Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

ClassFormer: Transformers for Multivariate Time Series Classification

Bachelor's Thesis

Simon J. Bührer

sbuehrer@student.ethz.ch

Chair for Mathematical Information Science MINS
ETH Zürich

Supervisors:
Clemens Hutter
Ines Haymann
Prof. Dr. Helmut Bölcskei

December 4, 2025

Abstract

The world of multivariate time series classification is full of challenges, from figuring out human activities to spotting motion, dealing with ECG issues, and classifying audio. Many methods exist, using dense and convolution approaches. However, when it comes to really long time series, these methods struggle due to their rapidly increasing complexity, which makes the spotting of dependencies over long periods impossible.
This thesis uses Transformers and adapts various techniques used for Attention mechanism, resulting in a high-performing network evaluated on 18 different multivariate datasets. Key innovations include using continuous wavelet transform for frequency enhancement, data patching to reduce model complexity, and a hierarchical three-stage Attention mechanism capturing cross-dependencies across time, dimensions, and frequencies at different scales. Furthermore, data-driven masking was tested, which improves performance during training and introduces additional stability.

Acronyms

AUROC Area Under the Receiver Operating Characteristic Curve. 9, 12

CWT Continuous Wavelet Transform. 5, 14, 15, 19

DDM Data-Driven Masking. iii, 3, 4, 7, 9–11, 17

DPW Dimension-Patch-Wise. iii, 3, 5, 6, 11, 15–17, 19, A-1, C-2

DTW Dynamic Time Warping. 16, 17

MHA Multi-Headed Attention. 7, 10

MSE Mean Squared Error. 5, 9, 12

PCA Principal Component Analysis. 14

ReLU Rectified Linear Unit. 7, 8

t-SNE t-Distributed Stochastic Neighbor Embedding. 14, 15

TARNet Task-Aware Reconstruction Network. 2, 10

TSA Three-Stage Attention. iii, 3, 4, 6–11, A-1

UEA University of East Anglia. iv, 3, 13, 14, 18, B-1

Contents

Chapter 1

Introduction

The introduction of pure Attention-based networks, specifically the Transformer architecture [1], has brought a significant change in the fields of language processing and computer vision. Interestingly, Transformers have also gained a foothold in other areas, as indicated by a detailed survey [2] on their usage in time series for forecasting, classification and anomaly detection.
This thesis focuses on the use of Transformers for the classification of multivariate time series. These time series exhibit distinct characteristics, each contributing to the complexity of the classification task:

- **Multivariate:** Involving more than one input variable, often referred to as dimensions, which adds complexity to the data.
- **Multiscale:** Presence of critical features across various time and frequency scales, capturing the dynamics of the underlying phenomena.
- **Variable-Length:** Time series with varying lengths, offering flexibility at multiple temporal resolutions. Additionally the model must ensure efficiency by maintaining low memory and computational complexity.
- **Cross-Dependencies:** Relevant features are at single or multiple time steps across diverse dimensions.
- **Time-Shift Invariance:** Events are independent of absolute time but dependent on temporal distances between time steps, which requires a more general understanding of temporal patterns.
- **Noise:** Acknowledging and accounting for unwanted modifications introduced to the series, ensuring the model's robustness in real-world scenarios.
- **Class Imbalance:** Scenarios in which the distribution of observations across the known classes is uneven, necessitating the need for adaptive learning strategies.
- **Limited Training Data:** Situations in which only a small amount of training data is available, which can lead to overfitting.

Numerous approaches have been proposed to address such challenges, with prominent methods including dynamic time warping (DTW) [3], Canonical interval forest (CIF) [4], the random convolutional kernel transform (ROCKET) [5], and Residual network (ResNet) [6]. However, none of them attempt a Transformer based approach. This thesis aims to close this gap by utilizing and adapting concepts from existing Transformer-based models that have proven successful in solving similar problems. The code and a tutorial, in form of a Jupyter notebook, is publicly accessible through the GitLab[1] Repository.

[1]GitLab: `https://gitlab.ethz.ch/sbuehrer/transformers-for-multi-modal-time-series-classification/`

Chapter 2

Related Works

2.1 Attention Is All You Need

The "Attention Is All You Need" [1] paper introduces the Transformer for the first time, a revolutionary network architecture for sequence transduction tasks. Unlike traditional models based on recurrent or convolutional neural networks, the Transformer relies solely on Attention mechanisms, eliminating the need for recurrence and convolutions. Experimental results on machine translation tasks demonstrate the superiority of the Transformer in terms of quality, parallelizability, and training efficiency. The study also discusses the limitations of sequential computation in recurrent models and highlights the advantages of Attention mechanisms.

2.2 Crossformer

Crossformer [7] is a novel Transformer-based model designed for multivariate time series forecasting. Unlike existing Transformer models that primarily focus on temporal dependency, Crossformer addresses the crucial cross-dimension dependency by employing Dimension-Segment-Wise embedding and Two-Stage Attention layers. The proposed model establishes a Hierarchical Encoder-Decoder architecture, utilizing information at different scales for enhanced forecasting.

2.3 TARNet

The Task-Aware Reconstruction Network (TARNet) [8] is a model employing Transformers for time series representation learning. Unlike existing approaches that decouple data reconstruction from end-task learning in a pretraining step, TARNet introduces a task-aware reconstruction strategy. This involves designing a data-driven masking approach that samples important timestamps based on self-Attention scores from end-task training. These timestamps are then masked and reconstructed, creating a reconstruction task that is informed by the end task. TARNet alternately trains the reconstruction task and the end task, sharing parameters within a single model. Extensive experiments across numerous classification and regression datasets demonstrate that TARNet consistently outperforms state-of-the-art baseline models across all evaluation metrics.

Chapter 3

Methodology

3.1 Dataset

The University of East Anglia (UEA) archive [9] contains 30 preprocessed multivariate time series datasets, which are already cleaned and preprocessed. The archive includes a predefined train/test split and results from other state-of-the-art classification networks. The data and the results can be loaded using the aeon toolkit. Information about length and size of the dataset is provided in Table B.1. For more detailed information about the problems and other classifiers refer to [10].
All UEA datasets share the same structure, consisting of inputs $X \in \mathbb{R}^{N \times D \times T}$ and labels $Y \in c^N, c \in \{0, \cdots, C\}$, where N is the total number of observation, D the number of dimensions, T the timeseries length and C the number of classes. Time series of unequal length are padded using the symmetric mode in a first processing step.

3.2 Network Architecture

The overall architecture of the ClassFormer is shown in Figure 3.1, and we provide a detailed description of all components in the following subsections. The model starts with a wavelet transform layer (Section 3.2.1), which acts as a preprocessing layer and generates scalograms. The data is then embedded through a Dimension-Patch-Wise (DPW) layer (Section 3.2.2), leading to a hierarchical stack of Three-Stage Attention (TSA) layers (Section 3.2.3). Those outcomes serve as the input for a fully connected layer (Section 3.2.5) responsible for time series classification. In parallel, the same network is used to reconstruct scalograms which are masked using Data-Driven Masking (DDM) (Section 3.3.1). A concise overview of the notations employed is provided in Table A.1.

