy and robustness. Innovative learning strategies, such as transfer learning, enhance generalization and efficiency, while probabilistic forecasting elements allow users to trust model decisions and capture forecast uncertainty. Generative training enhance robustness. Moreover, there is a strong emphasis on reducing computational complexity from quadratic to linear, as shown in **Appendix A (Complexity Equations)**, and optimizing model structures to lower computational costs while maintaining performance.

However, there is no single state-of-the-art transformer model for time series forecasting due to factors like stochasticity, varying data characteristics, the use of unique private datasets and different evaluation metrics. Selecting a transformer model should depend on the specific needs and data characteristics. For privacy concerns, the Federated Transformer is suitable. For real-time forecasting using incremental training, InTrans is the only option for now. For probabilistic forecasting, GQFormer and TDT are worth considering. For non-stationary data, W-transformers or the non-stationary transformer are good options. For non-linear data, DDPformer is effective. If you need to use covariates, TFT is a strong choice. For datasets with high inter-variable correlation, JTFT and CNformer are excellent. Based solely on the lowest MSE metrics, MTST and Hidformer are top choices for multivariate forecasting, while InParformer excels for univariate forecasting.

Despite these advancements, challenges and gaps remain. There is limited understanding of transformer models' performance across diverse domains and datasets, as most studies focus on tailored models rather than cross-domain evaluations. The lack of standardized benchmarks complicates model assessment, and there is a significant need for more transparent and explainable models. While some models have attempted to improve interpretability through specialized attention mechanisms and causal inference, there is still a need for more transparent and explainable models.

Additionally, the bias analysis reveals transparency issues as a considerable amount of papers does not disclosing the sources of funding and a possible regional bias, as there is predominance of private datasets form only one country. The papers also often only show the best results obtained, with some papers not making the parameters used public. This highlights the need for more rigorous and transparent evaluation practices to ensure robust and unbiased model assessments.

Hierarchical and multi-scale approaches need further exploration to optimize configurations and trade-offs between complexity and performance. Scalability and computational efficiency are major concerns; despite the introduction of sparse and hybrid attention mechanisms, large-scale and real-time implementations remain challenging. Handling missing data and robustness to noisy datasets are critical areas needing attention, as most models assume well-structured, clean data, which is unrealistic in real-world scenarios.

From the future research directions stated by the papers there were some trends. A significant focus is on exploring better sparsity strategies in self-attention mechanisms and adapting models to handle small datasets effectively. This includes developing data-efficient methods, leveraging incremental computations, and adaptively learning hyper-parameters from the data to improve performance across different applications. Enhancing the interpretability of models and studying their learning dynamics is another common theme, with efforts directed towards improving the understanding of model decisions, preventing over-stationarization, and optimizing multi-headed attention mechanisms.

Also, some transformers for time series forecasting were inspired by existing strategies used by transformers in other domains. Now, there are state-of-the-art transformers specifically designed for time series forecasting that have not yet been tested in other fields, such as computer vision (CV) and natural language processing (NLP). Conversely, there are new state-of the art models in different areas that still haven’t been tested for time series forecasting.

For future work specific to this paper, it would be interesting to broaden its scope to transformers of various areas. Examining how these models adapt to different domains could uncover innovations or approached with potential to transfer to other areas.

BIBLIOGRAPHICAL REFERENCES

Abadi, A., Rajabioun, T., & Ioannou, P. A. (2015). Traffic Flow Prediction for Road Transportation Networks With Limited Traffic Data. *IEEE Transactions on Intelligent Transportation Systems*, *16*(2), 653–662. IEEE Transactions on Intelligent Transportation Systems. https://doi.org/10.1109/TITS.2014.2337238

Almarzooqi, A. H., Alhusin, M. O., Nikolakakos, I. P., Husnain, A., & Albeshr, H. M. (2023). Improved NaS Battery State of Charge Estimation by Means of Temporal Fusion Transformer. *2023 IEEE Texas Power and Energy Conference (TPEC)*, 1–6. https://doi.org/10.1109/TPEC56611.2023.10078625

Amadeo, A. J., Siento, J. G., Eikwine, T. A., Diana, & Parmonangan, I. H. (2023). Temporal Fusion Transformer for Multi Horizon Bitcoin Price Forecasting. *2023 IEEE 9th Information Technology International Seminar (ITIS)*, 1–7. https://doi.org/10.1109/ITIS59651.2023.10420330

Ashok, A., Marcotte, É., Zantedeschi, V., Chapados, N., & Drouin, A. (2024). *TACTiS-2: Better, Faster, Simpler Attentional Copulas for Multivariate Time Series* (arXiv:2310.01327). arXiv. https://doi.org/10.48550/arXiv.2310.01327

Ban, G., Chen, Y., Xiong, Z., Zhuo, Y., & Huang, K. (2024). The univariate model for long-term wind speed forecasting based on wavelet soft threshold denoising and improved Autoformer. *Energy*, *290*, 130225. https://doi.org/10.1016/j.energy.2023.130225

Bou, S., Amagasa, T., & Kitagawa, H. (2022). InTrans: Fast Incremental Transformer for Time Series Data Prediction. In C. Strauss, A. Cuzzocrea, G. Kotsis, A. M. Tjoa, & I. Khalil (Eds.), *Database and Expert Systems Applications* (pp. 47–61). Springer International Publishing. https://doi.org/10.1007/978-3-031-12426-6_4

Caldas, F. M., & Soares, C. (2023). A Temporal Fusion Transformer for Long-Term Explainable Prediction of Emergency Department Overcrowding. In I. Koprinska, P. Mignone, R. Guidotti, S. Jaroszewicz, H. Fröning, F. Gullo, P. M. Ferreira, D. Roqueiro, G. Ceddia, S. Nowaczyk, J. Gama, R. Ribeiro, R. Gavaldà, E. Masciari, Z. Ras, E. Ritacco, F. Naretto, A. Theissler, P. Biecek, … S. Pashami (Eds.), *Machine Learning and Principles and Practice of Knowledge Discovery in Databases* (pp. 71–88). Springer Nature Switzerland. https://doi.org/10.1007/978-3-031-23618-1_5

Cao, D., Jia, F., Arik, S. O., Pfister, T., Zheng, Y., Ye, W., & Liu, Y. (2024). *TEMPO: Prompt-based Generative Pre-trained Transformer for Time Series Forecasting* (arXiv:2310.04948). arXiv. https://doi.org/10.48550/arXiv.2310.04948

Cao, H., Huang, Z., Yao, T., Wang, J., He, H., & Wang, Y. (2023). InParformer: Evolutionary Decomposition Transformers with Interactive Parallel Attention for Long-Term Time Series

Forecasting. *Proceedings of the AAAI Conference on Artificial Intelligence*, *37*(6), Article 6. https://doi.org/10.1609/aaai.v37i6.25845

Cao, Y., & Zhao, X. (2023). Dwtformer: Wavelet decomposition Transformer with 2D Variation for Long-Term Series Forecasting. *2023 IEEE 6th Information Technology,Networking,Electronic and Automation Control Conference (ITNEC)*, *6*, 1548–1558. https://doi.org/10.1109/ITNEC56291.2023.10082078

Cen, S., & Lim, C. G. (2024). Multi-Task Learning of the PatchTCN-TST Model for Short-Term Multi-Load Energy Forecasting Considering Indoor Environments in a Smart Building. *IEEE Access*, *12*, 19553–19568. IEEE Access. https://doi.org/10.1109/ACCESS.2024.3355448

Chen, J., Fan, J., Liu, Z., Xiang, J., & Wu, J. (2023). Segformer: Segment-Based Transformer with Decomposition for Long-Term Series Forecasting. *2023 International Joint Conference on Neural Networks (IJCNN)*, 1–8. https://doi.org/10.1109/IJCNN54540.2023.10191412

Chen, P., Zhang, Y., Cheng, Y., Shu, Y., Wang, Y., Wen, Q., Yang, B., & Guo, C. (2023, October 13). *Pathformer: Multi-scale Transformers with Adaptive Pathways for Time Series Forecasting*. The Twelfth International Conference on Learning Representations. https://openreview.net/forum?id=lJkOCMP2aW

Chen, Y., Liu, S., Yang, J., Jing, H., Zhao, W., & Yang, G. (2023). *A Joint Time-frequency Domain Transformer for Multivariate Time Series Forecasting* (arXiv:2305.14649). arXiv. https://doi.org/10.48550/arXiv.2305.14649

Chimmula, V. K. R., & Zhang, L. (2020). Time series forecasting of COVID-19 transmission in Canada using LSTM networks. *Chaos, Solitons & Fractals*, *135*, 109864. https://doi.org/10.1016/j.chaos.2020.109864

Cirstea, R.-G., Guo, C., Yang, B., Kieu, T., Dong, X., & Pan, S. (2022). *Triformer: Triangular, Variable-Specific Attentions for Long Sequence Multivariate Time Series Forecasting--Full Version* (arXiv:2204.13767). arXiv. https://doi.org/10.48550/arXiv.2204.13767

Dai, T., Wu, B., Liu, P., Li, N., Bao, J., Jiang, Y., & Xia, S.-T. (2023, October 13). *Periodicity Decoupling Framework for Long-term Series Forecasting*. The Twelfth International Conference on Learning Representations. https://openreview.net/forum?id=dp27P5HBBt

ddz16. (2024). *Ddz16/TSFpaper* [Computer software]. https://github.com/ddz16/TSFpaper (Original work published 2022)

Deng, B., Wu, Y., Liu, S., & Xu, Z. (2022). Wind Speed Forecasting for Wind Power Production Based on Frequency-Enhanced Transformer. *2022 4th International Conference on Machine Learning, Big Data and Business Intelligence (MLBDBI)*, 151–155. https://doi.org/10.1109/MLBDBI58171.2022.00036

Domínguez-Cid, S., Larios, D. F., Barbancho, J., Salvador, A. G., Quintana-Ortí, E. S., & León, C. (2023). TEFNEN: Transformer for Energy Forecasting in Natural Environment. *2023 3rd International Conference on Electrical, Computer, Communications and Mechatronics Engineering (ICECCME)*, 1–6. https://doi.org/10.1109/ICECCME57830.2023.10253223

