Constructing Global Coherence Representations: Identifying Interpretability and Coherences of Transformer Attention in Time Series Data

Leonid Schwenke
Semantic Information Systems Group (SIS)
Osnabrück University, Osnabrück, Germany
leonid.schwenke@uni-osnabrueck.de

Martin Atzmueller
Semantic Information Systems Group (SIS)
Osnabrück University, Osnabrück, Germany
martin.atzmueller@uni-osnabrueck.de

***Abstract*—Transformer models have shown significant advances recently based on the general concept of *Attention* — to focus on specifically important and relevant parts of the input data. However, methods for enhancing their interpretability and explainability are still lacking. This is the problem which we tackle in this paper, to make Multi-Headed Attention more interpretable and explainable for time series classification. We present a method for constructing global coherence representations from Multi-Headed Attention of Transformer architectures. Accordingly, we present abstraction and interpretation methods, leading to intuitive visualizations of the respective attention patterns. We evaluate our proposed approach and the presented methods on several datasets demonstrating their efficacy.**

***Index Terms*—Deep Learning, Transformer, Attention, Visualization, Comprehensibility, Interpretability, Explainability, Time Series Classification, Global Class Representation**

I. Introduction

Transformers [1], [2] are a class of prominent Deep Learning models. One of their key concepts is the use of *Attention* [1], [3], such that — intuitively — the learning process focuses on specifically important and relevant parts of the input data. Such models showed significant advances recently. However, one of their major drawbacks — which they share with general Deep Learning (black box) models — concerns their interpretability and explainability, which are still lacking, e. g., [4]. This becomes even more noticeable in complex domains such as for sequential data, in particular for time series data with continuous value domains etc [5]. Here, explainability, transparency and understandability of the applied complex models are often crucial for practical applications. This concerns, for instance, enhancing trust in a specific modeling approach, the application of the particular model and its results, respectively, e. g., for decision support in sensitive domains [5], [6].

In this paper, we present an approach for constructing global coherence representations (GCRs) from the Multi-Head Attention (MHA) [1], [7] mechanism of the Transformer architecture. In particular, we identify interpretability and coherences in Transformer attention for time series classification. We present abstraction and interpretation methods, supported by visualizations of the respective attention patterns. In this way, the Transformer model itself as well as its task, get more accessible, understandable and comprehensible for humans.

We define a GCR as the aggregation of a set of attention matrices (MHA) into a set of unified symbol-to-symbol matrices (or representations based on them), indicating coherences i. e., relations between pairs of symbols at all considered time steps: that is, there is a strong coherence between two symbols if the attention value — as constructed by the applied attention information extraction method — in our representation is high. Regarding the interpretability, we focus on *global interpretability* [6] of Transformers on classification tasks, i. e., on the level of classes, in contrast to *local* approaches, e. g., [8], which target explainability on local attention matrices. Our proposed approach features three main steps: (1) Data to symbol abstraction, (2) extracting attention information from MHA, (3) creating a Coherence Representation. With this, we aim to obtain GCR models which are interpretable such that their classification decisions can be comprehended by humans, by extracting the attention information on a global per class level. Thus using attention as a medium to better understand the classified data. To the best of the authors' knowledge, this is the first time such a global approach has been proposed. Our contributions are summarized as follows:

1) We propose an approach for abstracting, extracting and identifying interpretability and coherence patterns in Transformer attention on time series data.
2) We present a novel visualization method with different levels of abstraction, enabling coherent and downsized attention to enhance the global interpretability of the underlying classification task.
3) Our evaluation demonstrates the efficacy of the proposed approach, where we show how even simple abstraction methods can extract relevant classification information.

The rest of the paper is organized as follows: Section II summarizes related work. After that, Section III presents our approaches on multiple GCR variants. Next, Section IV discusses our results. Finally, Section V concludes with a summary and interesting directions for future work.

The research leading to this work has been funded by Interreg NWE, project Di-Plast - Digital Circular Economy for the Plastics Industry (NWE729)

Preprint of: Leonid Schwenke and Martin Atzmueller (2021) Constructing Global Coherence Representations: Identifying Interpretability and Coherences of Transformer Attention in Time Series Data. IEEE International Conference on Data Science and Advanced Analytics

II. Related Work

In the following, we outline related work regarding Transformers, as well as visualization options for making them more understandable, and summarize symbolic abstraction methods.

A. Transformers and Attention

Transformers [1], [2] have emerged as a prominent Deep Learning architecture for handling sequential data [1], e. g., for natural language processing (NLP). Transformers have also recently started to be successfully applied to time series problems [9], e. g., addressing efficient architectures [2] and enhanced prediction approaches [10] on time series.

In [11] the pattern filter ability of Transformers in different attention layers and heads was analyzed. They showed that the first Transformer layers perform a more generic pattern detection referred to as global averaging. In contrast, the later layers are the ones still trained at the end while containing more class specific filtering. The authors further reasoned that the MHA is encoding and storing multiple patterns for performing classification. In this work, we build on this principle in order to provide such summarized patterns of interest, which are abstracted into global patterns for specific classes. Here, we specifically focus on the interpretability of those patterns with respect to the individual classes of a classification problem, and provide a process-based approach for this task.

Regarding our data domain, we focus — in particular — on time series tasks which are still quite infrequently researched in combination with Transformers due to their memory limitations; however, Transformers are especially interesting thanks to their ability to handle long-term dependencies [10]. It is important to note that the given limitations of Transformers are currently a rather strong research topic in general; recently many slightly modified Transformer architectures arose which take on different limitations of the original Transformer [2]. In comparison to those approaches we consider the MHA in its original form [1] to create a standard baseline. Therefore, we also do not focus on scalability/runtime, but rather on comparability between the given tasks under consideration of our abstraction, interpretation and visualization approaches.

B. Analysis and Interpretation of Attention on Time Series

Regarding the analysis of *Attention*, most of the methods for MHA analysis and visualization — in order to increase their understandability — are found in the context of computer vision (CV) [12] and NLP [13]–[15]; here, the input is already rather accessible for humans. Those techniques use the properties of attention to highlight relations between — e. g., words [13]–[15] — and can be used to analyze linguistic structures, e. g., cf. [16], or to detect unwanted biases, as described in [13]. While not limited to Transformers, eXplainable Artifical Intelligence (XAI) is an important research direction for providing interpretation and transparency onto the model and/or its results, yet still with several open challenges, e. g., [6], [17]: In particular this holds for time series data, due to the unintuitive nature of the data (compared to NLP or CV) [5]. E. g., on time series applicable methods like class activation mapping (CAM) [18], [19] are based on CNNs and use also a somewhat similar highlighting representation as attention (cf. [12]). Compared to the extra layer in CAM [18], the MHA is a core element in the learning process, such that the later part of the Transformer model *interprets* the attention matrices information, in order to approximate an interpretation. A simple form of Attention was also previously used on time series data with RNNs. [20] found though, that the interpretability properties only work sub-optimally. But they did not look into attention of Transformers itself. On the other hand, [4] showed that MHA is at least partly interpretable even though multiple heads can be pruned without reducing the accuracy [21]. In addition, [22] demonstrated that it is possible to reduce words from sentences via MHA, also showing that attention can abstract important key coherences, while inputs with lower attention can be neglected for the purpose of interpretability. As we discussed and presented in [8], those attention properties can be used to abstract time series data limited by attention thresholds, defined in a human-in-the-loop process. Our combined attention vector provided some form of noise reduction properties to filter out less important information, while maintaining quite a similar accuracy score. Therefore, MHA can in principle be used for transparent interpretations of classification tasks under certain conditions and with specific methods. However, this is currently being limited to local approaches for interpretation. Hence, we extend on this to gain a better understanding of MHA with time series data, while we focus on a *global abstraction and interpretation*. To the best of the authors' knowledge, this is the first time that such a global approach has been proposed.

