THÈSE DE DOCTORAT DE

L'UNIVERSITÉ DE RENNES 1
COMUE UNIVERSITE BRETAGNE LOIRE

ÉCOLE DOCTORALE N° 601
Mathèmatique et Sciences et Technologies de l'Information et de la Communication
Spécialité : Informatique

Par
Ricardo CARLINI SPERANDIO

TIME SERIES RETRIEVAL USING DTW-PRESERVING SHAPELETS

Thèse présentée et soutenue à Rennes le 18 décembre 2019
Unité de recherche : IRISA — UMR6074
Thèse N° : 00000

"*Moving on, is a simple thing, what it leaves behind is hard*".

Dave Mustaine *À Tout le Monde*

à Julia et Fernanda

Acknowledgements

The present document, results of four years of research, would not have been possible without the help, encouragements, and support of many persons. I thank all of them, and I present them all my gratitude.

I would like to thank Dr. Laurent Amsaleg and Dr. Guillaume Gravier, who directed my thesis. Laurent and Guillaume, thanks for providing me an opportunity to work on this exciting field, all their encouragement, considerations, and involvement at all times. The discussions, the insights, and all the rest of their support. Laurent, thank you for all the advice, encouragements, and tips. *Aucun mot ne peut exprimer ma gratitude pour ce que vous avez fait, je vous remercie énormément.*

I would like to thank Dr. Simon Malinowski for his contribution and support. Simon, your advises, and your previous work on time series and shapelets were of paramount importance in this thesis. Without his contribution, the achievement of this thesis would have been much more difficult. *Je te dois une grande bière, Simon !*

I would also like to extend special thanks to Dr. Silvio Guimarães and his team over the PUC/Minas. His friendship, suggestions, and advice were immeasurable. His participation in the jury of this thesis and his review of the text are highly appreciated too. *Silvio, muito obrigado por tudo, você foi muito importante nestes anos.*

I would like to thank Dr. Romain Tavernard for his collaboration and, especially, for the package TSLearn; with it, he helped me a lot, *merci.*

I would like to extend my gratitude to Professor Dr. Alexandre Termier for honoring my jury by presiding it. I also appreciate the participation of Professor Dr. Usue Mori (Université du Pays Basque), Professor Dr. Germain Forestier (Université de Haute-Alsace) and Professor Dr. Florent Masseglia (Université Montpellier 2), I appreciate the challenge they took in accepting to read my work. Thank you.

I would like to thanks Inria, IRISA and Université Rennes 1 for sheltering me during those years. Of course, I could not fail to thank Aurélie Patier, *merci beaucoup Aurélie !*

My sincere thanks to CNPq *C*onselho Nacional de Desenvolvimento Científico e Tecnolóógico (Process number 233209/2014–0) for providing the necessary funds to develop this work.

Undoubtedly, I would like to thank my family, in particular, my wife Julia, for her never-ending support. Julia, I never dreamed that I'd meet somebody like you! And also to my little daughter Fernanda, who came to the world during this thesis and brought a new sense to my life. *Fernanda, papai espera ter mais tempo para ti desde já. Amo vocês!*

Finally, I would like to thank all the guys from the Linkmedia team, you guys have been a source of laughter and fun, and have helped me immensely and, I cannot thank you all enough.

Résumé Étendú

Il vaut mieux faire que dire.

Alfred de Musset

Contents

Contexte

La recherche scientifique traitant de l'analyse et de l'exploitation de séries temporelles est très active, notamment en raison de nombreuses applications telles que la surveillance des électrocardiogrammes, la reconnaissance vocale ou la modélisation environnementale. Cette thèse s'inscrit dans cette veine et propose une nouvelle approche pour l'identification de séries temporelles similaires à une requête. Cette tâche est difficile pour plusieurs raisons.

D'abord, la fouille de données temporelles soulève de nombreux défis en raison de la spécificité des données manipulées. Les séries temporelles proviennent de mesures réelles et des séries temporelles ne différant que par des distorsions sur l'axe du temps ou par l'amplitude des valeurs enregistrées doivent être jugées similaires.

Ce résumé introduit les différentes problématiques abordées dans cette thèse. Dans un premier temps, nous développons un court argumentaire afin de montrer l'intérêt de la recherche de séries temporelles. Puis nous développons les travaux réalisés dans le cadre du doctorat. Et enfin, nous présentons les résultats et les contributions obtenues.

Recherche en données temporelles

Une série temporelle, ou série chronologique, est une suite finie de valeurs numériques représentant l'évolution d'une quantité spécifique au cours du temps. Les données de séries chronologiques sont produites massivement 24×7 par des millions d'utilisateurs, dans le monde entier, dans des domaines tels que la finance, l'agronomie, la santé, la surveillance de la terre, les prévisions météorologiques, le multimédia, etc. En raison des progrès de la technologie des capteurs et de leur prolifération, les applications peuvent produire des millions à des milliards de séries chronologiques par jour, ce qui rend encore plus difficile l'extraction des données des séries chronologiques.

La fouille de données temporelles soulève de nombreux défis en raison de la spécificité des données des séries chronologiques. Les séries temporelles proviennent de mesures réelles et des séries temporelles ne différant que par des distorsions sur l'axe du temps ou par l'amplitude des valeurs enregistrées doivent être jugées similaires. De nombreux facteurs peuvent produire ces distorsions, comme un capteur mal calibré ou imprécis, l'utilisation de différentes unités de mesure, ou le bruit, par exemple [EA12].

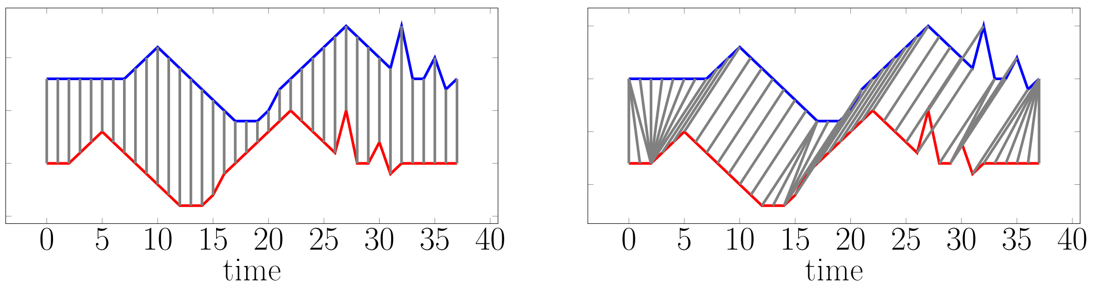
Cette thèse se concentre sur l'étude des mécanismes permettant de réaliser très efficacement des recherches de séries temporelles par similarité. Le scénario pour cette tâche est le suivant : soit τ une série temporelle de test (requête), la recherche consiste à identifier la série temporelle T^* parmi un ensemble de séries $\mathcal{T}$ appartenant à une base de données qui se trouve être l'élément plus proche de τ selon une mesure de la similarité D. En d'autres termes, il s'agit de trouver T^*, de telle sorte que :

$$T^* = \arg\min_{T_i \in \mathcal{T}} D(\tau, T_i) \tag{1}$$

Une méthode traditionnelle d'identification de T^* repose sur le calcul par force brute de D entre τ et toutes les séries de $\mathcal{T}$. Cette approche ne passe pas à l'échelle lorsqu'il s'agit de séries temporelles longues ou de bases de données énormes car calculer D est trop coûteux. Par conséquent, des méthodes doivent être employées pour réduire le coût de l'établissement de la similarité, comme s'appuyer sur l'indexation ou sur une approximation heuristique.

Initialement, comme dans d'autres domaines, la fonction D choisie pour mesurer la similarité des séries temporelles était une norme $\mathrm{L_p}$, comme la distance euclidienne (Euclidean distance (ED)). La mesure ED, en plus d'être rapide, est métrique, donc propice à l'indexation[1]. Toutefois, malgré ces avantages, la distance euclidienne ne donne pas de bons résultats avec les séries chronologiques car elle n'est pas robuste face aux distorsions sur l'axe du temps. Aussi, il est préférable de l'utiliser avec des données transformées ou d'utiliser une autre mesure plus robuste aux distorsions temporelles. Par conséquent, de nombreuses mesures de similarité et différents types de transformations de données ont été proposés pour les séries chronologiques, souvent destinés à résoudre des problèmes spécifiques à un domaine.

Parmi ces fonctions de distance proposées, une se distingue : l'alignement dynamique temporel, ou *Dynamic Time Warping (DTW) [SC78]*. La mesure DTW a la capacité de traiter les distorsions locales dans l'axe du temps, permettant un meilleur alignement entre les points des deux séquences, comme illustré par la Figure 1.



(a) $\mathrm{D}_{ED}(Q, R) = 19.3907$ (b) $\mathrm{D}_{DTW}(Q, R) = 0.0$

FIGURE 1 – (a) En utilisant l'ED, les deux séries temporelles ont une grande valeur de dissimilitude, contrairement à notre perception. (b) Un résultat plus intuitif est obtenu en utilisant DTW comme métrique pour la fonction de distance D, en raison de sa capacité à gérer les distorsions locales de l'axe des temps.

Cependant, la flexibilité de la DTW s'accompagne d'un coût de calcul quadratique [DTS+08].

Réduire la complexité de la DTW constitue l'objectif de nombreux chercheurs mettant au point de nombreuses optimisations élégantes, inventant des bornes inférieures de diverses nature ou concevant d'autres mécanismes [EA12]. Mais la quête de la meilleure performance dans le traitement de collections de séries temporelles extrêmement volumineuses est toujours active.

[1]Les distances métriques permettent d'utiliser l'inégalité triangulaire pour réduire le nombre de calculs dans les systèmes d'indexation.

D'autre part, une autre façon d'explorer le problème consiste à transformer l'ensemble de données et de lui donner une nouvelle représentation, en se concentrant uniquement sur les caractéristiques spécifiques de l'ensemble de données, réduisant ainsi sa complexité. Cette simplification permet l'utilisation de fonctions de distance plus simples et donc plus rapides.

La grande majorité de ces transformations exploite les caractéristiques globales de la série, telles que la transformation de Fourier discrète, ou *Discrete Fourier Transformation (DFT) [FRM94 ; AFS93]*, pour citer un exemple. Cependant, la similitude basée sur la forme n'est pas toujours globale. Par exemple, considérons un électrocardiogramme (ECG) pour un patient dont l'arythmie à un seul battement est indicative d'un problème cardiaque. Si ceci était capturé comme une série temporelle et comparé à une série de comportements normaux, il serait difficile de détecter une différence en raison de la présence de nombreux battements cardiaques réguliers. La caractéristique discriminatoire, dans ce cas, serait décrite par la présence d'une petite forme locale dans la série indiquant un battement irrégulier, qui serait probablement manquée dans les domaines de la fréquence et du temps car la structure et la forme globale des données seraient encore très similaires.

Nous envisageons plutôt d'extraire de petites sous-séquences représentatives des séries chronologiques pour détecter les similitudes locales basées sur la forme entre les séries. Ces petites sous-séquences représentatives, connues sous le nom de *shapelets*, constituent un domaine important de la recherche en fouille de données dans les séries chronologiques depuis sa proposition dans [YK09], et de nombreuses approches visent à étendre son application, pour accélérer son extraction ou même de redéfinir comment l'obtenir.

Parmi ces approches, une se distingue : la transformation à base de *shapelets* (*Shapelets transform (ST)* [BDHL12]). La ST utilise des *shapelets* pour créer une représentation vectorielle des séries temporelles, ce qui permet leur utilisation dans des algorithmes traditionnellement utilisés pour les données vectorielles. Toutefois, en général, les transformations à base de *shapelets* proposées dans la littérature sont des méthodes supervisées. Elles sont donc peu pratiques pour la tâche de recherche de séries chronologiques similaires, par nature non supervisée. La seule exception est l'approche d'apprentissage des shapelets préservant de la DTW (*Learning DTW-preserving shapelets (LDPS)*) [LMTA17].

La LDPS est la première approche de transformation de *shapelet* basée sur un apprentissage non supervisé. L'objectif n'est pas d'essayer d'apprendre des *shapelets* pour classes les plus discriminantes, mais plutôt d'apprendre des shapelets qui préservent au mieux la vraie mesure DTW dans le nouveau espace de plongement.

Dans cette thèse, nous proposons d'utiliser la LDPS comme base d'une système de recherche par similarité de séries temporelles (*Time Series Retrieval (TSR)*).

Objectif et vue d'ensemble de la thèse

C'est l'identification *rapide* de la ou ses séries les plus similaires à une série requête qui est sans aucun doute l'un des plus grands, sinon le plus grand, défi dans la recherche par similarité des séries chronologiques.

Dans cette thèse, nous essayons d'élucider une question : est-il possible de combiner la robustesse aux distorsions selon l'axe temporel de la mesure DTW avec la vitesse de calcul de la mesure ED ? C'est pourquoi nous proposons une transformation qui réunit le meilleur des deux univers : la robustesse aux distorsions locales selon l'axe des temps, offert par la DTW, et la vitesse découlant de l'utilisation de la mesure Euclidienne.

Nous proposons ici une approche approximative pour la recherche de séries temporelles similaires fondée sur un changement de représentation au travers d'un processus de plongement. L'idée est de proposer une façon de transformer les séries chronologiques en représentations vectorielles. Cette transformation est produite de manière à permettre qu'une recherche euclidienne puisse ensuite être appliquée efficacement entre les séries transformées afin de trouver le voisin le plus proche de la requête transformée que l'on veut identique à celui qui aurait pu être déterminé si l'on avait utilisé la mesure DTW. Naturellement, la transformation doit être soigneusement conçue pour que la recherche approximative soit précise. Un autre point crucial est lié au coût de calcul de la transformation. Au moment du test, la requête doit d'abord être transformée avant d'être comparée aux séries temporelles transformées de l'ensemble de données. La transformation ne doit donc pas être trop coûteuse.

Nous commençons par proposer l'utilisation de la LDPS pour la tâche de recherche par similarité, proposition nouvelle. Cette transformation est basée sur un apprentissage non supervisé telle que la distance euclidienne relative dans l'espace transformé reflète bien les mesures originales obtenues en utilisant la DTW.

Hors ligne, l'ensemble $\mathcal{S}$ avec d *shapelets* est appris de $\mathcal{T}$, comme décrit dans [LMTA17]. Toutes les séries temporelles dans $\mathcal{T}$ sont ensuite transformées et forment $\overline{\mathcal{T}}$, qui est stocké dans une base de données.

En ligne, une série temporelle requête T_q est transformée en $\overline{T}_q$ en utilisant $\mathcal{S}$. Les k voisins les plus proches de $\overline{T}_q$ sont ensuite recherchés dans $\overline{\mathcal{T}}$ et conservés dans une liste temporaire de séries temporelles classées en fonction de leur proximité avec $\overline{T}_q$.

Le résultat final peut ensuite être construit selon deux options : (a) la série temporelle brute associée au plus proche voisin trouvé dans l'espace transformé est considérée comme le plus proche voisin de la requête brute, ou (b) la vraie DTW est ensuite calculée entre la requête brute et les versions brutes des k séries temporelles transformées et identifiée, puis la série la plus proche selon la DTW est retournée.

Ce plongement préservant la DTW est tel que le classement dans l'espace transformé est une approximation du classement qui serait produit dans l'espace d'origine conformément à la mesure DTW. Cependant, ce classement basé sur L_2 est obtenu beaucoup plus rapidement car les distances euclidiennes sont moins coûteuses à calculer que les mesures DTW.

Nous savons qu'une transformation d'une série chronologique a un coût, c'est pour cela que dans la tâche de la recherche de séries similaires un compromis entre la qualité de la réponse et le temps de transformation doit être atteint. Dans le cas de la LDPS, le coût de transformation est lié au nombre de *shapelets* utilisées lors de la transformation. Par conséquent, notre objectif est d'obtenir une transformation qui utilise le moins de *shapelets* possible tout en préservant au maximum l'acuité de la réponse.

Ainsi, au lieu d'utiliser $\mathcal{S}$ comme spécifié par [LMTA17], il peut être préférable de conserver et d'utiliser pour la transformation uniquement un sous-ensemble de *shapelets* constitué d'éléments soigneusement sélectionnés.

Pour appliquer l'esprit des algorithmes de sélection de caractéristiques au cas d'une transformation de *shapelets*, il est nécessaire de définir : (a) une métrique afin d'évaluer la qualité respective de chaque sous-ensemble de *shapelets*, (b) une stratégie pour construire des sous-ensembles de *shapelets* de plus en plus importants, (c) un critère d'arrêt interrompant la recherche de sous-ensembles plus grands.

Nous détaillons maintenant ces trois points.

Métrique d'évaluation pour comparer les sous-ensembles de *shapelet* et pour le choix des sous-ensembles

Nous commençons par construire $\mathcal{S}$ comme spécifié dans [LMTA17]. Il n'est pas utilisé pour transformer des séries temporelles dans $\mathcal{T}$ mais plutôt comme un réservoir dans lequel seront ensuite piochées les *shapelets* les plus utiles.

Pour comparer les performances de différents sous-ensembles de *shapelet*, nous avons besoin d'une vérité terrain basée sur la vraie distance DTW entre les séries chronologiques. Pour la construire, la DTW entre toutes les paires de séries temporelles dans l'ensemble d'apprentissage de $\mathcal{T}$ est calculée et nous enregistrons pour chaque série temporelle l'identifiant de son plus proche voisin. Cet ensemble d'entraînement est ensuite divisé en 10 parties, dont 9 sont utilisées pour les tâches d'entraînement détaillées ci-dessous, l'une étant utilisée pour la validation, dans un contexte classique de validation croisée.

Ensuite, nous sélectionnons d' *shapelets* dans $\mathcal{S}$ et utilisons ces d' *shapelets* pour transformer toutes les séries temporelles appartenant à ensemble d'apprentissage courant. En utilisant les mêmes d' *shapelets*, nous transformons également chaque série temporelle de l'échantillon de validation et nous les utilisons en tant que requêtes.

Les séries temporelles transformées à partir de l'échantillon d'entraînement sont ensuite classées avec leur distance L_2 à chaque requête. Il est donc possible de déterminer à quel rang correspond la série chronologique vraie la plus proche pour cette requête. Nous répétons cette opération pour toutes les séries temporelles de validation et pour tous les échantillons. Cela revient à construire un histogramme des rangs où le vrai voisin le plus proche basé sur DTW apparaît.

Nous utilisons cet histogramme pour construire une fonction de distribution cumulative, puis nous calculons l'aire associée sous la courbe de cette fonction (*area under the curve (AUC)*). Nous considérons cette valeur de l'AUC comme la mesure de la performance permettant d'évaluer la qualité d'un sous-ensemble de *shapelet.* Plus cette AUC est élevée, meilleur est le sous-ensemble de *shapelet.* Cette métrique est bien adaptée à la tâche d'extraction du plus proche voisin car elle favorise le classement élevé du voisin le plus proche vrai dans la liste approchée.

La métrique étant définie, nous utilisons ensuite un algorithme glouton pour choisir les *shapelets*, en commençant par une liste vide. Nous y ajoutons la meilleure *shapelet*, puis la meilleure paire étant donné le choix fait plus tôt, etc.

Nous définissons quatre critères d'arrêt différents, qui déterminent quand arrêter d'ajouter des shapelets à l'ensemble actuel des shapelets sélectionnées, allant d'un choix minimal de shapelets à l'utilisation de toutes :

- $\mathrm{DPSR_g}$: Les *shapelets* sont ajoutées une à une jusqu'à ce qu'il ne reste plus de *shapelets.* À la fin, le sous-ensemble qui conduit à la meilleure AUC globale est sélectionné.
- $\mathrm{DPSR_t}$: Nous calculons le gradient normalisé entre l'AUC du sous-ensemble actuellement sélectionné et celle obtenue en ajoutant la *shapelet* qui améliore le mieux l'AUC. Si cette gradient est inférieure à 1, la sélection de *shapelet* est arrêtée.
- $\mathrm{DPSR_l}$: La sélection de shapelet est arrêtée dès que l'ajout d'une *shapelet* n'améliore pas la valeur de l'AUC.
- $\mathrm{DPSR_f}$: utilise toutes les *shapelets.*

La Figure 2 affiche les valeurs de l'AUC à chaque itération de l'algorithme de sélection de *shapelet* sur l'ensemble de données `Ham` (à partir de l'archive UCR-UEA [CKH+15]).

Dans la littérature sur la sélection des caractéristiques, la méthode de sélection décrite ci-dessus peut être classée dans la classe *wrapper*, où un algorithme d'apprentissage est appliqué pour évaluer la qualité respective de différents sous-ensembles de caractéristiques, de manière itérative. Cette approche est la plus courante et, malgré son amélioration par rapport à la recherche exhaustive, elle reste un processus très coûteux pour l'analyse de nombreuses caractéristiques. Par conséquent, le filtrage précoce des caractéristiques non pertinentes ou redondantes améliorerait la vitesse de l'algorithme de sélection de caractéristiques puisqu'il y en aurait moins à analyser. Ceci est illustré par la Figure 3.

Les filtres sont généralement peu coûteux et rapides, et appliquent souvent une analyse statistique de base sur l'ensemble des caractéristiques. Le filtre proposé ici repose sur la construction

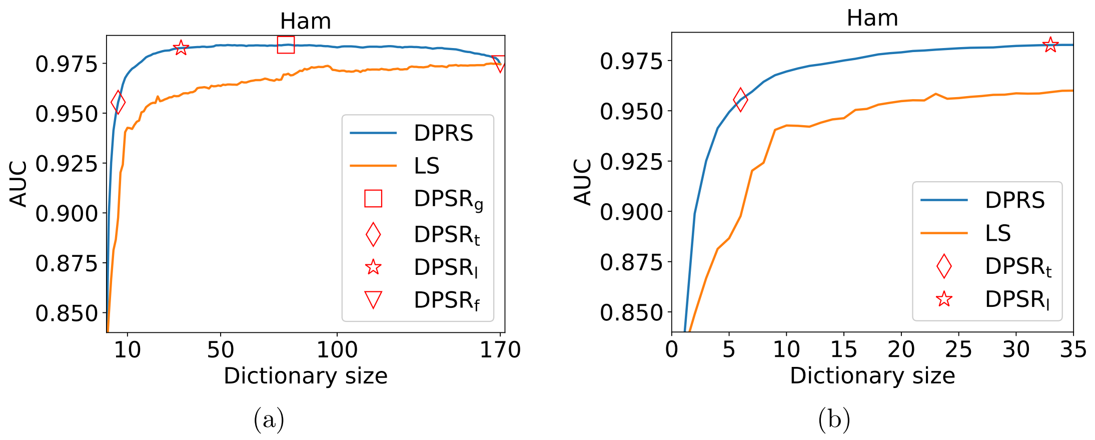


Figure 2 – (a) : Comparaison des valeurs de l'AUC pour la sélection des caractéristiques basée sur le DPSR et le score laplacien.(b) : Zoom sur (a)sur les 35 premiers éléments du dictionnaire.

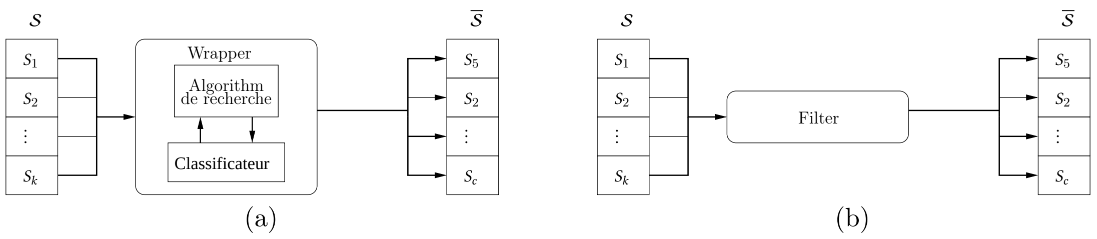


Figure 3 – (a) La procédure du modèle *wrapper*. (b) La procédure du modèle *filter*.

d'un graphe de *shapelets* basé sur la corrélation de Pearson, puis trouve des cliques dans ce graphe. Une seule *shapelet* par clique est conservée, les autres étant filtrées, comme indiqué dans la Figure 4.

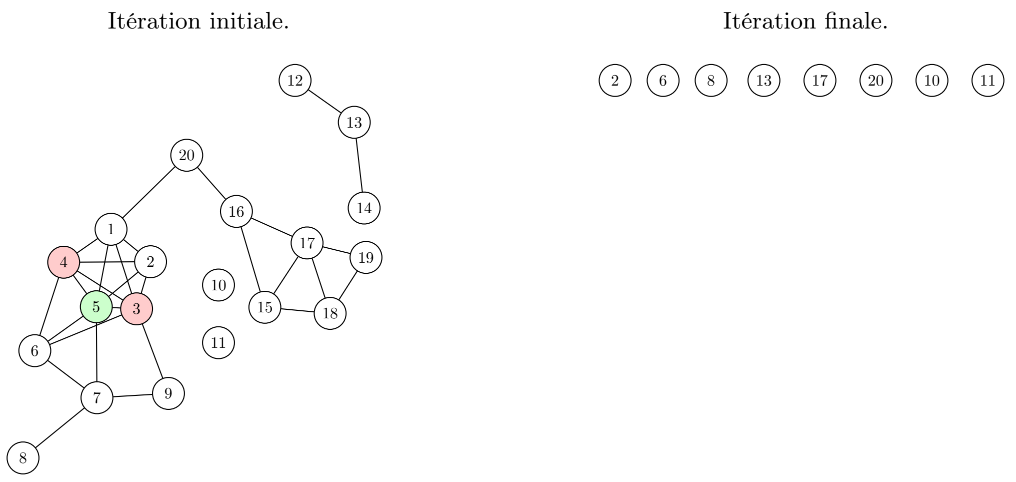


Figure 4 – Nous commençons par un graphe fragmenté et finissons par un graphe totalement déconnecté.

Les algorithmes DPSR et de filtrage décrit ci-dessus peuvent être étendus pour fonctionner avec des séries chronologiques à plusieurs variables. Pour étendre le DPSR (et le LDPS) aux séries multidimensionnelles, trois approches ont été proposées :

(i) M-DPSR$_\text{i}$: independent multivariate DPSR : La première variante consiste à appliquer M fois l'algorithme de [LMTA17] pour apprendre M dictionnaires indépendants, un par dimension de la série chronologique multivariée à transformer.

(ii) M-DPRS$_\text{d}$: dependent multivariate DPSR : La seconde variante apprend un ensemble de *shapelets multidimensionnels.*

(iii) M-DPSR$_\text{s}$: slack multivariate DPSR : La troisième variante apprend un ensemble de *shapelets multidimensionnels*, comme avec M-DPSR$_\text{d}$. Mais c'est la transformation qui diffère : La fenêtre coulissante du *shapelet* n'est pas rigide entre les dimensions.

Des expérimentations démontrent les bonnes performances de nos propositions, des tests statistiques ont montré que DPSR est significativement meilleur que PAA et LB_Keogh.

Résumé des principales contributions

Dans ce manuscrit, nous avons fait quatre contributions importantes :

(i) il explique comment les *shapelets* préservant de la DTW peuvent être utilisées dans le contexte spécifique de la récupération des séries temporelles

(ii) il propose quelques stratégies de sélection de *shapelets* pour faire face à l'échelle, c'est-à-dire pour faire face à une collection massive de séries temporelles ;

(iii) il présente un nouveau filtre multidimensionnel pour la sélection non supervisée de caractéristiques ;

(iv) il explique en détail comment traiter les séries chronologiques univariées et multivariées, couvrant ainsi tout le spectre des problèmes de recherche de séries chronologiques.

Le coeur de la contribution présentée dans ce manuscrit nous permet d'arbitrer facilement entre la complexité de la transformation et la précision de l'extraction.

Des expérimentations à grande échelle ont été menées à l'aide des archives de classification des séries chronologiques UCR [CKH+15] et des plus récentes archives de classification des séries chronologiques multivariées UEA [BDL+18], pour appuyer cette thèse et démontrer les importantes améliorations de performance par rapport aux techniques de pointe.

Pistes pour des travaux futurs

Dans cette thèse, nous avons montré que les *shapelets* pourraient aider à représenter des données pour la recherche, en particulier après le choix d'un sous-ensemble approprié. La prochaine étape directe consiste à s'intégrer à un système d'indexation, à explorer l'inégalité triangulaire, évitant ainsi les calculs exhaustifs de la distance euclidienne et à améliorer davantage les performances. Une telle approche peut être avantageusement utilisée pour l'indexation à tout moment de séries temporelles.

Notre approche impliquant des *shapelets* préservant de la DTW montre des résultats très prometteurs, tant pour les tâches univariées que multivariées. Cependant, une question ouverte est d'améliorer l'apprentissage des *shapelets* originellement décrit par Lods et al. dans [LMTA17]. Cette approche privilégie la prise en compte de distances moyennes, sans s'attacher à préserver plutôt les distances aux k premiers éléments, ce qui est plus pertinent pour la recherche des séries temporelles similaires à une requête. Nous entrevoyons deux adaptations de l'algorithme LDPS. L'une consisterait à ajouter un mécanisme permettant de choisir les top-k *shapelets* au cours de la phase d'apprentissage. L'autre consisterait à modifier la fonction de perte afin qu'elle soit minimisée pour ne pas conserver les distances moyennes DTW mais plutôt les k premiers rangs.

Summary

Chapter 1

Introduction

Begin at the beginning, the King said gravely, "and go on till you come to the end: then stop."

Lewis Carroll, *Alice in Wonderland*

Contents

From the earliest days, even before the development of the writing, the humans had already started to store knowledge (and data) by painting or sculpting in rocks and caves[1]. Now, after centuries of innovations, improvements in technology and gains in knowledge, humanity has reached a point where we can create 2.5 Quintilian bytes of data every day [Clo17], and there is nothing to indicate that this production will slow down.

(a)

(b)

Figure 1.1 – Examples of human storing data: (a) The oldest known human preserved data: A figurative painting dated as over 40,000 years old, "stored" into the Lubang Jeriji Saléh cave. It is believed to be the representation of decapitation of a 1.5-meter-banteng bull [ASO+18]. (b) One of the dozens of Facebook's data-centers, this in Sweden. On those servers, even the tracking of the Facebook users' mouse movement is stored.

Storing large amounts of data brings some challenges concerning hardware boundaries, where limitations are mainly encountered in issues such as cost, capacity, and bandwidth limits. However, our biggest challenge lies in how to handle automatically and extract knowledge from this large volume of data. If for many years the manual extraction of patterns from collections of

[1]Parietal art.

the dataset was sufficient, since the *Information Age*[2], due to the growth of dataset's size and complexity, this process needs to be performed by computers.

The process of discovering patterns in large data sets is named Data mining (DM), and it involves methods at the intersection of statistics, Machine learning (ML), and database systems [KDD06]. Algorithms for Data mining can be classified into three main groups:

Supervised algorithms: in this approach, the algorithm learns on a labeled (training) dataset. The supervised algorithm, using this provided training dataset, tries to model relationships and dependencies between the input data and the target prediction output (ground truth). Such that we can predict the output values for new data based on those relationships which it learned from the training dataset. The mass of practical machine learning uses supervised learning.

Unsupervised algorithms: The goal is to model the underlying structure or the distribution in the data, concerning to learn more about the data. The dataset is provided without explicit instructions on what to do with it. As there is no ground truth element to the data, unsupervised learning is a task more difficult than supervised learning. Once, it is hard to measure the accuracy of an algorithm trained without the aid of labels. However, for many situations, acquiring labeled data is prohibitive, or even we do not know *a priori* what kind of pattern we are looking for on the data.

Semi-supervised algorithms: It is a hybrid approach between supervised and unsupervised approaches. It uses a training dataset with both labeled and unlabeled data. This kind of method is particularly useful when extracting relevant features from the dataset is difficult, and the labeling task is costly and time-intensive for experts.

Approaches designed for data mining, need not only to be effective but also efficient, as we need to deal between an acceptable time response and an adequate quality of the returned answer. Besides that, in data mining tasks, we are exposed to scale-related issues. Algorithms for data mining need to present excellent performance with minimal impact according to the size of the dataset or the complexity of the analyzed data, i. e. the data's dimensionality. The complexity of the data imposes a new question: how to compare these complex data? The typical approach is to use distance functions to measure how similar (or dissimilar) are two objects. Nevertheless, the algorithms' performance is directly influenced by the dimensionality of the data and the number of objects to compare. The problem is when increasing the dimensionality of the data (its complexity), the volume of the space where this data is represented increases so fast that the available data becomes sparse and the measure distance becomes meaningless. In high-dimensional data, there is little difference in the distances between different pairs of objects. Those phenomena are called the *curse of dimensionality* [Bel03].

Some possible ways to handle the *curse of dimensionality* and the scale of the dataset are: (i) the use of dimensionality reduction techniques; (ii) the use of data transformation; (iii) and the acceleration of massive distance computation when comparing sets of objects by the use of lower-bounding functions and data structures.

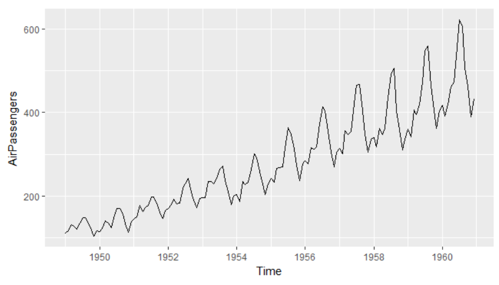
Conventional approaches for dimensionality reduction are based in feature selection or extraction, where, instead of working the raw data (all of its dimensions), only the relevant ones are preserved or extracted. Data transformation consists of using some transformation function, create a new representation for the raw data. Whereas, a lower-bounding function is about replacing a costly function by another less costly, which produces a good approximation of the *real* distance.

Data analyzed in data-mining-related tasks vary from long texts to video, passing trough audio, sequences, images, among others. Our interest is in a particular class of sequences, the time series (TS).

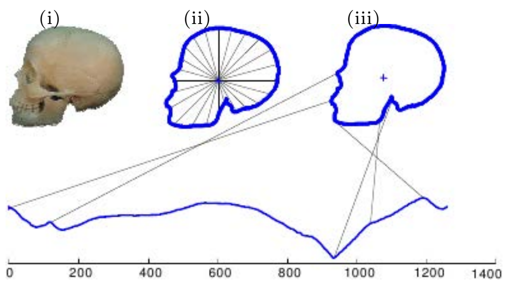
[2]The historical period in the 21st century characterized by the rapid shift from traditional industry that the Industrial Revolution brought through industrialization, to an economy based on information technology.

Doing Data Mining with “Time”

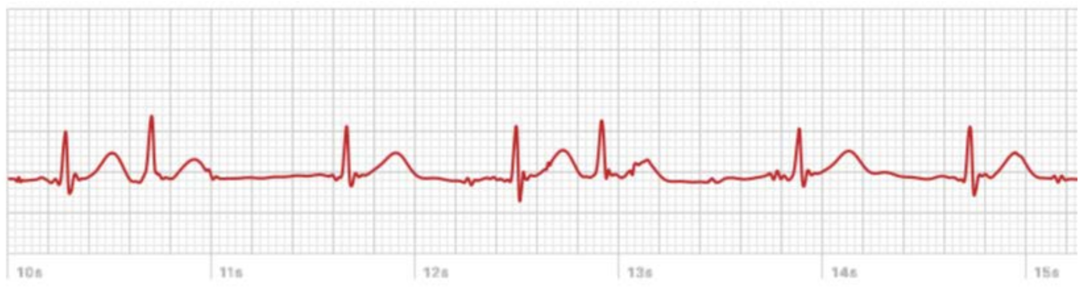
In a straightforward definition, time series is a special kind of data that represents a collection of values obtained from successive equally spaced measurements over time. However, this definition can be relaxed in such a way as to shelter in the “temporal-axis” other kind of measured sequences, like DNA or contours, among others. Figure 1.2 draws some examples of data represented as time series.



(a)



(b)



(c)

Figure 1.2 – Examples of time series: (a) The evolution and trend of the monthly air passengers. (b) A human skull is represented as a time series by firstly finding its outline (i). Then the distance from the center of the skull to each point on the skull’s outline is measured (ii). Finally, those distances are represented as a time series (iii). Lines are used to representing the relationship between the skull contours and the time series shape. In this case, we started at the skull’s mouth and went clockwise [KWX+06]. In (c), an ECG measured by an Apple Watch.

Time series are massively produced 24×7 by millions of users, worldwide, in domains such as finance, weather forecasting, health, earth monitoring, agronomy, among others [EA12]. For example, in Figure 1.2, we can observe the monthly air passengers for a given time interval (fig. 1.2a), this is a common expected scenario for time series. In fig. 1.2b, we observe a time series where the *time* represents the position in the contours sequence. Moreover, finally, in fig. 1.2c, Electrocardiography (ECG) data captured using a personal device. Now, due to the proliferation of the Internet of Things (IoT), time-series data generation can grow exponentially.

Time series can not only have a sizeable linear dimension (on the sense of its length along the time axis) but can also have more than one value varying over time. We call this case of time series as multivariate (or multichannel, or even multidimensional) time series. The adjective varies from the domain of study.

Therefore, algorithms need to be not only effective and efficient in dealing with temporal distortion but also when are vectors varying along the time.

Time Series Data Mining (TSDM) is a comprehensive research domain dedicated to the development of tools and techniques that allow the automatic discovery of meaningful knowledge from time-series data. It provides techniques and algorithms to perform machine learning tasks on time series for assignments as diverse as classification, segmentation, clustering, retrieval, prediction, forecasting, motif detection, subsequences matching, anomaly detection, among others.

Time-series mining exists as a specific field because time series has its specifics properties and challenges. In particular, the meaningful information in the time series is encoded across the time-axis with trends, shapes, or subsequences usually with distortions. Also, this *time-based* factor not only makes it difficult but sometimes impossible to use data mining methods traditionally applied with massive success in other domains.

An interesting feature of time series analysis is that humans have an extraordinary capacity to visualize the *shape* of data, detecting similarities between patterns instantly [EA12]. Our exceptional ability, delivered by the human neural cortex, allows us to ignore temporal distortions, noises, and enable us to deal with what is imperative, avoiding local fluctuations and noise in order to focus globally, and so, developing this overall notion of shape. Hence, time-series data mining emerges from the desire to materialize our natural ability.

For example, by giving an observed ECG from some patient, a medical doctor can make use of a time series retrieval system to look for some similar pattern in a time series ECG dataset. This retrieved information can help to provide the correct diagnoses.

Nevertheless, programming a computer to reproduce our natural capability is a hard problem, and the difficulty arises in capturing the ability to match patterns with some notion of *fuzziness* [BC94]. According to Esling and Agón, these constraints show us that three major issues are involved:

(i) Data representation: How can the fundamental shape characteristics of a time series be represented?

(ii) Similarity measurement: How can any pair of time series be distinguished or matched?

(iii) Indexing method: How should a massive set of time series be organized to enable fast querying?

These implementation components represent the core aspects of the vast majority of time series data mining algorithms. Note that, due to their peculiarities, not all TSDM tasks require these three characteristics. For example, in forecasting, the notion of similarity is not necessary. Time series forecasting is more related to statistical analysis. If a few years ago, the omnipresent approaches were based in someway in flavours of Auto-regressive (AR) models, or Singular Spectral Analysis (SSA) [GAD+02], now deep neural networks, as Convolutional Neural Networks (CNN) or Long Short-Term Memory Networks (LSTM) models are dividing the attention.

On the other hand, tasks like classification, clustering, or retrieval are directly affected by how to assess similarity or how the data are represented. While in classification, the most common supervised task, for a given time series input, the classifier tries to identify what is the category of the input, based on learned data. In the clustering task, a traditional unsupervised approach, we want to identify groups of related time series without any previous knowledge about the data. In its turn, the retrieval task is based on the idea of given a query time series, search on a time-series dataset for the (group of) most similar time series to the query.