Drouin, A., Marcotte, É., & Chapados, N. (2022). *TACTiS: Transformer-Attentional Copulas for Time Series* (arXiv:2202.03528). arXiv. http://arxiv.org/abs/2202.03528

Du, D., Su, B., & Wei, Z. (2023). Preformer: Predictive Transformer with Multi-Scale Segment-Wise Correlations for Long-Term Time Series Forecasting. *ICASSP 2023 - 2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 1–5. https://doi.org/10.1109/ICASSP49357.2023.10096881

Eisenach, C., Patel, Y., & Madeka, D. (2022). *MQTransformer: Multi-Horizon Forecasts with Context Dependent and Feedback-Aware Attention* (arXiv:2009.14799). arXiv. https://doi.org/10.48550/arXiv.2009.14799

Fan, J., Wang, B., & Bian, D. (2023). TEDformer: Temporal Feature Enhanced Decomposed Transformer for Long-term Series Forecasting. *IEEE Access*, 1–1. IEEE Access. https://doi.org/10.1109/ACCESS.2023.3287893

Fernandes, K., Muñoz, A. G., Ramirez-Villegas, J., Agudelo, D., Llanos-Herrera, L., Esquivel, A., Rodriguez-Espinoza, J., & Prager, S. D. (2020). *Improving Seasonal Precipitation Forecasts for Agriculture in the Orinoquía Region of Colombia*. https://doi.org/10.1175/WAF-D-19-0122.1

Garg, L., McClean, S. I., Barton, M., Meenan, B. J., & Fullerton, K. (2012). Intelligent Patient Management and Resource Planning for Complex, Heterogeneous, and Stochastic Healthcare Systems. *IEEE Transactions on Systems, Man, and Cybernetics - Part A: Systems and Humans*, *42*(6), 1332–1345. IEEE Transactions on Systems, Man, and Cybernetics - Part A: Systems and Humans. https://doi.org/10.1109/TSMCA.2012.2210211

Ghimire, S., Nguyen-Huy, T., AL-Musaylh, M. S., Deo, R. C., Casillas-Pérez, D., & Salcedo-Sanz, S. (2023). A novel approach based on integration of convolutional neural networks and echo state network for daily electricity demand prediction. *Energy*, *275*, 127430. https://doi.org/10.1016/j.energy.2023.127430

Gong, M., Zhao, Y., Sun, J., Han, C., Sun, G., & Yan, B. (2022). Load forecasting of district heating system based on Informer. *Energy*, *253*, 124179. https://doi.org/10.1016/j.energy.2022.124179

Gong, Z., Tang, Y., & Liang, J. (2023). *PatchMixer: A Patch-Mixing Architecture for Long-Term Time Series Forecasting* (arXiv:2310.00655). arXiv. https://doi.org/10.48550/arXiv.2310.00655

Guo, L., Li, R., & Jiang, B. (2021). A Data-Driven Long Time-Series Electrical Line Trip Fault Prediction Method Using an Improved Stacked-Informer Network. *Sensors*, *21*(13), Article 13. https://doi.org/10.3390/s21134466

Hachimi, C. E., Belaqziz, S., Khabba, S., Sebbar, B., Dhiba, D., & Chehbouni, A. (2023). Smart Weather Data Management Based on Artificial Intelligence and Big Data Analytics for Precision Agriculture. *Agriculture*, *13*(1), Article 1. https://doi.org/10.3390/agriculture13010095

Ham, S., Kim, S., Lee, N., Kim, P., Eom, I., Lee, B., Tsai, P.-J., Lee, K., & Yoon, C. (2017). Comparison of data analysis procedures for real-time nanoparticle sampling data using classical regression and ARIMA models. *Journal of Applied Statistics*, *44*(4), 685–699. https://doi.org/10.1080/02664763.2016.1182132

He, K., Zhang, X., Ren, S., & Sun, J. (2015). *Deep Residual Learning for Image Recognition* (arXiv:1512.03385). arXiv. https://doi.org/10.48550/arXiv.1512.03385

Honjo, K., Zhou, X., & Shimizu, S. (2022). CNN-GRU Based Deep Learning Model for Demand Forecast in Retail Industry. *2022 International Joint Conference on Neural Networks (IJCNN)*, 1–8. https://doi.org/10.1109/IJCNN55064.2022.9892599

Huang, J., Ma, M., Dai, Y., Hu, J., & Du, S. (2023). DBAFormer: A Double-Branch Attention Transformer for Long-Term Time Series Forecasting. *Human-Centric Intelligent Systems*, *3*(3), 263–274. https://doi.org/10.1007/s44230-023-00037-z

Huang, S., Zhang, J., He, Y., Fu, X., Fan, L., Yao, G., & Wen, Y. (2022). Short-Term Load Forecasting Based on the CEEMDAN-Sample Entropy-BPNN-Transformer. *Energies*, *15*(10), Article 10. https://doi.org/10.3390/en15103659

Huang, Y., & Wu, Y. (2023). Short-Term Photovoltaic Power Forecasting Based on a Novel Autoformer Model. *Symmetry*, *15*(1), Article 1. https://doi.org/10.3390/sym15010238

Jawed, S., & Schmidt-Thieme, L. (2022). GQFormer: A Multi-Quantile Generative Transformer for Time Series Forecasting. *2022 IEEE International Conference on Big Data (Big Data)*, 992–1001. https://doi.org/10.1109/BigData55660.2022.10020927

Jiang, Y., Gao, T., Dai, Y., Si, R., Hao, J., Zhang, J., & Gao, D. W. (2022). Very short-term residential load forecasting based on deep-autoformer. *Applied Energy*, *328*, 120120. https://doi.org/10.1016/j.apenergy.2022.120120

Jittanon, S., Mensin, Y., & Termritthikun, C. (2023). Intelligent Forecasting of Energy Consumption using Temporal Fusion Transformer model. *2023 IEEE International Conference on Cybernetics and Innovations (ICCI)*, 1–5. https://doi.org/10.1109/ICCI57424.2023.10112297

Junankar, T., Sondhi, J. K., & Nair, A. M. (2023). Wheat Yield Prediction using Temporal Fusion Transformers. *2023 2nd International Conference for Innovation in Technology (INOCON)*, 1–6. https://doi.org/10.1109/INOCON57975.2023.10101144

Kim, T., & Kim, H. Y. (2019). Forecasting stock prices with a feature fusion LSTM-CNN model using different representations of the same data. *PLOS ONE*, *14*(2), e0212320. https://doi.org/10.1371/journal.pone.0212320

Lee, S., Hong, J., Liu, L., & Choi, W. (2024). TS-Fastformer: Fast Transformer for Time-series Forecasting. *ACM Transactions on Intelligent Systems and Technology*, *15*(2), 24:1-24:20. https://doi.org/10.1145/3630637

Li, B., Cui, W., Zhang, L., Zhu, C., Wang, W., Tsang, I. W., & Zhou, J. T. (2023). DifFormer: Multi-Resolutional Differencing Transformer With Dynamic Ranging for Time Series Analysis. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *45*(11), 13586–13598. IEEE Transactions on Pattern Analysis and Machine Intelligence. https://doi.org/10.1109/TPAMI.2023.3293516

Li, M., Chen, Q., Li, G., & Han, D. (2022). Umformer: A Transformer Dedicated to Univariate Multistep Prediction. *IEEE Access*, *10*, 101347–101361. IEEE Access. https://doi.org/10.1109/ACCESS.2022.3208139

Li, S., Jin, X., Xuan, Y., Zhou, X., Chen, W., Wang, Y.-X., & Yan, X. (2019). Enhancing the Locality and Breaking the Memory Bottleneck of Transformer on Time Series Forecasting. *Advances in Neural Information Processing Systems*, *32*. https://proceedings.neurips.cc/paper_files/paper/2019/hash/6775a0635c302542da2c32aa19d86be0-Abstract.html

Li, T., Liu, Z., Shen, Y., Wang, X., Chen, H., & Huang, S. (2023). *MASTER: Market-Guided Stock Transformer for Stock Price Forecasting* (arXiv:2312.15235). arXiv. https://doi.org/10.48550/arXiv.2312.15235

Li, W., Meng, X., Chen, C., Mi, H., & Wang, H. (2023). Bidformer: A Transformer-Based Model via Bidirectional Sparse Self-Attention Mechanism for Long Sequence Time-Series Forecasting. *2023 IEEE International Conference on Systems, Man, and Cybernetics (SMC)*, 4076–4082. https://doi.org/10.1109/SMC53992.2023.10394310

Li, Y., Lu, X., Xiong, H., Tang, J., Su, J., Jin, B., & Dou, D. (2023). *Towards Long-Term Time-Series Forecasting: Feature, Pattern, and Distribution* (arXiv:2301.02068). arXiv. https://doi.org/10.48550/arXiv.2301.02068

Li, Y., Qi, S., Li, Z., Rao, Z., Pan, L., & Xu, Z. (2023). *SMARTformer: Semi-Autoregressive Transformer with Efficient Integrated Window Attention for Long Time Series Forecasting*. *3*, 2169–2177. https://doi.org/10.24963/ijcai.2023/241

Li, Y., Wang, H., Li, J., Liu, C., & Tan, J. (2022). ACT: Adversarial Convolutional Transformer for Time Series Forecasting. *2022 International Joint Conference on Neural Networks (IJCNN)*, 1–8. https://doi.org/10.1109/IJCNN55064.2022.9892791

Liang, Y., Xia, Y., Ke, S., Wang, Y., Wen, Q., Zhang, J., Zheng, Y., & Zimmermann, R. (2022, November 29). *AirFormer: Predicting Nationwide Air Quality in China with Transformers*. arXiv.Org. https://arxiv.org/abs/2211.15979v1