C. Symbolic Abstraction — SAX

The Symbolic Aggregate Approximation (SAX) is one prominent example of an aggregation and abstraction technique in the area of time series analysis, [5], [23], [24]. It transforms the continuous time series into a discrete symbolic string representation — both facilitating interpretation as well as abstraction, thus resulting in a high-level representation of time series data. Due to the numeric nature of time series [5], [9], [10], most MHA interpretation methods are not as accessible. Thus, the visualization and its interpretation is harder to make sense of for human non-experts. Here SAX can act as connection to the NLP context, which is more accessible for humans and the methods described in Section II-B. In [8], we already applied SAX on time series data with Transformers and showed its abstraction potential and benefit — limited to a local setting, while we focus on a global one in this paper.

III. Method

In this section, we describe our proposed approach. We first outline an according process model capturing the respective processing pipeline. After that, we discuss the applied Transformer model and describe the different GCR variants as well as the methods for global abstraction and interpretation. Finally, we discuss global class validation — as an (integrated) validation step of the respective representation.

A. Process Model.

Figure 1 details the process on our approach for identifying coherences towards interpretability of Transformer attention in time series data. The presented process adapts ideas from visualization approaches, e. g., [8] on Transformer attention; however, our proposed model and pipeline — as already discussed above — does not focus on the local interpretation but on the global one — as a novel contribution and method. In particular, this relates to the extraction of attention data into the proposed coherence representations.

As shown in Figure 1, first the data is preprocessed (scaling, SAX). After that, a Transformer model is trained. Next, we extract the attention information for creating different GCR variations: that is, applying one of the aggregation strategies described in Sections III-C-III-D on all individual attention matrices of the whole trainings set, which are contained in the MHA structure of the Transformer. The resulting matrix/matrices can then be visualized in order to enhance human comprehensibility and understandability of the GCR model, and to further also provide a view on the underlying problem with respect to classification of the time series data. In order to assess our attention coherence representation and visualization — as an abstraction of the original model's attention — we also include a further validation step: We perform a classification with only our global class representation, to show to which extent the extracted representation is representative.

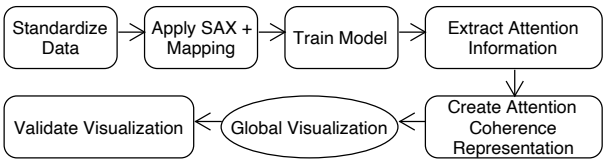

Fig. 1. Processing pipeline, from preprocessing the data, training the model, to extracting, visualizing and validating the obtained representation.

B. Transformer Model

In its original form [1], a Transformer consists of an encoder and a decoder, but for classification problems typically only the encoder is used — as we do in this work. At its core is the MHA which uses attention to learn important points to focus on. This attention comprises an attention matrix, which highlights relations between elements of the inputs. To be concrete, an attention matrix shows basically how strongly one input at a specific position attends another input at another position. These matrices are calculated and applied inside the Scaled Dot-Product, for which multiple ones exist inside the MHA, e. g., see Figure 2. In most cases the so-called self attention is applied, where all inputs to the MHA are the same (i. e., V, K, and Q in Figure 2). Basically, we want to make use of the coherence property of the attention matrices to group the most relevant features into a global class representation.

Figure 2 shows an example of a Transformer encoder corresponding to our applied model. It is important to note that we do not use a classic word embedding, but simply map each symbol to a number in the interval of $[-1, 1]$ as we already suggested in [8], because it preserves position information in the y-axis; otherwise, this would need to be approximated using an embedding, which was, however, in contrast to e. g., NLP problems found to be a sub-optimal solution, e. g., as we further discussed in [8] in detail.

1) Data Preprocessing: First, all data is scaled to unit variance with the Sklearn [25] standard-scaler, before all values of each time series are transformed into symbols using SAX — both fitted on the training data.

To support easy human interpretability while accommodating discriminating power at the same time, we can apply several scales and granularity in the number of symbols. In our experimentation described below, we abstracted to five symbols (5 bins), i. e., to a value range of very low, low, medium, high and very high, where we used a uniform distribution to calculate the bins. It is important to note that by using the SAX algorithm some important classification information can get lost. This needs to be considered when choosing the number of symbols for the respective dataset, which can be optimized in an interactive approach. We defined a mapping of the ordered set of symbols to the interval $[-1, 1]$ and then mapped the values of the original sequence accordingly as we suggested in [8]; in our context, we map the respective 5 symbols to the ordered values (-1, -0.5, 0, 0.5, 1). In this way, we keep the ordering information of the values of the time series, thus preserving the known trend information, rather than approximating it with a word embedding.

2) Model: As for the model[1], we decided to use a quite simple attention model with acceptable accuracies on all datasets, to make the results of all datasets somewhat comparable. We did not optimize each model for one dataset nor for the combination of all, but tried out different parameters, which performed quite similar with respect to the accuracy and abstraction. We used a 2-layered Transformer encoder, based on the original paper [1], with 6 heads, a head size of 6 and a dropout of 0.3, followed by a dense layer which takes in the flattened encoder output. As a final output layer we used a sigmoid-based dense layer with one neuron for each output-class. For training, we used an Adam optimizer with 10000 warm-up steps; for the loss function we took the mean squared error. The architecture of our model is shown in Figure 2.

C. Global Coherence Representation (GCR)

Compared to a NLP or CV context, time series data is even harder to take into relation due to the real-valued input space. As we already suggested in [8], a symbolification of a time series reduces this limitation. By exploiting the special useful characteristic of symbolified time series (compared to NLP), namely the small vocabulary size, we can summarize the model's attention to multiple symbol-based GCRs, which provide a more global view on the accumulated attention values per symbols for each position, as shown in the Figures 3-9.

[1]Model with code is provided at:
https://github.com/cslab-hub/GlobalTimeSeriesCoherenceMatrices

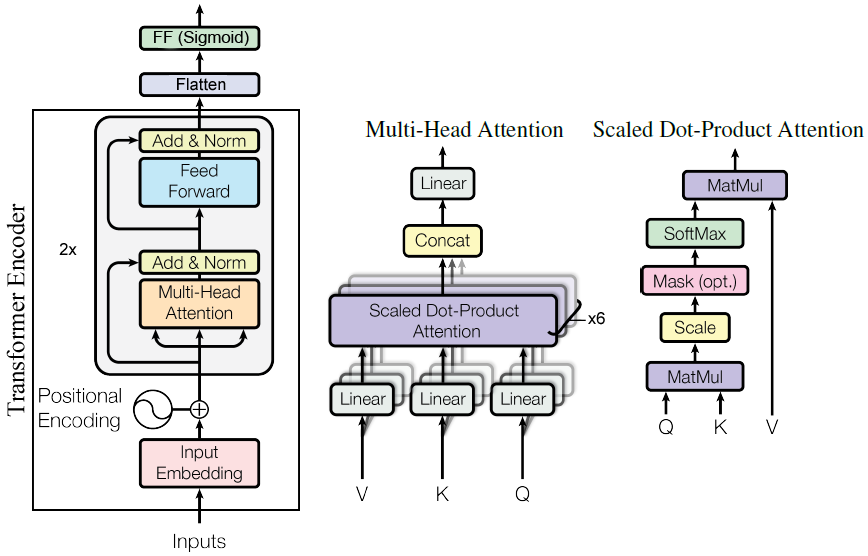

Fig. 2. Our applied architecture including the Transformer encoder, adapted from the architecture in [1].

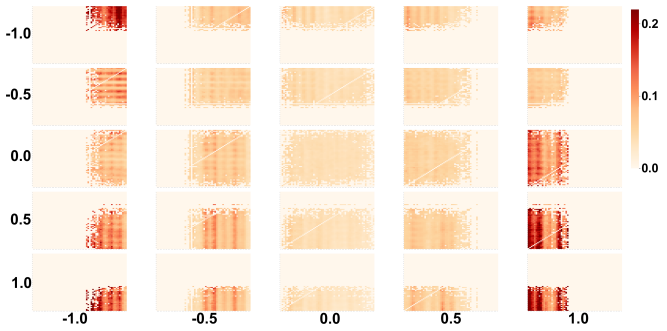

Fig. 3. FCAM via the relative average GSWA and *Max-Sum* LAAM, w.r.t. class 4 from the Synthetic dataset; representing a slowly falling trend.

