CDPS: Constrained DTW-Preserving Shapelets

Hussein El Amouri, Thomas Lampert, Pierre Gançarski, Clément Mallet

CDPS: Constrained DTW-Preserving Shapelets *

Hussein El Amouri ✉[1], Thomas Lampert ✉[1], Pierre Gançarski ✉[1], and Clément Mallet ✉[2]

[1] ICube, University of Strasbourg, France.
{helamouri,lampert,gancarski}@unistra.fr
[2] Univ Gustave Eiffel, IGN, ENSG, Saint-Mande, France.
clement.mallet@ign.fr

Abstract. The analysis of time series for clustering and classification is becoming ever more popular because of the increasingly ubiquitous nature of IoT, satellite constellations, and handheld and smart-wearable devices, etc. The presence of phase shift, differences in sample duration, and/or compression and dilation of a signal means that Euclidean distance is unsuitable in many cases. As such, several similarity measures specific to time-series have been proposed, Dynamic Time Warping (DTW) being the most popular. Nevertheless, DTW does not respect the axioms of a metric and therefore Learning DTW-Preserving Shapelets (LDPS) have been developed to regain these properties by using the concept of shapelet transform. LDPS learns an unsupervised representation that models DTW distances using Euclidean distance in shapelet space. This article proposes constrained DTW-preserving shapelets (CDPS), in which a limited amount of user knowledge is available in the form of must link and cannot link constraints, to guide the representation such that it better captures the user's interpretation of the data rather than the algorithm's bias. Subsequently, any unconstrained algorithm can be applied, e.g. K-means clustering, k-NN classification, etc, to obtain a result that fulfils the constraints (without explicit knowledge of them). Furthermore, this representation is generalisable to out-of-sample data, overcoming the limitations of standard transductive constrained-clustering algorithms. CLDPS is shown to outperform the state-of-the-art constrained-clustering algorithms on multiple time-series datasets. An open-source implementation based on PyTorch is available[1], which takes full advantage of GPU acceleration.

Keywords: Shapelets · Semi-supervised Learning · Clustering · Constrained-Clustering · Time series · Learning representation.

* This work was supported by the HIATUS (ANR-18-CE23-0025) and HERELLES (ANR-20-CE23-0022) ANR projects. We thank Nvidia Corporation for donating GPUs and the Centre de Calcul de l'Université de Strasbourg for access to the GPUs used for this research.

[1] From: https://git.unistra.fr/helamouri/constrained-dtw-preserving-shapelets

1 Introduction

The availability of time series data is increasing rapidly with the development of sensing technology and the increasing number of fields that uses such technology. This increase in data volume means that providing ground truth labels becomes difficult due to the time and cost needed. Labelling difficulty is exacerbated when making exploratory analyses and when working in nascent domains for which classes are not well defined. For that reason, supervised approaches such as classification become unfeasible and unsupervised clustering is often preferred. However, unsupervised approaches may lead to irrelevant or unreliable results since they have no knowledge about the user's requirements and are instead lead by the algorithm's bias. On the other hand, semi-supervised algorithms try to remove the rigid requirements of supervised approaches but retain the ability of a user to guide the algorithm to produce a meaningful output. This can be achieved by providing a set of constraints to the algorithm that encode some expert knowledge. These can take many forms but this work is concerned with must-link and cannot-link constraints since they are the easiest to interpret and provide.

Must-link and cannot-link constraints do not define what a sample represents (a class), instead they label pairs of samples as being the same (must-link), thus belong to the same cluster, or not (cannot-link). In this way the algorithm is guided to converge on a result that is meaningful to the user without explicitly, nor exhaustively labelling samples. Generally, time series are characterised by trend, shapes, and distortions either to time or shape [22] and therefore exhibit phase shifts and warping. As such, the Euclidean distance is unsuitable and several similarity measures specific to time-series have been proposed [17], for example compression-based measures [7], Levenshtein Distance [10], Longest Common Subsequnce [25] and Dynamic Time Warping (DTW) [19,20]. DTW is one of the most popular since it overcomes these problems by aligning two series through the computation of a cost function based on Euclidean distance [8], it is therefore known as an elastic measure [17]. Moreover, Paparrizos et al. show that DTW is a good basis for calculating embeddings, an approach that employs a similarity to construct a new representation. Time series also exhibit complex structure which are often highly correlated [22]. This makes their analysis difficult to achieve and time consuming, indeed several attempts to accelerate DTW's computation have been proposed [1,22]. Shapelets [30] offer a simpler approach to increase the accuracy of time-series analysis. Shapelets are phase-independent discriminative sub-sequences extracted or learnt to form features that map a time-series into a more discriminative representational space, therefore increasing the reliability and interpretability of downstream tasks. Since DTW does not respect the axioms of a metric, LDPS [13] extends shapelets to preserve DTW distances in a Euclidean embedding.

The contribution of this article is to introduce constrained DTW-preserving shapelets (CDPS), in which a time series representation is learnt to overcome time series distortions by approximating DTW and is influenced by a limited amount of user knowledge by providing constraints. Thus CDPS can model a

user's interpretation, rather than being influenced by the algorithm's bias. Subsequently, any unconstrained algorithm can be applied to the embedding, e.g. K-means clustering, k-NN classification, etc, to obtain a result that fulfils the constraints (without explicit knowledge of them). The proposed embedding process is studied in a constrained clustering setting, on multiple datasets, and is compared to COP-KMeans [27], FeatTS [23], and unsupervised DTW-preserving shapelets [13].

The representational embedding that is learnt by CDPS is generalisable to out-of-sample data, overcoming the limitations of standard constrained-clustering algorithms such as COP-KMeans. It is interpretable, since the learnt shapelets can themselves be visualised as time-series. Finally, since CDPS results in a vectorial representation of the data, they and the constraints can be analysed using norm-base measures, something that is not possible when using DTW as a similarity measure [8]. This opens up the possibility of measuring constraint informativeness [3] and constraint consistency [26] in time-series clustering, and explaining and interpreting the constraints, which is a concern for future work. Such measures, and notions of density, are needed to develop novel interactive and active constrained clustering processes for time-series.