In common, for the three approaches previously cited, metrics to establish similarities between time series are specific in the sense that they must be able to take into account the differences in the values making the series as well as distortions along the timeline. However, the retrieval task faces distinct challenges. Firstly, querying needs to be fast; thus, even the choice of the data transformation function is crucial. Second, it is common to handle the scale problem, which forces the use of optimized distance function, the choice of some data structuring. The comparison between pairs of time series can be dealt with by a more sophisticated measuring function; however, when the number of comparisons increases, we need to restrict the use of those complex functions.

Without any doubt, we can say that the most popular similarity measure is DTW. Nonetheless, despite its ability in handling the notion of *shape similarity*, as we will see in Section 2.3.1.2, it is costly to compute, and using it against numerous or very long time series is difficult in practice. Consequently, numerous attempts to accelerate the DTW were proposed; however, scaling the DTW remains a significant difficulty.

Working with raw time series is an arduous task, not only because of the dimensionality of its data but also due to the noise and temporal distortions. The presence of distortions makes unfeasible the use of less robust (and faster) similarity functions.

Therefore it is necessary to develop robust (and consequently computationally expensive) distance functions to deal with distortions or to define a new representation for the raw series, trying to preserve only the relevant information.

Luckily, one characteristic of the time series is that its consecutive values are usually not independent but highly correlated, thus with much redundancy. Such redundancy allows us to develop representation models that exploit these characteristics, such as correlation-based models or representations based on mean values.

A good representation is useful not only due to the data reduction, but it can also help in creating some transformation which allows the substitution of the costly DTW by a less expensive distance function with a minimal impact on the result's quality.

Naturally, the quality of this representation largely depends on the embedding process. Moreover, doing the right transformation, DM algorithms will not only be speed-up but can also produce better results than on the original data. Once the new representation will focus only on the relevant information, removing the noise and undesired parts of the data.

In this aspect, the family of contributions relying on the new concept of shapelets has proved to work particularly well. Originally proposed as a primitive for Time Series Classification (TSC), time series shapelets are phase-independent subsequences extracted or learned from time series to form discriminatory features. An evolution of the concept is to use shapelets to transform the raw time series in high-dimensional vectors and then use those vectors to feed a traditional machine learning classifier.

More recently, Lods *et al.* in [LMTA17] present the Learning DTW-preserving shapelets (LDPS), where they propose to embed the time series into a Euclidean space such that the distances in this embedded space well approximates the true measured by DTW. This work, focusing on the time series clustering, uses learned DTW-preserving shapelets to conduces the transformation.

In this manuscript, we propose the use of DTW-preserving shapelets for the specific context of large scale time-series retrieval.

We mainly focus firstly on developing the framework that can handle univariate and multivariate time series. Then, we evaluate different approaches to handle multivariate data-transformation. Thus, we propose how to evaluate the quality of a single shapelet and a set-of them. The challenge here is to define how good is a shapelet without any provided information.

We have elaborated and developed a new method for feature selection based on clique-elimination, our method presented excellent results in our experiments, proving able to eliminate redundant information. All the proposed methods are evaluated with traditional benchmark datasets.

Contributions

This manuscript emphasized the representation of the information contained in the time series to support the retrieval task.

In this manuscript, we have made four significant contributions:

(i) it explains how DTW-preserving shapelets can be used in the specific context of time series retrieval;

(ii) it proposes some shapelet selection strategies in order to cope with scale, that is, in order to deal with a massive collection of time series;

(iii) it presents a new multidimensional filter for unsupervised feature selection;

(iv) it details how to handle both univariate and multivariate time series, hence covering the whole spectrum of time series retrieval problems.

The contribution's core presented in this manuscript allows us to easily trade-off the transformation's complexity against the retrieval's accuracy.

Large scale experimentation was conducted using the UCR [CKH+15] time series classification archive and the newest UEA multivariate time series classification archive [BDL+18], providing support for this thesis and demonstrating the vast performance improvements compared to state-of-the-art techniques.

Organization

As mentioned early, the purpose of this thesis is to bring new solutions for the time series retrieval problem. We have organized this manuscript in five chapters in order to fulfill this goal.

Chapter 2 gives an overview of the notation and the concepts used in the manuscript and a thorough review of the state-the-art, from basic concepts to advanced algorithms for the time series retrieval is carried out. As we are proposing a new greedy-warping-like algorithm for selecting the best shapelets and also a clique-elimination-based filter approach for shapelet selection, a short revision in the feature selection domain is also necessary and done.

Next, in Chapter 3, we present our approach for TSR based on shapelet transformation, the DTW-Preserving Shapelet Retrieval (DPSR). This chapter also introduces our contribution to the work of Lods *et al.* We have generalized the LDPS to handle multivariate time series by proposing three new multivariate transformations. Besides, we will present our analysis of how to evaluate the quality of the learned shapelets in the original LDPS approach concerning the retrieval task. Based upon that analysis, we have proposed two new methods for feature selection: a greedy-based warp algorithm and a clique-elimination-based filter.

In turn, in Chapter 4, we use the proposed approach for DPSR on UCR time series classification archive and UEA multivariate time series archive, comparing the obtained results against two main competitors: the LB_Keogh lower bound (LB_Keogh) [Keo02] and the Piecewise Aggregate Approximation (PAA) [YF00; KCPM01]. The experimental result shows that DPSR can achieve a better retrieval performance on most datasets.

Finally, we conclude this thesis and present our perspectives in Chapter 5.

Chapter 2

Technical Background and Related Work

We can only see a short distance ahead, but we can see plenty there that needs to be done.

Alan Turing, *Computing machinery and intelligence*

Contents

This chapter gives an overview of the state-of-the-art Time Series Retrieval (TSR) algorithms, from basic concepts to advanced algorithms. First, we introduce notations and definitions related to TSR, and then we present the common distortions viewed in time-series data. TSR depends on how the implemented algorithms handle the distortions efficiently. Many algorithms use raw time series whereas others change the time series representation before the retrieval step. We thus categorize the different time series retrieval models that exist and review some of them with their most relevant characteristics. We start with the shape-based time series retrieval models, then we review the structure-based, finally we introduce the based on features. In this chapter, we only aim at categorizing the different types of time series retrieval algorithms as well as detailing the most famous and most competitive ones; and refer the curious reader to the numerous existing papers on this problem [YJF98; Fu11; RCM+12; EA12]. We conclude this chapter by briefly reviewing other Time Series Data Mining (TSDM) related to the retrieval task.

2.1 Basic Definitions and Notations

In the following, we provide definitions[1] as well as useful notations in order to characterize the time series retrieval problem.

A time series is an ordered sequence of real values, resulting from the observation of an underlying process from measurements usually made at uniformly (time) spaced instants according to a given sampling rate [EA12].

In time-series data, the dependence between the N positions is induced by their relative closeness in the time-axis, that is, considering two possible positions n_h and n_i, they will often

[1] Although expressed primarily for the univariate case, such definitions are straightforwardly extended to the multivariate case; we will make throughout the text, the distinction when necessary.

be highly dependent if $|h - i|$ is small, with decreasing dependency as $|h - i|$ increases [Jol02, p. 296]. A time series can gather all the observations of a process, which can result in a very long sequence. Also, a time series can result in a stream and be semi-infinite.

We can define a time series T as[2]

Definition 2.1 *Time series. In a nutshell, a univariate time series (or only time series) T of length $|T| = N$ is a sequence of real-ordered values typically collected in equally spaced intervals and formalized as:*

$$T = \{t_1, t_2, \ldots, t_N\} \tag{2.1}$$

where t_i denotes the i^{th} element of the time series T.

When the collected values are in a space $\mathbb{R}^M$ such that $M > 1$, these time series are known as multivariate time series, sometimes called multidimensional or even multichannel.

Definition 2.2 *Multivariate time series. A multivariate time series T is a set of M instances of $T \in \mathbb{R}^N$ univariate time series, expressed as:*

$$\mathrm{T} = \{T_1, \ldots, T_M\} \in \mathbb{R}^{N \times M} \tag{2.2}$$

where $t_{i,j}$ is the j^{th} element of the i^{th} channel of T.

For some application, it is more interesting to focus on the local shape properties of a time series rather than the global statistical properties, such as (global) mean, standard derivation, skewness, among others. These contiguous pieces of a long time series are called subsequences.

Definition 2.3 *Time series subsequence. Given a time series T of length N, a subsequence S of T is a continuous sampling of T, with length $|S| = l \leq N$, and starting at the position p of T, expressed as:*

$$S = \{t_p, t_{p+1}, \ldots, t_{p+l-1}\} \tag{2.3}$$

where the value p is an arbitrary position in T such that $1 \leq p \leq N - l + 1$.

Figure 2.1 draws an example of subsequence.

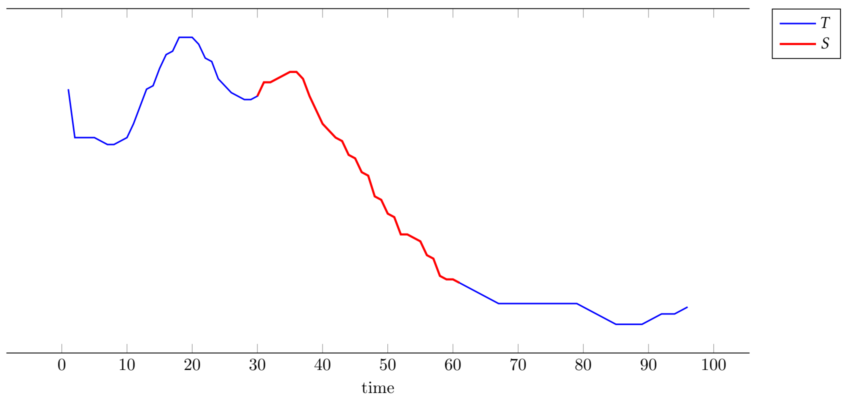


Figure 2.1 – Illustration of subsequence S (in red) of a time series T.

Time series data mining related tasks suffer from the *curse of dimensionality* due to the high dimensionality of the data. Hence, it is common to work with a simplified representation of the series, defined as:

Definition 2.4 *Time series representation. Given a time series T, with $|T| = N$, a representation of T is a model $\overline{T} = \{\overline{t}_1, \ldots, \overline{t}_c\}$ where $c \ll N$, obtained by applying a transformation function F such that $\overline{T} = F(T)$.*

[2]For ease of notation, we will use the letters Q and R to represent a time series when necessary.

In general, besides the significant reduction of the data dimensionality, those transformations aims to emphasize fundamental shape characteristics or features, handling with noise and time-shifting.

For easier storage, massive time series-sets are usually organized in a dataset $\mathcal{T}$.

Definition 2.5 *Time series dataset. A time series dataset is a set of* $|\mathcal{T}| = K$ *unsorted time series, formalized as:*

$$\mathcal{T} = \{T_1, \dots, T_K\} \tag{2.4}$$

We will discuss some aspects of time series distortions in the following section, and after in the subsequent sections, we will present the methods for Time Series Retrieval (TSR).

2.1.1 Time Series Distortions

TSDM brings up many challenges due to the specificity of time series data. Time series comes from real-world measurements: even similar time series will present distortions on its time-axis or in the amplitude of its recorded values.

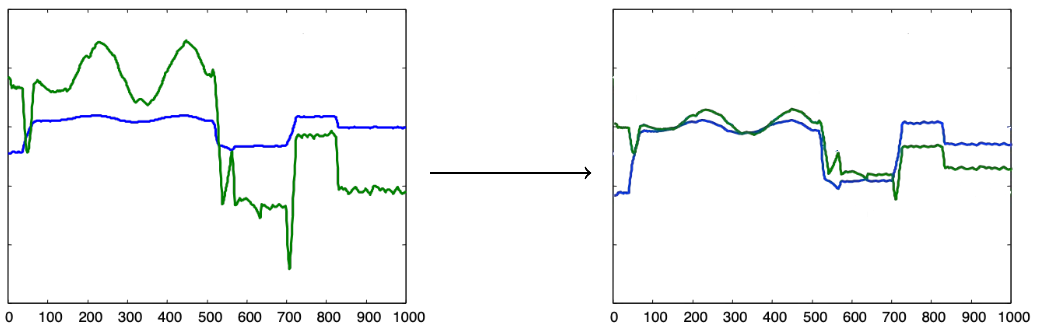
Many factors can produce those distortions, like a miscalibrated or inaccurate sensor, the use of different measurements units, or noise, for example [EA12]. In this section, we propose a brief overview of the commonly encountered in time series.

Distortion on Amplitude and Trend

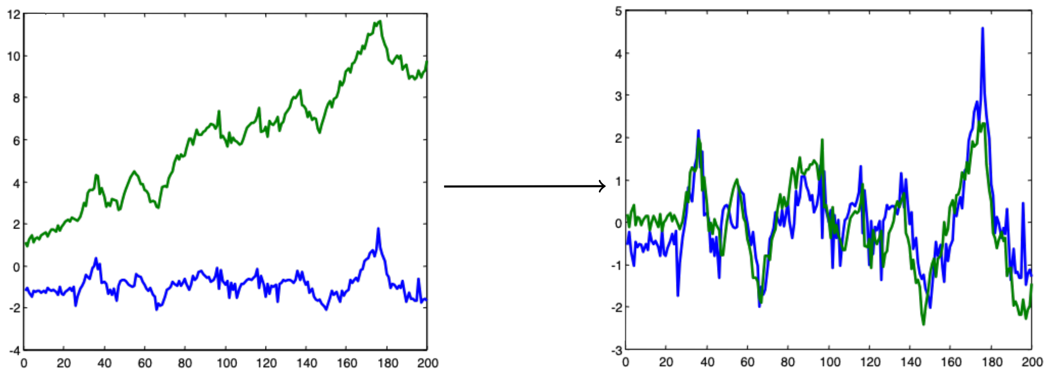
One common problem when dealing with time-series data is that they can present similar shapes while their values are on different scales. Many factors may influence the amplitude, for example, differences in the recorded volume (in the case of sound), illumination (image), data measured with different units like temperature measured in Fahrenheit or Celsius. Those factors can change the amplitude of the measured data while the intrinsic shape is comparable, therefore, if our interest is in looking for similarity on the shape, scaling the time series by removing the amplitude factor is a *sine qua non* condition.

Similarly, even if both time series have identical amplitudes, they may have different trends (or offset), in other words, it presents varying mean over time, and for some scenarios, the offset in the values (resulting from the trend) breaks the distance function. Similarly, even if both time series have identical amplitudes, they may have different trends (or offset); in other words, it presents varying mean over time. For some scenarios, the offset in the values (resulting from the trend) breaks the distance function. Some distance measures, such as the Euclidean distance, are sensitive to this family of distortion and fails to recognize the similarity of time series exclusively due to differences in the amplitude.

In this way, a usual pre-processing stage consists in applying a *z-normalization* to the data before evaluating the similarity, as illustrated in Figure 2.2a, or by scaling and removing the trend factor, as showed in Figure 2.2b.



(a)



(b)

FIGURE 2.2 – (a) Amplitude scaling: If compared before amplitude scaling (left), these time series appear very different. After applying the z-normalization, it appears more similar. (b) Applying the amplitude scaling plus offset translation and linear trend removal, allow to observe the similarity between the series.

Local Distortion on the Time-axis

As common as the previous case, however, more complex to be handled, we can visualize the local distortions on the time-axis as acceleration or deceleration in the phenomenon registered by the data points on the time series. Local distortion on the time-axis is the most common phenomenon handled in TSR and is showed in Figure 2.3.

Conventional approaches to handle this type of distortion are based on using a similarity function able to match non-aligned points between the sequences, i. e. , an elastic measure. Alternatively, creating a new representation focusing on the relevant features of the time series, and then doing the similarity evaluation on the transformed space.

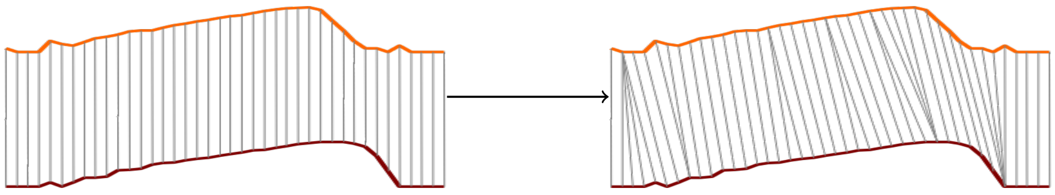


FIGURE 2.3 – Local distortion on the time-axis: The lines represent the comparison point-to-point. While in the traditional non-elastic measures (left) points are matched one-to-one, in the elastic measure, it allowed matching one-to-many (right).

Distortion on Phase

In some scenarios, the phenomenon of interest may be randomly positioned on the time series. For example, when using time series to represent a contour of a two-dimensional object, the starting point of the time series may not be adequately positioned causing a phase mismatch, as observed in Figure 2.4.

There are two solutions for this issue: either the whole time series are comparable and all the possible alignments must be tested [BKTdS14], or by using some representation based on feature extraction, and then doing the comparison on the transformed space.

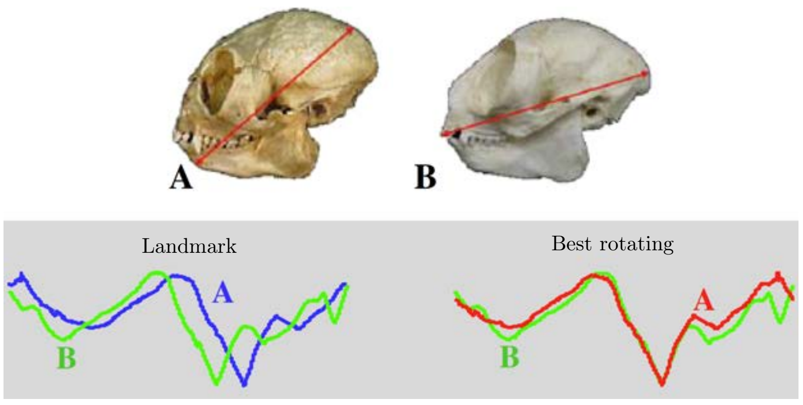


FIGURE 2.4 – Phase distortion: Two primate skulls (A and B) represented as a time series of its contour. In the Landmark matching, the major axis is defined as the starting point (red line), phase distortion is observed and can conduce to incorrect similarity measure (left). Considering the best rotation, the measured similarity increases (right). Adapted from [KWX+09].

Occlusion-based Distortion

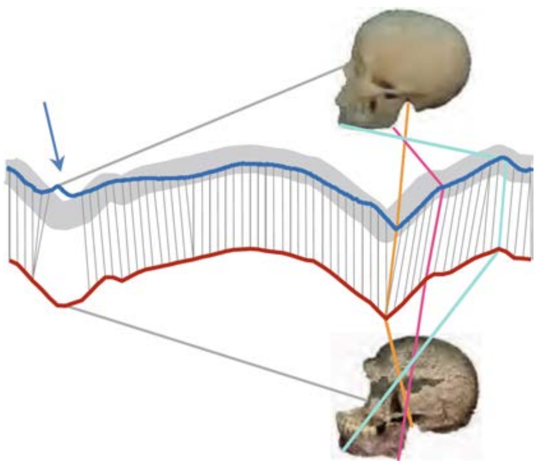
This distortion occurs when small subsequence(s) of a time series may be missing or presenting an abnormal pattern. For example, some information can be lost while collecting or transmitting by a sensor, or it may be disturbed by some phenomena.

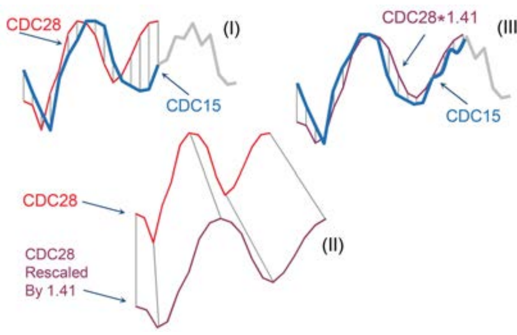
One way to handle occlusion-based distortion is allowing the elastic measure function to ignore sections (with the possibility of some penalty) that are difficult to match. Figure 2.5a shows a more visually intuitive example, where, probably, mutations throughout evolution have led to disturbances between the observed shapes.

Uniform Distortion on the Time-axis

Uniform distortion on the time-axis, in contrast to the localized distortion, can be seen as a case where the whole time series is uniformly warped in time. In other words, the time series is compressed or stretched, as showed in Figure 2.5b.

Methods proposed to handle local distortion does not perform well on a uniform distortion case. It is therefore preferable to re-scale one (or both) series before comparing them. However, the challenge in handling uniform scaling is that there is not an easy way to know the scale-factor before ahead.


(a) Occlusion.



(b) Uniform scaling.

FIGURE 2.5 – Occlusion and uniform scaling distortions: (a) Occlusion distortion: The nose region of the ancient skull (bottom) has no correspondence and is missing on the modem skull (top). A way to out-pass this issue is ignoring the non-matching segment (blue arrow). (b) Distortion on uniform scaling: (I) Without any pre-processing, a full gene expression, CDC28, matches poorly to the prefix of a related gene, CDC15. (II) If it is re-scaled by a factor of 1.41, it matches CDC15 much more closely (III). Adapted from [BWK11; BKTdS14].

2.2 Time Series Retrieval

The retrieval task (or query by content) is the most active area of research in time series analysis [EA12]. It aims to locate objects in a dataset that are most similar to a query object Q provided by the user.

According if it handles or not the raw time series, retrieval can be classified in shape-based, structure-based or feature-based. Also, the queries can be called on complete time series (whole matching) or to find every subsequence of the series matching the query (subsequence matching). In this thesis, we focus on feature-based whole-matching retrieval.

For the TSR task, the notion of similarity is a crucial concept, and the measurement of similarity between time series is a paramount subroutine not only for the TSR but also almost every TSDM task requires at least an indirect notion of similarity between pairs of time series.

Its purpose is to compare two instances by computing a single value evaluating their similarity. A similarity (dissimilarity) measure is a real-valued function that measures how close (far) two instances are to each other. The closer the instances are, the larger the similarity is. There exists many (dis)similarity measures and definitions of it, amongst them we indicate a special case: distance metrics.

In the case of two time series Q and T with, respectively, length of N_Q and N_T, the distance function D is defined as:

Definition 2.6 *Distance between time series.* $\mathrm{D}(Q,T)$ *is a distance function that takes time series* T *and* Q *as inputs and returns a non-negative value* d*, which is said to be the distance or (dis)similarity between* Q *and* T*.*

The distance function D can also be used to measure the distance between two subsequences of the same length since the subsequences, in this case, can be viewed as an entire series. However, we will also need to measure the similarity between a short subsequence and a (potentially much) more extended time series. We therefore define the distance between two time series T and S, with $|S| \ll |T|$ as:

Definition 2.7 *Distance from a time series to the subsequence.* $\mathrm{subD}(T,S)$ *is a distance function that takes time series* T *and subsequence* S *as inputs and returns a non-negative value* d*, which is said to be the distance or (dis)similarity between* T *and* S*.*

$$\mathrm{subD}(T,S) = \min\left(\mathrm{D}(S,S^{'})\right) \;\forall\; S^{'} \in \mathcal{S}_T^{|S|}, \tag{2.5}$$

where $\mathcal{S}_T^{|S|}$ *is the set of all possible subsequences in* T *with length* $|S|$*.*

Based on the concept of similarity distance, we can define the time series retrieval task as follows:

Definition 2.8 *Time series retrieval. Given a time series* Q*, the time series retrieval problem is to find* $T^* \in \mathcal{T}$ *that is the most similar to* Q *according to the similarity measure* D*:*

$$T^* = \operatorname*{argmin}_{T_i \in \mathcal{T}} \mathrm{D}(Q,T_i) \tag{2.6}$$

We can generalize the previously definition to retrieval not only the closest element T^*, but also all elements inside an area defined by a threshold-range φ (Figure 2.6b) – φ-range retrieval, defined as:

Definition 2.9 *Time series range search. Given a query time series* Q*, a time series dataset* $\mathcal{T}$*, and a range threshold* φ*, the range search will return a subset* $\mathcal{R} \subset \mathcal{T}$ *such that:* $\mathcal{R} = \{T_i \in \mathcal{T} \mid \mathrm{D}(Q,T_i) < \varphi\}$*, for a given similarity measure* D*.*

Or even the k-nearest neighbors (Figure 2.6c) – k-NN retrieval, defined as:

Definition 2.10 *Time series* k*-nearest neighbors search. Given a query time series* Q*, a time series dataset* $\mathcal{T}$*, and a integer* k*, the* k*-NN search will return a subset* $\mathcal{R} \subset \mathcal{T}$*, where the time series* $T_k \in \mathcal{R}$ *are the* k *most similar to* Q*, according to the similarity measure* D*.*

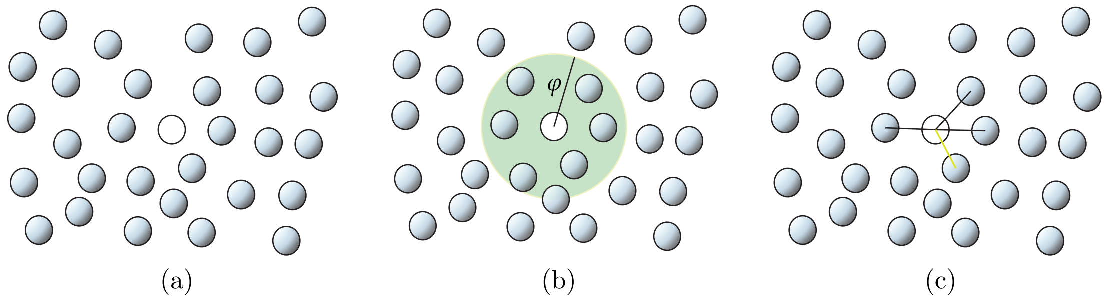


FIGURE 2.6 – Diagram of a typical query by content task represented in a 2-dimensional search space. Each point in this space represents a object whose coordinates are associated with its features. (a) A query object (orange point) is first transformed into the same representation as that used for other data points. Two types of queries can then be computed. (b) A φ-range query will return the set of series that are within distance φ of the query. (c) A k-nearest neighbors query will return the k points closest to the query, the yellow line represents closest.

Figure 2.6 illustrates a typical query by content task in a 2-dimensional search space.

In turn, subsequence retrieval requires searching all subsequences in the dataset, therefore, to compare a subsequence query C, $|C| = l$ and a time series T, $|T| = N$, where $l \ll N$, we need to extract all subsequences $S \subset T$ with length l. We can obtain all subsequences by using a sliding window of the appropriate size.

Definition 2.11 *Sliding Window. Given a time series T, $|T| = N$ and a subsequence of length l, all possible subsequences of T can be extracted by sliding a window of size l across T and considering each subsequence S_p^l. Here, the superscript l indicates the length of the subsequence and the subscript p is the starting position of the sliding window over T.*

The set of all subsequences of length l extracted from T can be defined as:

Definition 2.12 *Set of all subsequences of length l in T. Given a time series T, $|T| = N$ and a subsequence of length l. The set of all subsequences of length l extracted from T is expressed by:*

$$\mathcal{S}_T^l = \{S_p^l \text{ of } T, \text{ for } 1 \leq p \leq N - l + 1\}. \tag{2.7}$$

We use $\mathcal{S}_{\mathcal{T}}^l$ to represent the set of all subsequences of length L in the dataset $\mathcal{T}$.

Although it seems trivial, the retrieval task faces numerous challenges, and due to many reasons depending on the data, this task is far from easy. The main challenges stem from:

(i) Even a time series considered short has dozens of positions on the time-axis. That is, time series is high-dimensional data.

(ii) Usually, on TSR tasks, the dataset is massive and needs to support adding and removing objects. Therefore, approaches for TSR need to work at scale in open datasets.

(iii) Surely the time series data in the dataset will present some distortion on the time-axis. Consequently, the similarity measurement function needs to be able to handle distortions.

(iv) The query response time needs to be fast. Thus, the similarity function needs to be effective in handling distortions but also efficient in doing it faster.

Three main ways were found out to handle those previously listed challenges. The first is defining similarity measure functions that are robust to time-axis distortions. The second is proposing data-transformation functions able to extract features expressing what "really means" in the time series data, and consequently mitigating the temporal distortions. Finally, the combination of both, i. e. a transformation is applied to simplify the data together with a similarity function able to handle distortions.

In the following sections, we will present the most representative and used similarity measures and data transformations applied to time series retrieval.

2.3 Data Representation and Distance Functions

Data representation formats and distance functions are closely related to each other. Therefore, we are reviewing together current techniques on these two issues. Generally, we can classify these techniques into two main categories: distance functions on raw representations, or distance functions on data-transformed representations.

Following, we will discuss the characteristics of the main types of distance functions applied to time series. Our objective in this section is not to be exhaustive on time series distance measures but rather to give some instances of the preeminent families of distance measures: we encourage the reader to refer to dedicated surveys such as [DTS+08; WMD+13; KK03].

2.3.1 Distance Functions on Raw Representation

As previously defined in the Definition 2.1, a time series data can be represented as a sequence of real-ordered values, $T = \{t_1, \ldots, t_N\}$. T is called raw representation of the time series data. Many distance functions have been proposed based on raw representation.

The definitions of distance functions presented in this section use univariate time series data ($M = 1$) as an example, but they can be easily extended to multi-dimensional data.

2.3.1.1 Euclidean Distances and $\mathrm{L_p}$ Norms

The $\mathrm{L_p}$ distance family contains the most straightforward similarity measures for time series. Also known as Minkowski distance, the $\mathrm{L_p}$ distance corresponds to a family of functions, where $p \in \mathbb{R}_+^*$, and can be considered as a generalization of Manhattan ($p = 1$), Euclidean ($p = 2$), and Chebyshev ($p = \infty$) distances.

Given two time series T and Q of the same length N, the $\mathrm{L_p}$-norm (or Minkowski) distance between T and Q is:

$$\mathrm{L}_p(T, Q) = \|T - Q\|_p = \sqrt[p]{\sum_{i=1}^{N} |t_i - q_i|^p} \tag{2.8}$$

Undoubtedly, the most known and most used distance measure is the Euclidean distance (ED) due to its simplicity to understand and effortlessness to compute. To compare the time series T and Q with the same length using the Euclidean distance, we compare each-other point-to-point, where the similarity is defined by:

$$\mathrm{D}(T, Q) = \mathrm{L}_2(T, Q) = \sqrt{\sum_{i=1}^{N} (t_i - q_i)^2}. \tag{2.9}$$

Figure 2.7 shows a visual intuition of the Euclidean distance metric.

As in other data domains, the traditional $\mathrm{L_p}$-norm have been the first choice to measure the similarity in time series data [YF00; EA12]. It has the advantage of being easy to compute, parameter-free, and the computation cost is linear in terms of time series length. However,

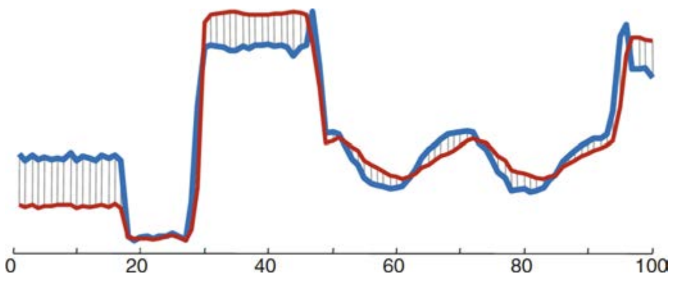


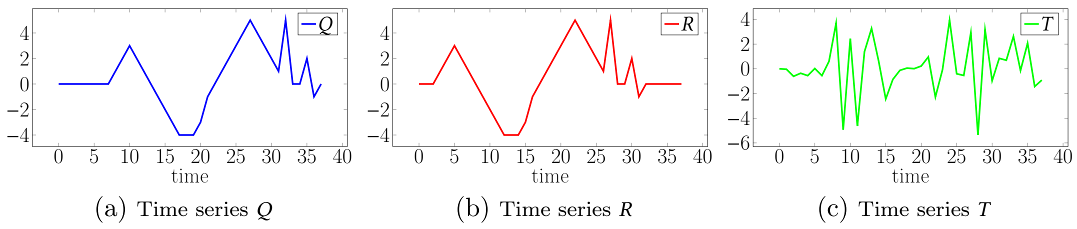
FIGURE 2.7 – Viewing the Euclidean distance: The Euclidean distance between two time series can be visualized as the square root of the sum of the squared length of the vertical hatch lines. Adapted from [BWK11].

unlike other data types where it achieves good similarity measure, on time series data L_p-norm tends to lead results that do not reflect the human perception [DTS+08; EA12].

Humans have an extraordinary capability to observe similar patterns. The evolution brought us an ability to filter noise, phase shift, time distortions, scaling, amplitude, outliers allowing to focus essentially on the shape of the time series.

Therefore, it is evident that the simplicity of the L_p-norm fails to reach such level of abstraction, may causing then a totally unexpected similarity behavior.

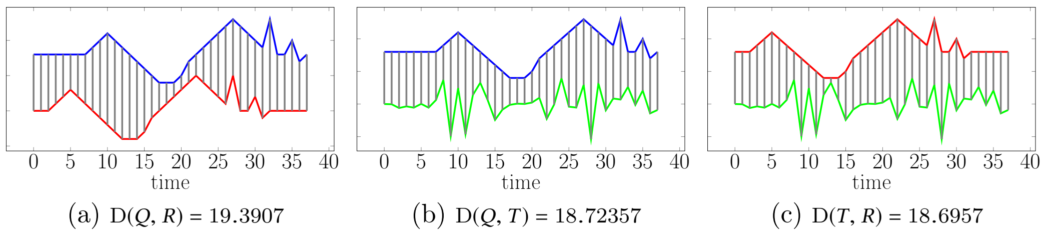
We can resort a toy example to exhibit this phenomenon. Considering a time series dataset $\mathcal{T} = \{Q, R, T\}$ with $|Q| = |R| = |T| = N$, from Figure 2.8.



(a) Time series Q (b) Time series R (c) Time series T

FIGURE 2.8 – Toy example of a time series dataset $\mathcal{T}$ with three time series.

Based on the behavior of our natural perception of shape, we believe that the pair (Q, R) is the most similar pair in $\mathcal{T}$. However, the opposite is observed when we are using the L_2-norm, and this pair of sequences is considered the most dissimilar, as observed in Figure 2.9.



(a) $D(Q, R) = 19.3907$ (b) $D(Q, T) = 18.72357$ (c) $D(T, R) = 18.6957$

FIGURE 2.9 – The use of L_2-norm as distance function D conduces to an unexpected measure of similarity. In this context, the pair (Q, R) (in (a)) is evaluated as the least similar pair, contradicting the human perception.

Below we will show how other metrics can be used to mitigate this issue of the Euclidean distance.

2.3.1.2 Dynamic Time Warping

The DTW is a well-known algorithm in many fields, with application in speech and gesture recognition, handwriting and online signature matching, music and signal processing, protein sequence alignment and chemical engineering, time series data mining, among others [Mül07b, p. 80].

DTW algorithm has become extremely popular due to its efficiency as time series similarity measure [KL83; BC94; Mül07a]. The great advantage of the DTW-based similarity over traditional L_p-norm is its ability to effectively handle the effects of local shifting, and time series with different lengths. In some sense, DTW preserves the human sense of shape-similarity by allowing *elastic* transformation of time series in order to detect similar shapes across different phases [EA12].

The objective of DTW is to find an optimal alignment between two time series Q and T that achieves minimum global cost while ensuring time continuity.

Before going to discuss how it works, we will observe the results of the DTW measure applied to the same previous toy example from Figure 2.8. In Figure 2.10 we can see how DTW handles time distortions and delivery an expected similarity based on the human perception of shape similarity.

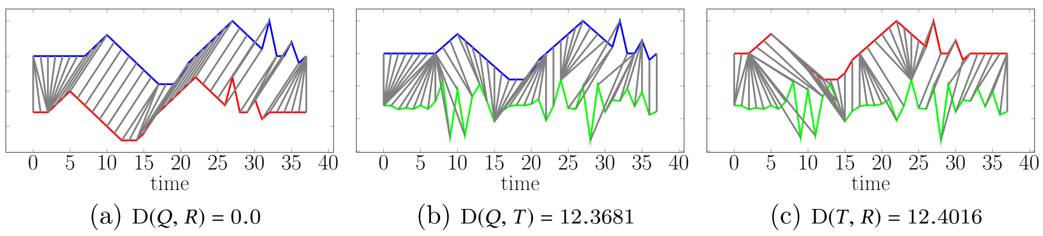


FIGURE 2.10 – A more intuitive result is obtained when using DTW as a metric for distance function D. Due to its ability to handle time shifts and distortions, DTW defines the pair (Q, R) the most similar, which is the intuitive similarity expected.

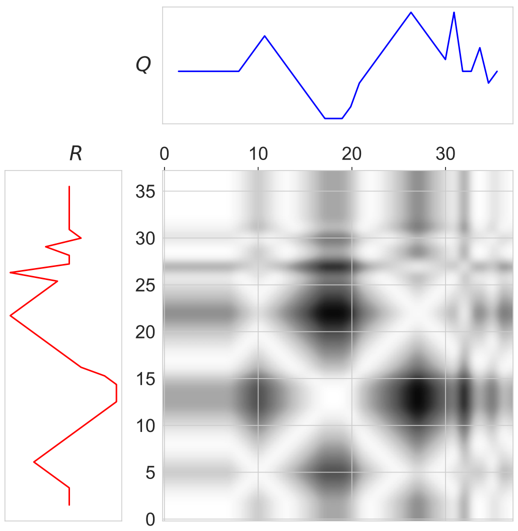
The DTW distance between two time series R and Q of lengths N_R and N_Q, respectively, is defined as:

$$\mathrm{DTW}(R_{:i}, Q_{:j}) = \delta(r_i, q_j) + \min \begin{cases} \mathrm{DTW}(R_{:i}, Q_{:j-1}), \\ \mathrm{DTW}(R_{:i-1}, Q_{:j-1}), \\ \mathrm{DTW}(R_{:i-1}, Q_{:j}) \end{cases} \tag{2.10}$$

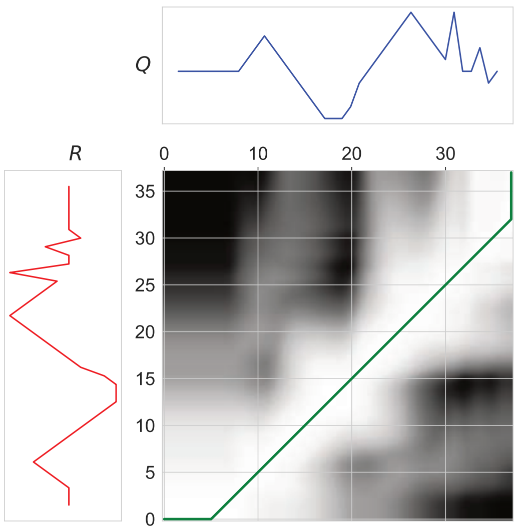
where $Q_{:i}$ represents the subsequence $\{q_1, q_2, \dots q_i\}$, and $\delta(r_i, q_j)$ is the local cost function that measures the distance between two elements. Traditionally, Euclidean distance (L_2-norm) or squared Euclidean distance is used as the local cost function δ, although nothing prevents the use of some other exotic metric.

DTW works by finding an optimal alignment between two time series Q and R that achieves minimum global cost while ensuring time continuity. The global cost is the summation of the cost between each pair of points in the alignment. Intuitively, such an optimal alignment runs along a "valley" of low cost within the local cost matrix, see Figure 2.11b for an illustration.

DTW allows some elements to be replicated (i. e. the one to many links observed in Figure 2.10) to accommodate the cases where elements are similar but out of phrase (time-wise). Comparing Figure 2.9 and Figure 2.10 reveals how DTW can handle local time shifting whereas Euclidean distance cannot.



(a)



(b)

FIGURE 2.11 – (a) Local cost matrix of the time series Q and R using the L$_2$-norm as local cost δ. (b) Accumulated cost matrix with the optimal warping path. In both (a) and (b), regions of low cost are indicated by light colors, and regions of high cost are indicated by dark colors.