Liao, H., & Radhakrishnan, K. K. (2022). Short-Term Load Forecasting with Temporal Fusion Transformers for Power Distribution Networks. *2022 IEEE Sustainable Power and Energy Conference (iSPEC)*, 1–5. https://doi.org/10.1109/iSPEC54162.2022.10033079

Lim, B., Arik, S. O., Loeff, N., & Pfister, T. (2020). *Temporal Fusion Transformers for Interpretable Multi-horizon Time Series Forecasting* (arXiv:1912.09363). arXiv. https://doi.org/10.48550/arXiv.1912.09363

Lin, T., Wang, Y., Liu, X., & Qiu, X. (2021). *A Survey of Transformers* (arXiv:2106.04554). arXiv. https://doi.org/10.48550/arXiv.2106.04554

Lin, Y., Koprinska, I., & Rana, M. (2021). *SSDNet: State Space Decomposition Neural Network for Time Series Forecasting* (arXiv:2112.10251). arXiv. https://doi.org/10.48550/arXiv.2112.10251

Liu, D., Wang, Y., Liu, C., Yuan, X., & Yang, C. (2024). Multirate-Former: An Efficient Transformer-Based Hierarchical Network for Multistep Prediction of Multirate Industrial Processes. *IEEE Transactions on Instrumentation and Measurement*, *73*, 1–13. IEEE Transactions on Instrumentation and Measurement. https://doi.org/10.1109/TIM.2023.3331407

Liu, S., Yu, H., Liao, C., Li, J., Lin, W., Liu, A. X., & Dustdar, S. (2022, February 28). *Pyraformer: Low-Complexity Pyramidal Attention for Long-Range Time Series Modeling and Forecasting*. International Conference on Learning Representations. https://openreview.net/forum?id=0EXmFzUn5I

Liu, Y., Hu, T., Zhang, H., Wu, H., Wang, S., Ma, L., & Long, M. (2024). *iTransformer: Inverted Transformers Are Effective for Time Series Forecasting* (arXiv:2310.06625). arXiv. https://doi.org/10.48550/arXiv.2310.06625

Liu, Y., Wang, Z., Yu, X., Chen, X., & Sun, M. (2022). Memory-based Transformer with shorter window and longer horizon for multivariate time series forecasting. *Pattern Recognition Letters*, *160*, 26–33. https://doi.org/10.1016/j.patrec.2022.05.010

Liu, Y., Wu, H., Wang, J., & Long, M. (2023). *Non-stationary Transformers: Exploring the Stationarity in Time Series Forecasting* (arXiv:2205.14415). arXiv. https://doi.org/10.48550/arXiv.2205.14415

Liu, Z., Cao, Y., Xu, H., Huang, Y., He, Q., Chen, X., Tang, X., & Liu, X. (2024). Hidformer: Hierarchical dual-tower transformer using multi-scale mergence for long-term time series forecasting. *Expert Systems with Applications*, *239*, 122412. https://doi.org/10.1016/j.eswa.2023.122412

Ma, J., & Dan, J. (2023). Long-Term Structural State Trend Forecasting Based on an FFT–Informer Model. *Applied Sciences*, *13*(4), Article 4. https://doi.org/10.3390/app13042553

Ma, S., Zhang, T., Zhao, Y.-B., Kang, Y., & Bai, P. (2023). TCLN: A Transformer-based Conv-LSTM network for multivariate time series forecasting. *Applied Intelligence*, *53*(23), 28401–28417. https://doi.org/10.1007/s10489-023-04980-z

Madhusudhanan, K., Burchert, J., Duong-Trung, N., Born, S., & Schmidt-Thieme, L. (2022). *Yformer: U-Net Inspired Transformer Architecture for Far Horizon Time Series Forecasting* (arXiv:2110.08255). arXiv. https://doi.org/10.48550/arXiv.2110.08255

Ni, Z., Yu, H., Liu, S., Li, J., & Lin, W. (2024). *BasisFormer: Attention-based Time Series Forecasting with Learnable and Interpretable Basis* (arXiv:2310.20496). arXiv. https://doi.org/10.48550/arXiv.2310.20496

Nie, Y., Nguyen, N. H., Sinthong, P., & Kalagnanam, J. (2023). *A Time Series is Worth 64 Words: Long-term Forecasting with Transformers* (arXiv:2211.14730). arXiv. http://arxiv.org/abs/2211.14730

Nivron, O., Parthipan, R., & Wischik, D. J. (2023). *Taylorformer: Probabilistic Predictions for Time Series and other Processes* (arXiv:2305.19141). arXiv. http://arxiv.org/abs/2305.19141

Ouyang, Z., Jabloun, M., & Ravier, P. (2023a). Rankformer: Leveraging Rank Correlation for Transformer-based Time Series Forecasting. *2023 IEEE Statistical Signal Processing Workshop (SSP)*, 85–89. https://doi.org/10.1109/SSP53291.2023.10207937

Ouyang, Z., Jabloun, M., & Ravier, P. (2023b). STLformer: Exploit STL Decomposition and Rank Correlation for Time Series Forecasting. *2023 31st European Signal Processing Conference (EUSIPCO)*, 1405–1409. https://doi.org/10.23919/EUSIPCO58844.2023.10290126

Pan, K., Lu, J., Li, J., & Xu, Z. (2023). A Hybrid Autoformer Network for Air Pollution Forecasting Based on External Factor Optimization. *Atmosphere*, *14*(5), Article 5. https://doi.org/10.3390/atmos14050869

Phetrittikun, R., Suvirat, K., Pattalung, T. N., Kongkamol, C., Ingviya, T., & Chaichulee, S. (2021). Temporal Fusion Transformer for forecasting vital sign trajectories in intensive care patients. *2021 13th Biomedical Engineering International Conference (BMEiCON)*, 1–5. https://doi.org/10.1109/BMEiCON53485.2021.9745215

Qu, K., Si, G., Shan, Z., Wang, Q., Liu, X., & Yang, C. (2024). Forwardformer: Efficient Transformer With Multi-Scale Forward Self-Attention for Day-Ahead Load Forecasting. *IEEE*

Transactions on Power Systems, *39*(1), 1421–1433. IEEE Transactions on Power Systems. https://doi.org/10.1109/TPWRS.2023.3266369

Ribeiro, A. H., Tiels, K., Aguirre, L. A., & Schön, T. B. (2020). *Beyond exploding and vanishing gradients: Analysing RNN training using attractors and smoothness* (arXiv:1906.08482). arXiv. https://doi.org/10.48550/arXiv.1906.08482

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

Shabani, M. A., Abdi, A. H., Meng, L., & Sylvain, T. (2023, February 1). *Scaleformer: Iterative Multi-scale Refining Transformers for Time Series Forecasting*. The Eleventh International Conference on Learning Representations. https://openreview.net/forum?id=sCrnllCtjoE

Shen, L., & Wang, Y. (2022). TCCT: Tightly-Coupled Convolutional Transformer on Time Series Forecasting. *Neurocomputing*, *480*, 131–145. https://doi.org/10.1016/j.neucom.2022.01.039

Shen, L., Wei, Y., Wang, Y., & Li, H. (2023). *Take an Irregular Route: Enhance the Decoder of Time-Series Forecasting Transformer* (arXiv:2312.05792). arXiv. https://doi.org/10.48550/arXiv.2312.05792

Su, C.-H., & Cheng, C.-H. (2016). A hybrid fuzzy time series model based on ANFIS and integrated nonlinear feature selection method for forecasting stock. *NEUROCOMPUTING*, *205*, 264–273. https://doi.org/10.1016/j.neucom.2016.03.068

Su, H., Wang, X., & Qin, Y. (2021). AGCNT: Adaptive Graph Convolutional Network for Transformer-based Long Sequence Time-Series Forecasting. *Proceedings of the 30th ACM International Conference on Information & Knowledge Management*, 3439–3442. https://doi.org/10.1145/3459637.3482054

Su, L., Zuo, X., Li, R., Wang, X., Zhao, H., & Huang, B. (2023). *A Systematic Review for Transformer-based Long-term Series Forecasting* (arXiv:2310.20218). arXiv. https://doi.org/10.48550/arXiv.2310.20218

Tang, B., & Matteson, D. S. (2021). Probabilistic Transformer For Time Series Analysis. *Advances in Neural Information Processing Systems*, *34*, 23592–23608. https://proceedings.neurips.cc/paper_files/paper/2021/hash/c68bd9055776bf38d8fc43c0ed283678-Abstract.html

Tay, Y., Dehghani, M., Bahri, D., & Metzler, D. (2022). *Efficient Transformers: A Survey* (arXiv:2009.06732). arXiv. https://doi.org/10.48550/arXiv.2009.06732

Tevare, V., & Revankar, P. S. (2023). Forecasting Stock Prices with Stack Transformer. *2023 International Conference on Circuit Power and Computing Technologies (ICCPCT)*, 1262–1269. https://doi.org/10.1109/ICCPCT58313.2023.10245652

Thwal, C. M., Tun, Y. L., Kim, K., Park, S.-B., & Hong, C. S. (2023). Transformers with Attentive Federated Aggregation for Time Series Stock Forecasting. *2023 International Conference on Information Networking (ICOIN)*, 499–504. https://doi.org/10.1109/ICOIN56518.2023.10048928

Tong, G., Ge, Z., & Peng, D. (2024). RSMformer: An efficient multiscale transformer-based framework for long sequence time-series forecasting. *Applied Intelligence*, *54*(2), 1275–1296. https://doi.org/10.1007/s10489-023-05250-8

Tong, J., Xie, L., & Zhang, K. (2023). Probabilistic Decomposition Transformer for Time Series Forecasting. In *Proceedings of the 2023 SIAM International Conference on Data Mining (SDM)* (pp. 478–486). Society for Industrial and Applied Mathematics. https://doi.org/10.1137/1.9781611977653.ch54

van Heerden, L., van Staden, C., & Vermeulen, H. J. (2023). Temporal Fusion Transformer for Day-Ahead Wind Power Forecasting in the South African Context. *2023 IEEE International Conference on Environment and Electrical Engineering and 2023 IEEE Industrial and Commercial Power Systems Europe (EEEIC / I&CPS Europe)*, 1–5. https://doi.org/10.1109/EEEIC/ICPSEurope57605.2023.10194737

