K-Means with Fréchet Mean under Dynamic Time Warping for Data Imputation in Univariate Time Series

SAMAN AKBARI, Technische Universität Berlin, Germany
EFTHIMIOS-ENIAS GOJKA, Technische Universität Berlin, Germany
ANTON BALTHASAR JAEKEL, Technische Universität Berlin, Germany
ELENA KRANZ, Technische Universität Berlin, Germany
MATS SALEWSKI, Technische Universität Berlin, Germany

Missing or unknown data in time series has been a common drawback in the machine learning domain. Consecutively, missing values would result in serious information loss if simply dropped from the dataset or ignored. To solve this pre-processing problem, there are both machine learning approaches, and methods imported from statistical learning theory to impute the missing values. The aim of this project is to analyze the missing data problem within univariate time series by comparing selected well-known methods with k-means imputation under dynamic time warping using the Fréchet mean.

Keywords: time series, imputation, dynamic time warping, k-means, fréchet mean, missing values

1 INTRODUCTION

Nowadays, data is generated almost everywhere, ranging from finance, meteorology to healthcare. Often, this data consists of multiple, and more or less similar time series. Examples are the stock prices, the wind speed over a specific period, or simply the daily sales from our local corner shop. Many of these real world applications suffer from missing or unknown data. The reasons range from mechanical or electronic failures during the data acquisition process, up to data corruption. Inappropriate treatment of missing data can cause large errors or false results in machine learning tasks like classification [6].

Methods to impute the missing values in time series differ based on the dimensionality of the data. The approaches can be divided into imputation methods for (i) *multivariate time series*, as well as (ii) *univariate time series*. Multivariate time series are dependent on more than one variable, while a univariate time series is a sequence of single observations $y_1, y_2, ..., y_n$ at successive points $t_1, t_2, ..., t_n$ in time [8]. The focus of this paper is set on univariate time series.

A standard method to impute the missing values in multiple univariate time series is by just taking the arithmetic mean of the whole dataset. To that end, a time series located in the center of a given sample of time series is constructed in the Euclidean space. This approach leads to temporal inaccuracies, and therefore significant information loss, as shown in Figure 1. Temporal variations of the time series are not considered. Thus, important features such as characteristic subsequences may disappear.

To solve this problem, we are going to use the Fréchet mean under dynamic time warping (DTW). Thus, the alignment of time series is performed with respect to the DTW distance and then synthesized to an average. We suppose that

Authors' addresses: Saman Akbari, Technische Universität Berlin, Germany, saman.akbari@campus.tu-berlin.de; Efthimios-Enias Gojka, Technische Universität Berlin, Germany, gojka@campus.tu-berlin.de; Anton Balthasar Jaekel, Technische Universität Berlin, Germany, anton.b.jaekel@campus.tu-berlin.de; Elena Kranz, Technische Universität Berlin, Germany, kranz@campus.tu-berlin.de; Mats Salewski, Technische Universität Berlin, Germany, m.salewski@campus.tu-berlin.de.

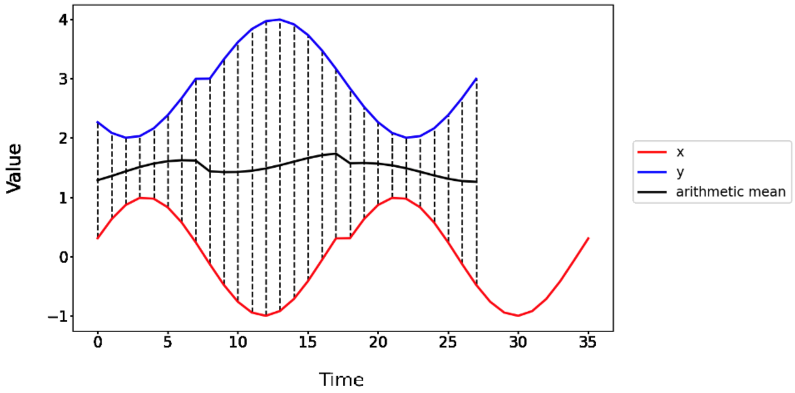

Fig. 1. Arithmetic mean example for a univariate time series.

$X = \left(x^{(1)}, ..., x^{(n)}\right)$ is a sample of N time series $x^{(i)}$. So a (sample) mean in DTW spaces is a time series that minimizes the Fréchet function [5]

$$F(x) = \frac{1}{N} \sum_{k=1}^{N} dtw^2 \left(x, x^{(k)}\right), \tag{1}$$

where dtw is the DTW distance. The Fréchet function is non-differentiable and non-convex. Currently, there is no polynomial-time algorithm for finding a global minimum [2]. This problem is usually approached by iteratively searching for a sample mean that is sufficient close to the optimum to be used for further computations. This paper will make use of the Stochastic Subgradient Mean algorithm (SSG) as proposed in [12].

Different to the Euclidean distance, DTW does not align pointwise. Instead, it tries to find more natural alignments considering multiple, alternative points. As visualized in Figure 2, DTW is able to synthesize the aligned time series to a mean considering temporal variations.

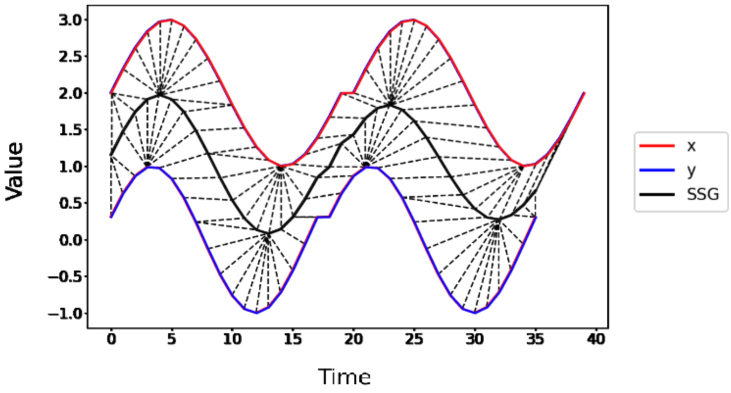

Fig. 2. Stochastic subgradient mean under dynamic time warping.

One problem of unsupervised machine learning tasks is that it is not known if the data can be categorized into multiple subsets with similar features. If so, only one mean for the whole dataset would not consider all the characteristics within each subset. An example is the traffic volume in a city over a day. In this case, clustering the data into the time of the days, i.e, morning, noon, evening and night, would most likely improve the performance since the traffic volume presumably is dependent on it. If not, the characteristics within each cluster would be ignored by synthesizing only one mean for the whole dataset. Thus, the imputation would suffer from significant information loss.

In this paper, we investigate the effect of synthesizing k different cluster centroids on the imputation performance. The idea is to cluster the dataset into k different subsets with the k-means algorithm under DTW *(DTW k-means)*. K-Means is an iterative clustering algorithm used to classify unsupervised data into a specified number k of groups. By clustering the data under DTW, every time series is assigned to the cluster centroid when the DTW distance is minimal. Thus, important features within one cluster are not ignored anymore. After clustering, the algorithm finds a Fréchet-mean under DTW for each cluster. These are used for imputing the missing values of the time series in each respective cluster.

For evaluation, this paper reviews three additional missing data imputation techniques for comparison. The literature distinguishes between statistical imputation methods and imputation based on machine learning methods [6]. We choose two naive statistical imputation methods, arithmetic mean and linear interpolation. Moreover, we decided to take k-nearest-neighbor under DTW, as it is based on DTW like our approach.

The remainder of the paper is structured as follows. Section 2 discusses related work and provides background material for the analyzed imputation methods. In section 3, we go into more detail about the methodology of our algorithm DTW k-means. We follow in section 4 by explaining the experiment and illustrating the results and discuss our two experiment in section 5. Finally, in section 6 the main conclusion is drawn.

2 BACKGROUND AND RELATED WORK

The analysis and imputation of missing values depends on the missing mechanism. In this section, we explain the different missing data mechanisms, and explain further the methods we choose for the imputation and classification task. Additional to handling missing values, applying DTW k-means concerns the problem of pattern classification. The approaches in the literature for tackling those two problems can be grouped into statistical imputation methods, and imputation methods based on machine learning. Arithmetic mean and linear interpolation belong to the statistical imputation methods, while DTW kNN and DTW k-means are more sophisticated machine learning methods.

2.1 Missing Data Types

We start by introducing different types of missing data. The appropriate imputation method depends in most cases on how data became lost. The literature distinguishes between three types of missing data mechanisms, missing completely at random (*MCAR*), missing at random (*MAR*) and not missing at random (*NMAR*) [6]. The majority of research assumes the missing process is MCAR or MAR. Thus, researcher can ignore the reasons for missing data, which simplifies the methods for missing data analysis.

