CDPS: CONSTRAINED DTW-PRESERVING SHAPELETS

Anonymous authors
Paper under double-blind review

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. Euclidean distance is unsuitable because of potential phase shift, differences in sample duration, and compression and dilation of characteristic signals. 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 DTW-preserving shapelets have been developed to regain these properties. This unsupervised approach to representation learning models DTW properties through the shapelet transform. 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 fulfills 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. The proposed algorithm is studied on multiple time-series datasets, and its advantages over classical constrained clustering algorithms and unsupervised DTW-preserving shapelets demonstrated. An open-source implementation based on PyTorch is available to take full advantage of GPU acceleration.

1 INTRODUCTION

Time series are produced in different domains such as finance, weather forecasting, health, remote sensing etc. The volume of time series data is increasing with the development of IoT, smart handheld devices, and personal health devices, etc. This expanse of data increases the difficulty to provide a ground truth labeling due to the ever increasing time and cost needed. Time-series are relatively hard to interpret, compared to images which is a data form that is natural to us. Labelling difficulty is exacerbated when making exploratory analyses and when working in nascent domains for which classes are not well defined. For that reason, 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. 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 is 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. A must-link constraint tells the algorithm that two points should be contained within the same cluster, and a cannot-link the contrary. In this way the algorithm is guided to converge on a result that is meaningful to the user without explicitly nor exhaustively labelling samples. Note that there is no notion of class, these constraints do not define what a sample represents, they only label pairs of samples as being the same or not.

Generally, time series are characterised by trend, shapes, distortions either to time or shape (Sperandio, 2019) 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, for example compression-based measures (Keogh et al., 2004), Levenshtein Distance (Levenshtein, 1966), Longest Common Subsequnce (Vlachos et al., 2006) and Dynamic Time Warping (DTW) (Sakoe & Chiba, 1971; 1978). DTW is the most popular since it overcomes these problems by aligning two series through the computation of a cost function based on Euclidean distance (Lampert et al., 2018). Time series also exhibit complex structure which are often highly correlated (Sperandio, 2019). This makes their analysis difficult to achieve and time consuming, indeed several attempts to accelerate DTW's computation have been proposed (Sperandio, 2019; Cai et al., 2020). A simpler approach to increase the accuracy of time-series classification was introduced by Ye & Keogh (2009), called Shapelets. These 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, Shapelets were extended to DTW-preserving shapelets to regain these properties. This unsupervised approach to representation learning models DTW properties through the shapelet transform.

The contribution of this article is to introduce constrained DTW-preserving shapelets (CDPS), in which a time-series representation is influenced by a limited amount of user knowledge (must link and cannot link constraints) to better capture the user's interpretation of the data rather than 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 fulfills the constraints (without explicit knowledge of them). The proposed embedding process is studied in a constrained clustering setting, on multiple datasets, and its advantages over COP-KMeans (Wagstaff et al., 2001) and unsupervised DTW-preserving shapelets demonstrated (Lods et al., 2017).

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 (Lampert et al., 2018). This opens up the possibility of measuring constraint informativeness (Davidson & Ravi, 2006) and constraint consistency (Wagstaff et al., 2006) in time-series clustering. 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 the literature on shapelets is reviewed, in Section 3 the Constrained DTW-Preserving Shapelets (CDPS) algorithm is presented with definitions and notations, in Section 4 CDPS is evaluated in comparison to constrained and unconstrained approaches and the results are discussed, and finally in Section 5 the conclusions are drawn.

2 Related Work

Shapelets were originally defined as a method to extract subsequences of time-series that are discovered such that they discriminate between the time-series using a tree based classifier (Ye & Keogh, 2009; 2011). As such, the shapelets themselves were chosen from a set of candidate shapelets, which is exhaustive and contains all possible sub-sequences of the times series in the dataset. Rakthanmanon & Keogh (2013) propose to first project the time-series into a symbolic representation to increase the speed of discovering the shapelets. Subsequently, Mueen et al. (2011) introduce logical shapelets, which combines shapelets with complex rules of discrimination to increase the reliability of the discovered shapelets and their ability to discriminate between the time-series. Sperandio (2019) present a detailed review of early shapelet approaches.