The rest of this article is organised as follows: in Section 2 related work is reviewed, in Section 3 the Constrained DTW-Preserving Shapelets (CDPS) algorithm is presented, in Section 4 CDPS is compared to constrained/semi-supervised and unconstrained approaches from the literature, and finally Section 6 presents the conclusions and future work.

2 Related Work

This section will present works related to shapelets and constrained clustering.

2.1 Shapelets

Shapelets are sub-sequences of time-series that were originally developed to discriminate between time-series using a tree based classifier [30,31]. As such, the shapelets themselves were chosen from a set of all possible sub-sequences of the set of time series being analysed, which is time consuming and exhaustive. Different approaches are proposed to increase the speed of finding shapelets. Rakthanmanon and Keogh [18] propose to first project the time-series into a symbolic representation to increase the speed of discovering the shapelets. Subsequently, Mueen et al. [14] introduce logical shapelets, which combines shapelets with complex rules of discrimination to increase the reliability of the shapelets and their ability to discriminate between the time-series. Sperandio [22] presents a detailed review of early shapelet approaches.

Lines et al. [12] proposed a new way of handling shapelets that separated classification from transformation. This was later extended by Hills et al. [6] to the shapelet transform, which transforms the raw data into a vectorial representation in which the shapelets define the representation space's bases. It was

proved that this separation leads to more accurate classification results even with non-tree based approaches.

2.2 Learning Shapelets

In order to overcome the exhaustive search for optimal shapelets, Grabocka et al. [4] introduce the concept of learning shapelets in a supervised setting. In this approach the optimal shapelets are learnt by minimising a classification objective function. The authors consider shapelets to be features to be learnt instead of searching for a set of possible candidates, they report that this method provides a significant improvement in accuracy compared to previous search based approaches. Other supervised approaches have been proposed, Shah et al. [21] increase accuracy by learning more relevant and representative shapelets. This is achieved by using DTW similarity instead of Euclidean distance, since it is better adapted to measure the similarity between the shapelets and the time-series. Another approach for learning shapelets is to optimise the partial AUC [29], in which shapelets are learnt in conjunction with a classifier.

2.3 Unsupervised Shapelets

Zakaria et al. [32] introduced the first approach for clustering time-series with shapelets, called unsupervised-shapelets or u-shapelets. U-shapelets are those that best partition a subset of the time series from the rest of the data set. The shapelets are chosen from a set of all possible sub-sequences by partitioning the dataset and removing the time series that are similar to the shapelet, this process is repeated until no further improvements (i.e. partitions) can be made. It is therefore an exhaustive search, as were the early supervised approaches. U-shapelets have been used in several works since their initial introduction [24,33]. Since these unsupervised methods take a similar approach to the original supervised shapelets, they have the same drawbacks. To overcome these, Zhang et al. [34] propose to combine learning shapelets with unsupervised feature selection to learn the optimal shapelets. Learning DTW-preserving shapelets (LDPS) expands the learning paradigm for shapelets by integrating additional constraints on the learnt representation. In LDPS these constrain the representation space to model the DTW distances between the time-series, instead of focusing on learning shapelets that best discriminate between them.

A multitude of other unsupervised approaches to build an embedding space for time series exist (other than shapelets) and Paparrizos et al. [17] provide an extensive study of them. Generic Representation Learning (GRAIL) [15], Shift Invariant Dictionary Learning (SIDL) [35], Preserving Representation Learning method (SPIRAL) [9], and Random Warping Series (RWS) [28] are different approaches to building such representations. Since these are unsupervised they are not of concern in this article.

2.4 Constrained Clustering

Constrained clustering algorithms are those that add expert knowledge to the process such as COP-Kmeans [27] and Constraint Clustering via Spectral Regularization (CCSR) [11]. Constraints can be given in different forms such as cluster level constraints and instance level constraints. Must-link and cannot-link constraints between samples fall under the latter.

Many constrained clustering algorithms have been proposed, some of which have been adapted to time-series. For a full review, the reader is referred to [8]. Here, those relevant to this study are mentioned. COP-KMeans is an extension to k-Means that often offers state-of-the-art performance without the need to choose parameters [8]. Cluster allocations are validated using the constraint set at each iteration to verify that no constraints are violated. For use with time-series the DTW distance measure is often used along with an appropriate averaging method such as DTW barycenter averaging (DBA) [8] to calculate the cluster centres. Another semi-supervised approach developed specifically for clustering time series is FeatTS [23]. FeatTS uses a percentage of labeled samples to extract relevant features used to calculate a co-occurrence matrix from a graph created by the features. The co-occurrence matrix is then used to cluster the dataset.

Other approaches to time-series clustering exist, such as k-shape [16], however being unsupervised, these fall outside the scope of this article.

3 Constrained DTW-Preserving Shapelets

This section proposes Constrained DTW-Preserving Shapelets (CDPS), which learns shapelets in a semi-supervised manner using ML and CL constraints. Therefore allowing expert knowledge to influence the transformation learning process, while also preserving DTW similarity and interpretability of the resulting shapelets. Definitions and notations are presented in sub-section 3.1, and the algorithm in sub-section 3.3.

3.1 Definitions and Notations

Time series: *is an ordered set of real-valued observations. Let* $\mathcal{T} = \{T_1, T_2, \ldots, T_N\}$ *be a set of* N *uni-dimensional time series (for simplicity of notation, nevertheless CDPS is also applicable to multi-dimensional time series).* L_i *is the length of a time series such that* T_i *is composed of* L_i *elements (each time-series may have different lengths), such that*

$$T_i = T_{i,1}, \ldots, T_{i,L_i}. \tag{1}$$

A segment of a time series T_i *at the* m^{th} *element with length* L *is denoted as* $T_{i,m:L} = \{T_{i,m}, \ldots, T_{i,L}\}$.