MCAR occurs when the probability that a variable is missing is independent of the variable itself. A typical example of MCAR is when a blood sample of a study object is lost or broken by accident. There is no reason for missingness neither a relation to any other patient blood characteristics. The process how values get lost can be described as probability for the missing indicator M with the complete set X^{obs} and the missing set X^{mis} given. For MCAR, the probability depend on an independent set of parameters ξ [9]:

$$Pr\left(M \middle| X^{obs}, X^{mis}, \xi\right) = Pr\left(M \middle| \xi\right). \tag{2}$$

For MAR mechanism, the data is missing due to some external reason. For example, when a sensor fails occasionally because of power outage. However, the missingness is unrelated to the variable. In the third type NMAR the missing variable cannot be predicted only from the available variables in the database. This happens for example when a sensor cannot measure outside a certain range. Due to these circumstances, valuable information gets lost [6]. We assume the MCAR mechanism in this paper.

2.2 Statistical Methods

The statistical methods *linear interpolation* and *arithmetic mean* are often used for comparison to show the improvement due to more sophisticated machine learning methods [4, 8, 9, 13]. In this chapter, we are going to highlight some advantages and disadvantages.

2.2.1 *Linear Interpolation.*

Linear interpolation takes two data points given by the coordinates (x_a, y_a) and (x_b, y_b), and draws a straight line between start and endpoint. For imputation, the interpolant is defined as:

$$y = y_a + (y_b - y_a)\,\frac{x - x_a}{x_b - x_a} \tag{3}$$

at the point (x, y).

In Figure 3 an example imputation with linear interpolation is shown. The green dashed line is the original time series, and the red points show the removed values. The graph colored in magenta presents the result of the respective imputation method.

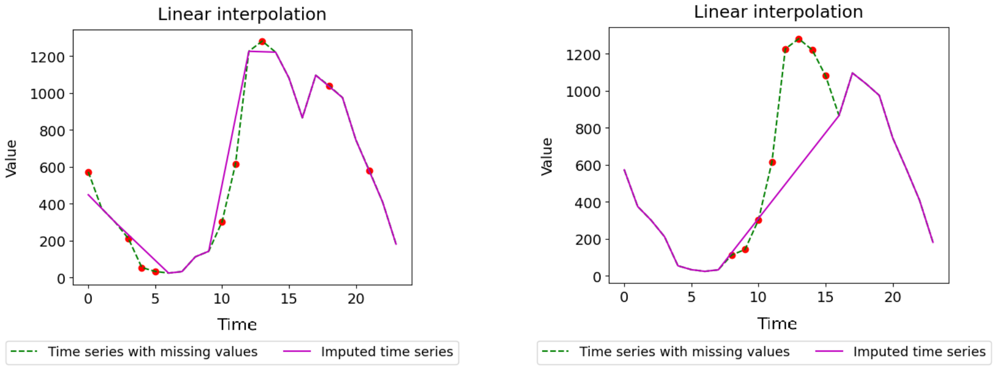

Fig. 3. Linear interpolation in different scenarios.

On the right graphic of Figure 3, it can be seen that linear interpolation does not perform accurate with wide missing intervals. Instead, as shown on the left graph, it can be very precise even with a high missing rates when the interval length is small.

2.2.2 *Arithmetic Mean*.

One of the earliest used method of imputation is the arithmetic mean, also known as unconditional mean imputation. In this approach, missing values m_t = at a point of time t are filled in by the average value of all observed time series $\bar{x_t}$ at that exact point of time t. The mean estimator is computed by:

$$\bar{x}_t = \frac{1}{N_{obs,t}} \sum_{n=1}^{N} \left(1 - m_{t,n}\right) a_{t,n} \tag{4}$$

where $N_{obs,t}$ is the number of observed values in a_t [6]. If all time series have missing values at time t, the arithmetic mean is not defined. therefore, optionally a consecutive imputation algorithm, such as linear interpolation, can be applied at the end to prevent this.

Figure 4 illustrates two scenarios of the performance of the arithmetic mean imputation. Again, the green dashed line is the original time series and the red points show the removed values. The graph colored in magenta presents the result of the respective imputation method.

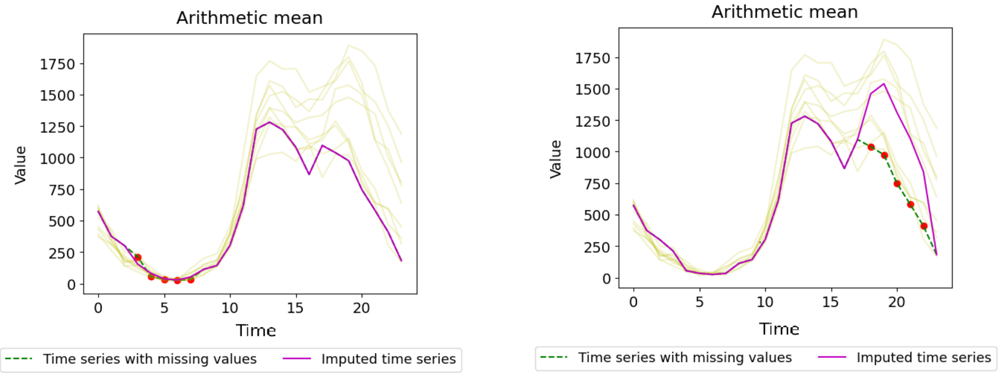

Fig. 4. Arithmetic mean imputation in different scenarios.

The arithmetic mean performs well as long there are no temporal distortions and the differences in amplitudes are small. The left graph in Figure 1 illustrates that in the time range of 5 to 10 the imputed time series is quite similar to the original one, even with a wide missing interval and high missing rate. However, in the right graph in the time range of 15 to 20 it can be seen that when the variance between the time series increases, the gap between the imputed time series and the original one is widening.

2.3 Dynamic-Time-Warping-based Methods

Dynamic time warping (DTW) provides a similarity metric between two datasets while relaxing the distance computation of the feature along a given axis. This is especially useful in the domain of time series analysis where feature observations of two time series might be shifted along the time axis but still be regarded as similar, e.g. speech patterns of the same word spoken with different speeds. In such a case, the usual euclidean metric would not be as usable as DTW. For that reason, DTW is a well known and commonly used method in time series analysis. For a summary of the DTW algorithm, refer to Algorithm 4 or refer to work that originally introduced the method [11].

2.3.1 ***DTW K-Nearest-Neighbors***. The standard DTW kNN, such as in [9], starts with the pre-processing of the source data using linear interpolation in order to avoid missing values. Then, the k nearest neighbors of each time series in the dataset have to be computed, for which the DTW distance is used as the distance measure. Following, the time series with missing values and its nearest neighbors are pair-wise aligned with DTW. Finally, the missing values are imputed by taking the arithmetic mean of the k neighbors at the missing value index. The linear interpolation in the beginning was done to prevent missing values in this step. This is repeated for every time series in the dataset, see Algorithm 1.

```
Algorithm 1: Dynamic time warping with k-nearest-neighbors imputation
Data:
$x\_missing$: univariate time series with missing values
$neighbors$: number of neighbors for DTW kNN
Result:
$x\_imputed$: imputed univariate time series
$x\_interpolated \leftarrow linear\_interpolation(x\_missing)$ // cf. Section 2.2.1
// the next line uses a kNN implementation, e.g. scikit-learn's [1] NearestNeighbors
$x\_neighbors \leftarrow nearest\_neighbors(x\_interpolated, neighbors)$
for $i = 0, .., length(x\_interpol)$ do
    $source$ list of neighboring time series used for imputation
    $x\_single\_neighbor\_imputed$ list of time series imputed using one of the k neighbors
    for $j = 0, ..., length(x\_neighbors_i)$ do
        $append(source, x\_interpolated_j)$
    end
    forall neighboring time series $s \in source$ do
        // the next line uses Algorithm 7
        $imputed \leftarrow dtw\_zero\_cost\_imputation(s, x\_missing_i)$
        $append(x\_single\_neighbor\_imputed, imputed)$
    end
    $x\_imputed_i \leftarrow mean(x\_single\_neighbor\_imputed)$
end
return $x\_imputed$
```

By using DTW kNN for imputation, the average of the k-nearest neighbors for a time series are most likely more similar than the average of the entire dataset. However, we have the addition of a hyperparameter k, which is the number of neighbors. To determine the best k for optimal results is not trivial and performance can vary greatly depending on the chosen k, see Figure 5. In this particular case of DTW-4NN, having only few neighbors to rely on for the imputation makes the quality decrease due to the lack of information contained in only the four nearest time series. With 12 neighbors, the quality seems to improve for imputations around the same region.