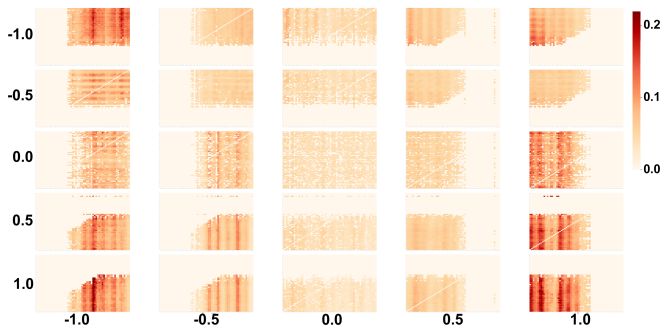

Fig. 4. FCAM via the relative average GSWA and *Max-Sum* LAAM, w.r.t. class 6 from the Synthetic dataset; representing a sudden falling trend.

This means that we can now highlight where an individual symbol is mostly present and how its attention values cohere to other symbols. To give a more class-centered view we create a GCR for each class, hence only using data from the represented class. Below, we present three variants how to build a GCR, namely a *full*, *column reduced*, and *global trend* GCR, where each one builds upon the previous representation and gets more abstract per step. For each representation multiple variations based on different attention aggregations can be created. We present our analysed aggregations in Section III-D after introducing the general idea of each representation. This aggregation can be viewed as a special attention interpretation step, i. e., respective to how a Transformer applies/exploits the attention information in a more complex manner.

1) Full Coherence Attention Matrices (FCAM): The FCAM highlight all relations from symbol x to symbol y, for all $x, y \in V$, with vocabulary V, for each given class via multiple matrices. Hence, it shows a more globally structured overview of how one symbol highlights any other symbol at any given position. Accordingly, we create $n^2 \times k$ matrices where n is the vocabulary size and k is the number of classes. For NLP tasks, this would not be feasible, e. g., due to the large vocabulary sizes (in the range of thousands of elements). The values of one FCAM matrix are aggregated and mapped over to the corresponding attention values (symbol x to y at position (i, j)) — we call this step Global Symbol Wise Aggregation (GSWA) — inside of a Local Attention Aggregation Matrix (LAAM) of each training sample. LAAM is the matrix resulting after aggregating (in any specific way) all attention matrices of the Transformer formed by one input (i. e., referring to the combination step in [8]). More details about the different options for LAAM and GSWA are presented in Section III-D.

Below, we always display the mapped discrete numbers of each symbol to better represent the value relation between each symbol. The examples in Figures 3-4 show how such FCAM look like. It can further be seen how multiple structural features per class emerge which can indicate the global pattern per class. The figures show two rather close classes from the Synthetic dataset. The left class represents a falling trend over a longer period of time, while the right one represents a

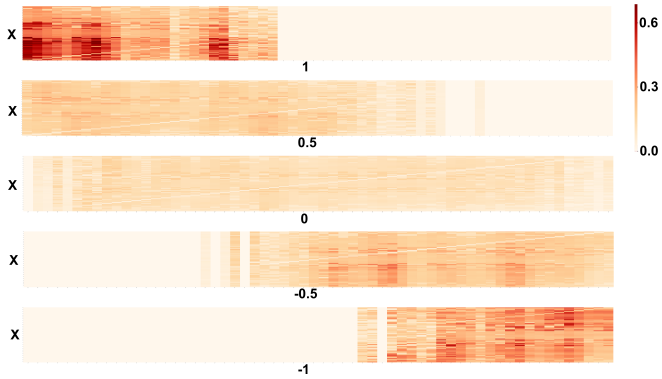

Fig. 5. CRCAM over the relative average GSWA and *Max-Sum* LAAM for class 4 from the Synthetic dataset; representing a slowly falling trend.

Fig. 6. GTM with maximum GVA, r. average GSWA and *Sum-Sum* LAAM for class 4 from the Synthetic dataset; representing a slowly falling trend.

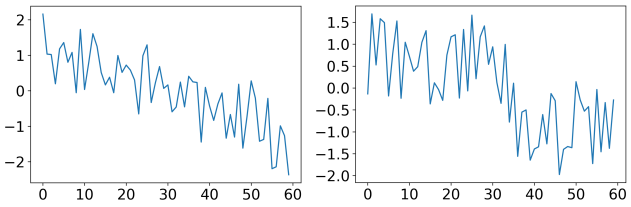

Fig. 7. Example original time series for class 4 (left; slowly falling trend) and class 6 (right; sudden falling trend) from the Synthetic dataset.

rapid trend decrease in the data. The sequence positions of the symbols on the y-axis (rows) and x-axis (columns) both start in the bottom left corner, hence the bottom left shows how the first symbols highlight themselves. A FCAM matrix can be interpreted as follows: Given one symbol and its position, how strongly does it highlight another symbol at any other position. In Figure 3 we observe, e. g., that matrix -1 to -1 is focused on the top right corner while 1 to 1 is focused on the bottom left. This shows, e. g., that -1 is mostly found at the end of the time series and 1 mostly at the beginning, showing a falling trend. In contrast, in Figure 4 a similar distribution can be seen, but the attention blocks in the corners are broader, hence indicating a longer period of 1s and -1s which represents

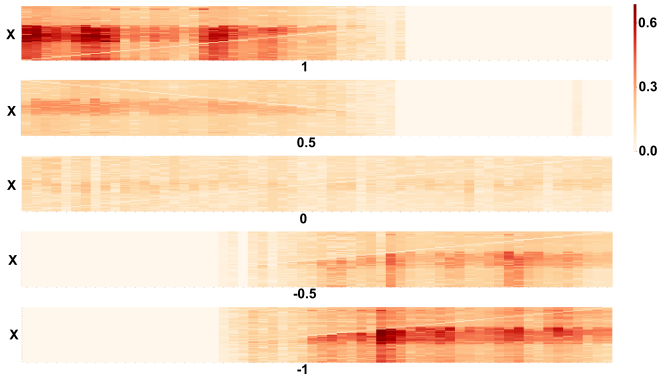

Fig. 8. CRCAM over the relative average GSWA and *Max-Sum* LAAM for class 6 from the Synthetic dataset; representing a sudden trend fall.

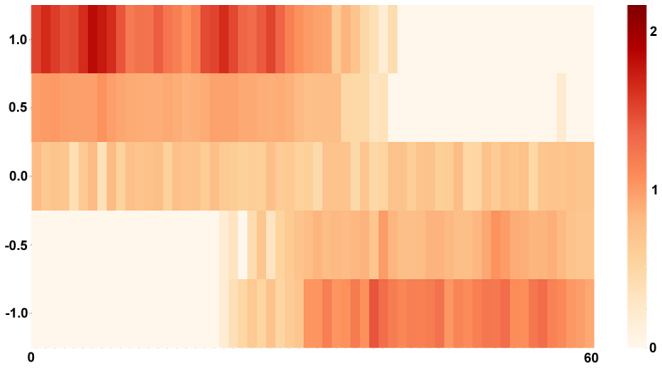

Fig. 9. GTM with maximum GVA, r. average GSWA and *Sum-Sum* LAAM for class 6 from the Synthetic dataset; representing a sudden trend fall.

a later sharper falling trend. Regarding the interpretability of both examples, we observe that the MHA mostly focuses on the duration of 1s and -1s at the beginning and end of the time series for classification. For comparison, Figure 7 shows examples on how the original classes could look like.

2) Column Reduced Coherence Attention Matrices (CRCAM): To bring up the difference of each class even more clearly and more compactly we accumulate all symbols in the y-axis to the one corresponding symbol in the x-axis for all matrices from the FCAM (e. g., the most left column in Figure 3 is combined to one matrix). Correspondingly, we obtain a new set of matrices as shown in the Figures 5-8 with the same class example as before. Each matrix now shows how strongly any possible symbol $x \in V$ at a given position highlights a specific other symbol at each of the possible positions. In this visualization format the difference and general trend of each class can be observed even better; e. g., Figure 5 shows a small iterative down shift in time. Also, the attention blocks of Figure 8 are bigger and somewhat separated into two bigger blocks, indicating a fast falling trend somewhere in the middle.