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L., & Polosukhin, I. (2023). *Attention Is All You Need* (arXiv:1706.03762). arXiv. http://arxiv.org/abs/1706.03762

Wang, N., & Zhao, X. (2023a). Enformer: Encoder-Based Sparse Periodic Self-Attention Time-Series Forecasting. *IEEE Access*, *11*, 112004–112014. IEEE Access. https://doi.org/10.1109/ACCESS.2023.3322957

Wang, N., & Zhao, X. (2023b). Time Series Forecasting Based on Convolution Transformer. *IEICE Transactions on Information and Systems*, *E106.D*(5), 976–985. https://doi.org/10.1587/transinf.2022EDP7136

Wang, X., Liu, H., Du, J., Yang, Z., & Dong, X. (2023). CLformer: Locally grouped auto-correlation and convolutional transformer for long-term multivariate time series forecasting. *Engineering Applications of Artificial Intelligence*, *121*, 106042. https://doi.org/10.1016/j.engappai.2023.106042

Wang, X., Liu, H., Yang, Z., Du, J., & Dong, X. (2023). CNformer: A convolutional transformer with decomposition for long-term multivariate time series forecasting. *Applied Intelligence*, *53*(17), 20191–20205. https://doi.org/10.1007/s10489-023-04496-6

Wang, X., Xia, M., & Deng, W. (2023). MSRN-Informer: Time Series Prediction Model Based on Multi-Scale Residual Network. *IEEE Access*, *11*, 65059–65065. IEEE Access. https://doi.org/10.1109/ACCESS.2023.3289824

Wang, Y., Long, H., Zheng, L., & Shang, J. (2024). Graphformer: Adaptive graph correlation transformer for multivariate long sequence time series forecasting. *Knowledge-Based Systems*, *285*, 111321. https://doi.org/10.1016/j.knosys.2023.111321

Wang, Y., Zhu, J., & Kang, R. (2023). DESTformer: A Transformer Based on Explicit Seasonal–Trend Decomposition for Long-Term Series Forecasting. *Applied Sciences*, *13*(18), Article 18. https://doi.org/10.3390/app131810505

Wang, Z., Chen, Z., Yang, Y., Liu, C., Li, X., & Wu, J. (2023). A hybrid Autoformer framework for electricity demand forecasting. *Energy Reports*, *9*, 3800–3812. https://doi.org/10.1016/j.egyr.2023.02.083

Wang, Z., Ran, H., Ren, J., & Sun, M. (2024). PWDformer: Deformable transformer for long-term series forecasting. *Pattern Recognition*, *147*, 110118. https://doi.org/10.1016/j.patcog.2023.110118

Wen, Q. (2024a). *Qingsongedu/awesome-AI-for-time-series-papers* [Computer software]. https://github.com/qingsongedu/awesome-AI-for-time-series-papers (Original work published 2022)

Wen, Q. (2024b). *Qingsongedu/time-series-transformers-review* [Computer software]. https://github.com/qingsongedu/time-series-transformers-review (Original work published 2022)

Wu, H., Li, R., Sheng, G., & Wilson, D. (2023). D-Transformer: A Deep Learning Model for Time Series Prediction. *2023 Asia-Pacific Conference on Image Processing, Electronics and Computers (IPEC)*, 313–316. https://doi.org/10.1109/IPEC57296.2023.00061

Wu, H., Xu, J., Wang, J., & Long, M. (2021). *Autoformer: Decomposition Transformers with Auto-Correlation for Long-Term Series Forecasting* (arXiv:2106.13008). arXiv. http://arxiv.org/abs/2106.13008

Wu, H., Xu, J., Wang, J., & Long, M. (2022). *Autoformer: Decomposition Transformers with Auto-Correlation for Long-Term Series Forecasting* (arXiv:2106.13008). arXiv. https://doi.org/10.48550/arXiv.2106.13008

Wu, S., Xiao, X., Ding, Q., Zhao, P., Wei, Y., & Huang, J. (2020). Adversarial Sparse Transformer for Time Series Forecasting. *Advances in Neural Information Processing Systems*, *33*, 17105–17115. https://proceedings.neurips.cc/paper/2020/hash/c6b8c8d762da15fa8dbbdfb6baf9e260-Abstract.html

Xie, H., & He, J. (2022). Transformer Based on Deconstruction and Dot Product for Time Series Forecasting of Air Quality. *2022 International Conference on Computers and Artificial Intelligence Technologies (CAIT)*, 31–37. https://doi.org/10.1109/CAIT56099.2022.10072078

xiyuanzh. (2024). *Xiyuanzh/time-series-papers* [Computer software]. https://github.com/xiyuanzh/time-series-papers (Original work published 2021)

Yang, Y., & Lu, J. (2023). Foreformer: An enhanced transformer-based framework for multivariate time series forecasting. *Applied Intelligence*, *53*(10), 12521–12540. https://doi.org/10.1007/s10489-022-04100-3

Yang, Z., Liu, L., Li, N., & Tian, J. (2022). Time Series Forecasting of Motor Bearing Vibration Based on Informer. *Sensors*, *22*(15), Article 15. https://doi.org/10.3390/s22155858

Ye, J., Zhao, B., & Liu, D. (2023). Temporal Decomposition Transformer for Probabilistic Energy Forecasting. *2022 First International Conference on Cyber-Energy Systems and Intelligent Energy (ICCSIE)*, 1–5. https://doi.org/10.1109/ICCSIE55183.2023.10175223

Yu, C., Wang, F., Shao, Z., Sun, T., Wu, L., & Xu, Y. (2023). *DSformer: A Double Sampling Transformer for Multivariate Time Series Long-term Prediction* (arXiv:2308.03274). arXiv. https://doi.org/10.48550/arXiv.2308.03274

Zeng, A., Chen, M., Zhang, L., & Xu, Q. (2022). *Are Transformers Effective for Time Series Forecasting?* (arXiv:2205.13504). arXiv. https://doi.org/10.48550/arXiv.2205.13504

Zeng, P., Hu, G., Zhou, X., Li, S., & Liu, P. (2023). Seformer: A long sequence time-series forecasting model based on binary position encoding and information transfer regularization. *Applied Intelligence*, *53*(12), 15747–15771. https://doi.org/10.1007/s10489-022-04263-z

Zeng, P., Hu, G., Zhou, X., Li, S., Liu, P., & Liu, S. (2022). Muformer: A long sequence time-series forecasting model based on modified multi-head attention. *Knowledge-Based Systems*, *254*, 109584. https://doi.org/10.1016/j.knosys.2022.109584

Zhang, C., Zhao, L., Yin, Z., & Zhang, Z. (2023). Causalformer: Causal Discovery-based Transformer for Multivariate Time Series Forecasting. *2023 16th International Congress on Image and Signal Processing, BioMedical Engineering and Informatics (CISP-BMEI)*, 1–6. https://doi.org/10.1109/CISP-BMEI60920.2023.10373365

Zhang, X., Jin, X., Gopalswamy, K., Gupta, G., Park, Y., Shi, X., Wang, H., Maddix, D. C., & Wang, Y. (2022). *First De-Trend then Attend: Rethinking Attention for Time-Series Forecasting* (arXiv:2212.08151). arXiv. https://doi.org/10.48550/arXiv.2212.08151

Zhang, Y., Ma, L., Pal, S., Zhang, Y., & Coates, M. (2024). *Multi-resolution Time-Series Transformer for Long-term Forecasting* (arXiv:2311.04147). arXiv. https://doi.org/10.48550/arXiv.2311.04147

Zhang, Y., Wu, R., Dascalu, S. M., & Harris, F. C. (2024). Multi-Scale Transformer Pyramid Networks for Multivariate Time Series Forecasting. *IEEE Access*, *12*, 14731–14741. IEEE Access. https://doi.org/10.1109/ACCESS.2024.3357693

Zhang, Z., Meng, L., & Gu, Y. (2024). SageFormer: Series-Aware Framework for Long-Term Multivariate Time Series Forecasting. *IEEE Internet of Things Journal*, 1–1. IEEE Internet of Things Journal. https://doi.org/10.1109/JIOT.2024.3363451

Zhong, Y., Zhou, J., Zhang, T., Yang, J., Li, P., & Deng, D. (2023). NTformer: Near Time Transformer for multi-step prediction of wellhead pressure in fracturing operations. *Geoenergy Science and Engineering*, *231*, 212407. https://doi.org/10.1016/j.geoen.2023.212407

Zhou, H., Zhang, S., Peng, J., Zhang, S., Li, J., Xiong, H., & Zhang, W. (2020). *Informer: Beyond Efficient Transformer for Long Sequence Time-Series Forecasting* (arXiv:2012.07436). arXiv. https://doi.org/10.48550/arXiv.2012.07436

Zhou, H., Zhang, S., Peng, J., Zhang, S., Li, J., Xiong, H., & Zhang, W. (2021). Informer: Beyond Efficient Transformer for Long Sequence Time-Series Forecasting. *Proceedings of the AAAI Conference on Artificial Intelligence*, *35*(12), Article 12. https://doi.org/10.1609/aaai.v35i12.17325

Zhou, T., Ma, Z., Wen, Q., Wang, X., Sun, L., & Jin, R. (2022). *FEDformer: Frequency Enhanced Decomposed Transformer for Long-term Series Forecasting* (arXiv:2201.12740). arXiv. http://arxiv.org/abs/2201.12740

Zhu, J., Zhao, Z., Zheng, X., An, Z., Guo, Q., Li, Z., Sun, J., & Guo, Y. (2023). Time-Series Power Forecasting for Wind and Solar Energy Based on the SL-Transformer. *Energies*, *16*(22), Article 22. https://doi.org/10.3390/en16227610

Zhu, S., Zheng, J., & Ma, Q. (2023). MR-Transformer: Multiresolution Transformer for Multivariate Time Series Prediction. *IEEE Transactions on Neural Networks and Learning Systems*, 1–13. IEEE Transactions on Neural Networks and Learning Systems. https://doi.org/10.1109/TNNLS.2023.3327416

APPENDIX A (COMPLEXITY EQUATIONS)

Lowering the computational complexity without sacrificing information utilization is a big challenge, and different transformers approach this differently, arriving at different equations.