Shapelet: *is an ordered set of real-valued variables, with a length smaller than, or equal to, that of the shortest time series in the dataset. Let a Shapelet be denoted as S having length L_k. Let $\mathcal{S} = \{S_1, \ldots, S_K\}$ be a set of K shapelets, where $S_k = S_{j,1:L_k}$. In our work, the set $\mathcal{S}$ can have shapelets with different lengths, but for the simplicity we will use shapelets with same length in the formulation.*

Squared Euclidean Score: *is the similarity score between a shapelet S_k and a time series sub-sequence $T_{i,m:L_S}$, such that*

$$D_{i,k,m} = \frac{1}{l} \sum_{x=1}^{l} (T_{i,m+x-l} - S_{k,x})^2. \tag{2}$$

Euclidean Shapelet Match: *represents the matching score between shapelet S_k and a time series T_i, such that*

$$\overline{T}_{i,k} = \min_{m \in \{1:L_i - L_k + 1\}} D_{i,k,m}. \tag{3}$$

Shapelet transform: *is the mapping of time series T_i using Euclidean shapelet matching with respect to the set of shapelets $\mathcal{S}$. Where the new vectorial representation is*

$$\overline{T}_i = \{\overline{T}_{i,1}, \ldots, \overline{T}_{i,K}\}. \tag{4}$$

Constraint Sets: *Let C_k be the k^{th} cluster, ML be the set containing time series connected by a must link and CL the set such that they are connected by a cannot link. Thus, $\forall\ T_i, T_j$ such that $i, j \in \{1, \ldots, N\}$ and $i \neq j$ we have*

$$ML = \{(i,j) | \forall\ k \in \{1, \ldots, K\}, T_i \in\ C_k \Leftrightarrow T_j \in C_k\}, \tag{5}$$

$$CL = \{(i,j) | \forall\ k \in \{1, \ldots, K\}, \neg(T_i \in\ C_k \wedge T_j \in C_k)\}. \tag{6}$$

3.2 Objective Function

In order to achieve a guided constrained learning approach, a new objective function is introduced based on contrastive learning [5] that extends the loss function used in LDPS [13] to a semi-supervised setting. The loss between two time-series takes the form

$$\mathcal{L}(T_i, T_j) = \frac{1}{2} \left(DTW(T_i, T_j) - \beta Dist_{i,j}\right)^2 + \phi_{i,j}, \tag{7}$$

where $DTW(T_i, T_j)$ is the dynamic time warping similarity between time-series T_i and T_j, $Dist_{i,j} = ||\overline{T}_i - \overline{T}_j||_2$ is the similarity measure between T_i and T_j in the embedded space such that $||\cdot||_2$ is the L_2 norm, and β scales the time-series similarity (distance) in the embedded space to the corresponding DTW similarity. The term $\phi_{i,j}$ is inspired by the contrastive loss and is defined, such that

$$\phi_{i,j} = \begin{cases} \alpha Dist_{i,j}^2, & \text{if } (i,j) \in \text{ML}, \\ \gamma \max(0, w - Dist_{i,j})^2, & \text{if } (i,j) \in \text{CL}, \\ 0, & \text{otherwise}, \end{cases} \tag{8}$$

where α, γ are weights that regularise the must-link and cannot-link similarity distances respectively, and w is the minimum distance between samples for them to be considered well separated in the embedded space (after which, there is no influence on the loss) and is calculated using the following function $w = \max_{\forall i,\forall j}(DTW(T_i,T_j)) - \log(\frac{DTW(T_i,T_j)}{\max_{\forall i,\forall j}(DTW(T_i,T_j)}),$ such that $i \neq j$.

The overall loss function is therefore defined, such that

$$\mathcal{L}(\mathcal{T}) = \frac{2}{K(K-1)} \sum_{i=1}^{K} \sum_{j=i+1}^{K-1} \mathcal{L}(T_i, T_j). \tag{9}$$

3.3 CDPS algorithm

Algorithm 1 CDPS algorithm

Input: $\mathcal{T}$ a set of Time-series,
ML and CL constraint sets,
$L_{\min}$ minimum length of shapelets,
$S_{\max}$ maximum number of shapelet blocks,
n_{epochs}, s_{batch}, c_{batch}
Output: Set S of shapelets,
Embeddings of $\mathcal{T}$.

ShapeletBlocks ← GET_SHAPELET_BLOCKS($L_{\min}$, $S_{\max}$, L_i)
Shapelets ← INITIALIZE_SHAPELETS(ShapeletBlocks)
for $i \leftarrow 0$ to n_{epochs} **do**
  for 1 to $|\mathcal{T}|/s_{\text{batch}}$ **do**
    minibatch ← GET_BATCH($\mathcal{T}$, ML, CL, S_{batch}, C_{batch})
    Compute the DTW between the $T_{i's}$ and $T_{j's}$ in minibatch
    Update the Shapelets and β by descending the gradient $\nabla\mathcal{L}(T_i, T_j)$
Embeddings ← SHAPELET_TRANSFROM($\mathcal{T}$)

Algorithm 1 describes CDPS's approach to learning the representational embedding. In which ShapeletBlocks is a dictionary containing $S_{\max}$ pairs, {shapelet length; shapelet number}, where shapelet length is $L_{\min} \cdot b_{\text{ind}}$, $L_{\min}$ is the minimum shapelet length and $b_{\text{ind}} \in \{1, \ldots, S_{\max}\}$ is the index of the shapelet block. The number of shapelets for each block is calculated using the same approach as LDPS [13]: $10\log(L_i - L_{\min} \cdot b_{\text{ind}}) \times 10$. The parameter C_{batch} defines the number of constraints in each training batch, the aim of this parameter is to increase the importance of the constrained time-series in face of the large number of the unconstrained time-series. INITIALIZE_SHAPELETS initialises the shapelets either randomly or rule-based. Here the following rule-based approach is taken: (1) Shapelets are initialised by drawing a number of time series samples then reshaping them into sub-sequences with length equal to that of the shapelets; (2) k-means clustering is then performed on the sub-sequences and the cluster centers are extracted to form the initial shapelets. GET_BATCH generates

batches containing both constrained and unconstrained samples. If there are insufficient constraints to fulfil C_{batch} then they are repeated. For speed and to take advantage of GPU acceleration, the above algorithm can be implemented as a 1D convolutional neural network in which each layer represents a shapelet block composed of all the shapelets having the same length followed by max-pooling in order to obtain the embeddings. The derivation of the gradient of $\mathcal{L}(\mathcal{T})$, $\nabla\mathcal{L}(T_i, T_j)$ (Algorithm 1, Line 7), is given in the supplementary material.

