Transformers for Time-Series Forecasting

Ben Breitinger

Born 5th January 2001 in Bensheim, Germany

June 27, 2024

Master's Thesis Mathematics

Advisor: Prof. Dr. Jochen Garcke

Second Advisor: Prof. Dr. Jürgen Dölz

Institute for Numerical Simulation

Mathematisch-Naturwissenschaftliche Fakultät der

Rheinischen Friedrich-Wilhelms-Universität Bonn

Acknowledgements

I would like to express my deepest gratitude to my advisor, Prof. Dr. Jochen Garcke, for proposing the research topic, guiding me in formulating an engaging research question, and supporting me with his extensive experience throughout the process. His assistance has been invaluable, especially in navigating the complexities of a rapidly evolving field.

I am also thankful to my partner, Anne, for the insightful discussions on this thesis' topics and her emotional support.

Lastly, I extend my gratitude to the Studienstiftung des deutschen Volkes for the financial support, which allowed me to concentrate fully on my studies.

Contents

Chapter 1

Introduction

Time-series data is widespread across various fields and significantly influences decision-making processes. In finance, time-series data plays a crucial role, especially in developing and utilising high-frequency trading algorithms. These algorithms exploit fleeting market opportunities with remarkable precision, while portfolio optimisation strategies rely heavily on historical market data. This enables the creation of diversified investment portfolios tailored to investors' risk preferences and financial goals. Additionally, risk management models leverage time-series data to assess and mitigate financial risks, providing valuable insights into market volatility and asset performance.

In weather forecasting, time-series data forms the foundation for accurate predictions of meteorological phenomena. Meteorologists analyse historical weather patterns alongside real-time atmospheric conditions to forecast severe weather events such as hurricanes and tornadoes. This aids in implementing timely evacuation plans and mitigating potential damage. Similarly, in agricultural planning, time-series data helps farmers optimise planting schedules and irrigation strategies based on historical climate data and seasonal trends. This enhances agricultural productivity and improves food security.

In resource management, time-series data is pivotal for forecasting demand across supply chains. It enables streamlined inventory management and distribution logistics by predicting fluctuations in consumer demand based on historical sales data and market trends. Additionally, time-series data facilitates energy consumption forecasting for utilities, enabling proactive grid management to meet fluctuating electricity demand efficiently while minimising waste and environmental impact. Moreover, in urban planning, analysing historical traffic patterns alongside real-time congestion data informs strategies to optimise traffic flow and enhance transport infrastructure, reducing congestion in urban centres.

As organisations increasingly harness the predictive potential of time-series data, there is a growing demand for accurate and scalable forecasting methods. This demand drives the development of advanced analytical techniques and robust predictive models across multiple sectors, fostering innovation and efficiency.

In the early 20th century, foundational work by statisticians like George Udny Yule and Andrey Kolmogorov laid the groundwork for time-series forecasting. Yule's autoregressive

models, introduced in 1927, provided a framework for understanding temporal dependencies in sequential data [Yul27]. Meanwhile, Kolmogorov's contributions to probability theory and stochastic processes laid the theoretical foundation for modeling random fluctuations in time-series data.

The mid-20th century saw significant advancements with the rise of computer technology, enabling the application of mathematical modeling techniques on a larger scale. The Box-Jenkins methodology, developed in the 1970s, formalised model selection, estimation, and diagnostic checking for autoregressive integrated moving average (ARIMA) models [BJ77]. Concurrently, linear regression techniques gained prominence for trend analysis, enabling quantification and extrapolation of linear relationships over time.

The late 20th and early 21st centuries saw transformative changes in time-series forecasting, driven by the exponential growth in data availability and computational power. Neural network-based methodologies emerged, starting with artificial neural networks (ANNs) in the late 1980s. Multilayer perceptrons (MLPs) demonstrated effectiveness in assimilating historical data for precise predictions [LF87].

Further evolution led to recurrent neural networks (RNNs) in the 1990s, which incorporated feedback loops for sequential data processing. However, challenges such as the vanishing gradient problem limited their effectiveness in capturing long-term dependencies. The introduction of Long Short-Term Memory (LSTM) networks addressed these challenges, enhancing the accuracy of neural network-based methodologies for time-series forecasting [HS97].

In 2017, the Transformer architecture introduced a major paradigm shift by leveraging self-attention mechanisms to discern global dependencies within data sequences [Vas+17]. Unlike traditional RNNs, Transformers excel in capturing intricate temporal patterns and nonlinear relationships, revolutionising domains like natural language processing (NLP) and computer vision (CV). Models such as BERT and GPT have set new benchmarks in NLP tasks [Dev+19; Bro+20], while Vision Transformers (ViTs) challenge the dominance of convolutional neural networks (CNNs) in computer vision [Dos+21].

Deploying Transformers in time-series forecasting exploits their ability to analyse historical data effectively, promising enhanced accuracy and scalability. This marks a significant advancement in predictive analytics, ushering in an era of innovation and efficiency across diverse industries.

Objectives of this work

The multivariate time-series forecasting problem is defined as the task of making a multi-step forecast of length T for all N_{in} variates of the time-series, given a lookback of the most recent L_{in} historic observations per variate. A formal definition of this problem will be provided in the next chapter.

In recent years, a plethora of Transformer models have been developed with the specific goal of multivariate time-series forecasting in mind. The following chapter will provide an overview of the emerging literature on this subject. While many works incorporate inductive biases specific to time-series data into the model, the focus has recently shifted towards the

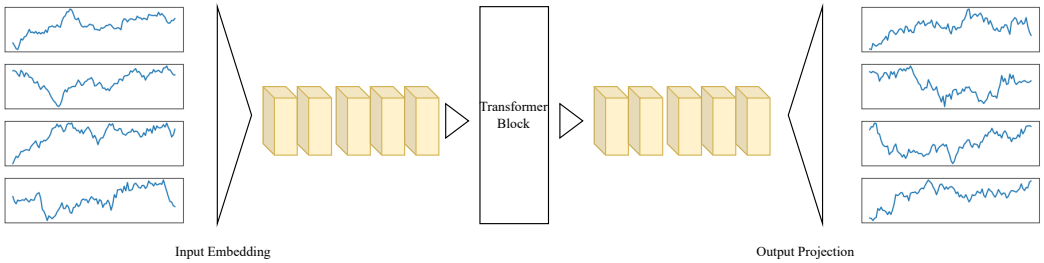

Figure 1.1: Illustration of the flow through a simple Transformer model: All variates of length L_{in} are combined in a single token for each timestamp and then processed by the Transformer block that computes a representation that is then used to forecast the next T timestamps.

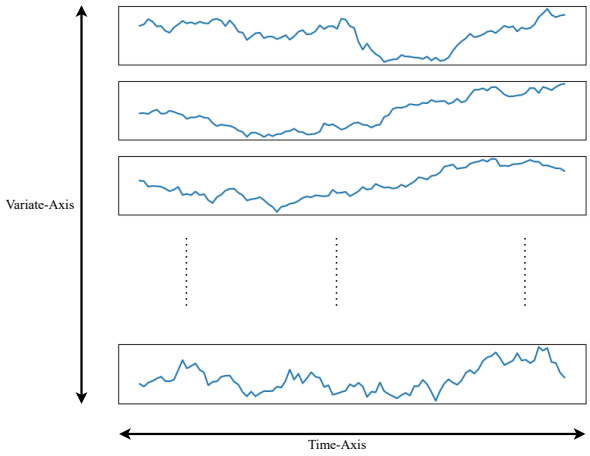

Figure 1.2: We distinguish the time- and variate-axis for a multivariate time-series.

study of different model architectures on a higher level. Previously, for each timestamp, all observations of a multivariate time-series were combined into a single token (see Figure 1.1). Subsequently, self-attention, the fundamental component of the Transformer, is employed to extract the underlying dynamics by combining the different tokens.

Goal 1: Systematic Study of Self-Attention and MLPs along the Time- and Variate-Axis

Recently, [Liu+24; ZY23; Wan+24; Nie+23; Zen+23] have questioned the combined embedding of many variates into one single token and the application of self-attention along the time axis (see Figure 1.2). Moreover, models that rely on MLPs as their central mechanism for combining different observations of the time-series have received a revival in recent years. This is evidenced by [Zha+22; Che+23]. These developments have resulted in the emergence of numerous models with varying structures that have demonstrated empirical success in multivariate time-series forecasting. To the author's knowledge, there is no comprehensive study of the different architectures beyond narrow ablation studies of the proposed models in their respective works with respect to forecasting performance.

A systematic study of the different architectures must provide a common reference point. In light of the above, we put forth the FlexibleTransformer architecture, which builds upon the architectures in [Nie+23; ZY23; Wan+24] and in which we can formulate current Transformer- and MLP-based models as examples. This serves as a foundation for extending the techniques proposed in [Kob+21; Kob+24] to the realm of time-series. Kobayashi et al. analyse contextualisation in the BERT and GPT language models by decomposing the Transformer's components. This approach enables the analysis of the impact of MLPs and self-attention mechanisms along the time- and variate-axes. The proposed decomposition is essentially a feature attribution method, which also provides a novel interpretation of model results for time-series data by attributing predictions to the models' inputs.

Goal 2: Study of Self-Attention Approximation Schemes along the Variate Axis

The success of methods that do not combine variates in the input embedding has sparked the development of architectures combining different variates by applying self-attention [Liu+24]. Due to the squared complexity of the self-attention mechanism in the input sequence length, there is a well-established body of research studying efficient attention approximation, which brings down the computational cost to $O(L_{\text{in}} \log L_{\text{in}})$ or even $O(L_{\text{in}})$. A substantial body of research has been conducted on this topic, with a focus on developing methods that are agnostic to the specific application. These methods can be broadly classified into two categories: those that assume sparsity of the attention score matrix and those that assume a low-rank property [Che+21; KKL20; Cho+21]. Approaches that are specific to time-series data are designed to be applied to self-attention along the time axis.

The recent advances in methods applying self-attention along the variate-axis have raised the question of efficient self-attention approximation along this axis, which remains an open problem. As most modern time-series data is very high-dimensional, efficient self-attention approximation is of particular importance in this context. Furthermore, we hope that by enabling the study of self-attention along the variate-axis in high-dimensional settings, we can pave the way for future research in this new setting.

Outline

In chapter 2 we give a more detailed overview of the current state of the literature in the field of applying Transformers to time-series data and introduce the basic structure of the vanilla Transformer encoder. This allows us to study some selected modifications of the Transformer architecture that are relevant for this work. Afterwards in chapter 3, we propose the FlexibleTransformer that provides a new framework for jointly studying recent Transformer- and MLP-based architectures. We use this as a starting point for the systematic analysis of self-attention and MLPs in different settings in chapter 4 by using feature attribution methods. In chapter 5, we then tackle the question of efficient self-attention approximation along the variate axis. Finally, we conclude and give an outlook in chapter 6.

Contributions

This work's contributions are

- We propose the FlexibleTransformer which provides a framework for a systematic analysis of popular Transformer-based time-series forecasting models. Hereby, we abstract from the concrete architecture and identify the common building blocks amongst seemingly unrelated architectures.
- We introduce the MLP token-mixer in the FlexibleTransformer framework. This carries the model architecture from TSMixer [Che+23] over to our setting.
- We then conduct a comprehensive empirical analysis of configurations within the FlexibleTransformer, including model architectures that have not yet been explored in the literature.
- We provide an end-to-end decomposition of the FlexibleTransformer based on the ideas from [Kob+21; Kob+24] and make Transformer-based time-series models interpretable.
- We explore the application of common efficient self-attention mechanisms along the variate-axis in the FlexibleTransformer to allow for modern large-scale applications.

Chapter 2

Transformers for Time-Series Forecasting

This chapter aims to introduce the relevant algorithmic descriptions and theoretical foundations of current Transformer model architectures. Given the multitude of different Transformer models for a wide range of applications, we aim to streamline the presentation towards our application of time-series forecasting. One of the main challenges is to present the concepts in sufficient generality so that we can later abstract from the concrete model architecture, while also providing an in-depth review of the relevant models' components for the novice to Transformer architectures.

This chapter is organised as follows: We begin by explaining the motivation behind the use of Transformers for time-series problems and defining the problem to be solved. We then present a reminiscent of the vanilla Transformer [Vas+17] tailored towards time-series forecasting. While the exposition of this condensed form of the vanilla Transformer should be viewed in an exemplary light, it provides us with the opportunity to introduce the essential concept of self-attention. In the following section, we will delve deeper into the current state-of-the-art modifications of the Transformer architecture. We will begin this discussion by providing a review of the broad literature on such modifications tailored towards time-series tasks. Subsequently, a number of specific models will be discussed in greater detail, forming the basis for further investigation throughout the remainder of the thesis and serving as a foundation for the subsequent chapter.

2.1 Motivation

Before embarking on an exposition of the Transformer, it is first necessary to provide a heuristic motivation behind the Transformer.

Prior to the invention of Transformers, LSTMs had been the state-of-the-art since the 1990s for time-series forecasting tasks. LSTMs are based on the idea of RNNs, namely that the input sequence is processed step by step and a hidden state is updated with each timestamp by the very same neural network. RNNs can be conceptualised as extremely deep neural networks. Their depth is essentially the length of the sequence, which presents a significant

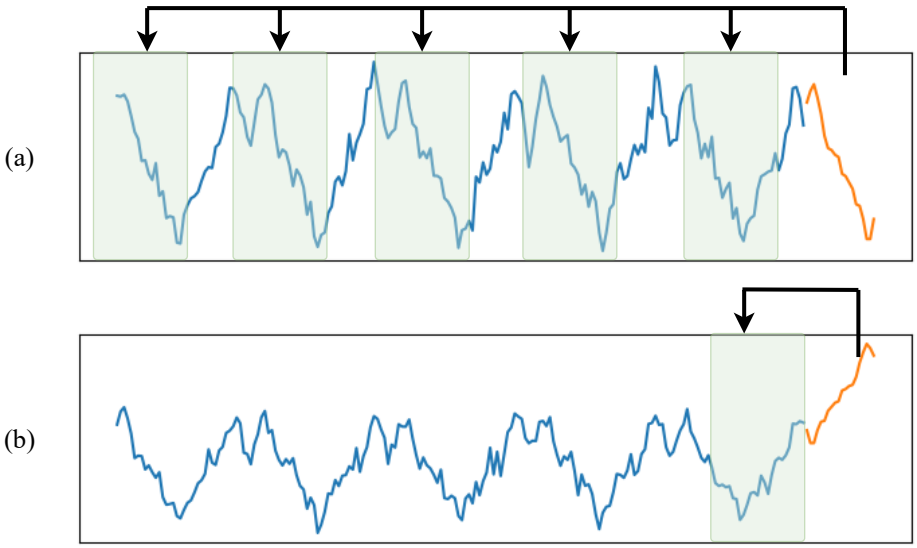

Figure 2.1: The self-attention mechanism of the Transformer (a) is able to capture long-range patterns in the data while LSTMs (b) have a more localised receptive field due to the sequential processing.

challenge since the gradients vanish or explode in backpropagation due to the chain rule. This issue was addressed by LSTMs, which introduced a gating mechanism to only partially update the hidden state with new information. However, a commonality among these model architectures is that the input sequence is processed sequentially. This has two inherent drawbacks: Firstly, training and inference are both considerably slower with long sequences due to the inability to parallelise the operations. Secondly, it is extremely challenging to learn patterns based on observations that are widely spaced in the input sequence.

These are two significant challenges that Transformers aim to address. The innovative concept behind the ability to overcome the limitations of sequential models is the concept of self-attention. Self-attention can be conceptualised as a general approach to conducting pairwise comparisons of distant elements within an input sequence (see Figure 2.1 (a)). This enables the Transformer to identify global patterns and excel in long-range forecasting tasks. An alternative way of expressing this insight is that Transformers represent a special case of graph neural networks (GNNs) that correspond to a fully connected graph (see [Zho+20] for an introduction).

It has been demonstrated that these pairwise comparisons can be parallelised, which results in the entire model having a training and inference time that is $O(1)$ in the input sequence length L_{in}.

2.2 The Time-Series Forecasting Problem

For the sake of completeness, we want to define the concept of a *stationary multivariate time-series* and the task that we want to solve. Let $\xi = (\xi_i)_{i\in\mathbb{N}}$ be a time-discrete $\mathbb{R}^{N_{\text{in}}}$-valued

stationary stochastic process on some probability space $(\Omega, \mathcal{F}, \mathbb{P})$. We will neither explicitly mention ξ nor the underlying probability space in the coming sections and chapters.

Definition 2.1. A *stationary multivariate time-series X of ξ* of length $L_{\text{in}} \in \mathbb{N}$ is a realisation of L_{in} successive steps of a stationary $\mathbb{R}^{N_{\text{in}}}$-valued stochastic process ξ, i.e. $X \in \mathbb{R}^{N_{\text{in}} \times L_{\text{in}}}$ such that there is $\omega_X \in \Omega$ with

$$X_{ni} = (\xi_i(\omega_X))_n, \qquad n \in \{1, \dots, N_{\text{in}}\},\ i \in \{1, ..., L_{\text{in}}\}.$$

While, the case of non-stationary time-series has recently been treated by the introduction of the non-stationary Transformer in [Liu+22b], we exclusively focus on the stationary case. This case captures most of the challenges in modeling and is the classical setting for time-series forecasting. Furthermore, we are not going to explicitly treat the case $N_{\text{in}} = 1$ in this thesis. Hence, we abbreviate by simply using *time-series* for the definition above. We further remark that X is well-defined thanks to the stationarity of ξ in the definition above.

The *time-series forecasting problem for ξ with history sequence length L_{in} and prediction length T* consists of predicting

$$Y := (\xi_{L_{\text{in}}+1}(\omega_X), \dots, \xi_{L_{\text{in}}+T}(\omega_X)) \in \mathbb{R}^{N_{\text{in}} \times T}$$

for some time-series X of length L_{in}. The typical approach in the Transformer literature is to model this as a supervised learning problem with a training dataset $\mathcal{D}$ based on a long time-series of ξ containing samples of the form

$$((\xi_{l+1}(\omega), \dots, \xi_{l+L_{\text{in}}}(\omega)), (\xi_{l+L_{\text{in}}+1}(\omega), \dots, \xi_{l+L_{\text{in}}+T}(\omega))) \in \mathbb{R}^{N_{\text{in}} \times L_{\text{in}}} \times \mathbb{R}^{N_{\text{in}} \times T},$$

where l is the sample index and $\omega \in \Omega$ is corresponding to the given time-series. In training, the first component of the samples above is the input to the model and the second component serves as the label. We use the L^2-loss function for training and call a model's output $\hat{Y} \in \mathbb{R}^{N_{\text{in}} \times T}$ the *forecast* of length T.

We call the rows of X *variates* and write X_n or $X_{n,:}$ for the n-th row of X and $X_{:,i}$ for the i-th column of X.

2.3 Vanilla Transformer

The ideas of this chapter have been originally presented in [Vas+17] for machine translation. Since 2017, Transformers have been adapted to many domains including computer vision [Dos+21; Ram+21], speech recognition [DXX18] and music composition [Kes23]. The objective is to streamline the presentation of the vanilla Transformer's components in order to facilitate their subsequent utilisation in our later applications. Contrary to [Vas+17], we replace the Transformer decoder with a simple linear layer. This simplifies the model and is currently the state-of-the-art for tasks based on time-series after the effectiveness of linear decoder has been demonstrated in [Zen+23].

Hence, we have the following three building blocks that we want to study in this section:

- Input encoding: The input encoding computes so-called *tokens* from the inputted time-series X. The canonical input encoding following [Vas+17] is to produce one

token for each timestamp of the time-series. Such an input encoding is composed of the input embedding and the positional encoding.

- The transformer encoder represents the core component of the transformer. It processes a sequence of tokens derived from the input encoding, extracting the essential semantics of the time-series that are pertinent to the forecasting task. In more concrete terms, the transformer encoder is comprised of multiple stacked transformer blocks. Each of these blocks contains a token-mixer, which interrelates the tokens in the sequence, and a token-processor, which is applied independently to each token. In the original Transformer, the token-mixer is self-attention, while the token-processor is an MLP. Additionally, we use the layer normalisation introduced in [Xio+20] and residual connections around the token-mixer and token-processor proposed in [He+16] to stabilise and improve learning.

- Linear decoder: This is a great simplification of the Transformer architecture compared to the iterative decoding in [Vas+17] that follows the classical Seq2Seq-approach from [SVL14]. The linear decoder is simply a linear layer that produces the T forecasting outputs given the Transformer encoder representation.

We have hinted at the notion of a token above since they are the elementary building blocks of the models' internal data representation which contains several tokens at each step. We introduce the following new notation.

Definition 2.2. A *token* $\boldsymbol{x}$ is a vector in $\mathcal{X} := \mathbb{R}^d$, where d denotes the *model dimension*. The *token representation* is given by $(\boldsymbol{x}_i)_{i\in\mathcal{I}}$, where $\mathcal{I}$ is a model dependent index set. We further write

$$\mathcal{T} := \mathcal{X}^{\mathcal{I}} = \{(\boldsymbol{x}_i)_{i\in\mathcal{I}} \ : \boldsymbol{x}_i \in \mathbb{R}^d \text{ for } i \in \mathcal{I}\}$$

for the *token space*.

As we will shortly see, the index set $\mathcal{I}$ can for example be chosen as $\{1, \dots, L_{\text{in}}\}$, i.e. the set of timestamps. We typically use bold letters for tokens and understand all standard vector operations token-wise if not otherwise mentioned.

The structure of the vanilla Transformer is outlined in Figure 1.1. We can also formulate the vanilla Transformer for the time-series forecasting task with a linear decoder algorithmically (we define the appearing operations in the next sections) in Algorithm 1.

We have slightly changed the notation with respect to [Vas+17]. We call the *feed-forward layer* that is applied to every token independently *token MLP* and the input embedding *single token input embedding* instead of just *input embedding*. The reason for this is that we are going to introduce a more general class of models in chapter 3 for which the vanilla Transformer will just prove to be a special case.

We are now going to define the operations that appear in Algorithm 1 and provide the theoretical foundations to explore modifications and generalisations. We refer the reader to appendix A.1 for an introduction on commonly used notation and to appendix A.2 for standard operations.

Algorithm 1: VanillaTransformer

input : Time-series $X \in \mathbb{R}^{N_{\text{in}} \times L_{\text{in}}}$
output: Forecast $\hat{Y} \in \mathbb{R}^{N_{\text{in}} \times T}$

$X^{(0)} \leftarrow \text{SingleTokenInputEmb}(X) + \text{TimePosEnc}(L_{\text{in}})$;

for $l \leftarrow 1$ **to** n_{layers} **do**
  $\tilde{X}^{(l)} \leftarrow \text{ResConn}_{\text{SelfAttn}}\left(X^{(l-1)}\right)$;
  $\tilde{X}^{(l)} \leftarrow \text{LayerNorm}\left(\tilde{X}^{(l)}\right)$;
  $X^{(l)} \leftarrow \text{ResConn}_{\textbf{TokenMLP}}\left(\tilde{X}^{(l)}\right)$;
  $X^{(l)} \leftarrow \text{LayerNorm}\left(X^{(l)}\right)$;

$\hat{Y} \leftarrow \text{LinDecoder}\left(X^{(n_{\text{layers}})}\right)$

2.3.1 Input Encoding

The input encoding is composed of an input embedding that is responsible for embedding the multivariate time-series in the token space $\mathcal{T}$, that is suited for the application of self-attention, and a positional encoding, that provides a fixed representation for each timestamp, allowing the model to discern the same value at different positions. In the case of the vanilla Transformer, we embed tokens timestamp-wise with the *single token input embedding* and thus consider a *time positional encoding*. These are parts of the Transformer architecture that are subject to modifications in subsequent versions of the Transformer.

Single Token Input Embedding

The purpose of the input embedding becomes evident when we consider the domain of NLP. In this context, the input sequence X is not a time-series, but rather a sequence of words (or stems of words). Consequently, it is not yet ready to be further processed by any self-attention mechanism that generally requires a sequence of real vectors. Therefore, the first step is to encode the input sequence into the token space $\mathcal{T}$.

We will later explore different types of encodings, but focus now on the *single token input embedding* that the vanilla Transformer makes use of. The idea is the following: For every timestamp $i \in \{1, \ldots, L_{\text{in}}\}$, we want to compute a token $\boldsymbol{x}_i$. Hence, the index set appearing in definition 2.2 is $\mathcal{I} := \{1, \ldots, L_{\text{in}}\}$ for the vanilla Transformer.

Since we have very little control over the number of variates N_{in}, having an input embedding $\mathbb{R}^{N_{\text{in}}} \to \mathbb{R}^d$ is a natural choice. The model dimension d appears as a model hyperparameter and depends on the kind of data and application. For NLP tasks, relatively large values of $d = 512$ are common, whereas, we typically choose d to be (much) smaller in time-series based applications.

The reason why we call this type of embedding *single token input embedding* is that we reduce the originally present N_{in} observations at each timestamp to a single token in $\mathbb{R}^d$. Furthermore, as we will see shortly, the time positional encoding of a token is a function of the token position that carries most of the information in only a few of the d model dimensions. Hence, the input embedding allows to "make space" in $\mathbb{R}^d$ for the quickly

oscillating components of positional encoding that carry most of the positional information.

Definition 2.3. The *single token input embedding* for a time-series of length L_{in} and N_{in} variates is defined as a linear layer, given by

$$\text{SingleTokenInputEmb}\,(X) := \mathbf{Lin}_{N_{\text{in}} \to d}\left(X_{\underline{N_{\text{in}}} \times L_{\text{in}}}\right)$$

Time Positional Encoding

The time positional encoding for a token is a vector in $\mathbb{R}^d$ only depending on the position of that token in the input sequence. The main reason for having a positional encoding is the permutation-equivariance of the self-attention mechanism which requires us to add information about the timestamp to the tokens computed by the single token input embedding to allow the model to discern the same value of the input time-series in different positions. In comparison to other models such as RNNs, we have to somehow inject the sequential nature of the data, while allowing for parallel processing.

Generally, there are two approaches to (time) positional encoding. While it is possible to learn positional encoding from scratch, we typically choose a fixed function. In its classical form from [Vas+17], the positional encoding is defined in the following way.

Definition 2.4. The *time positional encoding of length* L is an element in $\mathcal{T}$ with $\mathcal{I} = \{1, \ldots, L\}$ given by

$$(\text{TimePosEnc}(L)_n)_i := \begin{cases} \sin\left(\frac{n}{C^{i/d}}\right) & \text{if } i \equiv 0 \mod 2, \\ \cos\left(\frac{n}{C^{(i-1)/d}}\right) & \text{else} \end{cases} \tag{2.1}$$

for $i \in \{1, \ldots, d\}$ and $n \in \{1, \ldots, L\}$ where $C \gg 0$.

A typical choice for C is $C = 10000$. An intuitive view on definition 2.4 is that it corresponds to a continuous analgue of a binary encoding of the timestamp (as a natural number).

Bronstein et al. provide an interesting perspective on the positional encoding in [Bro+21]. They remark that based on the work [DB21], time positional encoding implements the sequential structure of the time-series because its columns are approximately the first components of the real and imaginary parts of the eigenvectors of a circulant matrix. It turns out that the graph having such circulant matrices as their graph Laplacians are long "loops".

2.3.2 Token-Mixer: Self-Attention

The token-mixer represents the central component of the Transformer architecture. As previously mentioned in section 2.1, the token-mixer enables the Transformer to learn long-ranging patterns in the data by considering and comparing all pairs of tokens. In concrete terms, the use of self-attention as the token-mixer along the time-axis represents the great innovation described in [Vas+17].

As we have seen in Algorithm 1, we input a token representation $X^{(l-1)} = (\boldsymbol{x}_i)_{i \in \mathcal{I}} \in \mathcal{T}$ to self-attention. For the rest of the section, we simply write X instead of $X^{(l-1)}$ for $(\boldsymbol{x}_i)_{i \in \mathcal{I}} \in \mathcal{T}$. This is not to be confused with the inputted time-series which we do not

mention for the rest of this section. We say that we apply self-attention along the time-axis in Algorithm 1 because the index set $\mathcal{I}$ corresponds to timestamps. We are going to see other applications of self-attention in the second half of this chapter. Thus, we abstract from the setting in the vanilla Transformer and consider the self-attention operation on a general collection of tokens, where $\mathcal{I}$ is not fixed to the natual numbers up to L_{in}.

The idea behind self-attention is that we want to compare every possible selection of two tokens. This results in an attention score matrix which we use to produce new tokens that are weighted combinations of the original tokens. In NLP tasks, this corresponds to identifying how much attention each words pays to every other word and using that information to recombine the original sequence of words.

The self-attention operation consists of the application of several self-attention heads. Let H be the number of such self-attention heads. We begin by introducing a single self-attention head, before integrating it into the complete model. Then we want to take a step back and look at attention more generally through the lens of probability kernels.

Let $h \in \{1, \dots, H\}$. The h-th self-attention head computes queries, keys and values from the token representation $X \in \mathcal{T}$. This is simply done by linear layers projecting each token into the attention space $\mathbb{R}^{d_{\text{attn}}}$ with dimension d_{attn}. We give some guidelines about the choice of d_{attn} in remark 2.2. Hence, we get the queries, keys and values

$$\boldsymbol{\mathcal{Q}}^{(h)} = \mathbf{Lin}^{(\mathcal{Q},h)}_{d \to d_{\text{attn}}}(X), \quad \boldsymbol{\mathcal{K}}^{(h)} = \mathbf{Lin}^{(\mathcal{K},h)}_{d \to d_{\text{attn}}}(X), \quad \boldsymbol{\mathcal{V}}^{(h)} = \mathbf{Lin}^{(\mathcal{V},h)}_{d \to d_{\text{attn}}}(X). \tag{2.2}$$

The prefix "self" comes from the fact that the queries, keys and values are all derived from the same input X. We further take a similarity kernel to compare queries and keys

$$\begin{aligned} \phi : \mathbb{R}^{d_{\text{attn}}} \times \mathbb{R}^{d_{\text{attn}}} &\to \mathbb{R} \\ (\boldsymbol{q}, \boldsymbol{k}) &\mapsto \frac{\boldsymbol{q}^t \boldsymbol{k}}{\sqrt{d_{\text{attn}}}} \end{aligned} \tag{2.3}$$

In the literature, this concrete similarity kernel ϕ is known under the name *scaled dot-product* and is one of many possibilities to establish the "closeness" of two vectors. Other choices include for example the cosine-similarity which is a non-linear transformation of the scaled dot-product. Having established the similarity between queries and keys, we can define the attention head and self-attention, which is based on [Vas+17].

Definition 2.5. Let $X = (\boldsymbol{x}_i)_{i \in \mathcal{I}} \in \mathcal{T}$ be a token representation with $\mathcal{X} = \mathbb{R}^d$ and $\mathcal{Q}^{(h)} = (\boldsymbol{q}_i^{(h)})_{i \in \mathcal{I}}$, $\mathcal{K}^{(h)} = (\boldsymbol{k}_i^{(h)})_{i \in \mathcal{I}}$ and $\mathcal{V}^{(h)} = (\boldsymbol{v}_i^{(h)})_{i \in \mathcal{I}}$ be queries, keys and values in $\mathbb{R}^{d_{\text{attn}}}$ for $h = 1, \dots, H$ as computed in equation (2.2). Let $\phi : \mathbb{R}^{d_{\text{attn}}} \times \mathbb{R}^{d_{\text{attn}}} \to \mathbb{R}$ be the similarity kernel in (2.3). Then, we can define

$$\text{SelfAttn}(X) := \sum_{h=1}^{H} \mathbf{Lin}^{(\text{out},h)}_{d_{\text{attn}} \to d} \left[\text{AttnHead}\left(\mathcal{Q}^{(h)}, \mathcal{K}^{(h)}, \mathcal{V}^{(h)} \right) \right],$$

where $\text{Lin}^{(\text{out},h)}_{d_{\text{attn}} \to d}$ are output projections for each head $h = 1, \dots, H$ and where the attention head is given by

$$\text{AttnHead}\left(\mathcal{Q}^{(h)}, \mathcal{K}^{(h)}, \mathcal{V}^{(h)} \right)_j := \sum_{i \in \mathcal{I}} \text{SoftMax}\left(\phi\left(\boldsymbol{q}_j^{(h)}, \boldsymbol{k}_{i'}^{(h)} \right)_{i' \in \mathcal{I}} \right)_i \boldsymbol{v}_i^{(h)} \tag{2.4}$$

for $j \in \mathcal{I}$.

We can recognise self-attention as a map from $\mathcal{T}$ to itself. It should be noted that this is not necessary and that we could easily tweak the set of queries and hence get a different image of SelfAttn. Furthermore, we remind the reader that the softmax with inverse temperature $\beta > 0$ is given by

$$\mathrm{SoftMax}_\beta(a_1, ..., a_n)_k := \frac{\exp(\beta a_k)}{\sum_{i=1}^n \exp(\beta a_i)}.$$

If not explicitly mentioned, we assume $\beta = 1$.

Remark 2.1. *The normalising factor $\sqrt{d_{attn}}$ in the similarity kernel ϕ stabilises the attention operation. This claim is heuristically justified by the following argument: Assume that the query $\boldsymbol{q}$ and keys $(\boldsymbol{k}_s)_{s=1}^S$ are independent having mean zero and unit variance with respect to $\mathbb{P}$. Then $\mathbb{E}(\boldsymbol{q}^t \boldsymbol{k}_s) = 0$ and*

$$\mathrm{Var}\left(\boldsymbol{q}^t \boldsymbol{k}_s\right) = \mathrm{Var}\left(\boldsymbol{q}\right)^t \mathrm{Var}\left(\boldsymbol{k}_s\right) = d_{attn}$$

for all $s = 1, \dots S$. Hence, we get stability of the attention operation by dividing by $\sqrt{d_{attn}}$, which is especially useful in training where the gradients for the SoftMax *vanish as the argument vector grows in norm.*

A More Abstract View on Self-Attention

Now, let us take a step back and introduce attention in a fairly general way that will serve us later. For the moment, we ignore where the queries, keys and values come from in the Transformer's self-attention operation. One can think of attention as a fuzzy key-value lookup table with $|\mathcal{I}_K|$ key-value pairs $(\boldsymbol{k}_i, \boldsymbol{v}(\boldsymbol{k}_i))_{i \in \mathcal{I}_K}$, where $\mathcal{I}_K$ is a suitable index set for the key-value pairs. Hereby, we assume that $\boldsymbol{k}_i \in \mathcal{H}$ for some pre-Hilbert space $(\mathcal{H}, \langle \cdot, \cdot \rangle)$ and $\boldsymbol{v}(\boldsymbol{k}_i) \in V$ for some real vector space V for all $i \in \mathcal{I}_K$. We now want to query these key-value pairs by using a query $\boldsymbol{q} \in \mathcal{H}$. This means, that each query $\boldsymbol{q}$ induces a probability distribution $p(\boldsymbol{q}, \cdot)$ on the keys $K = \{\boldsymbol{k}_i\}_{i \in \mathcal{I}_K}$. Attention can then be written as the expected value of the corresponding values with respect to $p(\boldsymbol{q}, \cdot)$:

$$\mathrm{Attn}\left(\boldsymbol{q}, (\boldsymbol{k}_i)_{i \in \mathcal{I}_K}, (\boldsymbol{v}(\boldsymbol{k}_i))_{i \in \mathcal{I}_K}\right) := \int_K \boldsymbol{v}(\boldsymbol{k})\, p(\boldsymbol{q}, d\boldsymbol{k}) \tag{2.5}$$

The attention head AttnHead is obtained by considering multiple queries $(\boldsymbol{q}_j)_{j \in \mathcal{I}_Q}$, i.e.

$$\mathrm{AttnHead}\left((\boldsymbol{q}_j)_{j \in \mathcal{I}_Q}, (\boldsymbol{k}_i)_{i \in \mathcal{I}_K}, (\boldsymbol{v}(\boldsymbol{k}_i))_{i \in \mathcal{I}_K}\right)_{j'} = \mathrm{Attn}\left(\boldsymbol{q}_{j'}, (\boldsymbol{k}_i)_{i \in \mathcal{I}_K}, (\boldsymbol{v}(\boldsymbol{k}_i))_{i \in \mathcal{I}_K}\right). \tag{2.6}$$

For reference, the following choices have been made in the discussion of the AttnHead above:

$$\mathcal{I}_K := \mathcal{I}_Q := \mathcal{I}, \qquad p(\boldsymbol{q}, d\boldsymbol{k}) := \sum_{i \in \mathcal{I}} \mathrm{SoftMax}\left(\phi\left(\boldsymbol{q}, \boldsymbol{k}_{i'}\right)_{i' \in \mathcal{I}}\right)_i \delta_{\boldsymbol{k}_i}(d\boldsymbol{k})$$

This somewhat more general presentation of attention allows us to identify the key components.

Complexity analysis of (self-)attention

We analyse the complexity for the attention mechanism of the vanilla Transformer. Each query requires the evaluation of the integral in equation (2.5). Hence, the summation of $|\mathcal{I}_K|$ d_{attn}-dimensional weighted values leads to an $O(|\mathcal{I}_K| d_{\text{attn}})$ time-complexity for a single evaluation of Attn. Since we have a total of $|\mathcal{I}_Q|$ qeuries, we get that the computational complexity of AttnHead is $O(|\mathcal{I}_Q|\,|\mathcal{I}_K|\,d_{\text{attn}})$. We also have to consider the embedding of each token from $\mathbb{R}^d$ to $\mathbb{R}^{d_{\text{attn}}}$, which has a computational complexity of $O((|I_Q|+|I_K|)d_{\text{attn}}d)$ for each head's query, key, value and output embedding. By considering the total of H heads, we have proven

Lemma 2.1. The computational complexity of SelfAttn in its general form is

$$O\left(Hd_{\text{attn}}(|I_K|\,|I_Q| + (|I_Q| + |I_K|)d)\right).$$

Remark 2.2. *The choice of d_{attn} is typically made in a way so that the computational complexity of* SelfAttn *does not depend on H, i.e. $d_{attn} \propto 1/H$.*

We notice that the computational cost of self-attention in the vanilla Transformer scales quadratically in the sequence length L_{in}.

2.3.3 Layer Normalisation

Batch and layer normalisation techniques are designed to enhance the speed at which a model can be trained, enabling the use of larger learning rates. However, one challenge is the phenomenon of covariate shift, which was initially addressed by batch normalisation [IS15]. This refers to the shift in the input distribution that a model encounters when processing different batches. This distributional shift leads to a shift of the distribution of the model's parameters, which in turn shifts the distribution of the model's activations. The hypothesis put forth in [IS15] is that stabilizing the distribution of the internal model activations improves model training stability and speed. The proposed method is to normalise the model activations by considering batch statistics and normalising the first and second moments of this distribution in a dedicated BatchNorm layer.

While BatchNorm achieves this goal, [Xio+20] proposes replacing this layer with layer normalisation. The key argument is that Batch Normalisation is difficult to carry out in large distributed training settings, is not suitable for some model classes such as RNNs and overloads the batch size as a hyperparameter. In their paper, Xiong et al. propose normalising the mean and variance of the distribution of all hidden units corresponding to a single sample. This approach does not introduce dependencies between different training samples and therefore does not depend on the batch size. In the context of the Transformer, the samples in question are the tokens, as defined in [Xio+20]. Layer normalisation is applied after the self-attention layer and the MLP, respectively.

Definition 2.6. Let $X = (\boldsymbol{x}_i)_{i\in I} \in \mathcal{T}$ be the input to layer normalisation. Let $\gamma \in \mathbb{R}^d$ and $\beta \in \mathbb{R}^d$ be learnable weights, then we define

$$\text{LayerNorm}(X) := \left(\frac{\boldsymbol{x}_i - \hat{\mu}(\boldsymbol{x}_i)}{\hat{\sigma}(\boldsymbol{x}_i)} \odot \gamma + \beta\right)_{i\in\mathcal{I}},$$

where $\odot$ denotes the element-wise multiplication and for $\boldsymbol{y} \in \mathbb{R}^d$

$$\hat{\mu}(\boldsymbol{y}) = \frac{1}{d}\sum_{j=1}^{d} \boldsymbol{y}_j \qquad \text{and} \qquad \hat{\sigma}(\boldsymbol{y}) = \sqrt{\frac{1}{d}\sum_{j=1}^{d}(\boldsymbol{y}_j - \hat{\mu}(\boldsymbol{y})^2)}$$

2.3.4 Residual Connection

In the paper [He+16], residual connections are proposed as a means of more efficiently training deep neural networks. These connections form the basis of the well-known residual nets (ResNets) in computer vision, and make training of super deep neural networks with more than 150 layers possible. Moreover, residual nets can be regarded as the discrete analogue of Neural ODEs [Che+18; SAP22].

In the context of our training of Transformer models, we utilise them as a means of training our models in a more efficient manner. However, we analyse their contextualisation abilities in greater detail in chapter 4.

Definition 2.7. The residual connection around a model component $M : \mathcal{T} \to \mathcal{T}$ is given by

$$\text{ResConn}_M(X) := M(X) + X.$$

We remind the reader that the addition is token-wise.

2.3.5 Token-Processor: Multilayer Perceptron

MLPs are employed to process tokens. The token-processor is applied to each token independently, in a manner analogous to layer normalisation.

Definition 2.8. The *token multilayer perceptron (TokenMLP)* with one hidden layer, hidden layer size d_{hidden} and activation function $g : \mathbb{R} \to \mathbb{R}$ is defined as

$$\text{TokenMLP}(X) := \mathbf{MLP}_{d \to 1 \times d_{\text{hidden}} \to d, g}\,(X)$$

where $X = (\boldsymbol{x}_i)_{i \in \mathcal{I}} \in \mathcal{T}$.

We typically use MLPs with one hidden layer and a hidden dimension that is greater than d. Common choices for the activation function are ReLU and GeLU.

Some research has been conducted into the role of MLPs in the Transformer, with findings presented in [Gev+21; Gev+22]. At present, the role of MLPs (also known as *feed-forward layers* in the literature) is understood to involve the encoding of vocabulary concepts in key-value pairs. This can be conceptualised as a process of understanding the concept through the probing of learned questions. However, the precise role of MLPs in the context of time-series contextualisation has not yet been subjected to a comprehensive investigation.

2.3.6 Linear Decoder

The objective of the decoder is to compute the forecast based on the encoder representation. In the original presentation of the Transformer in [Vas+17], an iterative decoder producing

one forecasted timestamp after another is employed. This approach follows the Seq2Seq methodology outlined in [SVL14] and is still employed in NLP tasks. While this pattern had also been utilised for Transformers in the context of time-series forecasting, Zeng et al. demonstrated the efficacy of linear decoders in their work [Zen+23], introducing DLinear.

Definition 2.9. The *linear decoder* maps a collection of tokens $X \in \mathcal{T}$ to the time-series forecast of length T and is defined as

$$\text{LinDecoder}(X) := \text{Reshape}_{(N_{\text{in}}, T)}(\text{Lin}_{d\,|\mathcal{I}| \to N_{\text{in}} T}\,(\text{Flatten}_{\mathcal{I}}(X))).$$

A concrete definition of the Reshape and $\text{Flatten}_{\mathcal{J}}$ operations is given in appendix A.3. An additional path for investigation is the potential to alter the basis upon which the forecast is generated, i.e. forecasting coefficients for a representation of the forecast in another basis, to introduce an inductive bias. One such basis could be a wavelet basis. However, this is beyond the scope of the current work and is therefore left for future research.

2.4 Modifications of the Transformer Architecture

This section presents some of the modifications to the Transformer for time-series forecasting from the literature that are relevant to this work. The range of such modifications is very broad, which is why we start off by reviewing the literature. We then proceed to present three types of models (PatchTST, iTransformer, TSMixer). We then present a common generalisation for these models in chapter 3.

2.4.1 Literature Review

While there is a long-standing and active research community focusing on more general methods in time-series analysis, we restrict ourselves to recent developments based on neural networks and begin with the invention of the Transformer in 2017 by Vaswani et al.

Originally, the Transformer was developed for NLP tasks [Vas+17; Dev+19]. However, it has since been applied to a range of other applications. Applications such as computer vision [Dos+21] and speech processing [DXX18] have also benefited from the use of Transformers. Transformers have been demonstrated to be highly effective, with numerous state-of-the-art models relying on Transformers in their respective domains. This work will focus on the evolution of Transformers in the context of time-series forecasting.

It is important to note that there are numerous additional tasks based on time-series data, including classification [Zer+21] and anomaly detection [Xu+21; TCJ22]. These tasks are beyond the scope of this discussion, but it is worth mentioning that they are also areas where Transformers have been applied.

The modifications of the original Transformer can be broadly categorised into two groups: those at the component level and those at the architecture level.

The component-level modifications are further divided into two main branches: the first being the modifications of the self-attention mechanism. The $O(L_{\text{in}}^2)$ time-complexity of self-attention in the input sequence length L_{in} was the first significant drawback of self-

attention that was addressed in the literature. Two main approaches have been proposed to address this issue: the use of sparse and low-rank attention approximation schemes.

In LogTrans [LI+19], sparsity is introduced by reducing the set of keys considered in self-attention to the order $O(\log L_{\text{in}})$. More concretely, only keys at $O(\log L_{\text{in}})$-many fixed positions are selected. This achieves a time-complexity of $O(L_{\text{in}} \log L_{\text{in}})$. The utilisation of convolutional self-attention in LogTrans involves the application of a convolutional layer prior to the implementation of self-attention. Furthermore, it incorporates the inductive bias that observations occurring in close temporal proximity provide valuable semantic information.

In contrast to the fixed selection of $O(\log L_{\text{in}})$ keys in LogTrans, Informer proposes ProbSparse self-attention as an alternative [Zho+21]. The entropy of the probability distributions induced by the queries on the set of keys is analysed, and self-attention is conducted only with respect to the most sparse $O(\log L_{\text{in}})$ queries. The rationale is that queries inducing probability distributions, that are close to uniform, carry very litte information. This method is also able to reduce the time complexity to $O(L_{\text{in}} \log L_{\text{in}})$, but achieves a better performance compared to LogTrans because of the increased relevancy of the selected keys. Informer is also the first model to challenge the Seq2Seq decoder approach proposed in [SVL14] for time-series forecasting. It designs a one-step decoder that avoids the accumulation of errors encountered in iterative forecasting.

Pyraformer [Liu+22a] implements a pyramidal tree-like structure to self-attention, which effectively reduces the time complexity to $O(L_{\text{in}})$. In this tree-like hierarchical structure, information is aggregated from the local level to the global level with the help of self-attention. This is achieved while maintaining the signal communication path length constant with respect to the input sequence length L_{in} under mild conditions.

The Autoformer model [Wu+21] and the FEDformer model [Zho+22] represent a novel approach to the problem, departing from previous models in significant ways. The models incorporate the inductive bias of periodicity in time-series into the model by conducting self-attention on an overlay of periodic patterns based on autocorrelations in the time-series in the case of Autoformer and in the Fourier-space in the case of FEDformer. Both models reduce the time complexity by selecting relevant frequencies before the application of self-attention. Both approaches have heuristic arguments in favour of them, and Zhou et al. provides formal arguments in favour of FEDformer in their paper. However, these arguments cannot be confirmed from a mathematical standpoint.

While the aforementioned models primarily aim to reduce the time complexity of self-attention, there are also models that focus on other components of the Transformer. The non-stationary Transformer [Liu+22b] employs normalisation, de-normalisation and lastly de-stationary attention to accommodate shifts in the input time-series distribution.

A recent development is to change the model architecture and focus less on modifying the self-attention component. The patch embedding, which was first popularised in the Transformer literature by the Vision Transformer [Dos+21], is used in PatchTST [Nie+23]. Furthermore, the author of the aforementioned paper introduces the concept of variate independence (see Figure 2.2). The Crossformer model [ZY23] builds upon the ideas presented in [Nie+23] and introduces the use of self-attention along variates. The iTransformer

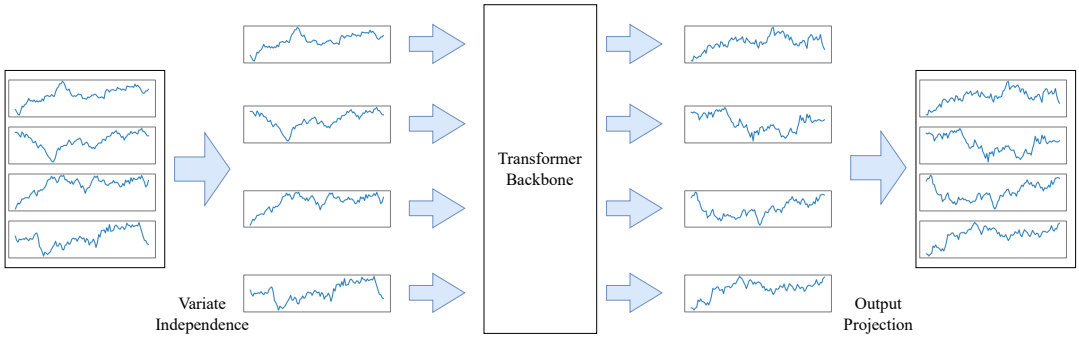

Figure 2.2: Illustration of variate independence; figure has been reproduced based on [Nie+23]

[Liu+24], completely discards the use of self-attention along the time and replaces it with an MLP-structure.

It should be noted that approaches from computer vision have also influenced the recent time-series forecasting literature. TimesNet [Wu+23] builds upon the ideas presented in Autoformer [Wu+21] and FEDformer [Zho+22]. However, it employs components from CV to extract interdependencies between observations belonging to the same frequency.

A parallel line of research was sparked by [Zen+23]. The authors question the efficacy of the previously proposed Transformer-based models and argue that their performance can be beaten with a simple linear model (DLinear) that produces a multistep forecast. While subsequent models, such as PatchTST, yield better results in numerical experiments, this line of research was continued. MLP-based models such as TSMixer [Che+23] or LightTS [Zha+22] employ analogous methodologies to those employed by the aforementioned Transformer-based models [Nie+23; Liu+24], but replace self-attention with an MLP.

2.4.2 Patching Time: PatchTST

PatchTST, an acronym for *patch time-series Transformer*, is a model proposed in [Nie+23]. It is a variate-independent model that employs a patch encoding.

The ideas behind PatchTST can be summarized as follows: Firstly, by patching several timestamps of the input time-series into one token, local information is aggregated in the tokens. Now, tokens are conceptually much closer to word-embeddings, because they carry very localised semantic information. Secondly, patching in time reduces the computational complexity from $O(L_{\text{in}}^2)$ to $O((L_{\text{in}}/P)^2)$, where P denotes the patch length. While it is questionable whether one can retain a competitive performance of the resulting model, it seems possible to scale up P with longer input sequences. Finally, the concept of variate independence, also known as *channel independence*, suggests that each variate is treated independently from the others, with the linear decoder aggregating information from several variates (see Figure 2.2). The work [Nie+23] is the first application of variate- or channel-independence in the Transformer architecture. Previous work in this direction was mostly done on convolutional neural networks (CNNs) and linear models [Zhe+14; Zen+23].

As stated by Nie et al., there are several reasons for the success of variate-independent ap-

proaches. Despite the application of the same self-attention and MLP layers to all variates, the model was still able to identify different attention patterns for each variate. This adaptability is attributed to the model's ability to utilise distinct regions of the attention space for different variate types. Furthermore, the size of commonly used time-series datasets may be insufficient for non-variate independent approaches. Experimentally, variate independent models converge more rapidly with less available training data. We wish to present another heuristic argument: in high-dimensional time-series data, period lengths are not uniform across all variates. Consequently, combining all observations at each timestamp into one token may be inappropriate. The contextualisation patterns of variate-independent approaches are studied in chapter 4.

Let us now define the operations of PatchTST that we need.

Definition 2.10. Let $X \in \mathbb{R}^{N_{\text{in}} \times L_{\text{in}}}$ be a time-series of length L_{in}. Let $P \in \{1, \dots L_{\text{in}}\}$ denote the patch length. Then PatchEmb maps a time-series to the token-space $\mathcal{T}_{\text{PatchTST}}$ corresponding to $\mathcal{X} := \mathbb{R}^d$ and $\mathcal{I}_{\text{PatchTST}} := \{1, \dots, N_{\text{in}}\} \times \{1, ..., L\}$ with $L = \lceil L_{\text{in}}/P \rceil$ by

$$\text{PatchEmb}(X) = \mathbf{Lin}_{P \to d}\left(\left(\text{Reshape}_{(N_{\text{in}}, L, P)}(X)\right)_{N_{\text{in}} \times L \times \underline{P}}\right),$$

where we potentially pad the input time-series X by repeating its last observation so that $L_{\text{in}} = LP$.

In contrast to the vanilla Transformer, the entire token representation X is not fed into self-attention; rather, self-attention is applied along the time-axis.

Definition 2.11. For $X = (\boldsymbol{x}_i)_{i \in \mathcal{I}_{\text{PatchTST}}} \in \mathcal{T}_{\text{PatchTST}}$. Self-attention along the time-axis is then defined as

$$\text{SelfAttn}_{\text{time}}(X) := \bigvee_{N_{\text{in}} \times \underline{L}} \mathbf{SelfAttn}\left(X_{N_{\text{in}} \times \underline{L}}\right).$$

We summarise PatchTST in Algorithm 2 and remark that the summation of PatchEmb(X) and TimePosEnc(L) is to be understood that the time positional encoding is added to each variate of the output of the patch embedding.

Algorithm 2: PatchTST

input : Time-series $X \in \mathbb{R}^{N_{\text{in}} \times L_{\text{in}}}$
output: Forecast $\hat{Y} \in \mathbb{R}^{N_{\text{in}} \times T}$

$X^{(0)} \leftarrow \text{PatchEmb}(X) + \text{TimePosEnc}(L)$;

for $l \leftarrow 1$ **to** n_{layers} **do**
  $\tilde{X}^{(l)} \leftarrow \text{ResConn}_{\text{SelfAttn}_{\text{time}}}\left(X^{(l-1)}\right)$;
  $\tilde{X}^{(l)} \leftarrow \text{LayerNorm}\left(\tilde{X}^{(l)}\right)$;
  $X^{(l)} \leftarrow \text{ResConn}_{\mathbf{TokenMLP}}\left(\tilde{X}^{(l)}\right)$;
  $X^{(l)} \leftarrow \text{LayerNorm}\left(X^{(l)}\right)$;
end

$\hat{Y} \leftarrow \text{LinDecoder}\left(X^{(n_{\text{layers}})}\right)$

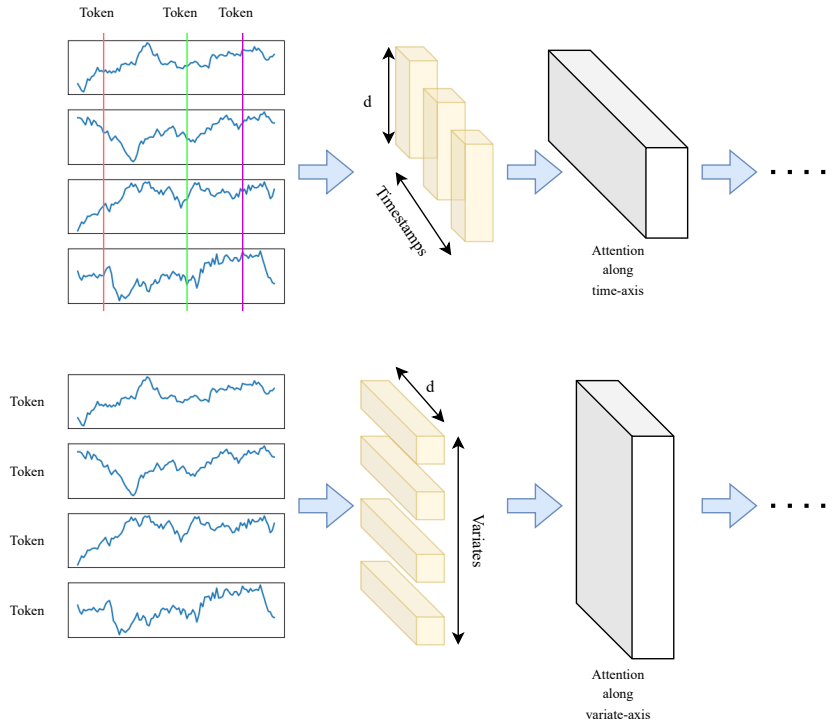

Figure 2.3: Idea of iTransformer: The Transformer does not mix time-based tokens, but rather variate-based tokens. Thus, we can see it as an inversion of the vanilla Transformer model. The illustration is reproduced from [Liu+24]

.

2.4.3 Self-Attention along Variate-Axis: iTransformer

Thus far, we have applied self-attention along the time-axis by comparing tokens corresponding to different timestamps in the input time-series. This has been challenged by Liu et al., who propose *iTransformer*, an inverted Transformer, to consider self-attention along the variate-axis [Liu+24]. This addresses the critique on self-attention along the time-axis by [Zen+23] and the recent success of MLP-based models, which we introduce as a baseline in the next section.

The key concept behind iTransformer is to use MLPs along the time-axis and self-attention along the variate-axis to extract patterns across several variates. The concept is illustrated in Figure 2.3.

The iTransformer can be readily formalised by simply swapping the time- and variate-axis in the vanilla Transformer, thereby transposing the single token input embedding. This results in the time positional encoding being dropped, which in turn means that the self-attention layer is unable to discern very similar time-series. Consequently, we can formalise the iTransformer as in Algorithm 3.

However, one significant challenge that remains unresolved in [Liu+24] is the efficient com-

Algorithm 3: iTransformer

input : Time-series $X \in \mathbb{R}^{N_{\text{in}} \times L}$
output: Forecast $\hat{Y} \in \mathbb{R}^{N_{\text{in}} \times T}$

$\mathcal{X}^{(0)} \leftarrow \text{SingleTokenInputEmb}(X^t)$;

for $l \leftarrow 1$ **to** n_{layers} **do**
  $\tilde{X}^{(l)} \leftarrow \text{ResConn}_{\text{SelfAttn}}\left(X^{(l-1)}\right)$;
  $\tilde{X}^{(l)} \leftarrow \text{LayerNorm}\left(\tilde{X}^{(l)}\right)$;
  $X^{(l)} \leftarrow \text{ResConn}_{\textbf{TokenMLP}}\left(\tilde{X}^{(l)}\right)$;
  $X^{(l)} \leftarrow \text{LayerNorm}\left(X^{(l)}\right)$;

$\hat{Y} \leftarrow \text{LinDecoder}\left(X^{(n_{\text{layers}})}\right)$

putation of attention along variates. In real-world time-series datasets containing thousands of variates, the self-attention operation has an $O(N_{\text{in}}^2)$ time complexity, which represents a significant computational bottleneck. In their attempt to address this issue, Liu et al. adopt a flexible training strategy. This is made possible by the inverted design of the model and the absence of a positional encoding, which allows for the use of a flexible number of variates. Consequently, the authors randomly select a subset of the variates for training and testing with all variates. They demonstrate that this strategy can retain the majority of the performance gains of the iTransformer while simultaneously significantly reducing training time. However, it should be noted that inference still has the $O(N_{\text{in}}^2)$ complexity. Furthermore, it is uncertain whether this strategy allows for the learning of complex patterns involving many variates in very complex datasets. Consequently, we later investigate the potential of efficient attention approximation schemes along variates for efficient inference and training using all variates at once in chapter 5.

2.4.4 Questioning Self-Attention: MLP-based models

As demonstrated by the iTransformer, an MLP is employed to process the tokens, which are computed from a variate. It is not self-attention that is used to extract any temporal patterns, but rather an MLP.

The rationale behind the use of MLPs instead of self-attention can be traced back to [Tol+21], who questioned the necessity of CNNs and Transformers in computer vision tasks. This proved to be a valuable extension of the critical linear ideas from [Zen+23] in the realm of time-series forecasting, leading to the development of the first adaptations of the MLP-Mixer from [Tol+21] to this field. Consequently, TSMixer was proposed in [Che+23], which was the first instance of such an application in the field of time-series forecasting. This involved the use of MLPs to mix along both the time- and variate-axes. This work focuses on the conceptually simpler case of TSMixer, rather than further developments such as LightTS [Zha+22], which constitute even more sparse models.

The authors of [Tol+21] present a conceptually straightforward rationale for the efficacy of linear models (and MLP extensions) in time-series forecasting tasks. They idenitify that, in practical applications, time-series data is typically characterised by smoothness

and periodicity. In all other cases, it is very challenging to accurately predict a time-series forecast. Now, consider a perfectly P-periodic and deterministic time-series X, where $X_{:,t} = X_{:,t+P}$ for $t = 1, \ldots, L - P$, where $P < L$. In this case, we have a linear model that perfectly predicts each variate of X. This linear model has a bias of zero and

$$A_{ij} = \begin{cases} 1, & \text{if } j = L - P + (i \bmod P), \\ 0, & \text{else} \end{cases}$$

as its weight matrix. Furthermore, the authors demonstrate that linear models exhibit robust predictive capabilities in scenarios where the time-series can be decomposed into a sum of a P-periodic component and a Lipschitz-continuous component. The Lipschitz constant is proportional to an upper bound on the model's error. This justifies the investigation of linear and hence MLP-based models.

