Université de Strasbourg

msii ÉCOLE DOCTORALE

École Doctorale Mathématiques, Sciences de
l'Information et de l'Ingénieur
Laboratoire Icube UMR 7357

Dissertation

Presented by

Hussein El Amouri

for the Fulfillment of the Requirements for the Degree of

Doctor of Philosophy

in Computer Science.

Version submitted to the jury members - 01 June 2023

Learning Explainable Constrained Representation for Time-Series

Jury members

President	—	Mr. Cédric Wemmert	Professor, University of Strasbourg, France
Supervisor	—	Mr. Pierre Gançarski	Professor, University of Strasbourg, France
Supervisor	—	Mr. Thomas Lampert	Assistant Professor, University of Strasbourg, France
Supervisor	—	Mr. Clément Mallet	Director of Research, IGN, University Gustave Eiffel, France
Reporter	—	Mr. Romain Tavenard	Professor, University of Rennes 2, France
Reporter	—	Ms. Seham Amer-Yehia	Director of Research, CNRS
Examinator	—	Ms. Thi-Bich-Hanh Dao	Assistant Professor, University of Orléans, France

"The human being is only a reed, the most feeble in nature; but he is a thinking reed. Let us make it our task, then, to think well: here is the principle of morality."

–Blaise Pascal

ACKNOWLEDGEMENTS

I would like to express my heartfelt gratitude to my esteemed thesis supervisors, Pierre Gançarski, Thomas Lampert, and Clément Mallet, for their exceptional support, unwavering guidance, and significant contributions throughout my research journey. Their combined efforts have played a pivotal role in shaping the outcome of this thesis.

I am immensely grateful to Pierre Gançarski and Clément Mallet for their perspectives, constructive comments, and expert guidance. Their invaluable input and critical insights have profoundly influenced the trajectory of my work, enabling me to broaden my understanding, challenge my assumptions, and significantly enhance the overall quality of this thesis. Without their guidance, this research would have been considerably more arduous to accomplish.

I extend my deepest appreciation to Thomas Lampert for his consistent follow-up and unwavering dedication. His continuous guidance, invaluable feedback, and unwavering commitment to my progress have been indispensable in shaping the direction of my research and refining my ideas.

I would like to extend my sincere appreciation to the exceptional SDC team of ICube laboratory. Their unwavering support, fruitful collaboration, and stimulating discussions have been pivotal in shaping the direction of my research and fostering a dynamic and intellectually stimulating research environment. Their collective expertise, encouragement, and camaraderie have been instrumental in my growth as a researcher.

Furthermore, I would like to express my heartfelt thanks to my beloved family, particularly my mother and father, for their unwavering support, boundless love, and continuous encouragement throughout this academic endeavour. Their unwavering belief in my abilities, constant backup, and the nurturing environment they provided have been a constant source of motivation, inspiration, and strength.

I would also like to acknowledge the administrative staff at Strasbourg and the HPC Center (*Centre de Calcul de l'Université de Strasbourg*) for their provision of necessary resources, and technical assistance in conducting the computational aspects of this research. Their diligent efforts and assistance have been crucial in enabling the execution of complex computations and facilitating the smooth progress of my work.

Lastly, I extend my sincere gratitude to all the individuals who have supported and inspired me along this challenging journey. Your unwavering encouragement, constructive feedback, and steadfast belief in my potential have been invaluable. I am deeply grateful for your contributions to my academic and personal growth.

DECLARATION

I hereby declare that this thesis, entitled “Learning Explainable Constrained Representation for Time Series”, submitted in partial fulfillment of the requirements for the degree of Doctor of Philosophy (PhD) in Computer Science, at the University of Strasbourg, is my original work. It has not been submitted for any other degree or examination in any other institution.

The work presented in this thesis gave rise to two publications:

- H. El Amouri, T. Lampert, P. Gançarski, C. Mallet, “Constrained DTW Preserving Shapelets for Explainable Time-Series Clustering”, Pattern Recognition, accepted.
- H. El Amouri, T. Lampert, P. Gançarski, C. Mallet, “CDPS: Constrained DTW-Preserving Shapelets.” Joint European Conference on Machine Learning and Knowledge Discovery in Databases, 2022.

I furthermore declare that I have properly acknowledged and cited all sources, including publications, articles, books, and any other materials used or consulted in the preparation of this thesis. All contributions from other individuals or organizations have been duly acknowledged.

I take full responsibility for the accuracy and originality of the content presented in this thesis. Any assistance received in terms of technical, intellectual, or financial support has been duly acknowledged. I understand that any act of plagiarism or academic dishonesty is a serious offence and may result in severe consequences.

I hereby grant the University of Strasbourg the non-exclusive right to reproduce and distribute copies of this thesis, either in print or electronic format, for scholarly purposes.

Contents

Declaration **v**

Table of Contents **viii**

List of Figures **ix**

List of Tables **xv**

Résumé **xvii**

General Introduction **xxiii**

I Introduction and State-of-the-Art **1**

1 Introduction to Time Series **3**
- 1.1 Time Series . . . 4
  - 1.1.1 Phenomena . . . 4
  - 1.1.2 Time series Analysis and Applications . . . 6
  - 1.1.3 Time Series Representation . . . 7
  - 1.1.4 Distance Measures . . . 9
- 1.2 Problem and Motivation . . . 15
- 1.3 Contribution . . . 16

2 Background and Related Work **19**
- 2.1 Shapelets . . . 20
  - 2.1.1 Definitions and Notations . . . 21
  - 2.1.2 Shapelet Discovery Algorithms . . . 21
  - 2.1.3 Learning DTW Preserving Shapelets . . . 25
- 2.2 Constrained Clustering . . . 27
  - 2.2.1 Clustering Definition . . . 27
  - 2.2.2 Clustering Algorithms . . . 28
  - 2.2.3 Cluster Validation . . . 30
  - 2.2.4 User Constraints . . . 32
  - 2.2.5 Measuring Constraint Properties . . . 34
  - 2.2.6 Constrained Clustering Algorithms . . . 37
- 2.3 Time Series Constrained Clustering . . . 42

2.4 Conclusions 47

II Contributions 49

3 Constrained DTW Preserving Shapelets 51
3.1 Contrastive Learning 52
3.2 Constrained DTW Preserving Shapelets 55
3.2.1 Proposed Loss Function 56
3.2.2 Model Architecture 60
3.2.3 Learning Procedure 62
3.3 Evaluation 63
3.3.1 Experimental Setup 63
3.3.2 Clustering 65
3.4 Discussion 70
3.4.1 Results 70
3.4.2 Model Selection 71
3.4.3 Sensitivity Study 73
3.5 Conclusions 76

4 Shapelet Cluster Explanation 77
4.1 Explanation Framework Definitions 78
4.2 Cluster Explanation: A Case Study 85
4.2.1 Shapelet Selection 90
4.3 Effect of Constraints on the Representation 93
4.4 Conclusions 102

III Conclusions and Prospective Work 105

5 Conclusions 107
5.1 Perspectives 109

Bibliography 111

IV Appendices 127

A Dynamic Time Warping 129

B Datasets 131

C Results 133

LIST OF FIGURES

1 Signaux de séries temporelles montrant différentes informations sur la météo à Strasbourg, France, pendant le mois de février 2023, tirés de Weather Underground. xviii

2 Time-series signals showing different information about the weather in Strasbourg-France during the month of February 2023, taken from Weather Underground. xxiv

1.1 The original time series (first row) and its component: Trend (second row), Seasonality :=Season (third row), and Residue :=Resid (last row). 5

1.2 Common types of distortions. 6

1.3 Different representations of multivariate time series: (a) feature-wise – Case 1 (see text) the time series is a set of univariate time series $(f_1, f_2, f_3, \dots)$, and (b) time-wise – Case 2 (see text) the time series is a set of vectors of dimension f-features $(f_1, f_2, f_3, \dots)$. . . . 8

1.4 Illustration depicting that perceptually similar time series and dissimilar ones can have a close Euclidean distance. Time series A and C follow a similar trend unlike B. 11

1.5 Illustration showing perceptually similar time series (A and C) and dissimilar ones (A and C) are correctly identified by DTW. Time series A and C follow a similar trend while B follows a different one. 12

2.1 Illustration of shapelet S that captures the local similarity of time series T_1 and discriminates it from time series T_2. 21

2.2 Shapelet decision tree for the arrow head dataset. The found shapelets are able to differentiate between the two types of arrow heads (Clovis and Avonlea). Since shapelet I has better quality (lower value indicates better separation), it is used as the root for the decision tree. Adapted from [7]. 22

2.3 Shapelet discovery process. In (a) the bag of all possible shapelets is generated, in (b) for each candidate the distance is calculated to the all time series, then in (c) the maximum information gain is recorded. Adapted from [35]. 23

2.4 Illustration of Fast shapelets principal. A subsequence of the time series is projected into a symbolic representation "adbacc" (top left). Using a sliding window technique multiple subsequences are shown in the lower left part. The time series represents the skull of a lizard. Adapted from [36]. 23
2.5 Illustration of logical shapelets. (a) shows two classes of synthetic time series. (b) Demonstrates the inability of the shapelets to effectively separate the classes (any other shapelet will fail as well). (c) Using both shapelets in the logical sense "and" (i.e. both should exist) successfully achieves the separation of the classes. Adapted from [37]. 24
2.6 Shapelet Transform adapted from [35]. Each time series is mapped to a vectorial representation each component is the minimum distance between the time series and shapelets. 25
2.7 Learning DTW Preserving Shapelets (LDPS) [32]. Time series T_1 and T_2 are mapped to Euclidean space using the shapelets S_1 and S_2 where the distance between the time series in the new space approximates DTW distance. Adapted from [32]. 26
2.8 General example of clustering data points. 27
2.9 Examples of user constraints, showing the different types of constraints. The figure shows three different clusters (dashed ellipsis) of instances (black points). The constraints shown are Instance level (must-link 'ML' in green and cannot-link 'CL' in red), and Cluster level constraints (δ, ϵ, and γ in blue) representing the minimum separation, the neighbourhood radius, and the maximum diameter respectively. The ML and CL constraints specify the relation between objects. ML indicates that the objects should be within the same cluster while CL should not. Adapted from [1]. 32
2.10 Illustration of informativeness (a) and coherence (b). The red lines with X indicate cannot link constraints while the green line indicates must link constraints. Adapted from [4]. 35
2.11 Illustration of calculating constraint coherence, showing three cases of computing the projected overlap between constraints a and b; where they constraint the points (a_1, a_2) and (b_1, b_2) respectively. The points indexed 1 always appear to the left of the other. Adapted from [4]. 35
2.12 The DCC method architecture. Adapted from [160]. 46
2.13 The algorithmic pipeline of FeatTS, adapted from [158]. 47

3.1 Illustration of the general idea behind contrastive learning. (b) Shows contrastive relations according to colour (blue, white) and in (c) according to brands (Toyota, Mercedes). 53
3.2 Illustration of the loss function L against the energy $\mathcal{D}_W$, where the red dashed line is the loss of positive pairs and the blue solid line is for the negative pairs. Adapted from [172]. 55

3.3 Graph of the margin $m_{i,j}$ (represented as m in the figure, which acts upon CL constraints, see Equation 3.8) against $DTW(T_i, T_j)$ and with $\max_{\forall i,\forall j}(DTW(T_i, T_j)) = 20$. 58

3.4 Illustration of the effect of the proposed loss function on the distance of the points in the new space. Different cases are illustrated depending on whether the points are linked with a must-link (green) or cannot-link (red) constraint, blue indicates the effect of approximating DTW. 59

3.5 Illustration of the CDPS model. ST is the shapelet transform, S_k are the weights of the shapelets, T_i ($i = \{1, 2\}$) are the input generated by the batch layer. 61

3.6 Illustration of the shapelet layer. $D_{i,k,w}$ is the Shapelet Euclidean Score (Definition 4), L_{S_i} is the length of the shapelet block, NUM$_i$ is the number of shapelets in a shapelet block, and $\overline{T}_{i,k}$ is the embeddings with respect to each shapelet block. 62

3.7 A Transductive comparison between CDPS+K-means (y-axis) and Raw-TS+(constrained K-means) (x-axis) with different constraint fractions. Each dot represents a dataset from the UCR archive, blue dots represent COP-Kmeans performance and red dots MIP-Kmeans performance. The datasets located in the blue triangle are those for which CDPS performs better than COP-Kmeans and MIP-Kmeans, and the white triangle is the opposite. The term “wins” indicate the number of datasets in which either CDPS have a higher score compared to the comparison (in this case wins is located on top of each figure) or vice versa (wins located at the bottom). 66

3.8 An inductive comparison between CDPS+K-means (y-axis) and Raw-TS+(constrained K-means) (x-axis) with different constraint fractions. Each dot represents a dataset from the UCR archive, blue dots represent COP-Kmeans performance and red dots MIP-Kmeans performance. The datasets located in the blue triangle are those for which CDPS performs better than COP-Kmeans and MIP-Kmeans, and the white triangle is the inverse. “wins” indicates the number of datasets in which either CDPS has a higher score compared to the comparison (in this case wins is located on top of each figure) or vice versa (wins is located at the bottom). 68

3.9 A comparison between CDPS+FeatTS and FeatTS with respect to NMI score. The top wins, indicates how many datasets FeatTS with CDPS features outperforms FeatTS, and the number of FeatTS wins is located at the bottom. 69

3.10 Clustering quality (NMI) as a function of the number of epochs for each dataset, using a constraint fraction of 30%. 72

3.11 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. 73

3.12 Heatmaps showing CDPS sensitivity to α and γ parameters, using 6 random univariate datasets (brighter colours indicate better performance). 74
3.13 Heatmaps of CDPS sensitivity to shapelet parameters (brighter colour indicates better performance). 75

4.1 Time-series samples from each category of the Synthetic Control dataset are presented. Colours correspond to the clustering in Figures 4.3, 4.6 and 4.7. 85
4.2 The normalised cumulative Information Gain calculated using GS-SCE. 86
4.3 Successive global explanation (GS-SCE). The top three shapelets are presented with the corresponding scatter plots of the data projected into each space, defined by adding each shapelet in succession (starting from S_{54}). 87
4.4 NMI of K-means clustering and normalized cumulative information gain NCIG relative to the number of shapelets used to form the representation (ordered by GS-SCE). The labels used for NMI are the clustering labels output by CDPS. The plateau in NCIG indicates that these shapelets add no further information. 88
4.5 Comparison of Clustering Performance on Increasing Number of Shapelets Ranked by GS-SCE (termed SCE) and Increasing Number of PCA Components (termed PCA). Note that the cluster labels obtained from the entire shapelet space are used as class labels for computing the Normalized Mutual Information (NMI) scores. 89
4.6 Independent global explanation (GI-SCE). The top three shapelets are presented with the corresponding scatter plots of the data projected into each space, defined by adding each shapelet in succession. 90
4.7 Combined global explanation (GC-SCE). The top three shapelets are presented with the corresponding scatter plots of the data projected into each space, defined by adding each shapelet in succession. 91
4.8 NMI of K-means clustering relative to the number of shapelets used to form the representation ordered by GI-SCE (Figure 4.8a) and by GC-SCE (Figure 4.8b). The labels used for NMI are the clustering labels output by CDPS. 92
4.9 Comparison of distance map between the actual DTW (a) and approximate DTW using all shapelets (b) and a subset of shapelets (c) for the Plane dataset. The colour bar indicates the distance between the different samples, the brighter the larger the distance. 92
4.10 Visualization of ML/CL: the ratio of the average distance between ML constraints to the average distance between CL constraints across different spaces LDPS, CDPS, and RAW time series space using DTW. The ratio was calculated for ten different univariate datasets. The constraint sets, for each constraint fraction, that were considered under the transductive study are used. 93

4.11 Visualization of the constraints on the LDPS and CDPS representational space using PaCMAP dimensionality reduction; for three different datasets. Must-link constraints are represented in solid lines coloured according to the ground truth labels and cannot-link constraints are the red dashed lines. 95

4.12 Visualization of the constraints on the MDS embedded space calculated using DTW, LDPS and CDPS distance maps; for three different datasets. Must-link constraints are represented in solid lines coloured according to the ground truth labels and cannot-link constraints are the red dashed lines. 96

4.13 Illustration of the effect of the constraints for different tasks on the same dataset. The visualisation is on LDPS and CDPS using PaCMAP dimensionality reduction. The dataset used contains samples of pointing a gun. (a) and (b) shows the task of grouping the samples based on age while (c) and (d) are based on the movement itself. 97

4.14 Illustration of the GunPointAgeSpan classes. It shows 20 samples chosen at random for each class. Each sample is recorded over five seconds with 30 frames per second, the 150 frames are represented in the x-axis. The y-axis shows the position of the abscissa component of the hand's centroid in each frame. 98

4.15 Distribution of coherence of the constraints calculated for different datasets based on DTW, LDPS and CDPS. For DTW the coherence is shown in the MDS embedding space with dimensions equal to a number of shapelets $|\mathcal{S}|$ is presented as well. 99

4.16 Scatter plot pf coherence versus NMI calculated for different datasets using the LDPS, CDPS and DTW with MDS embeddings with dimension equal to $|\mathcal{S}|$. The figure also shows the best fit of the data with the correlation factor. 99

4.17 Average distance between the must-link and cannot-link constraints ML/CL for the reduced CDPS representation space in comparison to using all the space. GS-SCE was used to reduce the original representational space and compared to PCA dimensionality reduction. 101

4.18 This figure shows the coherence for the reduced CDPS representation space in comparison to using all the space. GS-SCE was used to reduce the representational space and compared to PCA dimensionality reduction. 102

4.19 Illustration of the samples and constraints for the reduced CDPS representation space using PCA and GS-SCE. The example used in Section 4.2 is used herein to visually compare PCA with two principal components to GS-SCE. (a) and (b) show the scatter plots of the data points using the first two principal components of PCA and the first two shapelets ranked using GS-SCE. (c) and (d) show visualizations of a subsample of the constraints in each of the spaces shown in (a) and (b) respectively. 103

A.1 Illustration of mapping between points based on Euclidean distance (a) and DTW similarity (b). The wrapping path of DTW is shown in (c). 129

LIST OF TABLES

4.1 Clustering scores for different datasets using all shapelets (100% NCIG) and a subset of the shapelets (thresholded to 80% NCIG). . . 91

B.1 List of UCR datasets used in the main study. 131

C.1 Transductive NMI results for CDPS on the UCR archive. 133
C.2 Inductive NMI results for CDPS on the UCR archive. 137
C.3 Transductive NMI results. KM, CKM, MKM represents K-means, COP-Kmeans, MIP-Kmeans respectively. 141
C.4 Inductive NMI results. KM, CKM, MKM represents K-means, COP-Kmeans, MIP-Kmeans respectively. 144

Résumé

"Tous les hommes aspirent par nature à la connaissance."
–Aristotle

Contexte

Dans les années 1970, a débuté l'ère de l'information, caractérisée par une augmentation du volume et de la complexité des données. Dans les années 1990, de nouvelles stratégies et techniques étaient nécessaires pour faire face aux défis de l'analyse de ces données, regroupées sous le terme de « Data mining ». Le Data mining est le processus d'identification de modèles et de corrélations dans les données à l'aide d'approches mathématiques (par exemple, le théorème de Bayes - XVIIIe siècle, l'analyse de régression - XIXe siècle) et de l'informatique, en particulier le domaine de l'apprentissage automatique (réseaux neuronaux - années 1950, arbres de décision - années 1960). Dans les années 2000, l'augmentation est devenue exponentielle et est connue sous le nom de "Big Data". Parallèlement, le développement de l'apprentissage profond, de l'apprentissage automatique et des serveurs GPU a permis au Data mining de faire face efficacement à des domaines divers et à des défis spécifiques aux données.

Cependant, comme les données peuvent être catégorisées selon diverses caractéristiques ou structures, par exemple "Étiquetées vs Non étiquetées", "Temporelles vs Statiques", "Structurées vs Non structurées", "Numériques vs Catégorielles", et "Continues vs Discrètes", le nombre de méthodes de Data mining est très important et varié. Il est donc nécessaire de prendre en compte ces caractéristiques et structures lors du choix de la méthode d'analyse à utiliser.

Données de Séries Temporelles

Dans ce travail, nous nous intéressons à la catégorie de données temporelles, c'est-à-dire des séquences dépendantes du temps, plus spécifiquement les données de séries temporelles (Figure 1). Ce domaine est considéré relativement nouveau en apprentissage automatique, car il a commencé à susciter de l'intérêt dans les années 1970, mais n'a que récemment commencé à être sérieusement étudié. De telles

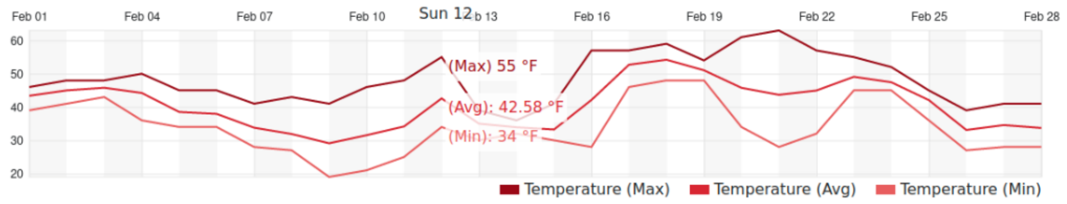


Figure 1: Signaux de séries temporelles montrant différentes informations sur la météo à Strasbourg, France, pendant le mois de février 2023, tirés de Weather Underground.

données suivent l'évolution des phénomènes au fil du temps, à intervalles de temps constants ou variables (taux d'échantillonnage), ce qui donne une séquence ordonnée de valeurs réelles, voir Section 1.1. En général, tout type de données respectant cette condition peut être considéré comme une série temporelle. Par exemple, la collecte de données météorologiques à des fins de prévision, la surveillance de la santé et la télédétection. D'autres objets pouvant être représentés sous la forme d'une séquence ordonnée de valeurs réelles peuvent être considérés de manière similaire, tels que la représentation unidimensionnelle d'objets, etc.

Les données de séries temporelles deviennent de plus en plus complexes et volumineuses en raison des avancées des technologies de détection. Cela a entraîné un certain nombre de défis, tels que les distorsions, l'étiquetage des données et l'explication des résultats.

Les distorsions dans les séries temporelles se manifestent dans le temps (décalage) et l'amplitude (mise à l'échelle). Ces distorsions rendent difficile l'utilisation de mesures de distance de type un-à-un (comme la distance euclidienne) car elles ne tiennent pas compte des décalages et de l'échelle. Cela rend inefficaces un grand nombre d'algorithmes.

Pour atténuer ce défi, deux approches peuvent être utilisées : soit l'utilisation d'une mesure adaptée à ce type de données (par exemple, la DTW - Dynamic Time Warping), soit la transformation des données pour les projeter dans un espace où les méthodes conventionnelles (distance euclidienne) peuvent être utilisées directement.

La première approche vise à définir des mesures basées sur la distance qui établissent des correspondances entre les points de données, en prenant en compte les distorsions. De telles mesures sont connues sous le nom de "mesures élastiques". La Dynamic Time Warping (DTW) est l'une des mesures élastiques les plus couramment utilisées, dans laquelle la distance est calculée en fonction du chemin optimal qui relie les points similaires entre eux. Cependant, son calcul a tendance à être lent lorsqu'il est utilisé avec de grands ensembles de données. Il convient de noter que la plupart des algorithmes doivent être adaptés pour incorporer ces mesures lorsqu'ils sont utilisés avec des séries temporelles [1].

La deuxième approche consiste à apprendre une nouvelle représentation de la série temporelle, qui peut être utilisée ultérieurement avec des algorithmes d'apprentissage automatique standard. Quelques exemples de méthodes visant à apprendre une telle transformation sont la transformation en shapelet, l'approche Symbolic Aggregate Approximation (SAX), l'approximation en morceaux (Piecewise Aggregate Approximation - PAA), etc.

Analyse des Séries Temporelles

Les méthodes visant à regrouper les séries temporelles en classes ou en clusters relèvent des termes de l'apprentissage supervisé, de l'apprentissage non supervisé et de l'apprentissage semi-supervisé, où :

Apprentissage supervisé Utilise des connaissances préalables sur les données en fournissant des étiquettes pendant le processus d'entraînement pour apprendre les dépendances cachées et les relations entre les données d'entrée et les étiquettes fournies. Les performances du processus d'apprentissage sont ensuite mesurées sur un ensemble de test non vu. Étant donné que le processus d'étiquetage des données temporelles est fastidieux, coûteux et chronophage, les techniques supervisées ne sont souvent pas préférées.

Apprentissage non supervisé Traite des données non étiquetées, sans aucune information préalable, afin de découvrir la structure sous-jacente et la distribution, uniquement en fonction du biais de l'algorithme. En raison de l'absence d'étiquettes, la mesure de la performance est difficile et les résultats peuvent ne pas correspondre à l'intuition de l'expert.

Apprentissage semi-supervisé dans le but d'éviter ces deux problèmes (difficultés à obtenir de bons exemples en raison du manque d'informations préalables et du coûteux processus d'étiquetage de l'ensemble des échantillons), des approches dites semi-supervisées ont émergé. Elles impliquent l'incorporation de l'expert dans le processus. Par exemple, l'expert est autorisé à guider le processus en injectant des informations telles que des contraintes (apprentissage contraint), en validant à des moments clés (apprentissage actif), en fournissant un faible pourcentage d'étiquettes, etc.

Dans ce travail, nous nous concentrons sur l'utilisation de contraintes pour apprendre le regroupement. Ces contraintes représentent des informations préalables (intuition de l'expert) fournies à l'algorithme. Des exemples d'algorithmes de ce type sont le *Constrained k-means* [2] (CK-means), le *Deep Constrained Clustering* (DCC) [3], etc. Ces contraintes peuvent prendre différentes formes, les plus connues étant les contraintes par paires (également appelées contraintes de niveau d'instance) : les contraintes de type "doivent être liées" (must-link) indiquent que deux échantillons sont similaires, tandis que les contraintes de type "ne doivent pas être liées" (cannot-link) indiquent qu'ils ne le sont pas. Les algorithmes de regroupement contraint visent à respecter ces contraintes tout en attribuant des points à différents groupes en fonction de leurs distances.

Dans le travail de Davidson et al. [4] sur les propriétés des contraintes par paires, ils ont formalisé les notions d'**informativité** et de **cohérence** des contraintes. L'**informativité** quantifie l'utilité des informations fournies par l'expert au processus d'apprentissage, informations que l'algorithme seul ne serait pas capable d'apprendre uniquement en se basant sur son biais. La **cohérence** mesure le degré de conflit entre les contraintes selon la métrique définie, où les contraintes "doivent être liées" sont considérées comme exerçant une force attractive et les contraintes "ne doivent pas être liées" une force répulsive. Avoir différents types de contraintes à proximité peut entraîner des conflits lors de l'attribution des échantillons aux groupes. L'informativité dépend de l'algorithme, tandis que la cohérence est indépendante de l'algorithme. Selon la définition originale, la cohérence nécessite un espace métrique pour calculer le chevauchement entre les contraintes, c'est-à-dire le conflit, ce qui n'est pas possible avec des mesures telles que DTW.

Un autre défi précédemment mentionné est d'expliquer la sortie du regroupement, car elle peut être complexe et difficile à interpréter, même pour les experts du domaine. Cette difficulté est amplifiée par la complexité des données de séries temporelles. Pour remédier à cela, certains algorithmes, tels que le *Maximal Frequent Item-Kmeans* [5] - MFI-Kmeans, la méthode d'apprentissage conceptuel basée sur le flou [6] - FCLM, utilisent le regroupement conceptuel pour apprendre des concepts de plus haut niveau à partir des données. Ces concepts sont utilisés pour expliquer les résultats du regroupement, en respectant strictement des conditions spécifiques. De plus, des techniques de visualisation, de réduction de dimension, des techniques basées sur des règles, etc., peuvent être utilisées pour faciliter l'explication des résultats du regroupement.

Travail Proposé

En résumé, le regroupement de séries temporelles est une tâche difficile qui nécessite des mesures de distance adaptées pour analyser et interpréter efficacement les données. De plus, les sorties des algorithmes de regroupement de séries temporelles peuvent être difficiles à interpréter. Pour remédier à cela, les experts peuvent fournir des connaissances préalables en utilisant des contraintes de niveau d'instance, qui guident l'algorithme et alignent les résultats sur les besoins de l'expert.

Cependant, l'incorporation de contraintes de niveau d'instance dans le processus de regroupement introduit le défi d'évaluer l'informativité et la cohérence des contraintes. De plus, l'intégration de techniques d'explication avec le paradigme de regroupement contraint demeure une question ouverte.

Dans ce travail, nous cherchons d'abord à développer une représentation basée sur les contraintes pour les séries temporelles qui tire parti des propriétés des mesures élastiques, en particulier la Dynamic Time Warping (DTW), en utilisant des shapelets et la transformation en shapelet. Notre approche vise à apprendre un espace où les propriétés des contraintes (par exemple, l'informativité et la cohérence) peuvent être calculées, et où la distance entre les objets approxime la mesure de distance DTW (Dynamic Time Warping) de manière à prendre en compte les distorsions dans les séries temporelles.

Deuxièmement, nous développons un cadre pour fournir des explications des résultats du regroupement de séries temporelles (au niveau des clusters, c'est-à-dire pour les clusters individuels par rapport à tout le reste, et au niveau global, c'est-à-dire le regroupement dans son ensemble), en nous basant uniquement sur la représentation découverte, c'est-à-dire les shapelets apprises. En tant que telle, la nouvelle représentation améliore les performances du regroupement tout en facilitant l'explication des résultats.

Pour obtenir un espace explicatif conforme aux mesures basées sur la métrique, nous utilisons la transformation en shapelet. La transformation en shapelet est une transformation des séries temporelles basée sur des sous-séquences discriminatives capables de différencier les séries temporelles en calculant la distance entre elles et les shapelets. Comme les shapelets peuvent être vues de la même manière que les séries temporelles (en tant que sous-séquences), elles devraient être plus faciles à interpréter pour l'expert [7]. Pour garantir que la distance entre les séries temporelles dans le nouvel espace tienne compte des distorsions, la distance entre leurs représentations (dans l'espace transformé) est contrainte pour approximer la distance DTW appliquée aux séries temporelles brutes. Enfin, pour intégrer l'apport de l'expert dans le processus d'apprentissage et le personnaliser à ses besoins, des contraintes par paires sont intégrées.

Plan de Thèse

Cette thèse est organisée comme suit :

Partie 1 : Introduction et État de l'Art

Chapitre 1 fournit les connaissances de base nécessaires et les définitions pour les données de séries temporelles et l'analyse de séries temporelles.

Chapitre 2 présente les travaux connexes et les connaissances de base nécessaires sur les shapelets, les contraintes et le regroupement.

Partie 2 : Shapelets Préservant la DTW Contrainte

Chapitre 3 présente notre approche proposée d'apprentissage de représentation contrainte, appelée Shapelets Préservant la DTW Contrainte (CDPS). Il présente également les expérimentations et les discussions menées pour évaluer la robustesse et l'efficacité de CDPS.

Chapitre 4 introduit plusieurs approches qui tirent parti de la représentation apprise à l'aide de CDPS pour expliquer les résultats de toute approche de regroupement utilisée pour analyser les données. Cette approche est appelée Explication du Regroupement avec Shapelets (SCE). Il démontre également le processus d'explication sur un cas d'utilisation et les résultats sont discutés.

Partie 3 : Conclusions et Perspectives

Chapitre 5 nous présentons les conclusions de nos contributions et les directions futures possibles.

GENERAL INTRODUCTION

"All men by nature desire knowledge."
–Aristotle

Context

In the 1970s, the era of information began, characterized by an increase in terms of volume and complexity of data. In the 1990s novel strategies and techniques were needed to address the challenges faced with analysing such data, falling under the term 'Data mining'. Data mining is the process of identifying patterns and correlations in data using mathematical approaches (e.g. Bayes' theorem–1700s, regression analysis–1800s) and computer science, particularly the field of machine learning (neural networks–1950s, decision trees–1960s). In the 2000s, the increase became exponential and is known as "Big Data". In parallel, the development of deep learning, machine learning and GPU servers enabled data mining to efficiently address diverse domains and data-specific challenges.

Nevertheless, as data can be categorized using various characteristics or structures, for example 'Labeled vs. Unlabeled', 'Temporal vs. Static', 'Structured vs. Unstructured', 'Numerical vs. Categorical', and 'Continuous vs. Discrete' the number of data mining methods are very large and various. Thus, it is necessary to take these characteristics and structures into account when choosing the analytical method to be used.

Time Series Data

In this work, we are interested in the category of temporal data, i.e. time-dependent sequences, more specifically Time Series data (Figure 2). This domain is considered relatively new in machine learning as it started to gain interest in the 1970s but has only recently started to receive serious attention. Such data tracks the change of phenomena through time, at constant or variable time intervals (sampling rate), resulting in an ordered sequence of real values, see Section 1.1. Generally, any type of data respecting this condition can be considered a time series. For example, recording weather data for forecasting purposes, health monitoring, and remote

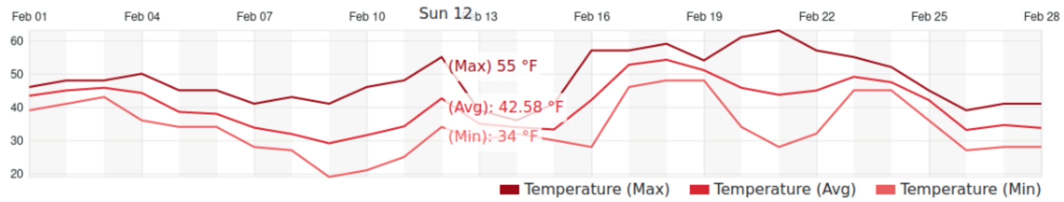


Figure 2: Time-series signals showing different information about the weather in Strasbourg-France during the month of February 2023, taken from Weather Underground.

sensing. Other objects that can be represented as an ordered sequence of real values can be considered in a similar manner, such as one-dimensional representation of objects, etc.

Time-series data is becoming increasingly complex and voluminous due to advancements in sensing technologies. This resulted in a number of challenges, e.g. distortions, data labelling, and result explanation.

Distortions in time series are exhibited in time (shift) and amplitude (scaling). These distortions make it challenging to use one-to-one based distance measures (such as Euclidean distance) since they do not take shifts and scale into account. This renders a wide number of algorithms inefficient. To mitigate this challenge, two approaches can be used: either using a measure adapted to this type of data (e.g. DTW – Dynamic Time Warping) or transforming the data to project it into a space in which conventional methods (Euclidean distance) can be used directly.

The first approach aims to define distance-based measures that establish correspondences between data points, taking distortions into account, such measures are known as 'elastic measures'. Dynamic Time Warping (DTW) is one of the most commonly used elastic measures, in which distance is calculated based on the optimal path that connects similar points together. Its calculation, however, tends to be slow when used with large datasets. It should be noted that most algorithms need to be adapted to incorporate these measures when used with time series [1].

The second approach involves learning a new representation of the time series, which can be used later with standard machine learning algorithms. Some examples of methods that aim to learn such a transformation are shapelet transform, Symbolic Aggregate approXimation (SAX), Piecewise Aggregate Approximation (PAA), etc.

Time Series Analysis

Methods to group time series into classes/clusters fall under the terms supervised, unsupervised, and semi-supervised, where:

Supervised learning uses prior knowledge of the data by providing labels during the training process to learn hidden dependencies and relationships between the input and the provided labels. The performance of the learning process is then measured on an unseen test set. Since the data labelling process for time series is tedious, expensive, and time-consuming, supervised techniques are often not preferred.

Unsupervised learning handles unlabeled data, without any prior information, to unravel underlying structure and distribution, based solely on the algorithm's bias. Due to the lack of labels, measuring the performance is challenging and the results may not be aligned with the expert's intuition.

Semi-supervised learning in an attempt to avoid these two problems (difficulties in obtaining good examples due to the lack of prior information and the expensive process of labelling all samples), so-called semi-supervised approaches have emerged. These involve incorporating the expert into the process. For example, the expert is allowed to guide the process by injecting information such as constraints (constrained learning), validating at key moments (active learning), providing a small percentage of labels, and so on.

In this work, we focus on using constraints to learn clustering. Where these constraints represent prior information (expert intuition) that is provided to the algorithm, examples of such algorithms are Constrained K-means [2] (COP-Kmeans), Deep Constrained Clustering (DCC) [3], etc. These constraints can take different forms, the most known are the pairwise constraints (also known as instance-level): must-link and cannot-link. Must-link constraints indicate that a pair of samples are similar while cannot-link constraints indicate that they are not. Constrained clustering algorithms aim to respect the constraints while assigning points to different clusters based on their distances.

In the work of Davidson et al. [4] on pairwise constraint properties, they formalized the **informativeness** and **coherence** of constraints. **Informativeness** quantifies the usefulness of the information provided by the expert to the learning process, where the algorithm alone may not be able to learn such information solely based on its bias. **Coherence** measures the degree of conflict between constraints according to the defined metric, where must-link are considered to exert an attractive force and cannot-link a repulsive force. Having different types of constraints in proximity may lead to conflict while assigning the samples to the clusters. Informativeness is algorithm dependent, while coherence is independent of the algorithm. According to the original definition coherence is verified by calculating the overlap between constraints, i.e. the conflict. This requires a metric space, which is not possible with measures such as DTW.

Another challenge previously mentioned is explaining the clustering output, since it can be complex and difficult to interpret, even for domain experts. This difficulty is further amplified by the complexity of time-series data. To address this, some algorithms, (such as Maximal Frequent Item-Kmeans [5] – MFI-Kmeans, Fuzzy-Based Concept Learning Method [6] – FCLM) employ conceptual clustering to learn higher-level concepts from the data. These concepts are used to explain the clustering results, with strict adherence to specific conditions. Additionally, visualization techniques, dimensionality reduction, rule-based techniques, etc., can be employed to facilitate the explanation of clustering results.

Proposed Work

In summary, time-series clustering is a challenging task that requires tailored distance measures to effectively analyze and interpret data. Moreover, time series clustering algorithm outputs can be hard to interpret. To address this, experts can provide prior knowledge using instance-level constraints, which guide the algorithm and align results with the expert's needs.

However, incorporating instance-level constraints in the clustering process introduces the challenge of assessing the informativeness and coherence of the constraints. Furthermore, integrating explanation techniques with the constrained clustering paradigm remains an open issue.

In this work, we first try to develop a constraint-based representation for time series that leverages elastic measure properties, specifically Dynamic Time Warping (DTW), using shapelets and the shapelet transform. Our approach aims to learn a space where constraint properties (e.g. informativeness and coherence) can be calculated, and the distance between objects approximates the DTW (Dynamic Time Warping) distance measure so that it can take into account distortions in the time series.

Second, we develop a framework to provide explanations for time-series clustering results (for cluster level, i.e. for individual clusters compared to everything else, and global level, i.e. the overall clustering itself), solely by relying on the discovered representation, that is the learned shapelets. As such, the new representation enhances clustering performance while facilitating an explanation of the results.

To achieve an explainable space that adheres to metric-based measures, we use the shapelet transform. The shapelet transform is a time series transformation based on discriminative sub-sequences that are able to differentiate between time series by calculating the distance between them and the shapelets. Since shapelets can be viewed in the same manner as time series (being sub-sequences), they should be easier for the expert to interpret [7]. To ensure that the distance between the time series in the new space accounts for distortions, the distance between their representations (in the transformed space) is constrained to approximate the DTW distance applied to the raw time series. Finally, to incorporate expert input into the learning process and to customize it to their needs, pairwise constraints are integrated.

Thesis Outline

This thesis is organized as follows:

Part 1: Introduction and State-of-the-Art

Chapter 1 provides the necessary background knowledge and definitions for time series data and time series analysis.

Chapter 2 provides the related work and background knowledge necessary for shapelets, constraints and clustering.

Part 2: Constrained DTW Preserving Shapelets

Chapter 3 presents our proposed constrained representation learning approach, called Constrained DTW Preserving Shapelets (CDPS). It also presents the experimentation and discussion carried out to evaluate CDPS's robustness and effectiveness.

Chapter 4 introduces several approaches that leverage the representation learnt using CDPS to explain the results of any clustering approach used to analyze the data, this approach is termed Shapelet Cluster Explanation (SCE). It also demonstrates the explanation process on a use case and the results are discussed.

Part 3: Conclusions and Prospective Work

Chapter 5 we present the conclusions of our contributions and possible future directions.

Part I

Introduction and State-of-the-Art

CHAPTER 1 INTRODUCTION TO TIME SERIES

"The journey of a thousand miles begins with one step."

– Lao Tzu

1.1 Time Series 4
  1.1.1 Phenomena 4
  1.1.2 Time series Analysis and Applications 6
  1.1.3 Time Series Representation 7
  1.1.4 Distance Measures 9
1.2 Problem and Motivation 15
1.3 Contribution 16

Studying a phenomenon often involves observing various aspects and variables of the phenomenon either at a specific time instant or its evolution over time. The resulting data from observing the evolution of phenomena over time is referred to as time series. Time series data contains valuable information that can reveal significant aspects which may not be apparent when studying the phenomenon at a single instant. Analyzing such data depends on the specific task at hand, such as anomaly detection in a production pipeline or predicting stock market trends. Analyzing time series data is challenging regardless of the task due to the presence of various types of distortions. These distortions may include missing data or temporal shifts between different observations. These complexities add an additional layer of difficulty when measuring similarity between time series data, as similarity measures need to account for and accommodate such distortions. In this chapter, we will introduce time series data, explore different approaches for analyzing such data, discuss measuring similarity between time series data, and provide the motivation behind our work.

1.1 Time Series

In the following section, we will discuss the definitions and notations used for time series data. We will also provide a detailed explanation of time series distortions and the different measures used for them. Furthermore, we will present various techniques for analyzing time series data and discuss their applications.

1.1.1 Phenomena

A time series generally corresponds to a sequence of observations (value of a stock, radiometry of a terrestrial object, etc.) of a phenomenon (evolution of the course of a stock, agricultural cycle, etc.) captured at intervals of time that may or may not be regular. The frequency of acquisition is the amount of time between two observations. The phenomena can exhibit recurring patterns known as seasonality, e.g. a cardiograph showing a heartbeat. It can follow a general trend which can be increasing, decreasing or both, e.g. the blood pressure of a nervous patient increases. It can have irregularities known as residuals, e.g. unexpected disruptions in the environment can cause sudden abnormalities in the subject being monitored. Figure 1.1 presents a visualization depicting these characteristics.

Analyzing such a series involves understanding the phenomena captured. For example, the analysis may be to learn a classification model (supervised learning) or look for regularities (frequent patterns), abnormal behaviour (fraud detection), or even groups (clustering) within these series. Such analysis is highly dependent on the phenomenon being studied and the data itself. For example, in remote sensing, the analysis of an agricultural crop yearly cycle depends on various factors, including the types of crop (wheat, maize, etc.) present in the area under consideration. Additionally, the growth cycle of a specific crop, wheat for example, can vary based on numerous parameters. Thus, for two different observations (instances) of the same phenomena we can observe a lag between the two cycles if the sowing dates are different, having different growing conditions (soil quality, irrigation, etc.) can lead to longer or shorter cycles and/or different yields, one-off events (hail, thunderstorms, etc.) can cause crops to end prematurely. Furthermore, studying the seasonality of crops can provide important information about the state of the soil at a given moment, having different annual weather patterns can disrupt cycles throughout the area thus complicating multi-year comparisons, the geographical location of the study area limits the types of crops grown and disrupts their cycles. In addition to understanding the phenomena at hand, the analysis also depends on the representative data. Some of the factors involved in this are as follows:

Acquisition frequency: how often values are observed and recorded, which must be sufficiently frequent to capture the phenomena being observed.

Acquisition quality: range of acquisition values, i.e. measurement resolution, absence or presence of noise, using an appropriate recording device, etc.

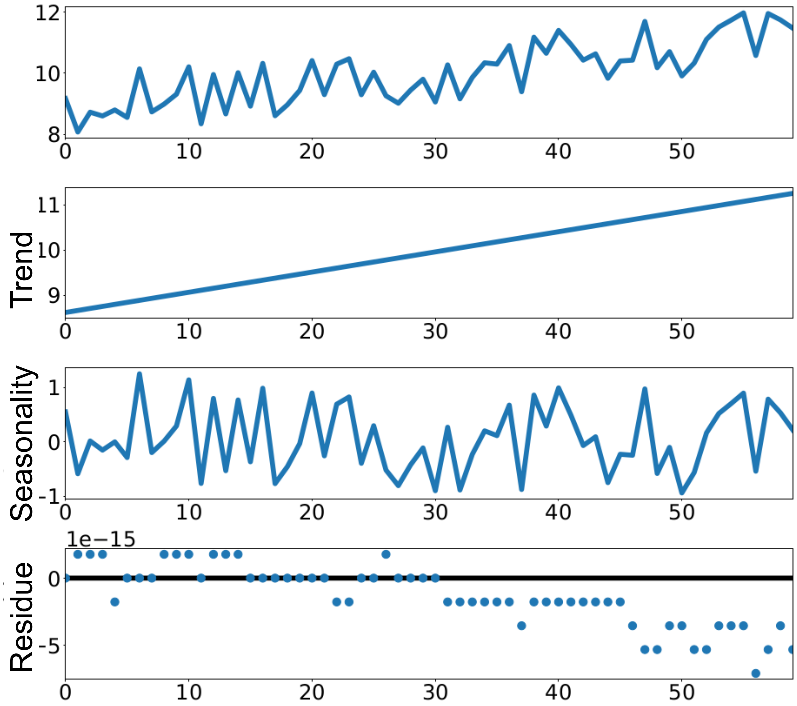


Figure 1.1: The original time series (first row) and its component: Trend (second row), Seasonality :=Season (third row), and Residue :=Resid (last row).

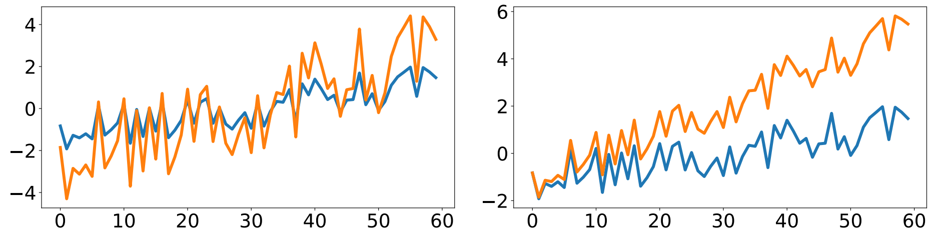
The possibility of missing data: for example, having an obstacle in front of the recording device such as in cloudy weather where it will be hard for the device to capture relevant information about the phenomena.

Thus, time series data corresponding to the same phenomena are subject to various distortions due to the phenomena itself, such as amplitude, trend, time, phase, missing values, and outliers [8]. The following is an explanation of each type of distortion.

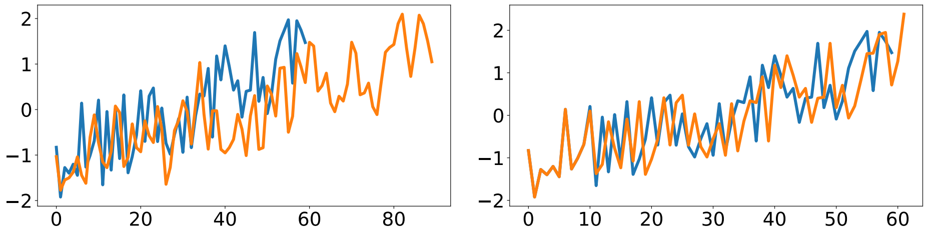
Amplitude distortion: This type of distortion refers to differences in amplitude scaling between two time series, while the intrinsic shape remains approximately the same, see Figure 1.2a.

Trend distortion: Also known as amplitude offset, it is caused by a varying mean over time, leading to distortions in the trend where each time series has an offset of different amounts in the amplitude axis, as shown in Figure 1.2b. Techniques such as z-normalization or detrending can help mitigate this type of distortion.

Time distortion: This can occur either globally or locally. Global distortion occurs when the entire time series undergoes a uniform deformation, such as time expansion or contraction (Figure 1.2c), resulting in a global shift of the entire time series. On the other hand, local time distortion occurs when certain segments of the time series undergo a local deformation, such as stretch-



(a) Amplitude distortion (b) Trend distortion



(c) Time distortion (d) Phase distortion

Figure 1.2: Common types of distortions.

ing or compressing, at a specific region or time segment, resulting in a shift in the phase component of the time series known as time warping.

Phase distortion: Refers to a shift of the time series in the time axis (Figure 1.2d). Phase distortion can be thought of as a type of time distortion, but it specifically affects the relative timing of different components of the signal without any compression or expansion. Phase shifting or phase alignment can be used to correct phase distortion.

Missing values and Outliers (residue): During the acquisition of the data, some instances might be missing or deviate significantly from the expected pattern, leading to outliers.

Out of the above-explained distortions, this work handles time series that suffer from time, phase, and amplitude distortions. Elastic measures, such as Dynamic Time Warping (DTW) explained in Section 1.1.4, are commonly used techniques to address this type of distortion.

1.1.2 Time series Analysis and Applications

Time series data is ubiquitous in our daily lives, and the analysis of this type of data can provide valuable insights into various applications. These applications range from predicting future trends to detecting anomalies in real-time systems. Time series analysis refers to a range of techniques used to analyze time series data. Depending on the specific problem or task at hand, different techniques may be applied [9].

Clustering groups similar time series together based on some distance measure or criterion without any prior knowledge or labels. Classification assigns time series to different categories or classes based on their properties or characteristics where ground truth information about the data samples is given. Forecasting makes use of past time series observations to predict future values. Anomaly detection detects unusual or unexpected patterns or events in a time series that may indicate a problem or anomaly. Retrieval searches for specific time series or patterns within a large collection of time series data. These are but a few of the many tasks and challenges that can be solved by using time series analysis methods. The specific problem at hand and the type of time series data being examined determine the strategy or method to be used.