Lines et al. (2012) proposed a new way of handling shapelets that separated classification from the transformation, which was later extended by Hills et al. (2014). The authors introduce the concept of the shapelet transform that aims to transform the raw data into a vectorial representation, where the shapelets define the bases of the representation space. The authors showed that this separation leads to stronger and more accurate classification results even with non-tree based approaches.

In order to overcome the exhaustive search for optimal shapelets, Grabocka et al. (2014) introduce the concept of learning shapelets. 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 approaches. Shah et al. (2016) 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 (Yamaguchi et al., 2020), in which shapelets are learnt in conjunction with a classifier for pAUC optimisation.

The approaches discussed this far have been supervised. Zakaria et al. (2012) introduced the first approach for clustering time-series with shapelets, called unsupervised-shapelets or u-shapelets. U-shapelets best partition a subset of the time series from the rest of the data set, which is repeated until no further improvements can be made. As such, this method suffers from the exhaustive search strategy as seen with early supervised approaches. U-shapelets have been used in several works since their initial introduction (Ulanova et al., 2015; Zakaria et al., 2016). Since these unsupervised methods take a similar approach to the original supervised shapelets, they have the same drawbacks. To overcome these, Zhang et al. (2016) propose to combine learning shapelets with unsupervised feature selection methods to auto-learn the optimal shapelets.

All these approaches learn to optimally discriminate time-series, either in a supervised or unsupervised manner. Learning DTW-preserving shapelets (LDPS) expands the learning paradigm for shapelets by integrating additional constraints on the learnt representation. In LDPS these constrain the representational space to model the DTW distances between the time-series. The learning process learns shapelets that form this space and transform a time-series into a high-dimensional Euclidean space in which DTW properties are preserved.

3 Constrained DTW-Preserving Shapelets

The methods discussed in the previous section fall under two categories of learning: supervised and unsupervised. This section proposes Constrained DTW-Preserving Shapelets (CDPS), which learns shapelets in a semi-supervised manner. Therefore allowing expert knowledge to influence the transformation learning process, while also preserving DTW properties and the interpretability of shapelets. The necessary preliminaries are presented in Section 3.1, CDPS's cost function in Section 3.2, and the overall algorithm in Section 3.3.

3.1 Definitions and Notations

Here the definitions and notations for time-series, shapelets and shapelet transform that will be used throughout this article are presented.

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

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

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, or equal, to that of the shortest time series in the dataset. Let a Shapelet be denoted as S having length L_S. Let $\mathcal{S} = \{S_1, .., S_K\}$ be a set of K shapelets, where $S_k = S_{j,1:L_S}$. 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.

Euclidean score: is the similarity score between a shapelet S_k and a time series subsequence $T_{i,m:L_S}$, 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

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

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

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

Constraint Sets: this work focuses on instance level constraints, which specify that two samples are the same using a Must-Link (ML) constraint or are different using a Cannot-Link (CL) constraint. Taking two time-series instances T_i and T_j, if they are linked with an ML constraint then they must be in the same cluster $\forall\ k \in \{1, \ldots, K\}, T_i \in\ C_k \Leftrightarrow T_j \in C_k$, where K is the number of clusters and C_k is the assigned cluster, and a CL constraint states that they cannot be in the same cluster, i.e. $\forall\ k \in \{1, \ldots, K\}, \neg(T_i \in\ C_k \wedge T_j \in C_k)$.

3.2 Objective Function

In order to achieve a guided constrained learning approach, a new objective function is introduced based on contrastive learning (Hadsell et al., 2006) that extends the loss function used in LDPS (Lods et al., 2017) to also preserve DTW properties in the transformed space.