The terminology used in the models introduced before differs significantly from that used in this work, which is why we begin by considering the raw time-series X of length L_{in} with N_{in} variates. A generalisation of TSMixer is provided in the following chapter, which allows the MLP-based model to be treated in a common framework with the other models presented in this chapter. The TSMixer with n_{layers} layers, activation function g and hidden dimension d_{hidden} is presented in Algorithm 4.

Algorithm 4: TSMixer

input : Time-series $X \in \mathbb{R}^{N_{\text{in}} \times L_{\text{in}}}$
output: Forecast $\hat{Y} \in \mathbb{R}^{N_{\text{in}} \times T}$

$X^{(0)} \leftarrow X$;

for $l \leftarrow 1$ **to** n_{layers} **do**

$$\tilde{X}^{(l)} \leftarrow \textbf{LayerNorm}\left(X^{(l-1)} + \bigvee\nolimits_{N_{\text{in}} \times \underline{L_{\text{in}}}} \boldsymbol{g}\left(\textbf{Lin}_{L_{\text{in}} \to L_{\text{in}}}\left(X^{(l-1)}_{N_{\text{in}} \times \underline{L_{\text{in}}}}\right)\right)\right);$$

$$X^{(l)} \leftarrow \textbf{LayerNorm}\left(\tilde{X}^{(l)} + \bigvee\nolimits_{\underline{N_{\text{in}}} \times L_{\text{in}}} \textbf{MLP}_{N_{\text{in}} \to 1 \times d_{\text{hidden}} \to N_{\text{in}}}\left(\tilde{X}^{(l)}_{\underline{N_{\text{in}}} \times L_{\text{in}}}\right)\right);$$

$$\hat{Y} \leftarrow \textbf{Lin}_{L_{\text{in}} \to T}\left(X^{(n_{\text{layers}})}_{N_{\text{in}} \times \underline{L_{\text{in}}}}\right);$$

It can be observed that the TSMixer architecture successively mixes time- and variate-information. It should be noted that the time-mixing operation in the TSMixer is not an MLP, but can be considered to be a reasonable approximation of an MLP structure.

Chapter 3

FlexibleTransformer

We think that a systematic study of existing ideas in the literature is important and can provide novel insights into the current state of the field of applying Transformers to time-series forecasting. Currently, it is the case that a comparison of models is only conducted in terms of performance on benchmark datasets without unifying the setting in which these architectures are compared. This perspective fails to acknowledge the intricacy and complexity of model architectures, which often comprise numerous interconnected components or employ disparate methodologies, such as the vanilla Transformer, PatchTST, iTransformer, and MLP-based models. It is challenging, if not impossible, to discern the fundamental elements of a given model architecture within this setup. Consequently, in order to study and compare the inner workings of these models, it is necessary to identify a common framework that is sufficiently abstract to provide interpretable flexibility while also being sufficiently concrete to allow for a systematic study. In this chapter, we introduce FlexibleTransformer, a novel generalisation of existing Transformer models. This framework allows us to formalise the models presented in chapter 2, with the goal of studying them in the next chapter from a standpoint of contextualisation. We begin by introducing the time-variate-token framework, which composes the model-internal data representation of the FlexibleTransformer. This is also where we formalise the token-mixing and token-processing operations that we have previously discussed. Finally, we examine how previously introduced models are implemented in the context of the FlexibleTransformer. While we do not seek to reinvent the Transformer and rely on the ideas presented in the literature, in particular the data representation proposed in [Nie+23] and the concept of operations along the variate-axis outlined in [Liu+24], our contributions can be summarised as follows:

- Formalisation: Introduction of the language of token spaces, token representations, operations along axes, definition of token-mixing and token-processing.
- Unification: Identification of common building blocks amongst current models following different architectural paradigms.
- MLP token-mixer: To sensibly compare MLP-based token-mixing with self-attention as a token-mixer, we have to lift the MLP to the level of combining different tokens while respecting the ideas from the token-unaware MLP-based models introduced in chapter 2.

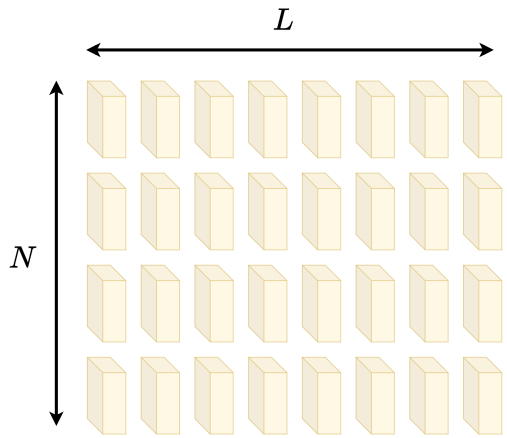

Figure 3.1: The time-variate-token framework formalises the idea of having a time- and a variate-axis and one token in $\mathbb{R}^d$ at each of the gridpoints.

It should be noted that the presentation of the models and their operations in the previous chapter has been aligned with the FlexibleTransformer. The notation and formalisation that has been used is not standard notation in the literature, but it is highly useful in our context.

3.1 Time-Variate-Token Framework

It should be noted that the concept of internal data representation by the time-variate-token framework is not entirely novel, but rather draws inspiration from the variate-independence approach outlined in [Nie+23].

This work builds upon the theoretical foundations established in chapter 2. Figure 3.1 illustrates the concept of a grid with a time- and a variate-axis, on which tokens are positioned.

Definition 3.1. Let $L \in \mathbb{N}$ denote the *internal number of timestamps*, $N \in \mathbb{N}$ the *internal number of variates* and $d \in \mathbb{N}$ the model dimension. Then the index set $\mathcal{I}$ for the token-space $\mathcal{T}$ in the *time-variate-token framework* is given by $\mathcal{I} := \{1, ..., N\} \times \{1, ..., L\}$ and $\mathcal{X} := \mathbb{R}^d$ in the definition of $\mathcal{T}$.

We elaborate on the choice of L and N later in this chapter and also give examples of corresponding input embeddings that are compatible with the choices of L and N. The reason why we choose $\mathcal{T}$ as in definition 3.1 is not because of variate independence as in [Nie+23], but because we want to flexibly mix tokens along the time- and variate-axis. We can now define precisely what we refer to by a token-mixer along an axis.

Definition 3.2. Let $\mathcal{T}$ be a token-space in the time-variate-token framework. A token-mixer is a map

$$\mathrm{TM} : \mathcal{X}^{\mathcal{J}} \to \mathcal{X}^{\mathcal{J}},$$

where $\mathcal{J} = \{1, ..., L\}$ in the case of token-mixer along the time-axis and $\mathcal{J} = \{1, ..., N\}$ in

the case of a token-mixer along the variate-axis. We further write

$$\begin{aligned} \mathrm{TM}_{\text{time}} : \mathcal{T} &\to \mathcal{T} \\ X &\mapsto \bigvee_{N \times \underline{L}} \mathbf{TM}\left(X_{N \times \underline{L}}\right) \end{aligned}$$

and

$$\begin{aligned} \mathrm{TM}_{\text{variate}} : \mathcal{T} &\to \mathcal{T} \\ X &\mapsto \bigvee_{\underline{N} \times L} \mathbf{TM}\left(X_{\underline{N} \times L}\right) \end{aligned}$$

Figure 3.2 illustrates the application of a token-mixer along a given axis. In the preceding chapter, we have seen self-attention as a token-mixer along the time- and the variate-axis. We can observe that SelfAttn is indeed a token-mixer in the sense of definition 3.2.

Let us now also give a definition for a token-processor.

Definition 3.3. Let $\mathcal{T}$ be a token space with index set $\mathcal{I}$ and $X = (\boldsymbol{x}_i)_{i \in \mathcal{I}} \in \mathcal{T}$. A token-processor is a map

$$\mathrm{TP} : \mathcal{X} \to \mathcal{X}.$$

We also write for $X = (\boldsymbol{x}_i)_{i \in \rangle}$ (with a slight abuse of notation)

$$\begin{aligned} \mathrm{TP} : \mathcal{T} &\to \mathcal{T} \\ X &\mapsto (\mathrm{TP}(\boldsymbol{x}_i))_{i \in \mathcal{I}}. \end{aligned}$$

We can identify MLPs and layer normalisation as examples of token-processors.

We want to close this section with the input embedding.

Definition 3.4. An *input embedding* for a time series of length L_{in} with N_{in} variates to a token space $\mathcal{T}$ is a map

$$\mathrm{InputEmb} : \mathbb{R}^{N_{\text{in}} \times L_{\text{in}}} \to \mathcal{T}.$$

The rather informal spirit of the input embedding (which we do not rigorously define) prevents the mixing of time and variates. However, the concrete choice of the input embedding is closely tied to the choice of $\mathcal{T}$. We will now summarise the ideas of this section by giving some examples of what the token spaces and the input embeddings can be. Examples include the following:

- Single token input embedding: We have seen this input embedding in the vanilla Transformer. It combines all variates into one token for each timestamp. Accordingly, $L = L_{\text{in}}$ and $N = 1$.
- Patch embedding: Here, local information is kept along time and variates by only combining nearby timestamps into patches. We thus have $L = \lceil L_{\text{in}}/P \rceil$ and $N = N_{\text{in}}$, where P denotes the patch length.
- Variate as token embedding: This is the input embedding of the iTransformer. It combines all timestamps of a variate into one token, i.e. $L = 1$ and $N = N_{\text{in}}$.

Figure 3.2: Illustration of how a token-mixer taking in all tokens (little towers representing elements in $\mathcal{X}$) along an axis is applied in the case of a (a) token-mixer along the time-axis and (b) token-mixer along the variate-axis. The colour indicates how the data is grouped before the application of the token-mixer.

3.2 Mixing Tokens with MLPs

As the reader may have observed, we have thus far only introduced one token-mixing operation: self-attention. In the previous chapter, we also considered MLP-based models in which MLP structures were employed to identify long-range patterns across time and variates. However, the notion of a token was absent, which is why we wish to introduce an MLP token-mixer structure that allows us to analyse these MLP-based models in the time-variate-token framework.

This is essentially a slightly new model that is conceptually very similar to the TSMixer. In order to support our argument that our proposal is appropriate, we also want to experimentally verify that the models are similar and identify the best model hyperparameters.

Let us begin by noting the key requirements for the proposed *MLP token-mixer*.

- MLP token-mixer is actually a token-mixer in the sense of definition 3.2.
- In the case $d = 1$, the MLP token-mixer corresponds to the variate-mixing step in the TSMixer architecture.
- As in TSMixer, the MLP token-mixer has a bottleneck structure, i.e. the hidden layer size is much smaller than the number of input layer size of the MLP.

In light of the aforementioned requirements, it appears that a natural extension of TSMixer

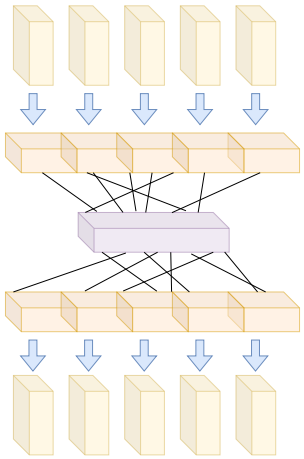

Figure 3.3: A sequence of tokens is mixed by the MLP token-mixer that flattens the tokens and uses an MLP to mix all the tokens information. Finally, the flattening is undone to produce a sequence of tokens again.

to the time-variate-token framework would be to flatten all tokens along an axis and pass them through an MLP with a modest hidden layer size. The output layer size would be the same as that of the input layer. Finally, the result would only have to be reshaped into tokens along the original dimensions. We thus formally define:

Definition 3.5. Let $\mathcal{X}$ and $\mathcal{J} = \{1, \ldots, A\}$ be as in definition 3.2 (with $A = L$ or $A = N$). We define the *MLP token-mixer* with hidden layer size d_{mixing} and activation function g as

$$\text{MLPTokenMixer}(X) := \left[\text{Reshape}_{(A,d)}\left(\text{MLP}_{dA \to d_{\text{mixing}} \to dA, g}\left(\text{Flatten}_{\mathcal{J}}(X)\right)\right)\right]_{A \times \underline{d}},$$

where $X = (\boldsymbol{x}_j)_{j \in \mathcal{J}}$

A detailed definition of the $\text{Flatten}_{\mathcal{J}}$ operation is given in appendix A.3. We illustrate the MLP token-mixer in Figure 3.3. It is immediate to see that this is precisely a variate-mixing step in TSMixer if we choose $d = 1$.

3.2.1 Experimental Comparison of MLP Token-Mixer and TSMixer

The objective of this experiment is to confirm that the MLP token-mixer layers in the time-variate-token framework produce similar results to the TSMixer. To this end, we will compare the model *pure MLP token-mixer model* outlined in Algorithm 5 with TSMixer. The choice of d_{mixing} does not have to be the same for the time- and the variate-mixer, but to allow for better comparability with the TSMixer model, we use the same mixing dimension d_{mixing} in both mixing-steps.

A comparison is presented between the performance of TSMixer and the pure MLP token-mixer model, with the optimal hyperparameters identified for various settings in terms of data and prediction length. This is presented in Table 3.1. While the objective is not to outperform existing benchmarks, it is of interest to utilise standard benchmark datasets to

Algorithm 5: PureMLPTokenMixerModel

input : Time-series $X \in \mathbb{R}^{N_{\text{in}} \times L_{\text{in}}}$
output: Forecast $\hat{Y} \in \mathbb{R}^{N_{\text{in}} \times T}$

$X^{(0)} \leftarrow X$;

for $l \leftarrow 1$ **to** n_{layers} **do**
  $\tilde{X}^{(l)} \leftarrow \textbf{LayerNorm}\left(X^{(l-1)} + \text{MLPTokenMixer}_{\text{time}}\left(X^{(l-1)}\right)\right)$;
  $X^{(l)} \leftarrow \textbf{LayerNorm}\left(\tilde{X}^{(l)} + \text{MLPTokenMixer}_{\text{variate}}\left(\tilde{X}^{(l-1)}\right)\right)$;

$\hat{Y} \leftarrow \text{LinDecoder}\left(X^{(n_{\text{layers}})}\right)$

Table 3.1: Experimental comparison of FlexibleTransformer with MLP token-mixer (identity and patch embedding) and TSMixer. The best models for each dataset are marked in red, the second best in blue.

Model	FlexibleTransformer					TSMixer		FlexibleTransformer				TSMixer	
Encoding	identity		patch			No Encoding		identity		patch		No Encoding	
	d_{mixing}	n_{layers}	d_{mixing}	d_{model}	n_{layers}	d_{hidden}	n_{layers}	MSE	MAE	MSE	MAE	MSE	MAE
dataset													
ETTh1	64	3	256	32	3	32	3	0.419	0.448	**0.414**	**0.441**	0.421	0.442
ETTh2	32	1	64	32	3	128	3	**0.247**	**0.350**	0.248	0.352	0.269	0.381
ETTm1	256	4	64	32	1	32	3	0.364	0.401	0.367	0.400	**0.352**	**0.394**
ETTm2	512	3	512	32	4	32	3	0.151	**0.258**	0.151	0.264	**0.151**	0.262
weather	64	4	32	32	4	32	1	0.269	**0.252**	0.271	0.254	**0.266**	0.267

facilitate comparability with a greater number of models. The datasets utilised are described in greater detail in appendix A.6. The optimal models are identified by determining the most effective parameter values for each model. We consider the ranges

$$\begin{aligned} d_{\text{mixing}} &\in \{32, 64, 128, 256, 512\} \\ d &= 32 \\ n_{\text{layers}} &\in \{1, 2, 3, 4\} \end{aligned}$$

for the FlexibleTransformer,

$$\begin{aligned} d_{\text{hidden}} &\in \{32, 64, 128, 256, 512\} \\ n_{\text{layers}} &\in \{1, 2, 3, 4\} \end{aligned}$$

for the TSMixer model and use training hyperparameters in

$$\begin{aligned} \text{dropout} &\in \{0.1, 0.3\} \\ \text{learning rate} &\in \{10^{-5}, 10^{-4}, 10^{-3}\} \\ \text{learning rate scheduler} &\in \{\text{one cycle}, \text{constant}\}, \end{aligned}$$

where the *one cycle* learning rate scheduler is inspired by [ST18].

Table 3.1 shows that the FlexibleTransformer with the MLP token-mixer as its time and variate token-mixer performs similarly to the TSMixer model on a time-series forecasting task for popular benchmark datasets with $L_{\text{in}} = T = 96$. The fitted hyperparameters and

Input Embedding	N	L
Single Token Input Embedding	1	L_{in}
Patch Embedding	N_{in}	$\lceil L_{\text{in}}/P \rceil$
iTransformer Embedding	N_{in}	1
TSMixer Embedding	N_{in}	L_{in}

Table 3.2: Parameters in various settings.

Model	Time Token-Mixer	Variate Token-Mixer	Token-Processor
Vanilla Transformer	Self-Attention	-	TokenMLP
PatchTST	Self-Attention	-	Token MLP
iTransformer	-	Self-Attention	TokenMLP
TSMixer	MLP Token-Mixer	MLP Token-Mixer	-

Table 3.3: Token-mixers for various models.

the mean squared error (MSE) and mean absolute error (MAE) are displayed in Table 3.1. Hereby, we have for $\hat{Y}, Y \in \mathbb{R}^{N_{\text{in}} \times T}$

$$\text{MSE}(\hat{Y}, Y) := \frac{1}{N_{\text{in}} T} \sum_{i=1}^{T} \sum_{n=1}^{N_{\text{in}}} (\hat{Y}_{n,i} - Y_{n,i})^2, \qquad \text{MAE}(\hat{Y}, Y) := \frac{1}{N_{\text{in}} T} \sum_{i=1}^{T} \sum_{n=1}^{N_{\text{in}}} |\hat{Y}_{n,i} - Y_{n,i}|.$$

The FlexibleTransformer with the identity embedding is the closest to the TSMixer model in terms of conceptual similarity. We have also included a model with a patch embedding (with patch size $P = 8$), which more closely resembles later model choices. The learning rate scheduler and dropout did not seem to have a major impact on training performance, whereas a learning rate of 10^{-4} proved to be optimal.

It can be concluded that the experimental evidence supports the claim that the MLP token-mixer is an appropriate analogue of the TSMixer component in the FlexibleTransformer architecture.

3.3 A Generalisation of Existing Models

In this section we want to illustrate that the models we have encountered so far are examples of the time-variate-token framework. Since we have only generalised the concepts from chapter 2, we will only indicate the components that have been used in the different settings.

Let us first consider the input embedding. We show the choices for N and L for the different input embeddings encountered so far in Table 3.2.

In Table 3.3, we identify the token-mixer along the time- and variate-axes and the token-processor used in the models discussed in chapter 2. Each model considered applies a layer normalisation after each mixing and processing step and the token-processors and token-mixers are wrapped in a ResConn layer.

We can see that the time-variate-token framework is flexible enough to accommodate a wide range of models. The FlexibleTransformer allows for even greater flexibility than just being a common framework for well-known time-series models, as we have seen above. In the

next section, we will systematically examine the impact of different architectural choices on forecasting performance.

3.4 Experiments

In this section we want to experimentally explore the FlexibleTransformer. As mentioned in the previous section, we want to go beyond the current model architectures that we have already generalised, and also explore architectural choices that have not yet been explored in the literature. The results of this section serve two purposes:

1. An extensive study of different model architectures with respect to performance, stability and size.

2. An exploration of well-performing model configurations to lay the foundation for an in-depth analysis of these models in the next chapter.

Our general approach is as follows: In general, we want to distinguish between different model architectures, characterised by the token-mixers and the token-processors. Since we have a wide range of possible hyperparameters (d, d_{hidden}, d_{mixing}, H, ...) to consider, we first find the best hyperparameters for the model architectures before comparing different architectures. We use the two standard evaluation metrics, the mean squared error and the mean absolute error.

We also want to report the sensitivity of the models to their hyperparameters, explore the stability of the models, and compare performance and model size.

3.4.1 Setup

In general, we want to explore all reasonable combinations of token-mixers and token-processors, but fix the input embedding by choosing the patch embedding with patch length $P = 8$. This patch size has shown to be adequate for this study's benchmark datasets [Nie+23] and is not of our primary interest. This helps us to reduce the total number of experiments to run. For brevity, we introduce the following notation for the central transformer component and write

$$\text{FT}_{d,N,L} : \text{TM}_{\text{time}}\,\text{TM}_{\text{variate}}\,\text{TP}$$

for the FlexibleTransformer architecture of size $\mathcal{I} = \{1, \dots, N\} \times \{1, \dots, L\}$ with the corresponding time and variate token-mixers and the corresponding token-processor. If a component is not used, we simply write "$-$" instead of SA^H for self-attention with H heads, $\text{MLP}_{\text{TM}}^{d_{\text{mixing}}}$ for the MLP token-mixer and $\text{MLP}_{\text{TP}}^{d_{\text{hidden}}}$ for the MLP token-processor. The choice of the activation function g is not our main goal, so we choose $g = \text{ReLU}$ in the MLP token-mixer and MLP token-processor. We want to explore all the different architectures that we can construct in the FlexibleTransformer setting with hyperparameters in the following range

$$\begin{aligned}
\text{TM}_{\text{time}}, \text{TM}_{\text{variate}} &\in \{\text{SA}^8\} \cup \{\text{MLP}_{\text{TM}}^{d_{\text{mixing}}} : d_{\text{mixing}} \in \{32, 64, 128\}\} \cup \{-\} \\
\text{TP} &\in \{\text{MLP}_{\text{TP}}^{256}\} \cup \{-\} \\
d &\in \{32, 64, 128, 256, 512\} \\
n_{\text{layers}} &\in \{1, 2, 3\}
\end{aligned}$$

The range of model architecture hyperparameters is informed by the research literature on TSMixer, PatchTST, iTransformer and vanilla Transformer, own exploratory experiments, and further constrained by computational resources. We fit a total of 750 models for each experimental setting.

We fix the training hyperparameters to

$$\begin{aligned} \text{dropout} &= 0.1 \\ \text{learning rate scheduler} &= \text{constant} \\ \text{learning rate} &= 10^{-4} \end{aligned}$$

as we cannot afford to further increase the number of models to fit and have found in early, exploratory runs that the training hyperparameters do not depend by much on the model architecture used. For optimisation, we use the Adam optimiser [KB14] to optimise the MSE-error training objective.

While the training time for a single model is relatively short, ranging from one to ten minutes on an Nvidia GA100, depending on the exact model size and N_{in} and L_{in}, we have a computational limit on the number of datasets and combinations of input lengths L_{in} and prediction lengths T for which we can run the experiments. Hence, we only consider

$$\begin{aligned} L_{\text{in}} &\in \{96, 512\}, \\ T &\in \{96, 192, 336, 512\} \\ \text{dataset} &\in \{\text{ETTh1}, \text{ETTh2}, \text{ETTm1}, \text{ETTm2}, \text{weather}\}. \end{aligned}$$

See appendix A.6 for more information about the datasets and their properties. We do not consider datasets with large N_{in} because of the $O(N_{\text{in}}^2)$ complexity of self-attention along the variate-axis. This problem is discussed in chapter 5. To the author's knowledge, this is the most systematic comparison of different time series architectures in different data settings.

3.4.2 Results and Analysis

We structure this section as follows. First, we present the results of our experiments. Since we fit a large number of models, we only present the raw results for one dataset and list the rest in the appendix. We then explore the dependence of model performance on key model hyperparameters, analyse how certain model components promote performance in different settings and finally relate model size to performance.

Raw Experiment Results

The total number of trained models is 6000 for each dataset. Therefore, it is impossible to present all the experimental results in this chapter or in the appendix. Instead, we present the raw experimental results for the ETTh1 dataset with $L_{\text{in}} = T = 96$ in Table 3.4. The other configurations for L_{in} and T are shown in appendix A.7. We refrain from presenting the raw experimental results for all other datasets and advise the interested reader to consult the accompanying repository, which contains all these data in digital form. There, we also present the l^∞-error. Due to the sheer number of models, we do not analyse the results with the aim of drawing conclusions at this stage, but defer this to later analyses.

		TP	No Processing										TokenMLP									
		Metric	MAE					MSE					MAE					MSE				
		d	32	64	128	256	512	32	64	128	256	512	32	64	128	256	512	32	64	128	256	512
VM	TM	n_{layers}																				
$\text{MLP}^{128}_{\text{TM}}$	$\text{MLP}^{128}_{\text{TM}}$	1	0.449	0.444	0.443	0.444	0.453	0.434	0.420	0.418	0.424	0.431	0.446	0.441	0.439	0.455	0.447	0.422	0.411	0.415	0.426	0.420
		2	0.446	0.442	0.440	0.440	0.441	0.427	0.416	0.412	0.415	0.416	0.444	0.441	0.440	0.448	0.444	0.417	0.419	0.417	0.422	0.419
		3	0.447	0.444	0.445	0.443	0.449	0.426	0.416	0.421	0.419	0.425	0.448	0.444	0.439	0.443	0.452	0.427	0.414	0.414	0.417	0.427
	No Mixing	1	0.443	0.437	0.437	0.442	0.442	0.432	0.422	0.424	0.427	0.432	0.438	0.434	0.432	0.432	0.431	0.423	0.417	0.413	0.414	0.414
		2	0.443	0.437	0.435	0.430	0.435	0.428	0.419	0.420	0.408	0.415	0.440	0.434	0.431	0.430	0.431	0.425	0.414	0.411	0.406	0.409
		3	0.439	0.435	0.433	0.430	0.434	0.420	0.414	0.411	0.404	0.414	0.439	0.433	0.430	0.431	0.435	0.422	0.412	0.407	0.407	0.415
	SA	1	0.441	0.435	0.436	0.438	0.438	0.427	0.415	0.416	0.418	0.414	0.439	0.439	0.436	0.440	0.437	0.423	0.423	0.418	0.416	0.411
		2	0.441	0.438	0.437	0.442	0.441	0.425	0.417	0.414	0.418	0.417	0.442	0.437	0.443	0.438	0.443	0.425	0.416	0.425	0.416	0.418
		3	0.441	0.438	0.438	0.436	0.447	0.424	0.416	0.414	0.413	0.422	0.443	0.442	0.440	0.445	0.448	0.426	0.418	0.415	0.420	0.424
$\text{MLP}^{32}_{\text{TM}}$	$\text{MLP}^{32}_{\text{TM}}$	1	0.452	0.447	0.444	0.443	0.444	0.447	0.436	0.427	0.427	0.429	0.449	0.436	0.437	0.443	0.442	0.435	0.414	0.420	0.421	0.421
		2	0.453	0.444	0.443	0.440	0.443	0.441	0.429	0.421	0.416	0.417	0.451	0.439	0.443	0.435	0.441	0.435	0.423	0.415	0.413	0.416
		3	0.451	0.445	0.441	0.442	0.440	0.437	0.425	0.417	0.419	0.413	0.450	0.445	0.443	0.440	0.444	0.432	0.422	0.425	0.420	0.419
	No Mixing	1	0.452	0.445	0.442	0.448	0.452	0.448	0.434	0.433	0.445	0.451	0.442	0.436	0.432	0.433	0.437	0.430	0.424	0.417	0.416	0.421
		2	0.450	0.445	0.440	0.437	0.440	0.443	0.436	0.428	0.422	0.425	0.443	0.440	0.436	0.430	0.432	0.429	0.425	0.421	0.410	0.412
		3	0.448	0.442	0.438	0.434	0.435	0.436	0.430	0.424	0.417	0.417	0.443	0.439	0.431	0.436	0.437	0.427	0.422	0.412	0.414	0.414
	SA	1	0.444	0.438	0.436	0.439	0.442	0.432	0.426	0.418	0.415	0.423	0.440	0.437	0.434	0.435	0.439	0.428	0.424	0.415	0.412	0.418
		2	0.445	0.439	0.439	0.438	0.435	0.430	0.423	0.421	0.415	0.409	0.446	0.439	0.440	0.435	0.443	0.433	0.423	0.420	0.410	0.420
		3	0.446	0.440	0.438	0.440	0.438	0.431	0.421	0.416	0.413	0.415	0.446	0.442	0.437	0.447	0.439	0.430	0.425	0.413	0.427	0.414
$\text{MLP}^{64}_{\text{TM}}$	$\text{MLP}^{64}_{\text{TM}}$	1	0.453	0.446	0.442	0.446	0.447	0.440	0.429	0.420	0.424	0.427	0.447	0.440	0.440	0.436	0.443	0.428	0.409	0.412	0.410	0.422
		2	0.452	0.444	0.441	0.440	0.442	0.439	0.426	0.422	0.414	0.421	0.445	0.442	0.438	0.439	0.446	0.423	0.420	0.412	0.413	0.422
		3	0.448	0.443	0.440	0.440	0.441	0.433	0.423	0.417	0.415	0.416	0.451	0.445	0.439	0.442	0.442	0.432	0.420	0.410	0.415	0.413
	No Mixing	1	0.446	0.442	0.439	0.442	0.443	0.438	0.433	0.427	0.431	0.431	0.441	0.434	0.433	0.434	0.432	0.429	0.419	0.417	0.418	0.414
		2	0.447	0.440	0.435	0.436	0.438	0.438	0.427	0.419	0.421	0.422	0.441	0.436	0.432	0.428	0.432	0.427	0.421	0.414	0.407	0.411
		3	0.444	0.438	0.437	0.433	0.433	0.431	0.420	0.417	0.411	0.411	0.441	0.435	0.431	0.431	0.432	0.425	0.418	0.409	0.407	0.412
	SA	1	0.441	0.438	0.436	0.434	0.438	0.428	0.420	0.417	0.409	0.414	0.442	0.441	0.436	0.442	0.441	0.429	0.427	0.416	0.418	0.418
		2	0.445	0.439	0.441	0.446	0.437	0.430	0.420	0.420	0.424	0.413	0.442	0.438	0.440	0.435	0.443	0.427	0.420	0.420	0.415	0.421
		3	0.444	0.439	0.439	0.441	0.437	0.428	0.422	0.421	0.417	0.415	0.443	0.439	0.445	0.438	0.436	0.426	0.417	0.420	0.414	0.411
No Mixing	$\text{MLP}^{128}_{\text{TM}}$	1	0.462	0.452	0.458	0.461	0.453	0.459	0.439	0.440	0.446	0.438	0.448	0.453	0.452	0.457	0.463	0.433	0.444	0.430	0.431	0.449
		2	0.461	0.455	0.460	0.452	0.464	0.452	0.437	0.440	0.437	0.442	0.456	0.457	0.457	0.476	0.492	0.431	0.437	0.437	0.464	0.491
		3	0.459	0.454	0.455	0.454	0.466	0.446	0.431	0.432	0.436	0.452	0.466	0.461	0.469	0.464	0.482	0.445	0.434	0.458	0.446	0.470
	$\text{MLP}^{32}_{\text{TM}}$	1	0.461	0.452	0.455	0.456	0.459	0.463	0.450	0.452	0.457	0.441	0.454	0.450	0.441	0.462	0.463	0.448	0.441	0.421	0.440	0.441
		2	0.456	0.452	0.450	0.451	0.460	0.452	0.451	0.436	0.448	0.447	0.456	0.449	0.459	0.459	0.451	0.444	0.436	0.436	0.451	0.437
		3	0.463	0.453	0.448	0.460	0.455	0.463	0.443	0.435	0.440	0.442	0.460	0.451	0.462	0.462	0.470	0.453	0.438	0.435	0.439	0.452
	$\text{MLP}^{64}_{\text{TM}}$	1	0.460	0.451	0.448	0.448	0.449	0.462	0.444	0.441	0.432	0.433	0.453	0.449	0.447	0.446	0.469	0.442	0.427	0.420	0.426	0.448
		2	0.460	0.451	0.459	0.463	0.456	0.454	0.438	0.444	0.446	0.440	0.455	0.452	0.462	0.455	0.465	0.436	0.427	0.439	0.432	0.442
		3	0.461	0.458	0.456	0.459	0.472	0.453	0.441	0.438	0.441	0.458	0.458	0.457	0.455	0.450	0.465	0.447	0.439	0.425	0.426	0.449
	No Mixing	1	0.471	0.468	0.464	0.466	0.468	0.488	0.486	0.482	0.485	0.486	0.445	0.441	0.438	0.434	0.443	0.440	0.435	0.430	0.424	0.432
		2	0.470	0.467	0.466	0.466	0.471	0.488	0.484	0.487	0.481	0.495	0.448	0.442	0.444	0.433	0.438	0.443	0.436	0.436	0.423	0.425
		3	0.470	0.468	0.466	0.467	0.470	0.487	0.486	0.485	0.485	0.485	0.446	0.442	0.438	0.437	0.435	0.441	0.435	0.432	0.425	0.421
	SA	1	0.451	0.448	0.447	0.449	0.447	0.450	0.449	0.443	0.440	0.439	0.448	0.444	0.445	0.444	0.449	0.444	0.438	0.434	0.434	0.445
		2	0.455	0.449	0.449	0.446	0.456	0.456	0.449	0.443	0.440	0.444	0.450	0.444	0.448	0.442	0.452	0.445	0.439	0.441	0.430	0.445
		3	0.452	0.448	0.445	0.449	0.449	0.449	0.444	0.437	0.445	0.439	0.447	0.445	0.442	0.448	0.446	0.443	0.439	0.434	0.434	0.435
SA	$\text{MLP}^{128}_{\text{TM}}$	1	0.449	0.464	0.513	0.560	0.501	0.433	0.451	0.531	0.598	0.505	0.454	0.462	0.460	0.479	0.507	0.444	0.456	0.452	0.474	0.513
		2	0.451	0.457	0.475	0.497	0.477	0.436	0.451	0.474	0.507	0.483	0.459	0.471	0.458	0.469	0.479	0.446	0.469	0.449	0.458	0.486
		3	0.456	0.487	0.457	0.475	0.491	0.441	0.489	0.447	0.471	0.490	0.466	0.472	0.465	0.476	0.556	0.447	0.470	0.452	0.467	0.608
	$\text{MLP}^{32}_{\text{TM}}$	1	0.445	0.443	0.443	0.450	0.447	0.436	0.432	0.425	0.438	0.428	0.445	0.442	0.449	0.445	0.466	0.437	0.428	0.438	0.436	0.457
		2	0.448	0.449	0.442	0.470	0.436	0.438	0.433	0.424	0.474	0.416	0.451	0.450	0.448	0.455	0.460	0.441	0.440	0.436	0.455	0.461
		3	0.449	0.456	0.450	0.452	0.452	0.438	0.448	0.434	0.444	0.445	0.455	0.453	0.448	0.450	0.477	0.447	0.437	0.437	0.437	0.468
	$\text{MLP}^{64}_{\text{TM}}$	1	0.448	0.448	0.465	0.451	0.531	0.441	0.432	0.449	0.431	0.580	0.451	0.450	0.459	0.491	0.482	0.442	0.441	0.453	0.498	0.497
		2	0.452	0.450	0.454	0.449	0.486	0.445	0.430	0.441	0.435	0.498	0.454	0.446	0.444	0.449	0.504	0.440	0.428	0.429	0.439	0.529
		3	0.455	0.452	0.468	0.465	0.461	0.443	0.437	0.469	0.461	0.447	0.456	0.454	0.464	0.466	0.491	0.436	0.440	0.451	0.456	0.499
	No Mixing	1	0.442	0.435	0.433	0.435	0.434	0.435	0.424	0.417	0.415	0.415	0.445	0.435	0.437	0.435	0.438	0.437	0.424	0.420	0.414	0.413
		2	0.440	0.437	0.435	0.441	0.434	0.435	0.426	0.419	0.420	0.418	0.445	0.437	0.437	0.435	0.433	0.438	0.427	0.421	0.414	0.415
		3	0.440	0.436	0.435	0.433	0.437	0.434	0.422	0.417	0.413	0.417	0.443	0.434	0.432	0.431	0.442	0.435	0.423	0.414	0.407	0.421
	SA	1	0.441	0.436	0.436	0.448	0.443	0.435	0.423	0.424	0.434	0.427	0.446	0.439	0.438	0.442	0.450	0.440	0.428	0.423	0.427	0.436
		2	0.441	0.440	0.444	0.448	0.444	0.434	0.428	0.424	0.437	0.439	0.444	0.437	0.439	0.448	0.455	0.437	0.422	0.422	0.432	0.442
		3	0.442	0.439	0.450	0.444	0.457	0.434	0.425	0.433	0.426	0.439	0.444	0.440	0.454	0.445	0.452	0.437	0.425	0.438	0.428	0.440

Table 3.4: Raw experiment results for ETTh1 dataset with $L_{\text{in}} = T = 96$. The rest of the results for this dataset can be found in appendix A.7. We mainly present this raw data for the sake of completeness. The reader may safely skip the detailed study of this table as we visualise the main relationships that we are interested in.

The fitted hyperparameters for all scenarios are shown in Table 3.7. We observe that d_{mixing} is mostly chosen to be rather small to form a bottleneck structure. This is consistent with the choice in [Che+23] for the TSMixer. On the other hand, d varies greatly between architectures, making it worthwhile to adjust this hyperparameter in practical settings. It is interesting to note that the number of layers is rarely chosen to be the highest in our setting.

Best Architectures and Stability

We refit the models with the best hyperparameter five times to analyse the stability of model performance. We show these results for the ETTh1 dataset with $L_{\text{in}} = 96$ in Table 3.5 and for $L_{\text{in}} = 512$ in Table 3.6. The results for other datasets are shown in appendix

T	Time-Mixer	Variate-Mixer	Token-Processor	MSE	MAE	MMaxError
96	MLP Token-Mixer	MLP Token-Mixer	No Processing	0.417 ± 0.004	0.443 ± 0.002	1.574 ± 0.010
			TokenMLP	0.421 ± 0.007	0.441 ± 0.003	1.589 ± 0.017
		No Mixing	No Processing	0.409 ± 0.002	0.432 ± 0.001	1.565 ± 0.007
			TokenMLP	0.412 ± 0.003	0.433 ± 0.003	1.563 ± 0.007
		Self-Attention	No Processing	0.420 ± 0.009	0.441 ± 0.005	1.581 ± 0.011
			TokenMLP	0.417 ± 0.006	0.439 ± 0.005	1.568 ± 0.007
	No Mixing	MLP Token-Mixer	No Processing	0.441 ± 0.005	0.460 ± 0.003	1.619 ± 0.009
			TokenMLP	0.430 ± 0.002	0.451 ± 0.003	1.595 ± 0.007
		No Mixing	No Processing	0.483 ± 0.001	0.466 ± 0.002	1.686 ± 0.008
			TokenMLP	0.426 ± 0.003	0.439 ± 0.003	1.589 ± 0.005
		Self-Attention	No Processing	0.442 ± 0.004	0.448 ± 0.002	1.618 ± 0.009
			TokenMLP	0.435 ± 0.002	0.447 ± 0.003	1.597 ± 0.008
	Self-Attention	MLP Token-Mixer	No Processing	0.436 ± 0.014	0.447 ± 0.007	1.605 ± 0.022
			TokenMLP	0.437 ± 0.003	0.448 ± 0.003	1.608 ± 0.012
		No Mixing	No Processing	0.415 ± 0.005	0.435 ± 0.005	1.562 ± 0.008
			TokenMLP	0.415 ± 0.003	0.435 ± 0.002	1.560 ± 0.005
		Self-Attention	No Processing	0.425 ± 0.002	0.437 ± 0.001	1.581 ± 0.004
			TokenMLP	0.427 ± 0.004	0.443 ± 0.002	1.575 ± 0.007
192	MLP Token-Mixer	MLP Token-Mixer	No Processing	0.502 ± 0.012	0.490 ± 0.008	1.895 ± 0.010
			TokenMLP	0.525 ± 0.012	0.517 ± 0.007	1.889 ± 0.019
		No Mixing	No Processing	0.458 ± 0.005	0.464 ± 0.003	1.809 ± 0.013
			TokenMLP	0.457 ± 0.002	0.464 ± 0.001	1.796 ± 0.011
		Self-Attention	No Processing	0.468 ± 0.007	0.474 ± 0.005	1.820 ± 0.016
			TokenMLP	0.469 ± 0.003	0.474 ± 0.002	1.820 ± 0.013
	No Mixing	MLP Token-Mixer	No Processing	0.544 ± 0.011	0.512 ± 0.004	1.986 ± 0.018
			TokenMLP	0.507 ± 0.004	0.494 ± 0.004	1.899 ± 0.005
		No Mixing	No Processing	0.534 ± 0.001	0.495 ± 0.000	1.954 ± 0.003
			TokenMLP	0.482 ± 0.003	0.471 ± 0.002	1.850 ± 0.011
		Self-Attention	No Processing	0.499 ± 0.007	0.482 ± 0.004	1.893 ± 0.012
			TokenMLP	0.495 ± 0.003	0.478 ± 0.002	1.878 ± 0.005
	Self-Attention	MLP Token-Mixer	No Processing	0.500 ± 0.017	0.487 ± 0.009	1.891 ± 0.025
			TokenMLP	0.502 ± 0.012	0.493 ± 0.008	1.868 ± 0.012
		No Mixing	No Processing	0.474 ± 0.002	0.472 ± 0.002	1.828 ± 0.006
			TokenMLP	0.472 ± 0.003	0.472 ± 0.003	1.809 ± 0.007
		Self-Attention	No Processing	0.487 ± 0.014	0.481 ± 0.010	1.847 ± 0.019
			TokenMLP	0.485 ± 0.004	0.476 ± 0.002	1.854 ± 0.015
336	MLP Token-Mixer	MLP Token-Mixer	No Processing	0.586 ± 0.003	0.535 ± 0.004	2.235 ± 0.005
			TokenMLP	0.573 ± 0.012	0.537 ± 0.008	2.177 ± 0.014
		No Mixing	No Processing	0.513 ± 0.006	0.500 ± 0.002	2.069 ± 0.017
			TokenMLP	0.509 ± 0.005	0.499 ± 0.002	2.046 ± 0.012
		Self-Attention	No Processing	0.537 ± 0.012	0.515 ± 0.007	2.116 ± 0.023
			TokenMLP	0.538 ± 0.015	0.516 ± 0.009	2.110 ± 0.028
	No Mixing	MLP Token-Mixer	No Processing	0.664 ± 0.036	0.579 ± 0.016	2.296 ± 0.029
			TokenMLP	0.583 ± 0.005	0.535 ± 0.004	2.214 ± 0.007
		No Mixing	No Processing	0.585 ± 0.002	0.524 ± 0.001	2.203 ± 0.005
			TokenMLP	0.540 ± 0.002	0.506 ± 0.001	2.103 ± 0.002
		Self-Attention	No Processing	0.563 ± 0.013	0.519 ± 0.007	2.170 ± 0.019
			TokenMLP	0.548 ± 0.003	0.512 ± 0.002	2.137 ± 0.007
	Self-Attention	MLP Token-Mixer	No Processing	0.572 ± 0.010	0.532 ± 0.009	2.194 ± 0.007
			TokenMLP	0.570 ± 0.013	0.533 ± 0.010	2.175 ± 0.019
		No Mixing	No Processing	0.537 ± 0.005	0.510 ± 0.003	2.102 ± 0.013
			TokenMLP	0.532 ± 0.006	0.505 ± 0.004	2.069 ± 0.012
		Self-Attention	No Processing	0.543 ± 0.006	0.512 ± 0.005	2.124 ± 0.010
			TokenMLP	0.566 ± 0.014	0.530 ± 0.008	2.102 ± 0.022
512	MLP Token-Mixer	MLP Token-Mixer	No Processing	0.646 ± 0.013	0.571 ± 0.005	2.485 ± 0.017
			TokenMLP	0.626 ± 0.013	0.575 ± 0.008	2.414 ± 0.023
		No Mixing	No Processing	0.564 ± 0.006	0.529 ± 0.003	2.336 ± 0.014
			TokenMLP	0.549 ± 0.009	0.527 ± 0.004	2.277 ± 0.032
		Self-Attention	No Processing	0.579 ± 0.015	0.542 ± 0.009	2.347 ± 0.026
			TokenMLP	0.577 ± 0.004	0.539 ± 0.002	2.339 ± 0.012
	No Mixing	MLP Token-Mixer	No Processing	0.685 ± 0.026	0.604 ± 0.012	2.464 ± 0.027
			TokenMLP	0.653 ± 0.023	0.582 ± 0.017	2.451 ± 0.028
		No Mixing	No Processing	0.619 ± 0.002	0.547 ± 0.001	2.420 ± 0.005
			TokenMLP	0.587 ± 0.003	0.537 ± 0.002	2.356 ± 0.007
		Self-Attention	No Processing	0.604 ± 0.003	0.544 ± 0.002	2.395 ± 0.005
			TokenMLP	0.594 ± 0.004	0.541 ± 0.003	2.363 ± 0.006
	Self-Attention	MLP Token-Mixer	No Processing	0.688 ± 0.015	0.604 ± 0.005	2.459 ± 0.028
			TokenMLP	0.618 ± 0.006	0.564 ± 0.005	2.424 ± 0.012
		No Mixing	No Processing	0.584 ± 0.005	0.543 ± 0.004	2.353 ± 0.012
			TokenMLP	0.577 ± 0.010	0.534 ± 0.007	2.336 ± 0.022
		Self-Attention	No Processing	0.582 ± 0.005	0.542 ± 0.004	2.348 ± 0.014
			TokenMLP	0.589 ± 0.005	0.541 ± 0.003	2.355 ± 0.007

Table 3.5: Errors for best models on ETTh1 dataset with $L_{\text{in}} = 96$ including standard deviations for a total of five runs. Entries marked in red are the best result for each T and entries marked in blue are the second best.

T	Time-Mixer	Variate-Mixer	Token-Processor	MSE	MAE	MMaxError
96	MLP Token-Mixer	MLP Token-Mixer	No Processing	0.443 ± 0.007	0.467 ± 0.006	1.614 ± 0.006
			TokenMLP	0.416 ± 0.003	0.449 ± 0.002	1.573 ± 0.009
		No Mixing	No Processing	0.410 ± 0.003	0.441 ± 0.001	1.559 ± 0.008
			TokenMLP	0.406 ± 0.003	0.441 ± 0.003	1.548 ± 0.005
		Self-Attention	No Processing	0.407 ± 0.002	0.443 ± 0.001	1.541 ± 0.004
			TokenMLP	0.410 ± 0.003	0.445 ± 0.002	1.555 ± 0.005
	No Mixing	MLP Token-Mixer	No Processing	0.474 ± 0.008	0.485 ± 0.006	1.659 ± 0.017
			TokenMLP	0.452 ± 0.011	0.471 ± 0.007	1.624 ± 0.018
		No Mixing	No Processing	0.451 ± 0.001	0.467 ± 0.001	1.608 ± 0.006
			TokenMLP	0.425 ± 0.001	0.450 ± 0.001	1.569 ± 0.003
		Self-Attention	No Processing	0.426 ± 0.002	0.453 ± 0.001	1.570 ± 0.002
			TokenMLP	0.430 ± 0.002	0.457 ± 0.001	1.571 ± 0.003
	Self-Attention	MLP Token-Mixer	No Processing	0.438 ± 0.010	0.460 ± 0.009	1.600 ± 0.011
			TokenMLP	0.436 ± 0.007	0.463 ± 0.009	1.587 ± 0.006
		No Mixing	No Processing	0.407 ± 0.001	0.438 ± 0.002	1.536 ± 0.004
			TokenMLP	0.424 ± 0.007	0.452 ± 0.003	1.552 ± 0.005
		Self-Attention	No Processing	0.413 ± 0.010	0.444 ± 0.007	1.540 ± 0.010
			TokenMLP	0.416 ± 0.002	0.444 ± 0.002	1.547 ± 0.004
192	MLP Token-Mixer	MLP Token-Mixer	No Processing	0.485 ± 0.006	0.495 ± 0.004	1.873 ± 0.012
			TokenMLP	0.505 ± 0.010	0.505 ± 0.005	1.909 ± 0.014
		No Mixing	No Processing	0.458 ± 0.005	0.475 ± 0.003	1.850 ± 0.007
			TokenMLP	0.449 ± 0.002	0.471 ± 0.002	1.814 ± 0.009
		Self-Attention	No Processing	0.455 ± 0.003	0.477 ± 0.001	1.807 ± 0.007
			TokenMLP	0.456 ± 0.004	0.476 ± 0.002	1.823 ± 0.011
	No Mixing	MLP Token-Mixer	No Processing	0.548 ± 0.011	0.529 ± 0.007	1.977 ± 0.016
			TokenMLP	0.533 ± 0.009	0.522 ± 0.005	1.938 ± 0.010
		No Mixing	No Processing	0.488 ± 0.003	0.491 ± 0.001	1.870 ± 0.007
			TokenMLP	0.465 ± 0.004	0.477 ± 0.002	1.816 ± 0.011
		Self-Attention	No Processing	0.469 ± 0.002	0.481 ± 0.002	1.834 ± 0.003
			TokenMLP	0.473 ± 0.006	0.485 ± 0.003	1.834 ± 0.010
	Self-Attention	MLP Token-Mixer	No Processing	0.530 ± 0.017	0.527 ± 0.010	1.918 ± 0.016
			TokenMLP	0.527 ± 0.016	0.520 ± 0.014	1.914 ± 0.015
		No Mixing	No Processing	0.454 ± 0.003	0.471 ± 0.003	1.788 ± 0.010
			TokenMLP	0.461 ± 0.004	0.477 ± 0.002	1.786 ± 0.006
		Self-Attention	No Processing	0.462 ± 0.009	0.478 ± 0.006	1.803 ± 0.010
			TokenMLP	0.469 ± 0.005	0.482 ± 0.005	1.806 ± 0.009
336	MLP Token-Mixer	MLP Token-Mixer	No Processing	0.563 ± 0.024	0.540 ± 0.015	2.197 ± 0.036
			TokenMLP	0.597 ± 0.026	0.563 ± 0.014	2.187 ± 0.025
		No Mixing	No Processing	0.494 ± 0.005	0.499 ± 0.001	2.098 ± 0.021
			TokenMLP	0.486 ± 0.002	0.497 ± 0.002	2.062 ± 0.009
		Self-Attention	No Processing	0.498 ± 0.010	0.507 ± 0.006	2.067 ± 0.022
			TokenMLP	0.496 ± 0.005	0.507 ± 0.002	2.059 ± 0.007
	No Mixing	MLP Token-Mixer	No Processing	0.636 ± 0.021	0.589 ± 0.009	2.222 ± 0.018
			TokenMLP	0.647 ± 0.032	0.593 ± 0.019	2.227 ± 0.028
		No Mixing	No Processing	0.521 ± 0.003	0.516 ± 0.002	2.103 ± 0.009
			TokenMLP	0.500 ± 0.000	0.503 ± 0.001	2.052 ± 0.003
		Self-Attention	No Processing	0.516 ± 0.006	0.515 ± 0.003	2.095 ± 0.018
			TokenMLP	0.511 ± 0.003	0.513 ± 0.001	2.078 ± 0.007
	Self-Attention	MLP Token-Mixer	No Processing	0.564 ± 0.030	0.556 ± 0.021	2.134 ± 0.016
			TokenMLP	0.574 ± 0.010	0.552 ± 0.010	2.174 ± 0.013
		No Mixing	No Processing	0.504 ± 0.007	0.507 ± 0.006	2.047 ± 0.008
			TokenMLP	0.500 ± 0.004	0.504 ± 0.002	2.036 ± 0.011
		Self-Attention	No Processing	0.498 ± 0.004	0.504 ± 0.003	2.042 ± 0.010
			TokenMLP	0.555 ± 0.044	0.546 ± 0.032	2.093 ± 0.045
512	MLP Token-Mixer	MLP Token-Mixer	No Processing	0.632 ± 0.018	0.580 ± 0.010	2.431 ± 0.021
			TokenMLP	0.641 ± 0.016	0.588 ± 0.007	2.410 ± 0.022
		No Mixing	No Processing	0.527 ± 0.005	0.524 ± 0.002	2.290 ± 0.016
			TokenMLP	0.533 ± 0.008	0.530 ± 0.004	2.301 ± 0.021
		Self-Attention	No Processing	0.542 ± 0.009	0.536 ± 0.006	2.320 ± 0.017
			TokenMLP	0.543 ± 0.007	0.538 ± 0.004	2.309 ± 0.011
	No Mixing	MLP Token-Mixer	No Processing	0.704 ± 0.031	0.627 ± 0.015	2.457 ± 0.031
			TokenMLP	0.668 ± 0.027	0.609 ± 0.011	2.424 ± 0.024
		No Mixing	No Processing	0.549 ± 0.004	0.537 ± 0.002	2.286 ± 0.012
			TokenMLP	0.546 ± 0.003	0.534 ± 0.002	2.277 ± 0.008
		Self-Attention	No Processing	0.552 ± 0.005	0.542 ± 0.003	2.283 ± 0.008
			TokenMLP	0.561 ± 0.008	0.545 ± 0.005	2.306 ± 0.019
	Self-Attention	MLP Token-Mixer	No Processing	0.668 ± 0.014	0.615 ± 0.007	2.393 ± 0.018
			TokenMLP	0.631 ± 0.014	0.592 ± 0.008	2.398 ± 0.022
		No Mixing	No Processing	0.543 ± 0.009	0.533 ± 0.006	2.266 ± 0.021
			TokenMLP	0.543 ± 0.004	0.535 ± 0.003	2.264 ± 0.003
		Self-Attention	No Processing	0.547 ± 0.004	0.540 ± 0.005	2.275 ± 0.017
			TokenMLP	0.546 ± 0.007	0.536 ± 0.006	2.276 ± 0.016

Table 3.6: Errors for best models on ETTh1 dataset with $L_{\text{in}} = 512$ including standard deviations for a total of five runs. Entries marked in red are the best result for each T and entries marked in blue are the second best.

L_{in}	T	Token Processor	Time Mixer / Variate Mixer	MLP Token-Mixer d_{mixing}	MLP Token-Mixer d	MLP Token-Mixer n_{layers}	No Mixing d_{mixing}	No Mixing d	No Mixing n_{layers}	Self-Attention d_{mixing}	Self-Attention d	Self-Attention n_{layers}
96	96	No Processing	MLP Token-Mixer	128	128	2	128	64	3	32	512	2
			No Mixing	128	256	3	-	256	2	-	256	3
			Self-Attention	32	512	2	-	128	3	-	64	1
		TokenMLP	MLP Token-Mixer	64	64	1	64	128	1	32	64	1
			No Mixing	128	256	2	-	512	3	-	256	3
			Self-Attention	32	256	2	-	256	2	-	128	2
	192	No Processing	MLP Token-Mixer	32	256	3	64	32	2	32	128	2
			No Mixing	128	256	3	-	64	3	-	128	2
			Self-Attention	128	128	3	-	64	3	-	128	1
		TokenMLP	MLP Token-Mixer	64	256	3	64	64	3	64	64	3
			No Mixing	128	128	3	-	256	1	-	256	3
			Self-Attention	128	256	3	-	64	1	-	64	1
	336	No Processing	MLP Token-Mixer	32	64	3	32	256	3	32	128	1
			No Mixing	64	512	3	-	128	1	-	64	2
			Self-Attention	32	256	3	-	64	3	-	32	3
		TokenMLP	MLP Token-Mixer	64	64	3	64	64	1	32	64	2
			No Mixing	64	256	3	-	256	2	-	128	3
			Self-Attention	64	128	3	-	64	2	-	256	2
	512	No Processing	MLP Token-Mixer	64	64	2	128	64	3	32	128	3
			No Mixing	128	512	2	-	128	3	-	128	1
			Self-Attention	32	256	3	-	32	2	-	64	1
		TokenMLP	MLP Token-Mixer	32	64	3	32	64	2	32	32	3
			No Mixing	128	256	3	-	256	2	-	128	1
			Self-Attention	64	32	3	-	32	2	-	32	3
512	96	No Processing	MLP Token-Mixer	64	64	2	32	32	2	32	64	1
			No Mixing	128	32	2	-	32	3	-	64	3
			Self-Attention	128	32	2	-	32	3	-	64	1
		TokenMLP	MLP Token-Mixer	128	64	2	32	64	2	64	64	1
			No Mixing	128	64	3	-	64	1	-	256	1
			Self-Attention	64	128	3	-	64	2	-	32	1
	192	No Processing	MLP Token-Mixer	64	32	3	32	32	1	64	32	3
			No Mixing	128	128	3	-	32	1	-	32	3
			Self-Attention	32	32	2	-	32	2	-	64	1
		TokenMLP	MLP Token-Mixer	32	64	2	32	32	1	32	64	2
			No Mixing	32	64	2	-	64	3	-	64	1
			Self-Attention	128	32	1	-	64	3	-	32	2
	336	No Processing	MLP Token-Mixer	32	64	2	64	32	3	32	64	2
			No Mixing	64	128	2	-	64	3	-	32	2
			Self-Attention	32	64	2	-	64	3	-	32	1
		TokenMLP	MLP Token-Mixer	64	32	3	32	64	2	32	32	1
			No Mixing	128	64	1	-	64	1	-	32	1
			Self-Attention	64	32	2	-	32	1	-	64	3
	512	No Processing	MLP Token-Mixer	32	32	2	128	32	1	64	32	3
			No Mixing	128	32	3	-	32	3	-	32	1
			Self-Attention	32	64	1	-	32	3	-	32	3
		TokenMLP	MLP Token-Mixer	32	32	2	32	32	1	64	32	1
			No Mixing	64	32	1	-	64	2	-	32	1
			Self-Attention	128	32	2	-	64	2	-	32	2

Table 3.7: Fitted hyperparameters for different choices of token-mixers and token processors in various settings for the ETTh1 dataset.

A.7. We advise the reader to also take a look at the results in the appendix. We observe the following:

- We do not observe major stability issues for the models.
- We observe similar results for all three evaluation metrics. One might remark that the use of self-attention along the time-axis brings about a slight relative improvement in the l^∞ (mean maximum error) evaluation metric.
- We observe that the best architectures strongly depend on the choice of the dataset, L_{in} and T. This requires a more detailed analysis which we conduct below.
- Two major trends can be observed, while their strength depends on the dataset at hand:
  - We generally tend to favor the use of no token-mixer along the variate-axis or the use of self-attention as a variate-mixer. Using MLP token-mixers along the variate-axis is rarely competitive. Using self-attention is only slightly worse than not using any token-mixer along the variate-axis.
  - As opposed to the case of the variate-axis, we want to use token-mixers along the time-axis. We tend to see the pattern that self-attention performs better for larger L_{in} and T relative to the MLP token-mixer for mixing along the time-axis, which is particularly pronounced for the weather dataset.

While we cannot give any theoretical reasons for the observed, we interpret these results as follows: The comparisons of the self-attention mechanism seem to yield the best performance in settings where many of them can be conducted or long-range patterns have to be found, i.e. large L_{in} or T. This agrees with the dominance of Transformers in NLP, where this model architecture yielded a breakthrough in many applications and especially in the understanding of long texts.

The rare strength of self-attention as the variate-mixer for the weather dataset could be connected with the relatively low correlation between the variates in the weather dataset (see appendix A.6). Such a possible relation should be explored further in the next chapter.