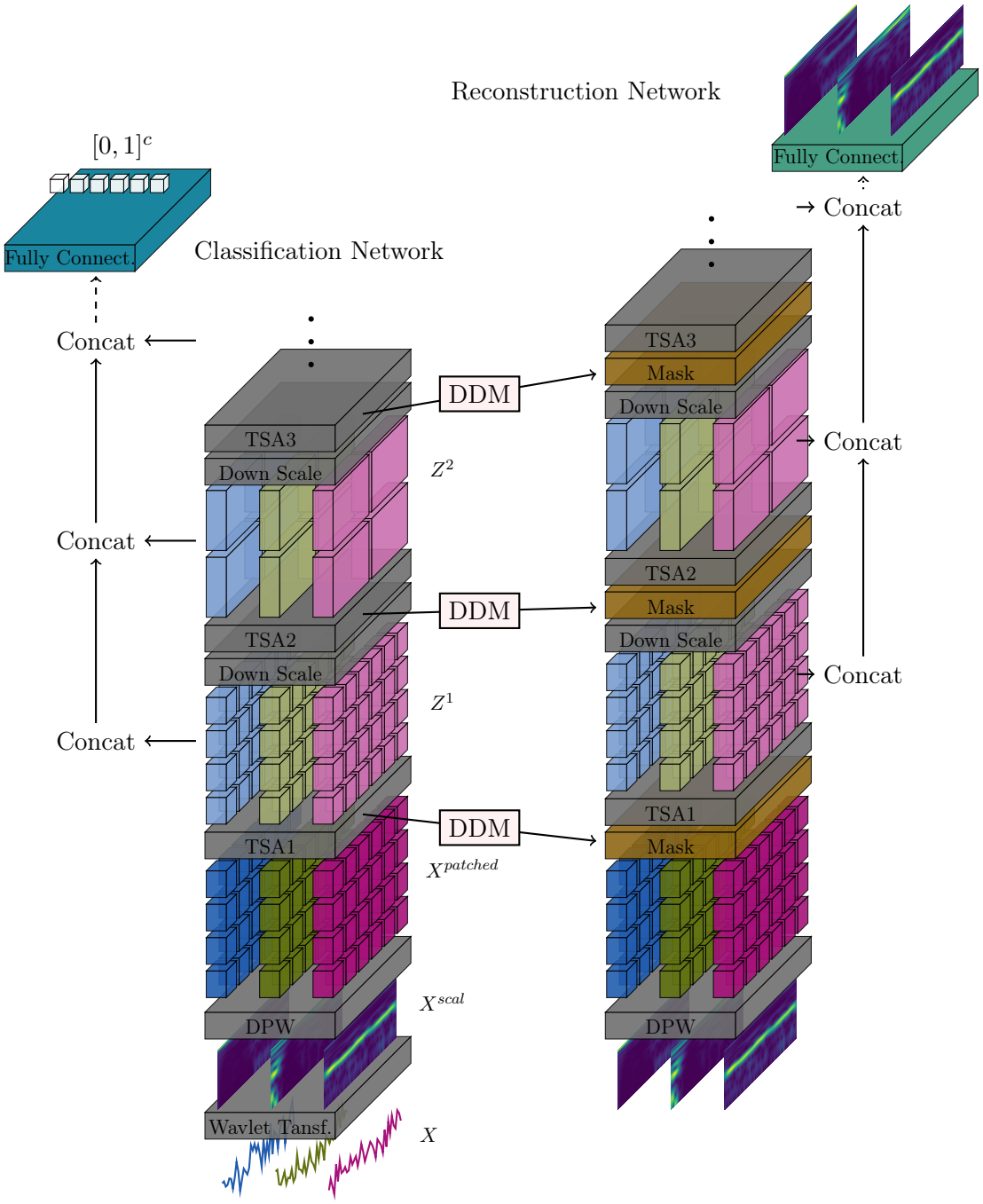


Figure 3.1: Overall architecture of the ClassFormer with three TSA layers. With classification network (left) and reconstruction network (right) sharing the same layers apart from the fully connected layer and the masking, which is created by DDM. The size of each patch denotes the covered time and frequency range. The network employs TSA layers as an Attention mechanism and Downscaling layers to capture dependencies at varying scales. The outcomes from TSA are concatenated and fed into a fully connected layer for classification or reconstruction, respectively.

3.2.1 Wavelet Transform

In many cases, important features of a time series are hidden in its frequency domain. To solve this problem, we propose using the Continuous Wavelet Transform (CWT) following the "FEDformer" [11] and the "Multi-Channel Vision Transformer for Epileptic Seizure Prediction" [12]. This approach offers superior time-frequency localization, increased robustness to noise and adaptability to non-stationary signals, providing a more informative representation.
Given an input $X \in \mathbb{R}^{B\times D\times T}$ the output (scalograms) of the CWT will be $X^{scal} \in \mathbb{R}_{+}^{B\times D\times T\times F}$, where B is the batch size and F the frequency domain size. An example is shown in Figure 3.2.

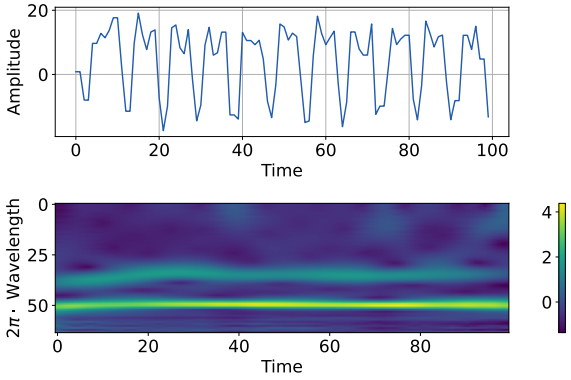


Figure 3.2: Observation 12 in dimension 0 of the "BasicMotions" dataset, annotated as "running". With time series plot (top) and resulting scalogram (bottom) obtained through the wavelet transform.

The complex morlet wavelet from scipy is being used, which is shown in Equation 3.1.

$$\psi(x) = \exp\left(\frac{jwx}{s} - \frac{x^2}{2s^2}\right) \cdot \pi^{-\frac{1}{4}} \cdot \sqrt{\frac{1}{s}} \tag{3.1}$$

Since the transform returns a complex array we compute the absolute value to get a real representation.

3.2.2 Dimension-Patch-Wise Embedding

Before applying Attention to the scalograms, normalization is an important preliminary step, especially since we use the scalograms as labels in the reconstruction process. Failure to normalize them might result in a large Mean Squared Error (MSE), introducing instability. To further counter the problem of exploding gradient, we opt for Batch Normalization [13] instead of layer Normalization.
Next, we break down the scalograms into smaller patches along the time and frequency axes. The reason behind this approach is highlighted in the CrossFormer paper [7], citing two main reasons for adopting this strategy:

- A single value at a step alone provides little information
- Attention values have a tendency to segment, i.e. close data points have similar Attention weights

Futhermore it allows us to reduce the number of weights in the Attention mechanism and is also often used by other Transformer classification networks [14, 10, 7, 15]. One can think of patching as slicing the scalogram into smaller arrays. The Figure 3.3 shows the entire process including the wavelet transform.

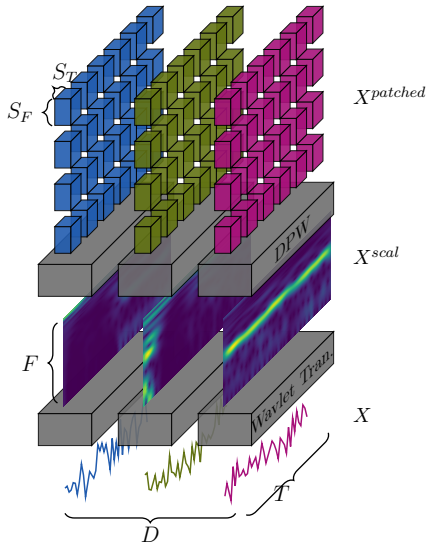


Figure 3.3: Wavelet Transform to process multivariate time series into scalograms. Following this, a subsequent DPW embedding layer creates a 3D array of vectors (collored boxes in the figure) that represents the multivariate time series, with each vector corresponding to a specific patch within the original series.

The DPW embedding patches by taking a set $x_{d,i,j} \in \mathbb{R}^{(S_t \cdot S_F)}$ of scalars from the scalogram $X^{scal} \in \mathbb{R}_+^{D \times T \times F}$. The scalar at dimension d, frequency f and time t in the scalogram X^{scal} is referred to as $x_{d,t,f} \in \mathbb{R}$. This sets $x_{d,i,j}$ are then linearly projected using a learnable matrix $H^{(proj)} \in \mathbb{R}^{d_{model} \times (S_T \cdot S_F)}$.
For simplicity we will from now on refer to the number of patches along the time axis as $L_T = \left\lceil \frac{T}{S_T} \right\rceil$ and along the frequency axis as $L_F = \left\lceil \frac{F}{S_F} \right\rceil$. We also assume for the sake of simplicity that L_F and L_T is a factor of $2^{N_{TSA}}$, where N_{TSA} is the number of TSA implemented and S_t respectively S_F represent the stride along the time and frequency axis. In cases where this statement dos not hold true we pad using zeros and mask them in the TSA.
In a next step a learnable positional embedding $H_{d,i,j}^{(pos)} \in \mathbb{R}^{\times d_{model}}$ is added. The result is a mapping from the scalogram to the patched data $X^{patched} \in \mathbb{R}_+^{D \times L_T \times L_F \times d_{model}}$. In mathematical terms the DPW embedding can be described as shown in Equation 3.2 .

$$\begin{aligned} x_{d,i,j} &= \{x_{d,t,f} \mid (i-1) \cdot S_T \leq t < i \cdot S_T,\ (j-1) \cdot S_F \leq f < j \cdot S_F\} \\ y_{d,i,j} &= H^{(proj)} x_{d,i,j} + H_{d,i,j}^{(pos)} \\ X^{patched} &= \{y_{d,i,j} \mid 1 \leq d \leq D,\ 1 \leq i \leq L_T,\ 1 \leq j \leq L_F\} \end{aligned} \tag{3.2}$$

3.2.3 Three-Stage Attention

In order to capture cross-dependencies, TSA is proposed. We try to capture dependencies between patches by stacking up three Transformer encoder on top of each other, where each pays Attention to another axis, as shown in Figure 3.4. The Transformer encoder are structured as described by the original Transformer [1]. The architecture is comparable

to the Two-Stage-Attention from "CrossFormer"[7], with the difference that there is an additional axis for the frequency domain.