Since real-world time series is hard to label, this work focuses on time series constrained clustering. We aim at grouping time series based on features extracted from discriminative patterns. The integration of constraints allows the clustering result to be aligned with the experts' intuition by encoding their knowledge and incorporating it into the clustering process. To account for the temporal distortions that can occur between different instances of the same phenomenon (presented in Section 1.1.1) it will be necessary to use distance measures that consider them. The different measures proposed in the literature are presented in Section 1.1.4.

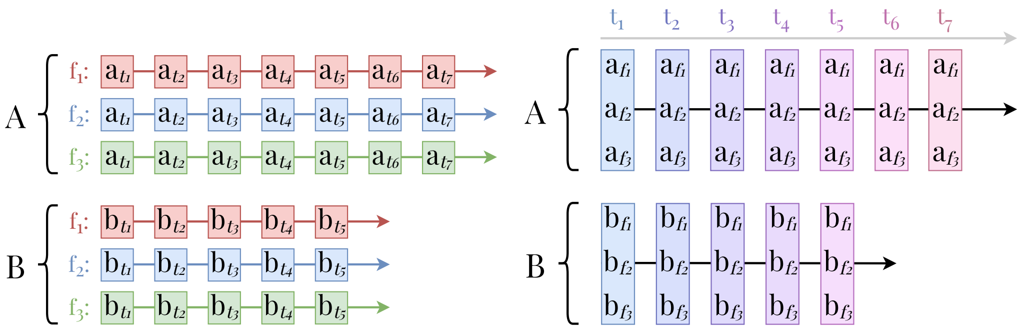
1.1.3 Time Series Representation

When studying phenomena one might be interested in observing one variable or multiple variables of the same phenomena, where these variables can be either observed synchronously or not, each using a different sensor. For example, by monitoring the growth of a crop, one can track the quality of the soil or add the variation in temperature, humidity, watering periods, level of carbon dioxide, etc. Generally in the literature, two different distinctions of the recorded observation are made, where it can be either a univariate or multivariate time series.

Definition 1 (Univariate Time Series) *A time series of length N can be defined as a sequence of real or symbolic (encoded using integers) values, represented as a vector $T \in \mathbb{R}^N$ or $T \in \mathbb{Z}^N$ (respectively) composed of a set of measurements, i.e.*

$$T = (T_1, \ldots, T_N). \tag{1.1}$$

Multivariate time series can be identified in two different ways: as a set of univariate time series, each corresponding to an (a)synchronous measurement of a different feature, or as a sequence of vectors representing the state of an object at each timestamp. Figure 1.3 illustrates these different representations for two multivariate time series, A and B. Figure 1.3a the series represented by features



(a) (b)

Figure 1.3: Different representations of multivariate time series: (a) feature-wise – Case 1 (see text) the time series is a set of univariate time series $(f_1, f_2, f_3, \ldots)$, and (b) time-wise – Case 2 (see text) the time series is a set of vectors of dimension f-features $(f_1, f_2, f_3, \ldots)$.

(characteristics, attributes, etc.), where each feature is independent of the others. In this representation, the sample's evolution is represented by F independent time series, where F is the number of features describing the object. On the other hand, Figure 1.3b combines the features in a single vector, thereby creating relationships between them by grouping them based on timestamps. In this representation, the sample's evolution is represented by a time series in which each timestamp is described by F features. It is important to note that it is possible to convert between these representations by creating vectors (which may be incomplete) or by disjointing vectors into independent time series. A formal definition of multivariate time series is given in Definition 2.

Definition 2 (Multivariate time series) *A time series of length N and F features can be defined as a sequence of univariate time series of (a)synchronous measurements, i.e. $T \in \mathbb{R}^{F \times N}$ and $T = \{T^f\}_{f=1,\ldots,F}$ (Case 1) or $T = \{T_n\}_{n=1,\ldots,N}$ (Case 2) where T^f represent time series of the f^{th} feature and T_n the vector of different features at timestamp n.*

When it comes to comparing two multivariate time series, there will be two different approaches due to the different time series representations (Cases 1 and 2). Let A and $B \in \mathbb{R}^{F \times N}$ be two multivariate time series, then the distance between them is:

$$D(A, B) = \begin{cases} \sum_{f=1}^{F} D_1(T_A^f, T_B^f), & \text{(Case 1)}, \\ \sum_{i=1}^{N} D_2(T_{A,i}, T_{B,i}), & \text{(Case 2)}, \end{cases} \tag{1.2}$$

where T_A^f denotes a time series of feature f, while $T_{A,i}$ represents a vector of different feature values at time i. The distance between A and B is determined using two distinct distance functions, D_1 and D_2. Consequently, these two distances can yield different results: D_1 is based on the comparison of individual values, whereas D_2 compares vectors. In this thesis, we will refer to (uni)multi-variate time series as time series since univariate can be considered as a special case of the multivariate with $F = 1$.

1.1.4 Distance Measures

A distance measure is a mathematical function or algorithm that quantifies the distance between two objects. This distance is calculated based on their characteristics or features [9]. Contrarily, a similarity measure is inversely proportional to a distance measure in which the more similar the objects are, the larger the similarity value and the smaller the distance is. The distance measures used in time series analysis can be classified into two main categories: metric and non-metric measures. Next, we will provide the necessary conditions for distance (metric and non-metric) and similarity measures.

To consider a distance measure a metric it needs to fulfil a set of conditions. Let D be the distance measure, and A, B, and C time series. The conditions for D to be a metric are:

Non-negativity: The distance between two objects is always non-negative, i.e.

$$D(A, B) \geq 0.$$

Identity of indiscernibles: The distance between an object and itself is zero, i.e.

$$D(A, B) = 0 \text{ iff } A = B.$$

Symmetry: The distance between two objects is the same regardless of the order in which they are compared, i.e.

$$D(A, B) = D(B, A).$$

Triangle inequality: The distance between two objects is always less than or equal to the sum of the distance between those objects and a third object, i.e.

$$D(A, B) + D(B, C) \geq D(A, C).$$

Failing to satisfy one or more of these conditions the distance measure is hence considered a non-metric measure.

On the other hand, A similarity measure S needs to respect the following conditions [10]:

Non-negativity: The distance between an object A and itself is greater than zero, intuitively the higher the value the better the similarity, i.e.

$$S(A, A) \geq 0.$$

Similarity: The similarity between an object and itself is always greater than its similarity to other objects, i.e.

$$S(A, A) \geq S(A, B).$$

Symmetry: This property is commutative, i.e.

$$S(A, B) = S(B, A).$$

Triangle inequality: the similarity between A and B through C is no greater than the direct similarity between A and C plus the self similarity of B, i.e.

$$S(A, B) + S(B, C) \leq S(A, C) + S(B, B).$$

Identity of indiscernibles: If the similarity between A and B is equal to the self-similarity of A and the self-similarity of B, then A and B are equivalent, i.e.

$$S(A, A) = S(B, B) = S(A, B) \text{ iff } A \equiv B.$$

Furthermore, different categories of distance measures can be identified [11]. These categories can be narrowed down to five groups: lock-step measures, elastic measures, embedding measures, kernel measures, and sliding measures. In this thesis, we are interested in lock-step since they are metric-based, elastic since they account for distortions, and embedding measures since they transform the time series into a lower vectorial representation highlighting important features. These are defined as follows.

Lock-step measures

A distance measure is said to be a lock-step measure if it is a one-to-one mapping, i.e. comparing the observed value at the i^{th} timestamp of time series A to the observed value at the i^{th} timestamp of time series B, each of length N, that is

$$D(A, B) = \sum_{i=1}^{t} D(A_i, B_i).$$

The most known lock-step measures are based on the L_p-norms. The L_p-norm of a vector A with N components is:

$$||A||_p = (|A_1^p| + |A_2^p| + \cdots + |A_t^p|)^{1/p},$$

where $|\cdot|$ is the absolute value.

Generally, L_p-norm is defined as the distance between a vector and the origin of the space, and when used between two vectors it is known as the Minkowski distance. The Minkowski distance between two vectors A and B is defined as:

$$D(A, B) = ||A - B||_p = \left(\sum_{i=1}^{N} |A_i - B_i|^p \right)^{\frac{1}{p}}.$$

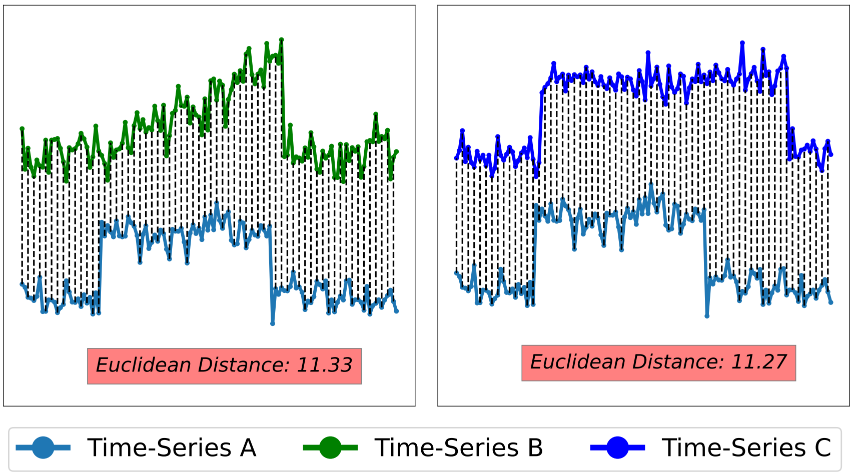


Figure 1.4: Illustration depicting that perceptually similar time series and dissimilar ones can have a close Euclidean distance. Time series A and C follow a similar trend unlike B.

When $p = 1$, the L_1-norm, also known as the Manhattan norm or Manhattan distance from the origin of the space, is defined as the sum of the absolute values of the components of A. When $p = 2$, the L_2-norm, or Euclidean norm or Euclidean distance from the origin of the space, is defined as the square root of the sum of the squares of the components of A. Measures such as Euclidean distance and Manhattan distance (Minkowski in general) satisfy the metric conditions stated previously. These measures are computationally efficient and can handle low-dimensional time series data. However, they may not be suitable for high-dimensional and complex time series data due to the curse of dimensionality. The curse of dimensionality refers to the difficulties and limitations that arise when working with data in high-dimensional spaces, in which the distance between points tends to be more uniform and the concept of proximity becomes less meaningful.

Another major drawback of these measures is that they may fail to capture the shape similarity and distortions that can be easily identified by human perception. As a result, they can measure time series as being similar even when they are perceptually different. Figure 1.4 shows three time series, taken from the CBF dataset from the UCR archive[1] [12], time series A and C belong to the same category, and time series B to another category. Perceptually, time series A and C are similar while B has a different trend. Hence one expects the Euclidean distance between A and C to be smaller than the distance between A and B, but the distance $D(A, B) = 11.33$ is very close in value to that of $D(A, C) = 11.27$, as reported in Figure 1.4, thus failing to quantify the dissimilarity between A and C.

---

[1] An open source collection of 128 univariate and 30 multivariate datasets from different domains that are commonly used in the field as a benchmark.

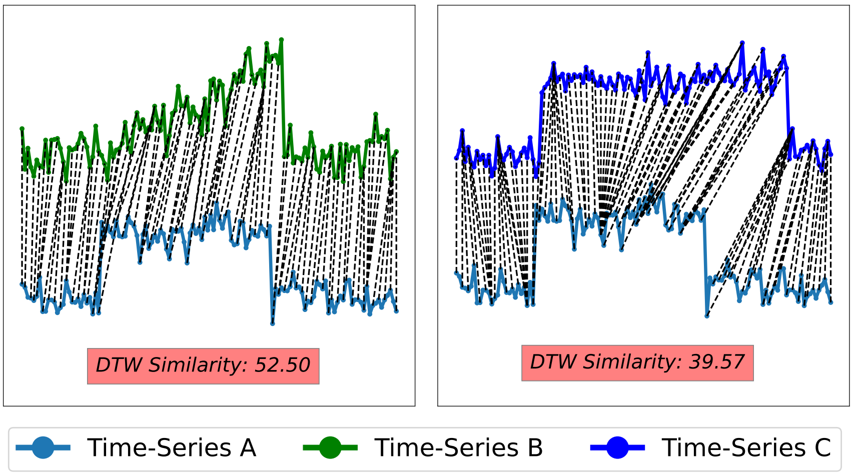


Figure 1.5: Illustration showing perceptually similar time series (A and C) and dissimilar ones (A and C) are correctly identified by DTW. Time series A and C follow a similar trend while B follows a different one.

Elastic Measures

In time series analysis, it is crucial to consider the form and distortions of the time series when determining similarity. The measures should be able to identify perceptually similar items, even when they are not numerically equal, aligning with human intuition. Elastic measures, such as Dynamic Time Warping (DTW) [13], Longest Common Sub-sequences [14], and Time-Wrap-Edit (TWE) [15] etc., allow for comparing time series based on one-to-many or one-to-none mapping of the points. The latter is possible with certain measures (such as TWE) because they are able to skip points and not assign them to any other point [16]. In comparison to other distance measures, Dynamic Time Warping has been proven to offer state-of-the-art and is one of the most used distance measures for time series data.

Figure 1.5 shows a comparison between Euclidean distance and DTW, which demonstrates that DTW, unlike Euclidean distance, captures the distortion. DTW is an elastic distance measure that uses a one-to-many mapping approach, thereby accommodating distortions in time series such as local shifting, phase shift, and time series having different lengths (time distortion). DTW aims at finding the optimal alignment (warping path) between two time series A and B, each of length t_A and t_B, respectively, by minimizing the global cost while maintaining time-continuity. Let D represent the local distance measure between individual samples of the time series, then the DTW can be defined as follows:

$$DTW(A_{1:i}, B_{1:j}) = D(A_i, B_j) + \min \begin{cases} DTW(A_{1:i-1}, B_{1:j-1}), \\ DTW(1:A_i, B_{1:j-1}), \\ DTW(1:A_{i-1}, 1:B_j), \end{cases}$$

where $A_{1:i} = A_1, A_2, \ldots, A_i$ and $B_{1:j} = B_1, B_2, \ldots, B_j$ represent sub-sequences of A and B of length i and j respectively.

Typically, the Euclidean distance or squared Euclidean distance is used, but other distance measures can also be employed. A warping path $\mathcal{P}$ is a set of pairs of indices that defines the best alignment of data points [17]. By definition, a (t_A, t_B)-warping path of length $P \in \mathbb{N}$ is a sequence

$$\mathcal{P} =< (i_1, j_1), (i_2, j_2), \ldots, (i_P, j_P) >,$$

where $(i_l, j_l) \in [1 : t_A] \times [1 : t_B]$ for $l \in [1 : P]$. A warping path must satisfy:

Boundary conditions: $(i_1, j_1) = (1, 1)$ and $(i_P, j_P) = (t_A, t_B)$.

Step-size condition: $0 \leq i_{l+1} - i_l \leq 1$ and $0 \leq j_{l+1} - j_l \leq 1$ for all $l < P$.

Monotonicity condition: $i_l \leq i_{i+1}$ and $j_l \leq j_{l+1}$ for all $l \leq P$.

The boundary conditions enforce the alignment of the first and the last elements of the time series. The step-size condition ensures that no element is omitted. Finally, monotonicity conditions enforce the path to respect the timing, if an element in A precedes other elements it should hold for B and vice versa. Algorithm 8 in Appendix A describes the algorithm for calculating and finding the warping path.

The formulation provided earlier performs well with time series having one feature (univariate). However, for multivariate time series, the independence or dependence of the features must be taken into account [18]. If features are dependent (i.e. the feature values at specific timestamps are dependent on each other) , then the warping path should be calculated on all features simultaneously, this will be termed as dependent DTW. If they are distinct (i.e. the time component across the dimensions is not important) a warping path for each feature can be found and the total cost will be the sum of the individual costs for each path , termed as independent DTW.

Let DTW_D represent the dependent approach and DTW_I the independent one. Given two multivariate time series $A \in \mathbb{R}^{N_A,F}$ and $B \in \mathbb{R}^{N_B,F}$, $DTW_I(A, B)$ finds the optimal path independently for each dimension and then adds their costs, i.e. transform the multivariate time series to F-univariate time series and find the wrapping path for each. $DTW_I(A, B)$ can therefore be written as:

$$DTW_I(A, B) = \sum_{f=1}^{F} DTW(A_f, B_f).$$

Contrarily, $DTW_D(A, B) = DTW(A, B)$, hence finds one warping path simultaneously for the F-features. The choice of DTW_D and DTW_I highly impacts the result accuracy [18]. Although DTW is a well-known distance measure that can quantify perceptual similarity with great quality, it has some drawbacks.

First, DTW becomes impractical for large datasets due to its quadratic computational complexity $O(t_A \times t_B)$. As a result, researchers have developed a number of DTW variants with the goal of reducing its computational complexity [19, 20, 21, 22]. Second, false similarity can occur when the generated path assigns one element from one time series to numerous subsequent elements from the other, or vice versa. And finally, one major drawback of DTW is that it is not a metric because it fails to satisfy the triangle inequality and uniqueness i.e. $D(x, y) = 0$ implies $x = y$. Thus, it is inapplicable to research that relies on a metric space, such as measuring constraint properties in constrained clustering, which will be discussed in Section 2.2.5. This means that an average time series cannot be calculated, which impacts all learning methods that calculate the distance between samples and the average sample, such as the K-means algorithm [23] (which will be discussed further in Section 2.2.2 in Algorithm 1).

In order to use such algorithms, a suitable averaging method should be used. Most of the averaging methods are based on pairwise averaging, i.e. a one-to-one mapping, thus failing to capture the distortion of the time series. In order to mitigate this problem, Petitjean et al. [24] proposed DTW Barycenter averaging (DBA). DBA consists of a heuristic technique that uses DTW to discover an average time series (barycenter) by repeatedly refining a randomly chosen initial barycenter until it becomes the barycenter of the set of input time series. By iteratively matching the input time series with the current barycenter and then computing an updated barycenter that is the point-wise average of the aligned time series, the approach minimizes the sum of DTW distances between each input time series and the current barycenter.

Embedding Measures

The above-described measures all work on the raw time series. However, for certain applications, mapping the time series to a different representation reveals crucial characteristics or features of the time series that are not generally apparent in the raw form, this is largely due to the diversity of the data and their characteristics [9]. Embedding measures aim to use distance measures to find a mapping of the time series to a new representation, also known as representation learning for time series. The newfound representation can be used for further analysis using simple distance measures such as L_p-distance or more sophisticated ones tailored for the representation. Another important aspect of embedding measures is their ability to map the time series to a lower-dimensional space if necessary. Generally, these measures are distance-preserving, which means that the comparison between two representations with L_p-distance approximates the comparison of the original time series using the original distance measure.

In the literature, several representations have been proposed [25], each targeting different features of the time series, such as capturing global structures, local structures, or other physical qualities. Discrete Fourier transform [26] and Discrete Wavelet Transform [27] are two representations that seek to transform time series into the frequency domain. Another form of representation is Symbolic Aggregate Approximation (SAX) [27], which quantizes the time series.

More recent studies to learn representations include Generic RepresntAtIon Leaning (GRAIL), which uses kernels, specifically the SINK kernel [28]; Similarity Preserving Representational Learning (SPIRAL), which uses DTW [29]; Shift-invariant Dictionary Learning which preserves the similarity between time series [30]; and Random Warping series that uses a Global Alignment Kernel [31]. In these methods, the algorithm aims to learn a representation (embedding) either following a supervised or a non-supervised approach where most of them employ neural networks to learn the embeddings.

Most of the aforementioned techniques are designed to transform time series into a new domain through the discovery of global shape similarities. On the other hand, time series data often exhibit local similarities. One state-of-the-art approach that uses local similarities is known as the shapelet transform, which will be explained in detail in Section 2.1.

As in other works (Learning DTW Preserving Shapelets – LDPS [32]), shapelets will be learned such that the transformation is constrained to model the DTW distance. The objective is to learn a transformation that can effectively map time series data to a metric space where distortions are accounted for. DTW plays a crucial role in handling distortions by allowing flexible alignments between time series. Meanwhile, the shapelet transform enables a mapping into a metric space, which facilitates the comparison and analysis of time series data based on relevant features. The combination of DTW and the shapelet transform is instrumental in addressing distortions and establishing a meaningful metric space for time series analysis. In our work, we extend this concept to learn such transformation but also allow the expert knowledge to influence it through the use of pairwise constraints indicating if samples are similar (should be close) or not.

1.2 Problem and Motivation

Having discussed the various aspects of time series data, including the distance measures and analysis techniques, we will now delve into the problem at hand. Analyzing time series data using supervised methods, such as classification, requires the expert to provide ground truth labels for all the data. These labels need to be perfectly known and defined, and the data provided for learning needs to be sufficiently large and of high quality. On the other hand, the complexity of the time series data with a large number of samples makes it hard and taxing for the expert to provide information on all the samples, hence, supervised approaches are difficult to use and may be unsuitable. An alternative approach, that is often used with time series data, is clustering. However, standard clustering techniques need to be adapted to take into account the distortion and complexity of the time series.

One major problem of clustering is that it is ill-posed [33], where the goal is to cluster samples based on their distances and to have homogeneous clusters without a prior definition of their meaning. For example, a set of observations can have two different clusterings depending on how the observations are perceived; suppose the observations represent a recording of running activity, the same observations can be clustered based on gender or age. Clustering is highly dependent on the approach used and the conditions specified. The results produced might not align with the expert's intuition and/or expectations. The expert's intuition is defined as what the expert expects of the results based on their prior knowledge about the domain.

To overcome this issue, a different type of clustering approach is often used, these methods integrate the expert's intuition, i.e. prior knowledge, into the clustering algorithm in order to guide the learning process. The aim of this is to converge to a clustering where the results align more closely with the expert's intuition. This is done by providing constraints to the clustering process. One way of providing constraints, which is the interest of this work, is by using instance-level constraints that specify the relationship between samples. These indicate whether two samples should belong to the same group or not. Unlike labels where the expert needs to be certain about the sample semantics, constraints can vary in their informativeness [4], meaning they can provide additional information to the learning process, or not. Similarly, constraints can also vary in their coherence [4], indicating that the provided constraints may or may not be conflicting.

When using constrained clustering approaches it is assumed that the constraints provided by the expert are both informative for the clustering algorithm and coherent, in which they encode relevant background knowledge about the instances. Measuring these properties for time series data is challenging since a metric space is generally needed. As mentioned before, using a metric space for time series data is not as effective when compared to elastic distances. So if we want to measure these properties for time series clustering data we need to map the data to a metric space in which it is possible.

In addition to guiding the clustering process and measuring the properties of the used constraints, clustering explanation is another major challenge. Recently, cluster explanation and interpretability have started to gain interest in a wide range of domains since it is important for experts to know 'why' such results are achieved and 'how'. Most of the work centres around finding concepts during the clustering process or techniques to rank the features based on their contribution to reaching a certain result.

1.3 Contribution

The contributions of this study are to address the problem of constructing a metric space that takes into account time series distortion and expert intuition in the form of pairwise relationships between samples defining whether they are similar or not. This is achieved by approximating an elastic measure (DTW) with an embedding measure under the semi-supervised paradigm by the use of pairwise constraints. As such, the elastic measure will enforce the space's invariance to distortions, and the embedding measure (shapelets) will enable the transformation of data to a metric space. Constraints are used to guide the learning process to obtain a transformation closer to the expert's intuition. This will enable the further analysis of constraint properties such as the informativeness and coherence of the constraints that require distance metrics. We also address the question of the explainability of the results, in which we leverage the interpretability of the shapelets to provide an explanation to experts of the clustering result. We provide cluster-level explanations unique to each cluster and a global-level explanation, this is achieved by ranking the shapelets according to their influence on the clustering results.

CHAPTER 2

BACKGROUND AND RELATED WORK

"If I have seen further than others, it is by standing upon the shoulders of giants."
– Sir Isaac Newton

2.1 Shapelets 20
  2.1.1 Definitions and Notations 21
  2.1.2 Shapelet Discovery Algorithms 21
  2.1.3 Learning DTW Preserving Shapelets 25
2.2 Constrained Clustering 27
  2.2.1 Clustering Definition 27
  2.2.2 Clustering Algorithms 28
  2.2.3 Cluster Validation 30
  2.2.4 User Constraints 32
  2.2.5 Measuring Constraint Properties 34
  2.2.6 Constrained Clustering Algorithms 37
2.3 Time Series Constrained Clustering 42
2.4 Conclusions 47

Time series data often exhibit valuable local patterns that can hold important information. To handle this information, distance measures for time series are commonly used to capture local similarities between them. However, these measures often do not satisfy the metric condition, making them unsuitable for metric-based algorithms. Another approach is to learn transformations, such as shapelets, that can capture local similarities. An important task in analyzing time series is grouping them based on these local similarities. Clustering, typically performed with distance metrics, is commonly used for this purpose. However, when dealing with time series data, measures that do not satisfy the metric conditions are often preferred. One challenge with clustering is its inherent ill-posed nature, as multiple results can be obtained for the same dataset and task. To address this issue, constrained clustering approaches have been proposed to incorporate expert knowledge and intuition into the clustering process. In this chapter we first discuss shapelets, which are features extracted from time series data that can be used for transforming them into a metric space. We emphasize a specific approach called Learning DTW Preserving Shapelets (LDPS) and provide the relevant literature on shapelets. Next, we introduce clustering in general, emphasizing its limitations, and focusing on the K-means algorithm as it serves as a baseline for unsupervised clustering approaches. Then, we explore constrained clustering approaches that aim to overcome the limitations of traditional clustering methods by incorporating expert knowledge into the clustering process. Finally, we delve into constrained clustering algorithms specifically for time series, offering a comprehensive explanation of the latest approaches and those used for comparison with our own approach. Introducing shapelets (specifically LDPS) and constrained clustering algorithms is essential for understanding the contributions of this work, as they employ concepts (transformation and constraints) from both approaches.

2.1 Shapelets

When we examine time series data, we may discern differences between several time series based on local or global similarities. Local similarities can be observed between patterns in subsequences of time series, whereas global similarities can be observed across the entire time series. Because local similarities focus on events occurring at specific intervals, they are more difficult to capture than global similarities. While some methods emphasize capturing global similarities during transformation, it is vital to consider local similarities because they may contain critical information, such as a time series expressing a recorded variation of temperature, where local similarities can refer to extreme weather events like thunderstorms or heatwaves and these can help in understanding the weather during a period of time to aid future forecasting. These local similarities can be identified using shapelets [34]. Figure 2.1 shows an example of shapelet S capturing a local feature of tim series T_1, which discriminates it from T_2.

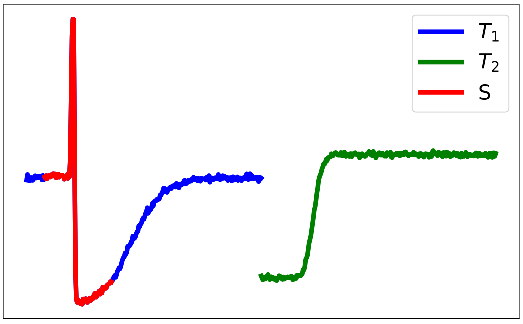


Figure 2.1: Illustration of shapelet S that captures the local similarity of time series T_1 and discriminates it from time series T_2.

2.1.1 Definitions and Notations

Definition 3 (Shapelets) *Are phase-independent discriminative sub-sequences of time series. Let a Shapelet be denoted as S having length L_k. A set of K shapelets be denoted as $\mathcal{S} = \{S_1, \ldots, S_K\}$, such that $S_k = S_{j,1:L_k}$. Although shapelets of different lengths may be included in the set $\mathcal{S}$, for simplicity, we only take shapelets of the same length into account in the formulation.*

Definition 4 (Squared Euclidean Score) *Measures the distance between a shapelet S_k and a time series sub-sequence $T_{i,w:L_S}$, such that*

$$D_{i,k,w} = \frac{1}{L_s} \sum_{x=1}^{L_s} (T_{i,w+x-1} - S_{k,x})^2. \tag{2.1}$$

Definition 5 (Euclidean Shapelet Match) *Represents the time-independent matching score between a shapelet S_k and a time series T_i (of length L_i), such that*

$$\overline{T}_{i,k} = \min_{w \in \{1:L_i - L_k + 1\}} D_{i,k,w}. \tag{2.2}$$

2.1.2 Shapelet Discovery Algorithms

The original work on shapelets was developed by Ye and Keogh [34, 7]. The approach builds a shapelet tree, which is a decision tree classifier constructed by recursively searching for discriminatory sub-sequences of time series across the entire dataset. The found sub-sequences, i.e. the shapelets, are used to split the data instances into two groups: one group comprises instances that contain subsequences similar to the shapelet, while the other group comprises instances that do not. Figure 2.2 shows the decision tree used to classify the different data samples in the arrow head dataset[1] [34].

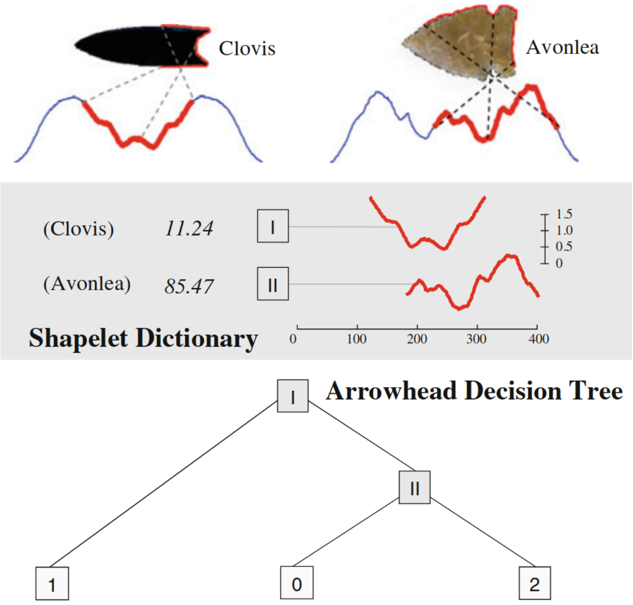


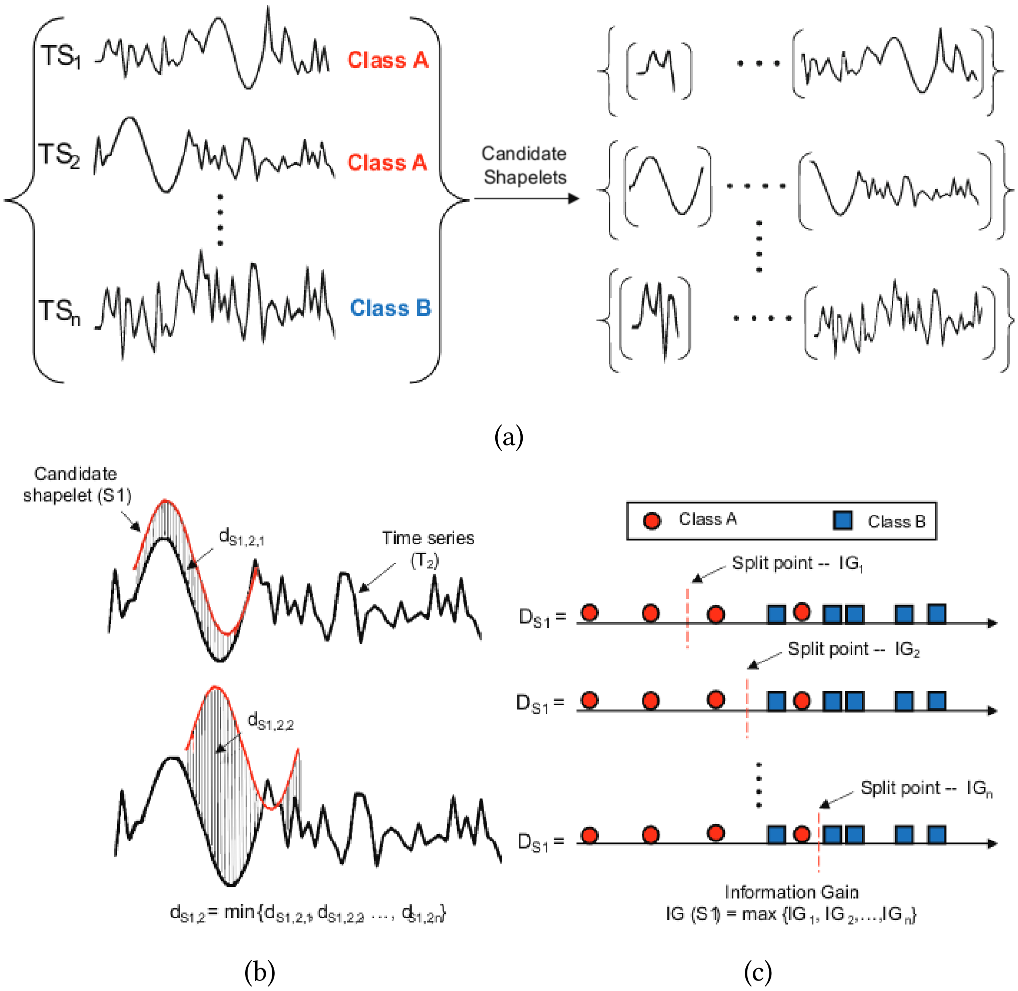
Figure 2.2: Shapelet decision tree for the arrow head dataset. The found shapelets are able to differentiate between the two types of arrow heads (Clovis and Avonlea). Since shapelet *I* has better quality (lower value indicates better separation), it is used as the root for the decision tree. Adapted from [7].

Figure 2.3 illustrates the process of finding the shapelets. The algorithm employs a brute-force search for each candidate shapelet (Figure 2.3a). To evaluate the quality of a shapelet candidate, the algorithm constructs an ordered histogram, also known as an orderline (Figure 2.3c), recording the distances between the sub-sequences and the candidate shapelet (Figure 2.3b). This histogram allows for finding a threshold that best splits it into two partitions by using a quality measure that calculates the purity of the partitions. The sub-sequences from different time series that are close to the shapelet are expected to be closer to each other, resulting in lower distance values (located on the left side of the histogram), while sub-sequences from other time series are further away (located on the right side of the histogram). The quality measure used is the Information Gain. It is a measure that identifies if a particular split yields a better partition of the data considering the class labels. After calculating the information gain for each shapelet, the shapelets are ordered in descending order. Since this approach requires searching the entire space of all possible shapelets in the dataset, it is time-consuming and memory-intensive.

Most of the literature on shapelets focuses on speed efficiency. Rakthanmanon and Keogh [36] suggest projecting the time series into a symbolic representation (SAX), which is known as FastShapelet, shown in Figure 2.4. The use of this new representation is to decrease the length of the time series and smooth the data. The discretization of the data speeds up the pruning of the shapelet candidates, which greatly improves the discovery process.

---

[1] Dataset taken from the UCR archive.


TS1
TS2
TSn
Class A
Class A
Class B
Candidate Shapelets
Candidate shapelet (S1)
ds1,2,1
Time series (T2)
ds1,2,2
ds1,2 = min{ds1,2,1, ds1,2,2 ..., ds1,2n}
Class A
Class B
Split point -- IG1
Split point -- IG2
Split point -- IGn
DS1 =
DS1 =
DS1 =
Information Gain
IG (S1) = max {IG1, IG2,...,IGn}

(a)

(b) (c)

Figure 2.3: Shapelet discovery process. In (a) the bag of all possible shapelets is generated, in (b) for each candidate the distance is calculated to the all time series, then in (c) the maximum information gain is recorded. Adapted from [35].

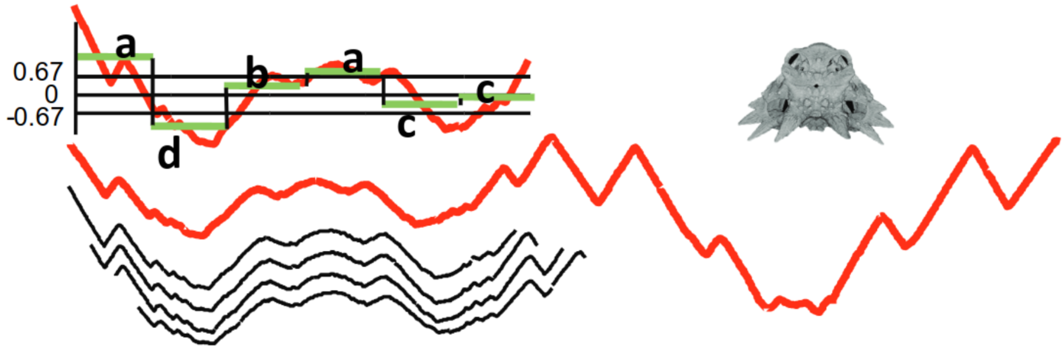

0.67
0
-0.67
a
d
b
a
c
c

Figure 2.4: Illustration of Fast shapelets principal. A subsequence of the time series is projected into a symbolic representation “adbacc” (top left). Using a sliding window technique multiple subsequences are shown in the lower left part. The time series represents the skull of a lizard. Adapted from [36].

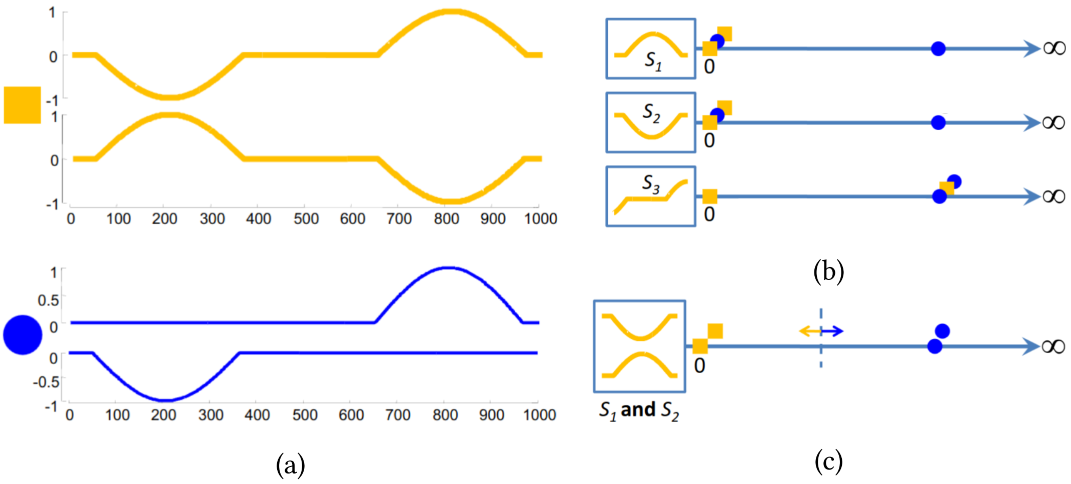


Figure 2.5: Illustration of logical shapelets. (a) shows two classes of synthetic time series. (b) Demonstrates the inability of the shapelets to effectively separate the classes (any other shapelet will fail as well). (c) Using both shapelets in the logical sense "and" (i.e. both should exist) successfully achieves the separation of the classes. Adapted from [37].

On the other hand, Mueen et al. [37] introduce logical shapelets, which are an adaptation of the shapelet tree algorithm that finds logical combinations of shapelets, e.g. AND, OR, XOR, to better handle complex problems. An example is shown in Figure 2.5, where it can be seen using a combination of the shapelet S_1 and S_2 leads to better separation of the points.

Several techniques to accelerate shapelet discovery have also been developed by exploiting graphic processing units to reduce the search time [38] or by changing the quality criterion used to determine the quality of a possible candidate shapelet [39, 40]. To overcome the exhaustive search for shapelet candidates from a bag of all possible shapelets extracted beforehand, Grabocka et al. [41] propose a supervised approach to model the shapelets as features to be learned, rather than searched. The algorithm learns the weights for the shapelets and a logistic regression (to model the classes) jointly, using a loss function that defines a linear relation between the shapelet and the time series. This increases the speed of finding shapelets but comes with a trade-off with respect to the interpretability of the shapelets.

Zakaria et al. [42] introduce the original work on clustering time series with shapelets, called unsupervised-shapelets or u-shapelets. The shapelets are chosen and extracted from a set of all possible sub-sequences by partitioning the dataset and removing the time series that are similar to the shapelets. This approach is computationally expensive, much like the earlier described techniques. A major advantage of clustering with u-shapelets is that it can work with time series of different lengths. This is not the case for the majority of approaches, which assume time series have equal length (and must be trimmed if not) [43]. However, u-shapelet clustering is able to ignore irrelevant parts of the time series data [44]. Multiple approaches have adopted the use of u-shapelets but still share the same

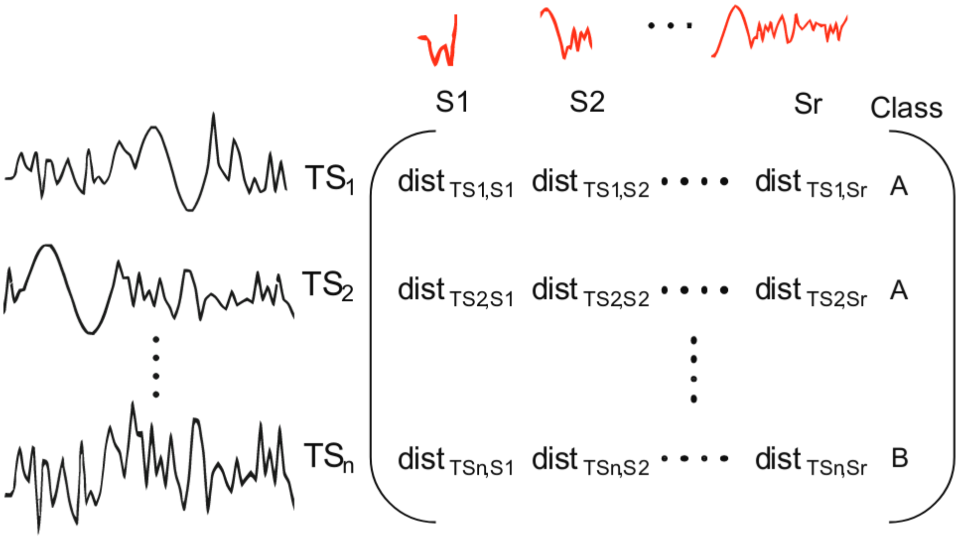

S1
S2
Sr
Class
TS1
TS2
TSn
dist TS1,S1 dist TS1,S2 dist TS1,Sr A
dist TS2,S1 dist TS2,S2 dist TS2,Sr A
dist TSn,S1 dist TSn,S2 dist TSn,Sr B

Figure 2.6: Shapelet Transform adapted from [35]. Each time series is mapped to a vectorial representation each component is the minimum distance between the time series and shapelets.

exhaustive search shortcoming.

All of the previously mentioned approaches lack the flexibility to learn a classification or clustering model and extract features or transformations separately. Separating the classification task from learning the time series representation was proposed by Lines et al. [39] and Hills et al. [45]. This work introduces the concept of the shapelet transform, the transformation maps the raw time series into a vectorial representation in which the shapelets define the representation space's bases. Figure 2.6 shows the matrix form of the transformed time series where the shapelets represent the features (columns) of the time series in the new space where the components are the distances between the time series and the shapelets.

Definition 6 (Shapelet Transform) *Is the mapping of the time series T_i using Euclidean shapelet match (Equation 5) with respect to the set of shapelets $\mathcal{S}$. The new vectorial representation is therefore:*

$$\overline{T}_i = \{\overline{T}_{i,1}, \ldots, \overline{T}_{i,K}\}, \tag{2.3}$$

where $\overline{T}_{i,k}$, $k \in [1, K]$ is the Euclidean shapelet match between the i^{th} time series and the k^{th} shapelet.

2.1.3 Learning DTW Preserving Shapelets

Inspired by the work of Grabocka et al. [41] and making use of the shapelet transform [39, 45], in which they focus only on learning shapelets without any classification or clustering, Lods et al. [32] propose an unsupervised way of learning

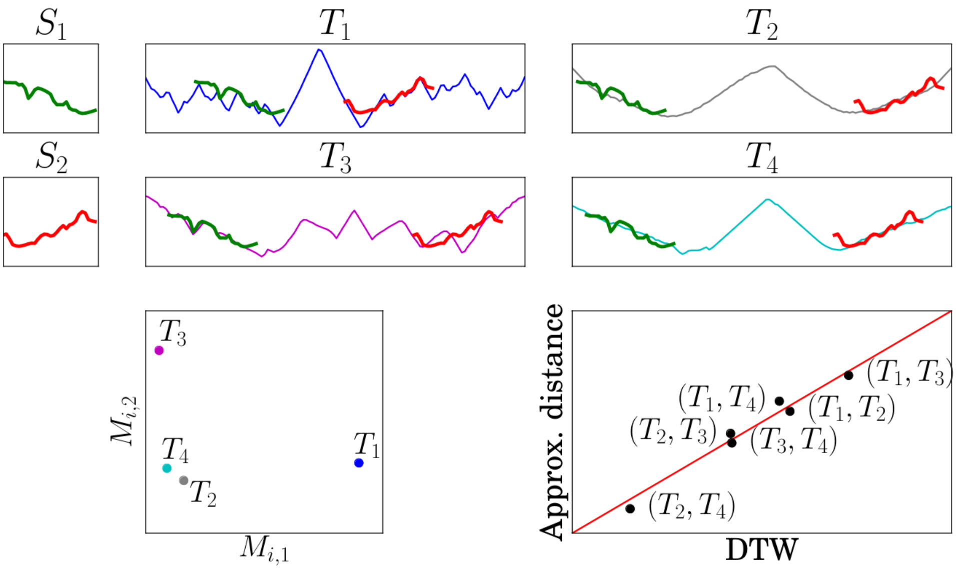


Figure 2.7: Learning DTW Preserving Shapelets (LDPS) [32]. Time series T_1 and T_2 are mapped to Euclidean space using the shapelets S_1 and S_2 where the distance between the time series in the new space approximates DTW distance. Adapted from [32].

shapelets, called Learning DTW Preserving Shapelets (LDPS). This approach learns shapelets that approximate the DTW distance in the transformed space. The DTW approximation guarantees that the shapelets learned to take into account time series distortions. Learning the shapelets is achieved by minimizing the following loss function:

$$\mathcal{L}(T_i, T_j) = \frac{1}{2}\left(\text{DTW}(T_i, T_j) - \beta \cdot \text{Dist}_{ij}\right)^2, \tag{2.4}$$

where T_i and T_j are time series from $\mathcal{T}$, β is a scaling parameter to be optimized during the learning process, and $\text{Dist}_{ij} = ||\overline{T_i} - \overline{T_j}||_2$ is the approximated DTW distance between the transformed time series $\overline{T_i}$ and $\overline{T_j}$.

Figure 2.7 presents an illustration of LDPS, where the shapelets S_1 and S_2 are used to map time series T_1 and T_2 into a two-dimensional space shown in the bottom left. The Euclidean distance between the representation of the time series is compared to the DTW between the original time series in the bottom right, it can be seen that these distances become approximately equal.

Our work builds upon LDPS and the shapelet transform, which helps with interpretability and the DTW distance approximation, in order to achieve our objective of having an interpretable transformation approximating DTW distance while respecting user prior knowledge.

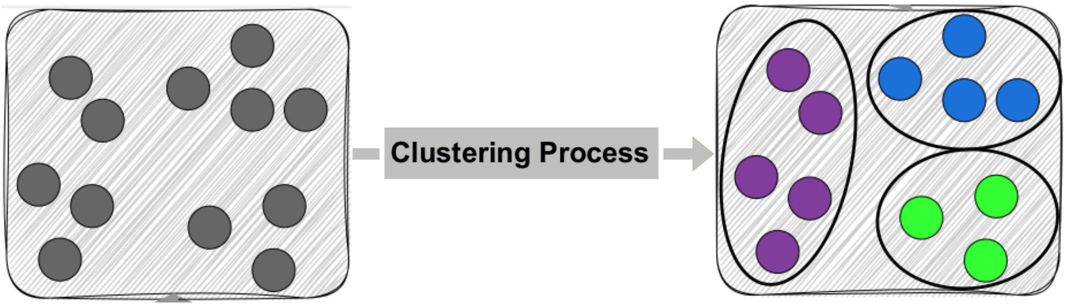


Figure 2.8: General example of clustering data points.

2.2 Constrained Clustering

Next, we will briefly introduce clustering, the necessary notations, and definitions with a non-exhaustive list of the clustering approaches found in the literature. We then focus on explaining constrained clustering in detail and how these approaches are adapted to deal with time series data.

2.2.1 Clustering Definition

Clustering is an unsupervised learning approach that groups data samples from a dataset with no predetermined labels, these groups are called *clusters*. A cluster contains the most homogeneous data objects that are as distinct as possible from the other clusters [9, 46, 47], it is achieved by discovering regularities and structures in the data instance whether they are explicitly present or not. Generally, regardless of the choice of the clustering approach, the procedure of clustering consists of four basic steps:

1. Feature selection or extraction, some algorithms work directly on the raw data while others require transformation to decrease the dimensions and generate novel features.
2. Clustering algorithm selection, depending on the data different clustering algorithms can be used. A brief description of the different clustering algorithms is presented in Section 2.2.2 and constrained algorithms in Section 2.2.6.
3. Applying the chosen clustering algorithm.
4. Cluster validation. There exists a number of approaches to validate the quality of the clustering results, which will be explored in Section 2.2.3.

In the next section, we will give a non-exhaustive explanation of the most used approaches in the literature.

2.2.2 Clustering Algorithms

Formally, given a set of samples $\mathcal{T} = \{T\}_{n,l} \in \mathbb{R}^{N \times L}$, where N is the number of data samples and L is the number of features, the clustering process aims at finding a partitioning $\mathcal{C} \in \mathbb{R}^{K \times L}$ of $\mathcal{T}$, where K is the number of clusters in $\mathcal{C}$. In order to find the optimal partitioning, different clustering techniques can be used, we first distinguish three main approaches: Hard-clustering (crisp) [48], Hierarchical-clustering [49], and Fuzzy-clustering (soft) [50].