3) Global Trend Matrix (GTM): Our goal with this reduction is to show the general learned shape of the attention layers. One challenge though is that a single reduced time series shape does not adequately capture the problem's com-

plexity. Therefore, we reduce our matrices further into a single *Global Trend Matrix*, which highlights the typical flow of the time series with an additional visualization of the typical attention strength. Using Global Vector Aggregation (GVA) we aggregate the y-axis of each matrix from the CRCAM to a vector, and combine all vectors to a single matrix based on the relative position of each symbol. An example for class 4 and 6 of the Synthetic dataset can be seen in the Figures 6-9. Here, the trend as seen before is represented in one single simple matrix. When interpreting the matrices it is important to keep in mind that attention is only partly interpretable and therefore the *Global Trend Matrix* needs to be analyzed with caution, which is in general also the case for other similar attention visualizations. Nevertheless, this visualization can compactly represent the learned state of the MHA, presenting the underlying problem and open up further analyzing steps — which we further show and discuss in Sections IV and IV-E.

It is worth noting, that some patterns of each visualization partially result from the typical position of each datapoint for a class — where no data exists there cannot be any attention — but gets further highlighted by the attention strength. To demonstrate the impact of our visualizations, we present our aggregation and validation processes in the next section.

D. Global Class Validation

Overall, we aim to use the GCRs to see if we can interpret them as an approximation for a global classification summary w.r.t. the given classes. One open question is then how to aggregate and evaluate those representations, because during the LAAM construction often simply the average of all attention matrices and their flat attention values is taken [15], [26], [27]. Below, we describe multiple other combination strategies based on simple arithmetic operations and descriptive statistics — which makes them conceptually simple to evaluate and interpret; since our process is extensible in general, these can then be complemented as needed. Furthermore, we introduce an evaluation method for GCR validation, to further analyze and understand the nature of attention in a global interpretability context on time series data. This could also help to filter out less important information for each class visualization, similar to what [8] proposed in a local context. Because the Transformer architecture can somehow use attention to improve the model's accuracy for multiple problems, we just need to find out how to handle attention correctly.

1) Aggregations: To combine the attention matrices of the heads and layers for one given input to get a LAAM, we introduce four aggregations, constructed with all possible combination of the sum and the maximum, when we first intertwine the heads and afterwards the layers. *Max-Sum* for example would stand for the maximum of the attention matrices of all heads per each layer and the sum of the resulting attention matrices of each layer. The resulting LAAM per possible training input are the base on which we calculate the GCR variations described in Section III-C.

For the GSWA step to create the GCR, all attention values inside a LAAM of the training data needs to be mapped into the corresponding symbol-to-symbol structure of the FCAM and aggregated via e. g., the *sum*. Thus, the bigger and more complete the training data is, the better is the global representation. One alternative for the GSWA step is to divide each element of the matrix, i. e., each data point by how often a value is given for this particular matrix position. This is done for each matrix when calculating the FCAM. We call this the *relative average* (r. average). When calculating CRCAM, it is always based on FCAM constructed by one of those two GSWA approaches. We decided to use those as the main strategies because the *sum* strongly depends on the quantity of each data point while reducing the influence of the flat attention value; vice versa for the *relative average*. Therefore, we argue that both approaches can have their benefits depending on the given distribution and application.

When performing the GVA to construct the GTM, we try for each base GSWA — the *sum* or the *r. average* — three options, namely the *maximum*, the *average* and the *median*. It is worth keeping in mind that taking the sum is as meaningful as taking the classic average if each sample count (e. g., head count) is always the same for each matrix point. We therefore did not additionally look into the average or vice versa.

2) Evaluation: For evaluation, we take samples from the test set and classify them with the help of our GCR. For this, each value of the time series is taken and all related attention values inside the GCR will be summed into a score per class. For the FCAM and CRCAM, this means that for each given value, the attention to each other value in the time series is taken, resulting in n^2 attention values per test trial. For the GTM, we just need to consider n attention values. After combining all kinds of matrices to a global representation, it is not straightforward to compare the attention values; thus, we define an attention maximum as the maximal attention score any time series could get per class. The final classification score of a trial is then calculated by summing up all related attention values for each class and dividing them by the class maximum. The class with the highest percentile fit is then predicted for classification. Just taking the flat attention value as a score performed considerably worse, compared to our more advanced/normalized approach.

As a result we obtain 10×4 accuracy scores to show all analyzed aggregations for each dataset, as depicted in the result tables in Section IV. We are aware that those combination methods are not complete and a lot more can be formalized, in principle. However, the ones we propose facilitate simple application and evaluation, already provided good results while giving a better insight into the characteristics of MHA from the Transformer model. In addition, our approach is modular in that sense that it can be extended by further combination strategies for constructing specialized further representations.

IV. Results

In this section, we present the results of our experimentation evaluating our approach using four datasets described below. We applied a standard 5 fold-cross validation procedure. The final validation and test statistics are given by the averages

TABLE I
POWER DATASET: ABSTRACTION ACCURACIES RESULTS ON DIFFERENT GCR VARIATIONS

Power	Max-Max	Max-Sum	Sum-Max	Sum-Sum
FCAM				
Sum	**0.8322** ± 0.0209	0.8256 ± 0.0307	0.8133 ± 0.0395	0.8000 ± 0.0344
R. Average	0.5222 ± 0.0306	0.5410 ± 0.0568	0.5433 ± 0.0565	0.5800 ± 0.0826
CRCAM				
Sum	0.8911 ± 0.0108	**0.8956** ± 0.0289	0.8778 ± 0.0478	0.8489 ± 0.0356
R. Average	0.5833 ± 0.1192	0.6122 ± 0.1217	0.6767 ± 0.0924	0.7367 ± 0.0431
GTM				
Max of sum	0.8978 ± 0.0198	**0.9244** ± 0.0090	0.9100 ± 0.0301	0.9000 ± 0.0423
Max of r. average	0.5111 ± 0.0105	0.5089 ± 0.0130	0.5144 ± 0.0185	0.5656 ± 0.1038
Average of sum	0.8956 ± 0.0151	0.8933 ± 0.0266	0.8678 ± 0.0514	0.8467 ± 0.0384
Average of r. average	0.6011 ± 0.1528	0.6322 ± 0.1634	0.6811 ± 0.1408	0.7511 ± 0.0860
Median of sum	0.8844 ± 0.0244	0.8856 ± 0.0414	0.8633 ± 0.0517	0.8344 ± 0.0432
Median of r. average	0.6089 ± 0.6089	0.6756 ± 0.1645	0.7389 ± 0.1232	0.7811 ± 0.0696

TABLE II
PLANE DATASET: ABSTRACTION ACCURACIES RESULTS ON DIFFERENT GCR VARIATIONS

Plane	Max-Max	Max-Sum	Sum-Max	Sum-Sum
FCAM				
Sum	0.8457 ± 0.0152	**0.8476** ± 0.0159	0.8343 ± 0.0166	**0.8476** ± 0.0159
R. Average	0.7219 ± 0.0505	0.7200 ± 0.0424	0.7486 ± 0.0656	0.7619 ± 0.0532
CRCAM				
Sum	**0.7943** ± 0.0777	0.7905 ± 0.0752	0.7695 ± 0.0809	0.7676 ± 0.0827
R. Average	0.7695 ± 0.0378	0.7410 ± 0.0729	0.7505 ± 0.0947	0.7486 ± 0.0954
GTM				
Max of sum	0.8171 ± 0.0690	0.8019 ± 0.0456	0.8038 ± 0.0714	0.7924 ± 0.0729
Max of r. average	**0.8190** ± 0.0346	0.7848 ± 0.0354	0.7733 ± 0.0279	0.7333 ± 0.0921
Average of sum	0.8000 ± 0.0792	0.7905 ± 0.0752	0.7714 ± 0.0797	0.7657 ± 0.0829
Average of r. average	0.7829 ± 0.1341	0.7695 ± 0.1277	0.7410 ± 0.1165	0.7505 ± 0.1208
Median of sum	0.7733 ± 0.0777	0.7714 ± 0.0745	0.7543 ± 0.0690	0.7714 ± 0.0745
Median of r. average	0.8019 ± 0.0353	0.7867 ± 0.0411	0.8019 ± 0.0647	0.7676 ± 0.0461

TABLE III
AVERAGE ACCURACIES OVER ALL 5 BINS OF OUR MODEL WITH ORIGINAL AND WITH SYMBOLIFIED INPUT.