The loss between two time-series takes the form

$$\mathcal{L}(T_i, T_j) = \frac{1}{2}\left(DTW(T_i, T_j) - \beta||\overline{T}_i - \overline{T}_j||_2\right)^2 + \phi_{i,j}, \tag{5}$$

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

$$\phi_{i,j} = \begin{cases} \alpha Dist_{i,j}^2, & \text{if } (i,j) \in ML, \\ \gamma \max(w, Dist_{i,j})^2, & \text{if } (i,j) \in CL, \\ 0, & \text{otherwise}, \end{cases} \tag{6}$$

where α, γ are the weights of the must-link and cannot-link constraints respectively, and where w is the minimum distance between samples for them to be considered well separated in the embedded space. The overall loss function is therefore defined such that

$$\mathcal{L}(\mathcal{T}) = \frac{2}{K(K-1)} \sum_{i=1}^{K} \sum_{j=i+1}^{K-1} \mathcal{L}(T_i, T_j). \tag{7}$$

The derivation of the gradient of $\mathcal{L}(\mathcal{T})$, $\nabla\mathcal{L}(T_i, T_j)$, is given in Appendix A.

3.3 Learning Process

Algorithm 1 CDPS algorithm

```
Input: $\mathcal{T}$, ML, CL, ShapeletBlocks, $n_{\text{epochs}}$, $s_{\text{batch}}$, $c_{\text{batch}}$
Output: Shapelets (the learnt shapelets), Embeddings (the new time-series representation)
Shapelets ← INITSHAPELETS(ShapeletBlocks)
for $i \leftarrow 0$ to $n_{\text{epochs}}$ do
    for $1$ to $|\mathcal{T}|/s_{\text{batch}}$ do
        minibatch ← BATCHSET($\mathcal{T}$, ML, CL, $S_{\text{batch}}$, $C_{\text{batch}}$)
        Update Shapelets by descending the gradient $\nabla\mathcal{L}(T_i, T_j)$
Embeddings ← SHAPELETTRANSFROM($\mathcal{T}$)
```

Algorithm 1 defines CDPS's approach to learning the representational embedding. In which ShapeletBlock is a dictionary of size S_{max} containing $\{$shapelet length; shapelet number$\}$ pairs, where shapelet length is $L_{\min} \cdot b_{\text{ind}}$, $b_{\text{ind}} \in \{1, \ldots, S_{\max}\}$, and $L_{\min}$ is the minimum shapelet length. The number of shapelets for each scale is calculated using the same approach as LDPS (Lods et al., 2017): $10 \log(L_{TS} - L_T)$. C_{batch} defines the number of constraints in each batch during training, 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. INITSHAPELETS initialises the shapelets, which can be random or rule-based. In CDPS, shapelets are initialised by extracting all shapelet length subsequences from the time-series and applying k-means clustering. The cluster centres are the initial shapelets (therefore the number of clusters equals the number of shapelets). BATCHSET takes the constraints set, the dataset and the percentage of constraints to be included to generate the batch having both constrained and unconstrained samples. If there are insufficient constraints to fulfill C_{batch} then they are repeated.

The parameters subject to optimisation are the scale parameter β and the shapelets themselves, see Appendix A. 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 maxpooling in order to obtain the embeddings.

Finally, clusters can be found in the embedding returned by Algorithm 1 using k-means clustering.

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; 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.

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, $\alpha = 2$, $\gamma = 5$. Different values of α and γ were evaluated in preliminary experiments (on different datasets) and the algorithm was found to be stable to variations. The minimum shapelet length $L_{min} = 0.15 \cdot L_{TS}$, and the maximum number of shapelets $S_{max} = 3$ are taken to be the same as used in LDPS (Lods et al., 2017). All models are trained for 500 epochs using the Adam optimiser.

K-means and COP-KMeans (Wagstaff et al., 2001) 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. Eleven datasets from the UCR repository (Dau et al., 2018) are used for evaluation (the same as used by the authors of LDPS (Lods et al., 2017)) and are detailed in Table 1. 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 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.

CDPS'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 dataset: 5%, 10%, 15%, 20%, 25%, 30%. Each 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). 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. This is repeated until the correct number of constraints are collected.

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

Dataset	Train size	Test size	Length	No. of Classes
CBF	30	900	128	3
CricketX	390	390	300	12
FaceFour	24	88	350	4
FaceAll	560	1690	131	14
FiftyWords	450	455	270	50
Lightning2	60	61	637	2
Lightning7	70	73	319	7
OSULeaf	200	242	427	6
SwedishLeaf	500	625	128	15
SyntheticContorl	300	300	60	6
Trace	100	100	275	4

4.2 Results and Discussion

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

Transductive: In which each dataset's train and test sets are combined and used for clustering, i.e. classic (transductive) clustering. Table 2 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 LDPS outperforms or ties with k-means in almost all cases (except in the CricketX dataset).