However, this approach for the DTW, sometimes called as naïve or classical, presents three main issues:

(i) in the best case, the computation cost of DTW is quadratic, $O(N_Q \cdot N_T)$, – where N_Q and N_T are the lengths of two compared time series, respectively –, with dynamic programming; therefore, when the time series length or the database size increases, enormous time requires to be spent on computing DTW to answer a query unless some indexing methods can be used to save the number of computations.

(ii) Unfortunately, DTW does not follow the triangle inequality, which precludes its use with many indexing structures, such as M-Tree family [CPZ97], Slim-Tree [JTSF00; SJdPG15] or D-index [DGSZ03], among others.

(iii) Finally, it may not give the best mapping according to our need because it will try to find the minimal distance, which in some situations it can result in unwanted path. For example, in Figure 2.12, without any path limitation, DTW will finds the optimal path between the two time series, but this minimal cost path, may is not the best choice according to our perspectives.

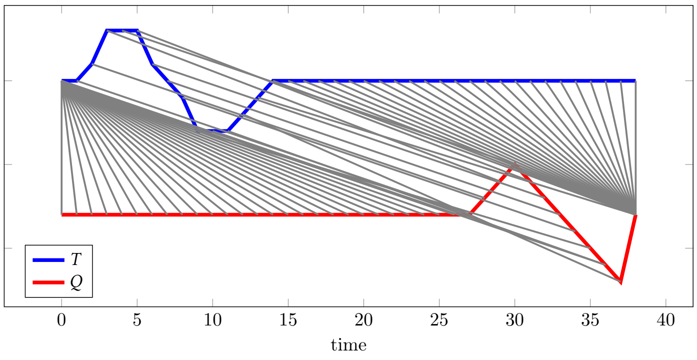


FIGURE 2.12 – An unwanted warping between time series Q and T.

Consequently, innumerable approaches have been proposed to mitigate the quadratic time complexity of the naïve DTW.

Variations of DTW

In order to mitigate the previous listed issues of the naïve DTW algorithm various modification were proposed. This section outlines the major modifications to suite the DTW to a specific domain such as step size condition, step weighting, global path constraints, and invariance of prefix and suffix.

As mentioned before, one of the main drawbacks of the naïve DTW is its tendency to generate unwanted paths, by assigning a single element of one sequence to many consecutive elements of the other sequence, or vice versa. These phenomena can be observed in Figure 2.12, and its lead to vertical and horizontal long segments in the optimal warping path, and we can interpreter this vertical (or horizontal) segments on the optimal warping path as a local acceleration (or deceleration) by a large factor. Therefore, in scenarios where we have similar velocity between the series evaluated this optimal path can be considered invalid.

To avoid such degeneration, many modifications of the step size condition constraint were proposed to try to restrict paths quite away from the diagonal. One can modify the pattern of the step size condition to restrict the slope of the admissible warping path.

Sakoe and Chiba in [SC78] define a factor $p = \frac{k_d}{k_l}$ in the cost matrix, where k_d is the number of stepping in its diagonal and k_l is the number of stepping in the direction of i (or j)-axis.

The factor p defines how strict is the allowed path. When $p = 0$ there are no restrictions, as in the naïve implementation. At the other extreme, when $p = \infty$ (that is $k_l = 0$) the path is restricted to diagonal line $j = i$, therefore, becoming equivalent to the traditional Euclidean distance.

Another way of affecting the optimal warping path is varying the weight for each possible direction. Previously we defined the measure of distance between time series as the accumulated cost function of the optimal warping path, which is a summation of pairwise distances between corresponding points at time-series Q and T. By construction, the original implementation based on the recursive has a preference of the diagonal alignment direction, since one diagonal step (cost of one cell) corresponds to the combination of one horizontal and one vertical step (cost of two cells). However, we can favor the vertical, horizontal, or diagonal direction in the alignment, by introducing an additional weight vector $\langle w_d, w_v, w_h \rangle$ to the DTW function.

Another widely used modification is the global path constraint also referenced as windowing functions or bands. As well as local constraints, are used to impose conditions to the admissible warping paths.

The global constraints define the region of cells in the accumulated cost matrix that are search for the optimal path. Cells outside this region are not searched [BC94]. Essentially, the global constraint define the overall stretching or compression allowed for the matching procedure. Therefore, it is clear that such windowing functions do not only prevent unwanted alignments but also speed up DTW computation by limiting the number of local cost computations.

Sakoe-Shiba band (SC-band) [SC78] is one of the simplest and most commonly used bands to limit calculation of cells in the cost matrix. SC-band runs along the main diagonal allowing only values inside the band $(2r + 1)$. Where r is a user-defined parameter which specifies the length of the window. Obviously, when comparing time series with different length, the upper limit r must be correctly defined.

Another traditional approach is the Itakura parallelogram [Ita75]. Contrary to the SC-Band, where r is independent of i (and of j) and constant along the diagonal of the cost matrix, in the Itakura parallelogram the bound limit of the window is a function of i (and j).

Mostly, the Itakura parallelogram describes a restriction that constraints the slope p of a warping path. An interesting characteristic of the Itakura parallelogram is that horizontal transitions are allowed, but not successively. Thus, Itakura constraints do not allow long horizontal paths, corresponding to the $p = \infty$ slope [TPKC10, p. 334].

However, the Itakura parallelogram it was designed to limit warping at the start and end of the naïve DTW warping path, where the first and last warping points are exactly known, limiting its use for purpose of searching subsequences and other DTW modifications.

In order to introduce more flexibility than a fixed authorized shape, some researchers prefer to learn an adaptive window shape from the data as presented in [KRGI11; Mar18; RK04b; KSM16].

It should be noted that, despite the positive impact in the execution time for long time series, the problem stills in a quadratic time complexity $O((r \cdot (N_Q + N_T))$, and as the optimal warping path can traverse cells outside the window, it may lead to undesirable alignment results. Furthermore, they heavily restrict the number of candidates for the optimal path which may cause undesired alignment results as seen in Figure 2.13.

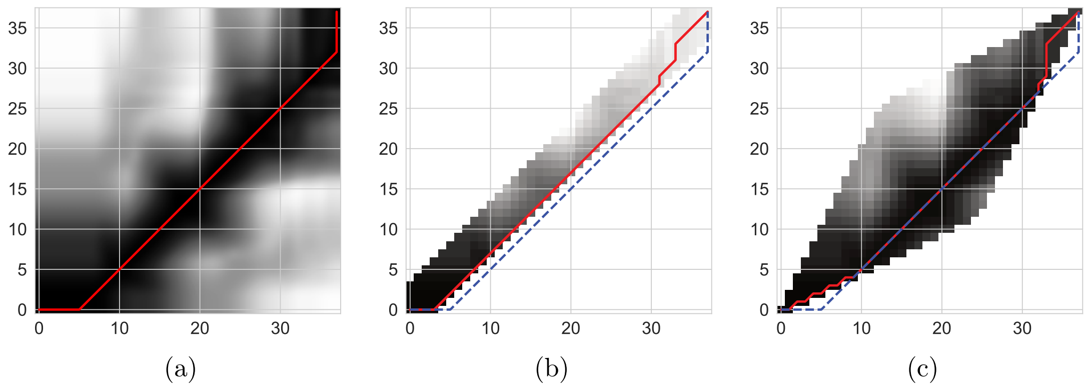


FIGURE 2.13 – (a) Optimal warping path without global constraint. (b) Restricted optimal warping path (in red) using the Sakoe-Shiba band with $r = 4$, in blue the optimal warping path. (c) Restricted optimal warping path (in red) using the Itakura paralelogram. Regions of low cost are indicated by light colors, and regions of high cost are indicated by dark colors.

On the other hand, other smart optimizations were proposed to speed-up the DTW computation in huge datasets. One way to address it is the early abandoning of DTW, where during the computation of the optimal accumulated distance, if we note that the current value exceeds the best-so-far, then we can stop the calculation. Another well-known strategy is to use a fast lower bounding function to avoid the need to apply the DTW on time series that are impossible to be the best match [KR05].

Basically, a lower bounding measure for DTW has two desirable properties:

- It must be *really* faster then the DTW ;
- It must tightly approximate the true DTW distance.

In general, the challenge to define a lower-bound function is to handle the trade-off between the tightness of the lower bound and how fast it is to compute.

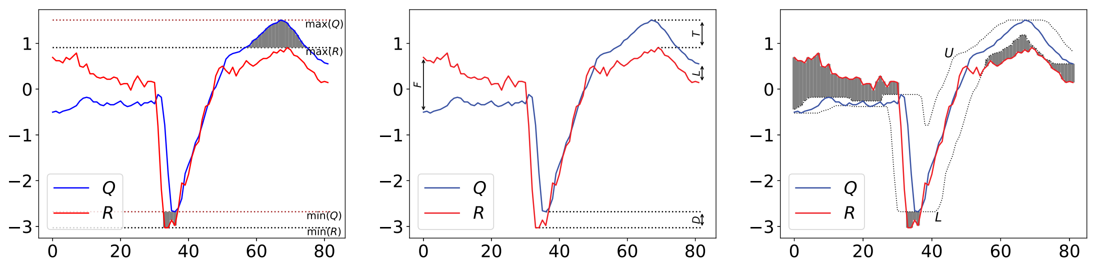
Once the lower bounding function is defined, an algorithm to speed-up the sequential scan search can be easily defined as in Algorithm 2.1.

Yi, Jagadish, and Faloutsos in [YJF98] introduced the first approach to lower bound DTW. In this function, referred LB_Yi, the authors take advantage of the observation that the final DTW distance is directly dependent to all the points in one sequence that are larger (smaller)

Algorithm 2.1 A lower-bounding based algorithm to perform sequential scan.

```
 1: input: $Q$, $\mathcal{T}$ {Time series query and dataset}
 2: output: $C$ {True 1-NN}
 3: $C \leftarrow -1$
 4: $best\_dist \leftarrow \infty$
 5: $true\_dist \leftarrow \infty$
 6: for all $T_i \in \mathcal{T}$ do
 7:    $lb\_dist \leftarrow \mathrm{LB}(T_i, Q)$
 8:    if $lb\_dist < best\_dist$ then
 9:       $true\_dist \leftarrow \mathrm{DTW}(T_i, Q)$
10:       if $true\_dist < best\_dist$ then
11:          $best\_dist \leftarrow true\_dist$
12:          $C \leftarrow T_i$
13:       end if
14:    end if
15: end for
16: return $C$
```

than the maximum (minimum) of the other sequence, as we can see in the Figure 2.14a. After



(a) (b) (c)

FIGURE 2.14 – Lower bounding for DTW applied in two time series from the TwoLeadECG dataset [CKH+15]. (a) in LB_Yi the sum of the squared length of gray lines represents the minimum of the corresponding points contribution to the overall DTW distance, and thus can be returned as the lower bounding measure. (b) LB_Kim looks at specific points to evaluate the lower-bounding. Finally, (c) LB_Keogh explores the ideia of local and global constraints to create an envelope U and L to evaluate the lower-bounding.

this first work of Yi, Jagadish, and Faloutsos, more than 18 new approaches for lower bound DTW were proposed [RCM+12]. Two of the most used are the LB_Kim and LB_Keogh, described below.

Kim, Park, and Chu in [KPC01] introduced the LB_Kim, a lower-bound function which works by extracting a four-tuple feature vector from each sequence. The features are the first and last elements of the sequence, together with the maximum and minimum values, as we can observe in the Figure 2.14b. The maximum squared differences of corresponding features are reported as the lower bound. Such a representation of the series by this four-tuple mitigates the *curse of dimensionality* and, consequently allows efficient use of spatial indexing methods. We can observe that, like LB_Yi, this function has complexity $O(N)$. An $O(1)$ extension of LB_Kim, called LB_KimLF, consists of using only the first (F) and last point (L), however, sacrificing the tightness.

Inspired by the work of Yi, Jagadish, and Faloutsos, Keogh introduced the LB_Keogh [Keo02]. They have assumed that global and local constraints are naturally desirable over the naïve DTW, thus LB_Keogh may be seen as an enhancement of LB_Yi for this scenario.

As discussed earlier, local constraints or windowing restrict the indices on warping path inside a range defined by the windowing function or the user defined parameter, as the radius of the window, the step weighting, among others.

Therefore, given a time series T, LB_Keogh uses the restriction R to create two new time series $L = [l_1, \ldots, l_{|L|}]$ and $U = [u_1, \ldots, u_{|U|}]$ defined as:

$$\forall i \in [1, \ldots N_T] \wedge r_i \in \mathrm{R}, u_i = \max(t_{i-r_i}, \ldots, t_{i+r_i}) \tag{2.11}$$

$$\forall i \in [1, \ldots N_T] \wedge r_i \in \mathrm{R}, l_i = \min(t_{i-r_i}, \ldots, t_{i+r_i}) \tag{2.12}$$

Those time series corresponds to the Lower and Upper limits of an envelope that represents the influence of each point in the warping path as we can see in Figure 2.14c.

To be able to compare the sequence Q, L and U, these time series must be of the same size, so when Q and T have a distinct length it is necessary to, for example, re-sample the sequence Q. The lower-bounding distance is given by the Equation (2.13).

$$\text{LB\_Keogh}(Q, T) = \sqrt{\sum_{i=1}^{N_Q} \begin{cases} (t_i - u_i)^2 & \text{if } t_i > u_i \\ (t_i - l_i)^2 & \text{if } t_i < l_i \\ 0, & \text{otherwise} \end{cases}} \tag{2.13}$$

The lower-bounding function, despite being a big step in improving the performance of DTW-based retrieval does not reduce the complexity of the original DTW, and depending on its tightness a considerable execution number of DTW meadures will be necessary.

Other proposed methods tries to speed-up the DTW based on the idea to perform the alignment on low resolution versions $\overline{Q}$ and $\overline{T}$ of the sequences Q and T and then scale up to full resolution cost matrix, in order to find the optimal warping path.

Keogh and Pazzani in [KP00] presented the algorithm, Piecewise Dynamic Time Warping (PDTW), a modification of DTW that operates on the PAA reduced dimensionality representation. They show that the resulting alignments are very similar to those produced by naïve DTW, with no significant loss of accuracy for classification and clustering tasks, although, it effectively generates a speedup of one to three orders of magnitude as ilustrated in Figure 2.15.

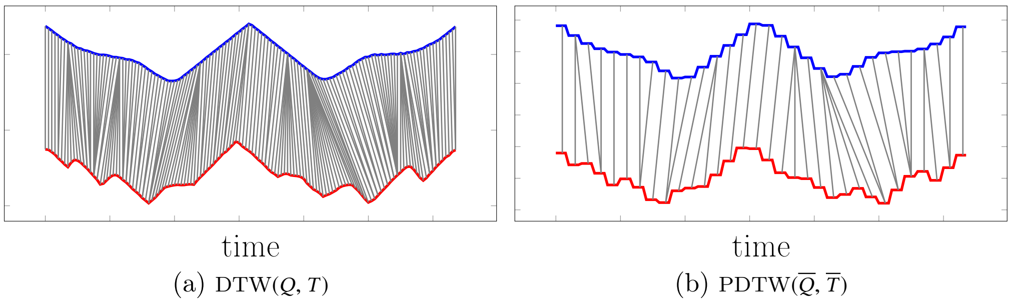


Figure 2.15 – (a) Two time series Q and T and the alignment between them, as discovered by naïve DTW. (b) The same time series in their PAA representation, and the alignment discovered by PDTW.

Salvador and Chan in [SC07] propose a method of multi-scale analysis, called Fast DTW for dynamic alignment. The FastDTW [SC07] algorithm uses ideas from both the constraints and abstraction approaches and provides optimal or near-optimal alignments with an $O(|Q| + |T|)$ time and memory complexity, in contrast to the $O(N_Q \cdot N_T)$ requirement for the naïve DTW algorithm.

It first searching for the optimal path for sub-sampled versions of the Q and T, and recursively projects a solution from a coarser resolution and refines the projected solution.

FastDTW algorithm solely comprises of following three main phases:

Coarsening: reduces the length of a time series by averaging adjacent pairs of points. The resulting time series is a factor of two smaller than the original time series. It is called successively times until the base condition (the lowest dimension given by a user parameter) is met, producing different low-level resolutions for time series. Figure 2.16 shows four resolutions that are created when running the FastDTW algorithm.

Projection: DTW is run to find an optimal warp path for the lowest resolution (Figure 2.16a) and determines what cells it passes through in a higher resolution. Since the resolution is increasing by a factor of two, a single point in the low-resolution warp path will map to at least four points at the higher resolution (Figure 2.16b). This projected path is then used as a heuristic during refinement to find a warp path at the higher resolution.

Refinement: finds an optimal warp path in the neighborhood of the projected path. To refine the projected path, a constrained DTW algorithm is run with the very specific constraint that only cells in the projected warp path plus additional neighborhood cells on each side are evaluated (Figure 2.16d). The size of the neighborhood is controlled by the radius parameter.

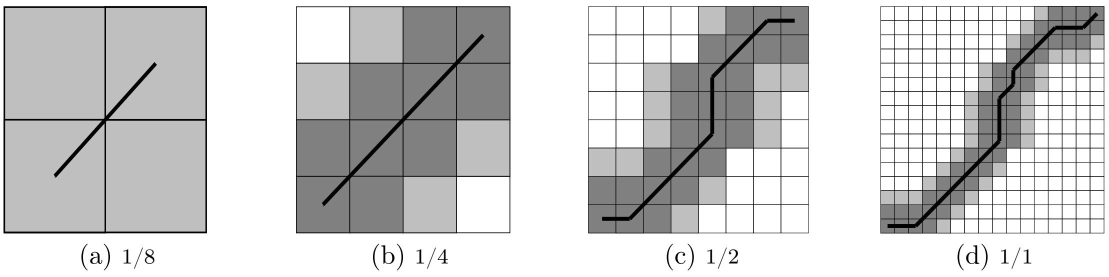


Figure 2.16 – The four different resolutions evaluated during the execution of FastDTW algorithm. Dark gray cells correspond to cells that need to be evaluated, according to the path discovered on the previous scale. The light gray cells are neighborhoods add by the radius parameter to mitigate the possibility of missing an optimal path. Adapted from [SC07]

We can note that FastDTW gradually shrinks the warping band by repeating the projection and refinement steps. That is, FastDTW dynamically reduces the warping band while the traditional bands, like Sakoe-Chiba, are not changed.

As previously discussed, naïve DTW is an $O(N^2)$ algorithm, once it is necessary to fill all cells in the cost matrix to ensure an optimal answer is found, and the size of the matrix grows quadratically with the size of the time series. On the other hand, in multilevel approaches, like FastDTW, the cost matrix is only filled in the neighborhood of the path projected from the previous resolution. In this way, the multilevel approach is an $O(N)$ algorithm, since the length of the warp path grows linearly with the size of the input time series.

It should be noted that due to its recursive construction, the FastDTW algorithm presents a big constant value, which degrades the execution time. For example, in some scenarios (small-length time series, high ratio between radius parameter and the time series length), its performance is lower than that obtained by naïve DTW [SC07; GSZ11].

In addition to those previously listed, the DTW-based metric presents another big issue: unlike in Euclidean space where an average method is easily defined, in the DTW-based space defining a average method is harder.

The lack of a well-defined method for averaging impacts in a wide variety of distance-based learning algorithms, once they require at their core an averaging method, and highly depend on the quality of this averaging. For example, the well-know k-means algorithm [Mac67] needs to average a set of objects, by finding a mean of the set, repeating this kind of operation until converging. However, in the DTW-based space it fails, as the arithmetic mean does not take time warping into account, as observed in Figure 2.17a.

Most of averaging methods proposed in the literature are based on pairwise averaging algorithms. Such strategies are really dependent of the order in which the series are taken into account with no guarantee to obtain the same results with a different order [NR07]. Because of such a limitation, recently two kinds of global approaches have been proposed: The DTW Barycenter averaging (DBA) and the Soft-DTW. The fundamental advantage of those methods is that the time series are averaged all together, and therefore there is no impact on the order of consideration of the time series.

The DTW Barycenter Averaging (DBA), presented in [PKG11], was the first successful algorithm able to handle an averaging method for DTW. It consists of a heuristic strategy (finding the optimal multiple alignment is a NP-Complete problem), designed as a global averaging method and can be summarized as: given an (potentially arbitrary) initial sequence, refine it, iteratively, in order to minimize its squared distance (DTW) to averaged sequences. This refinement if done coordinate by coordinate, therefore, each coordinate of the averaged sequence is the barycenter of the associated coordinates of the set of series.

The Soft-DTW [CB17] is defined by the authors as a “Differentiable Loss Function for Time-Series”. DTW defines the similarity between two time series as their minimal alignment cost. Soft-DTW proposes to replace this minimum by a soft minimum. In other words, Soft-DTW will consider all possible alignments and retrieve the soft-minimum of the distribution of all costs spanned by all alignments.

Just like naïve DTW, Soft-DTW is quadratic in time and space complexity (it is necessary to reaffirm that when there is no need to preserve the optimal warping path, the DTW algorithm becomes linear in space). However, the main advantage of Soft-DTW arises from the fact that it is a differentiable loss function, and its gradient can also be computed in quadratic time.

The algorithm depends on a smoothing parameter. When this smoothing parameter is set to 0, the original DTW distance is recovered. At the opposite, when this smoothing parameter tends to infinity, soft-DTW converges to the sum of all costs.

Some drawbacks of the Soft-DTW is unlike other distances, it can return negative values, there is no guarantee that $\mathrm{softDTW}(T, Q) = 0$ and it does not fulfill the triangle inequality, however, it is always symmetric.

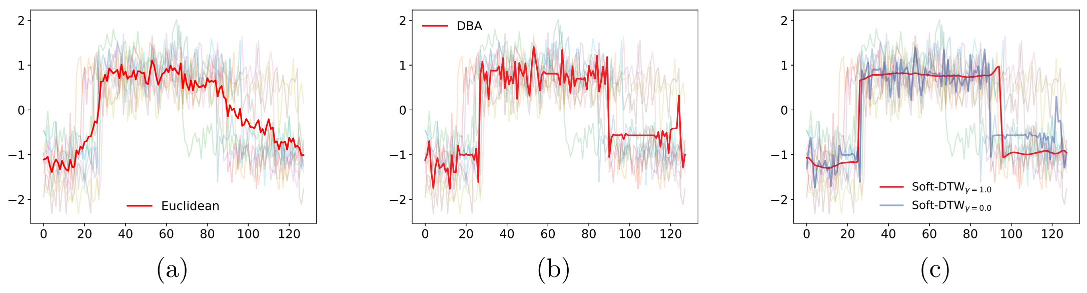


Figure 2.17 – Considering the CBF dataset from [CKH+15]. (a) The barycenter defined by the traditional arithmetic mean produces a spurious relation. (b) The average sequence of the cluster using the DBA approach. (c) In Soft-DTW, we can observe the effect of the smoothing parameter. Without it, which corresponds to the DBA case, the obtained curve is very granular (red) and it becomes smoother as the smoothing parameter increases (blue).

Multivariate Approaches for DTW

For a long time, most time series studies were focused on one-dimensional ones, sometimes neglecting generalization for multivariate series. However, thanks to the widespread of affordable motion sensor devices, data-mining-related tasks on multivariate time series have become increasingly popular.

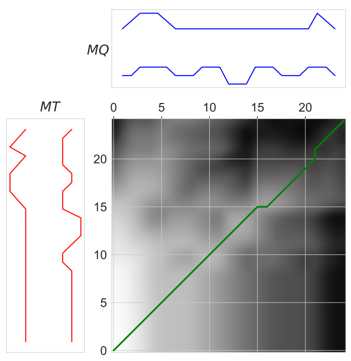
The generalization of the DTW distance (Equation (2.10)) for the multivariate case falls in one of two ways, dependent or independent warping path [SHJ+17], and the choice of the best approach is not trivial and depends of the nature on the data.

Both extensions are simple and involves minimal changing in the original algorithm. In independent path DTW ($\mathrm{DTW_i}$), the DTW distance is the cumulative distance of all independently measured DTW for all dimensions. Considering two M-dimensional multivariate time series Q and T, and assuming that T^m is the m-channel of T, we can define $\mathrm{DTW_i}$ as:

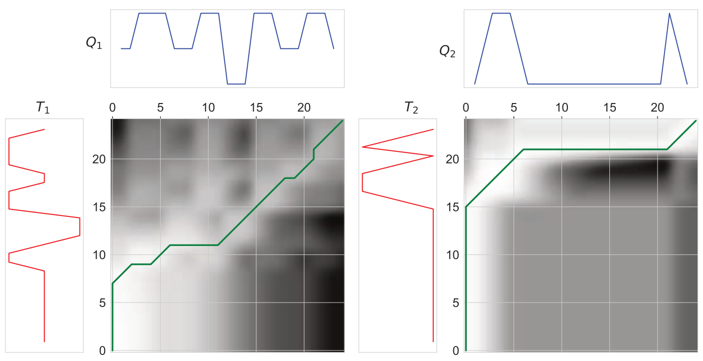
$$\mathrm{DTW_i}(\mathrm{Q}, \mathrm{T}) = \sum_{m=1}^{M} \mathrm{DTW}(\mathrm{Q}^m, \mathrm{T}^m) \tag{2.14}$$

In the other hand, the dependent path DTW ($\mathrm{DTW_d}$) is a direct modification of the local cost function δ.While in univariate case the distance function δ is related to single data points, in $\mathrm{DTW_d}$ it is, instead, the distance of M data points.

As demonstrated by Shokoohi-Yekta, Wang, and Keogh in [SWK15], the choice between $\mathrm{DTW_d}$ and $\mathrm{DTW_i}$ not only produce different values for the measured distance, but also it impacts on data-mining-related tasks, especially on classification and clustering.



(a) $\mathrm{DTW_d}$



(b) $\mathrm{DTW_i}$

FIGURE 2.18 – For two multidimensional time series T and Q. (a) The $\mathrm{DTW_d}(\mathrm{Q}, \mathrm{T}) = 0.559$. (b) The $\mathrm{DTW_i}(\mathrm{Q}, \mathrm{T}) = \mathrm{DTW}(\mathrm{Q}^1, \mathrm{T}^1) + \mathrm{DTW}(\mathrm{Q}^2, \mathrm{T}^2) = 0.214 + 0.0 = 0.214$. In both (a) and (b), regions of low cost are indicated by light colors, and regions of high cost are indicated by dark colors.

Finally, both versions of multivariate DTW have a $O(M \cdot N_Q \cdot N_T)$ computational complexity, where $|\mathrm{Q}| = N_Q$ and $|\mathrm{T}| = N_T$, respectively. The space complexity of DTW_d is $O(N_Q \cdot N_T)$ when we want to retain the optimal warping path or $O\left(\min(N_Q, N_T)\right)$ otherwise. In its turn, for DTW_i we have M warping path computations, consequently, the space complexity becomes $O(M \cdot N_Q \cdot N_T)$ when preserving all optimal warping paths or $O\left(M \cdot \min(N_T, N_Q)\right)$ otherwise.

2.3.1.3 Summary on DTW

DTW is a kind of *Swiss Army knife* used to measure the (dis)similarity between time series, it is a simple and elegant algorithm able to deal with temporal shifts and distortions.

However, as increasing the dimensionality of the time series in the temporal or spatial axis, the performance degrades due to its $O(N^2)$ time complexity. In addition, the naïve DTW can produces unexpected unwanted matches.

Thus, many speed-up techniques have been proposed to reduce, or at least, mitigate the quadratic complexity of DTW.

If on the one hand, global constraints or can mitigate the time complexity by reducing from $O(N^2)$ to $(O(N \cdot r)$, it remains quadratic, since r determines only a region proportional to N. On the other hand, it does not make DTW less sensitive to noise, outliers, nor make it a metric.

Lower bound functions are also helpful to deal with the time complexity, yet it can introduce an overhead, specially due to the *tightness* × complexity factor.

Other approaches are designed for specific scenarios, introducing extra parameters that need to be well tuned in training data.

However, even though all research, the quest for processing extremely large collections of time series is still active.

2.3.1.4 Longest Common SubSequences

The Longest Common SubSequence (LCSS) is proposed to handle time series data that may contain possible noise [EA12]. This noise could be introduced by hardware failures, disturbance signals, and transmission errors. The intuition of LCSS is to remove the noise effects by only counting the number of matched elements between two time sequences.

The LCSS score between two time series R and Q of lengths N_R and N_Q, respectively, is defined as:

$$\mathrm{LCSS}(R_{:i}, Q_{:j}) = \begin{cases} 0, & if\, N_R = 0\, or\, N_Q = 0. \\ \mathrm{LCSS}(R_{:i-1}, Q_{:j-1}) + 1 & if\, \delta(r_1; q_1) \leq \epsilon \\ \max\left(\mathrm{LCSS}(R_{:i-1}, Q_{:j}), \mathrm{LCSS}(R_{:i}, Q_{:j-1})\right), otherwise. \end{cases} \tag{2.15}$$

In the definition of LCSS, instead of using distance, *score* is used. Thus, the higher the score, the more similar are two time series. The LCSS score can be converted into distance using the following formula:

$$\mathrm{D}_{\mathrm{LCSS}} = 1 - \frac{\mathrm{LCSS}(R, Q)}{\min(N_R, N_Q)} \tag{2.16}$$

The matching threshold ϵ is a hyper-parameter which needs to be set up by the user to determine whether two elements match. LCSS is an alignment method that can handle outliers and temporal shifts, because the matching threshold quantizes the distance between two elements to two values, 0 and 1, which removes the larger distance effects caused by noise. However, LCSS does not differentiate time series with similar subsequences but various *gap* differences between its similar subsequences, which leads to inaccuracies. Also, like DTW it is not a metric and has a quadratic complexity.

2.3.1.5 Summary on Distance Functions on Raw Representation

Numerous similarity measures for raw time series exist, we detailed in this manuscript a small selection of them. Each one of them can be used for the retrieval task. A comparison of their characteristics can be found in Table 2.1.

Although robust enough to handle distortions in the time-axis, the choosing DTW or LCSS over L_p-norm is not an easy choice. The speed of the L_p-norm is often a sufficient condition to accept it losses in accuracy, especially when dealing with huge databases.

Despite being robust to handle outliers, the LCSS does not differentiate time series with similar subsequences, restricting its use to scenarios where there is a paramount need to deal specifically with outliers. In its turn, the DTW has as a significant setback its computational cost, and complexity of execution and numerous variations have been proposed to mitigate this issue. These modifications do not wholly solve DTW issues, and often need to be adjusted according to dataset-dependent parameters. Finally, the L_p-norm, and specifically, the Euclidean distance often requires specific preprocessing for the dataset, rising costs, but without achieving the same robustness as other functions.

Table 2.1 – Comparison among three distance functions on raw representations of timeseries data.

Distance Function	Handle $\|Q\| \neq \|T\|$	Local Time Shifting	Noise	User-defined matching threshold	Computational complexity	Metric
L_p-nom	No	No	No	No	$O(N)$	Yes
DTW	Yes	Yes	No	No	$O(N^2)$	No
LCSS	Yes	Yes	Yes	Yes	$O(N^2)$	No

Besides raw format representation, there are several other representations, together with distance functions, to measure the similarity between time series. These representations are proposed either for particular applications or dimensionality reduction. Next, we will discuss the most commonly used representations.

2.3.2 Distance Functions on Other Representations

The similarity measure can be performed on both raw and transformed time series. The most natural representation of time series is the raw representation, i. e. the ordered list of valued data points. However, there are many tasks for which a representation that highlights some specifies of time series is the most suitable one, e. g. frequency for speech recognition. Figure 2.19 shows some of the most common representations for time series data.

Due to the wide range of possible applications for time series data, numerous representations have been proposed by the time series community from Fourier and Wavelet Transformations to Shapelet-based Representation, through Piecewise Approximations. In the following, we focus on the most well-known time series representations.

2.3.2.1 Frequency Transformation Representation and Distance Functions

Numerous Frequency domain transformation methods have been proposed in order to reduce the dimensionality of time series, we detail here the most relevant ones.

The frequency domain transformations aim to move from the time-based domain to the frequency-based domain [EA12]. The two most used are the Fourier Transformation and the Discrete Wavelet Transformation. While Fourier transformation represents a time series as a sum of sinusoidal functions, wavelet transformation approximates a time series by a set of orthonormal representations. Both, Discrete Fourier Transformation (DFT) [FRM94; AFS93] and Discrete Wavelet Transformation (DWT) [CF99] reduce the dimensionality of the time series and have the property to be a lower-bounding function to the ED on the original space.

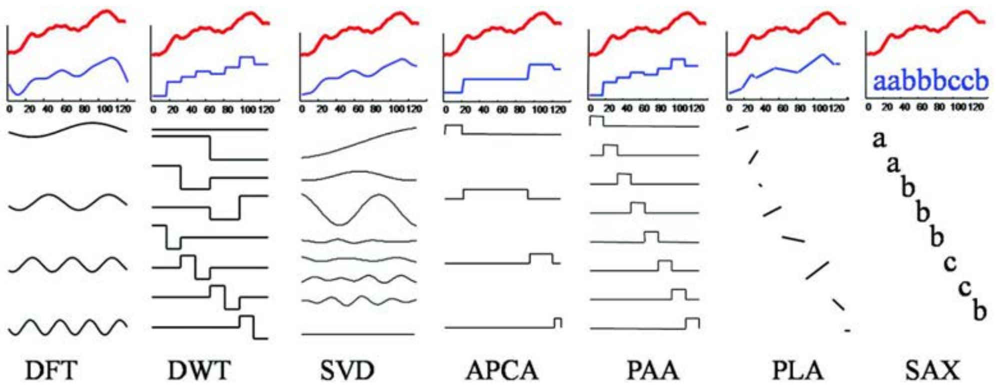


Figure 2.19 – Example of techniques that can significantly reduce the dimensionality of time series. For each transformation approach is represented the raw time series T (red), its representation $\overline{T}$ (blue) and the transformation components (bottom). Adapted from [RLG+10].

- **Discrete Fourier Transformation**: The first technique suggested for dimensionality reduction of time series was the DFT. The DFT is based on the idea of spectral decomposition, where a time series (or signal) T_c of length N, no matter how complex, can be represented as a linear combination of $\frac{N}{2}$ cosines and sines waves, and these ones, represented by a single complex number know as a Fourier coefficient, can be recombined into the original signal.

  Nonetheless, many of the Fourier coefficients have very low amplitude and thus have a small contribution to reconstructed signal. These low amplitude coefficients can be discarded without significant loss of information thereby reducing the dimensionality of the data and saving storage space. The Figure 2.20 shows the DFT approximation of the IBM stock price time series. It is evident that increasing the number of coefficients the DFT approximation gets better.

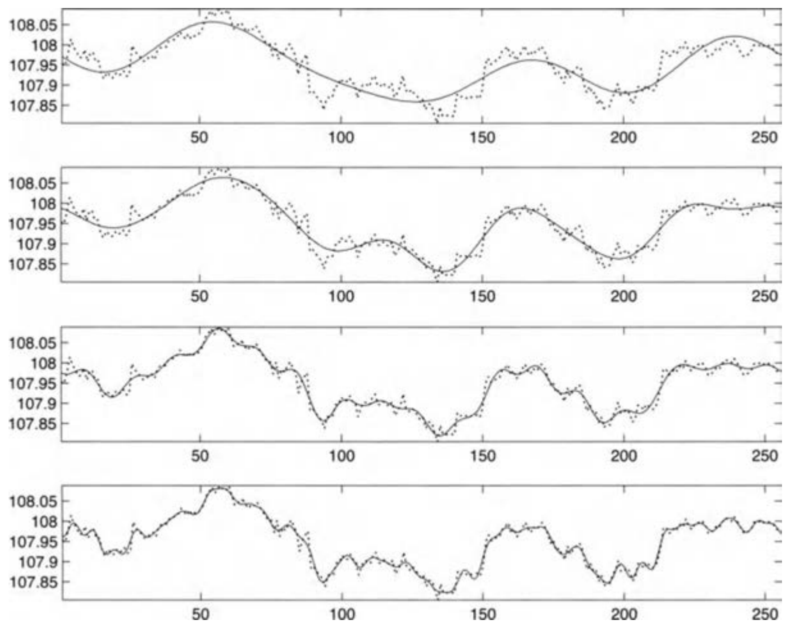


Figure 2.20 – Approximation of IBM stock price time series with a DFT. From top to bottom, the time series is approximated by 10, 20, 40 and 80 DFT coefficients, respectively. Extracted from [SZ04].

- **Discrete Wavelet Transform**: Wavelet bases are able to represent both frequency and location information inside a time series. It represents the data in terms of the sum and difference of a prototype function, so called the "analyzing" or "mother" wavelet. The main difference to the DFT is while Fourier coefficients always represent the contribution of the all data points in the time series, in the wavelets they are localized in time, i. e. some wavelet coefficients represent small, local subsections of the data being studied [RLG+10].

The first proposed DWT is the Haar wavelet, which mother wavelet corresponds to a sequence of continuous square-shaped functions that approximates the time series. DWT supports the multi-resolution analysis of the data, while first few coefficients contain an overall, coarse approximation of the data, the extra additional coefficients can be imagined as "zooming-in" to areas of high detail.

2.3.2.2 Piecewise Aproximative Representations and Distance Functions

Numerous Piecewise Aproximations methods have been proposed in order to reduce the dimensionality of time series, we detail here the most relevant ones.

- **Piecewise Linear Approximation**: The main idea of Piecewise Linear Approximation (PLA) [PH74] is to split the series into most representative segments, and then fit a polynomial model for each segment. Three basic approaches are distinguished [EA12]:

  i) Sliding windows: a segment is grown until it exceeds some error threshold;

  ii) Top-down approach: it consists in recursively partitioning a time series until some stopping criterion is met;

  iii) Bottom-up approach: it consists in starting from the finest approximation, sucessively new segments are iteratively merged.

  In [PH74], they introduced an algorithm which allows for a viable number of segments. However, an open question is how to best choose k, the "optimal" number of segments used to represent a particular time series [RLG+10].

- **Piecewise Aggregate Approximation**: Yi and Faloutsos and Keogh *et al.* independently introduced the Piecewise Aggregate Approximation (PAA) [YF00; KCPM01] as a dimensionality reduction technique for time series indexing.

  The idea is to approximate a time series by dividing it into equal-length segments and recording the mean value of the data points that fall within each segment. This representation reduces the data from N dimensions to w dimensions, $1 \leq w \leq N$ by dividing the time series into w "frames". The time series is represented by a vector $\overline{T} \in \mathbb{R}^w$, where the i^{th} element of $\overline{T}$ is calculated by the following equation:

$$\bar{t}_i = \frac{w}{N} \sum_{j=\frac{N}{w}(i-1)+1}^{\frac{N}{w}i} t_j \tag{2.17}$$

  The average value of the data included in a frame is calculated, and a vector of these values becomes the reduced representation of the data. When $w = N$, the transformed representation is identical to the original representation. When $w = 1$, the transformed representation is simply the mean of the original sequence. The main advantages of PAA are the interpretability and the simplicity of the method, PAA is a compression-based technique.

- **Adaptive Piecewise Constant Approximation**: Chakrabarti *et al.* introduced an extension to the PAA representation, the Adaptive Piecewise Constant Approximation (APCA) [CKMP02]. The difference to the original PAA is that this representation allows the segments to have arbitrary lengths. APCA has two parameters for each segment: its value (the mean of the segment) and its length. APCA is thus more flexible than PAA leading to a better approximation of time series.

2.3.2.3 ARIMA Model and LPC Cepstral Coefficient Representation and Distance Function

In statistical analysis of time series data, a time series is considered to be a sequence of observations of a particular variable. In this way, a time series is compound of four components: a trend, a cycle, a stochastic persistence component, and a random element. The four components can be captured by some Auto-regressive (AR) model, like the Auto-Regressive Integrated Moving Average (ARIMA) model. Therefore, the similarity between time series data can be modeled by the similarity between two corresponding ARIMA models. The advantage of using this model is that it requires a few fixed number of parameters compare regardless of the lengths of time series.

