# Data Streams: Investigating Data Structures for Multivariate Asynchronous Time Series Prediction Problems

Christopher Vox[1], David Broneske[2], Istiaque Mannafee Shaikat[1] and Gunter Saake[3]
[1]*Volkswagen AG, Wolfsburg, Germany*
[2]*German Centre for Higher Education Research and Science Studies, Hannover, Germany*
[3]*Otto-von-Guericke-Universität Magdeburg, Magdeburg, Germany*

Keywords: Multivariate Asynchronous Time Series, Deep Learning, Convolutional Neural Network, Recurrent Neural Network.

Abstract: Time series data are used in many practical applications, such as in the area of weather forecasting or in the automotive industry to predict the aging of a vehicle component. For these practical applications, multivariate time series are present, which are synchronous or asynchronous. Asynchronicity can be caused by different record frequencies of sensors and often causes challenges to the efficient processing of data in data analytics tasks. In the area of deep learning, several methods are used to preprocess the data for the respective models appropriately. Sometimes these data preprocessing methods result in a change of data distribution and thus, to an introduction of data based bias.
Therefore, we review different data structures for deep learning with multivariate, asynchronous time series and we introduce a lightweight data structure which utilizes the idea of stacking asynchronous data for deep learning problems. As data structure we create the Triplet-Stream with decreased memory footprint, which we evaluate for one classification problem and one regression problem. The Triplet-Stream enables excellent performance on all datasets compared to current approaches.



## 1 INTRODUCTION

Time series data are generated, gathered and processed all over the world. Thereby, these data are of diverse nature, as there are, among others, periodically sent short text messages, employees availability recordings, measurements of patient data for clinical purposes, but also manifold sensor measurements, which enable digitalized and autonomous services. If we consider one recording at a specific time point, then this measurement can have different dimensionality, such as images or text messages result in value matrices and value vectors, whereas temperature sensors generate one value per time point. The time series evaluated within this work can be represented by the measurement information $I: \{(x,t) \subseteq I \mid x_{(n,m)} \in \mathbb{R}^z, t \in \mathbb{R}^+\}$. The measurement $x$ is defined by the three-dimensional space spanned over $n,m,z$. In the following, we limit ourselves to measurements of one dimensional nature ($n = m = z = 1$) at a certain time point $t$.

A specific challenge is that the introduced information $I$ can have different properties such as missing or incorrect measurements (Sun et al., 2020; Weerakody et al., 2021). Furthermore, measurements from different sources can have different record frequencies, which causes an asynchronous multivariate space (Vox et al., 2022). According to (Wu et al., 2018), multivariate time series data can be characterized as synchronous or asynchronous, as regular or irregular and as co-cardinal or multi-cardinal. Asynchronicity can be defined as multivariate time series state, where the measurements of the univariate time series are not synchronized to a uniform time pattern and thus not aligned on the time domain. Moreover, when the amount of measurements of univariate time series within a multivariate time series space are different, the time series space can be described as

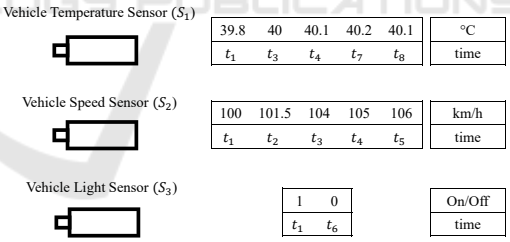

Figure 1: Characteristic of vehicle sensor data.



Vox, C., Broneske, D., Shaikat, I. and Saake, G.
Data Streams: Investigating Data Structures for Multivariate Asynchronous Time Series Prediction Problems.
DOI: 10.5220/0011737300003411
In *Proceedings of the 12th International Conference on Pattern Recognition Applications and Methods (ICPRAM 2023)*, pages 686-696
ISBN: 978-989-758-626-2; ISSN: 2184-4313

multi-cardinal. In addition, when all univariate time series have the same amount of measurement points, the multivariate time series space is co-cardinal.

The data source we investigate is shown in Figure 1. Three different sensors (one temperature sensor ($S_1$), one speed sensor ($S_2$) and one light control sensor ($S_3$)) are presented, which have unevenly spaced measurements due to an event-based recording. Thus, the multivariate sensor space is asynchronous, multicardinal and irregular but is referred to in simplified terms as asynchronous time series in the following.

As already introduced, different variations of time series exist whereby the investigation of this work is based on time series of one-dimensional characteristic. The investigation covers two prediction problems in the context of Multivariate asynchronous Time Series (MaTS). The first prediction problem is based on a clinical dataset, which describes the in-hospital mortality of patients. The second problem under investigation is based on a vehicle component aging dataset.

Within literature, different models have been introduced, which solve the prediction problems with complex model architectures or with the help of synchronizing aggregations (Che et al., 2018; Sass et al., 2019; Horn et al., 2020; Shukla and Marlin, 2021).

Particularly, over the last years the depth of Deep Neural Networks (DNNs) grew significantly to optimize the reliability and the goodness of estimation. In contrast, (Schwartz et al., 2020) elaborated the importance of the consideration of the efficiency of machine learning to minimize the required computational effort. They identified that only a small percentage of publications evaluated by them focused on the efficiency of deep Artificial Neural Networks (ANNs) also called DNNs. Especially, with respect to sustainability and with regard to the rebound effects of digitalization, machine learning models should process time series data efficiently. Hence, the focus of the development of DNNs should be on small and optimized models with reduced data preprocessing effort. One benefit of this optimization can be an efficient and highly optimized continuous operation of DNNs.

For this purpose, we investigate present data structures for DNNs which are used to solve prediction problems based on MaTS. Likewise, we evaluate the models that utilize the data structures, based on their performance and capacity. On the basis of the established solutions, we develop a lightweight but highly efficient data structure to solve asynchronous time series prediction problems. The main contributions of this paper are as follows:

- Investigating data structures and models for prediction problems based on MaTS. Focusing on introduced bias by manipulating the data distribution and on the main memory footprint of the data structures.
- Developing a lightweight but highly efficient data structure to enable Convolutional Neural Networks (CNNs) and Recurrent Neural Networks (RNNs) to solve two asynchronous time series prediction problems which are one binary classification problem and one regression problem.