4 Evaluation

In this section CDPS is evaluated with respect to different constraint sets under two cases: the classical constrained clustering setting in which clusters are extracted from a dataset, called transductive learning; and the second, which is normally not possible using classical constrained clustering algorithms, in which the constraints used to learn a representation are generalised to an unseen test set, called inductive learning.

4.1 Experimental Setup

Algorithm 1 is executed using mini-batch gradient descent with a batch size $s_{\text{batch}} = 64$, $c_{\text{batch}} = 16$ constraints in each batch for the transductive setting, while $s_{\text{batch}} = 32$, $c_{\text{batch}} = 8$ for the inductive setting (since there are fewer samples). The influence of α and γ on accuracy were evaluated and the algorithm was found to be stable to variations in most of the cases and for that reason the value for both is fixed to 2.5. The minimum shapelet length $L_{min} = 0.15 \cdot L_i$, and the maximum number of shapelets $S_{max} = 3$ are taken to be the same as used in LDPS [13]. All models are trained for 500 epochs using the Adam optimiser.

K-means and COP-KMeans [27] are used as comparison methods (unconstrained and constrained respectively) since k-means based algorithms are the most widely applied in real-world applications, offering state-of-the-art (or close to state-of-the-art) performance. CDPS is also compared to FeatTS [23], which is a semi-supervised algorithm that extracts features and uses k-Mediods clustering.

Thirty-five datasets [2] chosen randomly from the UCR repository [2] are used for evaluation. The number of clusters is set to the number of classes in each dataset. The Normalised Mutual Information (NMI), which measures the coherence between the true and predicted labels, is measured to evaluate the resulting clusters with 0 indicating no mutual information and 1 a perfect correlation.

For the first use case, termed Transductive, the training and test sets of the UCR datasets are combined, this reflects the real-world transductive case in which a dataset is to be explored and knowledge extracted. In the second, termed Inductive, the embedding is learnt on the training set and its generalised performance on the test set is evaluated. This inductive use-case is something that

[2] Details on the datasets used are provided in the supplementary material.

is not normally possible when evaluating constrained clustering algorithms since clustering is a transductive operation and this highlights one of the key contributions of CDPS - the ability generalise constraints to unseen data. The third use case, highlights the importance of CDPS shapelets as features and their general ability to be integrated into any downstream algorithm. As such, FeatTS's semi-supervised statistical features are replaced with the dataset's CDPS embedding.

Each algorithm's performance is evaluated on each dataset with increasing numbers of constraints, expressed in percentages of samples that are subject to a constraint in the 5%, 15%, 25%. These represent a very small fraction of the total number of possible constraints, which is $\frac{1}{2}N(N-1)$. Each clustering experiment is repeated 10 times, each with a different random constraint set, and each clustering algorithm is repeated 10 times for each constraint set (i.e. there are 100 repetitions for each percentage of constraints[3]). The constraints are generated by taking the ground truth data, randomly selecting two samples, and adding an ML or CL constraint depending on their class labels until the correct number of constraints are created.

In the FeatTS comparison, both the train and test sets were used (i.e. transductive). FeatTS and CDPS were evaluated using 25% of the ground truth information: FeatTS takes this information in the form of labels; while CDPS in the form of ML/CL constraints, CDPS embeddings are generated and replace FeatTS's features (CDPS+FeatTS). The number of features used for FeatTS was 20, as indicated in the author's paper. With both feature sets, k-Mediods was applied on the co-occurrence matrix to obtain the final clustering [23].

4.2 Results

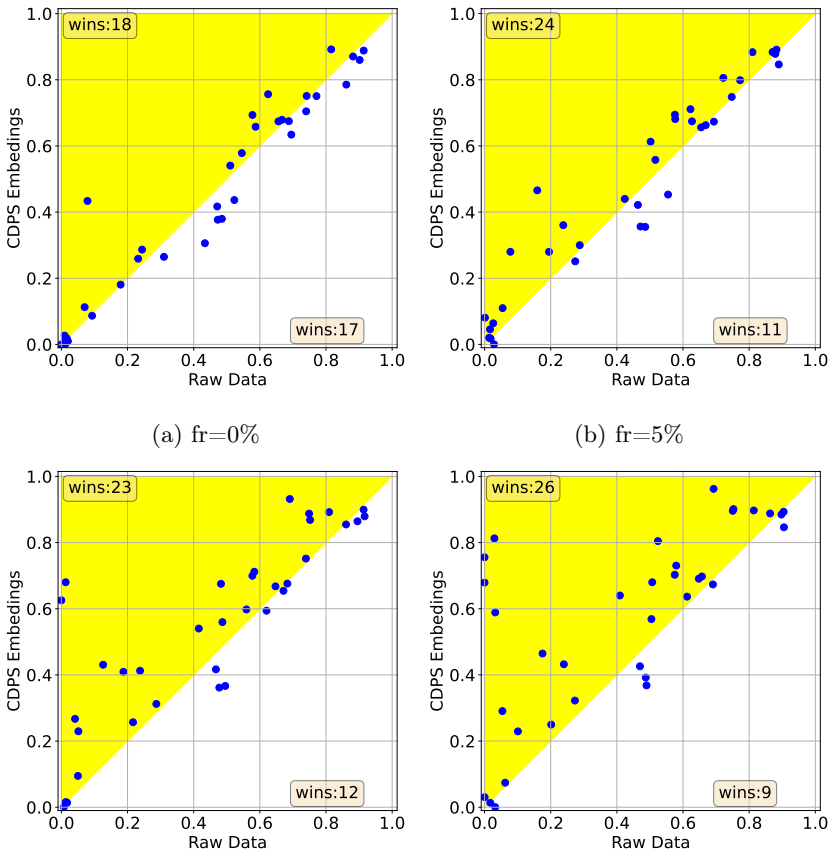
In this section the results of each use case (described in Section 4.1) are presented.