Hard clustering algorithms cluster data samples into one and only one cluster, hence samples can represent one and only one cluster, i.e. the clusters are disjoint. For every $C_i, C_j \in \mathcal{C}$ where $i, j \in \{1, \ldots, K\}$, a clustering is said to be hard if:

$$\begin{aligned} \bigcup_{\forall i} C_i = \mathcal{T}, \\ C_i \neq \emptyset, \\ C_i \cap C_j = \emptyset, i \neq j. \end{aligned}$$

Contrarily, fuzzy clustering may allow one sample to belong to different clusters with different degrees of membership [51], such as fuzzy C-means [52]. Let $u_{i,j}$ define the membership of the j^{th} object in the $i^t h$ cluster, fuzzy clusters satisfies:

$$\sum_{i=1}^{K} u_{i,j} = 1, \forall j \quad \text{and} \quad \sum_{j=1}^{N} u_{i,j} < N, \forall i.$$

Hierarchical clustering creates a dendrogram, which is a tree-like structure that captures the relationship between clusters. The dendrogram is constructed in a way such that clusters are successively merged or split. By cutting the dendrogram at a specific level, disjoint clusters can be obtained. Let $\mathcal{H} = \{H_1, \ldots, H_Q\}$ represent the dendrogram to be obtained where $Q \leq N$ such that $\forall i, j \neq i, m, l = 1 \ldots Q$, we have:

$$(C_i, C_j) \subseteq H_m \times H_l|_{(m>l)} \Rightarrow C_i \subseteq C_j \quad \text{or} \quad C_i \cap C_j = \emptyset.$$

Many clustering algorithms have been proposed, each belonging to one of the previously mentioned approaches [53]. These include partition [54], density-based [55], spectral clustering [56], conceptual [57], deep clustering using neural networks [58], collaborative [59], ensemble [60], and interactive clustering techniques [61].

Partitioning algorithms involve specifying an initial number of disjoint clusters and iteratively reallocating objects among the clusters until convergence is reached [52, 62]. These algorithms typically determine all clusters at once. One of the most widely used heuristic methods for partitioning is the K-means algorithm [47, 63], it is one of the simplest and most effective algorithms. As such, we explain the K-means algorithm in detail and use it in this work. K-means, as described in Algorithm 1, starts by assigning K random cluster centers. Then each

Algorithm 1 K-MEANS ALGORITHM

```
Input:: data points $x_1, x_2, \ldots, x_n$, number of clusters : $k$
Output: Cluster assignments for each data point $\{C_1, \ldots, C_K\}$
Initialize cluster centroids $\mu_1, \mu_2, \ldots, \mu_k$ randomly
while not converged do
    // Assign each data point $x_i$ to its closest centroid:
    for each $x_i$ do
        $c_i \leftarrow \arg\min_{j \in \{1,\ldots,K\}} ||x_i - \mu_j||^2$
    end for
    // Update each centroid as the mean of the data points assigned to it:
    for each $C_j$ do
        $\mu_j \leftarrow \frac{1}{|C_j|} \sum_{xi \in C_j} xi$
    end for
end while
return $\{C_1, \ldots, C_K\}$
```

sample is assigned to the cluster whose centre is nearest, to form K clusters. Then the refinement of the centroids starts by recalculating their centre using an averaging method (such as the arithmetic mean) and re-assign the points to the closest updated centroid; this process is repeated until there is no improvement in the centroids. The complexity of the algorithm is $O(nkt)$ which allows it to handle large datasets. Note that K-means requires the computation of the cluster centre making it valid for the data types where this computation is possible. In addition to this, since the algorithm calculates the mean of the objects it is prone to outliers and noise in the data [63], in these cases other approaches can be used, for example, mean-shift (kernel-based) [64], etc.The performance of K-means is influenced by the distinctness and density of the samples used for clustering, which affects its ability to accurately capture the underlying structure.

Density-based clustering identifies clusters based on density conditions, searching for regions with high density that are separated by low density, such as 'Density-Based Algorithm for Discovering Clusters' (DBSCAN) [65]. Spectral clustering uses the concept of spectral proprieties of a matrix to transform the objects into a space where the new information encodes the similarity between the objects, most techniques under this approach make use of the Laplacian graph [66]. Conceptual clustering finds clusters that represent a concept (such as COBWEB algorithm [67, 68], MFI-Kmeans [5], and FCLM [6]). Ensemble and collaborative clustering both use multiple algorithms to reach a consensus. Where in ensemble approaches each algorithm reaches a clustering result independently [69] and these are combined. And in collaborative clustering, the algorithms collaborate and share information during the clustering process [70, 71, 72] as the information is considered to be complementary [73]. The methods can be of different nature or the same with different initialization [74, 75]. Collaborative clustering can be considered as an extension of ensemble clustering, by adding a refinement step before the unification step, such as in SAMARAH [70, 71], where each clustering algorithm refines its result according to all other clustering algorithms until all results are strongly similar; the final unification can be achieved through a voting algorithm. In active clustering [76, 77, 78], the user provides feedback to the algorithm to determine which points belong to each cluster.

Most of the algorithms mentioned in this section find the clustering by using optimization criteria to assert the homogeneity and separation of a cluster [54]. The majority of the criteria are based on similarity or dissimilarity measures. These influence concepts such as the diameter of a cluster, the separation between clusters, the within-sum of dissimilarities of a cluster, the within-sum of squares of samples belonging to a cluster, etc. [79].

2.2.3 Cluster Validation

The quantitative evaluation of the clustering method is known as cluster validation and the methods used to perform this evaluation as cluster validity methods or indices. Clustering validation falls into three families of criteria: internal, relative, and external [47, 53, 80]. The internal criteria evaluate the quality of a clustering result using only quantities and features inherited from the algorithm and the dataset. Most measures in the internal criteria are based on measuring the intra-cluster compactness and inter-cluster separation. The external criteria compare the clustering output to the ground truth labels, hence the data labels should be known. On the contrary, relative criteria seek to identify the optimal clustering conditions of an algorithm by comparing different clustering results obtained under varying assumptions and parameters. This approach aids in determining the most suitable parameters for a clustering algorithm.

Ezugwu et al. [47] give a detailed explanation of the different criteria, and the following is a non-exhaustive list of the most used validation criteria:

- Internal criteria:
  - Sum of squared error is defined as $\text{SSE} = \sum_{i=k}^{K} \sum_{\forall x_i \in C_k} ||T_i - \mu_k||^2$, where μ_k is the mean of the cluster k. It is one of the simplest and widely used measures in clustering [79].
  - Silhouette index [81] is based on the intra-cluster compactness and the inter-cluster separation of the clusters. It measures for each object how well the object belongs to its own cluster compared to the other clusters. Formally the silhouette index for the i^{th} object T_i in cluster C_k is defined as:

    $$s(i) = \sum \frac{b(i) - a(i)}{\max(a(i), b(i))},$$

    where $-1 \leq s(i) \leq 1$ such that $a(i) = \frac{1}{|C_k|} \sum_{\forall T_j \neq T_i \in C_k} D(T_i, T_j)$ is the average distance between the T_i and the remaining objects within the cluster C_k, hence measuring the compactness. The separation T_i from all other cluster $C'_k \neq C_k$ is given by $b(i) = \min_{\forall k' \neq k} \frac{1}{|C'_k|} \sum_{\forall T_j \in C'_k \neq T_i \in C_k} D(T_i, T_j)$. The average silhouette index measures the performance of a clustering algorithm, and a value of one (the measure's maximum value) indicates the best performance.

- External criteria:
  - The Rand Index (RI) is a similarity measure between the clustering results $\mathcal{C}$ and the dataset underlying structure $\mathcal{P}$ [82]. It is defined as the number of similar assignments of point-pairs normalized by the total number of point-pairs [83], i.e.

$$\text{RI} = \frac{\left[\binom{N}{2} - \frac{1}{2}\left[\sum_i \left(\sum_j n_{ij}\right)^2 + \sum_j \left(\sum_i n_{ij}\right)^2 - \sum\sum n_{ij}^2\right]\right]}{\binom{N}{2}},$$

where $n_{ij} = |C_i \cap P_j|$ represents the number of objects that are placed in cluster C_i having underlying structure P_j. N is the total number of samples, $i = 1, \ldots, |\mathcal{C}|$ and $j = 1, \ldots, |\mathcal{P}|$. The value of the Rand Index falls between zero and one, a major problem with this measure is that the expected value for two random clustering is not constant.

  - Adjusted Rand Index [83, 84] overcomes the disadvantage of the Rand Index by normalizing with respect to the expected value of the Rand Index,

$$\text{ARI} = \frac{\sum_{i,j}\binom{n_{ij}}{2} - \sum_i \binom{n_i}{2} \sum_j \binom{n_j}{2} / \binom{N}{2}}{\frac{1}{2}[\sum_i \binom{n_i}{2} + \sum_j \binom{n_j}{2}] - \sum_i \binom{n_i}{2} \sum_j \binom{n_j}{2} / \binom{N}{2}},$$

where $n_{ij} = |C_i \cap P_j|$, $n_i = \sum_j n_{ij}$, and $n_j = \sum_i n_{ij}$. N is the number of samples, $i = 1, \ldots, |\mathcal{C}|$ and $j = 1, \ldots, |\mathcal{P}|$.

  - Entropy quantifies randomness [85]. The entropy of a single cluster is defined as $E_j = \sum_i p_{ij} \log(p_{ij})$, where p_{ij} is the probability of object T_i being in class C_j citezhao2001criterion. The total entropy is:

$$E = \sum_{j=1}^{|\mathcal{C}|} \frac{n_j}{n} \sum_i p_{ij} log(p_{ij}).$$

  - Normalized Mutual Information (NMI) is the mutual information normalized by the entropy of the predicted labels and the ground truth labels [74]. It indicates the amount of information one can extract from a distribution regarding a second one. It is defined as:

$$\text{NMI}(\mathcal{C}, \mathcal{P}) = \frac{I(\mathcal{C}, \mathcal{P})}{\sqrt{E(\mathcal{C})E(\mathcal{P})}}, \tag{2.5}$$

where $I(\mathcal{C}, \mathcal{P}) = E(\mathcal{C}) - E(\mathcal{C}|\mathcal{P})$ is the mutual information between the predicted and ground truth labels.

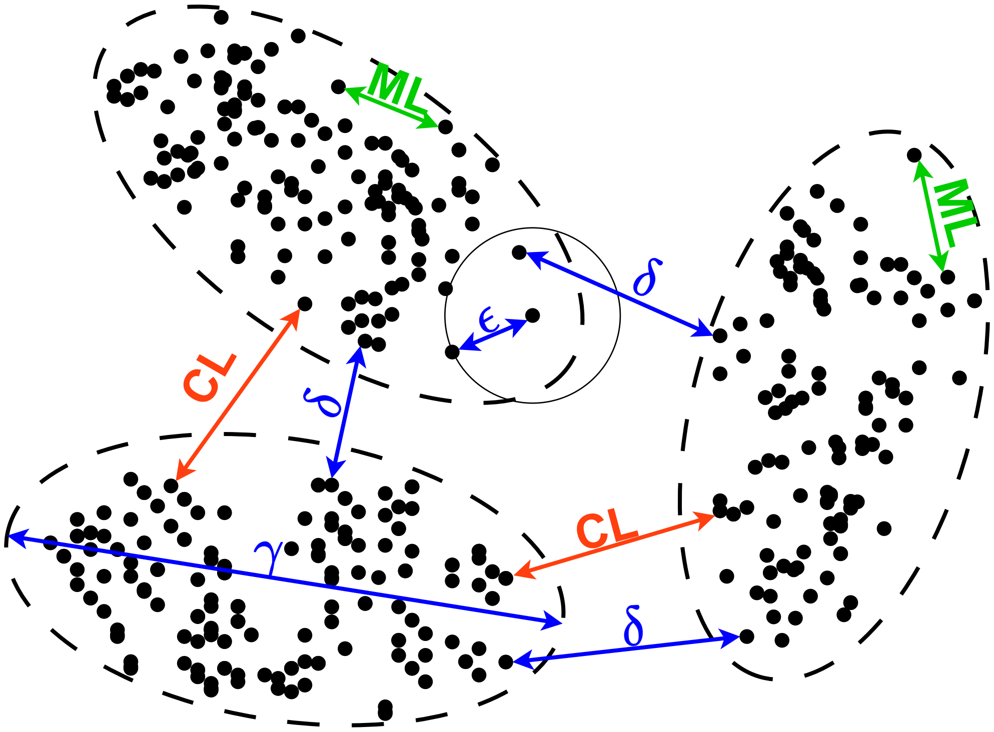


Figure 2.9: Examples of user constraints, showing the different types of constraints. The figure shows three different clusters (dashed ellipsis) of instances (black points). The constraints shown are Instance level (must-link 'ML' in green and cannot-link 'CL' in red), and Cluster level constraints (δ, ϵ, and γ in blue) representing the minimum separation, the neighbourhood radius, and the maximum diameter respectively. The ML and CL constraints specify the relation between objects. ML indicates that the objects should be within the same cluster while CL should not. Adapted from [1].

2.2.4 User Constraints

Following the definition of clustering, it can be inferred that the problem is ill-posed due to the ambiguity of data and algorithmic bias. It is rare to have one clustering solution for the same dataset, i.e. with different initial conditions the results will vary. To overcome these limitations, constrained clustering approaches have been developed. Very often the expert has some intuition of the data that can help in guiding and mitigating the ill-posed nature of clustering. Constrained clustering is an approach that leverages this information by integrating expert knowledge into the clustering process in the form of constraints. It aims to balance the bias of the optimization criteria and the constraints provided by the expert. There are different ways of providing constraints for the clustering algorithm. The most known are instance-level constraints, introduced by Wagstaff and Cardie [86], and cluster-level constraints, as in the work of Basu et al. [87], shown in Figure 2.9.

Instance-level constraints are constraints that specify the relationships between individual data points. These fall into two types: must-link and cannot-link constraints. Must-link constraints specify that two data points must be assigned to the same cluster, while cannot-link constraints specify that two data points cannot be assigned to the same cluster. Instance-level constraints are relatively easy to define and incorporate into the clustering process and can provide fine-grained control over the clustering results.

Definition 7 (Instance-level Constraints) *Let C_k be the k^{th} cluster, ML be the set containing time series indices for those connected by must-link constraints, and CL the set for those connected by cannot-link constraints. Thus, $\forall\ T_i, T_j$ such that $i, j \in \{1, \dots, N\}$ and $i \neq j$, we have*

$$ML = \{(i,j) | \forall\ k \in \{1, \dots, K\}, T_i \in\ C_k \Leftrightarrow T_j \in C_k\}, \tag{2.6}$$

$$CL = \{(i,j) | \forall\ k \in \{1, \dots, K\}, \neg(T_i \in\ C_k \wedge T_j \in C_k)\}. \tag{2.7}$$

Instance level constraints encode interesting properties. Must-link constraints are symmetric, reflexive, and transitive, which means if objects a and b are connected by an ML constraint along with b and c, this infers that a and c are also connected by a must-link constraint. Although cannot-link constraints do not have such properties, additional cannot-link constraints can be inferred from the must-link constraints.

Cluster-level constraints are constraints that specify conditions on the cluster's number, size, and individual objects belonging to the cluster. The following is a list of cluster-level constraints.

- The number of clusters K.
- The size of the cluster, constraining the capacity of a cluster by expressing the maximal or the minimal limit on the number of objects in each cluster. Let α represent the minimum number of elements in a cluster and β the maximum number of elements. Hence, the minimal constraint can be formulated as $\forall k \in \{1, \dots, K\}\ |C_k| \geq \alpha$ and the maximal constraint as $\forall k \in \{1, \dots, K\},\ |C_k| \leq \beta$.
- The maximum diameter of the cluster γ. This constraint provides an upper bound on the diameter of each cluster. Formally, $\forall k \in \{1, \dots, K\},\ \forall T_i, T_j \in C_k,\ D(T_i, T_j) \leq \gamma$.
- The split between clusters δ, indicates the minimum separation between the clusters $\forall k, k' \in \{1, \dots, K\},\ k \neq k',\ \forall T_i \in C_k,\ \forall T_j \in C_{k'},\ \ D(T_i, T_j) \geq \delta$.
- The neighbourhood of objects ϵ specifies the minimum radial distance for an object to have at least one other object in the same cluster. It was introduced

by Davidson and Ravi [88]. $\forall k \in \{1, \ldots, K\},\ \forall T_i \in C_k,\ \exists T_j \in C_k,\ T_i \neq T_j,\ \ D(T_i, T_j) \leq \epsilon$. This can be generalized to have at least m objects within the neighbourhood of the object.

In addition to the mentioned constraints, the user can take advantage of the features that describe the instances and provide constraints on the properties of the clusters [89]. Such as cardinality constraints, which constrain the number of objects with a specific property within a cluster. Another type of property constraint is the density constraint. Unlike cardinality constraints which are applied to the entire cluster, density constraints are applied to a subset of the objects within a cluster. Geometric constraints can also be provided bounding the clusters to some geometric property. Finally, complex logical constraints can also be provided, expressing logical combinations of the constraints. Note that combinations of different types of constraints can be used [88] and hence providing more information.

Note that for instance-level constraints the assumption is made that they are informative and coherent with one another and do not conflict or overlap. This can be challenging in practice, as it can be difficult to measure the informativeness and coherence of user-provided constraints. In Section 2.2.5 we present a detailed review of the definitions of informativeness and coherence for pairwise constraints.

In our work, we argue that constraints can be used in learning a transformation that reflects the expert intuition in which the space will assert that samples perceived as similar by the expert are closer to each other and those that are not similar are sufficiently far from each other. Section 3.2 provides a detailed explanation of the importance of the constraints and how we integrate them into our work.

2.2.5 Measuring Constraint Properties

The quality and usefulness of instance-level-based constrained clustering depends highly on the constraint set provided [90]. Wagstaff et al. [90] and Davidson et al. [4] show that constraint sets can increase or decrease clustering performance, hence the urgency to formulate and identify some properties to measure which constraint sets are useful or not. The authors proposed two approaches to measure the importance of the constraints, one is algorithm dependent called 'informativeness'—which is also known as inconsistency [4]—and the other is algorithm independent, named 'coherence'. Coherence was later reformulated, to take into account all different aspects of conflicts by Lampert et al. [1].

Informativeness

Definition 8 (Informativeness) *Measures the amount of information added to the algorithm bias through the constraints. Hence, it measures the number of constraints*

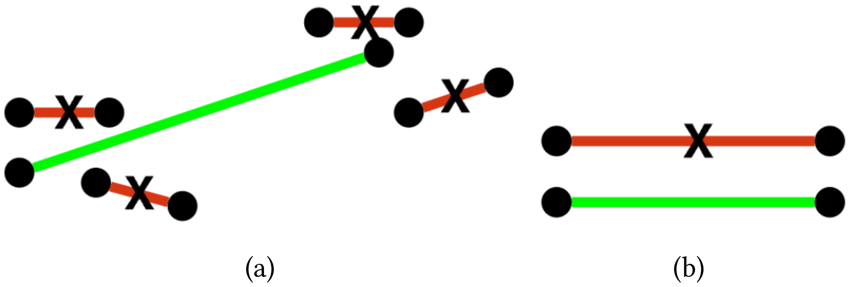


Figure 2.10: Illustration of informativeness (a) and coherence (b). The red lines with X indicate cannot link constraints while the green line indicates must link constraints. Adapted from [4].

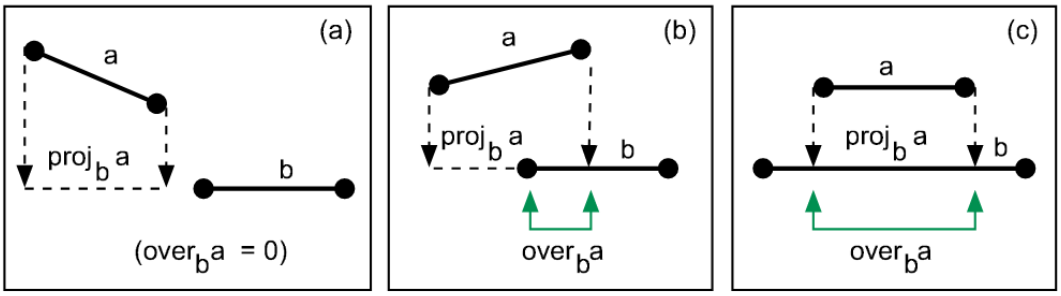


Figure 2.11: Illustration of calculating constraint coherence, showing three cases of computing the projected overlap between constraints a and b; where they constraint the points (a_1, a_2) and (b_1, b_2) respectively. The points indexed 1 always appear to the left of the other. Adapted from [4].

that the clustering algorithm cannot satisfy using its default bias, i.e. without any prior knowledge [90].

Given an incomplete set of constraints Γ, i.e. that does not represent a unique partitioning, and an algorithm $\mathcal{A}$, we generate a partitioning $\mathcal{P}_{\mathcal{A}}$ by applying $\mathcal{A}$ on a dataset $\mathcal{T}$ without any constraints. The informativeness $\mathcal{I}_{\mathcal{A}}(\Gamma)$ is hence calculated as the fraction of constraints in Γ that are unsatisfied by $\mathcal{P}_{\mathcal{A}}$ [90]:

$$\mathcal{I}_{\mathcal{A}}(\Gamma) = \frac{1}{|\Gamma|} \sum_{\gamma \in \Gamma} \text{unsat}(\gamma, \mathcal{P}_{\mathcal{A}}), \tag{2.8}$$

$$\text{s.t. unsat}(\gamma, \mathcal{P}_{\mathcal{A}}) = \begin{cases} 1 \ \mathcal{P}_{\mathcal{A}} \text{ does not satisfy } \gamma, \\ 0 \text{ otherwise,} \end{cases}$$

where unsat returns whether a constraint γ is satisfied by the partitioning $\mathcal{P}_{\mathcal{A}}$ or not.

Coherence

Definition 9 (Coherence) *Evaluates the degree to which the constraints are consistent and in agreement with one another, with respect to a distance metric. It is model-independent in that it does not require any partitioning and is computed directly on the constraints and data.*

The motivation behind this measure is that the must-link, ML, (cannot-link, CL) constraints impose an attractive (repulsive) force within the vicinity of the constrained points and in the direction of the line connecting the pair of points. To calculate the coherence (as defined in [1]), the constraints are treated as line segments and the projected overlap (by projecting each constraint onto the others) is measured.

Let $\vec{a}$ and $\vec{b}$ be vectors connecting the points constrained by $\boldsymbol{a}$ and $\boldsymbol{b}$ respectively, i.e. (a_1, a_2) and (b_1, b_2), see Figure 2.11. We first project the points bound by constraint $\boldsymbol{a}$ onto the line that is defined by the points bound by constraint $\boldsymbol{b}$, such that:

$$\begin{matrix} a_1' = ((a_1 - b_1)\boldsymbol{e})\boldsymbol{e} + b_1 \\ a_2' = ((a_2 - b_1)\boldsymbol{e})\boldsymbol{e} + b_1 \end{matrix}, \quad \text{where } \boldsymbol{e} = \frac{\boldsymbol{b}}{|\boldsymbol{b}|}.$$

Hence, the projection of the points to the 1D space can be written as:

$$a_i'' = a_i'\boldsymbol{e}, \quad b_i'' = b_i\boldsymbol{e}, \qquad \text{where } i \in \{1, 2\}.$$

Next, the points are sorted such that $a_1'' \leq a_2''$ and $b_1'' \leq b_2''$. If this assumption is satisfied, the overlap of constraint $\boldsymbol{a}$ on constraint $\boldsymbol{b}$ can be written as:

$$o_a^b = \max\{0, \min\{a_2'', b_2''\} - \max\{a_1'', b_1''\}\}.$$

According to the stated definition, two constraints are coherent if there is no overlap between them. Formally, this can be written as:

$$\mathrm{coh}_{cm} = \begin{cases} 1, & \text{if } o_c^m = 0 \text{ and } o_m^c = 0, \\ 0, & \text{otherwise.} \end{cases}$$

Finally, the overall coherence of a set of constraints is defined as the fraction of coherent constraints within the set, such that:

$$\mathrm{COH}(C) = \frac{\sum\limits_{c \in \mathrm{CL}, m \in \mathrm{ML}} \mathrm{coh}_{cm}}{|\mathrm{CL}||\mathrm{ML}|}.$$

In addition to the instance-level properties, we can also measure the satisfaction of constraints and the average distance between must-link and cannot-link constraints [1]. Constraint satisfaction indicates how many constraints have been fulfilled by the algorithm and not violated. The latter measures the ratio between the average distance of the must-link pairs and the cannot-link pairs, and therefore measures whether the must-link pairs are closer together than the cannot-link pairs globally, i.e. the ratio is less than one. If the value is one, this indicates that the distance between cannot-link pairs and must-link pairs is the same which means that the constraints are not well represented in the space. Note that this measure provides a high-level overview of the general distribution and an investigation of individual constraints might be needed to draw more detailed conclusions.

2.2.6 Constrained Clustering Algorithms

Earlier we described the concept of constrained clustering and how constraints can model an expert's knowledge and intuition so that the results are closer to their needs. In this section, we will describe some of the algorithms that have been adapted to or developed for constrained clustering, such as K-means, metric learning, spectral clustering, etc.

Let us first highlight the two main approaches when working with constraints: hard and soft constraint satisfaction. In the hard satisfaction approach, all the constraints must be satisfied in the resulting clustering. If any constraint is violated, the algorithm will not converge to a solution. However, to mitigate this issue, some algorithms relax this requirement and allow for the clustering algorithm to violate some constraints, giving it some freedom to follow its bias. This is referred to as the soft satisfaction approach.

Most of the proposed clustering algorithms in the literature can be extended to integrate constraints; the constraints integrated are either instance-level constraints (the majority of cases) or cluster-level constraints. Some of these algorithms are fast and find approximate solutions and hence don't guarantee the satisfaction of all constraints and therefore fall under soft constraint satisfaction. Constrained clustering algorithms fall into the same categories of clustering algorithms mentioned in Section 2.2.2 (partition clustering, metric learning, spectral graph theory, density based, etc). Next, we provide a non-exhaustive review of the most used techniques in the literature highlighting the Constrained K-means algorithm (COP-Kmeans), as it is used as a reference comparison for the constrained approaches in this thesis (Section 3.3).

Constrained K-means, are approaches that extend K-means into a constrained version using instance-level constraints either by enforcing pairwise constraints or by using the constraints to define penalties in the objective function [91]. A multitude of algorithms adopts the approach of enforcing constraint satisfaction, such as COP-COBWEB [86] which extends the original COBWEB algorithm. Another extension called COP-Kmeans was proposed by Wagstaff et al. [92], it chooses a reassignment that does not violate any constraints at each iteration, hence it is a greedy approach. An improved version of COP-Kmeans was proposed by Tan et al. [93] and Rutayisire et al. [94], which tries to solve the problem of constraint violation by either modifying the assignment order, based on either a measure of certainty computed for each instance or a sequenced assignment of cannot-link pairs. Basu [95] proposed two derivatives, the first being Seed-Kmeans and the second being Constrained-Kmeans. These approaches use the concept of seeds to identify the clusters, and the difference is the possibility to change the centres or not. A different approach was taken by Huang et al. [96] for A Semi-supervised Clustering Algorithm Based on Must-Link Set (MLC-KMeans). The authors propose to use the must-link constraints to introduce assistant centroids that help in assigning instances to clusters. Assistant centroids are based on the centroid of the must-link set which is a set containing all the instances linked by a must-link constraint for a given cluster.

A recent work proposed by Vouros and Vasilaki [97] called ‘A semi-supervised sparse K-means algorithm’, PCSKM, uses constraints in ‘sparsity K-means clustering’ (SKM) [98] to guide the learning process. This preserves the sparsity capabilities of the SKM algorithm, i.e. it finds clusters that are in distinct subsets of the features. PCSKM uses the constraints to penalize the clustering process if a constraint is violated. In order to handle high-dimensional sparse data, Tang et al. [99] proposed the SCREEN method for constraint-guided feature projection. The approach learns a lower-dimensional space such that the distance between any pair of instances is minimized if linked by a must-link constraint or maximized if linked by a cannot-link constraint. Afterwards, the spherical K-means [100] algorithm is used to avoid the violation of cannot-link constraints. This work has some similarities to our proposition where we try to minimize the distance between similar samples under must-link constraint and maximize the distance between dissimilar samples under cannot-link constraint. Note that this approach is not suited for time series and is therefore not included as a comparison.

Algorithms that define penalties on the objective function make a balance between clustering performance and satisfying as many constraints as possible. Such as the work of Demiriz et al. [101] by incorporating dispersion and impurity measures of the objective function. Davidson and Ravi [88] in their work Constrained Vector Quantization Error (CVQE), penalize constraint violation. If a must-link is violated then the penalty is the distance between the centriods of the clusters that the samples are in, and if a cannot-link is violated then the distance between the centriod of the cluster they are in and the nearest centriod. Later the work of Pelleg and Baras [102] improves CVQE, called linear-time CVQE. It incorporates the coordinates of the involved instances in the penalty calculations, while also avoiding the need to check all possible assignments for the cannot-link constraints.

A combined objective function, in which the first term is the sum of the total squared distances between the points and their cluster centres and the second term the cost of violating any pairwise constraints, was proposed by Basu et al. [103] and termed Active Semi-Supervision for Pairwise Constrained Clustering (PCK-Means). PCK-Means proposes an active learning approach to provide constraints to the algorithm based on the farthest-first traversal scheme, they showed that using this scheme gives better performance than when using random ones but it suffers from outliers and noisy constraints. The work of Ganji et al. [104], Lagrangian constrained clustering, formulates the problem into an unconstrained optimization problem. The objective is to minimize the Euclidean distance between the samples while also penalising the violation of cannot-link constraints. To account for must-link constraints, instances subject to these constraints are aggregated into super-instances. Super-instances are a mapping of must-linked instances where any two such instances are mapped in the same super-instance. The method employs Lagrangian relaxation with increasing penalties to address unsatisfied constraints during the iterative clustering process. The extension of Fuzzy C-means to include constraints was proposed by Grira et al. [105], they introduce a penalty term into the loss function that measures the cost of violating pairwise constraints. Grira et al. [106], propose an active approach for constrained-based fuzzy C-means, called Active Fuzzy Constrained Clustering (AFCC). AFCC minimizes a competitive agglomeration cost function with a fuzzy term that corresponds to the pairwise constraints, the constraints are selected and provided following an active mechanism. Beside pairwise constraints, Ge et al. [107] put constraints on the number of objects in a cluster and the minimum variance of a cluster. The work of Demiriz et al. [108] integrates the minimal size constraints. While Banerjee and Ghosh [109] constrained the clusters to be balanced. In order to avoid empty clusters, Bradley et al. [2] adds capacity constraints that specify the minimum number of samples in a cluster to avoid empty clusters or those with few samples.

Metric learning learns a distance metric for the input data to discriminate between the samples in the input space [110, 111]; the learned metric preserves the distance between the training data. Since most algorithms rely on a distance measure, metric learning can be considered as a prepossessing step, that is finding a good metric will increase the performance of the subsequent algorithm.

Formally, metric learning learns the distance D_M which is equivalent to learning a distance map matrix M, in order to satisfy the conditions of a metric. M needs to be a positive semi-definite real-valued matrix. For constrained clustering, Mahalanobis distance parameterized by matrix M is usually used, i.e. $D_M(T_i, T_j) = ||T_i - T_j||_M$. using Mahalanobis distance enables the measure to take into account correlations between attributes [112].

The integration of constraints in metric learning means that the metric reduces the distance between must-link pairs and the distance between cannot-link pairs increases, this will give similar points a higher chance to be in the same cluster while increasing cluster separation. This concept is similar to what we use in our approach Constrained DTW Preserving Shapelets explained Chapter 3 where we also aim at decreasing the distance between similar points and increasing it between dissimilar ones. Xing et al. [113] proposed to learn a metric constrained by must-link and cannot-link constraints by formulating the problem as an optimization problem. The objective function is the minimization of the distances for the must-link pairs under the condition that the sum of the distances for the cannot-link pairs is larger than a constant c, i.e.

$$\min \sum_{ML(T_i,T_j)} D^2_M(Ti, T_j), \quad \text{s.t} \sum_{CL(T_i,T_j)} D_M(Ti, T_j) \geq c.$$

In order to avoid over-fitting due to learning only on constrained objects, unlabeled data can be introduced. Several methods are proposed to learn a distance while including unlabeled data. Such works of Klein et al. [110] on Euclidean distance and shortest path, Bar-Hillel et al. [114, 115] on Mahanalobis distance, Cohn et al. [116] on Kullback-Leibler divergence, Bilenko and Mooney [117] on string-edit distance, and Hoi et al. [118] on Laplacian regulariser metric learning for clustering. Bilenko et al. [119] propose MPCK-Means, which learns a metric for each cluster. In order to avoid the intensive computational cost of the matrix M due to the semi-definite condition, Yi et al. [120] propose to learn the matrix using regression analysis and matrix completion technique to rectify the pairwise constraints.

Spectral-based methods are modified to either integrate pairwise constraints or labels, which can be taken into account either in a hard or soft approach. Kamvar et al. [121] were the first to propose incorporating instance-level constraints into spectral clustering. They proposed to modify the affinity matrix by setting must-link points to 1 and cannot-link to zero. It was later extended by Alzate and Suykens [122] to out-of-sample and soft constraints through the use of regularisation. Later, Wang and Davidson [123] and Wang et al. [124] proposed a framework that allows for measuring constraint satisfaction because constraints are modelled by a matrix with values of 1 if must-link and -1 if cannot-link. In the previous approach, a value of zero (CL) does not necessarily guarantee that objects belong to different clusters [125]. In this work, it is possible to use soft constraints by allowing real values or by using fuzzy cluster membership. Wang and Davidson [126] introduce a user-defined lower bound on the level of constraint satisfaction.

Ensemble clustering aims at reducing the bias and/or variance of the clustering algorithms by applying a consensus function on the final results of multiple independent methods [69]. The integration of constraints into ensemble clustering can be done in two manners: constraints are given independently to each method or integrated with the consensus function. Methods following the first approach limit the advantage of ensemble learning since constrained-based approaches tend to have low variance (diversity). Iqbal et al. [127] propose SCEV (Semi-supervised Clustering Ensembles by Voting) to balance diversity by using different constrained algorithms and a weighted voting approach to combine the results. On the other hand, methods following the second approach exist, such as the work of Al-Razgan and Domeniconi [128], Xiao et al. [129], and Dimitriadou et al. [130]. These approaches divide the consensus function into four different steps to create and partition a sparse graph constructed from a similarity matrix based on the set of clustering results. The most important step is the partitioning step where the constraints are integrated, to split or merge clusters. This approach mitigates the drawback of the first approach (low variance) but it adds the difficulty of defining a consensus function that takes into account the constraints and the cluster information.

Collaborative clustering can be extended to constrained clustering through three stages according to Forestier et al. [131]. The first stage is the generation of the final results using label constraints, which is the simplest and easiest to implement as it does not interfere with the collaborative process. The second stage, guiding the collaborative process by directly including the constraints in the collaborative step is highly dependent on how information is shared and exchanged. And the third stage, using the constrained-based agents, this method is expensive as it requires extensive modification of each clustering method. This final approach is limited since it limits the diversity of the algorithms and therefore increases error rates [132]. To address this, Domeniconi and Al-Razgan [132] propose a hybrid approach to integrate constraints in the second and the third stages, in which the algorithms also collaborate using the constraints. SAMARAH [131] also follows a similar approach, where it refines the results according to the constraints with the goal of resolving conflicts [133].

Declarative clustering generalizes the framework of constrained clustering by allowing the expert to explicitly state the constraints, which can be of different types, in the objective function. This allows the search for a global optimum that satisfies all constraints. Generally, optimization tools are used, such as integer linear programming (ILP), Boolean Stability Solver (SAT), or constraint programming (CP). These approaches allow the direct integration of cluster-level constraints, and for different optimization criteria, unlike the previously mentioned approaches that are developed for a particular criterion. Mueller and Kramer [134], Ouali et al. [135] developed a framework based on ILP that can integrate different kinds of constraints. Davidson et al. [136] proposed an SAT-based approach that integrates different types of constraints (pairwise constraints, maximum diameter, and minimum split). Dao et al. [137] introduced a CP-based framework for distance-based constrained clustering. In Duong et al. [138], they show that the problem of finding both compact and well-separated clusters can be solved by iteratively changing the objective function and adding constraints to the other objective value and properties, making the clustering actionable [89].

Incremental and active constrained clustering, these approaches explicitly include the expert during the clustering process (unlike the collaborative approach where the interaction ends after explicitly providing the constraints to the objective function) where the algorithm can query the expert at each iteration. This query enables to the expert to guide the results and for the algorithm to gain additional information, which can take the form of additional constraints to be taken into account in the next clustering iteration. Cohn et al. [116] proposed an approach to iteratively express agreement or disagreement regarding the clustering results using pairwise constraints. In addition, they propose the idea of adding comments on samples themselves if they are “good”, implying it should be maintained in the next clustering, or “bad”, implying they should be changed. The work of Davidson et al. [139] proposes the idea that updating an existing clustering to satisfy new and old constraints is more efficient than re-clustering from scratch. Under the active constrained paradigm [76], where the algorithm proposes potential constraints to the user in the form of queries, and the user validates or rejects them. The works of Lewis [140] and Settles and Craven [141] focus on constraint informativeness and uncertainty by selecting samples with the lowest confidence. The selection process is achieved through a learner that uses a defined strategy and criteria to assess the uncertainty, the authors proposed a number of approaches, such as Fisher Information based selection strategy, Information Density, etc. While the proposition of Basu et al. [103], Settles and Craven [141], and Van Craenendonck et al. [142] emphasize the importance of coherence, they choose constraints by selecting samples that will have the greatest effect on the clustering result or those that reduce clustering error [143, 144].

Deep clustering approaches are based on neural networks, and grouped into two approaches: the first divides the process into transformation learning and clustering [145, 146, 147, 3, 148] and the second learns them both simultaneously [149, 150, 151]. In order to integrate prior knowledge into the learning process, Ohi et al. [145] proposes to train a Siamese network (the weights between parallel networks are not shared) to generate meaningful embeddings from constraints and then perform clustering on them. The model is trained to minimize the pairwise distance of the embeddings, where it adds a distance hyperparameter to assert that cannot-link points are always a specific distance apart (the distance will be bound by the maximum defined by the hyperparameter) and set this distance to zero if the pair of samples has a must-link constraint. This approach is similar in spirit to the concept of metric-constrained learning and our approach. On the other hand, 'Deep constrained clustering – DCC' is proposed by Zhang et al. [3], which is based on a deep clustering method called Deep Embedding Clustering [149] and its improved version, IDEC [152]. This approach is explained more in detail in the following section since we use a time series variant of it to compare our contribution to.

Besides these approaches, efforts to extend density-based clustering, hierarchical, and graph-based approaches exist in the literature, for a detailed overview of each the reader is referred to the work of Cai et al. [91]. Here, we compare algorithms that were created or extended to time series clustering, and these generally fall under deep clustering and partition-based approaches. These algorithms will be highlighted in detail in the next section.

2.3 Time Series Constrained Clustering

In order to be applied to time series, the clustering algorithm must consider both the temporal and dimensional aspects of the data type. It is therefore important to address the challenges related to dimensionality, distortion, volume, and variability, among others, as explained in Chapter 1. In this section, we will outline various approaches proposed in the literature to tackle these issues. Additionally, we will provide a detailed explanation of the methods we compare to.

In the literature, three categories can be identified for time series clustering: whole clustering, subsequence clustering, and time point clustering.

Whole time series clustering considers the entire time series as the object to be clustered. Hence each cluster consists of multiple, complete time series. This approach is suitable for applications where the entire time series is analysed or modelled, such as in remote sensing, medical, or financial data analysis.

Subsequence clustering methods cluster subsequences of time series [153], i.e. each object to cluster is a subsequence or a segment of the time series extracted using a sliding window. This approach might be useful for applications where only certain segments of the time series are relevant, such as in speech or gesture recognition and usually, it is given in the form of streaming time series.

Time Point clustering methods cluster time points within a time series according to a combination of their temporal proximity and similarity [154, 155]. It is similar to the subsequence approach, but in time point clustering not all parts of the time series need to be clustered. This approach is useful for applications where the time series data is irregular or sparse, such as in environmental monitoring or sensor networks.

In this thesis, we work with whole-time series clustering where the approaches can be categorized into model, feature, shape, and shape-feature-based approaches [46]. The shape approach tries to match similar time series accounting for distortions, it employs classical clustering on raw time series with measures modified to time series, such as using DTW distance. In the feature approaches, the time series are mapped into a vector of lower dimension, where the vector components are based on features that are either based on statistical features or learned or extracted from the time series. In the model-based approaches, the time series is transformed into model parameters (parametric model) and then a clustering algorithm is applied [156]. Using one or some of these approaches together is applicable and depends on the problem.

Next, we will provide an explanation of the methods used later in the experimental study in Section 3.3. The methods include COP-Kmeans, which was adapted to time series by Lampert et al. [1], deep constrained clustering as proposed by Zhang et al. [3] and further adapted by Lafabregue et al. [157], and FeatTS clustering as introduced by Tiano et al. [158]. The first two approaches belong to the shape-based category as they use DTW and DBA techniques to handle time series. DCC falls under the model-based approach, while FeatTS under both the shape and model category.

Constrained Kmeans (COP-Kmeans) has been extended to time series by adopting the DTW distance instead of the usual Euclidean distance and use of DTW averaging (DBA) to find cluster centres [1]. The algorithm is described in Algorithm 2. In Line 4, the DTW distance between the cluster centre and the time series instances is calculated so it can be associated with the closest cluster. In Line 8, DBA is used to calculate the centroid for each cluster. As explained in Section 1.1.4 (Chapter 1), DBA is a heuristic aiming to minimize the sum of squared DTW distances of the set of time series and hence results in an average sequence. In line 5 the algorithm asserts that there is no constraint violation, if a constraint violation exists the algorithm will end and not converge.

MIP-Kmeans: extends the K-means algorithm using mixed integer linear programming, hence the acronym MIP-Kmeans. The problem can be considered as

Algorithm 2 COP-KMEANS ALGORITHM

```
Input: data points $x_1, x_2, \ldots, x_n$, number of clusters$K$, must-link constraints $ML$, cannot-link constraints $CL$
Output: Cluster assignments for each data point $\{C_1, \ldots, C_K\}$
Initialize cluster centroids $\mu_1, \mu_2, \ldots, \mu_k$ randomly
while not converged do
// Assign each data point $x_i$ to its closest centroid:
    for each point $x_i$ do
        $k \leftarrow \arg\min_{j \in \{1,\ldots,K\}} DTW(x_i, \mu_j)$
// Check if the assignment of points violates the constraints:
        if violate_constraint($x_i, C_k, CL, ML$) is True return {} else continue
    end for
// Update each centroid as the mean of the data points assigned to it:
    for each cluster $C_j$ do
        $\mu_j \leftarrow DBA(C_j)$
    end for
end while
return $C_1, \ldots, C_K$
Function violate_constraints($x_i, C_k, CL, ML$)
For each $(x_i, x_j) \in ML$ if $x_j \notin C_k$, return True
For each $(x_i, x_j) \in CL$ if $x_j \in C_k$, return True
return False
```

a minimization of the distance between the objects and the cluster centroids, thus minimizing the intra-group distances. The constraints are incorporated in the form of conditions to respect. Let $x_{i,j}$ represent the i^{th} object belonging to the j^{th} cluster, hence the grouping of the objects and the centroids can be represented by $D(x_i, \mu_j)\, x_{i,j}$, where D is the DTW distance and μ_j is the centroid of the j^{th} cluster. The mathematical model can be written as:

$$\min_{\mu_j} \sum_{i=1}^{N} \sum_{j=1}^{K} D(x_i, \mu_j)\, x_{i,j}, \tag{2.9}$$

Subject to:

$$x_{i_1,j} = x_{i_2,j},\ (i_1, i_2) \in ML, \tag{2.10a}$$

$$x_{i_1,j} + x_{i_2,j} \leq 1,\ (i_1, i_2) \in CL, \tag{2.10b}$$

$$\forall j \in \{1, \ldots, K\},\ \sum_{i=1}^{N} x_{i,j} \geq 1, \tag{2.10c}$$

$$\forall i \in \{1, \ldots, N\},\ \sum_{j=1}^{K} x_{i,j} = 1. \tag{2.10d}$$

Algorithm 3 MIP-Kmeans algorithm

Input: data points $x_1, x_2, \ldots, x_n$, number of clusters : K, must-link constraints ML, cannot-link constraints CL

Output: Cluster assignments for each data point $\{C_1, \ldots, C_K\}$

```
Initialize cluster centroids $\mu_1, \mu_2, \ldots, \mu_k$ randomly
Initialize MIP solver with $ML$, $CL$, $\mu_j$ and $x_i$ to reflect Equations 2.9 and 2.10
while not converged do
    Assign time series to the closest centroid
    Update cluster centroids
    Update MIP solver
    Resolve pairwise constraints by adjusting cluster assignments
end while
return $\{C_1, \ldots, C_K\}$
```

The conditions in Equations 2.10a and 2.10b assert that the constraints are respected, the condition in Equation 2.10c makes sure that each cluster has at least one object, and the last condition in Equation 2.10d guarantees that each instance belongs to one and only one cluster (hard clustering). The MIP-Kmeans algorithm is represented in Algorithm 3.

Deep Constrained Clustering (DCC): similar to DEC, DCC first trains an autoencoder ($x_i = g(f(x_i))$) and then removes the decoder part (see Figure 2.12). The embeddings z_i returned by the encoder are used to learn hard clustering for the instances. This is achieved by fine-tuning the encoder ($z_i = f(x_i)$) by optimizing an objective function based on Kullback-Leiber divergence L_c between two different distributions Q and P, which represent the cluster centres as Gaussian distributions (Q) and the target distribution P as the normalised square of Q (pulling points towards the cluster centres). The soft cluster assignment for instance i is represented as a vector q_i of length K, it is computed from z_j, which indicates the degree of belonging to cluster j. On the other hand, P represents the target distribution based on Q which indicates the hard assignment of instance i to only one cluster. The following is the mathematical definition for L_c, q_{ij} and p_{ij},

$$L_c = KL(P||Q) = \sum_{i=1}^{N} \sum_{j=1}^{K} p_{ij} \log \frac{p_{ij}}{q_{ij}},$$

$$q_{ij} = \frac{(1 + ||z_i - \mu_j||^2)^{-1}}{\sum_j (1 + ||z_i - \mu_j||^2)^{-1}}, \quad p_{ij} = \frac{q_{ij}^2 / \sum_i q_{ij}}{\sum_j (q_{ij}^2 / \sum_i q_{ij})},$$

where μ is the set of centriods initialized using K-means on z, q_{ij} is the similarity between z and μ based on Student's t-distribution [159]. N is the number of instances and K is the number of clusters. In order to preserve the representativeness of the embedded features and not be affected after removing the decoder, Guo et al. [152] propose to keep the reconstruction loss. Hence, the overall loss function is:

$$L = L_r + \gamma L_c,$$

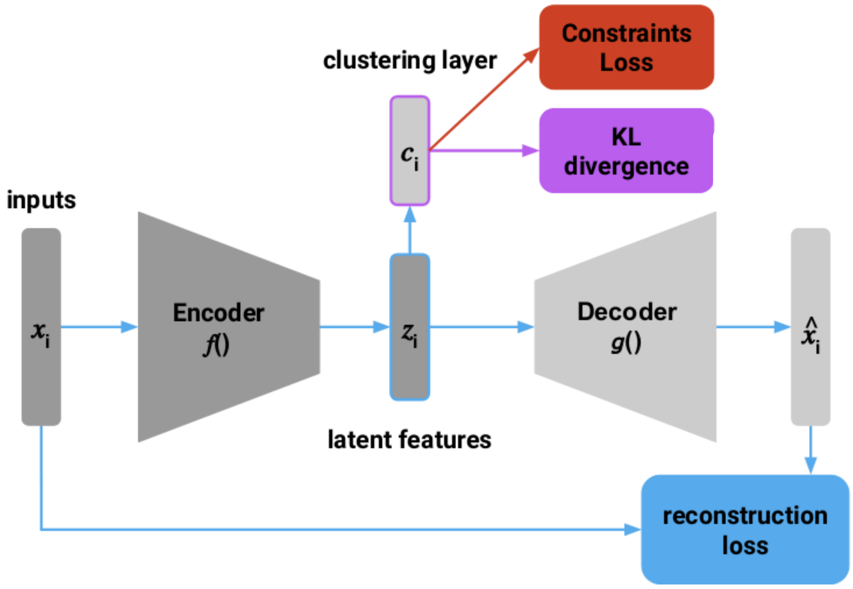


Figure 2.12: The DCC method architecture. Adapted from [160].

where L_r is the reconstruction loss and written as,

$$L_r = \frac{1}{n}\sum_{i=1}^{n} ||g(f(x_i)) - x_i||^2.$$

Zhang et al. [3] proposes four types of constraints, but the pairwise constraints is the only one that concern this work. These aim to maximize the similarity between the latent representations of objects for must-link constraints and respectively minimize it for cannot-link constraints. The must-link ML and cannot-link CL constraint losses are defined as:

$$l_{ML} = L_r - \gamma_{ML} \cdot \sum_{(a,b)\in ML} \log \sum_j q_{aj} \cdot q_{bj},$$

$$l_{CL} = \sum_{(a,b)\in CL} \log(1 - \sum_j q_{aj} \cdot q_{bj}).$$

Moreover, Lafabregue et al. [157] show that by modifying DCC to use 1D-convolutional layers, better performance is achieved with time series [161]. In the modified version, these are also followed by batch normalization layers and the embedding layer is preceded by a global average pooling layer. The final embedding layer remains fully connected.