The remainder of this paper is structured as follows. We describe the related work in Section 2. In Section 3 we describe preliminaries and the methodology. According to the introduced method, we present in Section 4 the experimental setup, the data source and the evaluations. Finally, Section 5 summarizes our findings of this research.

## 2 RELATED WORK

The datasets which are considered within this work correspond to those of MaTS. Based on the findings, which have been gathered through the extensive evaluation and processing of synchronous time series, in recent years, different approaches and methods have been developed to process and represent the rather complex time series asynchronicity with the help of statistical models and DNNs. For problems based on asynchronous time series, discretization can be seen as a basic procedure to use statistical models and kernel-based methods which require a fixed-length input.

Besides discretization methods, asynchronous time series can be processed in two ways (Sun et al., 2020; Weerakody et al., 2021). Firstly, by interpreting missing and irregular measurements as informative, through which different imputation strategies have been developed (Little and Rubin, 2019). Secondly, by processing the measurements as close as possible in their original measuring state.

### 2.1 Synchronous Time Series Modeling

Discretization of time series generates a synchronized time pattern and a uniform distance between consecutive measurements. Hence, Support Vector Machines (SVMs), Gaussian Process Regression (GPR), Random Forest Regression, Hidden Markov Models, Bayesian Networks and Kalman Filters can be applied to solve multivariate time series related problems (Rabiner, 1989; Ramati and Shahar, 2010; Shih et al., 2019).

Besides the introduced statistical models, DNNs have been utilized to solve time series related problems. (Fawaz et al., 2020) evaluated CNNs for their capability to solve multivariate time series classification problems. They identified, that Fully Convolutional Networks (FCNs) and residual networks are well suited for their problem under investigation. Based on these findings and also continuing the work of (Szegedy et al., 2016), who developed the Inception-Network for computer vision tasks, they developed the InceptionTime network for multivariate time series.

Thirdly, DNNs designed as RNNs, such as the Long Short-Term Memory (LSTM)(Hochreiter and Schmidhuber, 1997) and the Gated Recurrent Unit (GRU)(Cho et al., 2014) have been applied to time series related prediction and forecasting problems and have shown impressive results (Shih et al., 2019; Yang et al., 2019; Hewamalage et al., 2021).

Fourthly, attention mechanisms are used in DNNs for positional and time encoding. While utilizing self-attention for positional encoding, Transformers by (Vaswani et al., 2017) have been a break-through in terms of splitting the input into learnable representations. Subsequent work applied positional or time-based attention layers to the time series domain. Different combinations of those attention layers with Multi-Layer Perceptrons (MLPs) and RNNs have shown impressive results for prediction tasks (Song et al., 2018; Ma et al., 2019; Horn et al., 2020; Shukla and Marlin, 2021).

## 2.2 Missingness as Information

Discretization or a join between different modalities can generate a global time axis with aligned measurements. This allows to consider the problem as missing value problem and not as an irregular value problem (Lipton et al., 2016). In this regard, (Lipton et al., 2016) investigated whether missing indicators support LSTM networks for classification problems. They evaluated different combinations of imputation strategies (zero imputation and forward interpolation) with and without missing indicators. The investigation showed, that the combination of zero value imputation and missing indicators result in the best classification performance.

Besides the work of (Lipton et al., 2016), (Neil et al., 2016) improved the LSTM architecture to be capable to process event based unevenly spaced time series. Therefore, they used the time information within a time gate, which they incorporated into the LSTM cell. The time gate is controlled by an oscillation, which is parameterized by three learnable parameters. Only when the time gate is open, the LSTM states are updated. When the time gate is closed, the last cell state maintains although a defined leak of important gradients is allowed. Due to the sparse updates, the memory capability of the LSTM could be improved.

Subsequently, (Che et al., 2018) investigated GRU networks with different imputation strategies, such as mean value and zero value imputation as well as forward interpolation, to evaluate the impact to the classification performance. Therefore, the zero imputation strategy was enriched by missing indicators and delta time information. Moreover, they modified the GRU network and introduced trainable exponential decays which they applied to the input and the hidden state between observations. The so called GRU-D was able to outperform the different baseline GRU imputation strategies at two benchmark classification datasets.

(Lipton et al., 2016) as well as (Che et al., 2018) were able to empirically support the hypothesis of (Ramati and Shahar, 2010) who formulated inefficient training based on artificial generated samples. Also, they were able to indicate inefficient modeling due to data bias based on artificially imputed information, when we consider zero imputation as negligible bias.

## 2.3 Raw Processing & End-to-End Learning

Based on the work of (Rasmussen and Williams, 2006; Li and Marlin, 2016; Futoma et al., 2017), who utilized Gaussian Process (GP) and Multitask Gaussian Process (MGP) to approximate and process irregular time series, (Shukla and Marlin, 2019) developed an end-to-end algorithm without calculation intensive Gaussian Processes (GPs). They used Radial Basis Functions (RBFs) as semi-parametric, feed-forward interpolation layer. With the help of reference time points, each series of the multivariate space is aggregated to a fixed representation. Followed by a concatenation, the multiple fixed-length representations are processed by a cross-dimensional interpolation layer to obtain learnable correlations. In their experiment they combined the interpolation stage with a GRU network. The resulting Interpolation Prediction Network (IPNet) was able to outperform the GP-GRU as well as the GRU-D architecture within a classification benchmark.

(Rubanova et al., 2019) developed a generalization of the continuous state transition for recurrent networks. The basic idea is to understand the hidden state of RNNs as solution of an Ordinary Differential Equation (ODE), which ensures a continu-

ous dynamic representation. As a consequence of the continuous state transition, the modeling of unequally space time series could be optimized. The Latent-ODE (L-ODE-ODE) was able to outperform existing Recurrent Neural Network (RNN) based approaches on a classification benchmark dataset.