Dataset	Base Acc.	SAX Acc.
ECG Val.	0.9540 ± 0.0102	0.9440 ± 0.0185
ECG Test	0.9364 ± 0.0030	0.9316 ± 0.0038
Synth Val.	0.9700 ± 0.0306	0.9567 ± 0.0270
Synth Test	0.9573 ± 0.0077	0.9394 ± 0.0056
Plane Val.	0.9714 ± 0.0233	0.9810 ± 0.0233
Plane Test	0.9619 ± 0.0120	0.9771 ± 0.0047
Power Val.	0.6722 ± 0.1442	0.9667 ± 0.0324
Power Test	0.6567 ± 0.1320	0.9478 ± 0.0134

obtained across all 5 folds. Each coherence matrix was evaluated on all fold-models. It is important to note, that we only used a standard state-of-the-art Transformer model (described in Section III-B) which performed quite well on all given datasets, rather than optimizing it for all given classification tasks. With this, we aim to make the results more comparable at a general task-level, in order to inhibit over-optimization influences due to different model parameters.

A. Datasets

For evaluating our presented approach, we applied four datasets. Each univariate dataset has a quite small sequence length — due to the memory complexity of the Transformer.

1) The first one (**Synth**) is the Synthetic Control Chart time series [28], [29], which contains synthetic data for 6 different data trends. The train and test data both contain 300 samples; each sequence has a length of 60 and each class occurrence is balanced.
2) The second one (**ECG**) is an ECG5000 dataset [29], [30], which contains preprocessed ECG samples for 5 classes of length 140. The class distribution is unbalanced and the training size is 500, while the test data amounts to 4500 samples. This makes this dataset quite challenging, especially for the more infrequent classes.
3) For the third dataset (**Plane**), we chose a plane outline dataset [29] with 7 classes and a sequence length of 144. The test and train size is 105; class occurrences are balanced.
4) For the fourth dataset (**Power**), a balanced 2 class Power Consumption dataset [29] is taken which has 180 train and test samples of length 144. It differentiates between the power consumption of a household in warm and cold months.

B. Base Accuracies

Table III shows the validation and test results on all four datasets for the original data and symbolized data. Each model's performances was above 93% which can be observed as quite good without special fine tuning. One exception are the base accuracies on the Power dataset, but because we only focus our analysis on the symbolic data and on the comparability between them, this is acceptable in this context.

TABLE IV
SYNTHETIC CONTROL DATASET (SYNTH): ABSTRACTION ACCURACIES RESULTS ON DIFFERENT GCR VARIATIONS

Synth	Max-Max	Max-Sum	Sum-Max	Sum-Sum
FCAM				
Sum	0.7293 ± 0.0301	0.7253 ± 0.0448	0.7087 ± 0.0445	0.6987 ± 0.0492
R. Average	0.8267 ± 0.0284	**0.8660** ± 0.0217	0.8213 ± 0.0486	0.8533 ± 0.0313
CRCAM				
Sum	0.6427 ± 0.0747	0.6167 ± 0.0822	0.5433 ± 0.1052	0.5127 ± 0.0967
R. Average	0.8713 ± 0.0352	**0.8813** ± 0.0340	0.8367 ± 0.0781	0.8353 ± 0.080
GTM				
Max of sum	0.6827 ± 0.0168	0.6813 ± 0.0396	0.6567 ± 0.0508	0.6140 ± 0.0879
Max of r. average	0.7560 ± 0.0486	0.8547 ± 0.0335	0.8307 ± 0.0585	**0.9080** ± 0.0247
Average of sum	0.6413 ± 0.0742	0.6147 ± 0.0837	0.5427 ± 0.1144	0.5140 ± 0.1133
Average of r. average	0.8560 ± 0.0360	0.8560 ± 0.0402	0.8300 ± 0.0764	0.8253 ± 0.0542
Median of sum	0.6340 ± 0.0667	0.6160 ± 0.0783	0.5333 ± 0.1101	0.5033 ± 0.1099
Median of r. average	0.8147 ± 0.0510	0.8533 ± 0.0402	0.774 ± 0.0875	0.8007 ± 0.0837

TABLE V
ECG DATASET: ABSTRACTION ACCURACIES RESULTS ON DIFFERENT GCR VARIATIONS

ECG	Max-Max	Max-Sum	Sum-Max	Sum-Sum
FCAM				
Sum	0.7976 ± 0.0416	0.7108 ± 0.1110	0.5813 ± 0.2117	0.5606 ± 0.2173
R. Average	0.8627 ± 0.0223	**0.8807** ± 0.0086	0.8715 ± 0.0172	0.8688 ± 0.0264
CRCAM				
Sum	0.4696 ± 0.2186	0.4223 ± 0.1628	0.3542 ± 0.0914	0.3047 ± 0.0401
R. Average	0.8400 ± 0.0452	**0.8418** ± 0.0413	0.8315 ± 0.0440	0.8264 ± 0.0473
GTM				
Max of sum	0.4962 ± 0.2467	0.4749 ± 0.2262	0.4669 ± 0.2167	0.4170 ± 0.1575
Max of r. average	0.8788 ± 0.0122	0.8779 ± 0.0109	0.8764 ± 0.0157	0.8745 ± 0.0168
Average of sum	0.4620 ± 0.2109	0.4138 ± 0.1532	0.3339 ± 0.0711	0.3009 ± 0.0372
Average of r. average	0.8622 ± 0.0297	0.8736 ± 0.0201	0.8777 ± 0.0174	**0.8800** ± 0.0161
Median of sum	0.4600 ± 0.2092	0.4114 ± 0.1504	0.3240 ± 0.0687	0.2976 ± 0.0329
Median of r. average	0.8568 ± 0.0334	0.8706 ± 0.0177	0.8763 ± 0.0174	0.8790 ± 0.0140

C. Coherence Matrices Evaluation

In this section, we show the results from all 10×4 GCR variants — described in Section III-C and III-D — for each dataset from Section IV-A. Comparing the scores of all GCRs, we observe that we can achieve quite good results with up to 92.44% for the Power dataset, while only using the reformatted attention matrices. Therefore, we can use the visualizations with the highest accuracy to somewhat determine the most important positions for the given classification task (at least in the boundaries determined by the evaluation accuracy of the selected representation). The Plane dataset performed worst with up to 84.76% even though it had the highest SAX accuracy. Additionally, it is interesting to see that the best combination approach is depending on the dataset. While the *Max-Sum* LAAM method performed best most of the time — but not always and with mostly rather small differences (row vise) — the GSWA and GVA selection (columns vise) had a huge impact on the final results. Overall the relative average (GSWA) performed best — for the Synth, ECG and quite good on the Plane dataset; however, the Power dataset shows that this is not always the case. Further, the Plane dataset performed quite well on both GSWA methods. When comparing the results of the GTM/GVA variants, we observe even more differences for the best combination method. We would argue this is due to some form of individuality of attention, where attention acts as some sort of preprocessing but still needs a solution on how to interpret the processed data. This could also be further underlined by the findings and hypotheses of [4], [8], [11], [21].

The performance loss between the SAX and the GCR accuracy was highest for the Plane dataset (with 13%) and the lowest for the PowerCon dataset (with 2.3%). This also indicates that attention is not interpretable per se but can be, if processed correctly. Each GCR type (FCAM, CRCAM, GTM) obtained at least an accuracy of 79% (using the best aggregation); this varied between datasets, showing that also choosing the right visualization and matrix combination can be crucial for interpretation and classification. Additionally, some combination methods had a higher variance due to strong outliers, which also shows the influence of the initial training data and the need to find a stable combination method.

D. Low Accuracy Models — Weak Model Boosting

To show that due to the coherence abilities of the MHA quite a lot of classification information is present early, we trained each dataset for 15 epochs and with warm-up, resulting in the poor accuracies shown in Table VII; here, we applied our coherence representation evaluation on the new models.

Table VI shows the accuracy results for the poorly trained models for the *Max-Sum* LAAM — because it performed on the good models overall the best. When the results are compared to the ones in the Tables I-V, we observe that our process still performs quite good, while the baseline model is

TABLE VI
GCR ACCURACIES WITH *Max-Sum* LAAM FOR EACH DATASET, WHEN USING THE POOR ACCURACY MODELS.