Autocorrelation functions (ACFs) and partial autocorrelation functions (PACFs) are usually used to determine models to which time series data belong. Thus, time series with similar ACFs or PACFs are likely to belong to the same statistical model., and therefore similar. Wang and Wang in [WW00] propose to use ACFs and PACFs to represent time series data and measure similarity between two ACFs or PACFs using Euclidean distance. If the distance is smaller than a predefined threshold, two ACFs or PACFs will have the similar shapes or movement patterns, and the two time series represented by ACFs or PACFs are similar.

Kalpakis, Gada, and Puttagunta in [KGP01] propose using Euclidean distance between Linear Predictive Coding (LPC) centrums of two ARIMA models to measure the similarity between two time series data. They prove experimentally that using LPC centrum is more effective and efficient than applying DFT, DWT, and PCA on ARIMA models in tasks that involve clustering of time series data.

2.3.2.4 Symbolic Representation and Distance Function

The symbolic representations aim to further reduce the dimensionality of the time series by discretization of the time series into a sequence of symbols. Lin *et al.* introduced the Symbolic Aggregate approXimation (SAX) [LKWL07; SK08] method that converts a time series into a symbolic form representation. The SAX representation has been shown to preserve meaningful information from the original data and produce competitive results for classifying and clustering time series [EA12; RLG+10]. It relies on the PAA to produce the symbolic sequence. In order to transform a time series, SAX proceeds in three steps:

i) The time series is normalized to zero mean and unit standard deviation;

ii) The time series is segmented using a PAA representation;

iii) The PAA representation is finally quantized: a discretization step maps each real value into a symbol. Over the whole series, each symbol is given the same probability of appearance. This step is based on the hypothesis of a normal distribution of the values over along the time series.

SAX converts a time series into a sequence of symbols of length W, Figure 2.21 illustrates the SAX representation. The alphabet of symbols has a length $\alpha > 2$, $\alpha \in [1, \ldots, W]$. The number of symbols is to be determined by the user.

2.3.2.5 Shapelet-based Representation and Distance Function

All previously listed approaches are designed for transforming time series into a new domain for uncovering global similarity in shape. However, the shape-based similarity is not always

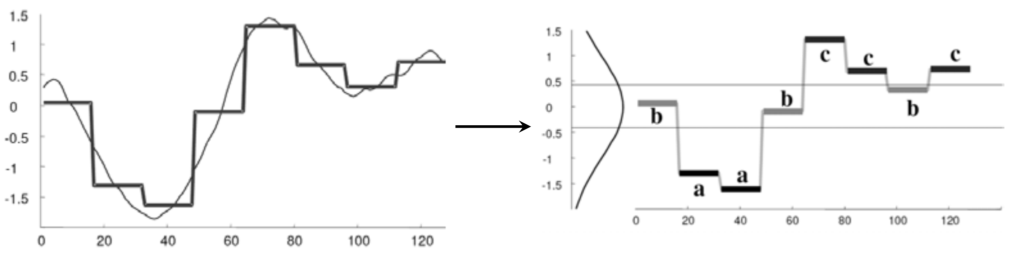


FIGURE 2.21 – An example of a time series being converted first to its PAA representation then to the string baabccbc. Adapted from [RLG+10].

global. For example, consider an Electrocardiography (ECG) heartbeat for a patient where a single beat arrhythmia is indicative of a heart condition. If this were captured as a time series and compared to a series of normal ECG behavior, it would be challenging to detect a difference because of the presence of many regular heartbeats.

The discriminatory feature, in this case, would be described by the presence of a small local shape within the series indicating an irregular beat, which would likely be missed in the frequency and time domains as the structure and global shape of the data would still be very similar. We consider extracting time series small representative subsections for detecting local shape-based similarity between series.

This small representative subsections, know as shapelets, have been a prominent area of time series data-mining research since its proposition in [YK09], and numerous approaches are proposed to expand its application, to speed up its extraction or even to redefine how to obtain it.

In the following subsections, we will present an overview of the state-of-art in time series shapelets, from its original definition to the DTW-preserving shapelet transformation.

2.3.2.6 Shapelets Tree

Shapelets Tree is the original approach for Time Series Shapelet [YK09; YK11]. In this first work, shapelets are used to build a decision tree classifier by recursively searching for a discriminatory subsequence on the entire data set. This subsequence is used to split the data into branches.

Basically, all subsequences $\mathcal{S}^*_{\mathcal{T}}$ in $\mathcal{T}$ are potential candidates to be elected a shapelet. However, to reduce the number of possible candidates it is convenient to define two limits: MINLEN and MAXLEN, respectively the minimum and maximum length of the candidates. In this way, considering $|T_i| = N_i$, the number of candidates can be defined as:

$$\sum_{l=\text{MINLEN}}^{\text{MAXLEN}} \sum_{T_i \in \mathcal{T}} (N_i - l + 1). \tag{2.18}$$

In the brute-force approach, for each candidate, the algorithm calculates the subsequence distance to all time series in $\mathcal{T}$, measured by the normalized Euclidean distance. Then, a histogram H ordered by the values of subD is created, as defined by Equation (2.19).

$$H[S_i] = \text{sort}\left(\text{subD}\left(S_i, T_j\right) \ \forall T_j \in \mathcal{T}\right) \forall S_i \in \mathcal{S} \tag{2.19}$$

The idea with this distance-ordered histogram is that for a "good" subsequence, most of the time series in one class of the dataset are close to the subsequence under subD, while most of the time series objects from the other class are far away from it, as drawn in Figure 2.22.

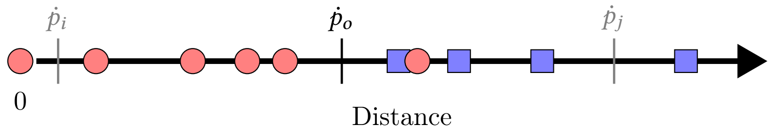


FIGURE 2.22 – A "good" subsequence allows discriminating between classes. The optimal splitting point $\dot{p}_o$ and two other possible splitting points are represented.

Therefore, some metric is necessary to evaluate how well the selected shapelet can divide the entire combined dataset into two classes. The authors proposed to use the Information gain (IG) [Qui86]. Thus, a shapelet is the subsequence which for a given optimal split point produces the highest IG, so, building a decision three using shapelets needs two factors: a shapelet and the corresponding optimal split point as showed in Figure 2.23.

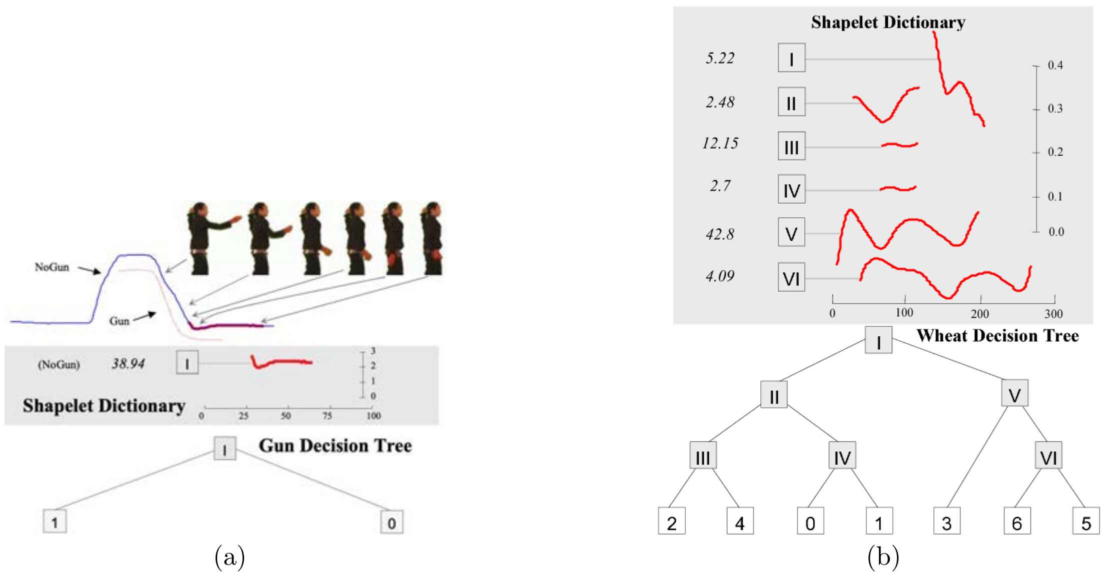


FIGURE 2.23 – Two examples of shapelet-based decision tree:(a) the dictionary of shapelets, with the thresholds is used to build a decision tree for the `Gun_point` dataset [CKH+15]. In (b) a multilevel tree need to classify the `Wheat` dataset [CKH+15]. Adapted from [YK11].

This naïve brute force algorithm is extremely space inefficient, requiring the storage of all subsequences candidates. Besides that, the algorithm suffers from high time complexity. Considering $|\mathcal{T}| = K$, and the average length of each time series $T_i \in \mathcal{T}$ as $\bar{N}$, the size of the candidate set is $O(K\bar{N}^2)$. Evaluating one candidate requires its comparison with $O(K\bar{N})$ subsequences, as each comparison is based on the ED, it takes $O(\bar{N})$ on average. Hence, the overall time complexity of the algorithm is $O(K^2\bar{N}^4)$, which makes its usage for real-world problems intractable.

Thus, the majority of shapelet research has focused on mitigate the performance drawback of shapelet discovery. Staying in Ye and Keogh [YK09] approach's, some smart optimizations were proposed. One is to reduce the cost of the subsequence distance computation using early abandon in the brute force approach, and the other is the early entropy pruning.

The re-usage of computations and pruning of the search space was proposed by Mueen, Keogh, and Young [MKY11] in the work "Logical-shapelets: an expressive primitive for time series classification".

This work defines Logical-shapelets as an adaptation to the shapelet tree algorithm aiming to combine shapelets to form more complex rules to better handle challenging separating problems. It uses a statistics caching optimization to speedup search for shapelets, and extends the original work by introducing an augmented, more expressive shapelet representation where a combination of shapelets is used in conjunction with each other for determining the class separation.

Figure 2.24 reproduces one of the original motivating examples which was presented in the original work. In the diagram it is represented a two-class problem where each class has two time series. The red circle class, contains two sinusoidal patterns with both positive and negative peaks, while the blue square class has only one positive or one negative peak. The classical shapelet definition is not powerful to discriminate these classes, and there is no way to do so, due to its absence of expressiveness to represent this concept.

On the other hand, combining those two sequences allow discriminating between the classes, and this is one relevant contribution of this work, it was the first to propose to combine shapelets.

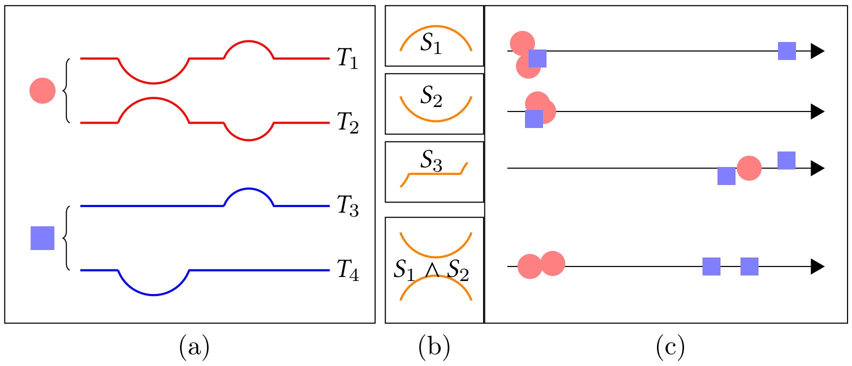


Figure 2.24 – (a) Two classes of synthetic time series. (b) The shapelet dictionary with three single shapelets and one pair connected using an *and* operation. (c) A good separation is done only using the combination of two shapelets. Adapted from [MKY11]
.

Keogh and Rakthanmanon, in [KR13], propose the *Fast Shapelet*, improving the efficiency of the original shapelets algorithm by exploiting the projections into a SAX-based symbolic representation [LKWL07]. The algorithm employs a number of techniques to speed up the finding and pruning of shapelet candidates. Using the SAX representation, the algorithm aims to reduce the length of the time series as well smoothing the data. The use of SAX words allows a discretization of the data, and support the use of a collision table metric to speed-up the pruning of shapelet candidates. This collision table metric is highly correlated with the IG, reducing the amount of work performed in the quality measure stage.

Other speed-ups have also been attempted by using hardware-based implementations, such as the usage of the processing power of Graphics processing unit (GPU) for boosting the search time [CDHR12]. Furthermore, the training time has been reduced by mining infrequent shapelet candidates [HDZ+12].

Another way to speed-up the shapelet discovery is based on alternative approaches for the quality measure. Instead of the IG, Lines *et al.*, in [LDHB12], propose the F-stats quality criterion, and Lines and Bagnall, in [LB12], propose to use either the Kruskal-Wallis test or the Mood's Median test. They allege that while the classification performances are not significantly different, the time required for the shapelet discovery can be reduced by $\approx 18\%$, in average, in comparison with the IG, due to the lower number of computations required by these tests. Nevertheless, scalability remains the bottleneck.

The use of shapelets for other tasks than classification, started with the Unsupervised Shapelets – u-shapelets work in [ZMK12; UBK15; ZMKY16]. Designed for time series clustering, the algorithm extracts the *u-shapelets*, which are subsequences that can divide the time series dataset into well separated groups.

Considering $\mathcal{T}_A$ and $\mathcal{T}_B$ two subsets for $\mathcal{T}$, such that $\mathcal{T}_A \cap \mathcal{T}_B = \emptyset$, basically, an u-shapelet S' is a subsequence of a time series $T \in \mathcal{T}$ for which the $\mathrm{subD(S', T_a)} \ll \mathrm{subD(S', T_b)}$ for all $T_a \in \mathcal{T}_A$ and $T_b \in \mathcal{T}_B$.

Inspirited in the idea of increasing the Information Gain from the shapelet tree approach, here the goal is to maximize the separation gab between two subsets of $\mathcal{T}$. In essence, this algorithm can be seen as a greed search algorithm.

The separation measure is formally defined as:

$$gap = \mu_{\mathcal{T}_B} - \sigma_{\mathcal{T}_B} - \left(\mu_{\mathcal{T}_A} - \sigma_{\mathcal{T}_A}\right) \tag{2.20}$$

where, $\mu_{\mathcal{T}_x}$ and $\sigma_{\mathcal{T}_x}$ represents, respectively, the mean value and the standard deviation for $\mathrm{subD}\left(S', T_x\right)$ $\forall\, T_x \in \mathcal{T}_x$.

Clustering time series with u-shapelets has several advantages. Firstly, u-shapelets clustering is defined for datasets in which time series have different lengths, which cannot be handled by most techniques described in the literature. Indeed, in many cases, the equal length assumption is a requirement, and the trimming to equal length is done by exploiting expensive human skills [UBK15]. Secondly, u-shapelets clustering is much more expressive regarding representational power. Indeed, it allows clustering only time series that can be clustered and do not cluster those that do not belong to any cluster [FNV18].

2.3.2.7 Shapelet Transform

One major drawback of the previous approaches is the lack of flexibility in learning and classification due to its integration in just one task. Whilst decision trees are highly interpretable, they are often outperformed by other classifiers and have a tendency to over-fit unless post-pruned or used with a conservative stopping condition. Additionally, the recursive nature of the decision tree algorithm means that the relatively time-consuming shapelet detection method is called repeatedly. Hence, the number of papers proposing the optimization of the shapelet extraction task.

Moreover, Bagnall *et al.* in [BDHL12] demonstrate the importance of separating the transformation from the classification algorithm with an ensemble approach, where each member of the ensemble is constructed on a different transformation of the original data. Thus, a separation between the shapelet extraction, which can be understood as feature discovery, and the learning decision can allow the use of specific learner designed to perform the classification.

Proposed by Lines *et al.* in [LDHB12] and further expanded upon in [HLB+14], the Shapelet Transform (ST) algorithm disconnect the process of finding the shapelets from the step of building the decision tree from original shapelet tree algorithm.

As seen previously, the original shapelet tree approach embeds the shapelet discovery algorithm in a decision tree, and use Information gain (IG) to evaluate the quality of candidates. For each node of the tree a shapelet is found by a exhaustive enumerative search, recursively subdividing the data. This results in the brute force search being performed a number of times, which makes it intractable on large problems.

Shapelet Transform (ST) is a first step towards the generalization of the shapelet principle. Instead of discovering iteratively the shapelets at each node of a shapelet-tree, it uses a single scan algorithm to finds the top-k shapelets in terms of quality measure (i. e. IG) in $\mathcal{T}$.

The transformation shapelet process is divided in three main stages:

1. The algorithm perform a single scan in $\mathcal{T}$, to extract the best k shapelets, where k is a cut-off value for determining the maximum number of shapelets to store, without any effect on the quality of the individual shapelets candidates.

2. The set of the top-k shapelets can be reduced, either by clustering the k shapelets or ignoring the elements below a cut-off point.

3. Using the top-k extracted shapelets, the algorithm creates a dictionary with k entries and use them to transform the original raw time series T, with $|T| = N$ into a vector $\overline{T}$, with $|\overline{T}| = k$, whose components represent the subD between the time series and the shapelets, as illustrated in Figure 2.25.

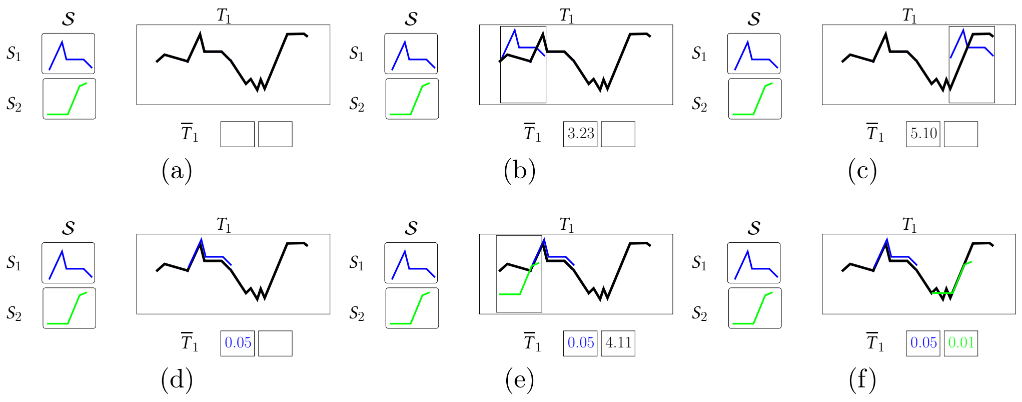


FIGURE 2.25 – Example of windowing for shapelet transform with $|\mathcal{S}| = 2$. (a) Starting with the shapelet S_1 and the time series T_1, the algorithm build the first component of the vector $\overline{T}_1$. S_1 slides over T_1 from begin (b) to the end (c). The subD is then used to build $\overline{T}_{1,S_1}$ (d). The same is done for S_2, starting from begin of T_1 (e) and going to the end. Finally, the subD is used to build $\overline{T}_{1,S_2}$.

Transforming a time series into its vectorial representation requires to slide each shapelet against that time series in order to compute and find the best matching location. The cost of the shapelet transform is, therefore, highly dependent on the number of shapelets used to create vectorial representation.

Another observed problem stems from the fact that each shapelet must be highly discriminant independently of the others. However, many of the shapelets generated by the transformation are similar to one another. That similarity reduces the interpretability of the data and can put down the accuracy of the classifier if it is not robust to handle correlated attributes.

To mitigate this phenomenon, Hills *et al.* propose a post-transform clustering procedure to group similar shapelets, without substantially reducing the accuracy of the classifier [HLB+14].

The transformation process is defined in the Algorithm 2.2.

Algorithm 2.2 The Shapelet Transform algorithm.

```
1:  input: $\mathcal{S}$ {Set of shapelets}, $\mathcal{T}$ {Time series dataset}
2:  output: $\overline{\mathcal{T}}$ {Transformed dataset}
3:  $\overline{\mathcal{T}} \leftarrow \emptyset$
4:  for $T_i \in \mathcal{T}$ do
5:      $\overline{T} \leftarrow \emptyset$
6:      for $S_k \in \mathcal{S}$ do
7:          $M \leftarrow$ subD$(T_i, S_k)$
8:          $\overline{T} \leftarrow \overline{T} \bigcup \{M\}$
9:      end for
10:     $\overline{\mathcal{T}} \leftarrow \overline{\mathcal{T}} \bigcup \{\overline{T}\}$
11: end for
12: return $\overline{\mathcal{T}}$
```

It is carried out using the subsequence distance calculation, and the resultant Euclidean shapelet match between a shapelet $S_k \in \mathcal{S}$ and a time series $T_i \in \mathcal{T}$ is defined as:

$$M_{i,k} = \text{subD}(T_i, S_k) \tag{2.21}$$

Then, each time series $T_i \in \mathcal{T}$ is represented by a vector $\overline{T}_i \in \mathbb{R}^{|\mathcal{S}|}$, where each k^{th} component of $\overline{T}_i$ corresponds to the value of $M_{i,k}$, formalized by:

$$\overline{T}_i = [M_{i,1}, \ldots, M_{i,|\mathcal{S}|}] \tag{2.22}$$

2.3.2.8 Learning Time-Series Shapelets

Grabocka *et al.* propose a new look to the shapelet approach, the Learning time-series shapelets (LTS) [GSWS14]. Instead of performing an enumerative search, as all previous proposed approaches, the LTS algorithm learns the best k shapelets using the minimization of a classification objective function. This function is defined as a linear combination of the minimal distance between the shapelets and the time series of $\mathcal{T}$.

Initially, a set of random shapelets is clustered using k-means. Then the centroids from these clusters are repeatedly refined using a stochastic gradient descent algorithm which is used to perform the minimization. Since the minimal distance between the shapelets and the time series is not derivable, they propose to estimate it using a soft minimum. Each iteration allows fitting the shapelet.

One advantage of the approach is that the shapelets are fitted, taking into account their interaction. However, the approach has some drawbacks: the shapelets found are not guaranteed to exist within the training data., and often do not. Contrary to Shapelets tree or Shapelet transform-based approaches where shapelets not only exist in the dataset but also provide interpretable features.

Another issue is related to the number of hyper-parameters, which is much larger than a classical shapelet discovery based on shapelet candidates extracted from $\mathcal{T}$. Apart from specific parameters for the minimization (learning rate, number of iterations, regularization parameter, soft-minimum precision), the number of desired shapelets and their lengths have to be precised. Finally, the convergence of the minimization also impacts the computational complexity through the number of iterations, while this parameter does not exist in conventional shapelet discovery.

The LTS algorithm begins by finding a number of subsequences in the original training data which require two hyper-parameters, R and L, where L alters the length of subsequences considered and R determines the coverage of the shapelets, and widen the search space.

These hyper-parameters are of paramount importance and directly affect the quality of the resulting shapelets. For example, if we define $R = 3$ and $L = 0.2$ (the typical values defined in the original experiments) we would find shapelets whose length are 20%, 40% and 60% of the raw time series. Then, these initial subsequences are clustered using k-means. Finally, the set of k centroids is used with a gradient descent model. Each shapelet is refined through a defined derivative function, minimizing the entropy loss. This process continues for a max number of iterations, or until the model converges.

Among all shapelet-based algorithms, LTS is one of the two most accurate and advanced one (with the Shapelet Transform algorithm).

This approach offers a significant improvement in term of accuracy compared to approaches searching shapelets. It is also the first shapelet-based method which actually learns the shapelets instead of searching them.

2.3.2.9 Learning DTW-preserving Shapelets

Virtually all previously discussed approaches are designed to supervised tasks, therefore unfeasible on retrieval tasks. The exception is the u-shapelets, which reveal the use of shapelets in the unsupervised context. However, its construction is, in some way, similar to the shapelet-tree algorithm, therefore, impractical to be used for transforming a time series into a new representation.

Inspired by the LTSalgorithm [GSWS14], Lods *et al.* propose a different approach for learning time series shapelets, the "Learning DTW-Preserving Shapelets" [LMTA17].

LDPS was the first approach for shapelet transformation based on unsupervised learning, the goal is not trying to learn shapelets to best discriminating classes, but instead, they aim to learn shapelets that best preserve the true DTW measure in the embedded space. Therefore, although they proposed this transformation for the clustering task, there is no restriction to using the forged shapelets in other tasks. In this thesis, we propose the use of LDPS as a basis for Time Series Retrieval (TSR) framework. A significant differential of the retrieval task is the need for the transformation to be as fast as possible while preserving accuracy, ergo, part of our work will be to evaluate the quality of the learned shapelets trying to find the best compromise between the number os shapelets used in the transformation and the accuracy.

Before we discuss the use of LDPS in the retrieval task, we will present a review of it.

LDPS is an algorithm that embeds time series into a metric space such that the Euclidean distance in the transformed space approximates the DTW in the original time series space. An illustration of the LDPS algorithm is schematically given in Figure 2.26.

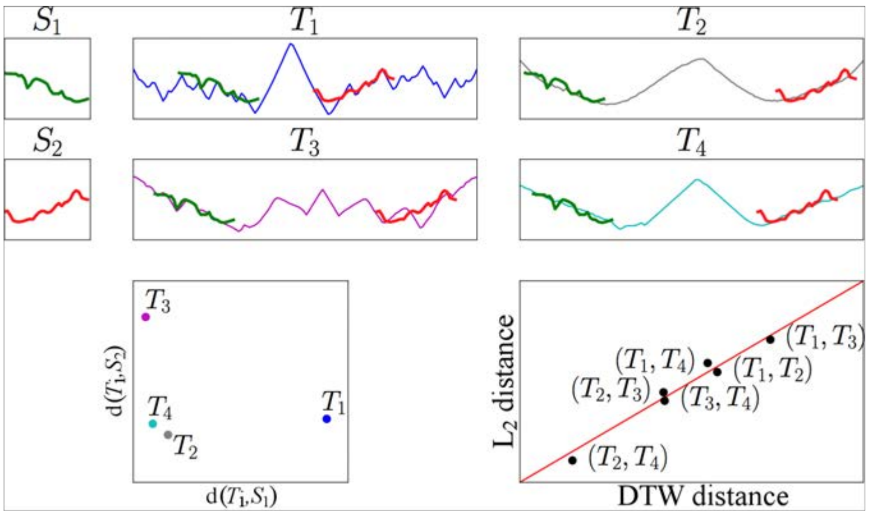


FIGURE 2.26 – Illustration of LDPS. The learned shapelets S_1 and S_2 are used to embed all TS from $\mathcal{T}$ into a 2-dimensional space $\overline{\mathcal{T}}$, in which the ED is a good approximation of the true DTW measure between the pairs in the original space. In real settings, more than two shapelets are learned to obtain higher-dimensional embedding and hence richer representations. Extracted from [LMTA17].

To learn a set $\mathcal{S} = \{S_1, \ldots, S_K\}$ shapelets, the algorithm depends on the Euclidean score and Euclidean match, respectively, defined as:

Definition 2.13 *Euclidean score. The Euclidean score between a shapelet S_k with $|S| = l$ and $T_{i,p:l}$, a subsequence of a time series T_i started at the position p, with length l is defined as:*

$$D_{i,k,p} = \frac{1}{l} \sum_{x=1}^{l} (T_{i,p+x-1} - S_{k,x})^2 \tag{2.23}$$

Definition 2.14 *Euclidean shapelet match. The Euclidean shapelet match between S_k and T_i is defined as:*

$$M_{i,k} = \min_{p \in \{1:N-l+1\}} D_{i,k,p} \tag{2.24}$$

Considering the learned dictionary $\mathcal{S}$, the representation $\overline{T}$ of T can be represented as:

$$\overline{T}_i = \mathrm{ST}(T_i, \mathcal{S}) = \{M_{i,1}, \dots, M_{i,|\mathcal{S}|}\} \tag{2.25}$$

Loss Function to be Minimized

The authors aim at learning a set of shapelets such that the ST preserves as well as possible the DTW measure. In other words, shapelets are forged such that the ED between transformed time series best approximates the DTW between raw time series. Hence, the objective function used to learn the shapelets is different from the ones of traditional works based on shapelets learning.

Thus, the problem is now the minimization of a loss function defined as:

$$\mathcal{L}(T_i, T_j) = \frac{1}{2}\left(\mathrm{DTW}(T_i, T_j) - \beta \left\|\overline{T}_i - \overline{T}_j\right\|_2\right)^2, \tag{2.26}$$

where β is a scale parameter, learned and $\overline{T_i}$ represents the shapelet transform of T_i. The overall loss for a dataset $\mathcal{T}$ of K time series is given by:

$$\mathcal{L}(\mathcal{T}) = \frac{2}{K(K-1)} \sum_{i=1}^{K} \sum_{j=i+1}^{K-1} \mathcal{L}(T_i, T_j). \tag{2.27}$$

The minimization of this loss is done via a stochastic gradient descent with respect to β and $\mathcal{S}$. Once the shapelets and the parameter β are learned, they can be used to transform every time series into a Euclidean vector.

Stochastic Gradient Descent

The LDPS method aims at learning, for a training set $\mathcal{T}$ of K time series, a set $\mathcal{S}$ of Q shapelets and a scale parameter β that minimize the overall loss defined in Equation (2.27).

A stochastic gradient descent framework is adopted to learn the $Q \cdot L + 1$ coefficients that lead to minimize $\mathcal{L}(\mathcal{T})$. In this framework, the gradients of $\mathcal{L}(T_i; T_j)$ with respect to these coefficients need to be computed. Considering $\hat{Y}_{i,j} = \left\|\overline{T}_i - \overline{T}_j\right\|_2$ and $\Delta_{i,j,q} = M_{i,q} - M_{j,q}$, the framework is defined as:

$$\frac{\partial \mathcal{L}(T_i, T_j)}{\partial \beta} = \hat{Y}_{i,j}\left(\beta \hat{Y}_{i,j} - \mathrm{DTW}(T_i, T_j)\right) \tag{2.28}$$

$$\frac{\partial \mathcal{L}(T_i, T_j)}{\partial S_{q,l}} = \frac{\partial \mathcal{L}(T_i, T_j)}{\partial \hat{Y}_{i,j}} \frac{\partial \hat{Y}_{i,j}}{\partial \Delta_{i,j,a}} \left(\frac{\partial \overline{T}_{i,q}}{\partial S_{q,l}} - \frac{\partial \overline{T}_{j,q}}{\partial S_{q,l}}\right) \forall q, l \tag{2.29}$$

Straight-forward derivations give:

$$\frac{\partial \mathcal{L}(T_i, T_j)}{\partial \hat{Y}_{i,j}} = \beta \left(\beta \hat{Y}_{i,j} - \mathrm{DTW}(T_i, T_j) \right) \tag{2.30}$$

$$\frac{\partial \hat{Y}_{i,j}}{\partial \Delta_{i,j,q}} = \begin{cases} \frac{\Delta_{i,j,q}}{\|\overline{T}_i - \overline{T}_j\|_2}, & \text{if } \overline{T}_i \neq \overline{T}_j \\ 0, & \text{if } \overline{T}_i = \overline{T}_j \end{cases} \tag{2.31}$$

$$\frac{\partial M_{i,q}}{\partial S_{q,l}} = \sum_o \frac{\partial M_{i,q}}{\partial D_{i,q,o}} \frac{\partial D_{i,q,o}}{\partial S_{q,l}}, \text{ where : } \frac{\partial D_{i,q,o}}{\partial S_{q,l}} = \frac{2}{L} \left(S_{q,l} - T_{i,o+l-1} \right) \tag{2.32}$$

These gradients are used to update the coefficients at each iteration of the algorithm with a learning rate of α, like for any gradient descent algorithm.

As the loss function Equation (2.27) is not convex, the model initialize by a set of Q shapelets generated by applying the k-means algorithm. Once the initial shapelets fixed, an initial value β_{ini} is selected for β by randomly sampling a set $\mathcal{P}$ of 100 time series pairs and computing the corresponding optimal least square solution to the mono-dimensional regression problem that relates distances between ST and DTW between original time series.

Lods *et al.* observed a strong negative correlation between the loss associated to a model and clustering quality obtained with this model. This seems to indicate that the value of the loss can be used as a model selection criterion without using any ground truth information. For a given data set, several models can be learned (different initialization and variants of LDPS). The one leading to the smallest loss will be selected. This model selection criterion was applied in the clustering task by the authors.

2.3.2.10 Summary on Shapelet-based Transformation

Shapelet transformation emerged as an extension to the initial Shapelet-tree approach. While on the one hand the idea of shapelet being a representative subsequence to determine a class was lost, on the other hand the way was opened for its use with traditional classifiers, breaking the decision tree barrier.

In turn, LTS made possible for shapelets not to be extracted from the dataset anymore, but forged, thus breaking completely with the original idea that they were descriptive subsequences. However, the results obtained in the classification task using LTS are excellent, surpassing the original Shapelet-tree algorithm.

The LDPS s a very interesting approach for dealing with time series. By using DTW and not labels for the shapelet evaluation, the LDPS allows its use for tasks other than clustering or classification. The authors themselves suggested using them in clustering tasks, where, using the traditional k-Means clustering algorithm, they achieved results superior to those obtained using approaches specifically designed to manipulate time series.

In addition, by definition, LDPS allows to apply the learned transformation to new time series, therefore applicable for open datasets. Hence our interest in using LDPS as the basis for an applied framework for the retrieval task.

2.3.3 Summary on Data Representation and Distance Functions

Many algorithms have been described, and some remarks can be summarized. The Euclidean distance is the faster approach for the retrieval task, and several transformations aim to convert the time series into a new representation for later application of the Euclidean distance. Besides that, the Euclidean distance is metric; however, it can benefit from triangular inequality in indexing systems.

On the other hand, DTW is arguably one of the best functions for dealing with local distortions on the time-axis. As a result, several approaches have been proposed to enhance DTW's performance; however, often restricting its virtue of dealing with distortions or reducing the quality of the response.

Finally, the Learning DTW-preserving shapelets (LDPS) is a good bet to try to unify DTW's rugged with Euclidean distance performance. Since it was initially designed for the clustering task, the cost of the transformation was not considered. Therefore, its use in retrieval should be studied in order to enable a good relationship between preserving DTW accuracy and ED performance.

Chapter 3

Time Series Retrieval Using DTW-preserving Shapelets

It's always good to take an orthogonal view of something. It develops ideas.

Ken Thompson, *C, Unix and beyond an interview with:*

Contents

As discussed previously, similarity-based time series retrieval should be fast and also robust to handle distortions and noise. L_p-norms are easy to compute, but they cannot handle local time shifting, in its turn, DTW can deal with local time shifting, but it is a non-metric distance function and has quadratic time-cost complexity. Besides that, the efficiency of similarity search using the DTW, is a particular problem since it is a non-metric distance function, and violates the triangle inequality that renders most indexing structures inapplicable. To this end, studies on this topic propose various smart optimizations, like global constraints or lower bounds on the actual distance to guarantee no false dismissals and others [YJF98; KPC01; Keo02; BC94; TWP17]. However, those lower bounds can admit a high percentage of false positives, and much of the optimizations are based on statistical characteristics of the dataset. Therefore, the quest for processing extremely large collections of time series is still active.

On the other hand many approaches go in another direction making use of features to represent the raw time series data. Those features, are then used with traditional L_p-norms [FRM94; CKMP02; SK08]. Therefore, these new representations allow the reduction of the amount of data to handle and also enabling the use of traditional data index structures. However, those representations fail in preserving the time-axis related data.

In this chapter, with the consideration of above issues, the following questions are explored:

- *Is there a way to combine the metric properties of the L_p-norms and some data transformation emulating the elastic property of the DTW, so that one can get the best of both worlds – namely being able to support local time shifting and being a metric distance function?*
- *We can make this representation able to work in open data tasks, and fast enough for the retrieval task?*

Proposed for the time series clustering, the Learning DTW-preserving shapelets (LDPS) [LMTA17] focus on learning, without class label information, shapelets such that Euclidean distances in the shapelet-transformed-space approximate well the true DTW. As opposed to the most part of the shapaet transform approach, the DTW-preserving shapelet transformation method is an unsupervised method, making it adaptable to the retrieval task.

In this thesis, we propose to extend the use of DTW-preserving shapelets transformation [LMTA17] from clustering to the retrieval. As in other data transformation approaches, the LDPS has a cost and, when applying with the retrieval task, two issues raise and must be carefully addressed efficient in terms of speed and quality:

(i) The complexity of the transformation should be small to reduce the overhead induced by transforming the query;

(ii) The true nearest neighbor has to be as close as possible to the first element in the list of approximate nearest neighbors — in other words, the transformation should preserve the ranking.

In our work, we propose a trade-off between accuracy of the retrieval and its cost by selecting a varying number of shapelets for computing the vectorial representations. Three shapelet selection strategies are proposed in order to identify and rank shapelets that are more suited to the retrieval task (via a dedicated measure). This step has two purposes: reducing the number of shapelet for the transformation of the query (to save computing time), while keeping a good trade-off with accuracy of the retrieval.

The rest of the chapter is organized as follows: Section 3.1 presents the overview of the dtw-preserving shapelet-based time series retrieval framework. Strategies to evaluate, rank and select subsets of shapelets are discussed in Section 3.2. In Section 3.3, a novel graph-based method for unsupervised multidimensional feature selection is presented, this method can be used to remove irrelevant shapelets prior to the evaluating task, with gains in the offline phase. Finally, Section 3.4 presents the extention to the multivariate case.

3.1 Overview

Off-line, the dictionary $\mathcal{S}$ of d shapelets is learned from train dataset $\mathcal{T}_{\text{train}}$, as described by Lods *et al.* in [LMTA17].

Lods *et al.*, based on [GSWS14], define the shapelet lengths l to 15%, 30% and 45% of time series length N. Also, the dictionary size d is defined as a function of the time series length N and the shapelets length l:

$$d = 10 \cdot \sum_{l \in \{0.15N, 0.3N, 0.45N\}} \log(N - l) \tag{3.1}$$

All time series in $\mathcal{T}$ are then transformed and form $\overline{\mathcal{T}}$, which is stored in a database. This process is revealed in Figure 3.1.

On-line, a query time series Q is transformed into $\overline{Q}$ using the dictionary of shapelets $\mathcal{S}$ previously learned. The k nearest neighbors of $\overline{Q}$ are then searched in $\overline{\mathcal{T}}$ and kept in a transient list of time series ranked according to their proximity to $\overline{Q}$.

The final result can typically be built according to two options:

Directly: The raw time series associated with the nearest neighbor found in the transformed space is considered to be the nearest neighbor of the raw query, or

Refined: The true DTW is subsequently computed between the raw query and the raw versions of the k transformed time series that have been identified, and the nearest is returned.

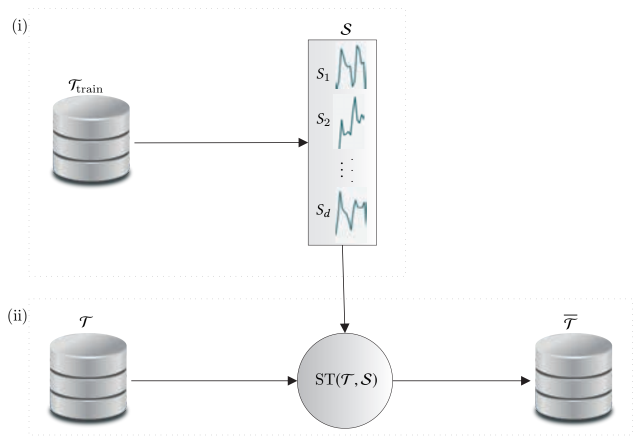


FIGURE 3.1 – Diagram: (i) The shapelet dictionary $\mathcal{S}$ is learned using the training-set $\mathcal{T}_{\text{train}}$. (ii) Using $\mathcal{S}$, the dataset $\mathcal{T}$ is transformed in $\overline{\mathcal{T}}$ by the shapelet transform ST. All steps are done off-line.