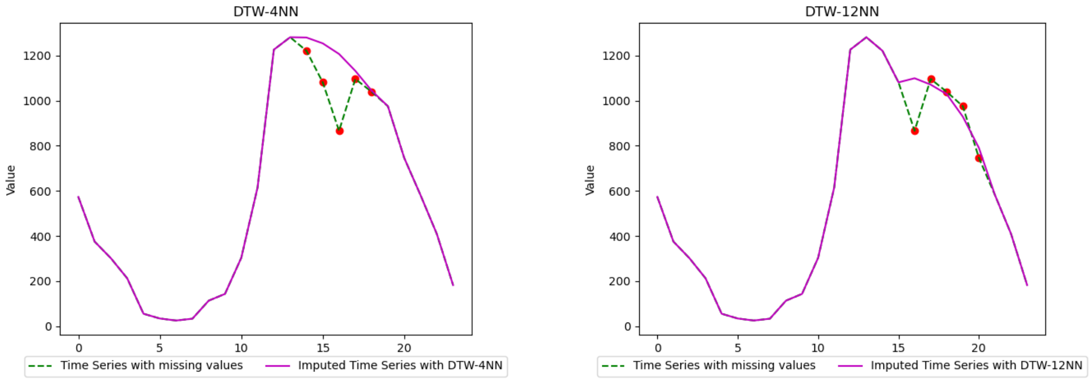

Fig. 5. DTW kNN imputation in different scenario.

3 DTW K-MEANS

This work implements a variation of DTW-based imputation methods, by utilizing clustering to find a meaningful centroid for a subset of a time series dataset. This centroid is then used to impute the missing values of the time series of the centroid's respective cluster. The detailed implementation is given in Algorithm 2. Furthermore, our method utilizes a so called zero-cost heuristic and the Fréchet mean, which will be elaborated in the following.

Algorithm 2: Dynamic time warping with k-means imputation

```
Data:
$X$ : Time series dataset with missing values
$k$ : Number of clusters for k-means computation
Result:
$I$ : Time series dataset $X$ with imputed missing values
// Initial imputation of X using naive aritmetic mean imputation
$X_{init} \longleftarrow$ arithmetic mean imputation of $X$ // cf. Section 2.2.2
// the next line uses Algorithm 3
$centroid, cluster \longleftarrow$ Compute k-means clustering $(X_{init}, k)$
$i \longleftarrow 0$
forall $x$ of $X$ do
    // the next line uses Algorithm 7
    $I_i \longleftarrow$ Compute imputations with $(centroid_j, x_j)$ // $where\ \ j \in 0,..,n\ \ for\ \ n\ \ clusters$
    $i \longleftarrow i + 1$
end
return I
```

3.0.1 ***Fréchet Mean.*** Rather than utilizing the mean in euclidean space, the imputation method implemented in this work applies the Fréchet mean, which is defined by the optimal solution of equation (1). In this work, this mean is used for the computation of the cluster centroids when computing the k-means clustering of a time series dataset. Utilizing the Fréchet mean is advantageous in the context of this work, since the DTW-metric is not applicable to the euclidean space metrics. The Fréchet function avoids this issue since it optimizes the mean over the variance of the

Algorithm 3: K-means under DTW-distance and Fréchet mean

```
Data:
$X$ : List of $n \times m$ source time series from which to impute
$k$ : number of clusters
Result:
$centroids$ : Computed k-Means centroids
$cluster$ : Computed k-Means cluster
$cluster(0, .., m) \leftarrow 0$
$centroids(0, .., k) \leftarrow k$ random samples of $X$
$new\_centroid(k \times m) \leftarrow 0$
while $centroids - new\_centroids \neq 0$ do
    forall $X_i \in X$ do
        $distance \leftarrow \infty$
        forall $c_j \in centroid$ do
            // the next line uses Algorithm 4
            $d \leftarrow$ Compute DTW-distance of $(c_j, X_i)$
            if $distance > d$ then
                $d \leftarrow distance$
                $cluster_i \leftarrow j$
            end
        end
        forall $X_j \in X$ do // $X_j \in X \mid j \in 0, .., n$ for n clusters
            /* the next line uses the Stochastic Subgradient Method as proposed in [12],
               Algorithm 3                                                          */
            $new\_centroids_i \leftarrow stochastic\_subgradient\_mean(X_j)$
        end
    end
    $centroids \leftarrow new\_centroids$
end
return $centroids, cluster$
```

given metric (here DTW), i.e. minimizing this variance. As mentioned in the introduction, finding an optimal solution of the Fréchet function (1) is a NP-hard problem and therefore has to be approximated iteratively. This work uses the Stochastic Subgradient Mean algorithm (SSG) for this computation, cf. [12].

The described properties of the Fréchet function also allow for its use as variance of the data it is used on. Therefore, this work also uses the value of the Fréchet function as an evaluation metric for the inertia of the clusters after applying k-means and the imputation, in the following referred to as *Fréchet variance* (F-Var). Since this value can vary widely over all the used datasets, the Fréchet variance is normalized for each dataset with its absolute maximum.

3.0.2 ***Zero-Cost Heuristic.*** The original DTW algorithm (cf. Algorithm 4) is not capable of handling time series containing missing values without adaptation, like deleting or estimating these occurrences beforehand or by enabling DTW to neglect them. Since missing values in time series are of special interest in this work, it also makes use of a heuristic, hereafter referred to as zero-cost heuristic, for handling missing values with DTW, when estimation of these values is not desired. The zero-cost heuristic assumes that for the missing value occurrence, DTW would be able to find

Algorithm 4: DTW metric and path computation

```
Data:
$s1$ : time series without missing values of length $N$
$s2$ : time series with missing values of length $M$
Result:
$d$ : DTW distance between the two time series
$p$ : computed warping path
/* Since this work uses the tslearn package [14] for DTW computations by modifying it to
   apply the a zero-cost heuristic, this algorithm, and the ones it makes use of are
   intentionally close to it's original implementation                                */
$missing\_value\_indices \leftarrow \arg(s2 \text{ is missing})$
// the next line uses Algorithm 5
$C \leftarrow$ Compute accumulated cost matrix with $(s1, s2, missing\_value\_indices)$
$d \leftarrow sqrt(C(N, M))$
// the next line uses Algorithm 6
$p \leftarrow$ Compute warping path with $(C, missing\_value\_indices)$
return $p, d$
```

an optimal sub-path in the respective cost matrix where the cost at the location of the missing value after imputation of the value would be zero. Respectively, after aligning the two time series along a computed warping path, the missing value after its imputation would coincide with its counterpart of the other time series.

Algorithm 5: DTW zero-cost heuristic: Accumulated cost matrix computation

```
Data:
$s1$ : time series without missing values of length $N$
$s2$ : time series with missing values of length $M$
$missing\_value\_indices$ : All indices where $s2$ has missing values
Result:
$C(N \times M)$ : Accumulated cost matrix
Initialize $C(N + 1 \times M + 1)$ with $\infty$
$C(0, 0) \leftarrow 0$
Increment all values of $s2$
forall $i \in 0, ..., N$ do
    forall $j \in 0, ..., M$ do
        if $(i$ is finite$)$ and $(j$ is finite$)$ then
            if $j \in missing\_value\_indices$ then
                $steps \leftarrow (C(i, j), C(i, j + 1), C(i + 1, j))$
                $C(i + 1, j + 1) \leftarrow (s1_i - s2_j)^2 + min(steps)$
            else // Zero-cost heuristic implementation
                $steps \leftarrow (C(i, j), C(i, j + 1))$
                $C(i + 1, j + 1) \leftarrow 0 + min(steps)$
            end
        end
    end
end
return $C(1...N, 1...M)$
```

Algorithm 6: DTW zero-cost heuristic: Warping path computation

```
Data:
$C(N \times M)$ : Accumulated cost matrix
$missing\_value\_indices$ : All indices where $s2$ has missing values
Result:
$p$ : computed warping path
$p_0 \leftarrow (N-1, M-1)$
$c \leftarrow 0$
while $p_c \neq (0,0)$ do
    $i, j \leftarrow p_c$
    if $i = 0$ then
        $p_{c+1} \leftarrow (0, j-1)$
    else if $j = 0$ then
        $p_{c+1} \leftarrow (i-1, 0)$
    else
        if $j \in missing\_value\_indices$ then
            $steps = (C(i-1, j-1), C(i, j-1), C(i-1, j))$
        else // Zero-cost heuristic implementation
            $steps = (C(i-1, j-1), C(i, j-1))$
        end
        if $argmin(steps) = 0$ then
            $p_{c+1} \leftarrow (i-1, j-1)$
        else if $argmin(steps) = 1$ then
            $p_{c+1} \leftarrow (i, j-1)$
        else
            $p_{c+1} \leftarrow (i-1, j)$
        end
    end
    $c \leftarrow c + 1$
end
return $reversed(p)$
```

Our implementation of the zero-cost heuristic also assumes that only one of the given time series contains missing values, since the other time series is needed for imputation. The implementation of the zero-cost heuristic for DTW differs from the original as follows:

(1) The cost at the location of a missing value index is set to zero when constructing the cost matrix.
(2) When computing the accumulated cost matrix, the algorithm cannot accumulate the cost along the axis of a missing value.
(3) When computing the optimal warping path of the accumulated cost matrix, the path cannot go along the axis of a missing value.