Weak Models	Power	Plane	Synth	ECG
FCAM				
Sum	0.7933 ± 0.0163	**0.8210** ± 0.0291	0.5440 ± 0.0259	0.4796 ± 0.1833
R. Average	**0.8822** ± 0.0491	0.7810 ± 0.0413	**0.7947** ± 0.0069	0.8722 ± 0.0396
CRCAM				
Sum	0.8578 ± 0.0174	0.7562 ± 0.0722	0.2067 ± 0.0110	0.2879 ± 0.0168
R. Average	0.8578 ± 0.0114	0.7867 ± 0.1170	0.6260 ± 0.0628	0.7785 ± 0.0590
GTM				
Max of sum	0.8611 ± 0.0183	0.7581 ± 0.0717	0.2107 ± 0.0106	0.2882 ± 0.0167
Max of r. average	0.8567 ± 0.0200	0.7219 ± 0.0983	0.2533 ± 0.0187	**0.8754** ± 0.0201
Average of sum	0.8578 ± 0.0174	0.7562 ± 0.0722	0.2060 ± 0.0112	0.2876 ± 0.0171
Average of r. average	0.7889 ± 0.0176	0.7695 ± 0.1167	0.4780 ± 0.0526	0.8732 ± 0.0171
Median of sum	0.8578 ± 0.0174	0.7543 ± 0.0685	0.2060 ± 0.0112	0.2878 ± 0.0169
Median of r. average	0.8344 ± 0.0330	0.7714 ± 0.0276	0.6820 ± 0.0474	0.8596 ± 0.0236

TABLE VII
AVERAGE ACCURACIES OF ALL 5 BINS OF OUR MODEL WITH ORIGINAL AND WITH SYMBOLIFIED INPUT, BUT ONLY WITH WARM-UP AND 15 TRAINING EPOCHS.

Dataset	Base Acc.	SAX Acc.
ECG Val.	0.6280 ± 0.0911	0.5840 ± 0.0049
ECG Test	0.6345 ± 0.1015	0.5819 ± 0.0036
Synth Val.	0.1833 ± 0.0615	0.1927 ± 0.0456
Synth Test	0.1667 ± 0.0000	0.1667 ± 0.0000
Plane Val.	0.1714 ± 0.0774	0.1295 ± 0.0794
Plane Test	0.1429 ± 0.0522	0.1486 ± 0.0344
Power Val.	0.5000 ± 0.0000	0.5056 ± 0.0086
Power Test	0.5444 ± 0.1033	0.4967 ± 0.1002

doing a bad job. Some aggregations on the weak models even performed better than for the good models, with sometimes even an extreme behavior e. g., the Power dataset where the relative average GSWA performs on par or even better than the sum. But the best GCR accuracy was still always obtained using the good models. This shows that MHA can quite well and early highlight important key features, but we still need to interpret them correctly. When comparing these results to Section IV-C, this further strengthens the hypothesis that the attention weights increase the important coherence points for classification — as some sort of general preprocessing mechanism facilitating interpretability on our representations.

E. Discussion

Our results demonstrate the strong performance of our GCRs and aggregation methods (see Section IV-C) and the ability of the GCRs to highlight important patterns even on weak models (see Section IV-D). It is important that while attention is a general highlighting mechanism regardless of the class — as already argued by [4], [8], [22] — we showed in this work that attention can be interpreted to further enhance common elements per class. [21] showed how attention heads can be pruned; we follow up with an even bigger reduction in attention but show that key elements can still be preserved.

1) Coherence Representations: Figures 3-9 show multiple global class representations — which we call GCRs — with different levels of detail where we argue and demonstrated via example patterns that the general class shape can be interpreted quite well. We further supported this by showing an evaluation technique which can even be used for classification. Because the evaluation method is completely transparent, we can comprehend how the attention values need to be handled, thus showing which data points the model perceives as important for the given task in relation to how good the GCR performs in the evaluation. Based on the same principle as [13], [14], but based on a more compact formalization, we argue that we can thus support the finding of unwanted biases in the trained representation, in a more compact representation. For example, Figure 9 shows a rather weak attention for the value 1 between two higher attention sequences. This could indicate that not enough or no training data covers this space, which could be unwanted and would correspondingly be a first approach to optimize the model. While the different levels of detail from the GCRs did not vary a lot in performance (comparing the best aggregations), they can provide a trade off between details (e. g., for debugging) and a more intuitive presentation.

When looking at our results in the Tables I-V, we observe that it is possible to quite simply construct an aggregation of all attention matrices which have a nice visualization and can even help with the interpretation of the underlying Transformer model. One emerged interesting property of attention which can be quite easily seen in the Tables I-V is that attention seems to have some form of individuality. This would make sense, because each problem needs its own solution.

2) Global Coherence Representation Validation: Even though the combination needs to be somewhat individual, we further have shown that using our approach quite good results can be extracted also using only poorly trained and thus initially weak models. We argue therefore, that this is due to the coherence ability of the MHA which already emerges after the warm up step and thus highlights interesting patterns which can already provide a quite good global class overview as shown in Table VI. This could be due to a general problem dividing property of MHA, where [31] even showed that with not much retraining, new problems can be learned. Other work further solidifies this assumption. We combine therefore multiple works; that with the pattern saving and retrieving ability of MHA [11] and its noise reduction functionality [8], [22], it can act as some form of preprocessing to separate data

and has the ability to highlight interesting pattern in all classes [4], [8], [11]. In other words, the MHA simplifies the problem for the dense layers and thus the dense layer represents an approximated interpretation method. Accordingly, w.r.t. our findings we suspect that a dynamic attention process can be found to interpret attention. Further, we hypothesize that a successful combination can somewhat be used to bootstrap the initial weights of all followup layer for faster learning success. Last but not least we would suspect, that it should be possible to further improve the coherence representations by extracting weights from the Transformer model into it. These hypothesis are to be investigated in future work.

V. Conclusions

In this paper, we presented an approach to make Multi-Headed Attention more interpretable and explainable for time series classification. Our proposed method aims at constructing a GCR from Multi-Headed Attention of Transformer architectures, for classification of time series data. Accordingly, we present abstraction and interpretation methods, leading to intuitive visualizations of the respective attention patterns. We evaluated our proposed approach and the presented methods on four time series datasets. Our results demonstrate the impact and efficacy of the proposed approach for constructing GCRs to identify interpretability and coherences of the Transformer attention. The GCRs show convincing results, in particular also for weak models, where their performance could be significantly improved applying the proposed approach. Additionally we discussed and emphasized the individuality of attention indicated by the different aggregation strategies.

For future work, we aim to apply the proposed approach on further datasets — extending towards multivariate time series, scalability, identity concerns [27] and further XAI measurements [5]. In addition, here the construction of according enhanced combination strategies becomes important, which is another area to consider for future work.

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," *arXiv preprint arXiv:1706.03762*, 2017.

[2] Y. Tay, M. Dehghani, D. Bahri, and D. Metzler, "Efficient transformers: A survey," *arXiv preprint arXiv:2009.06732*, 2020.

[3] S. Chaudhari, V. Mithal, G. Polatkan, and R. Ramanath, "An attentive survey of attention models," *arXiv preprint arXiv:1904.02874*, 2019.

[4] J. Baan, M. ter Hoeve, M. van der Wees, A. Schuth, and M. de Rijke, "Understanding multi-head attention in abstractive summarization," *arXiv preprint arXiv:1911.03898*, 2019.

[5] T. Rojat, R. Puget, D. Filliat, J. Del Ser, R. Gelin, and N. Díaz-Rodríguez, "Explainable artificial intelligence (xai) on timeseries data: A survey," *arXiv preprint arXiv:2104.00950*, 2021.

[6] D. V. Carvalho, E. M. Pereira, and J. S. Cardoso, "Machine learning interpretability: A survey on methods and metrics," *Electronics*, vol. 8, no. 8, p. 832, 2019.

[7] J.-B. Cordonnier, A. Loukas, and M. Jaggi, "Multi-head attention: Collaborate instead of concatenate," *arXiv preprint arXiv:2006.16362*, 2020.