Feature-based Time Series Clustering (FeatTS), is a semi-supervised approach that uses a small portion of labelled data to select relevant features from the time series. These features encode global similarities between time series. Note that this approach is different from the constrained algorithms presented above, as it assumes that the thematic labels (classes) are known for the few instances labelled

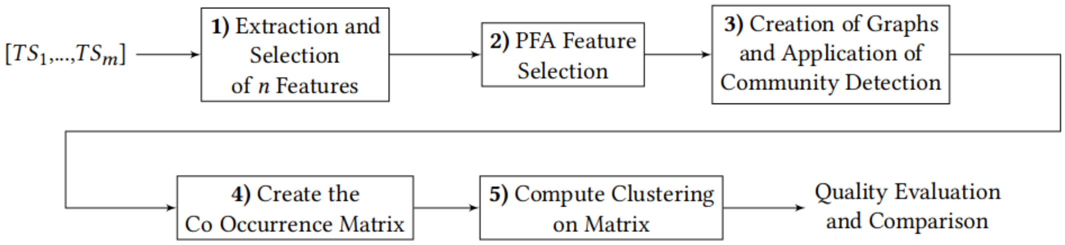

$[TS_1,...,TS_m]$
1) Extraction and Selection of *n* Features
2) PFA Feature Selection
3) Creation of Graphs and Application of Community Detection
4) Create the Co Occurrence Matrix
5) Compute Clustering on Matrix
Quality Evaluation and Comparison

Figure 2.13: The algorithmic pipeline of FeatTS, adapted from [158].

by the expert. Hence, this approach takes much more information compared to constrained algorithms because the labels represent N(N-1)/2 constraints, given N labels.

As shown in Figure 2.13, FeatTS starts by (1) extracting and selecting n features, using the TSfresh [162] method. Since not all features are equally relevant for clustering, the authors propose to use a supervised procedure called Benjamini-Yekutieli [163] to rank the features. Then (2) Principal Factor Analysis (PFA) feature selection (a variant of principal component analysis) is used to choose the features necessary to create a graph encoding. PFA is used since it preserves the original values and leverages the concept of explained variance.

Once the features are selected, FeatTS computes the global relationships between the time series based on their statistical features. It uses graph networks (3) to encode these relationships. Each time series is converted into a node, and weighted edges are created between them. The edge weight represents the difference between the values of two different time series using the selected features. The graphs are pruned based on a threshold and a Community Detection algorithm is applied to obtain the global relationships among the time series. The results of the communities are merged into a Co-Occurrence matrix (4), on which a clustering algorithm (K-Medoid) is applied (5).

2.4 Conclusions

This chapter provided a detailed overview of the related work, background knowledge, and state-of-the-art techniques. We described shapelets in depth, which are subsequences of time series capable of distinguishing between different categories of time series. These shapelets can be extracted or learned through supervised or unsupervised approaches. We discussed various approaches from the literature that aim to find optimal shapelets and highlighted their strengths and weaknesses for different tasks. Furthermore, we focused on one specific approach called Learning DTW Preserving Shapelets. This approach aims to learn shapelets in an unsupervised manner while approximating the DTW similarity measure (as presented in Chapter 1), which takes into account the distortions present in time series data. Next, we presented the task addressed in this thesis, which is time series clustering. We discussed different approaches to clustering time series data, emphasizing the challenging nature of the problem due to the lack of background knowledge. We then introduced a semi-supervised approach for clustering, specifically falling under the guided approaches where the expert provides their intuition to guide the algorithm through constraints aligned with their expectations. We described various constrained clustering approaches and highlighted constrained clustering methods specifically tailored for time series data. In the evaluation study of our approach, we provided a detailed explanation of four constrained clustering algorithms for time series. The first is constrained K-means, referred to as COP-Kmeans, which incorporates expert knowledge through must-link and cannot-link constraints to ensure clusters adhere to these constraints. Another variant, called MIP-Kmeans, transforms the clustering problem into a linear integer programming problem subject to a set of constraints encoding must-link and cannot-link relationships. We also explained a constrained algorithm utilizing neural networks, known as deep constrained clustering (DCC), which learns a time series embedding using an autoencoder. The embedding is then transformed using the encoder part, followed by a clustering layer. Additionally, we presented a semi-supervised approach using labels instead of constraints for clustering, known as FeatTS. This approach aims to extract statistical features from time series data and applies KMedoids clustering on a graph constructed using these features. Overall, these approaches represent different perspectives on constrained clustering for time series data and are utilized in the evaluation study of our proposed approach.

Part II

Contributions

CHAPTER 3

CONSTRAINED DTW PRESERVING SHAPELETS

> *"The key to artificial intelligence has always been the representation."*
> – Jeff Hawkins

3.1 Contrastive Learning 52
3.2 Constrained DTW Preserving Shapelets 55
  3.2.1 Proposed Loss Function 56
  3.2.2 Model Architecture 60
  3.2.3 Learning Procedure 62
3.3 Evaluation 63
  3.3.1 Experimental Setup 63
  3.3.2 Clustering 65
3.4 Discussion 70
  3.4.1 Results 70
  3.4.2 Model Selection 71
  3.4.3 Sensitivity Study 73
3.5 Conclusions 76

We have previously pointed out the ill-posed nature of clustering and that it can be mitigated by using constrained clustering, where we emphasized our interest in pairwise constrained clustering approaches. We highlighted that in order to measure the properties of constraints, a metric measure is necessary. This is hard to obtain for time series data due to distortions that need to be accounted for. In order to resolve these problems, i.e. a space where a metric measure can be used and to adapt it to time series, we propose Constrained DTW Preserving Shapelets, termed CDPS. We argue that using constraints in learning a time series representation can help in obtaining a space where the expert's intuition is respected. This transformation can be used with unsupervised (unconstrained) clustering approaches, inherently adding the expert's intuition to the clustering process. This has multiple advantages, for example, there is no need to adapt the clustering algorithm to take constraints into account or to create complex algorithms to account for this information. In order to achieve this, we adopt the concept of contrastive learning, in which the model learns distinctions between samples, based on a comparison between similar and dissimilar pairs of data.

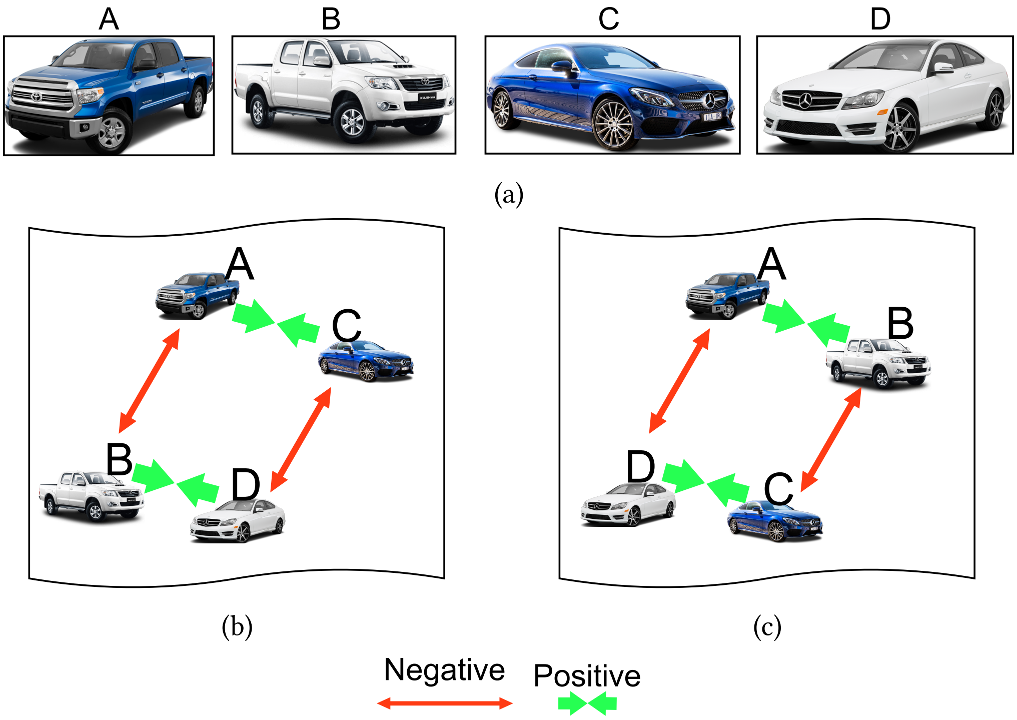
First, we explain the concept behind contrastive learning in Section 3.1, which is used to model the pairwise constraints. In Section 3.2, the CDPS model is presented in detail and discussed in detail the sections to follow.

3.1 Contrastive Learning

Generally, contrastive means showing and emphasizing the difference and distinctions between things, by identifying contrasting elements, characteristics, or features. For example in image recognition, one can emphasise different groupings of cars based on colour and/or brand (Figure 3.1).

Contrastive learning is an approach falling under the paradigm 'learn-to-compare' that uses contrastive information to learn a model for representation [164, 165], similarity measure [166, 167], classification [168], clustering [169], and more [170]. Most of these approaches were developed for images (classification and segmentation) and natural language processing. Where they use the concept of positive pairs, i.e. similar samples (e.g. cars with similar colours and/or brands), and negative pairs, i.e. distinct samples (e.g. cars with different colours and/or brands), to identify and learn the contrastive relation between samples.

In supervised contrastive learning approaches, the relationships between objects are inferred from labels, and represent a large amount of information. It is equivalent to all the possible constraints between the labelled points, i.e. $N(N-1)/2$ constraints if we had N labels. This enormous number of pairs increases the computational complexity of the model and the memory load. On the other hand, in unsupervised approaches, data augmentation is usually used to build positive pair samples, and self-supervised learning is used to generate pseudo labels. The choice of data augmentation method is of high importance in unsupervised contrastive learning and using multiple data augmentation operations is essential in



(a)

(b)

(c)

Figure 3.1: Illustration of the general idea behind contrastive learning. (b) Shows contrastive relations according to colour (blue, white) and in (c) according to brands (Toyota, Mercedes).

yielding effective representations [171]. Chen et al. [171] argue that unsupervised learning benefits more from data augmentation than supervised learning. A major drawback of unsupervised contrastive learning is the batch size[1]. The batch size is large so that it can account for enough contrastive information, but this increases the computational and memory complexity.

In general, the learning process depends on the loss used. There are two categories for contrastive loss: distance and probabilistic-based.

Distance-based: The objective is to maximize the distance between negative pairs and minimize the distance between positive pairs in the learned representation space. The loss function encourages the model to learn representations that can distinguish between similar and dissimilar samples based on their distances. Chopra et al. [166], Hadsell et al. [172] proposed the first classical contrastive loss in a supervised paradigm.

Probabilistic-based: The objective is to compare the distributions of embeddings in order to identify the contrastive relationship between samples [164]. Instead of directly optimizing distances, these loss functions focus on modelling the probability or likelihood of samples being similar or dissimilar [165].

---

[1] Batch size is the number of input samples to be fed into the model to train during each training iteration.

> Schroff et al. [173] proposed the triplet loss that uses the concept of an anchor sample and compares a positive (hence maximizing the distance) and a negative sample (minimizing the distance) to it.

Out of these two approaches we are interested in using distance base contrastive learning as it allows us to model the information in the form of the distance between samples, which serves as an important component in the CDPS algorithm. Next, we will explain in detail the loss proposed by Hadsell et al. [172] in 'Dimensionality Reduction by Learning an Invariant Mapping', as the loss proposed for CDPS is based upon this concept. The idea behind this work was to learn to map the data into a low-dimensional manifold using neighbourhood relationships.

Let $\mathcal{X} = \{X_1, \ldots, X_N\}$ be set of N instances where $X_i \in \mathbb{R}^F$ is the i^{th} sample with F-features, and G_W a parametric transformations $G_W : \mathbb{R}^F \rightarrow \mathbb{R}^{F'}$ where $F' << F$, such that

- the distance in the output space approximates the neighbourhood relationships in the input space;
- the mapping should be able to learn invariances to complex transformations;
- should model samples whose neighbourhood relationships are unknown.

Assuming that for each X_i there exist a set of positive pairs P_{X_i}. Using this information generates a binary label Y indicating the contrastive relation between X_i and any other instance, such that the value of Y is zero for every other instance in P_{X_i} and one for instances that are not in P_{X_i}. Since the problem is to learn a representation based on relationships, the authors propose to learn a parametrized distance function defined over the transformations of two input instances X_1 and X_2. The distance function to be learnt is written as:

$$\mathcal{D}_W(X_1, X_2) = ||G_W(X_1) - G_W(X_2)||_2,$$

and the overall loss function as:

$$\mathcal{L}(W) = \sum_{i=1}^{N} L\left(W, (Y, X_1, X_2)^i\right), \tag{3.1}$$

where W represents the weights to be learnt of a neural network representing the transformation and $(Y, X_1, X_2)^i$ is the i^{th} labelled sample pair, and

$$L\left(W, (Y, X_1, X_2)^i\right) = (1 - Y)L_S(\mathcal{D}_W^i) + Y L_D(\mathcal{D}_W^i), \tag{3.2}$$

where $\mathcal{D}_W^i$ is the distance between the i^{th} pair, such that L_S is the loss function for similar pair instances (positive) and L_D for the dissimilar ones (negative).

The authors provide an analogy to the physical spring model, known as Hooks law, which models the forces acting on a spring, where they assume that $\mathcal{D}_W^i$ is

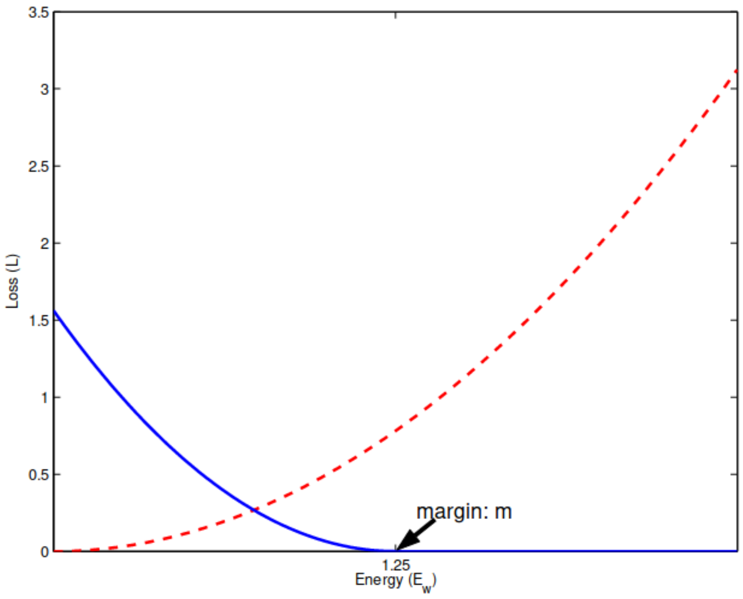


Figure 3.2: Illustration of the loss function L against the energy $\mathcal{D}_W$, where the red dashed line is the loss of positive pairs and the blue solid line is for the negative pairs. Adapted from [172].

the energy such that the minimization of L leads to the minimization of L_S and maximization of L_D as shown in Figure 3.2. The exact solution of Equation 3.2 that satisfies this condition will be

$$L_S(W, X_1, X_2) = \mathcal{D}_W(X_1, X_2), \tag{3.3}$$

$$L_D(W, X_1, X_2) = \frac{1}{2}\left(\max\{0, m - \mathcal{D}_W(X_1, X_2)\}\right)^2, \tag{3.4}$$

where m is the margin in which after this distance the points are considered to be distinct enough, preventing them from being pushed towards infinity. While training, the gradient with respect to W will play the role of either an attractive force (in the spring analogy) of positive pairs or a repulsive force of negative pairs. In the work of Hadsell et al. [172] and similar works for learning contrastive representations, the learning is based on either a convolutional-based neural network that learns embeddings, or is based on more recent architectures. One drawback of these approaches is that the transformation is a black box where the learnt weights cannot be interpreted by the expert. Moreover, as mentioned earlier, the amount of information required is relatively large, which can prohibit its use in some domains.

3.2 Constrained DTW Preserving Shapelets

In our work, we use the contrastive concept presented earlier such that similar points are those associated with a must-link constraint (positive pairs) while dissimilar points are those associated with a cannot-link constraint (negative pairs) and we aim to minimise the distance between ML points and increase the distance between CL points by adapting the loss proposed by Hadsell et al. [172]. Unlike the unsupervised and supervised approaches that use huge amounts of contrastive pairs or big batch sizes, we show that using the training procedure we propose is sufficient to learn a representation guided by the contrastive information given by the expert.

As a reminder, in our work, we aim to learn a guided transformation with explainable capabilities that take into account the distortions exhibited in time series. We argue that combining the transformation offered by shapelets with the capabilities of the contrastive loss will allow us to encode expert intuition and model DTW distance. Next, we explain the proposed loss followed by a description of the architecture used.

3.2.1 Proposed Loss Function

Given a set of N time series $\mathcal{T} = \{T_1, \dots, T_N\}$ with F features and length L, therefore $T_i \in \mathbb{R}^{F \times L}$ (theoretically the time series can be of varying length but for simplicity, we work with uniform length). Let ML be a set of must-link pairs and CL a set of cannot-link pairs, given by the expert. We aim to learn a set of shapelets $\mathcal{S} = \{S_1, \dots, S_K\}$ of length L_{S_k} such that the shapelet transform

$$\begin{aligned} \mathrm{ST} :& \mathbb{R}^{F \times L} \to \mathbb{R}^{F \times K}, \\ & T_i \to \overline{T}_i, \end{aligned}$$

is subject to the conditions

$$D(\overline{T}_i, \overline{T}_j) \approx \mathrm{DTW}(T_i, T_j), \tag{3.5}$$

$$D(\overline{T}_i, \overline{T}_j) \to 0 \text{ if } (T_i, T_j) \in \mathrm{ML}, \tag{3.6}$$

$$D(\overline{T}_i, \overline{T}_j) \to m_{i,j} \text{ if } (T_i, T_j) \in \mathrm{CL}, \tag{3.7}$$

where $\overline{T_i} = \{\overline{T}_{i,1}, \dots, \overline{T}_{i,K}\}$ is the transformation of time series T_i, $\overline{T}_{i,k}$ is the Euclidean shapelet match between time series T_i and shapelet S_k explained in Definition 5 in Chapter 2, and $D = || \cdot ||_2$ is the Euclidean distance between the transformed time series in the new space. Equation 3.5 forces the transformation to approximate the DTW distance and Equation 3.6 asserts that if the expert indicates the samples are similar, the distance should be pushed as close as possible, and if the samples are indicated as dissimilar (CL), Equation 3.7 forces the instance to be pushed further apart in the new space (this distance is limited by the variable $m_{i,j}$). The value of $m_{i,j}$ is of great importance, if it is not sufficiently large it will not separate the points enough and if it was too large it will lead CL points biasing the loss's gradient. The choice of $m_{i,j}$ will be explained in detail later. In what follows, the distance $D(\overline{T}_i, \overline{T}_j)$ will be written as $\mathcal{D}_{i,j}$ for simplicity.

In the unsupervised LDPS approach [32] (explained in detail in Section 2.1.3), in which the space is learnt only to model the DTW distance, the loss function only satisfies Equation 3.5 and is defined such that

$$\mathcal{L}_{\text{LDPS}}(T_i, T_j) = \frac{1}{2}\left(\text{DTW}(T_i, T_j) - \beta \cdot \mathcal{D}_{i,j}\right)^2,$$

where β is a parameter to scale the approximated distance to the same order of the DTW similarity. This measures the squared error between the DTW similarity and approximated distance and hence minimising $\mathcal{L}_{\text{LDPS}}$, therefore, measures the squared error between the DTW distance and approximated distance and hence minimising $\mathcal{L}_{\text{LDPS}}$, minimises the approximation's error.

In order to also take into account the information provided by the expert through constraints, we propose to modify $\mathcal{L}_{\text{LDPS}}$ to integrate the contrastive paradigm. The proposed loss function can be written as:

$$\mathcal{L}_{\text{CDPS}}(T_i, T_j) = \mathcal{L}_{\text{LDPS}} + \phi_{i,j},$$

where $\phi_{i,j}$ is the contrastive term. The term $\phi_{i,j}$ is inspired by the contrastive loss and is defined, such that

$$\phi_{i,j} = \begin{cases} \alpha \cdot \mathcal{D}_{i,j}^2, & \text{if } (i,j) \in \text{ ML}, \\ \gamma \cdot \max(0, m_{i,j} - \mathcal{D}_{i,j})^2, & \text{if } (i,j) \in \text{ CL}, \\ 0, & \text{otherwise}, \end{cases} \tag{3.8}$$

where α, γ are weights that regularise the must-link and cannot-link distances respectively, and m 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). The term m is calculated using

$$m_{i,j} = \max_{\forall i, \forall j}(DTW(T_i, T_j)) + \log\left(\frac{DTW(T_i, T_j)}{\max_{\forall i, \forall j}(DTW(T_i, T_j))}\right), \tag{3.9}$$

such that $i \neq j$. An illustration of Equation 3.9 is presented in Figure 3.3. In order to understand the use of this function (and therefore Figure 3.3), let us consider the following scenario. The expert provides the algorithm with cannot-link constraints on pairs of samples that are already 'well-separated' by DTW distance we argue that there is no need to push these samples further apart as they are already considered far by approximating DTW (hence the plateau in the value of m as visualized in Figure 3.3). On the other hand, if these cannot-link (dissimilar) samples have a small DTW distance (which can be due to problems such as misclassification, etc. as reported in Chapter 1), we aim to adjust and correct the small distance in the new space by forcing them to be pushed further apart, limited by m. This is reflected in the small DTW values in Figure 3.3, for which the value of m increases sharply. Consequently, the overall loss function over the entire dataset $\mathcal{T}$ is formalized such that:

$$\mathcal{L}(\mathcal{T}) = \frac{2}{K(K-1)} \sum_{i=1}^{K} \sum_{j=i+1}^{K-1} \mathcal{L}_{\text{CDPS}}(T_i, T_j). \tag{3.10}$$

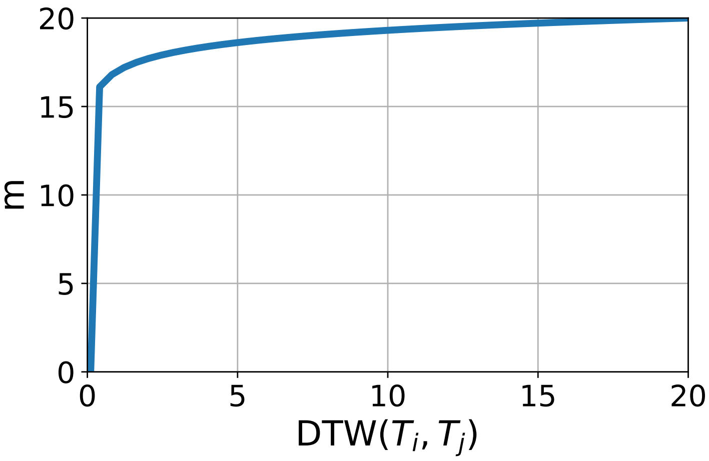


Figure 3.3: Graph of the margin $m_{i,j}$ (represented as m in the figure, which acts upon CL constraints, see Equation 3.8) against $DTW(T_i, T_j)$ and with $\max_{\forall i,\forall j}(DTW(T_i, T_j)) = 20$.

Minimization of $\mathcal{L}_{\text{CDPS}}$ with respect to β and $S_{k,l}$

Now, we will explain the effect of minimizing the loss function on the distance between the time series in the transformed space. Using the chain rule, the partial derivative of $\mathcal{L}_{\text{CDPS}}$ with respect to a shapelet parameter $S_{k,l}$ (k^{th} shapelet at the l^{th} element) over points (T_i,T_j) can be written as:

$$\begin{aligned}\frac{\partial \mathcal{L}_{\text{CDPS}}(T_i,T_j)}{\partial S_{k,l}} &= \frac{\partial \mathcal{L}_{\text{LDPS}}(T_i,T_j)}{\partial S_{k,l}} + \frac{\partial \phi_{i,j}}{\partial S_{k,l}} \\ &= \underbrace{\frac{\partial \mathcal{L}_{\text{LDPS}}(T_i,T_j)}{\partial \mathcal{D}_{i,j}}\frac{\partial \mathcal{D}_{i;j}}{\partial S_{k,l}}}_{\mathbb{A}} + \underbrace{\frac{\partial \phi_{i,j}}{\partial \mathcal{D}_{i,j}}\frac{\partial \mathcal{D}_{i,j}}{\partial S_{k,l}}}_{\mathbb{B}}.\end{aligned} \tag{3.11}$$

First, we study the effect of the $\mathcal{L}_{\text{LDPS}}$ loss, which will be referred to as term $\mathbb{A}$ and can be further expanded to

$$\mathbb{A} = -\beta(\text{DTW}(T_i,T_j) - \beta \cdot \mathcal{D}_{i,j})\frac{\partial \mathcal{D}_{i,j}}{\partial S_{k,l}}$$

and $\mathbb{B}$ into

$$\mathbb{B} = \begin{cases} 2\alpha\mathcal{D}_{i,j}\frac{\partial \mathcal{D}_{i,j}}{\partial S_{k,l}}, & \text{if } (i,j) \in ML, \\ -2\gamma(m_{i,j} - \mathcal{D}_{i,j})\frac{\partial \mathcal{D}_{i,j}}{\partial S_{k,l}}, & \text{if } (i,j) \in CL, \\ 0, & \text{otherwise.} \end{cases}$$

Following a similar analogy as Hadsell et al. [172], we treat $\mathbb{A} = \frac{\partial \mathcal{L}_{\text{LDPS}}(T_i,T_j)}{\partial S_{k,l}}$ and $\mathbb{B} = \frac{\partial \phi_{i,j}}{\partial S_{k,l}}$ as forces acting on $\overline{T}_i$ to push it closer to or further apart from $\overline{T}_j$. Figure 3.4 gives an illustration of this analogy. For the different cases shown, $\mathbb{A}$ is represented by the blue arrow and its purpose is to push/pull the points to model

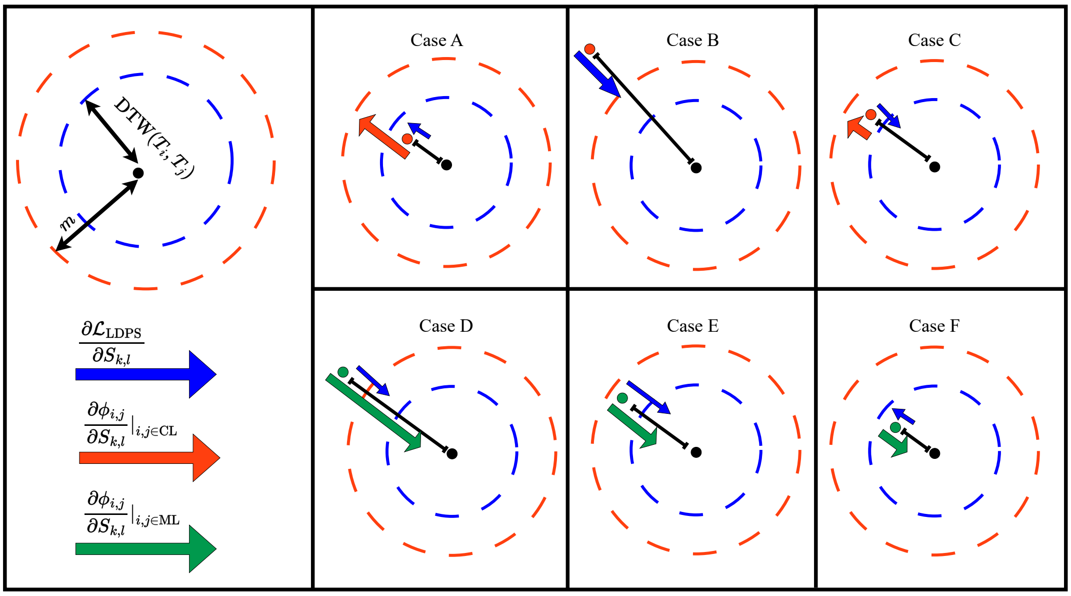

$DTW(T_i,T_j)$
m
Case A
Case B
Case C
Case D
Case E
Case F
$\frac{\partial \mathcal{L}_{\text{LDPS}}}{\partial S_{k,l}}$
$\frac{\partial \phi_{i,j}}{\partial S_{k,l}}|_{i,j\in \text{CL}}$
$\frac{\partial \phi_{i,j}}{\partial S_{k,l}}|_{i,j\in \text{ML}}$

Figure 3.4: Illustration of the effect of the proposed loss function on the distance of the points in the new space. Different cases are illustrated depending on whether the points are linked with a must-link (green) or cannot-link (red) constraint, blue indicates the effect of approximating DTW.

their DTW distance (the blue dotted circle). The negative sign in $\mathbb{A}$ asserts that this force will always be opposite to DTW$(T_i, T_j) - \beta \cdot \mathcal{D}_{i,j}$, meaning that if this term is negative then $\mathcal{D}_{i,j}$ is larger than DTW and the points need to be pulled closer to each other and vice versa. The term $\mathbb{B}$ has two different roles if the points are connected by a must-link (i.e. $(i, j) \in$ ML) then they should be closer to each other (represented by a green arrow) and if connected by cannot-link (i.e. $(i, j) \in$ CL) should be pushed further apart (represented by the red arrow). When $(i, j) \in$ ML, the absence of the sign means that $\overline{T}_i$ will always be pushed to $\overline{T}_j$ and the α parameter can either increase the impact of this force or decrease it. In contrast, when $(i, j) \in$ CL, the points will always be pushed away from each other until the distance between them is m, after which the force has no effect.

As illustrated in Figure 3.4 the force brought by $\mathbb{A}$ (responsible for approximating the DTW distance, the blue arrow) is present in all the cases (but would cancel out when $\beta \cdot \mathcal{D}_{i,j} = DTW(T_i, T_j)$), The interaction between the other forces shown in Figure 3.4 are as follows: in Case A, the points are dissimilar and the forces exerted are the repulsive force $\mathbb{B}$ (red arrow) and the force for DTW approximation $\mathbb{A}$ (blue arrow), after training for a while the points might be pushed sufficiently far apart that the repulsive force is zero (the points are distinct enough), as shown in Case B, by continuing training the forces should reach an equilibrium, Case C, where the points are sufficiently far from each other (but not necessarily equal to the actual DTW distance between the time series). The same reasoning can be made for the samples connected by a must-link (cases D, E, and F).

Having explained the effect of the loss on the space, we provide the necessary derivation. The derivative of the distance between the samples in the transformed space $\mathcal{D}_{i,j}$ by $S_{k,l}$ is

$$\frac{\partial \mathcal{D}_{i,j}}{\partial S_{k,l}} = \frac{\partial \mathcal{D}_{i,j}}{\partial \Delta_{i,j,k}} \frac{\partial \Delta_{i,j,k}}{\partial S_{k,l}},$$

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

$$\frac{\partial \mathcal{D}_{i,j}}{\partial \Delta_{i,j,k}} = \frac{\Delta_{i,j,k}}{\mathcal{D}_{i,j}}, \qquad \text{where } \mathcal{D}_{i,j} \neq 0,$$

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

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

In order to calculate the derivative with respect to the shapelet score, we use soft-minimum as an approximation of the minimum, as used in LDPS [32], which gives $\frac{\partial \overline{T}_{i,k}}{\partial D_{i,k,w}} = \delta_{w,w^*}$ and hence the above can be written as:

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

and,

$$\begin{aligned}
\frac{\partial D_{i,k,w}}{\partial S_{k,l}} &= \frac{1}{L_{S_k}} \frac{\partial}{\partial S_{k,l}} \left[\sum_{x=1}^{L_{S_k}} (T_{i,w+x-1} - S_{k,x})^2 \right]_{x=l} \\
&= -\frac{2}{L_{S_k}} (T_{i,w+l-1} - S_{k-l}).
\end{aligned}$$

The derivative with respect to the scaling parameter β is

$$\begin{aligned}
\frac{\partial \mathcal{L}_{\text{CDPS}}}{\partial \beta} &= \frac{\partial \mathcal{L}_{\text{CDPS}}}{\partial \beta} + \frac{\partial \phi_{ij}}{\partial \beta} \\
&= -\beta[DTW(T_i, T_j) - \beta \mathcal{D}_{i,j}] + \cancelto{0}{\frac{\partial \phi_{ij}}{\partial \beta}},
\end{aligned}$$

where ϕ_{ij} is used to penalize the approximation of DTW according to the expert regardless of the actual DTW so there is no need to include β in $\phi_{i,j}$. This means that the adjustment of β will only depend on the error between actual DTW(T_i, T_j) and the approximated distance $\mathcal{D}_{ij}$.

3.2.2 Model Architecture

We employ a Siamese architecture, which is well-suited for contrastive learning. Our chosen architecture closely resembles the one described in LDPS. It consists of

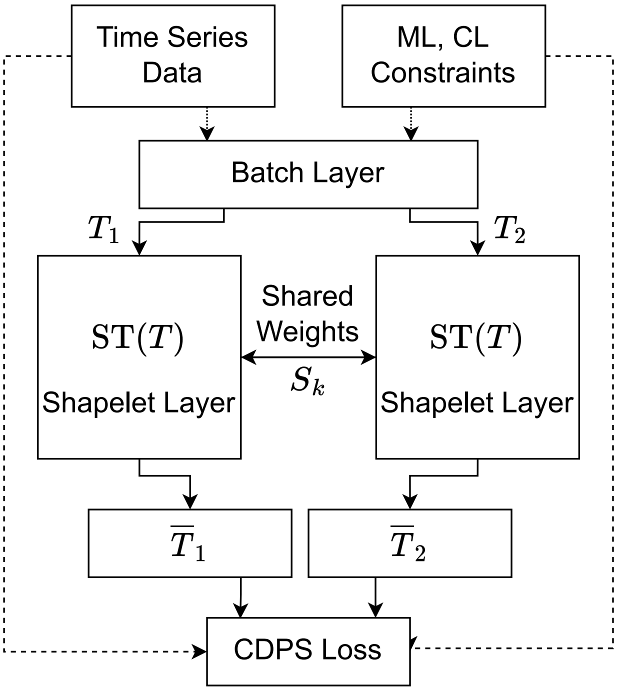


Figure 3.5: Illustration of the CDPS model. ST is the shapelet transform, S_k are the weights of the shapelets, T_i ($i = \{1, 2\}$) are the input generated by the batch layer.

layers representing shapelets, called ‘shapelet blocks’. The shapelet blocks have no interconnections i.e. they are parallel. The shapelet blocks are followed by a min-pooling to calculate the best shapelet match over the whole time series. Figure 3.5 provides a visual representation of the model architecture.

In contrast to LDPS, we use a batch input layer and train the model using batch gradient descent instead of stochastic descent. The batch layer takes the time series and the constraints set as its input so that it can generate batches containing samples that are constrained and those that are not. Using batch gradient descent will lead to less noise in the backward propagation especially when having contrasting information. Note that the batch size in CDPS is typically smaller compared to what is commonly used in contrastive learning. Using a very large batch size will make the backward propagation smoother but will lead to a local optimum, choosing a smaller batch size on the other hand will allow the training to escape the local optimum and converge to a better solution. The training procedure will be described in more detail in the subsequent sections.

Figure 3.6 shows an illustration of the shapelet layer. For simplicity, we illustrate uni-variate time series (those with one feature) and three shapelets blocks, where each can have a different shapelet length (denoted as L_{S_i}, $i = \{1, 2, 3\}$) and a different number of shapelets (denoted as NUM_{S_i}, $i = \{1, 2, 3\}$). After receiving the input batches, the shapelet layer takes the time series as input and calculates the Euclidean Shapelet Match $D_{i,k,w}$ (defined in Definition 4, Chapter 2) with respect to the shapelets in each shapelet block. $D_{i,k,w}$ will return all the possible distances between each shapelet in the blocks and each possible subsequence of the time se-

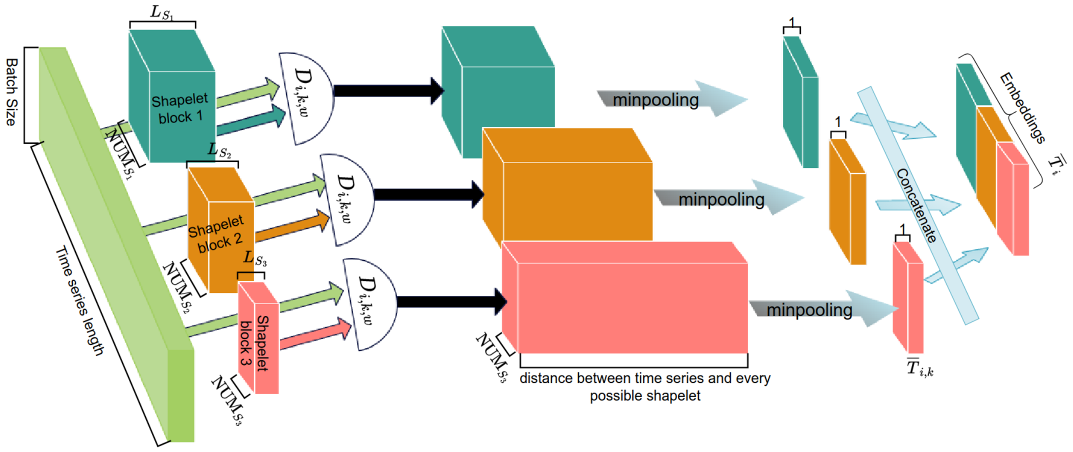

Batch Size
L_{S_1}
Shapelet block 1
NUM_{S_1}
L_{S_2}
Shapelet block 2
NUM_{S_2}
L_{S_3}
Shapelet block 3
NUM_{S_3}
Time series length
$D_{i,k,w}$
$D_{i,k,w}$
$D_{i,k,w}$
minpooling
minpooling
minpooling
NUM_{S_3}
distance between time series and every possible shapelet
1
1
1
Concatenate
$\overline{T}_{i,k}$
Embeddings
$\overline{T}_i$

Figure 3.6: Illustration of the shapelet layer. $D_{i,k,w}$ is the Shapelet Euclidean Score (Definition 4), L_{S_i} is the length of the shapelet block, NUM$_i$ is the number of shapelets in a shapelet block, and $\overline{T}_{i,k}$ is the embeddings with respect to each shapelet block.

ries, which is then fed into a min-pooling layer to get the best match (since we are using one feature, the output of the min-pooling will be: number of time series × number of shapelet × one).

3.2.3 Learning Procedure

Algorithm 4 describes the CDPS approach to learning the representational embedding. ShapeletBlocks is a dictionary containing S_{max} pairs, $\{$shapelet length L_{S_k}; shapelet number $\}$, where shapelet length is $L_{\text{min}} \cdot b_{\text{ind}}$, L_{min} is the minimum shapelet length and $b_{\text{ind}} \in \{1, \ldots, S_{\text{max}}\}$ is the index of the shapelet block. The number of shapelets for each block is calculated using the same approach as LDPS [32]: $10 \times \log(L_i - L_{\text{min}} \cdot b_{\text{ind}})$. INITIALIZE_SHAPELETS initializes the shapelets either randomly or rule-based. Here the following rule-based approach is taken: (1) Shapelets are initialized by drawing a number of time series samples and then reshaping them into sub-sequences with lengths equal to that of the shapelets; (2) K-means clustering is then performed on the sub-sequences and the cluster centres are extracted to form the initial shapelets. GET_BATCH($\mathcal{T}$, ML, CL, s_{batch}, c_{batch}) returns batches containing both constrained and unconstrained samples, where the batch size is s_{batch} and the number of constraints in a batch is c_{batch}. The introduction of c_{batch} ensures that constrained time series are frequently observed during training, increasing their impact in the presence of a large number of 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.

Algorithm 4 Constrained Distance Preserving Shapelets (CDPS)

```
Input:  $\mathcal{T}$ a set of time-series,
        ML and CL constraint sets,
        $L_{\text{min}}$ minimum length of shapelets,
        $S_{\text{max}}$ maximum number of shapelet blocks,
        $n_{\text{epochs}}$, $s_{\text{batch}}$, $c_{\text{batch}}$
Output: Set $S$ of shapelets,
        Embeddings of $\mathcal{T}$.
ShapeletBlocks ← GET_SHAPELET_BLOCKS($L_{\text{min}}$, $S_{\text{max}}$, $L_i$)
Shapelets ← INITIALIZE_SHAPELETS(ShapeletBlocks)
for $i \leftarrow 0$ to $n_{\text{epochs}}$ do
    for 1 to $|\mathcal{T}|/s_{\text{batch}}$ do
        minibatch ← GET_BATCH($\mathcal{T}$, ML, CL, $s_{\text{batch}}$, $c_{\text{batch}}$)
        Compute the DTW between the $T_{i's}$ and $T_{j's}$ in minibatch
        Update the Shapelets and $\beta$ by descending the gradient $\nabla\mathcal{L}(T_i, T_j)$
    end for
end for
Embeddings ← SHAPELET_TRANSFORM($\mathcal{T}$)
```

3.3 Evaluation

After providing a comprehensive explanation of CDPS and its ability to create a representational space, our next step is to evaluate CDPS through multiple experimental protocols to highlight its strengths and weaknesses. We first describe the experimental setup.

3.3.1 Experimental Setup

Unsupervised (K-means, and DCC without constraints) and semi-supervised clustering algorithms (COP-Kmeans, MIP-Kmeans, FeatTS, and DCC) are used as baselines; these algorithms were explained in detail in Chapter 2, Section 2.2.6. Since CDPS takes constraints during training, none are given to the algorithms that use CDPS' representation. The normalised Mutual Information (NMI, defined in Chapter 2 Equation 2.5) is used to measure clustering performance between the true and predicted labels, where zero signifies no mutual information and one a perfect correlation with the ground truth labels.

The study is carried out on the UCR repository [12, 174], which is a well-known archive for evaluating time series classification and clustering algorithms. The archive is comprised of a diverse collection of time series datasets covering various domains, including human activity recognition, medical diagnostics, environmental monitoring, sensor data analysis, synthetic data, and more. The datasets have different characteristics, such as different temporal patterns (trend, cycle, etc.), the number of classes, the length of the time series, and different distributions (different sampling, variability of the domain,). All the datasets are split into train and test sets, the train and test sets may have different numbers of samples and the classes may or may not be imbalanced (the properties of these datasets are presented in Table B.1 of Appendix B). These datasets, therefore, represent a wide variety of potential real-world applications and offer a means to demonstrate an algorithm's ability without focusing directly on an application domain. The reader is referred to articles published by Bagnall et al. [175] and Bagnall et al. [176] that describe the univariate and multivariate datasets (respectively) found in the archive in detail.

Due to the computational complexity of COP-Kmeans and MIP-Kmeans, thirty-five univariate and fifteen multivariate datasets were chosen at random. In total fifty datasets are used to evaluate the performance of CDPS in comparison to COP-Kmeans and MIP-Kmeans. The randomly chosen datasets encompass a wide range of domains, representing different distributions and exhibiting varying severity and complexity in terms of distortions. Since CDPS and DCC are more computationally efficient, their performance on all the datasets was evaluated and is presented in Appendix C (the details of the datasets such as the number of samples per training and test sets, the number of classes and dimensions are provided in Appendix B). Testing on all the datasets investigates the performance of CDPS on a wide range of applications where the results showed that CDPS outperforms DCC in most of the results. Using the UCR archive will allow us to investigate the capabilities of CDPS by analyzing its performance in comparison to other approaches. We can identify scenarios where CDPS outperforms alternative methods and recognize situations where it may fall short. This comprehensive evaluation using the UCR archive will help us to gain insight into the CDPS' weaknesses and strengths.

Three use cases are evaluated. In the first, termed **Transductive Clustering**, the training and test sets (as given by the repository) are combined, this reflects the real-world case in which a dataset is to be explored and knowledge extracted. In the second, termed **Inductive Clustering**, the embedding is learnt on the training set and its generalisation is evaluated on the test set. This inductive use case is not normally possible with constrained clustering algorithms since clustering is a transductive operation. This highlights a key contribution of CDPS, its ability to generalise constraints to unseen data. The third use case termed **Representation Learning**, highlights the importance of CDPS shapelets as general features able to be integrated into other downstream algorithms. As such, FeatTS' semi-supervised statistical features are replaced with the dataset's CDPS embedding. The algorithms are evaluated with varying constraint levels, represented as a percentage of constrained samples: 5%, 15%, 25%. These represent a very small fraction of the total possible constraints, $\frac{1}{2}N(N-1)$. Experiments are repeated 10 times with different random constraint sets, and each clustering algorithm is repeated 10 times per constraint set (i.e. 100 repetitions for each constraint percentage[1]). Constraints are generated by randomly selecting two samples, and adding a constraint

---

[1] Despite multiple initialisations, some COP-Kmeans results did not converge when applied to certain constraint sets, resulting in fewer repetitions.

(ML/CL) depending on their class labels until the desired number of constraints is reached.

Training is carried out using mini-batch gradient descent, batch size $s_{\text{batch}} = 64$ and number of constraints in batch $c_{\text{batch}} = 16$ for the transductive setting, and $s_{\text{batch}} = 32$ and $c_{\text{batch}} = 8$ for the inductive setting (since there are fewer samples). Experiments were performed on the high-performance computing cluster provided by the University of Strasbourg. The influence of α and γ on accuracy were evaluated and the algorithm showed stability in most cases, leading to a fixed value of 2.5 for both. The minimum shapelet length $L_{min} = 0.15 \cdot L_i$, and the maximum number of shapelets $S_{max} = 3$ are taken following those in LDPS [32]. All models (CDPS) are trained for 500 epochs using the Adam optimiser. The DCC implementation and parameter values were taken from [157]. Note that more details about model performance and an investigation into the model's sensitivity and space is presented in Section 3.4.3.

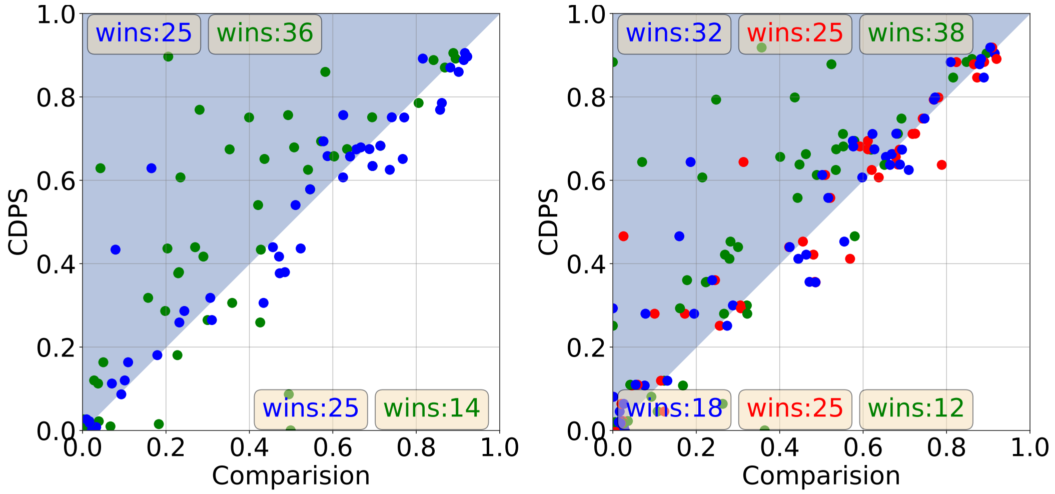
3.3.2 Clustering

We now present the results for each use case described earlier and discuss them in detail. In general, the results are expected to be favourable when using CDPS compared to using the clustering algorithm directly. This expectation arises from the fact that CDPS aims to learn a discriminative space based on expert intuition. When using CDPS, the data samples are transformed into a representation that encourages separate and concentrated clusters following the constraints provided by the expert. This characteristic aligns well with the requirements of clustering algorithms, which tend to perform better when the data samples form distinct and well-defined clusters. Hence, we anticipate that CDPS will enhance the performance of the clustering algorithms when compared to using them directly. By evaluating the results of each use case, we can further analyze and validate these expectations.

Transductive Clustering

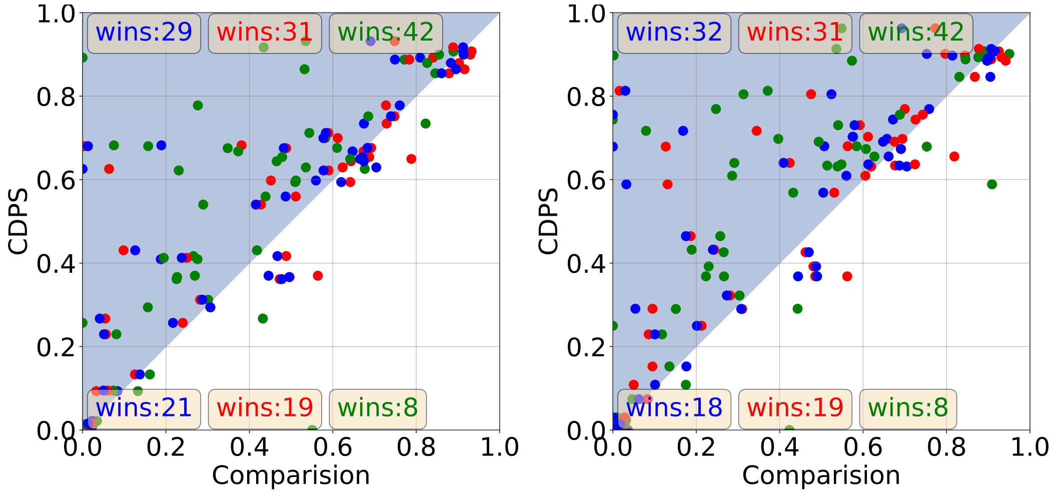
Figure 3.7 summarises the Transductive results (the full results presented in Appendix C, Table C.1). The figure shows the NMI scores for CDPS (Euclidean K-means performed on the CDPS embeddings) compared to the other algorithms applied to the raw time series (COP-Kmeans, MIP-Kmeans, and DCC), LDPS (CDPS with no constraints reduces to LDPS) and (unconstrained) K-means are presented as a basis for the constrained algorithms to study how constraints benefit each.

As expected LDPS and K-means offer similar performance, see Figure 3.7a, although some datasets do favour the unconstrained K-means algorithms, outperforming LDPS. Nevertheless, CDPS efficiently uses the information given by constraints to outperform the other algorithms in almost all the different constraint fractions and datasets.