Hyperparameter dependence

For practical reasons, it is extremely useful to have an overview of the dependence of model performance on its hyperparameters. This also informs the choice of hyperparameters for models that appear later in this thesis. We want to give an overview of how the model hyperparameters d, n_{layers} and d_{mixing} affect the model performance. We restrict ourselves to visualise the case $L_{\text{in}} = T = 96$. We further reduce the interdependencies between the different hyperparameters in our plots by selecting the best performing model (in terms of l^2 error) for each fixed hyperparameter value. The relative dependence on the hyperparameters is shown for all datasets in Figures 3.4, 3.5 and 3.6. We plot the relative dependence, i.e. for the hyperparameters $h_1, \dots, h_n$ we observe the prediction l^2-error $e_{h_1}, \dots, e_{h_n}$ in which case we plot

$$h_k \mapsto \frac{e_{h_k}}{\min\{e_{h_1}, \dots, e_{h_n}\}}.$$

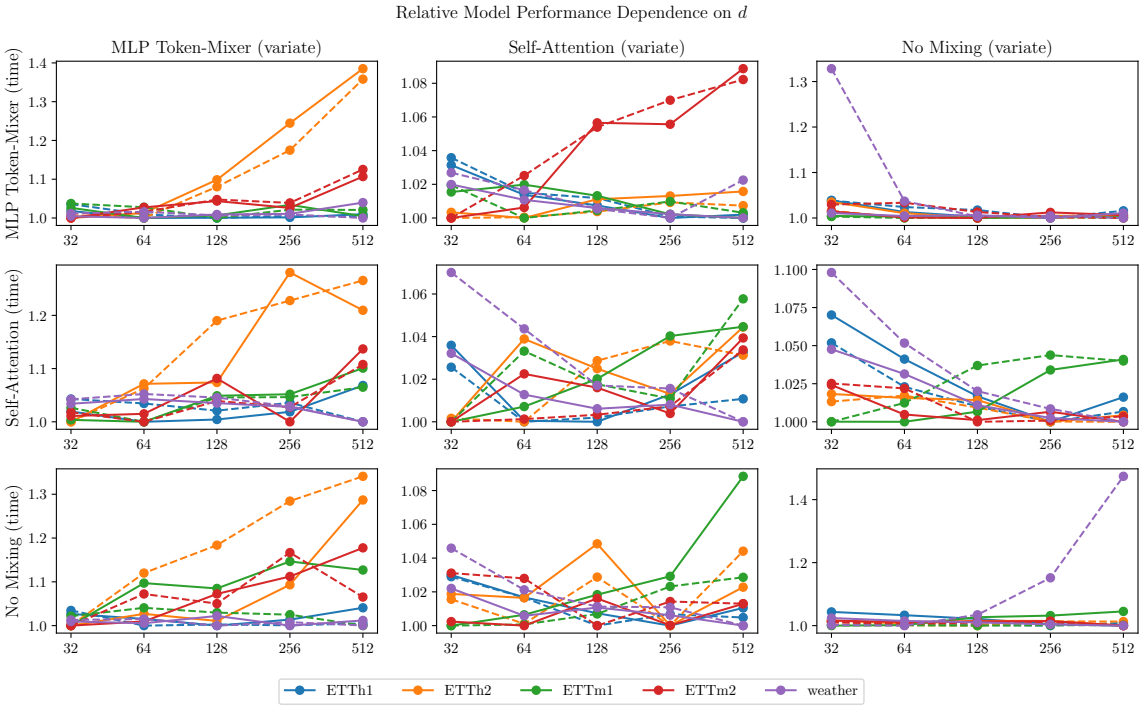

Figure 3.4: Dependence of the model performance on the hyperparameter d. Dashed lines indicate that the MLP token-processor was not used in the model.

This allows us to compare hyperparameter dependencies across several datasets.

We observe in Figure 3.4 that lower model dimensions d seem to be preferred in cases where an MLP token-mixer is used to mix along the variate-axis. In cases where we do not use the MLP token-mixer, the situation is less clear. The relationship between d and model performance does not seem to be monotonic when mixing with self-attention along the variate-axis. Surprisingly, we observe monotonic but strongly dependendent (on the dataset) relation when we mix with self-attention along the time-axis and do not mix along the variate-axis. In these cases, we suggest fitting this hyperparameter in practical applications.

We cannot see a clear pattern between model performance and n_{layers} in Figure 3.5. The actual relationship seems to depend mostly on the dataset used.

In cases where an MLP token-mixer is used, we observe in Figure 3.6 that we prefer lower values for d_{mixing} when mixing along the variate-axis. As we can see, this relationship depends on how strong the dependence of the different variates is. When mixing along the time-axis, higher values for d_{mixing} improve model performance. This is interesting since it shows that the semantic information along the time-axis requires a larger hidden dimension in the MLP token-mixer.

In all cases, the use of the MLP token processor can positively or negatively affect model performance, and we do not observe a strict preference depending on particular hyperparameter combinations.

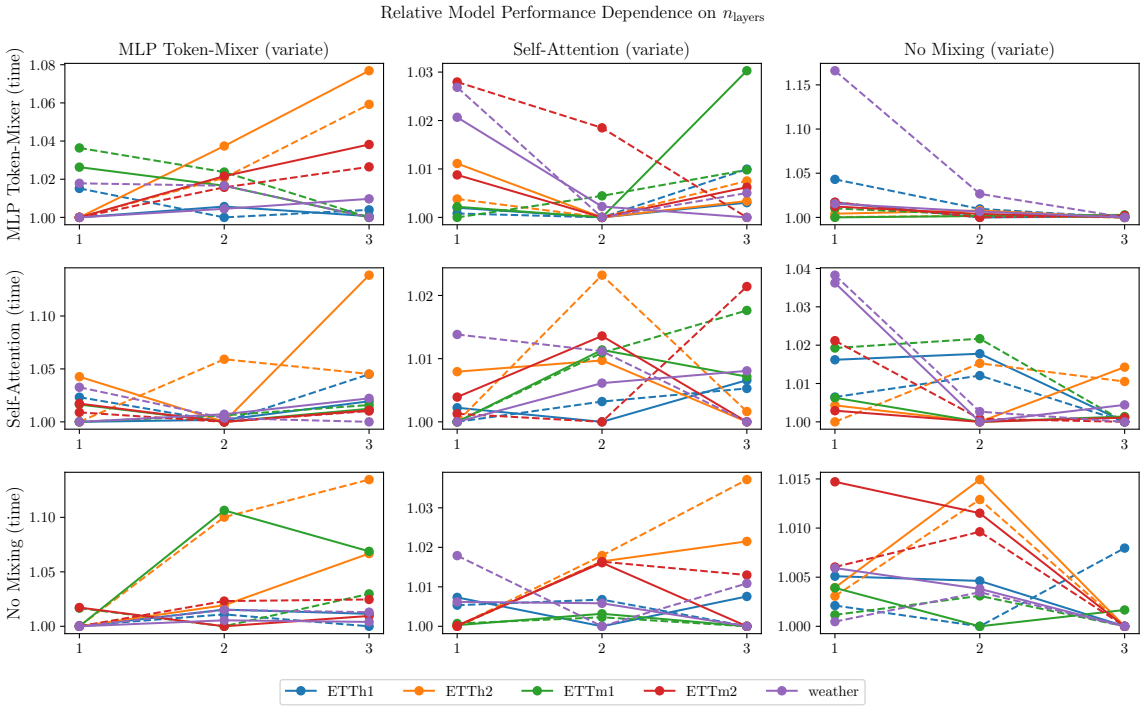

Figure 3.5: Dependence of the model performance on the hyperparameter n_{layers}. Dashed lines indicate that the MLP token-processor was not used in the model.

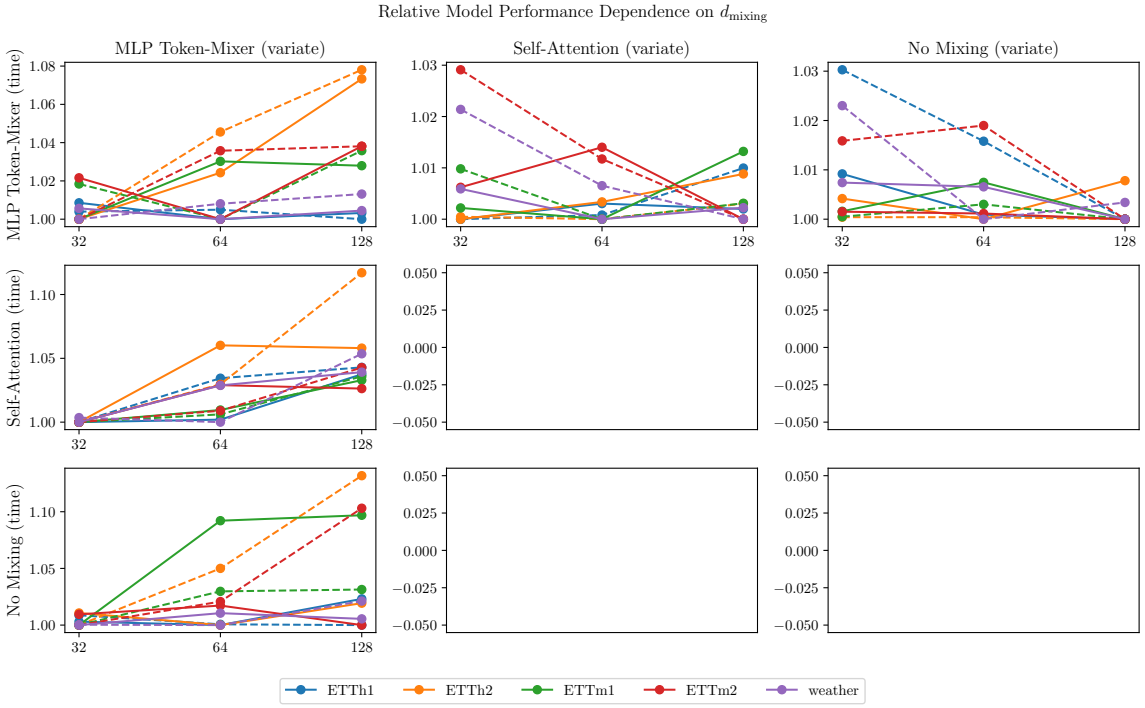

Figure 3.6: Dependence of the model performance on the hyperparameter d_{mixing}. Dashed lines indicate that the MLP token-processor was not used in the model. Some plots are empty because no MLP token-mixer was used in these cases.

Performance Promotion by Component in Different Settings

We now want to analyse in greater detail how the performance of the model changes depending on the configuration of the FlexibleTransformer, i.e. the token-mixers and token-processors used, and the setting, i.e. L_{in} and T. Before running the experiments, we expected that certain model components would perform better in some settings than in others, for the following reasons:

- The innate structure of mixing across time- and variate-axes is different. While we have temporal dependencies in time-mixing, the variates themselves are interchangeable. This may make self-attention better suited to mixing along the variate-axis due to its permutation equi-variance.

- The structure of long-term and short-term forecasts is different. In long-term forecasting, it is much more important to capture long-range patterns in the data. The thesis of the literature on the application of Transformers to time-series is that Transformer-based models are well suited to extracting non-linear long-range patterns in time-series data.

We show the promotion of performance by using the token-mixers and token-processors described in this chapter over not using any token-mixer (respectively token-processor) in Figure 3.7 for the ETTh1 dataset. The plots for the other datasets can be found in appendix A.7. The reader is advised to take a look at them.

Before analysing performance promotion, it should be noted that the baseline in Figure 3.7 is *no mixing*. The shown relative performance value is computed by

$$1 - \frac{\text{MSE of model with respective model-component (along axis)}}{\text{MSE of model without token-mixer (respectively token-processor) (along axis)}}.$$

This is not equivalent to having no token-mixers at all, but simply marks the relative improvement by fixing the time-mixer (or variate-mixer or token-processor) to a proper token-mixer (or token-processor) along the respective axis instead of not mixing along this axis.

We observe clear patterns in Figure 3.7: Fixing the token-processor has a better effect for larger forecast horizons T. This is shared for all datasets. The relative effect of setting the MLP token-mixer to mix along the variate-axis is worse than setting the variate-mixer to self-attention. This is consistent with our previous evaluation results. The slight preference for MLP token-mixing along the time-axis can also be observed in Figure 3.7. The dependence on the effects described above is (almost monotonically) stronger for longer forecast horizons T. The choice between $L_{\text{in}} \in \{96, 512\}$ doubles the effect on the choice of token-mixer, but has a negligible effect on the choice of token-processor. The nature of the patterns observed in this study are fairly insensitive to the dataset at hand, as can be seen for the other datasets in appendix A.7. The only difference is that we have a vertical shift for some of the datasets with a generally better or worse performance. We attribute this to the unique properties of the datasets.

As a rough guide we can say the following: For longer forecast horizons T and longer input time series L_{in} we have to be more careful with the choice of the right token-mixer and token-processor along the time-axis. For shorter forecasts, we would rather not use

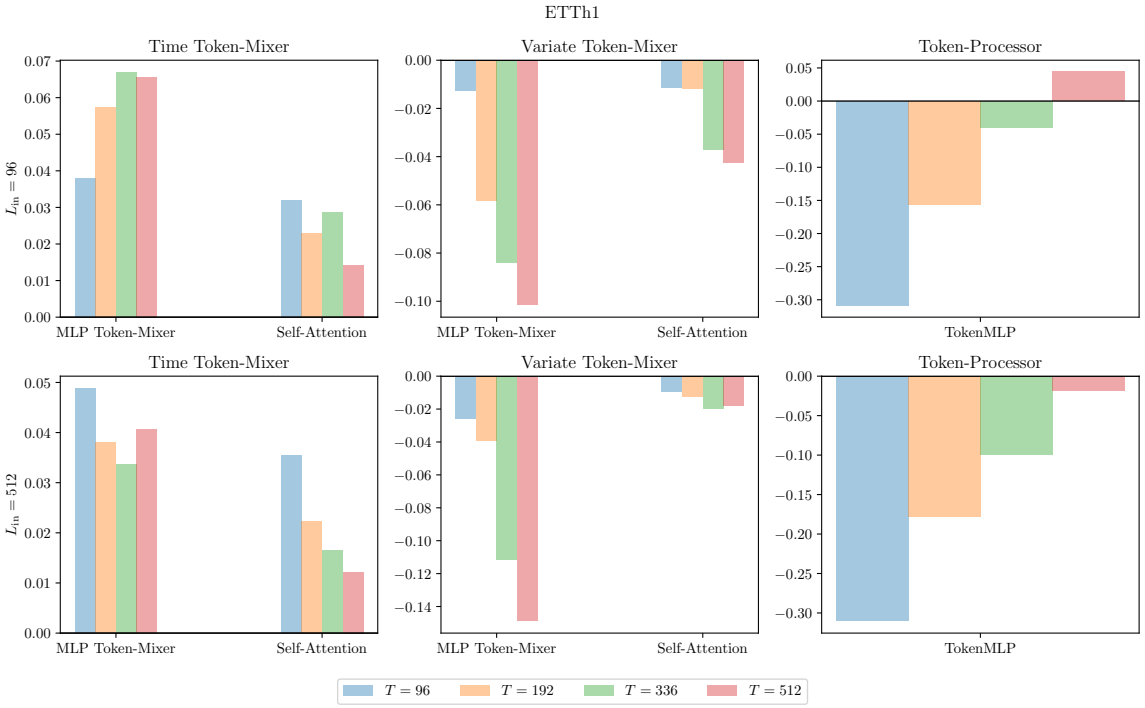

Figure 3.7: Performance promotion by using different token-mixers and token-processors for the ETTh1 dataset. The baseline is not mixing (and respectively not processing). Positive values indicate that using the best model with the respective token-mixer/token-processor along the respectve axis is better than the best model with not using a token-mixer/token-processor along the respective axis.

the token-processor as opposed to for longer forecast horizons. We can generally advise against using a token-mixer along the variate-axis. The use of the token-processor should be determined based on T. For longer forecasting horizons, it might be useful to use a token-processor.

Model Size

We also want to look at the relationship between model size and performance. In particular, we are interested in whether there are models that are small but still perform well. Model size is mainly influenced by the choice of hyperparameters that influence the size of the models' layers. Figure 3.8 shows this relationship. Using the MLP token-mixer to mix along the time-axis, using the MLP token-processor and not using any token-mixers to mix along the variate-axis seems to be a good compromise in most situations where low model size is paramount.

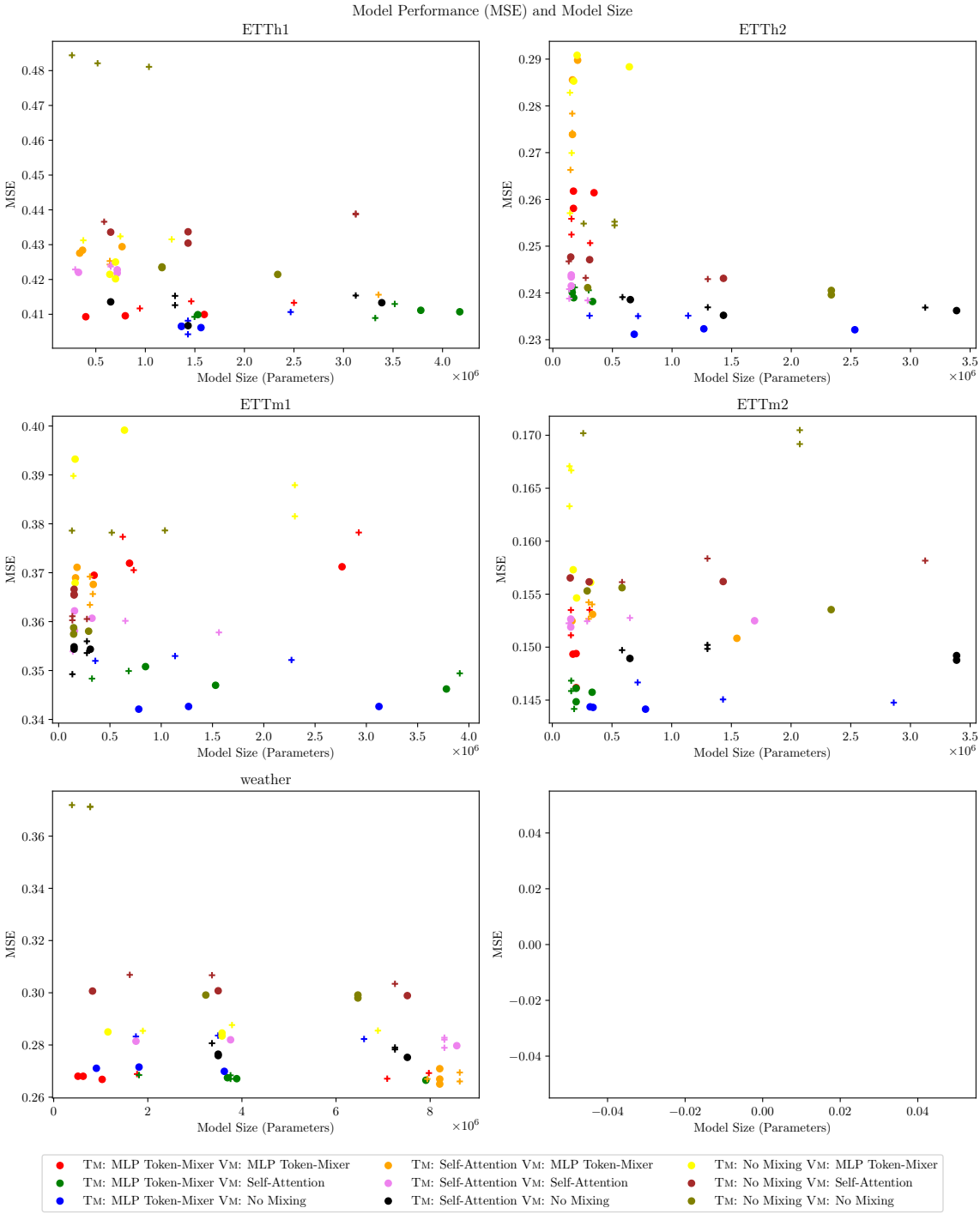

Figure 3.8: Plot depicts relation between model size (number of parameters) and MSE loss for each dataset. We display the best three hyperparameter model configurations for each combination of token-mixers. Round markers indicate that MLP token-processors are included, while this is not the case for crossed markers.

Chapter 4

Contextualisation in the FlexibleTransformer

The objective of this chapter is to conduct a systematic analysis of the contextualisation patterns in the FlexibleTransformer. This analysis will be conducted by examining different token-mixing schemes along different axes and by using different methods. To the author's knowledge, this is the first such study in the context of time-series.

The general approach is as follows: In order to relate the token representation at each step in the forward network flow to the inputs, feature attribution methods will be employed. Subsequently, we examine contextualisation patterns and apply a methodology analogous to the norm-based analysis proposed in [Kob+21]. Afterwards, we conduct a series of comprehensive experiments to find the relationship between contextualisation patterns and the properties of the input time-series.

The idea for this research was inspired by Kobayashi et al., who analysed the contextualisation patterns of feed-forward networks in BERT [Dev+19] and GPT [Bro+20] models in [Kob+24].

Our contributions in this chapter are the following:

- Extend the norm-based analysis to the time-variate-token framework.
- Decompose all token-mixers in the time-variate-token framework and adapt the results from [Kob+21] for self-attention, layer normalisation and residual connections and the results from [Kob+24] for MLPs to this extended setting.
- Change the attribution scores to be end-to-end attribution: In contrast to the approaches proposed in [Kob+21; Kob+24], our method does not compute attribution scores for a single layer in isolation. Instead, we relate the outputs of each layer to the inputted time-series. This approach has the advantage of respecting that some updates performed by later layers are relatively unimportant because the inputs have already been scaled down.
- As we construct an end-to-end feature attribution, we are able to provide interpretable

time-series Transformer models.

In the initial section of this chapter, we present the contextualisation approach previously outlined. This allows us to demonstrate how model forecasts can be related to input time-series data. Subsequently, we employ the contextualisation approach to conduct a comprehensive study of the FlexibleTransformer in the latter part of the chapter.

4.1 Decomposition Analysis

The purpose of this analysis is to examine how specific model components influence the internal data representation, with the objective of identifying the crucial components in the FlexibleTransformer. The approach presented below is based on the concept of *norm-based analysis*, which was popularised in [Kob+21; Kob+24]. Kobayashi et al. decomposes model components for BERT and GPT and study how the norm of each token changes from one layer to the next. We adopt the idea of decomposing a layer's output but do not consider the previous layer's output as the baseline for decomposition; rather, we consider the inputted time-series. A more detailed discussion of this topic can be found in section 4.1.2. The residual connections enable a coupling between a layer's output and its input, which allows us to avoid relying on norms and instead study the change of the whole token vectors in $\mathcal{X}$. The concrete contextualisation metrics, that we use to measure the changes in contextualisation throughout the model, will be introduced in section 4.1.1.

We will now introduce the relevant formal definitions and concepts that will be used throughout this chapter. We will place ourselves in the time-variate-token framework with token space $\mathcal{T} = \mathcal{X}^{\mathcal{I}}$, where we typically have $\mathcal{X} = \mathbb{R}^d$ with d being the model dimension. We consider a time-series $X \in \mathbb{R}^{N_{\text{in}} \times L_{\text{in}}}$ of length L_{in} with N_{in} variates as the input.

Definition 4.1. Let $M : \mathbb{R}^{N_{\text{in}} \times L_{\text{in}}} \to \mathbb{R}^{N_{\text{in}} \times T}$ be a FlexibleTransformer time-series forecasting model and let the *model steps* $M_{\text{in}} : \mathbb{R}^{N_{\text{in}} \times L_{\text{in}}} \to \mathcal{T}$, $M_k : \mathcal{T} \to \mathcal{T}$ for $k \in \{1, \dots, K\}$ and $M_{\text{out}} : \mathcal{T} \to \mathbb{R}^{N_{\text{in}} \times T}$ factorise the model in the sense of

$$M = M_{\text{out}} \circ M_K \circ \cdots \circ M_2 \circ M_1 \circ M_{\text{in}}.$$

Let $1 \leq k \leq K$. Then the *cumulative intermediate model step* corresponding to k is given by

$$\tilde{M}_k := M_k \circ \cdots \circ M_1 \circ M_{\text{in}}. \tag{4.1}$$

In practice, these model steps are simply the token-mixer layers, token-processor layers, normalisation layers, residual connections, possible embeddings and decoder layers that we use to build the FlexibleTransformer. A cumulative intermediate model step has outputs in the token-space $\mathcal{T} = \mathcal{X}^{\mathcal{I}}$.

The aim is to express each component of each token in $\tilde{M}_k(X)$, i.e. $(\tilde{M}_k(X)_i)_j$ for $i \in \mathcal{I}$ and $j \in \{1, \dots, d\}$, as a linear combination of elements of the inputted time-series X. This will be achieved in the following definition, where the relevant notion will be introduced.

Definition 4.2. Let $\tilde{M} : \mathbb{R}^{N_{\text{in}} \times L_{\text{in}}} \to \mathcal{T}$ be a cumulative intermediate model step. The

decomposition of $\tilde{M}$ given $X \in \mathbb{R}^{N_{\text{in}} \times L_{\text{in}}}$ is $(\boldsymbol{F}^{\tilde{M}}(X), \boldsymbol{b}^{\tilde{M}}(X))$, whereas

$$\boldsymbol{F}^{\tilde{M}}(X) := \left((F^{\tilde{M}}_{n,l}(X,i))_{(n,l)\in\{1,\dots,N_{\text{in}}\}\times\{1,\dots,L_{\text{in}}\}} \right)_{i\in\mathcal{I}} \in \left(\left(\mathbb{R}^d\right)^{N_{\text{in}}\times L_{\text{in}}} \right)^{\mathcal{I}}$$
$$\boldsymbol{b}^{\tilde{M}}(X) := \left(b^{\tilde{M}}(X,i) \right)_{i\in\mathcal{I}} \in \left(\mathbb{R}^d\right)^{\mathcal{I}}$$

such that

$$\tilde{M}(X)_i = \sum_{\substack{n\in\{1,\dots,N_{\text{in}}\} \\ l\in\{1,\dots,L_{\text{in}}\}}} F^{\tilde{M}}_{n,l}(X,i) + b^{\tilde{M}}(X,i) \in \mathbb{R}^d.$$

We can see that $\boldsymbol{F}^{\tilde{M}}(X)$ provides a decomposition of a cumulative intermediate model step $\tilde{M}$ in the sense that the cumulative intermediate model step's output can be directly related to the input sequence.

Remark 4.1. *We remark that the decomposition $(\boldsymbol{F}^{\tilde{M}}(X), \boldsymbol{b}^{\tilde{M}}(X))$ is not unique. In fact, we have to carefully make sure that we capture the idea that $F^{\tilde{M}}_{n,l}(X,i)$ is the contribution of the l-th timestamp of the n-th variate for input X at token i after the application of $\tilde{M}$. This idea is not captured in definition 4.2. This is only non-trivial in cases where $\tilde{M}$ is non-linear (i.e.* SelfAttn*,* TokenMLP*,* MLPTokenMixer *and* LayerNorm*).*

We have tried to find a definition that captures this idea. One possibility is the following: We have thought about introducing the concept of context *in the definition above and requiring*

$$F^{\tilde{M}}_{n,l}(X,i) = f_{i,n,l}(X_{n,l}, c_i(X)),$$

for $i \in \mathcal{I}$, $f_{i,n,l} : \mathbb{R} \times \mathcal{C} \to \mathbb{R}^d$. Here $c_i : \mathbb{R}^{N_{in}\times L_{in}} \to \mathcal{C}$ denotes context that is shared for all n, l. We can proof that the decomposition of SelfAttn*, that we provide below in lemma 4.2, and the decompositions of* LayerNorm *would satisfy this extended version of the definition above. For example, in the case of self-attention, the context would mainly capture the attention scores. However, we cannot relate the decomposition of* MLPTokenMixer *and* TokenMLP *to this definition because of the use of integrated gradients.*

4.1.1 Contextualisation Metrics

The contextualisation of two model components can now be compared using contextualisation metrics. While there is a contextualisation metrics in the norm-based approach proposed in [Kob+24], we propose an additional metric based on amplification that does not fit within the norm-based approach, but rather uses the unique structure of our decomposition.

Definition 4.3. Let $\boldsymbol{F}^{\tilde{M}}(X), \boldsymbol{b}^{\tilde{M}}(X)$ and $\boldsymbol{F}^{\tilde{\tilde{M}}}(X), \boldsymbol{b}^{\tilde{\tilde{M}}}(X)$ be two decompositions in the sense of definition 4.2. Let

$$g_{\text{CM}} : \mathbb{R}^{\mathcal{J}} \times \mathbb{R}^{\mathcal{J}} \to \mathbb{R},$$

where $\mathcal{J} := \mathcal{I} \times \{1,\dots,N_{\text{in}}\} \times \{1,\dots,L_{\text{in}}\} \times \{1,\dots,d\}$. Then the contextualisation metric

$\mathrm{CM}_{\tilde{M}\to\tilde{\tilde{M}}}$ based on g_{CM} is a map

$$\mathrm{CM}_{\tilde{M},\tilde{\tilde{M}}} : \mathbb{R}^{N_{\mathrm{in}}\times L_{\mathrm{in}}} \to \mathbb{R}$$
$$X \mapsto g_{\mathrm{CM}}\left(\left(F_{n,l}^{\tilde{M}}(X,i)_j\right)_{(i,n,l,j)\in\mathcal{J}}, \left(F_{n,l}^{\tilde{\tilde{M}}}(X,i)_j\right)_{(i,n,l,j)\in\mathcal{J}}\right).$$

The goal of the contextualisation metric is to capture the changes of relevances of tokens before and after the application of one or several model layers, i.e. where $\tilde{\tilde{M}} = S \circ \tilde{M}$ and $S : \mathcal{T} \to \mathcal{T}$. We typically choose S to be either one specific layer, i.e.

$$S = M_k, \qquad \tilde{M} = M_{k-1} \circ \cdots \circ M_1 \circ M_{\mathrm{in}}$$

for $1 \leq k \leq K$ or can alternatively also capture cumulative contextualisation behaviour by choosing

$$S = M_k \circ \cdots \circ M_1, \qquad \tilde{M} = M_{\mathrm{in}}.$$

Let us begin by presenting a contextualisation metric from the literature that is part of the norm-based approach.

Spearman Contextualisation Metric in the Norm-based Approach

In the norm-based approach, we consider the norm $\|F_{n,l}(X,i)\|$ instead of the individual components of $F_{n,l}(X,i) \in \mathcal{X}$. This simplifies the comparison of $\boldsymbol{F}^{\tilde{M}}(X)$ and $\boldsymbol{F}^{\tilde{\tilde{M}}}(X)$.

In [Kob+24], the authors introduce a contextualisation metric based on the Spearman rank correlation coefficient.

Definition 4.4. The *Spearman mixing-metric* is a contextualisation metric that is based on g_{Spearman}. Let $\boldsymbol{F}^{\tilde{M}}(X), \boldsymbol{F}^{\tilde{\tilde{M}}}(X)$ and $\mathcal{J}$ be as in definition 4.3. Then

$$g_{\mathrm{Spearman}}\left(\left(F_{n,l}^{\tilde{M}}(X,i)_j\right)_{(i,n,l,j)\in\mathcal{J}}, \left(F_{n,l}^{\tilde{\tilde{M}}}(X,i)_j\right)_{(i,n,l,j)\in\mathcal{J}}\right)$$
$$:= 1 - \rho_S\left(\left(\|F_{n,l}^{\tilde{M}}(X,i)\|_2\right)_{i,n,l}, \left(\|F_{n,l}^{\tilde{\tilde{M}}}(X,i)\|_2\right)_{i,n,l}\right).$$

We remind the reader of the definition of the Spearman rank correlation coefficient in appendix A.4.

The Spearman mixing metric is an interesting approach to mixing tokens in that it treats mixing as a permutation of their relative importance (in the sense of their norm). However, this can lead to issues in practice if the amplitude of the tokens varies greatly and there are only a few important tokens having large norms. The non-dominant tokens (with small norms) do not significantly influence the model's prediction, but may greatly alter the Spearman mixing metric.

Thus, we want to introduce another contextualisation metric that does not compare norms on the token-level.

Amplification Mixing-Metric

A significant challenge when comparing two decompositions $\boldsymbol{F}^{\tilde{M}}(X)$ and $\boldsymbol{F}^{\tilde{\tilde{M}}}(X)$ is that the scales of these two decompositions are not necessarily aligned. If we are able to compare them on the same scale, we can simply consider the difference between them and then average out the absolute values of the differences. This leads us to consider the amplification mixing-metric.

Definition 4.5. The *amplification mixing-metric* is a contextualisation metric that is based on g_{amp}. Let $\boldsymbol{F}^{\tilde{M}}(X), \boldsymbol{F}^{\tilde{\tilde{M}}}(X)$ and $\mathcal{J}$ be as in definition 4.3. Then

$$\begin{aligned} &g_{\text{amp}}\left(\left(F^{\tilde{M}}_{n,l}(X,i)_j\right)_{(i,n,l,j)\in\mathcal{J}}, \left(F^{\tilde{\tilde{M}}}_{n,l}(X,i)_j\right)_{(i,n,l,j)\in\mathcal{J}}\right) \\ &:= \frac{1}{|\mathcal{I}|N_{\text{in}}L_{\text{in}}d}\sum_{i\in\mathcal{I}}\left\|\left(\frac{F^{\tilde{M}}_{n,l}(X,i)_j - \mu\left(\boldsymbol{F}^{\tilde{M}}\right)}{\sigma\left(\boldsymbol{F}^{\tilde{M}}(X)\right)} - \frac{F^{\tilde{\tilde{M}}}_{n,l}(X,i)_j - \mu\left(\boldsymbol{F}^{\tilde{\tilde{M}}}\right)}{\sigma\left(\boldsymbol{F}^{\tilde{\tilde{M}}}(X)\right)}\right)_{n,l,j}\right\|_1 \end{aligned}$$

where

$$\begin{aligned} \mu(\boldsymbol{F}(X)) &:= \frac{1}{|\mathcal{I}|N_{\text{in}}L_{\text{in}}d}\sum_{(i',n',l',j')\in\mathcal{J}} F_{n',l'}(X,i')_{j'} \\ \sigma(\boldsymbol{F}(X)) &= \left(\frac{1}{|\mathcal{I}|N_{\text{in}}L_{\text{in}}d}\sum_{(i',n',l',j')\in\mathcal{J}}\left(F_{n',l'}(X,i')_{j'} - \mu(\boldsymbol{F}(X))\right)^2\right)^{\frac{1}{2}} \end{aligned}$$

We will see examples of these contextualisation metrics once we have decomposed the FlexibleTransformer.

4.1.2 Advantages of Decomposing over Attention Scores

These techniques were originally developed for the purpose of analysing Transformer language models with the objective of a deeper insight into the self-attention mixing operation. A naive approach would be to consider simply attention scores as the key object in a decomposition approach, i.e. the matrix

$$\left(\text{SoftMax}\left(\phi\left(\boldsymbol{q}_i^{(h)}, \boldsymbol{k}_{j'}^{(h)}\right)_{j'\in\mathcal{I}}\right)_j\right)_{(i,j)\in\mathcal{I}\times\mathcal{I}}$$

in equation (2.4). However, as [Kob+21] argues, this overlooks the crucial point that the queries $(\boldsymbol{q}_i^{(h)})_{i\in\mathcal{I}}$ and keys $(\boldsymbol{k}_j^{(h)})_{j\in\mathcal{I}}$ have been projected into $\mathbb{R}^{d_{\text{attn}}}$ using linear layers. These linear layers can already recombine, amplify and dampen certain tokens. According to Kobayashi et al., the analysis of attention scores is meaningless if isolated. The decomposition approach takes all these surrounding operations into account.

However, we believe that the approach proposed by Kobayashi et al. does not fully address the issue. While the model is composed of each component in a linear manner, with each component decomposed into its input, the analysis does not combine the results of multiple decompositions of successive components.

4.2 Decomposition of the FlexibleTransformer

The key ingredient for the norm-based approach introduced in the previous section is the decomposition $(\boldsymbol{F}^{\tilde{M}}(X), \boldsymbol{b}^{\tilde{M}}(X))$ of $\tilde{M}$ given $X \in \mathbb{R}^{N_{\text{in}} \times L_{\text{in}}}$. Hence, we have to decompose

- the different input embeddings,
- layer normalisation and residual connection operations,
- the different token-mixers (MLP token-mixer and self-attention) and
- the TokenMLP token-processor.

In order to relate the forecast to the input time-series in the same fashion, it is also necessary to decompose the linear decoder.

The proposed strategy is as follows: Assume that $\tilde{\tilde{M}} = S \circ \tilde{M}$, where we have already decomposed $\tilde{M}$. Hence, we compute $(\boldsymbol{F}^{\tilde{\tilde{M}}}(X), \boldsymbol{b}^{\tilde{\tilde{M}}}(X))$ based on $(\boldsymbol{F}^{\tilde{M}}(X), \boldsymbol{b}^{\tilde{M}}(X))$.

We move the decomposition of the residual connection and layer normalisation to the appendix, as their decomposition is not central to our exposition. As an example, in order to provide the reader with a basic understanding of the definitions, we will nevertheless carry out the decomposition of the single token input embedding of the vanilla Transformer in the main text.

For the sake of brevity, we implicitly assume that $(n, l) \in \{1, \ldots, N_{\text{in}}\} \times \{1, \ldots, L_{\text{in}}\}$ and $(m, k) \in \{1, \ldots, N\} \times \{1, \ldots, L\}$. We further make use of the Dirac delta $\delta_{x,y}$, that equals one if x and y agree and zero else.

Lemma 4.1. The decomposition of the single token input embedding is given by

$$\begin{aligned} F_{n,l}^{\text{STIEmb}}(X, (1, k)) &= W_{:,n} X_{n,k} \delta_{k,l} \\ b^{\text{STIEmb}}(X, (1, k)) &= b \end{aligned}$$

where $W \in \mathbb{R}^{d \times N_{\text{in}}}$, $b \in \mathbb{R}^d$ denote the weight matrix and the bias of the linear layer of the single token input embedding.

Proof. We want to write

$$\begin{aligned} \text{SingleTokenInputEmb}(X)_{(1,k)} &= \mathbf{Lin}_{N_{\text{in}} \to d}\left(X_{\underline{N_{\text{in}}} \times L}\right)_k \\ &= \text{Lin}_{N_{\text{in}} \to d}(X_{:,k}) \\ &= \sum_{\substack{n \in \{1,\ldots,N_{\text{in}}\} \\ l \in \{1,\ldots,L_{\text{in}}\}}} F_{n,l}^{\text{STIB}}(X, (1, k)) + b^{\text{STIB}}(X, (1, k)). \end{aligned}$$

With the definition of the linear layer, we get

$$\begin{aligned}\mathrm{Lin}_{N_{\mathrm{in}}\to d}\left(X_{:,k}\right) &= WX_{:,k} + b \\ &= \sum_{n\in\{1,\ldots,N_{\mathrm{in}}\}} W_{:,n}X_{n,k} + b \\ &= \sum_{\substack{n\in\{1,\ldots,N_{\mathrm{in}}\}\\ l\in\{1,\ldots,L_{\mathrm{in}}\}}} W_{:,n}X_{n,l}\delta_{k,l} + b.\end{aligned}$$

This concludes the proof. □

In the next proofs, we will be somewhat briefer and will focus on the decomposition of the key steps. In the remainder of this section, we will decompose the major components of the FlexibleTransformer that are of particular interest to us. We will begin with the decomposition of self-attention, which is relatively straightforward. Then, we will turn to the MLP-based components.

The non-linearity of their activation function necessitates the utilisation of a specialised technique, namely *integrated gradients*, which was originally introduced in [STY17]. Having introduced and justified this technique, it is then employed in the decomposition of the MLP token-mixer and MLP token-processor.

4.2.1 Decomposition of Self-Attention

This section builds upon the ideas presented in [Kob+21] by extending them to the time-variate-token paradigm.

While our formalism allows for the straightforward application of self-attention along both the time- and variate-axes in the time-variate-token paradigm, we limit our discussion to the case where self-attention is applied along the time-axis, i.e. $\mathrm{SelfAttn}_{\mathrm{time}}$. The decomposition for self-attention along the variate-axis is analogous to the decomposition for self-attention along the time-axis. However, the components of $(m,k) = i \in \mathcal{I}$ and $(n,l) \in \{1,\ldots,N_{\mathrm{in}}\} \times \{1,\ldots,L_{\mathrm{in}}\}$ must be interchanged, respectively. We have the following result.

Lemma 4.2. Let $\tilde{M} : \mathbb{R}^{N_{\mathrm{in}}\times L_{\mathrm{in}}} \to \mathcal{T}$ be the cumulative intermediate model step before the application of self-attention with H heads. Let $(\boldsymbol{F}^{\tilde{M}}(X), \boldsymbol{b}^{\tilde{M}}(X))$ be the decomposition up until right before the self-attention component. The decomposition of $\tilde{M}$ together with self-attention is

$$F^{\mathrm{SA}_{\mathrm{time}}\circ\tilde{M}}_{n,l}(X,(m,k)) = \sum_{h=1}^{H}\sum_{k'=1}^{L} W^{(\mathrm{out},h)}A^{(h,m)}_{k,k'}W^{(\mathcal{V},h)}F^{\tilde{M}}_{n,l}(X,(m,k')),$$

$$b^{\mathrm{SA}_{\mathrm{time}}\circ\tilde{M}}(X,(m,k)) = \sum_{h=1}^{H}\left(\sum_{k'=1}^{L}\left(W^{(\mathrm{out},h)}A^{(h,m)}_{k,k'}\left(W^{(\mathcal{V},h)}b^{\tilde{M}}(X,(m,k)) + b^{(\mathcal{V},h)}\right)\right) + b^{(\mathrm{out},h)}\right)$$

where $W^{(\mathcal{V},h)}, b^{(\mathcal{V},h)}$ and $W^{(\mathrm{out},h)}, b^{(\mathrm{out},h)}$ denote the weights and biases of $\mathrm{Lin}^{(\mathcal{V},h)}_{d\to d_{\mathrm{attn}}}$ and $\mathrm{Lin}^{(\mathrm{out},h)}_{d_{\mathrm{attn}}\to d}$ for $h = 1,\ldots,H$ and

$$A^{(h,m)}_{k,k'} := \mathrm{SoftMax}\left(\phi\left(\mathrm{Lin}^{(\mathcal{Q},h)}_{d\to d_{\mathrm{attn}}}\left(\tilde{M}(X)_{m,k}\right), \mathrm{Lin}^{(\mathcal{K},h)}_{d\to d_{\mathrm{attn}}}\left(\tilde{M}(X)_{m,p}\right)\right)_{p\in\{1,\ldots,L\}}\right)_{k'} \in \mathbb{R}.$$

Proof. The proof consists merely of cleverly rewriting $(\mathrm{SelfAttn}_{\mathrm{time}} \circ \tilde{M})(X)_{m,k}$ from definition 2.5. In each $\mathrm{SelfAttn}_{\mathrm{time}}$ operation, we compute an attention score matrix for each token (m, k). Based on definitions 2.5 and 3.2, we identify the (k, k')-th component of the attention score matrix as $A^{(h,m)} \in \mathbb{R}^{L \times L}$.

This allows us to write using definitions 2.5 and 3.2 and the decomposition of $\tilde{M}$ given X

$$\tilde{M}(X)_{(m,k)} = \sum_{\substack{n \in \{1,\dots,N_{\mathrm{in}}\} \\ l \in \{1,\dots,L_{\mathrm{in}}\}}} F^{\tilde{M}}_{n,l}(X, (m,k)) + b^{\tilde{M}}(X, (m,k))$$

that:

$$\begin{aligned}
&\mathrm{SelfAttn}_{\mathrm{time}}(\tilde{M}(X))_{(m,k)} \\
&= \sum_{h=1}^{H} \mathrm{Lin}^{(\mathrm{out},h)}_{d_{\mathrm{attn}} \to d} \left(\sum_{k'=1}^{L} A^{(h,m)}_{k,k'} \mathrm{Lin}^{(\mathcal{V},h)}_{d \to d_{\mathrm{attn}}} (\tilde{M}(X)_{m,k'}) \right) \\
&= \sum_{h=1}^{H} \left[W^{(\mathrm{out},h)} \left(\sum_{k'=1}^{L} A^{(h,m)}_{k,k'} \left(W^{(\mathcal{V},h)} \tilde{M}(X)_{m,k'} + b^{(\mathcal{V},h)} \right) \right) + b^{(\mathrm{out},h)} \right] \\
&= \sum_{h=1}^{H} \left[W^{(\mathrm{out},h)} \left(\sum_{k'=1}^{L} A^{(h,m)}_{k,k'} \left(W^{(\mathcal{V},h)} \left(\sum_{\substack{n \in \{1,\dots,N_{\mathrm{in}}\} \\ l \in \{1,\dots,L_{\mathrm{in}}\}}} F^{\tilde{M}}_{n,l}(X, (m,k')) + b^{\tilde{M}}(X, (m,k')) \right) \right. \right. \right. \\
&\qquad \left. \left. \left. + b^{(\mathcal{V},h)} \right) \right) + b^{(\mathrm{out},h)} \right] \\
&= \sum_{\substack{n \in \{1,\dots,N_{\mathrm{in}}\} \\ l \in \{1,\dots,L_{\mathrm{in}}\}}} \sum_{h=1}^{H} \sum_{k'=1}^{L} W^{(\mathrm{out},h)} A^{(h,m)}_{k,k'} W^{(\mathcal{V},h)} F^{\tilde{M}}_{n,l}(X, (m,k')) \\
&\qquad + \sum_{h=1}^{H} \left(\sum_{k'=1}^{L} \left(W^{(\mathrm{out},h)} A^{(h,m)}_{k,k'} \left(W^{(\mathcal{V},h)} b^{\tilde{M}}(X, (m,k')) + b^{(\mathcal{V},h)} \right) \right) + b^{(\mathrm{out},h)} \right)
\end{aligned}$$

We can identify the postulated expression for $F^{\mathrm{SA}_{\mathrm{time}} \circ \tilde{M}}_{n,l}(X, (m,k))$. □

4.2.2 Integrated Gradients for Feature Attribution

Before proceeding to the decomposition of the MLP token-mixer and the MLP token-processor, it is necessary to motivate and introduce a technique for the decomposition of the non-linear activation function g in the MLP.

The decomposition of self-attention is relatively straightforward due to the linear structure of recombining the tokens through the value projection with the attention scores as weights. However, with the non-linear activation in the MLP, we are now in a different position because a simple linear decomposition is not possible anymore. Consequently, we pursue the approach proposed in [Kob+24], which involves utilising *integrated gradients (IG)* to decompose this non-linear activation. Kobayashi et al. were the first to apply IG to the MLP structure in language transformer models.

The method of integrated gradients is a valuable contribution to the diverse range of feature attribution approaches. Previous works on this topic have included the use of the product of the model's gradients and the features [Bae+09], deconvolutional networks [ZF14], and guided backpropagation in convolutional neural networks [Spr+14]. Furthermore, the layer-wise relevancy propagation approach proposed in [Bin+16] and the Deep Lift method developed in [SGK17] should be mentioned. The IG approach taken by Sundararajan, Taly, and Yan in [STY17] is axiomatic in nature. In this approach, axioms are postulated for feature attribution methods. *Path methods* are identified as the only feature decomposition methods that satisfy these axioms. IG represents a specific instance of a path method, and arguably represents the simplest such instance.

In this section, we aim to provide a concise overview of the axiomatic approach described above, as it plays a pivotal role in our ability to decompose the FlexibleTransformer. We will present the axioms postulated in [STY17] and introduce path methods. Finally, we will conclude by introducing the IG method.

Axioms for Feature Attribution Methods

The following definition of a feature attribution method is a slight modification of definition 1 in [STY17].

Definition 4.6. Let $g : \mathbb{R}^k \to \mathbb{R}$ represent a neural network and let $x \in \mathbb{R}^k$ be an input to g. An *attribution* of the prediction of g at input x relative to a baseline input $x' \in \mathbb{R}^k$ is a vector $A_g(x, x') \in \mathbb{R}^k$ where $A_g(x, x')_i$ is the contribution of x_i to the prediction $g(x)$.

The selection of the baseline input x' can be thought of as a zero prediction. In image processing tasks, this could be the black image. Another basic strategy is to choose x' such that $g(x') = 0$, which is consistent with the completeness axiom that we present shortly.

Sundararajan, Taly, and Yan postulate the following four axioms for an attribution A_g for g:

1. **Sensitivity**: An attribution A_g for g is said to satisfy the sensitivity axiom if for all g which do not depend (mathematically) on the i_0-th component of its input x for $i_0 \in \{1, \dots, k\}$, then $A_g(x, x')_{i_0} = 0$ holds for all $x, x' \in \mathbb{R}^k$.

2. **Linearity**: An attribution A is said to satisfy the linearity axiom if the map

$$\begin{aligned} A : \mathcal{F}(\mathbb{R}^k, \mathbb{R}) &\to \mathcal{F}(\mathbb{R}^k \times \mathbb{R}^k, \mathbb{R}) \\ g &\mapsto A_g \end{aligned}$$

   is linear. Here, $\mathcal{F}(\mathbb{R}^k, \mathbb{R})$ and $\mathcal{F}(\mathbb{R}^k \times \mathbb{R}^K, \mathbb{R})$ represent the set of functions from $\mathbb{R}^k$ and $\mathbb{R}^k \times \mathbb{R}^k$ to $\mathbb{R}$.

3. **Completeness**: An attribution A_g for a model g is said to satisfy the completeness axiom if

$$g(x) - g(x') = \sum_{i=1}^{k} A_g(x, x')_i.$$

4. **Implementation invariance**: Two networks g and g' are said to be functionally

equivalent if they agree as functions from $g, g' : \mathbb{R}^k \to \mathbb{R}$ despite having different implementations. We require that an attribution is invariant with respect to the implementation, i.e. for two functionally equivalent networks g, g', we require $A_g = A_{g'}$.

For instance, it can be demonstrated that the gradients $\nabla_x g$ are implementation invariant. However, popular methods such as DeepLift and layer-wise relevancy propagation do not adhere to this implementation invariance, as they utilise discrete gradients for which the chain rule does not hold.

Concrete counterexamples for the stated violations of these axioms for popular methods can be found in appendix B of [STY17].

Path Methods

We can formulate a class of attributions that we call *path methods*. It has been demonstrated that path methods are the only attribution methods that satisfy all the axioms above. We are however not going to prove the latter result and refer the interested reader to [Fri04, Theorem 1] for a proof and a formal definition of the axiomns.

Definition 4.7. Let $\gamma_{x,x'} \in C^\infty([0,1], \mathbb{R}^k)$ be a smooth path from $\gamma_{x,x'}(0) = x'$ to $\gamma_{x,x'}(1) = x$. Then, the *path integrated gradient* (PIG) for a function $g \in C^1(\mathbb{R}^k, \mathbb{R})$ is given by

$$\mathrm{PIG}^{\gamma_{x,x'},g} := \int_0^1 \nabla g(\gamma_{x,x'}(t))\, \gamma'_{x,x'}(t)\, dt$$

We can observe that path integrated gradients satisfy inplementation invariance because they are defined only using the gradients of the function g. It is immediate from the fundamental theorem of calculus that we further have

$$g(x) - g(x') = \sum_{i=1}^{k} \mathrm{PIG}_i^{\gamma_{x,x'},g}$$

since

$$\sum_{i=1}^{k} \mathrm{PIG}_i^{\gamma_{x,x'},g} = \int_0^1 \sum_{i=1}^{k} \frac{\partial}{\partial x_i} g(\gamma_{x,x'}(t))\, (\gamma_{x,x'})'_i(t)\, dt = \int_0^1 \frac{\partial}{\partial t}(g \circ \gamma_{x,x'})(t)\, dt = g(x) - g(x').$$

This also proves that path methods satisfy the completeness axiom. We further remark that completeness is a very desirable property and is ideally suited for our goal of decomposing the MLP-based components of the FlexibleTransformer. The linearity and sensitivity axiom follow from the linearity of the gradient and the fact that $\partial/\partial x_{i_0} g(x_{i_0}) = 0$ if g does not depend on the i_0-th component.

The path integrated gradients introduced above have been known and used in the cost-sharing literature in economics and also go under the name Aumann-Shapley method [AS74].

Integrated Gradients as the Unique Symmetry-Preserving Path Integrated Gradient Method

The consideration of another desirable property of the attribution method, that we call *symmetry-preservation*, leads us to integrated gradients. We say that an attribution method

A_g for g is symmetry-preserving if for all pairs $(i,j) \in \{1,\dots,k\}^2$, $i < j$ such that

$$g(x) = g(x_0,\dots,x_{i-1},x_j,x_{i+1},\dots,x_{j-1},x_i,x_{j+1},\dots,x_k) \qquad \text{for all } x \in \mathbb{R}^k$$

we have

$$A_g(x,x')_i = A_g(x,x')_j \qquad \text{for all } x,x' \in \mathbb{R}^k,$$

where $x_i = x_j$ and $x'_i = x'_j$. Let us now define IG.

Definition 4.8. Integrated gradients is an integrated path gradient method with the path $\tilde{\gamma}_{x,x'}(t) := (1-t)x' + tx$. We write $\mathrm{IG}^g(x,x') := \mathrm{PIG}^{\tilde{\gamma}_{x,x'},g}$.

IG is not only appealing because of its simplicity thanks to the straight line path, but also thanks to the following proposition that is proven in [STY17, Appendix A].

Proposition 4.1. The *integrated gradients* method is the unique integrated path method that satisfies symmetry-preservation.

4.2.3 Decomposition of the MLP Token-Mixer

Recall that the MLP token-mixer was defined in definition 3.5

$$\mathrm{MLPTokenMixer}\left(\hat{X}\right) := \left[\mathrm{Reshape}_{(A,d)}\left(\mathrm{MLP}_{dA\to d_{\mathrm{mixing}}\to dA,g}\left(\mathrm{Flatten}_{\mathcal{J}}\left(\hat{X}\right)\right)\right)\right]_{A\times \underline{d}},$$

where $\hat{X} \in \mathcal{X}^{\mathcal{J}}$ with $\mathcal{J} = \{1,\dots,A\}$. The concrete choice of A depends on whether the MLP token-mixer is applied along the time- or the variate-axis. As with the decomposition of self-attention, we only consider the case where the MLP token-mixer is applied along the time-axis, i.e. we decompose $\mathrm{MLPTokenMixer}_{\mathrm{time}}$. We continue to use the naming conventions from the previous section on decomposing self-attention.

We decompose the j-th component of the token at position (m,k) with the previous application of a cumulative intermediate model step $\tilde{M} : \mathbb{R}^{N_{\mathrm{in}}\times L_{\mathrm{in}}} \to \mathcal{T}$. As before, we write $\tilde{X} = \tilde{M}(X)$ with model input $X \in \mathbb{R}^{N_{\mathrm{in}}\times L_{\mathrm{in}}}$. Using the definition of MLPTokenMixer, we can write

$$\begin{aligned}
&\left(\mathrm{MLPTokenMixer}_{\mathrm{time}}\left(\tilde{M}(X)\right)_{(m,k)}\right)_j \\
=&\mathrm{MLP}_{dL\to d_{\mathrm{mixing}}\to dL,g}\Big(\Big(\tilde{M}(X)_{(m,1)}\Big)_1 \cdots \Big(\tilde{M}(X)_{(m,1)}\Big)_d \\
&\qquad \cdots \quad \cdots \quad \cdots \quad \Big(\tilde{M}(X)_{(m,L)}\Big)_1 \cdots \Big(\tilde{M}(X)_{(m,L)}\Big)_d\Big)_{kd+j}
\end{aligned}$$

The MLP has one hidden layer and hence is composed of linear input and linear output projections with weights $W^{(0)} \in \mathbb{R}^{d_{\mathrm{mixing}}\times dL}$, $b^{(0)} \in \mathbb{R}^{d_{\mathrm{mixing}}}$ and $W^{(1)} \in \mathbb{R}^{dL\times d_{\mathrm{mixing}}}, b^{(1)} \in \mathbb{R}^{dL}$ respectively to and from $\mathbb{R}^{d_{\mathrm{mixing}}}$ with a non-linear activation g in between. We can write $\iota : \hat{X} \mapsto W^{(0)}\mathrm{Flatten}_{\mathcal{J}}(\hat{X}) + b^{(0)}$ for $\hat{X} \in \mathcal{X}^{\mathcal{J}}$ and

$$\begin{aligned}
F^{\mathrm{Pre}\,g}_{n,l}(X,m) := W^{(0)}\Big(&F^{\tilde{M}}_{n,l}(X,(m,1))_1 \cdots F^{\tilde{M}}_{n,l}(X,(m,1))_d \\
&\cdots \quad \cdots \quad \cdots \quad F^{\tilde{M}}_{n,l}(X,(m,L))_1 \cdots F^{\tilde{M}}_{n,l}(X,(m,L))_d\Big) \\
b^{\mathrm{Pre}\,g}(X,m) := W^{(0)}\Big(&b^{\tilde{M}}(X,(m,1))_1 \quad \cdots \quad b^{\tilde{M}}(X,(m,1))_d \\
&\cdots \quad \cdots \quad \cdots \quad b^{\tilde{M}}(X,(m,L))_1 \quad \cdots \quad b^{\tilde{M}}(X,(m,L))_d\Big) + b^{(0)}
\end{aligned}$$

so that

$$\iota\left(\left(\tilde{M}(X)_{N\times\underline{L}}\right)_m\right) = \sum_{\substack{n\in\{1,\dots,N_{\text{in}}\}\\ l\in\{1,\dots,L_{\text{in}}\}}} F_{n,l}^{\text{Pre}\,g}(X,m) + b^{\text{Pre}\,g}(X,m) \in \mathbb{R}^{d_{\text{hidden}}}$$

We now face the non-linear activation function $g : \mathbb{R} \to \mathbb{R}$. As we however do not want to consider the sum $\sum_{n,l} F_{n,l}^{\text{Pre}\,g}(X,m) + b^{\text{Pre}\,g}(X,m)$ as the sole information that we input to g, but rather the individual contributions $(F_{n,l}^{\text{Pre}\,g}(X,m))_{n,l}$ and $b^{\text{Pre}\,g}(X,m)$, we define

$$\begin{aligned}\tilde{g} : \mathbb{R}^{N_{\text{in}}L_{\text{in}}+1} &\to \mathbb{R}\\ x &\mapsto g\left(\sum_{\substack{n\in\{1,\dots,N_{\text{in}}\}\\ l\in\{1,\dots,L_{\text{in}}\}}} x_{(n-1)L_{\text{in}}+l} + x_{N_{\text{in}}L_{\text{in}}+1}\right).\end{aligned}$$

Applying the feature attribution method IG to $\tilde{g}$ with baseline $\mathbf{0}$, we get for the p-th component of the activated hidden representation

$$\begin{aligned}&g\left(\sum_{\substack{n\in\{1,\dots,N_{\text{in}}\}\\ l\in\{1,\dots,L_{\text{in}}\}}} F_{n,l}^{\text{Pre}\,g}(X,m)_p + b^{\text{Pre}\,g}(X,m)_p\right)\\ &= \sum_{\substack{n\in\{1,\dots,N_{\text{in}}\}\\ l\in\{1,\dots,L_{\text{in}}\}}} \text{IG}^{\tilde{g}}_{(n-1)L_{\text{in}}+l}\Big(\Big(F_{1,1}^{\text{Pre}\,g}(X,m)_p \quad \cdots \quad F_{1,L_{\text{in}}}^{\text{Pre}\,g}(X,m)_p\\ &\qquad\qquad \cdots \quad F_{N_{\text{in}},1}^{\text{Pre}\,g}(X,m)_p \quad \cdots \quad F_{N_{\text{in}},L_{\text{in}}}^{\text{Pre}\,g}(X,m)_p \quad b^{\text{Pre}\,g}(X,m)_p\Big), \mathbf{0}\Big)\\ &\quad + \text{IG}^{\tilde{g}}_{N_{\text{in}}L_{\text{in}}+1}\Big(\Big(F_{1,1}^{\text{Pre}\,g}(X,m)_p \quad \cdots \quad F_{1,L_{\text{in}}}^{\text{Pre}\,g}(X,m)_p\\ &\qquad\qquad \cdots \quad F_{N_{\text{in}},1}^{\text{Pre}\,g}(X,m)_p \quad \cdots \quad F_{N_{\text{in}},L_{\text{in}}}^{\text{Pre}\,g}(X,m)_p \quad b^{\text{Pre}\,g}(X,m)_p\Big), \mathbf{0}\Big).\end{aligned}$$

Together with the output projection, we have proven the following lemma.