To summarize, the original DTW algorithm allows the accumulation of the cost matrix C and the path computation to follow three directions:

$$C(i,j), C(i,j+1), C(i+1,j) \ . \tag{5}$$

Algorithm 7: DTW zero-cost heuristic: Imputation

```
Data:
$C(N \times M)$ : Accumulated cost matrix
$missing\_value\_indices$ : All indices where $s2$ has missing values $s$ : source time series without missing values of length $N$
$t$ : target time series with missing values of length $M$
Result:
$t$ : $t$ with missing values imputed
$StopImputation \leftarrow False$
$p_0 \leftarrow (N-1, M-1)$
$c \leftarrow 0$
while $p_c \neq (0,0)$ do
    $i, j \leftarrow p_c$
    if $j \in missing\_value\_indices$ then
        $t_j \leftarrow s_i$
    end
    if $i = 0$ then
        $p_{c+1} \leftarrow (0, j-1)$
    end
    else if $j = 0$ then
        $p_{c+1} \leftarrow (i-1, 0)$
        $StopImputation \leftarrow True$
    end
    else
        if $j \in missing\_value\_indices$ then
            $steps = (C(i-1, j-1), C(i, j-1), C(i-1, j))$
        end
        else // Zero-cost heuristic implementation
            $steps = (C(i-1, j-1), C(i, j-1))$
        end
        if $argmin(steps) = 0$ then
            $p_{c+1} \leftarrow (i-1, j-1)$
        end
        else if $argmin(steps) = 1$ then
            $p_{c+1} \leftarrow (i, j-1)$
        end
        else
            $p_{c+1} \leftarrow (i-1, j)$
        end
    end
    $c \leftarrow c + 1$
end
if $(StopImputation = False)$ and $(0 \in missing\_value\_indices)$ then
    $t_0 \leftarrow s_0$
end
return $t$
```

The zero-cost heuristic used in this work reduces the directions that can be followed in the case that j is an index of a missing value, to

$$C(i, j), C(i, j+1)\ . \tag{6}$$

Furthermore, in this case the paired distance is also set to zero, cf. Algorithm 5, line 11. The complete algorithm with zero-cost adaptation consists of Algorithm 4, 5 and 6.

4 EXPERIMENT I: SHORT TIME SERIES

The following research questions will be answered.

(i) Comparison of DTW k-means with Fréchet mean to other common TS imputation methods: linear interpolation, arithmetic mean, and DTW k-nearest-neighbors
(ii) Analysis of the impact of missing value distribution, number of neighbors (DTW kNN) and number of clusters (DTW k-means) on imputation performance.

4.1 Data & Design

We used the time series from the most popular time series archive *UCR Time Series Classification* [3]. Over the years, this archive grew from originally 16 datasets to 128 datasets. Due to not having access to a cluster for the computation, we chose to select time series with a maximum length of 96, containing a maximum of 600 time series for the training. therefore, 19 datasets were included (Appendix A.1).

As the imputation performance metric, the R^2 score between the original and imputed data is used, which is an extension of the common model performance measure MSE, with variance of the dataset for normalization (Formula 7). Since we use multiple datasets from the UCR archive, the R^2 score is suitable to evaluate imputation performance of time series, and is commonly used in the literature [9, 17]. The R^2 score is defined from $]\infty, 1]$ with higher values indicating better imputation performance.

$$R^2 = 1 - \frac{\sum(y_i - \hat{y})^2}{\sum(y_i - \bar{y})^2} = 1 - \frac{\text{MSE}}{\text{Var}(y)} \tag{7}$$

where:

$$y = \textit{original time series}$$
$$\hat{y} = \textit{imputed value of } y$$
$$\bar{y} = \textit{ mean value of } y$$

For imitating missing values for the UCR datasets by deleting values at random, the missing rate and missing interval have to be chosen. The missing rate describes the percentage of missing values in a time series, while the missing interval refers to the length of missing value sequences. In case of DTW K-Means or DTW kNN imputation, we will have to choose the number of clusters and number of neighbors accordingly. For DTW K-Means, we chose to factor the number of clusters against the number of actual classes denoted by the UCR archive. We ran each hyperparameter combination per imputation method. The hyperparameter configuration can be seen in Table 1.

Area	**Parameter**	**Values**
Missing values	Missing rate	0.1, 0.2, 0.3, 0.4
	Missing interval	1, 2, 4, 8
DTW kNN	Neighbors	1, 2, 4, 8
DTW KM	Clusters	Classes * [0.25, 0.5, 1, 1.5, 2]

Table 1. Hyperparameter configuration for experiment 1.

We implemented our imputation methods in Python. For DTW-related implementations, we used the package TSLearn [14]: DTW distance with `metrics.dtw`, warping path using `metrics.dtw_path`, and the averaging of time series with Stochastic Subgradient Mean (SSG) using `barycenters.dtw_barycenter_averaging_subgradient`. For the DTW zero cost heuristic, we extended the DTW implementation of TSLearn. Linear interpolation was done with `interpolate.interp1d` from SciPy [15]. For finding the nearest neighboring time series for DTW K-Nearest-Neighbors, we used `neighbors.NearestNeighbors` from Scikit-Learn [10]. The clustering in DTW K-Means was implemented manually using TSLearn's DTW distance for measuring the distance to the centroids, and their SSG implementation for calculating new centroids.

4.2 Experimental Process

For each hyperparameter combination from Table 1, we created missing values in the complete dataset from Appendix A.1 with the missingness mechanism MAR (*Missing at random*). Next, we imputed the time series with missing values with one of the previously mentioned imputation methods: *linear interpolation*, *arithmetic mean*, *DTW kNN*, *DTW k-means*. We finished this step by calculating the R^2 score between the original time series and the imputed time series. This procedure was repeated for every imputation method with every hyperparameter combination on all datasets from Appendix A.1. Furthermore, 10 runs per dataset were made and averaged for more meaningful results.

4.3 Results

In this section, we present the results of our described experiment. We divide our analysis into two parts. In the first part of the analysis, we take a look at the overall performance results of the different imputation methods based on R^2. In the second part, we deep-dive into the effects of the different hyperparameters on the performance of individual imputation methods.

4.3.1 ***Imputation Performance.*** From the results in 6, we can see that our k-means method has the best overall performance when looking at the mean R^2 over all 10 runs. Arithmetic mean imputation also performs similarly, albeit slightly worse in both mean and median R^2. Linear interpolation takes the short end of the stick, and it is the worst performing method based on the mean R^2 by a significant margin. When we take the median R^2 of all runs into account, our k-means DTW imputation still presents the best value, with the rest of the methods displaying similar performance. It is worth noting that the DTW k-means and arithmetic mean based imputation have consistent values in terms of mean and median R^2.

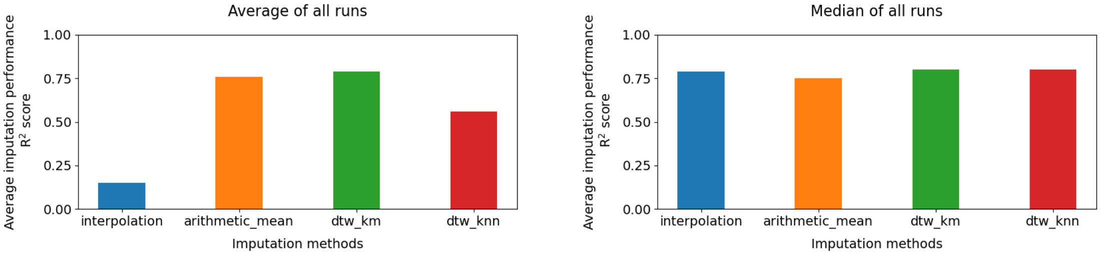

Fig. 6. Overall imputation performance (R²) per method on short datasets. Arithmetic mean over all 10 runs (left), and median (right)

4.3.2 *Effects of Hyperparameters on Imputation Performance.*

Missing Value Hyperparameters. As described in our experimental design and process in section 4.1, we take the original time series datasets and remove values based on two hyperparameters: **missing rate** and **missing interval**. In figures 7, and Figure 16 in Appendix A.2, we display how the average or median R² changes for all imputation methods depending on the value of the missing rate and missing value interval. For a missing interval of 1, we notice that linear interpolation seems to perform the best at smaller missing rates for small sequence lengths, but DTW k-means does the best at higher missing rates and longer missing intervals (average). The higher the missing rate intervals, the more it becomes clear that the k-means DTW based imputation has the highest R² performance, be that mean or median. At the highest missing interval of 8 and missing rate of 0.4, DTW kNN and linear interpolation have very low mean and median R² scores, which drop significantly faster for the mean R² scores, compared to the median.