Transductive: Figure 1 shows the NMI scores for CDPS (Euclidean k-means performed on the CDPS embeddings) compared to k-means (on the raw time-series), COP-Kmeans (also on the raw time-series), and LDPS (Euclidean k-means on the LDPS embeddings). Unconstrained k-means and LDPS are presented as a reference for the constrained algorithms (COP-kmeans and CDPS respectively) to give insight into the benefit of constraints for each. It can be seen that overall LDPS and k-means offer similar performance.

It can also be seen that CDPS uses the information gained by constraints more efficiently, outperforming COP-Kmeans in almost all the different constraint fractions for most datasets.

It appears, nevertheless, that some datasets lend themselves to (unconstrained) k-means based algorithms since it outperforms LDPS. Nevertheless, CPDS exhibits an increase in performance as the number of constraints increase, whereas COP-Kmeans tends to stagnate. This can be seen as the cloud of points move

[3] Note that it is not always possible for COP-KMeans to converge on a result due to constraint violations, although many initialisation were tried to obtain as many results as possible some of the COP-KMeans results represent fewer repetitions.



(a) fr=0% (b) fr=5%

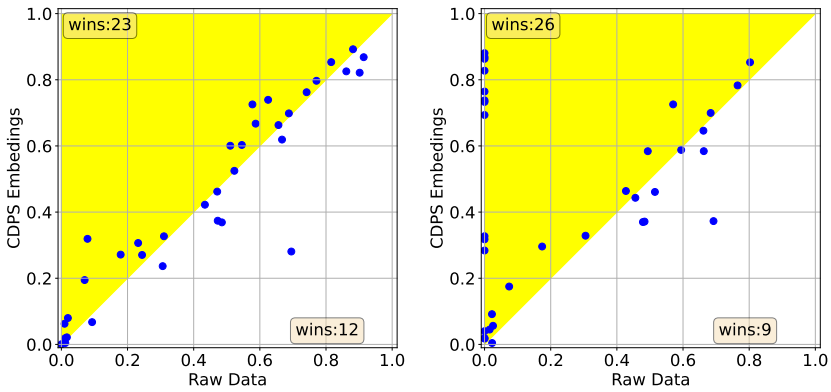
(c) fr=15% (d) fr=25%

Fig. 1: A Transductive comparison between CDPS+kmeans and Raw-TS+CopKmeans with different constraint fractions.

upwards (CDPS score increases) as more constraints are given. We can also observe that for some datasets the constrained algorithms behave similarly with 5% constraints, i.e. the cloud of points in the lower left corner, but again CDPS benefits most from increasing the number of constraints and significantly outperforms COP-KMeans with larger constraint percentages.

Inductive: Figure 2 presents the Inductive results, in which the embedding space is learnt on the training set and the generalisation performance evaluated on the unseen test set.

It should be noted that when training on the train set, there are significantly fewer constraints then when using the merged datasets for the same constraint percentage. It can therefore be concluded that even in the face of few data and



(a) fr=0% (b) fr=5%

(c) fr=15% (d) fr=25%

Fig. 2: An Inductive comparison between CDPS+kmeans and Raw-TS+CopKmeans with different constraint fractions.

constraints, CDPS is still able to learn a generalisable representation and attain (within a certain margin) the same clustering performance then when trained on the merged dataset. This is probably explained by the fact that having a smaller number of samples with few constraints means that they are repeated in the mini-batches (see Section 3.3), and this allows CDPS to focus on learning shapelets that are discriminative and preserve DTW rather than shapelets that model larger numbers of time series. Thus the resulting representation space is more faithful to the constraints, allowing better clustering of unseen time-series.

FeatTS comparison: This study investigates the significance of the shapelets learnt using CDPS as features over the semi-supervised statistical features extracted using FeatTS. Figure 3 shows the NMI scores of CDPS+FeatTS and

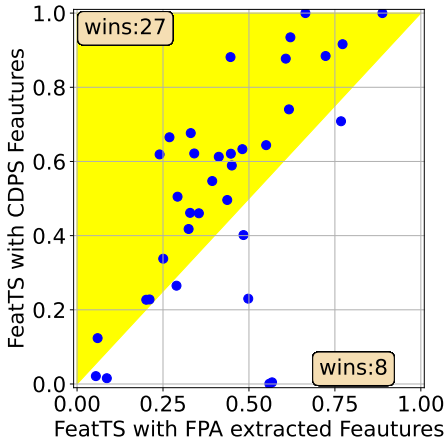


Fig. 3: A comparison between CDPS+FeatTS and FeatTS with respect to NMI score.

FeatTS, and we observe that, out of 35 datasets, CDPS+FeatTS outperforms FeatTS in 27. This indicates that the shapelets learnt using CDPS are better for clustering than the statistical features. For the datasets that achieved around zero NMI with respect to CDPS+FeatTS while high NMI with FeatTS (e.g. GunPointAgeSpan CDPS+FeatTS: 0.001, FeatTS: 0.559) it appears that the shapelets learnt in these cases are not discriminative enough, which is confirmed by CDPS' low scores (CDPS+Kmeans: 0.004).

5 Discussion

Since LDPS only models DTW distance, the comparisons between it and k-means (Figs. 2a & 1a) give approximately equal performance. Nevertheless, CDPS is better able to exploit the information contained in the constraints when they are introduced, giving more accurate clustering results overall. Both LDPS and CDPS result in a metric space, which is beneficial for further analysis and processing.

Being a hard constraint algorithm, COP-KMeans offers no guarantee of convergence, which was evident in the presented study where several of the results were missing after multiple tries. This is due to the difficulty of clustering with an elastic distance measure such as DTW. In these experiments, all constraints can be considered as coherent since they are generated from the ground truth data, however, in real-world situations this problem would be exacerbated by inconsistent constraints, particularly considering time-series since these are very hard to label. CDPS, does not suffer from such limitations.