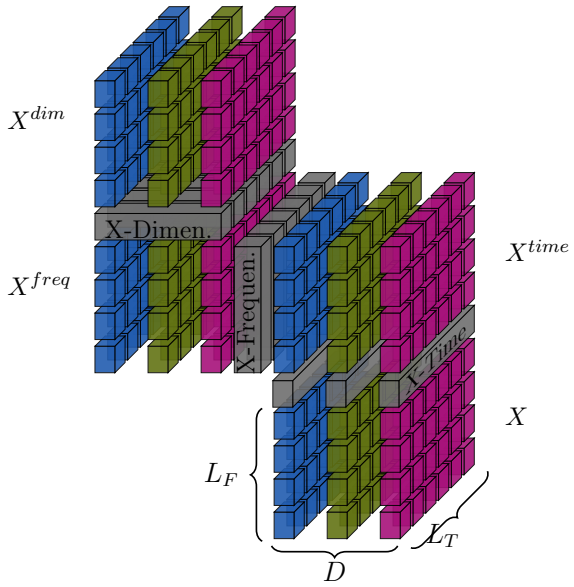


Figure 3.4: Utilizing TSA to handle a 3D vector array, where each box corresponds to a patch of the original scalogram. The entire vector array undergoes the Cross-Time Stage (X-Time), the Cross-Frequency Stage (X-Frequen.) and Cross-Dimension Stage (X-Dimen.) to catch corresponding dependencies.

Given a 3D array of vectors $X \in \mathbb{R}^{D \times L_T \times L_F \times d_{model}}$ the Transformer encoder with Attention on the frequency axis using Multi-Headed Attention (MHA) can be described as shown in Equation 3.3 and Equation 3.4.

$$\tilde{X}^{freq}_{d,i,:} = \text{LayerNorm}(\text{Mask}^{RC} \cdot X_{d,i,:} + \text{MHA}(X_{d,i,:}, X_{d,i,:}, X_{d,i,:}, \text{Mask}^{AS})) \tag{3.3}$$

$$X^{freq}_{d,i,:} = \text{LayerNorm}(\tilde{X}^{freq}_{d,i,:} + \text{Dense}(\tilde{X}^{freq}_{d,i,:})) \tag{3.4}$$

Where LayerNorm represents Layer Normalization, Dense stands for a feed-forward layer, which in our setup is a single dense layer with a Rectified Linear Unit (ReLU) activation function. We use a dropout layer for the dense layer during training to prevent overfitting. The notation MHA(Q, K, V, Mask^{AS}) indicates the multi-head Attention layer, where Q, K, V and Mask^{AS} are used for queries, keys, values and mask for attention score, which is used for DDMand to ignore padding. Mask^{RC} denotes the residual connection mask (Section 3.3.1). All dimensions ($1 \leq d \leq D$) and time steps ($1 \leq i \leq L_T$) share the same MHA layer. By combining Equation 3.3 and Equation 3.4, we formulate the encoder as follows:

$$X^{freq} = \text{Encoder}^{freq}(X) \tag{3.5}$$

The three encoder applied in succession, with Attention on different axis, gives us the TSA:

$$\begin{aligned} X^{time} &= \text{Encoder}^{time}(X) \\ X^{freq} &= \text{Encoder}^{freq}(X^{time}) \\ X^{dim} &= \text{Encoder}^{dim}(X^{freq}) \\ \hat{Z} &= X^{dim} = \text{TSA}(X) \end{aligned} \tag{3.6}$$

Where $\hat{Z}, X \in \mathbb{R}^{D \times L_T \times L_F \times d_{model}}$ represent the input respectively output of the TSA layer.

3.2.4 Downscaling Layer

The Downscaling layer allows us to create hierarchical structures with multiple TSA layers stacked on top of each other, which is necessary to capture information at different scales. The layer merges four patches into one by linearly projecting them with a projection matrix $H_{down} \in \mathbb{R}^{d_{model} \times 4d_{model}}$. Given an input $\hat{Z}^n \in \mathbb{R}^{D \times \frac{L_T}{2^n} \times \frac{L_F}{2^n} \times d_{model}}$ the output becomes $Z^{n+1} \in \mathbb{R}^{D \times \frac{L_T}{2^{n+1}} \times \frac{L_F}{2^{n+1}} \times d_{model}}$.

$$Z^{n+1}_{d,i,j} = H_{down} \begin{bmatrix} \hat{Z}^n_{d,2i,2j} \\ \hat{Z}^n_{d,2i,2j+1} \\ \hat{Z}^n_{d,2i+i,2j} \\ \hat{Z}^n_{d,2i+1,2j+1} \end{bmatrix}, \forall d \in \{1, \cdots, D\},\ i \in \{1, \cdots, \frac{L_T}{2}\},\ j \in \{1, \cdots, \frac{L_F}{2}\} \tag{3.7}$$

$$Z^{n+1} = \text{DownScale}(\hat{Z}^n) \tag{3.8}$$

3.2.5 Fully Connected Layer

The fully connected layer differs between the classification and the reconstruction network as also indicated by the different color in Figure 3.1. For this layer, the output from the TSA serves as the input. Initially, a dense layer is applied individually to each TSA output along the time axis, incorporating ReLU activation. This dense layer features a constant number of hidden nodes, which are subsequently flattened and concatenated.
To enhance the network's robustness, a dropout layer is introduced at this stage, employing the same rate as the dense layer within the TSA. Following the dropout layer, the hidden nodes undergo two additional ReLU layers with either d_{ff} nodes for the classification network or d^{recon}_{ff} for the reconstruction network. In the concluding stage, a final processing step is implemented. This involves either a dense layer with sigmoid activation, followed by a softmax operation to generate one-hot encoded classification labels, or in case of reconstruction, a linear activation function, with its shape matching the pooled normalized scalogram of fixed-size.
By restricting the action of the dense layer to the time axis in the initial layer, we effectively reduce the memory complexity of the weight matrix from $\mathcal{O}(DT)$ to $\mathcal{O}(D+T)$, where D represents the number of dimensions and T denotes the time series length. To maintain this complexity class in the reconstruction network, the scalogram output used for reconstruction undergoes 2D average pooling to achieve a constant size C_T, thereby reducing the memory complexity of the final dense layers weight matrix to $\mathcal{O}(D)$. The entire process is shown in Equation 3.9.

$$\begin{aligned} \tilde{X}^n_{d,:,f} &= \text{Dense}(\tilde{Z}^n_{d,:,f}),\ \forall d \in \{1, \cdots, D\},\ j \in \{1, \cdots, \frac{L_F}{2^n}\},\ n \in \{0, \cdots, N_{TSA}-1\} \\ \tilde{\tilde{X}} &= \text{Flatten}(\tilde{X}^0) \oplus \text{Flatten}(\tilde{X}^1) \oplus \cdots \oplus \text{Flatten}(\tilde{X}^{N_{TSA}-1}) \\ X^{final} &= \text{Dense}(\text{Dense}(\text{Dropout}(\tilde{\tilde{X}}))) \end{aligned} \tag{3.9}$$

In case of reconstruction

$$\hat{Y}^{reco} = \text{Dense}(X^{final})$$

In case of classification

$$\hat{Y}^{class} = \text{Softmax}(\text{Dense}(X^{final}))$$

Where $\tilde{X}^n_{d,:,f} \in \mathbb{R}^{20}$ are the nodes in the first hidden layer for dimension d, frequency f and TSA number n, $\tilde{X}^n \in \mathbb{R}^{D \times \frac{L_F}{2^n} \times d_{model} \times 20}$ is the tensor of all hidden nodes in the first layer over all dimensions and frequencies. $\tilde{\tilde{X}} \in \mathbb{R}^{D \cdot L_F \cdot d_{model} \cdot 20 \cdot \sum_{n=0}^{N_{TSA}-1} \frac{L_F}{2^n}}$ is the concatenation of all the flattened tensors we previously defined. Whereby $\oplus$ denotes the concatenation operator. The output of the of the n-TSA is defined as $\tilde{Z}^n \in \mathbb{R}^{D \times \frac{L_T}{2^n} \times L_F 2^n \times d_{model}}$. Finally $\hat{Y}_{class} \in [0,1]^C$ the classification output with C the number of classes and $\hat{Y}_{reco} \in \mathbb{R}_+^{D \times C_T \times F}$ the reconstructed pooled scalogram.

3.3 Training

In the initial phase of training, a list of all training tasks is generated, and these tasks are subsequently executed by the worker, where the worker corresponds to the number of available GPU devices.
To achieve a split between validation and training during parameter tuning, we use k-fold cross-validation after shuffling the dataset. This approach proves to be particularly effective for small datasets. Following common recommendations [16], we use 10-fold cross-validation. This robust technique ensures a comprehensive evaluation and increases the reliability of the model optimization process. To fine-tune a total of 15 hyper-parameters, we employed a hybrid strategy involving both grid and random search methods. A predefined grid served as the foundation, with random uniform values selected from within this grid. This iterative process was repeated 32 times. Subsequently, the hyper-parameters yielding the best performance metrics were chosen for the final configuration.
The ClassFormer follows a sequential training process. Initially, a training step is conducted for the classification task, utilizing Categorical Crossentropy as the loss function. Afterward, the model is evaluated on the validation data, using metrics such as Accuracy and Area Under the Receiver Operating Characteristic Curve (AUROC). Subsequently, the Attention scores are used to compute the DDM, and a training step is performed for the reconstruction task with MSE as the loss function. Validation is again conducted using MSE as the metric. This entire process is repeated for each epoch across all batches. The summarized representation of this process can be found in Algorithm 1.