Lemma 4.3. Let $\tilde{M} : \mathbb{R}^{N_{\text{in}}\times L_{\text{in}}} \to \mathcal{T}$ be the model part before the application of the MLP token-mixer. Let $(\boldsymbol{F}^{\tilde{M}}(X), \boldsymbol{b}^{\tilde{M}}(X))$ be the decomposition up until right before the MLP token-mixer component with hidden dimension d_{mixing} and activation function g. The decomposition of $\text{MLPTokenMixer}_{\text{time}} \circ \tilde{M}$ is given by

$$\begin{aligned}&F_{n,l}^{\text{MLPTokenMixer}_{\text{time}}\circ\tilde{M}}(X,(m,k))_j\\ &= \sum_{p=1}^{d_{\text{mixing}}} W^{(1)}_{kd+j,p}\text{IG}^{\tilde{g}}_{(n-1)L_{\text{in}}+l}\Big(\Big(F_{1,1}^{\text{Pre}\,g}(X,m)_p \quad \cdots \quad F_{1,L_{\text{in}}}^{\text{Pre}\,g}(X,m)_p\\ &\qquad\qquad \cdots \quad F_{N_{\text{in}},1}^{\text{Pre}\,g}(X,m)_p \quad \cdots \quad F_{N_{\text{in}},L_{\text{in}}}^{\text{Pre}\,g}(X,m)_p \quad b^{\text{Pre}\,g}(X,m)_p\Big), \mathbf{0}\Big),\end{aligned}$$

and

$$
\begin{aligned}
&b^{\mathrm{MLPTokenMixer}_{\mathrm{time}}\circ\tilde{M}}(X,(m,k))_j \\
&= \sum_{p=1}^{d_{\mathrm{mixing}}} W^{(1)}_{kd+j,p}\mathrm{IG}^{\tilde{g}}_{N_{\mathrm{in}}L_{\mathrm{in}}+1}\Big(\Big(F^{\mathrm{Pre}\,g}_{1,1}(X,m)_p \quad \cdots \quad F^{\mathrm{Pre}\,g}_{1,L_{\mathrm{in}}}(X,m)_p \\
&\qquad\qquad \cdots \quad F^{\mathrm{Pre}\,g}_{N_{\mathrm{in}},1}(X,m)_p \quad \cdots \quad F^{\mathrm{Pre}\,g}_{N_{\mathrm{in}},L_{\mathrm{in}}}(X,m)_p \quad b^{\mathrm{Pre}\,g}(X,m)_p\Big),\mathbf{0}\Big) + b^{(1)}_j,
\end{aligned}
$$

where $j \in \{1,\dots,d\}$ and

$$
\begin{aligned}
F^{\mathrm{Pre}\,g}_{n,l}(X,m) &:= W^{(0)}\Big(F^{\tilde{M}}_{n,l}(X,(m,1))_1\cdots F^{\tilde{M}}_{n,l}(X,(m,1))_d \\
&\qquad\qquad \cdots \quad \cdots \quad \cdots \quad F^{\tilde{M}}_{n,l}(X,(m,L))_1\cdots F^{\tilde{M}}_{n,l}(X,(m,L))_d\Big) \\
b^{\mathrm{Pre}\,g}(X,m) &:= W^{(0)}\Big(b^{\tilde{M}}(X,(m,1))_1\cdots b^{\tilde{M}}(X,(m,1))_d \\
&\qquad\qquad \cdots \quad \cdots \quad \cdots \quad b^{\tilde{M}}(X,(m,L))_1\cdots b^{\tilde{M}}(X,(m,L))_d\Big) + b^{(0)}.
\end{aligned}
$$

4.2.4 Decomposition of the MLP Token-Processor

The decomposition of the MLP token-processor is very similar to the decomposition of the MLP token-mixer, because again we have an MLP as the key part to be decomposed with IG. We will not give the proof of the following lemma, because it is essentially the same as the proof of lemma 4.3.

Lemma 4.4. Let $\tilde{M} : \mathbb{R}^{N_{\mathrm{in}}\times L_{\mathrm{in}}} \to \mathcal{T}$ be the model part before the application of the MLP token-processor. Let $(\boldsymbol{F}^{\tilde{M}}(X), \boldsymbol{b}^{\tilde{M}}(X))$ be the decomposition up until right before the MLP token-processor component with hidden dimension d_{hidden} and activation function g. The decomposition of $\tilde{M}$ together with the MLP token-processor is given by

$$
\begin{aligned}
&F^{\mathrm{TokenMLP}\circ\tilde{M}}_{n,l}(X,i)_j \\
&= \sum_{p=1}^{d_{\mathrm{hidden}}} W^{(1)}_{j,p}\mathrm{IG}^{\tilde{g}}_{(n-1)L_{\mathrm{in}}+l}\Big(\Big(F^{\mathrm{Pre}\,g}_{1,1}(X,i)_p \quad \cdots \quad F^{\mathrm{Pre}\,g}_{1,L_{\mathrm{in}}}(X,i)_p \\
&\qquad\qquad \cdots \quad F^{\mathrm{Pre}\,g}_{N_{\mathrm{in}},1}(X,i)_p \quad \cdots \quad F^{\mathrm{Pre}\,g}_{N_{\mathrm{in}},L_{\mathrm{in}}}(X,i)_p \quad b^{\mathrm{Pre}\,g}(X,i)_p\Big),\mathbf{0}\Big)
\end{aligned}
$$

and

$$
\begin{aligned}
&b^{\mathrm{TokenMLP}_{\mathrm{time}}\circ\tilde{M}}(X,i)_j \\
&= \sum_{p=1}^{d_{\mathrm{hidden}}} W^{(1)}_{j,p}\mathrm{IG}^{\tilde{g}}_{N_{\mathrm{in}}L_{\mathrm{in}}+1}\Big(\Big(F^{\mathrm{Pre}\,g}_{1,1}(X,i)_p \quad \cdots \quad F^{\mathrm{Pre}\,g}_{1,L_{\mathrm{in}}}(X,i)_p \\
&\qquad\qquad \cdots \quad F^{\mathrm{Pre}\,g}_{N_{\mathrm{in}},1}(X,i)_p \quad \cdots \quad F^{\mathrm{Pre}\,g}_{N_{\mathrm{in}},L_{\mathrm{in}}}(X,i)_p \quad b^{\mathrm{Pre}\,g}(X,i)_p\Big),\mathbf{0}\Big) + b^{(1)}_j,
\end{aligned}
$$

where $j \in \{1,\dots,d\}$ and

$$
\begin{aligned}
F^{\mathrm{Pre}\,g}_{n,l}(X,i) &:= W^{(0)}\Big(F^{\tilde{M}}_{n,l}(X,i)_1\cdots F^{\tilde{M}}_{n,l}(X,i)_d\Big) \\
b^{\mathrm{Pre}\,g}(X,i) &:= W^{(0)}b^{\tilde{M}}(X,i) + b^{(0)}.
\end{aligned}
$$

4.3 Experimental Contextualisation Analysis

We want to apply the theory developed in the previous sections to practice and analyse the contextualisation of different components in the FlexibleTransformer. It would be very nice if we could decompose the models from the experimental part of the previous chapter. However, the number of integrals to be computed for the IG-attribution of an MLP token-processor is $O(d_{\text{hidden}} N_{\text{in}} L_{\text{in}} NL)$. This makes it impractical to use this decomposition in large models, and is a natural goal for future research to make contextualisation analysis and interpretability available to large real-world applications. While we were able to compute the decomposition for several batches of size 32 on an Nvidia GA100, we were unable to compute the model decomposition for the entire ETTh1 test dataset (17420 timestamps in the original time series) in less than a day, which prevents us from performing meaningful large-scale analysis. We only present some examples of contextualisation in section 4.4.

So we construct smaller, artificial data that we think is suitable to study the contextualisation patterns of different FlexibleTransformer components.

4.3.1 Synthetic Data Generation

We have the following goals in mind when constructing synthetic datasets to study contextualisation:

- Dataset size: We want the dataset to allow for a study with significantly lower L_{in} and N_{in} to reduce the total computational load of the IG-attribution.
- Different ratios $L_{\text{in}}/N_{\text{in}}$: We are interested in how the model behaves in different scenarios. This reflects the challenges encountered in the real world, as classical time-series data have very few variates, while many modern applications have hundreds to thousands of variates.
- Different strengths of coupling between the variates: We want to interpolate between the extreme cases of independent variates and the scenario where the variates are highly coupled in the sense that it is necessary to use all other variates to predict each of them.

A key consideration is to decide on the nature of the interaction between the variates. Options include the following

- Periodic data: This is quite close to real-world applications, since most real-world datasets have some periodicity. On the one hand, a form of independence of the variates can be introduced by considering period lengths that are relatively prime to each other. On the other hand, a dependence of the variates can be achieved if the periods are multiples of each other. A disadvantage of this is that the periods can still become relatively large in the dependent case. The problem of overfitting to data seen during training can be alleviated to some extent by adding noise.
- Purely random data: We can achieve any kind of (in)dependence by, for example, controlling the covariance of a Gaussian process. While this is conceptually further away from real datasets than the previous aspect, the strength of this approach lies in the complete control (and easy mathematical analysis) of the process.

After careful consideration and experimental exploration, we decided against the conceptually appealing choice of using Gaussian processes because it is very difficult to actually train our models to detect patterns in such purely random data. We observed highly unstable training behaviour with a strong dependence on the (random) weight initialisation in cases where we chose non-trivial covariance functions for the Gaussian process. Examples for covariance functions that we have tried include shift relations of certain variates (some variates are predictable with other variates), variates as means of other variates, and non-linear interactions. Particularly in settings with large N_{in}, it was almost impossible to fit models reliably to the data. We observed a strong dependence on the weight initialisation. This negative result for Gaussian processes is contrasted with the use of periodic data. Here we can cleanly construct datasets that have the desired properties.

Construction of Periodic Data

As motivated above, we want to use data based on periodic patterns. A basic approach is as follows: We fix $N_{\text{in}} \in \{2^k : k \in \mathbb{N}\}$ and L_{in} to fit our computing resources. To control the interaction between the N_{in} variates, we group them into *blocks* of size $B \in \{1, 2, 4, \ldots, N_{\text{in}}\}$, where the total number of blocks is N_{in}/B. Each block has an associated frequency, which we use to construct the variates that belong to that block. We perturb the variates with Gaussian noise. Thus, for the periods $p_1, \ldots, p_{N_{\text{in}}/B}$, we have for the variate n

$$X_{n,l} = \sin\left(\frac{2\pi l}{p_k}\right) + \gamma e_{n,l} \qquad (e_{n',l'})_{\substack{n' \in \{1,\ldots,N_{\text{in}}\} \\ l' \in \{1,\ldots,L_{\text{in}}\}}} \sim \mathcal{N}(0,1) \quad \text{i.i.d.},$$

where $k = \lceil n/B \rceil$, for $l \in \{1, \ldots, L_{\text{in}}\}$ and $\gamma > 0$.

Our specific choices for the experiments in this section, which are within our computational limits, are as follows:

$$\begin{aligned} L_{\text{in}} &= 20 \\ N_{\text{in}} &\in \{16, 32\} \\ B &\in \{2, 4, 8, 16\} \\ \{p_1, \ldots, p_B\} &= \{p > 10 : p \text{ prime}\} \\ \gamma &= 0.3 \end{aligned}$$

Figure 4.1 shows this data in the case, where $L_{\text{in}} = 40$, $N_{\text{in}} = 16$, $B = 4$, $\gamma = 0.3$. We can observe that the data is split into several blocks.

4.3.2 Setup

The goal of this experiment is to better understand the contextualisation behaviour of the token-mixers and token-processor introduced in chapter 3. Therefore, we look at the token-processor and the token-mixers

$$\begin{aligned} \text{TM}_{\text{time}}, \text{TM}_{\text{variate}} &\in \{\text{SA}^2, \text{MLP}^{16}_{\text{TM}}\} \\ \text{TP} &= \text{MLP}^4_{\text{TP}}. \end{aligned}$$

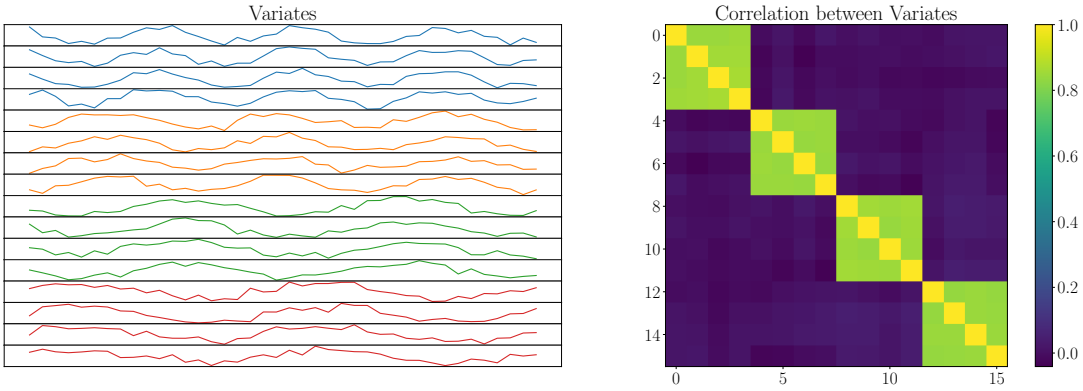

Figure 4.1: Example of periodic data with $L_{\text{in}} = 40$, $N_{\text{in}} = 16$, $B = 4$, $\gamma = 0.3$. Sample time-series (left) and correlation plot (right) show that the data is split into blocks of size $B = 4$. We use a larger L_{in} as compared to our experiments only for visualisation purposes.

To keep the experiment small, we fix H, d_{mixing} and d_{hidden}. We further use the following model and training hyperparameters:

$$\begin{aligned} d &= 8 \\ n_{\text{layers}} &= 1 \\ \text{dropout} &= 0.1 \\ \text{learning rate} &= 0.05 \\ \text{learning rate scheduler} &= \text{constant} \end{aligned}$$

The patch embedding with patch length one serves as the embedding.

4.3.3 Results and Analysis

Table 4.1 shows the results of the standard evaluation metrics. We consider the contextualisation of each component and the contextualisation of each cumulative intermediate model step. The following naming convention allows us to effectively refer to model components: We use two-letter abbreviations for each model component

TM : Time-Mixer	VM : Variate-Mixer
TP : Token-Processor	LN : Layer Normalisation

and chain several of these abbreviations together. For example, a 1-layer FlexibleTransformer with time- and variate-mixing and a token-processor would correspond to TMLNVMLNTPLN, and if we consider the same model only up to the variate-mixing step without the subsequent layer normalisation and token-processing, we would write TMLNVM.

In Table 4.2 we show the contextualisation for each individual component. To emphasise that we are only considering one component, we underline the component for which we are calculating the contextualisation metrics, i.e. TMLN$\underline{\text{VM}}$ for an isolated contextualisation study of the variate-mixer.

N	B	Time Mixer	Variate Mixer	MSE	MAE	MMaxError
16	2	MLP Token-Mixer	MLP Token-Mixer	0.196	0.355	0.818
			Self-Attention	0.189	0.347	0.808
		Self-Attention	MLP Token-Mixer	0.184	0.343	0.795
			Self-Attention	0.396	0.508	1.122
	4	MLP Token-Mixer	MLP Token-Mixer	0.167	0.321	0.770
			Self-Attention	0.168	0.322	0.768
		Self-Attention	MLP Token-Mixer	0.173	0.329	0.780
			Self-Attention	0.326	0.455	1.018
	8	MLP Token-Mixer	MLP Token-Mixer	0.170	0.335	0.769
			Self-Attention	0.165	0.327	0.771
		Self-Attention	MLP Token-Mixer	0.177	0.340	0.790
			Self-Attention	0.189	0.345	0.818
	16	MLP Token-Mixer	MLP Token-Mixer	0.176	0.338	0.767
			Self-Attention	0.165	0.327	0.750
		Self-Attention	MLP Token-Mixer	0.165	0.326	0.750
			Self-Attention	0.165	0.326	0.748
32	2	MLP Token-Mixer	MLP Token-Mixer	0.218	0.376	0.853
			Self-Attention	0.199	0.361	0.821
		Self-Attention	MLP Token-Mixer	0.194	0.356	0.815
			Self-Attention	0.449	0.530	1.132
	4	MLP Token-Mixer	MLP Token-Mixer	0.184	0.340	0.794
			Self-Attention	0.177	0.334	0.786
		Self-Attention	MLP Token-Mixer	0.170	0.328	0.769
			Self-Attention	0.420	0.521	1.131
	8	MLP Token-Mixer	MLP Token-Mixer	0.191	0.350	0.815
			Self-Attention	0.178	0.338	0.792
		Self-Attention	MLP Token-Mixer	0.193	0.351	0.822
			Self-Attention	0.340	0.466	1.049
	16	MLP Token-Mixer	MLP Token-Mixer	0.165	0.324	0.771
			Self-Attention	0.164	0.323	0.767
		Self-Attention	MLP Token-Mixer	0.162	0.321	0.760
			Self-Attention	0.201	0.356	0.833

Table 4.1: Evaluation metrics for contextualisation experiment.

We also analyse the contextualisation of the cumulative intermediate model steps in Table 4.3. Here we highlight the contextualisation by underlining all steps, i.e. TmLnVm for a contextualisation study of all three model components up to the variate-mixer.

We can observe from the tabulated data that there seems to be a relation between B and the error metrics which is also shown in Figure 4.2. In particular, we see that the discrepancies in performance mostly occur in the setting where the block size B is small. For small numbers of blocks, the models achieve similar performances in all three evaluation metrics.

However, there are a couple of notable observations. First, the empirical strength of the

			Component	TM		TMLN		TMLNVM		TMLNVMLN		TMLNVMLNTP		TMLNVMLNTPLN	
			Contextualisation Metric	spearman	amp	spearman	amp	spearman	amp	spearman	amp	spearman	amp	spearman	amp
N	B	Time Mixer	Variate Mixer												
16	2	MLP Token-Mixer	MLP Token-Mixer	0.472	0.058	0.718	0.057	0.000	0.000	0.219	0.033	0.001	0.016	0.008	0.068
			Self-Attention	0.705	0.100	0.629	0.098	0.050	0.009	0.022	0.075	0.000	0.004	0.001	0.053
		Self-Attention	MLP Token-Mixer	0.771	0.072	0.000	0.050	0.000	0.000	0.586	0.082	0.000	0.003	0.000	0.086
			Self-Attention	0.772	0.121	0.000	0.088	0.587	0.025	0.001	0.069	0.001	0.079	0.002	0.054
	4	MLP Token-Mixer	MLP Token-Mixer	0.492	0.087	0.699	0.048	0.749	0.207	0.041	0.122	0.002	0.019	0.006	0.075
			Self-Attention	0.467	0.050	0.716	0.090	0.267	0.107	0.008	0.098	0.002	0.023	0.003	0.082
		Self-Attention	MLP Token-Mixer	0.771	0.096	0.000	0.044	0.660	0.207	0.040	0.202	0.001	0.031	0.011	0.141
			Self-Attention	0.771	0.087	0.000	0.045	0.621	0.149	0.000	0.082	0.001	0.103	0.004	0.145
	8	MLP Token-Mixer	MLP Token-Mixer	0.002	0.000	0.902	0.017	0.095	0.152	0.203	0.059	0.001	0.054	0.001	0.063
			Self-Attention	0.781	0.190	0.582	0.093	0.511	0.311	0.003	0.200	0.000	0.035	0.001	0.171
		Self-Attention	MLP Token-Mixer	0.770	0.015	0.000	0.023	0.876	0.200	0.088	0.070	0.019	0.067	0.023	0.084
			Self-Attention	0.770	0.043	0.000	0.049	0.745	0.198	0.005	0.127	0.002	0.035	0.009	0.144
	16	MLP Token-Mixer	MLP Token-Mixer	0.779	0.187	0.581	0.050	0.524	0.607	0.009	0.135	0.002	0.260	0.007	0.363
			Self-Attention	0.590	0.056	0.665	0.042	0.433	0.129	0.008	0.083	0.001	0.031	0.002	0.049
		Self-Attention	MLP Token-Mixer	0.771	0.099	0.000	0.071	0.668	0.293	0.029	0.115	0.009	0.130	0.024	0.211
			Self-Attention	0.771	0.094	0.000	0.065	0.624	0.138	0.000	0.097	0.000	0.116	0.002	0.149
32	2	MLP Token-Mixer	MLP Token-Mixer	0.516	0.024	0.781	0.036	0.000	0.000	0.180	0.021	0.001	0.006	0.041	0.038
			Self-Attention	0.760	0.058	0.706	0.056	0.296	0.014	0.048	0.055	0.000	0.003	0.018	0.049
		Self-Attention	MLP Token-Mixer	0.773	0.046	0.000	0.027	0.000	0.000	0.700	0.051	0.000	0.002	0.000	0.053
			Self-Attention	0.775	0.102	0.000	0.053	0.708	0.067	0.004	0.127	0.005	0.082	0.003	0.100
	4	MLP Token-Mixer	MLP Token-Mixer	0.718	0.042	0.724	0.047	0.000	0.000	0.207	0.031	0.000	0.006	0.025	0.046
			Self-Attention	0.754	0.061	0.710	0.066	0.125	0.008	0.033	0.055	0.000	0.000	0.006	0.034
		Self-Attention	MLP Token-Mixer	0.773	0.048	0.000	0.027	0.000	0.000	0.701	0.060	0.000	0.006	0.000	0.053
			Self-Attention	0.774	0.077	0.000	0.045	0.710	0.071	0.004	0.086	0.004	0.040	0.001	0.075
	8	MLP Token-Mixer	MLP Token-Mixer	0.608	0.065	0.750	0.037	0.402	0.213	0.117	0.113	0.002	0.023	0.015	0.112
			Self-Attention	0.778	0.125	0.700	0.078	0.438	0.205	0.005	0.167	0.000	0.000	0.001	0.125
		Self-Attention	MLP Token-Mixer	0.774	0.055	0.000	0.048	0.000	0.000	0.701	0.061	0.000	0.004	0.000	0.047
			Self-Attention	0.774	0.081	0.000	0.056	0.737	0.296	0.001	0.241	0.005	0.187	0.008	0.122
	16	MLP Token-Mixer	MLP Token-Mixer	0.747	0.113	0.708	0.058	0.642	0.420	0.037	0.124	0.019	0.152	0.005	0.182
			Self-Attention	0.777	0.138	0.701	0.063	0.403	0.242	0.003	0.195	0.000	0.016	0.001	0.131
		Self-Attention	MLP Token-Mixer	0.774	0.061	0.000	0.055	0.523	0.166	0.290	0.099	0.005	0.078	0.010	0.117
			Self-Attention	0.773	0.035	0.000	0.033	0.800	0.190	0.001	0.066	0.000	0.016	0.003	0.100

Table 4.2: Contextualisation metrics of individual model components. We present this table with numerical values only for completeness. The reader may safely skip the detailed study of this table.

			Component	TM		TMLN		TMLNVM		TMLNVMLN		TMLNVMLNTP		TMLNVMLNTPLN	
			Contextualisation Metric	spearman	amp	spearman	amp	spearman	amp	spearman	amp	spearman	amp	spearman	amp
N	B	Time Mixer	Variate Mixer												
16	2	MLP Token-Mixer	MLP Token-Mixer	0.472	0.058	0.904	0.073	0.904	0.073	0.904	0.082	0.904	0.078	0.904	0.078
			Self-Attention	0.705	0.100	0.904	0.110	0.904	0.114	0.904	0.126	0.904	0.125	0.904	0.106
		Self-Attention	MLP Token-Mixer	0.771	0.072	0.771	0.089	0.771	0.089	0.904	0.111	0.904	0.110	0.904	0.095
			Self-Attention	0.772	0.121	0.771	0.126	0.904	0.140	0.904	0.128	0.904	0.138	0.904	0.146
	4	MLP Token-Mixer	MLP Token-Mixer	0.492	0.087	0.904	0.085	0.906	0.197	0.906	0.173	0.906	0.172	0.906	0.157
			Self-Attention	0.467	0.050	0.904	0.085	0.905	0.144	0.905	0.190	0.905	0.186	0.905	0.122
		Self-Attention	MLP Token-Mixer	0.771	0.096	0.771	0.119	0.903	0.260	0.905	0.243	0.905	0.240	0.905	0.204
			Self-Attention	0.771	0.087	0.771	0.111	0.904	0.221	0.904	0.221	0.904	0.197	0.904	0.140
	8	MLP Token-Mixer	MLP Token-Mixer	0.002	0.000	0.903	0.017	0.906	0.151	0.906	0.149	0.906	0.134	0.906	0.111
			Self-Attention	0.781	0.190	0.908	0.162	0.924	0.369	0.924	0.384	0.924	0.382	0.925	0.227
		Self-Attention	MLP Token-Mixer	0.770	0.015	0.770	0.029	0.906	0.194	0.906	0.176	0.906	0.154	0.906	0.138
			Self-Attention	0.770	0.043	0.770	0.057	0.904	0.195	0.904	0.188	0.904	0.180	0.904	0.128
	16	MLP Token-Mixer	MLP Token-Mixer	0.779	0.187	0.907	0.170	0.952	0.608	0.952	0.544	0.953	0.477	0.954	0.135
			Self-Attention	0.590	0.056	0.904	0.064	0.904	0.163	0.904	0.183	0.904	0.169	0.904	0.141
		Self-Attention	MLP Token-Mixer	0.771	0.099	0.772	0.109	0.906	0.342	0.906	0.352	0.906	0.327	0.907	0.244
			Self-Attention	0.771	0.094	0.772	0.120	0.905	0.233	0.905	0.253	0.905	0.219	0.906	0.144
32	2	MLP Token-Mixer	MLP Token-Mixer	0.516	0.024	0.932	0.043	0.932	0.043	0.932	0.045	0.932	0.045	0.932	0.040
			Self-Attention	0.760	0.058	0.932	0.067	0.932	0.078	0.932	0.087	0.932	0.086	0.932	0.061
		Self-Attention	MLP Token-Mixer	0.773	0.046	0.773	0.055	0.773	0.055	0.932	0.064	0.932	0.064	0.932	0.055
			Self-Attention	0.775	0.102	0.774	0.097	0.932	0.161	0.932	0.134	0.932	0.092	0.932	0.100
	4	MLP Token-Mixer	MLP Token-Mixer	0.718	0.042	0.932	0.061	0.932	0.061	0.932	0.061	0.932	0.061	0.932	0.056
			Self-Attention	0.754	0.061	0.932	0.078	0.932	0.082	0.932	0.093	0.932	0.093	0.932	0.075
		Self-Attention	MLP Token-Mixer	0.773	0.048	0.773	0.062	0.773	0.062	0.932	0.076	0.932	0.076	0.932	0.054
			Self-Attention	0.774	0.077	0.774	0.092	0.932	0.151	0.932	0.133	0.932	0.125	0.932	0.132
	8	MLP Token-Mixer	MLP Token-Mixer	0.608	0.065	0.932	0.065	0.934	0.208	0.934	0.183	0.934	0.185	0.934	0.102
			Self-Attention	0.778	0.125	0.933	0.115	0.937	0.246	0.937	0.263	0.937	0.263	0.938	0.143
		Self-Attention	MLP Token-Mixer	0.774	0.055	0.773	0.077	0.773	0.077	0.932	0.084	0.932	0.084	0.932	0.061
			Self-Attention	0.774	0.081	0.774	0.095	0.933	0.347	0.932	0.258	0.932	0.143	0.932	0.122
	16	MLP Token-Mixer	MLP Token-Mixer	0.747	0.113	0.933	0.097	0.961	0.402	0.963	0.349	0.962	0.300	0.962	0.140
			Self-Attention	0.777	0.138	0.933	0.116	0.940	0.295	0.940	0.290	0.940	0.279	0.940	0.161
		Self-Attention	MLP Token-Mixer	0.774	0.061	0.773	0.063	0.882	0.205	0.932	0.195	0.932	0.164	0.932	0.101
			Self-Attention	0.773	0.035	0.773	0.034	0.932	0.196	0.932	0.211	0.932	0.213	0.932	0.138

Table 4.3: Contextualisation of cumulative intermediate model steps. We present this table with numerical values only for completeness. The reader may safely skip the detailed study of this table.

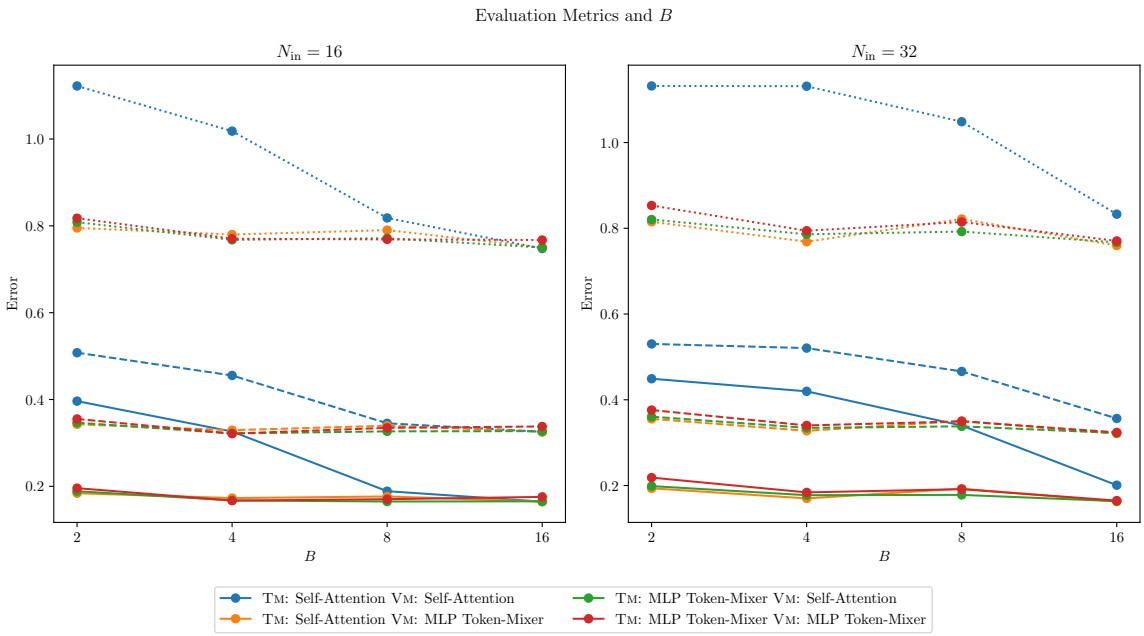

Figure 4.2: Relationship between number of blocks B and model error for different architectures. The solid line shows the mean squared error, the dashed line the mean absolute error and the dotted line the mean maximum error.

architecture using the MLP token-mixer along the time-axis and self-attention along the variate-axis seems to be superior in settings where there is little sparsity in the correlations between the different variates of the data (here $B \in \{4, 8, 16\}$ in the case $N_{\text{in}} = 16$ and $B = 8$ in the case $N_{\text{in}} = 32$). In the other cases, the use of self-attention along the time-axis and MLP token-mixer along the variate axis appears to work better in high-sparsity settings. This hints at a superior performance of self-attention along the variate-axis in cases, where there are many dependent variates (also visually observable in Figure 4.2). Secondly, we observe a sharp drop for the FlexibleTransformer configuration, which uses self-attention to mix along the time- and variate-axes. We want to pay particular attention to these results when analysing the contextualisation patterns of these models.

Relationship between Model Performance and Contextualisation

The goals of the contextualisation study were twofold before we began this study: First, we wanted to better understand the internal flow of the model and better identify the important architectural components for the FlexibleTransformer. Secondly, we had a vague hope that there was a relationship between contextualisation and model performance. The concrete hypothesis was that we could find a correlation between better performing models and higher contextualisation.

While all the data has been presented in Tables 4.1, 4.2 and 4.3, let us visualise the relationship we are interested in. We plot this in Figures 4.3 for the case $N_{\text{in}} = 16$ and in 4.4 for the case $N_{\text{in}} = 32$.

There is no discernible pattern and, in particular, no negative correlation between the mean

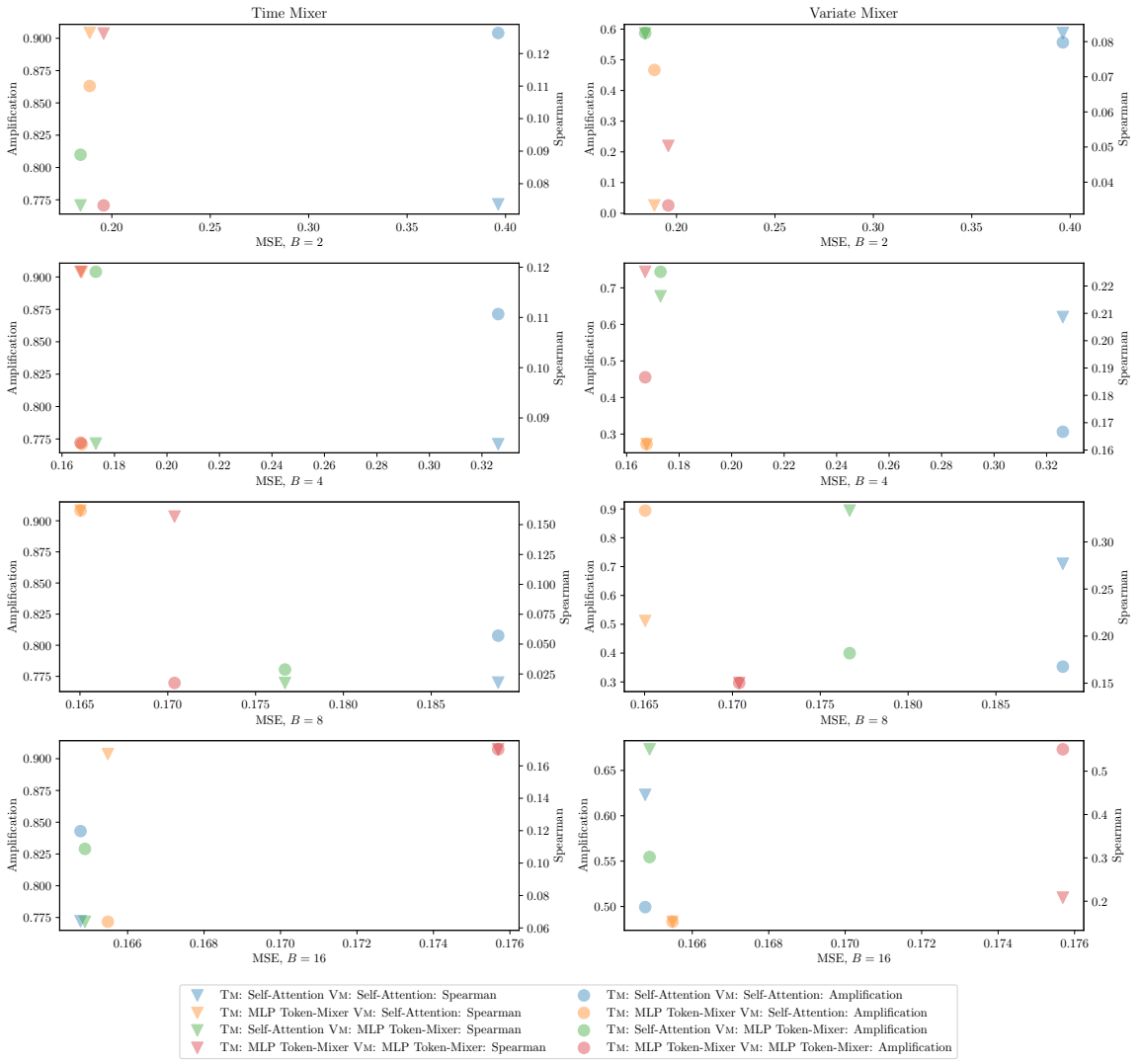

Figure 4.3: Relationship between contextualisation metrics and MSE for $N_{\text{in}} = 16$. Only displays mean contextualisation for each FlexibleTransformer configuration.

MSE and the mean of any of the contextualisation metrics between different architectures. Therefore, we also look at the data on a finer scale in Table 4.4 by computing the Pearson correlation coefficients between the MSE and the contextualisation metrics for each token-mixer along each axis and in each setting. Unfortunately, we cannot generally conclude that higher contextualisation in either the time-mixer or the variate-mixer leads to better model performance in terms of MSE.

Contextualisation Flow

Next, we want to pursue the first goal listed above and better understand which are the important components of the FlexibleTransformer as measured by contextualisation. After the previous negative result, it seems important to first answer whether our contextualisation metrics actually depend on factors that we can control. Therefore, we ask whether the contextualisation of the variate-mixing step is higher in settings with larger blocks, where

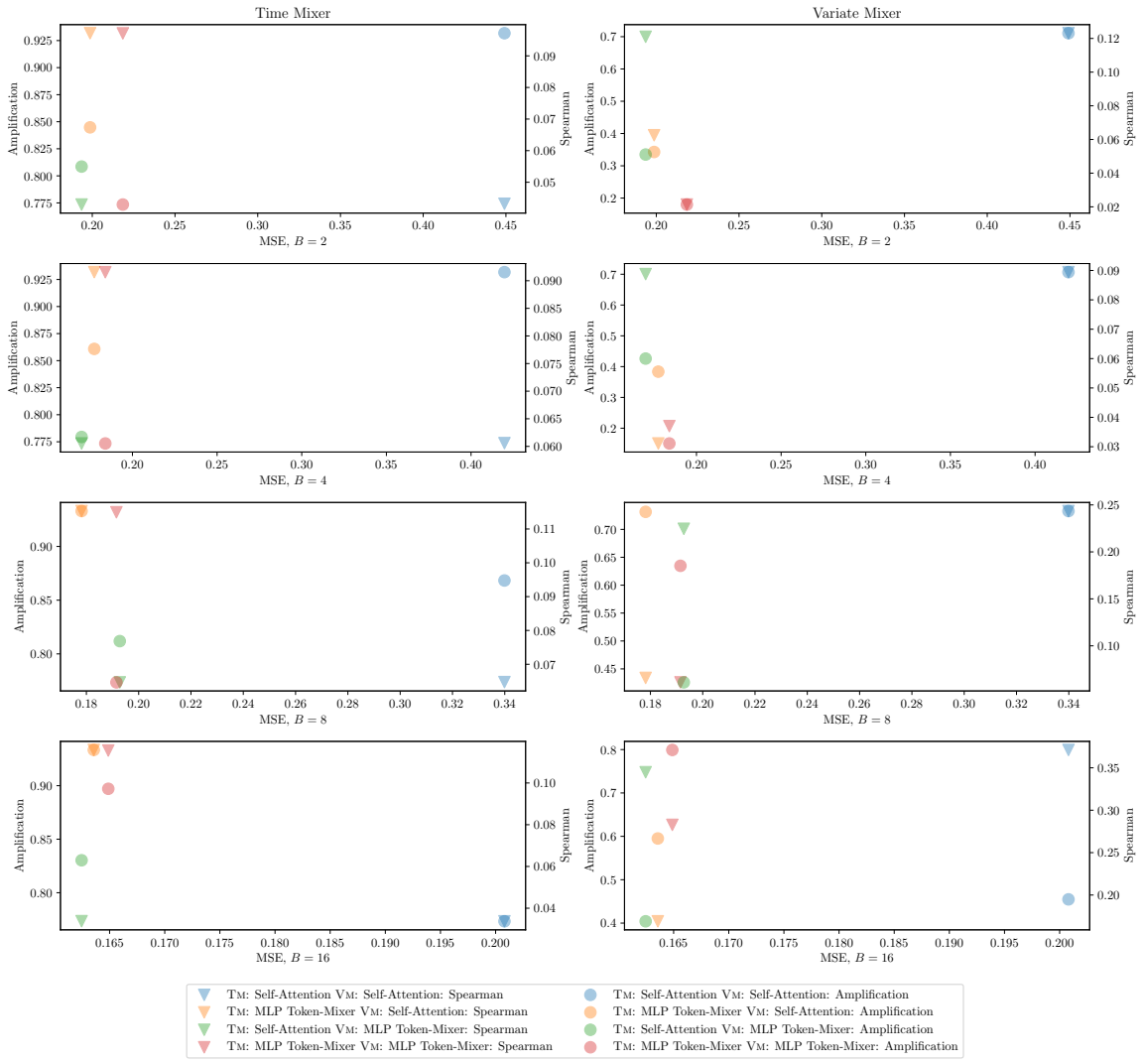

Figure 4.4: Relationship between contextualisation metrics and MSE for $N_{\text{in}} = 32$. Only displays mean contextualisation for each FlexibleTransformer configuration.

	Token-Mixer	MLP Token-Mixer				Self-Attention			
	Axis	Time		Variate		Time		Variate	
		Amplification	Spearman	Amplification	Spearman	Amplification	Spearman	Amplification	Spearman
N	B								
16	2	0.175	0.204	0.079	-0.088	-0.008	0.073	0.382	0.117
	4	0.290	-0.137	0.263	0.063	0.117	-0.243	0.244	-0.273
	8	-0.098	-0.304	-0.096	-0.020	-0.276	-0.113	0.066	0.133
	16	0.063	0.250	0.062	-0.117	-0.162	0.211	-0.043	0.323
32	2	0.361	-0.222	0.249	0.349	0.057	-0.085	0.021	-0.061
	4	-0.125	0.458	-0.146	0.040	0.261	-0.279	-0.243	0.048
	8	-0.072	-0.073	0.074	-0.167	0.015	-0.269	-0.070	-0.196
	16	0.056	-0.156	0.127	-0.150	0.472	0.036	0.017	0.096

Table 4.4: Correlation computed over all test samples for all settings between contextualisation metrics and MSE.

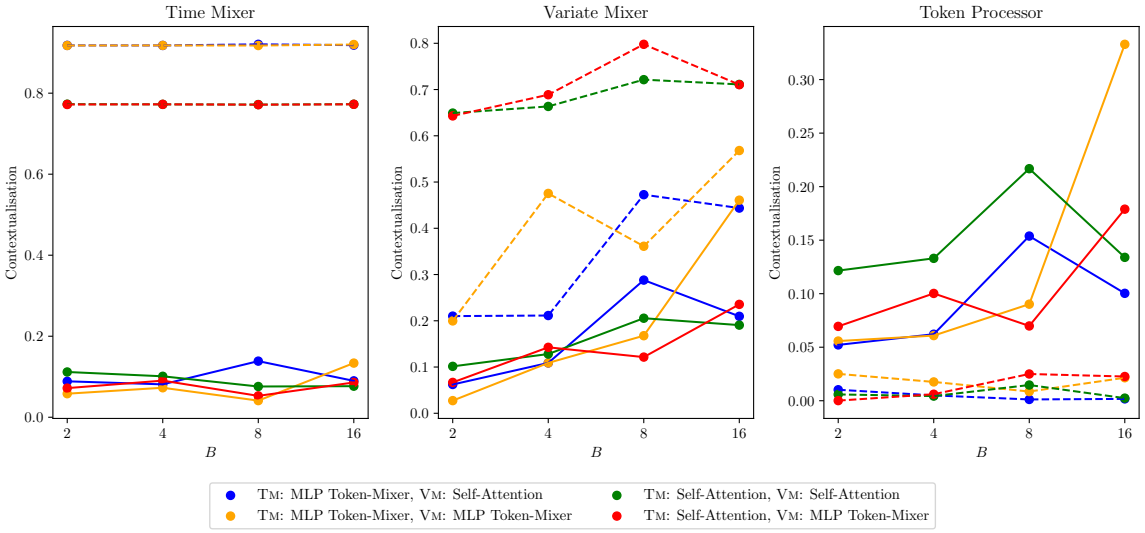

Figure 4.5: Spearman contextualisation metric (dotted) and amplification contextualisation mertic with respect to different sizes of blocks B.

more variates can be combined to achieve better forecasting results. In Figure 4.5 we observe that contextualisation does not increase with increasing block size B in the time-mixer, but that there is a sharp increase in contextualisation for both contextualisation metrics for the variate-mixer. These results are consistent with our previous hypothesis.

Interestingly, we also observe an increase in the amplification contextualisation metric for the token-processor which underlines the importance of token-processing in the (Flexible)Transformer architecture. The concrete role of token-processing is currently not properly understood. We can however point to [Gev+21; Gev+22] where research on the role of the MLP token-processor is conducted.

To even better understand the internal dynamics within the model and to analyse the interplay of the different components, we examine the contextualisation flow through our network. We show the contextualisation of each individual model component and also the contextualisation of the respective cumulative intermediate model step in Figure 4.6 for $N_{\text{in}} = 16$ and in 4.7 for $N_{\text{in}} = 32$.

We can make the following observations:

- The choice of contextualisation metric has a strong influence on the qualitative result

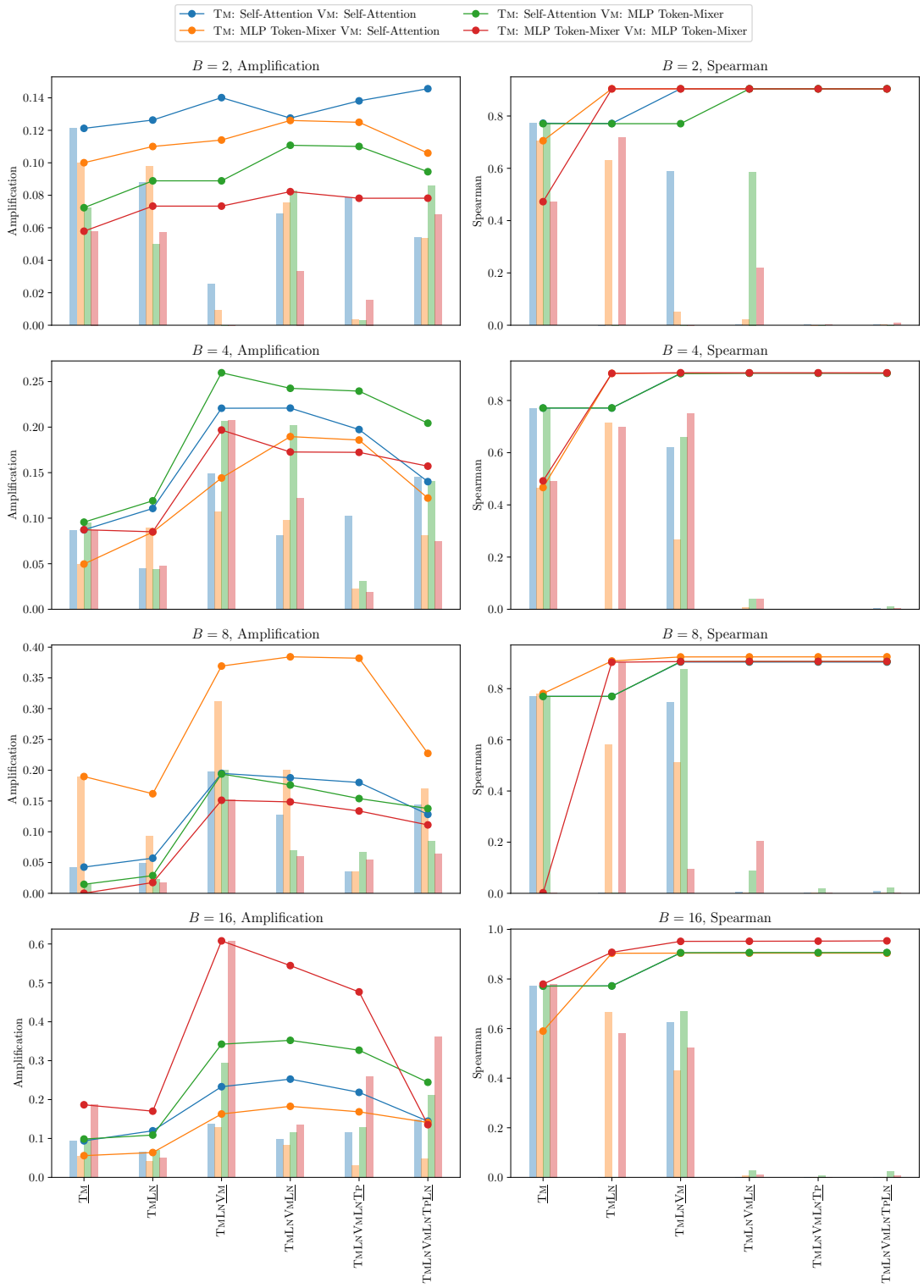

Figure 4.6: Contextualisation flow for $N_{\text{in}} = 16$. Bars indicate contextualisation of isolated model step and line shows contextualisation for the respective cumulative intermediate model step.

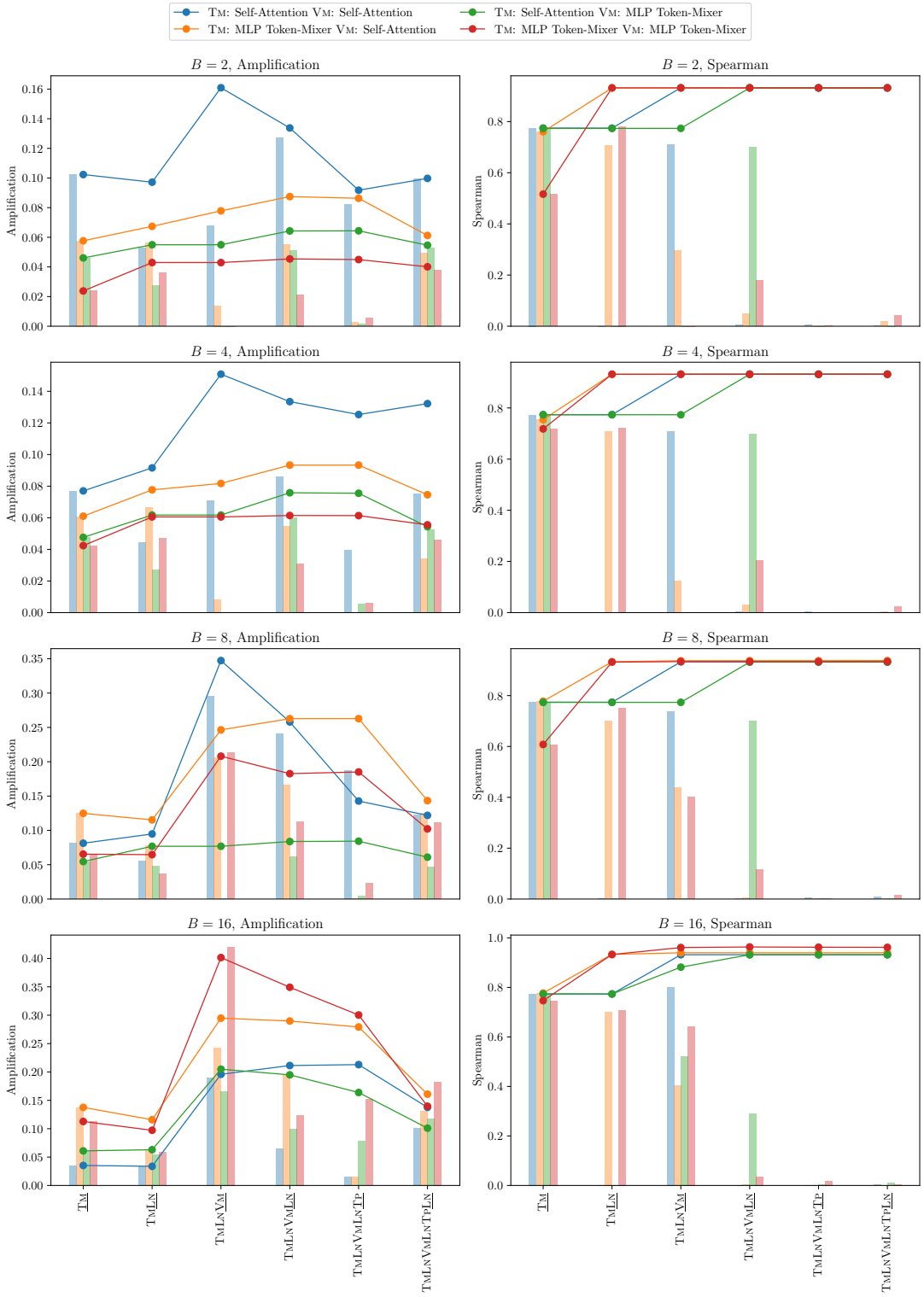

Figure 4.7: Contextualisation flow for $N_{\text{in}} = 32$. Bars indicate contextualisation of isolated model step and line shows contextualisation for the respective cumulative intermediate model step.

observed: If we measure the contextualisation for all model steps with the spearman contextualisation metric, the effects of the layer normalisation and the token-processor are hardly noticeable.

- In general, the cumulative contextualisation is not monotonic, as measured by the amplification contextualisation metric. We have a strong contextualisation by the token-mixers that is reversed by subsequent model steps (in particular by LayerNorm). This pattern is most evident when the MLP token-mixer is used to mix along the time-axis. The model seems to overshoot, which needs to be corrected by strong contextualisation in the last LayerNorm layer.

- As expected, the contextualisation of individual model steps is greatest for the token-mixers. This is most evident when measured by the Spearman contextualisation metric.

- Comparing the contextualisation flow between the cases $N_{\text{in}} = 16$ and $N_{\text{in}} = 32$, we observe a similar behaviour for the Spearman contextualisation metric (except for the higher effect of the first LayerNorm). Furthermore, the cumulative contextualisation measured by the amplification contextualisation metric is less variable in the case $N_{\text{in}} = 32$ and the effect of the variate-mixer is particularly pronounced.

Conclusion

In conclusion, contrary to our hopes, we do not find a relationship between contextualisation and model performance.

However, we find that the combination of mixing with the MLP token-mixer along the time-axis and self-attention along the variate-axis is preferable in cases where there is litte sparsity between the variates, i.e. when many variates have to be combined. Using the MLP token-mixer along the variate-axis is preferable in settings where there is a high sparsity between variates.

The contextualisation procedure carried out in this chapter appears to be appropriate and able to capture the changing contextualisation that occurs in the model and which we can relate to the synthetic datasets.

We have found that the cumulative contextualisation flow is surprisingly non-monotonic in terms of the amplification contextualisation metric. A positive answer to the question of whether a more monotonic cumulative contextualisation flow is associated with better model performance would require a serious re-evaluation of the layer normalisation in the Transformer architecture (at least for time-series applications). This requires careful experimental design and is beyond the scope of this work.

4.4 Model Interpretability

This section shows how we can use the decomposition framework to interpret the output of the model. We saw the decomposition of intermediate steps in the last section, which gave us $(\boldsymbol{F}^{\tilde{M}}(X), \boldsymbol{b}^{\tilde{M}}(X))$.

It is straightforward to also decompose the linear decoder, which leads us to have a decomposition of each variate at each prediction timestamp. Thus we have

$$\boldsymbol{F}^{\text{complete}}(X) = \left(\left(F_{n,l}^{\text{complete}}(X,(m,t)) \right)_{\substack{n\in\{1,\dots,N_{\text{in}}\} \\ l\in\{1,\dots,L_{\text{in}}\}}} \right)_{\substack{m\in\{1,\dots,N_{\text{in}}\} \\ t\in\{1,\dots,T\}}}$$

$$\boldsymbol{b}^{\text{complete}}(X) = \left(b^{\text{complete}}(X,(m,t)) \right)_{\substack{m\in\{1,\dots,N_{\text{in}}\} \\ t\in\{1,\dots,T\}}}$$

such that

$$\text{FlexibleTransformer}(X)_{m,t} = \sum_{\substack{n\in\{1,\dots,N_{\text{in}}\} \\ l\in\{1,\dots,L_{\text{in}}\}}} F_{n,l}^{\text{complete}}(X,(m,t)) + b^{\text{complete}}(X,(m,t)),$$

for $(m,t) \in \{1,\dots,N_{\text{in}}\} \times \{1,\dots,T\}$.

We present this model decomposition visually for a number of examples to illustrate the strength of this approach for model interpretability tasks. In the examples below, we show only the attribution of a predicted value and colour the values in the input sequence according to $\boldsymbol{F}^{\text{complete}}(X)$. Since we are not focusing on the architecture of the model in this section, but only on the interpretability, we fix the architecture of the model whose decomposition we show. We use the FlexibleTransformer with the MLP token-mixer to mix along the time- and variate-axes.

We want to show the interpretability of the FlexibleTransformer in synthetic and real environments, using the artificial datasets from the previous section, but also examples from the benchmark datasets listed in appendix A.6. In all examples shown below, we interpret the prediction of a timestamp from the first variate (also marked red in the plots). In Figure 4.8 we show an interpretability plot for the synthetic data from this chapter, which shows that the model has learned well the block of variates relevant to the first variate. In Figure 4.9 we can see how different patterns affect the prediction. This type of interpretability plot has the potential to inform expert decisions.

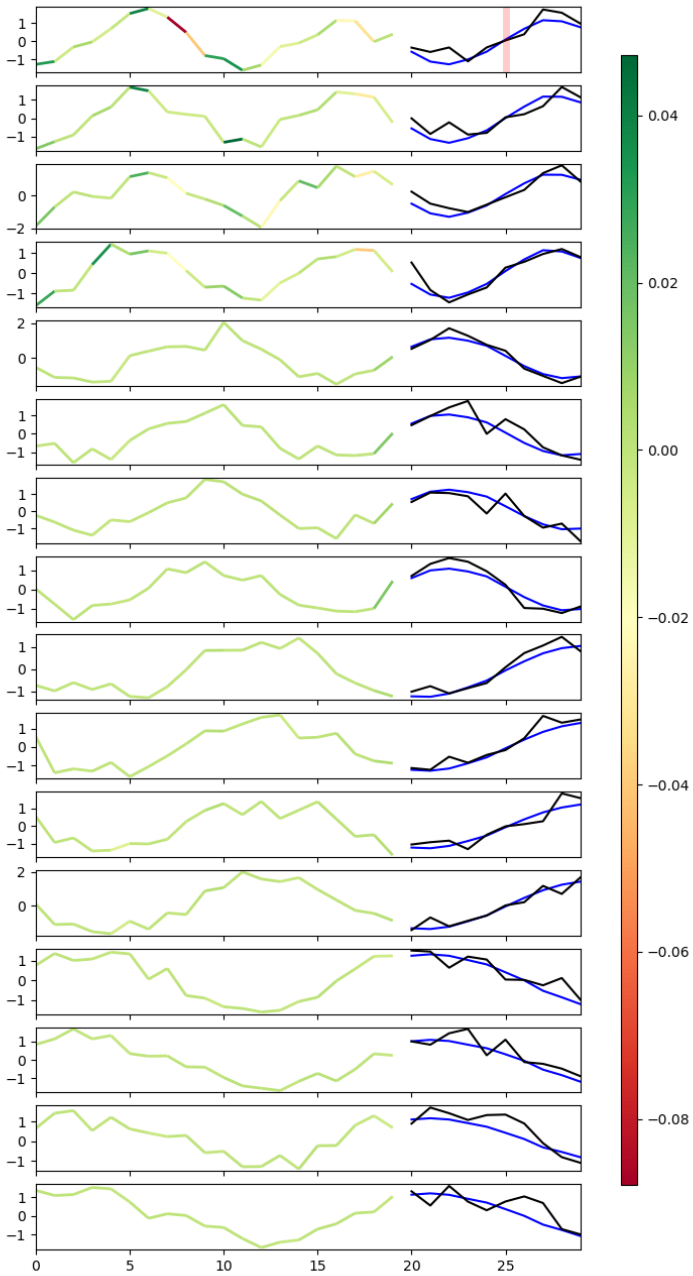

Figure 4.8: Interpretability plot of synthetic dataset as described in section 4.3.1 with $N_{\text{in}} = 16$, $T = 10$ and $B = 4$. Colour indicates value of $F_{n,l}^{\text{complete}}(X, (1, T/2))$.

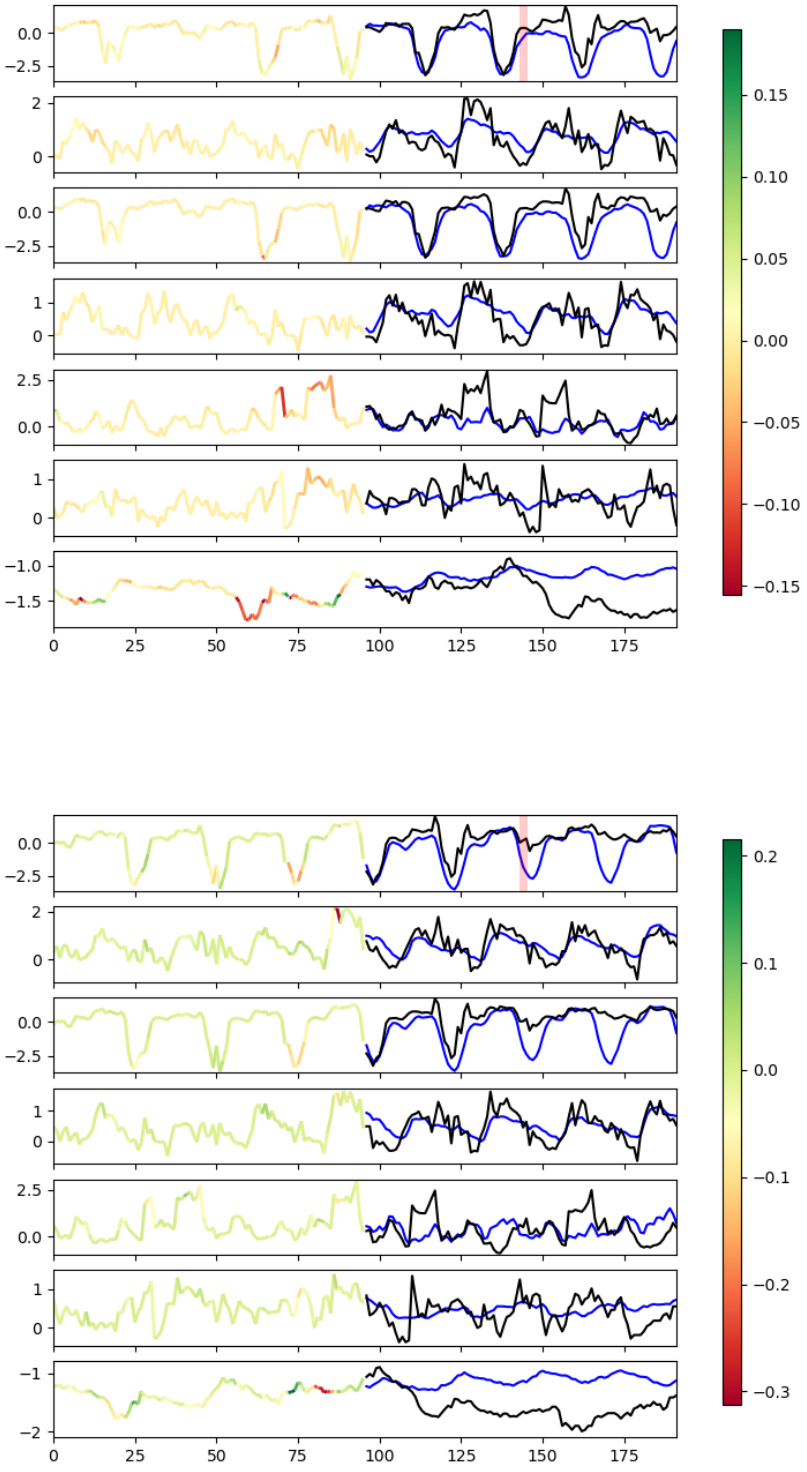

Figure 4.9: Interpretability plots for samples from ETTh1 dataset with $L_{\text{in}} = T = 96$. Colour indicates value of $F_{n,l}^{\text{complete}}(X, (1, T/2))$.