Parallel to the continuous development of RNN architectures for MaTS, (Horn et al., 2020) developed an architecture, called Set Functions for Time Series (SeFT), based on Set-Functions (Zaheer et al., 2017), which directly process the unequally spaced measurements. The SeFT model does not need a prior ordering of sequential measurements. The measurements are converted to sets (time, value, modality indicator) and are processed disordered. The information of events and the respective occurrence in time is preserved and learned by the model with the help of a time encoding layer and a Transformer like aggregation network. A weighted mean aggregation approach is utilized to separate important and unimportant time encoded sets. The many weights approach results in a multi-head attention layer. The output of the multi-head attention is subsequently processed by an aggregation network.

(Shukla and Marlin, 2021) have utilized attention to create a deep learning interpolation network, which they called Multi Time Attention Networks (mTANs). The authors combined a time encoding layer (generated via a learnable embedding) with a multi-time attention layer. The multi-time attention layer processes the multiple generated time embeddings to create a fixed representation based on reference time points. The fixed output of the discretized mTANs (mTAND) architecture can then be processed by a black-box classifier. Thus, the mTAND can be interpreted as interpolation module which is not build upon GPs and Radial Basis Function (RBF) kernels, as they have been introduced by (Li and Marlin, 2016; Shukla and Marlin, 2019). In their experiments they combined mTAND with a GRU and trained the combined mTAND model in an end-to-end fashion. They achieved impressive results on two classification benchmark datasets where they were able to outperform the most other recently published models.

# 3 PRELIMINARIES AND METHODOLOGY

This section begins with the depiction of the related work problem. Connected to the related work problem, different data structures are visualized, which are used in relevant model implementations. Based on the present data structures a new data structure is introduced. Lastly, models are described, which are used to process the variable feature space we have devised.

## 3.1 Related Work Problem

According to Section 1, and in particular with regard to sustainable application of AI, the development of efficient DNNs is important. In recent years, different methods have been developed, which can process MaTS (cf. Section 2). The models have been continuously improved to achieve even higher prediction accuracy based on benchmark datasets. The most common quality characteristics were training time per epoch and prediction accuracy estimated with the use of a test dataset. The evaluation of model size (also known as model capacity) has received little to no attention.

Moreover, the research of the last seven years stated, that normal RNNs are insufficient to solve missing value problems. Thus, advanced and modified RNNs have been developed. Also, attention mechanisms were introduced into end-to-end learning concepts which enabled even higher performance on benchmark datasets. However, can we utilize existing general purpose DNNs, such as the RNN, to reach equal performance compared to recently published models?

## 3.2 Data Structures of MaTS

When we are considering vehicle sensor networks, different sensors and also different network protocols interact. Hence, if we gather diverse sensor information in a central node, then this multivariate measurement space is asynchronous, as Figure 1 depicts. Besides the vehicle specific sensor measurements, different DNNs have been introduced, which tried to reach optimal prediction accuracy applied to irregular sampled time series benchmark datasets, such as the PhysioNet dataset (Goldberger et al., 2000; Silva et al., 2012). However, as complex as the models have become, the models consume and process the asynchronous data in similar ways. Generally, the models can be ordered to one of the three data structures shown in Figure 2. The parameter $T$ represents the time information and $S_i$ represents the time series of the measurement interval $t_{[1,8]}$.

The data structure shown in a) indicates the concept of interpolation and imputation of missing values indicated by $\sigma$. Strategies such as forward interpolation or mean value imputation have been used to generate a synchronized multivariate time series of a defined dimensionality (Che et al., 2018). The data structure of b) shows a mask matrix (also known as missing indicator), which indicates whether the val-

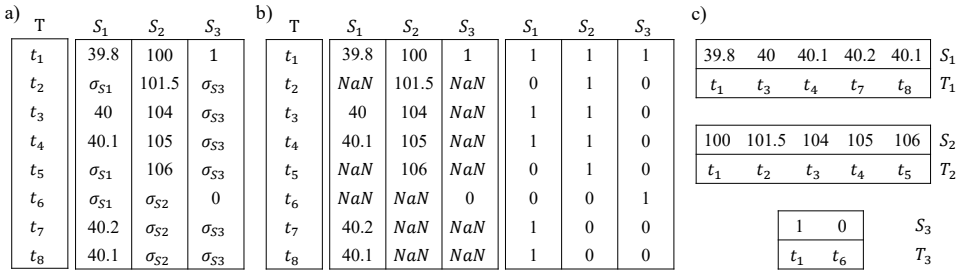


Figure 2: General MaTS data structures for Deep Neural Networks. Structure a) depicts the interpolation and imputation strategy, structure b) shows missing information as valuable information with the use of a masking layer and structure c) depicts the rather non-preprocessed processing of multivariate sensor data.

ues of the NaN-matrix (note that the NaNs can be imputed diversely) are real measurements or negligible values for the deep neural network. (Che et al., 2018) used the mask as missing indicator to enrich the information, which is given to the DNN. In contrast, (Shukla and Marlin, 2021) used the mask to extract valid measurements. The data structure of c) represents an array for each time series. To process the data structure of c), methods can be applied, which process each time series individually. Afterwards, the outputs are aggregated to a latent space of length dependent on the amount of time series. The latent space is then used within a final fixed-input size predictor. As we identified, the data structure of c) minimizes the data preprocessing effort.

## 3.3 Data Streams

The main optimization aspects for processing MaTS are the model as well as the data structure which represents the given information. Hence, we fully utilize the property that RNNs and CNNs can process variable feature spaces. Therefore, the feature space is created with the fundamental idea to change the raw measurements and the existing array, which holds the measurements in the main memory, as less as possible, to reduce the data preprocessing effort. As further requirements we define:

- The data distribution of the raw measurements should not be changed, compared to when no imputation strategy is applied.
- When considering hardware constraints on embedded devices, the input data structure should be as small as possible. Thus, missing value indicators are rather inappropriate.

Thus, we developed a vertically stacked data structure named as *Triplet-Stream*, which is shown in Figure 3. We assume that CNNs as well as RNNs should be capable to generalize over a multivariate feature space, when the features are arranged vertically. The

| T | $t_1$ | $t_3$ | $t_4$ | $t_7$ | $t_8$ | $t_1$ | $t_2$ | $t_3$ | $t_4$ | $t_5$ | $t_1$ | $t_6$ |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| V | 39.8 | 40 | 40.1 | 40.2 | 40.1 | 100 | 101.5 | 104 | 105 | 106 | 1 | 0 |
| ID | $S_1$ | $S_1$ | $S_1$ | $S_1$ | $S_1$ | $S_2$ | $S_2$ | $S_2$ | $S_2$ | $S_2$ | $S_3$ | $S_3$ |

Figure 3: Concatenated data structure defined as Triplet-Stream.