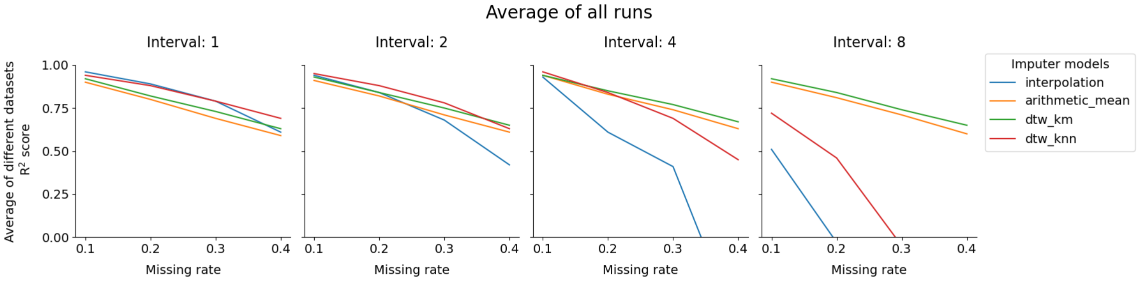

Fig. 7. Average imputation performance with varying missing interval length and missing rates on short datasets.

Nearest Neighbors. We analyzed the impact of neighbors size on DTW kNN performance, namely 1, 2, 4 or 8 nearest neighbors, see Figure 8 for the average of all runs, and Figure 17 in Appendix A.2 for the median. For shorter missing value intervals, the hyperparameter does not have a large impact on the R² across missing rates, but it is clear that 1NN performs the worst. As the missing value interval and the missing rates increase, the number of nearest neighbors greatly impacts imputation performance. We observe a clear trend here: the higher the k, the higher the R². Furthermore, a 'snapping' point is observed in all curves after the missing rate increases past a certain level, after which the R² drops quickly. This snapping point is also delayed with a higher k. In terms of mean and median R² we do not observe significant differences.

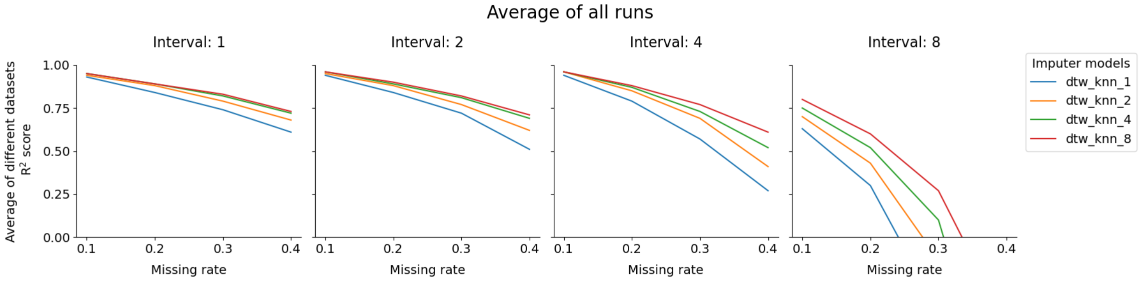

Fig. 8. Average DTW kNN imputation performance with varying missing interval length, missing rates, and number of nearest neighbors on short datasets.

Cluster Number. For our k-means DTW imputation method, we needed to tune the hyperparameter k, which controls the number of clusters the time series get grouped into. Our assumption was that the k would significantly impact imputation performance because it directly influences how many time series are taken into account for calculating the mean values with which we impute missing values. The value of k is a coefficient which we multiply with the number of classes in the original dataset, with k=1 being the true number of classes denoted by the UCR archive. The results of $k = 0.25, 0.5, 1, 1.5, 2$ are plotted in Figures 9, and Figure 18 in Appendix A.2. Just like with DTW kNN, for shorter missing value intervals the number of clusters does not have a large impact on the R² across missing rates, but it is clear that the lowest cluster factor performs the worst. As the missing value interval and the missing rates increase, the imputation performance gap between the cluster factors become bigger.

Just like with DTW kNN, the cluster factor 2 (the highest number of clusters) is the best performer in terms of mean and median R². However, the difference is not nearly as notable. When we take a look at lower levels of k, cluster factor 0.25 and cluster factor 0.5 are consistently poor. In the DTW kNN imputation, we observed a 'snapping point', which is not the case for DTW k-means. Overall, the performance of our DTW k-means imputation stays relatively consistent as we increase missing rate and missing interval length, both in mean and median R², regardless of the number of clusters.

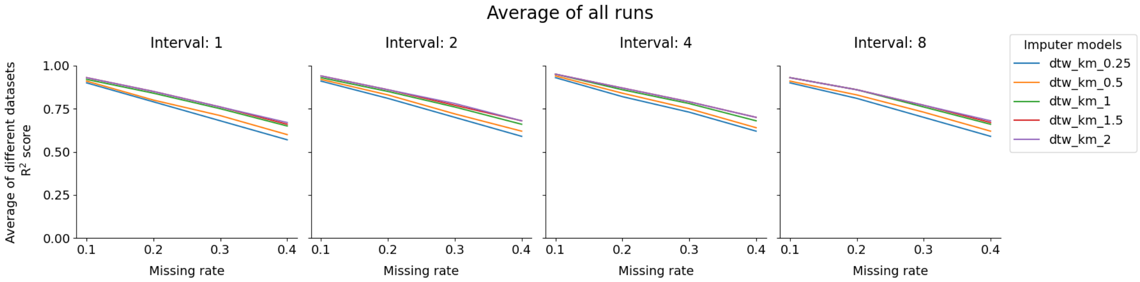

Fig. 9. Average DTW k-means performance with varying missing interval length, missing rates, and number of clusters on short datasets

4.4 Discussion

The imputation of missing time series data is ubiquitous in the literature, ranging from missing gene expressions from microarray datasets [16], to missing value imputation of time series air-quality data via deep neural networks [7]. The methods to address the missing data imputation problem are also manifold in this regard, as pointed out in [6], with the focus on statistical imputation methods like mean- or regression imputation, as well as machine learning-based methods like kNN- or neural network imputation. With our work, we present a new machine learning-based imputation algorithm for univariate time series. We use the k-means algorithm under dynamic time warping, extended with the zero-cost heuristic for handling missing values, and the Fréchet distance between the time series and cluster centroids. The imputation performance of our algorithm was evaluated on various UCR datasets [3] using the R^2 score in comparison with conventional imputation methods, namely arithmetic mean, linear interpolation and DTW kNN.

Overall, the algorithm presented by us achieves the best result, although only slightly better than the arithmetic mean, and is very consistent while doing so regardless of interval length. The good performance of the arithmetic mean method is surprising, since in other experiments [9], it usually performs worse than more sophisticated methods. If we look at the median, all algorithms achieve similar scores. Looking at the performance depending on the interval length and the missing rate, we find DTW kNN and interpolation react similar to changes and get worse with increasing missing rate. This is due to the fact that linear interpolation was chosen as the initialization method and therefore seems to have a big impact on performance.

If we consider the performance as a function of the Fréchet variance (see Fig. 10,normalization described in Section), we find a negative correlation. The greater the variance (including the Fréchet distance) within the clusters, the worse the R^2 score. This is to be expected, because the difference between imputation and true value increases with increasing variance. The difference can again become smaller the more clusters there are, which can also be seen from the fact that the data points with a higher number of clusters are located in the upper left part of the figure.

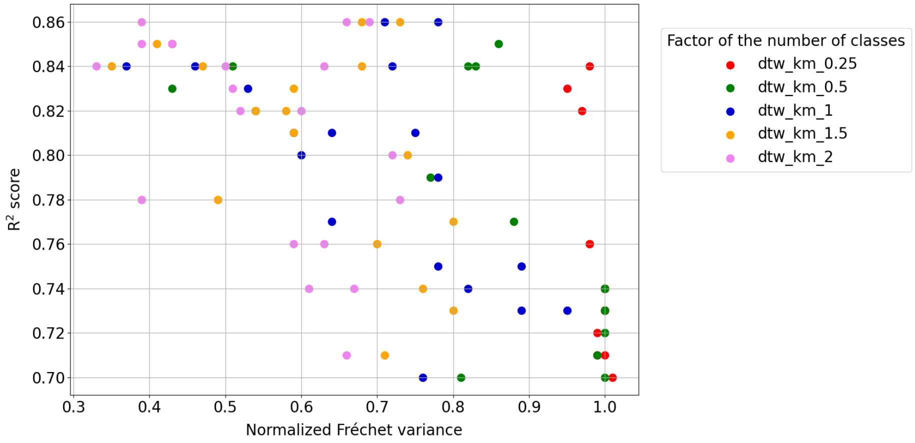

Fig. 10. R^2 Score of the DTW k-means algorithm as a function of the normalized Fréchet variance for small datasets.

Due to hardware limitations, it is not possible for us to perform more extensive tests on a greater number of datasets. The calculation of time series of large length are particularly computationally intensive. In this respect, our results only reflect the performance calculated on relatively short time series (maximum 94 data points). In a second experiment, we therefore investigate how our algorithm behaves with longer time series for a limited number of datasets. Future work could test our method on more datasets for more meaningful results

5 EXPERIMENT II: LONG TIME SERIES

In Section 4, we evaluated the imputation performance of linear interpolation, arithmetic mean, DTW kNN and DTW k-means on short time series. The performance was assessed for varying missing interval lengths, missing rates, cluster sizes and number of neighbors. The following experiment will investigate whether the imputation performance varies significantly for longer time series.