Algorithm 1 Training of ClassFormer during Hyper-parameter tuning

Input: ds_{train}, ds_{test}
Parameter: $k, N_{epochs}, t_{patience}$
Output: Model, μ, σ

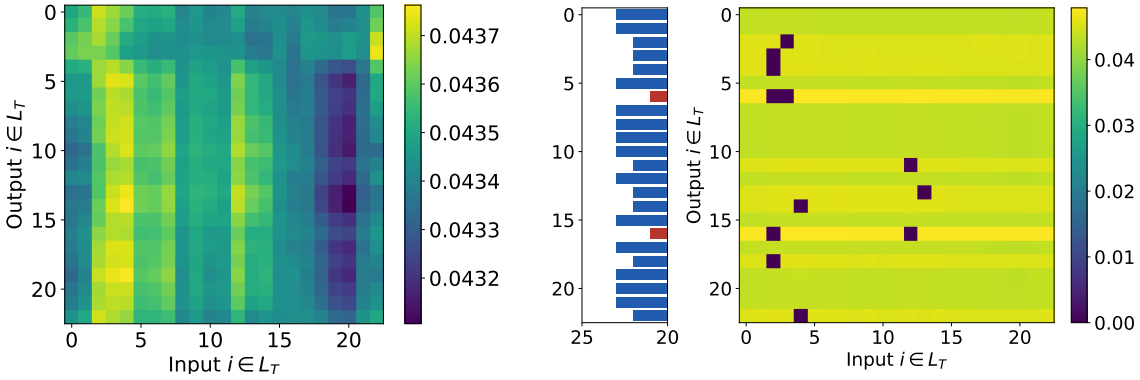
$ds_{train}, ds_{val} \leftarrow$ k-Fold $(k, epoch, ds_{train})$
for $rep = 1, \ldots, k$ **do**
    Model $\leftarrow$ Model.init()
    for $epoch = 1, \ldots, N_{epochs}$ **do**
        S, Model $\leftarrow$ Model.optimize(ds_{train}, recon= $False$) ▷ S: Attention Scores
        $metrics^{class}_{rep} \leftarrow$ Model.eval(ds_{val})
        Mask $\leftarrow$ DDM(S)
        Model $\leftarrow$ Model.optimize(ds_{train}, Mask, recon= $True$)
        $metrics^{recon}_{rep} \leftarrow$ Model.eval(ds_{val})
        if EarlyStopping($metrics^{class}_{rep}, t_{patience}$) **then**
            pass
        end if
    end for
end for
$\mu, \sigma \leftarrow$ mean($metrics$), std($metrics$)
return Model,μ, σ

Following parameter tuning, the final results were determined with training on the entire train dataset without validation data. Subsequently, the test data was used to extract the final metrics. To increase the stability of the results, a replication approach was used. The model was trained and initialized with identical hyper-parameters but with different weights for $N_{\text{rep}} = 3$ iterations. The results reported in Section 4 were derived from the mean and standard deviation obtained from the subsequent inferences on the test datasets.

3.3.1 Data-Driven Masking

Data-Driven Masking (DDM), introduced by TARNet [8], outlines the process of selecting values in the Attention mechanism to mask based on previously calculated Attention scores. In contrast to randomly chosen values, as often employed in pretraining strategies [17] this strategy has an advantage by better aligning with the real task performance.
The DDM we use differs from the one used in TARNet. Instead of masking the input to the MHA directly, we use two boolean masks. One for the attention scores and one for the residual connection, as depicted in Equation 3.3. The mask for the attention score can be described as follows. Initially, the top $\alpha \in [0.05, 0.2]$ values are selected from each Attention score derived from the TSA. An illustrative example is provided in Figure 3.5a. Subsequently, $\beta \in [0.05, 0.2]$ values are randomly chosen after classification and are masked in the Attention score during reconstruction, as demonstrated in Figure 3.5b. The mask has identical dimensions to the attention scores and can therefore simply be fed into the MHA.
This process alone may not suffice due to the presence of a residual connection alongside the MHA. Therefore, the Attention mask is summed up along the output axis, and the lowest β values are selected to form a mask for the residual connection. This mask has the same dimensions as the residual connection and sets all values that are set to false in the mask to 0.

Both masks are used on every transformer encoder. So for each TSA layer and each axis, i.e. time, frequency and dimension. This approach anticipates that the model learns to prioritize the reconstruction of the most critical values for the classification task, enhancing performance by capturing the underlying concepts of the data and emphasizing less dominant features. The results of using DDM are depicted in the ablation study in Section 4.5.



(a) Pre-DDM visualization highlighting pronounced Attention towards patches in the first 3-5 positions, evident from the brighter colors, and relatively reduced Attention towards patches around position 20, as indicated by the darker colors.

(b) Post-DDM visualization. Dark spots indicate masked Attention weights. On the left, the calculated mask for the residual connection is visualized. This mask is generated by counting the number of non-zero values in each row, with the rows designated for masking in the residual connection highlighted in red.

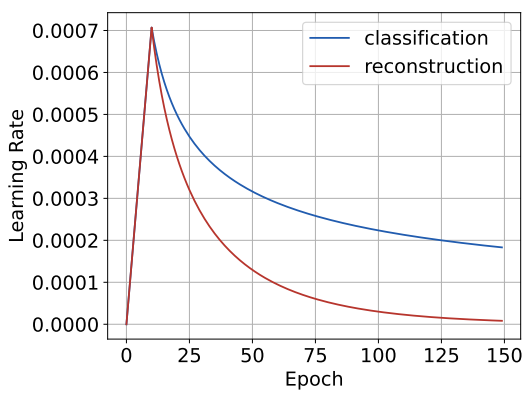
Figure 3.5: Heatmap depicting Attention scores calculated using softmax($\frac{QK^T}{\sqrt{d_k}}$) for the "ArticularyWordRecognition" dataset, specifically for observation 2 and the first head with Attention focused along the time axis in the first TSA. To enhance clarity, padded data from DPW is masked out. Notably, each row of the heatmap sums up to 1, consistent with the applied softmax function.

3.3.2 Optimizer

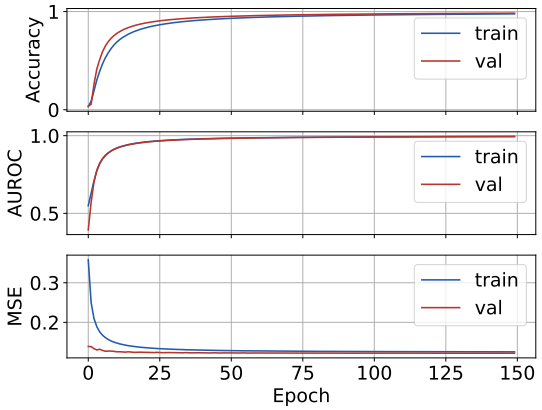
There are two distinct optimizers, one for the reconstruction network and one for the classification network, both leveraging the Adam optimizer [18] with a dynamically adjustable learning rate as also proposed by the original Transformer [1]. The formulation of the learning rate is defined in Equation 3.10.

$$lrate = d_{model}^{-0.5} \cdot \min(epoch^{-0.5} \cdot 0.8^{(epoch-e_w)\cdot s}, epoch \cdot e_w^{-1.5}) \cdot 10^{-2} \tag{3.10}$$

Where *epoch* denotes the current epoch, e_w are the warm-up epochs, and s represents the exponential decay factor applied after the warm-up steps. Specifically, in our classification scenario, s is set to 0, while for reconstruction, s is assigned a value of 0.1. This decay mechanism enables a seamless transition to the classification task after the warm-up phase. A line plot of the learning rates is shown in Figure 3.6a Through empirical testing, we determined that a warm-up period of $e_w = 10$ epochs proved to be a reasonable duration for this initial phase. The adjustment of 10^{-2} was necessary to prevent exploding gradients. A sample history of metrics with these configuration is shown in Figure 3.6b.



(a) Dynamic learning rate schedule for the classification (blue) and the reconstruction network (orange). Notably, the reconstruction learning rate experiences a more rapid decay after the initial 10 warm-up steps, as illustrated in the plot.



(b) Visualization of metric trends across 150 epochs on the "ArticularyWordRecognition" dataset, with the validation (red) and training (blue) metrics. Notably, the AUROC and accuracy behave like $1 - e^{-k \cdot epoch}$, while the reconstruction MSE exhibits an exponential decay pattern.

Figure 3.6: Learn rate schedule and resulting metrics for 150 epochs.