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 (with the exception of FiftyWords, Lightning2, and CricketX). It appears that CricketX lends itself to k-means based algorithms since, in the unconstrained setting, k-means outperforms LDPS. Nevertheless, CPDS exhibits an increase in performance as the number of constraints increase, whereas COP-Kmeans tends to stagnate. For FiftyWords, the performance is almost tied between the unconstrained k-means and LDPS which is also reflected in the constrained version of the algorithms where only a slight variation can be observed. In FaceFour the constrained algorithms behave similarly with 5% constraints but again CDPS benefits most from increasing the number of constraints and significantly outperforms COP-KMeans with larger constraint percentages.

Thus overall, the CDPS algorithm leads to better clustering results since it is able to better exploit the information brought to the learning process by the constraints. These bias CDPS to find shapelets that define a representation that respects the constraints 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.

Inductive: In which the embedding space is learnt on the training set and the generalisation performance evaluated on the unseen test set. Table 3 presents the generalised NMI scores. Comparing these to Table 2 (the transductive results) reveals that CDPS efficiently generalises constraints and quite often demonstrates an increase in NMI.

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 (since the training sets are significantly smaller, see Table 1). It can therefore be concluded that even in the face of few data and 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 DTW preserving rather than trying to find shapelets that

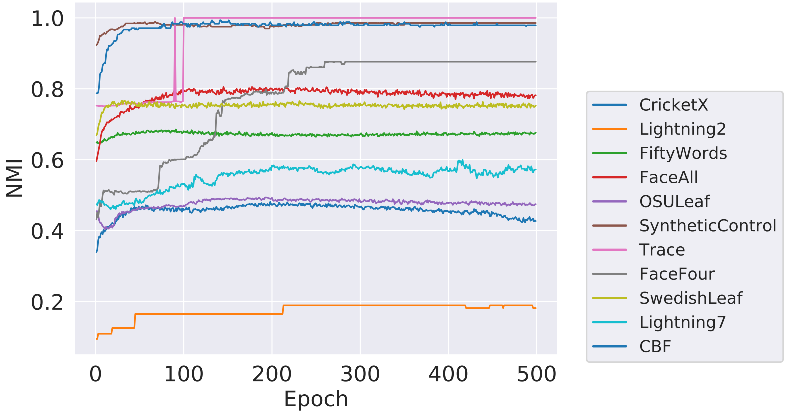


Figure 1: Clustering quality (NMI) as a function of the number of epochs for each dataset, using a constraint fraction of 30%.

model larger numbers of time series. Thus the resulting representation space is more faithful to the constraints, allowing better clustering of unseen time-series.

These two studies 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).

4.2.1 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.

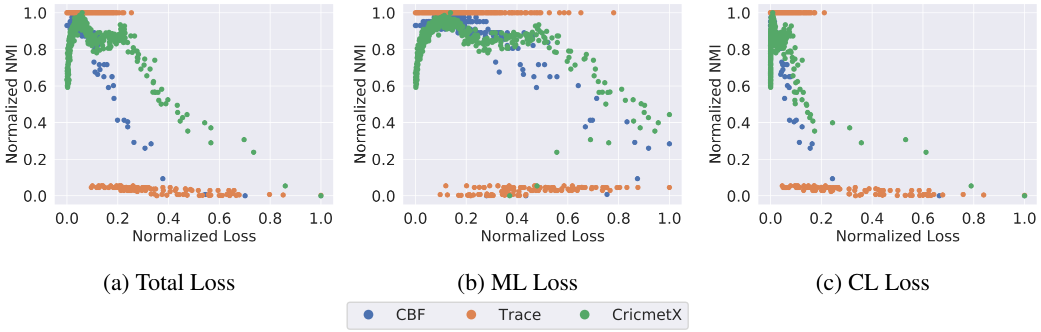
Figure 1 presents the CDPS clustering quality (NMI) as a function of the number of epochs for each dataset (using 30% constraints). It demonstrates that generally most of the models converge within a small number of epochs, with FaceFour taking the most epochs to converge. Moreover, the quality of the 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.