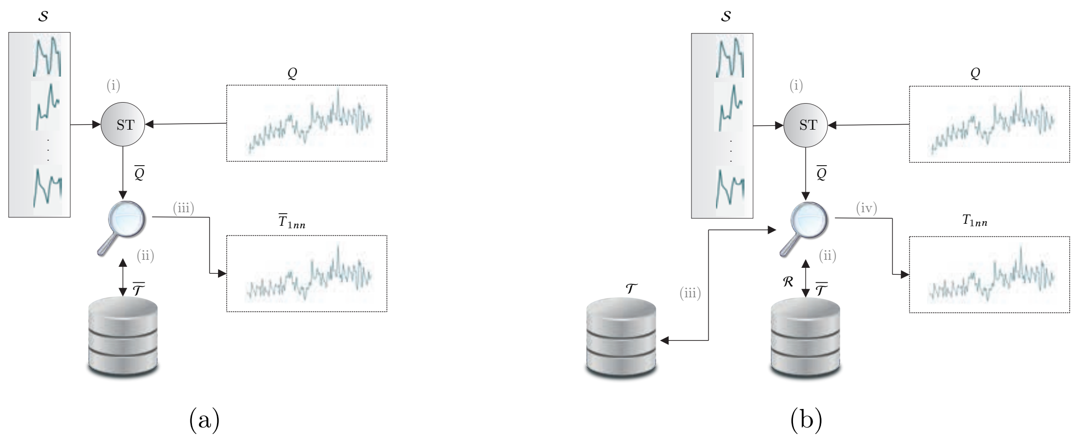


FIGURE 3.2 – Diagram illustrating the difference between methods directly (a) and refined (b). In (a), the retrieval start by transforming the query Q to $\overline{Q}$ using the dictionary $\mathcal{S}$; then using $\overline{Q}$, we search for the closest entry in $\overline{\mathcal{T}}$ (ii); finally the closest element $\overline{T}_{1nn}$ is returned (iii). In (b), after transforming Q in $\overline{Q}$ (i), the search initially return the set $\mathcal{R}$ of the closest elements (ii); then, a refinement is done by comparing Q and the entries in $\mathcal{T}$ appointed by the entries in $\mathcal{R}$ using the DTW (iii); finally, the closest element is returned (iv).

This DTW-preserving embedding is such that the ranking in the transformed space is an approximation of the ranking that would be produced in the original space according to the DTW measure. However, this L_2-based ranking is obtained much faster, as Euclidean distances are cheaper to compute compared to DTW measurements.

With respect to the final result returned to the users:

Directly: It puts a lot of pressure on the quality of the shapelet transform because the nearest time series under DTW has also to be the nearest in the Euclidean space, and this for any time series in the dataset and any query. Odds of degrading quality in comparison to what the true DTW would determine are high, but this method is extremely fast.

Refined: It is more demanding because extra DTW are computed, but it also returns better quality results as $k > 1$ time series are analyzed. In this case, it matters that the nearest time series under DTW belongs to these first k (embedded) elements, instead of being ranked first, strictly.

Transforming Q into $\overline{Q}$ has a cost, which cannot be neglected because it is part of the on-line phase. This cost is essentially due to sliding the d shapelets from $\mathcal{S}$ against Q in order to determine the d minimal distances which are kept and form $\overline{Q}$. This cost is particularly high for long time series and when the number of learned shapelets is high.

3.2 Ranking and Selecting Shapelets

General considerations about high-dimensional representations suggest that components of vectors might not all be equally useful. This is well-known and led to designing dimensional reduction methods doing feature selection.

Such algorithms are perfectly relevant when considering the embedding of time series into high-dimensional vectors: the d dimensions of time series in $\overline{\mathcal{T}}$ might as well not be all equally useful. Eliminating poorly informative dimensions, and in turn considering a reduced number of shapelets, speeds up the transform, hopefully without losing too much quality, result wise. Therefore, instead of using $\mathcal{S}$ as specified by [LMTA17], it might be better to keep and use for the transform only a subset of it, made of carefully selected shapelets.

To apply the spirit of feature selection algorithms to the case of shapelet transform, it is necessary to define:

(i) a metric in order to evaluate the respective quality of each subset of shapelets,

(ii) a strategy for building increasingly large subsets of shapelets until,

(iii) it becomes appropriate to stop considering anymore shapelet.

This metric, building strategy and stop criterion establish the essence of most existing feature selection approaches such as the ones detailed in [SIL07; LCW+18; BL97; MR15].

In the next section, we will detail how they instantiate in the case of time series retrieval.

3.2.1 Evaluation Metric to Compare Shapelet Subsets

The framework starts by constructing a dictionary $\mathcal{S}$ from $\mathcal{T}_{\text{train}}$ as proposed by Lods *et al.* in [LMTA17]. We do not use $\mathcal{S}$ to transform the time series, but instead as a reservoir from which the most useful shapelets will then be picked. The following motivates and presents strategies for building $\mathcal{S}_s$ from $\mathcal{S}$, where $\mathcal{S}_s$ contains the shapelets eventually retained for transforming $\mathcal{T}$ into $\overline{\mathcal{T}}$, as well as transforming Q into $\overline{Q}$.

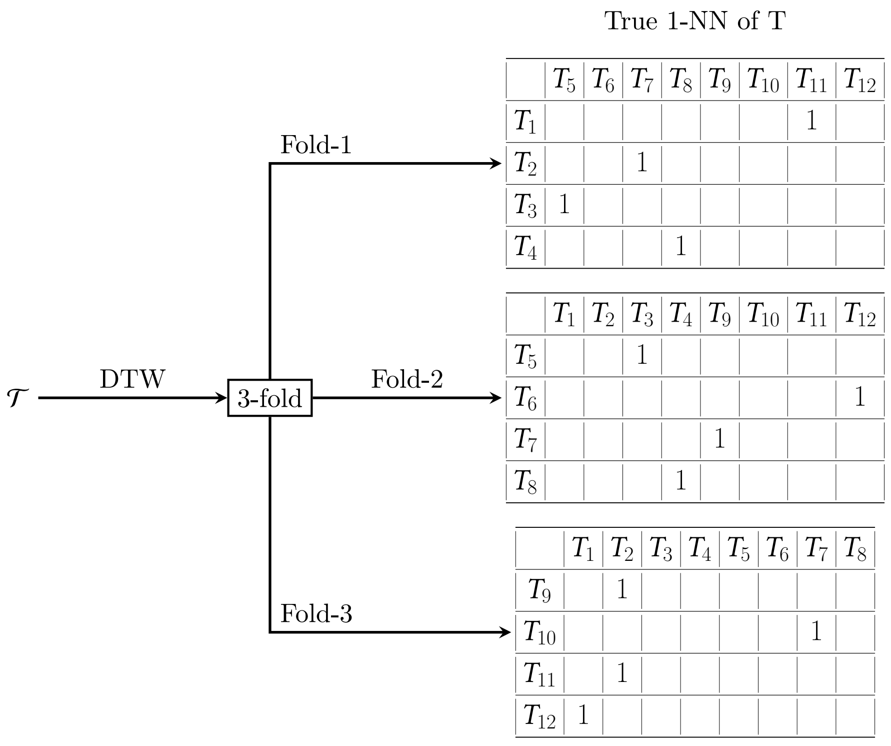
To compare the performance of different shapelet subsets, we need a rank ground-truth based on the true DTW distances between all time series in $\mathcal{T}_{\text{train}}$. We compute the distances once only[1], off-line.

We start by splitting the training set $\mathcal{T}_{\text{train}}$ into 10 parts, 9 being used for the subsequent training tasks detailed below, one used for validation, in a traditional 10-fold cross-validation context.

The ground-truth is build using the previously computed DTW distances. We record on each fold for each time series the identifier of its 1-nearest-neighbour, Figure 3.3 depicts an example of 1-NN-DTW ground-truth.

Then, we pick a dictionary $\mathcal{S}_s \subset \mathcal{S}$, and use these $\mathcal{S}_s$ shapelets to transform all the time series that belong to the current training fold. Using the same $\mathcal{S}_s$ shapelets, we also transform each time series from the validation fold and use them as queries. The transformed time series

[1]During the execution of the LDPS, a (sparse) matrix of the DTW distances between some pairs is made. We use this matrix as base, to reduce the computational time.



	T_5	T_6	T_7	T_8	T_9	T_{10}	T_{11}	T_{12}
T_1							1	
T_2			1					
T_3	1							
T_4				1				

	T_1	T_2	T_3	T_4	T_9	T_{10}	T_{11}	T_{12}
T_5			1					
T_6								1
T_7					1			
T_8				1				

	T_1	T_2	T_3	T_4	T_5	T_6	T_7	T_8
T_9		1						
T_{10}							1	
T_{11}		1						
T_{12}	1							

FIGURE 3.3 – A toy example of the ground-truth construction: for a time series dataset $\mathcal{T} = \{T_1, \dots, T_{12}\}$, we build the ground-truth of its true 1-nn based on the DTW measure, each row represents the query element, for example, the true 1-nn of T_1 is T_{11}. In this example, we are using a 3-fold setup.

from the training fold are then ranked w.r.t. their L_2 distance to each query. It is therefore possible to determine at which rank the true nearest time series is (w.r.t. DTW) for that query, as observed in Figure 3.4. Repeating this operation for all the validation time series and for all the folds amounts to building an histogram of the ranks at which the true DTW-based nearest-neighbour appear.

This histogram can also be interpreted as the empirical probability of observing the true 1-nearest-neighbor time series at any particular rank after the transform. And, after a proper normalization, it can therefore be considered to be a Probability Density Function (PDF).

The PDF is defined as follows:

Definition 3.1 *PDF The Probability Density Function (PDF) of a discrete random variable X is a function that satisfies the following properties:*

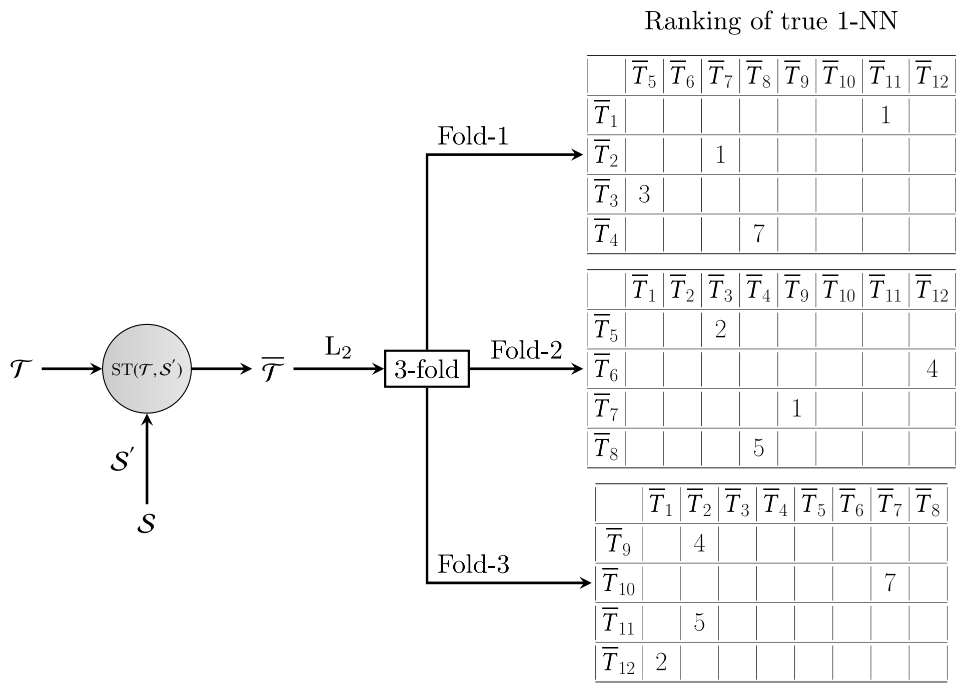
1. $P(X = x) = f(x) > 0$ *if* $x \in \mathcal{S}$;
2. $\sum_{x \in \mathcal{S}} f(x) = 1$
3. $P(X \in \mathcal{A}) = \sum_{x \in \mathcal{A}} f(x)$

From this PDF, it is straightforward to construct its natural counterpart which is the Cumulative Distribution Function (CDF), and to compute the Area Under Curve (AUC).

Definition 3.2 *Cumulative Distribution Function (CDF) The Cumulative Distribution Function (CDF) of discrete random variable X is defined as*

$$F_X(t) = P(X \leq t) \tag{3.2}$$

The notation $F_X(t)$ means that F is the CDF for the random variable X, however, it is a function of t. And, it has the following properties:



	$\overline{T}_5$	$\overline{T}_6$	$\overline{T}_7$	$\overline{T}_8$	$\overline{T}_9$	$\overline{T}_{10}$	$\overline{T}_{11}$	$\overline{T}_{12}$
$\overline{T}_1$							1	
$\overline{T}_2$			1					
$\overline{T}_3$	3							
$\overline{T}_4$				7				

	$\overline{T}_1$	$\overline{T}_2$	$\overline{T}_3$	$\overline{T}_4$	$\overline{T}_9$	$\overline{T}_{10}$	$\overline{T}_{11}$	$\overline{T}_{12}$
$\overline{T}_5$			2					
$\overline{T}_6$								4
$\overline{T}_7$					1			
$\overline{T}_8$				5				

	$\overline{T}_1$	$\overline{T}_2$	$\overline{T}_3$	$\overline{T}_4$	$\overline{T}_5$	$\overline{T}_6$	$\overline{T}_7$	$\overline{T}_8$
$\overline{T}_9$		4						
$\overline{T}_{10}$							7	
$\overline{T}_{11}$		5						
$\overline{T}_{12}$	2							

FIGURE 3.4 – To evaluate a shapelet dictionary $\mathcal{S}' \subset \mathcal{S}$, we start by transforming the dataset $\mathcal{T}$ in $\overline{\mathcal{T}}$ using as dictionary $\mathcal{S}'$. Then, we build the ranking of the pairs according to the L$_2$ distance, and finally we use it to determine the rank position of the true 1-NN.

1. *$F_X(t)$ is a non-decreasing function of $t \in\ ]-\infty, \infty[$;*
2. *$F_X(t) \in [0, 1]$;*
3. *If X is a discrete random variable whose minimum value is a, then $F_X(a) = P(X \le a) = P(X = a) = f_X(a)$. If c is less than a, $F_X(c) = 0$;*
4. *If the maximum values of X is b, then $F_X(b) = 1$.*

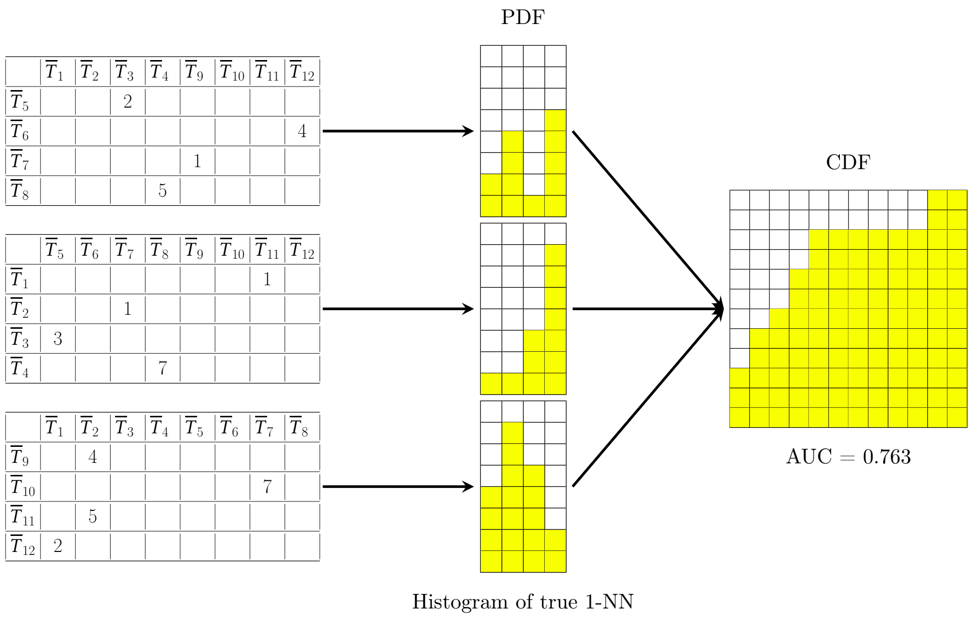
We consider this AUC value as the performance measure to evaluate the quality of a shapelet subset. The higher that Area Under Curve (AUC), the better the shapelet subset. This metric is well adapted to the task of nearest-neighbour retrieval as it favors high ranking of the true nearest-neighbor in the approximated list.

3.2.2 Shapelet Subset Selection

To select the best subset of shapelets, an exhaustive selection method can be applied to form all possible subsets $\mathcal{S}_s \in \mathcal{S}$ with $|\mathcal{S}| \in [1, |\mathcal{S}|]$. However, in this case, the computational cost is prohibitively high, as the set off all combination can be expressed as:

$$\sum_{k=1}^{|\mathcal{S}|} \binom{\mathcal{S}}{k} = 2^{|\mathcal{S}|} \Rightarrow O\left(2^{|\mathcal{S}|}\right). \tag{3.3}$$

Therefore, some heuristic method is necessary. We have based our approach in a wrapper greedy-based forward selection method, which is classically used in the feature selection domain [KJ97]. Like all greedy-based algorithms, our algorithm, presented in Algorithm 3.1, follows the problem solving heuristic of making the locally optimal choice at each step with the intent of finding the global optimum. Of course, as the subset shapelets selection does



	$\overline{T}_1$	$\overline{T}_2$	$\overline{T}_3$	$\overline{T}_4$	$\overline{T}_9$	$\overline{T}_{10}$	$\overline{T}_{11}$	$\overline{T}_{12}$
$\overline{T}_5$			2					
$\overline{T}_6$								4
$\overline{T}_7$					1			
$\overline{T}_8$				5				

	$\overline{T}_5$	$\overline{T}_6$	$\overline{T}_7$	$\overline{T}_8$	$\overline{T}_9$	$\overline{T}_{10}$	$\overline{T}_{11}$	$\overline{T}_{12}$
$\overline{T}_1$							1	
$\overline{T}_2$			1					
$\overline{T}_3$	3							
$\overline{T}_4$				7				

	$\overline{T}_1$	$\overline{T}_2$	$\overline{T}_3$	$\overline{T}_4$	$\overline{T}_5$	$\overline{T}_6$	$\overline{T}_7$	$\overline{T}_8$
$\overline{T}_9$		4						
$\overline{T}_{10}$							7	
$\overline{T}_{11}$		5						
$\overline{T}_{12}$	2							

FIGURE 3.5 – Construction the PDF and CDF of an arbitrary dictionary $\mathcal{S}' \subset \mathcal{S}$.

not present an optimal substructure, the algorithm is used to approximate the globally optimal solution in a reasonable amount of time.

The procedure selecting the best shapelet subset begins with an empty list. After, it evaluates all shapelets once, and the best, according to the evaluation metric described in Section 3.2.1, is selected and added to the list. Then, it iteratively adds to the list, one-by-one, the shapelet that best improve the quality of the subset by measuring the resulting AUC. This process is repeated until a stopping criterion is met. In this scenario, the worst-case occurs when all $|\mathcal{S}|$ interactions are necessary, resulting in:

$$\frac{(1 + |\mathcal{S}|) \cdot |\mathcal{S}|}{2} \Rightarrow O\left(|\mathcal{S}|^2\right) \tag{3.4}$$

operations, and it depends on the stooping criterion used.

It is important to realize this procedure is *greedy* and that it identifies first the best shapelet, then the best pair given the choice made earlier, and successively.

3.2.2.1 Stopping Criterion

As previously discussed, one of the steps of a wrapper-based method for feature selection resides in the criterion used to determine when it needs to stop looking for a new element (feature) for the final set. In our work, four different stopping criteria is defined, determining when to stop adding shapelets to the current set of selected shapelets:

DPSR$_\mathbf{g}$: Shapelets are added one by one until no more shapelets are available. At the end, the subset that leads to the best overall AUC is selected. This finds the number of shapelets giving *global maximum* of the AUC-based performance.

DPSR$_\mathbf{t}$: We compute the normalized slope between the AUC value of the current selected subset and the one obtained by adding the shapelet that best improves the AUC. If this slope is less than 1, then the shapelet selection is stopped.

DPSR$_\mathbf{l}$: The shapelet selection is stopped as soon as adding a shapelet does not improve the AUC value. This finds the *first local maximum* of the AUC-based performance.

Algorithm 3.1 The DPSR algorithm — Creating the dictionary of shapelets used for the DTW-preserving transform

```
1: input: $\mathcal{S}$ {Dictionary of shapelets}, $\mathcal{T}$ {Time series train set}
2: output: $\mathcal{S}_s$ {Ranked list of selected shapelets}
3: $\mathcal{S}_a \leftarrow \mathcal{S}$ {Shapelets available, initially all}
4: $\mathcal{S}_s \leftarrow \emptyset$ {Dictionary of selected shapelets, initially empty}
5: $stop \leftarrow$ FALSE
6: repeat
7:    $score_b \leftarrow -1$
8:    for all $S \in \mathcal{S}_a$ do
9:       $\mathcal{S}' \leftarrow \mathcal{S}_s \cup S$
10:      $score \leftarrow$ AUC value of the set $\mathcal{S}'$
11:      if $score > score_b$ then
12:         $score_b \leftarrow score$
13:         $S_b \leftarrow S$
14:      end if
15:   end for
16:   $\mathcal{S}_s \leftarrow \mathcal{S}_s \cup S_b$
17:   $\mathcal{S}_a \leftarrow \mathcal{S}_a \setminus S_b$
18:   if Stopping criterion is met then
19:      $stop \leftarrow$ TRUE
20:   end if
21: until $stop$ = TRUE
22: return $\mathcal{S}_s$
```

DPSR$_\mathbf{f}$: All shapelets are used, none is excluded.

Figure 3.6 clarifies the four stopping criteria on the AUC curve. It is important to note during each evaluation step, it is necessary, for each shapelet, to transform, compute and rank the distances for all time in the validation set. Hence, in the worst case, when all shapelets are evaluated the computational complexity is given by:

$$\overbrace{O(|T|)\cdot O(|\mathcal{S}|^2)\cdot O(|\mathcal{T}|)}^{①}\cdot\overbrace{O(|\mathcal{S}|\cdot|\mathcal{T}|\cdot\log(|\mathcal{T}|))}^{②}\Leftrightarrow O\left(|T|\cdot|\mathcal{S}|^3\cdot|\mathcal{T}|^2\log|\mathcal{T}|\right) \tag{3.5}$$

where ① is the computational complexity to transform all time-series from the training dataset, for all shapelets in the remained list, and ② is the computational complexity to compute and ranking the distances in the $\mathcal{S}_s$-based embedded space.

Although it does not directly impact query time, as it is an offline operation, this cost impacts directly on the database construction time. With that in mind, a way to reduce this database build time can be achieved by a previously elimination of redundant or irrelevant shapelets, before applying the proposed evaluation method.

In the next section, we present a solution to filtering irrelevant and redundant shapelets based on graph representation and clique elimination.

3.3 Filtering Out Irrelevant Shapelets

From the feature selection literature, the selection method described previously can be categorized into the wrapper class, where a learning algorithm is applied to evaluate the respective quality of different feature subsets, interactively. This is the most common approach, and despite its improvement over the exhaustive search, it remains a very costly process when analyzing many features. Consequently, early filtering out irrelevant or redundant features would typically

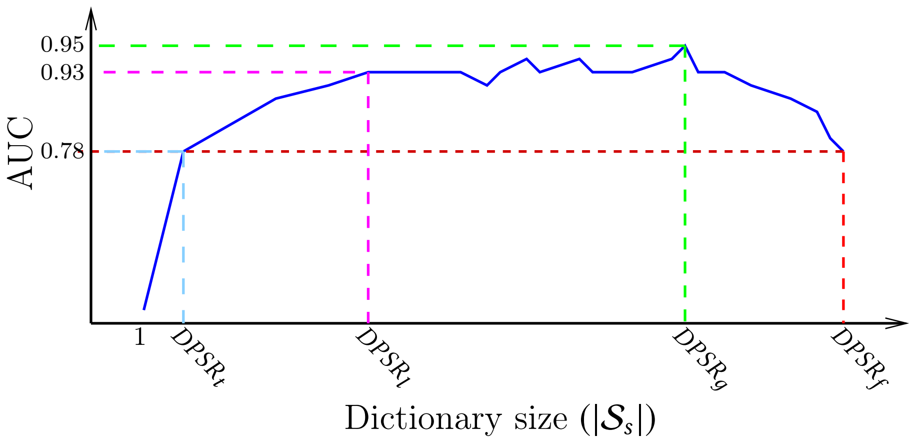


FIGURE 3.6 – An illustration of the four proposed stopping criteria. The algorithm starts with an empty set, then it identifies first the best shapelet, and, successively, identify the best in the remained list. In $\mathrm{DPSR_t}$ and $\mathrm{DPSR_l}$, the evaluation stops immediately after the condition is met. $\mathrm{DPSR_g}$ needs to run until there is no more shapelet, then, the best value can be selected.

improve the speed of the subsequent feature wrapper algorithm as it would have fewer features to analyze.

Filters are usually cheap and fast, and often apply a basic statistical analysis over the feature set. Whereas supervised feature filtering methods can not be used here, unsupervised filter-based methods typically rely on term variance [Fis19], Laplacian score [HCN05] or Spectral selection [ZL07]. In its turn, [dSRdE+11] takes into account the possible dependencies between features.

In this section, we will present a novel graph-based method for unsupervised multidimensional feature filtering, which can efficiently and effectively deal with both irrelevant and redundant features. Recently, the use of graph-based methods for filtering features has played an essential role in machine learning because of their ability to encode similarity relationships among pieces of data.

Representing the feature space in a graph can provide a universal and flexible framework that reflects the underlying manifold structure and relationships between the features. Two well-known graph-based methods are the Fisher Score (FS) and Laplacian Score (LS).

The proposed method consists of three main steps:

(i) Graph representation of the problem space;

(ii) Enumeration of all cliques in the graph;

(iii) Search for the best representative shapelet from each clique, by looking at some relevance criterion.

In the first step, the shapelet (feature) set, is represented as a weighted-undirected graph in which each node in the graph denotes the shapelet and each edge weight indicates the similarity value between its corresponding shapelets. The second step looks for all cliques in the graph,and we consider that cliques represent sets of well-similar shapelets. Finally, in the third step, using some relevance criterion, the most representative node of each clique is preserved, and the others are removed from the graph.

The additional details are described in the following sections.

3.3.1 Graph Representation of the Transformed Space

To recap, when we use a shapelet dictionary $\mathcal{S}$ to transform a time series T into a high-dimensional vector $\overline{T} = \{\overline{t}_1, \ldots, \overline{t}_{|\mathcal{S}|}\}$ each dimension $\overline{t}_i$ is a feature which corresponds to the match distance between the time series T and the shapelet $S_i \in \mathcal{S}$. Thus, we can compare, in the transformed dataset, if some shapelet is redundant or not, based on the resultant values associated with its transformation.

In this way, we can discover, in the transformed dataset, if some shapelet is redundant or not, based on the resultant values associated with its transformation. For example, a shapelet which produces transformation with a minimal variance indicates that the measured distances tend to be very close to the mean; consequently, this specific shapelet probably does not present a good discriminating power, thus irrelevant. Also, if two shapelets produce very close measures or very close distance-based ranks, one of them is redundant. Thus, we can infer the quality of the shapelet based only on the statistical analysis of its resultant transformation.

For this purpose, we construct a graph $G = (V, E, w)$ where $V = \{1, \ldots, |\mathcal{S}|\}$ represents the index of each shapelet $S_i \in \mathcal{S}$, $E = \{(V_i, V_j) \forall V_i, V_j \in V, i \neq j\}$ represents that there is an observed similarity between V_i and V_j, and $w_{i,j}$ denotes the measured similarity between the transformation delivered by the pair of shapelets S_i and S_j, connected by the edge (V_i, V_j).

In our method, we start by computing the variance in the distances to transformed time series when using each shapelet in isolation. It is then easy to rank shapelets by increasing variance of their term. Shapelets ranked below the α-first percentile are immediately filtered out and are not considered anymore. The p ones that remain form $\mathcal{S}^+$. The similarity between the shapelets in $\mathcal{S}'$ are then computed using the absolute value of the Pearson's product-moment correlation coefficient [HCBV10]:

$$w_{i,j} = \left| \frac{\sum_p (M_i - \bar{\mathcal{M}}_i) \cdot (\mathcal{M}_j - \bar{\mathcal{M}}_j)}{\sqrt{\sum_p (\mathcal{M}_i - \bar{\mathcal{M}}_i)^2} \cdot \sqrt{\sum_p (\mathcal{M}_j - \bar{\mathcal{M}}_j)^2}} \right|, \tag{3.6}$$

where $\mathcal{M}_i$ and $\mathcal{M}_j$ refer to the i^{th} and j^{th} dimension of the transformed space induced by the shapelets $S_i, S_j \in \mathcal{S}'$. Other correlation methods can be used, in our experiments, we have tried with the more expensive Spearman correlation and Kendall-tau measures and could not see any major gain.

The shapelets $\mathcal{S}_i$ and $\mathcal{S}_j$ in the graph are connected if and only if the value of $w_{i,j} \geq \delta$. Where α and δ are threshold values determined by the user. For example, considering a learned dictionary $\mathcal{S}$, with $|\mathcal{S}| = 21$, after computing the variance and the correlation in the distances per shapelet, and considering $\delta = 0.8$ and $\alpha = 0.05$, we obtain for a toy example dataset $\mathcal{T}$, the values represented in Table 3.1, and the graph-based representation showed in Figure 3.7.

3.3.2 Enumeration of all Cliques

In a straightforward way, in graph theory, we can define a clique as:

Definition 3.3 *Clique. Clique is a subset of vertices of an undirected graph such that every two distinct vertices in the clique are adjacent; that is, its induced subgraph is complete.*

We are interested in a special kind of clique, the maximal clique, defined as:

Definition 3.4 *Maximal clique. In a graph $G = (V, E)$, for each node V^*, a maximal clique for V^* is a largest completed subgraph containing V^*.*

In our approach, we use the set of all maximal cliques as a surrogate for clusters, once all elements into the clique are strongly related due to its high correlation. The maximal

Table 3.1 – Statistics of the dictionary $\mathcal{S}$ for a toy example $\overline{\mathcal{T}}$.

Shapelet/Node	Variance
1	0.00225
2	0.00285
3	0.00231
4	0.00241
5	0.00278
6	0.00321
7	0.00281
8	0.00151
9	0.00214
10	0.00200
11	0.00215
12	0.00281
13	0.00314
14	0.00301
15	0.00281
16	0.00300
17	0.00325
18	0.00324
19	0.00299
20	0.00210
21	0.00002

Edge		Correlation
1	2	0.854
1	3	0.831
1	4	-0.912
1	5	0.935
2	3	0.801
2	4	0.932
2	5	0.995
3	4	0.953
3	5	0.803
4	5	0.875
3	6	0.865
4	6	0.866
5	6	0.932
5	7	0.911
6	7	0.913
3	9	0.991
7	9	0.941
12	13	0.843
13	14	0.885
15	16	0.809
15	17	-0.921
16	17	0.851
15	18	0.903
17	18	0.811
17	19	0.888
18	19	0.876
16	20	0.809
1	20	0.909

clique enumeration is a well-known problem in graph theory, and despite being classified as NP-complete[2], it has well-performing algorithms on sparse graphs [MU04], as the Bron–Kerbosch algorithm [BK73].

Due to the choice of the δ threshold parameter, the resulting graph G, build upon the output of the previous step, is typically sparse (as observed in the example of Figure 3.7), and consequently computationally feasible.

Our method works by doing node elimination, as a clique is detected we look for the most relevant node into the clique, preserving it and removing the others, in this way, this elected node is "qualified" to represent its adjacent set (inside the clique). However, before looking directly at the set $\mathcal{C} = \{C_1, \ldots, C_n\}$ of all cliques in G, we look for the set $\mathcal{I} = \{I_1, \ldots, I_m\}$ of all maximal-clique-like intersections in G.

Definition 3.5 *Maximal-clique-like intersections. A maximal-clique-like intersection I between two cliques C_1 and C_2 is the set of the all common vertices of C_1 and C_2, such that $|I| \geq 2$.*

$$I = \begin{cases} \emptyset, & \text{if } |C_1 \cap C_2| < 2 \\ C_1 \cap C_2, & \text{otherwise} \end{cases} \tag{3.7}$$

[2]For comparison, the traditional and ubiquitous clustering-method k-means is classified as NP-Hard.

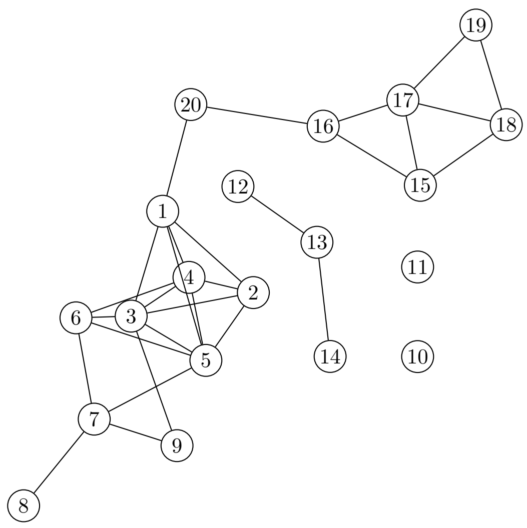


FIGURE 3.7 – Our initial graph-based representation of the feature space. For a dictionary $\mathcal{S}$, $|\mathcal{S}| = 21$, due to the pruning parameters α and δ, the graph G is disconnected, with 20 nodes and 28 edges.

We focus initially on the maximal-clique-like intersections based on the idea that those nodes are common to more than one clique. Consequently, they have some "inter-clique" redundancy. Thus, by eliminating the intersection, we can do a better distinguish between the cliques.

We use the algorithm Cliques [BK73; TTT06; CK08] to find all maximal cliques in the graph-based representation. The pseudo-code for the maximal clique enumeration method is given by Algorithm 3.2.

For example, when applied to the graph G given by Figure 3.7, we obtain the list $\mathcal{C}$ of cliques enumerated by the Table 3.2a and the list of intersection I determined by the Definition 3.5. Those lists, showed in Table 3.2, are used to select and preserve relevant features.

TABLE 3.2 – Enumeration of all cliques (a), intersections of cliques (b), and singleton nodes (c) in the graph G from Figure 3.7.

(a)

$C_i \in \mathcal{C}$	List of nodes	$\lvert C \rvert$
C_1	{1, 2, 3, 4, 5}	5
C_2	{3, 4, 5, 6}	4
C_3	{5, 6, 7}	3
C_4	{3, 9}	2
C_5	{7, 8}	2
C_6	{12, 13}	2
C_7	{13, 14}	2
C_8	{15, 16, 17}	3
C_9	{17, 18, 19}	3
C_{10}	{16, 20}	2
C_{11}	{1, 20}	2
C_{12}	{15, 17, 18}	3

(b)

$I_i \in \mathcal{I}$	List of nodes	$\lvert I \rvert$
I_1	{3, 4, 5}	3
I_2	{5, 6}	2
I_3	{15, 17}	2
I_4	{17, 18}	2

(c)

$$\left\{ \textcircled{10}, \textcircled{11} \right\}$$

Algorithm 3.2 Output all maximal complete subgraphs.

```
1: input: $G$ {Graph}
2: output: $C$ {Set of all maximal cliques in G.}
3: adj ← list all adjacent pairs in G
4: Q ← $\emptyset$ {Queue of candidates}
5: subg ← list of all nodes in G
6: cand ← list of all nodes in G {Candidates to be evaluated}
7: u ← node in subg with highest cardinality
8: ext_u ← list of nodes in cand except the set of adjacent of u
9: stack ← $\emptyset$ {Execution stack}
10: while true do
11:   if ext_u ≠ $\emptyset$ then
12:     q ← POP(ext_u) {next node in the ext_u list.}
13:     cand ← cand  q {remove q node from cand.}
14:     Q ← PUSH(Q,q) {Push q in the list Q.}
15:     adj_q ← list of adjacent of q
16:     subg_q ← subgraph with all adjacents of q
17:     if subg_q = $\emptyset$ then
18:       $C$ ← Q
19:     else
20:       cand_q ← set of nodes in cand adjacent to q
21:       if cand_q ≠ $\emptyset$ then
22:         stack ← stack ∪ {(subg, cand, ext_u)}
23:         Q ← Q ∪ { }
24:         subg ← subg_q
25:         cand ← cand_q
26:         u ← node in subg with highest cardinality
27:         ext_u ← list of nodes in cand except the set of adjacent of u
28:       end if
29:     end if
30:   else
31:     POP(Q)
32:     subg, cand, ext_u ← POP(stack)
33:   end if
34: end while
35: return $C$
```

3.3.3 Search for the Best Shapelets

Once the list of maximal cliques $\mathcal{C}$ is obtained, and after enumerating all maximal-clique-like intersections $\mathcal{I}$ in $\mathcal{C}$, we rank the entries in $\mathcal{I}$ by its cardinality, then we search for the largest entry, i. e. the element I^* such that:

$$I^* = \max_{I_k \in \mathcal{I}}(|I_k|) \tag{3.8}$$

When $\mathcal{I}$ become empty, we look directly on the cliques enumerated in $\mathcal{C}$, also ranked according to its cardinality, choosing the largest entry C^*:

$$C^* = \max_{C_k \in \mathcal{C}}(|C_k|) \tag{3.9}$$

All the vertices found in a maximal-clique-like intersection (clique) are filtered, and the node with the highest term variance[3] is preserved. Then, the lists $\mathcal{C}$ and $\mathcal{I}$, and, indirectly, the graph

[3]As the dimensions of the vectors $\overline{T} \in \overline{\mathcal{T}}$ are defined by the subD method, he values of variances are by construction normalized.

G are updated. The filtering algorithm finishes when the graph G is fully disconnected, i. e. the list C is empty, and, consequently, G contains only singletons vertices.

The pseudo-code for the proposed multivariate-unsupervised filter selection method is given in Algorithm 3.3.

Algorithm 3.3 Filtering out shapelets before building the dictionary.

```
1: input: $\overline{\mathcal{T}}$ {Shapelet-based-transformed training data set,}
          $\mathcal{S}$ {Set of learned shapelets}
          $\delta$ {Minimal correlation between shapelets}
          $\beta$ {Minimal term variance.}
2: output: $\mathcal{S}_f$ {Set of preserved shapelets.}
3: Compute variance for each shapelet. Rank shapelets on variance.
4: $\mathcal{S}^+ \leftarrow \mathcal{S} \setminus$ shapelets ranked $< \alpha$-percentile
5: Compute Pearson's correlation matrix $M$ for all shapelets in $\mathcal{S}^+$
6: Build an oriented graph $G$ connecting $S_i$ and $S_j$ if $|M[i,j]| \geq \delta$
7: repeat
8:   if There is intersection between the cliques in $G$ with cardinality $\geq 2$ then
9:     $C \leftarrow$ the largest clique intersection in $G$
10:  else
11:    $C \leftarrow$ the largest clique in $G$
12:  end if
13:  $N \leftarrow$ node in $C$ with highest variance
14:  $G \leftarrow G \setminus (C \setminus N)$
15: until There is no more cliques in $G$
16: return $\mathcal{S}'$ list of all singletons nodes in $G$.
```

The output of the Algorithm 3.3 can be directly used or passed to the wrapper method described in Section 3.2.2 building a hybrid approach.