Although it was included in this study in order to have a comparison method, using COP-KMeans in an inductive use-case is not usual practice for a classical

clustering algorithm. It was simulated by providing COP-KMeans with the combined 'training' dataset, its constraints, and the test data to be clustered. CDPS, on the other hand, offers a truly inductive approach in which new data can be projected into the resulting space, which inherently models user constraints. In this setting the difference between COP-KMeans is reduced, however, it should be noted that CDPS does not 'see' the training data during the inductive setup, whereas COP-KMeans used all the data to derive the clusters. This also exposes the infeasibility of using COP-KMeans in this way, the data needs to be stored, and accessed each time new data should be clustered, which will become computationally expensive as its size grows. Finally, CDPS's embedding can be used for tasks other than clustering (classification, generation, etc).

CDPS's inductive complexity (once the space has been determined) is $\mathcal{O}(NL_kK)$, plus kmeans' complexity $\mathcal{O}(NkKi)$, where k is the number of clusters, i the number of iterations until convergence, and the complexity of COP-KMeans is $\mathcal{O}(NkKi|\mathrm{ML} \cup \mathrm{CL}|)$.

Overall, the CDPS algorithm leads to better clustering results since it is able to exploit the information brought to the learning process by the constraints. Relatively, it can be seen that the number of datasets in which CDPS outperforms COP-KMeans increases in line with the number of constraints. In absolute terms, COP-KMeans' performance tends to decrease as more constraints are introduced, and the opposite can be said for CDPS. These constraints bias CDPS to find shapelets that define a representation that respects them while retaining the properties of DTW. Although the focus of this article is not to evaluate whether clustering on these datasets benefits from constraints, it can be observed that generally better performance is found when constraints are introduced.

The studies in the previous section show that the transformed space not only preserves the desirable properties of DTW but also implicitly models the constraints given during training. Although it was not evaluated, it is also possible to use COP-Kmeans (constrained) clustering in the Inductive CDPS embedding, thus allowing another mechanism to integrate constraints after the embedding has been learnt. Although CDPS has several parameters, it has been shown that these do not need to be fine-tuned for each dataset to achieve state-of-the-art performance (although better performance may be achieved if this is done).

5.1 Model Selection

When performing clustering there is no validation data with which to determine a stopping criteria. It is therefore important to analyse the behaviour of CDPS during training to give some general recommendations.

Figure 4 presents the CDPS clustering quality (NMI) as a function of the number of epochs for each dataset (using 30% constraints). It demonstrates that generally most of the models converge within a small number of epochs, with FaceFour taking the most epochs to converge. Moreover, the quality of the learnt representation does not deteriorate as the number of epochs increases, i.e. neither the DTW preserving aspect nor the constraint influence dominate the loss and diminish the other as epochs increase.

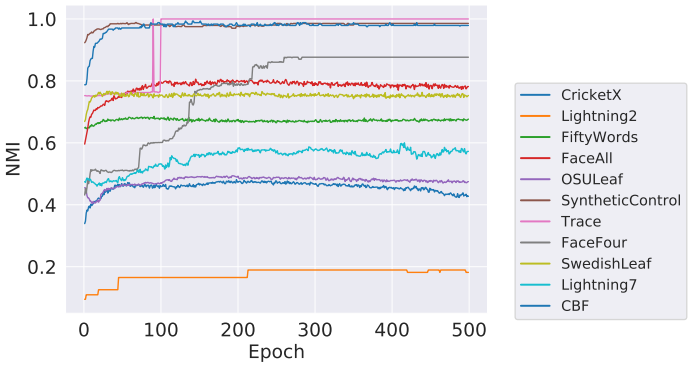
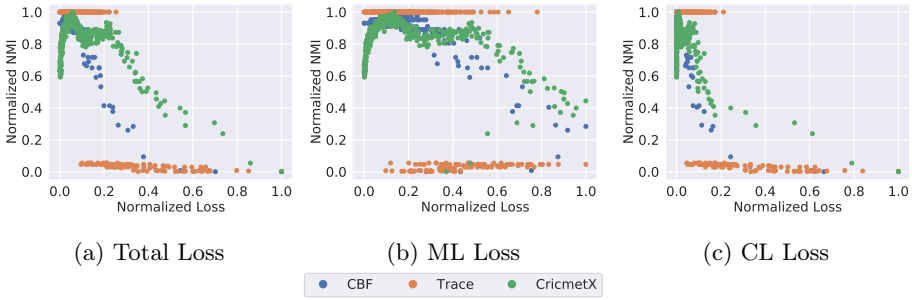


Fig. 4: Clustering quality (NMI) as a function of the number of epochs for each dataset, using a constraint fraction of 30%.



(a) Total Loss (b) ML Loss (c) CL Loss

Fig. 5: Relationship between NMI and CDPS Loss for each dataset. To highlight the relationship between datasets, both loss and NMI have been scaled to between 0 and 1.

Figure 5 presents scatter-plots of the NMI and CDPS loss (both normalised to between 0 and 1) for several datasets. In addition to the total loss, both the ML and CL losses have been included. The general trend observed in the overall loss is that a lower loss equates to a higher NMI.

These show that the loss can be used as a model selection criterion without any additional knowledge of the dataset. For practical application, the embedding can be trained for a fixed large enough number of epochs (as done in this study) or until stability is achieved. This is in line with the typical manner in which clustering algorithms are applied.

6 Conclusions

This article has presented CDPS, an approach for learning shapelet based time-series representations that respect user constraints while also respecting the

DTW similarity of the raw time-series. The constraints take the form of must-link and cannot-link pairs of samples provided by the user. The influence of the constraints on the learning process is ensured through the use of mini-batch gradient descent in which a fraction of each batch contains samples under constraint. The resulting space removes many limitations inherent with using the DTW similarity measure for time-series, particularly interpretability, constraint analysis, and the analysis of sample density. CDPS therefore paves the way for new developments in constraint proposition and incremental (active) learning for time-series clustering. The representations learnt by CDPS are general purpose and can be used with any machine learning task. The presented study focused on its use in constrained clustering. By evaluating the proposed method on thirty-five public datasets, it was found that using unconstrained k-means on CPDS representations outperforms COP-Kmeans, unconstrained k-means (on the original time-series), and LDPS with k-means. Also, CDPS is shown to outperform FeatTS that uses statistical features. It was also shown that the representation learnt by CDPS is generalisable, something that is not possible with classic constrained clustering algorithms and when applied to unseen data, CDPS outperforms COP-KMeans.