5.1 Data & Design

The design from Section 4 was replicated. We used the time series from the most popular time series archive *UCR Time Series Classification* [3]. Previously, due to not having access to a cluster for the computation, we chose to select only short time series. To validate our evaluation results, we will repeat our experiment for a small selection of longer time series (*time series length: [463, 637]*). In total, 7 datasets were selected (Appendix A.2). Again, we used the R² score between the original and imputed data as our imputation performance metric. Our hyperparameter configuration is identical, except that we were able to evaluate the imputation performance for higher sequence lengths due to the time series being longer (Table 2). We use the same implementations, as described in Section 4.

Area	**Parameter**	**Values**
Missing values	Missing rate Missing interval	0.1, 0.2, 0.3, 0.4 1, 5, 15, 30, 45
DTW kNN	Neighbors	1, 2, 4, 8
DTW KM	Clusters	Classes * [0.25, 0.5, 1, 1.5, 2]

Table 2. Hyperparameter configuration for experiment 2.

5.2 Experimental Process

The experimental process is identical to Section 4, but using the hyperparameter configuration from Table 2 applied to the datasets in Appendix A.2.

5.3 Results

In this section, we present the results of our second experiment. We divide our analysis into two parts, analogous to Section 4.3.

5.3.1 ***Imputation Performance***. Compared to the experiment on short time series, our experiment on longer time series (*length: [463, 637]*) provides slightly different average and median results. Unlike with the short time series, the arithmetic mean and k-means DTW imputation provide the lowest average and median R^2, whereas linear interpolation and DTW kNN seem to do the best. The linear interpolation median R^2 is by far the highest, almost perfect. When we look at the results of DTW kNN, the imputation performance is a lot better on average than on the short datasets. The median R^2 on the other hand is similar. As for k-means DTW, the imputation performance is very consistent, achieving similar results on long and short datasets.

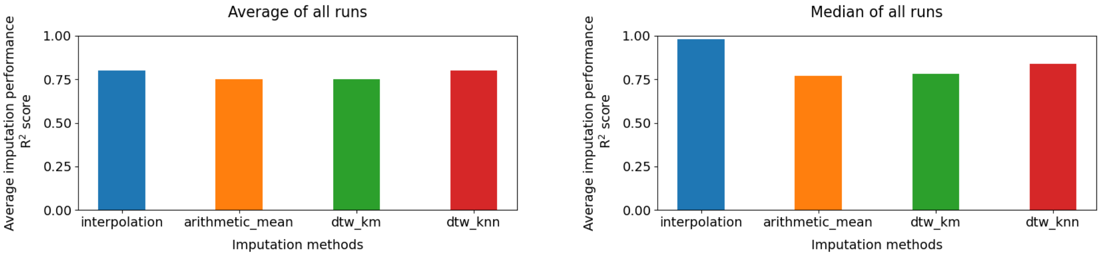

Fig. 11. Overall imputation performance (R^2) per method on long datasets. Arithmetic mean over all 10 runs (left), and median (right).

5.3.2 *Effects of Hyperparameters on Imputation Performance*.

Missing Value Hyperparameters. In Figure 12, we present the average performance for different imputation methods on long time series datasets (*length: [463, 637]*). The median performance can be found in the appendix A.2. For interval lengths between 1 and 15, the linear interpolation dominates clearly over the other methods. For the missing interval length of 30, the differences between the methods become smaller. When we extend the interval length to 45, linear interpolation performance deteriorates significantly, whereas DTW kNN stays very consistent.

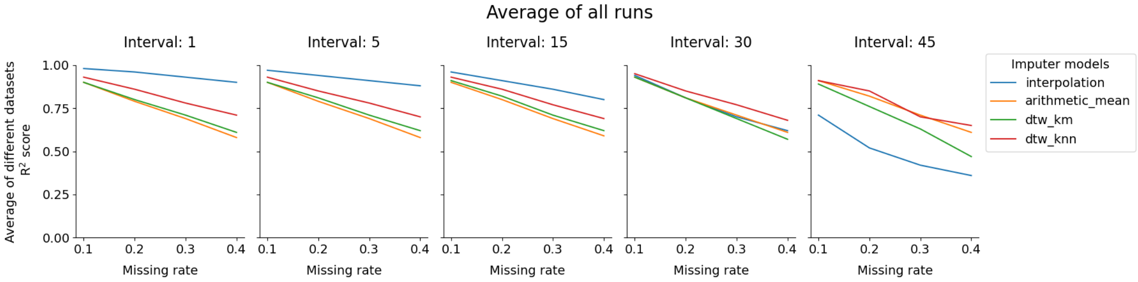

Fig. 12. Average imputation performance with varying missing interval length and missing rates on long datasets.

Nearest Neighbors. For long time series datasets, the DTW kNN imputation performs better on average than in short datasets (see Figure 13). The R^2 values also does not drop as drastically with increasing missing rate and missing interval length. Just like with short datasets, the higher the number of neighbors, the better the imputation quality. The only situation where this does not apply is for missing interval length 45, where the neighbors size 2 suddenly increases in performance for missing rate 0.4.

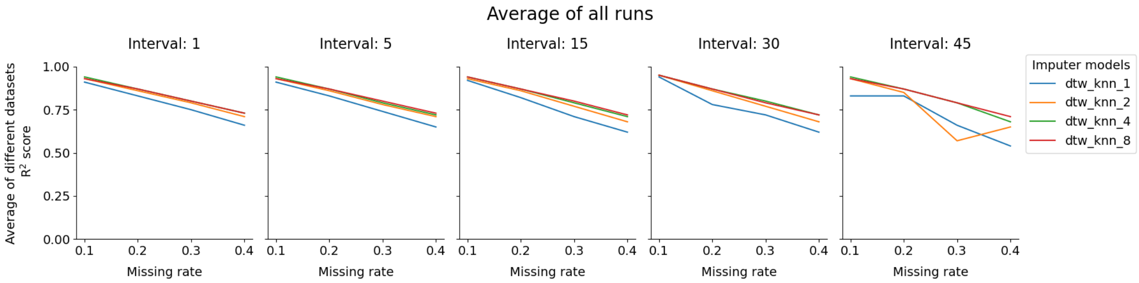

Fig. 13. Average DTW kNN performance with varying missing interval length, missing rates, and number of nearest neighbors on long datasets.

Cluster Number. The impact of the k in DTW k-means is not as big on long time series according to our experimental results, see Figure 14. The curves for the different approaches are very similar for missing intervals 1 to 30. However, a noticeable difference as we increase missing interval length is that the lower number of clusters end up providing the higher performance compared to k coefficients of 1, 1.5, and 2. The most significant performance drop comes from k=2, where it ranks best for shorter intervals but drops to the bottom position for longer missing intervals. Unlike with DTW kNN, the increase in k does not provide consistently better results for both experiments.

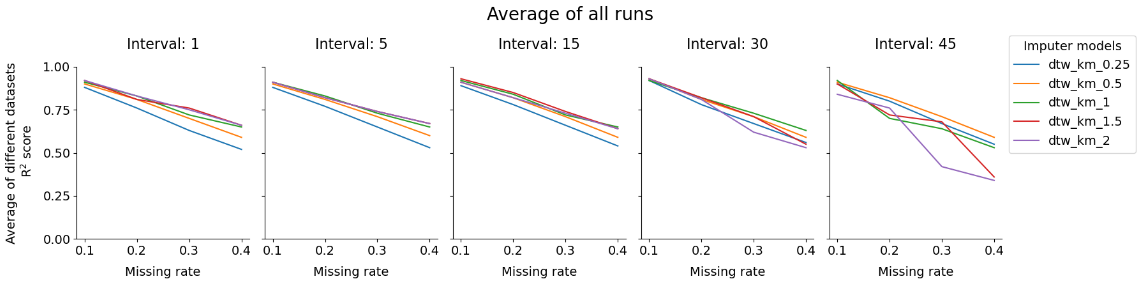

Fig. 14. Average DTW k-means performance with varying missing interval length, missing rates, and number of nearest neighbors on long datasets.

5.4 Discussion

The first thing that immediately stands out looking at the results compared to the first experiment is the performance jump for the interpolation and DTW kNN algorithm. Interpolation is usually the worst imputation method (see [9]), but remaining the highest compared to the other methods up to a missing interval length of 30. Since we chose longer time series, the absolute missing interval length has a smaller impact on longer time series. This scenario favors the interpolation algorithm and explains why it achieves such good results this time. Moreover, it is the initialization

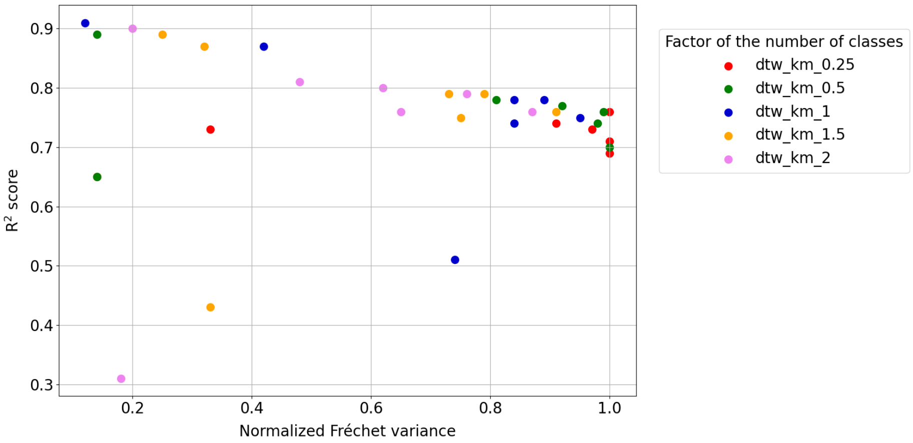

Fig. 15. R^2 score of the DTW k-means algorithm as a function of the normalized Fréchet variance on large datasets.

method for the DTW kNN algorithm and thus has a positive effect on it.