(a) LDPS / CDPS 0%

(b) 5%



(c) 15%

(d) 25%

Figure 3.7: A Transductive comparison between CDPS+K-means (y-axis) and Raw-TS+(constrained K-means) (x-axis) with different constraint fractions. Each dot represents a dataset from the UCR archive, blue dots represent COP-Kmeans performance and red dots MIP-Kmeans performance. The datasets located in the blue triangle are those for which CDPS performs better than COP-Kmeans and MIP-Kmeans, and the white triangle is the opposite. The term "wins" indicate the number of datasets in which either CDPS have a higher score compared to the comparison (in this case wins is located on top of each figure) or vice versa (wins located at the bottom).

CDPS performance increases with the number of constraints, whereas COP-Kmeans and MIP-Kmeans tends to stagnate. This can be seen as the cloud of points move upwards (CDPS' NMI score increases) as more constraints are given. DCC tends to work better for the datasets that have low NMI scores for COP-Kmeans and MIP-Kmeans. Indicating complex or similar structures with which DTW struggles to capture the detailed differences. In contrast, DCC is able to find more discriminative features beyond the structural similarity of the time series, resulting in improved performance. We can also observe that for some datasets the constrained algorithms behave similarly with 5% constraints, where COP-Kmeans and MIP-Kmeans have almost similar performance while DCC is a bit worse. Again CDPS benefits the most from increasing the number of constraints and significantly outperforms the comparison algorithms with larger constraint percentages.

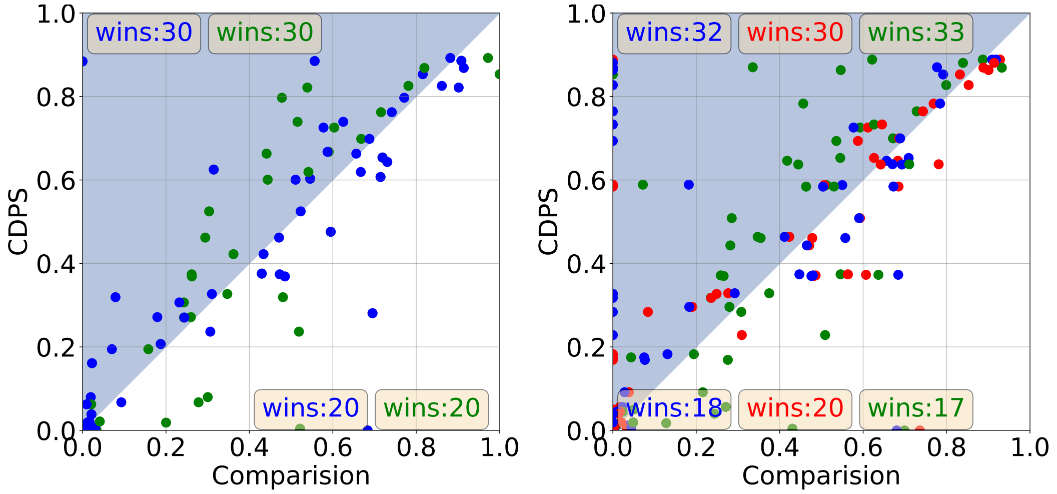
Overall COP(MIP)-Kmeans perform similarly, however, MIP-Kmeans was able to converge on datasets for which COP-Kmeans was not. COP-Kmeans fails if a constraint is violated, however, MIP-Kmeans's formulation as a mixed-integer programming optimisation allows it to search the space for an optimal or near-optimal solution. Furthermore, CDPS exploits constraints better than other algorithms, and DCC seems to exploit them the least. For some datasets, all algorithms perform equally (see top right and bottom left corners) which can be due to the fact that the datasets are either very difficult (hence low NMI) or very easy to cluster.

Inductive Clustering

Figure 3.8 summarises the Inductive results (the full results are presented in Appendix C, Table C.4), i.e. embedding learnt on the training set, generalisation performance evaluated on the unseen test set. Note that for the same constraint percentage, there are significantly fewer constraints than in the transductive setting.

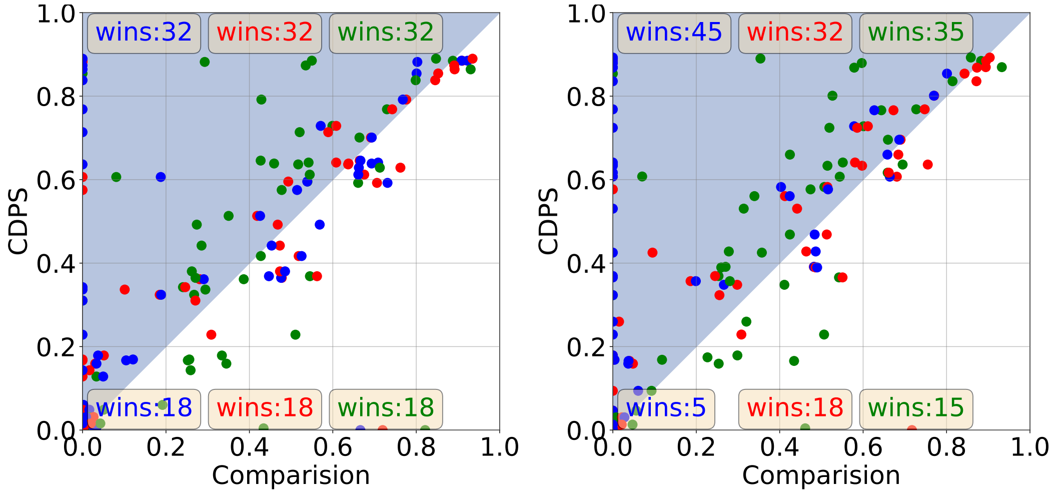
It can therefore be concluded that even with fewer data and constraints, CDPS is still able to learn a generalisable representation and attain (within a certain margin) the same clustering performance as when trained with the merged datasets. This is probably explained by the fact that having fewer samples and constraints means they are repeated in the mini-batches (see Section 3.2.2). Allowing CDPS to focus on learning shapelets that are highly discriminative and preserve DTW rather than shapelets that model larger numbers of time series. Thus, the resulting representational space better adheres to the constraints, allowing better clustering of unseen time series.

It can also be observed that COP-Kmeans tends to struggle with a small number of constraints, since it increases the risk of constraint violations, while MIP-Kmeans is able to overcome this. In this setting, DCC outperforms both COP-Kmeans and MIP-Kmeans for some of the challenging datasets (low NMI – lower left corner) and shows similar performance to the other algorithms on less complex data (top right corner).



(a) LDPS / CDPS 0%

(b) 5%



(c) 15%

(d) 25%

Figure 3.8: An inductive comparison between CDPS+K-means (y-axis) and Raw-TS+(constrained K-means) (x-axis) with different constraint fractions. Each dot represents a dataset from the UCR archive, blue dots represent COP-Kmeans performance and red dots MIP-Kmeans performance. The datasets located in the blue triangle are those for which CDPS performs better than COP-Kmeans and MIP-Kmeans, and the white triangle is the inverse. "wins" indicates the number of datasets in which either CDPS has a higher score compared to the comparison (in this case wins is located on top of each figure) or vice versa (wins is located at the bottom).

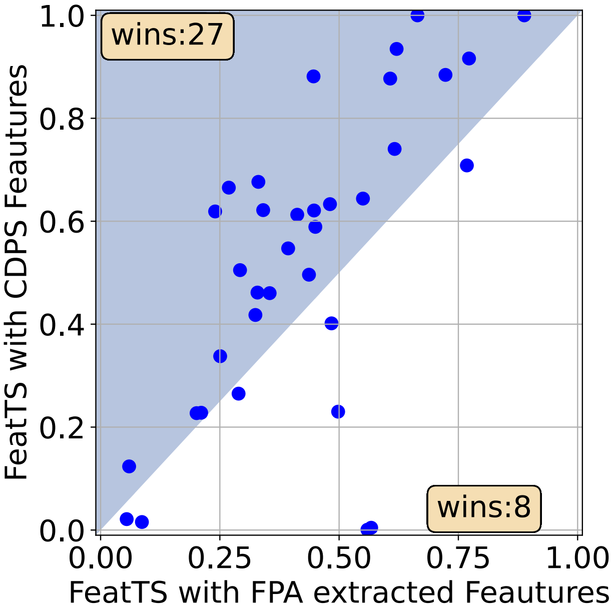


Figure 3.9: A comparison between CDPS+FeatTS and FeatTS with respect to NMI score. The top wins, indicates how many datasets FeatTS with CDPS features outperforms FeatTS, and the number of FeatTS wins is located at the bottom.

Representation Learning

In this section, we aim to demonstrate that the CDPS representation is versatile and compatible with various downstream algorithms. To achieve this, we compare CDPS with FeatTS. Firstly, let's provide a brief overview of FeatTS (more information can be found in Section 2.3). FeatTS is a semi-supervised approach that extracts statistical features from labelled data. For this study, we will use 25% labelled samples which is a lot more than the information contained in 25% constraints used to train CDPS. These features are used to generate a co-occurrence matrix, which, in turn, is used with a K-medoids clustering algorithm to cluster the data. In order to compare the features obtained from FeatTS with CDPS, we employ the following approach. We provide the FeatTS algorithm with the CDPS representation as input features. By doing so, we allow FeatTS to generate the co-occurrence matrix and perform clustering using the CDPS representation. This comparison serves as a proof of concept, demonstrating the versatility of the CDPS representation and its possible compatibility with different downstream algorithms. It allows us to evaluate and compare the performance of FeatTS when operating on CDPS features.

Figure 3.9 shows the NMI scores of CDPS+FeatTS (FeatTS using CDPS shapelets as features) and the original FeatTS algorithm, carried on the 35 univariate datasets (since FeatTS is limited to univariate time-series). We observe that, out of 35 datasets, CDPS+FeatTS outperforms FeatTS in 27. Indicating that CDPS' shapelets are more informative for clustering tasks when compared to FeatTS' statistical features. For the datasets that achieved around zero NMI with respect to CDPS+FeatTS while high NMI with FeatTS (e.g. with the GunPointAgeSpan dataset CDPS+FeatTS gives 0.001, while FeatTS gives 0.559), it appears that the shapelets learnt in these cases are not discriminative enough, which is confirmed by CDPS' low scores (CDPS+K-means: 0.004).

3.4 Discussion

This section discusses the results, describes the model selection strategy and investigates the model's sensitivity to several parameters: α, γ, the number of shapelets, and the length of shapelets. Furthermore, we present an exploration of the representational space. Specifically, we measure the coherence of constraints (refer to Definition 9, Section 8) It should be noted that this coherence measurement could not be calculated for classical time series data when employed with elastic measures like DTW.

3.4.1 Results

Due to the fact that LDPS only models DTW distance, its induced clustering performs approximately equal to K-means with DBA averaging, as shown in Figure 3.8a, in which the results are clustered around the diagonal. However, both LDPS and CDPS result in a metric space, which is beneficial for further analysis and processing. For example, in a constrained clustering setting, it will be possible to measure the coherence property of the pairwise constraints.

Generally, the results shown earlier conclude that CDPS is more capable of exploiting the information contained in the constraints when they are introduced, giving more accurate clustering results overall. Furthermore, CDPS avoids the problem of non-convergence that can arise with hard-constrained clustering techniques such as COP-Kmeans, which is significantly more challenging when using an elastic measurement such as DTW. In these experiments, all constraints can be considered 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.

While K-means-based methods were included in this study for the purpose of comparison, it is not a common practice to use them in an inductive use case for classical clustering algorithms. The inductive setting was simulated by providing COP-Kmeans and MIP-Kmeans with the combined 'training' dataset, along with its associated constraints, and the test data to be clustered (without constraints). However, this approach introduces certain challenges. The need to store and access this entire dataset can limit the feasibility of such use cases, not to mention the significantly high computational cost involved. In contrast, CDPS offers a truly inductive approach in which new data can be projected into the resulting space, which inherently models user constraints. This demonstrates the new possibilities that can arise from such 'constrained representations' since CDPS' embedding can be used for tasks other than clustering (classification, generation, etc.). By leveraging CDPS, we open up new possibilities and overcome the limitations associated with traditional clustering algorithms. CDPS' inductive complexity (once the space has been determined) is $\mathcal{O}(NL_kK)$, plus K-means' complexity $\mathcal{O}(NMKi)$, where M is the number of clusters, K is the number of shapelets, L_k is the length of the shapelets, i is the number of iterations until convergence, and the complexity of COP-Kmeans is $\mathcal{O}(NMKi|\text{ML} \cup \text{CL}|)$.

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. In absolute terms, the performance of other algorithms tends to decrease or stagnate as more constraints are introduced, while it increases with CDPS. These constraints bias CDPS to find shapelets that define a representation that respects both DTW and the constraints. Although the focus of this work 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 added.

Although CDPS outperforms the competitors in the majority of the datasets, as discussed, it still falls short in a number of datasets. This suggests that CDPS exhibits certain limitations. Looking at the CDPS formulation these limitations arise when dealing with complex datasets that lead to noisy DTW similarity, therefore affecting its performance. As described in Section 3.2.1, CDPS aims to find a balance in the neighbourhood of a sample based on the different forces, that approximate DTW (blue), that respect the cannot-link (red) and the must-link (green), as shown in Figure 3.4. If the dataset contains complex information with respect to the DTW distance, this can introduce conflicting information during training and may not guarantee a balance for the different forces. This issue will be further explored in a failure case in Chapter 4, Section 4.3. Additionally, there are other weaknesses that have not been addressed in CDPS. First, CDPS is not compatible with sparse time series since it does not consider missing values. Second, it is not suitable for time-dependent feature analysis because the shapelet transform used in CDPS is time-independent. Finally, CDPS's loss function does not inherently enforce dissimilarity between shapelets, which may result in multiple similar shapelets being learned. This is mitigated by the proposed shapelet ranking approach, Shapelet Cluster Explanation (SCE), that will be introduced in Chapter 4.

It is also conceivable to use a constrained clustering algorithm with CDPS embedding. Although this was not studied, it would allow another mechanism to integrate constraints after the embedding has been learnt.

3.4.2 Model Selection

During unsupervised learning, there is typically no validation data available to establish a definitive stopping criterion. Therefore, it becomes crucial to carefully examine the behaviour of CDPS during training in order to provide some useful

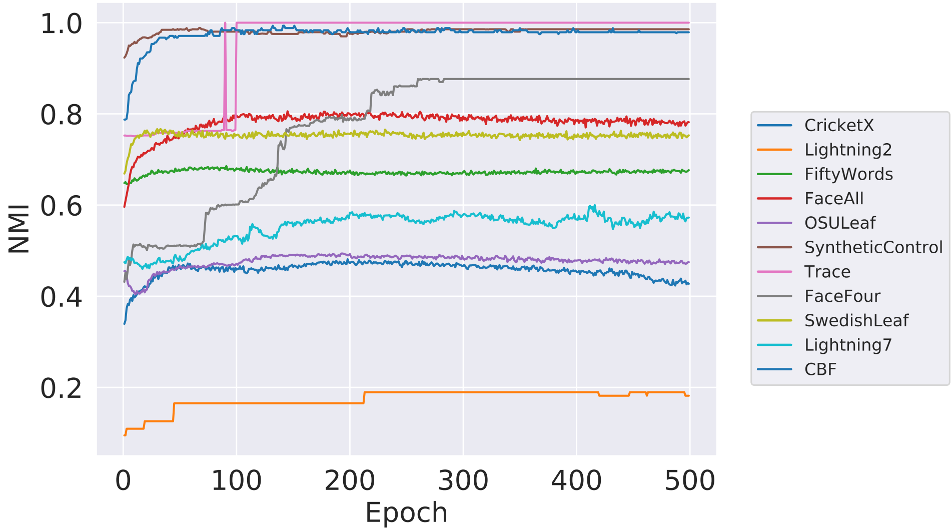
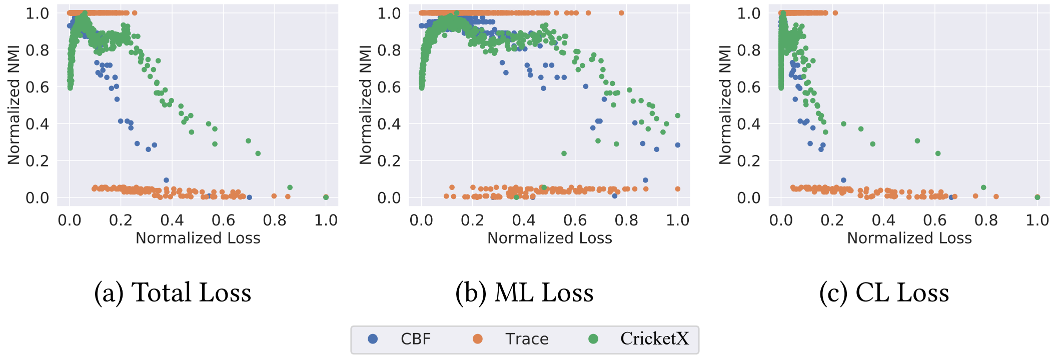


Figure 3.10: Clustering quality (NMI) as a function of the number of epochs for each dataset, using a constraint fraction of 30%.

guidelines for when to stop the training process. This can be done by analyzing the training dynamics and observing the performance trends of CDPS across different datasets. Such an analysis should help us achieve a satisfactory representation that leads to improved clustering results using CDPS. In what follows clustering using K-means (under the transductive setting) on the CDPS representation is used to evaluate and study CDPS performance during training. Different datasets are chosen at random, and the CDPS training is carried out with a 30% constraint fraction.

Figure 3.10 illustrates the clustering quality (measured by Normalized Mutual Information, NMI) of CDPS as the number of epochs progresses for different datasets. The figure reveals that the majority of models converge within a relatively small number of epochs, except for the FaceFour dataset, which requires more epochs to reach convergence. An interesting observation is that the quality of the learned representation does not deteriorate as the number of epochs increases. This indicates that neither the preservation of DTW distance nor the influence of constraints dominates the loss function to the extent that it diminishes the other. In other words, the competing objectives of preserving temporal relationships and incorporating user constraints remain balanced throughout the training process and reach an equilibrium (as explained in Section 3.2.1), resulting in a consistent and robust representation quality. This finding is significant as it suggests that increasing the number of epochs does not lead to overfitting or a decline in the overall performance of CDPS. Instead, it reinforces the reliability of CDPS in maintaining the desired characteristics of both DTW and constraints, leading to stable and good clustering results.

Figure 3.11 presents scatterplots showcasing the relationship between the NMI score and CDPS loss for multiple datasets. Both NMI and CDPS loss values are



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

Figure 3.11: 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.

normalized between zero and one. In addition to the scatterplot of the total loss, the individual losses for Must-Link (ML) and Cannot-Link (CL) constraints are also included (Figures 3.11b and 3.11c respectively). An evident trend observed in all the scatterplots is that a lower overall loss corresponds to a higher NMI. This indicates that as the CDPS loss decreases, the clustering quality, as measured by NMI, tends to improve. The correlation between the two metrics suggests that minimizing the loss function during training is sufficient to yield a better alignment between the learned representation and the underlying data structure.

Both these studies (NMI clustering score with respect to the epoch and normalized loss) show the importance of optimizing the loss function on achieving better cluster results and that it 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.

3.4.3 Sensitivity Study

Now, we will analyze the sensitivity of CDPS with regard to its various parameters. By investigating the impact of these parameters on the performance and behaviour of CDPS, we will be able to test the ability of the model to converge to a stable solution. The parameters to analyze are α, γ, the number of shapelets, and the length of shapelets. By systematically varying these parameters and observing their effects on CDPS, we can understand the extent to which each parameter influences the quality and characteristics of the learned representation. This analysis will provide valuable insights and recommendations for selecting optimal parameter values, guiding the user in making informed decisions when applying CDPS.

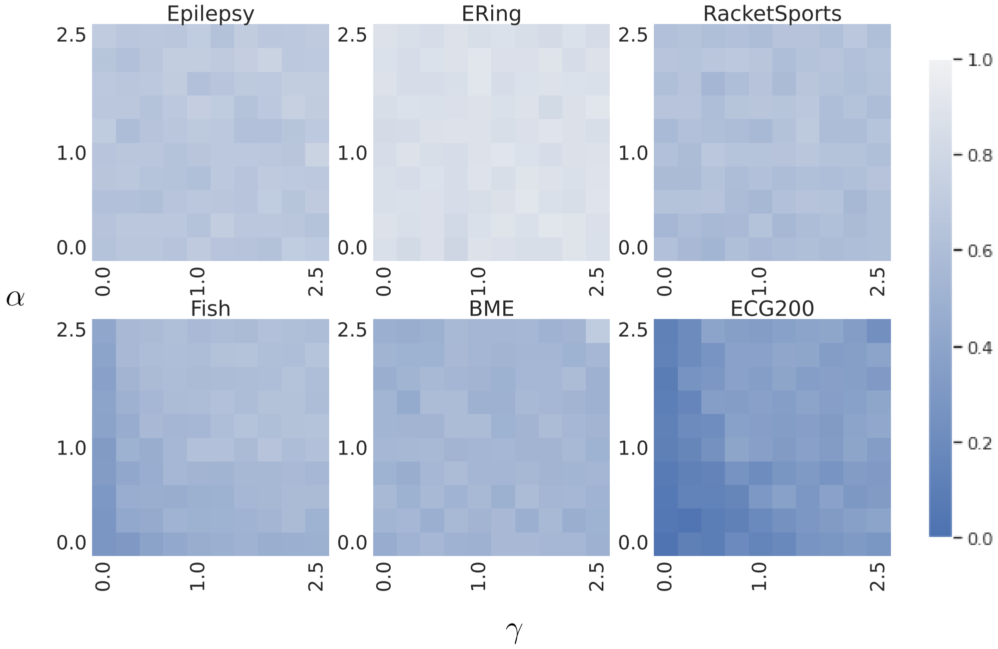


Figure 3.12: Heatmaps showing CDPS sensitivity to α and γ parameters, using 6 random univariate datasets (brighter colours indicate better performance).

Effect of α and γ

To test the stability of the learning process to different values of α and γ parameters, a grid search was performed over the range 0, 0.5, 1, 1.5, 2, 2.5. Each pair of parameter values was tested with 10 repetitions on a fixed set of constraints. Both univariate and multivariate datasets from the UCR archive were randomly selected.

Figure 3.12 presents these results, in which the clustering scores are presented for each pair of values (note that brighter colour indicates better performance). It can be observed that the impact of parameter values varies across different datasets. For datasets such as Fish (a one-dimensional representation of the fish shape by tracing its outline into a one-dimensional line), ECG200 (electrical activity of a heartbeat), and BME (synthetic dataset with three classes characterized by the presence of a peak in the series either at the beginning or end or not present.), higher values of the parameters lead to improved performance. However, for other datasets, the influence of these parameters is not as significant. As such, taking $\alpha = \gamma = 2.5$ (as in the previous experiments) is reasonable since they result in high scores over the majority of datasets, and hence CDPS can be considered robust with respect to α and γ values. This means that the user does not need to optimise their values, nor needs to have in-depth knowledge of their meaning to achieve state-of-the-art performance.

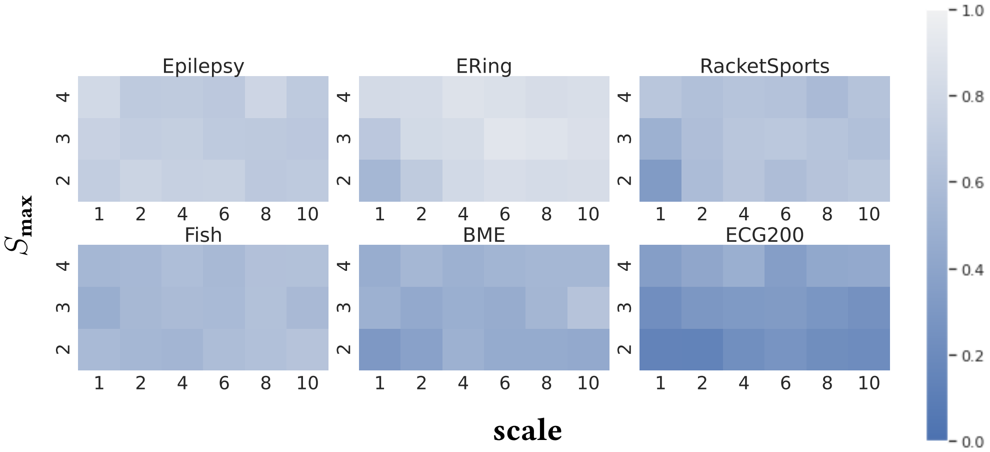


Figure 3.13: Heatmaps of CDPS sensitivity to shapelet parameters (brighter colour indicates better performance).

Effect of Shapelet Parameters

The shapelet parameters in CDPS consist of the number of shapelet blocks ($S_{\max}$), the minimum shapelet length ($L_{\min}$), and a multiplier that determines the number of shapelets per block (scale). This study aims to investigate the impact of the shapelet parameters, which are typically user provided, by determining if there is a saturation in the clustering performance induced by the CDPS representation after a specific set of parameter values. As mentioned in Section 3.3.1, the parameters of the shapelets are specified using the same rule proposed by LDPS [32]. That is, the number of shapelets per block are determined according to the following rule $\log(L_i - L_{\min} \cdot b_{\text{ind}}) \cdot$scale, where $b_{\text{ind}} \in \{1, \ldots, S_{\max}\}$ indicating the block number. In order to evaluate the sensitivity of the model to the parameters, we explore $S_{\max}$ and scale using the values $\{2, 4, 3\}$ and $\{1, 2, 4, 6, 8, 10\}$ respectively and we fix $L_{\min}$ to $0.15 \cdot L_i$ (since shapelet length is directly related to the block number).

Figure 3.13 displays the clustering scores for different combinations of $S_{\max}$ and scale; brighter colours indicate better performance. It can first be observed that for the majority of datasets, the clustering performance is often the same for multiple parameter combinations. This implies that we can choose a smaller multiplier for the shapelets per block as well as a smaller value for the number of shapelet blocks and still guarantee a similar clustering score. For instance, clustering on Epilepsy produced similar results for (4,1) and (4,8), but since (4,1) offers less complexity, it would be preferred. In addition, we can see that for some datasets, the performance degrades as the number of shapelets increases (whether in terms of S_{smax} or scale), as seen in the cases of BME, RacketSports, and EGC200.

Taking this study into account, we can infer that choosing the values for the maximum number of shapelet blocks and the scale as $S_{\max} = 3$ and scale $= 10$ generally results in performances on par with the state-of-the-art regardless of the dataset, even though fine-tuning these parameters might lead to better results. In conclusion, the investigation of shapelet parameters highlights the robustness of

CDPS and provides insights into choosing suitable values that offer competitive clustering performance without the need for extensive parameter optimization.

3.5 Conclusions

In this chapter, we have introduced the first contribution of this thesis, called Constrained DTW Preserving Shapelets (CDPS). We began by discussing the concept of contrastive learning and contrastive loss, which aims to learn information from pairs of data samples. We explained how CDPS uses this concept to incorporate user constraints in the form of pairwise constraints (must-link and cannot-link constraints) into the learning process of LDPS. This allows shapelets to be learnt that respect the user constraints while approximating the DTW distance measure.

We evaluated the accuracy of performing clustering on the CDPS representational space, in order to assess the quality of the learnt space, and compared it to other algorithms. K-means and LDPS were used as baselines for use cases without constraints information, while COP-Kmeans, MIP-Kmeans, and DCC were used as comparisons for constrained clustering algorithms. The results demonstrated that CDPS consistently outperformed the competitors in both transductive and inductive settings. The transductive setting is the classical approach used in clustering while the inductive setting means that constraints on one dataset are generalised to another. This showed that CDPS effectively incorporates the constraint information and exhibits generalization capabilities to unseen data. Furthermore, we provided a comparison with another semi-supervised algorithm, FeatTS, to showcase the potential of CDPS in enhancing the performance of downstream algorithms, where CDPS exhibited superior performance in most cases.

We also discussed the model selection strategy and sensitivity, revealing that the loss function can serve as a stopping criterion and that the algorithm is relatively robust to changes in the parameters. The representation of the CDPS space was examined, demonstrating its discriminative nature compared to LDPS and DTW-based representations. We also provided a discussion on the weakness of CDPS such as its inability to work with sparse datasets or time-dependent feature analysis, and its possible limitations due to the complexity of the dataset and therefore the possibility of conflicting information being introduced when using DTW distance.

CHAPTER 4

SHAPELET CLUSTER EXPLANATION

If you step back a little and say we want to do
A.I., then you will realize that A.I. needs
knowledge, reasoning, and explanation.
– Oren Etzioni

4.1 Explanation Framework Definitions 78
4.2 Cluster Explanation: A Case Study 85
  4.2.1 Shapelet Selection 90
4.3 Effect of Constraints on the Representation 93
4.4 Conclusions 102

Having an explanation for a clustering algorithm's results is important across various domains. Clustering algorithms are designed to uncover patterns and group similar instances together. However, when dealing with complex time series data, interpreting the underlying reasons behind clustering outcomes can be challenging. To tackle this challenge, one approach is to transform the time series data into a more interpretable space that captures essential features while discarding irrelevant ones. In the previous chapter, we introduced CDPS (Constrained DTW Preserving Shapelets) a method that achieves such a transformation but also includes expert intuition in the learning process. CDPS uses shapelets, which are interpretable subsequences that capture key characteristics of the time series while approximating DTW distance and preserving distortions. The transformed space, facilitated by CDPS, provides an interpretable representation of the time series data and offers insights into the "why" behind clustering results, which is more significant for the expert than just providing a validation of the clusters.

In this chapter, we will leverage the interpretability of shapelets to propose a framework that ranks them based on their discriminative ability to provide insights into the clustering algorithm and individual clusters. By extending previous works that individually rank shapelets, such as the work of Ye and Keogh [7], Zakaria et al. [42], Ulanova et al. [43], etc, this ranking identifies the most informative set of shapelets that accurately reflect a particular clustering scenario or the overall clustering. Additionally, it may enable us to reduce the number of shapelets required while achieving comparable clustering performance.

We will first provide the theory behind the framework followed by a case study. An in-depth study of the importance of the reduced space achieved by the Shapelet Clustering Explanation (SCE) framework is also presented at the end of this chapter. In Section 4.3 we will study the effects of constraints, by studying coherence and the average distance between must-link and cannot-link constraints ML/CL. Coherence can now be calculated accurately since CDPS results in a metric space, and ML/CL will assess the influence of the constraints on the compactness and separation of the space. The effect of constraints on the learned space is compared to the equivalent space learnt using Learning DTW Preserving (LDPS, i.e. without constraints) and to dimensionality reduction on the DTW distance map. We will also study if the influence of the constraints (coherence and ML/CL) persists when using a subset of shapelets that are ranked according to the SCE framework.

4.1 Explanation Framework Definitions

In this section, the necessary background and notations are briefly explained and the Shapelet Clustering Explanation (SCE) method is introduced. In most of the shapelet approaches, ranking shapelets has been widely used as a means of searching for the shapelets from a set of all possible shapelets. Previous studies, such as the work of Ye and Keogh [7], used an independent ranking method (as explained in Chapter 2, Section 2.1.2) which ranks the shapelets by their ability to split the data into more homogeneous groups. Information gain is used to quantify the quality, treating each shapelet individually. In our work, we will first present this independent ranking of shapelets as introduced by Ye and Keogh [7] . This will then be extended to two alternative approaches: combined ranking (ranks shapelets by combining different shapelets and studying the space formed by them) and successive ranking (also studying the space formed by different combinations of shapelets but measures how much information is added when an additional shapelet is included). These are contributions of this thesis that extend independent shapelet ranking to dependent ranking based on the combined information of the shapelets. These two novel approaches will be discussed in detail later in this chapter.

In order to rank the shapelets, we need a quality measure capable of assessing the separation of the samples into groups of samples similar to the shapelets and those that are not similar, hence assessing the discriminative capability of the shapelet. Hills et al. [45] tested with a number of quality measures (such as Information Gain [177], Kruskal-Wallis [178], etc.), and showed that there is no significant difference in performance between them. Out of the tested measures, Information Gain (IG) is the only one that is capable of assessing the quality of partitioning the data (using the shapelets) into groups that aim to model the clustering process, hence assessing the quality of the shapelets. In this work, as with the original independent ranking approach, we will use IG to quantify the quality of the shapelets. But first, the definitions necessary to understand information gain (IG) and the criteria to split the data are presented.

Definition 10 (Entropy) *Measures the randomness, thus purity, of a given set of class labels: homogeneous cluster labels give zero entropy, and uniformly random samples give the maximum value. Assuming a dataset $\mathcal{T}$ with $\mathcal{C}$ classes and N instances, each class c_i with probability $p(c_i)$ contains n_i samples. With $\ln$ the natural logarithm, $\mathcal{T}$'s entropy is defined, such that:*

$$E(\mathcal{T}) = -\sum_{i=1}^{C} p(c_i)\ln(p(c_i)), \quad s.t. \begin{cases} \sum n_i = N, \\ p(c_i) = \dfrac{n_i}{N}, \end{cases} \tag{4.1}$$

where $\ln$ is the natural logarithm. Note that any logarithm $\log_a$ can be used.

Definition 11 (Information Gain) *Measures the reduction of entropy, i.e. information gained, in a dataset after being partitioned into distinct groups based on a specific splitting criterion (e.g. threshold). It is evaluated by calculating the entropy of the dataset before and after the split. The Information Gain (IG) of a dataset $\mathcal{T}$ with split ζ is defined such that*

$$IG(\mathcal{T},\zeta) = E(\mathcal{T}) - E(\mathcal{T})|_\zeta, \tag{4.2}$$

where $E(\mathcal{T})|_\zeta$ is the entropy calculated after the split. Suppose D is partitioned into two sets $\mathcal{T}_l$ and $\mathcal{T}_r$ based on the split ζ, therefore

$$E(\mathcal{T})|_\zeta = \frac{N_l}{N}E(\mathcal{T}_l) + \frac{N_r}{N}E(\mathcal{T}_r), \quad s.t. \begin{cases} N_l = |\mathcal{T}_l|, \\ N_r = |\mathcal{T}_r|. \end{cases} \tag{4.3}$$

The shapelets learned by CDPS can be ranked using IG according to their ability to reproduce (i.e. explain) a full clustering, termed 'Global-Wise' explanation (GE), or a specific cluster, termed 'Cluster-Wise' explanation (CE). The term 'reproduce' refers to the ability to recreate the clustering output (GE) or a specific cluster therein (CE), when the time-series dataset is projected into the space defined by one or more shapelets (ranked according to their IG) by linear separation. This separation is performed using Linear Discriminative Analysis (LDA) [179]. LDA is used because it provides a linear separation of the data samples according to the cluster labels (cluster labels $\mathcal{C}$ are considered as the class labels for the purpose of LDA) by maximizing the separation between the clusters and minimizing the within-cluster variance, leading to effective linear decision boundaries. The IG is calculated on LDA's output predictions (partitioning) using the labels given in $\mathcal{C}$ as the ground labels.

More specifically, given a set of N time-series $\mathcal{T}$, a set of K learned shapelets $\mathcal{S}$, and a partitioning of $\mathcal{T}$ into M clusters $\mathcal{C} = \{C_1, \ldots, C_M\}$ (i.e. the result of an unsupervised clustering algorithm applied to the CDPS learnt representation), the goal is to rank the shapelets in importance for giving the best CE using C_i (GE using $\mathcal{C}$). In this manner, if a certain subset of shapelets is highly representative of C_i ($\mathcal{C}$), i.e. is highly ranked by IG, it allows the user to visually perceive how the cluster (clustering) came to be formed. Furthermore, once the shapelets have been ranked accordingly, the best d-shapelets, $\mathcal{S}_d$ ($|\mathcal{S}_d| = d$) can be adjusted according to further user needs (e.g. to reduce dimensionality, remove duplicate shapelets, etc). It should be noted that in the following formulations, ζ is an operator that partitions the space into M partitions if GE is employed or two partitions if CE is used, i.e. the partitioned space is denoted by $D = \{D_m\}_{m=1\ldots M}$, such that D_m should be representative of C_m. Thus, with CE, a thresholding criterion is used for ζ. Alternatively, with GE, Linear Discriminative Analysis is used for ζ, in which the cluster assignments $\mathcal{C}$ are treated as class labels and LDA splits the data according to the linear decision boundaries found.

Three approaches for ranking the shapelets, irrespective of the objective (either CE or GE), can be identified depending on the desired result.

Independent. In which the IG of each shapelet is examined independently of the other shapelets. Hence, $\mathcal{S}_d$ is defined such that:

$$\mathcal{S}_d = \{S_j \mid IG(\mathcal{T}, \zeta^j) > IG(\mathcal{T}, \zeta^i)\}, \tag{4.4}$$

where $IG(\mathcal{T}, \zeta^i) > IG(\mathcal{T}, \zeta^{i-1})\ \forall S_i, S_j, S_{i-1} \in \mathcal{S},\ S_i \neq S_j$, such that $\zeta^i = \zeta(\mathcal{T}, S_i)$ partitions the dataset $\mathcal{T}$ in the space defined by the shapelet S_i into D_i. Similarly $\zeta^j = \zeta(\mathcal{T}, S_j)$ and $\zeta^{i-1} = \zeta(\mathcal{T}, S_{i-1})$ partition the dataset $\mathcal{T}$ in the space defined by shapelets S_j and S_{i-1} into $\mathcal{D}_j$ and $\mathcal{D}_{i-1}$ respectively. In this way, all shapelets are ranked according to their IG.

Combined. Ranking the shapelets independently as above does not mean that when the top-d shapelets are combined to form a space they result in the highest possible Information Gain, i.e. results in the best linearly separable distribution, out of all the possible combinations of the d shapelets. Finding this space is NP-Hard, the combined approach, therefore, uses an exhaustive recursive search strategy to find the shapelet that adds the highest Information Gain[1], S_l, at each iteration to build the d-dimensional space. In this way, the basis of the transformed space is recursively constructed such that:

$$\mathcal{S}_d^j = (\mathcal{S}_d^{j-1}, S_l), \text{where} \begin{cases} \mathcal{S}_d^0 = (\phi), \\ IG(\mathcal{T}, \zeta_d^j) > IG(\mathcal{T}, \zeta_d^{j-1,i}), \end{cases} \tag{4.5}$$

and where $\mathcal{S}_d^j$ is the ordered list of the j best shapelets found so far, such that $\zeta_d^j = \zeta(\mathcal{T}, \mathcal{S}_d^j)$ and $\zeta_d^{j-1,i} = \zeta(\mathcal{T}, (\mathcal{S}_d^{j-1}, S_i))$ partition the dataset $\mathcal{T}$ in the space defined by $\mathcal{S}_d^j$ and $(\mathcal{S}_d^{j-1}, S_i)$ into $\mathcal{D}_j$ and $\mathcal{D}_{j-1,i}$ respectively, $l, i \in \{1, \ldots, K\}$, $i \neq l$, and $S_i, S_l \in \mathcal{S}$, $S_i \notin \mathcal{S}_d^j$.

Successive. Besides optimising linear separability and modelling the clustering result, the combined strategy does not impose diversity in the shapelet ordering. For example, assume that a shapelet is selected using the combined strategy and has the same IG as the previously discovered set of shapelets, this equality in IG does not necessarily imply that the selected shapelet is diverse nor increases separation in the space. The successive strategy alters the combined strategy by only calculating the Information Gain of a set of samples $\delta_j \subset \mathcal{T}$ whose partitioning differs or are incorrectly identified, after adding an additional shapelet, S_l, at each iteration, such that

$$\begin{aligned} \delta_j &= \Delta_{j,j-1} \cup \iota_j \\ \delta_j^{'} &= \Delta_{(j-1,i),j-1} \cup \iota_{j-1,i}, \end{aligned} \tag{4.6}$$

where Δ indicates the difference between the current partitioning $\mathcal{D}_j$ and the previous partitioning $\mathcal{D}_{j-1}$, such that:

$$\begin{aligned} \Delta_{j,j-1} &= \bigcup_{m=1}^{M} \mathcal{D}_{j,m} \setminus \mathcal{D}_{j-1,m}, \\ \Delta_{(j-1,i),j-1} &= \bigcup_{m=1}^{M} \mathcal{D}_{(j-1,i),m} \setminus \mathcal{D}_{j-1,m}, \end{aligned} \tag{4.7}$$

and ι are the samples that remained incorrectly identified in the current partition $\mathcal{D}_j$, such that:

$$\begin{aligned} \iota_j &= \{t \mid t \in \mathcal{D}_{j,m}, t \notin C_m\}, \\ \iota_{j-1,i} &= \{t \mid t \in \mathcal{D}_{(j-1,i),m}, t \notin C_m\}, \end{aligned} \tag{4.8}$$

where $\mathcal{D}_j = \{D_{j,m}\}_{m=1\ldots M}$ is the partitioning of the samples in the space defined by $\mathcal{S}_d^j$ (and $\mathcal{D}_0 = \mathcal{T}$) and $\mathcal{D}_{j-1,i} = \{D_{(j-1,i),m}\}_{m=1\ldots M}$ is the partitioning in the space defined by $(\mathcal{S}_d^{j-1}, S_i)$ such that $S_i \notin \mathcal{S}_d^j$, i.e. all the shapelets that

---

[1] The Information Gain is calculated based on the partitioning returned by linear decision boundaries, i.e. hyper-planes.

Algorithm 5 GI-SCE: Global Independent Shapelet Clustering Explanation

```
    Input: dists, a distance matrix between (shapelets, time-series)
           $\mathcal{C}$, clustering predictions for the time-series
    Output: $\mathcal{S}_d$, Shapelets ordered by IG
            IG, Information gain related to $\mathcal{S}_d$
 1: $IG,\ \mathcal{S}_d \leftarrow [],\ []$
 2: $S,\ K, N \leftarrow GetShapeletsIndicesAndSize(dists)$
 3: for $j = 1, \dots, K$ do
 4:     $IG_j^* \leftarrow 0$
 5:     for $S_l \in S$ and $S_l \notin \mathcal{S}_d$ do
 6:         embvector ← transpose($dists$)[$S_l$]
 7:         $\mathcal{D}_j$ ← get_Partitions_By_Shapelets(embvector, $\mathcal{C}$)
 8:         $IG_j \leftarrow IG(\mathcal{D}_j, S_l)$
 9:         if $IG_j^* < IG_j$ then
10:             $S_j,\ D_{j-1},\ IG_j^* \leftarrow S_l,\ D_j,\ IG_j$
11:     end for
12:     $\mathcal{S}_d \leftarrow [\mathcal{S}_d,\ S_j]$
13:     $IG \leftarrow [IG,\ IG_j^*]$
14: end for
```

do not add the maximal IG when combined with $\mathcal{S}_d^{j-1}$. Therefore, the best shapelet at iteration j can be found such that:

$$IG(\delta_j, \zeta_\delta^j)\frac{\mu_j}{N} > IG(\delta_j', \zeta_{\delta'}^{j-1,i})\frac{\mu_{j-1}}{N}, \tag{4.9}$$

where $\zeta_\delta^j = \{\delta_{j,m}\}_{m=1\dots M}$ and $\zeta_{\delta'}^{j-1,i} = \{\delta_{j,m}'\}_{m=1\dots M}$ are the partitions of δ_j and δ_j' respectively, and μ is the number of samples correctly identified in $\mathcal{D}_j$ and incorrectly in $\mathcal{D}_{j-1}$. The scaling factor $\frac{\mu}{N}$ is used to weight to shapelets proportionally to the number of correctly partitioned samples. In this way it focuses on constructing a space using more diverse shapelets, aiding explainability.

With this in place, several combinations of SCE measures can now be proposed, e.g. Global Combined SCE, GC-SCE; Local Independent SCE, LI-SCE, etc.

Next, we will explain the algorithms for each use case (independent, combined, and successive) for the GE approach.

We will first explain the GI-SCE algorithm presented in Algorithm 5.

Line 5 iterates through all the shapelets that are not ranked already i.e. not in $\mathcal{S}_d$, to find the next best independent shapelet to rank.

Line 6 “embvector” represent the embedding of the time series data (dists) using only the shapelet S_l.

Algorithm 6 GC-SCE: Global Combined Shapelet Clustering Explanation

```
    Input: dists, a distance matrix between (shapelets, time-series)
           $\mathcal{C}$, clustering predictions for the time-series
    Output: $\mathcal{S}_d$, Shapelets ordered by IG
            IG, Information Gain related to $\mathcal{S}_d$
 1: $S_j, IG^*_j, \mathcal{D}_{j-1} \leftarrow$ GetBestIndependentShapelet($dists, \mathcal{C}$)
 2: $\mathcal{S}_d,\ IG, \leftarrow [S_j],\ [IG^*_j]$
 3: $S,\ K, N \leftarrow GetShapeletsIndicesAndSize(dists)$
 4: for $j = 2, \dots, K$ do
 5:     $IG^*_j \leftarrow 0$
 6:     for $S_l \in S$ and $S_l \notin \mathcal{S}_d$ do
 7:         embvector $\leftarrow$ transpose($dists$)[$\mathcal{S}_d, S_l$]
 8:         $\mathcal{D}_j \leftarrow$ get_Partitions_By_Shapelets(embvector, $\mathcal{C}$)
 9:         $IG_j \leftarrow IG(\mathcal{D}_j, [\mathcal{S}_d, S_l])$
10:         if $IG^*_j < IG_j$ then
11:             $S_j,\ D_{j-1},\ IG^*_j \leftarrow S_l,\ D_j,\ IG_j$
12:     end for
13:     $\mathcal{S}_d \leftarrow [\mathcal{S}_d,\ S_j]$
14:     $IG \leftarrow [IG,\ IG^*_j]$
15: end for
```

Line 7 takes this vector and the cluster labels to get the partitions of the data based on ζ, where, in this case, it is a thresholding criterion.

Line 9 through 11 calculate first the Information Gain of the current shapelets and if it is better than the best one so far, the IG, shapelet and partitions are stored to compare with.

Subsequently, we will explain the algorithm of GC-SCE provided in Algorithm 6 and highlight the differences to GI-SCE.

Line 1 the shapelet that best splits the dataset into $M = |\mathcal{C}|$ partitions is found based on the independent approach. Its index S_l, corresponding IG^* and partitions $\mathcal{D}_{j-1}$ are stored.

Line 7 unlike the GI-SCE approach that returns "embvector" based on one shapelet, now we will return the embeddings of the time series using the best-found d-shapelets so far in $\mathcal{S}_d$ and the current shapelet S_k under study, hence building the space as specified in Equation 4.5.

Line 8 get_partitions_by_shapelets returns the best possible partitioning by applying ζ (LDA) to the data-samples obtained via embvector.

Line 8 the IG is computed as defined in the combined case.

Algorithm 7 GS-SCE: Global Successive Shapelet Clustering Explanation

```
    Input: dists, a distance matrix between (shapelets, time-series)
           $\mathcal{C}$, clustering predictions for the time-series
    Output: $\mathcal{S}_d$, Shapelets ordered by IG
            IG, Information Gain related to $\mathcal{S}_d$
 1: $S_j, IG_j^*, \mathcal{D}_{j-1} \leftarrow$ GetBestIndependentShapelet($dists$, $\mathcal{C}$)
 2: $\mathcal{S}_d,\ IG \leftarrow [S_j],_{[} IG_j^*]$
 3: $S,\ K, N \leftarrow GetShapeletsIndicesAndSize(dists)$
 4: for $j = 2, \dots, K$ do
 5:     $IG_j^* \leftarrow 0$
 6:     for $S_l \in S$ and $S_l \notin \mathcal{S}_d$ do
 7:         embvector $\leftarrow$ transpose($dists$)[$\mathcal{S}_d$, $S_l$]
 8:         $\mathcal{D}_j \leftarrow$ get_Partitions_By_Shapelets(embvector, $\mathcal{C}$)
 9:         $\delta_j \ \leftarrow \ \Delta_{j,j-1} \cup \iota_j$
10:         $\mu_j \leftarrow NumberOfCorrectPts(\delta_j)$
11:         $IG_j \leftarrow IG(\delta_j, [\mathcal{S}_d, S_l]) \cdot \frac{\mu_j}{N}$
12:         if $IG_j^* < IG_j$ then
13:             $S_jD_{j-1}, IG_j^* \leftarrow S_lD_j, IG_j$
14:     end for
15:     $IG, \mathcal{S}_d \leftarrow [IG, IG_j^*], [\mathcal{S}_d, S_j]$
16: end for
```

Finally, the GS-SCE algorithm (Algorithm 7) is explained and the differences to the other approaches are highlighted.

Lines 1 through 8 are similar to the GC-SCE approach.

Line 9 Stores misidentified samples and those whose partition differs.

Line 10 The number of correct samples in δ_j is returned.

Line 11 The Information Gain defined in the successive case is calculated.

In order to achieve the Cluster-Wise explanation (CE), the only change to the presented algorithms is to use binary labels instead of the cluster labels: the cluster labels are C_i (the cluster being explained) and $\overline{C_i}$ (all other clusters). To rank the shapelets that best represent a specific cluster, and when Global-Wise explanation (GE) is concerned the original clustering labels, $\mathcal{C}$, are used as the labels. It is worth noting that these strategies may be used with any shapelet-based method, independent of the learning process.

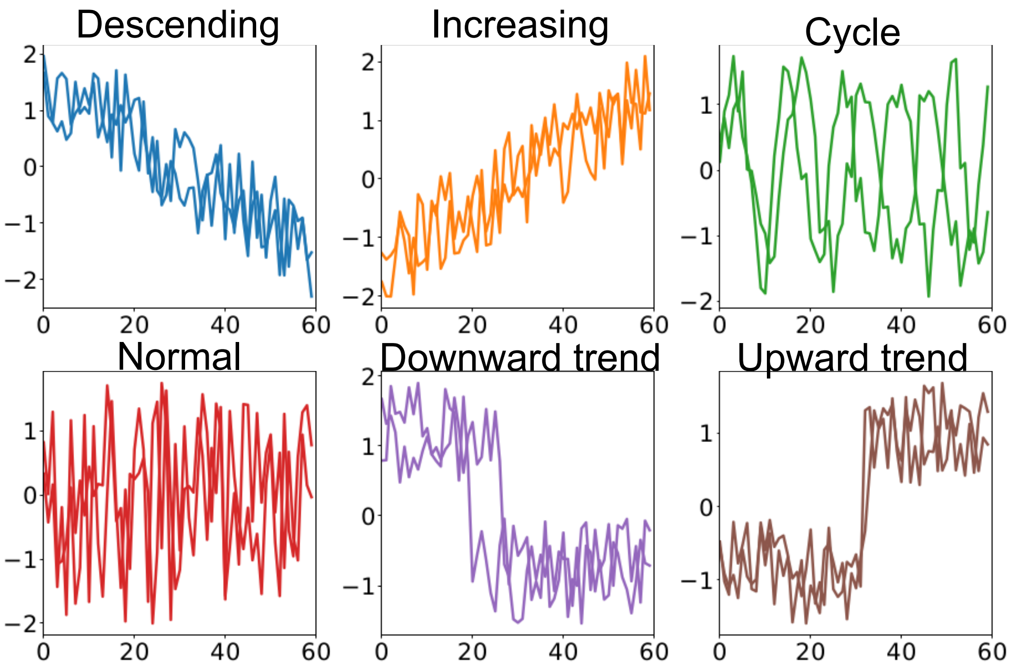


Figure 4.1: Time-series samples from each category of the Synthetic Control dataset are presented. Colours correspond to the clustering in Figures 4.3, 4.6 and 4.7.