Chapter 5

Efficient Attention Approximation along Variates

We have seen in chapters 2 and 3 that we can apply self-attention along the variate-axis. While efficient self-attention schemes have been thoroughly explored in transformer-based time series forecasting models [LI+19; Zho+21; Liu+22a; Wu+21; Zho+22], the recent need to apply self-attention along the variate-axis ([Liu+24] and chapter 3) requires us to investigate attention approximation approaches in high-dimensional settings.

The time-complexity of self-attention is further exacerbated by the structure of modern datasets. Often there are hundreds or even thousands of variates, and in time-critical applications such as high-frequency trading, we cannot rely on methods that only reduce the training time of such models, but also require reduced time-complexity during inference.

To the author's knowledge, the only approach that has been explored to make self-attention along the variate-axis more efficient has been proposed in [Liu+24]. The authors' approach is to exploit the fact that we do not use positional encoding for embedding along the variate-axis, and thus can simply drop variates during training. This is only made possible by the permutation equi-variance of self-attention. The main disadvantage of this approach is that we use only some of the variates in each training iteration, and we cannot use this approach during inference and still benefit from the interdependencies between *all* the variates.

Therefore, we want to take a first step towards exploring existing attention approximation schemes for variate-mixing. Since a sequence of tokens along the variate-axis does not have the same properties as a time-series, we rely on general-purpose attention approximation schemes instead of those mentioned in the literature review in chapter 2.

There are many general approaches to improve transformer inference [Chi+23]. Examples include pruning, quantisation, hardware-aware optimisation, and the design of efficient attention approximation schemes. We will focus on the latter. Two polar opposites in the literature on attention approximation schemes are methods that focus on finding the best approximation of the most important key-value pairs [KKL20] or those that focus on finding a low-dimensional representation of the attention matrix [Wan+20; Qin+22; Cho+21]. We call these two approaches *sparse attention approximation* and *low-rank attention approxi-*

mation. We want to study the *locality sensitive hashing (LSH)* method from the Reformer model [KKL20] as a method with sparse attention approximation and *fast attention via positive orthorgonal random features (FAVOR+)* from the Performer model [Cho+21] as a method with a low-rank method for efficient attention approximation in high-variate settings.

In the first part of the chapter, we give a theoretical presentation of the attention approximation methods FAVOR+ and LSH. We then introduce metrics that we use in our experiments to relate model performance to properties of the attention matrix and the data. Finally, we perform numerical experiments on synthetic data and relate model performance to the previously introduced metrics.

5.1 FAVOR+

Fast Attention Via positive Orthorgonal Random features (FAVOR+) is an approach for efficient attention approximation presented by Choromanski et al. in [Cho+21]. It can also be applied to kernelisable attention approximation for kernels beyond the SoftMax-kernel introduced in chapter 2.

As in chapter 2, let $\boldsymbol{\mathcal{Q}} = (\boldsymbol{q}_j)_{j\in\mathcal{J}}$, $\boldsymbol{\mathcal{K}} = (\boldsymbol{k}_j)_{j\in\mathcal{J}}$ and $\boldsymbol{\mathcal{V}} = (\boldsymbol{v}_j)_{j\in\mathcal{J}}$ denote the queries, keys and values for one attention head that mixes along variates. Since $\mathcal{J} := \{1, \dots, N\}$ we can identify $\mathcal{Q}, \mathcal{K}$ and $\mathcal{V}$ with matrices in $\mathbb{R}^{N\times d_{\text{attn}}}$. We recall the definition of the attention head as

$$\text{AttnHead}(\mathcal{Q}, \mathcal{K}, \mathcal{V})_i = \sum_{j=1}^{N} \text{SoftMax}\left(\phi\left(\boldsymbol{q}_i, \boldsymbol{k}_{i'}\right)_{i'\in\{1,\dots,N\}}\right)_j \boldsymbol{v}_j.$$

Definition 5.1. Let $K : \mathbb{R}^{d_{\text{attn}}} \times \mathbb{R}^{d_{\text{attn}}} \to \mathbb{R}_+$ and AttnHead as in equation (5.1). If we have

$$\text{AttnHead}(\mathcal{Q}, \mathcal{K}, \mathcal{V}) = \boldsymbol{D}^{-1}\boldsymbol{A}\boldsymbol{\mathcal{V}} \qquad \boldsymbol{A}_{ij} := K(\boldsymbol{\mathcal{Q}}_i, \boldsymbol{\mathcal{K}}_{\boldsymbol{j}}) \qquad \boldsymbol{D} := \text{diag}(\boldsymbol{A}\boldsymbol{1}_N),$$

for all $\boldsymbol{\mathcal{Q}}, \boldsymbol{\mathcal{K}}, \boldsymbol{\mathcal{V}} \in \mathbb{R}^{N\times d_{\text{attn}}}$, then we call AttnHead kernelisable with the kernel K and $\boldsymbol{A} \in \mathbb{R}^{N\times N}$ the (unnormalised) attention matrix for $\boldsymbol{\mathcal{Q}}, \boldsymbol{\mathcal{K}}, \boldsymbol{\mathcal{V}} \in \mathbb{R}^{N\times d_{\text{attn}}}$.

The term $\boldsymbol{D}^{-1}$ corresponds to the normalisation of the kernel to yield a probability measure on the set of keys and $\boldsymbol{1}_N \in \mathbb{R}^N$ is the vector of ones.

It is immediate to see that AttnHead is kernelisable with

$$K(\boldsymbol{\mathcal{Q}}_i, \boldsymbol{\mathcal{K}}_j) := \exp\left(\frac{\boldsymbol{\mathcal{Q}}_i \boldsymbol{\mathcal{K}}_j^t}{\sqrt{d_{\text{attn}}}}\right) \tag{5.1}$$

since we have SoftMax-activations and the scaled dot-product as the similarity kernel. We can now introduce the FAVOR+ mechanism.

Definition 5.2. Let $(\Omega, \mathcal{F}, \mathbb{P})$ be a probability space. If there is $\psi : \mathbb{R}^{d_{\text{attn}}} \to \text{Map}\left(\Omega, \mathbb{R}_+^r\right)$ for some $r \in \mathbb{N}$ such that

$$K(\boldsymbol{q}, \boldsymbol{k}) = \mathbb{E}\left[\psi(\boldsymbol{q})^t\, \psi(\boldsymbol{k})\right] \tag{5.2}$$

holds for all $\boldsymbol{q}, \boldsymbol{k} \in \mathbb{R}^{d_{\text{attn}}}$, we call ψ *in accordance with* K and define the estimator $\widehat{\text{AttnHead}}$ for AttnHead by

$$\widehat{\text{AttnHead}}_{\psi}(\boldsymbol{\mathcal{Q}}, \boldsymbol{\mathcal{K}}, \boldsymbol{\mathcal{V}}) := \hat{\boldsymbol{D}}^{-1}\left(\boldsymbol{\mathcal{Q}}'\left(\left(\boldsymbol{\mathcal{K}}'\right)^t \boldsymbol{\mathcal{V}}\right)\right),$$

where

$$\boldsymbol{\mathcal{Q}}'_i := \psi(\boldsymbol{\mathcal{Q}}_i) \qquad \boldsymbol{\mathcal{K}}'_i := \psi(\boldsymbol{\mathcal{K}}_i) \qquad \hat{\boldsymbol{D}} := \text{diag}\left(\boldsymbol{\mathcal{Q}}'\left(\left(\boldsymbol{\mathcal{K}}'\right)^t \mathbf{1}_N\right)\right)$$

We further call $\psi(\boldsymbol{x})$ a *random feature map* and the brackets define the order of computation.

Typically, we choose $r \ll N$ and have a reduced computational complexity of $O(Nrd_{\text{attn}})$ for $\widehat{\text{AttnHead}}_{\psi}$. We give theoretical estimates for the errors of the estimator after we have identified concrete ψ for the AttnHead from chapter 2.

5.1.1 Kernels

We want to find probability spaces $(\Omega, \mathcal{F}, \mathbb{P})$ and maps ψ in accordance with the kernel K defined in equation (5.1) that allow us to find estimators for AttnHead as described above.

Let $m \in \mathbb{N}$. To motivate the concrete choice of ψ for FAVOR+, we start with a naive choice for such a random feature map. We will only see the downsides of this choice after we have introduced it. One possibility is to have independent random variables $X_1, \ldots, X_m \sim \mathcal{N}(0, \frac{1}{\sqrt{d_{\text{attn}}}} I_{d_{\text{attn}}})$ and define

$$\psi_m^{\text{trig}}(\boldsymbol{x})(\omega) := \frac{1}{\sqrt{m}} \exp\left(\frac{\|x\|^2}{2\sqrt{d_{\text{attn}}}}\right)(\sin(X_1(\omega)^t \boldsymbol{x}), \ldots, \sin(X_m(\omega)^t \boldsymbol{x}), \cos(X_1(\omega)^t \boldsymbol{x}), \ldots, \cos(X_m(\omega)^t \boldsymbol{x})).$$

The proof of the following lemma relies on ideas from [RR07].

Lemma 5.1. ψ_m^{trig} satisfies equation (5.2).

Proof. We can identify the characteristic function of X_j for $j \in \{1, \ldots, m\}$ as

$$\mathbb{E}\left[\exp\left(iX_j^t(\boldsymbol{q} - \boldsymbol{k})\right)\right] = \exp\left(-\frac{\|\boldsymbol{q} - \boldsymbol{k}\|^2}{2\sqrt{d_{\text{attn}}}}\right).$$

Since the right-hand side is real, we have

$$\mathbb{E}\left[\exp\left(iX_j^t(\boldsymbol{q} - \boldsymbol{k})\right)\right] = \mathbb{E}\left[\cos(X_j^t(\boldsymbol{q} - \boldsymbol{k}))\right].$$

The trigonometric identity

$$\cos(x - y) = \sin(x)\sin(y) + \cos(x)\cos(y)$$

for $x, y \in \mathbb{R}$ leads to

$$\begin{aligned}
&\mathbb{E}\left[\psi^{\text{trig}}(\boldsymbol{q})^t \psi^{\text{trig}}(\boldsymbol{k})\right] \\
=&\frac{1}{m}\exp\left(\frac{\|\boldsymbol{q}\|^2+\|\boldsymbol{k}\|^2}{2\sqrt{d_{\text{attn}}}}\right)\mathbb{E}\left[\sum_{j=1}^{m}\left(\sin(X_j^t\boldsymbol{q})\sin(X_j^t\boldsymbol{k})+\cos(X_j^t\boldsymbol{q})\cos(X_j^t\boldsymbol{k})\right)\right] \\
=&\exp\left(\frac{\|\boldsymbol{q}\|^2+\|\boldsymbol{k}\|^2}{2\sqrt{d_{\text{attn}}}}\right)\mathbb{E}\left[\cos(X^t(\boldsymbol{q}-\boldsymbol{k}))\right] \\
=&\exp\left(\frac{\boldsymbol{q}^t\boldsymbol{k}}{\sqrt{d_{\text{attn}}}}\right).
\end{aligned}$$

This concludes the proof. □

It turns out that ψ_m^{trig} leads to a high variance of the estimator $\widehat{\text{AttnHead}}_{\psi_{\text{trig}}}$ since most of the values of $\boldsymbol{A}$ are close to zero which is the region where sin is the most unstable. This is theoretically undermined by [Cho+21, Lemma 2], which states that the variance of $\psi_m^{\text{trig}}(\boldsymbol{q})^t\psi_m^{\text{trig}}(\boldsymbol{k})$ is of order $O(1/K(\boldsymbol{q},\boldsymbol{k})^2)$ for $\boldsymbol{q},\boldsymbol{k} \in B \subset \mathbb{R}^{d_{\text{attn}}}$ with bounded B and fixed m. Furthermore, having random feature kernels (strictly speaking, ψ^{trig} cannot be a random feature map since it can possibly be negative) that can take on negative values includes the risk of exacerbating numerical instabilities that can lead to absurd behaviour (e.g. entries of $\boldsymbol{D}$ being negative). Thus, Choromanski et al. propose a robust mechanism consisting of positive random features for SoftMax. The following lemma corresponds to Lemma 1 in [Cho+21].

Lemma 5.2. Let $\boldsymbol{q},\boldsymbol{k} \in \mathbb{R}^{d_{\text{attn}}}$ and denote $\boldsymbol{u} = \boldsymbol{q}+\boldsymbol{k}$. With independent $X_1,\dots,X_m \sim \mathcal{N}(0, \frac{1}{\sqrt{d_{\text{attn}}}} I_{d_{\text{attn}}})$ and the kernel K given in equation (5.1), we have that the random feature maps defined by

$$\psi_m^+(\boldsymbol{x}) := \frac{1}{\sqrt{m}}\exp\left(-\frac{\|\boldsymbol{x}\|^2}{2\sqrt{d_{\text{attn}}}}\right)\left(\exp\left(\frac{X_1^t\boldsymbol{x}}{\sqrt{d_{\text{attn}}}}\right),\dots,\exp\left(\frac{X_m\boldsymbol{x}}{\sqrt{d_{\text{attn}}}}\right)\right)$$

are in accordance with K.

Proof. We introduce a factor of one and compute

$$\begin{aligned}
\exp\left(\frac{\|\boldsymbol{q}+\boldsymbol{k}\|^2}{2\sqrt{d_{\text{attn}}}}\right) &= \exp\left(\frac{\|\boldsymbol{q}+\boldsymbol{k}\|^2}{2\sqrt{d_{\text{attn}}}}\right)(2\pi\sqrt{d_{\text{attn}}})^{-d_{\text{attn}}/2}\int_{\mathbb{R}^{d_{\text{attn}}}}\exp\left(-\frac{\|x-(\boldsymbol{q}+\boldsymbol{k})\|^2}{2\sqrt{d_{\text{attn}}}}\right)dx \\
&= (2\pi\sqrt{d_{\text{attn}}})^{-d_{\text{attn}}/2}\int_{\mathbb{R}^{d_{\text{attn}}}}\exp\left(-\frac{\|x\|^2}{2\sqrt{d_{\text{attn}}}}+\frac{x^t(\boldsymbol{q}+\boldsymbol{k})}{\sqrt{d_{\text{attn}}}}-\frac{\|\boldsymbol{q}+\boldsymbol{k}\|^2}{2\sqrt{d_{\text{attn}}}}+\frac{\|\boldsymbol{q}+\boldsymbol{k}\|^2}{2\sqrt{d_{\text{attn}}}}\right)dx \\
&= (2\pi\sqrt{d_{\text{attn}}})^{-d_{\text{attn}}/2}\int_{\mathbb{R}^{d_{\text{attn}}}}\exp\left(-\frac{\|x\|^2}{2\sqrt{d_{\text{attn}}}}+\frac{x^t(\boldsymbol{q}+\boldsymbol{k})}{\sqrt{d_{\text{attn}}}}\right)dx \\
&= (2\pi\sqrt{d_{\text{attn}}})^{-d_{\text{attn}}/2}\int_{\mathbb{R}^{d_{\text{attn}}}}\exp\left(-\frac{\|x\|^2}{2\sqrt{d_{\text{attn}}}}\right)\exp\left(\frac{x^t\boldsymbol{q}}{\sqrt{d_{\text{attn}}}}\right)\exp\left(\frac{x^t\boldsymbol{k}}{\sqrt{d_{\text{attn}}}}\right)dx \\
&= \mathbb{E}\left[\exp\left(\frac{X_j^t\boldsymbol{q}}{\sqrt{d_{\text{attn}}}}\right)\exp\left(\frac{X_j^t\boldsymbol{k}}{\sqrt{d_{\text{attn}}}}\right)\right].
\end{aligned}$$

for any $j \in \{1, \ldots, m\}$. The lemma now follows directly from this observation

$$\begin{aligned}
\mathbb{E}\left[\psi_m^+(\boldsymbol{q})^t \psi_m^+(\boldsymbol{k})\right] &= \frac{1}{m} \exp\left(-\frac{\|\boldsymbol{q}\|^2 + \|\boldsymbol{k}\|^2}{2\sqrt{d_{\text{attn}}}}\right) \mathbb{E}\left[\sum_{j=1}^{m} \exp\left(\frac{X_j^t \boldsymbol{q}}{\sqrt{d_{\text{attn}}}}\right) \exp\left(\frac{X_j^t \boldsymbol{k}}{\sqrt{d_{\text{attn}}}}\right)\right] \\
&= \exp\left(-\frac{\|\boldsymbol{q}\|^2 + \|\boldsymbol{k}\|^2}{2\sqrt{d_{\text{attn}}}}\right) \exp\left(\frac{\|\boldsymbol{q} + \boldsymbol{k}\|^2}{2\sqrt{d_{\text{attn}}}}\right) \\
&= \exp\left(\frac{\boldsymbol{q}^t \boldsymbol{k}}{\sqrt{d_{\text{attn}}}}\right) = K(\boldsymbol{q}, \boldsymbol{k})
\end{aligned}$$

We conclude the proof by observing that $\psi_m^+(x)(w) > 0$ for all $x \in \mathbb{R}^{d_{\text{attn}}}$, $\omega \in \Omega$. □

The following result shows that the kernel estimator arising from ψ_m^{trig} has a higher asymptotic variance than the one corresponding to ψ_m^+.

We refer the reader to [Cho+21, Lemma 2] for the purely computational proof.

Lemma 5.3. The following is true:

$$\begin{aligned}
\operatorname{Var}(\psi_m^{\text{trig}}(\boldsymbol{q})^t \psi_m^{\text{trig}}(\boldsymbol{k})) &= \frac{\exp\left(\frac{\|\boldsymbol{q}+\boldsymbol{k}\|^2}{\sqrt{d_{\text{attn}}}}\right)\left(1 - \exp\left(-\frac{\|\boldsymbol{q}-\boldsymbol{k}\|^2}{\sqrt{d_{\text{attn}}}}\right)\right)^2}{2m\, K(\boldsymbol{q}, \boldsymbol{k})^2} \\
\operatorname{Var}(\psi_m^+(\boldsymbol{q})^t \psi_m^+(\boldsymbol{k})) &= \frac{1}{m} \exp\left(\frac{\|\boldsymbol{q} + \boldsymbol{k}\|^2}{\sqrt{d_{\text{attn}}}}\right)\left(1 - \exp\left(-\frac{\|\boldsymbol{q} + \boldsymbol{k}\|^2}{\sqrt{d_{\text{attn}}}}\right)\right)^2 K(\boldsymbol{q}, \boldsymbol{k})^2
\end{aligned}$$

Lemma 5.3 shows that as $K(\boldsymbol{q}, \boldsymbol{k}) \to 0$, we have

$$\operatorname{Var}(\psi_m^{\text{trig}}(\boldsymbol{q})^t \psi_m^{\text{trig}}(\boldsymbol{k})) \to \infty, \quad \operatorname{Var}(\psi_m^+(\boldsymbol{q})^t \psi_m^+(\boldsymbol{k})) \to 0.$$

We can further reduce the variance by requiring the random variables $X_1, \ldots, X_m$ to be orthogonal. We denote the orthogonalised version of ψ_m^+ as $\psi_m^{\text{ort}+}$. Orthogonality can be achieved in practice by using the Gram-Schmidt orthogonalisation procedure. We do not spell out the proof, that $\psi_m^{\text{ort}+}$ is still in accordance with K, since it simply suffices to acknowledge, that for each $j \in \{1, \ldots, m\}$ we have $X_j \sim \mathcal{N}(0, \frac{I_{d_{\text{attn}}}}{\sqrt{d_{\text{attn}}}})$, because the multivariate normal distribution is isotropic. The independence was not necessary to show the unbiasedness of the estimator.

Theorem 2 in [Cho+21] asserts that this method can further reduce the variance of the estimator for K.

Lemma 5.4. For $0 < m \leq d_{\text{attn}}$, we have

$$\begin{aligned}
&\operatorname{Var}(\psi_m^{\text{ort}+}(\boldsymbol{q})^t \psi_m^{\text{ort}+}(\boldsymbol{k})) \\
&\qquad \leq \operatorname{Var}(\psi_m^+(\boldsymbol{q})^t \psi_m^+(\boldsymbol{k})) - \frac{2(m-1)}{m(d+2)}\left(K(\boldsymbol{q}, \boldsymbol{k}) - \exp\left(-\frac{\|\boldsymbol{q}\|^2 + \|\boldsymbol{k}\|^2}{2}\right)\right)^2.
\end{aligned}$$

5.2 Locality Sensitive Hashing Attention

We want to contrast the exposition of FAVOR+ with a sparse attention approximation method, namely *locality-sensitive hashing (LSH)* proposed in [KKL20] for the Reformer model.

The basic assumption behind LSH attention is sparsity of the attention matrix, i.e. that only few query-key pairs contribute to the SoftMax in the AttnHead operation. Hence, we can approximate $\mathrm{SoftMax}(\phi(\boldsymbol{q}_i, \boldsymbol{k}_{i'})_{i'=1}^N)$ by only considering the pairs $(\boldsymbol{q}_i, \boldsymbol{k}_{i'})$, for which $\phi(\boldsymbol{q}_i, \boldsymbol{k}_{i'})$ is large.

Since ϕ denotes a similarity kernel, we want to group "similar" query-key pairs together. This can be achieved with an appropriate locality-sensitive hashing method which groups similar (in a sense depending on the concrete LSH method) queries and keys together.

Since, we have $\phi(\boldsymbol{q}, \boldsymbol{k}) = \boldsymbol{q}^t\boldsymbol{k}/\sqrt{d_{\text{attn}}}$ for the standard attention, that we consider in this work, Kitaev, Kaiser, and Levskaya propose to use an LSH scheme that is based on angular distance [And+15]. We now explore the rough idea of this concept.

5.2.1 The Hash Function

As previously mentioned, we use the hash function to establish "closeness" between queries and keys. We can formulate simple requirements for the hash function such that we can control the number of possible hash buckets.

It is beyond the scope of this work to formally introduce the concrete LSH function (practical cross-polytope LSH) and the associated theoretical results. Rather, we want to present the rough idea. Andoni et al. solve a version of the *Approximate Near Neighbour Problem* in [And+15]. To this end, they propose a hash family called *cross-polytope LSH*. Cross-polytope LSH works in dimension $d \in \mathbb{N}$ by hashing a point $x \in \mathbb{R}^d$ by computing

$$\underset{\{\pm e_i\}_{1\leq i\leq d}}{\arg\inf} f_x \qquad \text{where } f_x(y) := \left\| \frac{y}{\|y\|} - \frac{Ax}{\|Ax\|} \right\|,$$

where $\{e_i\}_{1\leq i\leq d}$ denotes the euclidian basis of $\mathbb{R}^d$ and $A \in \mathbb{R}^{d\times d}$ is a random matrix with Gaussian entries.

It turns out that the computation of Ax is of time-complexity $O(d^2)$, which is too expensive for practical applications (we want to go below square complexity). To solve this problem Andoni et al. propose to use feature hashing which consists of computing the cross-polytope hash not in $\mathbb{R}^d$ as described above but in $\mathbb{R}^{d'}$. This can be achieved by considering not $A \in \mathbb{R}^{d\times d}$, but $\tilde{A} \in \mathbb{R}^{d'\times d}$ with $d' \ll d$. The resulting number of possible hashes is $2d'$ in this case.

5.2.2 Attention Approximation

Recall that

$$\mathrm{AttnHead}(\mathcal{Q}, \mathcal{K}, \mathcal{V})_i = \sum_{j=1}^{N} \mathrm{SoftMax}\left(\phi\left(\boldsymbol{q}_i, \boldsymbol{k}_{i'}\right)_{i'\in\{1,\dots,N\}}\right)_j \boldsymbol{v}_j.$$

Let $P_i^{\text{standard}} = \{1, \ldots, N\}$ denote the set of keys that each query pays attention to. We want to find smaller $\tilde{P}_i$ for each $i \in \{1, \ldots, N\}$ such that

$$\widehat{\text{AttnHead}}(\mathcal{Q}, \mathcal{K}, \mathcal{V})_i = \sum_{j \in \tilde{P}_i} \text{SoftMax}\left(\phi\left(\boldsymbol{q}_i, \boldsymbol{k}_{i'}\right)_{i' \in \tilde{P}_i}\right)_j \boldsymbol{v}_j.$$

To find suitable sets $(\tilde{P}_i)_{i \in \{1,\ldots,N\}}$, we carry out the construction in several steps.

Step 1: Given the hash function h we can define

$$P_i^{(1)} := \{j \in \{1, \ldots, N\} \, : \, h(\boldsymbol{q}_i) = h(\boldsymbol{k}_j)\}.$$

The sets $P_i^{(1)}$ encode the idea that a query should only pay attention to keys that are similar in the sense of the hash function.

Step 2: We want to have an equal number of keys and queries in each bucket so that we can easily compute full attention within each bucket. One way to achieve this is to identify keys with queries and thus modify the attention mechanism slightly. This equates to

$$P_i^{(2)} := \{j \in \{1, \ldots, N\} \, : \, h(\boldsymbol{q}_i) = h\left(\boldsymbol{q}_j\right)\}.$$

In other words we now have that

$$h(\boldsymbol{q_i}) = h(\boldsymbol{q_j}) \qquad \Leftrightarrow \qquad P_i^{(2)} = P_j^{(2)}$$

compared to as before. We remark that the resulting operation is not equivalent to standard self-attention. We restrict the flexibility of the attention mechanism by not using the key embedding $\text{Lin}_{d \to d_{\text{attn}}}^{(\mathcal{K})}$ for the keys, but instead using the query embedding $\text{Lin}_{d \to d_{\text{attn}}}^{\mathcal{Q}}$ We can also phrase this differently by just setting $\boldsymbol{k}_i := \boldsymbol{q}_i$.

Step 3: We repeat the process from the previous steps several times to reduce the likelihood of similar queries falling into different buckets due to artefacts in the hashing function. To do this, let $n_{\text{rounds}} \in \mathbb{N}$ be the number of iterations and $h^{(1)}, \ldots, h^{(n_{\text{rounds}})}$ be different hash functions. This lets us define

$$P_i^{(3)} := \bigcup_{r=1}^{n_{\text{rounds}}} P_i^{(2,r)}$$

where $P_i^{(2,r)}$ denotes $P_i^{(2)}$ with hash function $h^{(r)}$, i.e.

$$P_i^{(2,r)} := \left\{j \in \{1, \ldots, N\} \, : \, h^{(r)}(\boldsymbol{q}_i) = h^{(r)}\left(\boldsymbol{q}_j\right)\right\}.$$

Step 4: The sets $P_i^{(3)}$ can still be of vastly different size for different indices $i \in \{1, \ldots, N\}$. Hence, we rebalance into equally sized buckets of size $m = N / n_{\text{buckets}}$. An easy way to achieve such a rebalancing is to sort all indices by their first appearance in a bucket. More formally, let $\tilde{P}_1^{(3)} := P_1^{(3)}$ and iteratively $\tilde{P}_i^{(3)} = P_i^{(3)} \setminus \tilde{P}_{i-1}^{(3)}$ for $i \in \{2, \ldots, N\}$. Let $(t_j^{(i)})_{j \in \{1,\ldots,|\tilde{P}_i^{(3)}|\}}$ be an enumeration for each $\tilde{P}_i^{(3)}$, i.e.

$$\tilde{P}_i^{(3)} := \{\boldsymbol{q}_{t_1^{(i)}}, \ldots, \boldsymbol{q}_{t_{|\tilde{P}_i^{(3)}|}^{(i)}}\}.$$

and denote $p_i := |\tilde{P}_i^{(3)}|$. Then, we can define the permutation $(s(j))_{j\in\{1,\dots,N\}}$ by

$$s(j) = t^{(n_j+1)}_{j-\sum_{l=1}^{n_j} p_l} \qquad n_j = \sup\left\{ n \in \{0,1,\dots,N-1\} \,:\, \sum_{l=1}^{n} p_l < j \right\}.$$

We now essentially reorder the queries $(\boldsymbol{q}_j)_{j\in\{1,\dots,N\}}$ with the permutation $j \mapsto s(j)$ to get $(\boldsymbol{q}_{s(j)})_{j\in\{1,\dots,N\}}$. Here, the queries are sorted by their first appearance in a bucket and the buckets' queries appear as subsequences. We now cut into new buckets of size m with This leads us to consider

$$\tilde{P}_i := s^{-1}\left(\left\{ \left\lfloor \frac{i-1}{m} \right\rfloor + 1, \dots, \left\lceil \frac{i}{m} \right\rceil \right\} \right)$$

which defines buckets of equal size (up to boundary effects).

5.3 Experiments

Our goal is to experimentally compare LSH attention and the FAVOR+ mechanism to efficiently compute attention along the variate-axis. In this section, we first introduce the experimental setting and the datasets we use for this comparison. We then present the results and dive into the analysis. Our general hypothesis is that higher sparsity in the data favours the LSH mechanism, while the FAVOR+ mechanism performs better in a low-sparsity setting.

The general experimental structure is as follows: We construct data with different degrees of sparsity (in an appropriate sense). We then fit a standard FlexibleTransformer model that uses self-attention along the variate-axis to these data. We briefly present one measure that allows us to measure sparsity in the attention score matrix and another measure that we can use directly on the data. We then fit the respective models with the standard self-attention mechanism along the variate-axis replaced by the LSH and FAVOR+ mechanisms. This allows us to test the hypothesis formulated above.

5.3.1 Measures for Sparsity

We introduce a measure that can be computed directly from the data and one measure that requires a trained standard self-attention along the variate-axis.

Correlation Sparsity

This model-independent measure is quite simple. The idea is that sparsity is characterised by a sparse correlation matrix of the variates for a time-series $X \in R^{N_{\text{in}}\times L_{\text{in}}}$. Let $C \in \mathbb{R}^{N_{\text{in}}\times N_{\text{in}}}$ be the correlation matrix of the variates computed over the entire time-series X. We normalise the correlation matrix with the l^1 norm to $\mathbb{R}^{N_{\text{in}}\times N_{\text{in}}}$

$$\tilde{C}_{ij} := \frac{|C_{ij}|}{\|C\|_1}$$

so that $\tilde{C}$ corresponds to a probability distribution μ_C on $\{1,\dots,N_{\text{in}}\}^2$ and $\|\cdot\|_1$ is the l^1 norm. This allows us to compute sparsity as the Kullback-Leibler divergence between μ_C and $\mu_{\mathcal{U}} := \mathcal{U}(\{1,\dots,N_{\text{in}}\}^2)$:

$$\text{CorrSparsity}(X) := \text{KL}(\mu_C \,\|\, \mu_{\mathcal{U}}) = \sum_{i,j=1}^{N_{\text{in}}} \mu_C(\{(i,j)\}) \log\left(\frac{\mu_C(\{(i,j)\})}{\mu_{\mathcal{U}}(\{(i,j)\})} \right)$$

Attention Entropy

Suppose we are given a concrete FlexibleTransformer configuration. We will usually want to consider the configuration in which we want to replace the standard self-attention, which mixes along the variate-axis, with one of the efficient self-attention approximation schemes. For this measure, however, we consider the full standard self-attention because we need to access the attention score matrix. Let $\boldsymbol{A} \in \mathbb{R}^{N_{\text{in}} \times N_{\text{in}}}$ be the attention score matrix for some concrete input, i.e.

$$\boldsymbol{A}_{ij} = \text{SoftMax}\left(\phi(\boldsymbol{q}_i, \boldsymbol{k}_{i'})_{i'=1,\dots,N_{\text{in}}}\right)_j .$$

It is immediately clear that $\boldsymbol{A}$ is a stochastic matrix in the sense that $\sum_{j=1}^{N_{\text{in}}} \boldsymbol{A}_{ij} = 1$ for all $i \in \{1, \dots, N_{\text{in}}\}$. Therefore, we can get a measure of the sparsity of this attention score matrix by considering the mean of the entropy of each row, i.e.

$$\text{AttnEntropy}(\boldsymbol{A}) := -\frac{1}{N_{\text{in}}} \sum_{i=1}^{N_{\text{in}}} \sum_{j=1}^{N_{\text{in}}} A_{ij} \log(A_{ij}).$$

We will see concrete examples of these metrics when we introduce the datasets for the experiments in this chapter.

5.3.2 Setup

We want to control the sparsity of our data. Therefore, we will continue to use the datasets we constructed in section 4.3.1. More concretely, we want to work in a higher dimensional setting (in which it is still feasible to compute standard self-attention to compute AttnEntropy). So our concrete choices are

$$\begin{aligned} N_{\text{in}} &\in \{64, 128, 256\} \\ B &\in \{4, 8, 16, 32, 64\} \\ \gamma &= 0.3. \end{aligned}$$

Figure 5.1 shows the absolute correlation for the datasets defined above and Table 5.1 shows the sparsity metrics we introduced above.

Since in chapter 3 we found the choice of the MLP token-mixer to be promising in conjunction with self-attention along the variate-axis, we use this configuration of the FlexibleTransformer to test the efficient self-attention mechanisms. Since we are not investigating token-mixing along the time-axis, we simply fix $L_{\text{in}} = T = 96$. We know the appropriate hyperparameters for the FlexibleTransformer from previous experiments and fix them as

$$\begin{aligned} d &= 64 \\ d_{\text{mixing}} &= 64 \\ d_{\text{hidden}} &= 256 \\ n_{\text{layers}} &= 1 \end{aligned}$$

For the LSH mechanism, we use the values $n_{\text{rounds}} = 4$ and the bucket size $m = 4$ from the original paper [KKL20]. The standard choice for the number of random features in FAVOR+ is $d_{\text{attn}} \log(d_{\text{attn}})$. We run the experiments with the following training hyperparameters:

$$\begin{aligned} \text{dropout} &= 0.1 \\ \text{learning rate} &= 10^{-4} \\ \text{learning rate scheduler} &= \text{constant} \end{aligned}$$

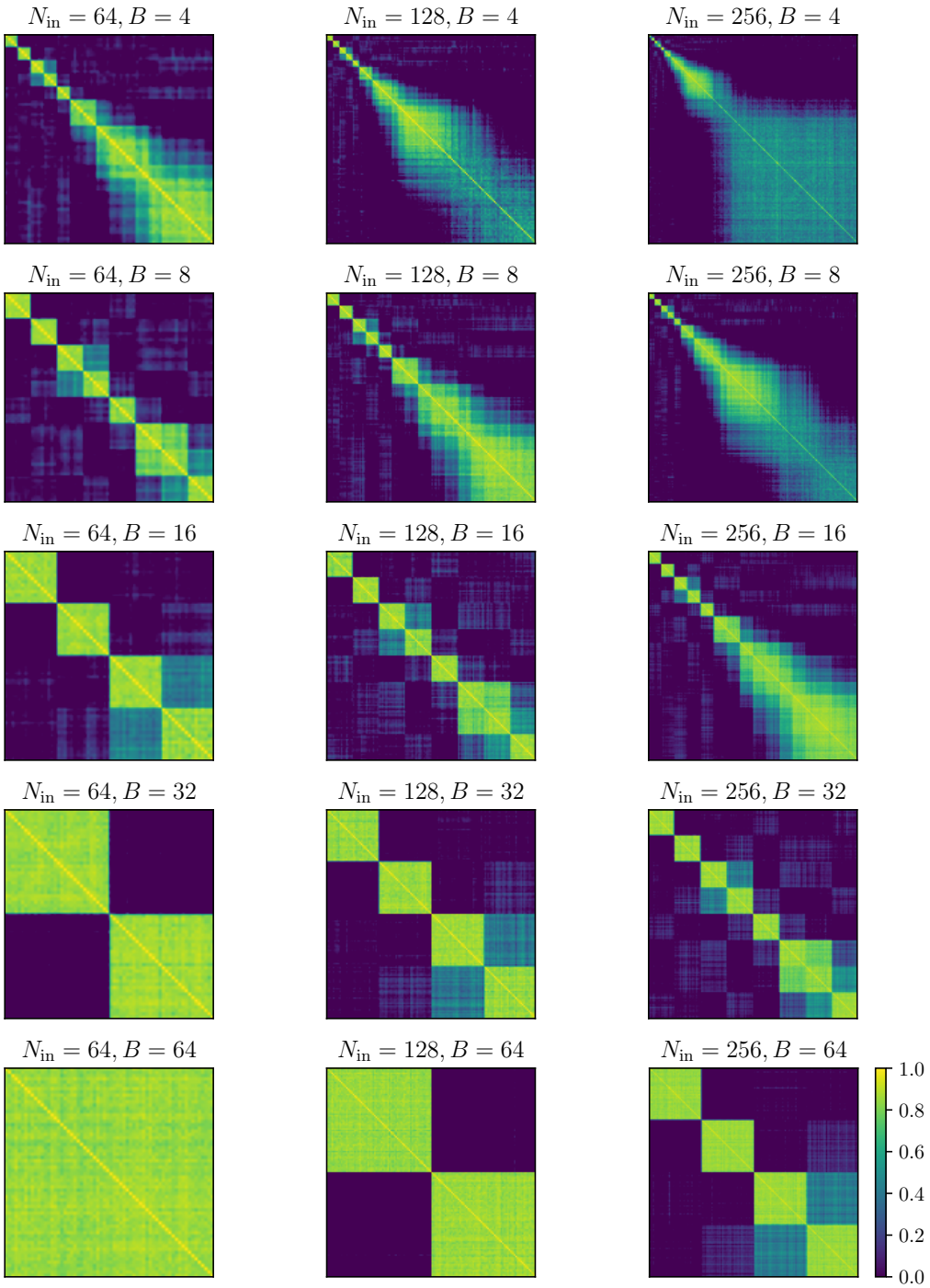

Figure 5.1: Absolute correlation between variates for different choices of N_{in} and B. Note that we get artifacts as we increase N_{in}/B which is due to having finite L_{in}.

N_{in}	B	AttnEntropy	CorrelationSparsity
64	4	2.482	0.499
	8	2.347	0.538
	16	2.838	0.499
	32	3.577	0.227
	64	4.145	0.001
128	4	3.740	0.301
	8	2.908	0.503
	16	3.239	0.542
	32	3.488	0.488
	64	4.228	0.459
256	4	4.986	0.187
	8	4.288	0.297
	16	4.435	0.488
	32	3.771	0.534
	64	4.228	0.459

Table 5.1: Measures for sparsity evaluated on the data that we use in this chapter's experiments. Note that the measures for sparsity are non-monotonic due to the artifacts in the data as observed in Figure 5.1. This is not a problem since we only need that different datasets have different properties that we measure with the attention entropy and the correlation sparsity.

5.3.3 Results and Analysis

We present the results in Table 5.2 and also visualise the relative performance of LSH over FAVOR+ in Figure 5.2.

First of all, we note that both methods perform as well as the standard self-attention. We do observe the expected behaviour that models with larger B perform (ever so slightly) better because more examples from the same period are available.

Contrary to our hypothesis that LSH performs better in sparse settings, we do not observe a difference in the performance of the models in terms of mean squared error. This may however be mostly due to the fact that both methods perform as well as the full self-attention and can hence be seen as perfectly sufficient to approximate self-attention in these settings. In all cases, FAVOR+ performs slightly better.

A reason for the above could be the following: We might have chosen N_{in} too small in our experiments as we are able to approximate (in the sense of final model performance) self-attention in both settings. By increasing N_{in}, we lose the control of comparing with full self-attention as we are simply not able to fit models including full self-attention in very big settings.

	Mechanism	FAVOR+			LSH			Standard Attention		
		MAE	MMaxError	MSE	MAE	MMaxError	MSE	MAE	MMaxError	MSE
N_{in}	B									
64	4	0.320	1.092	0.161	0.323	1.101	0.163	0.323	1.101	0.163
	8	0.322	1.101	0.163	0.323	1.104	0.163	0.323	1.104	0.164
	16	0.325	1.108	0.165	0.322	1.098	0.162	0.322	1.098	0.162
	32	0.320	1.096	0.160	0.321	1.100	0.162	0.319	1.095	0.160
	64	0.319	1.093	0.160	0.318	1.088	0.159	0.318	1.087	0.158
128	4	0.319	1.093	0.160	0.321	1.099	0.162	0.320	1.096	0.161
	8	0.319	1.094	0.160	0.322	1.102	0.163	0.322	1.102	0.163
	16	0.320	1.095	0.161	0.322	1.101	0.163	0.322	1.100	0.163
	32	0.323	1.105	0.164	0.321	1.099	0.162	0.321	1.099	0.162
	64	0.319	1.090	0.159	0.321	1.096	0.161	0.319	1.091	0.160
256	4	0.321	1.099	0.162	0.323	1.104	0.164	0.325	1.109	0.165
	8	0.320	1.098	0.161	0.322	1.103	0.163	0.320	1.097	0.161
	16	0.320	1.096	0.161	0.322	1.101	0.163	0.322	1.100	0.163
	32	0.322	1.102	0.162	0.322	1.103	0.163	0.322	1.103	0.163
	64	0.319	1.090	0.159	0.321	1.096	0.161	0.319	1.091	0.160

Table 5.2: Test errors for LSH and FAVOR+ on all datasets.

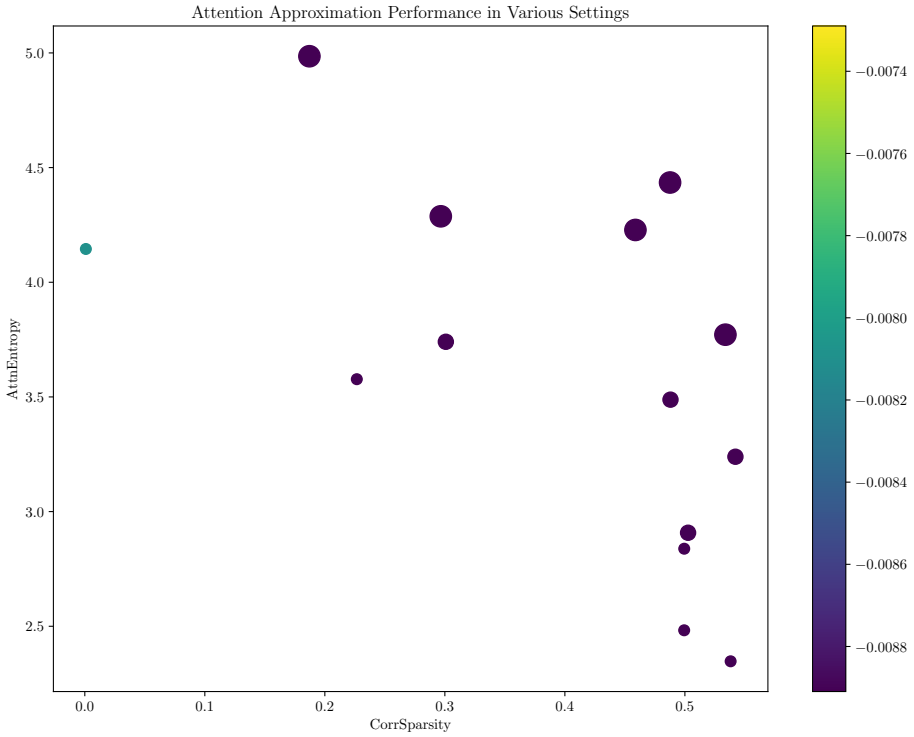

Figure 5.2: Relative advantage of using LSH over FAVOR+ for all datasets. We plot the sparsity measures on the axes, the size of the circle indicates B and the color is the mean squared error for the respective LSH-based model over the mean squared error for the FAVOR+-based model minus one.

	Mechanism	Data Properties	FAVOR+			LSH		
		CorrSparsity	MAE	MMaxError	MSE	MAE	MMaxError	MSE
N_{in}	B							
1024	4	0.279	0.323	1.103	0.165	0.323	1.102	0.165
	128	0.694	0.380	1.301	0.266	0.379	1.298	0.265
	512	0.545	0.559	1.916	0.581	0.559	1.917	0.581
	1024	0.287	0.798	2.731	0.999	0.798	2.731	0.999

Table 5.3: Experiment results for increased number of variates N_{in}.

Modifications of the Experiment

At this stage, we cannot draw any further conclusions from this experiment alone. Therefore, we want to extend the previous setting and replace the data we use to create this experiment. Note that increasing N_{in} prevents us from calculating the AttnEntropy measure and the full self-attention model as a baseline. Nevertheless, we suggest repeating the above experiment with data where N_{in} is larger.

Increase N_{in}

We repeat the previous experiment, but with synthetic data that now has significantly more variates. More concretely, we choose

$$\begin{aligned} N_{\text{in}} &= 1024 \\ B &\in \{4, 128, 512, 1024\}. \end{aligned}$$

We do not evaluate the attention entropy metric as we are not able to train models with standard self-attention along the variate-axis for this value of N_{in}. Hence, we only show the correlation sparsity in Table 5.3. As we can see, both models perform approximately equally well. We cannot observe a difference between the models' performances. The only relationship that we can observe is between B and the model performance, which differentiates this large setting from the previous experiment, where we did not observe such a positive correlation between B and MSE, MAE and MMaxError. We can now observe that FAVOR+ as well as LSH both fail to learn anything in the case where $B = 1024$ and perform better in high sparsity settings.

Conclusion

We cannot find a difference in performance for the two efficient attention approximation mechanisms LSH and FAVOR+ and observe that they are both able to approximate full self-attention in modestly large settings. Both seem to be equally well suited for the artificial tasks that we have tested them in.

However, while we still observe the same performance by both FAVOR+ and LSH, their performance degrades in large settings where we have many highly correlated variates.

From a theoretical standpoint, it seems to be interesting to further investigate general properties of LSH and FAVOR+ especially in settings where the input tokens are semantically very similar (our low sparsity case).

It would also be interesting to experimentally study self-attention in well-controlled, general settings (not necessarily in the context of time-series) with differing degrees of semantic similarity between the tokens that are fed into self-attention.

From the standpoint of time-series forecasting, we do not think, that both of these points are particularly urgent, as we do have working efficient attention approximation schemes that work well in modelstly large and high sparsity settings.

Chapter 6

Conclusion

We want to conclude this thesis by highlighting the main findings and giving an outlook on future avenues of research within the framework introduced in this work.

After having introduced the vanilla Transformer and having given an overview of current developments within the literature applying Transformers to time-series, we have identified the need to systematise and unify the analysis of various model architectures including PatchTST [Nie+23], iTransformer [Liu+24] and TSMixer [Che+23]. This has led us to introduce and study the FlexibleTransformer which is a model that allows us to flexibly generalise many of the previous models and even study yet unexplored architectures.

In thorough experiments, we have explored a wide range of possible configurations of the FlexibleTransformer, conducted a hyperparameter study and investigated the trade-off between model size and performance. We have identified architectures mainly mixing along the time-axis as promising.

To better understand the key moving parts in the FlexibleTransformer architecture, we modified the contextualisation approach from [Kob+24; Kob+21] to fit the time-series setting. Extending the previous results to a cumulative decomposition allows us to provide interpretable time-series models. Furthermore, we analysed the contextualisation of different model components. We could not confirm the hypothesis that a stronger contextualisation is associated with better model performance. Future research that analyses individual model components on a more granular basis, by for example boosting or restraining individual components can maybe shed more light on this relationship.

As modern applications oftentimes require models to deal with hundreds to thousands of variates, the question for efficient attention mechanisms along the variate-axis arises. We analyse how two popular general-purpose attention approximation schemes perform in various settings. We have experimentally observed that both of the considered attention approximation schemes have performed equally well and we did not find a relationship between the inductive bias of the attention approximation scheme and constructed metrics measuring properties of the data aimed at identifying these inductive biases.

Outlook

This work explores new concepts and many questions have arisen. We think that we have provided a thorough analysis of the FlexibleTransformer in classical terms. Since the application of self-attention along the variate-axis did not emerge as strikingly powerful and since we have identified efficient self-attention approximation schemes that work well along the variate-axis, we do not identify the further improvement of efficient attention along the variate-axis as particularly urgent.

The contextualisation approach to Transformer-based time-series models appears to be the most promising idea from this work and should be explored further. In particular, we identify the following two questions that should be explored in future research:

- A more fine-grained analysis of the relationship between contextualisation and model performance. In particular, can we modify individual components that have been identified with the contextualisation approach discussed in this work and observe a performance gain by boosting this component such that it can contextualise even more?

- More efficient feature attribution methods for the MLP-based model components. As mentioned in chapter 4, we had to restrict the experiments in size due to very long runtimes. To allow for larger contextualisation studies and real-world model interpretability, more efficient contextualisation approaches are needed.

Appendix A

Appendix

A.1 Notation for Application of Model-Components

We establish the convention that model components written in **bold** letters are vectorised versions, where the component is applied along the first level index set, i.e.

$$\mathbf{M}\left((\boldsymbol{x}_i)_{i\in\mathcal{I}}\right) := (\mathrm{M}(\boldsymbol{x}_i))_{i\in\mathcal{I}}\,,$$

where

$$\begin{aligned}\mathrm{M} &: \mathcal{X} \to \mathcal{X}\\ \mathbf{M} &: \mathcal{X}^{\mathcal{I}} \to \mathcal{X}^{\mathcal{I}}\end{aligned}$$

and $\boldsymbol{x}_i \in \mathcal{X}$ for $i \in \mathcal{I}$.

Oftentimes, we consider objects $X = (\boldsymbol{x}_i)_{i\in\mathcal{I}} \in \mathcal{X}^{\mathcal{I}}$, where $\mathcal{I} = I_1 \times I_2 \times \cdots \times I_k$. We want to extract one dimension of this index set with the idea of applying M on a slice of X. Hence, we introduce the following notation

$$\begin{aligned}&X_{I_1\times\cdots\times I_{l-1}\times\underline{I_l}\times I_{l+1}\times\cdots I_k}\\ &:= \Big(\big(X_{(i_1,\ldots,i_{l-1},i_l,i_{l+1},\ldots,i_k)}\big)_{i_l\in I_l}\Big)_{(i_1,\ldots,i_{l-1},i_{l+1},\ldots,i_k)\in I_1\times\cdots\times I_{l-1}\times I_{l+1}\times\cdots\times I_k}\\ &\in \big(\mathcal{X}^{I_l}\big)^{I_1\times\cdots\times I_{l-1}\times I_{l+1}\times\cdots\times I_k}.\end{aligned} \tag{A.1}$$

that marks the dimension of interest with an underline. In the case, where $I_j = \{1,\ldots,a_j\}$ with $a_j \in \mathbb{N}$ for $j \in \{1,\ldots,k\}$, we may equivalently write $X_{a_1\times\cdots\times a_{l-1}\times\underline{a_l}\times a_{l+1}\times\cdots\times a_k}$ for the lefthandside in equation (A.1).

We also want to be able to reverse this operation and hence introduce the following notation for an index set of the form $\mathcal{I} = A \times B$

$$\bigvee_{\underline{A}\times B} X := (\boldsymbol{x}_{a,b})_{(a,b)\in A\times B} \quad \text{for } X = \big((\boldsymbol{x}_{a,b})_{a\in A}\big)_{b\in B} \in \big(\mathcal{X}^A\big)^B$$

$$\text{and} \quad \bigvee_{A\times\underline{B}} X := (\boldsymbol{x}_{a,b})_{(a,b)\in A\times B} \quad \text{for } X = \big((\boldsymbol{x}_{a,b})_{b\in B}\big)_{a\in A} \in \big(\mathcal{X}^B\big)^A.$$

A.2 Common Components of Neural Nets

We use the term for "learnable weights" when we refer to weights that are subject to optimisation during the model learning stage. The components introduced in this section can be considered standard. As we, however, pay attention to formality, we want to define them nevertheless.

Linear Layer

The probably most basic building block is a linear layer.

Definition A.1. The *linear layer* $\mathrm{Lin}_{A\to B} : \mathbb{R}^A \to \mathbb{R}^B$ has learnable weights $W \in \mathbb{R}^{B\times A}$, $b \in \mathbb{R}^B$ and is given by

$$\mathrm{Lin}_{A\to B}(\boldsymbol{x}) = W\boldsymbol{x} + b$$

for $x \in \mathbb{R}^A$.

Multilayer Perceptron

The multilayer perceptron (MLP) is also often called "feed-forward layer" in the literature. We, however, think that MLP is more descriptive compared to feed-forward layer.

Definition A.2. The *multilayer perceptron (MLP)* with H hidden layers, input layer size d_{in}, hidden layer size d_{hidden}, output layer size d_{out} and activation function $g : \mathbb{R} \to \mathbb{R}$ is a function

$$\mathrm{MLP}_{d_{\text{in}}\to H\times d_{\text{hidden}}\to d_{\text{out}},g} : \mathbb{R}^{d_{\text{in}}} \to \mathbb{R}^{d_{\text{out}}}$$

and is defined as

$$\begin{aligned}
\boldsymbol{x}^{(0)} &:= \boldsymbol{g}(W^{(0)}\boldsymbol{x} + b^{(0)}) \\
\boldsymbol{x}^{(h)} &:= \boldsymbol{g}\left(W^{(h)}\boldsymbol{x}^{(h-1)} + b^{(h)}\right) \qquad \text{for } h = 1, \dots, H-1 \\
\mathrm{MLP}_{d_{\text{in}}\to H\times d_{\text{hidden}}\to d_{\text{out}},g}(\boldsymbol{x}) &:= W^{(H)}\boldsymbol{x}^{(H-1)} + b^{(H)}
\end{aligned}$$

for $\boldsymbol{x} \in \mathbb{R}^{d_{\text{in}}}$, where $W^{(0)} \in \mathbb{R}^{d_{\text{hidden}}\times d_{\text{in}}}$, $b^{(0)} \in \mathbb{R}^{d_{\text{hidden}}}$, $W^{(h)} \in \mathbb{R}^{d_{\text{hidden}}\times d_{\text{hidden}}}$, $b^{(h)} \in \mathbb{R}^{d_{\text{hidden}}}$ for $h = 1, \dots, H-1$ and $W^{(H)} \in \mathbb{R}^{d_{\text{out}}\times d_{\text{hidden}}}$, $b^{(H)} \in \mathbb{R}^{d_{\text{out}}}$ are learnable weights. The function $\boldsymbol{g} : \mathbb{R}^{d_{\text{hidden}}} \to \mathbb{R}^{d_{\text{hidden}}}$ is the component-wise application of g.

A.3 Flatten and Reshape

The operations of flattening and reshaping a vector are not very important in the grander scheme of things, but we want to define them nevertheless for formal completeness. The concrete definitions of these operations are fairly similar to popular implementations in most well-known deep learning libraries.

Definition A.3. Let $\mathcal{J} \subset \mathcal{I}$, where $\mathcal{I}$ is the index set of a token space. Let $\mathcal{J} = \{j_1, \dots, j_J\}$ be an enumeration of $\mathcal{J}$ and let $X = (\boldsymbol{x}_{j_i})_{i\in\{1,\dots,J\}} \in \left(\mathbb{R}^d\right)^{\mathcal{J}}$. The flattening operation maps to $\mathbb{R}^{d\,|\mathcal{J}|}$ and is concretely given by

$$\mathrm{Flatten}_{\mathcal{J}}(X)_p = (\boldsymbol{x}_{j_i})_k\,,$$

where $p = id + k$ and $k \in \{0, \dots, d-1\}$.

We can analoguously define the reshaping operation:

Definition A.4. Let $y \in \mathbb{R}^{d_1 \times \cdots \times d_K \times d}$, $d = \prod_{i=K+1}^{N} d_i$ and $0 \leq K \leq N$. Then, we define

$$\text{Reshape}_{(d_1,\ldots,d_N)}(y)_{i_1,\ldots,i_N} := y_{i_1,i_2,\ldots,i_K,i_{K+1}d_{K+2}\cdots d_N + i_{K+2}d_{K+3}\cdots d_N + \cdots + i_N}$$

where $i_j \in \{1, \ldots, d_j\}$ for $j \in \{1, \ldots, N\}$.

A.4 Spearman Rank Correlation Coefficient

The Spearman rank correlation coefficient is a non-parametric measure of rank correlation. Let $n \in \mathbb{N}$ and $\boldsymbol{X} := (x_i)_{i\in\{1,\ldots,n\}}$ and $\boldsymbol{Y} := (y_i)_{i\in\{1,\ldots,n\}}$, whereas $x_i, y_i \in \mathbb{R}$ for $i \in \{1, \ldots, n\}$. Denote $R : \mathbb{R}^n \to \{1, \ldots, n\}^n$ a rank function, i.e. $i \mapsto R(z)_i$ is a bijection on $\{1, \ldots, n\}$ and $z_{R(z)_k} \leq z_{R(z)_l}$ for $1 \leq l < k \leq n$ for $z \in \mathbb{R}^n$. Then

$$\rho_S(\boldsymbol{X}, \boldsymbol{Y}) := \frac{\sum_{i=1}^{n} \left(R(x_i) - \frac{1}{n}\sum_{j=1}^{n} R(x_j)\right)\left(R(y_i) - \frac{1}{n}\sum_{j=1}^{n} R(y_j)\right)}{\left(\sum_{i=1}^{n} \left(R(x_i) - \frac{1}{n}\sum_{j=1}^{n} R(x_j)\right)^2\right)^{\frac{1}{2}} \left(\sum_{i=1}^{n} \left(R(y_i) - \frac{1}{n}\sum_{j=1}^{n} R(y_j)\right)^2\right)^{\frac{1}{2}}}$$

defines the spearman rank correlation coefficient ρ_S. We have $\rho_S \in [-1, 1]$, whereas ρ_S being close to one hints at a positive monotonic relation between $\boldsymbol{X}$ and $\boldsymbol{Y}$ and ρ_S being close to minus one at a negative monotonic relation.

A.5 Further Decompositions

We decompose the operations ResConn and LayerNorm. We fix the token space $\mathcal{T} = \mathcal{X}^{\mathcal{I}}$ with arbitrary $\mathcal{I}$ and $\mathcal{X} = \mathbb{R}^d$ and write $X \in \mathbb{R}^{N_{\text{in}} \times L_{\text{in}}}$ for the input time-series.