References

1. Cai, B., Huang, G., Xiang, Y., Angelova, M., Guo, L., Chi, C.H.: Multi-scale shapelets discovery for time-series classification. Int. J. Inf. Technol. Decis. Mak **19**(03), 721–739 (2020)
2. Dau, H.A., et al.: The UCR time series classification archive (October 2018), `https://www.cs.ucr.edu/~eamonn/time_series_data_2018/`
3. Davidson, I., Ravi, S.: Identifying and generating easy sets of constraints for clustering. In: AAAI. pp. 336–341 (2006)
4. Grabocka, J., Schilling, N., Wistuba, M., Schmidt-Thieme, L.: Learning time-series shapelets. In: SIGKDD. pp. 392–401 (2014)
5. Hadsell, R., Chopra, S., LeCun, Y.: Dimensionality reduction by learning an invariant mapping. In: CVPR. vol. 2, pp. 1735–1742 (2006)
6. Hills, J., Lines, J., Baranauskas, E., Mapp, J., Bagnall, A.: Classification of time series by shapelet transformation. Data Min Knowl Discov **28**(4), 851–881 (2014)
7. Keogh, E., Lonardi, S., Ratanamahatana, C.A.: Towards parameter-free data mining. In: SIGKDD. pp. 206–215 (2004)
8. Lampert, T., Lafabregue, B., Serrette, N., Forestier, G., Crémilleux, B., Vrain, C., Gancarski, P., et al.: Constrained distance based clustering for time-series: a comparative and experimental study. Data Min Knowl Discov **32**(6), 1663–1707 (2018)
9. Lei, Q., Yi, J., Vaculin, R., Wu, L., Dhillon, I.S.: Similarity preserving representation learning for time series clustering. In: IJCAI. pp. 2845–2851 (2017)
10. Levenshtein, V.: Binary codes capable of correcting deletions, insertions, and reversals. Soviet Physics Doklady **10**(8), 707–710 (1966)
11. Li, Z., Liu, J., Tang, X.: Constrained clustering via spectral regularization. In: CVPR. pp. 421–428. IEEE (2009)
12. Lines, J., Davis, L.M., Hills, J., Bagnall, A.: A shapelet transform for time series classification. In: SIGKDD. pp. 289–297 (2012)

13. Lods, A., Malinowski, S., Tavenard, R., Amsaleg, L.: Learning DTW-preserving shapelets. In: IDA (2017)
14. Mueen, A., Keogh, E., Young, N.: Logical-shapelets: an expressive primitive for time series classification. In: SIGKDD. pp. 1154–1162 (2011)
15. Paparrizos, J., Franklin, M.J.: Grail: efficient time-series representation learning. VLDB Endowment **12**(11), 1762–1777 (2019)
16. Paparrizos, J., Gravano, L.: k-shape: Efficient and accurate clustering of time series. In: SIGMOD. pp. 1855–1870 (2015)
17. Paparrizos, J., Liu, C., Elmore, A.J., Franklin, M.J.: Debunking four long-standing misconceptions of time-series distance measures. In: ACM SIGMOD. pp. 1887–1905 (2020)
18. Rakthanmanon, T., Keogh, E.: Fast shapelets: A scalable algorithm for discovering time series shapelets. In: SDM. pp. 668–676 (2013)
19. Sakoe, H., Chiba, S.: Dynamic-programming approach to continuous speech recognition. In: ICA. pp. 65–69 (1971)
20. Sakoe, H., Chiba, S.: Dynamic programming algorithm optimization for spoken word recognition. IEEE Tans Acoust, Speech, Signal Process **26**(1), 43–49 (1978)
21. Shah, M., Grabocka, J., Schilling, N., Wistuba, M., Schmidt-Thieme, L.: Learning DTW-shapelets for time-series classification. In: ACM IKDD CODS. pp. 1–8 (2016)
22. Sperandio, R.C.: Recherche de séries temporelles à l'aide de DTW-preserving shapelets. Ph.D. thesis, Université Rennes 1 (2019)
23. Tiano, D., Bonifati, A., Ng, R.: Feature-driven time series clustering. In: EDBT. pp. 349–354 (2021)
24. Ulanova, L., Begum, N., Keogh, E.: Scalable clustering of time series with u-shapelets. In: SDM. pp. 900–908 (2015)
25. Vlachos, M., Hadjieleftheriou, M., Gunopulos, D., Keogh, E.: Indexing multidimensional time-series. The VLDB Journal **15**(1), 1–20 (2006)
26. Wagstaff, K., Basu, S., Davidson, I.: When is constrained clustering beneficial, and why? In: IAAI (2006)
27. Wagstaff, K., Cardie, C., Rogers, S., Schrödl, S.: Constrained k-means clustering with background knowledge. In: ICML. vol. 1, pp. 577–584 (2001)
28. Wu, L., Yen, I.E.H., Yi, J., Xu, F., Lei, Q., Witbrock, M.: Random warping series: A random features method for time-series embedding. In: AISTATS. pp. 793–802 (2018)
29. Yamaguchi, A., Maya, S., Maruchi, K., Ueno, K.: Ltspauc: Learning time-series shapelets for optimizing partial auc. In: SDM. pp. 1–9 (2020)
30. Ye, L., Keogh, E.: Time series shapelets: a new primitive for data mining. In: SIGKDD. pp. 947–956 (2009)
31. Ye, L., Keogh, E.: Time series shapelets: a novel technique that allows accurate, interpretable and fast classification. Data Min Knowl Discov **22**(1), 149–182 (2011)
32. Zakaria, J., Mueen, A., Keogh, E.: Clustering time series using unsupervised-shapelets. In: ICDM. pp. 785–794 (2012)
33. Zakaria, J., Mueen, A., Keogh, E., Young, N.: Accelerating the discovery of unsupervised-shapelets. Data Min Knowl Discov **30**(1), 243–281 (2016)
34. Zhang, Q., Wu, J., Yang, H., Tian, Y., Zhang, C.: Unsupervised feature learning from time series. In: IJCAI. pp. 2322–2328 (2016)
35. Zheng, G., Yang, Y., Carbonell, J.: Efficient shift-invariant dictionary learning. In: ACM SIGKDD. pp. 2095–2104 (2016)