The **vanilla** transformer's complexity is $\mathcal{O}(L)^2$ - where L is the sequence length - because each query attends to all keys, requiring $L \times L$ computations.

Sparse Attention Mechanism

(S. Li et al., 2019), in **LogSparse**, achieve a complexity of $\mathcal{O}(L(\log L)^2)$ through a sparse attention mechanism. This mechanism reduces the computational load by computing attention scores only for a set of positions that are logarithmically spaced.

(H. Zhou et al., 2020), in **Informer**, reach a complexity of $\mathcal{O}(L\ \log L)$. It focuses on on the most informative queries by using the ProbSparse mechanism which identifies a sparse subset of key-query pairs with the highest attention scores.

(G. Tong et al., 2024), in **RSMformer**, achieve $\mathcal{O}(L\ \log L)$, through the introduction of the Residual Sparse Attention (RSA) mechanism, that focuses on the most dominant queries, by leveraging sparse matrix operations and residual connections.

(W. Li et al., 2023), in **Bidformer**, reach $\mathcal{O}((\log L)^2)$ through the introduction of the bidirectional sparse self-attention mechanism (Bis-Attention). The Bis-Attention mechanism works by sparsifying the self-attention matrix, focusing only on the most relevant parts of the input sequence.

Frequency Domain and Fourier-based Methods

(H. Wu et al., 2021), in **Autoformer**, achieve a complexity of $\mathcal{O}(L\ \log L)$ by leveraging the Fast Fourier Transform (FFT) to compute the autocorrelation efficiently. The process involves calculating the autocorrelation of the time series to identify the most significant periodic lags, aggregating similar sub-series efficiently.

(T. Zhou et al., 2022), in **FEDformer**, reach a complexity of $\mathcal{O}(L)$ by transforming the input data into the frequency domain using Fourier and wavelet transforms and then processing only a fixed number of selected frequency components.

(Ouyang et al., 2023a), in **Rankformer**, arrive at $\mathcal{O}(L\ \log L)$, by integrating the Fast Fourier Transform (FFT) and the Wiener-Khinchin theorem for calculating the Autocorrelation Function (ACF) of ranked time series data.

(Y. Wang et al., 2023), in **DESTformer**, achieve $\mathcal{O}(L)$ by employing the Fast Fourier Transform (FFT) is used to decompose the time series data into seasonal and trend components, and by

further sampling the frequency components and applying the specialized Multi-View and Multi-Scale Attention mechanisms.

Patch-based and Segmentation Approaches

(Cirstea et al., 2022), in **Triformer**, achieve linear complexity, $\mathcal{O}(L)$. This is achieved through the introduction of the "Patch Attention" mechanism, dividing the time series into smaller patches, and within each patch, a pseudo timestamp is used to compute attention scores. This reduces the number of attention computations needed within each patch to linear complexity. Additionally, the Triformer employs a triangular structure for stacking multiple layers of Patch Attention, where the size of each subsequent layer decreases exponentially, thus maintaining the overall linear complexity for the entire model.

(Nie et al., 2023), in **PatchTST**, achieve $\mathcal{O}(L/S)^2)$, by segmenting the input time series into patches, effectively reducing the number of input tokens L by a factor related to the patch length P and stride S, significantly reducing the computational load and memory requirements when $S > 1$.

(J. Chen et al., 2023), in **Segformer**, reach $\mathcal{O}(\left(\frac{L}{l}\right)^2)$, where l is the segment length, by dividing the entire sequence into smaller segments of length l and focusing the attention mechanism on these segments rather than the entire sequence.

(Lee et al., 2024), in **TS-Fastformer**, reduce the complexity to $\mathcal{O}(\left(\frac{L}{l}\right)^2)$, by segmenting the input sequence into smaller sub-windows of length l.

(Shen et al., 2023), in **FPPformer**, reach $\mathcal{O}(L \times P)$ determined by its use of patch-wise and element-wise attention mechanisms, where P is the size of each patch.

Hierarchical and Multi-scale Attention Mechanisms

(Tang & Matteson, 2021), in **ProTran**, achieve a complexity of $\mathcal{O}(L^2\, d)$, where T is the length of the time series and d is the dimensionality of the latent space. This complexity arises from the use of the attention mechanisms that involves operations on matrices of size $T \times d$. Each attention operation requires calculating pairwise interactions between all time steps, leading to a quadratic dependence on the sequence length T. Additionally, the hierarchical structure with multiple layers of latent variables adds a linear factor to the complexity.

(S. Liu et al., 2022), in **Pyraformer**, achieve linear complexity, $\mathcal{O}(L)$, by limiting the scope of attention to local neighborhoods and summarizing information at progressively coarser levels. This is done through its hierarchical and pyramidal structure, which employing a sparser attention mechanism across multiple scales. Each node in this structure attends only to a selected subset of other nodes, both within its own scale and across scales.

Simplified and Local Attention Mechanisms

(Drouin et al., 2022), in **TACTiS**, reach a complexity of $\mathcal{O}(n^2 \cdot l_{max} + n \cdot {l_{max}}^2)$, where n represents the number of different time series and l_{max} is the length of the longest time series. This complexity is achieved through the model's strategic use of attention mechanisms across different layers; attention is applied in two dimensions - across different variables and across different time steps within each variable.

(Bou et al., 2022), in **InTrans**, reach $\mathcal{O}(S)$, by improving computational efficiency with incremental computations, where S is the length of the non-overlapping part of the input, reducing redundant operations.

(Y. Li, Lu, et al., 2023), in **Conformer**, reach $\mathcal{O}(L)$ by introducing the sliding-window attention mechanism, where each element only attends to a fixed number of nearby elements within a predefined window, rather than the entire sequence.

(Y. Yang & Lu, 2023), in **Foreformer**, reach $\mathcal{O}(L^2 \cdot d)$, where d is the dimensionality of each input vector. This is due to the self-attention mechanism, where each element in the sequence attends to every other element, resulting in an $L \times L$ matrix of attention scores. Therefore, the total computational complexity for the attention mechanism becomes $\mathcal{O}(L^2 \cdot d)$, reflecting the quadratic relationship with the sequence length and the linear relationship with the dimensionality of the data.

Discussion

Sparse attention mechanisms significantly cut down computational complexity by focusing attention only on the most relevant parts of the input sequence. Frequency-domain methods utilize Fast Fourier Transform (FFT) and related techniques to transform and process data efficiently. Patch-based and segmentation approaches break down sequences into smaller segments or patches, reducing the number of computations required. Hierarchical and multi-scale attention mechanisms achieve efficiency by organizing attention hierarchically across different scales or layers. Lastly, simplified and local attention mechanisms focus on local neighbourhoods or incremental computations to streamline the attention process. This trend towards diverse, specialized methods indicates a robust effort to balance computational efficiency with effective information utilization in transformer architectures.

APPENDIX B (PAPERS' STUDY EXTRA)

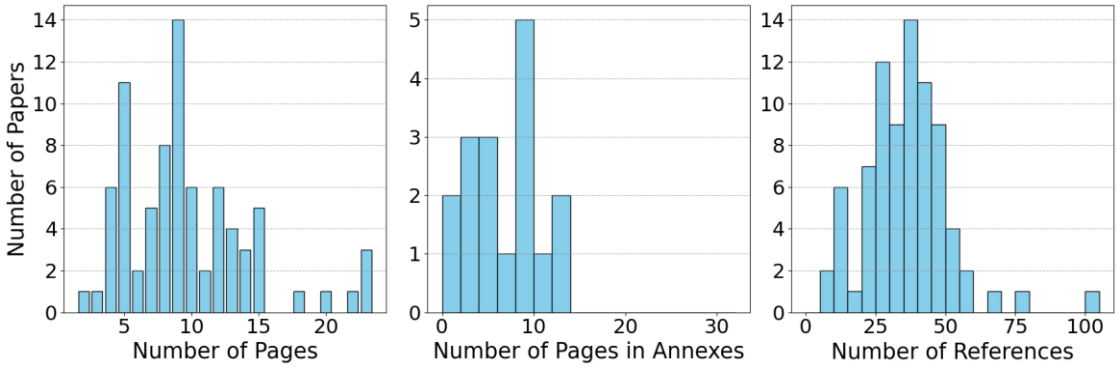

Figure 0.1 – Bar plot of the number of pages of the papers, histogram of the number of pages in the annexes, histogram of the number of references

Most papers have around 10 pages, 79 of the papers have no annexes and the ones that do mostly have from 4 to 10 pages – the annexes mostly focus on proofs, sensitivity studies, ablation studies and details about training, tuning, and the datasets. The Distribution of the number of references is centered around 40.

56 of the papers were published in conference proceedings and the 46 remaining ones were published in books. The most common conferences were ICLR, NeurIPS, AAAI, CIKM, ACML, IJCAI, IJCNN, and 27 unique conferences. The most common books were Applied Intelligence, IEEE Access and other IEEE books, Energy, Sensors, and 13 unique books.

The authors of each paper were retrieved and then analysed. 22 of the authors have contributed to 2 of the papers, and the remaining 392 only contributed to one.