Lemma A.1. Let $\tilde{M} : \mathbb{R}^{N_{\text{in}} \times L_{\text{in}}} \to \mathcal{T}$ be the model part before the application of the LayerNorm component. Let $(\boldsymbol{F}^{\tilde{M}}(X), \boldsymbol{b}^{\tilde{M}}(X))$ be the decomposition up until right before the LayerNorm component with weights $\gamma \in \mathbb{R}^d$ and $\beta \in \mathbb{R}^d$. The decomposition of $\text{LayerNorm} \circ \tilde{M}$ is given by

$$F_{n,l}^{\text{LayerNorm}\circ\tilde{M}}(X, i) = \gamma \odot \frac{1}{\hat{\sigma}(\tilde{M}(X)_i)} \left(F_{n,l}^{\tilde{M}}(X, i) - \frac{1}{d}\sum_{k=1}^{d} F_{n,l}^{\tilde{M}}(X, i)_k\right)$$

$$b^{\text{LayerNorm}\circ\tilde{M}}(X, i) = \gamma \odot \frac{1}{\hat{\sigma}(\tilde{M}(X)_i)} \left(b^{\tilde{M}}(X, i) - \frac{1}{d}\sum_{k=1}^{d} b^{\tilde{M}}(X, i)_k\right) + \beta$$

Proof. We compute for $i \in \mathcal{I}$

$$\begin{aligned}
\text{LayerNorm} \circ \tilde{M}(X)_i &= \gamma \odot \frac{\tilde{M}(X)_i - \hat{\mu}(\tilde{M}(X)_i)}{\hat{\sigma}(\tilde{M}(X)_i)} + \beta \\
&= \gamma \odot \frac{1}{\hat{\sigma}(\tilde{M}(X)_i)} \left(\sum_{\substack{n \in \{1,\dots,N_{\text{in}}\} \\ l \in \{1,\dots,L_{\text{in}}\}}} F_{n,l}^{\tilde{M}}(X,i) + b^{\tilde{M}}(X,i) \right. \\
&\qquad \left. - \frac{1}{d} \sum_{k=1}^{d} \left(\sum_{\substack{n \in \{1,\dots,N_{\text{in}}\} \\ l \in \{1,\dots,L_{\text{in}}\}}} F_{n,l}^{\tilde{M}}(X,i)_k + b^{\tilde{M}}(X,i)_k \right) \right) + \beta \\
&= \sum_{\substack{n \in \{1,\dots,N_{\text{in}}\} \\ l \in \{1,\dots,L_{\text{in}}\}}} \gamma \odot \frac{1}{\hat{\sigma}(\tilde{M}(X)_i)} \left(F_{n,l}^{\tilde{M}}(X,i) - \frac{1}{d} \sum_{k=1}^{d} F_{n,l}^{\tilde{M}}(X,i)_k \right) \\
&\qquad + \gamma \odot \frac{1}{\hat{\sigma}(\tilde{M}(X)_i)} \left(b^{\tilde{M}}(X,i) - \frac{1}{d} \sum_{k=1}^{d} b^{\tilde{M}}(X,i)_k \right) + \beta.
\end{aligned}$$

This concludes the proof. □

Lemma A.2. Let $\tilde{M} : \mathbb{R}^{N_{\text{in}} \times L_{\text{in}}} \to \mathcal{T}$ be the model part before the application of the ResConn_S component with $S : \mathcal{T} \to \mathcal{T}$. Let $(\boldsymbol{F}^{\tilde{M}}(X), \boldsymbol{b}^{\tilde{M}}(X))$ be the decomposition up until right before the ResConn and $(\boldsymbol{F}^{S \circ \tilde{M}}(X), \boldsymbol{b}^{S \circ \tilde{M}}(X)$ be decomposition of $S \circ \tilde{M}$. The decomposition of $\text{ResConn}_S \circ \tilde{M}$ is given by

$$\begin{aligned}
F_{n,l}^{\text{ResConn}_S \circ \tilde{M}}(X,i) &= F_{n,l}^{S \circ \tilde{M}}(X,i) + F^{\tilde{M}}(X,i) \\
b^{\text{ResConn}_S \circ \tilde{M}}(X,i) &= b^{S \circ \tilde{M}}(X,i) + b^{\tilde{M}}(X,i)
\end{aligned}$$

Proof. We compute

$$\begin{aligned}
\text{ResConn}_S \circ \tilde{M}(X)_i &= S(\tilde{M}(X))_i + \tilde{M}(X)_i \\
&= \sum_{\substack{n \in \{1,\dots,N_{\text{in}}\} \\ l \in \{1,\dots,L_{\text{in}}\}}} \left(F_{n,l}^{S \circ \tilde{M}}(X,i) + F^{\tilde{M}}(X,i) \right) + b^{S \circ \tilde{M}}(X,i) + b^{\tilde{M}}(X,i)
\end{aligned}$$

□

A.6 Benchmark Datasets Used in Experiments

In the field of time-series forecasting, several key datasets are commonly used to evaluate model performance. This section introduces the datasets utilized in our study: ETTh1, ETTh2, ETTm1, ETTm2, electricity and weather. To give a rough overview, Table A.1 compiles key figures for each of the datasets. We now want to give a short description and their respective goals for each dataset.

Dataset	N_{in}	number of datapoints	mean correlation between variates
ETTh1	7	17420	0.329
ETTh2	7	17420	0.345
ETTm1	7	69680	0.331
ETTm2	7	69680	0.345
electricity	321	26304	0.464
weather	21	52696	0.163

Table A.1: Key figures for the benchmark datasets that were used in this work.

ETT (Electricity Transformer Temperature) Datasets

The ETT datasets consist of four parts: ETTh1, ETTh2, ETTm1, and ETTm2. These datasets were proposed by [Zho+21], come from the State Grid Corporation of China and include data at two different resolutions: hourly (h) and minute (m). The subsets are:

- ETTh1: Hourly data from one transformer, showing its temperature over time.
- ETTh2: Hourly data from a second transformer, similar to ETTh1 but from another source.
- ETTm1: Minute-level data from one transformer, offering more detailed time intervals.
- ETTm2: Minute-level data from a second transformer, similar to ETTm1 but from another source.

It should be noted that the electricity transformer has no connection with the Transformer models explored in this work. These datasets are useful for testing models' ability to handle both long-term and short-term patterns in time-series data.

Electricity Dataset

The electricity dataset includes the electricity usage of 321 customers, recorded every 15 minutes over several years. This dataset helps evaluate models for forecasting electricity demand, highlighting their ability to capture regular patterns and sudden changes in consumption.

Weather Dataset

The weather dataset contains meteorological data from the Weather Station of Max Planck Institute for Biogeochemistry. It includes 21 different weather variables, such as temperature, humidity and wind speed, recorded every 10 minutes. This dataset is important for models focused on forecasting weather conditions, requiring them to handle multiple types of data.

		TP	No Processing										TokenMLP									
		Metric	MAE					MSE					MAE					MSE				
		d	32	64	128	256	512	32	64	128	256	512	32	64	128	256	512	32	64	128	256	512
VM	TM	n_{layers}																				
$\text{MLP}_{\text{TM}}^{128}$	$\text{MLP}_{\text{TM}}^{128}$	1	0.498	0.494	0.491	0.517	0.513	0.521	0.509	0.502	0.529	0.527	0.490	0.483	0.485	0.511	0.519	0.498	0.486	0.485	0.521	0.537
		2	0.487	0.491	0.485	0.501	0.505	0.490	0.505	0.494	0.514	0.519	0.489	0.482	0.492	0.504	0.526	0.494	0.484	0.492	0.501	0.549
		3	0.494	0.488	0.487	0.499	0.510	0.505	0.496	0.493	0.503	0.533	0.491	0.485	0.487	0.517	0.519	0.496	0.483	0.480	0.542	0.540
	No Mixing	1	0.477	0.472	0.468	0.472	0.474	0.489	0.478	0.470	0.482	0.486	0.473	0.469	0.466	0.463	0.467	0.477	0.469	0.464	0.461	0.465
		2	0.474	0.468	0.465	0.467	0.468	0.481	0.471	0.464	0.461	0.468	0.472	0.467	0.464	0.462	0.465	0.474	0.467	0.460	0.456	0.459
		3	0.474	0.467	0.465	0.463	0.471	0.479	0.467	0.461	0.458	0.464	0.472	0.465	0.461	0.463	0.471	0.472	0.460	0.452	0.453	0.468
	SA	1	0.475	0.473	0.470	0.481	0.482	0.483	0.474	0.464	0.479	0.480	0.474	0.470	0.473	0.474	0.484	0.478	0.470	0.468	0.468	0.485
		2	0.476	0.470	0.469	0.474	0.497	0.481	0.469	0.470	0.469	0.496	0.474	0.477	0.471	0.475	0.482	0.475	0.480	0.476	0.468	0.477
		3	0.473	0.472	0.470	0.480	0.475	0.474	0.472	0.460	0.473	0.467	0.476	0.474	0.473	0.466	0.471	0.478	0.473	0.466	0.458	0.465
$\text{MLP}_{\text{TM}}^{32}$	$\text{MLP}_{\text{TM}}^{32}$	1	0.497	0.491	0.488	0.487	0.506	0.528	0.518	0.511	0.502	0.528	0.490	0.482	0.478	0.488	0.496	0.503	0.493	0.488	0.499	0.504
		2	0.496	0.491	0.487	0.490	0.490	0.520	0.512	0.503	0.503	0.500	0.491	0.486	0.482	0.477	0.512	0.502	0.498	0.489	0.479	0.518
		3	0.494	0.490	0.487	0.482	0.495	0.515	0.508	0.496	0.488	0.499	0.493	0.481	0.482	0.484	0.503	0.502	0.485	0.483	0.483	0.520
	No Mixing	1	0.481	0.475	0.472	0.478	0.483	0.502	0.490	0.485	0.496	0.506	0.474	0.467	0.467	0.465	0.464	0.480	0.475	0.473	0.465	0.461
		2	0.480	0.473	0.470	0.467	0.472	0.491	0.486	0.480	0.464	0.477	0.473	0.471	0.464	0.462	0.463	0.480	0.475	0.465	0.459	0.456
		3	0.480	0.474	0.469	0.472	0.468	0.489	0.487	0.474	0.472	0.467	0.474	0.475	0.470	0.465	0.467	0.479	0.481	0.473	0.458	0.459
	SA	1	0.477	0.473	0.480	0.477	0.477	0.486	0.479	0.479	0.473	0.477	0.475	0.470	0.473	0.476	0.485	0.483	0.476	0.483	0.477	0.487
		2	0.477	0.472	0.475	0.481	0.490	0.481	0.475	0.476	0.478	0.488	0.481	0.472	0.471	0.486	0.480	0.489	0.478	0.472	0.488	0.478
		3	0.476	0.472	0.471	0.476	0.483	0.480	0.472	0.466	0.472	0.483	0.476	0.474	0.478	0.479	0.479	0.480	0.476	0.477	0.480	0.478
$\text{MLP}_{\text{TM}}^{64}$	$\text{MLP}_{\text{TM}}^{64}$	1	0.497	0.498	0.490	0.494	0.501	0.522	0.522	0.506	0.507	0.515	0.491	0.484	0.480	0.497	0.516	0.502	0.496	0.486	0.505	0.525
		2	0.496	0.490	0.484	0.495	0.493	0.516	0.509	0.495	0.499	0.506	0.491	0.482	0.480	0.515	0.521	0.502	0.490	0.485	0.533	0.535
		3	0.491	0.491	0.483	0.493	0.504	0.497	0.507	0.492	0.501	0.524	0.488	0.486	0.495	0.484	0.510	0.489	0.492	0.506	0.479	0.536
	No Mixing	1	0.480	0.473	0.468	0.470	0.485	0.497	0.486	0.473	0.477	0.511	0.472	0.466	0.463	0.465	0.465	0.478	0.471	0.462	0.468	0.464
		2	0.479	0.472	0.469	0.463	0.466	0.494	0.482	0.473	0.459	0.462	0.472	0.467	0.464	0.466	0.467	0.476	0.471	0.463	0.466	0.462
		3	0.478	0.471	0.466	0.464	0.466	0.490	0.476	0.466	0.458	0.463	0.473	0.467	0.467	0.465	0.471	0.476	0.465	0.465	0.456	0.462
	SA	1	0.477	0.473	0.479	0.473	0.482	0.486	0.478	0.481	0.471	0.478	0.474	0.469	0.470	0.471	0.483	0.481	0.472	0.471	0.472	0.481
		2	0.477	0.473	0.469	0.474	0.482	0.484	0.478	0.468	0.467	0.478	0.476	0.475	0.467	0.483	0.474	0.480	0.476	0.463	0.480	0.468
		3	0.479	0.475	0.478	0.469	0.475	0.484	0.480	0.475	0.463	0.468	0.478	0.477	0.475	0.480	0.472	0.482	0.483	0.473	0.475	0.466
No Mixing	$\text{MLP}_{\text{TM}}^{128}$	1	0.514	0.503	0.508	0.523	0.529	0.553	0.531	0.532	0.555	0.566	0.499	0.514	0.516	0.523	0.523	0.522	0.523	0.533	0.548	0.551
		2	0.517	0.510	0.530	0.522	0.527	0.530	0.541	0.566	0.551	0.557	0.502	0.491	0.532	0.545	0.535	0.509	0.501	0.558	0.580	0.560
		3	0.524	0.519	0.523	0.517	0.535	0.534	0.537	0.545	0.541	0.569	0.509	0.535	0.519	0.531	0.529	0.523	0.563	0.537	0.560	0.560
	$\text{MLP}_{\text{TM}}^{32}$	1	0.502	0.498	0.500	0.504	0.505	0.539	0.532	0.535	0.540	0.537	0.495	0.491	0.495	0.497	0.515	0.516	0.514	0.518	0.512	0.541
		2	0.510	0.508	0.509	0.506	0.518	0.548	0.543	0.529	0.523	0.554	0.491	0.494	0.504	0.503	0.523	0.508	0.516	0.521	0.523	0.549
		3	0.513	0.511	0.508	0.514	0.515	0.555	0.551	0.527	0.538	0.532	0.496	0.491	0.494	0.499	0.507	0.520	0.503	0.508	0.515	0.522
	$\text{MLP}_{\text{TM}}^{64}$	1	0.508	0.506	0.504	0.518	0.529	0.548	0.522	0.539	0.545	0.555	0.500	0.496	0.516	0.522	0.534	0.525	0.520	0.524	0.540	0.574
		2	0.498	0.514	0.516	0.537	0.534	0.512	0.554	0.543	0.581	0.570	0.495	0.488	0.510	0.526	0.534	0.513	0.504	0.517	0.546	0.569
		3	0.507	0.515	0.514	0.538	0.514	0.519	0.554	0.542	0.589	0.544	0.501	0.489	0.513	0.534	0.533	0.526	0.497	0.527	0.554	0.580
	No Mixing	1	0.499	0.496	0.495	0.495	0.497	0.538	0.535	0.535	0.535	0.539	0.477	0.473	0.470	0.470	0.483	0.494	0.488	0.484	0.480	0.495
		2	0.498	0.495	0.497	0.496	0.500	0.537	0.535	0.538	0.539	0.546	0.479	0.476	0.472	0.469	0.471	0.496	0.491	0.484	0.481	0.481
		3	0.499	0.495	0.495	0.496	0.497	0.538	0.535	0.535	0.538	0.545	0.480	0.474	0.470	0.477	0.476	0.496	0.488	0.484	0.492	0.484
	SA	1	0.484	0.482	0.489	0.483	0.489	0.507	0.504	0.515	0.505	0.513	0.478	0.474	0.483	0.491	0.499	0.494	0.491	0.503	0.507	0.511
		2	0.484	0.485	0.491	0.493	0.497	0.508	0.508	0.510	0.518	0.522	0.479	0.479	0.481	0.488	0.485	0.495	0.497	0.499	0.502	0.499
		3	0.485	0.482	0.492	0.492	0.500	0.506	0.498	0.511	0.521	0.522	0.484	0.485	0.481	0.484	0.487	0.500	0.502	0.491	0.501	0.507
SA	$\text{MLP}_{\text{TM}}^{128}$	1	0.500	0.497	0.509	0.556	0.514	0.528	0.519	0.531	0.613	0.542	0.492	0.490	0.514	0.546	0.580	0.510	0.504	0.536	0.582	0.657
		2	0.506	0.494	0.501	0.585	0.579	0.528	0.515	0.517	0.665	0.654	0.494	0.487	0.501	0.587	0.544	0.502	0.497	0.513	0.659	0.581
		3	0.505	0.491	0.498	0.536	0.559	0.528	0.494	0.512	0.569	0.612	0.510	0.502	0.553	0.554	0.567	0.520	0.506	0.602	0.604	0.619
	$\text{MLP}_{\text{TM}}^{32}$	1	0.489	0.495	0.488	0.498	0.501	0.510	0.518	0.507	0.520	0.525	0.497	0.490	0.481	0.502	0.513	0.520	0.507	0.493	0.526	0.537
		2	0.495	0.493	0.477	0.496	0.516	0.514	0.514	0.484	0.512	0.537	0.501	0.485	0.492	0.512	0.507	0.521	0.494	0.500	0.530	0.534
		3	0.502	0.486	0.497	0.492	0.501	0.521	0.500	0.514	0.507	0.523	0.488	0.487	0.492	0.520	0.552	0.500	0.498	0.490	0.541	0.607
	$\text{MLP}_{\text{TM}}^{64}$	1	0.497	0.491	0.490	0.492	0.513	0.520	0.508	0.506	0.500	0.536	0.494	0.491	0.494	0.520	0.533	0.513	0.508	0.504	0.551	0.561
		2	0.496	0.486	0.479	0.534	0.518	0.514	0.499	0.488	0.582	0.546	0.494	0.485	0.502	0.532	0.567	0.511	0.491	0.517	0.563	0.633
		3	0.502	0.490	0.549	0.501	0.567	0.523	0.500	0.599	0.522	0.623	0.501	0.487	0.504	0.573	0.563	0.520	0.489	0.525	0.632	0.621
	No Mixing	1	0.475	0.470	0.472	0.476	0.469	0.489	0.477	0.470	0.472	0.472	0.476	0.470	0.472	0.472	0.479	0.487	0.478	0.475	0.476	0.484
		2	0.472	0.472	0.468	0.477	0.472	0.485	0.479	0.469	0.478	0.472	0.476	0.471	0.468	0.476	0.474	0.488	0.480	0.471	0.478	0.475
		3	0.473	0.469	0.472	0.475	0.477	0.484	0.475	0.472	0.473	0.491	0.477	0.471	0.472	0.470	0.472	0.490	0.478	0.478	0.471	0.481
	SA	1	0.473	0.474	0.476	0.505	0.491	0.486	0.481	0.480	0.522	0.504	0.481	0.475	0.477	0.478	0.486	0.492	0.482	0.483	0.484	0.492
		2	0.476	0.477	0.490	0.497	0.486	0.488	0.483	0.499	0.506	0.495	0.476	0.484	0.481	0.496	0.496	0.487	0.491	0.488	0.502	0.509
		3	0.474	0.474	0.478	0.481	0.491	0.486	0.482	0.486	0.485	0.496	0.482	0.488	0.483	0.485	0.487	0.496	0.496	0.490	0.489	0.495

Table A.2: Raw experiment results for $L_{\text{in}} = 96$ and $T = 192$ on ETTh1 dataset.

A.7 Further Experiment Results for Chapter 3

We present the experimental results for the cases that we have left out in the main text for the sake of brevity.

Raw Results

The raw experiment results for the configurations $L_{in} \in \{96, 512\}$, $T \in \{96, 192, 336, 720\}$, that have not been treated in chapter 3, can be found in Tables A.2, A.3, A.4, A.5, A.6, A.7 and A.8.

Architecture Comparison

We show the architecture comparison results for the datasets ETTh2, ETTm1, ETTm2 and weather and for the cases $L_{\text{in}} = 96$ and $L_{\text{in}} = 512$ in Tables A.9, A.10, A.11, A.12, A.13, A.14, A.15 and A.16.

		TP	No Processing										TokenMLP									
		Metric	MAE					MSE					MAE					MSE				
		d	32	64	128	256	512	32	64	128	256	512	32	64	128	256	512	32	64	128	256	512
VM	TM	n_{layers}																				
	$\text{MLP}_{\text{TM}}^{128}$	1	0.546	0.541	0.544	0.576	0.583	0.606	0.594	0.588	0.651	0.669	0.539	0.536	0.580	0.587	0.626	0.575	0.572	0.642	0.646	0.735
		2	0.540	0.539	0.560	0.544	0.567	0.587	0.589	0.608	0.586	0.633	0.529	0.527	0.561	0.576	0.621	0.560	0.554	0.606	0.634	0.729
		3	0.544	0.547	0.567	0.566	0.562	0.586	0.593	0.622	0.625	0.626	0.533	0.535	0.555	0.572	0.590	0.567	0.563	0.591	0.635	0.675
	No Mixing	1	0.508	0.503	0.503	0.503	0.508	0.544	0.534	0.531	0.533	0.545	0.504	0.499	0.500	0.499	0.500	0.529	0.521	0.523	0.519	0.518
$\text{MLP}_{\text{TM}}^{128}$		2	0.506	0.510	0.502	0.500	0.501	0.536	0.542	0.529	0.526	0.516	0.506	0.500	0.498	0.498	0.499	0.530	0.521	0.513	0.515	0.508
		3	0.506	0.501	0.498	0.496	0.502	0.533	0.524	0.517	0.509	0.516	0.506	0.498	0.499	0.499	0.504	0.528	0.516	0.516	0.506	0.521
	SA	1	0.508	0.506	0.511	0.524	0.518	0.535	0.529	0.527	0.551	0.543	0.512	0.505	0.507	0.508	0.526	0.542	0.528	0.530	0.531	0.555
		2	0.507	0.507	0.511	0.518	0.543	0.531	0.534	0.531	0.542	0.578	0.507	0.510	0.518	0.535	0.523	0.530	0.537	0.535	0.565	0.548
		3	0.510	0.507	0.512	0.517	0.528	0.536	0.529	0.534	0.534	0.551	0.508	0.509	0.509	0.513	0.515	0.529	0.533	0.535	0.528	0.529
	$\text{MLP}_{\text{TM}}^{32}$	1	0.547	0.540	0.539	0.556	0.575	0.612	0.604	0.595	0.633	0.668	0.534	0.526	0.527	0.567	0.599	0.580	0.563	0.561	0.619	0.679
		2	0.537	0.539	0.533	0.557	0.557	0.590	0.596	0.582	0.622	0.611	0.532	0.515	0.538	0.575	0.574	0.570	0.546	0.575	0.640	0.628
		3	0.538	0.531	0.536	0.544	0.551	0.593	0.581	0.586	0.593	0.605	0.533	0.536	0.535	0.572	0.564	0.570	0.576	0.565	0.613	0.621
	No Mixing	1	0.511	0.508	0.509	0.512	0.517	0.550	0.547	0.552	0.559	0.574	0.505	0.498	0.503	0.497	0.502	0.532	0.526	0.534	0.515	0.523
$\text{MLP}_{\text{TM}}^{32}$		2	0.512	0.506	0.502	0.499	0.509	0.550	0.538	0.535	0.521	0.537	0.504	0.499	0.496	0.503	0.509	0.531	0.522	0.515	0.530	0.537
		3	0.511	0.505	0.504	0.500	0.508	0.546	0.536	0.532	0.521	0.538	0.507	0.500	0.500	0.504	0.503	0.531	0.520	0.514	0.519	0.515
	SA	1	0.507	0.507	0.506	0.519	0.523	0.537	0.538	0.536	0.541	0.554	0.509	0.505	0.518	0.526	0.515	0.539	0.535	0.547	0.559	0.540
		2	0.510	0.506	0.510	0.512	0.517	0.539	0.532	0.529	0.535	0.544	0.509	0.524	0.507	0.512	0.540	0.536	0.551	0.535	0.536	0.571
		3	0.509	0.510	0.505	0.503	0.509	0.534	0.531	0.527	0.521	0.526	0.517	0.507	0.506	0.512	0.547	0.546	0.533	0.527	0.533	0.586
	$\text{MLP}_{\text{TM}}^{64}$	1	0.551	0.541	0.541	0.565	0.562	0.621	0.597	0.599	0.644	0.636	0.533	0.526	0.552	0.581	0.608	0.570	0.562	0.595	0.647	0.706
		2	0.540	0.534	0.552	0.554	0.571	0.594	0.582	0.611	0.610	0.634	0.529	0.522	0.563	0.578	0.583	0.564	0.548	0.614	0.637	0.662
		3	0.535	0.537	0.554	0.548	0.559	0.583	0.586	0.603	0.602	0.618	0.529	0.523	0.554	0.574	0.593	0.561	0.544	0.592	0.637	0.675
	No Mixing	1	0.513	0.506	0.504	0.509	0.508	0.555	0.544	0.536	0.549	0.545	0.503	0.502	0.499	0.500	0.503	0.528	0.528	0.525	0.527	0.519
$\text{MLP}_{\text{TM}}^{64}$		2	0.509	0.505	0.502	0.504	0.500	0.543	0.537	0.529	0.531	0.518	0.502	0.499	0.497	0.497	0.500	0.526	0.520	0.515	0.515	0.511
		3	0.508	0.505	0.501	0.499	0.497	0.540	0.531	0.525	0.514	0.508	0.505	0.498	0.496	0.495	0.500	0.528	0.515	0.509	0.502	0.507
	SA	1	0.510	0.506	0.514	0.517	0.518	0.540	0.538	0.540	0.553	0.539	0.506	0.506	0.511	0.512	0.525	0.532	0.529	0.535	0.537	0.550
		2	0.507	0.505	0.510	0.522	0.517	0.532	0.531	0.536	0.552	0.537	0.506	0.507	0.510	0.536	0.521	0.530	0.534	0.539	0.570	0.546
		3	0.511	0.511	0.503	0.524	0.541	0.534	0.532	0.526	0.549	0.572	0.508	0.513	0.509	0.520	0.526	0.531	0.530	0.524	0.541	0.548
	$\text{MLP}_{\text{TM}}^{128}$	1	0.563	0.557	0.619	0.592	0.614	0.642	0.631	0.737	0.685	0.727	0.543	0.542	0.583	0.633	0.610	0.600	0.588	0.654	0.753	0.702
		2	0.566	0.577	0.587	0.573	0.619	0.645	0.651	0.667	0.643	0.748	0.543	0.569	0.594	0.596	0.594	0.595	0.625	0.675	0.669	0.666
		3	0.567	0.567	0.579	0.581	0.566	0.639	0.628	0.648	0.656	0.644	0.549	0.582	0.591	0.597	0.574	0.602	0.641	0.669	0.682	0.638
	$\text{MLP}_{\text{TM}}^{32}$	1	0.554	0.564	0.563	0.570	0.582	0.629	0.653	0.654	0.649	0.675	0.548	0.537	0.537	0.582	0.624	0.608	0.589	0.592	0.642	0.725
		2	0.559	0.574	0.575	0.609	0.581	0.637	0.676	0.664	0.747	0.655	0.548	0.534	0.550	0.626	0.572	0.606	0.585	0.607	0.720	0.632
		3	0.565	0.571	0.573	0.562	0.602	0.646	0.661	0.650	0.623	0.709	0.540	0.538	0.569	0.562	0.579	0.588	0.580	0.621	0.635	0.653
	$\text{MLP}_{\text{TM}}^{64}$	1	0.565	0.555	0.563	0.585	0.616	0.649	0.637	0.645	0.669	0.741	0.550	0.527	0.566	0.582	0.621	0.613	0.571	0.621	0.659	0.729
No Mixing		2	0.566	0.558	0.586	0.590	0.602	0.640	0.627	0.675	0.686	0.710	0.545	0.543	0.589	0.559	0.610	0.599	0.590	0.650	0.608	0.711
		3	0.570	0.587	0.609	0.563	0.582	0.649	0.686	0.721	0.631	0.670	0.532	0.577	0.598	0.590	0.590	0.579	0.634	0.663	0.662	0.661
	No Mixing	1	0.528	0.526	0.525	0.526	0.529	0.588	0.588	0.584	0.589	0.596	0.511	0.506	0.506	0.507	0.505	0.547	0.540	0.542	0.543	0.540
		2	0.528	0.525	0.525	0.526	0.527	0.587	0.585	0.585	0.589	0.593	0.510	0.508	0.506	0.504	0.509	0.547	0.545	0.540	0.538	0.543
		3	0.528	0.526	0.526	0.525	0.529	0.588	0.587	0.587	0.586	0.592	0.513	0.508	0.506	0.506	0.510	0.550	0.544	0.540	0.541	0.544
	SA	1	0.515	0.517	0.523	0.534	0.529	0.562	0.560	0.573	0.597	0.588	0.511	0.510	0.520	0.522	0.525	0.547	0.545	0.564	0.562	0.561
		2	0.517	0.518	0.538	0.537	0.547	0.563	0.561	0.596	0.606	0.612	0.514	0.509	0.537	0.527	0.538	0.551	0.544	0.584	0.570	0.592
		3	0.517	0.515	0.557	0.551	0.543	0.559	0.556	0.632	0.628	0.598	0.517	0.516	0.517	0.528	0.545	0.555	0.549	0.555	0.569	0.604
	$\text{MLP}_{\text{TM}}^{128}$	1	0.552	0.551	0.585	0.620	0.631	0.617	0.610	0.658	0.725	0.760	0.542	0.551	0.589	0.620	0.657	0.597	0.594	0.653	0.717	0.805
		2	0.556	0.563	0.582	0.647	0.664	0.623	0.620	0.645	0.783	0.821	0.550	0.568	0.606	0.590	0.690	0.597	0.618	0.692	0.666	0.882
		3	0.550	0.589	0.585	0.611	0.647	0.609	0.660	0.651	0.718	0.799	0.571	0.587	0.587	0.600	0.586	0.622	0.655	0.658	0.688	0.665
	$\text{MLP}_{\text{TM}}^{32}$	1	0.546	0.529	0.533	0.585	0.570	0.607	0.578	0.576	0.662	0.633	0.545	0.522	0.579	0.558	0.581	0.603	0.563	0.642	0.609	0.660
		2	0.539	0.544	0.570	0.656	0.604	0.592	0.600	0.643	0.816	0.701	0.526	0.531	0.572	0.603	0.581	0.568	0.562	0.632	0.697	0.643
		3	0.552	0.546	0.566	0.619	0.613	0.611	0.603	0.619	0.723	0.717	0.531	0.549	0.556	0.616	0.573	0.573	0.594	0.591	0.718	0.653
	$\text{MLP}_{\text{TM}}^{64}$	1	0.543	0.539	0.541	0.628	0.647	0.604	0.592	0.584	0.744	0.799	0.536	0.537	0.559	0.584	0.596	0.583	0.582	0.608	0.640	0.673
SA		2	0.546	0.558	0.567	0.641	0.630	0.602	0.624	0.627	0.777	0.767	0.538	0.542	0.603	0.666	0.602	0.591	0.578	0.688	0.804	0.684
		3	0.542	0.559	0.558	0.611	0.647	0.593	0.622	0.616	0.705	0.751	0.542	0.555	0.599	0.615	0.590	0.586	0.596	0.681	0.716	0.662
	No Mixing	1	0.508	0.510	0.515	0.518	0.524	0.541	0.538	0.543	0.550	0.558	0.510	0.504	0.509	0.504	0.511	0.545	0.534	0.536	0.529	0.540
		2	0.508	0.505	0.513	0.515	0.523	0.539	0.531	0.540	0.545	0.558	0.507	0.508	0.502	0.511	0.514	0.542	0.537	0.528	0.538	0.543
		3	0.510	0.509	0.511	0.511	0.508	0.541	0.536	0.536	0.537	0.534	0.512	0.509	0.500	0.508	0.520	0.545	0.539	0.523	0.538	0.553
	SA	1	0.507	0.510	0.525	0.533	0.511	0.538	0.540	0.557	0.580	0.541	0.505	0.516	0.520	0.515	0.520	0.538	0.545	0.556	0.551	0.554
		2	0.510	0.513	0.530	0.532	0.523	0.540	0.541	0.562	0.565	0.552	0.511	0.515	0.528	0.511	0.535	0.543	0.544	0.563	0.535	0.571
		3	0.508	0.514	0.548	0.522	0.522	0.537	0.546	0.593	0.547	0.557	0.518	0.529	0.512	0.525	0.524	0.550	0.569	0.540	0.549	0.560

Table A.3: Raw experiment results for $L_{\text{in}} = 96$ and $T = 336$ on ETTh1 dataset.

Promotion Results

We show the evaluation plots for the performance promotion for the datasets ETTh2, ETTm1, ETTm2 and weather in Figures A.1, A.2, A.3 and A.4.

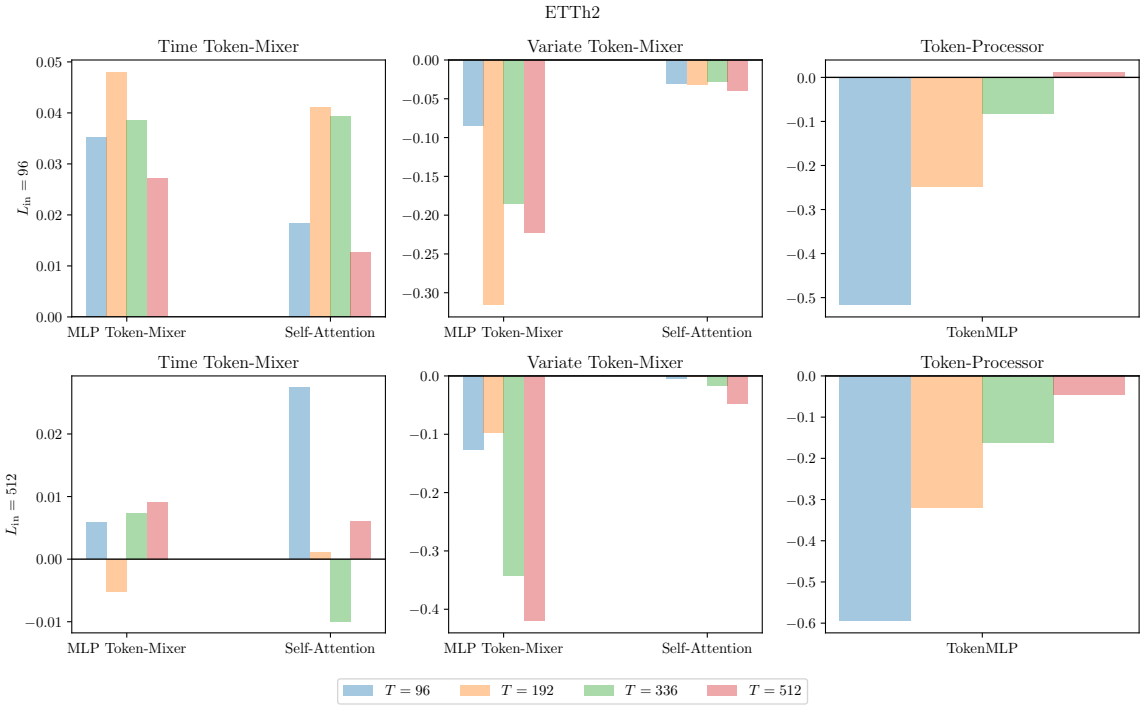

Figure A.1: Performance promotion by using different token-mixers and token-processors for the ETTh2 dataset. The baseline is not mixing (and respectively not processing).

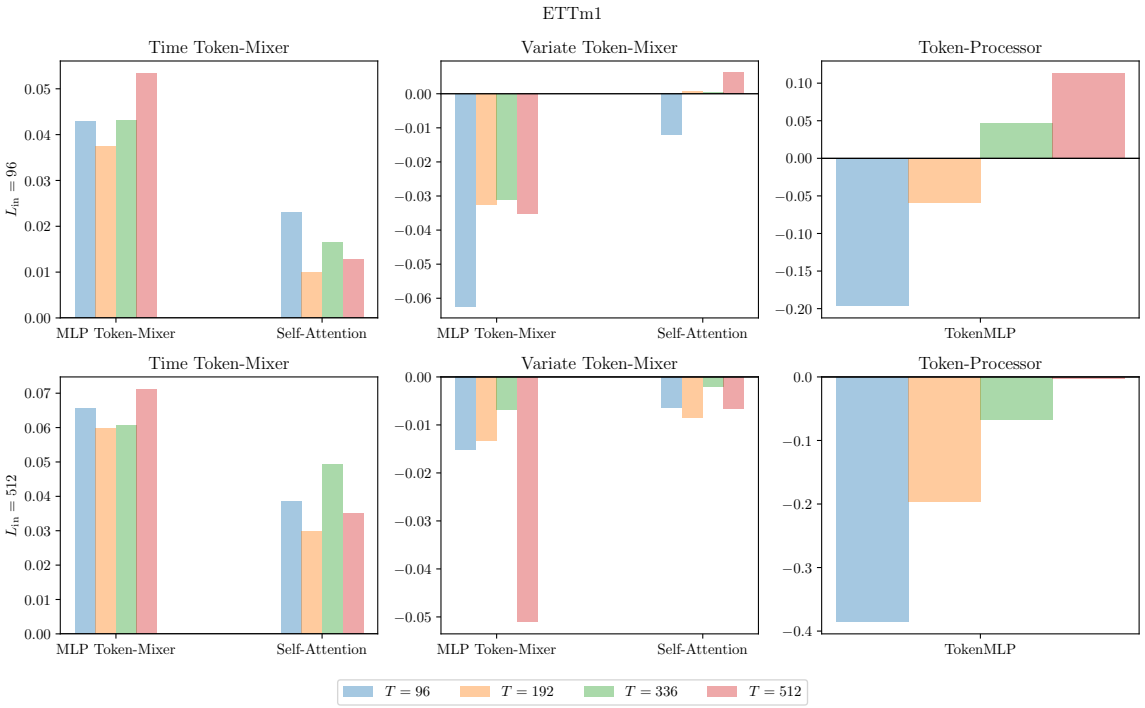

Figure A.2: Performance promotion by using different token-mixers and token-processors for the ETTm1 dataset. The baseline is not mixing (and respectively not processing).

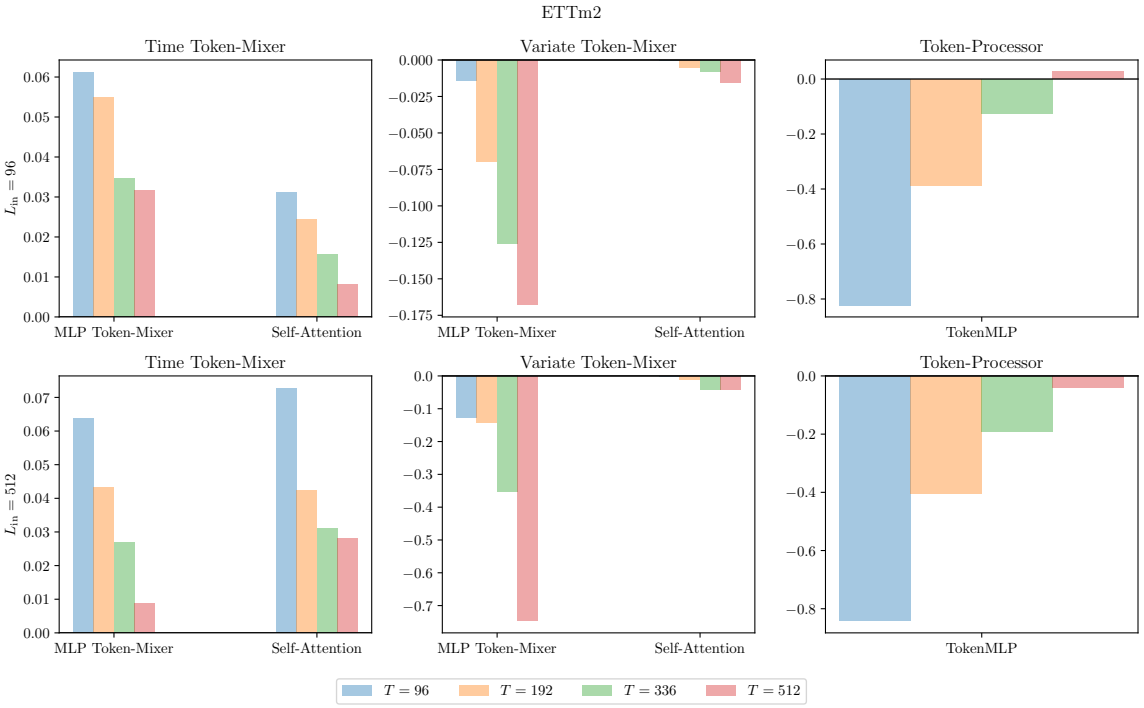

Figure A.3: Performance promotion by using different token-mixers and token-processors for the ETTm2 dataset. The baseline is not mixing (and respectively not processing).

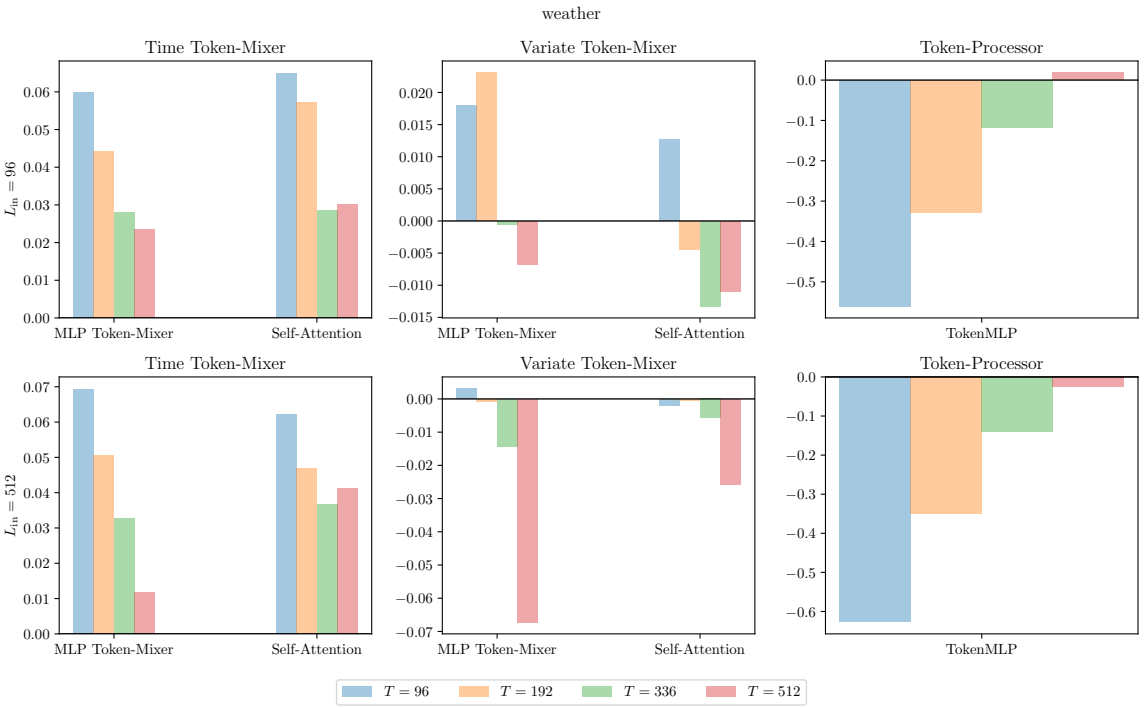

Figure A.4: Performance promotion by using different token-mixers and token-processors for the weather dataset. The baseline is not mixing (and respectively not processing).

		TP	No Processing										TokenMLP									
		Metric	MAE					MSE					MAE					MSE				
		d	32	64	128	256	512	32	64	128	256	512	32	64	128	256	512	32	64	128	256	512
VM	TM	n_{layers}																				
	$\text{MLP}_{\text{TM}}^{128}$	1	0.574	0.578	0.592	0.605	0.603	0.651	0.646	0.663	0.681	0.686	0.565	0.596	0.621	0.640	0.644	0.620	0.652	0.696	0.741	0.746
		2	0.577	0.590	0.583	0.587	0.588	0.650	0.649	0.637	0.646	0.654	0.561	0.608	0.596	0.627	0.631	0.604	0.667	0.646	0.703	0.728
		3	0.575	0.578	0.583	0.579	0.583	0.621	0.626	0.637	0.632	0.640	0.562	0.585	0.631	0.627	0.618	0.601	0.626	0.714	0.728	0.706
	No Mixing	1	0.532	0.535	0.532	0.530	0.531	0.578	0.586	0.578	0.575	0.579	0.531	0.529	0.524	0.532	0.527	0.566	0.567	0.556	0.567	0.555
$\text{MLP}_{\text{TM}}^{128}$		2	0.532	0.528	0.529	0.528	0.524	0.575	0.569	0.571	0.563	0.553	0.531	0.527	0.525	0.527	0.531	0.566	0.562	0.560	0.559	0.566
		3	0.530	0.527	0.525	0.524	0.526	0.570	0.565	0.562	0.555	0.555	0.529	0.532	0.526	0.525	0.537	0.562	0.568	0.552	0.543	0.563
	SA	1	0.538	0.536	0.540	0.539	0.552	0.582	0.575	0.581	0.576	0.594	0.535	0.535	0.542	0.545	0.549	0.577	0.573	0.589	0.586	0.593
		2	0.532	0.532	0.532	0.545	0.554	0.569	0.570	0.571	0.584	0.604	0.533	0.536	0.531	0.547	0.557	0.568	0.572	0.569	0.591	0.605
		3	0.534	0.538	0.543	0.547	0.563	0.569	0.579	0.582	0.580	0.609	0.537	0.547	0.539	0.544	0.547	0.570	0.590	0.572	0.576	0.583
	$\text{MLP}_{\text{TM}}^{32}$	1	0.578	0.566	0.563	0.581	0.594	0.668	0.648	0.636	0.656	0.684	0.570	0.556	0.602	0.593	0.643	0.632	0.606	0.673	0.655	0.751
		2	0.564	0.566	0.573	0.568	0.574	0.642	0.642	0.645	0.614	0.641	0.567	0.556	0.593	0.603	0.612	0.625	0.602	0.645	0.668	0.688
		3	0.564	0.557	0.565	0.581	0.569	0.637	0.622	0.623	0.652	0.628	0.561	0.560	0.598	0.594	0.595	0.615	0.598	0.656	0.650	0.664
	No Mixing	1	0.537	0.531	0.532	0.537	0.539	0.587	0.581	0.586	0.596	0.604	0.529	0.525	0.526	0.526	0.525	0.567	0.562	0.565	0.561	0.563
$\text{MLP}_{\text{TM}}^{32}$		2	0.533	0.532	0.530	0.530	0.526	0.581	0.579	0.579	0.574	0.564	0.534	0.528	0.525	0.523	0.533	0.575	0.564	0.562	0.557	0.561
		3	0.532	0.529	0.525	0.526	0.525	0.577	0.572	0.566	0.555	0.559	0.533	0.530	0.526	0.526	0.532	0.574	0.569	0.552	0.550	0.553
	SA	1	0.536	0.535	0.542	0.552	0.550	0.579	0.580	0.583	0.608	0.596	0.539	0.535	0.535	0.537	0.564	0.586	0.577	0.582	0.575	0.618
		2	0.534	0.532	0.557	0.549	0.559	0.574	0.573	0.606	0.589	0.608	0.544	0.535	0.538	0.548	0.555	0.592	0.576	0.581	0.590	0.601
		3	0.537	0.534	0.531	0.536	0.570	0.575	0.572	0.568	0.566	0.625	0.536	0.542	0.538	0.549	0.560	0.576	0.584	0.575	0.591	0.606
	$\text{MLP}_{\text{TM}}^{64}$	1	0.579	0.571	0.581	0.594	0.600	0.668	0.653	0.658	0.682	0.699	0.560	0.565	0.599	0.632	0.637	0.613	0.616	0.659	0.725	0.735
		2	0.562	0.567	0.588	0.575	0.589	0.629	0.609	0.652	0.633	0.656	0.557	0.574	0.597	0.624	0.603	0.605	0.625	0.654	0.693	0.676
		3	0.571	0.566	0.578	0.594	0.589	0.637	0.617	0.637	0.668	0.661	0.570	0.586	0.583	0.602	0.584	0.629	0.636	0.624	0.669	0.642
	No Mixing	1	0.534	0.532	0.532	0.535	0.535	0.583	0.580	0.583	0.589	0.599	0.531	0.525	0.525	0.525	0.534	0.566	0.561	0.562	0.559	0.567
$\text{MLP}_{\text{TM}}^{64}$		2	0.533	0.528	0.530	0.527	0.530	0.578	0.571	0.575	0.565	0.560	0.530	0.525	0.523	0.525	0.529	0.566	0.562	0.556	0.559	0.554
		3	0.533	0.526	0.525	0.524	0.530	0.577	0.564	0.561	0.559	0.564	0.532	0.526	0.525	0.523	0.532	0.570	0.558	0.553	0.546	0.555
	SA	1	0.539	0.535	0.544	0.538	0.553	0.583	0.578	0.585	0.575	0.610	0.536	0.537	0.540	0.550	0.547	0.577	0.579	0.577	0.598	0.589
		2	0.535	0.531	0.536	0.549	0.556	0.574	0.571	0.575	0.593	0.606	0.537	0.542	0.538	0.559	0.560	0.573	0.587	0.578	0.615	0.612
		3	0.533	0.534	0.541	0.556	0.550	0.570	0.572	0.575	0.593	0.589	0.534	0.533	0.535	0.555	0.558	0.568	0.569	0.571	0.595	0.609
	$\text{MLP}_{\text{TM}}^{128}$	1	0.594	0.623	0.616	0.634	0.636	0.698	0.723	0.712	0.744	0.758	0.574	0.631	0.609	0.635	0.631	0.640	0.726	0.681	0.733	0.729
		2	0.599	0.609	0.626	0.604	0.622	0.674	0.697	0.732	0.684	0.728	0.574	0.624	0.623	0.622	0.624	0.636	0.711	0.710	0.715	0.710
		3	0.594	0.582	0.606	0.609	0.604	0.661	0.636	0.686	0.706	0.687	0.617	0.614	0.642	0.622	0.620	0.695	0.703	0.745	0.703	0.707
	$\text{MLP}_{\text{TM}}^{32}$	1	0.596	0.587	0.595	0.599	0.617	0.705	0.689	0.700	0.719	0.739	0.571	0.560	0.571	0.621	0.642	0.646	0.614	0.648	0.710	0.760
		2	0.598	0.600	0.595	0.590	0.585	0.705	0.706	0.697	0.669	0.656	0.582	0.553	0.607	0.610	0.640	0.663	0.608	0.690	0.702	0.752
		3	0.601	0.613	0.609	0.591	0.622	0.710	0.729	0.714	0.675	0.729	0.575	0.585	0.584	0.600	0.612	0.652	0.643	0.659	0.674	0.702
	$\text{MLP}_{\text{TM}}^{64}$	1	0.590	0.590	0.585	0.618	0.636	0.689	0.694	0.678	0.726	0.777	0.570	0.586	0.603	0.638	0.640	0.645	0.649	0.678	0.745	0.741
No Mixing		2	0.595	0.593	0.620	0.602	0.602	0.694	0.687	0.728	0.692	0.677	0.581	0.597	0.627	0.596	0.645	0.658	0.663	0.717	0.659	0.759
		3	0.600	0.598	0.609	0.613	0.603	0.705	0.676	0.702	0.712	0.691	0.556	0.592	0.625	0.610	0.638	0.610	0.651	0.716	0.685	0.744
	No Mixing	1	0.550	0.548	0.546	0.546	0.550	0.621	0.620	0.615	0.617	0.627	0.538	0.534	0.539	0.536	0.539	0.591	0.585	0.591	0.588	0.592
		2	0.549	0.548	0.548	0.547	0.547	0.619	0.621	0.620	0.618	0.619	0.537	0.534	0.535	0.531	0.540	0.586	0.583	0.585	0.581	0.591
		3	0.550	0.549	0.544	0.549	0.546	0.620	0.621	0.613	0.621	0.616	0.541	0.536	0.533	0.536	0.534	0.592	0.585	0.584	0.587	0.583
	SA	1	0.544	0.544	0.552	0.551	0.563	0.606	0.608	0.626	0.618	0.642	0.544	0.541	0.549	0.551	0.551	0.601	0.593	0.605	0.604	0.613
		2	0.541	0.546	0.551	0.565	0.559	0.596	0.611	0.625	0.638	0.639	0.539	0.549	0.549	0.572	0.570	0.589	0.604	0.608	0.650	0.634
		3	0.542	0.548	0.577	0.552	0.565	0.598	0.612	0.660	0.618	0.637	0.542	0.553	0.555	0.565	0.570	0.594	0.611	0.609	0.630	0.649
	$\text{MLP}_{\text{TM}}^{128}$	1	0.584	0.574	0.646	0.650	0.665	0.674	0.637	0.775	0.784	0.809	0.575	0.627	0.625	0.636	0.645	0.645	0.720	0.720	0.737	0.757
		2	0.579	0.631	0.622	0.638	0.646	0.649	0.735	0.706	0.747	0.753	0.572	0.612	0.650	0.637	0.624	0.633	0.685	0.759	0.743	0.720
		3	0.589	0.599	0.626	0.650	0.651	0.665	0.668	0.725	0.762	0.779	0.600	0.607	0.625	0.651	0.641	0.655	0.676	0.718	0.781	0.756
	$\text{MLP}_{\text{TM}}^{32}$	1	0.577	0.564	0.582	0.629	0.644	0.662	0.634	0.656	0.749	0.775	0.577	0.563	0.582	0.581	0.595	0.662	0.621	0.654	0.638	0.662
		2	0.575	0.566	0.598	0.611	0.606	0.661	0.625	0.676	0.718	0.688	0.561	0.564	0.615	0.635	0.601	0.623	0.619	0.698	0.749	0.682
		3	0.599	0.566	0.570	0.609	0.659	0.695	0.621	0.614	0.684	0.792	0.559	0.582	0.628	0.634	0.634	0.616	0.637	0.713	0.740	0.746
	$\text{MLP}_{\text{TM}}^{64}$	1	0.579	0.566	0.586	0.616	0.637	0.667	0.631	0.662	0.707	0.749	0.572	0.573	0.595	0.613	0.615	0.644	0.631	0.660	0.685	0.716
SA		2	0.580	0.567	0.595	0.638	0.616	0.664	0.628	0.665	0.762	0.717	0.566	0.605	0.595	0.626	0.657	0.625	0.668	0.661	0.706	0.821
		3	0.580	0.568	0.638	0.619	0.644	0.651	0.618	0.756	0.707	0.769	0.571	0.628	0.668	0.687	0.610	0.618	0.714	0.820	0.844	0.707
	No Mixing	1	0.538	0.543	0.539	0.546	0.548	0.582	0.586	0.577	0.588	0.591	0.542	0.535	0.532	0.530	0.555	0.592	0.576	0.573	0.575	0.617
		2	0.541	0.542	0.545	0.551	0.545	0.589	0.581	0.593	0.601	0.589	0.540	0.536	0.537	0.535	0.549	0.586	0.581	0.582	0.576	0.594
		3	0.539	0.538	0.541	0.552	0.549	0.584	0.580	0.583	0.600	0.594	0.540	0.532	0.535	0.551	0.540	0.587	0.574	0.578	0.592	0.582
	SA	1	0.537	0.538	0.566	0.551	0.567	0.583	0.578	0.615	0.602	0.632	0.536	0.534	0.556	0.561	0.565	0.582	0.581	0.606	0.614	0.627
		2	0.537	0.539	0.567	0.569	0.546	0.581	0.587	0.621	0.629	0.586	0.538	0.540	0.557	0.556	0.553	0.587	0.585	0.606	0.599	0.599
		3	0.536	0.551	0.579	0.554	0.547	0.580	0.597	0.644	0.588	0.590	0.538	0.554	0.563	0.575	0.568	0.580	0.599	0.607	0.631	0.631

Table A.4: Raw experiment results for $L_{\text{in}} = 96$ and $T = 512$ on ETTh1 dataset.