3.3.3 Hardware and Schedule

The training of all models was conducted on 4 NVIDIA GeForce GTX TITAN X GPUs, each equipped with 12GB of memory. The duration of one epoch varied depending on the dataset, ranging from 2 second (ArtialFibrillation) to 2 minute (PhonemeSpectra). On average, it took approximately 5 hours to train all 18 datasets (Table 4.1) for 150 epochs.

Chapter 4

Results

4.1 Performance Metrics Benchmark

The model was tested on 18 multivariate UEA datasets and compared to the baseline of other 8 classifiers [10]. The results are shown in Table 4.1.

	ClassFormer		RISE	TSF	RSF	ResNet	1NN-DTW-I	cBOSS	DrCIF	ROCKET
ArticularyWordRecognition	0.939	± 0.007	0.947	0.950	0.987	0.980	0.927	0.983	0.980	**0.997**
AtrialFibrillation	0.289	± 0.031	0.267	0.267	0.067	**0.333**	**0.333**	0.133	0.200	0.200
BasicMotions	0.983	± 0.024	**1.000**	**1.000**	**1.000**	**1.000**	0.700	0.975	**1.000**	0.975
PhonemeSpectra	0.248	± 0.005	0.272	0.150	0.221	0.324	0.103	0.192	**0.313**	0.294
Cricket	0.861	± 0.030	0.986	0.917	0.986	**1.000**	0.958	0.944	0.986	**1.000**
Epilepsy	**1.000**	± 0.000	0.978	0.978	0.964	0.978	0.710	**1.000**	0.978	0.986
ERing	0.836	± 0.021	0.848	0.933	0.900	0.904	0.896	0.822	**0.985**	0.981
EthanolConcentration	0.392	± 0.005	**0.487**	0.441	0.335	0.232	0.289	0.433	0.654	0.449
FingerMovements	0.490	± 0.000	0.480	0.450	0.560	0.520	0.590	0.500	0.540	**0.620**
HandMovementDirection	0.297	± 0.019	0.203	0.486	0.297	0.297	0.324	0.351	0.392	**0.446**
Handwriting	0.292	± 0.019	0.160	0.376	0.368	**0.624**	0.347	0.478	0.345	0.549
Heartbeat	0.722	± 0.000	0.727	0.722	**0.741**	0.576	0.561	0.722	0.737	0.707
Libras	0.852	± 0.017	0.789	0.767	0.750	**0.933**	0.761	0.800	0.872	0.900
NATOPS	0.796	± 0.009	0.761	0.750	0.833	**0.950**	0.744	0.794	0.806	0.894
RacketSports	0.759	± 0.027	0.789	0.862	0.921	0.901	0.803	0.882	0.862	**0.941**
JapaneseVowels	**0.842**	± 0.046	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
SelfRegulationSCP2	0.513	± 0.022	0.544	0.494	0.472	0.511	0.472	0.439	**0.572**	0.528
PenDigits	0.962	± 0.002	0.868	0.945	0.965	0.994	0.991	0.953	0.993	**0.996**

Table 4.1: Accuracy of the ClassFormer on 18 UEA test datasets compared to 8 benchmark classifiers [10]. The best accuracy score per dataset is marked bold. Additionally the standard deviation (±) for the ClassFormer is given for $N_{Rep} = 3$.

For a more informative comparison of the models, a critical difference diagram [19, 20] was constructed, as illustrated in Figure 4.1. The positions of the models on the diagram reflect their mean ranks across all outcomes. Lower ranks indicate that a treatment consistently outperforms its competitors, while higher ranks suggest lesser success. Treatments with indistinguishable outcomes in terms of statistical significance, as determined by the pairwise Wilcoxon test, are connected with a horizontal line on the diagram.

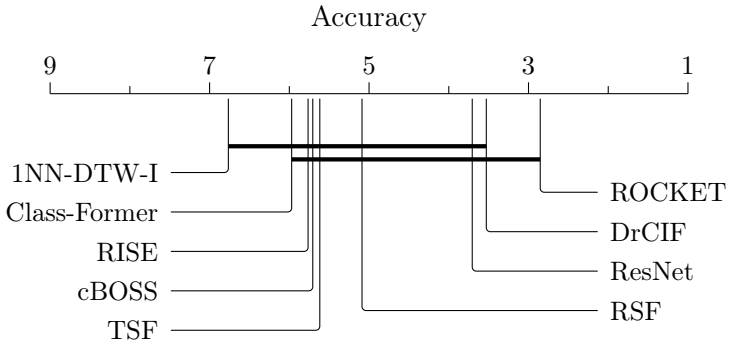


Figure 4.1: Critical difference diagrams [19, 20] for 10 classifiers on 18 multivariate UEA datasets in Table. 4.1 using pairwise Wilcoxon test with $\alpha = 0.05$ to form cliques.

In general, the performance of ClassFormer is not notably inferior, or at least not significantly compared to other classifiers. The difference in accuracy between the classifiers extremely depends on the dataset. Therefore it does not make sense to determine an overall winner. Instead, it is more meaningful to designate a winner for each specific problem. The one-to-one comparison in Figure 4.2 between ClassFormer and the best classifications algorithm to date, "ROCKET" [5], shows that there are also situations in which ClassFormer should be used.

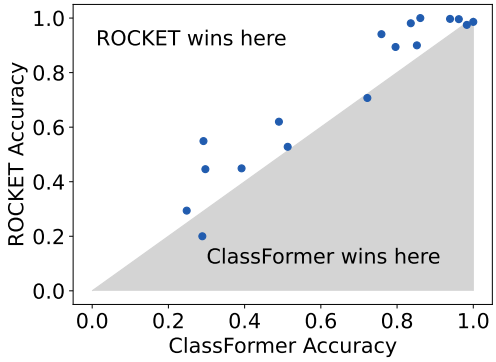
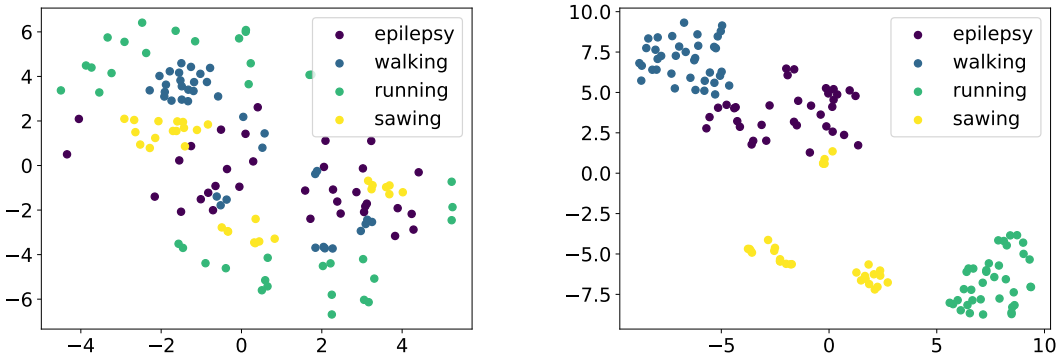


Figure 4.2: Scatter plot of accuracy of 18 UEA problems for ROCKET [5] and ClassFormer

4.2 Analysis of Wavelet Transform

To assess the effectiveness of the CWT, we flatten the time series and the scalogram for each observation to a single multidimensional vector. Subsequently we reduce the number of dimensions to 500 by using Principal Component Analysis (PCA). Following this dimension reduction, we leverage t-Distributed Stochastic Neighbor Embedding (t-SNE) [21] to generate a two-dimensional representation of our multidimensional data, as illustrated in Figure 4.3. This approach allows us to gain insights into the multidimensional data of the CWT, revealing patterns and relationships within the data that might be obscured in the original tensor.



(a) Raw time series before CWT. No clear boundaries between the labeled datapoints are existing.

(b) Scalogram after CWT. The data-point cluster according to their label. Boundaries should be learnable by deep dense layer in multidimensional space.

Figure 4.3: 2D scatter plot of the t-SNE result on the "Epilepsy" test dataset.

The advantages of employing a scalogram become evident in datasets characterized by high levels of quantization (resolution) and where frequency holds underlying significance. This is notably observed in datasets such as "Epilepsy," "BasicMotions," or "ArtialFibrillation". In datasets like "EthanolConcentration", where the data already includes three spectra representing the spectral envelope, using a CWT is unnecessary. In such cases, the model struggles to classify the data successfully because the data's characteristics do not align well with the added complexity introduced by the wavelet transform.

4.3 Analysis of Dimension-Patch-Wise Embedding

The DPW employs learnable weights for position encoding, which are shown in Figure 4.4. The illustration reveals no recognizable pattern across time, frequency or dimension. However neighboring values tend to noticeable distinctions, suggesting that the model tries to maintain separation among them. This behavior resembles to the absolute positional encoding used in Transformers [1]. The learnable positional embedding variables in the ClassFormer are initialized with $X \sim \mathcal{N}(0, 0.05)$. An interesting approach for enhancing the encoding process could be to initialize the weights using learnable Fourier features [22].