Figure 2 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.

5 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.



(a) Total Loss

(b) ML Loss

(c) CL Loss

Figure 2: 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.

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 11 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. It was also shown that the representation learnt by CDPS is generalisable, something that is not possible with classic constrained clustering algorithms. When applied to unseen data, CDPS outperforms COP-KMeans even when the latter has been explicitly trained with constraints defined on the test dataset (while CDPS generalises those from the training set).

Potential future directions of research are to improve the interpretability and discriminative property of the shapelets learnt by CDPS. Therefore providing an explanation and interpretation of the resulting clusters.

Table 2: NMI scores for transductive clustering. Presented are the averages of 10 repetitions applied to 10 different constraint sets, i.e. 100 repetitions in total. Bold indicates the highest NMI score in each dataset and constraint percentage.

			Constrained											
	Unconstrained		**5%**		**10%**		**15%**		**20%**		**25%**		**30%**	
Datasets	kmeans	LDPS	COPKmeans	CDPS	COPKmeans	CDPS	COPKmeans	CDPS	COPKmeans	CDPS	COPKmeans	CDPS	COPKmeans	CDPS
CBF	0.769	0.784	0.768	**0.821**	0.757	**0.878**	0.753	**0.994**	0.757	**0.993**	0.753	**0.997**	0.744	**0.999**
CricketX	**0.445**	0.392	**0.481**	0.37	**0.481**	0.39	**0.481**	0.402	**0.479**	0.412	**0.479**	0.424	**0.474**	0.412
FaceAll	0.67	**0.705**	0.661	0.665	**0.659**	0.649	0.66	**0.681**	0.651	**0.72**	0.653	**0.733**	0.657	**0.759**
FaceFour	0.598	**0.629**	0.531	0.516	0.568	0.579	0.547	**0.662**	0.524	**0.633**	0.495	**0.75**	0.54	**0.784**
FiftyWords	0.67	0.68	0.688	0.693	0.688	0.687	0.687	0.688	0.687	0.675	0.69	0.677	0.689	0.662
Lightning2	0.077	**0.105**	0.052	**0.127**	0.053	**0.11**	0.049	**0.082**	**0.067**	0.055	0.044	**0.06**	0.044	**0.073**
Lightning7	0.491	**0.567**	0.501	**0.559**	0.505	**0.565**	0.5	**0.563**	0.505	**0.551**	0.498	**0.567**	0.488	**0.586**
OSULeaf	0.24	0.254	0.245	**0.407**	0.237	**0.404**	0.242	**0.44**	0.238	**0.438**	0.236	**0.441**	0.24	**0.462**
SwedishLeaf	0.575	**0.658**	0.581	**0.701**	0.578	**0.707**	0.577	**0.729**	0.575	**0.739**	0.574	**0.744**	0.572	**0.751**
SyntheticControl	0.883	**0.965**	0.893	**0.963**	0.892	**0.945**	0.892	**0.951**	0.902	**0.956**	0.919	**0.974**	0.916	**0.972**
Trace	0.751	0.751	0.733	**0.898**	0.751	**0.975**	0.752	**0.981**	0.75	**1.0**	0.743	**1.0**	0.746	**1.0**

Table 3: NMI scores for inductive learning in which embeddings are learnt on the training set and generalisation evaluated on the test set. Presented are the averages of 10 repetitions applied to 10 different constraint sets, i.e. 100 repetitions in total. Bold indicates the highest NMI score in each dataset and constraint percentage.