Triplet-Stream ensures separability of the vertically stacked unequally spaced measurements by the third column named as *ID*. Therefore, each time series gets a unique indicator information. The original and raw time information is given within column $T$. The values of the time series are represented by the column $V$.

For a supervised machine learning problem we can then define a dataset $D = \{(Z_k, l_k)|k \in \mathcal{N}^+\}$. Where each sample is described by the label information $l$ (the label is dataset and use-case dependent) and by the feature representation $Z$. The feature representation $Z$ can further be formulated as $Z = \{(t_i, v_i, s_i)|i \in \mathcal{N}^+\}$. The index $i$ defines the recording (row) of the feature space and $t$ the time where the value $v$ was measured for the signal indicator $s$ respectively.

## 3.4 The Ways of Encoding

The usage of a signal indicator and thus the unambiguous assignment of an *ID* to each measurement leads to the problem of encoding the *ID*. The most statistical models and the most established DNNs process numerical data and numerical representations of features.

The sensor *ID* can be interpreted as categorical (string) variable. In the context of tabular data, a categorical variable can be of nominal or ordinal nature (Cerda et al., 2018). Categorical data of ordinal nature are classes which can be transferred to a scalable and finite numerical system. Otherwise, a categorical variable can be nominal, such as the gender of a person is a nominal category. (Cerda et al., 2018) formulated that categorical variables of a tabular data are

standardized when the dataset is defined completely. In this context, standardization can be understood as a state, where the categories of the categorical variable are known a-priori, finite and mutually exclusive. In a non-standardized tabular dataset the categories of a categorical variable are not known a-priori.

Within the statistical analysis the problem arises, when models have to process unknown nominal categories of a new collected dataset. The identification, whether a new category belongs to an already existing entity or whether the new category describes a completely new entity is challenging and problematic (Cerda et al., 2018; Cerda and Varoquaux, 2022).

To solve this problem, different algorithms have been developed to convert the categorical classes to a numerical feature representation to ensure separability between categories. In principle, those algorithms can be divided into two approaches. Firstly, *Constant Encoding*, where the categorical variable is converted to a constant numerical representation. Secondly, *Similarity Encoding*, where each category gets a trainable representation.

As constant encoding, such as One-Hot-Encoding (OHE), does not depict the similarity between words such as *book* and *publication*, algorithms have been developed to encode numerical similarity based on distances. For example, Word-Embeddings of fixed size, implemented as initial layer for categorical variables, can be trained in a supervised fashion in DNNs to obtain the best suited encoded representation (Guo and Berkhahn, 2016).

### 3.5 The Method

In our method, the Triplet-Stream is processed by DNNs, which are capable to process variable feature spaces. For this, the modality indicators need to be encoded. The categories of the analyzed datasets are finite, mutually exclusive and known a-priori. Due to the disadvantages of *Constant Encoding* algorithms, we choose the Word-Embedding approach introduced by (Guo and Berkhahn, 2016), cf. Section 3.4. Thereby, the embedding size can be considered as further hyperparameter.

The method we propose is shown in Figure 4. It can be seen that the modality indicators are processed by the trainable Embedding-Network in advance. Subsequently, the encoded categories are concatenated with the original time and value information. The combined representation is then processed by the prediction network. To identify, which DNN architecture is best suited as *Prediction Head*, cf. Figure 4, we evaluate the GRU network designed by (Cho et al., 2014) and the InceptionTime network designed

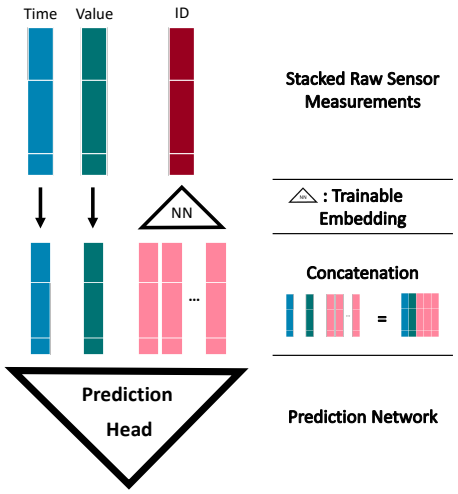


Figure 4: The generic Deep Neural Network (DNN)-Architecture for MaTS.

by (Fawaz et al., 2020). The GRU-based approach is referred to as GRU-Triplet-Stream (GRUTS) and the InceptionTime-based approach is referred to as InceptionTime-Triplet-Stream (ITTS).

GRU based models have shown promising results applied on MaTS datasets (Che et al., 2018). In contrast, the performance of the InceptionTime network applied to MaTS datasets has barely been studied. The InceptionTime network is designed modular with the help of blocks. To overcome the vanishing gradient problem residual connections are used. Each block consists of a dimension-reducing bottleneck layer and convolution layers with different kernel-sizes. The different kernel-sizes generate a broader and improved field-of-vision. Subsequently, the outputs of the convolution layers are concatenated with the output of a MaxPool layer, which is applied to the input of the block. Following the last inception block a global average pooling is applied to generate a fixed dimensional space which is forwarded to a linear layer. The amount of blocks, the size of kernels but also the amount of bottleneck channels are important hyperparameters of the model.

## 4 EXPERIMENT

In the experiments we evaluated the GRUTS and the ITTS on two irregular sampled and asynchronous multivariate time series datasets: The first one is a binary classification task based on the PhysioNet-2012 dataset and the second one is a regression task based on the Vehicle Engine Component Aging (VECA) dataset.

