d_symb playground: an interactive tool to explore large multivariate time series datasets

Sylvain W Combettes, Paul Boniol, Charles Truong, Laurent Oudre

d_{symb} playground: an interactive tool to explore large multivariate time series datasets

Sylvain W. Combettes, Paul Boniol, Charles Truong, and Laurent Oudre
Université Paris-Saclay, Université Paris Cité, ENS Paris-Saclay, CNRS, SSA, INSERM, Centre Borelli
{firstname.name}@ens-paris-saclay.fr

***Abstract*—Exploring and comparing non-stationary multivariate time series is an important problem in many domains and real-world applications. In recent work, we introduced d_{symb}, a symbolic representation that transforms multivariate time series into interpretable symbolic sequences that comes along with a compatible and efficient distance measure to compare the obtained symbolic sequences. We have shown how d_{symb} can handle the non-stationarity of multivariate physiological signals, how interpretable the symbolization is, and how suitable the distance measure is compared to Dynamic Time Warping (DTW) variants. We have also empirically shown that the computation time when using d_{symb} on a clustering time is significantly smaller than with DTW variants (typically 100 times faster). In this demonstration, we present the d_{symb} playground, an interactive web-based tool to interpret and compare a large multivariate time series dataset quickly. We showcase the relevance of this tool in several scenarios based on real-world datasets.**

I. Introduction

The last decade has seen the proliferation of sensors in fields as diverse as meteorology, finance, healthcare, surveillance, and epidemiology. As a result, the amount of time series data has increased, ranging from univariate to multivariate or even multimodal data, where several quantities are recorded simultaneously. When collected over a long period of time, these time series are also likely to be non-stationary, making them even more challenging to study. Due to this increased complexity, these time series are often difficult to analyze and explore visually, creating a gap between the data and the practitioner. Another related issue is the ability to define a distance measure between multivariate time series, which could help to perform data mining tasks such as clustering, in order to study and form groups of time series with similar characteristics.

In this context, the main challenge is to define a distance measure between multivariate and possibly non-stationary time series that remains interpretable for the practitioner. Defining such a distance is challenging because it needs to consider both interactions between dimensions and abrupt changes in the time series caused by non-stationarity. There are several distance measures for multivariate signals in the literature [1]. For example, many variants of the popular Dynamic Time Warping (DTW) can be applied to multivariate time series, tackling temporal misalignments and allowing comparison between time series of different lengths. However, DTW distances are designed to find alignments directly on the raw waveforms, and the concept of *stationary phases* is not explicitly considered in the distance calculation. As shown in [2], this property makes them less suitable and less interpretable in the context of complex time series.

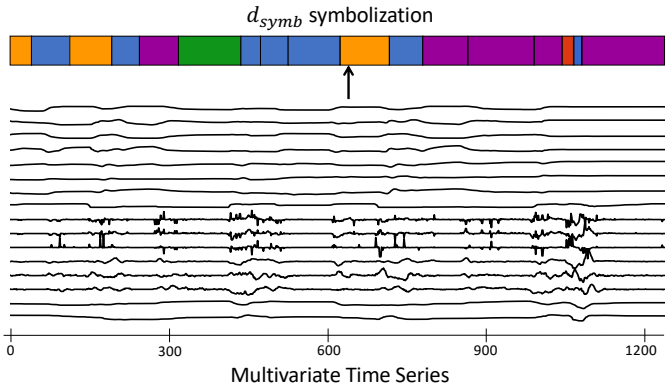


Fig. 1. Illustration of the d_{symb} symbolization applied to a multivariate time series from the JIGSAWS dataset [4].