4.2 Cluster Explanation: A Case Study

A case study of SCE using global wise (GI-SCE, GC-SCE, and GS-SCE) cluster explanation is presented herein. We will first describe the dataset used for this study, and then present the case study for the successive approach. It is followed by a discussion in comparison to the other two approaches independent and combined. We will also examine reducing the number of shapelets by choosing the best-ranked ones under the GS-SCE approach.

The Synthetic Control dataset, from the UCR archiveis used. This dataset was chosen since it contains multiple categories which will allow us to show the importance of the global setting, i.e. having more than two labels. The dataset contains 600 samples, that represents synthetically generated control charts each of length 60. The dataset is divided into six categories: 1-Normal, 2-Cyclic, 3-Increasing trend, 4-Decreasing trend, 5-Upward shift, and 6-Downward shift. Figure 4.1 shows two instances from each category. The Transductive approach is used, and the aim is to provide insight and explanation of the clustering result. Recall that the GS-SCE algorithm, described in Algorithm 7, takes clustering labels and transformed time series as input and returns a shapelet ranking based on the information added by each.

Figure 4.2 presents the Normalised Cumulative Information Gain, NCIG, scaled between 0 and 1. After a certain number of shapelets NCIG plateaus, indicating that any further addition of the remaining shapelets does not improve the space

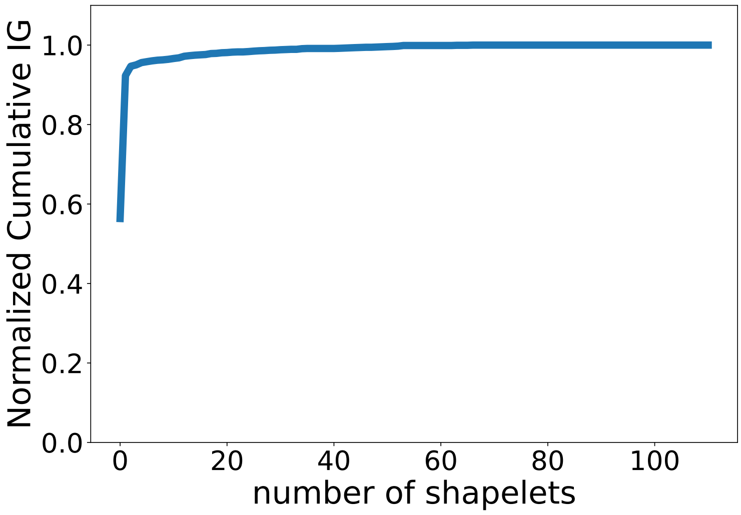


Figure 4.2: The normalised cumulative Information Gain calculated using GS-SCE.

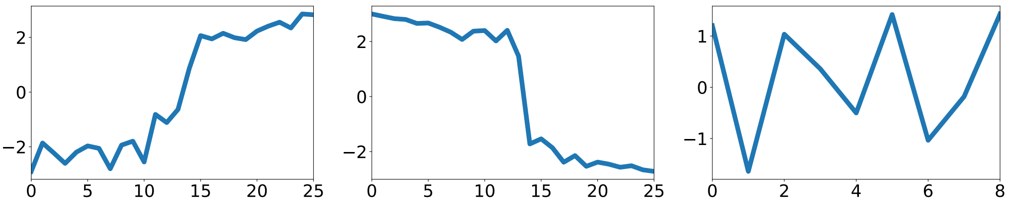
relative to the clustering partitioning. In this case, dimensionality reduction up to the number of shapelets indicated by the elbow can be considered. In some cases, Information Gain may not plateau, indicating that all shapelets are necessary.

Figure 4.3 presents the best 3 shapelets, the order-lines[1] for each shapelet, and the scatter plots of the data samples in the space when used with 2 and 3 shapelets. These 3 shapelets represent an additive Information Gain of 0.67, 0.42, and 0.027 respectively, together capturing 1.117 of the total cumulative IG which is equal to 97% of NCIG. This means that these shapelets hold 97% of the information in the overall shapelet space and are therefore considered to be the best shapelets for reflecting the overall clustering.

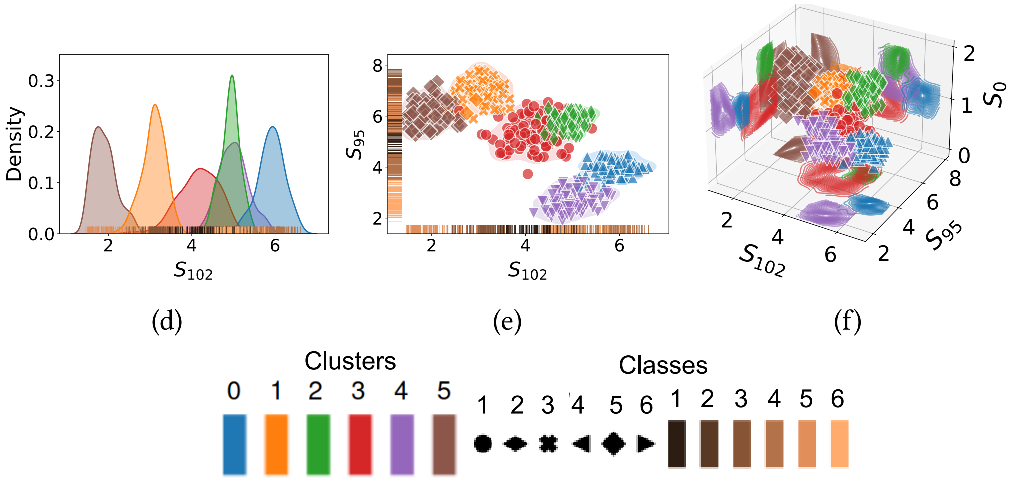
Figure 4.3d presents the order-line of the most informative shapelet (S_{102}, Figure 4.3a), which exhibits an increasing trend. The figure demonstrates that despite having well separated-clusters, not all categories can be distinguished. It is clear that cluster zero (blue) which exhibits a decreasing trend and cluster five (brown) which exhibits an increasing trend are well separated and the other categories slightly overlap each other except for clusters two (green) and red (violet), which overlap each other by a considerable amount since they exhibit both increasing and decreasing trends, see Figure 4.1. Shapelet S_{95}, Figure 4.3b, represents a decreasing trend in the data and therefore has a low distance to clusters zero and four and a high distance to clusters one and five (having increasing trends), as seen in the y-axis of Figure 4.3e. Clusters two and three (green and red, Figure 4.1), however, exhibit both upward and downward trends, hence their overlap in feature space (Figure 4.3e). Observing this scatter plot further, we can see that the complementing information of the shapelets separates the clusters, except for clusters two and three that exhibit both trends. Adding shapelet S_0, which models both upward and downward trends, see Figure 4.3c, increases the separation between the second and

---

[1] An order-line is simply a representation of the distance between the time-series and the shapelet ordered from lowest to highest distance.



(a) Shapelet S_{102} (b) Shapelet S_{95} (c) Shapelet S_0



(d) (e) (f)

Figure 4.3: Successive global explanation (GS-SCE). The top three shapelets are presented with the corresponding scatter plots of the data projected into each space, defined by adding each shapelet in succession (starting from S_{54}).

the third cluster (visually S_0 has more similarity to three than two), see Figure 4.3f.

In light of this, one can ask whether SCE can also help to achieve equivalent clustering with fewer, informative shapelets. As such, K-means clustering is applied to the embeddings obtained using each subset returned by GS-SCE and compared to the results of the original clustering. As expected, NMI follows the same trend as IG, see Figure 4.4, and it can be observed that using just 20% of the shapelets (around 10 shapelets) results in approximately the same performance as the original clustering. As such the NCIG can be used to threshold the number of shapelets in a similar way to the eigenvalues in principal component analysis.

For example, taking a threshold of 80% NCIG, would result in two shapelets (S_{102} and S_{95}), accumulating 0.94 of the NCIG, while retaining an NMI of 0.8 relative to the original clustering. Adding a third shapelet (S_0) increases the NCIG to 0.96 and results in an NMI of 0.9. It should be noted, however, that the reduced dimensional space will not model DTW distance, as will be explored in Section 4.2.1.

In order to further explore the possibility of using SCE as a dimensionality reduction technique for selecting a reduced set of shapelets, we conducted a comparison with Principal Component Analysis (PCA) [180]. The comparison evalu-

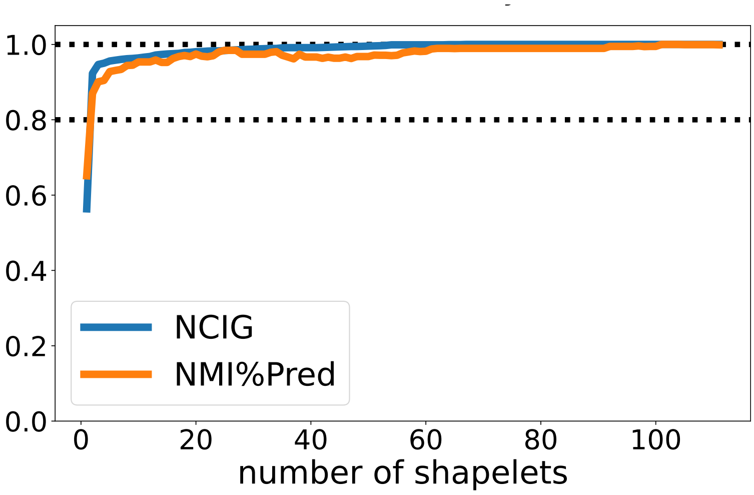


Figure 4.4: NMI of K-means clustering and normalized cumulative information gain NCIG relative to the number of shapelets used to form the representation (ordered by GS-SCE). The labels used for NMI are the clustering labels output by CDPS. The plateau in NCIG indicates that these shapelets add no further information.

ates the clustering scores obtained from clustering based on PCA components and compares them to the clustering results achieved using SCE shapelets. Where we assess clustering performance while increasing the number of shapelets or principal components. Figure 4.5 presents the NMI scores obtained from clustering using an increasing number of shapelets based on the ranking provided by GS-SCE, as well as PCA components ranked by their eigenvalues. The figure indicates that PCA performs slightly better than SCE in terms of clustering accuracy with lower dimensionalities because it is able to extract linear combinations of the shapelet embeddings, however, it loses the interpretability offered by ranking shapelets themselves, as with SCE. We should recall that the embedding created by CDPS is the minimum matching distance of each shapelet over the whole time series, therefore we cannot simply create a linear combination of the shapelets to interpret the PCA results because the minimum position of the shapelets represented by the linear combination produced by PCA do not necessarily occur at the same point in the time series.

Comparison to GI-SCE and GC-SCE

In order to better understand the idea and the reasoning behind the proposed GS-SCE explanation approach and to highlight the difference, this section provides visual examples of GI-SCE and GC-SCE. The first three best shapelets are used to infer the difference between the approaches presented in Subsection 4.1.

Figure 4.6 presents the space with the different shapelets ranked with GI-SCE, which ranks shapelets independently of each other. Looking at the scatter plots (Figures 4.6e and 4.6f) and the shapelets (Figures 4.6a, 4.6b and 4.6c) it is clear that in the **Independent** case the features are highly correlated. This is expected as there

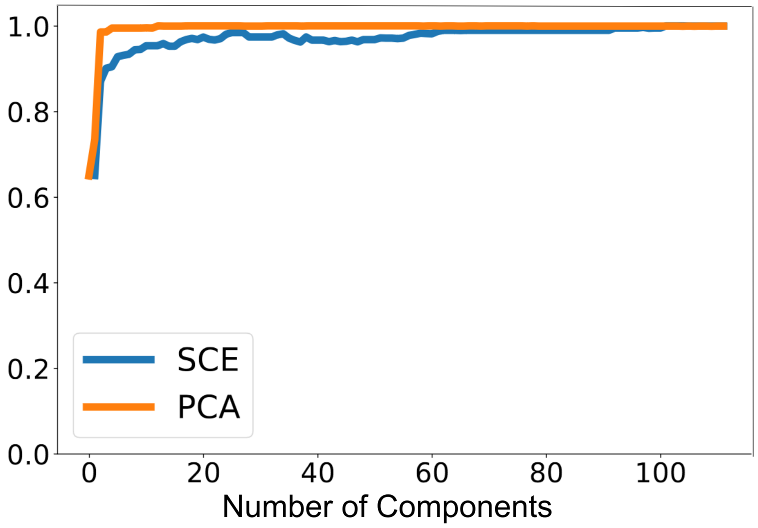
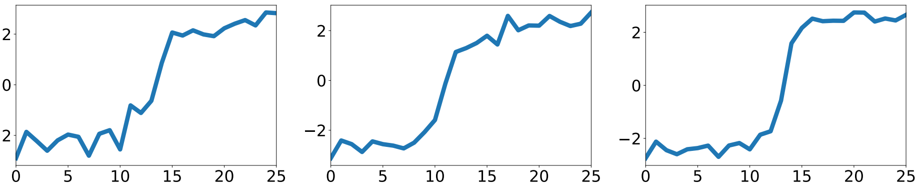


Figure 4.5: Comparison of Clustering Performance on Increasing Number of Shapelets Ranked by GS-SCE (termed SCE) and Increasing Number of PCA Components (termed PCA). Note that the cluster labels obtained from the entire shapelet space are used as class labels for computing the Normalized Mutual Information (NMI) scores.

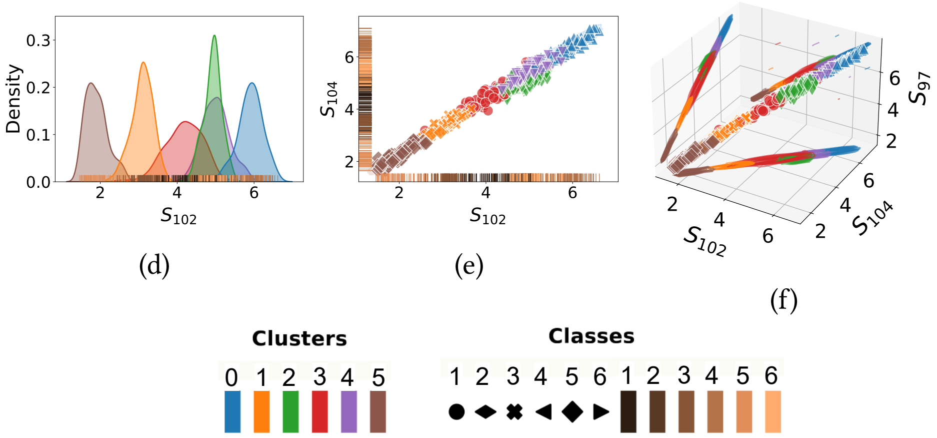
is no requirement for the ordering of the shapelets to reflect complementary information and hence highly correlated shapelets will be ranked first since they have equivalent IGs. Figure 4.7 presents the space with the different shapelets ranked with GC-SCE. Contrarily to the independent approach, the **Combined** approach tries to find diverse shapelets. In this use case, the shapelets for the combined approach are diverse (see Figure 4.7a, 4.7b and 4.7c) but this is not guaranteed as there is no explicit condition specifying their diversity and how much information they add. The scatterplots obtained using the three shapelets in GS-SCE (see Figure 4.3f) and GC-SCE (Figure 4.7f) seem similar, but the third shapelet found by GS-SCE (according to the definition) results in most points being moved to the correct cluster, and hence a better representation of the clustering result. Although in this example the difference is not great, in other datasets this can cause a very different shapelet to be found.

The Normalised Cumulative Information Gain (NCIG) plots of the GI-SCE and GC-SCE reflect these findings, see Figure 4.8, in which the Successive approach quickly finds the shapelets that maximise Information Gain. These figures also include the NMI score of K-means clustering performed in the subspace relative to the number of shapelets (plotted in orange). It becomes clear that the Information Gain found by GI-SCE (Figure 4.8a) and GC-SCE (Figure 4.8b) does not represent NMI, and would therefore result in an overestimation in the required number of shapelets. On the other hand, GS-SCE (Figure 4.4) results in a compact representation (fewer shapelets) with a comparable NMI score.

As noted in Section 4.1, the Cluster-Wise explanations (CI-SCE, CC-SCE, and CS-SCE, which explain the clustering of a specific cluster rather than the global



(a) Shapelet S_{102} (b) Shapelet S_{104} (c) Shapelet S_{97}



(d) (e) (f)

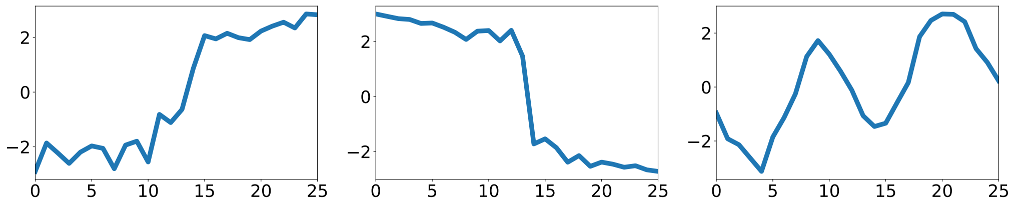
Figure 4.6: Independent global explanation (GI-SCE). The top three shapelets are presented with the corresponding scatter plots of the data projected into each space, defined by adding each shapelet in succession.

clustering) can be obtained using the global algorithms but by providing $\mathcal{C}$ as binary labels. These labels should be true for samples contained in the cluster to be explained C_j and false for all others $\overline{C}_j$. The same discussion and reasoning can be conducted to reflect the recently presented findings.

4.2.1 Shapelet Selection

Section 4.2 illustrated cluster explanation using GW approaches and the possibility of reducing the number of shapelets while achieving results equivalent to those obtained in the original space. Aside from the intended use-case of offering insight into the clustering results, this demonstrates further possibilities for the information obtained using such an approach. We further illustrate this point with 10 datasets using transductive clustering, (explained in Chapter 3, Section 3.3.1) with 25% constraint fraction.

Table 4.1 presents the clustering score achieved with the subset of shapelets that result in a cumulative Information Gain of 80%. For the majority of the datasets, this reduced dimensionality results in more than 0.7 NMI when compared to the clustering achieved in the original space, even though in most cases only 1 or 2 (~2%) shapelets are retained. Having said this, the shapelets (and therefore the information) that are removed are needed to respect the original DTW distances. This



(a) Shapelet S_{102} (b) Shapelet S_{95} (c) Shapelet S_{109}

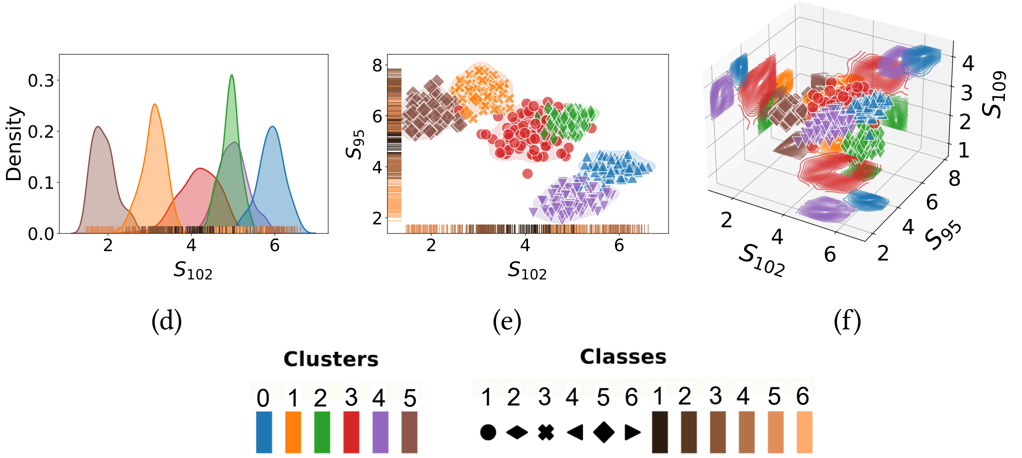
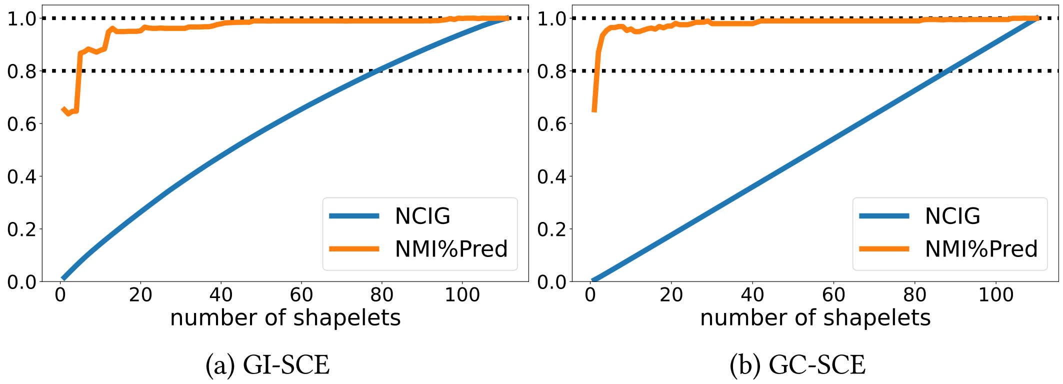


Figure 4.7: Combined global explanation (GC-SCE). The top three shapelets are presented with the corresponding scatter plots of the data projected into each space, defined by adding each shapelet in succession.

Table 4.1
Clustering scores for different datasets using all shapelets (100% NCIG) and a subset of the shapelets (thresholded to 80% NCIG).

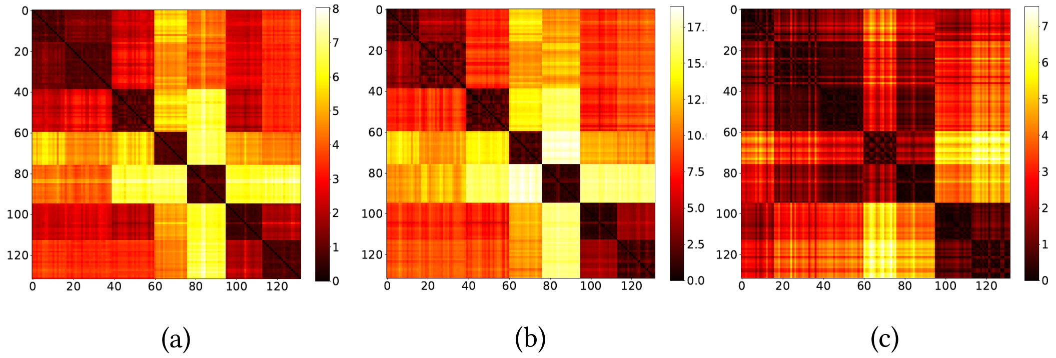
Dataset	#Shapelets (Total)	NCIG	NMI
BME	1 (68)	96%	0.77±0.000
CBF	2 (68)	100%	0.86±0.000
ECG200	1 (64)	96%	0.70±0.000
GunPoint	1 (70)	99%	0.88±0.000
GunPointAgeSpan	2 (70)	98%	0.72±0.000
Herring	1 (89)	86%	0.66±0.000
MoteStrain	1 (62)	88%	0.64±0.001
OSULeaf	12 (86)	80%	0.67±0.006
Plane	2 (70)	99%	0.82±0.003
Symbols	2 (85)	96%	0.82±0.002

is because the representational space is learnt to approximate the DTW distances using all the shapelets and not a subset of the space or the shapelets themselves.



(a) GI-SCE (b) GC-SCE

Figure 4.8: NMI of K-means clustering relative to the number of shapelets used to form the representation ordered by GI-SCE (Figure 4.8a) and by GC-SCE (Figure 4.8b). The labels used for NMI are the clustering labels output by CDPS.



(a) (b) (c)

Figure 4.9: Comparison of distance map between the actual DTW (a) and approximate DTW using all shapelets (b) and a subset of shapelets (c) for the Plane dataset. The colour bar indicates the distance between the different samples, the brighter the larger the distance.

Figure 4.9 shows example distance maps for the Plance dataset. Figure 4.9a shows the distance map of the actual DTW distance, Figure 4.9b shows with respect to the CDPS space, and Figure 4.9c the reduced dimensionality CDPS space (2 shapelets). It can be seen that the original DTW and CDPS result in similar distance maps while the reduced CDPS dimensional space has a very different distance map since it looses some of the DTW approximation (but retains the ability to differentiate between clusters). The loss of DTW approximation is due to the fact that the distance between the samples in the overall shapelet space approximates the DTW and choosing any fewer shapelets will break this approximation. As such, one can choose to use all the shapelets, retaining the DTW approximation in the space while respecting constraints, or use fewer shapelets for better performance. Note that using fewer shapelets does not necessarily means lower clustering performance, the experts may deem the clustering achieved using fewer shapelets is better than that using all the shapelets, even though DTW approximation is lost.

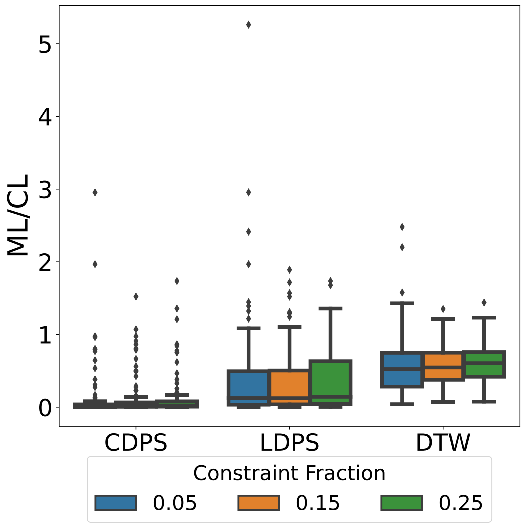


Figure 4.10: Visualization of ML/CL: the ratio of the average distance between ML constraints to the average distance between CL constraints across different spaces LDPS, CDPS, and RAW time series space using DTW. The ratio was calculated for ten different univariate datasets. The constraint sets, for each constraint fraction, that were considered under the transductive study are used.

4.3 Effect of Constraints on the Representation

In this section, our objective is to investigate the impact of constraints on the representational space and determine the validity of using contrastive loss for bringing similar samples closer together and pushing dissimilar samples further apart. In this way, the expert constraints are modelled, as described in Section 3.1 and 3.2. We use the same ten randomly selected datasets as presented in the previous subsection to calculate the statistics presented in the remainder of this chapter, using the transductive setting (Section 3.3.2). A random subset of three datasets is selected for visualization to save space.

To begin with, we will examine the ratio between the average distance of the must-link samples and the average distance of the cannot-link samples, denoted as ML/CL. We then calculate the coherence in all ten repetitions of the 25% constraint sets. Measuring the ML/CL ratio provides a general understanding of the distances between the samples in the space, as explained in Chapter 2, Section 2.2.5. We expect that CDPS exhibits the lowest ML/CL ratio, followed by LDPS and DTW.

Figure 4.10 presents the boxplots of the ML/CL ratio calculated over LDPS, CDPS and time series space using DTW. The study is done for the three different constraint fractions (5%, 15%, and 25%) using the ten constraint sets that were used in the transductive setting (Section 3.3.2). It is evident that DTW yields the largest ratios (overall constraint fractions), CDPS has the smallest ratios, and LDPS is approximately equal to DTW. This strengthens the assumption that CDPS exploits constraints to bring similar samples closer together and dissimilar ones further apart, leading to a decrease in ML/CL.

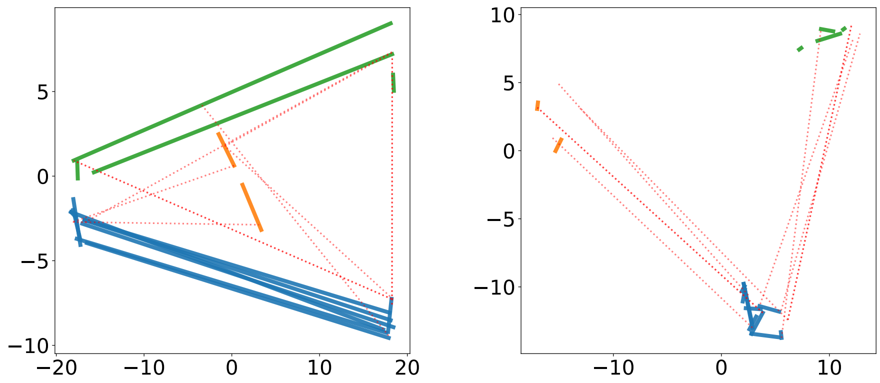
Now we compare the LDPS (CDPS without constraints) with the CDPS space. This will allow us to observe the effects of the constraints on the space by visually interpreting the ML/CL ratio. In order to visualise the spaces in two dimensions, we use Pairwise Controlled Manifold Approximation (PaCMAP) dimensionality reduction [181]. PaCMAP is a recent approach developed to take into account both the global and the local structure of the space to be reduced. It optimizes the low-dimensional embeddings using three kinds of pairs of points; neighbouring pairs, mid-near pairs, and distant pairs.

Figure 4.11 shows the scatterplots depicting a subset of the constrained samples. Must-link connected samples are linked by solid lines, coloured according to the ground truth labels, while cannot-link samples are connected by red dotted lines. Analyzing these figures provides a general overview of how CDPS makes use of the constraints provided. Figures 4.11a and 4.11c demonstrates that LDPS separates clusters in the BME and CBF datasets, but CDPS further consolidates similar points, as illustrated in Figures 4.11b and 4.11d. Examining the Symbols dataset, LDPS exhibits confusion between four clusters (red and purple) and (blue and orange) due to the relatively close distance between these pairs of classes. In comparison, CDPS manages to push the clusters further apart, thanks to the constraints, but still encounters challenges in capturing some points, such as the red must-link point connected to the purple region. This discrepancy may stem from the requirement of approximating DTW distance.

Next, we will compare the clustering results obtained using the DTW distance. However, since DTW is not a metric we cannot directly visualise it. Instead, we must use Multidimensional Scaling (MDS) [182] embeddings to map DTW's distance matrix to a two-dimensional space. The same is performed for LDPS and CDPS in order to perform a fair comparison and therefore the clustering results obtained from each of them can be compared in a two-dimensional space. This should further solidify the previous discussion.

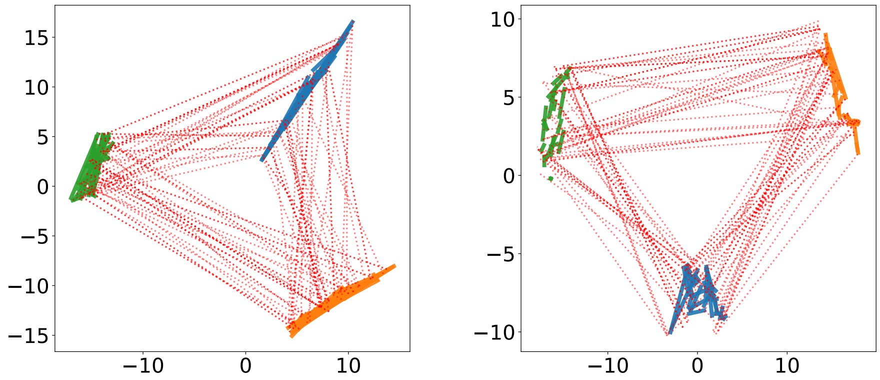
Figure 4.12 shows these visualisations for the three chosen datasets. These show that LDPS and DTW have generally the same trend, in which the distance between must-link pairs is relatively large and the grouping of points is similar across both methods, with overlapping must-link and cannot-link constraints. On the other hand, CDPS shows more compact clustering, similar to what was observed with the PaCMAP plots, where similar points are closer and dissimilar points are pushed further apart. The rest of the analysis follows the same line of reasoning as seen with PaCMAP.

An interesting observation can be made by investigating the GunPoint and GunPointAgeSpan dataset found in the UCR archive. First, we will explain these datasets. Both datasets are records of two different actions performed by a female and a male actor. The first action, termed Gun, is to take a gun from a waist holster and aim it at an eye-level target. The second action termed Point (or NoGun), is to point at the target without a gun. The GunPoint dataset was recorded in



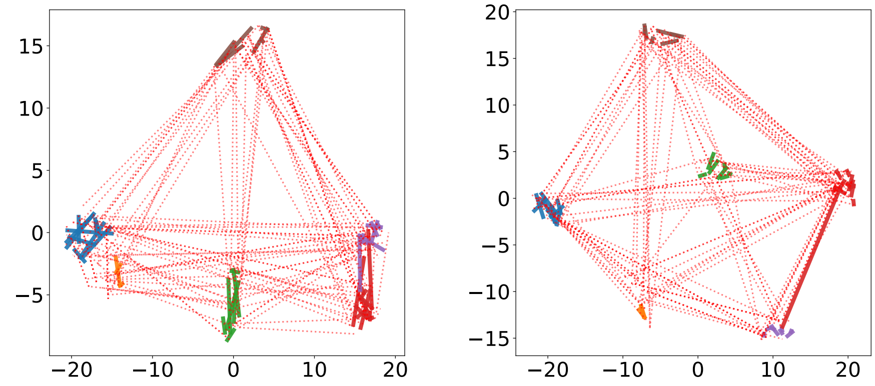
(a) BME – LDPS.

(b) BME – CDPS.



(c) CBF – LDPS.

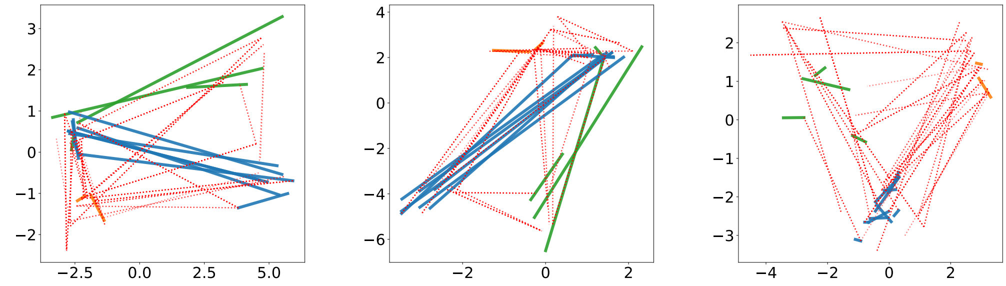
(d) CBF – CDPS.



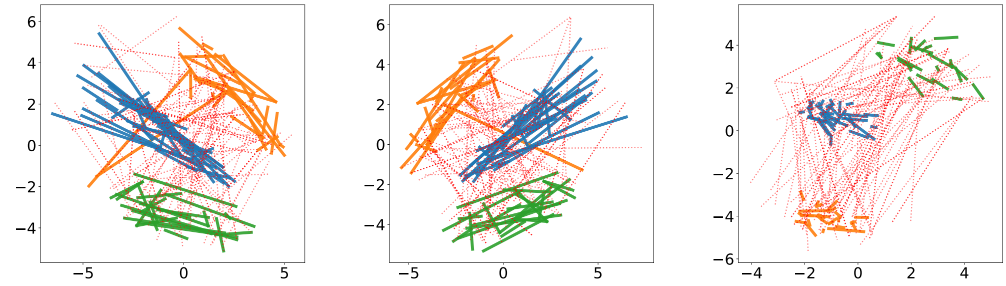
(e) Symbols – LDPS.

(f) Symbols – CDPS.

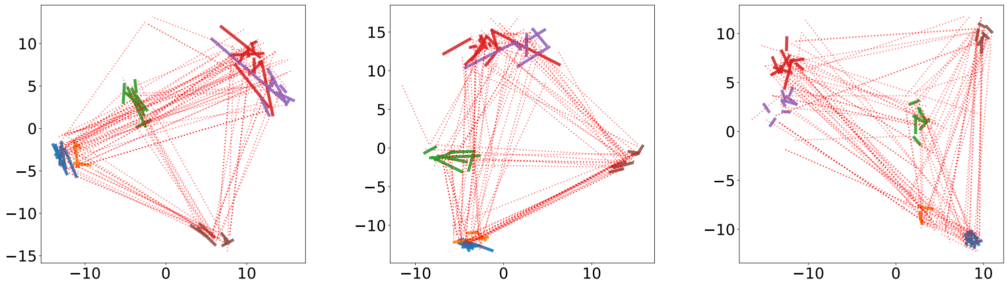
Figure 4.11: Visualization of the constraints on the LDPS and CDPS representational space using PaCMAP dimensionality reduction; for three different datasets. Must-link constraints are represented in solid lines coloured according to the ground truth labels and cannot-link constraints are the red dashed lines.



(a) BME – LDPS. (b) BME – DTW. (c) BME – CDPS.

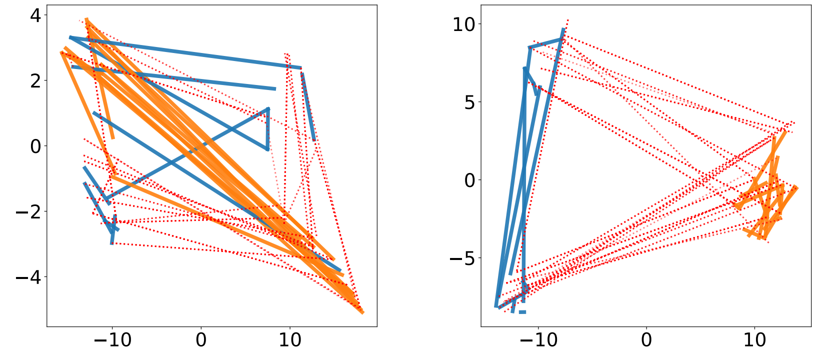


(d) CBF – LDPS. (e) CBF – DTW. (f) CBF – CDPS.

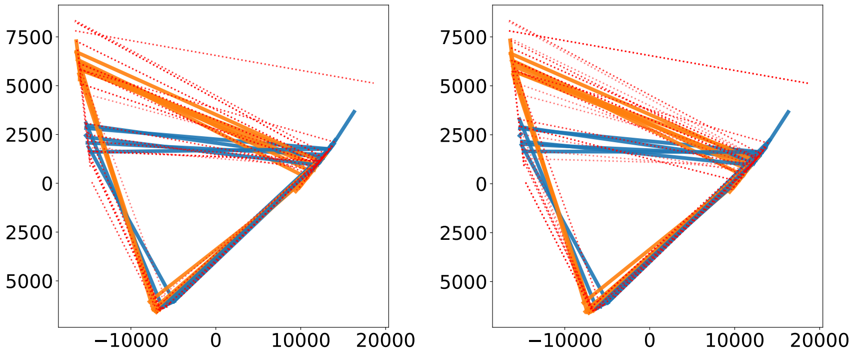


(g) Symbols – LDPS. (h) Symbols – DTW. (i) Symbols – CDPS.

Figure 4.12: Visualization of the constraints on the MDS embedded space calculated using DTW, LDPS and CDPS distance maps; for three different datasets. Must-link constraints are represented in solid lines coloured according to the ground truth labels and cannot-link constraints are the red dashed lines.



(a) GunPoint — LDPS (b) GunPoint — CDPS



(c) GunPointAgeSpan — LDPS (d) GunPointAgeSpan — CDPS

Figure 4.13: Illustration of the effect of the constraints for different tasks on the same dataset. The visualisation is on LDPS and CDPS using PaCMAP dimensionality reduction. The dataset used contains samples of pointing a gun. (a) and (b) shows the task of grouping the samples based on age while (c) and (d) are based on the movement itself.

2008 and the classes are Gun (first action) and Point (second action) where the model needs to learn a distinction between pointing with and without a gun. While the GunPointAgeSpan dataset contains data from actors recorded in 2003 and repeated recording by the same actors after 15 years, in 2018. The classes in the GunPointAgeSpan dataset are also Gun (first action) and Point (second action), but now each class contains more variability.

Figure 4.13 presents PaCMAP embedding visualisations of the constraints and their impact on the GunPoint and GunPointAgeSpan datasets. CDPS is compared to LDPS to investigate its effectiveness in capturing the constraints. For the GunPoint dataset, CDPS successfully uses the constraints, resulting in two distinct groupings of data samples. In contrast, LDPS shows overlapping clusters as it relies only the DTW distance. In the case of the GunPointAgeSpan dataset, both LDPS and CDPS have the same results, with no clear separation between the clusters. This finding is

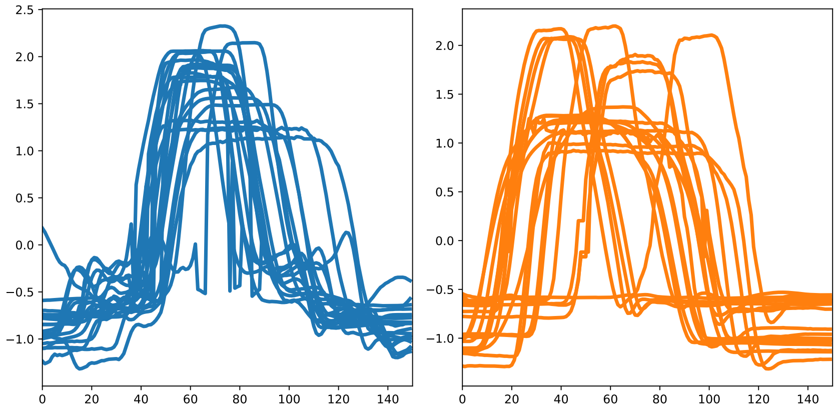


Figure 4.14: Illustration of the GunPointAgeSpan classes. It shows 20 samples chosen at random for each class. Each sample is recorded over five seconds with 30 frames per second, the 150 frames are represented in the x-axis. The y-axis shows the position of the abscissa component of the hand's centroid in each frame.

intriguing because it suggests that CDPS fails to effectively model the constraints for this particular dataset. Figure 4.14 displays samples from each class (only 20 samples per class shown), it can be observed that the actions performed by the actors after a 15-year gap exhibit notable differences. This means that the constraints give vastly conflicting information compared to the DTW distance and CDPS is not able to resolve it. Thus, this indicates that CDPS's failure on the GunPointAgeSpan dataset can be attributed to the approximation of DTW distance, which struggles to distinguish the differences between the clusters. Consequently, this limitation affects both LDPS and diminishes the effectiveness of constraints in CDPS. Note that this study was done with $\alpha = \gamma = 2.5$ and adjusting this value and increasing the number of constraints might give better results but these will still be limited by the need to approximate DTW.

Measuring Constraints Coherence

Now that we have a metric space in which time series can be transformed, we can now also calculate the coherence of the constraints directly in the resulting CDPS space. Recall that DTW distance is not a metric measure and hence the coherence of constraints cannot be calculated. Instead, we use an embedding approach (Multi-Dimensional Scaling, MDS) to find a representation that matches the distance matrix as closely as possible. This gives a reduced metric space in which coherence can be indirectly calculated. And we can then compare this to the coherence that can now be calculated in the CDPS representation. Furthermore, we also calculate an MDS embedding for LDPS and CDPS in order to have a fair comparison to DTW.

Figure 4.15 presents the distribution of constraint coherence values calculated upon the ten datasets used in the previous discussion. In order to compare DTW to LDPS and CDPS, MDS embeddings of DTW were calculated with the number

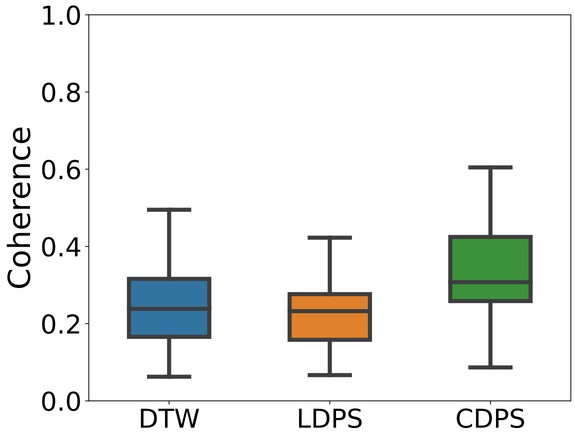
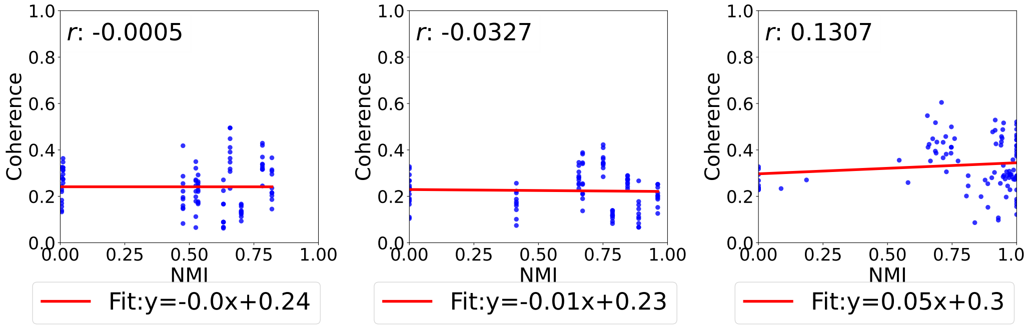


Figure 4.15: Distribution of coherence of the constraints calculated for different datasets based on DTW, LDPS and CDPS. For DTW the coherence is shown in the MDS embedding space with dimensions equal to a number of shapelets $|\mathcal{S}|$ is presented as well.



(a) DTW MDS d=$|\mathcal{S}|$ (b) LDPS Original (c) CDPS Original

Figure 4.16: Scatter plot pf coherence versus NMI calculated for different datasets using the LDPS, CDPS and DTW with MDS embeddings with dimension equal to $|\mathcal{S}|$. The figure also shows the best fit of the data with the correlation factor.

of dimensions equal to the number of shapelets. Analyzing the figure, we observe that CDPS demonstrates the highest coherence values, indicating that the constraints are more consistent with each other in CDPS space. This result aligns with our expectations since CDPS is designed to respect and incorporate the provided constraints. On the other hand, DTW and LDPS exhibit similar coherence values, which is expected since LDPS approximates DTW. The coherence measure provides valuable insights into the degree of consistency among the constraints, independent of any partitioning or specific clustering algorithm. The higher coherence in CDPS confirms its effectiveness in modelling constraints.

Now that we have established that CDPS is effective in modelling the constraints, we will investigate the correlation between coherence and clustering performance. A correlation study between them will be carried out for each DTW, LDPS, and CDPS, following the same concept explained earlier. Recall that coherence indicates whether the constrained samples in the space are conflicting or not. Higher coherence means fewer conflicts. In general, if there are no conflicts, the constraints will not necessarily improve the clustering as they may not be informative. However, if the conflict between constraints is too great, this may impede clustering as they provide conflicting information. In this comparison (between coherence and NMI) we will be able to identify if higher coherence (fewer conflicts) in the CDPS space is positively correlated with the NMI score, and if coherence is correlated to NMI when DTW and LDPS are used.

Figure 4.16 shows the scatter plot for coherence versus NMI clustering scores. The figure also displays the fitted line and reports the correlation factor. It is clear that CDPS exhibits a positive correlation between coherence and the clustering score, unlike both DTW and LDPS, where there is almost no correlation (the factor is zero) or a slightly negative correlation. This strengthens the fact that CDPS leads to better results and is capable of respecting and modelling constraints. It must be emphasised here that these results are not definitive and should be interpreted with caution. First, the DTW coherence is calculated on an MDS embedding of the DTW distance matrix, which may or may not be valid (although the similar LDPS results give weight to its validity). Secondly, assessing the positive effect of coherence on the clustering results is also complicated. Nevertheless, since it demonstrates the only way to calculate a measure such as coherence when using DTW distance, these results do show that measuring it with the proposed CDPS embedding seems to be more accurate.

Effect of Constraints on Reduced Representation

In this section, we will analyze the reduced space obtained using the GS-SCE algorithm. We aim to investigate whether the observations and discussions presented in Section 4.2 regarding the clustering performance and shapelet rankings are also reflected in the coherence of constraints and the ratio between the average distance of the must-link (ML) samples and cannot-link (CL) ones, represented as ML/CL. We will follow the same strategy used in Section 4.3, where the effect of the constraints on the representation when using all shapelets was investigated. Therefore, to maintain consistency, we will employ the same datasets and experimental settings used in Chapter 3, Section 4.3 to examine the reduced representation achieved using GS-SCE with a threshold of 80% NCIG. As such, only the number of shapelets that result in 80% NCIG is used for clustering using shapelets. For PCA, the number of principal components used was equal to the number of shapelets yielding 80% NCIG.