[8] L. Schwenke and M. Atzmueller, "Show me what you're looking for: Visualizing abstracted transformer attention for enhancing their local interpretability on time series data," in *Proc. 34th International Florida Artificial Intelligence Research Society Conference (FLAIRS-2021)*. North Miami Beach, FL, USA: FLAIRS, 2021.

[9] B. Lim, S. O. Arik, N. Loeff, and T. Pfister, "Temporal fusion transformers for interpretable multi-horizon time series forecasting," *arXiv preprint arXiv:1912.09363*, 2019.

[10] S. Li, X. Jin, Y. Xuan, X. Zhou, W. Chen, Y.-X. Wang, and X. Yan, "Enhancing the locality and breaking the memory bottleneck of transformer on time series forecasting," *arXiv preprint arXiv:1907.00235*, 2019.

[11] H. Ramsauer, B. Schäfl, J. Lehner, P. Seidl, M. Widrich, L. Gruber, M. Holzleitner, M. Pavlović, G. K. Sandve, V. Greiff *et al.*, "Hopfield networks is all you need," *arXiv preprint arXiv:2008.02217*, 2020.

[12] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly *et al.*, "An image is worth 16x16 words: Transformers for image recognition at scale," *arXiv preprint arXiv:2010.11929*, 2020.

[13] J. Vig, "Visualizing attention in transformer-based language representation models," *arXiv preprint arXiv:1904.02679*, 2019.

[14] B. Škrlj, N. Eržen, S. Sheehan, S. Luz, M. Robnik-Šikonja, and S. Pollak, "Attviz: Online exploration of self-attention for transparent neural language modeling," *arXiv preprint arXiv:2005.05716*, 2020.

[15] A. M. Braşoveanu and R. Andonie, "Visualizing transformers for nlp: a brief survey," in *2020 24th International Conference Information Visualisation (IV)*. IEEE, 2020, pp. 270–279.

[16] K. Clark, U. Khandelwal, O. Levy, and C. D. Manning, "What does bert look at? an analysis of bert's attention," *arXiv preprint arXiv:1906.04341*, 2019.

[17] S. Vollert, M. Atzmueller, and A. Theissler, "Interpretable machine learning: A brief survey from the predictive maintenance perspective," in *Proc. IEEE International Conference on Emerging Technologies and Factory Automation (ETFA 2021)*. IEEE, 2021.

[18] B. Zhou, A. Khosla, A. Lapedriza, A. Oliva, and A. Torralba, "Learning deep features for discriminative localization," in *Proc. IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 2921–2929.

[19] F. Oviedo, Z. Ren, S. Sun, C. Settens, Z. Liu, N. T. P. Hartono, S. Ramasamy, B. L. DeCost, S. I. Tian, G. Romano *et al.*, "Fast and interpretable classification of small x-ray diffraction datasets using data augmentation and deep neural networks," *npj Computational Materials*, vol. 5, no. 1, pp. 1–9, 2019.

[20] S. Serrano and N. A. Smith, "Is attention interpretable?" *arXiv preprint arXiv:1906.03731*, 2019.

[21] D. Pruthi, M. Gupta, B. Dhingra, G. Neubig, and Z. C. Lipton, "Learning to deceive with attention-based explanations," *arXiv preprint arXiv:1909.07913*, 2019.

[22] C. Wang, X. Liu, and D. Song, "Language models are open knowledge graphs," *arXiv preprint arXiv:2010.11967*, 2020.

[23] J. Lin, E. Keogh, S. Lonardi, and B. Chiu, "A Symbolic Representation of Time Series, with Implications for Streaming Algorithms," in *Proc. 8th ACM SIGMOD Workshop on Research Issues in Data Mining and Knowledge Discovery*. New York, NY, USA: ACM, 2003, pp. 2–11.

[24] J. Lin, E. Keogh, L. Wei, and S. Lonardi, "Experiencing SAX: A Novel Symbolic Representation of Time Series," *Data Mining and Knowledge Discovery*, vol. 15, no. 2, pp. 107–144, 2007.

[25] L. Buitinck, G. Louppe, M. Blondel, F. Pedregosa, A. Mueller, O. Grisel, V. Niculae, P. Prettenhofer, A. Gramfort, J. Grobler *et al.*, "Api design for machine learning software: experiences from the scikit-learn project," *arXiv preprint arXiv:1309.0238*, 2013.

[26] G. Brunner, Y. Liu, D. Pascual, O. Richter, M. Ciaramita, and R. Wattenhofer, "On identifiability in transformers," *arXiv preprint arXiv:1908.04211*, 2019.

[27] S. Abnar and W. Zuidema, "Quantifying attention flow in transformers," *arXiv preprint arXiv:2005.00928*, 2020.

[28] R. J. Alcock, Y. Manolopoulos *et al.*, "Time-series similarity queries employing a feature-based approach," in *7th Hellenic conference on informatics*, 1999, pp. 27–29.

[29] H. A. Dau, E. Keogh, K. Kamgar, C.-C. M. Yeh, Y. Zhu, S. Gharghabi, C. A. Ratanamahatana, Yanping, B. Hu, N. Begum, A. Bagnall, A. Mueen, G. Batista, and Hexagon-ML, "The ucr time series classification archive," October 2018.

[30] A. L. Goldberger, L. A. Amaral, L. Glass, J. M. Hausdorff, P. C. Ivanov, R. G. Mark, J. E. Mietus, G. B. Moody, C.-K. Peng, and H. E. Stanley, "Physiobank, physiotoolkit, and physionet: components of a new research resource for complex physiologic signals," *circulation*, vol. 101, no. 23, pp. e215–e220, 2000.

[31] K. Lu, A. Grover, P. Abbeel, and I. Mordatch, "Pretrained transformers as universal computation engines," *arXiv preprint arXiv:2103.05247*, 2021.

APPENDIX A
ADDITIONAL GTM VISUALIZATIONS

To show some further examples of our visualizations, we present the GTM for all classes from all datasets which performed overall the best — here, always the last fold is visualized. Figure 10 shows all six trend classes of the Synthetic dataset. The Power dataset (2 classes) is visualized in Figure 11. Next, Figure 12 visualizes all 7 plane shapes from the Planes dataset. Last but not least, Figure 13 shows 4 of the 6 classes from the ECG dataset. For class 5 and 6, no training sample was present in the given fold, which shows how unbalanced some class occurrences for this dataset were.

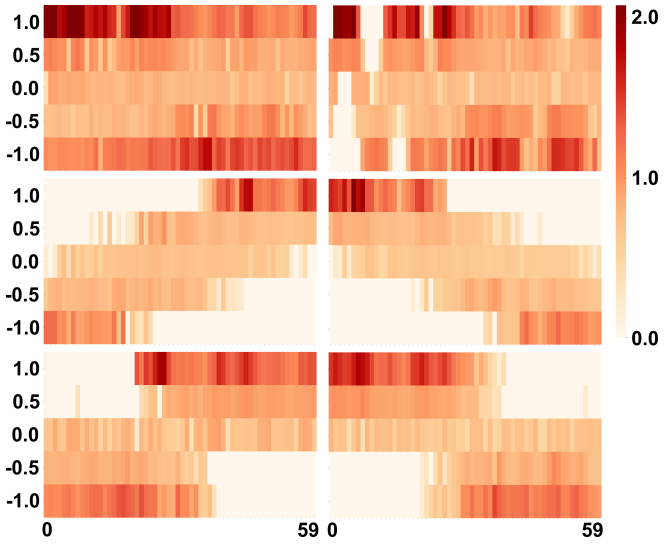

Fig. 10. Global Trend Matrix for all classes of the Synthetic dataset with *Sum-Sum* LAAM and maximum GVA of the relative average GSWA.