In our previous work [2], we introduced an adaptive symbolization illustrated in Figure 1, that transforms a multivariate time series into a symbolic sequence that can be displayed as a colorbar. Each color is assigned to a symbol that is directly interpretable, as it represents a specific type of behavior within the time series. We also constructed a distance measure called d_{symb}, inspired by the general edit distance [3] and specifically crafted for these symbolic sequences. On a clustering task, we have shown that d_{symb} is more suitable than variants of DTW for comparing multivariate non-stationary time series. Moreover, because d_{symb} distances are computed on symbolic sequences of smaller lengths, the computation time for a standard clustering task is significantly lower than that of DTW or variants (typically 100 times faster).

In this demonstration, we introduce the d_{symb} playground to quickly explore, interpret, and compare multiple multivariate time series from a dataset. This interactive demonstration allows users to upload and visualize their multivariate time series and their d_{symb} symbolizations using the colorbars. With a single glance at the colorbars, the symbolization provides an immediate and comprehensive understanding of a dataset. Users can also visualize the d_{symb} pairwise distance matrix between the symbolic sequences. Furthermore, users can assess the relevance of the d_{symb} distance measure, with regards to 9 other distance measures, including variants of DTW, on the JIGSAWS dataset [4].

II. The d_{symb} symbolic representation and distance measure

We briefly describe our previous work, d_{symb} [2][1], a distance measure between non-stationary multivariate time series, using an interpretable symbolic representation as an intermediate step. This distance measure is designed to handle non-stationarity and to be interpretable. The proposed distance is computed in several steps:

1) The multivariate time series is partitioned into stationary segments using a change-point detection procedure,
2) Each stationary segment is assigned a symbol through K-means clustering,
3) Each multivariate time series is transformed into a symbolic sequence,
4) The final d_{symb} distance is computed as the general edit distance between the symbolic sequences.

Let $Q = (q_1, \dots, q_n)$ and $C = (c_1, \dots, c_m)$ be two real-valued multivariate time series of dimension d, of lengths n and m respectively. We assume that each dimension of Q and C have been normalized to zero mean and unit variance.

A. Step 1: Adaptive Segmentation

First, the raw time series are segmented using an adaptive change-point detection technique. Change-point detection aims at finding the w^* unknown instants $t_1^* < t_2^* < \ldots < t_{w^*+1}^*$ where some characteristics (here, the mean) of Q change abruptly. A recent review of such methods is given in [5]. In the context of our symbolization, the number of changes w^* is unknown and must be estimated too.

The change-points $\hat{t}_1, \dots, \hat{t}_{\hat{w}+1}$ (here $\hat{w}$ is the number of detected changes) can be estimated by solving the following optimization problem:

$$\begin{aligned}&\left(\hat{w}, \hat{t}_1, \dots, \hat{t}_{\hat{w}+1}\right)\\ &:= \underset{(w,t_1,\dots,t_{w+1})}{\arg\min} \sum_{k=0}^{w+1} \sum_{t=t_k}^{t_{k+1}-1} \|q_t - \bar{q}_{t_k:t_{k+1}}\|^2 + \lambda(w+1) \quad (1)\end{aligned}$$

where $\bar{q}_{t_k:t_{k+1}}$ is the empirical mean of $\{q_{t_k}, \dots, q_{t_{k+1}-1}\}$ and $\lambda > 0$ is a penalization parameter. (By convention, $t_0 := 0$ and $t_{w+1} := n$.) The penalized formulation (1) seeks a compromise between the reconstruction error given by the sum of quadratic errors and the complexity given by the number of change-points. Problem (1) is solved using the Pruned Exact Linear Time (PELT) algorithm [6], which is shown to have $\mathcal{O}(n)$ complexity under the assumption that the segment lengths are randomly drawn from a uniform distribution.

Intuitively, the λ parameter penalizes introducing a new change-point: when λ is small, many change-points are detected. We use the standard scaling $\lambda = \ln(n)$ [5] for calibration purposes.