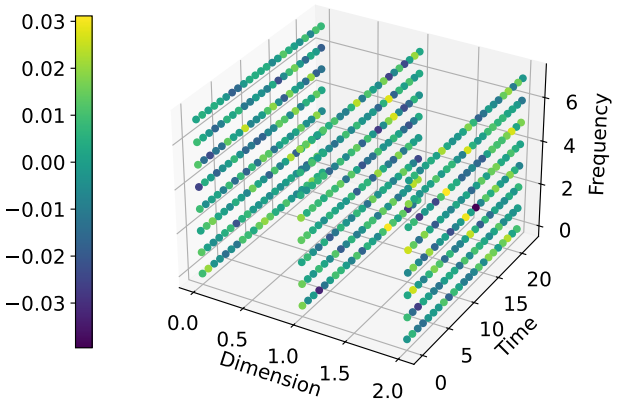
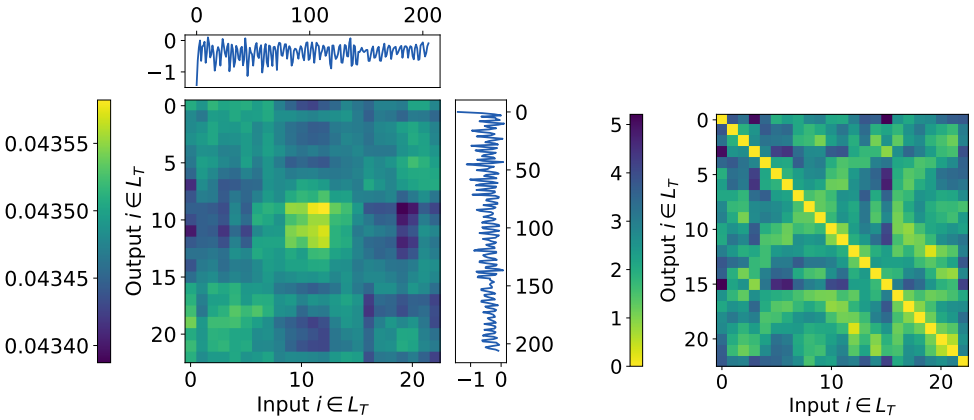


Figure 4.4: 3D scatter plot of positional encoding weights in DPW layer for "Epilepsy" dataset. Darker color indicate negative values while brighter indicate positive.

4.4 Visualization and Analysis of the Attention Score

The Attention scores vary greatly between individual observations and between dimensions as well as frequency and time section. For this reason, clear interpretations of individual Attention scores are hardly possible. However, general statements can be made. As shown in Figure 4.5a the Self-Attention is always local.
Since the model is also trained on reconstruction, one would basically expect attention to be placed on patches with similar values or at least on segments from the original time series that are similar, so that the signal can easily be reconstructed. However, a comparison between Dynamic Time Warping (DTW) [3] with the Attention score shows us that this is not the case and that the ClassFormer is capable of recognizing more complex features. The comparison is shown in Figure 4.5.



(a) Attention score from the mean of all heads. With line plots on the sides showing the respective time series. Self Attention is recognisable in the middle of the heatmap indicated by its brighter color.

(b) DTW [3] calculated from the time series. Brighter color indicates high resemblance while darker color indicates disparity.

Figure 4.5: Sample Observation from "Epilepsy" dataset in second dimension and first frequency range with Attention to patches along its time axis. Patches padded by DPW are masked out for clarification.

A complete sample with all Attention scores along time axis for all dimensions and frequency ranges is shown in Figure C.2. Whereby conspicuous stripes become recognizable. Vertical stripes indicate high respectively low values in the patch and the importance of the time segment over the entire period. While horizontal stripes indicate Attention or Ignorance over the entire period, which rarely occurs.

4.5 Ablation Study

To assess the effectiveness of the key components, a ablation study has been performed by training the entire dataset with identical parameters for an equal number of epochs, excluding the use of these components. The results are shown in Table 4.2.

4.5.1 Ablation of Data-Driven Masking

From the results in Table 4.2 it can be seen that the reconstruction has an overall improving effect in terms of accuracy. In situations where it is better without reconstruction, the metric is usually only a little better, but this can also be argued with insufficient number of epochs or incorrectly set learning rate schedule. The main advantage is to be seen in the standard deviation. The introduction of reconstruction network leads to significantly more stable results. In a specific instance, the reconstruction mechanism demonstrates notably superior outcomes compared to the network operating without it. In the context of "PenDigits", the introduced stability proves to be crucial for the network to skillfully capture the fundamental concepts embedded in the data.

Reconstruction	true		true		false	
Warm-up	true		false		true	
ArticularyWordRecognition	**0.939**	± 0.007	0.931	± 0.008	0.937	± 0.013
AtrialFibrillation	0.289	± 0.031	**0.433**	± 0.033	0.432	± 0.031
BasicMotions	**0.983**	± 0.024	0.975	± 0.030	0.963	± 0.013
PhonemeSpectra	**0.248**	± 0.005	0.239	± 0.008	0.229	± 0.003
Cricket	0.861	± 0.030	**0.910**	± 0.021	0.875	± 0.028
Epilepsy	**1.000**	± 0.000	**1.000**	± 0.000	0.996	± 0.004
ERing	0.836	± 0.021	**0.854**	± 0.006	0.833	± 0.001
EthanolConcentration	**0.392**	± 0.001	0.386	± 0.002	0.367	± 0.002
FingerMovements	**0.490**	± 0.000	**0.490**	± 0.000	0.487	± 0.025
HandMovementDirection	**0.297**	± 0.019	0.189	± 0.054	0.196	± 0.007
Handwriting	0.292	± 0.019	0.263	± 0.018	**0.309**	± 0.022
Heartbeat	**0.722**	± 0.000	**0.722**	± 0.001	0.717	± 0.005
Libras	**0.852**	± 0.017	0.839	± 0.006	0.850	± 0.022
NATOPS	0.796	± 0.009	**0.803**	± 0.003	0.800	± 0.028
RacketSports	**0.759**	± 0.027	0.757	± 0.030	0.714	± 0.003
JapaneseVowels	**0.842**	± 0.046	0.833	± 0.002	0.839	± 0.007
SelfRegulationSCP2	0.513	± 0.022	**0.544**	± 0.017	0.528	± 0.050
PenDigits	**0.962**	± 0.002	0.536	± 0.432	0.960	± 0.004

Table 4.2: Accuracy of the ablation study ClassFormer on 18 UEA test datasets. The best accuracy score per dataset is marked bold. The standard deviation (±) is given for $N_{\text{Rep}} = 3$.

4.5.2 Ablation of Warm-up in Learning Rate Schedule

In the paper "On Layer Normalization in the Transformer Architecture" [23] the need for a warm-up stage is questioned when using Pre-Layer Normalitaion Transformer layer. For ClassFormer a Post-Normalization Layer approach was used as in the original Transformer [1]. To see if the warm-up step or the whole schedule is necessary at all, the whole network was trained again with a constant learning rate of 0.003, which is close to the arithmetic mean of the learning rate as described in section 3.3.2 schedule. The results are shown in Table 4.2. The results indicate that the warm-up phase is necessary with ClassFormer, but the difference is very small. Further experiments with the learning rate are necessary to make significant statements.

Chapter 5

Conclusion

We have proposed ClassFormer as a network of various components. Starting with the representation of the data as a scalogram with subsequent DPW embedding followed by a hierarchical Attention mechanism. We used data driven masking to reduce the amount of labeled data and achieve better performance. For the first time we have a model that is able to mask frequency ranges in a data driven way and on different bandwidths. To summarize, we achieve state-of-the-art results on 18 different data sets and deepen the analysis and interpretation of the individual components.
The obtained results provide a robust foundation for further research in the field of multivariate time series classification. We extensively employed and tested various components, ensuring both their functionality and interpretability in this study.
In future work the focus should be on fine tuning the hyper-parameters. During training, we found significant differences in performance depending on the parameter configuration. Exploring Automated Machine Learning (AutoML), especially Neural Architecture Search (NAS) [24, 25], could be a viable approach for optimizing the hyper-parameters. In addition, a more differentiated investigation of CWT is necessary. The decision to use CWT should depend on the characteristics of the data set. If the dimensions already match the frequency domain, further decomposition may be of limited value. It may also prove beneficial to explore alternative types of wavelets based on data types.
So far, we have experimented with one form of data-driven methodology, and exploring alternative approaches will be a focus of future research. Investigating the applicability of unsupervised learning as a pre-training strategy is another approach worth exploring. Overall, these considerations highlight the potential opportunities for refining and extending the ClassFormer.

References

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

[2] Q. Wen, T. Zhou, C. Zhang, W. Chen, Z. Ma, J. Yan, and L. Sun, "Transformers in time series: A survey," 2023.

[3] S. Salvador and P. Chan, "Toward accurate dynamic time warping in linear time and space," *Intell. Data Anal.*, vol. 11, no. 5, p. 561–580, oct 2007. [Online]. Available: https://dl.acm.org/doi/10.5555/1367985.1367993

[4] M. Middlehurst, J. Large, and A. Bagnall, "The canonical interval forest (cif) classifier for time series classification," in *2020 IEEE International Conference on Big Data (Big Data)*, 2020, pp. 188–195.