Supplementary Material
CDPS: Constrained DTW-Preserving Shapelets

Hussein El Amouri ✉[1], Thomas Lampert ✉[1], Pierre Gançarski ✉[1], and Clément Mallet ✉[2]

[1] ICube, University of Strasbourg, France.
{helamouri,lampert,gancarski}@unistra.fr
[2] Univ Gustave Eiffel, IGN, ENSG, Saint-Mande, France.
clement.mallet@ign.fr

1 Derivation of the CDPS Loss Gradient

This section presents the derivation of the loss function's gradient. Let

$$Dist_{i,j} = ||\overline{T}_i - \overline{T}_j||_2,$$

$$\mathrm{L}(\mathrm{T}_i, T_j) = \tfrac{1}{2}\psi + \phi_{i,j}, \text{where,}$$

$$\psi = \frac{1}{2}\left(DTW(T_i, T_j) - \beta Dist_{i,j}\right)^2,$$

and

$$\phi_{i,j} = \begin{cases} \alpha Dist_{i,j}^2, & \text{if } (i,j) \in ML, \\ \gamma \max(0, w - Dist_{i,j})^2, & \text{if } (i,j) \in CL, \\ 0, & \text{otherwise,} \end{cases}$$

where w is a predefined constant.

Derivation with Respect to β

$$\frac{\partial \mathcal{L}(T_i, T_j)}{\partial \beta} = \frac{1}{2}\frac{\partial \psi}{\partial \beta} + \frac{\partial \phi}{\partial \beta} = \frac{1}{2}\frac{\partial \psi}{\partial \beta} = -\beta[DTW(T_i, T_j) - \beta Dist_{i,j}].$$

Derivation with Respect to the Shapelets

$$\frac{\partial \mathcal{L}(T_i, T_j)}{\partial S_{k,l}} = \frac{1}{2}\frac{\partial \psi}{\partial S_{k,l}} + \frac{\partial \phi}{\partial S_{k,l}}.$$

The derivations of ψ and ϕ with respect to the shapelets $S_{k,l}$ will be presented separately.

Using the chain rule, the derivation with respect to ψ can be written as such that

$$\frac{\partial \psi}{\partial S_{k,l}} = \frac{\partial \psi}{\partial Dist_{i,j}} \frac{\partial Dist_{i,j}}{\partial \Delta_{i,j,k}} \frac{\partial \Delta_{i,j,k}}{\partial S_{k,l}},$$

where $\Delta_{i,j,k} = \overline{T}_{i,k} - \overline{T}_{j,k}$. The derivation of each term is straight-forward:

$$\frac{\partial \psi}{\partial Dist_{i,j}} = -2\beta(DTW(T_i, T_j) - \beta Dist_{i,j}),$$

$$\frac{\partial Dist_{i,j}}{\partial \Delta_{i,j,k}} = \frac{\Delta_{i,j,k}}{Dist_{i,j}}, \quad \text{where } Dist_{i,j} \neq 0,$$

and

$$\frac{\partial \Delta_{i,j,k}}{\partial S_{k,l}} = \frac{\partial \overline{T}_{i,k}}{\partial S_{k,l}} - \frac{\partial \overline{T}_{j,k}}{\partial S_{k,l}},$$

where

$$\frac{\partial \overline{T}_{i,k}}{\partial S_{k,l}} = \frac{\partial \min(D_{i,k,m})}{\partial S_{k,l}} = \sum_m \frac{\partial \overline{T}_{i,k}}{\partial D_{i,k,m}} \frac{\partial D_{i,k,m}}{\partial S_{k,l}}.$$

Following the approximation used in LDPS [**?**] which gives $\frac{\partial \overline{T}_{i,k}}{\partial D_{i,k,m}} = \delta_{m,m^*}$ the above can be written as:

$$\frac{\partial \overline{T}_{i,k}}{\partial S_{k,l}} = \sum_m \delta_{m,m^*} \frac{D_{i,k,m}}{\partial S_{k,l}},$$

$$\frac{\partial \phi}{\partial S_{k,l}} = \frac{\partial \phi}{\partial Dist_{i,j}} \frac{\partial Dist_{i,j}}{\partial \Delta_{i,j,k}} \frac{\partial \Delta_{i,j,k}}{\partial S_{k,l}},$$

where

$$\frac{\partial \phi}{\partial Dist_{i,j}} = \begin{cases} 2\alpha Dist_{i,j}, & \text{if } (i,j) \in ML, \\ -2\gamma(w - Dist_{i,j}), & \text{if } (i,j) \in CL, \\ 0, & \text{otherwise,} \end{cases}$$

where w is a predefined constant.

2 Dataset

Dataset	Train size	Test size	Length	No. of Classes
ShapesAll	600	600	512	60
OSULeaf	200	242	427	6
Coffee	28	28	286	2
ScreenType	375	375	720	3
Phoneme	214	1896	1024	39
CBF	30	900	128	3
Rock	20	50	2844	4
CricketZ	390	390	300	12
Lightning7	70	73	319	7
Mallat	55	2345	1024	8
Car	60	60	577	4
CricketX	390	390	300	12
MoteStrain	20	1252	84	2
Fungi	18	186	201	18
Symbols	25	995	398	6
GunPointAgeSpan	135	316	150	2
Lightning2	60	61	637	2
SwedishLeaf	500	625	128	15
ECG200	100	100	96	2
Adiac	390	391	176	37
Fish	175	175	463	7
BirdChicken	20	20	512	2
PowerCons	180	180	144	2
BME	30	150	128	3
Plane	105	105	144	7
Meat	60	60	448	3
SyntheticControl	300	300	60	6
Beef	30	30	470	5
FacesUCR	200	2050	131	14
GunPoint	50	150	150	2
FaceFour	24	88	350	4
Herring	64	64	512	2
FiftyWords	450	455	270	50
FaceAll	560	1690	131	14
CricketY	390	390	300	12

Table 1: List of UCR datasets used in the study.