[1]http://www.laurentoudre.fr/publis/ICDM2023.pdf

B. Step 2: Quantization

Once the segment boundaries have been determined for all multivariate time series in our dataset, the mean per segment (in dimension d) is computed for each multivariate time series. The means per segment are centered and scaled to unit variance. Then, these means per segment, from all segments of all multivariate time series in our dataset, are clustered using the K-means algorithm, where the number of clusters is the desired number of symbols A. Finally, each segment is attributed a symbol: the label of its associated cluster.

C. Step 3: Compute the d_{symb} distance measure

The proposed d_{symb} distance measure leverages the popular general edit distance [3], which is the minimal cost of a sequence of operations that transform a string into another using substitutions, insertions, and deletions.

In d_{symb}, the operation costs of the edit distance are defined to take into account the dissimilarity between individual symbols:

- The substitution cost $\mathrm{sub}(\mathtt{a}, \mathtt{b})$ for individual symbols $\mathtt{a}$ and $\mathtt{b}$ is the Euclidean distance between the cluster center $G_{\mathtt{a}}$ of symbol $\mathtt{a}$ and the cluster center $G_{\mathtt{b}}$ of symbol $\mathtt{b}$

$$\mathrm{sub}(\mathtt{a}, \mathtt{b}) = \|G_{\mathtt{a}} - G_{\mathtt{b}}\|_2 . \quad (2)$$

- For all characters, the insertion and deletion costs are fixed to $\mathrm{sub}_{\max}$, where $\mathrm{sub}_{\max}$ is the maximum value of the modified substitute costs in Formula (2).

The total cost is the sum of the costs of the elementary operations. Note that, given the costs, d_{symb} should do more substitutions than insertions or deletions. Moreover, d_{symb} can handle symbolic sequences of varying lengths, thanks to the insertion and deletion operations.

We incorporate the segment length information into the symbolic sequences by replicating each symbol proportionally to its segment length. Precisely, if ℓ is the minimum segment length on the data set, the symbol associated with a segment of length l will be replicated $\lfloor \frac{l}{\ell} \rfloor$ times. Finally, $d_{symb}(Q, C)$ is equal to the general edit distance between these replicated symbolic sequences. Since the general edit distance calculation is quadratic in the sequence length, the decimation provided by the ℓ factor results in a super fast distance calculation. For a typical value of $\ell = 10$, the number of operations is divided by $\ell^2 = 100$. This feature is very useful when dealing with large datasets, as will be seen in the demonstration.

III. The d_{symb} playground: system overview

In this section, we describe the d_{symb} playground, available online[2][3], and built using Python 3.9 and the Streamlit framework [7]. The d_{symb} playground, summarized in Figure 2, is a web interactive tool to explore large multivariate time series datasets. Our system is based on d_{symb} (described in the previous section) and inputs a multivariate time series dataset. The GUI is composed of three main frames, shown in

[2]https://dsymb-playground.streamlit.app

[3]https://github.com/boniolp/dsymb-playground

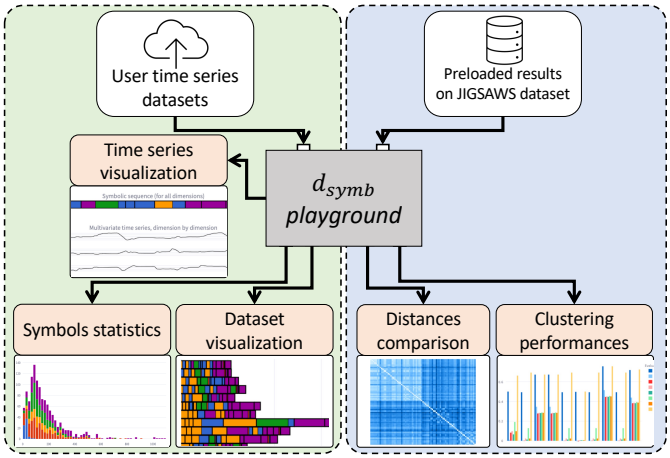


Fig. 2. Summary of d_{symb} playground inputs and features.