Continuing with the example graph G from Figure 3.7, we start by the state showed in Table 3.2, as the set $\mathcal{I} \neq \emptyset$, the algorithm looks into the entry with the greatest cardinality $I_1 = \{3, 4, 5\}$ and preserves the node with the highest term variance i. e. the node 5, discarding the others. This process is repeated, until the set $\mathcal{I}$ is empty, as showed in Figure 3.8 from iteration 1 to 3. When $\mathcal{I} = \emptyset$, the algorithm looks directly for the remaining cliques, as showed in Figure 3.8 from iteration 4 to 8. Table 3.3 describes the selected intersection/clique and the preserved node in each interaction.

Table 3.3 – Iterations of the clique-removal filter.

Iteration	Clique	Node Selected
1	{3, 4, 5}	5
2	{5, 6, 7}	6
3	{15, 16, 17}	17
4	{17, 18, 19}	17
5	{13, 14}	13
6	{1, 2}	2
7	{12, 13}	13

Finally, at the end we have as output a disconnected graph $G' = \{S', \emptyset\}$, where the vertices are the entries of the subset $\mathcal{S}' \subset \mathcal{S}$ of selected features (shapelets).

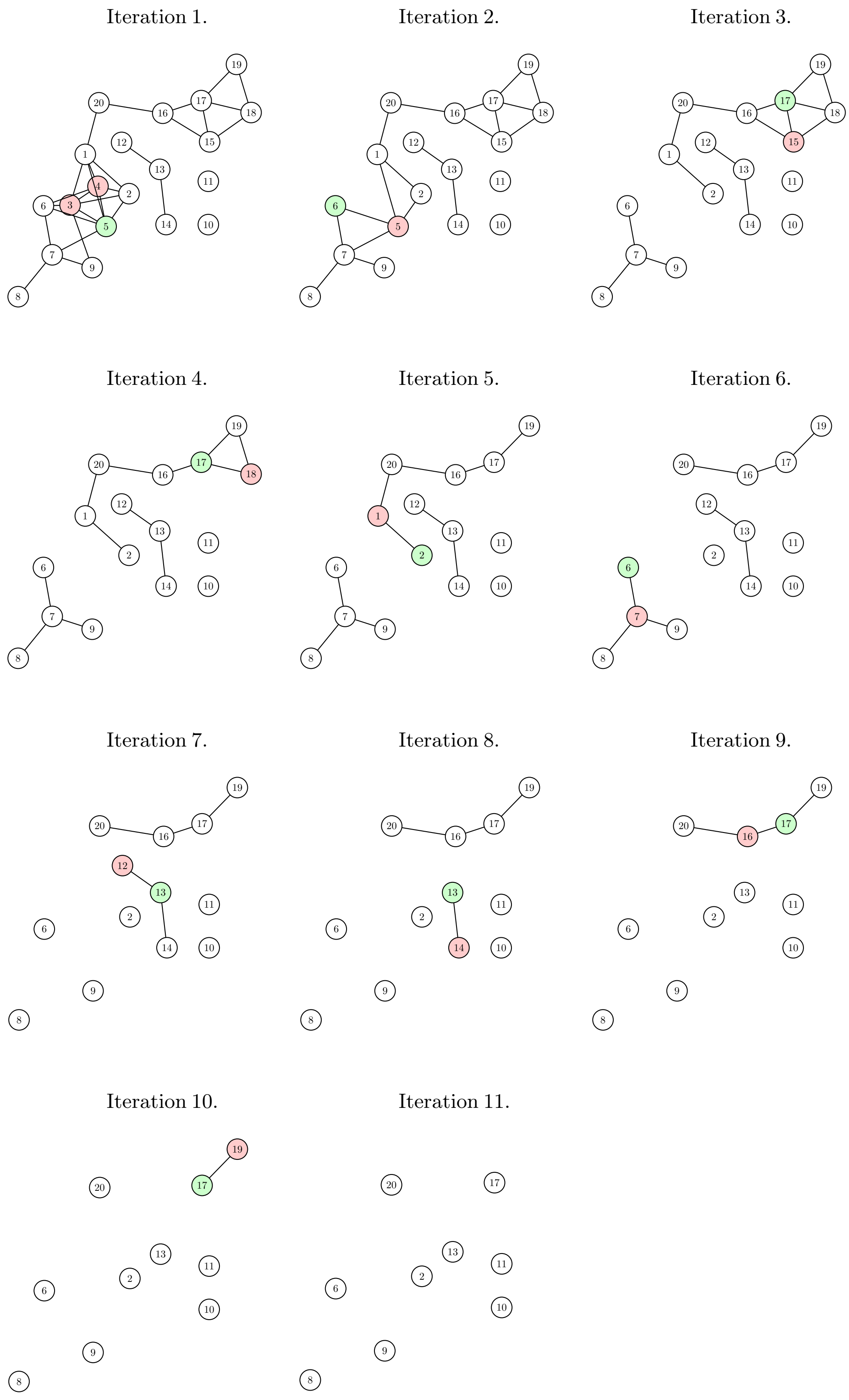


FIGURE 3.8 – Iterations of the clique-removal filter.

3.4 Extending to Multivariate Time Series

In this section, we extend the DPSR and the filtering algorithms that were described in the preceding sections to work with multivariate time series (MTS). In this way, the original approach for LDPS [LMTA17] needs also to be generalized. Hence, we are further describing three variations of the LDPS adding MTS support.

3.4.1 Multivariate Time Series: from DTW to Shapelets

As defined in Definition 2.2, multivariate time series (MTS), also know as multichannel or even multidimensional time series is a generalization of univariate time series, where each observation is not a scalar anymore but a vector in a M-dimensional space.

Basically, there are two ways to deal with MTS and DTW. One is reducing the M-dimensional multivariate time series into a univariate time series, by concatenating all M dimensions into a new $N \times M$ long univariate time series, or by applying some dimensionality reduction technique, like Principal Component Analysis (PCA) [JC16] [YS04]. The other is by generalizing the DTW algorithm to handle multivariate time series.

We have previously discussed in Section 2.3.1.2 two approaches for multivariate DTW: the $\mathrm{DTW_i}$ and the $\mathrm{DTW_d}$. As formerly explained, $\mathrm{DTW_d}$ handle all dimensions together in a single dependent warping path. In its turn, $\mathrm{DTW_i}$ has M-independent warping paths, one per dimension. Therefore, M DTW are computed, and the final similarity between the two time series amounts to summing the M obtained values.

These two solutions to use DTW to measure the similarity between pairs of time series, conduce us to propose three possible approaches for multivariate-LDPS, and consequently to DPSR.

3.4.2 DPSR and Filtering for Multivariate Time Series

The DPSR approach has so far been detailed in a setting where it copes with univariate time series, also the LDPS were detailed in Section 2.3.2.9 for the univariate context. This section describes how they can be extended to cope with multivariate time series.

We now detail three variants that differ in the way they treat the multiple dimensions of the time series to embed. The first variant captures the spirit of the approach followed by the $\mathrm{DTW_i}$, that is, M sets of univariate shapelets are learned and subsequently used. The second variant captures the spirit of the $\mathrm{DTW_d}$, in the sense that a set of multivariate shapelets is learned. The third variant blends the two others.

3.4.2.1 M-DPSR$_\mathrm{i}$: Independent Multivariate DPSR

This is the most straightforward modification. It consists of learning independently a set of shapelets per channel in the multivariate time series. At the end of learning step, we will have a hyper-dictionary Σ with M independently dictionaries, one per dimension.

$$\zeta = \{\mathcal{S}_1, \ldots, \mathcal{S}_M\} \tag{3.10}$$

Thus, the representation $\overline{\mathrm{T}}_i$ of T_i is given by:

$$\overline{\mathrm{T}}_\mathrm{i} = \bigcup_{m \in M} (\mathrm{ST}(T_i^m, \mathcal{S}_m)) = \bigcup_{m=1}^{M} \{M_{i,1}, \ldots M_{i,|\mathcal{S}_m|}\} \tag{3.11}$$

The selection of the top-shapelets is also done individually dimension-by-dimension, and the hyper-dictionary Σ' is defined as:

$$\zeta' = \{\mathcal{S}'_1, \ldots, \mathcal{S}'_M\} \tag{3.12}$$

Once the filtering is done independently, it can result in different number of shapelets in the dictionary for each dimension.

With this variant, the embedding is fully flexible in the sense that it is the best representation that is picked *independently* dimension per dimension. This variant is referred to being M-DPSR$_\mathrm{i}$ in the following, as it is in some way, similar to the idea of DTW$_\mathrm{i}$.

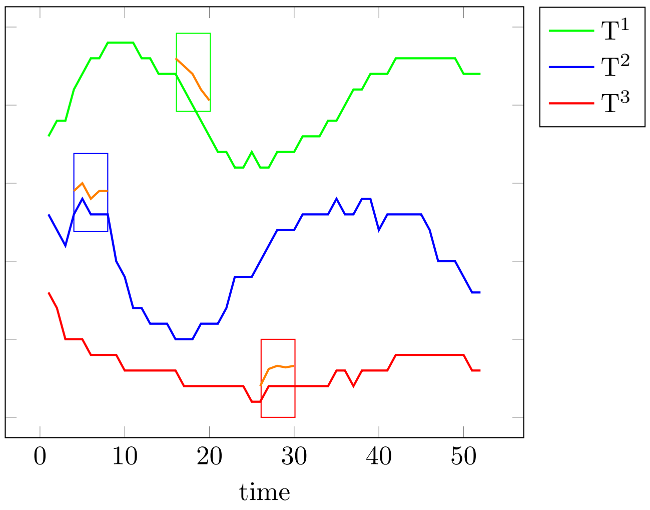


FIGURE 3.9 – The independent multivariate DPSR: each dimension is handled independently by its own sliding window and its associated learned $\mathcal{S}_m$ dictionary.

3.4.2.2 M-DPSR$_\mathrm{d}$: Dependent Multivariate DPSR

The second variant learns a set $\boldsymbol{\mathcal{S}}$ of K *multivariate* shapelets:

$$\boldsymbol{\mathcal{S}} = \{\mathrm{S}_1, \ldots, \mathrm{S}_K\} \tag{3.13}$$

It requires to generalize the approach by Lods *et al.* [LMTA17]: the loss function as well as the procedure to identify the best match have to take all dimensions into account. We need to generalize the definition Equation (2.23) by:

Definition 3.6 *Multivariate Euclidean score. The multivariate Euclidean score between a multivariate shapelet with M channels S_k, $|\mathrm{S}| = l$ and $\mathrm{T}_{i,p:l}$, a multivariate subsequence with length l starting at the position p, of a multivariate time series T_i with M channels defined as:*

$$D_{i,k,p} = \frac{1}{l}\sum_{x=1}^{l}\sum_{m=1}^{M}\left(T^m_{i,p+x-1} - S^m_{k,x}\right)^2 \tag{3.14}$$

Then, we can use the Equation (2.24) to obtain $\overline{\mathrm{T}}_i \in \mathbb{R}^{|\boldsymbol{\mathcal{S}}|}$

$$\overline{\mathrm{T}}_i = \mathrm{ST}(\mathrm{T}_i, \mathcal{S}) = \{M_{i,1}, \ldots, M_{i,|\boldsymbol{\mathcal{S}}|}\} \tag{3.15}$$

Finally, we can apply the greedy selection and the filtering procedure described above directly on $\boldsymbol{\mathcal{S}}$ to obtain $\boldsymbol{\mathcal{S}}'$.

Since all the dimensions are involved at once in this distance computation, this variant is referred to being the *dependent* M-DPSR$_\mathrm{d}$ in the following, once it is similar to the idea of warping dependency in DTW$_\mathrm{d}$. Figure 3.10 draws the multivariate sliding window over the time seties T_i.

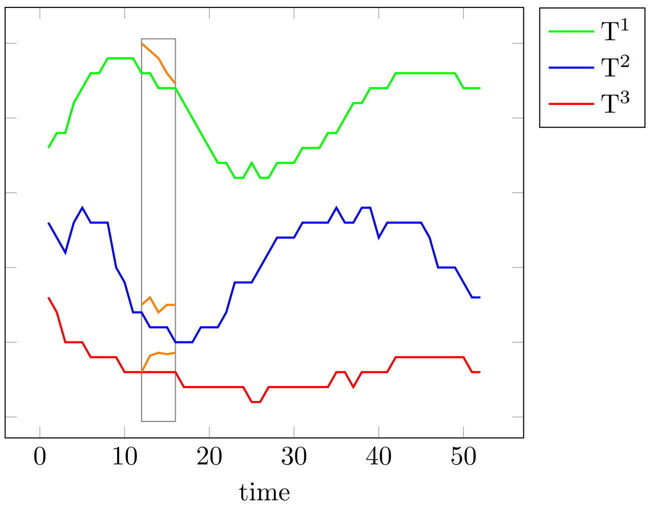


Figure 3.10 – The dependent multivariate DPSR: all dimension are handled together, consequently, we have just one (M-dimensional) sliding window.

3.4.2.3 M-DPSR$_\mathrm{s}$: Slack Multivariate DPSR

The third variant learns a set $\boldsymbol{\mathcal{S}}$ of multidimensional shapelets, just as with M-DPSR$_\mathrm{d}$. However, here the subsequence distance matching method emulates the behavior of the DTW$_\mathrm{i}$, i. e. the window slides independently per dimension, as in M-DPSR$_\mathrm{i}$.

It requires to generalize the approach by Lods *et al.* [LMTA17]: the loss function as well as the procedure to add support to multivariate shapelets. We need to generalize the definition Equation (2.23) by:

Definition 3.7 *Multivariate slack Euclidean score. The multivariate slack Euclidean score between a multivariate shapelet with M channels S_k, $|\mathrm{S}| = l$ and $\mathrm{T}_{i,p:l}$, a multivariate subsequence with length l starting at the position p, of a multivariate time series T_i with M channels defined as:*

$$D^m_{i,k,p} = \frac{1}{l}\sum_{x=1}^{l}\left(T^m_{i,p+x-1} - S^m_{k,x}\right)^2 \tag{3.16}$$

In this hybrid approach, embedding the multivariate time series means determining the sum of all M-minimal distances (one per dimension), therefore, the Euclidean shapelet match (Equation (2.24)) needs to be redefined as:

Definition 3.8 *Multivariate slack Euclidean shapelet match. The multivariate slack Euclidean shapelet match between S_k, with M channels, and T_i, with M channels, is defined as:*

$$M_{i,k} = \sum_{m=1}^{M} \min_{p\in\{1:N-l+1\}} D^m_{i,k,p} \tag{3.17}$$

In this way, the representation $\overline{\mathrm{T}}$ lies in $\mathbb{R}^{|\boldsymbol{\mathcal{S}}|}$, like in M-DPSR$_\mathrm{d}$. Figure 3.11 shows the independence of the sliding window in the M-DPSR$_\mathrm{s}$ approach.

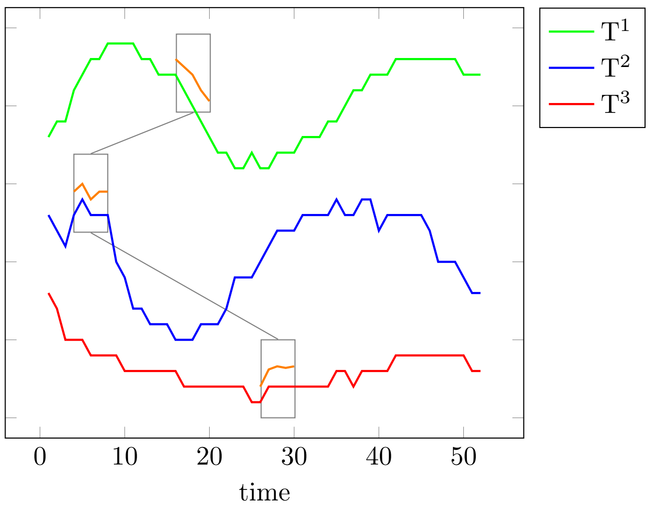


FIGURE 3.11 – The slack multivariate DPSR: the dimension are handled individually, however in the transformation step, the transformation is done by the accumulated sum per dimensions.

The motivation for this method is that we believe whilst the shapelet learned is dependent on the features being in phase, the places where they occur in other series could be independent of one another.

Chapter 4

Experiments

The best and safest method of philosophizing seems to be first to inquire diligently into the properties of things, and establishing those properties by experiments, and then to proceed more slowly to hypotheses for the explanation of them.

ISAAC NEWTON, *The Life of*

Contents

In this chapter we present the experiments and results of our proposed method for time series retrieval based on DTW-preserving shapelets.

The objective of the experiments it to evaluate the relevance of advanced feature (shapelet) selection in the DTW-preserving shapelet transformation on Time Series Retrieval (TSR) task. Firstly, we describe the datasets used, them the experimental setup and finally the obtained results.

4.1 Datasets

In this section we present both, the univariate and the multivariate datasets used to experimentally compare (M-)DPSR against the proposed competitors.

For the univariate case, we use the 85 datasets from the well-known UCR Time Series Archive [CKH+15]. Queries used for retrieval are the test sets series from the archive.

Before running any experiment, we built a full DTW-based groundtruth. The true DTW measurements between all time series pairs in each of the 85 families are determined. It is then straightforward to identify the nearest-neighbour of each time series.

In its turn, for the multivariate case, we use a subset with 20 datasets from the new UEA Multivariate Dataset Archive [BDL+18]. Again, the test set series are used as queries.

Once at least two multivariate approaches for DTW are commonly used, the DTW_i and the DTW_d, we have built for each dataset two grandtruths, one based on DTW_i and another based on DTW_d.

4.1.1 The UCR Time Series Classification Archive

The UCR Time Series Classification Archive [CKH+15] was built in order to improve the quality of papers using time series data. First arising in 2003 with 20 datasets, the UCR repository has grown to include 85 problems, each split into training and testing sets.

In spite of does not aim at representing all real-life problems, the UCR database provides, however, a large panel of problems, from very small datasets to larger ones. In Table 4.1 we can observe resumed details while the description per dataset is given by Table A.1.

Table 4.1 – UCR database in numbers.

	From	Up to	Average
Time Series Length (N)	25	2710	423.2
Training Set Size ($\lvert\mathcal{T}\rvert$)	16	8926	432.9
Testing Set Size	20	8236	1164.7
Ratio Train/Test Size	0.48	62.6	7.1

The datasets can be broadly separated into seven categories. These are image outline; sensor reading; motion capture; spectrographs; ECG measurements; electric devices and simulated datasets. In Table 4.2 we detail the number of problems in each type, these datasets have different characteristics and its elements (time series) can be subjected to temporal shifts, temporal distortions, noise, outliers, among others distortions in the data.

Table 4.2 – Number of datasets in UCR by problem type.

Type of problem	N° of datasets
Image Outline	29
Sensor Reading	16
Motion Capture	14
Spectographs	7
ECG Meassurements	7
Electric Devices	6
Simulated	6
Total	85

4.1.2 The UEA Multivariate Time Series Classification Archive

The UEA multivariate time series classification archive [BDL+18] was introduced in 2018 and provides 30 multivariate time series datasets with a wide range of cases, dimensions (channels) and series lengths.

The most part of the dataset can be classified into five groups whereas three of them are not easily labeled in some group. Table 4.3 shows the number of datasets per class of problem.

Tables 4.4 and 4.5 give the sizes of training and test sets as well as the number of dimensions, length and the “total dimensionality” which is the amount of data per multivariate time series, i. e. the number of dimensions times the time series length.

Table 4.3 – Number of datasets in UEA by problem type.

Type of problem	Nº datasets available	Nº of datasets evaluated
Human Activity Recognition	9	8
Motion Classification	4	2
ECG Classification	2	2
EEG/MEG Classification	6	4
Audio Spectra Classification	6	2
Other Problems	3	2
Total	30	20

Table 4.4 – The full UEA database in numbers.

	From	Up to	Average
Time Series Length (N)	8	17984	1073.4
Training Set Size ($\|\mathcal{T}\|$)	12	30000	2038.3
Testing Set Size	15	20000	1359.0
Time Series Dims (M)	2	1345	98.5
Total Dim	16	363150	30130.2

The Learning-DTW preserving algorithm has a run-time complexity of $O\left(maxI \cdot M(N^2 + l \cdot K)\right)$, where $maxI$ is the number of iterations on the algorithm, l is the length of the shapelet and K is the number of shapelets learned. Thus, its many of the multivariate datasets in the full UEA database are infeasible to learn with at least 100 iterations of the algorithm. For example, on the DuckDuckGeese dataset, where the “total dimensionality” is 363150, the algorithm needs $\approx$ 300 days to complete 100 iterations steps.

On the other hand, the wrapper algorithm has a time complexity of $O\left(M \cdot N \cdot K^3 \cdot |\mathcal{T}|^2 \log |\mathcal{T}|\right)$, and even using the proposed filter approach, the algorithm stills unfeasible, for example, for the `InsectWingbeat` dataset, which $|\mathcal{T}|$ is 30000.

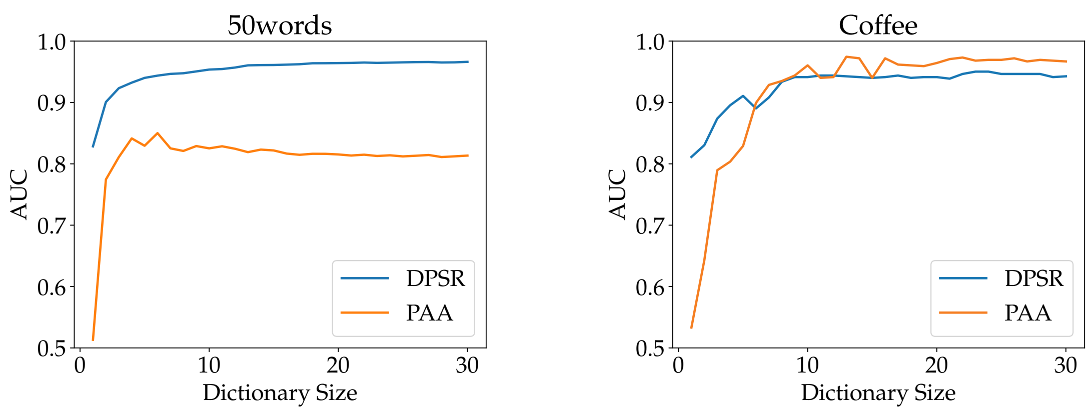
Thereafter, to complete our experiments we have used a subset with 20 (of 30) datasets of the UCE multivariate dataset, described in Table 4.5. The full description per dataset is done in Table A.2.

Table 4.5 – The subset of 20 datasets from the UEA database in numbers.

	From	Up to	Average
Time Series Length	29	2500	457.5
Training Set Size	12	2459	350.5
Testing Set Size	15	2466	384.0
Time Series Dims	2	61	10.1
Total Dims	90	24705	3342.9

4.2 Experimental Setup

Our experiments are performed on a 24-core 2.8 CHz Intel Xeon ES-2630 with 64 GB of memory. All algorithms and structures are implemented in `Python3`, `Cython` and `NumPy`. Although



(a) (b)

FIGURE 4.1 – Comparing DPSR and PAA for two specific time series. Observed AUC as the dictionary size increases.

the machine has 24 cores, the only operations using parallelism are the distance computations handled by `NumPy` during the course of each experiment. No other parallelism is enforced. The `TSLearn` [Tav17] toolkit was used for the computation of PAA and LB_Keogh.

For the PAA and LB_Keogh, the parameters w (word size) and r (window length) are learned by 10-fold cross-validation.

In its turn, for the LDPS algorithm, a set of DTW-preserving shapelets is learned using the algorithm proposed in [LMTA17], with default parameters. To learn high-quality shapelets, 500,000 iterations of the gradient descent algorithm are performed. In addition, two sets of shapelets are learned for each family of time series, and the set with the smallest overall loss is selected.

4.3 Dictionary Size vs. Accuracy: Reaching a Plateau

This first experiment compares the performance of PAA and DPSR for univariate time series retrieval by recording the observed AUC as the size of the dictionary of shapelets increases.

Here, all shapelets are kept, none are filtered out, it is thus DPSR_f that is used. Transformations range from very rough approximations when using very few shapelets to finer grain representations when using a larger number of shapelets. For consistency, we compared PAA and DPSR for the same sizes, that is, the number of shapelets used to create the dictionary is equal to the number of pieces for PAA.

Figure 4.1 illustrates the behavior for two selected datasets, `50words`, and `Coffee`. Those two dataset are well representative of the scenarios observed in the 85 datasets analyzed.

With `Gun_Point`, DPSR outperforms PAA for all dictionary sizes. The results for `Coffee` are more contrasted: when the time series is split into more than 7 pieces, PAA outperforms the shapelet based approach. Please note that these figures show quality measures for the first 30 elements of the dictionary only, larger dictionaries providing no improvements.

Table 4.6 compares PAA and DPSR for all 85 time series by counting the number of times each method performs better than the other, along with the observed average AUC values for dictionaries made of 5, 10, 20 and 30 shapelets (or pieces in the case of PAA). Overall, this table shows that DPSR consistently outperforms PAA for all dictionary sizes. We can also observe that both methods seem to reach an AUC plateau, sooner for PAA than it is for DPSR.

For DPSR, this plateau highlights the importance of selecting a subset of all learned shapelets. It indicates that Lods *et al.* learn too many shapelets, and that a lot of them do not contribute significantly to enhancing the resulting vectorial representation when considering retrieval. Sliding them at transform time is therefore a waste of resources. This is important especially because reducing the cost of transforming the query time series is paramount.

Table 4.6 – Comparing DPSR and PAA for the full UCR Archive, varying dictionary sizes. Number of times each method outperforms the other in terms of AUC. Average AUC values over the 85 datasets.

Dictionary Size	5	10	20	30
# DPSR wins	70	69	73	72
# PAA wins	15	16	12	13
Average AUC for PAA	0.838	0.847	0.844	0.841
Average AUC for DPSR	0.906	0.921	0.929	0.932

For the multivariate case, as presented in Section 3.4 we have three possible approaches for the M-DPSR: M-DPSR$_\mathrm{i}$, M-DPSR$_\mathrm{d}$, and M-DPSR$_\mathrm{s}$. Considering the three proposed approaches and the multivariate implementation of PAA we have observed a similar behavior, that is, a plateau is observed. Figure 4.2 shows the behavior for four selected datasets, `AtrialFibrilation`, `SelfRegulationSCP1`, `CharacterTrajectories`, and `ERing`. Here we need to highlight that while for the M-DPSR each entry S in the dictionary is in $\mathbb{R}$, for (M)PAA, each piece W is in $\mathbb{R}^M$, where M is the number of dimensions (channels) in the raw time series.

4.4 Shapelet Selection Strategies for DPSR

In Section 3.2.2, strategies for stopping aggregating selected features were presented. We now evaluate their effectiveness, which is a trade-off between their accuracy in terms of AUC and the transformation cost they cause.

Typically, small subsets allow for very fast transforms (just a few shapelets need to be slid at test time) but quality is typically low, whereas in contrast larger subsets improve AUC but cause more expensive transform operations. Here, no irrelevant shapelets are filtered out.

To observe this trade-off, we selected by cross-validation on the training sets of each dataset the best shapelet subset for the different stopping criterion. We then used these subsets to transform test series for retrieval. The corresponding AUC that are observed for the 85 datasets from the archive with each stopping strategy are recorded and averaged. The resulting AUC value is given in Table 4.7, left, together with the average number of selected shapelets. The last line of Table 4.7 (DPSR$_\mathrm{f}$) corresponds, in the column “No filtering”, to the case where no shapelet selection is performed (i. e. the whole shapelet set learned beforehand is used).

Table 4.7 – Average AUC and dictionary sizes for feature subset selection strategies. With and without filtering out irrelevant shapelets.

	No filtering		Filtering	
	AUC	Dict. size	AUC	Dict. size
DPSR$_\mathrm{t}$	0.906	4.4	0.899	3.8
DPSR$_\mathrm{l}$	0.928	27.4	0.929	25.2
DPSR$_\mathrm{g}$	0.935	54.0	0.935	41.5
DPSR$_\mathrm{f}$	0.934	156.1	0.931	95.9

We can observe a trade off between the accuracy of the retrieval (AUC) and the computational time of the transform (linear with the number of shapelets). DPSR$_\mathrm{t}$, the most aggressive

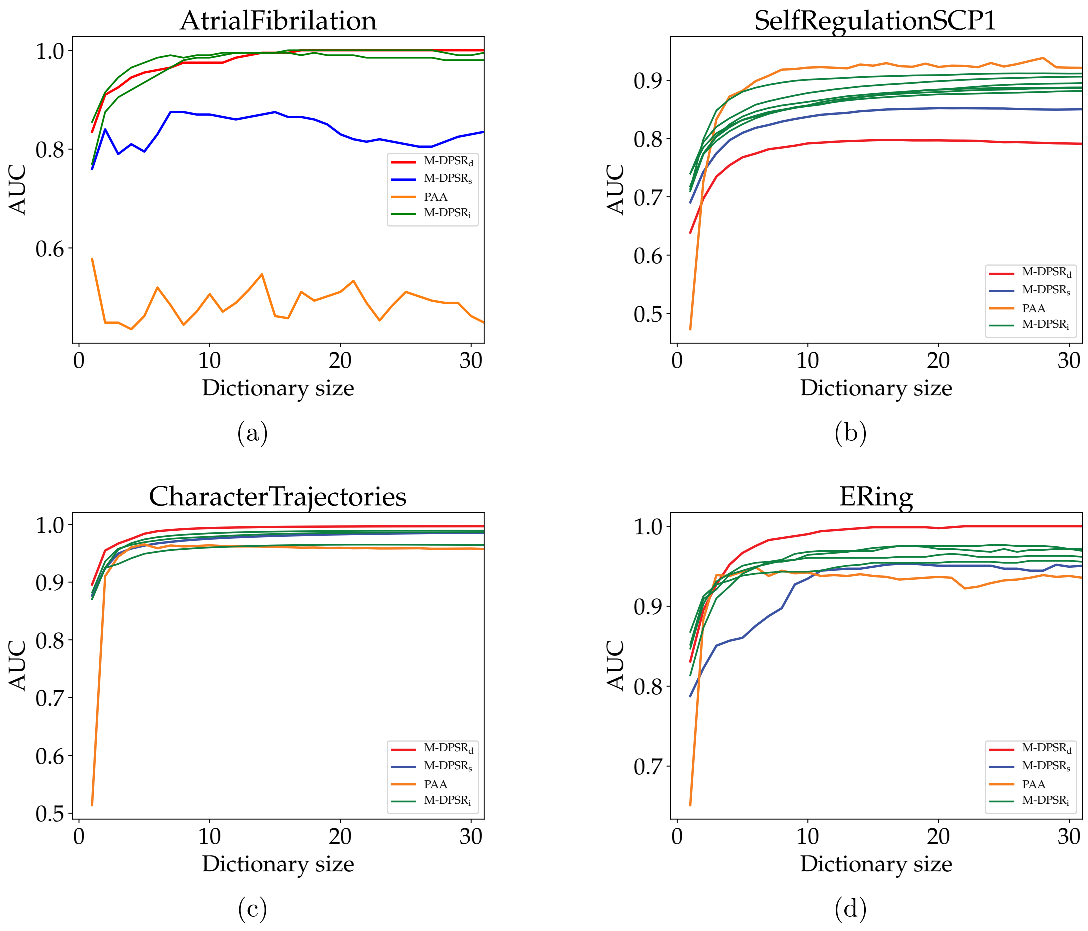


FIGURE 4.2 – Comparing the three approaches for M-DPSR and (M)PAA for four specific multivariate time series. Observed AUC as the dictionary size increases. Note that we have more than one entry for the M-DPSR$_i$ (green), as in this approach we apply the DPSR per dimension of the raw time series.

strategy, selects very few shapelets (a little bit more than four, on average) for an average AUC of 0.906. DPSR$_g$, more conservative, uses on average 54 shapelets, but the corresponding quality improvement is quite small: it goes from 0.906 to 0.935. This is a clear illustration of the trade off, also exemplified by the DPSR$_l$ strategy, which is in between these two other strategies.

We can also observe the importance of the feature selection itself: when no selection is made (DPSR$_f$), the average AUC is slightly lower than for DPSR$_g$ whereas the corresponding average number of shapelets is almost tripled.

4.5 Filtering out Before Selecting

We now test the impact on the accuracy when filtering out irrelevant shapelets before selecting them. Here, the correlation threshold is set to $\delta = 0.8$ and the term variance percentile is set to $\alpha = 0.5$. We again involve the full UCR archive.

The right-hand side of Table 4.7 gives the resulting observed AUC and dictionary sizes. It is to contrast with the left part of the same table where no filter is applied. For DPSR$_t$, it is observed a reduction of 15.6% in the average size of the dictionary with a decrease of 0.01% in the accuracy. For DPSR$_l$ and DPSR$_g$ the size of the dictionary compared to the non filtered case shrinks by 23.8% while preserving the average accuracy. Filtering and keeping all remaining shapelets with DPSR$_f$ reduces the dictionary size by 38.8% and decreases the accuracy by 0.03%. Filtering is very effective.

The severity of the filter mainly depends on δ, the threshold which determines if an edge will be present or not in the initial graph G. A smaller value for δ increases the number of edges in G, which in turn creates more cliques resulting in more aggressively filtering out shapelets. A higher value for δ gives fewer options when reducing cliques. Overall, we experienced with values for δ ranging between 0.3 and 0.9. The value 0.8 provide the best results.

4.6 Feature Selection vs. Instructed Feature Learning

The previous experiment demonstrated that only a small fraction of the learned shapelets are truly useful because they significantly contribute to the quality of the retrieval. We show here that it is the combination of the learning stage and the feature selection strategy that leads to such behavior. For that purpose, we compare the AUC performance of non-filtered DPSR$_t$ and DPSR$_l$ with a method where [LMTA17] is instructed to learn the same number of shapelets.

We know from the previous experiments that the average AUC for DPSR$_t$ over all the datasets is 0.906, with using 4.4 shapelets on average. When directly learning that same number of shapelets, then the average AUC over all datasets is 0.866. More precisely, DPSR$_t$ performs better in 74 cases (out of 85). Same conclusions can be drawn with DPSR$_l$. The average AUC of DPSR$_l$ is 0.928, whereas the AUC obtained when directly learning the same number of shapelets is equal to 0.905. DPSR$_l$ wins 71 times out of 85.

These results indicate that it is worth spending more time offline to learn a large set of shapelets and then selecting the more appropriate. This is better than trying to save computational time by learning fewer shapelets. Also, our feature selection strategy allows deciding on the number of shapelets in a data driven fashion, and not just heuristically.

4.6.1 Comparing Methods at Their Best

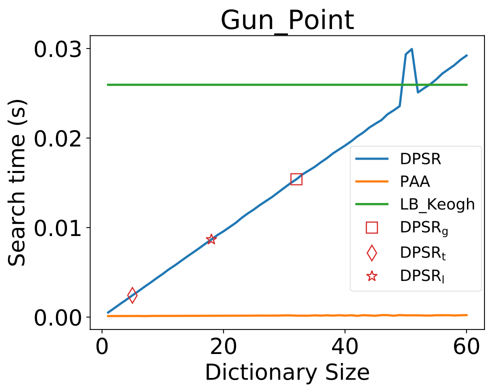
TABLE 4.8 – Comparing DPSR, PAA and LB_Keogh with their best parameters. Number of times each method outperforms the other in terms of AUC.

	DPSR vs. PAA				DPSR vs. LB_Keogh			
	# wins for DPSR		# wins for PAA		# wins for DPSR		# wins for LB_Keogh	
DPSR$_t$	58	(0.910, 4.5)	27	(0.866, 38.8)	34	(0.910, 4.5)	51	(0.908, —)
DPSR$_l$	69	(0.933, 29.8)	16	(0.866, 38.8)	64	(0.933, 29.8)	21	(0.908, —)
DPSR$_g$	75	(0.938, 54.6)	10	(0.866, 38.8)	67	(0.938, 54.6)	18	(0.908, —)
DPSR$_f$	79	(0.935, 140.9)	6	(0.866, 38.8)	67	(0.935, 140.9)	18	(0.908, —)

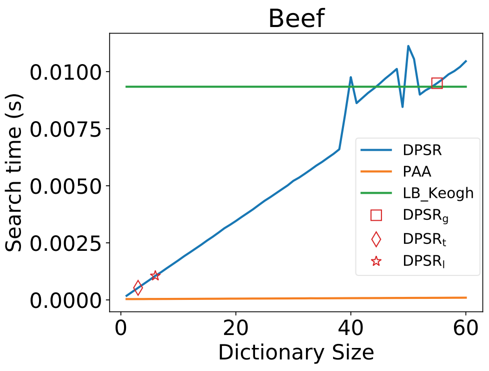
In this experiment, we compare the respective performance of DPSR, PAA and LB_Keogh when their parameters (number of segments for PAA, subset of shapelets for DPSR and window length for LB_Keogh) are cross-validated on the train set. The results comparing the performance of (i) DPSR and PAA methods and (ii) DPSR and LB_Keogh are given in Table 4.8. This table gives the number of times each method wins over the other, with the corresponding AUC and dictionary size between parentheses. Overall, DPSR outperforms PAA even for the aggressive DPSR$_t$ strategy.

Comparing the representation sizes when DPSR or PAA are winning is insightful. Consider for example DPSR$_t$, winning over PAA 58 times. Among these 58 wins, in 48 cases, the dictionary needed for DPSR$_t$ contains fewer items than the number of pieces for PAA. The dual point of view is also insightful: PAA outperforms DPSR$_t$ in 27 cases, but all PAA transformations need more pieces than DPSR$_t$ has elements in its dictionary. Not only DPSR wins more frequently than PAA, but when it wins, it is with shorter representations. This is also true for $DPSR_l$ and $DPSR_g$. The average AUC value for PAA is equal to 0.866 which is worse than the average AUC value of the three DPSR approaches. Against LB_Keogh, only $DPSR_t$ looses. $DPSR_l$ and $DPSR_g$ win over LB_Keogh by a large margin. Note also that, unlike LB_Keogh and PAA, DPSR allows comparing time series of different lengths. Overall, we observed that DPSR more often wins over its competitors when features are not filtered. Using more features maintains quality.

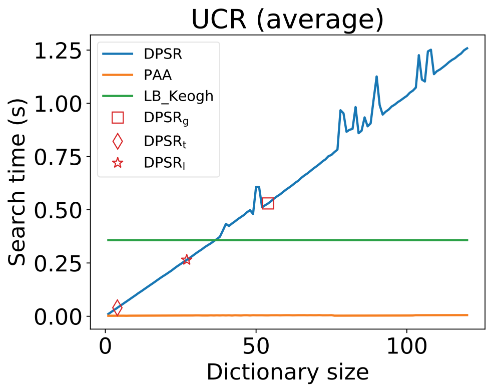
4.6.2 Search Costs



(a) Gun_Point



(b) Beef



(c) UCR

FIGURE 4.3 – Average Search Times for two specific time series and for the full UCR Archive. Varying dictionary size.

So far, only quality comparisons have been done. We observe now the computational costs of the approaches discussed here. We focus on the case where shapelets are not filtered out, which is the worst case in terms of performance. Experiments were performed for recording search times when the numbers of elements in the dictionary for DPSR and the number of pieces for PAA gradually increase. Search times for `Gun_Point` and `Beef` are plotted on Figure 4.3 (limited to size ≤ 60), and the average search times for the full UCR archive is on Figure 4.3c.

These figures also show the time it takes to only compute the envelopes of time series for LB_Keogh. That process, if then computing DTWs, is guaranteed to identify the correct nearest time series. The quality of any approximate search scheme can only be equal or lower. The time for solely computing the LB_Keogh value on all time series is the absolute minimal cost the real LB_Keogh+DTW could have.

TABLE 4.9 – Comparing M-DPSR$_d$, M-DPSR$_s$ and M-DPSR$_i$ ran against 20 datasets from UEA archive. Average AUC, dictionary size. Average ranked performance w.r.t. AUC. Number of Wins.