[5] A. Dempster, F. Petitjean, and G. I. Webb, "Rocket: exceptionally fast and accurate time series classification using random convolutional kernels," *Data Mining and Knowledge Discovery*, vol. 34, no. 5, p. 1454–1495, Jul. 2020. [Online]. Available: http://dx.doi.org/10.1007/s10618-020-00701-z

[6] Z. Wang, W. Yan, and T. Oates, "Time series classification from scratch with deep neural networks: A strong baseline," 2016.

[7] Y. Zhang and J. Yan, "Crossformer: Transformer utilizing cross-dimension dependency for multivariate time series forecasting," in *The Eleventh International Conference on Learning Representations*, 2023. [Online]. Available: https://openreview.net/forum?id=vSVLM2j9eie

[8] R. R. Chowdhury, X. Zhang, J. Shang, R. K. Gupta, and D. Hong, "Tarnet: Task-aware reconstruction for time-series transformer," in *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, ser. KDD '22. New York, NY, USA: Association for Computing Machinery, 2022, p. 212–220. [Online]. Available: https://doi.org/10.1145/3534678.3539329

[9] A. Bagnall, H. A. Dau, J. Lines, M. Flynn, J. Large, A. Bostrom, P. Southam, and E. Keogh, "The uea multivariate time series classification archive, 2018," 2018.

[10] A. P. Ruiz, M. Flynn, J. Large, M. Middlehurst, and A. Bagnall, "The great multivariate time series classification bake off: a review and experimental evaluation of recent algorithmic advances," *Data Mining and Knowledge Discovery*, vol. 35, no. 2, pp. 401–449, Mar. 2021. [Online]. Available: https://doi.org/10.1007/s10618-020-00727-3

[11] T. Zhou, Z. Ma, Q. Wen, X. Wang, L. Sun, and R. Jin, "Fedformer: Frequency enhanced decomposed transformer for long-term series forecasting," 2022.

[12] R. Hussein, S. Lee, and R. Ward, "Multi-channel vision transformer for epileptic seizure prediction," *Biomedicines*, vol. 10, no. 7, 2022. [Online]. Available: https://www.mdpi.com/2227-9059/10/7/1551

[13] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," 2015.

[14] D. Du, B. Su, and Z. Wei, "Preformer: Predictive transformer with multi-scale segment-wise correlations for long-term time series forecasting," 2022.

[15] Y. Nie, N. H. Nguyen, P. Sinthong, and J. Kalagnanam, "A time series is worth 64 words: Long-term forecasting with transformers," 2023.

[16] D. Anguita, L. Ghelardoni, A. Ghio, L. Oneto, S. Ridella *et al.*, "The'k'in k-fold cross validation." in *ESANN*, 2012, pp. 441–446.

[17] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "Bert: Pre-training of deep bidirectional transformers for language understanding," 2019.

[18] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," 2017.

[19] J. Demšar, "Statistical comparisons of classifiers over multiple data sets," *The Journal of Machine learning research*, vol. 7, no. 1, pp. 1–30, 2006.

[20] A. Benavoli, G. Corani, and F. Mangili, "Should we really use post-hoc tests based on mean-ranks?" *The Journal of Machine Learning Research*, vol. 17, no. 1, pp. 152–161, 2016.

[21] L. van der Maaten and G. Hinton, "Visualizing data using t-sne," *Journal of Machine Learning Research*, vol. 9, no. 86, pp. 2579–2605, 2008. [Online]. Available: http://jmlr.org/papers/v9/vandermaaten08a.html

[22] Y. Li, S. Si, G. Li, C.-J. Hsieh, and S. Bengio, "Learnable fourier features for multi-dimensional spatial positional encoding," 2021.

[23] R. Xiong, Y. Yang, D. He, K. Zheng, S. Zheng, C. Xing, H. Zhang, Y. Lan, L. Wang, and T.-Y. Liu, "On layer normalization in the transformer architecture," 2020.

[24] T. Elsken, J. H. Metzen, and F. Hutter, "Neural architecture search: A survey," 2019.

[25] D. Stamoulis, R. Ding, D. Wang, D. Lymberopoulos, B. Priyantha, J. Liu, and D. Marculescu, "Single-path mobile automl: Efficient convnet design and nas hyperparameter optimization," *IEEE Journal of Selected Topics in Signal Processing*, vol. 14, no. 4, p. 609–622, May 2020. [Online]. Available: http://dx.doi.org/10.1109/JSTSP.2020.2971421

Appendix A

Nomenclature

Notation	Size	Meaning
N	Constant	Number of observations
D	Constant	Number of dimensions
T	Constant	Length of time series
F	Constant	Length of frequency domain
B	Constant	Batch size
C	Constant	Number of classes
S_T	Constant	Stride along time axis
S_F	Constant	Stride along frequency axis
L_T	Constant	Number of patches along time axis
L_F	Constant	Number of patches along frequency axis
d_{model}	Constant	Size of the patch embedding (comparable to d_{model} [1])
N_{TSA}	Constant	Number of TSA
N_{Rep}	Constant	Number of repetition for calculating standard deviation
d_{ff}	Constant	Number of hidden nodes per layer in classification
d_{ff}^{recon}	Constant	Number of hidden nodes per layer in reconstruction
C_T	Constant	Reconstruction output scalogram size along time-axis
X	$\mathbb{R}^{N\times D\times T}$	All time series (network input)
X^{scal}	$\mathbb{R}_+^{D\times T\times F}$	Batch of scalograms
$X^{patched}$	$\mathbb{R}^{D\times L_T\times L_F\times d_{model}}$	Scalogram patched with DPW
X^{freq}	$\mathbb{R}^{D\times L_T\times L_F\times d_{model}}$	Output of Encoder with attention along freqency axis
X^{time}	$\mathbb{R}^{D\times L_T\times L_F\times d_{model}}$	Output of Encoder with attention along time axis
X^{dim}	$\mathbb{R}^{B\times D\times L_T\times L_F\times d_{model}}$	Output of Encoder with attention along dimension axis
Z_n	$\mathbb{R}^{D\cdot\frac{L_T}{2^n}\cdot\frac{L_F}{2^n}\cdot d_{model}}$	Input of the n-TSA
$\tilde{Z}_n$	$\mathbb{R}^{D\cdot\frac{L_T}{2^n}\cdot\frac{L_F}{2^n}\cdot d_{model}}$	Ouput of the n-TSA
$\hat{Y}_{reco}$	$\mathbb{R}_+^{B\times D\times C_T\times F}$	Reconstruction output (pooled scalogram)
$\hat{Y}_{class}$	$[0,1]^C$	Classification output
$\tilde{X}^n_{d,:,f}$	$\mathbb{R}^{20}$	Nodes in the first hidden layer
$\tilde{X}^n$	$\mathbb{R}^{D\times\frac{L_F}{2^n}\times d_{model}\times 20}$	Tensor of all hidden nodes in first layer
$\tilde{\tilde{X}}$	$\mathbb{R}^{D\cdot L_F\cdot d_{model}\cdot 20\cdot\sum_{n=0}^{N_{TSA}-1}\frac{L_F}{2^n}}$	Concatenation of all the flattened hidden node tensors

Table A.1: Meaning of the Notation

Appendix B

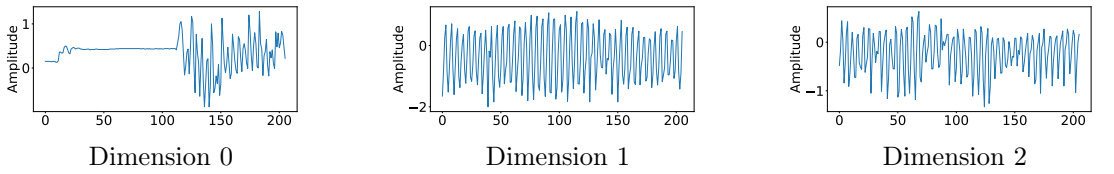
UEA Datasets Overview

Problem	Train Size	Test Size	Dimensions	Series Length	Classes	Normalised	Padded	Missing Values	Class Counts
ArticularyWordRecognition	275	300	9	144	25	true	false	false	11
AtrialFibrillation	15	15	2	640	3	false	false	false	5
BasicMotions	40	40	6	100	4	false	false	false	10
CharacterTrajectories	1422	1436	3	182	20	false	false	false	85
Cricket	108	72	6	1197	12	true	false	false	9
DuckDuckGeese	50	50	1345	270	5	false	false	false	10
EigenWorms	128	131	6	17984	5	false	false	false	55
Epilepsy	137	138	3	206	4	false	false	false	34
EthanolConcentration	261	263	3	1751	4	false	false	false	65
ERing	30	270	4	65	6	false	false	false	5
FaceDetection	5890	3524	144	62	2	false	false	false	2945
FingerMovements	316	100	28	50	2	false	false	false	159
HandMovementDirection	160	74	10	400	4	false	false	false	40
Handwriting	150	850	3	152	26	true	false	false	8
Heartbeat	204	205	61	405	2	false	false	false	57
InsectWingbeat	30000	20000	200	30	10	false	false	false	3000
JapaneseVowels	270	370	12	29	9	false	false	false	30
Libras	180	180	2	45	15	false	false	false	12
LSST	2459	2466	6	36	14	false	false	false	34
MotorImagery	278	100	64	3000	2	false	false	false	139
NATOPS	180	180	24	51	6	false	false	false	30
PenDigits	7494	3498	2	8	10	false	false	false	780
PEMS-SF	267	173	963	144	7	false	false	false	32
Phoneme	3315	3353	11	217	39	false	false	false	85
RacketSports	151	152	6	30	4	false	false	false	39
SelfRegulationSCP1	268	293	6	896	2	false	false	false	135
SelfRegulationSCP2	200	180	7	1152	2	false	false	false	100
SpokenArabicDigits	6599	2199	13	93	10	false	false	false	660
StandWalkJump	12	15	4	2500	3	false	false	false	4
UWaveGestureLibrary	120	320	3	315	8	true	false	false	15