	Constrained											
	5%		**10%**		**15%**		**20%**		**25%**		**30%**	
Datasets	COPKmeans	CDPS	COPKmeans	CDPS	COPKmeans	CDPS	COPKmeans	CDPS	COPKmeans	CDPS	COPKmeans	CDPS
CBF	0.766	**0.795**	0.762	**0.806**	0.752	**0.815**	0.752	**0.819**	0.742	**0.815**	0.73	**0.814**
CricketX	**0.496**	0.376	**0.492**	0.398	**0.496**	0.39	**0.498**	0.397	**0.496**	0.422	**0.499**	0.443
FaceAll	**0.666**	0.638	0.665	0.654	0.664	0.643	0.665	0.658	0.663	0.686	0.659	0.664
FaceFour	0.574	**0.648**	0.586	**0.637**	0.596	0.596	0.568	**0.635**	0.583	**0.626**	0.551	**0.623**
FiftyWords	0.726	0.72	0.729	0.718	0.725	0.723	0.719	0.719	0.734	0.716	0.739	0.718
Lightning2	0.075	**0.2**	0.065	**0.178**	0.07	**0.131**	0.091	**0.121**	0.056	**0.164**	0.079	**0.161**
Lightning7	0.533	**0.589**	0.536	**0.587**	0.525	**0.586**	0.526	**0.582**	0.542	**0.576**	0.532	**0.574**
OSULeaf	0.24	**0.353**	0.239	**0.382**	0.238	**0.386**	0.235	**0.378**	0.233	**0.399**	0.228	**0.401**
SwedishLeaf	0.59	**0.69**	0.588	**0.714**	0.584	**0.721**	0.585	**0.719**	0.584	**0.736**	0.582	**0.747**
SyntheticControl	0.878	**0.955**	0.875	**0.959**	0.882	**0.962**	0.863	**0.957**	0.879	**0.952**	0.895	**0.964**
Trace	0.756	**0.824**	0.767	**0.835**	0.767	**0.81**	0.778	**0.865**	0.776	**0.85**	0.783	**0.883**

Acknowledgments

Removed to comply with anonymisation.

Reproducibility Statement

Code to recreate all experiments will be publicly released upon acceptance and the data used throughout is a public benchmark repository (Dau et al., 2018).

References

Borui Cai, Guangyan Huang, Yong Xiang, Maia Angelova, Limin Guo, and Chi-Hung Chi. Multi-scale shapelets discovery for time-series classification. *International Journal of Information Technology & Decision Making*, 19(03):721–739, 2020.

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. The ucr time series classification archive, October 2018. `https://www.cs.ucr.edu/~eamonn/time_series_data_2018/`.

Ian Davidson and S. Ravi. Identifying and generating easy sets of constraints for clustering. In *AAAI*, pp. 336–341, 2006.

Josif Grabocka, Nicolas Schilling, Martin Wistuba, and Lars Schmidt-Thieme. Learning time-series shapelets. In *SIGKDD*, pp. 392–401, 2014.

Raia Hadsell, Sumit Chopra, and Yann LeCun. Dimensionality reduction by learning an invariant mapping. In *CVPR*, volume 2, pp. 1735–1742, 2006.

Jon Hills, Jason Lines, Edgaras Baranauskas, James Mapp, and Anthony Bagnall. Classification of time series by shapelet transformation. *Data Mining and Knowledge Discovery*, 28(4):851–881, 2014.

Eamonn Keogh, Stefano Lonardi, and Chotirat Ann Ratanamahatana. Towards parameter-free data mining. In *SIGKDD*, pp. 206–215, 2004.

Thomas Lampert, Baptiste Lafabregue, Nicolas Serrette, Germain Forestier, Bruno Crémilleux, Christel Vrain, Pierre Gancarski, et al. Constrained distance based clustering for time-series: a comparative and experimental study. *Data Mining and Knowledge Discovery*, 32(6):1663–1707, 2018.

Vladimir Levenshtein. Binary codes capable of correcting deletions, insertions, and reversals. In *Soviet Physics Doklady*, volume 10, pp. 707–710, 1966.

Jason Lines, Luke M Davis, Jon Hills, and Anthony Bagnall. A shapelet transform for time series classification. In *SIGKDD*, pp. 289–297, 2012.

Arnaud Lods, Simon Malinowski, Romain Tavenard, and Laurent Amsaleg. Learning DTW-preserving shapelets. In *IDA*, 2017.

Abdullah Mueen, Eamonn Keogh, and Neal Young. Logical-shapelets: an expressive primitive for time series classification. In *SIGKDD*, pp. 1154–1162, 2011.

Thanawin Rakthanmanon and Eamonn Keogh. Fast shapelets: A scalable algorithm for discovering time series shapelets. In *SDM*, pp. 668–676, 2013.

Hiroaki Sakoe and Seibi Chiba. Dynamic-programming approach to continuous speech recognition. In *International Congress on Acoustics*, pp. 65–69, 1971.