DTW kNN behaves similarly to the first experiment. It is noticeable that the snapping point observed in the shorter datasets is not as prominent on long datasets, as we increase the missing rate. This can be attributed to the absolute missing interval length having a smaller impact on longer time series. As with short time series, the arithmetic mean performs similar applied to large time series, only with a slight difference to DTW kNN and DTW k-means. The performance also remains constant over all interval lengths.

The DTW k-means algorithm presented by us succumbs to the DTW kNN algorithm in terms of performance at all interval lengths. The number of k classes seems to make only a slight difference in performance. This difference becomes increasingly larger as the interval length increase. Interestingly, the number of clusters this time has a slightly negative impact on performance, however we also have some strong outliers.

Looking at performance as a function of the normalized Fréchet variance (see Fig. 15, normalization described in Section), we again find a negative correlation. The larger the variance (including the Fréchet distance) within the clusters, the worse the R^2 value. However, the data points are significantly less scattered in this experiment. Even though it can be assumed that the number of clusters has a positive influence, no reliable statement can be made due to the lack of a larger number of datasets.

As in the first experiment, we are confronted with the problem of limited computational capacity. In particular, the results obtained by the linear interpolation are very surprising. No reliable statement can be made on the basis of this small number of datasets. A more extensive investigation is needed to critically examine these results. In addition, in future work, a method could be developed that allows to work independently of an initialization by means of another imputation method.

6 CONCLUSION

With this work, we introduced a new algorithm for imputing missing time series data using the k-means algorithm with the DTW distance, for which we implemented the zero-cost heuristic for handling missing values, and the Fréchet mean to calculate new centroids. For this purpose, we applied our algorithm to the UCR datasets [3] and used the R² score to compare the performance to conventional imputation methods. In doing so, our algorithm performed best with shorter datasets, albeit with a narrow lead over arithmetic mean in experiment 1 (see Chapter 4). Our method loses performance with longer time series, although here only a very small set of datasets was examined due to not having the computational capacities (see Chapter 5). As seen in experiment 1, it is likely superior to the conventional methods, which would need to be investigated by more extensive testing.

ACKNOWLEDGMENTS

We would like to thank *David Schultz* for his engagement and guiding throughout the semester, and therefore making this research possible.

REFERENCES

[1] Lars Buitinck, Gilles Louppe, Mathieu Blondel, Fabian Pedregosa, Andreas Mueller, Olivier Grisel, Vlad Niculae, Peter Prettenhofer, Alexandre Gramfort, Jaques Grobler, Robert Layton, Jake VanderPlas, Arnaud Joly, Brian Holt, and Gaël Varoquaux. 2013. API design for machine learning software: experiences from the scikit-learn project. In *ECML PKDD Workshop: Languages for Data Mining and Machine Learning*. 108–122.

[2] Laurent Bulteau, Vincent Froese, and Rolf Niedermeier. 2020. Tight Hardness Results for Consensus Problems on Circular Strings and Time Series. *SIAM Journal on Discrete Mathematics* 34 (2020), 1854–1883.

[3] Hoang Anh Dau, Eamonn Keogh, Kaveh Kamgar, Chin-Chia Michael Yeh, Yan Zhu, Shaghayegh Gharghabi, Chotirat Ann Ratanamahatana, Yanping, Bing Hu, Nurjahan Begum, Anthony Bagnall, Abdullah Mueen, Gustavo Batista, and Hexagon-ML. 2018. The UCR Time Series Classification Archive. https://www.cs.ucr.edu/~eamonn/time_series_data_2018/.

[4] Vincent Fortuin, Gunnar Rätsch, and Stephan Mandt. 2019. Multivariate Time Series Imputation with Variational Autoencoders. *ArXiv* abs/1907.04155 (2019).

[5] Maurice R. Fréchet. 1948. Les éléments aléatoires de nature quelconque dans un espace distancié. *Annales de l'institut Henri Poincaré* 10, 4, 215–310.

[6] Pedro J. García-Laencina, José-Luis Sancho-Gómez, and Aníbal R. Figueiras-Vidal. 2009. Pattern classification with missing data: a review. *Neural Computing and Applications* 19 (2009), 263–282.

[7] Taesung Kim, Jinhee Kim, Wonho Yang, Hunjoo Lee, and Jaegul Choo. 2021. Missing Value Imputation of Time-Series Air-Quality Data via Deep Neural Networks. *International Journal of Environmental Research and Public Health* 18, 22 (2021), 12213.

[8] Steffen Moritz, Alexis Sardá, Thomas Bartz-Beielstein, Martin Zaefferer, and Jörg Stork. 2015. Comparison of different Methods for Univariate Time Series Imputation in R. *ArXiv* abs/1510.03924 (2015).

[9] Stefan Oehmcke, Oliver Zielinski, and Oliver Kramer. 2016. kNN ensembles with penalized DTW for multivariate time series imputation. In *2016 International Joint Conference on Neural Networks (IJCNN)*. 2774–2781.

[10] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. 2011. Scikit-learn: Machine Learning in Python. *Journal of Machine Learning Research* 12 (2011), 2825–2830.

[11] Hiroaki Sakoe and Seibi Chiba. 1978. Dynamic programming algorithm optimization for spoken word recognition. *IEEE TRANSACTIONS ON ACOUSTICS, SPEECH, AND SIGNAL PROCESSING* 26 (1978), 43–49.

[12] David Schultz and Brijnesh Jain. 2018. Nonsmooth analysis and subgradient methods for averaging in dynamic time warping spaces. *Pattern Recognition* 74 (2018), 340–358.

[13] Aditya Sundararajan and Arif I. Sarwat. 2020. Evaluation of Missing Data Imputation Methods for an Enhanced Distributed PV Generation Prediction. In *Proceedings of the Future Technologies Conference (FTC) 2019*, Kohei Arai, Rahul Bhatia, and Supriya Kapoor (Eds.). 590–609.

[14] Romain Tavenard, Johann Faouzi, Gilles Vandewiele, Felix Divo, Guillaume Androz, Chester Holtz, Marie Payne, Roman Yurchak, Marc Rußwurm, Kushal Kolar, and Eli Woods. 2020. Tslearn, A Machine Learning Toolkit for Time Series Data. *Journal of Machine Learning Research* 21, 118 (2020), 1–6.

[15] Pauli Virtanen, Ralf Gommers, Travis E. Oliphant, Matt Haberland, Tyler Reddy, David Cournapeau, Evgeni Burovski, Pearu Peterson, Warren Weckesser, Jonathan Bright, Stéfan J. van der Walt, Matthew Brett, Joshua Wilson, K. Jarrod Millman, Nikolay Mayorov, Andrew R. J. Nelson, Eric Jones, Robert Kern, Eric Larson, C J Carey, İlhan Polat, Yu Feng, Eric W. Moore, Jake VanderPlas, Denis Laxalde, Josef Perktold, Robert Cimrman, Ian Henriksen, E. A. Quintero, Charles R. Harris, Anne M. Archibald, Antônio H. Ribeiro, Fabian Pedregosa, Paul van Mulbregt, and SciPy 1.0

Contributors. 2020. SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python. *Nature Methods* 17 (2020), 261–272.

[16] Rakhi Wajgi, Manali Kshirsagar, and Dipak Wajgi. 2017. Imputation of Missing Gene Expressions from Microarray Dataset: A Review. *International Journal of Computer Trends and Technology* 46 (04 2017), 15–22. https://doi.org/10.14445/22312803/IJCTT-V46P104

[17] L. Wijesekara and Liwan Liyanage. 2020. Comparison of imputation methods for missing values in air pollution data: Case study on Sydney air quality index. In *Future of Information and Communication Conference*. Springer, 257–269.