Figure 3: the **Individual analysis frame**, the **Dataset analysis frame**, and the **Benchmark frame**. The individual and dataset analysis frames enable users to explore and quickly gain insights thanks to the d_{symb} symbolization. The benchmark frame allows users to assess the performance of the d_{symb} distance compared to 9 existing distance measures on a real-world application.

As shown in Figure 3(B), for both the individual and dataset analysis frames, the user is required to upload their multivariate time series dataset and then select the number of symbols to be used in the d_{symb} symbolization. Each multivariate time series must be stored in a Comma-Separated Values (CSV) file of shape `(n_timestamps, n_dim)`. The user can choose the number of symbols as an integer between 2 and 25. Then, the d_{symb} computation is performed: the symbolization of all time series, as well as the pairwise distance matrix between the time series, are returned. We now describe the three main frames and their corresponding available actions in more detail.

[Individual analysis frame] The d_{symb} playground enables users to select a single time series and focus on its exploration. A visualization, shown in Figure 3(B), allows users to explore the raw multivariate time series along with its corresponding symbolic sequence represented as a colorbar. Therefore, users can interpret the multivariate segmentation from d_{symb}, which is based on changes in the mean, and investigate how it deals with the potential non-stationarity of the input time series. It also allows one to understand what a symbol represents with regard to real-world events: each symbol can be interpreted as an action, with a semantic meaning. For the plot of the raw multivariate time series, by default, the number of displayed dimensions on the same plot is capped at 20 for conciseness purposes. The user can investigate each group of 20 dimensions separately (while the displayed symbolic sequence is the one corresponding to all dimensions together). The user can also choose to visualize all dimensions at once.

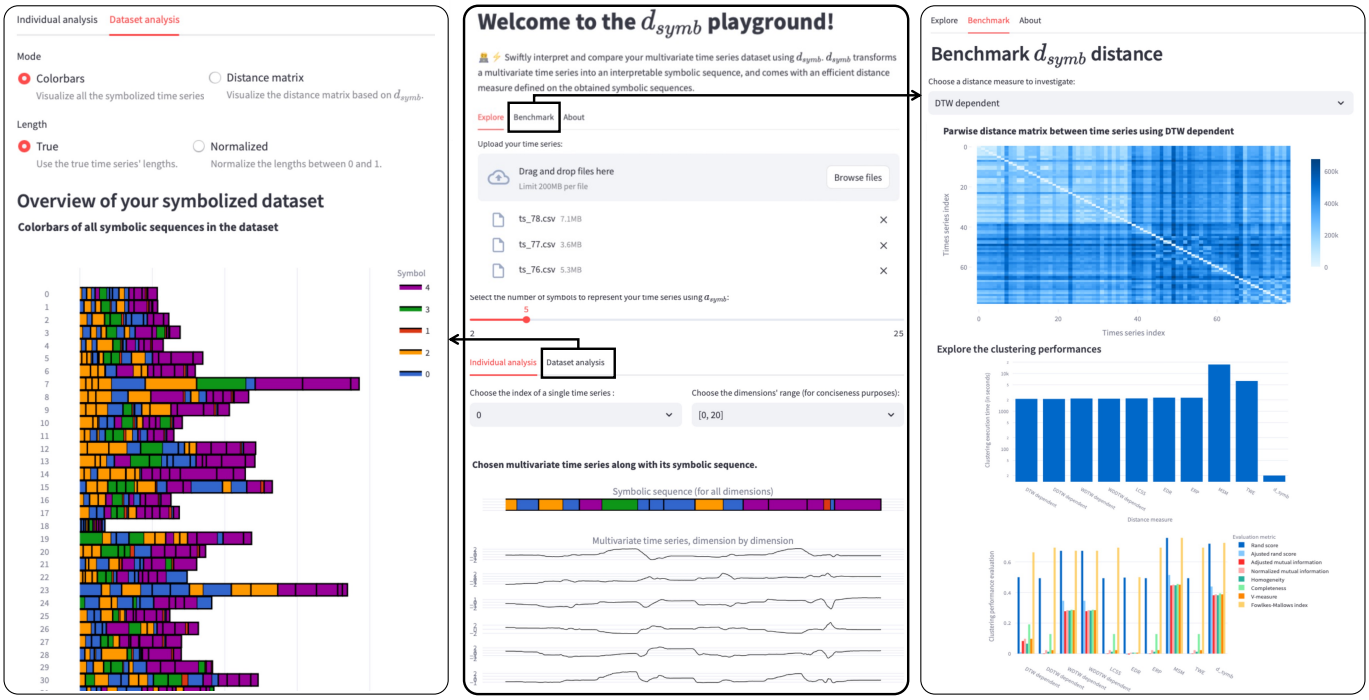
[Dataset analysis frame] Instead of focusing on a single time series, the dataset analysis frame explores the whole multivariate time series dataset at once. With a quick glance, the colorbars provide a compact representation of a dataset of multivariate time series, as displayed in Figure 3(A). Each row corresponds to the symbolic representation of each time series of the dataset. In a colorbar, black vertical lines illustrate change-points. Therefore, users can observe the different regimes that occur in the time series. The colorbars can be represented in two different ways: (i) the true lengths of the time series; or (ii) the normalized lengths. In the latter, all colorbars are stretched to have the same length. Scrolling down, more visualizations are available to help users understand the meaning of the symbols: (i) the histogram of the symbols, (ii) the distribution of the lengths for each symbol, (iii) the time stamps where each symbol occurs, and (iv) two figures illustrating the similarities between each individual symbol. Finally, the users can also visualize the pairwise distance matrix between the obtained symbolic sequences. Note that the users can modify the number of symbols at any time and, thanks to the fast computation of d_{symb}, all the visualizations described above are updated in real-time.