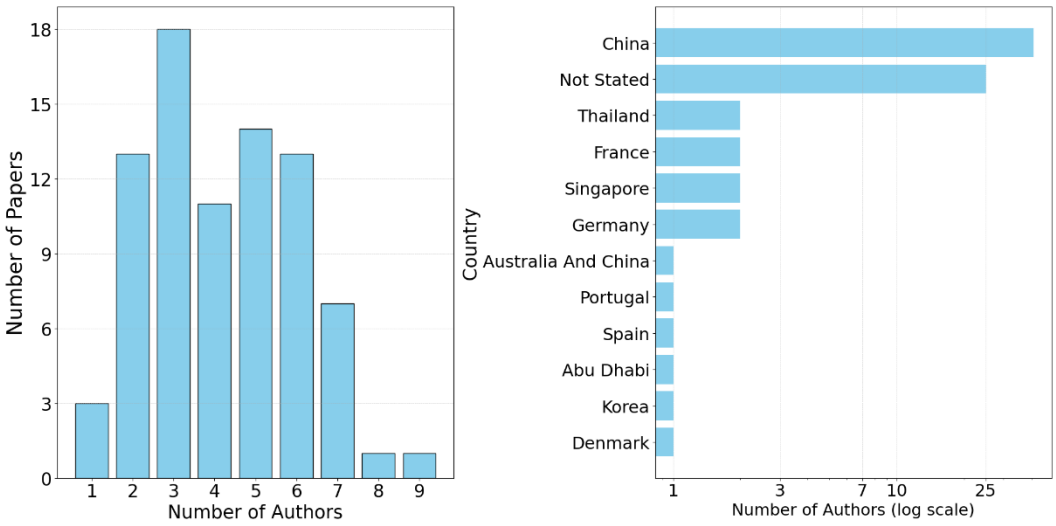

Figure 0.2 - Bar Plot of the Authors count per Paper; Distribution of the Authors Based on the Country

There are only 3 papers with a single author, most are published through a collaboration, but teams of more then seven are rare. This might suggest that the research is too complex to be done alone and that coordinating with more than 6 people becomes increasingly complex.

More than half of the Authors are in China, with a significant amount in the USA, and the rest distributed among other 21 countries.

APPENDIX C (TABLE OF SUMMARY OF CHANGES)

Table 0.1 has the list of the analysed transformers, the reference to the respective paper, if the code is publicly available, the summary of the changes in key-words and the motivations/ issues addressed by the proposed transformer.

Table 0.1 – Summary of the Changes in the Analysed Transformers, in chronological order of publication

Transformer	Reference	Code	Summed up changes	Issues Addressed and Motivations
Logtrans/Logsparse	(S. Li et al., 2019)	Yes	Sparse Convolutional Self Attention + Rolling Window	Locality-Agnostic + Memory Bottleneck
TFT	(Lim et al., 2020)	Yes	Static Covariate Encoders (GRN) + Gating Mechanisms + Sequence-to-Sequence Layer + Interpretable Multi-Head Attention + Temporal Fusion Decoder + Variable Selection Networks + Quantile Predictions	Interpretability + Covariates
AST	(S. Wu et al., 2020)	Yes	Covariates + α-entmax Sparse Attention + Adversarial training + GAN framework	Error Acumulation due to Auto-Regression + Probabilistic Forecasting
Informer	(H. Zhou et al., 2020)	Yes	ProbSparse self-attention (KL Divergence)+ Generative Decoder + Self-attention distilling (Max Pooling)	Memory Bottleneck + Speed Plunge
Stacked-Informer Network	(Guo et al., 2021)	No	Stacked Structure + Probabilistic Sparse Attention + Self-Attention Distilling + Gradient Centralization and Adam Optimizer + Generative Style Decoder	Accuracy and Speed on PV Forecasting
TCCT	(Shen & Wang, 2022)	Yes	CPSAttention + Self-attention Distilling (Dilated Casual Convolution) + Passthrough Mechanism	Memory Bottleneck + Insensitivity to Local Context + Existing CNN hybrids are Loosely-Coupled

AGCNT	(H. Su et al., 2021)	No	ProbSparse Adaptive Graph Self-Attention + Distilling + Generative Inference + Convolution	Probsparse Destroys Relationships Between Sequences – Doesn't Capture Correlation Between Sequences
SSDNet	(Y. Lin et al., 2021)	No	Covariates + SSM + NLL and MAE Loss Function + Probabilistic Forecasting	Interpretability + Interest in Kalman Filter with the Problem of Being Computationally Expensive
Autoformer	(H. Wu et al., 2021)	Yes	Decomposition Block + Auto-Correlation Inner Block	Memory Bottleneck + Capturing Intricate Temporal Patterns
ProTran	(Tang & Matteson, 2021)	No	SSM + Probabilistic Modeling + Stochastic Latent Variables + Variational Inference + KL Divergence and Variational Inference Loss Function	Capturing Non-Markovian Dynamics and Non-Linear Interactions
Pyraformer	(S. Liu et al., 2022)	Yes	Pyramidal Attention Model (Sparse) + Coarser-Scale Construction Module	To Build a Flexible but Parsimonious Model that Can Capture a Wide Range of Temporal Dependencies
CEEMDAN-Sample Entropy-BPNN-Transformer	(S. Huang et al., 2022)	No	Complete Ensemble Empirical Mode Decomposition with Adaptive Noise + Sample Entropy + Back Propagation Neural Network	Power Load Being Stochastic
FEDformer	(T. Zhou et al., 2022)	Yes	Frequency Enhanced Block + Discrete Wavelet Transform + Frequency Enhanced Attention + Mixture Of Expert Decomposition	Efficiency + Unable to Capture Trend
TACTIS	(Drouin et al.,	Yes	Covariates + timestamps +	Estimating the

	2022)		Time Series Tokens + Non-Parametric Copula + Attentional Copula Mechanism	Joint Predictive Distribution of High-dimensional Multivariate Time Series + Missing Values
ACT	(S. Wu et al., 2020)	No	Convolutional Attention Block + Decoding Mode + Adversarial training	Error Accumulation + Insensitive to Local Context + Not Modelling Distribution of Data
Triformer	(Cirstea et al., 2022)	Yes	Patch Attention (Patches and Pseudo Timestamps) + Recurrence + Triangular Stacking + Variable Specific model	Memory Bottleneck + Difficulty Capturing Different Variables' Temporal Dynamics
SWLHT	(Y. Liu et al., 2022)	Yes	Overlapping Segments + Memory-Efficient Module + Enhanced Attention Mechanism + Linear Autoregressive Module + Iterative Concatenation of Single-Step Forecasts	Complex and Non-linear Interdependence Between Time Steps and Different Variables + Prediction Fragmentation + Insensitivity to Data Scale + Lack of Data Training
InTrans	(Bou et al., 2022)	No	Incremental Positional Embedding + Incremental Temporal Embedding + Incremental Value Embedding (Convolution) + Incremental Self-attention	Informer Being Unable to Incrementally Learn + Long Time Series Forecasting
Yformer	(Madhusudhanan et al., 2022)	Yes	Downscaling + Upscaling + Contracting ProbSparse Self-Attention Blocks + Expanding ProbSparse Cross-Attention Block + Auxiliary Reconstruction Loss	Difficulties in Capturing Long-Range Dependencies Effectively + Sparse Architecture

				Leads to a Mismatch Between the Resolutions of the Input and Output
Umformer	(M. Li et al., 2022)	No	Univariate Feature Decomposition and Preprocessing + Feature Selection (GLUV3) + Shared Double-heads Probsparse Attention (SDHPA)	Univariate Forecasting
Muformer	(P. Zeng et al., 2022)	No	Multi Perceptual Domain + Multi-Granularity Attention Head Mechanism + Similar Pruning Heads Mechanism (KL Divergence)	Redundant Input Information
DDPformer	(Xie & He, 2022)	No	Deconstruction and Dot Product Attention Mechanism + Channel-Dependent Attention + Power Normalization + Layer Normalization	Nonlinear and Dynamic Characteristics of the Multivariate Air Quality Data
Non-Stationary Transformer	(Y. Liu et al., 2023)	Yes	Series Stationarization (Normalization Module + De-normalization Module) + De-stationary Attention	Difficulty Forecasting Non-Stationary Data in Which the Joint Distribution Changes Over Time + Over-stationarization
TDformer	(X. Zhang et al., 2022)	Yes	Seasonal-trend Decomposition (Average Filters and Adaptive Weights) + Fourier Attention Mechanism + MLP + Reversible Instance Normalization	Relationships Between Attention Models in Different Time and Frequency Domains
Seformer	(P. Zeng et al., 2023)	No	Binary Positional Encoding (ODEs and Segments) + Information Transfer Regularization	Neural ODE's Problems – Cost, Drift Problem, Information Loss and Not Being Directly Applicable to LSTF Problems
W-transformers	(Sasal et al.,	Yes	Maximal Overlap Discrete	Non-stationary

	2022)		Wavelet Transform	Data
Deep Autoformer	(Jiang et al., 2022)	No	Integrates Multi-Layer Perceptrons (MLPs) into the basic Autoformer Framework	Extreme Fluctuations and Irregular Data Patterns of VSTLF
GQFormer	(Jawed & Schmidt-Thieme, 2022)	Yes	Unique Time-series ID Embedding + Implicit Quantile Level Embedding + Covariates + Sparse Self-attention + Quantile Transposed Attention + Multi-Task Loss + Probabilistic Forecasting	Probabilistic Forecasting
TDT	(Ye et al., 2023)	No	Decomposition + Covariates + 2 Decoders + NLL Loss + Probabilistic Forecasting	Probabilistic Forecasting
ADAMS	(Y. Huang & Wu, 2023)	No	Decomposition (Moving Averages) + Multi-Scale Framework + De-Stationary Attention Module + Auto-Correlation Mechanism + Adaptive Huber Loss Function	PV Data
InParformer	(H. Cao et al., 2023)	No	Evolutionary Seasonal-Trend Decomposition + Interactive Parallel Attention + Interactive Partitioned Convolution	Context-agnostic and Capturing Overall Properties of Time Series
iTransformer	(Y. Liu et al., 2024)	Yes	Inversion of Data Dimensions + Attention Mechanism on Variate Tokens + Feed-Forward Networks	Larger Lookback Windows Leading to Performance Degradation and Computation Explosion
FFT-Informer	(J. Ma & Dan, 2023)	No	Feature Extraction (Segments + FFT) + Multi-head ProbSparse Self-Attention with Hampel Filter + One-forward Operation Forecasting	To Forecast the Structural State in a Step-by-step Manner, Extracting Feature of Structural State Trends and the Negative Effects of Data Collection Under