Hiroaki Sakoe and Seibi Chiba. Dynamic programming algorithm optimization for spoken word recognition. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 26(1):43–49, 1978.

Mit Shah, Josif Grabocka, Nicolas Schilling, Martin Wistuba, and Lars Schmidt-Thieme. Learning DTW-shapelets for time-series classification. In *ACM IKDD CODS*, pp. 1–8, 2016.

Ricardo Carlini Sperandio. *Recherche de séries temporelles à l'aide de DTW-preserving shapelets*. PhD thesis, Université Rennes 1, 2019.

Liudmila Ulanova, Nurjahan Begum, and Eamonn Keogh. Scalable clustering of time series with u-shapelets. In *SDM*, pp. 900–908, 2015.

Michail Vlachos, Marios Hadjieleftheriou, Dimitrios Gunopulos, and Eamonn Keogh. Indexing multidimensional time-series. *The VLDB Journal*, 15(1):1–20, 2006.

Kiri Wagstaff, Claire Cardie, Seth Rogers, and Stefan Schrödl. Constrained k-means clustering with background knowledge. In *ICML*, volume 1, pp. 577–584, 2001.

Kiri Wagstaff, Sugato Basu, and Ian Davidson. When is constrained clustering beneficial, and why? In *IAAI*, 2006.

Akihiro Yamaguchi, Shigeru Maya, Kohei Maruchi, and Ken Ueno. Ltspauc: Learning time-series shapelets for optimizing partial auc. In *SDM*, pp. 1–9, 2020.

Lexiang Ye and Eamonn Keogh. Time series shapelets: a new primitive for data mining. In *SIGKDD*, pp. 947–956, 2009.

Lexiang Ye and Eamonn Keogh. Time series shapelets: a novel technique that allows accurate, interpretable and fast classification. *Data Mining and Knowledge Discovery*, 22(1):149–182, 2011.

Jesin Zakaria, Abdullah Mueen, and Eamonn Keogh. Clustering time series using unsupervised-shapelets. In *ICDM*, pp. 785–794, 2012.

Jesin Zakaria, Abdullah Mueen, Eamonn Keogh, and Neal Young. Accelerating the discovery of unsupervised-shapelets. *Data Mining and Knowledge Discovery*, 30(1):243–281, 2016.

Qin Zhang, Jia Wu, Hong Yang, Yingjie Tian, and Chengqi Zhang. Unsupervised feature learning from time series. In *IJCAI*, pp. 2322–2328, 2016.

A Derivation of the CDPS Loss Gradient

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

$$\widehat{DTW}_{i,j} = ||\overline{T}_i - \overline{T}_j||_2, \qquad DTW_{i,j} = DTW(T_i, T_j), \qquad \mathcal{L}(T_i, T_j) = \frac{1}{2}\psi + \phi_{i,j},$$

where,

$$\psi = \frac{1}{2}\left(DTW(T_i, T_j) - \beta||\overline{T}_i - \overline{T}_j||_2\right)^2,$$

and

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

where w is a predefined constant.

A.1 Derivation with Respect to β

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

A.2 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

$$\frac{\partial \psi}{\partial S_{k,l}} = \frac{\partial \psi}{\partial \widehat{DTW}_{i,j}} \frac{\partial \widehat{DTW}_{i,j}}{\partial \Delta_{i,j,k}} \frac{\partial \Delta_{i,j,k}}{\partial S_{k,l}},$$

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

$$\frac{\partial \psi}{\partial \widehat{DTW}_{i,j}} = -2\beta(DTW_{i,j} - \beta \widehat{DTW}_{i,j}),$$

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

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 (Lods et al., 2017) which gives $\frac{\partial \overline{T}_{i,k}}{\partial D_{i,k,m}} = \delta_{m,m^*}$ the above can be written as:

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

$$\frac{\partial \phi}{\partial S_{k,l}} = \frac{\partial \phi}{\partial \widehat{DTW}_{i,j}} \frac{\partial \widehat{DTW}_{i,j}}{\partial \Delta_{i,j,k}} \frac{\partial \Delta_{i,j,k}}{\partial S_{k,l}},$$

where

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

where w is a predefined constant.