## 4.1 Datasets

The **PhysioNet-2012 Challenge** dataset contains 12000 Intensive Care Unit (ICU) admissions (Goldberger et al., 2000; Silva et al., 2012). At the beginning, only 4000 ICU admissions possessed labels (named as Set-a). The dataset was extended over the past few years, which results in currently $\sim$ 12000 labeled ICU admissions. Each patient was admitted to one of a variety of ICUs for e.g. medical, trauma and surgical reasons. The ICU stay of the patient needed to last at least 48 hours, otherwise the patients have been excluded from the dataset. The dataset contains 41 variables of which five generic descriptors are only measured once such as the age and the weight of the patient. The remainder of 36 variables represents time series with multiple observations. Each observation got a time stamp which describes the time, which has passed since the initial admission of the individual patient. Five outcome descriptors are available in the dataset such as length of stay or length of survival. Within this study we focused on the in-hospital mortality (1:=died; 0:=survived). This led to a dataset, which is strongly imbalanced. Less than 15% of the labeled ICU stays are positive.

To be comparable to the work of (Rubanova et al., 2019; Shukla and Marlin, 2021), we chose the Set-a with 4000 labeled patients as the first benchmark dataset. We refer to this as PhysioNet-a dataset. To further be comparable to the work of (Horn et al., 2020), we created a second benchmark dataset from the whole 12000 labeled patients and applied the same train-test-split as (Horn et al., 2020). We refer to this as PhysioNet-b dataset.

The **Vehicle Engine Component Aging** dataset has been introduced by (Sass et al., 2019). The component under investigation was the Exhaust Gas Recirculation (EGR) cooling system. The EGR system is used in diesel engines to regulate the combustion temperature within the combustion chamber. The aging represents the degree of pollution of the EGR-cooler. The investigated degree of aging of the prototype in the development stage is detected in service intervals by measuring the fresh air mass flow once with enabled EGR-system $\dot{m}_{open}$ and once with disabled EGR-system $\dot{m}_{closed}$. The aging over time is then defined as mass flow ratio $\dot{m}_r = \dot{m}_{open}/\dot{m}_{closed}$. While utilizing interpolation, the aging in the range of $(0,1)$ is continuously defined over the whole investigated time span.

The corresponding vehicle data are one dimensional in-vehicle sensor information recorded within the time of component aging. With respect to the investigation of (Sass et al., 2020), who identified and classified the importance of sensors to monitor the aging, we selected the seven most relevant sensors. Based on the properties of the underlying bus-network, the multivariate time series space is asynchronous as asynchronicity has been defined in Section 1. Furthermore, we increased the effect of asynchronicity and reduced the size of the MaTS space to 49.16 % of the original allocated size with the use of the Binary Shift Compression (BiSCo) algorithm (lossless configuration) introduced by (Vox et al., 2022).

The total component aging dataset contains seven unevenly spaced time series as features and one label time series, which describes the aging. In total, the encoded dataset is defined by 29.883.593 observations over the multivariate time series space. However, the technical requirement of a minute-by-minute prediction of component aging leads to 38.426 labeled samples which we created with a sliding window approach. Furthermore, we took the last aging information within the window to label the sample.

## 4.2 Competitors

As competitors we have chosen models which have shown suitable properties to process asynchronous vehicle sensor data. These models are listed below:

- The Phased-LSTM model (Neil et al., 2016)
- The GRU-D model (Che et al., 2018)
- The L-ODE-ODE model (Rubanova et al., 2019)
- The IPNet model (Shukla and Marlin, 2019)
- The SeFT model (Horn et al., 2020)
- The mTAND model (Shukla and Marlin, 2021)

All models are described in detail in Section 2.

## 4.3 Experimental Setup

All experiments have been written in the programming language Python, where we used the PyTorch and the Tensorflow framework (Abadi et al., 2016; Paszke et al., 2019). All experiments are executed with a GPU of type NVIDIA Tesla K80, which has access to 6 cores and 56 GB of RAM. Furthermore, we used the Adam optimizer during learning phase for all Triplet-Stream experiments (Kingma and Ba, 2015). To find the best hyperparameter configuration for the problems under investigation we used a Bayesian optimization approach called Heteroscedastic Evolutionary Bayesian Optimisation (HEBO) (Cowen-Rivers et al., 2022). HEBO has shown outstanding performance on a benchmark

dataset compared to other hyperparameter optimization algorithms (Turner et al., 2021). For reproducibility, the best hyperparameters of the individual models for the two benchmark datasets are presented in the APPENDIX.

## 4.4 Classification Results

Due to the imbalanced PhysioNet datasets (PhysioNet-a and PhysioNet-b) we used the Area Under the Receiver Operating Characteristic Curve (AUROC) score instead of the binary accuracy to compare the classification performance. The ITTS and the GRUTS model have been trained with the cross-entropy loss. Thereby, we applied a mini-batch stochastic training approach. Also, we evaluated the classification performance of the test dataset for each epoch. Through an extensive hyperparameter search we identified the best model configurations and simulated them ten times, for which the performance (mean and standard deviation) is shown in Table 1. The best model configurations are listed in Table 4 and 5 in the APPENDIX. It can be seen from the

Table 1: Visualization of the classification performance for medical health prediction problems.

| **Model** | **AUROC** | |
|---|---|---|
| | PhysioNet-a | PhysioNet-b |
| GRU-D | $0.818 \pm 0.008^{*}$ | $0.863 \pm 0.003^{\dagger}$ |
| Phased-LSTM | $0.836 \pm 0.003^{*}$ | $0.790 \pm 0.010^{\dagger}$ |
| IPNet | $0.819 \pm 0.006^{*}$ | $0.860 \pm 0.002^{\dagger}$ |
| SeFT | $0.795 \pm 0.015^{*}$ | $0.851 \pm 0.004^{\dagger}$ |
| L-ODE-ODE | $0.829 \pm 0.004^{*}$ | $0.857 \pm 0.006^{\dagger}$ |
| mTAND | $0.854 \pm 0.001^{*}$ | - |
| ITTS | $0.801 \pm 0.014$ | $0.828 \pm 0.003$ |
| GRUTS | $0.834 \pm 0.004$ | $0.857 \pm 0.002$ |

*Published by (Shukla and Marlin, 2021)
†Published by (Horn et al., 2020)

table that the mTAND model reached the highest classification performance for the PhysioNet-a dataset. It is worth noting that we were not able to reproduce the published accuracy of the mTAND model, which deviated in our experiments by approximately two percentage points to 0.83. For the PhysioNet-b dataset the GRU-D model reached the highest AUROC score.