				Abnormal Conditions are Big Challenges
DWTformer	(Y. Cao & Zhao, 2023)	No	Wavelet Transform + 2D-FeaturesBlock (CNN) + Frequency Enhanced Block (DTF) + Autocorrelation in the Attention Mechanism	Complex Intrinsic Features of the Series
Conformer	(Y. Li, Lu, et al., 2023)	Yes	Input representation (FFT) + Sliding-Window Attention + Stationary and Instant Recurrent Network (GRU and RNN) + Normalizing Flow Framework	Sparse Self-Attention Leading to Limited Information Utilization
D-Transformer	(H. Wu et al., 2023)	No	Integration of the Duffing Equation + Improvement in Initial Weight Settings + Dropout Regularization	Timing Forecasting
PDTrans	(J. Tong et al., 2023)	No	Covariates + Decomposition Framework (MLP and CNN) + Conditional Generative Model + Hierarchical Probabilistic Forecasting + NLL loss and KL divergence	Error Accumulation of the Autoregression + Mining Reliable Temporal Dependencies
Scaleformer	(Shabani et al., 2023)	Yes	Input Embedding (Value, Temporal, and Positional Embeddings) + Multi-scale Framework + Cross-Scale Normalization + Iterative Refinement + Adaptive Loss Function	Lack of Explicit Scale-aware Processing
PatchTST	(Nie et al., 2023)	Yes	Patching + Channel Independence + Representation Learning	Order and Context Agnostic
Autoformer- GA	(Pan et al., 2023)	No	Integration of Genetic Algorithm + Archive Storage Operator + Elite Voting Operator	Fail to Explain the Complex Relationships Between Prediction Targets and External Factors
Foreformer	(Y. Yang & Lu, 2023)	No	Novel Positional Encoding + Multi-Temporal Resolution Module (ELU Activation + Convolution) + Explicit Sparse Multi-Head Attention + Static	Insufficient Extraction of Temporal Patterns at Different Scales

			Covariates Processing Module	+ Not Using Covariates + Extraction of Irrelevant Information in the Self-attention
CLformer	(X. Wang, Liu, Du, et al., 2023)	No	Input Representation (Dilated Causal Convolution) Decomposition + Temporal Convolution Block + Locally Grouped Auto-Correlation (Segments)	Complexity + Unable to Capture Temporal Patterns at Multiple Scales
Convolution Transformer	(N. Wang & Zhao, 2023b)	No	ES Attention Mechanism + Frequency Enhanced Block (DTF) + Causal Dilated Convolution + Multi-layer Feature Fusion	Hybridization with CNN
Preformer	(Du et al., 2023)	Yes	Decomposition Blocks + Multi-Scale Segment-Correlation (MSSC) + Predictive MSSC	Complexity + Context Agnostic
Seformer	(P. Zeng et al., 2023)	No	Binary Positional Encoding (ODEs and Segments) + Information Transfer Regularization	Neural ODE's Problems – Cost, Drift Problem, Information Loss and Not Being Directly Applicable to LSTF Problems
Federated Transformer	(Thwal et al., 2023)	No	Time2Vec Embedding + Federated Attention + Federated Learning	Conventional Training Scheme Has Shown Deficiencies Regarding Model Overfitting, Data Scarcity, and Privacy Issues
Rankformer	(Ouyang et al., 2023a)	No	Rank Correlation Block + Multi-Level Decomposition Blocks (Multiple Moving Average)	Hidden Dependencies Not Being Properly Extracted, Especially Nonlinear Serial Dependencies
DBAFormer	(J. Huang et al.,	No	Double-Branch Attention	Sparse Attention

	2023)		Mechanism (Temporal and Variable Attention) + Query-Independent Attention + 1D and K-Dimensional Convolutional Layers	Limiting Information Utilization
TEFNEN	(Domínguez-Cid et al., 2023)	No	Input Representation (SNN and RNN) + dropout	Modelling Photovoltaic Data
STLformer	(Ouyang et al., 2023b)	No	STL Decomposition Block (LOESS) + Rank Correlation Block (Spearman's)	Complex Trends and Nonlinear Serial Dependencies
Stack Transformer	(Tevare & Revankar, 2023)	No	Decomposition + Time2Vec Embedding + Integrate Temporal Attention	Stock Data
SMARTformer	(Y. Li, Qi, et al., 2023)	No	Time-Independent Embedding + Integrated Window Attention + Semi-Autoregressive Decoder	Local Temporal Dependencies
CNformer	(X. Wang, Liu, Yang, et al., 2023)	Yes	Sequence Decomposition Block (Convolution) - Stacked Dilated Convolution Modules - Convolutional Attention	Temporal Dependencies and Inter-variable Dependencies in Intertwined Temporal Patterns Being Unreliable + Inability to Capture Both Short- and Long-term Dependencies Well
DESTformer	(Y. Wang et al., 2023)	No	FFT Decomposition + Multi-View Attention + Multi-Scale Attention (CNN)	Information Utilization Bottleneck (Ignoring the Fine-grained Temporal Patterns in the Trend View in the Series Decomposition Component)
Bidformer	(W. Li et al., 2023)	No	Bidirectional Sparse Self-Attention Mechanism +	Time and Memory Cost

			Shared-QK Method + Sparsity Measurement + Generative Inference	
Dsformer	(Yu et al., 2023)	No	Double Sampling Block + Temporal Variable Attention Block +Generative Decoder (MLP)	Not Making Full Use of Global Information, Local Information, and Variables Correlation
Causalformer	(C. Zhang et al., 2023)	No	Input Representation (Temporal and Causal Features) + Granger Causality Graph + ProbSparse-Attention + Generative Decoder + Sparsity Penalties	Neglecting the Interactions Among Multivariate Elements, and Lacking the Application of Causal Relationships.
DifFormer	(B. Li, Cui, et al., 2023)	No	DiffAttention + dynamic ranging + Δ-lagging + Temporal Stream (Down-Scaling + Patch Merging + Multi-Resolutional Differencing)	Representing Nuanced Seasonal, Cyclic, and Outlier Patterns
Enformer	(N. Wang & Zhao, 2023a)	No	Sparse Periodic Self-Attention (KL Divergence) + Coarse Matching Module (CNN)	Efficiency + Vector Information Redundancy + Processing
TEDformer	(Fan et al., 2023)	No	STL Decomposition (LOESS) + Temporal Feature Extraction (TCNs)	Not Fully Considering the Local Temporal Features of the Sequence, and Not Addressing the Impact of Sequence Anomalies on Decomposition and the Processing of Trend Terms
MSRN-Informer	(X. Wang, Xia, et al., 2023)	No	Multi-Scale Structure + Residual Network + Multi-Scale Convolutional Kernels + ProbSparse Self-Attention	Most Studies Focusing on Building-level Energy

				Forecasting Rather Than Individual Load Forecasting, Which Cannot Support Controlled Demand Response Programs
SL-Transformer	(J. Zhu et al., 2023)	No	Savitzky–Golay (SG) Filter + Local Outlier Factor (LOF) filter + LSTMs	Noise and Temporal Intricacies
FPPformer	(Shen et al., 2023)	Yes	Diagonal-Masked Self-Attention + Combined Patch-wise and Element-wise Attention + Hierarchical Feature Mapping + Composite Loss Function	Extracting the Features of Input Sequence and Seeking the Relations of Input and Prediction Sequence
BasisFormer	(Ni et al., 2024)	Yes	Basis Vectors + Coef Module + Bidirectional Cross-Attention + Regularization Term + Dual Loss Function + End-to-End Training	Using Bases
TCLN	(S. Ma et al., 2023)	No	Multi-kernel CNN Module + LSTM Encoder + Autoregressive Component	Lack of Depth in Extracting Spatial and Spatiotemporal Features Between Variables
NTformer	(Zhong et al., 2023)	No	Learnable Timestamp Encoding + Temporal and Non-temporal Processing + Temporal Attention + Temporal Dimension Correlation + Feature Conversion Mechanism + Loss Function with L2 Regularization	Dataset
HAFF	(Z. Wang et al., 2023)	No	Refine Auto-Correlation Mechanism of Autoformer	Difficult to Forecast the Electricity Demand Series Because of the

				Influence of Cyclical Factors and Small Data Amounts.
MR-Transformer	(S. Zhu et al., 2023)	No	Covariates + Adaptive Segmentation (DWT) + Variable-Specific Temporal Convolution + Variable-Specific Attention + Long Short-Term Attention	Context-agnostic
Forwardformer	(Qu et al., 2024)	No	Input Representation (Value, Timestamp and Information Embeddings) and + Multi-Scale Forward Self-Attention + Novel Loss Function	Ignore Special Events
RSMformer	(G. Tong et al., 2024)	No	Up-and-down Sampling + Residual Sparse Attention Mechanism + Cross-Scale Feature Relationships+ Adaptive Huber Loss Function	Constrained by Single Time Scale, the Quadratic Calculation Complexity of the Self-Attention Mechanism, and the High Memory Occupation.
Multirate-Former	(D. Liu et al., 2024)	No	Data Chunking + Coarse-Grained Data Complementation (CNN) + Self-Reconstruction Error + Sampling-Type Coding + Fine-Grained Complementation + Unsupervised Pretraining	Ubiquitous Multirate Sampling Characteristics of the Data
SageFormer	(Z. Zhang et al., 2024)	Yes	Global Tokens + Series-Aware Framework + Graph-enhanced Transformer + Sparsity + ForecastingHead	Prevailing Methods Either Marginalize Inter-series Dependencies or Overlook Them Entirely
MTPNet	(Y. Zhang, Wu, et al., 2024)	Yes	Input Representation (Dimension-Invariant Embedding + CNN + Patches) + Multi-scale Transformer Pyramid Network + Hierarchical Inter-Scale	Prior Work has Been Confined to Modeling Temporal Dependencies at Either a Fixed