Method	AUC	Dict. Size	Avg. Rank	# Wins
M-DPSR$_d$	0.843	62.1	1.8	7
M-DPSR$_s$	0.819	62.1	2.0	6
M-DPSR$_i$	0.773	587.5	2.2	8

These figures show that the time for computing envelopes with LB_Keogh is fixed, which is normal. They also show that PAA is the fastest approach. Its underlying principles are simple and cause light computations, the search times increasing slightly with the number of pieces. The time taken per search for DPSR is also increasing with the dictionary size: there are more and more shapelets to slide, and distance computations are more demanding. Three remarkable signs are placed on the search time plot for the DPSR approach. They refer to the search times observed when the number of shapelets in use correspond to what DPSR$_t$, DPSR$_l$ and DPSR$_g$ determined.

4.7 Multivariate Transformation versus Accuracy

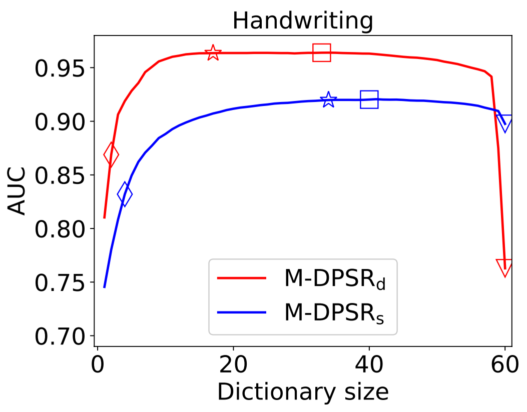
This first experiment observes the performance of the three variants for M-DPSR, in terms of AUC versus the size of the shapelet dictionary. To fairly compare M-DPSR$_i$ with M-DPSR$_s$ and M-DPSR$_d$, a comment is needed. M-DPSR$_s$ and M-DPSR$_d$ use multivariate shapelets whereas M-DPSR$_i$ uses M sets of univariate shapelets. Hence, by construction, the size of the dictionary for M-DPSR$_i$ is high compared to the two other variants.

Table 4.9 shows how accuracy and the size of the dictionary to transform time series relate. This table shows that M-DPSR$_d$, despite having the best average AUC, is the method with fewer wins. M-DPSR$_i$, however, wins more often but has the lowest average AUC. These results are reflected in the average rank, which is approximately 2.0 for all three variants. M-DPSR$_i$ needs approximately seven times more shapelets, as expected. The variants better differentiate once shapelets are selected and filtered, as shown next.

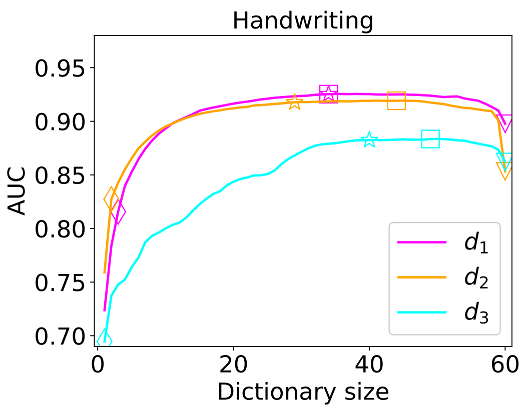
4.8 Selection, Filtering for M-DPSR

Overall, an AUC quality plateau is observed with multivariate time series, as it was observed with univariate series. Figure 4.4 shows how the AUC varies as the size of the dictionary increases for the specific `Handwriting` dataset. Figure 4.4a shows the cases where M-DPSR$_d$ and M-DPSR$_s$ are used, and when no irrelevant shapelet is filtered. Figure 4.4b shows the performance fot the M-DPSR$_i$ strategy. In this case, the `Handwriting` dataset is a 3-dimensional time series. For that reason, the Figure 4.4b shows three curves, one for each of the three dimensions. On all curves, the dictionary sizes corresponding to applying the stopping criterion for aggregating shapelets are indicated. Note that in the case of M-DPSR$_i$, the size on the x-axis corresponds to the dictionary for *one* dimension. The total number of shapelets used to transform one multivariate time series with M-DPSR$_i$ is the sum of the size of each per-dimension dictionary. For example, M-DPSR$_i$ applying the '**t**' stopping criterion result in picking 9 shapelets, 4 for dimension 1, 3 for dimension 2 and 2 for dimension 3. When the '**l**' stopping criterion is used then it results in 106 dimensions split in $(35, 30, 41)$. With '**g**', it is 130 with $(35, 45, 50)$.

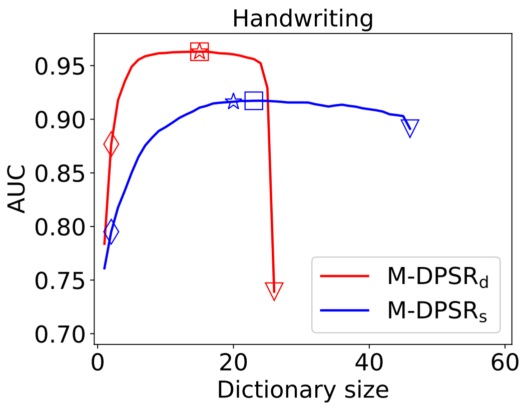
The average AUC values and average dictionary sizes over all multivariate datasets are given in Table 4.10. The columns '**t**', '**l**', '**g**' and '**f**' correspond respectively to the aggregation stopping criteria defined in Section 3.2.2.1. Here again, we observe a similar trade-off between accuracy



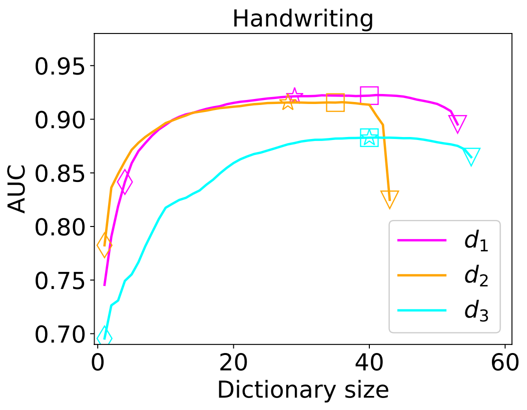
(a) M-DPSR$_\mathrm{d}$ and M-DPSR$_\mathrm{s}$



(b) M-DPSR$_\mathrm{i}$



(c) M-DPSR$_\mathrm{d}$ and M-DPSR$_\mathrm{s}$, with filter



(d) M-DPSR$_\mathrm{i}$, with filter

FIGURE 4.4 – Quality when number of shapelets varies for the `Handwriting` dataset. 4.4c and 4.4d: irrelevant shapelets are filtered out.

for retrieval time: few shapelets cause fast transforms but quality is lower compared to using more shapelets at a higher transform cost.

That same table shows also the quality and the dictionary sizes when filtering out irrelevant features. The number of preserved features is often very much reduced without observing any dramatic loss in quality. Figures 4.4c and 4.4d illustrate the effects of this filtering on the `Handwriting` time series. Figure 4.4c shows the effect of filtering for the M-DPSR$_\mathrm{d}$ and M-DPSR$_\mathrm{s}$ cases. Figure 4.4d shows the effect of filtering on each of the three dimensions for the M-DPSR$_\mathrm{i}$ case.

4.9 Comparing Methods at Their Best for Multivariate Data

It is also insightful to compare the performance of the multivariate methods at their best, i. e. when their parameters have been cross-validated on the training set. Here, the variants of M-DPSR are compared to the multivariate version of PAA and multivariate version of LB_Keogh. For each dataset, we record the performance when the best number of segments is used for PAA, the best window length for LB_Keogh. This performance is then compared to the best variant of M-DPSR when the *global* stopping criterion is used, and also to the best variant of M-DPSR using the best stopping criterion, denoted by M-DPSR$_\mathrm{best}$.

Figure 4.5a shows the results of comparing M-DPSR against PAA. M-DPSR$_\mathrm{d}$ outperforms PAA for 64.3% of the datasets. M-DPSR$_\mathrm{i}$ and M-DPSR$_\mathrm{s}$ are more on par with PAA. Yet,

Table 4.10 – Average AUC and dictionary sizes for multivariate feature selection strategies, with and without filtering irrelevant shapelets.

		t		l		g		f	
		AUC	Dim.	AUC	Dim.	AUC	Dim.	AUC	Dim.
M-DPSR$_d$	w. Filter	0.737	2.7	0.866	20.6	0.869	23.4	0.839	39.2
	w/o. Filter	0.787	3.4	0.866	24.2	0.871	29.2	0.843	62.1
M-DPSR$_s$	w. Filter	0.748	2.7	0.846	16.4	0.852	20.2	0.822	34.9
	w/o. Filter	0.767	3.3	0.836	20.0	0.848	26.1	0.819	62.1
M-DPSR$_i$	w. Filter	0.666	23.6	0.733	175.0	0.743	201.8	0.731	332.0
	w/o. Filter	0.728	31.2	0.774	226.9	0.748	303.4	0.773	586.5

PAA needs an average of 336.2 segments while M-DPSR$_d$, M-DPSR$_s$ and M-DPSR$_i$ need a dictionary of size 30, 27.9, and 243.6 respectively. Figure 4.5c is making the same comparison, using LB_Keogh, however.

Figures 4.5b and 4.5d show the cases where the best possible behavior for M-DPSR is compared to the best behaviors for both PAA and LB_Keogh. These figures clearly show that M-DPSR very often outperforms its competitors.

Table 4.11 summarizes the results presented using the best parameters for each method.

Table 4.11 – Comparing M-DPSR$_{best}$, PAA and LB_Keogh with their best parameter. Number of times each method outperforms the other, average ranking from AUC.

Method	# Wins	Avg. AUC	Avg Dims	Avg. Rank
M-DPRS$_{best}$	13	0.9019	105.1	1.65
PAA	6	0.8307	1668.7	2.30
LB_Keogh	3	0.8829	3342.9	2.05

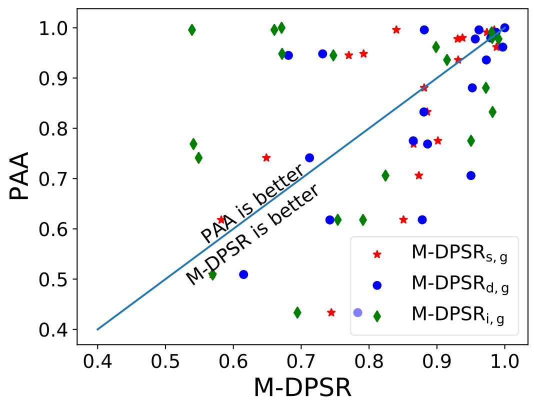
4.10 Search Costs on Multivariate Data

Several experiments are performed in order to compare the execution time of aforementioned methods. In these experiments, the execution times (in seconds) for each strategy against each dataset are reported in Table 4.12. Figure 4.6 shows the performance for three specific time series. It should be emphasized again that we are measuring the time for solely computing the LB_Keogh, which is the absolute minimum cost that real LB_Keogh+DTW could have.

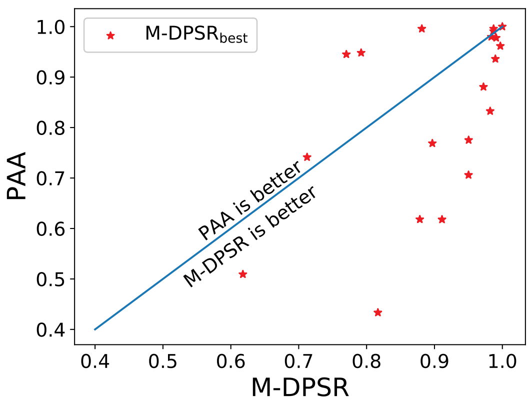
4.11 Discussion

Since the first paper on shapelets from Ye and Keogh, many approaches using time series shapelets were proposed. With this work we have extended the use of shapelets to cover the retrieval task, and few lessons can be drawn from our experiments.

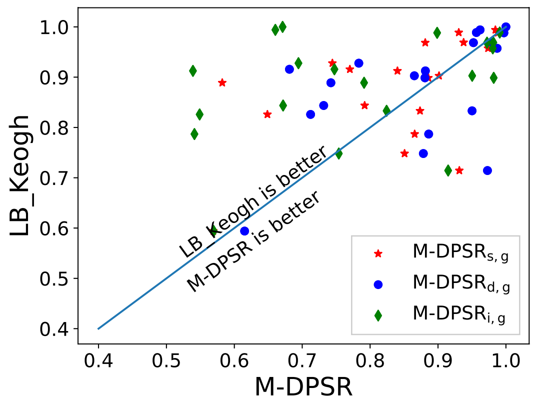
(i) It is extremely profitable to rank and then select a few of the shapelets learned by the state-of-the-art DTW-preserving Shapelet Transform by Lods *et al.* Even the most aggressive subset creation strategy that typically select very few shapelets (4.4 on average when considering the full UCR Time Series archive) provides very good approximations of the true DTW between time series, at a very low computational cost.



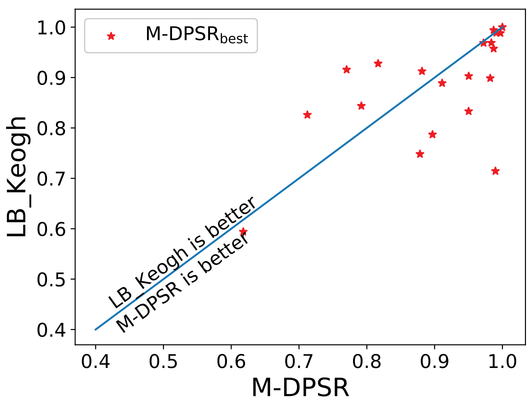
(a) M-DPSR (global)× PAA



(b) M-DPSR$_{\text{best}}$ × PAA



(c) M-DPSR (global)× LBK



(d) M-DPSR$_{\text{best}}$ × LBK

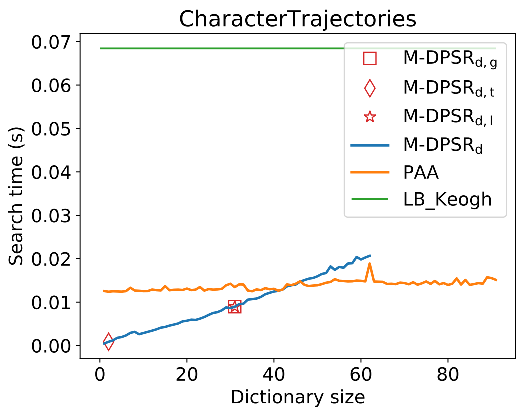
FIGURE 4.5 – Accuracy comparison (AUC) of M-DPSR variants using the global stopping criterion against the multivariate PAA (4.5a) and against the multivariate LB_Keogh (4.5c) for all datasets described in Table A.2. In 4.5b and 4.5b we choose the best approach for M-DPSR.

(ii) In the multivariate tasks, there is not clear a winner between M-DPSR$_{\text{d}}$, M-DPSR$_{\text{s}}$, and M-DPSR$_{\text{i}}$, highlighting the need of an adaptive approach.

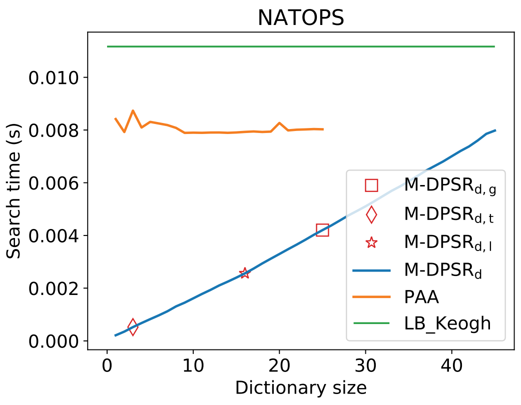
Considering the growing interest in time series data mining and specially in shapelets, this work could serve as a basis for future work on time series retrieval.

TABLE 4.12 – Average execution time (in seconds) of search over independent runs for each method.

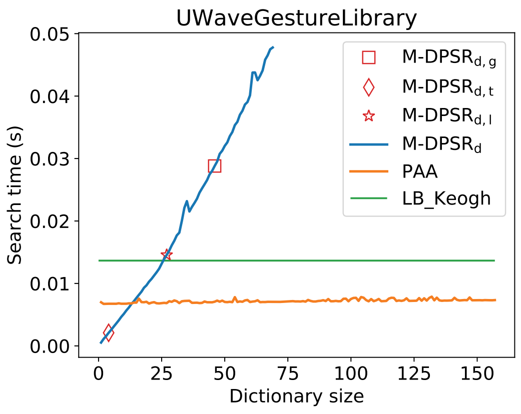
Dataset	M-DPRS$_t$	M-DPRS$_l$	M-DPRS$_g$	M-DPRS$_f$	PAA	LB_Keogh
ArticularyWordRecognition	0.0009	0.0208	0.0208	0.0210	0.0080	0.0220
AtrialFibrilation	0.0058	0.0116	0.0260	0.1910	0.0009	0.0120
BasicMotions	0.0008	0.0032	0.0042	0.0112	0.0005	0.0029
CharacterTrajectories	0.0009	0.0089	0.0089	0.0207	0.0151	0.0613
Epilepsy	0.0013	0.0091	0.0091	0.0242	0.0015	0.0083
EthanolConcentration	0.0843	0.2925	0.4117	2.0122	0.0098	0.0617
ERing	0.0006	0.0023	0.0032	0.0074	0.0004	0.0020
FingerMovements	0.0005	0.0038	0.0038	0.0086	0.0069	0.0073
HandMovementDirection	0.0109	0.0338	0.0338	0.1409	0.0120	0.0094
Handwriting	0.0007	0.0039	0.0082	0.0157	0.0043	0.0129
Heartbeat	0.0346	0.0806	0.1037	1.0997	0.0483	0.0457
JapaneseVowels	0.0004	0.0028	0.0034	0.0047	0.0058	0.0128
Libras	0.0004	0.0024	0.0027	0.0050	0.0015	0.0066
LSST	0.0004	0.0048	0.0048	0.0050	0.0210	0.0880
NATOPS	0.0005	0.0026	0.0042	0.0080	0.0080	0.0092
RacketSports	0.0004	0.0029	0.0029	0.0046	0.0013	0.0057
SelfRegulationSCP1	0.0227	0.0733	0.0733	0.5766	0.0087	0.0352
SelfRegulationSCP2	0.0320	0.7016	1.0267	1.4266	0.0106	0.0380
StandWalkJump	0.0860	0.3780	0.7593	6.1917	0.0394	0.0940
UWaveGestureLibrary	0.0021	0.0146	0.0288	0.0478	0.0073	0.0142



(a) Character_Trajectories



(b) NATOPS



(c) UWaveGestureLibrary

FIGURE 4.6 – Average Search Times for three specific multivariate time series. Varying dictionary size.

Chapter 5

Conclusions and Perspectives

"Stories don't end", he says.
"They just turn into new beginnings."

Lindsay Eagar, *Hour of the Bees*

Contents

Time series data can be found in many domains resulting in an explosion of interest in mining time series data in the last two decades. A wide number of algorithms have already been proposed to help with the time series retrieval task

The work in this thesis was initially aimed at using DTW-preserving Shapelets as a data transformation for the Time Series Retrieval (TSR). As presented in Section 2.3.2, in time series classification, shapelet based methods have been shown to be a very effective model for classification. Here, we are proposing, for the first time in the literature, the use of shapelets in other tasks than classification or clustering.

We initially wanted to answer two question: can we make the learning DTW-preservin shapelet transform efficient to the retrieval task? And, can we make the DTW-preserving shapelet transform faster without reducing accuracy?

The third aim of this thesis was, can we deal with multivariate time series? It is possible to be efficient in this context?

5.1 Results Summary

The first contribution of this thesis was improving the DTW-preserving shapelet transform to the retrieval task (DPSR). In our framework, DTW-preserving shapelets are learned, and then a subset is used to create a vectorial representation of the raw time series. The idea is that the ranked euclidean distances in the embedded space reflect the ranked distances measured by the DTW on the original space. In this way, we aim to preserve the quality of the retrieval,

We strive to maintain the quality of time series retrieval while the retrieval cost is reduced, therefore facilitating time series retrieval at scale. To this end, we propose a method to analyze the quality of shapelets subset, and how that subset can be chosen. These proposed improvements were designed to reduce the run-time while the accuracy of the retrieval task is preserved.

In our experiments, we observed that the original LDPS approach produces an excessive number of shapelets. If on the one hand, the high number of learned shapelets helps the transformation in approaching the value of the DTW, on the other hand, it makes indexing difficult by increasing dimensionality and slowing down the transformation, as more sliding window steps become necessary.

Considering that the goal of Lods *et al.* in [LMTA17] is, in short, to minimize the loss given by the average difference $L_2(\overline{T}_i, \overline{T}_j) - \mathrm{DTW}(T_i, T_j)$, increasing the number of shapelts can be opportune to minimize the average transformation loss. However, in this original work, by construction, the authors do not do any distinction between the contributions from the pairs $\langle T_i, T_j \rangle$ and $\langle T_i, T_k \rangle$, whereas $\mathrm{DTW}(T_i, T_j) \lll \mathrm{DTW}(T_i, T_k)$, i. e. there is no contrast between the input from the closest or farthest pairs.

Hence, our hypothesis is the transformation based on the set of learned shapelets $\mathcal{S}$ tends to preserve, in the transformed space, the average rank of the DTW-based distances, whereas, for retrieval, the ideal is to emphasize, in the embedded space, a useful distinction of the real k-first elements for the original space, and consequently, it does not matter if the furthest elements have their ranking positions preserved.

Furthermore, the LDPS approach does not have any criteria to evaluate the redundancy or the quality of a specific shapelet (or groups of them); consequently, the learned model may lead to problems such as redundancy, or over-fitting, among others. Whereas, on the one hand, transformation based on too many shapelets can be accepted for offline tasks, for online tasks such as retrieval, this number needs to be small, reducing the transformation cost at query time. Therefore, a study of a shapelets' optima-subset is obviously of paramount importance.

We then propose a metric based on the ranking of the DTW measure to evaluate the transformation quality. This metric is applied to (groups of) shapelets and allow us to select the optimal subset. A forward wrapper method for feature selection made upon the greedy strategy was presented, and three stop criteria were defined.

Our experiments in the 85 UCR univariate datasets comparing the proposed DPSR against two solid competitors, the PAA and LB_Keogh demonstrated the substantial gains in the retrieval quality, preserving a good response time. Our results demonstrated that the retrieval based on the most aggressive strategy ($\mathrm{DPSR_t}$), won over PAA 58 times (on 85 datasets), and, among these wins, in 48 cases the dictionary needed for $\mathrm{DPSR_t}$ contains fewer items than the number of pieces for PAA. This result highlights the quality of representation.

One drawback of the proposed method is the computational cost of the offline method used to evaluate the shapelets. However, our second contribution is a filter based on clique-removal, presented to reduce the time need on the offline step. This filter uses a graph representation of the transformed space, and we can look at the relations between the features and use the cliques as a surrogate for clusters. All elements into the clique are high-correlated each other, therefore, probably redundant. Consequently, we can remove the redundant and irrelevant ones.

A third contribution consists of the generalization of the LDPS method to handle multivariate time series. In this way, three transformation approach was proposed, trying to emulate the two most common approaches for multivariate DTW, the independent and the dependent warping methods. The first transformation is a straightforward use of the univariate LDPS on each m-channel of the multivariate time series. This approach produces the highest number of shapelets, as M independently dictionaries are learned. One advantage of this transformation is easy to take advantages of paralleled computation. We are planning to reduce the inter-channel redundancy by modifying the clique-removal to handle the inter-channel data.

We then propose an enhancement of an already existing algorithm LDPS to learn multivariate shapelets. Two multivariate transformations were proposed, the dependent and the slack. On the first, the window slides synchronous over all channels, while in the second it slides asynchronous. The motivation for the slack method is that we believe whilst the shapelet learned is dependent on the features being in phase, the places where they occur in other series could be independent of one another. One planed modification is to limit the offset of the phase between the channels.

The results on the UEA multivariate dataset showed that multivariate shapelet transformation could bet even in the retrieval time the (M-)PAA, well-known as one of the fastest methods for transformation in retrieval tasks. This is due to both multivariate transformation return a single vector in $\mathbb{R}^{|\mathcal{S}|}$, while in (M-)PAA or (M-)LB_Keogh the transformed data (or the envelope) sill multivariate.

However, one of the significant issues noticed during the large scale experimental work was that the DTW-preserving shapelet transform together with the wrapper method was intractable on the most massive multivariate problems even with the use of specialist HPC equipment. With the final goal of this thesis to design efficient univariate and multivariate shapelet-based retrieval, it became apparent that more drastic time and space reductions would be required as we were still struggling on univariate problems. We believe this could be overcome by modifying the learning procedure, eliminating the need to select the best shapelets later.

The final contribution in this thesis is one of the first large scale experiments on univariate and multivariate Time Series Retrieval (TSR) task using shapelet transform.

5.2 Future Work and Extensions

The massification of sensors and collected data brings new challenges for the efficient recovery of time series data, and our work can be seen as a step towards bringing closer shapelets and time series retrieval. Shapelets are recognized for their excellent results in classification and clustering tasks. However, nothing more natural than extending them to the retrieval task.

In this thesis, we showed shapelets could help represent data for retrieval, especially, after a proper subset is chosen. A next straightforward step is inputting this data in an indexing system, exploring the triangular inequality, so, avoiding the exhaustive Euclidean distance calculations, further improving performance. Such an approach can be advantageously used for *anytime indexing* of time series.

Our framework based on DTW-preserving Shapelets algorithm shows very promising retrieval results, in both univariate and multivariate tasks. However, an open issue is to improve the learning of shapelets as the process described by Lods *et al.* in [LMTA17]. s previously mentioned, in the original approach, it works by minimizing the loss between the Euclidean distance in the transformed space and the DTW distance in the raw data, which leads to missing a transformation helpful to preserve the top-k DTW-ranking prioritizing the overall DTW-ranking. The choice of subset besides providing a speed-up in the transformation and consequently in the query time has the side effect of allowing the transformation to prioritize shapelets that allow an improvisation in the top-k retrieved ranking. Nevertheless, the cost of the shapelets evaluation algorithm makes it difficult to use in large databases.

In this way, two possible modifications can be applied to the LDPS algorithm. One is by adding a mechanism for choosing top-k shapelets during the learning phase, and another is modifying the loss function to be minimized. In this case, choosing not to maintain the average DTW distances, but preserving the top-k ranked list.

Besides that, another open way to explore is addressing the similarity between LDPS and Convolutional Neural Networks (CNN). Where we can see a correspondence between the Shapelet Transform and the output of a CNN's single-layer. Therefore, an equivalency between the convolution filters and the shapelets. This equivalency paves the way for using optimized libraries, allowing for performance gains.

Finally, we show that DTW-preserving shapelets can handle both univariate and multivariate data in the retrieval task, showing promising retrieval results. However, further investigation is required, especially in order to better understand how to determine shapelets' length and number to deal with the retrieval task, also, how to learn those shapelets better, avoiding overfitting, and redundancy in the generated representation. Moreover, a fast implementation of both LDPS and DPSR exploiting state-of-art libraries must be implemented in the future to promote these methods.

Contents

List of Figures

List of Tables

List of Algorithms

List of Abbreviations

APCA	Adaptive Piecewise Constant Approximation
AR	Auto-regressive
AUC	Area Under Curve
CDF	Cumulative Distribution Function
CNN	Convolutional Neural Networks
DFT	Discrete Fourier Transformation
DM	Data mining
DPSR	DTW-Preserving Shapelet Retrieval
DTW	Dynamic Time Warping
DWT	Discrete Wavelet Transformation
ECG	Electrocardiography
ED	Euclidean distance
FS	Fisher Score
GPU	Graphics processing unit
IG	Information gain
IoT	Internet of Things
LB_Keogh	LB_Keogh lower bound
LCSS	Longest Common SubSequence
LDPS	Learning DTW-preserving shapelets
LS	Laplacian Score
LSTM	Long Short-Term Memory Networks
LTS	Learning time-series shapelets
ML	Machine learning
MTS	multivariate time series
PAA	Piecewise Aggregate Approximation
PCA	Principal Component Analysis
PDF	Probability Density Function
PLA	Piecewise Linear Approximation
SAX	Symbolic Aggregate approXimation
SSA	Singular Spectral Analysis
ST	Shapelet Transform
TS	time series
TSC	Time Series Classification
TSDM	Time Series Data Mining
TSR	Time Series Retrieval

Appendices

Appendix A

Specifications per Datasets

Table A.1 – The 85 datasets in the UCR time series archive.

Dataset	Length	Train	Test
ElectricDevices	97	8926	7711
StarLightCurves	1025	1000	8236
wafer	153	1000	6164
FordA	501	1320	3601
Two_Patterns	129	1000	4000
NonInvasiveFatalECG_Thorax1	751	1800	1965
NonInvasiveFatalECG_Thorax2	751	1800	1965
uWaveGestureLibrary_X	316	896	3582
uWaveGestureLibrary_Y	316	896	3582
uWaveGestureLibrary_Z	316	896	3582
UWaveGestureLibraryAll	946	896	3582
FordB	501	810	3636
ECG5000	141	500	4500
ChlorineConcentration	167	467	3840
PhalangesOutlinesCorrect	81	1800	858
FaceAll	132	560	1690
yoga	427	300	3000
InsectWingbeatSound	257	220	1980
FacesUCR	132	200	2050
Phoneme	1025	214	1896
HandOutlines	2710	370	1000
ShapesAll	513	600	600
SwedishLeaf	129	500	625
MedicalImages	100	381	760
Strawberry	236	370	613
50words	271	450	455
ProximalPhalanxOutlineCorrect	81	600	291
MiddlePhalanxOutlineCorrect	81	291	600
WordsSynonyms	271	267	638
DistalPhalanxOutlineCorrect	81	276	600
Adiac	177	390	391
Cricket_X	301	390	390
Cricket_Y	301	390	390
Cricket_Z	301	390	390
LargeKitchenAppliances	721	375	375
RefrigerationDevices	721	375	375
ScreenType	721	375	375
SmallKitchenAppliances	721	375	375
MALLAT	1025	55	2345

Table A.1 continued from previous page

Dataset	Length	Train	Test
synthetic_control	61	300	300
ProximalPhalanxOutlineAgeGroup	81	400	205
ProximalPhalanxTW	81	205	400
ItalyPowerDemand	25	67	1029
Computers	721	250	250
MiddlePhalanxOutlineAgeGroup	81	154	400
MiddlePhalanxTW	81	154	399
DistalPhalanxOutlineAgeGroup	81	139	400
DistalPhalanxTW	81	139	400
CinC_ECG_torso	1640	40	1380
InlineSkate	1883	100	550
OSULeaf	428	200	242
Haptics	1093	155	308
Earthquakes	513	139	322
FISH	464	175	175
CBF	129	30	900
TwoLeadECG	83	23	1139
SonyAIBORobotSurfaceII	66	27	953
MoteStrain	85	20	1252
Symbols	399	25	995
ECGFiveDays	137	23	861
Worms	901	77	181
WormsTwoClass	901	77	181
SonyAIBORobotSurface	71	20	601
Ham	432	109	105
Plane	145	105	105
ECG200	97	100	100
Trace	276	100	100
ToeSegmentation1	278	40	228
Gun_Point	151	50	150
ArrowHead	252	36	175
Lighting7	320	70	73
DiatomSizeReduction	346	16	306
ToeSegmentation2	344	36	130
Herring	513	64	64
Lighting2	638	60	61
Car	578	60	60
Meat	449	60	60
ShapeletSim	501	20	180
Wine	235	57	54
FaceFour	351	24	88
Beef	471	30	30
OliveOil	571	30	30
Coffee	287	28	28
BeetleFly	513	20	20
BirdChicken	513	20	20

Table A.2 – The 30 datasets from UEA multivariate time series archive.

Dataset	Train	Test	Dims	Length	Total	Skip
ArticularyWordRecognition	275	300	9	144	1296	N
AtrialFibrillation	15	15	2	640	1280	N
BasicMotions	40	40	6	100	600	N
CharacterTrajectories	1422	1436	3	182	546	N
Cricket	108	72	6	1197	7182	Y
DuckDuckGeese	60	40	1345	270	363150	Y
EigenWorms	128	131	6	17984	107904	Y
Epilepsy	137	138	3	206	618	N
EthanolConcentration	261	263	3	1751	5253	N
ERing	30	30	4	65	260	N
FaceDetection	5890	3524	144	62	8928	Y
FingerMovements	316	100	28	50	1400	N
HandMovementDirection	320	147	10	400	4000	N
Handwriting	150	850	3	152	456	N
Heartbeat	204	205	61	405	24705	N
JapaneseVowels	270	370	12	29	348	N
Libras	180	180	2	45	90	N
LSST	2459	2466	6	36	216	N
InsectWingbeat	30000	20000	200	78	15600	Y
MotorImagery	278	100	64	3000	192000	Y
NATOPS	180	180	24	51	1224	N
PenDigits	7494	3498	2	8	16	Y
PEMS-SF	267	173	963	144	138672	Y
Phoneme	3315	3353	11	217	2387	Y
RacketSports	151	152	6	30	180	N
SelfRegulationSCP1	268	293	6	896	5376	N
SelfRegulationSCP2	200	180	7	1152	8064	N
SpokenArabicDigits	6599	2199	13	93	1209	Y
StandWalkJump	12	15	4	2500	10000	N
UWaveGestureLibrary	120	320	3	315	945	N

Appendix B

Results on Univariate Dataset

Table B.1 – AUC for the PAA and LB_Keogh methods.

	PAA		LB_Keogh	
DATASET	DIMS	AUC %	WINDOW	AUC %
BeetleFly	16	0.9100	0.1	0.8775
BirdChicken	173	0.8325	0.05	0.9275
Coffee	111	0.9643	0.01	0.9745
OliveOil	276	0.9911	0.01	0.9056
Beef	170	0.9844	0.01	0.9900
Wine	116	0.9971	0.01	0.9620
Car	193	0.8383	0.1	0.8933
Meat	101	0.9944	0.01	0.9669
Lighting2	127	0.5760	0.05	0.6730
Herring	25	0.8613	0.05	0.9600
Lighting7	108	0.7755	0.05	0.7955
FaceFour	117	0.8120	0.05	0.8409
Trace	5	0.9128	0.1	0.9585
ECG200	16	0.9024	0.1	0.9311
Ham	214	0.9471	0.01	0.9299
Plane	7	0.9741	0.01	0.9714
ToeSegmentation2	7	0.8404	0.05	0.8207
Gun_Point	7	0.9213	0.1	0.9772
FISH	154	0.8299	0.05	0.9036
ArrowHead	89	0.9135	0.01	0.9254
ShapeletSim	15	0.6167	0.05	0.5997
Worms	3	0.7611	0.2	0.8068
WormsTwoClass	3	0.7611	0.2	0.8068
ProximalPhalanxOutlineAgeGroup	5	0.9388	0.01	0.9419
ToeSegmentation1	7	0.7762	0.1	0.8575
OSULeaf	13	0.8967	0.1	0.9415
Computers	3	0.6575	0.2	0.7118
ProximalPhalanxOutlineCorrect	5	0.9423	0.01	0.9508
synthetic_control	15	0.9571	0.05	0.9335
DiatomSizeReduction	142	0.9269	0.01	0.9575
Haptics	4	0.7864	0.1	0.8596
Earthquakes	7	0.5433	0.1	0.8893
LargeKitchenAppliances	3	0.6119	0.2	0.5990
RefrigerationDevices	7	0.6292	0.2	0.8416
ScreenType	3	0.7215	0.2	0.7952
SmallKitchenAppliances	2	0.5945	0.2	0.6502
Cricket_X	8	0.9197	0.1	0.9591
Cricket_Y	7	0.9178	0.1	0.9586

Table B.1 continued from previous page

	PAA		LB_Keogh	
DATASET	DIMS	AUC %	WINDOW	AUC %
Cricket_Z	5	0.9305	0.1	0.9553
Adiac	70	0.9034	0.01	0.9650
MiddlePhalanxTW	8	0.9398	0.01	0.9494
DistalPhalanxOutlineAgeGroup	8	0.9185	0.01	0.9301
DistalPhalanxTW	5	0.9024	0.01	0.9283
MiddlePhalanxOutlineAgeGroup	5	0.9396	0.01	0.9648
ProximalPhalanxTW	5	0.9269	0.01	0.9297
50words	6	0.8500	0.2	0.8984
InlineSkate	6	0.8881	0.1	0.9709
ShapesAll	10	0.9515	0.05	0.9509
DistalPhalanxOutlineCorrect	8	0.9292	0.01	0.9359
MiddlePhalanxOutlineCorrect	5	0.9456	0.01	0.9655
SonyAIBORobotSurface	7	0.8817	0.05	0.8090
Strawberry	47	0.9968	0.01	0.9948
SwedishLeaf	16	0.9125	0.05	0.9302
WordsSynonyms	6	0.8380	0.2	0.8823
MedicalImages	48	0.9521	0.01	0.9482
PhalangesOutlinesCorrect	5	0.9542	0.01	0.9765
ECGFiveDays	4	0.8917	0.1	0.9469
CBF	6	0.9008	0.2	0.8980
SonyAIBORobotSurfaceII	30	0.8418	0.05	0.9017
Symbols	7	0.9124	0.05	0.9263
HandOutlines	11	0.9359	0.01	0.8965
ItalyPowerDemand	11	0.9679	0.01	0.9609
TwoLeadECG	3	0.8550	0.05	0.9246
MoteStrain	7	0.8404	0.01	0.7909
CinC_ECG_torso	4	0.8785	0.2	0.9204
FaceAll	16	0.8301	0.05	0.9450
Phoneme	2	0.5599	0.1	0.8770
NonInvasiveFatalECG_Thorax1	44	0.9781	0.05	0.9868
NonInvasiveFatalECG_Thorax2	250	0.9846	0.01	0.9838
InsectWingbeatSound	4	0.8049	0.2	0.8525
FacesUCR	9	0.8570	0.05	0.9653
MALLAT	26	0.9593	0.01	0.9615
yoga	71	0.9700	0.1	0.9841
UWaveGestureLibraryAll	12	0.9493	0.05	0.9596
uWaveGestureLibrary_X	7	0.9361	0.1	0.9530
uWaveGestureLibrary_Y	5	0.9406	0.1	0.9576
uWaveGestureLibrary_Z	6	0.9306	0.1	0.9579
FordA	20	0.6509	0.05	0.9141
FordB	22	0.5875	0.1	0.8692
ChlorineConcentration	82	0.9987	0.01	0.9938
Two_Patterns	7	0.7703	0.1	0.9401
ECG5000	35	0.9807	0.01	0.9810
wafer	4	0.9701	0.1	0.9923
ElectricDevices	3	0.7426	0.2	0.8470
StarLightCurves	6	0.9199	0.2	0.9606
AVERAGE	38.833	0.866	0.074	0.9080

Table B.2 – Dimensionality per stopping criteria per dataset.