From the results it can be taken that the GRUTS approach achieved a classification performance comparable to advanced attention models and comparable to modified recurrent networks such as the GRU-D. Also, for the more transparent results of the PhysioNet-b benchmark the GRUTS model deviated only by 0.6 percentage points, which is interesting due to the fact that (Che et al., 2018) compared different GRU based methods on MaTS prediction problems and showed that their GRU-D model outperformed standard GRUs significantly. Moreover, from the results it can be taken that the ITTS approach did not perform well on the PhysioNet classification benchmarks.

## 4.5 Regression Results

To be able to fairly compare the performance for the VECA regression task of the individual models a train-test-split of 80%-20% has been applied to the VECA dataset. We excluded the L-ODE-ODE model because of the escalating training time per epoch caused by the ODE-solver, which made a fair comparison by an appropriate hyperparameter tuning impossible. Within the mini-batch based learning algorithm we used the Mean-Squared-Error (MSE) as loss term. All investigated models have been extensively hyperparameter tuned with HEBO, where we used the model implementations of (Horn et al., 2020) and (Shukla and Marlin, 2021). The resulting optimal hyperparameters are depicted in Table 6 in the APPENDIX. Furthermore, the regression performance of the best hyperparameter configuration for each model under investigation is shown in Table 2. The simulation for each model has been repeated three times to present the RMSE with mean and standard deviation. From the results of the regression benchmark it can be seen that the GRUTS model reached the best performance while the ITTS model enabled a more stable convergence based on the lower standard deviation with a comparable RMSE as the GRUTS. Without forcing the hyperparameter tuning algorithm to search for small models, the model with the lowest capacity and thus with the lowest number of trainable parameters is the IPNet model. If the epoch time quality characteristic is considered, then the SeFT model is the fastest model which we evaluated. While taking into account the RMSE, the epoch time and the capacity holistically the most suited model to solve the regression problem is the ITTS approach due to the Pareto optimum of the three quality measures.

## 4.6 Ablation Study

Besides the model size the sample data structure and the resulting array size in the main memory is a further important criterion for embedded devices such as vehicles and batch based training on graphic cards. Hence, we investigated the amount of bytes which have to be allocated for the data structures of Section 3.

Table 2: Visualization of the regression performance defined as Root-Mean-Squared-Error (RMSE) (mean ± std), the mean epoch time in seconds and the model capacity defined as trainable parameters. The RMSE evaluation metric was multiplied by 1000 to improve readability.

| Model | RMSE | Capacity | Epoch Time | Model Impl. |
|---|---|---|---|---|
| Phased-LSTM | $1.736 \pm 0.337$ | 482,305 | 6395.84 | (Horn et al., 2020)* |
| GRU-D | $3.214 \pm 0.598$ | 49,695 | 6740.68 | (Horn et al., 2020)* |
| IPNet | $3.304 \pm 0.510$ | 15,057 | 2095.66 | (Horn et al., 2020)* |
| SeFT | $2.933 \pm 0.817$ | 122,033 | 56.88 | (Horn et al., 2020)* |
| mTAND | $3.367 \pm 1.228$ | 105,733 | 1886.58 | (Shukla and Marlin, 2021)† |
| ITTS | $1.301 \pm 0.082$ | 82,094 | 138.11 | cf. Section 3.5† |
| GRUTS | $1.250 \pm 0.450$ | 108,064 | 437.80 | cf. Section 3.5† |

*Tensorflow v2.9.1; †PyTorch v1.12

In Table 3 the data structures are linked to the model implementations. Moreover, the table shows the mean sample size in Bytes of 30 randomly selected samples of the VECA dataset. It can be seen that the interpolation based data structure ($a_1$) results in a sample size which is by a factor of 1000 bigger than the other data structures. To ensure that each sensor information is represented without loss of information, we used an interpolation frequency of 10 Hz. Likewise, the standard deviation of the interpolated data structure is noteworthy because a few of the samples allocated more than 1 GB. Thus, the interpolated data structure is rather inappropriate. The mean value imputed data structure is approximately 50% smaller than the masked data structure, which is caused by the additional mask. Also, it can be taken from the table that the Triplet-Stream data structure is the second smallest. In comparison to the masked data structure, the Triplet-Stream reduced the needed memory space by approximately 80%.

### 4.7 Discussion

In the experiments we showed that the Triplet-Stream combined with CNN and RNN prediction heads can compete and even outperform state-of-the-art attention and recurrent models in the domain of MaTS. We showed that an efficient data structure and a suitable encoding of modality indicators enabled high performance of general purpose DNNs in the classification task and in the regression task.

For the classification task we used the published results of (Shukla and Marlin, 2021), because we were not able to reproduce them. Nevertheless, the GRUTS approach obtained sufficient performance within the classification benchmark. In the regression benchmark we showed that the Triplet-Stream is an efficient method for vehicle sensor data based regression problems.

## 5 CONCLUSION

In this paper we elaborated a need for research for processing MaTS in DNNs. Therefore, we developed the Triplet-Stream, which is a lightweight datastructure without missingness indicators or value imputation strategies. Thus, we did not change the raw sensor data distribution, to minimize data based bias. We combined the Triplet-Stream with an embedding stage and a subsequent prediction head. As prediction heads we evaluated the GRU model and the InceptionTime model. The GRUTS and the ITTS as well as other relevant models were benchmarked on a regression problem and on a classification problem.

The evaluation has shown that the GRUTS model achieved excellent performance on the classification problem. Furthermore, the Triplet-Stream based models outperformed existing state-of-the-art models on the regression dataset. The ITTS model achieved Pareto-optimal results regarding the quality measures accuracy, training time per epoch, model capacity and sample size. Consequently, we demonstrated an appropriateness of our method to enable a sustainable application of deep learning. Based on these results, the Triplet-Stream approach should further be studied and analyzed on different multivariate time series based prediction problems.

In future work, we will continue the investigation of suitable data structures for DNNs. We will evaluate their impact on the model size for vehicle sensor data based prediction problems. Also, we will study the impact of processing synchronous time series with the Triplet-Stream and compare the results to the standard approach. Likewise, we will extend the investigation to vehicle fleets and analyze the energy demand of DNNs. Through this, we want to evaluate the energy consumption of models when they are fully deployed.