A APPENDIX

A.1 Experiment I: datasets

The following datasets from the UCR Archive were used for the experiments (*maximum time series length* = *96*, *maximum train set size* = 600):

ID	Type	Name	Train	Test	Class	Length
16	Image	DistalPhalanxOutlineAgeGroup	400	139	3	80
17	Image	DistalPhalanxOutlineCorrect	600	276	2	80
18	Image	DistalPhalanxTW	400	139	6	80
20	ECG	ECG200	100	100	2	96
38	Sensor	ItalyPowerDemand	67	1029	2	24
44	Image	MedicalImages	381	760	10	99
45	Image	MiddlePhalanxOutlineAgeGroup	400	154	3	80
46	Image	MiddlePhalanxOutlineCorrect	600	291	2	80
47	Image	MiddlePhalanxTW	399	154	6	80
48	Sensor	MoteStrain	20	1252	2	84
56	Image	ProximalPhalanxOutlineAgeGroup	400	205	3	80
57	Image	ProximalPhalanxOutlineCorrect	600	291	2	80
58	Image	ProximalPhalanxTW	400	205	6	80
64	Sensor	SonyAIBORobotSurface1	20	601	2	70
65	Sensor	SonyAIBORobotSurface2	27	953	2	65
70	Simulated	SyntheticControl	300	300	6	60
74	ECG	TwoLeadECG	23	1139	2	82
91	Traffic	Chinatown	20	343	2	24
127	Simulated	SmoothSubspace	150	150	3	15

A.2 Experiment II: datasets

The following datasets from the UCR Archive were used for the experiments (*maximum time series length* = *637*), *maximum train set size* = 322):

ID	Type	Name	Train	Test	Class	Length
3	Spectro	Beef	30	30	5	470
5	Image	BirdChicken	20	20	2	512
6	Sensor	Car	60	60	4	577
19	Sensor	Earthquakes	322	139	2	512
28	Image	Fish	175	175	7	463
40	Sensor	Lightning2	60	61	2	637
112	EPG	InsectEPGSmallTrain	17	249	3	601

A.3 Further results

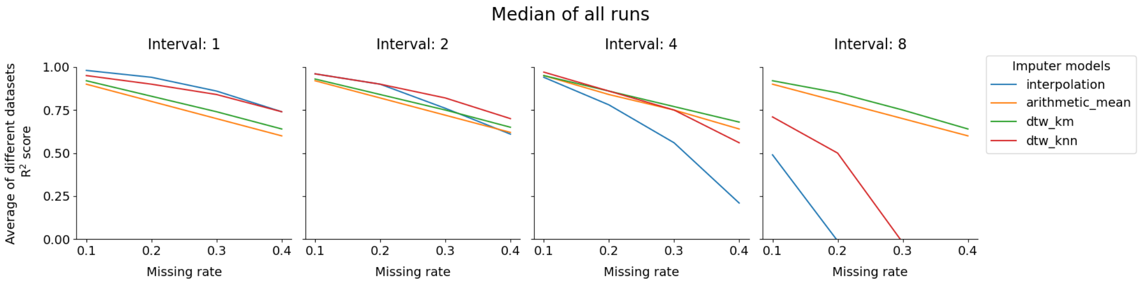

Fig. 16. Median imputation performance with varying missing interval length and missing rates on short datasets.

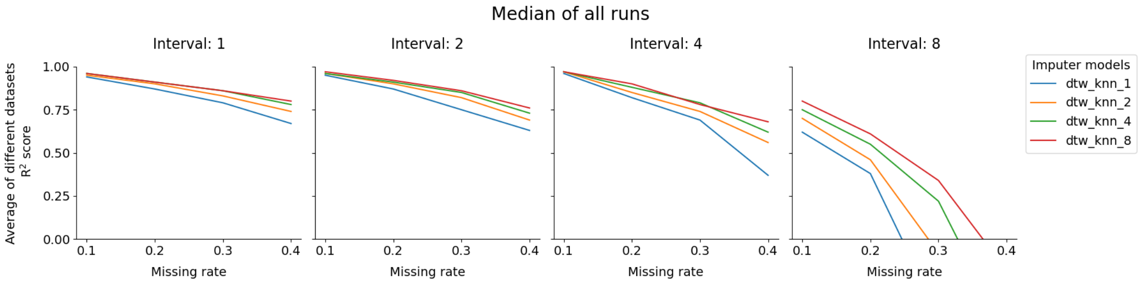

Fig. 17. Median DTW kNN imputation performance with varying missing interval length, missing rates, and number of nearest neighbors on short datasets.

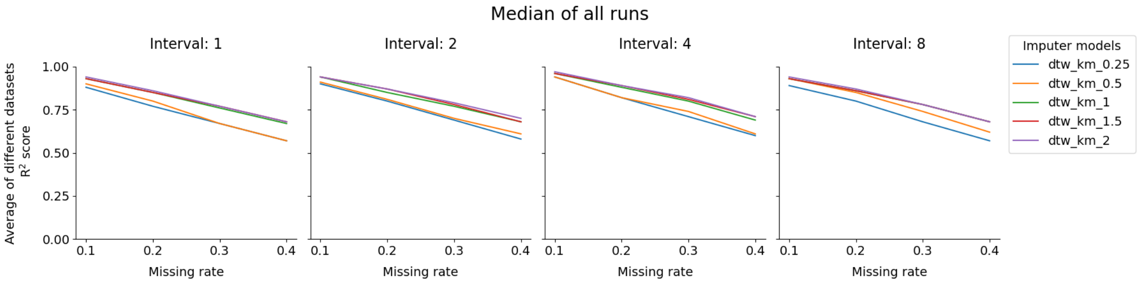

Fig. 18. Median DTW k-means performance with varying missing interval length, missing rates, and number of clusters on short datasets.

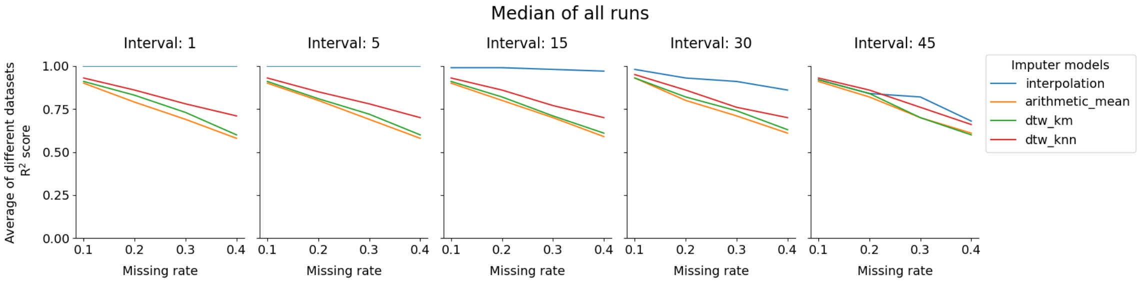

Fig. 19. Median imputation performance with varying missing interval length and missing rates on long datasets.

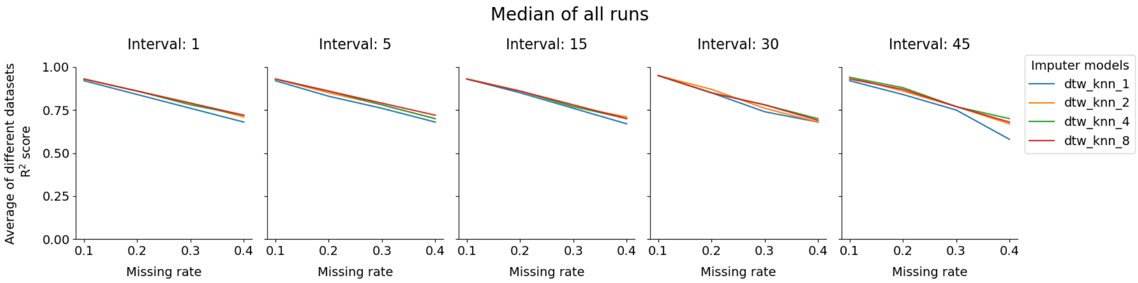

Fig. 20. Median DTW kNN performance with varying missing interval length, missing rates, and number of nearest neighbors on long datasets.

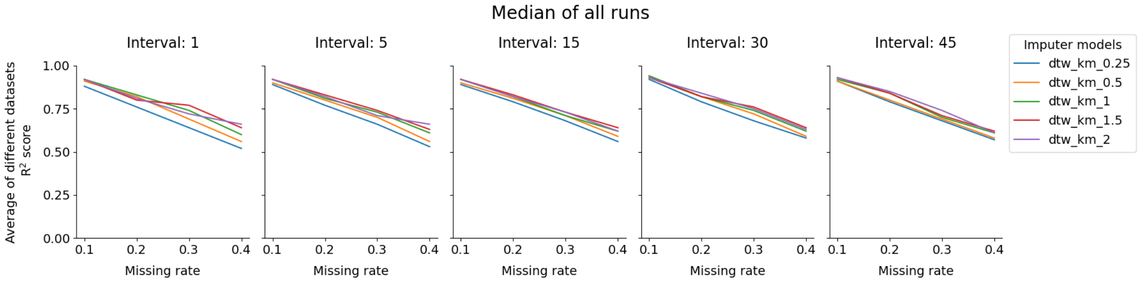

Fig. 21. Median DTW k-means performance with varying missing interval length, missing rates, and number of clusters on long datasets.