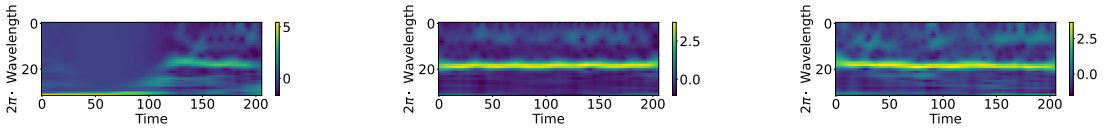
Table B.1: UEA archive multivariate time series classification problems [9]

Appendix C

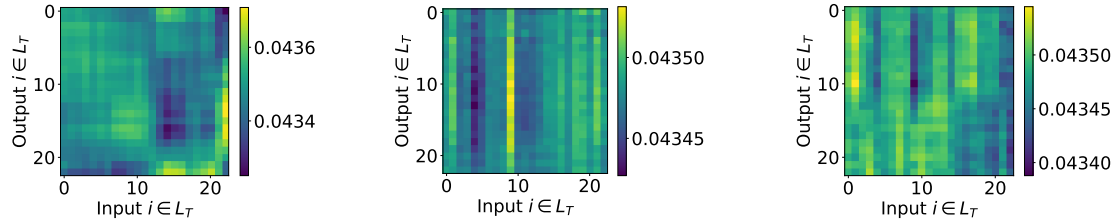
Complete Attention Score Sample



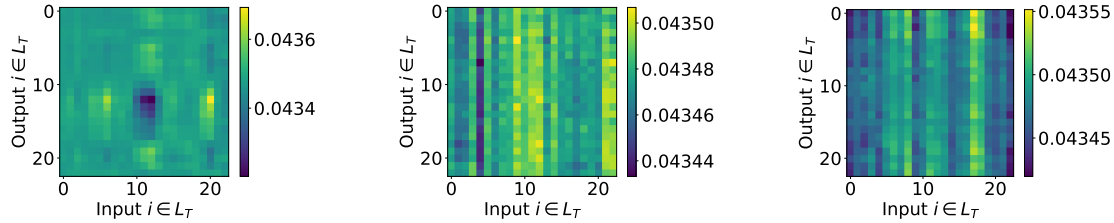
(a) Raw time series for all dimensions.



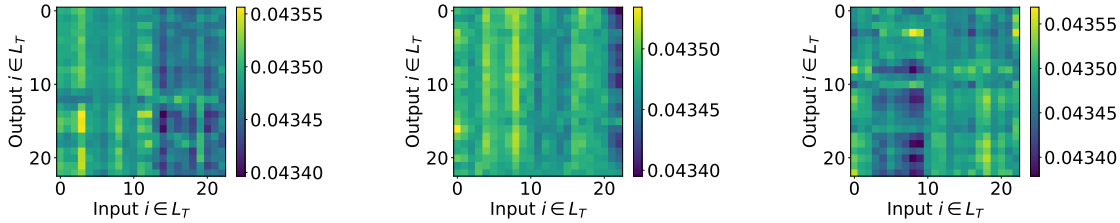
(b) Scalogram calculated from the time series in Figure C.1a.



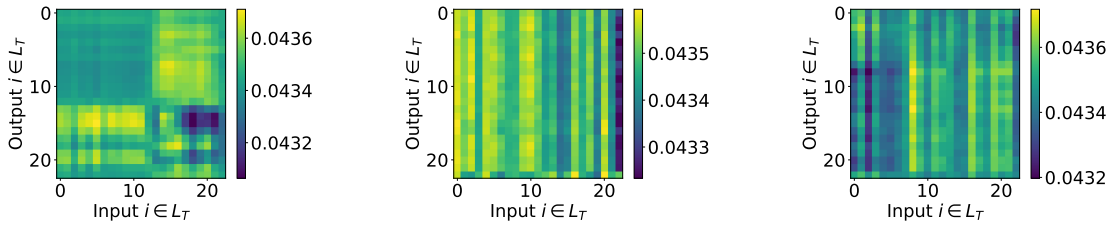
(c) Attention score for frequency range 0-4.



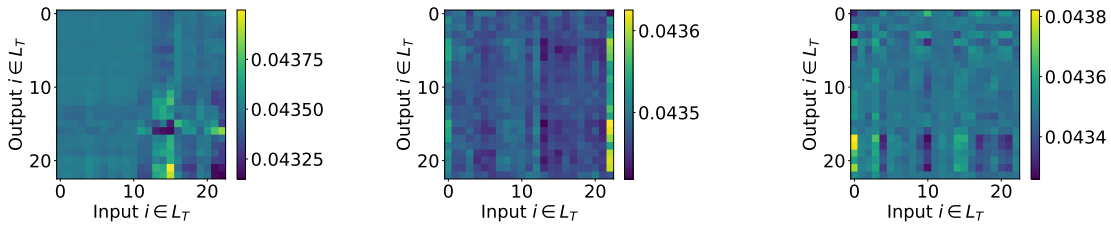
(d) Attention score for frequency range 4-8.



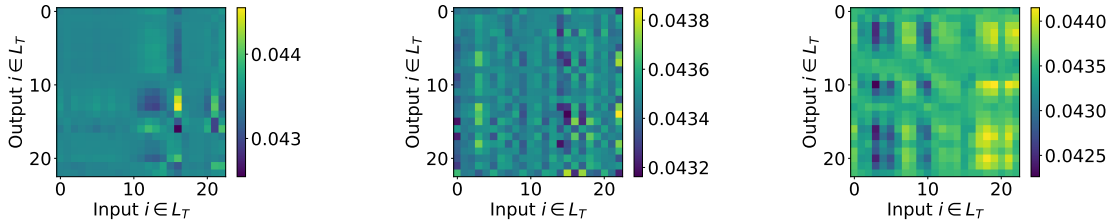
(e) Attention score for frequency range 8-12.



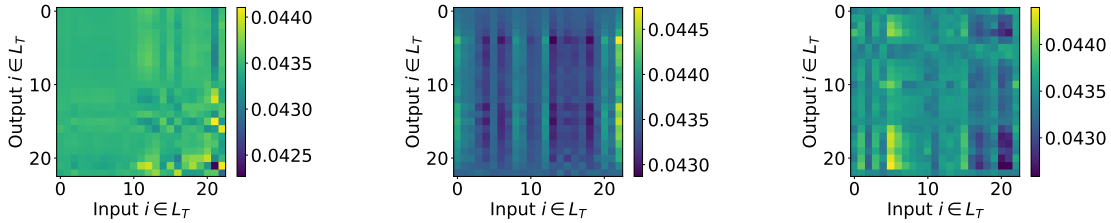
(a) Attention score for frequency range 12-16.



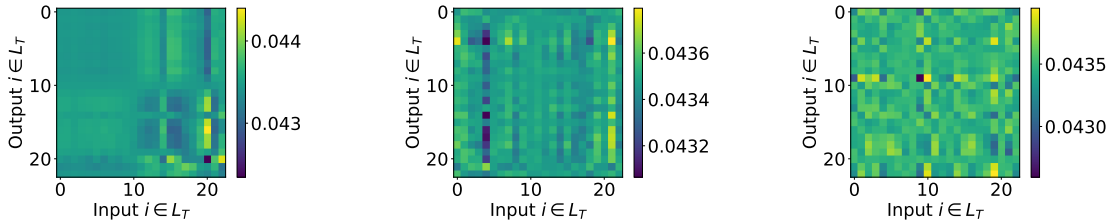
(b) Attention score for frequency range 16-20.



(c) Attention score for frequency range 20-24.



(d) Attention score for frequency range 24-28.



(e) Attention score for frequency range 28-32.

Figure C.2: Complete sample from "Epilepsy" test dataset with all Attention score along time axis for observation 30 labelled as "epilepsy". Patches padded by DPW are masked out for clarification.