Table 3: Combining the data structures of Figure 2 and Figure 3 with the models under investigation and the resulting VECA sample size.

| Data Structure | Sample Size in Byte | Model |
|---|---|---|
| Interpolated ($a_1$) | $4.017e8 \pm 1.060e9$ | - |
| Imputed ($a_2$) | $4.591e4 \pm 3.647e4$ | - |
| Masked ($b$) | $8.608e4 \pm 7.837e4$ | SeFT, IPNet, mTAND GRU-D, Phased-LSTM |
| Raw ($c$) | $1.148e4 \pm 9.116e3$ | - |
| Triplet-Stream | $1.722e4 \pm 1.367e4$ | GRUTS, ITTS |

# DISCLAIMER

The results, opinions, and conclusions expressed in this publication are not necessarily those of Volkswagen Aktiengesellschaft.

# REFERENCES


Abadi, M., Barham, P., Chen, J., Chen, Z., Davis, A., Dean, J., Devin, M., Ghemawat, S., Irving, G., Isard, M., et al. (2016). Tensorflow: A system for large-scale machine learning. In *12th USENIX Symposium on Operating Systems Design and Implementation (OSDI 16)*, pages 265–283.

Cerda, P. and Varoquaux, G. (2022). Encoding high-cardinality string categorical variables. *IEEE Transactions on Knowledge and Data Engineering*, 34(3):1164–1176.

Cerda, P., Varoquaux, G., and Kégl, B. (2018). Similarity encoding for learning with dirty categorical variables. *Machine Learning*, 107(8-10):1477–1494.

Che, Z., Purushotham, S., Cho, K., Sontag, D., and Liu, Y. (2018). Recurrent neural networks for multivariate time series with missing values. *Scientific reports*, 8(1):6085.

Cho, K., van Merrienboer, B., Gulcehre, C., Bahdanau, D., Bougares, F., Schwenk, H., and Bengio, Y. (2014). Learning phrase representations using rnn encoder–decoder for statistical machine translation. *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing, EMNLP*, pages 1724–1734.

Cowen-Rivers, A. I., Lyu, W., Tutunov, R., Wang, Z., Grosnit, A., Griffiths, R. R., Maraval, A. M., Jianye, H., Wang, J., and Peters, J. (2022). Hebo: pushing the limits of sample-efficient hyper-parameter optimisation. *Journal of Artificial Intelligence Research*, 74:1269–1349.

Fawaz, H. I., Lucas, B., Forestier, G., Pelletier, C., Schmidt, D. F., Weber, J., Webb, G. I., Idoumghar, L., Muller, P.-A., and Petitjean, F. (2020). Inceptiontime: Finding alexnet for time series classification. *Data Mining and Knowledge Discovery*, 34(6):1936–1962.

Futoma, J., Hariharan, S., and Heller, K. (2017). Learning to detect sepsis with a multitask gaussian process rnn classifier. *International Conference on Machine Learning, ICML*, 34:1174–1182.

Goldberger, A. L., Amaral, L. A. N., Glass, L., Hausdorff, J. M., Ivanov, P. C., Mark, R. G., Mietus, J. E., Moody, G. B., Peng, C.-K., and Stanley, H. E. (2000). Physiobank, physiotoolkit, and physionet: Components of a new research resource for complex physiologic signals. *Circulation*, 101(23):215–220.

Guo, C. and Berkhahn, F. (2016). Entity embeddings of categorical variables. *arXiv preprint arXiv:1604.06737*.

Hewamalage, H., Bergmeir, C., and Bandara, K. (2021). Recurrent neural networks for time series forecasting: Current status and future directions. *International Journal of Forecasting*, 37(1):388–427.

Hochreiter, S. and Schmidhuber, J. (1997). Long short-term memory. *Neural computation*, 9(8):1735–1780.

Horn, M., Moor, M., Bock, C., Rieck, B., and Borgwardt, K. (2020). Set functions for time series. *International Conference on Machine Learning, ICML*, 37:4353–4363.

Kingma, D. P. and Ba, J. (2015). Adam: A method for stochastic optimization. *International Conference on Learning Representations, ICLR*, 3.

Li, S. C.-X. and Marlin, B. (2016). A scalable end-to-end gaussian process adapter for irregularly sampled time series classification. *International Conference on Neural Information Processing Systems*, 30:1812–1820.

Lipton, Z., Kale, D., and Wetzel, R. (2016). Modeling missing data in clinical time series with rnn. *Proceedings of Machine Learning for Healthcare 2016*.

Little, R. J. A. and Rubin, D. B. (2019). *Statistical analysis with missing data*, volume 793. John Wiley & Sons.

Ma, J., Shou, Z., Zareian, A., Mansour, H., Vetro, A., and Chang, S.-F. (2019). Cdsa: Cross-dimensional self-attention for multivariate, geo-tagged time series imputation. *arXiv preprint arXiv:1905.09904*.

Neil, D., Pfeiffer, M., and Liu, S.-C. (2016). Phased lstm: Accelerating recurrent network training for long or event-based sequences. *International Conference on Neural Information Processing Systems*, 30:3889–3897.

Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., Desmaison, A., Kopf, A., Yang, E., DeVito, Z., Raison, M., Tejani, A., Chilamkurthy, S., Steiner, B., Fang, L., Bai, J., and Chintala, S. (2019). Pytorch: An imperative style, high-performance deep learning library. *Advances in Neural Information Processing Systems*, 32:8024–8035.

Rabiner, L. R. (1989). A tutorial on hidden markov models and selected applications in speech recognition. *Proceedings of the IEEE*, 77(2):257–286.

Ramati, M. and Shahar, Y. (2010). Irregular-time bayesian networks. *Conference on Uncertainty in Artificial Intelligence, UAI*, 26.

Rasmussen, C. E. and Williams, C. K. I. (2006). *Gaussian Processes for Machine Learning*. MIT Press.

Rubanova, Y., Chen, R. T. Q., and Duvenaud, D. (2019). Latent odes for irregularly-sampled time series. *Advances in Neural Information Processing Systems*, 32.