		TP	No Processing										TokenMLP									
		Metric	MAE					MSE					MAE					MSE				
		d	32	64	128	256	512	32	64	128	256	512	32	64	128	256	512	32	64	128	256	512
VM	TM	n_{layers}																				
$\text{MLP}_{\text{TM}}^{128}$	$\text{MLP}_{\text{TM}}^{128}$	1	0.460	0.461	0.475	0.484	0.504	0.429	0.439	0.448	0.471	0.493	0.451	0.461	0.453	0.490	0.492	0.421	0.429	0.422	0.469	0.476
		2	0.464	0.474	0.472	0.475	0.481	0.430	0.449	0.446	0.464	0.467	0.451	0.446	0.449	0.468	0.490	0.422	0.411	0.414	0.443	0.477
		3	0.459	0.460	0.469	0.487	0.495	0.428	0.431	0.446	0.468	0.494	0.454	0.453	0.466	0.480	0.501	0.422	0.416	0.440	0.462	0.500
	No Mixing	1	0.445	0.448	0.453	0.468	0.464	0.421	0.429	0.434	0.456	0.457	0.440	0.438	0.439	0.446	0.450	0.408	0.406	0.404	0.417	0.426
		2	0.439	0.440	0.444	0.454	0.455	0.404	0.413	0.422	0.445	0.434	0.439	0.439	0.443	0.450	0.458	0.403	0.404	0.415	0.419	0.432
		3	0.441	0.440	0.446	0.446	0.457	0.410	0.408	0.418	0.420	0.443	0.439	0.437	0.450	0.450	0.451	0.403	0.401	0.419	0.419	0.427
	SA	1	0.440	0.441	0.453	0.467	0.490	0.407	0.414	0.432	0.451	0.498	0.443	0.441	0.443	0.446	0.487	0.410	0.408	0.414	0.415	0.486
		2	0.442	0.442	0.445	0.454	0.463	0.405	0.410	0.409	0.425	0.446	0.441	0.446	0.445	0.453	0.459	0.405	0.411	0.411	0.423	0.433
		3	0.448	0.445	0.446	0.448	0.492	0.414	0.407	0.415	0.415	0.474	0.447	0.446	0.454	0.461	0.454	0.412	0.410	0.425	0.434	0.428
$\text{MLP}_{\text{TM}}^{32}$	$\text{MLP}_{\text{TM}}^{32}$	1	0.461	0.458	0.467	0.477	0.506	0.439	0.441	0.454	0.466	0.496	0.452	0.450	0.444	0.457	0.486	0.427	0.426	0.416	0.431	0.461
		2	0.460	0.459	0.471	0.476	0.511	0.435	0.438	0.454	0.463	0.511	0.451	0.447	0.448	0.462	0.476	0.421	0.418	0.416	0.438	0.458
		3	0.457	0.455	0.468	0.473	0.489	0.435	0.430	0.441	0.457	0.480	0.445	0.450	0.449	0.464	0.479	0.412	0.411	0.417	0.434	0.462
	No Mixing	1	0.459	0.457	0.462	0.469	0.487	0.437	0.446	0.452	0.465	0.497	0.437	0.442	0.437	0.440	0.456	0.405	0.416	0.410	0.411	0.438
		2	0.449	0.444	0.454	0.453	0.464	0.424	0.418	0.438	0.429	0.461	0.439	0.441	0.442	0.449	0.453	0.406	0.407	0.410	0.426	0.435
		3	0.445	0.443	0.446	0.459	0.464	0.419	0.413	0.417	0.442	0.454	0.447	0.442	0.443	0.443	0.450	0.412	0.408	0.413	0.409	0.421
	SA	1	0.440	0.442	0.448	0.447	0.485	0.409	0.412	0.427	0.422	0.477	0.445	0.443	0.449	0.449	0.477	0.419	0.414	0.417	0.422	0.469
		2	0.443	0.442	0.440	0.448	0.478	0.407	0.408	0.405	0.420	0.463	0.445	0.443	0.441	0.444	0.458	0.412	0.407	0.408	0.414	0.434
		3	0.448	0.444	0.445	0.463	0.451	0.412	0.409	0.405	0.434	0.427	0.445	0.443	0.448	0.454	0.462	0.409	0.408	0.413	0.425	0.452
$\text{MLP}_{\text{TM}}^{64}$	$\text{MLP}_{\text{TM}}^{64}$	1	0.462	0.473	0.486	0.492	0.516	0.436	0.450	0.464	0.489	0.520	0.450	0.448	0.458	0.464	0.496	0.425	0.420	0.427	0.436	0.477
		2	0.456	0.451	0.461	0.484	0.500	0.427	0.423	0.441	0.463	0.502	0.450	0.445	0.445	0.466	0.477	0.420	0.414	0.411	0.443	0.454
		3	0.459	0.460	0.463	0.467	0.503	0.433	0.433	0.435	0.443	0.505	0.449	0.448	0.454	0.469	0.481	0.416	0.419	0.427	0.445	0.470
	No Mixing	1	0.448	0.453	0.455	0.464	0.489	0.423	0.433	0.440	0.457	0.519	0.441	0.441	0.440	0.456	0.456	0.410	0.414	0.410	0.437	0.438
		2	0.445	0.443	0.449	0.451	0.464	0.418	0.413	0.426	0.430	0.453	0.439	0.439	0.442	0.444	0.457	0.403	0.406	0.413	0.418	0.427
		3	0.443	0.444	0.442	0.448	0.451	0.410	0.416	0.415	0.424	0.432	0.445	0.442	0.440	0.448	0.454	0.412	0.406	0.406	0.417	0.422
	SA	1	0.441	0.443	0.445	0.447	0.482	0.409	0.413	0.419	0.424	0.465	0.443	0.446	0.446	0.454	0.468	0.411	0.418	0.419	0.429	0.453
		2	0.446	0.447	0.444	0.453	0.470	0.412	0.413	0.406	0.426	0.448	0.446	0.444	0.447	0.450	0.469	0.413	0.410	0.416	0.422	0.455
		3	0.445	0.444	0.447	0.448	0.457	0.408	0.407	0.414	0.416	0.430	0.447	0.446	0.442	0.447	0.463	0.410	0.410	0.404	0.416	0.449
No Mixing	$\text{MLP}_{\text{TM}}^{128}$	1	0.499	0.484	0.496	0.527	0.535	0.483	0.470	0.482	0.518	0.537	0.477	0.472	0.498	0.548	0.591	0.463	0.450	0.484	0.552	0.626
		2	0.486	0.503	0.507	0.540	0.521	0.474	0.495	0.497	0.540	0.515	0.476	0.484	0.513	0.537	0.529	0.456	0.467	0.513	0.539	0.531
		3	0.502	0.516	0.518	0.513	0.528	0.490	0.504	0.515	0.509	0.534	0.516	0.489	0.511	0.514	0.550	0.509	0.473	0.500	0.505	0.566
	$\text{MLP}_{\text{TM}}^{32}$	1	0.474	0.480	0.504	0.522	0.515	0.458	0.472	0.500	0.539	0.522	0.471	0.474	0.512	0.517	0.535	0.453	0.464	0.508	0.511	0.544
		2	0.474	0.502	0.492	0.521	0.516	0.458	0.487	0.473	0.528	0.516	0.474	0.467	0.492	0.517	0.544	0.460	0.447	0.481	0.532	0.561
		3	0.490	0.507	0.516	0.528	0.542	0.478	0.502	0.516	0.529	0.562	0.477	0.473	0.506	0.510	0.551	0.458	0.453	0.496	0.499	0.552
	$\text{MLP}_{\text{TM}}^{64}$	1	0.477	0.478	0.515	0.517	0.516	0.461	0.467	0.505	0.516	0.508	0.474	0.514	0.536	0.523	0.525	0.458	0.507	0.550	0.523	0.522
		2	0.487	0.499	0.516	0.508	0.544	0.476	0.481	0.507	0.494	0.549	0.478	0.471	0.500	0.562	0.547	0.459	0.448	0.481	0.586	0.562
		3	0.512	0.515	0.514	0.528	0.544	0.497	0.510	0.504	0.534	0.548	0.470	0.499	0.515	0.520	0.529	0.451	0.494	0.512	0.517	0.536
	No Mixing	1	0.470	0.468	0.467	0.473	0.482	0.453	0.453	0.454	0.466	0.473	0.452	0.447	0.450	0.452	0.457	0.425	0.421	0.427	0.431	0.441
		2	0.470	0.468	0.467	0.477	0.478	0.455	0.454	0.454	0.465	0.476	0.450	0.451	0.450	0.454	0.463	0.423	0.424	0.426	0.433	0.445
		3	0.467	0.466	0.466	0.473	0.480	0.451	0.452	0.454	0.466	0.475	0.455	0.449	0.450	0.451	0.454	0.429	0.423	0.426	0.429	0.433
	SA	1	0.460	0.456	0.459	0.463	0.472	0.436	0.434	0.440	0.445	0.460	0.458	0.453	0.457	0.458	0.470	0.433	0.428	0.433	0.434	0.460
		2	0.457	0.456	0.458	0.470	0.477	0.432	0.432	0.433	0.459	0.455	0.457	0.454	0.458	0.464	0.481	0.430	0.426	0.437	0.448	0.477
		3	0.452	0.455	0.458	0.467	0.481	0.424	0.429	0.437	0.447	0.467	0.457	0.455	0.457	0.463	0.484	0.428	0.430	0.432	0.440	0.475
SA	$\text{MLP}_{\text{TM}}^{128}$	1	0.461	0.476	0.493	0.540	0.529	0.439	0.461	0.481	0.562	0.526	0.463	0.478	0.514	0.543	0.561	0.439	0.450	0.507	0.542	0.578
		2	0.471	0.482	0.526	0.519	0.535	0.449	0.467	0.528	0.513	0.536	0.463	0.480	0.509	0.553	0.528	0.440	0.460	0.498	0.574	0.526
		3	0.479	0.519	0.538	0.528	0.523	0.456	0.515	0.546	0.542	0.518	0.468	0.510	0.530	0.566	0.534	0.446	0.503	0.525	0.593	0.537
	$\text{MLP}_{\text{TM}}^{32}$	1	0.458	0.450	0.463	0.504	0.514	0.437	0.426	0.438	0.497	0.505	0.454	0.465	0.471	0.475	0.504	0.430	0.436	0.444	0.450	0.495
		2	0.466	0.461	0.490	0.488	0.522	0.452	0.440	0.475	0.475	0.519	0.455	0.464	0.462	0.532	0.559	0.428	0.441	0.441	0.527	0.573
		3	0.463	0.474	0.486	0.541	0.542	0.444	0.455	0.474	0.565	0.548	0.463	0.457	0.500	0.529	0.505	0.440	0.432	0.501	0.533	0.495
	$\text{MLP}_{\text{TM}}^{64}$	1	0.466	0.466	0.515	0.538	0.535	0.449	0.448	0.506	0.555	0.548	0.455	0.450	0.465	0.503	0.520	0.430	0.424	0.436	0.483	0.514
		2	0.466	0.467	0.471	0.541	0.535	0.445	0.442	0.450	0.547	0.531	0.464	0.460	0.490	0.539	0.589	0.446	0.437	0.472	0.537	0.639
		3	0.478	0.477	0.491	0.511	0.508	0.463	0.459	0.473	0.494	0.501	0.470	0.477	0.523	0.526	0.517	0.448	0.457	0.529	0.530	0.523
	No Mixing	1	0.444	0.440	0.446	0.451	0.461	0.419	0.411	0.415	0.421	0.437	0.445	0.444	0.445	0.444	0.447	0.416	0.416	0.414	0.410	0.417
		2	0.444	0.439	0.440	0.450	0.448	0.415	0.411	0.407	0.419	0.421	0.448	0.442	0.448	0.450	0.463	0.421	0.413	0.417	0.422	0.441
		3	0.440	0.438	0.445	0.443	0.448	0.409	0.406	0.414	0.414	0.422	0.446	0.442	0.449	0.443	0.449	0.419	0.411	0.419	0.411	0.424
	SA	1	0.443	0.439	0.444	0.458	0.495	0.414	0.408	0.413	0.428	0.488	0.444	0.448	0.482	0.452	0.531	0.416	0.418	0.454	0.424	0.536
		2	0.446	0.442	0.453	0.465	0.476	0.415	0.409	0.423	0.435	0.456	0.446	0.452	0.455	0.477	0.484	0.419	0.424	0.423	0.457	0.468
		3	0.450	0.445	0.463	0.486	0.551	0.420	0.413	0.433	0.469	0.586	0.454	0.449	0.473	0.481	0.521	0.428	0.419	0.450	0.461	0.519

Table A.5: Raw experiment results for $L_{\text{in}} = 512$ and $T = 96$ on ETTh1 dataset.

		TP	No Processing										TokenMLP									
		Metric	MAE					MSE					MAE					MSE				
		d	32	64	128	256	512	32	64	128	256	512	32	64	128	256	512	32	64	128	256	512
VM	TM	n_{layers}																				
		1	0.503	0.503	0.544	0.564	0.565	0.499	0.497	0.555	0.582	0.583	0.498	0.524	0.549	0.555	0.568	0.486	0.525	0.562	0.559	0.593
	$\text{MLP}^{128}_{\text{TM}}$	2	0.498	0.516	0.513	0.532	0.538	0.488	0.522	0.507	0.532	0.547	0.494	0.510	0.524	0.544	0.551	0.482	0.502	0.526	0.553	0.573
		3	0.504	0.532	0.524	0.541	0.557	0.496	0.539	0.528	0.557	0.587	0.487	0.511	0.524	0.540	0.546	0.467	0.505	0.516	0.556	0.572
		1	0.474	0.474	0.481	0.488	0.495	0.461	0.467	0.473	0.498	0.508	0.469	0.469	0.471	0.474	0.491	0.448	0.450	0.456	0.468	0.485
$\text{MLP}^{128}_{\text{TM}}$	No Mixing	2	0.471	0.472	0.476	0.475	0.488	0.457	0.458	0.460	0.457	0.480	0.474	0.471	0.476	0.481	0.485	0.453	0.447	0.455	0.467	0.473
		3	0.470	0.469	0.470	0.474	0.478	0.450	0.450	0.449	0.457	0.454	0.470	0.475	0.478	0.485	0.496	0.446	0.454	0.459	0.463	0.480
		1	0.477	0.476	0.476	0.481	0.490	0.457	0.457	0.459	0.466	0.488	0.473	0.475	0.475	0.498	0.528	0.449	0.455	0.461	0.496	0.550
	SA	2	0.475	0.476	0.478	0.483	0.515	0.451	0.453	0.456	0.473	0.522	0.475	0.477	0.479	0.475	0.545	0.450	0.460	0.462	0.453	0.572
		3	0.479	0.474	0.477	0.493	0.533	0.456	0.450	0.456	0.484	0.550	0.478	0.475	0.499	0.495	0.531	0.454	0.453	0.502	0.494	0.543
		1	0.499	0.499	0.515	0.523	0.553	0.492	0.502	0.523	0.543	0.579	0.494	0.486	0.511	0.539	0.551	0.484	0.473	0.519	0.541	0.574
	$\text{MLP}^{32}_{\text{TM}}$	2	0.492	0.496	0.512	0.542	0.539	0.486	0.490	0.511	0.550	0.551	0.487	0.479	0.517	0.529	0.547	0.473	0.461	0.510	0.529	0.560
		3	0.495	0.493	0.508	0.525	0.546	0.489	0.484	0.511	0.533	0.546	0.480	0.488	0.489	0.507	0.542	0.465	0.480	0.474	0.496	0.545
		1	0.480	0.482	0.487	0.492	0.506	0.476	0.481	0.494	0.503	0.526	0.468	0.466	0.470	0.475	0.500	0.448	0.448	0.451	0.465	0.504
$\text{MLP}^{32}_{\text{TM}}$	No Mixing	2	0.476	0.479	0.475	0.487	0.501	0.461	0.469	0.459	0.489	0.508	0.473	0.465	0.473	0.483	0.485	0.449	0.443	0.457	0.469	0.476
		3	0.483	0.472	0.476	0.483	0.485	0.467	0.456	0.468	0.474	0.491	0.473	0.474	0.480	0.480	0.497	0.450	0.449	0.455	0.461	0.490
		1	0.475	0.473	0.479	0.475	0.508	0.455	0.457	0.464	0.462	0.517	0.474	0.475	0.477	0.492	0.495	0.455	0.459	0.464	0.487	0.495
	SA	2	0.476	0.476	0.483	0.485	0.511	0.450	0.450	0.461	0.468	0.514	0.473	0.478	0.472	0.490	0.490	0.450	0.461	0.456	0.479	0.485
		3	0.477	0.476	0.486	0.478	0.497	0.451	0.454	0.471	0.462	0.501	0.479	0.480	0.480	0.483	0.503	0.456	0.459	0.460	0.470	0.510
		1	0.499	0.501	0.509	0.535	0.572	0.501	0.497	0.522	0.561	0.624	0.494	0.501	0.552	0.534	0.544	0.486	0.493	0.559	0.544	0.553
	$\text{MLP}^{64}_{\text{TM}}$	2	0.492	0.520	0.535	0.532	0.531	0.487	0.521	0.535	0.532	0.536	0.492	0.492	0.523	0.543	0.563	0.485	0.480	0.520	0.557	0.587
		3	0.492	0.502	0.511	0.531	0.519	0.483	0.501	0.501	0.546	0.505	0.488	0.492	0.539	0.537	0.556	0.472	0.477	0.553	0.546	0.594
		1	0.474	0.483	0.489	0.500	0.494	0.461	0.480	0.492	0.531	0.507	0.467	0.466	0.473	0.473	0.487	0.447	0.446	0.457	0.456	0.484
$\text{MLP}^{64}_{\text{TM}}$	No Mixing	2	0.473	0.470	0.472	0.488	0.488	0.455	0.458	0.458	0.497	0.493	0.471	0.468	0.475	0.483	0.490	0.446	0.447	0.451	0.463	0.480
		3	0.473	0.472	0.479	0.478	0.495	0.457	0.454	0.467	0.464	0.510	0.476	0.474	0.480	0.478	0.481	0.451	0.454	0.457	0.462	0.467
		1	0.473	0.475	0.476	0.489	0.520	0.455	0.460	0.461	0.487	0.536	0.473	0.481	0.475	0.477	0.520	0.453	0.470	0.458	0.465	0.527
	SA	2	0.477	0.476	0.475	0.476	0.510	0.453	0.452	0.454	0.456	0.514	0.475	0.476	0.479	0.476	0.502	0.453	0.453	0.462	0.457	0.496
		3	0.478	0.479	0.478	0.484	0.499	0.455	0.455	0.461	0.469	0.493	0.481	0.477	0.474	0.482	0.500	0.461	0.453	0.450	0.464	0.499
		1	0.541	0.541	0.586	0.569	0.556	0.573	0.551	0.614	0.578	0.562	0.526	0.539	0.588	0.618	0.633	0.542	0.544	0.640	0.671	0.694
	$\text{MLP}^{128}_{\text{TM}}$	2	0.541	0.549	0.569	0.586	0.605	0.560	0.547	0.586	0.614	0.655	0.534	0.556	0.623	0.614	0.621	0.543	0.583	0.690	0.663	0.690
		3	0.550	0.589	0.575	0.608	0.595	0.572	0.619	0.597	0.661	0.629	0.535	0.555	0.628	0.611	0.634	0.545	0.578	0.696	0.656	0.714
		1	0.523	0.526	0.595	0.621	0.610	0.533	0.544	0.678	0.699	0.673	0.505	0.532	0.550	0.597	0.645	0.508	0.549	0.573	0.635	0.728
	$\text{MLP}^{32}_{\text{TM}}$	2	0.539	0.545	0.564	0.618	0.585	0.559	0.575	0.598	0.695	0.632	0.531	0.543	0.542	0.638	0.652	0.550	0.570	0.553	0.739	0.748
		3	0.553	0.539	0.617	0.577	0.673	0.593	0.561	0.672	0.605	0.802	0.509	0.514	0.564	0.590	0.684	0.510	0.518	0.604	0.633	0.829
		1	0.520	0.524	0.574	0.584	0.593	0.534	0.535	0.594	0.619	0.638	0.512	0.514	0.621	0.597	0.609	0.516	0.513	0.684	0.627	0.664
No Mixing	$\text{MLP}^{64}_{\text{TM}}$	2	0.534	0.551	0.572	0.589	0.568	0.555	0.560	0.590	0.630	0.575	0.528	0.544	0.569	0.603	0.610	0.537	0.558	0.596	0.649	0.660
		3	0.556	0.558	0.581	0.589	0.633	0.580	0.567	0.616	0.632	0.719	0.516	0.664	0.565	0.585	0.613	0.521	0.800	0.591	0.617	0.645
		1	0.489	0.490	0.492	0.500	0.500	0.485	0.486	0.492	0.505	0.504	0.480	0.476	0.474	0.496	0.490	0.466	0.464	0.461	0.492	0.487
	No Mixing	2	0.491	0.490	0.492	0.492	0.498	0.492	0.489	0.489	0.493	0.502	0.482	0.477	0.477	0.480	0.482	0.469	0.464	0.464	0.470	0.475
		3	0.491	0.491	0.489	0.496	0.503	0.488	0.488	0.488	0.497	0.506	0.477	0.474	0.478	0.478	0.485	0.463	0.461	0.466	0.471	0.482
		1	0.484	0.484	0.487	0.501	0.496	0.474	0.472	0.481	0.508	0.503	0.483	0.489	0.485	0.494	0.494	0.472	0.481	0.477	0.489	0.497
	SA	2	0.482	0.482	0.484	0.500	0.508	0.470	0.471	0.472	0.503	0.516	0.483	0.483	0.483	0.498	0.510	0.469	0.472	0.472	0.503	0.517
		3	0.484	0.484	0.502	0.510	0.515	0.474	0.470	0.498	0.519	0.530	0.484	0.483	0.484	0.492	0.508	0.469	0.468	0.472	0.488	0.512
		1	0.511	0.525	0.609	0.581	0.602	0.515	0.523	0.653	0.607	0.647	0.518	0.530	0.539	0.596	0.612	0.515	0.524	0.532	0.624	0.657
	$\text{MLP}^{128}_{\text{TM}}$	2	0.523	0.558	0.595	0.646	0.564	0.514	0.570	0.628	0.721	0.578	0.509	0.625	0.575	0.628	0.591	0.496	0.684	0.604	0.684	0.634
		3	0.582	0.611	0.614	0.610	0.587	0.613	0.674	0.660	0.646	0.624	0.531	0.570	0.570	0.597	0.566	0.530	0.591	0.603	0.632	0.606
		1	0.497	0.507	0.524	0.560	0.513	0.494	0.503	0.525	0.571	0.507	0.497	0.498	0.524	0.554	0.589	0.494	0.491	0.530	0.572	0.615
	$\text{MLP}^{32}_{\text{TM}}$	2	0.527	0.495	0.573	0.547	0.572	0.533	0.483	0.601	0.572	0.589	0.537	0.488	0.571	0.615	0.566	0.558	0.479	0.611	0.654	0.607
		3	0.505	0.489	0.581	0.637	0.597	0.500	0.478	0.610	0.722	0.642	0.509	0.500	0.572	0.634	0.541	0.506	0.495	0.603	0.697	0.554
		1	0.520	0.501	0.534	0.581	0.590	0.537	0.494	0.534	0.611	0.628	0.513	0.503	0.514	0.597	0.638	0.518	0.490	0.503	0.634	0.736
SA	$\text{MLP}^{64}_{\text{TM}}$	2	0.503	0.525	0.631	0.603	0.556	0.497	0.529	0.702	0.637	0.565	0.506	0.535	0.584	0.566	0.599	0.499	0.533	0.615	0.580	0.650
		3	0.490	0.546	0.536	0.636	0.647	0.472	0.549	0.524	0.722	0.717	0.513	0.573	0.628	0.568	0.589	0.507	0.600	0.690	0.599	0.629
		1	0.471	0.471	0.473	0.478	0.495	0.456	0.454	0.457	0.459	0.485	0.473	0.472	0.474	0.499	0.497	0.458	0.455	0.460	0.489	0.490
	No Mixing	2	0.471	0.472	0.479	0.485	0.496	0.454	0.454	0.462	0.472	0.490	0.477	0.475	0.479	0.481	0.497	0.466	0.459	0.461	0.462	0.485
		3	0.469	0.475	0.473	0.479	0.484	0.451	0.456	0.454	0.464	0.470	0.475	0.478	0.493	0.480	0.487	0.461	0.460	0.480	0.458	0.477
		1	0.476	0.469	0.471	0.493	0.546	0.463	0.451	0.454	0.484	0.587	0.477	0.483	0.484	0.532	0.531	0.462	0.472	0.467	0.538	0.561
	SA	2	0.471	0.499	0.499	0.529	0.582	0.452	0.488	0.490	0.522	0.619	0.475	0.488	0.539	0.565	0.543	0.461	0.474	0.542	0.587	0.561
		3	0.475	0.473	0.500	0.521	0.572	0.457	0.451	0.481	0.501	0.607	0.496	0.482	0.541	0.608	0.570	0.490	0.469	0.549	0.657	0.612

Table A.6: Raw experiment results for $L_{\text{in}} = 512$ and $T = 192$ on ETTh1 dataset.

		TP	No Processing										TokenMLP									
		Metric	MAE					MSE					MAE					MSE				
		d	32	64	128	256	512	32	64	128	256	512	32	64	128	256	512	32	64	128	256	512
VM	TM	m_{layers}																				
	$\text{MLP}_{\text{TM}}^{128}$	1	0.546	0.564	0.611	0.647	0.622	0.561	0.599	0.678	0.731	0.685	0.562	0.639	0.623	0.639	0.656	0.589	0.721	0.691	0.716	0.756
		2	0.553	0.583	0.594	0.592	0.597	0.577	0.620	0.624	0.633	0.631	0.560	0.589	0.598	0.616	0.599	0.581	0.642	0.648	0.680	0.657
		3	0.554	0.562	0.583	0.581	0.597	0.575	0.587	0.621	0.612	0.642	0.558	0.564	0.579	0.597	0.581	0.583	0.586	0.623	0.662	0.627
	No Mixing	1	0.496	0.503	0.504	0.511	0.522	0.489	0.505	0.517	0.520	0.548	0.498	0.494	0.501	0.510	0.537	0.487	0.481	0.495	0.513	0.567
$\text{MLP}_{\text{TM}}^{128}$		2	0.496	0.500	0.495	0.496	0.506	0.488	0.495	0.487	0.489	0.502	0.501	0.505	0.520	0.525	0.552	0.490	0.496	0.515	0.525	0.569
		3	0.495	0.494	0.502	0.515	0.521	0.484	0.488	0.496	0.520	0.533	0.500	0.507	0.514	0.521	0.561	0.483	0.494	0.502	0.522	0.584
	SA	1	0.508	0.510	0.509	0.513	0.521	0.497	0.504	0.507	0.520	0.535	0.507	0.502	0.505	0.514	0.541	0.496	0.494	0.503	0.518	0.573
		2	0.506	0.517	0.514	0.514	0.527	0.494	0.515	0.514	0.517	0.540	0.512	0.508	0.517	0.525	0.538	0.504	0.499	0.523	0.543	0.554
		3	0.520	0.516	0.513	0.552	0.574	0.520	0.511	0.511	0.574	0.614	0.512	0.518	0.519	0.524	0.546	0.503	0.517	0.523	0.535	0.569
	$\text{MLP}_{\text{TM}}^{32}$	1	0.532	0.547	0.557	0.570	0.632	0.546	0.570	0.587	0.627	0.706	0.542	0.541	0.577	0.626	0.647	0.564	0.558	0.635	0.693	0.732
		2	0.543	0.524	0.588	0.572	0.572	0.569	0.535	0.633	0.585	0.587	0.541	0.564	0.561	0.578	0.595	0.554	0.598	0.582	0.607	0.645
		3	0.540	0.558	0.544	0.609	0.601	0.561	0.591	0.569	0.679	0.655	0.530	0.566	0.572	0.558	0.592	0.538	0.600	0.593	0.580	0.638
	No Mixing	1	0.501	0.501	0.501	0.513	0.526	0.498	0.501	0.501	0.513	0.548	0.495	0.496	0.501	0.511	0.515	0.485	0.489	0.496	0.514	0.524
$\text{MLP}_{\text{TM}}^{32}$		2	0.497	0.501	0.502	0.509	0.515	0.487	0.497	0.501	0.511	0.529	0.501	0.504	0.518	0.511	0.533	0.488	0.491	0.520	0.505	0.547
		3	0.500	0.509	0.506	0.506	0.526	0.488	0.502	0.503	0.502	0.551	0.499	0.513	0.514	0.529	0.542	0.485	0.505	0.503	0.533	0.558
	SA	1	0.504	0.510	0.507	0.513	0.571	0.495	0.505	0.508	0.515	0.592	0.506	0.515	0.507	0.504	0.536	0.496	0.518	0.503	0.509	0.561
		2	0.511	0.504	0.505	0.526	0.549	0.502	0.492	0.492	0.529	0.570	0.509	0.510	0.518	0.527	0.561	0.500	0.506	0.516	0.537	0.609
		3	0.506	0.506	0.517	0.522	0.536	0.494	0.495	0.514	0.525	0.552	0.509	0.515	0.512	0.538	0.569	0.500	0.508	0.516	0.562	0.622
	$\text{MLP}_{\text{TM}}^{64}$	1	0.538	0.557	0.578	0.643	0.648	0.552	0.589	0.619	0.733	0.725	0.555	0.571	0.605	0.646	0.651	0.579	0.610	0.668	0.750	0.760
		2	0.534	0.554	0.591	0.572	0.593	0.547	0.580	0.625	0.593	0.634	0.555	0.579	0.584	0.607	0.619	0.583	0.609	0.626	0.666	0.690
		3	0.544	0.543	0.578	0.567	0.591	0.568	0.557	0.611	0.590	0.626	0.535	0.564	0.567	0.567	0.573	0.537	0.586	0.592	0.600	0.609
	No Mixing	1	0.497	0.498	0.510	0.503	0.517	0.493	0.493	0.527	0.510	0.531	0.495	0.500	0.507	0.506	0.526	0.482	0.492	0.501	0.505	0.544
$\text{MLP}_{\text{TM}}^{64}$		2	0.495	0.492	0.492	0.507	0.512	0.486	0.484	0.481	0.507	0.532	0.497	0.498	0.513	0.507	0.528	0.485	0.486	0.510	0.502	0.539
		3	0.499	0.500	0.495	0.511	0.524	0.486	0.496	0.484	0.519	0.535	0.503	0.499	0.513	0.519	0.523	0.486	0.485	0.506	0.519	0.527
	SA	1	0.510	0.506	0.505	0.512	0.564	0.506	0.500	0.498	0.520	0.582	0.508	0.502	0.505	0.516	0.550	0.498	0.492	0.497	0.523	0.563
		2	0.505	0.514	0.513	0.535	0.532	0.493	0.510	0.514	0.552	0.555	0.505	0.513	0.521	0.526	0.532	0.491	0.504	0.530	0.546	0.554
		3	0.516	0.511	0.519	0.522	0.568	0.509	0.504	0.524	0.524	0.604	0.513	0.523	0.517	0.534	0.581	0.508	0.524	0.521	0.551	0.638
	$\text{MLP}_{\text{TM}}^{128}$	1	0.588	0.595	0.647	0.659	0.644	0.644	0.648	0.712	0.743	0.718	0.586	0.625	0.618	0.656	0.713	0.622	0.702	0.670	0.726	0.857
		2	0.607	0.631	0.654	0.669	0.686	0.676	0.685	0.736	0.779	0.808	0.599	0.642	0.632	0.667	0.715	0.661	0.732	0.706	0.759	0.859
		3	0.653	0.622	0.657	0.663	0.701	0.741	0.670	0.741	0.773	0.837	0.611	0.629	0.652	0.632	0.688	0.680	0.694	0.732	0.694	0.804
	$\text{MLP}_{\text{TM}}^{32}$	1	0.598	0.592	0.617	0.635	0.636	0.668	0.638	0.693	0.707	0.711	0.569	0.592	0.614	0.745	0.674	0.604	0.637	0.670	0.935	0.773
		2	0.593	0.605	0.622	0.642	0.642	0.651	0.670	0.677	0.722	0.721	0.577	0.567	0.609	0.676	0.675	0.609	0.585	0.683	0.805	0.788
		3	0.595	0.627	0.658	0.667	0.703	0.648	0.695	0.757	0.765	0.874	0.566	0.597	0.682	0.650	0.750	0.595	0.639	0.800	0.749	0.949
	$\text{MLP}_{\text{TM}}^{64}$	1	0.577	0.599	0.654	0.637	0.645	0.622	0.663	0.728	0.706	0.704	0.570	0.608	0.610	0.649	0.671	0.594	0.667	0.640	0.735	0.770
No Mixing		2	0.586	0.623	0.636	0.675	0.668	0.634	0.682	0.698	0.777	0.783	0.594	0.634	0.615	0.634	0.679	0.631	0.700	0.666	0.699	0.791
		3	0.584	0.629	0.637	0.672	0.675	0.620	0.701	0.695	0.800	0.791	0.596	0.597	0.630	0.683	0.701	0.647	0.644	0.691	0.801	0.823
	No Mixing	1	0.517	0.515	0.515	0.515	0.541	0.519	0.517	0.521	0.522	0.569	0.506	0.501	0.508	0.505	0.521	0.503	0.498	0.510	0.504	0.537
		2	0.515	0.514	0.518	0.531	0.525	0.516	0.518	0.525	0.556	0.541	0.506	0.511	0.504	0.507	0.527	0.502	0.512	0.502	0.508	0.541
		3	0.515	0.511	0.514	0.517	0.525	0.518	0.515	0.519	0.526	0.538	0.505	0.505	0.503	0.503	0.511	0.502	0.501	0.501	0.503	0.519
	SA	1	0.513	0.516	0.524	0.519	0.532	0.512	0.519	0.528	0.530	0.554	0.510	0.517	0.519	0.514	0.529	0.506	0.517	0.524	0.512	0.549
		2	0.512	0.513	0.520	0.553	0.539	0.509	0.512	0.530	0.591	0.573	0.513	0.512	0.512	0.517	0.605	0.509	0.512	0.512	0.519	0.699
		3	0.518	0.510	0.519	0.541	0.557	0.516	0.508	0.523	0.566	0.595	0.514	0.519	0.534	0.548	0.567	0.512	0.524	0.551	0.572	0.611
	$\text{MLP}_{\text{TM}}^{128}$	1	0.556	0.624	0.660	0.660	0.616	0.566	0.668	0.737	0.754	0.654	0.571	0.627	0.636	0.629	0.622	0.588	0.671	0.697	0.677	0.677
		2	0.608	0.606	0.622	0.658	0.625	0.647	0.638	0.664	0.743	0.666	0.583	0.623	0.636	0.662	0.606	0.619	0.679	0.698	0.752	0.644
		3	0.601	0.625	0.635	0.627	0.654	0.646	0.667	0.690	0.677	0.742	0.614	0.629	0.651	0.608	0.631	0.658	0.672	0.730	0.647	0.701
	$\text{MLP}_{\text{TM}}^{32}$	1	0.564	0.561	0.613	0.677	0.672	0.600	0.581	0.663	0.767	0.790	0.544	0.559	0.650	0.682	0.638	0.562	0.571	0.728	0.778	0.715
		2	0.564	0.551	0.657	0.652	0.653	0.597	0.555	0.755	0.731	0.722	0.568	0.604	0.614	0.592	0.678	0.581	0.638	0.650	0.631	0.786
		3	0.560	0.642	0.600	0.657	0.615	0.587	0.709	0.650	0.722	0.667	0.557	0.571	0.662	0.638	0.623	0.575	0.595	0.751	0.704	0.681
	$\text{MLP}_{\text{TM}}^{64}$	1	0.551	0.579	0.630	0.595	0.645	0.566	0.620	0.685	0.625	0.706	0.558	0.571	0.623	0.628	0.674	0.578	0.587	0.666	0.670	0.763
SA		2	0.555	0.662	0.653	0.694	0.623	0.561	0.748	0.715	0.818	0.679	0.572	0.600	0.667	0.613	0.671	0.592	0.639	0.750	0.655	0.769
		3	0.639	0.615	0.606	0.620	0.640	0.707	0.666	0.639	0.681	0.708	0.598	0.621	0.642	0.620	0.651	0.638	0.688	0.719	0.664	0.730
	No Mixing	1	0.502	0.500	0.515	0.526	0.511	0.498	0.496	0.515	0.534	0.510	0.502	0.514	0.526	0.534	0.518	0.499	0.510	0.523	0.537	0.511
		2	0.495	0.504	0.510	0.514	0.524	0.490	0.498	0.505	0.509	0.521	0.506	0.518	0.524	0.541	0.518	0.501	0.512	0.520	0.545	0.514
		3	0.498	0.497	0.508	0.518	0.540	0.496	0.491	0.503	0.515	0.559	0.510	0.507	0.540	0.524	0.522	0.506	0.502	0.544	0.520	0.533
	SA	1	0.502	0.531	0.586	0.602	0.589	0.496	0.533	0.610	0.633	0.653	0.507	0.584	0.616	0.675	0.673	0.501	0.615	0.650	0.783	0.776
		2	0.510	0.523	0.604	0.613	0.617	0.502	0.518	0.624	0.668	0.697	0.511	0.610	0.636	0.577	0.641	0.506	0.674	0.713	0.614	0.736
		3	0.506	0.510	0.643	0.613	0.673	0.501	0.503	0.715	0.664	0.805	0.510	0.505	0.665	0.543	0.603	0.508	0.496	0.766	0.550	0.648

Table A.7: Raw experiment results for $L_{\text{in}} = 512$ and $T = 336$ on ETTh1 dataset.

		TP	No Processing										TokenMLP									
		Metric	MAE					MSE					MAE					MSE				
		d	32	64	128	256	512	32	64	128	256	512	32	64	128	256	512	32	64	128	256	512
VM	TM	n_{layers}																				
		1	0.578	0.639	0.655	0.681	0.692	0.615	0.735	0.746	0.818	0.827	0.616	0.706	0.694	0.743	0.777	0.674	0.860	0.823	0.944	1.012
	$\text{MLP}_{\text{TM}}^{128}$	2	0.589	0.662	0.651	0.645	0.622	0.641	0.755	0.741	0.712	0.673	0.617	0.681	0.655	0.647	0.657	0.690	0.798	0.738	0.728	0.752
		3	0.604	0.624	0.644	0.607	0.610	0.659	0.711	0.722	0.662	0.664	0.622	0.654	0.637	0.618	0.621	0.685	0.771	0.728	0.693	0.692
		1	0.523	0.523	0.524	0.526	0.539	0.530	0.527	0.539	0.543	0.562	0.529	0.533	0.536	0.549	0.566	0.528	0.540	0.550	0.564	0.602
$\text{MLP}_{\text{TM}}^{128}$	No Mixing	2	0.527	0.522	0.521	0.525	0.531	0.529	0.525	0.520	0.529	0.536	0.529	0.531	0.568	0.563	0.564	0.528	0.534	0.594	0.585	0.590
		3	0.523	0.534	0.536	0.541	0.563	0.520	0.546	0.550	0.561	0.607	0.530	0.538	0.550	0.598	0.595	0.529	0.542	0.558	0.639	0.633
		1	0.533	0.539	0.548	0.561	0.569	0.530	0.551	0.567	0.594	0.619	0.539	0.540	0.535	0.553	0.617	0.542	0.545	0.547	0.586	0.676
	SA	2	0.540	0.542	0.555	0.557	0.560	0.550	0.555	0.576	0.580	0.586	0.532	0.546	0.562	0.557	0.568	0.529	0.559	0.604	0.583	0.603
		3	0.544	0.560	0.560	0.576	0.586	0.552	0.591	0.594	0.610	0.628	0.532	0.551	0.560	0.562	0.612	0.536	0.570	0.591	0.598	0.681
		1	0.561	0.589	0.595	0.680	0.684	0.601	0.642	0.664	0.800	0.838	0.582	0.607	0.671	0.688	0.719	0.637	0.668	0.797	0.829	0.894
	$\text{MLP}_{\text{TM}}^{32}$	2	0.561	0.605	0.609	0.609	0.630	0.597	0.675	0.687	0.658	0.706	0.567	0.611	0.632	0.666	0.631	0.603	0.690	0.712	0.783	0.710
		3	0.576	0.583	0.618	0.588	0.628	0.619	0.627	0.705	0.634	0.711	0.592	0.624	0.587	0.611	0.618	0.642	0.703	0.623	0.682	0.686
		1	0.529	0.529	0.536	0.530	0.547	0.536	0.533	0.555	0.548	0.577	0.528	0.527	0.525	0.550	0.537	0.530	0.533	0.530	0.567	0.552
$\text{MLP}_{\text{TM}}^{32}$	No Mixing	2	0.523	0.534	0.533	0.539	0.537	0.529	0.546	0.539	0.560	0.554	0.542	0.529	0.538	0.567	0.580	0.545	0.530	0.539	0.582	0.615
		3	0.527	0.546	0.540	0.559	0.537	0.526	0.565	0.563	0.593	0.559	0.539	0.551	0.555	0.588	0.566	0.538	0.553	0.569	0.623	0.590
		1	0.538	0.527	0.537	0.562	0.555	0.545	0.529	0.545	0.589	0.575	0.535	0.535	0.532	0.552	0.551	0.537	0.539	0.541	0.587	0.576
	SA	2	0.534	0.539	0.537	0.553	0.573	0.536	0.544	0.538	0.569	0.627	0.561	0.547	0.549	0.559	0.603	0.589	0.560	0.566	0.599	0.686
		3	0.540	0.550	0.575	0.562	0.659	0.551	0.564	0.609	0.601	0.796	0.544	0.542	0.572	0.585	0.636	0.557	0.559	0.608	0.630	0.763
		1	0.575	0.614	0.676	0.686	0.713	0.621	0.687	0.799	0.819	0.870	0.587	0.669	0.703	0.714	0.709	0.646	0.775	0.848	0.864	0.856
	$\text{MLP}_{\text{TM}}^{64}$	2	0.577	0.593	0.637	0.605	0.618	0.619	0.649	0.738	0.660	0.694	0.583	0.652	0.681	0.653	0.637	0.625	0.753	0.794	0.759	0.709
		3	0.574	0.598	0.625	0.610	0.607	0.616	0.659	0.709	0.658	0.654	0.600	0.649	0.628	0.617	0.618	0.655	0.758	0.724	0.685	0.682
		1	0.529	0.522	0.525	0.531	0.533	0.535	0.526	0.532	0.549	0.551	0.525	0.534	0.529	0.548	0.547	0.523	0.543	0.534	0.560	0.567
$\text{MLP}_{\text{TM}}^{64}$	No Mixing	2	0.523	0.523	0.536	0.528	0.545	0.525	0.525	0.552	0.535	0.571	0.545	0.529	0.533	0.550	0.573	0.549	0.527	0.534	0.563	0.600
		3	0.524	0.528	0.539	0.543	0.559	0.528	0.531	0.559	0.559	0.604	0.540	0.549	0.555	0.592	0.603	0.538	0.553	0.564	0.629	0.652
		1	0.537	0.531	0.535	0.537	0.626	0.533	0.530	0.544	0.545	0.680	0.533	0.539	0.534	0.532	0.610	0.533	0.544	0.538	0.547	0.661
	SA	2	0.534	0.543	0.542	0.555	0.573	0.533	0.550	0.551	0.584	0.610	0.532	0.539	0.561	0.573	0.611	0.532	0.551	0.595	0.628	0.683
		3	0.555	0.548	0.552	0.575	0.603	0.577	0.561	0.575	0.627	0.661	0.566	0.551	0.561	0.577	0.602	0.599	0.567	0.607	0.623	0.659
		1	0.616	0.693	0.724	0.702	0.686	0.676	0.813	0.884	0.824	0.811	0.655	0.732	0.701	0.777	0.779	0.747	0.894	0.839	0.988	1.011
	$\text{MLP}_{\text{TM}}^{128}$	2	0.627	0.685	0.729	0.698	0.733	0.689	0.789	0.914	0.827	0.904	0.643	0.735	0.737	0.713	0.736	0.715	0.906	0.910	0.858	0.903
		3	0.631	0.736	0.721	0.727	0.726	0.693	0.909	0.884	0.881	0.899	0.722	0.715	0.751	0.746	0.746	0.861	0.865	0.981	0.926	0.945
		1	0.640	0.653	0.712	0.661	0.687	0.734	0.766	0.869	0.759	0.809	0.606	0.647	0.686	0.725	0.733	0.666	0.729	0.817	0.870	0.899
	$\text{MLP}_{\text{TM}}^{32}$	2	0.633	0.652	0.660	0.686	0.719	0.743	0.744	0.763	0.800	0.911	0.609	0.726	0.727	0.720	0.717	0.674	0.869	0.894	0.885	0.864
		3	0.661	0.621	0.668	0.706	0.696	0.778	0.684	0.770	0.872	0.853	0.606	0.634	0.735	0.688	0.729	0.668	0.728	0.901	0.817	0.872
		1	0.612	0.636	0.710	0.712	0.748	0.682	0.726	0.864	0.870	0.963	0.627	0.646	0.716	0.747	0.716	0.699	0.738	0.864	0.940	0.867
No Mixing	$\text{MLP}_{\text{TM}}^{64}$	2	0.639	0.675	0.708	0.666	0.768	0.739	0.811	0.855	0.755	0.997	0.639	0.700	0.748	0.762	0.729	0.717	0.832	0.927	0.981	0.887
		3	0.639	0.687	0.742	0.676	0.702	0.715	0.803	0.930	0.781	0.838	0.618	0.645	0.708	0.733	0.713	0.681	0.710	0.858	0.919	0.884
		1	0.539	0.537	0.534	0.535	0.543	0.549	0.550	0.544	0.548	0.559	0.535	0.534	0.535	0.534	0.542	0.544	0.547	0.551	0.546	0.565
	No Mixing	2	0.537	0.534	0.537	0.538	0.552	0.548	0.544	0.552	0.554	0.578	0.536	0.532	0.531	0.532	0.540	0.547	0.542	0.542	0.542	0.562
		3	0.534	0.535	0.535	0.538	0.544	0.542	0.546	0.546	0.552	0.566	0.536	0.537	0.532	0.534	0.531	0.545	0.550	0.542	0.548	0.544
		1	0.543	0.545	0.544	0.555	0.548	0.557	0.560	0.560	0.582	0.566	0.543	0.542	0.555	0.594	0.555	0.555	0.559	0.583	0.654	0.583
	SA	2	0.547	0.545	0.552	0.559	0.587	0.559	0.560	0.574	0.588	0.642	0.542	0.539	0.538	0.611	0.732	0.555	0.549	0.551	0.680	0.948
		3	0.542	0.548	0.561	0.554	0.569	0.550	0.567	0.593	0.578	0.603	0.544	0.539	0.641	0.748	0.869	0.558	0.550	0.734	0.993	1.277
		1	0.605	0.666	0.676	0.658	0.666	0.654	0.757	0.774	0.745	0.755	0.651	0.655	0.678	0.697	0.647	0.739	0.736	0.776	0.805	0.715
	$\text{MLP}_{\text{TM}}^{128}$	2	0.611	0.691	0.686	0.682	0.674	0.660	0.813	0.804	0.767	0.772	0.703	0.685	0.683	0.700	0.703	0.826	0.794	0.784	0.834	0.838
		3	0.676	0.693	0.744	0.708	0.671	0.771	0.827	0.967	0.879	0.780	0.673	0.739	0.720	0.684	0.663	0.777	0.915	0.872	0.797	0.755
		1	0.608	0.605	0.634	0.696	0.639	0.677	0.653	0.710	0.786	0.724	0.619	0.649	0.665	0.642	0.624	0.690	0.731	0.756	0.715	0.678
	$\text{MLP}_{\text{TM}}^{32}$	2	0.597	0.614	0.643	0.622	0.694	0.650	0.679	0.728	0.685	0.812	0.606	0.611	0.717	0.686	0.655	0.655	0.677	0.867	0.800	0.739
		3	0.621	0.635	0.651	0.653	0.671	0.669	0.689	0.734	0.722	0.752	0.642	0.656	0.676	0.648	0.664	0.696	0.746	0.794	0.741	0.769
		1	0.614	0.661	0.635	0.692	0.723	0.685	0.745	0.707	0.821	0.885	0.601	0.665	0.662	0.734	0.680	0.645	0.754	0.737	0.896	0.800
SA	$\text{MLP}_{\text{TM}}^{64}$	2	0.608	0.653	0.671	0.679	0.658	0.654	0.724	0.784	0.761	0.748	0.674	0.653	0.707	0.745	0.645	0.778	0.725	0.848	0.933	0.712
		3	0.606	0.724	0.734	0.720	0.686	0.645	0.907	0.922	0.889	0.805	0.667	0.662	0.723	0.717	0.692	0.760	0.764	0.904	0.887	0.814
		1	0.528	0.538	0.532	0.541	0.559	0.536	0.550	0.541	0.550	0.585	0.535	0.549	0.561	0.567	0.566	0.545	0.560	0.570	0.578	0.579
	No Mixing	2	0.533	0.533	0.555	0.558	0.559	0.541	0.543	0.572	0.567	0.570	0.538	0.553	0.572	0.567	0.585	0.547	0.560	0.588	0.582	0.617
		3	0.541	0.530	0.534	0.564	0.600	0.553	0.540	0.537	0.597	0.650	0.553	0.549	0.573	0.575	0.582	0.567	0.557	0.596	0.600	0.617
		1	0.534	0.542	0.781	0.634	0.589	0.541	0.548	1.039	0.708	0.643	0.549	0.673	0.796	0.777	0.619	0.559	0.749	1.034	1.020	0.669
	SA	2	0.548	0.610	0.682	0.745	0.660	0.554	0.645	0.782	0.930	0.762	0.530	0.639	0.665	0.709	0.674	0.539	0.695	0.756	0.874	0.809
		3	0.533	0.531	0.789	0.587	0.661	0.535	0.536	1.018	0.636	0.759	0.540	0.564	0.800	0.703	0.639	0.551	0.589	1.173	0.829	0.739

Table A.8: Raw experiment results for $L_{\text{in}} = 512$ and $T = 512$ on ETTh1 dataset.

T	Time-Mixer	Variate-Mixer	Token-Processor	MSE	MAE	MMaxError
96	MLP Token-Mixer	MLP Token-Mixer	No Processing	0.267 ± 0.005	0.364 ± 0.003	1.006 ± 0.008
			TokenMLP	0.258 ± 0.008	0.360 ± 0.007	0.994 ± 0.012
		No Mixing	No Processing	0.242 ± 0.006	0.339 ± 0.004	0.968 ± 0.010
			TokenMLP	0.236 ± 0.002	0.335 ± 0.001	0.958 ± 0.002
		Self-Attention	No Processing	0.240 ± 0.002	0.338 ± 0.002	0.972 ± 0.004
			TokenMLP	0.246 ± 0.003	0.344 ± 0.002	0.976 ± 0.007
	No Mixing	MLP Token-Mixer	No Processing	0.265 ± 0.004	0.365 ± 0.004	0.997 ± 0.005
			TokenMLP	0.291 ± 0.014	0.383 ± 0.008	1.026 ± 0.014
		No Mixing	No Processing	0.257 ± 0.004	0.355 ± 0.002	0.975 ± 0.006
			TokenMLP	0.244 ± 0.007	0.339 ± 0.004	0.962 ± 0.013
		Self-Attention	No Processing	0.251 ± 0.010	0.346 ± 0.007	0.975 ± 0.015
			TokenMLP	0.245 ± 0.003	0.342 ± 0.002	0.965 ± 0.005
	Self-Attention	MLP Token-Mixer	No Processing	0.264 ± 0.007	0.364 ± 0.005	1.003 ± 0.010
			TokenMLP	0.287 ± 0.011	0.383 ± 0.008	1.032 ± 0.018
		No Mixing	No Processing	0.238 ± 0.006	0.336 ± 0.004	0.954 ± 0.011
			TokenMLP	0.239 ± 0.004	0.333 ± 0.002	0.953 ± 0.005
		Self-Attention	No Processing	0.241 ± 0.005	0.339 ± 0.004	0.964 ± 0.006
			TokenMLP	0.247 ± 0.007	0.345 ± 0.005	0.971 ± 0.011
192	MLP Token-Mixer	MLP Token-Mixer	No Processing	0.359 ± 0.026	0.421 ± 0.011	1.203 ± 0.019
			TokenMLP	0.480 ± 0.097	0.479 ± 0.043	1.284 ± 0.069
		No Mixing	No Processing	0.305 ± 0.007	0.385 ± 0.004	1.159 ± 0.010
			TokenMLP	0.295 ± 0.009	0.378 ± 0.006	1.141 ± 0.016
		Self-Attention	No Processing	0.297 ± 0.003	0.384 ± 0.002	1.151 ± 0.006
			TokenMLP	0.298 ± 0.003	0.383 ± 0.002	1.155 ± 0.005
	No Mixing	MLP Token-Mixer	No Processing	0.508 ± 0.026	0.490 ± 0.010	1.300 ± 0.016
			TokenMLP	0.443 ± 0.020	0.469 ± 0.010	1.259 ± 0.018
		No Mixing	No Processing	0.312 ± 0.004	0.394 ± 0.002	1.157 ± 0.007
			TokenMLP	0.301 ± 0.004	0.385 ± 0.002	1.146 ± 0.007
		Self-Attention	No Processing	0.304 ± 0.003	0.386 ± 0.001	1.146 ± 0.006
			TokenMLP	0.315 ± 0.010	0.392 ± 0.006	1.161 ± 0.015
	Self-Attention	MLP Token-Mixer	No Processing	0.466 ± 0.037	0.466 ± 0.017	1.273 ± 0.025
			TokenMLP	0.463 ± 0.050	0.469 ± 0.018	1.270 ± 0.028
		No Mixing	No Processing	0.288 ± 0.003	0.374 ± 0.001	1.129 ± 0.003
			TokenMLP	0.300 ± 0.012	0.381 ± 0.004	1.147 ± 0.017
		Self-Attention	No Processing	0.302 ± 0.005	0.385 ± 0.003	1.150 ± 0.006
			TokenMLP	0.299 ± 0.003	0.382 ± 0.002	1.143 ± 0.004
336	MLP Token-Mixer	MLP Token-Mixer	No Processing	0.424 ± 0.018	0.463 ± 0.008	1.393 ± 0.011
			TokenMLP	0.528 ± 0.042	0.517 ± 0.017	1.466 ± 0.029
		No Mixing	No Processing	0.339 ± 0.006	0.414 ± 0.003	1.316 ± 0.011
			TokenMLP	0.336 ± 0.007	0.412 ± 0.003	1.313 ± 0.012
		Self-Attention	No Processing	0.348 ± 0.007	0.417 ± 0.004	1.327 ± 0.007
			TokenMLP	0.341 ± 0.005	0.414 ± 0.002	1.326 ± 0.006
	No Mixing	MLP Token-Mixer	No Processing	0.453 ± 0.035	0.478 ± 0.017	1.411 ± 0.021
			TokenMLP	0.485 ± 0.054	0.496 ± 0.024	1.432 ± 0.031
		No Mixing	No Processing	0.347 ± 0.002	0.419 ± 0.001	1.308 ± 0.003
			TokenMLP	0.352 ± 0.007	0.419 ± 0.004	1.321 ± 0.010
		Self-Attention	No Processing	0.348 ± 0.005	0.416 ± 0.003	1.314 ± 0.009
			TokenMLP	0.357 ± 0.002	0.421 ± 0.001	1.320 ± 0.003
	Self-Attention	MLP Token-Mixer	No Processing	0.467 ± 0.046	0.474 ± 0.019	1.409 ± 0.032
			TokenMLP	0.479 ± 0.025	0.493 ± 0.010	1.424 ± 0.013
		No Mixing	No Processing	0.334 ± 0.006	0.407 ± 0.002	1.296 ± 0.012
			TokenMLP	0.339 ± 0.005	0.409 ± 0.002	1.305 ± 0.009
		Self-Attention	No Processing	0.350 ± 0.005	0.417 ± 0.002	1.322 ± 0.007
			TokenMLP	0.349 ± 0.004	0.415 ± 0.002	1.318 ± 0.004
512	MLP Token-Mixer	MLP Token-Mixer	No Processing	0.601 ± 0.086	0.553 ± 0.030	1.614 ± 0.036
			TokenMLP	0.518 ± 0.018	0.528 ± 0.008	1.584 ± 0.014
		No Mixing	No Processing	0.370 ± 0.007	0.436 ± 0.004	1.442 ± 0.012
			TokenMLP	0.372 ± 0.006	0.436 ± 0.004	1.442 ± 0.012
		Self-Attention	No Processing	0.376 ± 0.002	0.438 ± 0.001	1.456 ± 0.006
			TokenMLP	0.384 ± 0.005	0.442 ± 0.002	1.464 ± 0.006
	No Mixing	MLP Token-Mixer	No Processing	0.508 ± 0.070	0.514 ± 0.030	1.568 ± 0.036
			TokenMLP	0.559 ± 0.082	0.543 ± 0.035	1.596 ± 0.042
		No Mixing	No Processing	0.375 ± 0.005	0.440 ± 0.002	1.438 ± 0.010
			TokenMLP	0.385 ± 0.012	0.441 ± 0.006	1.444 ± 0.016
		Self-Attention	No Processing	0.378 ± 0.013	0.436 ± 0.006	1.435 ± 0.020
			TokenMLP	0.403 ± 0.014	0.451 ± 0.008	1.457 ± 0.012
	Self-Attention	MLP Token-Mixer	No Processing	0.651 ± 0.058	0.547 ± 0.019	1.623 ± 0.037
			TokenMLP	0.537 ± 0.043	0.529 ± 0.013	1.596 ± 0.015
		No Mixing	No Processing	0.370 ± 0.004	0.431 ± 0.001	1.425 ± 0.006
			TokenMLP	0.369 ± 0.004	0.431 ± 0.002	1.424 ± 0.008
		Self-Attention	No Processing	0.387 ± 0.008	0.442 ± 0.003	1.447 ± 0.007
			TokenMLP	0.391 ± 0.008	0.443 ± 0.004	1.447 ± 0.012

Table A.9: Errors for best models on ETTh2 dataset with $L_{\text{in}} = 96$ including standard deviations for a total of five runs. Entries marked in red are the best result for each T and entries marked in blue are the second best.

T	Time-Mixer	Variate-Mixer	Token-Processor	MSE	MAE	MMaxError
96	MLP Token-Mixer	MLP Token-Mixer	No Processing	0.304 ± 0.066	0.387 ± 0.029	1.032 ± 0.048
			TokenMLP	0.292 ± 0.029	0.385 ± 0.018	1.035 ± 0.029
		No Mixing	No Processing	0.236 ± 0.002	0.347 ± 0.003	0.949 ± 0.004
			TokenMLP	0.233 ± 0.003	0.340 ± 0.003	0.950 ± 0.007
		Self-Attention	No Processing	0.240 ± 0.003	0.349 ± 0.004	0.956 ± 0.004
			TokenMLP	0.236 ± 0.002	0.345 ± 0.003	0.952 ± 0.003
	No Mixing	MLP Token-Mixer	No Processing	0.328 ± 0.043	0.405 ± 0.018	1.051 ± 0.037
			TokenMLP	0.357 ± 0.048	0.419 ± 0.025	1.084 ± 0.031
		No Mixing	No Processing	0.246 ± 0.004	0.361 ± 0.003	0.951 ± 0.007
			TokenMLP	0.233 ± 0.003	0.345 ± 0.003	0.938 ± 0.008
		Self-Attention	No Processing	0.230 ± 0.003	0.343 ± 0.004	0.934 ± 0.005
			TokenMLP	0.232 ± 0.003	0.346 ± 0.005	0.932 ± 0.003
	Self-Attention	MLP Token-Mixer	No Processing	0.313 ± 0.040	0.391 ± 0.021	1.035 ± 0.033
			TokenMLP	0.320 ± 0.019	0.402 ± 0.009	1.051 ± 0.013
		No Mixing	No Processing	0.225 ± 0.001	0.334 ± 0.001	0.932 ± 0.002
			TokenMLP	0.228 ± 0.003	0.337 ± 0.002	0.939 ± 0.002
		Self-Attention	No Processing	0.226 ± 0.003	0.335 ± 0.002	0.935 ± 0.007
			TokenMLP	0.230 ± 0.003	0.339 ± 0.003	0.937 ± 0.005
192	MLP Token-Mixer	MLP Token-Mixer	No Processing	0.534 ± 0.158	0.502 ± 0.056	1.344 ± 0.088
			TokenMLP	0.330 ± 0.027	0.411 ± 0.014	1.196 ± 0.022
		No Mixing	No Processing	0.275 ± 0.001	0.376 ± 0.001	1.112 ± 0.003
			TokenMLP	0.291 ± 0.018	0.383 ± 0.006	1.126 ± 0.012
		Self-Attention	No Processing	0.283 ± 0.004	0.380 ± 0.001	1.127 ± 0.007
			TokenMLP	0.282 ± 0.006	0.379 ± 0.003	1.126 ± 0.009
	No Mixing	MLP Token-Mixer	No Processing	0.472 ± 0.100	0.486 ± 0.048	1.311 ± 0.072
			TokenMLP	0.452 ± 0.144	0.473 ± 0.062	1.294 ± 0.089
		No Mixing	No Processing	0.281 ± 0.002	0.387 ± 0.002	1.105 ± 0.004
			TokenMLP	0.274 ± 0.003	0.377 ± 0.001	1.096 ± 0.008
		Self-Attention	No Processing	0.274 ± 0.005	0.376 ± 0.001	1.106 ± 0.012
			TokenMLP	0.277 ± 0.002	0.383 ± 0.003	1.103 ± 0.005
	Self-Attention	MLP Token-Mixer	No Processing	0.517 ± 0.118	0.510 ± 0.052	1.342 ± 0.083
			TokenMLP	0.478 ± 0.057	0.497 ± 0.026	1.312 ± 0.041
		No Mixing	No Processing	0.274 ± 0.003	0.373 ± 0.002	1.101 ± 0.004
			TokenMLP	0.277 ± 0.001	0.374 ± 0.000	1.104 ± 0.001
		Self-Attention	No Processing	0.274 ± 0.004	0.373 ± 0.003	1.102 ± 0.008
			TokenMLP	0.287 ± 0.010	0.379 ± 0.006	1.111 ± 0.005
336	MLP Token-Mixer	MLP Token-Mixer	No Processing	0.594 ± 0.131	0.558 ± 0.059	1.589 ± 0.091
			TokenMLP	0.597 ± 0.090	0.568 ± 0.043	1.590 ± 0.064
		No Mixing	No Processing	0.319 ± 0.007	0.406 ± 0.003	1.278 ± 0.006
			TokenMLP	0.322 ± 0.012	0.405 ± 0.005	1.275 ± 0.003
		Self-Attention	No Processing	0.328 ± 0.011	0.409 ± 0.006	1.292 ± 0.014
			TokenMLP	0.332 ± 0.017	0.413 ± 0.009	1.294 ± 0.015
	No Mixing	MLP Token-Mixer	No Processing	0.569 ± 0.025	0.556 ± 0.013	1.582 ± 0.020
			TokenMLP	0.517 ± 0.067	0.535 ± 0.036	1.528 ± 0.063
		No Mixing	No Processing	0.321 ± 0.005	0.414 ± 0.002	1.268 ± 0.019
			TokenMLP	0.312 ± 0.002	0.405 ± 0.001	1.258 ± 0.005
		Self-Attention	No Processing	0.317 ± 0.003	0.408 ± 0.001	1.265 ± 0.009
			TokenMLP	0.367 ± 0.038	0.436 ± 0.022	1.310 ± 0.037
	Self-Attention	MLP Token-Mixer	No Processing	0.517 ± 0.041	0.529 ± 0.017	1.504 ± 0.030
			TokenMLP	0.508 ± 0.089	0.521 ± 0.043	1.481 ± 0.071
		No Mixing	No Processing	0.317 ± 0.005	0.403 ± 0.003	1.258 ± 0.006
			TokenMLP	0.315 ± 0.002	0.402 ± 0.001	1.255 ± 0.006
		Self-Attention	No Processing	0.318 ± 0.005	0.404 ± 0.002	1.261 ± 0.007
			TokenMLP	0.380 ± 0.024	0.440 ± 0.016	1.309 ± 0.021
512	MLP Token-Mixer	MLP Token-Mixer	No Processing	0.644 ± 0.043	0.596 ± 0.022	1.762 ± 0.038
			TokenMLP	0.673 ± 0.146	0.610 ± 0.068	1.751 ± 0.107
		No Mixing	No Processing	0.378 ± 0.029	0.437 ± 0.009	1.431 ± 0.020
			TokenMLP	0.345 ± 0.004	0.426 ± 0.002	1.399 ± 0.005
		Self-Attention	No Processing	0.372 ± 0.019	0.439 ± 0.009	1.436 ± 0.020
			TokenMLP	0.389 ± 0.027	0.446 ± 0.012	1.449 ± 0.024
	No Mixing	MLP Token-Mixer	No Processing	0.723 ± 0.064	0.642 ± 0.028	1.803 ± 0.052
			TokenMLP	0.721 ± 0.068	0.650 ± 0.034	1.799 ± 0.046
		No Mixing	No Processing	0.355 ± 0.004	0.437 ± 0.001	1.396 ± 0.010
			TokenMLP	0.346 ± 0.003	0.429 ± 0.002	1.394 ± 0.008
		Self-Attention	No Processing	0.360 ± 0.006	0.436 ± 0.002	1.416 ± 0.008
			TokenMLP	0.361 ± 0.007	0.435 ± 0.002	1.411 ± 0.007
	Self-Attention	MLP Token-Mixer	No Processing	0.624 ± 0.092	0.586 ± 0.053	1.687 ± 0.081
			TokenMLP	0.657 ± 0.081	0.596 ± 0.033	1.705 ± 0.067
		No Mixing	No Processing	0.345 ± 0.004	0.426 ± 0.002	1.394 ± 0.007
			TokenMLP	0.357 ± 0.016	0.430 ± 0.009	1.395 ± 0.007
		Self-Attention	No Processing	0.429 ± 0.054	0.463 ± 0.024	1.454 ± 0.045
			TokenMLP	0.454 ± 0.021	0.478 ± 0.010	1.477 ± 0.010

Table A.10: Errors for best models on ETTh2 dataset with $L_{\text{in}} = 512$ including standard deviations for a total of five runs. Entries marked in red are the best result for each T and entries marked in blue are the second best.