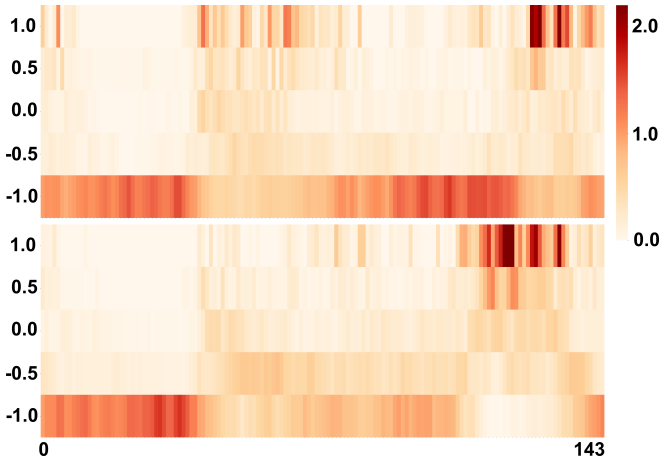

Fig. 11. Global Trend Matrix for all classes of the Power dataset with *Max-Sum* LAAM and maximum GVA of the sum GSWA.

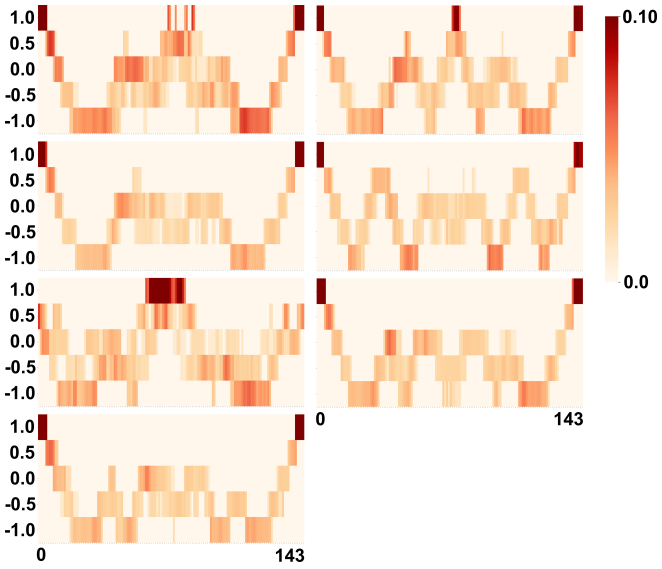

Fig. 12. Global Trend Matrix for all classes of the Plane dataset with *Max-Max* LAAM and maximum GVA of the relative average GSWA.

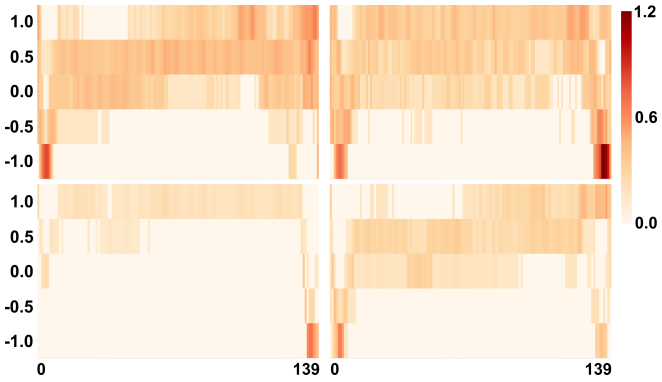

Fig. 13. Global Trend Matrix for all classes of the ECG dataset with *Sum-Sum* LAAM and average GVA of the relative average GSWA.

APPENDIX B
REPRODUCIBILITY

Our model is provided in a git repository[2]. Figure 2 provides a simple overview of our model, but to further enhance reproducibility we provide the following textual summary: We used *sklearn.preprocessing.StandardScaler* to standardize the data, which is fitted on the training data. Afterwards we applied *pyts.approximation.SymbolicAggregateApproximation* with 5 bins and the uniform strategy to symbolize the data. The symbols are mapped to the interval [-1,1] with equal distances. The processed data is now trained with a Transformer encoder model where *tensorflow_addons.layers.MultiHeadAttention* is used for the MHA. The Transformer encoder has 2 attention layers, a d_model of 16, 6 heads and a head size of 6 with a dropout of 0.3. We applied a positional encoding but used

[2]Source/code for reproducibility is available under: https://github.com/cslab-hub/GlobalTimeSeriesCoherenceMatrices

```
# attentions is a matrix with the shape (train_size, layers, heads, input_size, input_size), where:
#   train_size is the number of training samples
#   layers the number of layers in the model
#   heads the number of heads in the model
#   input_size the length of a training sample
# predicting the attention and labels for the train data with the trained model
prediction, attentions = model.predict(train_data)

# Create Max-Sum LAAM
# aggregate heads with max (as in max from the numpy package)
reduced_heads = max(attentions, axis=2)
# aggregate layers with sum (as in sum from the numpy package) to receive the LAAM
reduced_attentions = sum(reduced_heads, axis=1)

for train_index in train_data:
    original_time_series = train_data[train_index]
    attention_matrix = reduced_attentions[train_index]

    # initEmptyFCAM() initializes the following dict/array structure with zeros to do r. average GSWA
    # attention_count represents how often one specific position in all attention matrices was present in train_data.
    # r_average_FCAM represents the relative average attention for each class for each possible symbol pair
    # Both are in shape (labels, symbols, symbols, input_size, input_size) where:
    #   labels is the number classes of the classification task
    #   symbols is the number of abstracted symbols
    # r_average_CRCAM represents the column reduced r_average_FCAM with shape  (labels, symbols, input_size, input_size)
    attention_count, r_average_FCAM, r_average_CRCAM = initEmptyFCAM()

    for i in range(input_size):
        for j in range(input_size):

            #count all non-zero attentions
            if attention_matrix[i][j] != 0:
                #train_labels are the corresponding labels for the train input
                attention_count[train_labels][original_time_series[i]][original_time_series[j]][i][j] += 1

            #summing up the attention score for each position
            r_average_FCAM[train_labels][original_time_series[i]][original_time_series[j]][i][j] += attention_matrix[i][j]

    #iterate over all elements to calculate the relative average GSWA
    for label in r_average_FCAM.keys():
        for to_symbol in r_average_FCAM[lable].keys():
            for from_symbol in r_average_FCAM[lable].keys():
                for i in range(input_size):
                    for j in range(input_size):
                        current_count = attention_count[label][from_symbol][to_symbol][i][j]
                        current_attention_value =  r_average_FCAM[label][from_symbol][to_symbol][i][j]
                        r_average_FCAM[label][from_symbol][to_symbol][i][j] = current_attention_value/current_count
                        r_average_CRCAM[label][to_symbol][i][j] += r_average_FCAM[label][from_symbol][to_symbol][i][j]

            # calculate the GTM with the maximum, average and median GVA
            # (max, median and average are implemented as their counterparts in numpy)
            r_average_maximum_GTM = max(r_average_CRCAM[label][to_symbol], axis=0)
            r_average_average_GTM = average(r_average_CRCAM[label][to_symbol], axis=0)
            r_average_median_GTM = median(r_average_CRCAM[label][to_symbol], axis=0)
```

Fig. 14. Pseudo Python code: How to calculate the FCAM and CRCAM for different strategies; here, for the relative average and *Max-Sum* combination and all of the three proposed corresponding GTM variations.

no additional embedding. After processing via the encoder, we flattened the output, applied a 0.3 dropout and used a fully connected neural network layer with a sigmoid function to obtain the output classes. The number of outputs is equal to the number of classes of the trained dataset. As an error function we applied the mean squared error. We used 10000 warm-up steps and the standard Adam optimizer. Our batch size was 50, we had at most 500 epochs combined with early stopping where patience was 70 and the minimal delta 0. We selected the best model with the highest validation accuracy and smallest validation loss as the final model. To construct the coherence visualizations, we used the combinations described in Section III-C and Section III-D in detail.

To summarize how to compute the LAAM, GSWA und GVA we provide — besides our original implementation, pseudo code in Figure 14. Here, we show how to construct the FCAM, CRCAM representations — with the relative average and *Max-Sum* combination and all 3 corresponding GTM variations.

For calculation we used Python version 3.7.3, a 1080 ti and 32 GB RAM, with the following packages (with those versions):

1) tensorflow==2.2.0
2) tensorflow_addons==0.11.2
3) tensorflow_probability==0.7.0
4) seaborn==0.10.1
5) scipy==1.4.1
6) scikit-learn==0.23.2
7) pyts==0.11.0
8) pandas==1.0.0
9) numpy==1.18.5
10) matplotlib==3.3.1