Sass, Andreas Udo, Esatbeyoglu, E., and Fischer, T. (2019). Monitoring of powertrain component aging using in-vehicle signals. *Diagnose in mechatronischen Fahrzeugsystemen XIII*, pages 15–28.

Sass, Andreas Udo, Esatbeyoglu, E., and Iwwerks, T. (2020). Signal pre-selection for monitoring and prediction of vehicle powertrain component aging. *Proceedings of the 16th European Automotive Congress*, pages 519–524.

Schwartz, R., Dodge, J., Smith, N. A., and Etzioni, O. (2020). Green ai. *Communications of the ACM*, 63(12):54–63.

Shih, S.-Y., Sun, F.-K., and Lee, H.-y. (2019). Temporal pattern attention for multivariate time series forecasting. *Machine Learning*, 108(8-9):1421–1441.

Shukla, S. and Marlin, B. (2019). Interpolation-prediction networks for irregularly sampled time series. *International Conference on Learning Representations, ICLR*, 7.

Shukla, S. N. and Marlin, B. M. (2021). Multi-time attention networks for irregularly sampled time series. *International Conference on Learning Representations, ICLR*, 9.

Silva, I., Moody, G., Scott, D. J., Celi, L. A., and Mark, R. G. (2012). Predicting in-hospital mortality of icu patients: The physionet/computing in cardiology challenge 2012. In *2012 Computing in Cardiology*, pages 245–248.

Song, H., Rajan, D., Thiagarajan, J. J., and Spanias, A. (2018). Attend and diagnose: Clinical time series analysis using attention models. *Proceedings of the AAAI conference on artificial intelligence*, 32(1).

Sun, C., Hong, S., Song, M., and Li, H. (2020). A review of deep learning methods for irregularly sampled medical time series data. *arXiv preprint arXiv:2010.12493*.

Szegedy, C., Vanhoucke, V., Ioffe, S., Shlens, J., and Wojna, Z. (2016). Rethinking the inception architecture for computer vision. *Proceedings of the IEEE conference on computer vision and pattern recognition*.

Turner, R., Eriksson, D., McCourt, M., Kiili, J., Laaksonen, E., Xu, Z., and Guyon, I. (2021). Bayesian optimization is superior to random search for machine learning hyperparameter tuning: Analysis of the black-box optimization challenge 2020. *NeurIPS 2020 Competition and Demonstration Track*, pages 3–26.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L., and Polosukhin, I. (2017). Attention is all you need. *Advances in Neural Information Processing Systems*, 30.

Vox, C., Broneske, D., Piewek, J., Sass, A. U., and Saake, G. (2022). Integer time series compression for holistic data analytics in the context of vehicle sensor data. In *International Conference on Connected Vehicle and Expo (ICCVE)*, pages 1–7.

Weerakody, P. B., Wong, K. W., Wang, G., and Ela, W. (2021). A review of irregular time series data handling with gated recurrent neural networks. *Neurocomputing*, 441(7):161–178.

Wu, S., Liu, S., Sohn, S., Moon, S., Wi, C.-I., Juhn, Y., and Liu, H. (2018). Modeling asynchronous event sequences with rnns. *Journal of biomedical informatics*, 83:167–177.

Yang, B., Sun, S., Li, J., Lin, X., and Tian, Y. (2019). Traffic flow prediction using lstm with feature enhancement. *Neurocomputing*, 332(4):320–327.

Zaheer, M., Kottur, S., Ravanbakhsh, S., Poczos, B., Salakhutdinov, R. R., and Smola, A. J. (2017). Deep sets. *Advances in Neural Information Processing Systems*, 30.

# APPENDIX

Table 4: Classification Benchmark I.

| **Best hyperparameters for the PhysioNet-a dataset** |
|---|
| **ITTS:** n-blocks=3, bottleneck-channels=2, n-filters=12, kernel-sizes=[4, 8, 16], activation=hardswish, out-activation=sigmoid, n-embeddings=15, linear-hidden=[256], use-residual=True, min-max-scaling=False; **GRUTS:** rnn-hidden=55, rnn-depth=5, activation=gelu, linear-hidden=[65, 33], n-embeddings=69, min-max-scaling=False |

Table 5: Classification Benchmark II.

| **Best hyperparameters for the PhysioNet-b dataset** |
|---|
| **ITTS:** n-blocks=1, bottleneck-channels=12, n-filters=16, kernel-sizes=[4, 8, 16], activation=tanh, out-activation=linear, n-embeddings=20, linear-hidden=[128], use-residual=True, min-max-scaling=False; **GRUTS:** rnn-hidden=235, rnn-depth=7, activation=tanh, linear-hidden=[424], n-embeddings=176, min-max-scaling=False |

Table 6: Regression Benchmark.

| **Best hyperparameters for the VECA dataset** |
|---|
| **Phased-LSTM:** n-units=256, use-peepholes=True, leak=0.001, period-init-max=1000.0; **GRU-D:** n-units=120, dropout=0.0, recurrent-dropout=0.01; **IPNet:** n-units=60, dropout=0.0, recurrent-dropout=0.1, imputation-stepsize=1.0, reconst-fraction=0.01; **SeFT:** n-phi-layers=1, phi-width=165, phi-dropout=0.0, n-psi-layers=3, psi-width=28, psi-latent-width=121, dot-prod-dim=90, n-heads=7, attn-dropout=0.1, latent-width=40, n-rho-layers=4, rho-width=24, rho-dropout=0.0, n-positional-dims=4, max-timescale=100.0; **mTAND**: query-steps=196, rec-hidden=94, embed-time=88, num-heads=4, freq=9, learn-emb=True, regressor-layer-size=134; **ITTS:** n-blocks=6, bottleneck-channels=2, n-filters=13, kernel-sizes=[4, 32, 128], activation=relu, out-activation=sigmoid, n-embeddings=1, linear-hidden=[256], use-residual=True, min-max-scaling=True; **GRUTS:** rnn-hidden=64, rnn-depth=5, activation=hardswish, linear-hidden=[128, 64, 32, 16], n-embeddings=1, min-max-scaling=True |