T	Time-Mixer	Variate-Mixer	Token-Processor	MSE	MAE	MMaxError
96	MLP Token-Mixer	MLP Token-Mixer	No Processing	0.384 ± 0.007	0.402 ± 0.004	1.250 ± 0.007
			TokenMLP	0.385 ± 0.007	0.402 ± 0.004	1.252 ± 0.009
		No Mixing	No Processing	0.351 ± 0.003	0.387 ± 0.002	1.213 ± 0.004
			TokenMLP	0.345 ± 0.003	0.379 ± 0.002	1.198 ± 0.003
		Self-Attention	No Processing	0.356 ± 0.002	0.391 ± 0.001	1.219 ± 0.004
			TokenMLP	0.361 ± 0.012	0.399 ± 0.009	1.228 ± 0.014
	No Mixing	MLP Token-Mixer	No Processing	0.400 ± 0.004	0.414 ± 0.003	1.270 ± 0.003
			TokenMLP	0.384 ± 0.016	0.408 ± 0.008	1.260 ± 0.022
		No Mixing	No Processing	0.379 ± 0.001	0.407 ± 0.001	1.261 ± 0.002
			TokenMLP	0.359 ± 0.003	0.396 ± 0.002	1.215 ± 0.002
		Self-Attention	No Processing	0.362 ± 0.002	0.397 ± 0.002	1.227 ± 0.003
			TokenMLP	0.369 ± 0.003	0.403 ± 0.003	1.236 ± 0.002
	Self-Attention	MLP Token-Mixer	No Processing	0.372 ± 0.008	0.398 ± 0.005	1.238 ± 0.008
			TokenMLP	0.379 ± 0.011	0.401 ± 0.006	1.244 ± 0.014
		No Mixing	No Processing	0.355 ± 0.002	0.388 ± 0.002	1.216 ± 0.003
			TokenMLP	0.360 ± 0.003	0.392 ± 0.002	1.218 ± 0.006
		Self-Attention	No Processing	0.356 ± 0.003	0.391 ± 0.003	1.217 ± 0.002
			TokenMLP	0.365 ± 0.008	0.398 ± 0.007	1.226 ± 0.006
192	MLP Token-Mixer	MLP Token-Mixer	No Processing	0.425 ± 0.007	0.428 ± 0.005	1.510 ± 0.009
			TokenMLP	0.420 ± 0.005	0.424 ± 0.002	1.504 ± 0.011
		No Mixing	No Processing	0.402 ± 0.006	0.417 ± 0.005	1.481 ± 0.006
			TokenMLP	0.393 ± 0.002	0.411 ± 0.001	1.458 ± 0.003
		Self-Attention	No Processing	0.394 ± 0.002	0.416 ± 0.002	1.469 ± 0.004
			TokenMLP	0.398 ± 0.005	0.420 ± 0.004	1.477 ± 0.008
	No Mixing	MLP Token-Mixer	No Processing	0.426 ± 0.010	0.431 ± 0.005	1.515 ± 0.013
			TokenMLP	0.424 ± 0.006	0.431 ± 0.004	1.521 ± 0.009
		No Mixing	No Processing	0.437 ± 0.000	0.434 ± 0.000	1.551 ± 0.001
			TokenMLP	0.406 ± 0.002	0.418 ± 0.001	1.479 ± 0.003
		Self-Attention	No Processing	0.415 ± 0.002	0.424 ± 0.003	1.498 ± 0.007
			TokenMLP	0.413 ± 0.004	0.425 ± 0.003	1.489 ± 0.003
	Self-Attention	MLP Token-Mixer	No Processing	0.414 ± 0.006	0.428 ± 0.003	1.495 ± 0.008
			TokenMLP	0.416 ± 0.003	0.428 ± 0.003	1.497 ± 0.007
		No Mixing	No Processing	0.408 ± 0.006	0.418 ± 0.004	1.472 ± 0.006
			TokenMLP	0.405 ± 0.002	0.414 ± 0.002	1.475 ± 0.003
		Self-Attention	No Processing	0.412 ± 0.002	0.424 ± 0.002	1.486 ± 0.004
			TokenMLP	0.408 ± 0.004	0.418 ± 0.004	1.487 ± 0.006
336	MLP Token-Mixer	MLP Token-Mixer	No Processing	0.466 ± 0.007	0.458 ± 0.004	1.727 ± 0.008
			TokenMLP	0.468 ± 0.011	0.459 ± 0.005	1.723 ± 0.014
		No Mixing	No Processing	0.445 ± 0.005	0.444 ± 0.003	1.696 ± 0.010
			TokenMLP	0.444 ± 0.003	0.446 ± 0.002	1.685 ± 0.004
		Self-Attention	No Processing	0.443 ± 0.004	0.448 ± 0.003	1.688 ± 0.006
			TokenMLP	0.451 ± 0.007	0.454 ± 0.006	1.702 ± 0.006
	No Mixing	MLP Token-Mixer	No Processing	0.476 ± 0.012	0.464 ± 0.004	1.739 ± 0.020
			TokenMLP	0.476 ± 0.007	0.461 ± 0.004	1.764 ± 0.010
		No Mixing	No Processing	0.502 ± 0.005	0.466 ± 0.001	1.818 ± 0.009
			TokenMLP	0.460 ± 0.002	0.450 ± 0.002	1.710 ± 0.002
		Self-Attention	No Processing	0.461 ± 0.005	0.449 ± 0.004	1.722 ± 0.008
			TokenMLP	0.463 ± 0.003	0.453 ± 0.002	1.715 ± 0.006
	Self-Attention	MLP Token-Mixer	No Processing	0.466 ± 0.005	0.458 ± 0.003	1.727 ± 0.008
			TokenMLP	0.509 ± 0.023	0.481 ± 0.012	1.773 ± 0.023
		No Mixing	No Processing	0.459 ± 0.005	0.449 ± 0.003	1.700 ± 0.003
			TokenMLP	0.458 ± 0.006	0.447 ± 0.004	1.698 ± 0.007
		Self-Attention	No Processing	0.461 ± 0.002	0.449 ± 0.002	1.710 ± 0.004
			TokenMLP	0.471 ± 0.006	0.460 ± 0.004	1.721 ± 0.009
512	MLP Token-Mixer	MLP Token-Mixer	No Processing	0.491 ± 0.003	0.475 ± 0.003	1.880 ± 0.009
			TokenMLP	0.501 ± 0.007	0.485 ± 0.005	1.877 ± 0.012
		No Mixing	No Processing	0.474 ± 0.003	0.463 ± 0.001	1.854 ± 0.009
			TokenMLP	0.471 ± 0.004	0.466 ± 0.003	1.837 ± 0.007
		Self-Attention	No Processing	0.480 ± 0.005	0.473 ± 0.004	1.854 ± 0.010
			TokenMLP	0.478 ± 0.007	0.472 ± 0.004	1.848 ± 0.012
	No Mixing	MLP Token-Mixer	No Processing	0.507 ± 0.007	0.487 ± 0.006	1.899 ± 0.007
			TokenMLP	0.509 ± 0.008	0.484 ± 0.005	1.917 ± 0.016
		No Mixing	No Processing	0.540 ± 0.002	0.488 ± 0.001	2.006 ± 0.005
			TokenMLP	0.498 ± 0.002	0.473 ± 0.002	1.885 ± 0.004
		Self-Attention	No Processing	0.494 ± 0.002	0.469 ± 0.001	1.891 ± 0.006
			TokenMLP	0.496 ± 0.003	0.474 ± 0.002	1.889 ± 0.006
	Self-Attention	MLP Token-Mixer	No Processing	0.514 ± 0.005	0.488 ± 0.003	1.904 ± 0.008
			TokenMLP	0.515 ± 0.006	0.486 ± 0.004	1.901 ± 0.011
		No Mixing	No Processing	0.487 ± 0.002	0.465 ± 0.002	1.863 ± 0.004
			TokenMLP	0.493 ± 0.006	0.471 ± 0.006	1.871 ± 0.007
		Self-Attention	No Processing	0.501 ± 0.005	0.477 ± 0.005	1.884 ± 0.005
			TokenMLP	0.502 ± 0.006	0.476 ± 0.006	1.888 ± 0.005

Table A.11: Errors for best models on ETTm1 dataset with $L_{\text{in}} = 96$ including standard deviations for a total of five runs. Entries marked in red are the best result for each T and entries marked in blue are the second best.

T	Time-Mixer	Variate-Mixer	Token-Processor	MSE	MAE	MMaxError
96	MLP Token-Mixer	MLP Token-Mixer	No Processing	0.318 ± 0.004	0.381 ± 0.002	1.182 ± 0.006
			TokenMLP	0.315 ± 0.004	0.375 ± 0.004	1.178 ± 0.004
		No Mixing	No Processing	0.303 ± 0.000	0.364 ± 0.001	1.161 ± 0.001
			TokenMLP	0.306 ± 0.002	0.365 ± 0.002	1.160 ± 0.001
		Self-Attention	No Processing	0.310 ± 0.002	0.372 ± 0.002	1.164 ± 0.002
			TokenMLP	0.310 ± 0.001	0.372 ± 0.001	1.167 ± 0.000
	No Mixing	MLP Token-Mixer	No Processing	0.325 ± 0.003	0.385 ± 0.003	1.193 ± 0.006
			TokenMLP	0.335 ± 0.004	0.393 ± 0.002	1.202 ± 0.006
		No Mixing	No Processing	0.368 ± 0.001	0.408 ± 0.001	1.235 ± 0.002
			TokenMLP	0.342 ± 0.007	0.395 ± 0.006	1.200 ± 0.005
		Self-Attention	No Processing	0.340 ± 0.004	0.392 ± 0.002	1.196 ± 0.004
			TokenMLP	0.336 ± 0.002	0.389 ± 0.001	1.193 ± 0.004
	Self-Attention	MLP Token-Mixer	No Processing	0.334 ± 0.009	0.388 ± 0.006	1.204 ± 0.009
			TokenMLP	0.323 ± 0.004	0.378 ± 0.003	1.192 ± 0.006
		No Mixing	No Processing	0.314 ± 0.002	0.373 ± 0.002	1.172 ± 0.003
			TokenMLP	0.314 ± 0.004	0.371 ± 0.002	1.175 ± 0.004
		Self-Attention	No Processing	0.323 ± 0.008	0.382 ± 0.003	1.182 ± 0.011
			TokenMLP	0.321 ± 0.003	0.378 ± 0.003	1.180 ± 0.006
192	MLP Token-Mixer	MLP Token-Mixer	No Processing	0.359 ± 0.005	0.409 ± 0.003	1.416 ± 0.003
			TokenMLP	0.353 ± 0.003	0.406 ± 0.002	1.404 ± 0.006
		No Mixing	No Processing	0.356 ± 0.004	0.399 ± 0.002	1.420 ± 0.005
			TokenMLP	0.353 ± 0.003	0.397 ± 0.002	1.405 ± 0.002
		Self-Attention	No Processing	0.356 ± 0.005	0.402 ± 0.003	1.414 ± 0.006
			TokenMLP	0.352 ± 0.004	0.400 ± 0.002	1.406 ± 0.004
	No Mixing	MLP Token-Mixer	No Processing	0.374 ± 0.005	0.420 ± 0.002	1.443 ± 0.012
			TokenMLP	0.385 ± 0.006	0.429 ± 0.005	1.450 ± 0.010
		No Mixing	No Processing	0.416 ± 0.002	0.433 ± 0.001	1.508 ± 0.003
			TokenMLP	0.380 ± 0.004	0.414 ± 0.004	1.447 ± 0.003
		Self-Attention	No Processing	0.387 ± 0.002	0.416 ± 0.001	1.461 ± 0.005
			TokenMLP	0.379 ± 0.000	0.414 ± 0.001	1.445 ± 0.002
	Self-Attention	MLP Token-Mixer	No Processing	0.385 ± 0.013	0.416 ± 0.008	1.464 ± 0.019
			TokenMLP	0.383 ± 0.010	0.417 ± 0.006	1.457 ± 0.014
		No Mixing	No Processing	0.363 ± 0.002	0.401 ± 0.003	1.423 ± 0.003
			TokenMLP	0.355 ± 0.002	0.396 ± 0.003	1.412 ± 0.003
		Self-Attention	No Processing	0.371 ± 0.004	0.411 ± 0.004	1.428 ± 0.003
			TokenMLP	0.377 ± 0.004	0.413 ± 0.004	1.440 ± 0.010
336	MLP Token-Mixer	MLP Token-Mixer	No Processing	0.406 ± 0.006	0.441 ± 0.006	1.616 ± 0.006
			TokenMLP	0.406 ± 0.004	0.444 ± 0.003	1.614 ± 0.010
		No Mixing	No Processing	0.390 ± 0.003	0.424 ± 0.003	1.598 ± 0.004
			TokenMLP	0.392 ± 0.002	0.424 ± 0.001	1.602 ± 0.005
		Self-Attention	No Processing	0.394 ± 0.002	0.427 ± 0.000	1.606 ± 0.007
			TokenMLP	0.393 ± 0.004	0.427 ± 0.003	1.602 ± 0.007
	No Mixing	MLP Token-Mixer	No Processing	0.423 ± 0.015	0.449 ± 0.011	1.657 ± 0.022
			TokenMLP	0.440 ± 0.018	0.463 ± 0.011	1.669 ± 0.020
		No Mixing	No Processing	0.462 ± 0.002	0.455 ± 0.001	1.736 ± 0.004
			TokenMLP	0.419 ± 0.001	0.437 ± 0.002	1.648 ± 0.002
		Self-Attention	No Processing	0.427 ± 0.003	0.443 ± 0.002	1.670 ± 0.004
			TokenMLP	0.421 ± 0.005	0.440 ± 0.003	1.656 ± 0.005
	Self-Attention	MLP Token-Mixer	No Processing	0.500 ± 0.059	0.481 ± 0.031	1.752 ± 0.067
			TokenMLP	0.469 ± 0.027	0.464 ± 0.017	1.720 ± 0.036
		No Mixing	No Processing	0.405 ± 0.004	0.426 ± 0.001	1.627 ± 0.010
			TokenMLP	0.402 ± 0.004	0.421 ± 0.002	1.626 ± 0.009
		Self-Attention	No Processing	0.415 ± 0.003	0.434 ± 0.004	1.636 ± 0.004
			TokenMLP	0.412 ± 0.004	0.431 ± 0.004	1.636 ± 0.004
512	MLP Token-Mixer	MLP Token-Mixer	No Processing	0.447 ± 0.006	0.464 ± 0.005	1.794 ± 0.010
			TokenMLP	0.451 ± 0.007	0.470 ± 0.007	1.794 ± 0.007
		No Mixing	No Processing	0.418 ± 0.004	0.443 ± 0.003	1.752 ± 0.005
			TokenMLP	0.420 ± 0.003	0.444 ± 0.002	1.748 ± 0.007
		Self-Attention	No Processing	0.418 ± 0.001	0.444 ± 0.001	1.748 ± 0.006
			TokenMLP	0.422 ± 0.004	0.445 ± 0.003	1.765 ± 0.009
	No Mixing	MLP Token-Mixer	No Processing	0.480 ± 0.012	0.488 ± 0.008	1.847 ± 0.009
			TokenMLP	0.503 ± 0.009	0.500 ± 0.005	1.854 ± 0.019
		No Mixing	No Processing	0.490 ± 0.002	0.471 ± 0.000	1.898 ± 0.006
			TokenMLP	0.451 ± 0.003	0.456 ± 0.003	1.801 ± 0.003
		Self-Attention	No Processing	0.457 ± 0.003	0.457 ± 0.001	1.831 ± 0.012
			TokenMLP	0.455 ± 0.002	0.460 ± 0.003	1.817 ± 0.008
	Self-Attention	MLP Token-Mixer	No Processing	0.504 ± 0.030	0.487 ± 0.018	1.889 ± 0.029
			TokenMLP	0.553 ± 0.026	0.507 ± 0.015	1.956 ± 0.021
		No Mixing	No Processing	0.433 ± 0.003	0.445 ± 0.003	1.780 ± 0.007
			TokenMLP	0.436 ± 0.003	0.442 ± 0.002	1.787 ± 0.004
		Self-Attention	No Processing	0.456 ± 0.009	0.463 ± 0.008	1.803 ± 0.008
			TokenMLP	0.452 ± 0.007	0.453 ± 0.006	1.816 ± 0.019

Table A.12: Errors for best models on ETTm1 dataset with $L_{\text{in}} = 512$ including standard deviations for a total of five runs. Entries marked in red are the best result for each T and entries marked in blue are the second best.

T	Time-Mixer	Variate-Mixer	Token-Processor	MSE	MAE	MMaxError
96	MLP Token-Mixer	MLP Token-Mixer	No Processing	0.156 ± 0.005	0.265 ± 0.004	0.715 ± 0.006
			TokenMLP	0.151 ± 0.003	0.260 ± 0.002	0.706 ± 0.005
		No Mixing	No Processing	0.152 ± 0.009	0.259 ± 0.008	0.711 ± 0.015
			TokenMLP	0.149 ± 0.004	0.256 ± 0.004	0.703 ± 0.008
		Self-Attention	No Processing	0.149 ± 0.003	0.261 ± 0.003	0.705 ± 0.004
			TokenMLP	0.148 ± 0.001	0.259 ± 0.002	0.708 ± 0.002
	No Mixing	MLP Token-Mixer	No Processing	0.166 ± 0.001	0.273 ± 0.001	0.730 ± 0.004
			TokenMLP	0.161 ± 0.003	0.268 ± 0.003	0.715 ± 0.005
		No Mixing	No Processing	0.172 ± 0.002	0.283 ± 0.002	0.740 ± 0.006
			TokenMLP	0.158 ± 0.003	0.263 ± 0.001	0.712 ± 0.005
		Self-Attention	No Processing	0.162 ± 0.001	0.271 ± 0.001	0.727 ± 0.003
			TokenMLP	0.158 ± 0.004	0.267 ± 0.003	0.718 ± 0.007
	Self-Attention	MLP Token-Mixer	No Processing	0.157 ± 0.005	0.264 ± 0.005	0.714 ± 0.010
			TokenMLP	0.180 ± 0.012	0.281 ± 0.006	0.743 ± 0.007
		No Mixing	No Processing	0.152 ± 0.001	0.259 ± 0.001	0.706 ± 0.003
			TokenMLP	0.150 ± 0.002	0.255 ± 0.001	0.701 ± 0.005
		Self-Attention	No Processing	0.154 ± 0.002	0.263 ± 0.001	0.714 ± 0.004
			TokenMLP	0.152 ± 0.003	0.263 ± 0.003	0.712 ± 0.005
192	MLP Token-Mixer	MLP Token-Mixer	No Processing	0.226 ± 0.015	0.321 ± 0.008	0.903 ± 0.013
			TokenMLP	0.215 ± 0.007	0.315 ± 0.006	0.883 ± 0.008
		No Mixing	No Processing	0.199 ± 0.007	0.299 ± 0.005	0.867 ± 0.011
			TokenMLP	0.196 ± 0.006	0.296 ± 0.005	0.865 ± 0.012
		Self-Attention	No Processing	0.196 ± 0.003	0.301 ± 0.003	0.867 ± 0.006
			TokenMLP	0.197 ± 0.004	0.301 ± 0.004	0.871 ± 0.008
	No Mixing	MLP Token-Mixer	No Processing	0.276 ± 0.022	0.350 ± 0.010	0.945 ± 0.016
			TokenMLP	0.223 ± 0.004	0.319 ± 0.004	0.889 ± 0.005
		No Mixing	No Processing	0.215 ± 0.003	0.314 ± 0.002	0.887 ± 0.005
			TokenMLP	0.209 ± 0.006	0.303 ± 0.004	0.878 ± 0.011
		Self-Attention	No Processing	0.208 ± 0.004	0.310 ± 0.003	0.885 ± 0.009
			TokenMLP	0.205 ± 0.004	0.305 ± 0.003	0.882 ± 0.008
	Self-Attention	MLP Token-Mixer	No Processing	0.235 ± 0.009	0.326 ± 0.005	0.900 ± 0.011
			TokenMLP	0.228 ± 0.007	0.323 ± 0.003	0.893 ± 0.008
		No Mixing	No Processing	0.208 ± 0.007	0.302 ± 0.005	0.878 ± 0.012
			TokenMLP	0.200 ± 0.002	0.297 ± 0.002	0.864 ± 0.004
		Self-Attention	No Processing	0.207 ± 0.005	0.307 ± 0.004	0.879 ± 0.010
			TokenMLP	0.217 ± 0.006	0.314 ± 0.004	0.895 ± 0.009
336	MLP Token-Mixer	MLP Token-Mixer	No Processing	0.265 ± 0.007	0.362 ± 0.004	1.040 ± 0.005
			TokenMLP	0.270 ± 0.010	0.367 ± 0.007	1.038 ± 0.009
		No Mixing	No Processing	0.245 ± 0.012	0.336 ± 0.008	1.006 ± 0.019
			TokenMLP	0.267 ± 0.013	0.350 ± 0.010	1.039 ± 0.017
		Self-Attention	No Processing	0.241 ± 0.006	0.336 ± 0.004	1.011 ± 0.008
			TokenMLP	0.237 ± 0.004	0.334 ± 0.003	1.003 ± 0.007
	No Mixing	MLP Token-Mixer	No Processing	0.276 ± 0.011	0.367 ± 0.007	1.045 ± 0.011
			TokenMLP	0.290 ± 0.005	0.375 ± 0.003	1.050 ± 0.003
		No Mixing	No Processing	0.275 ± 0.021	0.360 ± 0.015	1.053 ± 0.034
			TokenMLP	0.254 ± 0.009	0.340 ± 0.006	1.019 ± 0.013
		Self-Attention	No Processing	0.247 ± 0.007	0.341 ± 0.004	1.011 ± 0.014
			TokenMLP	0.248 ± 0.008	0.342 ± 0.005	1.009 ± 0.011
	Self-Attention	MLP Token-Mixer	No Processing	0.326 ± 0.031	0.391 ± 0.021	1.097 ± 0.033
			TokenMLP	0.322 ± 0.026	0.390 ± 0.012	1.088 ± 0.021
		No Mixing	No Processing	0.245 ± 0.004	0.333 ± 0.003	1.000 ± 0.008
			TokenMLP	0.247 ± 0.006	0.334 ± 0.004	1.004 ± 0.009
		Self-Attention	No Processing	0.251 ± 0.009	0.342 ± 0.006	1.015 ± 0.010
			TokenMLP	0.257 ± 0.012	0.347 ± 0.007	1.024 ± 0.014
512	MLP Token-Mixer	MLP Token-Mixer	No Processing	0.341 ± 0.020	0.414 ± 0.010	1.187 ± 0.015
			TokenMLP	0.329 ± 0.008	0.414 ± 0.007	1.181 ± 0.007
		No Mixing	No Processing	0.291 ± 0.011	0.370 ± 0.006	1.137 ± 0.014
			TokenMLP	0.276 ± 0.007	0.363 ± 0.006	1.124 ± 0.011
		Self-Attention	No Processing	0.277 ± 0.005	0.365 ± 0.004	1.126 ± 0.007
			TokenMLP	0.286 ± 0.005	0.369 ± 0.003	1.137 ± 0.005
	No Mixing	MLP Token-Mixer	No Processing	0.334 ± 0.022	0.410 ± 0.013	1.177 ± 0.018
			TokenMLP	0.340 ± 0.010	0.415 ± 0.005	1.188 ± 0.009
		No Mixing	No Processing	0.305 ± 0.015	0.382 ± 0.011	1.157 ± 0.023
			TokenMLP	0.292 ± 0.005	0.370 ± 0.003	1.137 ± 0.007
		Self-Attention	No Processing	0.287 ± 0.005	0.371 ± 0.003	1.134 ± 0.006
			TokenMLP	0.282 ± 0.006	0.367 ± 0.004	1.129 ± 0.008
	Self-Attention	MLP Token-Mixer	No Processing	0.357 ± 0.017	0.417 ± 0.009	1.208 ± 0.017
			TokenMLP	0.336 ± 0.013	0.414 ± 0.010	1.192 ± 0.014
		No Mixing	No Processing	0.279 ± 0.006	0.359 ± 0.004	1.114 ± 0.007
			TokenMLP	0.279 ± 0.004	0.360 ± 0.003	1.115 ± 0.007
		Self-Attention	No Processing	0.299 ± 0.015	0.375 ± 0.008	1.138 ± 0.016
			TokenMLP	0.285 ± 0.009	0.369 ± 0.005	1.126 ± 0.012

Table A.13: Errors for best models on ETTm2 dataset with $L_{\text{in}} = 96$ including standard deviations for a total of five runs. Entries marked in red are the best result for each T and entries marked in blue are the second best.

T	Time-Mixer	Variate-Mixer	Token-Processor	MSE	MAE	MMaxError
96	MLP Token-Mixer	MLP Token-Mixer	No Processing	0.167 ± 0.005	0.284 ± 0.005	0.720 ± 0.007
			TokenMLP	0.167 ± 0.006	0.281 ± 0.004	0.719 ± 0.009
		No Mixing	No Processing	0.148 ± 0.001	0.260 ± 0.001	0.694 ± 0.003
			TokenMLP	0.144 ± 0.001	0.256 ± 0.001	0.684 ± 0.002
		Self-Attention	No Processing	0.151 ± 0.002	0.266 ± 0.002	0.696 ± 0.003
			TokenMLP	0.152 ± 0.001	0.265 ± 0.002	0.700 ± 0.002
	No Mixing	MLP Token-Mixer	No Processing	0.174 ± 0.005	0.291 ± 0.003	0.734 ± 0.008
			TokenMLP	0.178 ± 0.004	0.293 ± 0.002	0.743 ± 0.004
		No Mixing	No Processing	0.183 ± 0.002	0.301 ± 0.002	0.732 ± 0.004
			TokenMLP	0.163 ± 0.009	0.276 ± 0.007	0.713 ± 0.018
		Self-Attention	No Processing	0.168 ± 0.005	0.281 ± 0.004	0.726 ± 0.010
			TokenMLP	0.166 ± 0.001	0.279 ± 0.002	0.716 ± 0.003
	Self-Attention	MLP Token-Mixer	No Processing	0.193 ± 0.007	0.303 ± 0.005	0.765 ± 0.012
			TokenMLP	0.190 ± 0.010	0.301 ± 0.009	0.750 ± 0.013
		No Mixing	No Processing	0.144 ± 0.002	0.256 ± 0.002	0.687 ± 0.005
			TokenMLP	0.147 ± 0.002	0.258 ± 0.001	0.690 ± 0.005
		Self-Attention	No Processing	0.146 ± 0.002	0.260 ± 0.003	0.692 ± 0.003
			TokenMLP	0.164 ± 0.007	0.274 ± 0.005	0.715 ± 0.009
192	MLP Token-Mixer	MLP Token-Mixer	No Processing	0.239 ± 0.013	0.341 ± 0.006	0.902 ± 0.012
			TokenMLP	0.236 ± 0.019	0.342 ± 0.013	0.906 ± 0.018
		No Mixing	No Processing	0.192 ± 0.003	0.299 ± 0.002	0.851 ± 0.007
			TokenMLP	0.190 ± 0.002	0.293 ± 0.002	0.843 ± 0.004
		Self-Attention	No Processing	0.200 ± 0.004	0.311 ± 0.003	0.869 ± 0.009
			TokenMLP	0.196 ± 0.004	0.304 ± 0.003	0.856 ± 0.005
	No Mixing	MLP Token-Mixer	No Processing	0.239 ± 0.020	0.342 ± 0.015	0.909 ± 0.026
			TokenMLP	0.243 ± 0.007	0.341 ± 0.004	0.912 ± 0.009
		No Mixing	No Processing	0.215 ± 0.001	0.325 ± 0.001	0.863 ± 0.002
			TokenMLP	0.202 ± 0.011	0.308 ± 0.006	0.866 ± 0.020
		Self-Attention	No Processing	0.207 ± 0.009	0.312 ± 0.006	0.876 ± 0.017
			TokenMLP	0.207 ± 0.003	0.313 ± 0.002	0.867 ± 0.005
	Self-Attention	MLP Token-Mixer	No Processing	0.302 ± 0.023	0.371 ± 0.013	0.962 ± 0.021
			TokenMLP	0.253 ± 0.007	0.351 ± 0.006	0.953 ± 0.012
		No Mixing	No Processing	0.191 ± 0.003	0.295 ± 0.002	0.847 ± 0.006
			TokenMLP	0.191 ± 0.002	0.294 ± 0.002	0.844 ± 0.004
		Self-Attention	No Processing	0.204 ± 0.007	0.306 ± 0.005	0.859 ± 0.008
			TokenMLP	0.202 ± 0.007	0.307 ± 0.005	0.857 ± 0.008
336	MLP Token-Mixer	MLP Token-Mixer	No Processing	0.419 ± 0.079	0.445 ± 0.040	1.144 ± 0.050
			TokenMLP	0.368 ± 0.016	0.422 ± 0.011	1.120 ± 0.012
		No Mixing	No Processing	0.231 ± 0.004	0.330 ± 0.004	0.982 ± 0.007
			TokenMLP	0.230 ± 0.004	0.327 ± 0.002	0.980 ± 0.006
		Self-Attention	No Processing	0.235 ± 0.004	0.337 ± 0.003	0.985 ± 0.006
			TokenMLP	0.235 ± 0.005	0.336 ± 0.003	0.989 ± 0.008
	No Mixing	MLP Token-Mixer	No Processing	0.337 ± 0.019	0.400 ± 0.009	1.081 ± 0.014
			TokenMLP	0.368 ± 0.068	0.415 ± 0.030	1.108 ± 0.051
		No Mixing	No Processing	0.248 ± 0.002	0.349 ± 0.001	0.996 ± 0.006
			TokenMLP	0.239 ± 0.006	0.338 ± 0.004	0.998 ± 0.010
		Self-Attention	No Processing	0.239 ± 0.004	0.337 ± 0.003	0.996 ± 0.008
			TokenMLP	0.260 ± 0.005	0.353 ± 0.005	1.017 ± 0.008
	Self-Attention	MLP Token-Mixer	No Processing	0.352 ± 0.042	0.405 ± 0.019	1.100 ± 0.026
			TokenMLP	0.357 ± 0.026	0.417 ± 0.015	1.165 ± 0.023
		No Mixing	No Processing	0.231 ± 0.006	0.325 ± 0.003	0.980 ± 0.007
			TokenMLP	0.235 ± 0.006	0.329 ± 0.003	0.982 ± 0.008
		Self-Attention	No Processing	0.249 ± 0.012	0.344 ± 0.007	0.995 ± 0.012
			TokenMLP	0.268 ± 0.010	0.358 ± 0.007	1.024 ± 0.010
512	MLP Token-Mixer	MLP Token-Mixer	No Processing	0.510 ± 0.041	0.485 ± 0.016	1.292 ± 0.028
			TokenMLP	0.581 ± 0.041	0.514 ± 0.014	1.333 ± 0.018
		No Mixing	No Processing	0.265 ± 0.009	0.359 ± 0.006	1.101 ± 0.012
			TokenMLP	0.271 ± 0.009	0.359 ± 0.004	1.109 ± 0.014
		Self-Attention	No Processing	0.273 ± 0.005	0.364 ± 0.002	1.109 ± 0.007
			TokenMLP	0.275 ± 0.006	0.369 ± 0.004	1.113 ± 0.010
	No Mixing	MLP Token-Mixer	No Processing	0.575 ± 0.009	0.505 ± 0.002	1.320 ± 0.006
			TokenMLP	0.587 ± 0.028	0.507 ± 0.013	1.355 ± 0.023
		No Mixing	No Processing	0.271 ± 0.002	0.366 ± 0.001	1.096 ± 0.003
			TokenMLP	0.273 ± 0.007	0.363 ± 0.005	1.118 ± 0.011
		Self-Attention	No Processing	0.276 ± 0.007	0.364 ± 0.004	1.112 ± 0.012
			TokenMLP	0.279 ± 0.005	0.368 ± 0.004	1.108 ± 0.006
	Self-Attention	MLP Token-Mixer	No Processing	0.508 ± 0.047	0.475 ± 0.015	1.298 ± 0.031
			TokenMLP	0.573 ± 0.082	0.500 ± 0.026	1.329 ± 0.044
		No Mixing	No Processing	0.253 ± 0.001	0.348 ± 0.000	1.082 ± 0.002
			TokenMLP	0.264 ± 0.002	0.352 ± 0.001	1.091 ± 0.004
		Self-Attention	No Processing	0.298 ± 0.009	0.379 ± 0.005	1.122 ± 0.009
			TokenMLP	0.295 ± 0.006	0.378 ± 0.004	1.124 ± 0.005

Table A.14: Errors for best models on ETTm2 dataset with $L_{\text{in}} = 512$ including standard deviations for a total of five runs. Entries marked in red are the best result for each T and entries marked in blue are the second best.

T	Time-Mixer	Variate-Mixer	Token-Processor	MSE	MAE	MMaxError
96	MLP Token-Mixer	MLP Token-Mixer	No Processing	0.275 ± 0.006	0.252 ± 0.005	0.735 ± 0.009
			TokenMLP	0.272 ± 0.005	0.254 ± 0.005	0.721 ± 0.008
		No Mixing	No Processing	0.288 ± 0.004	0.263 ± 0.004	0.744 ± 0.005
			TokenMLP	0.272 ± 0.001	0.248 ± 0.002	0.724 ± 0.002
		Self-Attention	No Processing	0.269 ± 0.003	0.248 ± 0.001	0.720 ± 0.003
			TokenMLP	0.271 ± 0.003	0.248 ± 0.003	0.724 ± 0.004
	No Mixing	MLP Token-Mixer	No Processing	0.289 ± 0.001	0.273 ± 0.001	0.751 ± 0.001
			TokenMLP	0.292 ± 0.006	0.276 ± 0.007	0.755 ± 0.008
		No Mixing	No Processing	0.370 ± 0.001	0.328 ± 0.000	0.845 ± 0.001
			TokenMLP	0.299 ± 0.004	0.277 ± 0.004	0.763 ± 0.007
		Self-Attention	No Processing	0.309 ± 0.003	0.286 ± 0.005	0.771 ± 0.004
			TokenMLP	0.300 ± 0.009	0.279 ± 0.008	0.760 ± 0.004
	Self-Attention	MLP Token-Mixer	No Processing	0.276 ± 0.006	0.251 ± 0.003	0.729 ± 0.005
			TokenMLP	0.266 ± 0.003	0.246 ± 0.003	0.716 ± 0.005
		No Mixing	No Processing	0.278 ± 0.002	0.253 ± 0.003	0.732 ± 0.002
			TokenMLP	0.276 ± 0.001	0.253 ± 0.002	0.731 ± 0.002
		Self-Attention	No Processing	0.285 ± 0.002	0.260 ± 0.001	0.744 ± 0.003
			TokenMLP	0.283 ± 0.005	0.258 ± 0.003	0.738 ± 0.004
192	MLP Token-Mixer	MLP Token-Mixer	No Processing	0.327 ± 0.003	0.301 ± 0.003	0.993 ± 0.005
			TokenMLP	0.319 ± 0.005	0.296 ± 0.003	0.984 ± 0.006
		No Mixing	No Processing	0.330 ± 0.001	0.302 ± 0.001	0.988 ± 0.002
			TokenMLP	0.319 ± 0.001	0.292 ± 0.001	0.984 ± 0.002
		Self-Attention	No Processing	0.320 ± 0.001	0.296 ± 0.002	0.981 ± 0.004
			TokenMLP	0.321 ± 0.002	0.297 ± 0.002	0.981 ± 0.002
	No Mixing	MLP Token-Mixer	No Processing	0.341 ± 0.004	0.314 ± 0.004	1.012 ± 0.006
			TokenMLP	0.339 ± 0.006	0.309 ± 0.005	1.016 ± 0.008
		No Mixing	No Processing	0.422 ± 0.000	0.359 ± 0.000	1.111 ± 0.001
			TokenMLP	0.339 ± 0.001	0.306 ± 0.002	1.012 ± 0.004
		Self-Attention	No Processing	0.346 ± 0.002	0.315 ± 0.003	1.016 ± 0.002
			TokenMLP	0.342 ± 0.003	0.311 ± 0.001	1.014 ± 0.003
	Self-Attention	MLP Token-Mixer	No Processing	0.317 ± 0.005	0.291 ± 0.005	0.980 ± 0.007
			TokenMLP	0.326 ± 0.010	0.299 ± 0.007	0.993 ± 0.011
		No Mixing	No Processing	0.321 ± 0.002	0.290 ± 0.002	0.988 ± 0.004
			TokenMLP	0.318 ± 0.002	0.287 ± 0.002	0.985 ± 0.002
		Self-Attention	No Processing	0.332 ± 0.003	0.302 ± 0.003	1.005 ± 0.005
			TokenMLP	0.326 ± 0.003	0.297 ± 0.003	0.996 ± 0.004
336	MLP Token-Mixer	MLP Token-Mixer	No Processing	0.382 ± 0.007	0.342 ± 0.005	1.251 ± 0.008
			TokenMLP	0.372 ± 0.002	0.337 ± 0.002	1.224 ± 0.002
		No Mixing	No Processing	0.381 ± 0.001	0.339 ± 0.002	1.226 ± 0.003
			TokenMLP	0.373 ± 0.001	0.334 ± 0.002	1.230 ± 0.002
		Self-Attention	No Processing	0.374 ± 0.002	0.337 ± 0.002	1.224 ± 0.004
			TokenMLP	0.377 ± 0.002	0.339 ± 0.002	1.231 ± 0.004
	No Mixing	MLP Token-Mixer	No Processing	0.389 ± 0.004	0.349 ± 0.003	1.255 ± 0.007
			TokenMLP	0.386 ± 0.005	0.347 ± 0.003	1.247 ± 0.006
		No Mixing	No Processing	0.466 ± 0.001	0.387 ± 0.000	1.351 ± 0.001
			TokenMLP	0.388 ± 0.002	0.342 ± 0.002	1.251 ± 0.001
		Self-Attention	No Processing	0.397 ± 0.002	0.351 ± 0.002	1.255 ± 0.002
			TokenMLP	0.395 ± 0.001	0.348 ± 0.003	1.256 ± 0.002
	Self-Attention	MLP Token-Mixer	No Processing	0.382 ± 0.011	0.340 ± 0.007	1.243 ± 0.015
			TokenMLP	0.370 ± 0.008	0.335 ± 0.007	1.221 ± 0.009
		No Mixing	No Processing	0.375 ± 0.001	0.333 ± 0.004	1.240 ± 0.003
			TokenMLP	0.371 ± 0.001	0.328 ± 0.001	1.238 ± 0.002
		Self-Attention	No Processing	0.380 ± 0.002	0.338 ± 0.002	1.244 ± 0.003
			TokenMLP	0.381 ± 0.003	0.340 ± 0.003	1.249 ± 0.004
512	MLP Token-Mixer	MLP Token-Mixer	No Processing	0.430 ± 0.003	0.375 ± 0.003	1.460 ± 0.005
			TokenMLP	0.425 ± 0.005	0.375 ± 0.003	1.447 ± 0.006
		No Mixing	No Processing	0.436 ± 0.002	0.378 ± 0.005	1.451 ± 0.005
			TokenMLP	0.429 ± 0.001	0.372 ± 0.002	1.459 ± 0.003
		Self-Attention	No Processing	0.430 ± 0.002	0.375 ± 0.001	1.450 ± 0.003
			TokenMLP	0.429 ± 0.003	0.376 ± 0.003	1.450 ± 0.005
	No Mixing	MLP Token-Mixer	No Processing	0.439 ± 0.004	0.383 ± 0.003	1.472 ± 0.004
			TokenMLP	0.433 ± 0.002	0.381 ± 0.001	1.461 ± 0.005
		No Mixing	No Processing	0.511 ± 0.000	0.414 ± 0.000	1.574 ± 0.001
			TokenMLP	0.435 ± 0.001	0.372 ± 0.002	1.476 ± 0.003
		Self-Attention	No Processing	0.443 ± 0.002	0.384 ± 0.002	1.479 ± 0.003
			TokenMLP	0.441 ± 0.001	0.380 ± 0.002	1.477 ± 0.003
	Self-Attention	MLP Token-Mixer	No Processing	0.448 ± 0.014	0.385 ± 0.010	1.471 ± 0.017
			TokenMLP	0.431 ± 0.011	0.376 ± 0.006	1.454 ± 0.014
		No Mixing	No Processing	0.427 ± 0.001	0.370 ± 0.001	1.466 ± 0.003
			TokenMLP	0.419 ± 0.002	0.362 ± 0.003	1.461 ± 0.006
		Self-Attention	No Processing	0.430 ± 0.002	0.376 ± 0.004	1.462 ± 0.003
			TokenMLP	0.431 ± 0.007	0.374 ± 0.006	1.480 ± 0.013

Table A.15: Errors for best models on weather dataset with $L_{\text{in}} = 96$ including standard deviations for a total of five runs. Entries marked in red are the best result for each T and entries marked in blue are the second best.

T	Time-Mixer	Variate-Mixer	Token-Processor	MSE	MAE	MMaxError
96	MLP Token-Mixer	MLP Token-Mixer	No Processing	0.261 ± 0.002	0.248 ± 0.003	0.711 ± 0.002
			TokenMLP	0.255 ± 0.001	0.245 ± 0.001	0.708 ± 0.002
		No Mixing	No Processing	0.263 ± 0.003	0.249 ± 0.004	0.708 ± 0.007
			TokenMLP	0.258 ± 0.002	0.244 ± 0.003	0.703 ± 0.003
		Self-Attention	No Processing	0.258 ± 0.001	0.247 ± 0.002	0.705 ± 0.001
			TokenMLP	0.260 ± 0.001	0.248 ± 0.003	0.716 ± 0.003
	No Mixing	MLP Token-Mixer	No Processing	0.279 ± 0.002	0.273 ± 0.002	0.733 ± 0.003
			TokenMLP	0.277 ± 0.002	0.270 ± 0.001	0.732 ± 0.004
		No Mixing	No Processing	0.387 ± 0.001	0.364 ± 0.002	0.859 ± 0.003
			TokenMLP	0.283 ± 0.002	0.274 ± 0.004	0.746 ± 0.004
		Self-Attention	No Processing	0.290 ± 0.002	0.284 ± 0.002	0.756 ± 0.003
			TokenMLP	0.283 ± 0.001	0.278 ± 0.003	0.745 ± 0.002
	Self-Attention	MLP Token-Mixer	No Processing	0.262 ± 0.004	0.251 ± 0.003	0.717 ± 0.003
			TokenMLP	0.270 ± 0.004	0.256 ± 0.003	0.728 ± 0.004
		No Mixing	No Processing	0.261 ± 0.002	0.242 ± 0.002	0.710 ± 0.007
			TokenMLP	0.260 ± 0.001	0.243 ± 0.002	0.721 ± 0.004
		Self-Attention	No Processing	0.270 ± 0.001	0.256 ± 0.002	0.720 ± 0.002
			TokenMLP	0.266 ± 0.002	0.252 ± 0.005	0.717 ± 0.005
192	MLP Token-Mixer	MLP Token-Mixer	No Processing	0.310 ± 0.002	0.292 ± 0.001	0.972 ± 0.003
			TokenMLP	0.316 ± 0.005	0.302 ± 0.005	0.982 ± 0.006
		No Mixing	No Processing	0.312 ± 0.002	0.291 ± 0.002	0.960 ± 0.002
			TokenMLP	0.308 ± 0.001	0.288 ± 0.001	0.963 ± 0.001
		Self-Attention	No Processing	0.311 ± 0.003	0.292 ± 0.002	0.965 ± 0.004
			TokenMLP	0.311 ± 0.002	0.293 ± 0.002	0.969 ± 0.006
	No Mixing	MLP Token-Mixer	No Processing	0.326 ± 0.003	0.309 ± 0.003	0.974 ± 0.005
			TokenMLP	0.340 ± 0.006	0.318 ± 0.003	1.005 ± 0.005
		No Mixing	No Processing	0.417 ± 0.001	0.382 ± 0.001	1.094 ± 0.002
			TokenMLP	0.329 ± 0.001	0.309 ± 0.002	0.993 ± 0.004
		Self-Attention	No Processing	0.338 ± 0.002	0.321 ± 0.004	1.002 ± 0.003
			TokenMLP	0.330 ± 0.001	0.309 ± 0.002	0.993 ± 0.002
	Self-Attention	MLP Token-Mixer	No Processing	0.325 ± 0.013	0.304 ± 0.011	1.004 ± 0.018
			TokenMLP	0.336 ± 0.016	0.314 ± 0.011	1.014 ± 0.023
		No Mixing	No Processing	0.311 ± 0.001	0.289 ± 0.005	0.969 ± 0.003
			TokenMLP	0.310 ± 0.002	0.287 ± 0.003	0.970 ± 0.005
		Self-Attention	No Processing	0.323 ± 0.002	0.300 ± 0.002	0.973 ± 0.003
			TokenMLP	0.320 ± 0.002	0.299 ± 0.004	0.976 ± 0.007
336	MLP Token-Mixer	MLP Token-Mixer	No Processing	0.390 ± 0.012	0.356 ± 0.009	1.261 ± 0.013
			TokenMLP	0.385 ± 0.008	0.358 ± 0.006	1.243 ± 0.009
		No Mixing	No Processing	0.375 ± 0.002	0.340 ± 0.003	1.215 ± 0.002
			TokenMLP	0.372 ± 0.002	0.338 ± 0.002	1.228 ± 0.003
		Self-Attention	No Processing	0.377 ± 0.004	0.345 ± 0.003	1.227 ± 0.004
			TokenMLP	0.376 ± 0.002	0.343 ± 0.002	1.231 ± 0.003
	No Mixing	MLP Token-Mixer	No Processing	0.407 ± 0.010	0.361 ± 0.006	1.252 ± 0.010
			TokenMLP	0.405 ± 0.005	0.367 ± 0.003	1.257 ± 0.004
		No Mixing	No Processing	0.450 ± 0.001	0.402 ± 0.001	1.329 ± 0.001
			TokenMLP	0.380 ± 0.001	0.346 ± 0.002	1.237 ± 0.003
		Self-Attention	No Processing	0.391 ± 0.003	0.356 ± 0.004	1.249 ± 0.003
			TokenMLP	0.382 ± 0.002	0.344 ± 0.001	1.240 ± 0.003
	Self-Attention	MLP Token-Mixer	No Processing	0.425 ± 0.050	0.376 ± 0.035	1.312 ± 0.057
			TokenMLP	0.477 ± 0.028	0.404 ± 0.015	1.337 ± 0.025
		No Mixing	No Processing	0.364 ± 0.002	0.333 ± 0.003	1.205 ± 0.001
			TokenMLP	0.368 ± 0.003	0.334 ± 0.005	1.221 ± 0.002
		Self-Attention	No Processing	0.373 ± 0.002	0.340 ± 0.003	1.225 ± 0.003
			TokenMLP	0.379 ± 0.003	0.347 ± 0.002	1.240 ± 0.003
512	MLP Token-Mixer	MLP Token-Mixer	No Processing	0.453 ± 0.023	0.400 ± 0.015	1.493 ± 0.018
			TokenMLP	0.463 ± 0.011	0.407 ± 0.005	1.502 ± 0.010
		No Mixing	No Processing	0.424 ± 0.002	0.377 ± 0.001	1.443 ± 0.003
			TokenMLP	0.424 ± 0.001	0.376 ± 0.002	1.454 ± 0.001
		Self-Attention	No Processing	0.423 ± 0.005	0.380 ± 0.003	1.449 ± 0.007
			TokenMLP	0.452 ± 0.009	0.399 ± 0.006	1.500 ± 0.013
	No Mixing	MLP Token-Mixer	No Processing	0.463 ± 0.009	0.399 ± 0.006	1.489 ± 0.010
			TokenMLP	0.458 ± 0.010	0.401 ± 0.009	1.492 ± 0.012
		No Mixing	No Processing	0.480 ± 0.001	0.421 ± 0.001	1.539 ± 0.002
			TokenMLP	0.423 ± 0.002	0.374 ± 0.003	1.458 ± 0.002
		Self-Attention	No Processing	0.432 ± 0.002	0.385 ± 0.003	1.468 ± 0.003
			TokenMLP	0.427 ± 0.002	0.376 ± 0.004	1.467 ± 0.003
	Self-Attention	MLP Token-Mixer	No Processing	0.516 ± 0.015	0.430 ± 0.008	1.530 ± 0.011
			TokenMLP	0.480 ± 0.064	0.402 ± 0.033	1.524 ± 0.058
		No Mixing	No Processing	0.408 ± 0.001	0.366 ± 0.001	1.427 ± 0.002
			TokenMLP	0.414 ± 0.002	0.367 ± 0.003	1.442 ± 0.003
		Self-Attention	No Processing	0.417 ± 0.002	0.370 ± 0.003	1.447 ± 0.004
			TokenMLP	0.426 ± 0.010	0.378 ± 0.007	1.465 ± 0.015

Table A.16: Errors for best models on weather dataset with $L_{\text{in}} = 512$ including standard deviations for a total of five runs. Entries marked in red are the best result for each T and entries marked in blue are the second best.

Bibliography

[And+15] Alexandr Andoni et al. "Practical and Optimal LSH for Angular Distance". In: *Neural Information Processing Systems*. Vol. 1. Sept. 2015, pp. 1225–1233.

[AS74] R. J. Aumann and L. S. Shapley. "Values of Non-Atomic Games". In: Princeton University Press, 1974.

[Bae+09] David Baehrens et al. "How to Explain Individual Classification Decisions". In: 11 (Dec. 2009), pp. 1803–1831.

[Bin+16] Alexander Binder et al. "Layer-Wise Relevance Propagation for Neural Networks with Local Renormalization Layers". In: abs/1604.00825 (2016). Ed. by Alessandro E.P. Villa, Paolo Masulli, and Antonio Javier Pons Rivero.

[BJ77] George. E. P. Box and Gwilym M Jenkins. "Time Series Analysis: Forecasting and Control". In: 14 (1977), p. 269.

[Bro+20] Tom B. Brown et al. "Language Models Are Few-Shot Learners". In: *Advances in Neural Information Processing Systems*. Vol. 33. Curran Associates, Inc., 2020, pp. 1877–1901.

[Bro+21] Michael M. Bronstein et al. *Geometric Deep Learning: Grids, Groups, Graphs, Geodesics, and Gauges*. May 2021. arXiv: `2104.13478 [cs, stat]`. (Visited on 03/10/2024).

[Che+18] Ricky T. Q. Chen et al. "Neural Ordinary Differential Equations". In: *Advances in Neural Information Processing Systems*. Vol. 31. Curran Associates, Inc., 2018.

[Che+21] Beidi Chen et al. "Scatterbrain: Unifying Sparse and Low-rank Attention Approximation". In: *Advances in Neural Information Processing Systems*. Vol. 34. Curran Associates, Inc., 2021, pp. 17413–17426.

[Che+23] Si Chen et al. "TSMixer: An All-MLP Architecture for Time Series Forecasting". In: (2023). ISSN: 2835-8856.

[Chi+23] Krishna Teja Chitty-Venkata et al. "A Survey of Techniques for Optimizing Transformer Inference". In: 144 (2023), p. 102990. ISSN: 1383-7621.

[Cho+21] Krzysztof Choromanski et al. "Rethinking Attention with Performers". In: *International Conference on Learning Representations*. 2021.

[DB21] Vijay Prakash Dwivedi and Xavier Bresson. "A Generalization of Transformer Networks to Graphs". In: (2021).

[Dev+19] Jacob Devlin et al. "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding". In: *North American Chapter of the Association for Computational Linguistics*. 2019.

[Dos+21] A. Dosovitskiy et al. “An Image Is Worth 16x16 Words: Transformers for Image Recognition at Scale”. In: *International Conference on Learning Representations*. 2021.

[DXX18] Linhao Dong, Shuang Xu, and Bo Xu. “Speech-Transformer: A No-Recurrence Sequence-to-Sequence Model for Speech Recognition”. In: *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. 2018, pp. 5884–5888.

[Fri04] Eric J. Friedman. “Paths and Consistency in Additive Cost Sharing”. In: 32 (2004), pp. 501–518.

[Gev+21] Mor Geva et al. “Transformer Feed-Forward Layers Are Key-Value Memories”. In: *Empirical Methods in Natural Language Processing (EMNLP)*. 2021.

[Gev+22] Mor Geva et al. “Transformer Feed-Forward Layers Build Predictions by Promoting Concepts in the Vocabulary Space”. In: *Conference on Empirical Methods in Natural Language Processing*. arXiv, 2022, pp. 30–45.

[He+16] Kaiming He et al. “Deep Residual Learning for Image Recognition”. In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. arXiv, 2016, pp. 770–778.

[HS97] S. Hochreiter and J. Schmidhuber. “Long Short-Term Memory”. In: *Neural Computation* 9 (1997), pp. 1735–1780.

[IS15] Sergey Ioffe and Christian Szegedy. “Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift”. In: *Proceedings of the 32nd International Conference on Machine Learning*. Vol. 37. Proceedings of Machine Learning Research. PMLR, 2015, pp. 448–456. arXiv: `1502.03167 [cs]`. (Visited on 03/19/2024).

[KB14] Diederik P. Kingma and Jimmy Ba. “Adam: A Method for Stochastic Optimization”. In: abs/1412.6980 (2014).

[Kes23] Abhinav Kaushal Keshari. *Multi-Genre Music Transformer – Composing Full Length Musical Piece*. Jan. 2023. arXiv: `2301.02385 [cs, eess]`. (Visited on 06/04/2024).

[KKL20] Nikita Kitaev, Lukasz Kaiser, and Anselm Levskaya. “Reformer: The Efficient Transformer”. In: *ArXiv*. 2020.

[Kob+21] Goro Kobayashi et al. “Incorporating Residual and Normalization Layers into Analysis of Masked Language Models”. In: *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*. Ed. by Marie-Francine Moens et al. Association for Computational Linguistics, 2021, pp. 4547–4568. DOI: `10.18653/v1/2021.emnlp-main.373`.

[Kob+24] Goro Kobayashi et al. “Analyzing Feed-Forward Blocks in Transformers through the Lens of Attention Map”. In: *International Conference on Learning Representations*. arXiv, 2024. arXiv: `2302.00456 [cs]`.

[LF87] Lapedes and Farber. “Nonlinear Signal Processing Using Neural Networks: Prediction and System Modelling”. In: *IEEE International Conference on Neural Networks*. 1987.

[LI+19] SHIYANG LI et al. “Enhancing the Locality and Breaking the Memory Bottleneck of Transformer on Time Series Forecasting”. In: *Advances in Neural Information Processing Systems*. Vol. 32. Curran Associates, Inc., 2019.

[Liu+22a] Shizhan Liu et al. “Pyraformer: Low-Complexity Pyramidal Attention for Long-Range Time Series Modeling and Forecasting”. In: *International Conference on Learning Representations*. 2022.

[Liu+22b] Yong Liu et al. "Non-Stationary Transformers: Exploring the Stationarity in Time Series Forecasting". In: *Advances in Neural Information Processing Systems*. Vol. 35. Curran Associates, Inc., 2022, pp. 9881–9893. arXiv: 2205.14415 [cs, eess]. (Visited on 02/19/2024).

[Liu+24] Yong Liu et al. "iTransformer: Inverted Transformers Are Effective for Time Series Forecasting". In: *International Conference on Learning Representations*. 2024.

[Nie+23] Yuqi Nie et al. "A Time Series Is Worth 64 Words: Long-term Forecasting with Transformers". In: *International Conference on Learning Representations*. 2023.

[Qin+22] Zhen Qin et al. "cosFormer: Rethinking Softmax in Attention". In: *International Conference on Learning Representations*. 2022.

[Ram+21] Aditya Ramesh et al. "Zero-Shot Text-to-Image Generation". In: *Proceedings of the 38th International Conference on Machine Learning*. Vol. 139. PMLR, 2021.

[RR07] Ali Rahimi and Benjamin Recht. "Random Features for Large-Scale Kernel Machines". In: *Advances in Neural Information Processing Systems*. Vol. 20. Curran Associates, Inc., 2007.

[SAP22] Michael E. Sander, Pierre Ablin, and Gabriel Peyré. "Do Residual Neural Networks Discretize Neural Ordinary Differential Equations?" In: *36th Conference on Neural Information Processing Systems*. 2022.

[SGK17] Avanti Shrikumar, Peyton Greenside, and Anshul Kundaje. "Learning Important Features Through Propagating Activation Differences". In: *International Conference on Machine Learning*. arXiv, 2017.

[Spr+14] Jost Tobias Springenberg et al. "Striving for Simplicity: The All Convolutional Net". In: abs/1412.6806 (2014).

[ST18] Leslie N. Smith and Nicholay Topin. "Super-Convergence: Very Fast Training of Neural Networks Using Large Learning Rates". In: *Defense + Commercial Sensing*. 2018.

[STY17] Mukund Sundararajan, Ankur Taly, and Qiqi Yan. "Axiomatic Attribution for Deep Networks". In: *Proceedings of the 34th International Conference on Machine Learning*. 2017.

[SVL14] Ilya Sutskever, Oriol Vinyals, and Quoc V. Le. "Sequence to Sequence Learning with Neural Networks". In: *Advances in Neural Information Processing Systems*. Vol. 27. Curran Associates, Inc., 2014.

[TCJ22] Shreshth Tuli, Giuliano Casale, and Nicholas R. Jennings. "TranAD: Deep Transformer Networks for Anomaly Detection in Multivariate Time Series Data". In: 15 (2022), pp. 1201–1214.

[Tol+21] Ilya Tolstikhin et al. "MLP-Mixer: An All-MLP Architecture for Vision". In: *Advances in Neural Information Processing Systems*. Vol. 34. Curran Associates, Inc., 2021, pp. 24261–24272.

[Vas+17] Ashish Vaswani et al. "Attention Is All You Need". In: *31st Conference on Neural Information Processing Systems*. 2017.

[Wan+20] Sinong Wang et al. "Linformer: Self-Attention with Linear Complexity". In: abs/2006.04768 (2020). arXiv: 2006.04768 [cs, stat].

[Wan+24] Xue Wang et al. "Make Transformer Great Again for Time Series Forecasting: Channel Aligned Robust Dual Transformer". In: *The Twelfth International Conference on Learning Representations*. 2024.

[Wu+21] Haixu Wu et al. “Autoformer: Decomposition Transformers with Auto-Correlation for Long-Term Series Forecasting”. In: *Advances in Neural Information Processing Systems*. Vol. 34. arXiv, 2021.

[Wu+23] Haixu Wu et al. “TimesNet: Temporal 2D-Variation Modeling for General Time Series Analysis”. In: *International Conference on Learning Representations*. 2023.

[Xio+20] Ruibin Xiong et al. “On Layer Normalization in the Transformer Architecture”. In: *International Conference on Machine Learning*. 2020.

[Xu+21] Jiehui Xu et al. “Anomaly Transformer: Time Series Anomaly Detection with Association Discrepancy”. In: *International Conference on Learning Representations*. 2021.

[Yul27] George Udny Yule. “On a Method of Investigating Periodicities Disturbed Series, with Special Reference to Wolfer’s Sunspot Numbers”. In: 226 (1927), pp. 267–298.

[Zen+23] Ailing Zeng et al. “Are Transformers Effective for Time Series Forecasting?” In: *Proceedings of the AAAI Conference on Artificial Intelligence*. Vol. 37. 2023, pp. 11121–11128.

[Zer+21] George Zerveas et al. “A Transformer-based Framework for Multivariate Time Series Representation Learning”. In: *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*. Association for Computing Machinery, 2021, pp. 2114–2124.

[ZF14] Matthew D. Zeiler and Rob Fergus. “Visualizing and Understanding Convolutional Networks”. In: *Computer Vision – ECCV 2014*. Springer International Publishing, 2014, pp. 818–833.

[Zha+22] Tianping Zhang et al. *Less Is More: Fast Multivariate Time Series Forecasting with Light Sampling-oriented MLP Structures*. 2022. arXiv: `2207.01186 [cs]`. (Visited on 12/21/2023).

[Zhe+14] Yi Zheng et al. “Time Series Classification Using Multi-Channels Deep Convolutional Neural Networks”. In: *Web-Age Information Management*. Springer International Publishing, 2014.

[Zho+20] Jie Zhou et al. “Graph Neural Networks: A Review of Methods and Applications”. In: *AI Open* 1 (2020), pp. 57–81.

[Zho+21] Haoyi Zhou et al. “Informer: Beyond Efficient Transformer for Long Sequence Time-Series Forecasting”. In: *The Thirty-Fifth {AAAI} Conference on Artificial Intelligence*. Vol. 35. AAAI Press, 2021, pp. 11106–11115.

[Zho+22] Tian Zhou et al. “FEDformer: Frequency Enhanced Decomposed Transformer for Long-term Series Forecasting”. In: *Proc. 39th International Conference on Machine Learning (ICML 2022)*. 2022.

[ZY23] Yunhao Zhang and Junchi Yan. “Crossformer: Transformer Utilizing Cross-Dimension Dependency for Multivariate Time Series Forecasting”. In: *International Conference on Learning Representations*. 2023.