Figure 4.17 shows the ML/CL ratio in the reduced space (using PCA and SCE with threshold 80% NCIG to choose the number of components and shapelets) in comparison to the overall representational space (see Section 4.3, Figure 4.15 for comparison to LDPS and DTW). The illustration shows that the ratio of the reduced spaces is relatively small compared to the original space. This indicates that the constraints are still modelled well and the shapelets did not lose their separability

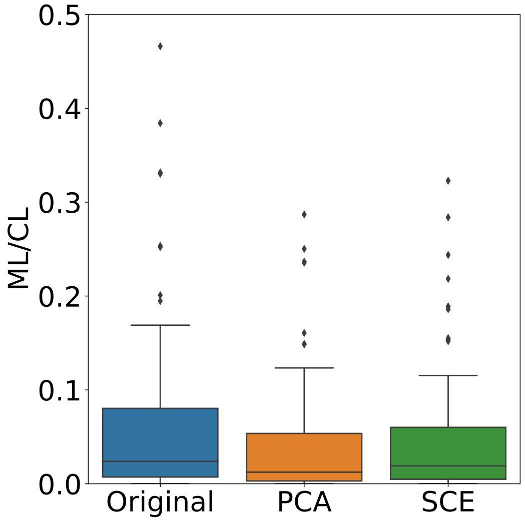


Figure 4.17: Average distance between the must-link and cannot-link constraints ML/CL for the reduced CDPS representation space in comparison to using all the space. GS-SCE was used to reduce the original representational space and compared to PCA dimensionality reduction.

power, in addition, lower values can be attributed to having smaller distances as the representational space is smaller and hence less sparse. Moreover, it seems that the SCE reduced space offers more separation than that obtained using PCA as the ratio in SCE is smaller. Figure 4.18 shows the coherence of constraints over the reduced space (using PCA and SCE) in comparison to the overall representational space (which was discussed and presented in Section 4.3, Figure4.15). This figure shows that the coherence in the reduced space is slightly better than the original space and comparable to the PCA reduced space. This strengthens the finding represented in Figure 4.17, that shapelets model the constraints and reflect expert intuition.

Figure 4.19 offers a visualization of the Synthetic Control dataset (used in Section 4.2) CDPS reduced representational space obtained using GS-SCE and PCA reduction. This illustration provides a visual comparison between the two reduced spaces and how the constraints are represented in them, offering a case study of the finding of Figures 4.17 and 4.18. Comparing PCA (Figure 4.19a) to CDPS GS-SCE space (Figure 4.19b) it is clear that there is less overlap between the points in the CDPS reduced space. This is reflected in the constraints presented in Figures 4.19c and 4.19d where there is less overlap between must-link belonging to different clusters in the CDPS reduced space than with PCA.

Overall, this study provides an in-depth study on the reduced space using GS-SCE and provides clarification on how the constraints affect the reduced space. It was shown that the constraints remain respected and hence reducing the number of shapelets using GS-SCE will provide comparable performance and separation when using the overall space.

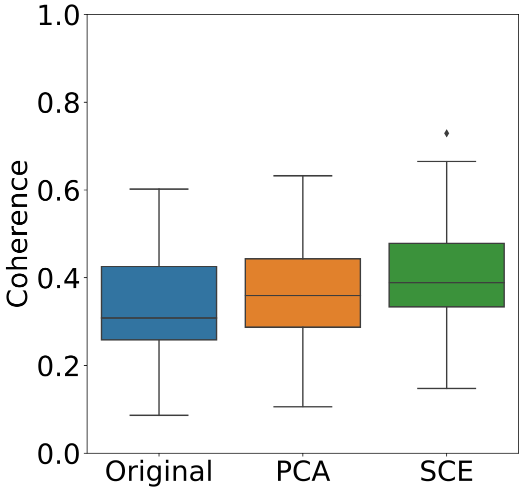


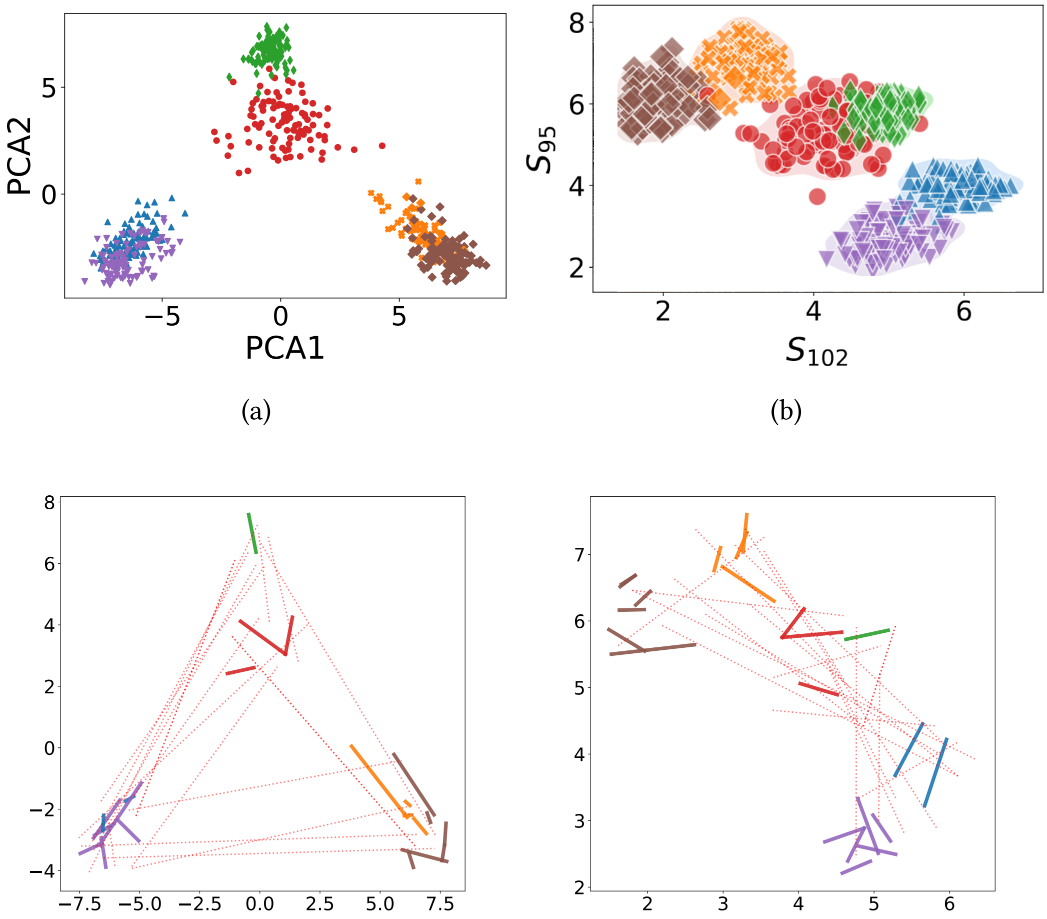
Figure 4.18: This figure shows the coherence for the reduced CDPS representation space in comparison to using all the space. GS-SCE was used to reduce the representational space and compared to PCA dimensionality reduction.

4.4 Conclusions

In this chapter, we presented the Shapelet Cluster Explanation (SCE) framework which offers a valuable solution for interpreting clustering algorithms' results for time series data. By leveraging the interpretability of shapelets, SCE ranks them based on their ability to provide insights into the clustering algorithm and individual clusters. This ranking helps identify the most informative set of shapelets, which accurately reflects specific clustering scenarios or the overall clustering outcome. Different approaches for ranking the shapelets were investigated termed as independent, combined and successive. These approaches can be used to achieve global-wise or cluster-wise explanations. It was shown that out of the proposed approaches the successive global wise approach, termed GS-SCE provided the best explanation for the clustering as it focus on ranking the shapelets based on the information brought by the addition of any new shapelet to the study.

Overall, the SCE framework not only offers interpretability for clustering results answering the "why" but also provides a means to reduce the number of shapelets required while maintaining comparable clustering performance. By leveraging shapelets' interpretability and the constraints modelled in the CDPS space, using SCE with CDPS offers a comprehensive approach to understanding and explaining clustering outcomes in complex time series data. Although SCE was only experimented on the representation learnt by CDPS the framework is applicable to any shapelet-based algorithm since it is independent of the shapelet learning process, the only inputs to the SCE approach are the cluster labels and the distance between the shapelets and the time series samples.

In addition to the study of the explainability of the space this chapter discusses



(a) (b)

(c) (d)

Figure 4.19: Illustration of the samples and constraints for the reduced CDPS representation space using PCA and GS-SCE. The example used in Section 4.2 is used herein to visually compare PCA with two principal components to GS-SCE. (a) and (b) show the scatter plots of the data points using the first two principal components of PCA and the first two shapelets ranked using GS-SCE. (c) and (d) show visualizations of a subsample of the constraints in each of the spaces shown in (a) and (b) respectively.

the representational space capabilities in modelling the constraints in both the original space and the reduced space. Studying the coherence of the constraints in the CDPS space in comparison to LDPS and DTW showed that CDPS is effective in modelling the constraints. The correlation study between the coherence and the clustering score showed that having better coherence might result in improved clustering and that CDPS does indeed improve the calculation of such measures. As well as studying the constraints in the overall space, a study over the reduced representational space is also presented where we found that the reduced space does model the constraints and retain the separability power of the representational space.

CDPS does, however, have some limitations (these were discussed in detail in relation to the GunPoint and GunPointAgeSpan datasets) that can be introduced by either the DTW distance or when insufficient constraints are provided by the expert. We discussed how the approximation to DTW may lead to reduced discrimination among data points and negatively affect constraint satisfaction. Note that this might be mitigated to a certain degree, as the parameters α and γ can be adjusted to increase the importance of constraints. However, this adjustment may introduce noise in the learning process, making it challenging to reach an equilibrium in the space.

Part III

Conclusions and Prospective Work

CHAPTER 5 | CONCLUSIONS

"The scientist is not a person who gives the right answers, he's one who asks the right questions."
– Claude Lévi-Strauss

This thesis presented the Constrained DTW Preserving Shapelets (CDPS) framework. This framework constructs a metric space that considers time series distortion and expert intuition. The key contribution of the study is the approximation of the elastic measure, Dynamic Time Warping (DTW) while modelling the expert intuition through the use of pairwise constraints. This framework falls under the semi-supervised paradigm and uses the concept of contrastive learning to learn and model the constraints of information. By leveraging the strengths of the elastic measure and an embedding measure, CDPS achieves a transformation of time series data into a metric space that is both distortion-invariant and aligned with expert intuition. Moreover, this space learns shapelets which are subsequences that represent discriminative features of the time series to discriminate between different categories. Since the shapelets are discriminative subsequences they are interpretable and hence the space learned by CDPS is explainable as well. The CDPS framework incorporates pairwise constraints, such as must-link and cannot-link constraints, provided by the expert through the contrastive loss, which ensures that similar points are closer to each other and dissimilar points are sufficiently far in the representational space. These constraints guide the learning process and ensure that the resulting transformation respects the expert's intuition. In order to approximate the DTW distance, the mean squared error between the actual DTW and the approximated one is minimized.

Using this space addresses two major questions in constraints clustering for time series data. First, how to calculate the properties of the constraints when most of the measures used to take distortion into account cannot be used to measure such properties. Second, to answer the question asked in "why" a clustering result is obtained. This work answers both concerns using the CDPS space and the second contribution of this work, Shapelet Cluster Explanation (SCE). SCE ranks the shapelets based on their ability to provide insights into the clustering algorithm and individual clusters by ranking them according to their informativeness (using the information gain). This ranking enables the identification of informative shapelets for cluster-level and global-level explanations, offering a comprehensive understanding of the clustering outcomes. At the cluster level, we provide an explanation of individual clusters in comparison to the other clusters and at the global level provide insights into the overall result of the clustering algorithm. In this work we distinguished between three different ways of ranking the shapelets, either taking each shapelet independently, termed independent, using combined shapelets by aggregating them, termed dependent, and finally ranking them based on the information they add to the previous best shapelet, termed successive. In addition to explaining the space using the successive approach for ranking the shapelets, the successive approach offers a way to reduce the number of shapelets without losing much in terms of clustering power.

To evaluate the effectiveness of CDPS, various experiments are conducted using multiple public time series datasets. The clustering performance of CDPS is compared with other algorithms, including unconstrained K-means, LDPS, COP-Kmeans, MIP-Kmeans, and DCC. The results consistently demonstrate the superiority of CDPS in both transductive and inductive settings. CDPS outperforms the competitors by effectively incorporating constraint information. The inductive study which cannot be used with classical clustering algorithms demonstrated the generalizable capabilities of the CDPS algorithm to unseen data.

Furthermore, the study explores the potential of CDPS for enhancing the performance of downstream algorithms. A comparison with the semi-supervised algorithm FeatTS showcases CDPS's ability to improve the performance of other algorithms by replacing their statistical features with the representation learned by CDPS. This shows that the features extracted by CDPS have more discriminative power than the statistical time series features. In addition, we discussed model selection, which identified that minimizing the loss function is a sufficient stopping criterion, and the algorithm is found to be relatively robust to changes in parameters. The discriminative nature of the CDPS space is examined, revealing its superiority compared to LDPS and DTW-based representations.

Studying the representation space of CDPS proved that CDPS is able to model the constraints in both the original space and using fewer shapelets, but in this case, it loses the approximation to DTW. This was achieved by analysing the coherence property of the constraints in comparison to DTW and LDPS space and measuring the average distance of the must-link points to those of the cannot-linked points. The coherence analysis of the constraints confirms that CDPS has the highest coherence out of LDPS and DTW and that the average distance is relatively low (less than one, which indicates good separation between the points). This analysis strengthens the fact that CDPS is able to model the constraints, and having a small ratio means that the CDPS was able to make use of the contrastive loss to decrease the distance between similar points and make it larger between non-similar points.

While CDPS demonstrates several advantages, it also has limitations. The approximation to DTW may lead to reduced discrimination among data points, negatively affecting constraint satisfaction. Depending on the constraints and DTW since in some cases the information between the two might be conflicting to the degree where it is not possible to reach an equilibrium and hence a solution. Adjusting CDPS's parameters can help in reducing and mitigating this issue but may introduce noise in the learning process. The study discusses these limitations in detail and provides insights into potential challenges and areas for improvement. Moreover, the current proposition cannot work with time series having missing data and domains where the time information is sensitive, since the shapelets are independent of the position they are found, although approximating DTW should account for this it is not guaranteed.

5.1 Perspectives

Having an explainable space that incorporates prior knowledge and models distance measures opens up many research directions. One interesting approach is to investigate whether this space can be used in active learning, where the expert can add constraints during representation training. The SCE framework enables the expert to gain insights into the clustering results. This insight can be valuable at any stage during active learning, allowing the expert to make informed decisions.

Another direction is to explore the use of different types of distance measures and determine if a dynamic approach for the distance measure can be implemented. This would allow for the learning of shapelets that better capture distortions while accommodating the constraints provided by the expert. These two directions represent the short-term perspective as implementing and testing these ideas should be simple and also allow for a concrete comparison to what comes next. Investigating the use of this space in a collaborative framework with other constraint algorithms is also an interesting avenue to explore, which will be the mid-term future work placing the building block to the long-term perspective. It would involve studying how this transformation affects collaborative learning between different algorithms. Furthermore, the proposed explanation framework in this work allows the expert to gain insights into the clustering results.

Another important question to answer is whether CDPS space can be used to guide the expert in proposing new constraints by providing suggestions to the expert. For example, after having a clustering of the data, either using the CDPS transformed data or other data representation, the CDPS transformation and SCE can be leveraged to provide explanations for the clustering results, this might enable a way forward to propose constraints based on the uncertain data samples and clusters identified through the analysis of shapelets and SCE explanations. This is the long-term perspective and that we desire to reach after investigating the necessary directions mentioned above. Thus, a closed loop can be established between the expert proposing constraints, the CDPS transformation, clustering and then proposing constraints, and so on, until a consensus is reached between the expert and the model. By establishing such a feedback loop, the expert's constraints and the model's learning process can continuously inform and influence each other, leading to a mutually agreed-upon representation.

BIBLIOGRAPHY

[1] Thomas Lampert, Thi-Bich-Hanh Dao, Baptiste Lafabregue, Nicolas Serrette, Germain Forestier, Bruno Crémilleux, Christel Vrain, and Pierre Gançarski. Constrained distance based clustering for time-series: a comparative and experimental study. *Data Mining and Knowledge Discovery*, 32:1663–1707, 2018.

[2] Paul S Bradley, Kristin P Bennett, and Ayhan Demiriz. Constrained k-means clustering. Technical Report MSR-TR-2000-65, Microsoft Research, 2000.

[3] Hongjing Zhang, Sugato Basu, and Ian Davidson. A framework for deep constrained clustering-algorithms and advances. In *Machine Learning and Knowledge Discovery in Databases*, pages 57–72. Springer, 2020.

[4] Ian Davidson, Kiri L Wagstaff, and Sugato Basu. Measuring constraint-set utility for partitional clustering algorithms. In *Knowledge Discovery in Databases in European Conference on Principles and Practice of Knowledge Discovery in Databases*, pages 115–126. Springer, 2006.

[5] Ling Zhuang and Honghua Dai. A maximal frequent itemset approach for web document clustering. In *International Conference on computer and Information Technology*, pages 970–977. IEEE, 2004.

[6] Yunlong Mi, Yong Shi, Jinhai Li, Wenqi Liu, and Mengyu Yan. Fuzzy-based concept learning method: Exploiting data with fuzzy conceptual clustering. *IEEE transactions on cybernetics*, 52(1):582–593, 2020.

[7] Lexiang Ye and Eamonn Keogh. Time series shapelets: a novel technique that allows accurate, interpretable and fast classification. *Data mining and knowledge discovery*, 22:149–182, 2011.

[8] Gustavo EAPA Batista, Eamonn J Keogh, Oben Moses Tataw, and Vinicius MA De Souza. CID: an efficient complexity-invariant distance for time series. *Data Mining and Knowledge Discovery*, 28:634–669, 2014.

[9] Philippe Esling and Carlos Agon. Time-series data mining. *ACM Computing Surveys*, 45(1):1–34, 2012.

[10] Shihyen Chen, Bin Ma, and Kaizhong Zhang. On the similarity metric and the distance metric. *Theoretical Computer Science*, 410(24-25):2365–2376, 2009.

[11] John Paparrizos, Chunwei Liu, Aaron J Elmore, and Michael J Franklin. Debunking four long-standing misconceptions of time-series distance measures. In *International conference on management of data*, pages 1887–1905, 2020.

[12] 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.

[13] 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.

[14] James W Hunt and Thomas G Szymanski. A fast algorithm for computing longest common subsequences. *Communications of the ACM*, 20(5):350–353, 1977.

[15] Eamonn J Keogh and Michael J Pazzani. An enhanced representation of time series which allows fast and accurate classification, clustering and relevance feedback. In *International Conference on Knowledge Discovery and Data Mining*, volume 98, pages 239–243, 1998.

[16] Anthony Bagnall, Jason Lines, Aaron Bostrom, James Large, and Eamonn Keogh. The great time series classification bake off: a review and experimental evaluation of recent algorithmic advances. *Data mining and knowledge discovery*, 31:606–660, 2017.

[17] Jason Lines and Anthony Bagnall. Time series classification with ensembles of elastic distance measures. *Data Mining and Knowledge Discovery*, 29:565–592, 2015.

[18] Mohammad Shokoohi-Yekta, Bing Hu, Hongxia Jin, Jun Wang, and Eamonn Keogh. Generalizing DTW to the multi-dimensional case requires an adaptive approach. *Data mining and knowledge discovery*, 31:1–31, 2017.

[19] 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.

[20] Fumitada Itakura. Minimum prediction residual principle applied to speech recognition. *IEEE Transactions on acoustics, speech, and signal processing*, 23 (1):67–72, 1975.

[21] Young-Seon Jeong, Myong K Jeong, and Olufemi A Omitaomu. Weighted dynamic time warping for time series classification. *Pattern recognition*, 44 (9):2231–2240, 2011.

[22] Victor Maus, Gilberto Câmara, Ricardo Cartaxo, Alber Sanchez, Fernando M Ramos, and Gilberto R De Queiroz. A time-weighted dynamic time warping method for land-use and land-cover mapping. *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, 9(8):3729–3739, 2016.

[23] J MacQueen. Classification and analysis of multivariate observations. In *Berkeley Symposium on Mathematical Statistics and Probability*, pages 281–297, 1967.

[24] François Petitjean, Alain Ketterlin, and Pierre Gançarski. A global averaging method for dynamic time warping, with applications to clustering. *Pattern recognition*, 44(3):678–693, 2011.

[25] Yaguang Tao, Alan Both, Rodrigo I Silveira, Kevin Buchin, Stef Sijben, Ross S Purves, Patrick Laube, Dongliang Peng, Kevin Toohey, and Matt Duckham. A comparative analysis of trajectory similarity measures. *GIScience & Remote Sensing*, 58(5):643–669, 2021.

[26] Christos Faloutsos, Mudumbai Ranganathan, and Yannis Manolopoulos. Fast subsequence matching in time-series databases. *ACM Sigmod Record*, 23(2): 419–429, 1994.

[27] Kin-Pong Chan and Ada Wai-Chee Fu. Efficient time series matching by wavelets. In *International Conference on Data Engineering*, pages 126–133, 1999.

[28] John Paparrizos and Michael J Franklin. Grail: efficient time-series representation learning. *Proceedings of the VLDB Endowment*, 12(11):1762–1777, 2019.

[29] Qi Lei, Jinfeng Yi, Roman Vaculin, Lingfei Wu, and Inderjit S Dhillon. Similarity preserving representation learning for time series clustering. *arXiv preprint arXiv:1702.03584*, 2017.

[30] Guoqing Zheng, Yiming Yang, and Jaime Carbonell. Efficient shift-invariant dictionary learning. In *International Conference on Knowledge Discovery and Data Mining*, pages 2095–2104, 2016.

[31] Lingfei Wu, Ian En-Hsu Yen, Jinfeng Yi, Fangli Xu, Qi Lei, and Michael Witbrock. Random warping series: A random features method for time-series embedding. In *International Conference on Artificial Intelligence and Statistics*, pages 793–802, 2018.

[32] Arnaud Lods, Simon Malinowski, Romain Tavenard, and Laurent Amsaleg. Learning DTW-preserving shapelets. In *Advances in Intelligent Data Analysis XVI*, pages 198–209, 2017.

[33] Eytan Domany. Superparamagnetic clustering of data—the definitive solution of an ill-posed problem. *Physica A: Statistical Mechanics and its Applications*, 263(1-4):158–169, 1999.

[34] Lexiang Ye and Eamonn Keogh. Time series shapelets: a new primitive for data mining. In *International conference on Knowledge discovery and data mining*, pages 947–956, 2009.

[35] Monica Arul and Ahsan Kareem. Applications of shapelet transform to time series classification of earthquake, wind and wave data. *Engineering Structures*, 228:111564, 2021.

[36] Thanawin Rakthanmanon and Eamonn Keogh. Fast shapelets: A scalable algorithm for discovering time series shapelets. In *International Conference on Data Mining*, pages 668–676, 2013.

[37] Abdullah Mueen, Eamonn Keogh, and Neal Young. Logical-shapelets: an expressive primitive for time series classification. In *International conference on Knowledge discovery and data mining*, pages 1154–1162, 2011.

[38] Kai-Wei Chang, Biplab Deka, Wen-Mei W Hwu, and Dan Roth. Efficient pattern-based time series classification on GPU. In *International Conference on Data Mining*, pages 131–140, 2012.

[39] Jason Lines, Luke M Davis, Jon Hills, and Anthony Bagnall. A shapelet transform for time series classification. In *International conference on Knowledge discovery and data mining*, pages 289–297, 2012.

[40] Jason Lines and Anthony Bagnall. Alternative quality measures for time series shapelets. In *Intelligent Data Engineering and Automated Learning*, pages 475–483. Springer, 2012.

[41] Josif Grabocka, Nicolas Schilling, Martin Wistuba, and Lars Schmidt-Thieme. Learning time-series shapelets. In *International conference on Knowledge discovery and data mining*, pages 392–401, 2014.

[42] Jesin Zakaria, Abdullah Mueen, and Eamonn Keogh. Clustering time series using unsupervised-shapelets. In *International Conference on Data Mining*, pages 785–794, 2012.

[43] Liudmila Ulanova, Nurjahan Begum, and Eamonn Keogh. Scalable clustering of time series with u-shapelets. In *International conference on data mining*, pages 900–908. SIAM, 2015.

[44] Vanel Steve Siyou Fotso, Engelbert Mephu Nguifo, and Philippe Vaslin. Frobenius correlation based u-shapelets discovery for time series clustering. *Pattern Recognition*, 103:107301, 2020.

[45] Jon Hills, Jason Lines, Edgaras Baranauskas, James Mapp, and Anthony Bagnall. Classification of time series by shapelet transformation. *Data mining and knowledge discovery*, 28:851–881, 2014.

[46] Saeed Aghabozorgi, Ali Seyed Shirkhorshidi, and Teh Ying Wah. Time-series clustering–a decade review. *Information systems*, 53:16–38, 2015.

[47] Absalom E. Ezugwu, Abiodun M. Ikotun, Olaide O. Oyelade, Laith Abualigah, Jeffery O. Agushaka, Christopher I. Eke, and Andronicus A. Akinyelu. A comprehensive survey of clustering algorithms: State-of-the-art machine learning applications, taxonomy, challenges, and future research prospects. *Engineering Applications of Artificial Intelligence*, 110:104743, 2022.

[48] Dibya Jyoti Bora, Dr Gupta, and Anil Kumar. A comparative study between fuzzy clustering algorithm and hard clustering algorithm. *arXiv preprint arXiv:1404.6059*, 2014.

[49] Fionn Murtagh and Pedro Contreras. Algorithms for hierarchical clustering: an overview. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, 2(1):86–97, 2012.

[50] Rui Xu and Donald Wunsch. Survey of clustering algorithms. *IEEE Transactions on neural networks*, 16(3):645–678, 2005.

[51] Lotfi A Zadeh. Fuzzy sets. *Information and control*, 8(3):338–353, 1965.

[52] Soumi Ghosh and Sanjay Kumar Dubey. Comparative analysis of k-means and fuzzy c-means algorithms. *International Journal of Advanced Computer Science and Applications*, 4(4), 2013.

[53] Maria Halkidi, Yannis Batistakis, and Michalis Vazirgiannis. Clustering algorithms and validity measures. In *International Conference on Scientific and Statistical Database Management. SSDBM 2001*, pages 3–22, 2001.

[54] Pierre Hansen and Brigitte Jaumard. Cluster analysis and mathematical programming. *Mathematical programming*, 79(1-3):191–215, 1997.

[55] Hans-Peter Kriegel, Peer Kröger, Jörg Sander, and Arthur Zimek. Density-based clustering. *Wiley interdisciplinary reviews: data mining and knowledge discovery*, 1(3):231–240, 2011.

[56] Ulrike Von Luxburg. A tutorial on spectral clustering. *Statistics and computing*, 17:395–416, 2007.

[57] Airel Pérez-Suárez, José F Martínez-Trinidad, and Jesús A Carrasco-Ochoa. A review of conceptual clustering algorithms. *Artificial Intelligence Review*, 52:1267–1296, 2019.

[58] Erxue Min, Xifeng Guo, Qiang Liu, Gen Zhang, Jianjing Cui, and Jun Long. A survey of clustering with deep learning: From the perspective of network architecture. *IEEE Access*, 6:39501–39514, 2018.

[59] Antoine Cornuéjols, Cédric Wemmert, Pierre Gançarski, and Younès Bennani. Collaborative clustering: Why, when, what and how. *Information Fusion*, 39:81–95, 2018.

[60] Sandro Vega-Pons and José Ruiz-Shulcloper. A survey of clustering ensemble algorithms. *International Journal of Pattern Recognition and Artificial Intelligence*, 25(03):337–372, 2011.

[61] Juhee Bae, Tove Helldin, Maria Riveiro, Sławomir Nowaczyk, Mohamed-Rafik Bouguelia, and Göran Falkman. Interactive clustering: A comprehensive review. *ACM Computing Surveys (CSUR)*, 53(1):1–39, 2020.

[62] Kristina P Sinaga and Miin-Shen Yang. Unsupervised K-means clustering algorithm. *IEEE access*, 8:80716–80727, 2020.

[63] Mohiuddin Ahmed, Raihan Seraj, and Syed Mohammed Shamsul Islam. The k-means algorithm: A comprehensive survey and performance evaluation. *Electronics*, 9(8):1295, 2020.

[64] Dorin Comaniciu and Peter Meer. Mean shift: A robust approach toward feature space analysis. *IEEE Transactions on pattern analysis and machine intelligence*, 24(5):603–619, 2002.

[65] Martin Ester, Hans-Peter Kriegel, Jörg Sander, Xiaowei Xu, et al. A density-based algorithm for discovering clusters in large spatial databases with noise. In *International Conference on Knowledge Discovery and Data Mining*, volume 96, pages 226–231, 1996.

[66] Russell Merris. Laplacian matrices of graphs: a survey. *Linear algebra and its applications*, 197:143–176, 1994.

[67] Francis Bach and Michael Jordan. Learning spectral clustering. *Advances in neural information processing systems*, 16, 2003.

[68] Douglas H Fisher. Knowledge acquisition via incremental conceptual clustering. *Machine learning*, 2:139–172, 1987.

[69] Stefan T Hadjitodorov and Ludmila I Kuncheva. Selecting diversifying heuristics for cluster ensembles. In *International Workshop on Multiple Classifier Systems*, pages 200–209, 2007.

[70] Cédric Wemmert, Pierre Gançarski, and Jerzy J. Korczak. A collaborative approach to combine multiple learning methods. *International Journal on Artificial Intelligence Tools*, 9(01):59–78, 2000.

[71] Pierre Gançarski and Cédric Wemmert. Collaborative multi-step mono-level multi-strategy classification. *Multimedia Tools and Applications*, 35:1–27, 2007.

[72] Witold Pedrycz. Collaborative fuzzy clustering. *Pattern Recognition Letters*, 23(14):1675–1686, 2002.

[73] Josef Kittler, Mohamad Hatef, Robert PW Duin, and Jiri Matas. On combining classifiers. *IEEE Transactions on pattern analysis and machine intelligence*, 20 (3):226–239, 1998.

[74] Alexander Strehl and Joydeep Ghosh. Cluster ensembles—a knowledge reuse framework for combining multiple partitions. *Journal of machine learning research*, 3(Dec):583–617, 2002.

[75] Ana LN Fred and Anil K Jain. Data clustering using evidence accumulation. In *International Conference on Pattern Recognition*, volume 4, pages 276–280. IEEE, 2002.

[76] Burr Settles. Active learning literature survey. Technical Report 1648, University of Wisconsin–Madison, 2009.

[77] Fan Min, Shi-Ming Zhang, Davide Ciucci, and Min Wang. Three-way active learning through clustering selection. *International Journal of Machine Learning and Cybernetics*, 11:1033–1046, 2020.

[78] Zalán Bodó, Zsolt Minier, and Lehel Csató. Active learning with clustering. In *Active Learning and Experimental Design workshop In conjunction with AISTATS*, pages 127–139, 2011.

[79] Lior Rokach and Oded Maimon. Clustering methods., 2005.

[80] Mayra Z Rodriguez, Cesar H Comin, Dalcimar Casanova, Odemir M Bruno, Diego R Amancio, Luciano da F Costa, and Francisco A Rodrigues. Clustering algorithms: A comparative approach. *Public Library of Science since (PloS) one*, 14(1):e0210236, 2019.

[81] Peter J. Rousseeuw. Silhouettes: A graphical aid to the interpretation and validation of cluster analysis. *Journal of Computational and Applied Mathematics*, 20:53–65, 1987. ISSN 0377-0427.

[82] William M Rand. Objective criteria for the evaluation of clustering methods. *Journal of the American Statistical association*, 66(336):846–850, 1971.

[83] Lawrence Hubert and Phipps Arabie. Comparing partitions. *Journal of classification*, 2:193–218, 1985.

[84] Matthijs J Warrens and Hanneke van der Hoef. Understanding the adjusted rand index and other partition comparison indices based on counting object pairs. *Journal of Classification*, 39(3):487–509, 2022.

[85] Alfréd Rényi. On measures of entropy and information. In *Berkeley Symposium on Mathematical Statistics and Probability, Volume 1: Contributions to the Theory of Statistics*, volume 4, pages 547–562, 1961.

[86] Kiri Wagstaff and Claire Cardie. Clustering with instance-level constraints. *American Association for Artificial Intelligence and Innovative Applications of Artificial Intelligence Conference*, 1097:577–584, 2000.

[87] Sugato Basu, Ian Davidson, and Kiri Wagstaff. *Constrained clustering: Advances in algorithms, theory, and applications.* CRC Press, 2008.

[88] Ian Davidson and SS Ravi. Clustering with constraints: Feasibility issues and the k-means algorithm. In *International conference on data mining*, pages 138–149. SIAM, 2005.

[89] Thi-Bich-Hanh Dao, Christel Vrain, Khanh-Chuong Duong, and Ian Davidson. A framework for actionable clustering using constraint programming. In *European Conference on Artificial Intelligence*, pages 453–461, 2016.

[90] Kiri L Wagstaff, Sugato Basu, and Ian Davidson. When is constrained clustering beneficial, and why? *Ionosphere*, 58(60.1):62–63, 2006.

[91] Jianghui Cai, Jing Hao, Haifeng Yang, Xujun Zhao, and Yuqing Yang. A review on semi-supervised clustering. *Information Sciences*, 2023.

[92] Kiri Wagstaff, Claire Cardie, Seth Rogers, Stefan Schrödl, et al. Constrained k-means clustering with background knowledge. In *International Conference on Machine Learning*, volume 1, pages 577–584, 2001.

[93] Wei Tan, Yan Yang, and Tianrui Li. An improved cop-kmeans algorithm for solving constraint violation. In *Computational Intelligence: Foundations and Applications*, pages 690–696. World Scientific, 2010.

[94] Tonny Rutayisire, Yan Yang, Chao Lin, and Jinyuan Zhang. A modified cop-kmeans algorithm based on sequenced cannot-link set. In *International Conference on Rough Sets and Knowledge Technology*, pages 217–225, 2011.

[95] Sugato Basu. Semi-supervised clustering by seeding. In *International Conference on Machine Learning*, 2002.

[96] Haichao Huang, Yong Cheng, and Ruilian Zhao. A semi-supervised clustering algorithm based on must-link set. In *Advanced Data Mining and Applications*, pages 492–499, 2008.

[97] Avgoustinos Vouros and Eleni Vasilaki. A semi-supervised sparse K-Means algorithm. *Pattern Recognition Letters*, 142:65–71, 2021.

[98] Daniela M Witten and Robert Tibshirani. A framework for feature selection in clustering. *Journal of the American Statistical Association*, 105(490):713–726, 2010.

[99] Wei Tang, Hui Xiong, Shi Zhong, and Jie Wu. Enhancing semi-supervised clustering: a feature projection perspective. In *International conference on Knowledge discovery and data mining*, pages 707–716, 2007.

[100] Inderjit S Dhillon and Dharmendra S Modha. Concept decompositions for large sparse text data using clustering. *Machine learning*, 42:143–175, 2001.

[101] Ayhan Demiriz, Kristin P Bennett, and Mark J Embrechts. Semi-supervised clustering using genetic algorithms. *Artificial neural networks in engineering*, pages 809–814, 1999.

[102] Dan Pelleg and Dorit Baras. K-means with large and noisy constraint sets. In *European Conference on Machine Learning*, pages 674–682, 2007.

[103] Sugato Basu, Arindam Banerjee, and Raymond J Mooney. Active semi-supervision for pairwise constrained clustering. In *International conference on data mining*, pages 333–344, 2004.

[104] Mohadeseh Ganji, James Bailey, and Peter J Stuckey. Lagrangian constrained clustering. In *International Conference on Data Mining*, pages 288–296, 2016.

[105] Nizar Grira, Michel Crucianu, and Nozha Boujemaa. Fuzzy clustering with pairwise constraints for knowledge-driven image categorisation. *IEEE Proceedings-Vision, Image and Signal Processing*, 153(3):299–304, 2006.

[106] Nizar Grira, Michel Crucianu, and Nozha Boujemaa. Active semi-supervised fuzzy clustering. *Pattern Recognition*, 41(5):1834–1844, 2008.

[107] Rong Ge, Martin Ester, Wen Jin, and Ian Davidson. Constraint-driven clustering. In *International conference on knowledge discovery and data mining*, pages 320–329, 2007.

[108] Ayhan Demiriz, Kristin P Bennett, and Paul S Bradley. Using assignment constraints to avoid empty clusters in k-means clustering. *Constrained clustering: advances in algorithms, theory, and applications*, 201, 2008.

[109] Arindam Banerjee and Joydeep Ghosh. Scalable clustering algorithms with balancing constraints. *Data mining and knowledge discovery*, 13:365–395, 2006.

[110] Dan Klein, Sepandar D Kamvar, and Christopher D Manning. From instance-level constraints to space-level constraints: Making the most of prior knowledge in data clustering. In *International conference on Machine learning*, 2002.

[111] Liu Yang and Rong Jin. Distance metric learning: A comprehensive survey. *Michigan State Universiy*, 2(2):4, 2006.

[112] Aurélien Bellet, Amaury Habrard, and Marc Sebban. Metric learning. *Synthesis lectures on artificial intelligence and machine learning*, 9(1):1–151, 2015.

[113] Eric Xing, Michael Jordan, Stuart J Russell, and Andrew Ng. Distance metric learning with application to clustering with side-information. *Advances in neural information processing systems*, 15, 2002.

[114] Aharon Bar-Hillel, Tomer Hertz, Noam Shental, Daphna Weinshall, and Greg Ridgeway. Learning a mahalanobis metric from equivalence constraints. *Journal of machine learning research*, 6(6), 2005.

[115] Aharon Bar-Hillel, Tomer Hertz, Noam Shental, and Daphna Weinshall. Learning distance functions using equivalence relations. In *International conference on machine learning*, pages 11–18, 2003.

[116] David Cohn, Rich Caruana, and Andrew McCallum. Semi-supervised clustering with user feedback. *Constrained clustering: advances in algorithms, theory, and applications*, 4(1):17–32, 2003.

[117] Mikhail Bilenko and Raymond J Mooney. Adaptive duplicate detection using learnable string similarity measures. In *International conference on Knowledge discovery and data mining*, pages 39–48, 2003.

[118] Steven CH Hoi, Wei Liu, and Shih-Fu Chang. Semi-supervised distance metric learning for collaborative image retrieval and clustering. *ACM Transactions on Multimedia Computing, Communications, and Applications (TOMM)*, 6(3):1–26, 2010.

[119] Mikhail Bilenko, Sugato Basu, and Raymond J Mooney. Integrating constraints and metric learning in semi-supervised clustering. In *International conference on Machine learning*, page 11, 2004.

[120] Jinfeng Yi, Rong Jin, Shaili Jain, Tianbao Yang, and Anil Jain. Semi-crowdsourced clustering: Generalizing crowd labeling by robust distance metric learning. *Advances in neural information processing systems*, 25, 2012.

[121] Kamvar Kamvar, Sepandar Sepandar, Klein Klein, Dan Dan, Manning Manning, and Christopher Christopher. Spectral learning. In *International Joint Conference of Artificial Intelligence*, 2003.

[122] Carlos Alzate and Johan AK Suykens. A regularized formulation for spectral clustering with pairwise constraints. In *International Joint Conference on Neural Networks*, pages 141–148, 2009.

[123] Xiang Wang and Ian Davidson. Active spectral clustering. In *International Conference on Data Mining*, pages 561–568, 2010.

[124] Xiang Wang, Buyue Qian, and Ian Davidson. On constrained spectral clustering and its applications. *Data Mining and Knowledge Discovery*, 28:1–30, 2014.

[125] Zhenguo Li, Jianzhuang Liu, and Xiaoou Tang. Constrained clustering via spectral regularization. In *Conference on Computer Vision and Pattern Recognition*, pages 421–428, 2009.

[126] Xiang Wang and Ian Davidson. Flexible constrained spectral clustering. In *International conference on Knowledge discovery and data mining*, pages 563–572, 2010.

[127] Ashraf Mohammed Iqbal, Abidalrahman Moh'd, and Zahoor Khan. Semi-supervised clustering ensemble by voting. *arXiv preprint arXiv:1208.4138*, 2012.

[128] Muna Al-Razgan and Carlotta Domeniconi. Clustering ensembles with active constraints. *Applications of Supervised and Unsupervised Ensemble Methods*, pages 175–189, 2009.

[129] Wenchao Xiao, Yan Yang, Hongjun Wang, Tianrui Li, and Huanlai Xing. Semi-supervised hierarchical clustering ensemble and its application. *Neurocomputing*, 173:1362–1376, 2016.

[130] Evgenia Dimitriadou, Andreas Weingessel, and Kurt Hornik. A mixed ensemble approach for the semi-supervised problem. In *Artificial Neural Networks—ICANN*, pages 571–576. Springer, 2002.

[131] Germain Forestier, Pierre Gançarski, and Cédric Wemmert. Collaborative clustering with background knowledge. *Data & Knowledge Engineering*, 69 (2):211–228, 2010.

[132] Carlotta Domeniconi and Muna Al-Razgan. Penta-training: Clustering ensembles with bootstrapping of constraints. In *Workshop on Supervised and Unsupervised Ensemble Methods and their Applications*, page 47, 2008.

[133] Germain Forestier, Cédric Wemmert, and Pierre Gancarski. Towards conflict resolution in collaborative clustering. In *International Conference Intelligent Systems*, pages 361–366, 2010.

[134] Marianne Mueller and Stefan Kramer. Integer linear programming models for constrained clustering. In *Discovery Science*, pages 159–173. Springer, 2010.

[135] Abdelkader Ouali, Samir Loudni, Yahia Lebbah, Patrice Boizumault, Albrecht Zimmermann, and Lakhdar Loukil. Efficiently finding conceptual clustering models with integer linear programming. In *International Joint Conference on Artificial Intelligence*, 2016.

[136] Ian Davidson, SS Ravi, and Leonid Shamis. A sat-based framework for efficient constrained clustering. In *International conference on data mining*, pages 94–105. SIAM, 2010.

[137] Thi-Bich-Hanh Dao, Khanh-Chuong Duong, and Christel Vrain. A declarative framework for constrained clustering. In *Machine Learning and Knowledge Discovery in Databases*, pages 419–434. Springer, 2013.

[138] Khanh-Chuong Duong, Christel Vrain, et al. Constrained clustering by constraint programming. *Artificial Intelligence*, 244:70–94, 2017.

[139] Ian Davidson, SS Ravi, and Martin Ester. Efficient incremental constrained clustering. In *International conference on Knowledge discovery and data mining*, pages 240–249, 2007.

[140] David D Lewis. A sequential algorithm for training text classifiers: Corrigendum and additional data. In *ACM Sigir Forum*, volume 29, pages 13–19. ACM New York, NY, USA, 1995.

[141] Burr Settles and Mark Craven. An analysis of active learning strategies for sequence labeling tasks. In *Conference on empirical methods in natural language processing*, pages 1070–1079, 2008.

[142] Toon Van Craenendonck, Wannes Meert, Sebastijan Dumančić, and Hendrik Blockeel. Cobrasts: A new approach to semi-supervised clustering of time series. In *International Conference on Discovery Science*, pages 179–193. Springer International Publishing, 2018.

[143] Yuhong Guo and Russell Greiner. Optimistic active-learning using mutual information. In *International Joint Conference on Artificial Intelligence*, volume 7, pages 823–829, 2007.

[144] Nicholas Roy and Andrew McCallum. Toward optimal active learning through monte carlo estimation of error reduction. *International Conference on Machine Learning*, 2:441–448, 2001.

[145] Abu Quwsar Ohi, Muhammad F Mridha, Farisa Benta Safir, Md Abdul Hamid, and Muhammad Mostafa Monowar. Autoembedder: a semi-supervised dnn embedding system for clustering. *Knowledge-Based Systems*, 204:106190, 2020.

[146] Fei Tian, Bin Gao, Qing Cui, Enhong Chen, and Tie-Yan Liu. Learning deep representations for graph clustering. In *Conference on Artificial Intelligence*, volume 28, 2014.

[147] Marek Śmieja, Łukasz Struski, and Mário AT Figueiredo. A classification-based approach to semi-supervised clustering with pairwise constraints. *Neural Networks*, 127:193–203, 2020.

[148] Jost Tobias Springenberg. Unsupervised and semi-supervised learning with categorical generative adversarial networks. *arXiv preprint arXiv:1511.06390*, 2015.

[149] Junyuan Xie, Ross Girshick, and Ali Farhadi. Unsupervised deep embedding for clustering analysis. In *International conference on machine learning*, pages 478–487. PMLR, 2016.

[150] Yazhou Ren, Kangrong Hu, Xinyi Dai, Lili Pan, Steven CH Hoi, and Zenglin Xu. Semi-supervised deep embedded clustering. *Neurocomputing*, 325:121–130, 2019.

[151] Pascal Vincent, Hugo Larochelle, Yoshua Bengio, and Pierre-Antoine Manzagol. Extracting and composing robust features with denoising autoencoders. In *International conference on Machine learning*, pages 1096–1103, 2008.

[152] Xifeng Guo, Long Gao, Xinwang Liu, and Jianping Yin. Improved deep embedded clustering with local structure preservation. In *International Joint Conference on Artificial Intelligence*, pages 1753–1759, 2017.

[153] Eamonn Keogh and Jessica Lin. Clustering of time-series subsequences is meaningless: implications for previous and future research. *Knowledge and information systems*, 8:154–177, 2005.

[154] Aristides Gionis and Heikki Mannila. Finding recurrent sources in sequences. In *International conference on Research in computational molecular biology*, pages 123–130, 2003.

[155] Fabian Mörchen, Alfred Ultsch, and Olaf Hoos. Extracting interpretable muscle activation patterns with time series knowledge mining. *International Journal of Knowledge-based and Intelligent Engineering Systems*, 9(3):197–208, 2005.

[156] T Warren Liao. Clustering of time series data—a survey. *Pattern recognition*, 38(11):1857–1874, 2005.

[157] Baptiste Lafabregue, Jonathan Weber, Pierre Gançarski, and Germain Forestier. Deep constrained clustering applied to satellite image time series. In *ECML/PKDD Workshop on Machine Learning for Earth Observation Data (MACLEAN)*, Würzburg, Germany, 2019.

[158] Donato Tiano, Angela Bonifati, and Raymond Ng. FeatTS: Feature-based time series clustering. In *International Conference on Management of Data*, pages 2784–2788, 2021.

[159] Laurens Van der Maaten and Geoffrey Hinton. Visualizing data using t-SNE. *Journal of machine learning research*, 9(11), 2008.

[160] Baptiste Lafabregue. *Clustering et apprentissage profond sous contraintes pour l'analyse de séries temporelles : application à l'analyse temporelle incrémentale en télédétection.* PhD thesis, 2021.

[161] Hassan Ismail Fawaz, Germain Forestier, Jonathan Weber, Lhassane Idoumghar, and Pierre-Alain Muller. Deep learning for time series classification: a review. *Data mining and knowledge discovery*, 33(4):917–963, 2019.

[162] Maximilian Christ, Nils Braun, Julius Neuffer, and Andreas W Kempa-Liehr. Time series feature extraction on basis of scalable hypothesis tests (tsfresh–a python package). *Neurocomputing*, 307:72–77, 2018.

[163] Yoav Benjamini and Daniel Yekutieli. The control of the false discovery rate in multiple testing under dependency. *Annals of statistics*, pages 1165–1188, 2001.

[164] Aaron van den Oord, Yazhe Li, and Oriol Vinyals. Representation learning with contrastive predictive coding. *arXiv preprint arXiv:1807.03748*, 2018.

[165] Phuc H Le-Khac, Graham Healy, and Alan F Smeaton. Contrastive representation learning: A framework and review. *IEEE Access*, 8:193907–193934, 2020.

[166] Sumit Chopra, Raia Hadsell, and Yann LeCun. Learning a similarity metric discriminatively, with application to face verification. In *Conference on Computer Vision and Pattern Recognition*, volume 1, pages 539–546. IEEE, 2005.

[167] Kilian Q Weinberger and Lawrence K Saul. Distance metric learning for large margin nearest neighbor classification. *Journal of machine learning research*, 10(2), 2009.

[168] Jing Wang, Jiangyun Li, Wei Li, Lingfei Xuan, Tianxiang Zhang, and Wenxuan Wang. Positive–negative equal contrastive loss for semantic segmentation. *Neurocomputing*, 535:13–24, 2023.

[169] Dejiao Zhang, Feng Nan, Xiaokai Wei, Shangwen Li, Henghui Zhu, Kathleen McKeown, Ramesh Nallapati, Andrew Arnold, and Bing Xiang. Supporting clustering with contrastive learning. *arXiv preprint arXiv:2103.12953*, 2021.

[170] Lanling Xu, Jianxun Lian, Wayne Xin Zhao, Ming Gong, Linjun Shou, Daxin Jiang, Xing Xie, and Ji-Rong Wen. Negative Sampling for Contrastive Representation Learning: A Review. *arXiv preprint arXiv:2206.00212*, 2022.

[171] Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A simple framework for contrastive learning of visual representations. In *International conference on machine learning*, pages 1597–1607. PMLR, 2020.

[172] Raia Hadsell, Sumit Chopra, and Yann LeCun. Dimensionality reduction by learning an invariant mapping. In *Conference on Computer Vision and Pattern Recognition*, volume 2, pages 1735–1742, 2006.

[173] Florian Schroff, Dmitry Kalenichenko, and James Philbin. Facenet: A unified embedding for face recognition and clustering. In *Conference on computer vision and pattern recognition*, pages 815–823, 2015.

[174] Eamonn Koegh Anthony Bagnall et al. The UCR Time Series Classification Archive, 2018.

[175] Anthony Bagnall, Hoang Anh Dau, Jason Lines, Michael Flynn, James Large, Aaron Bostrom, Paul Southam, and Eamonn Keogh. The UEA multivariate time series classification archive. *arXiv preprint arXiv:1811.00075*, 2018.

[176] A. Bagnall et al. The great time series classification bake off: a review and experimental evaluation of recent algorithmic advances. *Data mining and knowledge discovery*, 31:606–660, 2017.

[177] Claude Elwood Shannon and Warren Weaver. *The Mathematical Theory of Communication*. The University of Illinois Press, 1949.

[178] William H Kruskal. A nonparametric test for the several sample problem. *The Annals of Mathematical Statistics*, pages 525–540, 1952.

[179] Suresh Balakrishnama and Aravind Ganapathiraju. Linear discriminant analysis-a brief tutorial. *Institute for Signal and information Processing*, 18 (1998):1–8, 1998.

[180] Svante Wold, Kim Esbensen, and Paul Geladi. Principal component analysis. *Chemometrics and intelligent laboratory systems*, 2(1-3):37–52, 1987.