[Benchmark frame] The benchmark frame compares the d_{symb} distance measure to 9 existing distance measures on time series. We apply our benchmark to the real-world JIGSAWS dataset [4] with the goal of identifying surgeons' gestures based on kinematic time series. These signals are recorded during the use of robotic arms and grippers when performing surgical tasks. All results are precomputed (in order to save the users some computing time). In this dataset, we consider two surgical gestures: *Knot Tying* (39 multivariate time series) and *Needle Passing* (40 multivariate time series). The goal is to cluster (using an agglomerative clustering approach with complete linkage) and identify these two gestures, each time for several distance measures. As shown in Figure 3(C), we display the pairwise distance matrix for the chosen distance measure, as well as the clustering accuracy and the execution time (in seconds) for all distance measures in the benchmark.

IV. Demonstration Scenarios

In this demonstration, we propose three scenarios: (i) allowing the user to interpret the symbolization on a human locomotion dataset [8]; (ii) demonstrating the relevance of the d_{symb} visualization on a user-selected dataset; and (iii) challenging the user to evaluate the benefits and limitations of 10 distance measures, including d_{symb}, on a clustering task.

[Scenario 1: Interpreting d_{symb} on a human locomotion dataset] In this scenario, we provide the user with a dataset of gait signals [8] (i.e., time series), more precisely, the spectrograms of the gait signals, which correspond to non-stationary multivariate physiological time series. For each time series, the acquisition protocol of a patient is the following: standing still for 6 sec, walking 10 meters at the speed he felt comfortable with, turning around, walking back to the initial position, and standing for 2 sec. The goal of this scenario is two-fold: (i) make the user identify the protocol described above thanks to the d_{symb} symbolization, and (ii) allow the



(A) Dataset analysis frame (B) Individual analysis frame (C) Benchmark frame

Fig. 3. Illustration of the three main frames of the d_{symb} playground.

user to interpret each symbol as a real-world event from the protocol (such as standing still or walking).