			Connections	Scale or Multiple Scales that Exponentially Increase (Most with Base 2). This Limitation Impedes Their Capacity to Effectively Capture Diverse Seasonalities.
PatchTCN-TST	(Cen & Lim, 2024)	No	Segments (TCN) + Channel-Independent + TCN Residual Blocks + Instance Normalization and De-Normalization + Multi-Task Learning	Efficiency
Graphformer	(Y. Wang et al., 2024)	No	Learnable Tokens (GNsNs) + Graph Self-Attention Mechanism (Sparse) + Dilated Causal Convolution + Temporal Inertia Module + Multi-scale Feature Fusion	Cannot Effectively Exploit the Potential Spatial Correlation Between Variables
TS-Fastformer	(Lee et al., 2024)	Yes	Sub Window Tokenizer + Pre-trained Encoder + Past Attention Decoder	Sequential Relationships + Slow Speed
MASTER	(T. Li, Liu, et al., 2023)	Yes	Market-Guided Gating Mechanism + Momentary and Cross-Time Stock Correlation + Temporal Aggregation	First, Stock Correlations Often Occur Momentarily and in a Cross-time Manner. Second, the Feature Effectiveness is Dynamic with Market Variation, Which Affects Both the Stock Sequential Patterns and Their Correlations.
MOEA	(Ban et al., 2024)	No	Integrate Mixture Of Expert Decomposition and Wavelet Soft Threshold Denoising	Dataset
PWDformer	(Z. Wang et al.,	No	Position Weights + Frequency	Trend to Reduce

	2024)		Selection Module + Deformable-Local Aggregation Mechanism (Adaptive Windows)	Computational Complexity at the Expense of Time Information Aggregation Capability + Order information loss
Hidformer	(Z. Liu et al., 2024)	No	Segment-and-Merge Architecture + Replacement of Multi-Head Attention with Recurrence and Linear Attention	Permutation-invariant + Missing Multi-scale Local Features and Information from the Frequency Domain
JTFT	(Y. Chen et al., 2023)	Yes	Learnable Positional Encoding (Joint Time-Frequency Domain Representation + Customized Discrete Cosine Transform) + Low-Rank Attention Layer	Efficiency + Multivariate Forecasting
MTST	(B. Li, Cui, et al., 2023)	Yes	Relative Positional Encoding + Patch-Level Tokenization + Multi-resolution processing + Multi Scale Feature Learning + Adapted Attention Mechanism	Patch Size
PDF	(Dai et al., 2023)	Yes	Multi-periodic Decoupling Block (FFT and Segmentation) + Dual Variations Modeling Block (CNN) + Variations Aggregation Block	Intricate Temporal Patterns
Pathformer	(P. Chen et al., 2023)	Yes	Multi-Scale Router (Adaptive Segments) + Adaptive Pathways + Dual Attention	Challenging to Capture Different Characteristics Spanning Various Scales
TACTIS-2	(Ashok et al., 2024)	Yes	Dual-encoder (Marginal and Copula Distributions for Covariates and Input Representation) + Causal Attention Mechanism + Two-Stage Optimization Process +	Increase Flexibility

			Copulas	
TEMPO	(D. Cao et al., 2024)	No	Specialized Embeddings + Zero-shot Learning (Soft Prompting Mechanism) + Generative Pre-trained Transformer (GPT)	Explore Whether GPT-type Architectures Can Be Effective for Time Series

APPENDIX D (AGGREGATION OF RESULTS)

In this Section, the final results from different papers are aggregated, the excels "agglomeration" and "info" are available in link and link.

Table 0.1 – Evolution of Informer metrics throughout different papers

Metric		Informer		Informer in Linear		Informer in PatchTST		Informer in Conformer		Informer in DWTformer	
		MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
Electricity/ ECL	96	0,274	0,368			0,304	0,393	0,5423	0,5568	2,26	0,368
	192	0,296	0,386			0,327	0,417	0,5304	0,5549	0,284	0,384
	336	0,3	0,394			0,333	0,422	0,6429	0,5921	0,334	0,423
	720	0,373	0,439			0,351	0,427	0,9534	0,7789	0,592	0,589
Exchange	96	0,847	0,752			0,847	0,752	0,231	0,3841	0,395	0,453
	192	1,204	0,895			1,204	0,895	0,3079	0,4488	0,692	0,6
	336	1,672	1,036			1,672	1,036	0,5902	0,6306	0,809	0,649
	720	2,478	1,31			2,478	1,31	0,863	0,7953	1,111	0,769
Traffic	96	0,719	0,391			0,733	0,41			0,739	0,413
	192	0,696	0,379			0,777	0,435			0,762	0,424
	336	0,777	0,42			0,776	0,434			0,862	0,487
	720	0,864	0,472			0,827	0,466			1,056	0,53
Weather	96	0,3	0,384			0,354	0,405	0,3929	0,3231		
	192	0,598	0,544			0,419	0,434	0,4396	0,4332		
	336	0,578	0,523			0,583	0,543	0,5848	0,5197		
	720	1,059	0,741			0,916	0,705	0,7051	0,5881		
ILI	24	5,764	1,677			4,657	1,449				
	36	4,755	1,467			4,65	1,463				
	48	4,763	1,469			5,004	1,542				
	60	5,264	1,564			5,071	1,543				
ETTh1	96			0,865	0,713	0,941	0,769	0,8901	0,6498	0,905	0,751
	192			1,008	0,792	1,007	0,786	1,0463	0,7082	1,006	0,786
	336			1,107	0,809	1,038	0,784	1,1237	0,7383	1,036	0,783
	720			1,181	0,865	1,144	0,857	1,1047	0,7292	1,155	0,846
ETTh2	96			3,755	1,525	1,549	0,952			2,833	1,321
	192			5,602	1,931	3,792	1,542			2,625	2,091
	336			4,721	1,835	4,215	1,642			5,309	1,953
	720			3,647	1,625	3,656	1,619			3,989	1,665
ETTm1	96			0,672	0,571	0,626	0,56	1,0921	0,7023	0,623	0,559
	192			0,795	0,669	0,725	0,619	1,2657	0,7898	0,854	0,686
	336			1,212	0,871	1,005	0,741	1,3849	0,8459	1,074	0,769
	720			1,166	0,823	1,133	0,845	1,3537	0,8492	1,236	0,826
ETTm2	96	0,365	0,453			0,355	0,462			0,387	0,478
	192	0,533	0,563			0,595	0,586			0,808	0,706
	336	1,363	0,887			1,27	0,871			1,464	0,928
	720	3,379	1,338			3,001	1,267			3,904	1,466

In Table 0.1 – Evolution of Informer metrics throughout different papers, the 1st column was introduced in the Informer paper (H. Zhou et al., 2021). The 2nd column was introduced in the Linear paper (A. Zeng et al., 2022), and using the same hyper-parameters train the model for the missing datasets. The 3rd column was introduced in the PatchTST paper (Nie et al., 2023), changing the size of the look-back window. The 4th column was introduced in the Conformer model (Y. Li, Lu, et al., 2023), setting the sampling factor to 1. The 5th column was introduced in the DWTformer paper (Y. Cao & Zhao, 2023), using the same parameters as the first column. The values from these columns were copied in other papers.

This shows the inherent stochasticity associated with transformers by comparing the first and last column, as well as the impact of using different parameters in the performance of the model that was done in the 3rd and 4th columns.

Table 0.2 and Table 0.3 agglomerate the results of papers that use $R_{0.5}$ and CRPS. As said before, most of the papers use the MSE and MAE metrics. For multivariate results, there are 2 main forecasting windows {24, 48, 168, 336, 720} and {96, 192, 336, 720}. The first has 42 columns and 31 rows while the second has 82 columns and 38 rows. The same 2 forecasting windows are used for univariate forecasting having 14 columns and 27 rows, and 36 columns and 37 rows, respectively. This means that the tables cannot be shown here, but they are in excel sheet “Agglomeration”.

Table 0.2 – Agglomeration of results that use $R_{0.5}$ as the metric

R0.5	Logtrans	ARIMA	ETS	TRMF	DeepAR	DeepState	TFT	AST	Informer	ACT	PDTrans
Electricity-c 1d	0.059	0.154	0.101	0.084	0.075	0.083	0.055	0.042 ± 0.007	0.062 ± 0.005	0.040 ± 0.007	0.058
Electricity-c 7d	0.070	0.283	0.121	0.087	0.082	0.085		0.057 ± 0.010	0.072 ± 0.009	0.055 ± 0.010	0.068
Traffic-c 1d	0.122	0.223	0.236	0.186	0.161	0.167	0.095	0.093 ± 0.010	0.120 ± 0.019	0.091 ± 0.010	0.113
Traffic-c 7d	0.139	0.492	0.509	0.202	0.179	0.168		0.125 ± 0.019	0.130 ± 0.023	0.119 ± 0.019	0.126

Table 0.3 - Agglomeration of results that use CRPS as the metric

CRPS-Sum	ProTran	Auto-ARIMA	ETS	TimeGrad	GPVar	TACTiS	TACTiS-2
SOLAR	0.194 ± 0.030						
ELECTRICITY	0.016 ± 0.001	0.077 ± 0.016	0.059 ± 0.011	0.067 ± 0.028	0.035 ± 0.011	0.021 ± 0.005	0.020 ± 0.005
TRAFFIC	0.028 ± 0.001						
TAXI	0.084 ± 0.003						
WIKIPEDIA	0.047 ± 0.004						
fred-md		0.043 ± 0.005	0.037 ± 0.010	0.094 ± 0.030	0.067 ± 0.008	0.042 ± 0.009	0.035 ± 0.005
kdd-cup		0.625 ± 0.066	0.408 ± 0.030	0.326 ± 0.024	0.290 ± 0.005	0.237 ± 0.013	0.234 ± 0.011
solar-10min		0.994 ± 0.216	0.678 ± 0.097	0.540 ± 0.044	0.254 ± 0.028	0.311 ± 0.061	0.240 ± 0.027
traffic		0.222 ± 0.005	0.353 ± 0.011	0.126 ± 0.019	0.145 ± 0.010	0.071 ± 0.008	0.078 ± 0.008

NOVA Information Management School
Instituto Superior de Estatística e Gestão de Informação

Universidade Nova de Lisboa