[181] Yingfan Wang, Haiyang Huang, Cynthia Rudin, and Yaron Shaposhnik. Understanding how dimension reduction tools work: An empirical approach to deciphering t-SNE, UMAP, TriMap, and PaCMAP for data visualization. *Journal of Machine Learning Research*, 22(201):1–73, 2021.

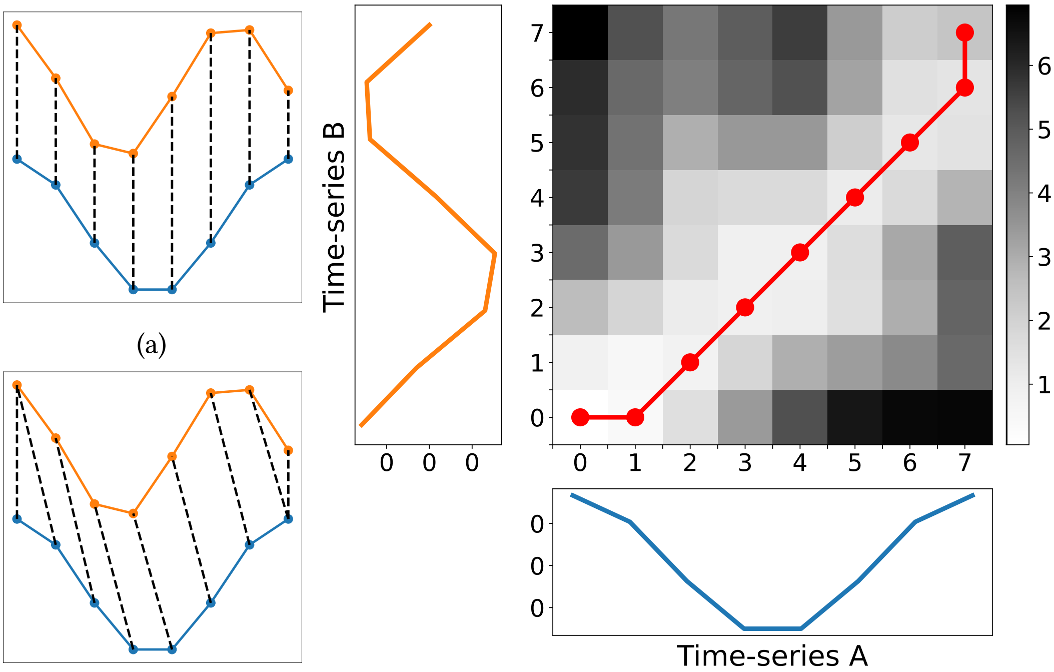
[182] Joseph B Kruskal. Multidimensional scaling by optimizing goodness of fit to a nonmetric hypothesis. *Psychometrika*, 29(1):1–27, 1964.

Part IV

Appendices

Appendix A | Dynamic Time Warping

This section presents Algorithm 8, which outlines the Dynamic Time Warping (DTW) algorithm. Additionally, Figure A.1 provides a visual representation of the optimal warping path for two distinct time series. Both the algorithm and the figure serve to reinforce and illustrate the concepts discussed and explained in Section 1.1.4.



(a)

(b)

(c)

Figure A.1: Illustration of mapping between points based on Euclidean distance (a) and DTW similarity (b). The wrapping path of DTW is shown in (c).

Algorithm 8 DYNAMIC TIME WARPING

Input: Two time series X and Y of lengths n and m respectively

Output: The DTW distance between X and Y

```
D [:, 0] = infinity
D[0, :] = infinity
D[0, 0] = 0
for i = 1 to n do
    for j = 1 to m do
        cost = distance(X[i], Y[j])
        D[i, j] = cost + min(D[i-1, j], D[i, j-1], D[i-1, j-1])
    end for
end for
return D[n, m]
```

Appendix B | Datasets

This section provides an overview of the datasets utilized in this thesis. The datasets are randomly chosen from the UCR archive [12].

Table B.1

List of UCR datasets used in the main study.

Dataset	Train size	Test size	Length	Classes	Dimensions
FaceAll	560	1690	131	14	1
MoteStrain	20	1252	84	2	1
Symbols	25	995	398	6	1
PenDigits	7494	3498	8	10	2
ScreenType	375	375	720	3	1
BasicMotions	40	40	100	4	6
ShapesAll	600	600	512	60	1
BME	30	150	128	3	1
Fungi	18	186	201	18	1
CBF	30	900	128	3	1
RacketSports	151	152	30	4	6
NATOPS	180	180	51	6	24
FiftyWords	450	455	270	50	1
Handwriting	150	850	152	26	3
OSULeaf	200	242	427	6	1
CricketY	390	390	300	12	1
SyntheticControl	300	300	60	6	1
Rock	20	50	2844	4	1
CricketX	390	390	300	12	1
GunPoint	50	150	150	2	1
Fish	175	175	463	7	1
Cricket	108	72	1197	12	6
Libras	180	180	45	15	2

(Continued on next page)

Table B.1 (continued)

Dataset	Train size	Test size	Length	Classes	Dimensions
BirdChicken	20	20	512	2	1
FaceFour	24	88	350	4	1
UWaveGestureLibrary	120	320	315	8	3
FacesUCR	200	2050	131	14	1
Phoneme	214	1896	1024	39	1
StandWalkJump	12	15	2500	3	4
Epilepsy	137	138	206	4	3
CricketZ	390	390	300	12	1
Mallat	55	2345	1024	8	1
AtrialFibrillation	15	15	640	3	2
EthanolConcentration	261	263	1751	4	3
PowerCons	180	180	144	2	1
HandMovementDirection	160	74	400	4	10
Lightning7	70	73	319	7	1
Plane	105	105	144	7	1
Lightning2	60	61	637	2	1
Adiac	390	391	176	37	1
Meat	60	60	448	3	1
SwedishLeaf	500	625	128	15	1
Heartbeat	204	205	405	2	61
ECG200	100	100	96	2	1
Car	60	60	577	4	1
Coffee	28	28	286	2	1
Herring	64	64	512	2	1
GunPointAgeSpan	135	316	150	2	1
ArticularyWordRecognition	275	300	144	25	9
Beef	30	30	470	5	1

Appendix C | Results

This section details the results discussed in Section 3.3.

Table C.1
Transductive NMI results for CDPS on the UCR archive.

	0%		5%		15%		25%	
	LDPS	DCC	CDPS	DCC	CDPS	DCC	CDPS	DCC
ACSF1	0.52	0.39	0.53	0.37	0.54	0.36	0.56	0.36
Adiac	0.69	0.57	0.69	0.58	0.7	0.58	0.7	0.58
ArrowHead	0.27	0.29	0.29	0.33	0.35	0.36	0.4	0.3
ArticularyWordRecognition	0.91	0.89	0.9	0.9	0.91	0.89	0.91	0.89
AtrialFibrillation	0.12	0.03	0.11	0.17	0.09	0.13	0.11	0.18
BME	0.44	0.2	0.45	0.28	0.68	0.35	0.8	0.31
BasicMotions	0.77	0.28	0.79	0.25	0.78	0.28	0.77	0.25
Beef	0.26	0.3	0.3	0.32	0.31	0.3	0.32	0.3
BeetleFly	0.16	0.05	0.21	0.09	0.29	0.06	0.41	0.11
BirdChicken	0.01	0.07	0.05	0.11	0.23	0.08	0.23	0.12
CBF	0.75	0.4	0.8	0.44	0.89	0.77	0.9	0.95
Car	0.18	0.23	0.28	0.27	0.41	0.28	0.46	0.26
Chinatown	0.4	0.02	0.73	0.47	0.83	0.47	0.84	0.39
ChlorineConcentration	0.0	0.0	0.01	0.0	0.01	0.0	0.01	0.0
CinCECGTorso	0.04	0.28	0.16	0.37	0.47	0.45	0.58	0.26
Coffee	0.63	0	0.71	0.55	0.93	0.53	0.96	0.55
Computers	0.05	0.01	0.04	0.01	0.03	0.02	0.02	0.02
Cricket	0.9	0.21	0.92	0.36	0.92	0.43	0.91	0.54
CricketX	0.38	0.23	0.36	0.22	0.36	0.23	0.39	0.23
CricketY	0.42	0.29	0.42	0.27	0.42	0.27	0.43	0.27
CricketZ	0.38	0.23	0.36	0.22	0.37	0.23	0.37	0.22
Crop	0.5	0.34	0.5	0.56	0.53	0.55	0.54	0.55
DistalPhalanxTW	0.55	0.47	0.52	0.58	0.52	0.59	0.53	0.63

(Continued on next page)

Table C.1 (continued)

	0%		5%		15%		25%	
Dataset	LDPS	DCC	CDPS	DCC	CDPS	DCC	CDPS	DCC
ECG200	0.02	0.18	0.06	0.26	0.27	0.43	0.29	0.44
ECG5000	0.43	0.55	0.62	0.73	0.64	0.77	0.65	0.76
ECGFiveDays	0.02	0.01	0.84	0.76	0.99	1.0	1.0	0.99
EOGHorizontalSignal	0.38	0.36	0.4	0.37	0.45	0.37	0.46	0.37
EOGVerticalSignal	0.32	0.34	0.35	0.33	0.39	0.33	0.41	0.33
Earthquakes	0.04	0.02	0.04	0.01	0.04	0.01	0.04	0.01
ElectricDevices	0.36	0.13	0.38	0.07	0.42	0.07	0.44	0.07
Epilepsy	0.63	0.04	0.64	0.07	0.68	0.08	0.72	0.08
EthanolConcentration	0.0	0.0	0.0	0.0	0.0	0.0	0.01	0.0
EthanolLevel	0.01	0.01	0.01	0.01	0.03	0.01	0.06	0.01
FaceAll	0.67	0.35	0.66	0.4	0.67	0.37	0.7	0.4
FaceFour	0.58	0	0.61	0.49	0.6	0.51	0.68	0.58
FacesUCR	0.68	0.51	0.66	0.46	0.65	0.48	0.69	0.49
FiftyWords	0.67	0.63	0.67	0.62	0.68	0.61	0.67	0.61
FingerMovements	0.0	0.01	0.0	0.01	0.0	0.01	0.0	0.01
Fish	0.31	0.36	0.44	0.3	0.54	0.29	0.64	0.29
FordA	0.0	0.0	0.39	0.01	0.53	0.01	0.56	0.01
FordB	0.05	0.0	0.36	0.0	0.52	0.0	0.56	0.0
FreezerRegularTrain	0.1	0.22	0.89	0.23	0.95	0.22	0.92	0.22
FreezerSmallTrain	0.1	0.23	0.87	0.23	0.92	0.24	0.92	0.22
Fungi	0.89	0.89	0.88	0.0	0.89	0.0	0.9	0.0
GunPoint	0.0	0.0	0.08	0.09	0.68	0.16	0.81	0.37
GunPointAgeSpan	0.0	0.5	0.0	0.36	0.0	0.55	0.0	0.42
GunPointMaleVersusFemale	0.58	0.08	0.8	0.32	0.82	0.84	0.86	0.84
GunPointOldVersusYoung	0.02	0.07	0.1	0.19	0.55	0.57	0.46	0.62
Ham	0.03	0.1	0.04	0.14	0.09	0.23	0.14	0.22
HandMovementDirection	0.02	0.04	0.02	0.04	0.02	0.03	0.02	0.03
Handwriting	0.44	0.27	0.41	0.28	0.37	0.27	0.37	0.27
Haptics	0.1	0.09	0.09	0.1	0.11	0.11	0.12	0.1
Heartbeat	0.01	0.0	0.01	0.0	0.01	0.0	0.01	0.0
Herring	0.03	0.0	0.02	0.01	0.01	0.01	0.03	0.01
HouseTwenty	0.53	0.13	0.57	0.13	0.66	0.17	0.67	0.22
InlineSkate	0.11	0.05	0.08	0.06	0.1	0.05	0.13	0.05
InsectEPGRegularTrain	0.36	0.13	0.43	0.24	0.53	0.3	0.62	0.3

(Continued on next page)

Table C.1 (continued)

	0%		5%		15%		25%	
Dataset	LDPS	DCC	CDPS	DCC	CDPS	DCC	CDPS	DCC
InsectEPGSmallTrain	0.36	0.18	0.44	0.25	0.53	0.27	0.62	0.34
InsectWingbeatSound	0.22	0.55	0.38	0.53	0.45	0.54	0.49	0.54
ItalyPowerDemand	0.01	0.01	0.64	0.81	0.77	0.67	0.79	0.08
LargeKitchenAppliances	0.13	0.02	0.13	0.03	0.17	0.03	0.24	0.03
Libras	0.63	0.54	0.62	0.53	0.63	0.54	0.63	0.54
Lightning2	0.11	0.04	0.11	0.04	0.09	0.07	0.07	0.05
Lightning7	0.54	0.42	0.56	0.44	0.56	0.44	0.57	0.43
Mallat	0.87	0.87	0.88	0.85	0.88	0.83	0.89	0.85
Meat	0.76	0.49	0.67	0.54	0.59	0.51	0.64	0.55
MedicalImages	0.33	0.26	0.27	0.23	0.31	0.25	0.33	0.24
MiddlePhalanxOutlineAgeGroup	0.4	0.39	0.39	0.42	0.38	0.41	0.39	0.43
MiddlePhalanxOutlineCorrect	0.0	0.02	0.0	0.03	0.0	0.02	0.0	0.02
MiddlePhalanxTW	0.41	0.45	0.41	0.46	0.42	0.5	0.42	0.49
MixedShapesRegularTrain	0.56	0.34	0.63	0.49	0.68	0.57	0.7	0.5
MixedShapesSmallTrain	0.55	0.47	0.62	0.52	0.68	0.58	0.69	0.52
MoteStrain	0.09	0.49	0.47	0.58	0.63	0.68	0.68	0.75
NATOPS	0.65	0.44	0.64	0.45	0.64	0.47	0.63	0.51
NonInvasiveFetalECGThorax1	0.7	0.7	0.74	0.71	0.81	0.71	0.82	0.71
OSULeaf	0.29	0.2	0.36	0.18	0.41	0.19	0.43	0.19
OliveOil	0.53	0.49	0.53	0.37	0.58	0.5	0.57	0.49
PenDigits	0.68	0	0.71	0.68	0.73	0.82	0.74	0
PhalangesOutlinesCorrect	0.0	0.02	0.01	0.03	0.01	0.01	0.0	0.01
Phoneme	0.32	0.16	0.29	0.16	0.29	0.16	0.29	0.15
PigAirwayPressure	0.63	0.55	0.62	0.54	0.63	0.54	0.63	0.54
PigArtPressure	0.84	0.63	0.84	0.63	0.83	0.62	0.84	0.62
PigCVP	0.65	0.55	0.67	0.56	0.68	0.56	0.69	0.56
Plane	0.89	0.84	0.89	0.86	0.9	0.85	0.89	0.88
PowerCons	0.43	0.43	0.28	0.32	0.43	0.42	0.59	0.91
ProximalPhalanxOutlineAgeGroup	0.53	0.44	0.49	0.51	0.49	0.52	0.51	0.5
ProximalPhalanxOutlineCorrect	0.08	0.07	0.08	0.1	0.08	0.12	0.08	0.11
ProximalPhalanxTW	0.58	0.57	0.55	0.6	0.55	0.6	0.56	0.6
RacketSports	0.61	0.23	0.61	0.21	0.62	0.23	0.61	0.29
RefrigerationDevices	0.05	0.02	0.05	0.02	0.06	0.02	0.07	0.02
Rock	0.26	0.43	0.25	0.0	0.26	0.0	0.25	0.0

(Continued on next page)

Table C.1 (continued)

	0%		5%		15%		25%	
Dataset	LDPS	DCC	CDPS	DCC	CDPS	DCC	CDPS	DCC
ScreenType	0.02	0.02	0.02	0.03	0.02	0.03	0.01	0.03
SemgHandGenderCh2	0.08	0.13	0.08	0.12	0.1	0.13	0.13	0.14
SemgHandMovementCh2	0.18	0.18	0.18	0.18	0.18	0.18	0.18	0.16
ShapeletSim	0.02	0.01	0.04	0.03	0.05	0.02	0.06	0.03
ShapesAll	0.75	0.69	0.75	0.69	0.75	0.68	0.76	0.69
SmallKitchenAppliances	0.24	0.01	0.24	0.01	0.25	0.01	0.26	0.01
SmoothSubspace	0.59	0.47	0.49	0.43	0.67	0.43	0.74	0.5
SonyAIBORobotSurface1	0.69	0.73	0.77	0.83	0.9	0.96	0.91	0.95
SonyAIBORobotSurface2	0.31	0.24	0.63	0.57	0.81	0.79	0.88	0.9
StandWalkJump	0.16	0.05	0.12	0.12	0.13	0.16	0.15	0.14
Strawberry	0.12	0.14	0.49	0.16	0.64	0.17	0.63	0.15
SwedishLeaf	0.66	0.6	0.68	0.55	0.71	0.54	0.73	0.54
Symbols	0.79	0.81	0.85	0.82	0.85	0.85	0.85	0.83
SyntheticControl	0.86	0.58	0.88	0.52	0.86	0.53	0.88	0.57
ToeSegmentation1	0.03	0.01	0.18	0.01	0.63	0.02	0.73	0.04
ToeSegmentation2	0.14	0.02	0.33	0.04	0.47	0.05	0.61	0.06
Trace	0.7	0.54	0.79	0.57	0.85	0.61	0.88	0.6
TwoLeadECG	0.06	0.0	0.92	0.67	1.0	0.54	1.0	0.12
TwoPatterns	0.91	0.02	0.69	0.02	0.76	0.03	0.78	0.04
UMD	0.37	0.37	0.32	0.21	0.41	0.21	0.46	0.23
UWaveGestureLibrary	0.66	0.6	0.64	0.65	0.65	0.64	0.66	0.63
UWaveGestureLibraryAll	0.5	0.72	0.5	0.72	0.66	0.75	0.71	0.71
UWaveGestureLibraryX	0.45	0.48	0.45	0.46	0.49	0.45	0.5	0.44
UWaveGestureLibraryY	0.42	0.44	0.42	0.4	0.44	0.4	0.46	0.38
Wafer	0.0	0.01	0.28	0.84	0.28	0.21	0.32	0.16
Wine	0.01	0.0	0.01	0.01	0.02	0.01	0.02	0.02
WordSynonyms	0.49	0.42	0.45	0.41	0.47	0.41	0.49	0.41
Worms	0.16	0.09	0.1	0.07	0.11	0.07	0.13	0.08
WormsTwoClass	0.03	0.01	0.02	0.02	0.03	0.02	0.04	0.03
Yoga	0.01	0.01	0.0	0.03	0.0	0.0	0.0	0.0

Table C.2
Inductive NMI results for CDPS on the UCR archive.

	0%		5%		15%		25%	
Dataset	LDPS	DCC	CDPS	DCC	CDPS	DCC	CDPS	DCC
ACSF1	0.57	0.31	0.54	0.35	0.55	0.39	0.56	0.35
Adiac	0.73	0.6	0.73	0.59	0.73	0.6	0.73	0.6
ArrowHead	0.25	0.46	0.36	0.48	0.37	0.47	0.42	0.45
ArticularyWordRecognition	0.89	0	0.89	0.89	0.88	0.89	0.88	0.88
AtrialFibrillation	0.16	0	0.17	0.28	0.17	0.25	0.18	0.3
BME	0.52	0.3	0.46	0.35	0.42	0.43	0.47	0.42
BasicMotions	0.88	0	0.87	0.34	0.88	0.29	0.89	0.35
Beef	0.33	0.35	0.33	0.37	0.36	0.39	0.35	0.41
BeetleFly	0.14	0.19	0.17	0.12	0.21	0.1	0.3	0.11
BirdChicken	0.08	0.3	0.09	0.22	0.18	0.33	0.16	0.25
CBF	0.8	0.48	0.78	0.46	0.79	0.43	0.8	0.53
Car	0.27	0.26	0.3	0.28	0.32	0.27	0.36	0.28
Chinatown	0.2	0.01	0.24	0.05	0.34	0.02	0.3	0.07
ChlorineConcentration	0.0	0.01	0.01	0.01	0.01	0.02	0.01	0.02
CinCECGTorso	0.06	0.24	0.15	0.3	0.19	0.33	0.21	0.35
Coffee	0.28	0	0.37	0.64	0.59	0.66	0.77	0.64
Computers	0.05	0.02	0.03	0.02	0.03	0.02	0.02	0.03
Cricket	0.88	0	0.89	0.62	0.89	0.55	0.88	0.6
CricketX	0.37	0.26	0.37	0.27	0.36	0.27	0.39	0.27
CricketY	0.46	0.29	0.44	0.28	0.44	0.29	0.43	0.28
CricketZ	0.37	0.26	0.37	0.26	0.38	0.26	0.39	0.26
Crop	0.49	0.46	0.47	0.56	0.5	0.56	0.52	0.56
DistalPhalanxTW	0.52	0.53	0.52	0.63	0.52	0.65	0.53	0.67
ECG200	0.02	0.2	0.06	0.27	0.16	0.34	0.17	0.43
ECG5000	0.44	0.62	0.55	0.67	0.61	0.76	0.61	0.79
ECGFiveDays	0.04	0.06	0.12	0.19	0.18	0.14	0.39	0.17
EOGHorizontalSignal	0.37	0.43	0.47	0.44	0.48	0.45	0.5	0.44
EOGVerticalSignal	0.39	0.35	0.38	0.39	0.43	0.39	0.44	0.38
Earthquakes	0.06	0.0	0.03	0.01	0.03	0.01	0.03	0.03
ElectricDevices	0.4	0.04	0.4	0.14	0.44	0.14	0.45	0.13
Epilepsy	0.62	0	0.59	0.07	0.61	0.08	0.61	0.07
EthanolConcentration	0.01	0	0.01	0.01	0.01	0.01	0.01	0.01
EthanolLevel	0.01	0.01	0.02	0.02	0.02	0.02	0.03	0.01

(Continued on next page)

Table C.2 (continued)

	0%		5%		15%		25%	
Dataset	LDPS	DCC	CDPS	DCC	CDPS	DCC	CDPS	DCC
FaceAll	0.66	0.44	0.65	0.42	0.65	0.43	0.66	0.42
FaceFour	0.6	0	0.59	0.51	0.6	0.54	0.58	0.51
FacesUCR	0.62	0.54	0.58	0.53	0.61	0.54	0.61	0.54
FiftyWords	0.7	0.67	0.7	0.67	0.7	0.66	0.7	0.66
Fish	0.42	0.36	0.46	0.35	0.51	0.35	0.56	0.34
FordA	0.01	0.01	0.32	0.01	0.5	0.01	0.55	0.01
FordB	0.02	0.02	0.26	0.0	0.41	0.01	0.44	0.0
FreezerRegularTrain	0.1	0.27	0.24	0.26	0.63	0.25	0.78	0.25
FreezerSmallTrain	0.1	0.33	0.22	0.35	0.35	0.37	0.38	0.4
Fungi	0.85	1.0	0.85	0.0	0.85	0.0	0.85	0.0
GunPoint	0.0	0.03	0.02	0.13	0.06	0.19	0.17	0.12
GunPointAgeSpan	0.0	0.52	0.0	0.43	0.0	0.43	0.0	0.46
GunPointMaleVersusFemale	0.61	0.15	0.65	0.24	0.82	0.3	0.85	0.73
GunPointOldVersusYoung	0.02	0.07	0.05	0.18	0.12	0.21	0.11	0.34
Ham	0.11	0.05	0.03	0.12	0.07	0.12	0.12	0.21
HandMovementDirection	0.04	0	0.05	0.05	0.05	0.05	0.05	0.06
Handwriting	0.38	0	0.37	0.55	0.37	0.55	0.37	0.54
Haptics	0.09	0.16	0.12	0.14	0.14	0.13	0.14	0.13
Heartbeat	0.01	0	0.01	0.0	0.01	0.0	0.01	0.0
Herring	0.06	0.02	0.04	0.02	0.03	0.02	0.03	0.01
HouseTwenty	0.33	0.26	0.56	0.19	0.66	0.19	0.63	0.25
InlineSkate	0.11	0.23	0.2	0.23	0.19	0.23	0.19	0.21
InsectEPGRegularTrain	0.32	0.28	0.35	0.34	0.42	0.33	0.42	0.37
InsectEPGSmallTrain	0.36	0.18	0.49	0.21	0.49	0.3	0.52	0.24
InsectWingbeatSound	0.24	0.57	0.32	0.58	0.39	0.57	0.42	0.57
ItalyPowerDemand	0.01	0.78	0.06	0.2	0.34	0.21	0.46	0.62
LargeKitchenAppliances	0.14	0.04	0.15	0.06	0.16	0.05	0.26	0.06
Libras	0.65	0	0.65	0.55	0.64	0.54	0.64	0.55
Lightning2	0.19	0.16	0.17	0.04	0.13	0.03	0.09	0.09
Lightning7	0.6	0.44	0.58	0.46	0.58	0.48	0.58	0.47
Mallat	0.89	0.97	0.87	0.93	0.86	0.93	0.87	0.93
Meat	0.74	0.52	0.73	0.63	0.64	0.52	0.62	0.66
MedicalImages	0.34	0.24	0.28	0.26	0.29	0.26	0.31	0.29
MiddlePhalanxOutlineAgeGroup	0.1	0.45	0.43	0.44	0.43	0.46	0.43	0.45

(Continued on next page)

Table C.2 (continued)

	0%		5%		15%		25%	
Dataset	LDPS	DCC	CDPS	DCC	CDPS	DCC	CDPS	DCC
MiddlePhalanxOutlineCorrect	0.07	0.02	0.0	0.03	0.0	0.05	0.0	0.04
MiddlePhalanxTW	0.41	0.47	0.43	0.49	0.43	0.51	0.43	0.53
MixedShapesRegularTrain	0.54	0.42	0.52	0.45	0.61	0.43	0.64	0.47
MixedShapesSmallTrain	0.53	0.5	0.54	0.54	0.58	0.56	0.59	0.55
MoteStrain	0.07	0.28	0.04	0.24	0.14	0.26	0.26	0.32
NATOPS	0.64	0	0.64	0.44	0.64	0.46	0.63	0.51
OSULeaf	0.27	0.24	0.32	0.23	0.34	0.24	0.37	0.25
OliveOil	0.65	0.37	0.57	0.55	0.59	0.56	0.59	0.55
PenDigits	0	0	0	0.7	0	0.82	0	0
PhalangesOutlinesCorrect	0.03	0.05	0.0	0.04	0.0	0.01	0.0	0.01
Phoneme	0.24	0.52	0.23	0.51	0.23	0.51	0.23	0.51
PigAirwayPressure	0.69	0.82	0.83	0.82	0.81	0.82	0.82	0.82
PigArtPressure	0.86	0.83	0.88	0.84	0.87	0.84	0.87	0.84
PigCVP	0.74	0.81	0.83	0.8	0.83	0.8	0.84	0.8
Plane	0.87	0.82	0.88	0.84	0.89	0.85	0.89	0.86
PowerCons	0.32	0.48	0.28	0.31	0.34	0.29	0.43	0.36
ProximalPhalanxOutlineAgeGroup	0.51	0.46	0.51	0.45	0.49	0.47	0.51	0.46
ProximalPhalanxOutlineCorrect	0.14	0.05	0.05	0.08	0.06	0.1	0.07	0.09
ProximalPhalanxTW	0.57	0.5	0.55	0.62	0.57	0.64	0.56	0.66
RacketSports	0.48	0	0.51	0.29	0.49	0.27	0.53	0.31
RefrigerationDevices	0.06	0.04	0.04	0.03	0.06	0.02	0.07	0.03
Rock	0.31	0.24	0.33	0.0	0.31	0.0	0.32	0.0
ScreenType	0.02	0.04	0.02	0.05	0.02	0.04	0.01	0.05
SemgHandGenderCh2	0.07	0.14	0.09	0.14	0.11	0.2	0.14	0.26
SemgHandMovementCh2	0.17	0.2	0.19	0.23	0.19	0.22	0.2	0.22
ShapeletSim	0.65	0.07	0.04	0.06	0.06	0.07	0.08	0.06
ShapesAll	0.76	0.72	0.76	0.73	0.77	0.73	0.77	0.73
SmallKitchenAppliances	0.3	0.01	0.2	0.01	0.2	0.02	0.2	0.01
SmoothSubspace	0.47	0.43	0.5	0.42	0.68	0.45	0.71	0.42
SonyAIBORobotSurface1	0.35	0.1	0.34	0.23	0.42	0.26	0.46	0.21
SonyAIBORobotSurface2	0.26	0.3	0.36	0.46	0.39	0.47	0.49	0.35
StandWalkJump	0.21	0	0.18	0.19	0.17	0.26	0.17	0.23
Strawberry	0.1	0.14	0.33	0.16	0.62	0.18	0.62	0.17
SwedishLeaf	0.67	0.59	0.69	0.54	0.71	0.52	0.72	0.52

(Continued on next page)

Table C.2 (continued)

	0%		5%		15%		25%	
Dataset	LDPS	DCC	CDPS	DCC	CDPS	DCC	CDPS	DCC
Symbols	0.83	0.78	0.83	0.8	0.84	0.8	0.84	0.81
SyntheticControl	0.82	0.54	0.86	0.55	0.87	0.54	0.87	0.58
ToeSegmentation1	0.09	0.03	0.05	0.02	0.16	0.03	0.32	0.04
ToeSegmentation2	0.44	0.01	0.51	0.01	0.59	0.03	0.66	0.01
Trace	0.74	0.56	0.73	0.59	0.8	0.62	0.83	0.63
TwoLeadECG	0.07	0.09	0.08	0.23	0.16	0.17	0.31	0.14
TwoPatterns	0.88	0.06	0.74	0.03	0.69	0.04	0.71	0.06
UMD	0.32	0.24	0.34	0.28	0.35	0.28	0.34	0.29
UWaveGestureLibrary	0.61	0	0.64	0.71	0.63	0.71	0.64	0.69
UWaveGestureLibraryAll	0.5	0.68	0.41	0.71	0.52	0.72	0.59	0.75
UWaveGestureLibraryX	0.45	0.47	0.43	0.46	0.46	0.46	0.46	0.47
UWaveGestureLibraryY	0.42	0.45	0.4	0.45	0.42	0.44	0.44	0.4
Wafer	0.0	0.01	0.01	0.16	0.04	0.78	0.05	0.32
Wine	0.0	0.09	0.02	0.07	0.02	0.07	0.03	0.07
WordSynonyms	0.5	0.5	0.51	0.52	0.53	0.52	0.54	0.5
Worms	0.18	0.09	0.11	0.09	0.11	0.1	0.12	0.1
WormsTwoClass	0.0	0.01	0.02	0.02	0.02	0.03	0.03	0.03
Yoga	0.0	0.01	0.0	0.01	0.0	0.04	0.01	0.03

Table C.3

Transductive NMI results. KM, CKM, MKM represents K-means, COP-Kmeans, MIP-Kmeans respectively.

	0%			5%				15%				25%			
Dataset	LDPS	KM	DCC	CDPS	CKM	MKM	DCC	CDPS	CKM	MKM	DCC	CDPS	CKM	MKM	DCC
Adiac	0.69	0.58	0.57	0.69	0.58	0.61	0.58	0.7	0.58	0.61	0.58	0.7	0.57	0.61	0.58
ArticularyWordRecognition	0.91	0.92	0.89	0.9	0.92	0.92	0.9	0.91	0.91	0.93	0.89	0.91	0.92	0.93	0.89
AtrialFibrillation	0.12	0.1	0.03	0.11	0.08	0.05	0.17	0.09	0.08	0.03	0.13	0.11	0.1	0.05	0.18
BME	0.44	0.52	0.2	0.45	0.55	0.46	0.28	0.68	0.48	0.49	0.35	0.8	0.52	0.48	0.31
BasicMotions	0.77	0.86	0.28	0.79	0.77	0.77	0.25	0.78	0.76	0.73	0.28	0.77	0.76	0.7	0.25
Beef	0.26	0.31	0.3	0.3	0.29	0.31	0.32	0.31	0.29	0.28	0.3	0.32	0.27	0.28	0.3
BirdChicken	0.01	0.02	0.07	0.05	0.02	0.12	0.11	0.23	0.05	0.06	0.08	0.23	0.1	0.09	0.12
CBF	0.75	0.77	0.4	0.8	0.77	0.78	0.44	0.89	0.75	0.78	0.77	0.9	0.75	0.8	0.95
Car	0.18	0.18	0.23	0.28	0.19	0.17	0.27	0.41	0.19	0.19	0.28	0.46	0.18	0.19	0.26
Coffee	0.63	0.7	0	0.71	0.62	0.72	0.55	0.93	0.69	0.75	0.53	0.96	0.69	0.77	0.55
Cricket	0.9	0.92	0.21	0.92	0.9	0.91	0.36	0.92	0.91	0.89	0.43	0.91	0.91	0.88	0.54
CricketX	0.38	0.49	0.23	0.36	0.47	0.48	0.22	0.36	0.48	0.47	0.23	0.39	0.49	0.48	0.23
CricketY	0.42	0.47	0.29	0.42	0.46	0.48	0.27	0.42	0.47	0.49	0.27	0.43	0.47	0.46	0.27
CricketZ	0.38	0.47	0.23	0.36	0.49	0.49	0.22	0.37	0.5	0.5	0.23	0.37	0.49	0.49	0.22
ECG200	0.02	0.01	0.18	0.06	0.03	0.02	0.26	0.27	0.04	0.05	0.43	0.29	0.05	0.1	0.44
Epilepsy	0.63	0.17	0.04	0.64	0.19	0.31	0.07	0.68	0.19	0.38	0.08	0.72	0.17	0.34	0.08
EthanolConcentration	0.0	0	0.0	0.0	0	0.01	0.0	0.0	0	0.0	0.0	0.01	0	0.01	0.0

(Continued on next page)

Table C.3 (continued)

	0%			5%				15%				25%			
Dataset	LDPS	KM	DCC	CDPS	CKM	MKM	DCC	CDPS	CKM	MKM	DCC	CDPS	CKM	MKM	DCC
FaceAll	0.67	0.66	0.35	0.66	0.65	0.68	0.4	0.67	0.65	0.67	0.37	0.7	0.66	0.69	0.4
FaceFour	0.58	0.55	0	0.61	0.5	0.51	0.49	0.6	0.56	0.45	0.51	0.68	0.51	0.56	0.58
FacesUCR	0.68	0.67	0.51	0.66	0.67	0.67	0.46	0.65	0.67	0.69	0.48	0.69	0.65	0.68	0.49
FiftyWords	0.67	0.69	0.63	0.67	0.69	0.69	0.62	0.68	0.68	0.69	0.61	0.67	0.69	0.69	0.61
Fish	0.31	0.43	0.36	0.44	0.42	0.42	0.3	0.54	0.42	0.43	0.29	0.64	0.41	0.42	0.29
Fungi	0.89	0.82	0.89	0.88	0.81	0.82	0.0	0.89	0.81	0.84	0.0	0.9	0.81	0.84	0.0
GunPoint	0.0	0.0	0.0	0.08	0.0	0.0	0.09	0.68	0.01	0.0	0.16	0.81	0.03	0.02	0.37
GunPointAgeSpan	0.0	0.01	0.5	0.0	0.03	0.02	0.36	0.0	0.01	0.0	0.55	0.0	0.03	0.04	0.42
HandMovementDirection	0.02	0.02	0.04	0.02	0.02	0.02	0.04	0.02	0.02	0.02	0.03	0.02	0.02	0.03	0.03
Handwriting	0.44	0.46	0.27	0.41	0.44	0.57	0.28	0.37	0.45	0.56	0.27	0.37	0.44	0.56	0.27
Heartbeat	0.01	0.03	0.0	0.01	0.03	0.0	0.0	0.01	0.02	0.0	0.0	0.01	0.01	0.0	0.0
Herring	0.03	0.01	0.0	0.02	0.01	0.02	0.01	0.01	0.02	0.01	0.01	0.03	0.0	0.03	0.01
Libras	0.63	0.74	0.54	0.62	0.71	0.62	0.53	0.63	0.7	0.62	0.54	0.63	0.7	0.62	0.54
Lightning2	0.11	0.07	0.04	0.11	0.05	0.06	0.04	0.09	0.05	0.06	0.07	0.07	0.06	0.08	0.05
Lightning7	0.54	0.51	0.42	0.56	0.52	0.52	0.44	0.56	0.49	0.51	0.44	0.57	0.5	0.53	0.43
Mallat	0.87	0.88	0.87	0.88	0.87	0.89	0.85	0.88	0.88	0.9	0.83	0.89	0.9	0.91	0.85
Meat	0.76	0.62	0.49	0.67	0.63	0.61	0.54	0.59	0.62	0.64	0.51	0.64	0.61	0.72	0.55
MoteStrain	0.09	0.09	0.49	0.47	0.16	0.03	0.58	0.63	0	0.06	0.68	0.68	0	0.13	0.75
NATOPS	0.65	0.77	0.44	0.64	0.69	0.68	0.45	0.64	0.67	0.64	0.47	0.63	0.69	0.68	0.51

(Continued on next page)

Table C.3 (continued)

	0%			5%				15%				25%			
Dataset	LDPS	KM	DCC	CDPS	CKM	MKM	DCC	CDPS	CKM	MKM	DCC	CDPS	CKM	MKM	DCC
OSULeaf	0.29	0.24	0.2	0.36	0.24	0.24	0.18	0.41	0.24	0.25	0.19	0.43	0.24	0.24	0.19
PenDigits	0.68	0.71	0	0.71	0.68	0.72	0.68	0.73	0.67	0.73	0.82	0.74	0.67	0.73	0
Phoneme	0.32	0.31	0.16	0.29	0	0.31	0.16	0.29	0.31	0.31	0.16	0.29	0.31	0.31	0.15
Plane	0.89	0.91	0.84	0.89	0.88	0.92	0.86	0.9	0.91	0.93	0.85	0.89	0.9	0.93	0.88
PowerCons	0.43	0.08	0.43	0.28	0.08	0.1	0.32	0.43	0.13	0.1	0.42	0.59	0.03	0.13	0.91
RacketSports	0.61	0.62	0.23	0.61	0.6	0.64	0.21	0.62	0.58	0.59	0.23	0.61	0.56	0.61	0.29
Rock	0.26	0.23	0.43	0.25	0.27	0.26	0.0	0.26	0.22	0.24	0.0	0.25	0.2	0.21	0.0
ScreenType	0.02	0.02	0.02	0.02	0.02	0.02	0.03	0.02	0.01	0.02	0.03	0.01	0.02	0.02	0.03
ShapesAll	0.75	0.74	0.69	0.75	0.75	0.74	0.69	0.75	0.74	0.75	0.68	0.76	0	0.74	0.69
StandWalkJump	0.16	0.11	0.05	0.12	0.13	0.12	0.12	0.13	0.14	0.13	0.16	0.15	0.18	0.1	0.14
SwedishLeaf	0.66	0.59	0.6	0.68	0.58	0.59	0.55	0.71	0.58	0.59	0.54	0.73	0.58	0.59	0.54
Symbols	0.79	0.86	0.81	0.85	0.89	0.87	0.82	0.85	0.86	0.88	0.85	0.85	0.91	0.87	0.83
SyntheticControl	0.86	0.9	0.58	0.88	0.88	0.87	0.52	0.86	0.9	0.92	0.53	0.88	0.9	0.94	0.57
UWaveGestureLibrary	0.66	0.64	0.6	0.64	0.66	0.79	0.65	0.65	0.67	0.79	0.64	0.66	0.66	0.82	0.63

Table C.4
Inductive NMI results. KM, CKM, MKM represents K-means, COP-Kmeans, MIP-Kmeans respectively.

	0%			5%				15%				25%			
Dataset	LDPS	KM	DCC	CDPS	CKM	MKM	DCC	CDPS	CKM	MKM	DCC	CDPS	CKM	MKM	DCC
Adiac	0.73	0.58	0.6	0.73	0.58	0.61	0.59	0.73	0.57	0.61	0.6	0.73	0.58	0.61	0.6
ArticularyWordRecognition	0.89	0.91	0	0.89	0.92	0.93	0.89	0.88	0.92	0.91	0.89	0.88	0	0.89	0.88
AtrialFibrillation	0.16	0.02	0	0.17	0.08	0	0.28	0.17	0.1	0	0.25	0.18	0	0	0.3
BME	0.52	0.52	0.3	0.46	0.56	0.48	0.35	0.42	0.53	0.52	0.43	0.47	0.48	0.51	0.42
BasicMotions	0.88	0.56	0	0.87	0.78	0	0.34	0.88	0.8	0	0.29	0.89	0	0	0.35
Beef	0.33	0.31	0.35	0.33	0.29	0.28	0.37	0.36	0.29	0.28	0.39	0.35	0.27	0.3	0.41
BirdChicken	0.08	0.02	0.3	0.09	0.03	0.04	0.22	0.18	0.04	0.05	0.33	0.16	0.04	0.05	0.25
CBF	0.8	0.77	0.48	0.78	0.78	0.77	0.46	0.79	0.77	0.78	0.43	0.8	0.77	0.77	0.53
Car	0.27	0.18	0.26	0.3	0.18	0.19	0.28	0.32	0.19	0.18	0.27	0.36	0.2	0.19	0.28
Coffee	0.28	0.7	0	0.37	0.68	0.61	0.64	0.59	0.73	0.71	0.66	0.77	0.63	0.67	0.64
Cricket	0.88	0	0	0.89	0.91	0	0.62	0.89	0.91	0	0.55	0.88	0	0	0.6
CricketX	0.37	0.49	0.26	0.37	0.48	0.48	0.27	0.36	0.48	0.48	0.27	0.39	0.48	0.48	0.27
CricketY	0.46	0.47	0.29	0.44	0.47	0.47	0.28	0.44	0.45	0.47	0.29	0.43	0.49	0.46	0.28
CricketZ	0.37	0.47	0.26	0.37	0.48	0.49	0.26	0.38	0.49	0.47	0.26	0.39	0.49	0.49	0.26
ECG200	0.02	0.01	0.2	0.06	0.02	0.02	0.27	0.16	0.03	0.03	0.34	0.17	0.04	0.04	0.43
Epilepsy	0.62	0.31	0	0.59	0.18	0	0.07	0.61	0.19	0	0.08	0.61	0	0	0.07
EthanolConcentration	0.01	0	0	0.01	0.04	0.01	0.01	0.01	0.02	0.01	0.01	0.01	0	0.01	0.01

(Continued on next page)

Table C.4 (continued)

	0%			5%				15%				25%			
Dataset	LDPS	KM	DCC	CDPS	CKM	MKM	DCC	CDPS	CKM	MKM	DCC	CDPS	CKM	MKM	DCC
FaceAll	0.66	0.66	0.44	0.65	0.66	0.68	0.42	0.65	0.67	0.66	0.43	0.66	0.66	0.68	0.42
FaceFour	0.6	0.55	0	0.59	0.55	0.51	0.51	0.6	0.54	0.49	0.54	0.58	0.4	0.51	0.51
FacesUCR	0.62	0.67	0.54	0.58	0.67	0.68	0.53	0.61	0.66	0.68	0.54	0.61	0.66	0.68	0.54
FiftyWords	0.7	0.69	0.67	0.7	0.69	0.69	0.67	0.7	0.69	0.69	0.66	0.7	0.69	0.69	0.66
Fish	0.42	0.43	0.36	0.46	0.41	0.42	0.35	0.51	0.43	0.42	0.35	0.56	0.42	0.41	0.34
Fungi	0.85	0.82	1.0	0.85	0.79	0.83	0.0	0.85	0.8	0.85	0.0	0.85	0.8	0.84	0.0
GunPoint	0.0	0.0	0.03	0.02	0.0	0.0	0.13	0.06	0.0	0.0	0.19	0.17	0.0	0.0	0.12
GunPointAgeSpan	0.0	0.01	0.52	0.0	0.0	0.03	0.43	0.0	0.03	0.0	0.43	0.0	0.0	0.01	0.46
HandMovementDirection	0.04	0.02	0	0.05	0.02	0	0.05	0.05	0.02	0	0.05	0.05	0	0	0.06
Handwriting	0.38	0.43	0	0.37	0.45	0.56	0.55	0.37	0.45	0.56	0.55	0.37	0	0.55	0.54
Heartbeat	0.01	0.03	0	0.01	0.04	0.0	0.0	0.01	0.03	0.0	0.0	0.01	0	0.0	0.0
Herring	0.06	0.01	0.02	0.04	0.01	0.02	0.02	0.03	0.01	0.03	0.02	0.03	0.03	0.02	0.01
Libras	0.65	0.72	0	0.65	0.71	0.63	0.55	0.64	0.71	0.61	0.54	0.64	0	0.58	0.55
Lightning2	0.19	0.07	0.16	0.17	0.08	0	0.04	0.13	0.05	0	0.03	0.09	0.06	0	0.09
Lightning7	0.6	0.51	0.44	0.58	0.5	0	0.46	0.58	0.51	0	0.48	0.58	0.52	0	0.47
Mallat	0.89	0.88	0.97	0.87	0	0.89	0.93	0.86	0	0.89	0.93	0.87	0	0.89	0.93
Meat	0.74	0.62	0.52	0.73	0	0.65	0.63	0.64	0	0.64	0.52	0.62	0	0.66	0.66
MoteStrain	0.07	0.09	0.28	0.04	0	0.01	0.24	0.14	0	0.02	0.26	0.26	0	0.01	0.32
NATOPS	0.64	0.73	0	0.64	0.69	0.64	0.44	0.64	0.69	0.64	0.46	0.63	0	0.6	0.51

(Continued on next page)

Table C.4 (continued)

	0%			5%				15%				25%			
Dataset	LDPS	KM	DCC	CDPS	CKM	MKM	DCC	CDPS	CKM	MKM	DCC	CDPS	CKM	MKM	DCC
OSULeaf	0.27	0.24	0.24	0.32	0	0.24	0.23	0.34	0	0.25	0.24	0.37	0	0.25	0.25
PenDigits	0	0.68	0	0	0.68	0.74	0.7	0	0.67	0.72	0.82	0	0	0.72	0
Phoneme	0.24	0.31	0.52	0.23	0	0.31	0.51	0.23	0	0.31	0.51	0.23	0	0.31	0.51
Plane	0.87	0.91	0.82	0.88	0	0.91	0.84	0.89	0	0.94	0.85	0.89	0	0.9	0.86
PowerCons	0.32	0.08	0.48	0.28	0	0.08	0.31	0.34	0	0.1	0.29	0.43	0	0.1	0.36
RacketSports	0.48	0.59	0	0.51	0.59	0.59	0.29	0.49	0.57	0.47	0.27	0.53	0	0.44	0.31
Rock	0.31	0.23	0.24	0.33	0	0.25	0.0	0.31	0	0.27	0.0	0.32	0	0.26	0.0
ScreenType	0.02	0.02	0.04	0.02	0	0.02	0.05	0.02	0	0.02	0.04	0.01	0	0.02	0.05
ShapesAll	0.76	0.74	0.72	0.76	0	0.74	0.73	0.77	0	0.74	0.73	0.77	0	0.75	0.73
StandWalkJump	0.21	0.19	0	0.18	0.13	0	0.19	0.17	0.12	0	0.26	0.17	0	0	0.23
SwedishLeaf	0.67	0.59	0.59	0.69	0	0.59	0.54	0.71	0	0.59	0.52	0.72	0	0.59	0.52
Symbols	0.83	0.86	0.78	0.83	0	0.85	0.8	0.84	0	0.85	0.8	0.84	0	0.87	0.81
SyntheticControl	0.82	0.9	0.54	0.86	0	0.9	0.55	0.87	0	0.89	0.54	0.87	0	0.87	0.58
UWaveGestureLibrary	0.61	0.71	0	0.64	0.67	0.78	0.71	0.63	0.66	0.76	0.71	0.64	0	0.76	0.69

Summary

Time-series clustering is a challenging task that requires tailored similarity measures to effectively analyze and interpret data. Moreover, the algorithm outputs can be hard to interpret. To address this, experts can provide prior knowledge using instance-level constraints, which guide the algorithm and align results with the expert's needs. However, incorporating this in the clustering process introduces the challenge of assessing the informativeness and coherence of the constraints. Furthermore, integrating explanation techniques with the constrained clustering paradigm remains an open issue. In this work, we first develop a constraint-based representation for time series that leverages elastic measure properties, specifically Dynamic Time Warping (DTW), using shapelets and the shapelet transform. Our approach aims to learn a metric space where constraint properties can be calculated, and the distance between objects approximates the DTW (Dynamic Time Warping) similarity measure. Second, we develop different approaches to provide explanations for time-series clustering results by using shapelets representations. As such, the new representation enhances clustering performance while facilitating an explanation of the results.

Keywords: Learning Shapelets, constrained clustering, Time Series, explaining clustering, time series transformation

Résumé

Le regroupement de séries temporelles est une tâche complexe qui nécessite des mesures de similarité sur mesure pour analyser et interpréter efficacement les données. De plus, les sorties des algorithmes peuvent être difficiles à interpréter. Pour remédier à cela, les experts peuvent fournir des connaissances préalables en utilisant des contraintes au niveau des instances, qui guident l'algorithme et alignent les résultats sur les besoins de l'expert. Cependant, incorporer cela dans le processus de regroupement introduit le défi d'évaluer l'informativité et la cohérence des contraintes. De plus, l'intégration de techniques d'explication avec le paradigme de regroupement contraint reste une question ouverte. Dans ce travail, nous développons d'abord une représentation basée sur des contraintes pour les séries temporelles qui exploite les propriétés de mesure élastique, en particulier la Dynamic Time Warping (DTW), en utilisant des shapelets et la transformation des shapelets. Notre approche vise à apprendre un espace métrique où les propriétés des contraintes peuvent être calculées, et la distance entre les objets approxime la mesure de similarité DTW (Dynamic Time Warping). Deuxièmement, nous développons différentes approches pour fournir des explications pour les résultats du regroupement de séries temporelles en utilisant des représentations de shapelets. Ainsi, la nouvelle représentation améliore les performances du regroupement tout en facilitant l'explication des résultats.

Keywords: Learning Shapelets, clustering sur contraint, séries temporelles, explication du clustering, transformation de séries temporelles.