[Scenario 2: Apply d_{symb} on a multivariate dataset chosen by the user] We will ask the user to upload a dataset of his choice, preferably one that he has worked with before and is familiar with its challenges. Similarly to Scenario 1, the goal is to let the user assess the benefit of the d_{symb} symbolization and distance on their own use cases. We will then ask the user to explain how accurate and efficient d_{symb} is.

[Scenario 3: Assess the relevance of d_{symb} on a clustering task] In the last scenario, we will ask the user to explore the precomputed benchmark showcased in our demo. 10 distance measures, including d_{symb} and variants of DTW, are applied to a clustering task on the JIGSAWS dataset [4]. We will discuss with the user the compromise between accuracy and efficiency, for each distance measure, and show that d_{symb} is much faster than existing distances (typically 100 times faster), and its accuracy is top 2.

V. Conclusions

We demonstrate the d_{symb} playground, a system that allows users to visualize and explore their multivariate time series dataset. Each time series is transformed into a symbolic sequence displayed as a colorbar. The latter allows users to understand their dataset at a glance, and each symbol can be linked to a real event. The d_{symb} distance measure is also shown to be accurate and fast compared to existing methods such as variants of DTW. More generally, such a tool can be beneficial as a preliminary step for more complex tasks, such as classification [9] and anomaly detection [10].

Acknowledgments

S. Combettes is supported by the IDAML chair (ENS Paris-Saclay) and UDOPIA (ANR-20-THIA-0013-01). C. Truong is funded by the PhLAMES chair (ENS Paris-Saclay).

References

[1] A. Shifaz, C. Pelletier, F. Petitjean, and G. I. Webb, "Elastic similarity and distance measures for multivariate time series," *Knowl Inf Syst*, 2023.

[2] S. W. Combettes, C. Truong, and L. Oudre, "An interpretable distance measure for multivariate non-stationary physiological signals," in *2023 IEEE International Conference on Data Mining Workshops (ICDMW)*, 2023, pp. 533–539.

[3] V. I. Levenshtein *et al.*, "Binary codes capable of correcting deletions, insertions, and reversals," in *Soviet Physics Doklady*, vol. 10, 1966, pp. 707–710.

[4] Y. Gao, S. S. Vedula, C. E. Reiley, N. Ahmidi, B. Varadarajan, H. C. Lin, L. Tao, L. Zappella, B. Béjar, D. D. Yuh *et al.*, "The jhu-isi gesture and skill assessment working set (jigsaws): A surgical activity dataset for human motion modeling," in *Modeling and Monitoring of Computer Assisted Interventions (M2CAI) – MICCAI Workshop*, 2014.

[5] C. Truong, L. Oudre, and N. Vayatis, "Selective review of offline change point detection methods," *Signal Processing*, vol. 167, p. 107299, 2020.

[6] R. Killick, P. Fearnhead, and I. A. Eckley, "Optimal detection of changepoints with a linear computational cost," *Journal of the American Statistical Association*, vol. 107, no. 500, pp. 1590–1598, 2012.

[7] Streamlit documentation. https://streamlit.io/.

[8] C. Truong, R. Barrois-Müller, T. Moreau, C. Provost, A. Vienne-Jumeau, A. Moreau, P.-P. Vidal, N. Vayatis, S. Buffat, A. Yelnik, D. Ricard, and L. Oudre, "A Data Set for the Study of Human Locomotion with Inertial Measurements Units," *Image Processing On Line*, vol. 9, pp. 381–390, 2019.

[9] T. L. Nguyen, S. Gsponer, and G. Ifrim, "Time series classification by sequence learning in all-subsequence space," *2017 IEEE 33rd International Conference on Data Engineering (ICDE)*, pp. 947–958, 2017.

[10] Y. Gao, J. Lin, and C. Brif, "Ensemble grammar induction for detecting anomalies in time series," in *Proceedings of the 23rd International Conference on Extending Database Technology, EDBT 2020, Copenhagen, Denmark*, 2020, pp. 85–96.