Dataset	DPRS_t	DPRS_l	DPRS_g	DPRS_f	F+DPRS_t	F+DPRS_l	F+DPRS_g	F+DPRS_f
BeetleFly	4	6	6	174	4	5	8	160
BirdChicken	4	7	16	174	5	7	15	100
Coffee	5	11	17	157	4	13	50	88
OliveOil	5	9	12	177	3	6	53	87
Beef	3	6	55	171	2	4	34	43
Wine	2	7	9	150	1	5	25	30
Car	4	20	68	178	4	18	37	96
Meat	3	7	71	171	3	17	29	59
Lighting2	5	14	59	181	5	16	33	129
Herring	4	25	70	174	4	16	56	91
Lighting7	5	25	46	161	6	15	48	120
FaceFour	5	13	52	163	6	11	55	115
Trace	2	11	27	156	1	5	5	11
ECG200	5	15	86	125	3	24	43	76
Ham	7	27	103	170	4	29	48	98
Plane	3	17	39	137	2	14	25	56
ToeSegmentation2	7	17	56	162	5	13	71	127
Gun_Point	2	10	23	138	1	6	6	26
FISH	6	26	41	171	6	29	74	119
ArrowHead	3	8	35	153	2	11	23	64
ShapeletSim	3	11	174	174	8	9	55	170
Worms	10	30	66	192	9	26	54	187
WormsTwoClass	8	37	54	192	9	39	47	182
ProximalPhalanxOutlineAgeGroup	4	29	47	119	3	18	37	75
ToeSegmentation1	7	23	61	156	9	19	47	144
OSULeaf	9	48	70	169	9	39	64	164
Computers	3	31	52	185	3	9	20	103
ProximalPhalanxOutlineCorrect	3	32	38	119	2	25	25	69
synthetic_control	4	33	36	110	3	22	44	88
DiatomSizeReduction	2	2	2	162	1	7	9	32

Table B.2 continued from previous page

Dataset	DPRS_t	DPRS_l	DPRS_g	DPRS_f	F+DPRS_t	F+DPRS_l	F+DPRS_g	F+DPRS_f
Haptics	5	18	32	197	4	23	31	124
Earthquakes	3	11	14	174	1	15	15	112
LargeKitchenAppliances	6	22	30	185	1	15	24	101
RefrigerationDevices	5	23	40	185	4	27	41	108
ScreenType	2	44	46	185	1	32	35	110
SmallKitchenAppliances	1	1	20	185	3	14	14	84
Cricket_X	5	59	93	159	5	65	97	141
Cricket_Y	5	45	50	159	5	71	83	134
Cricket_Z	6	68	102	159	5	70	96	139
Adiac	3	27	43	143	2	21	25	45
MiddlePhalanxTW	4	25	59	119	3	23	25	73
DistalPhalanxOutlineAgeGroup	3	26	84	119	2	26	33	48
DistalPhalanxTW	3	21	38	119	3	18	25	65
MiddlePhalanxOutlineAgeGroup	4	25	58	119	4	30	32	96
ProximalPhalanxTW	5	26	40	119	4	22	28	76
50words	5	47	78	156	4	46	81	135
InlineSkate	6	23	52	213	4	13	30	83
ShapesAll	6	41	46	174	5	46	51	148
DistalPhalanxOutlineCorrect	4	27	53	119	3	23	36	63
MiddlePhalanxOutlineCorrect	4	31	50	119	4	27	32	81
SonyAIBORobotSurface	5	9	31	114	6	15	32	96
Strawberry	3	26	84	151	2	16	36	44
SwedishLeaf	4	34	62	132	4	34	58	105
WordsSynonyms	4	57	78	156	4	52	65	126
MedicalImages	4	41	55	126	3	36	57	82
PhalangesOutlinesCorrect	2	37	53	119	2	27	27	57
ECGFiveDays	5	8	25	135	4	9	31	59
CBF	3	8	47	132	4	6	34	106
SonyAIBORobotSurfaceII	4	10	43	113	5	8	38	94
Symbols	3	5	7	167	2	10	21	39
HandOutlines	6	22	125	225	3	30	46	56

Table B.2 continued from previous page

Dataset	DPRS_t	DPRS_l	DPRS_g	DPRS_f	F+DPRS_t	F+DPRS_l	F+DPRS_g	F+DPRS_f
ItalyPowerDemand	3	17	35	84	2	13	28	48
TwoLeadECG	3	4	53	120	3	6	13	52
MoteStrain	4	8	38	120	2	7	17	66
CinC_ECG_torso	5	9	51	210	4	12	35	105
FaceAll	5	49	90	134	6	63	68	131
Phoneme	6	14	33	195	4	21	26	113
NonInvasiveFatalECG_Thorax1	4	68	86	186	2	49	49	66
NonInvasiveFatalECG_Thorax2	3	42	72	186	2	34	39	57
InsectWingbeatSound	6	42	46	153	5	44	54	119
FacesUCR	7	37	64	134	6	27	52	123
MALLAT	4	15	31	195	3	14	20	70
yoga	4	29	67	169	3	28	55	117
UWaveGestureLibraryAll	7	86	109	192	7	69	96	185
uWaveGestureLibrary_X	5	71	96	159	5	70	75	118
uWaveGestureLibrary_Y	5	60	69	159	5	44	57	113
uWaveGestureLibrary_Z	5	57	57	159	5	50	64	125
FordA	8	33	58	174	6	33	47	166
FordB	8	38	53	174	5	42	42	163
ChlorineConcentration	2	28	50	141	2	20	38	77
Two_Patterns	6	63	65	132	5	49	62	99
ECG5000	4	55	96	135	3	38	72	85
wafer	2	25	25	138	2	21	21	56
ElectricDevices	3	23	61	125	3	23	58	110
StarLightCurves	4	28	55	195	2	19	22	45
AVERAGE	4.4	27.4	54.0	156.1	3.8	25.2	41.5	95.9

TABLE B.3 – AUC per stooping criteria per dataset.

Dataset	DPRS_t	DPRS_l	DPRS_g	DPRS_f	F+DPRS_t	F+DPRS_l	F+DPRS_g	F+DPRS_f
BeetleFly	0.635	0.593	0.593	0.823	0.623	0.648	0.648	0.828
BirdChicken	0.790	0.810	0.845	0.883	0.805	0.815	0.830	0.890
Coffee	0.911	0.944	0.944	0.967	0.894	0.929	0.945	0.967
OliveOil	0.944	0.943	0.954	0.991	0.931	0.954	0.988	0.993
Beef	0.943	0.961	0.971	0.984	0.972	0.981	0.983	0.986
Wine	0.959	0.974	0.980	0.988	0.889	0.955	0.988	0.987
Car	0.941	0.956	0.964	0.963	0.935	0.963	0.970	0.956
Meat	0.963	0.972	0.980	0.980	0.947	0.981	0.982	0.966
Lighting2	0.874	0.898	0.932	0.930	0.917	0.917	0.921	0.925
Herring	0.934	0.970	0.977	0.975	0.930	0.959	0.974	0.979
Lighting7	0.923	0.947	0.945	0.957	0.934	0.944	0.948	0.953
FaceFour	0.760	0.760	0.833	0.818	0.781	0.779	0.819	0.814
Trace	0.980	0.985	0.985	0.958	0.970	0.982	0.982	0.964
ECG200	0.920	0.942	0.959	0.961	0.895	0.950	0.951	0.958
Ham	0.903	0.946	0.965	0.963	0.891	0.956	0.962	0.966
Plane	0.966	0.980	0.983	0.983	0.964	0.983	0.983	0.982
ToeSegmentation2	0.816	0.833	0.865	0.863	0.763	0.795	0.858	0.857
Gun_Point	0.961	0.981	0.985	0.983	0.953	0.977	0.977	0.970
FISH	0.927	0.964	0.967	0.973	0.925	0.962	0.972	0.973
ArrowHead	0.891	0.904	0.935	0.956	0.875	0.899	0.907	0.943
ShapeletSim	0.575	0.576	0.586	0.586	0.633	0.638	0.608	0.584
Worms	0.840	0.867	0.875	0.879	0.836	0.857	0.875	0.879
WormsTwoClass	0.840	0.883	0.888	0.885	0.828	0.882	0.881	0.884
ProximalPhalanxOutlineAgeGroup	0.943	0.986	0.987	0.975	0.932	0.977	0.983	0.969
ToeSegmentation1	0.789	0.825	0.832	0.843	0.803	0.815	0.834	0.844
OSULeaf	0.862	0.918	0.921	0.933	0.872	0.910	0.918	0.933
Computers	0.911	0.937	0.934	0.910	0.911	0.913	0.934	0.898
ProximalPhalanxOutlineCorrect	0.950	0.986	0.987	0.975	0.939	0.983	0.983	0.969
synthetic_control	0.940	0.952	0.952	0.956	0.911	0.945	0.952	0.955
DiatomSizeReduction	0.910	0.910	0.910	0.964	0.880	0.957	0.957	0.960
Haptics	0.932	0.946	0.949	0.889	0.925	0.948	0.947	0.896
Earthquakes	0.825	0.831	0.830	0.663	0.823	0.831	0.831	0.656
LargeKitchenAppliances	0.870	0.880	0.876	0.841	0.843	0.882	0.876	0.828

Table B.3 continued from previous page

Dataset	DPRS_t	DPRS_l	DPRS_g	DPRS_f	F+DPRS_t	F+DPRS_l	F+DPRS_g	F+DPRS_f
RefrigerationDevices	0.880	0.910	0.906	0.900	0.869	0.899	0.902	0.899
ScreenType	0.947	0.955	0.955	0.934	0.943	0.951	0.950	0.923
SmallKitchenAppliances	0.883	0.883	0.877	0.837	0.842	0.868	0.868	0.835
Cricket_X	0.930	0.968	0.970	0.974	0.940	0.970	0.975	0.976
Cricket_Y	0.922	0.966	0.966	0.975	0.935	0.972	0.972	0.975
Cricket_Z	0.935	0.971	0.974	0.972	0.934	0.971	0.973	0.974
Adiac	0.969	0.983	0.983	0.982	0.955	0.981	0.981	0.982
MiddlePhalanxTW	0.926	0.965	0.972	0.966	0.922	0.962	0.961	0.962
DistalPhalanxOutlineAgeGroup	0.930	0.962	0.955	0.952	0.897	0.936	0.937	0.932
DistalPhalanxTW	0.925	0.955	0.960	0.956	0.917	0.949	0.954	0.947
MiddlePhalanxOutlineAgeGroup	0.952	0.971	0.973	0.973	0.944	0.968	0.968	0.971
ProximalPhalanxTW	0.942	0.976	0.978	0.967	0.932	0.974	0.973	0.956
50words	0.940	0.965	0.972	0.968	0.914	0.960	0.968	0.964
InlineSkate	0.963	0.970	0.976	0.971	0.935	0.956	0.964	0.972
ShapesAll	0.961	0.975	0.975	0.974	0.948	0.977	0.978	0.973
DistalPhalanxOutlineCorrect	0.946	0.971	0.975	0.972	0.938	0.970	0.970	0.965
MiddlePhalanxOutlineCorrect	0.966	0.982	0.982	0.980	0.966	0.980	0.980	0.979
SonyAIBORobotSurface	0.810	0.811	0.865	0.897	0.765	0.853	0.855	0.894
Strawberry	0.988	0.995	0.996	0.997	0.977	0.996	0.997	0.997
SwedishLeaf	0.950	0.975	0.980	0.975	0.954	0.977	0.979	0.975
WordsSynonyms	0.927	0.967	0.966	0.957	0.919	0.964	0.965	0.956
MedicalImages	0.975	0.988	0.989	0.987	0.965	0.989	0.987	0.986
PhalangesOutlinesCorrect	0.963	0.991	0.991	0.988	0.954	0.989	0.989	0.985
ECGFiveDays	0.937	0.939	0.939	0.955	0.920	0.926	0.943	0.944
CBF	0.914	0.936	0.952	0.955	0.915	0.925	0.949	0.952
SonyAIBORobotSurfaceII	0.785	0.808	0.838	0.866	0.771	0.801	0.844	0.859
Symbols	0.882	0.883	0.887	0.944	0.916	0.921	0.927	0.935
HandOutlines	0.946	0.976	0.983	0.979	0.937	0.975	0.976	0.975
ItalyPowerDemand	0.936	0.965	0.968	0.976	0.906	0.967	0.974	0.976
TwoLeadECG	0.812	0.810	0.892	0.910	0.858	0.880	0.882	0.907
MoteStrain	0.888	0.902	0.910	0.919	0.822	0.898	0.897	0.914
CinC_ECG_torso	0.935	0.950	0.968	0.968	0.924	0.949	0.964	0.968
FaceAll	0.871	0.914	0.915	0.900	0.875	0.916	0.914	0.899
Phoneme	0.824	0.853	0.856	0.847	0.832	0.858	0.859	0.843
NonInvasiveFatalECG_Thorax1	0.981	0.994	0.994	0.993	0.971	0.994	0.994	0.991

Table B.3 continued from previous page

Dataset	DPRS_t	DPRS_l	DPRS_g	DPRS_f	F+DPRS_t	F+DPRS_l	F+DPRS_g	F+DPRS_f
NonInvasiveFatalECG_Thorax2	0.980	0.995	0.996	0.995	0.976	0.995	0.995	0.994
InsectWingbeatSound	0.928	0.957	0.957	0.937	0.913	0.956	0.957	0.930
FacesUCR	0.921	0.952	0.959	0.964	0.919	0.944	0.950	0.963
MALLAT	0.947	0.955	0.958	0.970	0.944	0.962	0.962	0.967
yoga	0.961	0.984	0.987	0.985	0.954	0.984	0.986	0.985
UWaveGestureLibraryAll	0.915	0.957	0.958	0.957	0.917	0.954	0.958	0.957
uWaveGestureLibrary_X	0.927	0.961	0.963	0.962	0.927	0.963	0.963	0.961
uWaveGestureLibrary_Y	0.939	0.969	0.970	0.963	0.943	0.969	0.969	0.966
uWaveGestureLibrary_Z	0.934	0.966	0.966	0.964	0.932	0.964	0.966	0.963
FordA	0.746	0.773	0.776	0.715	0.734	0.774	0.778	0.715
FordB	0.723	0.737	0.739	0.666	0.707	0.740	0.740	0.662
ChlorineConcentration	0.987	0.998	0.998	0.998	0.987	0.998	0.998	0.998
Two_Patterns	0.935	0.976	0.976	0.975	0.929	0.975	0.976	0.973
ECG5000	0.971	0.987	0.987	0.986	0.963	0.985	0.986	0.985
wafer	0.981	0.994	0.994	0.988	0.984	0.993	0.993	0.987
ElectricDevices	0.929	0.969	0.970	0.941	0.929	0.969	0.970	0.941
StarLightCurves	0.960	0.973	0.974	0.961	0.938	0.964	0.964	0.955
AVERAGE	0.907	0.928	0.935	0.934	0.899	0.929	0.935	0.931

Appendix C

Results on Multivariate Dataset

TABLE C.1 – AUC comparison between (M-)PAA, (M-)LB_Keogh, and M-DPSR$_{\text{best}}$.

	AUC					
Dataset	PAA	LB_keogh	M-DPRS$_{\text{best,t}}$	M-DPRS$_{\text{best,l}}$	M-DPRS$_{\text{best,g}}$	M-DPRS$_{\text{best,f}}$
ArticularyWordRecognition	0.9958	0.9941	0.9388	0.9842	0.9845	0.9871
AtrialFibrilation	0.6178	0.8889	0.8844	0.8489	0.7911	0.9111
BasicMotions	0.7688	0.7869	0.8500	0.8831	0.8862	0.8869
CharacterTrajectories	0.9616	0.9881	0.9163	0.9971	0.9971	0.9955
Epilepsy	0.6181	0.7482	0.8166	0.8784	0.8784	0.8332
EthanolConcentration	0.9360	0.7144	0.9276	0.9732	0.9728	0.9629
ERing	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000
FingerMovements	0.9959	0.9125	0.7951	0.8814	0.8814	0.8289
HandMovementDirection	0.7413	0.8259	0.6329	0.7123	0.7123	0.6800
Handwriting	0.7060	0.8331	0.8164	0.9485	0.9502	0.8686
Heartbeat	0.5092	0.5940	0.5705	0.6077	0.6152	0.6123
JapaneseVowels	0.9777	0.9887	0.9701	0.9905	0.9909	0.9711
Libras	0.9909	0.9574	0.9279	0.9858	0.9871	0.9822
LSST	0.8326	0.8987	0.9370	0.9820	0.9820	0.9804
NATOPS	0.8806	0.9685	0.9343	0.9686	0.9722	0.9552
RacketSports	0.7752	0.9028	0.8709	0.9480	0.9503	0.9407
SelfRegulationSCP1	0.9451	0.9157	0.7318	0.7702	0.7702	0.7614
SelfRegulationSCP2	0.9482	0.8437	0.7430	0.7919	0.7919	0.7583
StandWalkJump	0.4333	0.9278	0.6889	0.8056	0.7833	0.8167
UWaveGestureLibrary	0.9798	0.9690	0.9455	0.9786	0.9815	0.9837
AVERAGE	0.8307	0.8829	0.8449	0.8968	0.8939	0.8858

TABLE C.2 – The measured AUC per dataset considering the M-DPSR transformation and the stopping criterion.

	Tangent			Local			Global			Full		
Dataset	MDPSR$_s$	MDPSR$_i$	MDPSR$_d$	MDPSR$_s$	MDPSR$_i$	MDPSR$_d$	MDPSR$_s$	MDPSR$_i$	MDPSR$_d$	MDPSR$_s$	MDPSR$_i$	MDPSR$_d$
ArticularyWordRecognition	0.9388	0.5986	0.8959	0.9842	0.6435	0.9620	0.9845	0.6604	0.9620	0.9871	0.6056	0.9616
AtrialFibrilation	0.4667	0.8844	0.5556	0.4667	0.8489	0.6400	0.5822	0.7911	0.7422	0.8178	0.7378	0.9111
BasicMotions	0.8056	0.4931	0.8500	0.8712	0.5637	0.8831	0.8656	0.5413	0.8862	0.8775	0.5125	0.8869
CharacterTrajectories	0.9163	0.6946	0.9042	0.9879	0.8939	0.9971	0.9879	0.8987	0.9971	0.9783	0.8831	0.9955
Epilepsy	0.8166	0.6959	0.8088	0.8438	0.7469	0.8784	0.8507	0.7539	0.8784	0.8332	0.8330	0.8040
EthanolConcentration	0.8843	0.8539	0.9276	0.9286	0.9011	0.9732	0.9314	0.9149	0.9728	0.9169	0.9172	0.9629
ERing	1.0000	0.5822	1.0000	1.0000	0.6278	1.0000	1.0000	0.6711	1.0000	1.0000	0.7567	1.0000
FingerMovements	0.7951	0.4973	0.7534	0.8403	0.5512	0.8814	0.8403	0.5391	0.8814	0.8289	0.4228	0.7078
HandMovementDirection	0.5694	0.5499	0.6329	0.6443	0.5204	0.7123	0.6487	0.5489	0.7123	0.5064	0.5286	0.6800
Handwriting	0.7794	0.7394	0.8164	0.8679	0.8000	0.9485	0.8735	0.8243	0.9502	0.8686	0.8084	0.7835
Heartbeat	0.5423	0.5705	0.5699	0.5691	0.5363	0.6077	0.5691	0.5696	0.6152	0.5486	0.6123	0.5757
JapaneseVowels	0.8393	0.9701	0.8614	0.9306	0.9905	0.9599	0.9306	0.9909	0.9565	0.7817	0.9711	0.9226
Libras	0.8870	0.9279	0.8760	0.9727	0.9806	0.9858	0.9737	0.9807	0.9871	0.9773	0.9822	0.9783
LSST	0.7872	0.9370	0.7698	0.8858	0.9820	0.8808	0.8858	0.9820	0.8808	0.8782	0.9804	0.8811
NATOPS	0.7619	0.9343	0.8539	0.8792	0.9686	0.9485	0.8812	0.9722	0.9522	0.6806	0.9552	0.8856
RacketSports	0.7110	0.8709	0.7757	0.8974	0.9480	0.8652	0.9015	0.9503	0.8652	0.9017	0.9407	0.7703
SelfRegulationSCP1	0.7318	0.5713	0.6412	0.7702	0.7028	0.6812	0.7702	0.7474	0.6812	0.5788	0.7614	0.6791
SelfRegulationSCP2	0.7430	0.6209	0.6631	0.7919	0.6848	0.7310	0.7919	0.6718	0.7316	0.7583	0.6649	0.6836
StandWalkJump	0.4778	0.6278	0.6889	0.6611	0.6167	0.8056	0.7444	0.6944	0.7833	0.7222	0.6056	0.8167
UWaveGestureLibrary	0.8776	0.9455	0.8873	0.9324	0.9786	0.9712	0.9377	0.9815	0.9793	0.9316	0.9837	0.9832
AVERAGE	0.7666	0.7283	0.7866	0.8363	0.7743	0.8656	0.8475	0.7842	0.8708	0.8187	0.7732	0.8435

Bibliography

Bibliography

[02a] *Proceedings of the 2002 IEEE International Conference on Data Mining (ICDM 2002), 9-12 December 2002, Maebashi City, Japan*, IEEE Computer Society, 2002, ISBN: 0-7695-1754-4. [Online]. Available: http://ieeexplore.ieee.org/xpl/mostRecentIssue.jsp?punumber=8435.

[02b] *VLDB 2002, Proceedings of 28th International Conference on Very Large Data Bases, August 20-23, 2002, Hong Kong, China*, Morgan Kaufmann, 2002.

[04] *2004 IEEE International Conference on Acoustics, Speech, and Signal Processing, ICASSP 2004, Montreal, Quebec, Canada, May 17-21, 2004*, IEEE, 2004, ISBN: 0-7803-8484-9. [Online]. Available: http://ieeexplore.ieee.org/xpl/mostRecentIssue.jsp?punumber=9248.

[05a] *Advances in Neural Information Processing Systems 18 [Neural Information Processing Systems, NIPS 2005, December 5-8, 2005, Vancouver, British Columbia, Canada]*, 2005. [Online]. Available: http://papers.nips.cc/book/advances-in-neural-information-processing-systems-18-2005.

[05b] *Proceedings of the ACM/IEEE SC2005 Conference on High Performance Networking and Computing, November 12-18, 2005, Seattle, WA, USA, CD-Rom*, IEEE Computer Society, 2005, ISBN: 1-59593-061-2. [Online]. Available: http://ieeexplore.ieee.org/xpl/mostRecentIssue.jsp?punumber=10435.

[11] *Proceedings of the Eleventh SIAM International Conference on Data Mining, SDM 2011, April 28-30, 2011, Mesa, Arizona, USA*, SIAM / Omnipress, 2011, ISBN: 978-0-89871-992-5. DOI: 10.1137/1.9781611972818. [Online]. Available: https://doi.org/10.1137/1.9781611972818.

[12a] *11th International Conference on Machine Learning and Applications, ICMLA, Boca Raton, FL, USA, December 12-15, 2012. Volume 1*, IEEE, 2012, ISBN: 978-1-4673-4651-1. [Online]. Available: http://ieeexplore.ieee.org/xpl/mostRecentIssue.jsp?punumber=6403616.

[12b] *Proceedings of the Twelfth SIAM International Conference on Data Mining, Anaheim, California, USA, April 26-28, 2012*, SIAM / Omnipress, 2012, ISBN: 978-1-61197-232-0. DOI: 10.1137/1.9781611972825. [Online]. Available: https://doi.org/10.1137/1.9781611972825.

[13] *Proceedings of the 13th SIAM International Conference on Data Mining, May 2-4, 2013. Austin, Texas, USA*, SIAM, 2013, ISBN: 978-1-61197-262-7. DOI: 10.1137/1.9781611972832. [Online]. Available: https://doi.org/10.1137/1.9781611972832.

[17] *2017 IEEE International Conference on Data Science and Advanced Analytics, DSAA 2017, Tokyo, Japan, October 19-21, 2017*, IEEE, 2017, ISBN: 978-1-5090-5004-8. [Online]. Available: http://ieeexplore.ieee.org/xpl/mostRecentIssue.jsp?punumber=8255765.

[AFS93] R. Agrawal, C. Faloutsos, and A. N. Swami, "Efficient similarity search in sequence databases", *in Foundations of Data Organization and Algorithms, 4th International Conference, FODO'93, Chicago, Illinois, USA, October 13-15, 1993, Proceedings*, D. B. Lomet, Ed., ser. Lecture Notes in Computer Science, vol. 730, Springer, 1993, pp. 69–84, ISBN: 3-540-57301-1. DOI: 10.1007/3-540-57301-1_5. [Online]. Available: https://doi.org/10.1007/3-540-57301-1%5C_5.

[AGS11] C. Apté, J. Ghosh, and P. Smyth, Eds., *Proceedings of the 17th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, San Diego, CA, USA, August 21-24, 2011*, ACM, 2011, ISBN: 978-1-4503-0813-7.

[ASO+18] M. Aubert, P. Setiawan, A. A. Oktaviana, A. Brumm, P. H. Sulistyarto, E. W. Saptomo, B. Istiawan, T. A. Ma'rifat, V. N. Wahyuono, F. T. Atmoko, J.-X. Zhao, J. Huntley, P. S. C. Taçon, D. L. Howard, and H. E. A. Brand, "Palaeolithic cave art in Borneo", *Nature*, vol. 564, no. 7735, pp. 254–257, 2018, ISSN: 1476-4687. DOI: 10.1038/s41586-018-0679-9. [Online]. Available: https://doi.org/10.1038/s41586-018-0679-9.

[AST15] S. R. Aghabozorgi, A. S. Shirkhorshidi, and Y. W. Teh, "Time-series clustering - A decade review", *Inf. Syst.*, vol. 53, pp. 16–38, 2015. DOI: 10.1016/j.is.2015.04.007. [Online]. Available: https://doi.org/10.1016/j.is.2015.04.007.

[ATW17] N. M. Adams, A. Tucker, and D. J. Weston, Eds., *Advances in Intelligent Data Analysis XVI - 16th International Symposium, IDA 2017, London, UK, October 26-28, 2017, Proceedings*, vol. 10584, Lecture Notes in Computer Science, Springer, 2017, ISBN: 978-3-319-68764-3. DOI: 10.1007/978-3-319-68765-0. [Online]. Available: https://doi.org/10.1007/978-3-319-68765-0.

[BB04] C. Bahlmann and H. Burkhardt, "The writer independent online handwriting recognition system frog on hand and cluster generative statistical dynamic time warping", *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 26, no. 3, pp. 299–310, 2004. DOI: 10.1109/TPAMI.2004.1262308. [Online]. Available: https://doi.org/10.1109/TPAMI.2004.1262308.

[BB17] A. Bostrom and A. Bagnall, "A Shapelet Transform for Multivariate Time Series Classification", *arXiv e-prints*, arXiv:1712.06428, Dec. 2017. arXiv: 1712.06428 [cs.LG].

[BBL+07] M. W. Berry, M. Browne, A. N. Langville, V. P. Pauca, and R. J. Plemmons, "Algorithms and applications for approximate nonnegative matrix factorization", *Computational Statistics & Data Analysis*, vol. 52, no. 1, pp. 155–173, 2007. DOI: 10.1016/j.csda.2006.11.006. [Online]. Available: https://doi.org/10.1016/j.csda.2006.11.006.

[BBLL16] A. J. Bagnall, A. Bostrom, J. Large, and J. Lines, "The great time series classification bake off: an experimental evaluation of recently proposed algorithms. extended version", *CoRR*, vol. abs/1602.01711, 2016. arXiv: 1602.01711. [Online]. Available: http://arxiv.org/abs/1602.01711.

[BC94] D. J. Berndt and J. Clifford, "Using dynamic time warping to find patterns in time series", *in Knowledge Discovery in Databases: Papers from the 1994 AAAI Workshop, Seattle, Washington, USA, July 1994. Technical Report WS-94-03*, U. M. Fayyad and R. Uthurusamy, Eds., AAAI Press, 1994, pp. 359–370, ISBN: 0-929280-73-3.

[BDB+16] F. Bonchi, J. Domingo-Ferrer, R. A. Baeza-Yates, Z. Zhou, and X. Wu, Eds., *IEEE 16th International Conference on Data Mining, ICDM 2016, December 12-15, 2016, Barcelona, Spain*, IEEE Computer Society, 2016. [Online]. Available: http://ieeexplore.ieee.org/xpl/mostRecentIssue.jsp?punumber=7837023.

[BDHL12] A. J. Bagnall, L. M. Davis, J. Hills, and J. Lines, "Transformation based ensembles for time series classification", *in Proceedings of the Twelfth SIAM International Conference on Data Mining, Anaheim, California, USA, April 26-28, 2012.*, SIAM / Omnipress, 2012, pp. 307–318, ISBN: 978-1-61197-232-0. DOI: 10.1137/1.9781611972825.27. [Online]. Available: https://doi.org/10.1137/1.9781611972825.27.

[BDKS04] M. W. Berry, U. Dayal, C. Kamath, and D. B. Skillicorn, Eds., *Proceedings of the Fourth SIAM International Conference on Data Mining, Lake Buena Vista, Florida, USA, April 22-24, 2004*, SIAM, 2004, ISBN: 978-0-89871-568-2. DOI: 10.1137/1.9781611972740. [Online]. Available: https://doi.org/10.1137/1.9781611972740.

[BDL+18] A. J. Bagnall, H. A. Dau, J. Lines, M. Flynn, J. Large, A. Bostrom, P. Southam, and E. J. Keogh, "The UEA multivariate time series classification archive, 2018", *CoRR*, vol. abs/1811.00075, 2018. arXiv: 1811.00075. [Online]. Available: http://arxiv.org/abs/1811.00075.

[Bel03] R. Bellman, *Dynamic Programming*, ser. Dover Books on Computer Science Series. Dover Publications, 2003, ISBN: 9780486428093. [Online]. Available: https://books.google.fr/books?id=fyVtp3EMxasC.

[Bel15] R. E. Bellman, *Adaptive control processes: a guided tour*. Princeton university press, 2015, vol. 2045.

[BFOS84] L. Breiman, J. H. Friedman, R. A. Olshen, and C. J. Stone, *Classification and Regression Trees*. Wadsworth, 1984, ISBN: 0-534-98053-8.

[BK59] R. Bellman and R. Kalaba, "On adaptive control processes", *IRE Transactions on Automatic Control*, vol. 4, no. 2, pp. 1–9, Nov. 1959, ISSN: 0096-199X. DOI: 10.1109/TAC.1959.1104847.

[BK73] C. Bron and J. Kerbosch, "Finding all cliques of an undirected graph (algorithm 457)", *Commun. ACM*, vol. 16, no. 9, pp. 575–576, 1973.

[BKTdS14] G. E. A. P. A. Batista, E. J. Keogh, O. M. Tataw, and V. M. A. de Souza, "CID: an efficient complexity-invariant distance for time series", *Data Min. Knowl. Discov.*, vol. 28, no. 3, pp. 634–669, 2014. DOI: 10.1007/s10618-013-0312-3. [Online]. Available: https://doi.org/10.1007/s10618-013-0312-3.

[BL97] A. Blum and P. Langley, "Selection of relevant features and examples in machine learning", *Artif. Intell.*, vol. 97, no. 1-2, pp. 245–271, 1997. DOI: 10.1016/S0004-3702(97)00063-5. [Online]. Available: https://doi.org/10.1016/S0004-3702(97)00063-5.

[BLB+17] A. J. Bagnall, J. Lines, A. Bostrom, J. Large, and E. J. Keogh, "The great time series classification bake off: a review and experimental evaluation of recent algorithmic advances", *Data Min. Knowl. Discov.*, vol. 31, no. 3, pp. 606–660, 2017. DOI: 10.1007/s10618-016-0483-9. [Online]. Available: https://doi.org/10.1007/s10618-016-0483-9.

[Blu05] A. Blum, "Random projection, margins, kernels, and feature-selection", *in Subspace, Latent Structure and Feature Selection, Statistical and Optimization, Perspectives Workshop, SLSFS 2005, Bohinj, Slovenia, February 23-25, 2005, Revised Selected Papers*, C. Saunders, M. Grobelnik, S. R. Gunn, and J. Shawe-Taylor, Eds., ser. Lecture Notes in Computer Science, vol. 3940, Springer, 2005, pp. 52–68, ISBN: 3-540-34137-4. DOI: 10.1007/11752790_3. [Online]. Available: https://doi.org/10.1007/11752790%5C_3.

[BR14] V. Bettaiah and H. S. Ranganath, "An analysis of time series representation methods: data mining applications perspective", *in Proceedings of the 2014 ACM Southeast Regional Conference, Kennesaw, GA, USA, March 28 - 29, 2014*, K. E. Hoganson and J. (He, Eds., ACM, 2014, 16:1–16:6, ISBN: 978-1-4503-2923-1. DOI: 10.1145/2638404.2638475. [Online]. Available: https://doi.org/10.1145/2638404.2638475.

[BWK11] G. E. A. P. A. Batista, X. Wang, and E. J. Keogh, "A complexity-invariant distance measure for time series", *in Proceedings of the Eleventh SIAM International Conference on Data Mining, SDM 2011, April 28-30, 2011, Mesa, Arizona, USA*, SIAM / Omnipress, 2011, pp. 699–710, ISBN: 978-0-89871-992-5. DOI: 10.1137/1.9781611972818.60. [Online]. Available: https://doi.org/10.1137/1.9781611972818.60.

[Car15] Caroline Kleist, "Time Series Data Mining Methods: A Review", *Engineering Applications of Artificial Intelligence*, vol. 24, no. 533039, p. 61, 2015. [Online]. Available: http://linkinghub.elsevier.com/retrieve/pii/S0952197610001727.

[CB17] M. Cuturi and M. Blondel, “Soft-dtw: a differentiable loss function for time-series”, *in Proceedings of the 34th International Conference on Machine Learning, ICML 2017, Sydney, NSW, Australia, 6-11 August 2017*, D. Precup and Y. W. Teh, Eds., ser. Proceedings of Machine Learning Research, vol. 70, PMLR, 2017, pp. 894–903. [Online]. Available: `http://proceedings.mlr.press/v70/cuturi17a.html`.

[CD11] A. Cuzzocrea and U. Dayal, Eds., *Data Warehousing and Knowledge Discovery - 13th International Conference, DaWaK 2011, Toulouse, France, August 29-September 2,2011. Proceedings*, vol. 6862, Lecture Notes in Computer Science, Springer, 2011, ISBN: 978-3-642-23543-6. DOI: `10.1007/978-3-642-23544-3`. [Online]. Available: `https://doi.org/10.1007/978-3-642-23544-3`.

[CDHR12] K. Chang, B. Deka, W. W. Hwu, and D. Roth, “Efficient pattern-based time series classification on GPU”, *in 12th IEEE International Conference on Data Mining, ICDM 2012, Brussels, Belgium, December 10-13, 2012*, M. J. Zaki, A. Siebes, J. X. Yu, B. Goethals, G. I. Webb, and X. Wu, Eds., IEEE Computer Society, 2012, pp. 131–140, ISBN: 978-1-4673-4649-8. DOI: `10.1109/ICDM.2012.132`. [Online]. Available: `https://doi.org/10.1109/ICDM.2012.132`.

[CF99] K. Chan and A. W. Fu, “Efficient time series matching by wavelets”, *in Proceedings of the 15th International Conference on Data Engineering, Sydney, Australia, March 23-26, 1999*, M. Kitsuregawa, M. P. Papazoglou, and C. Pu, Eds., IEEE Computer Society, 1999, pp. 126–133, ISBN: 0-7695-0071-4. DOI: `10.1109/ICDE.1999.754915`. [Online]. Available: `https://doi.org/10.1109/ICDE.1999.754915`.

[CGM13] J. M. Cadenas, M. C. Garrido, and R. Martinez, “Feature subset selection filter-wrapper based on low quality data”, *Expert Syst. Appl.*, vol. 40, no. 16, pp. 6241–6252, 2013. DOI: `10.1016/j.eswa.2013.05.051`. [Online]. Available: `https://doi.org/10.1016/j.eswa.2013.05.051`.

[CK08] F. Cazals and C. Karande, “A note on the problem of reporting maximal cliques”, *Theor. Comput. Sci.*, vol. 407, no. 1-3, pp. 564–568, 2008. DOI: `10.1016/j.tcs.2008.05.010`. [Online]. Available: `https://doi.org/10.1016/j.tcs.2008.05.010`.

[CKH+15] Y. Chen, E. Keogh, B. Hu, N. Begum, A. Bagnall, A. Mueen, and G. Batista, *The UCR time series classification archive*, `www.cs.ucr.edu/~eamonn/time_series_data/`, Jul. 2015.

[CKHP02] S. Chu, E. J. Keogh, D. M. Hart, and M. J. Pazzani, “Iterative deepening dynamic time warping for time series”, *in Proceedings of the Second SIAM International Conference on Data Mining, Arlington, VA, USA, April 11-13, 2002*, R. L. Grossman, J. Han, V. Kumar, H. Mannila, and R. Motwani, Eds., SIAM, 2002, pp. 195–212, ISBN: 978-0-89871-517-0. DOI: `10.1137/1.9781611972726.12`. [Online]. Available: `https://doi.org/10.1137/1.9781611972726.12`.

[CKMP02] K. Chakrabarti, E. J. Keogh, S. Mehrotra, and M. J. Pazzani, “Locally adaptive dimensionality reduction for indexing large time series databases”, *ACM Trans. Database Syst.*, vol. 27, no. 2, pp. 188–228, 2002. DOI: `10.1145/568518.568520`. [Online]. Available: `https://doi.org/10.1145/568518.568520`.

[Clo17] I. M. Cloud, *10 key marketing trends for 2017 and ideas for exceeding customer expectations*, `https://public.dhe.ibm.com/common/ssi/ecm/wr/en/wrl12345usen/watson-customer-engagement-watson-marketing-wr-other-papers-and-reports-wrl12345usen-20170719.pdf`, Accessed: 2018-09-30, 2017.

[CLW01] N. Cercone, T. Y. Lin, and X. Wu, Eds., *Proceedings of the 2001 IEEE International Conference on Data Mining, 29 November - 2 December 2001, San Jose, California, USA*, IEEE Computer Society, 2001, ISBN: 0-7695-1119-8. [Online]. Available: `http://ieeexplore.ieee.org/xpl/mostRecentIssue.jsp?punumber=7762`.

[ÇMC15] M. S. Çetin, A. Mueen, and V. D. Calhoun, "Shapelet ensemble for multi-dimensional time series", *in Proceedings of the 2015 SIAM International Conference on Data Mining, Vancouver, BC, Canada, April 30 - May 2, 2015*, S. Venkatasubramanian and J. Ye, Eds., SIAM, 2015, pp. 307–315, ISBN: 978-1-61197-401-0. DOI: 10.1137/1.9781611974010.35. [Online]. Available: https://doi.org/10.1137/1.9781611974010.35.

[CN04] L. Chen and R. T. Ng, "On the marriage of lp-norms and edit distance", *in (e)Proceedings of the Thirtieth International Conference on Very Large Data Bases, VLDB 2004, Toronto, Canada, August 31 - September 3 2004*, M. A. Nascimento, M. T. Özsu, D. Kossmann, R. J. Miller, J. A. Blakeley, and K. B. Schiefer, Eds., Morgan Kaufmann, 2004, pp. 792–803, ISBN: 0-12-088469-0. DOI: 10.1016/B978-012088469-8.50070-X. [Online]. Available: http://www.vldb.org/conf/2004/RS21P2.PDF.

[CP08] M. Corduas and D. Piccolo, "Time series clustering and classification by the autoregressive metric", *Computational Statistics & Data Analysis*, vol. 52, no. 4, pp. 1860–1872, 2008. DOI: 10.1016/j.csda.2007.06.001. [Online]. Available: https://doi.org/10.1016/j.csda.2007.06.001.

[CPZ97] P. Ciaccia, M. Patella, and P. Zezula, "M-tree: an efficient access method for similarity search in metric spaces", *in VLDB'97, Proceedings of 23rd International Conference on Very Large Data Bases, August 25-29, 1997, Athens, Greece*, M. Jarke, M. J. Carey, K. R. Dittrich, F. H. Lochovsky, P. Loucopoulos, and M. A. Jeusfeld, Eds., Morgan Kaufmann, 1997, pp. 426–435, ISBN: 1-55860-470-7. [Online]. Available: http://www.vldb.org/conf/1997/P426.PDF.

[CS04] S. Chen and M. Shyu, Eds., *Proceedings of the Second ACM International Workshop on Multimedia Databases, ACM-MMDB 2004, Washington, DC, USA, November 13, 2004*, ACM, 2004, ISBN: 1-58113-975-6.

[CTTY13] H. Chen, F. Tang, P. Tiño, and X. Yao, "Model-based kernel for efficient time series analysis", *in The 19th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD 2013, Chicago, IL, USA, August 11-14, 2013*, I. S. Dhillon, Y. Koren, R. Ghani, T. E. Senator, P. Bradley, R. Parekh, J. He, R. L. Grossman, and R. Uthurusamy, Eds., ACM, 2013, pp. 392–400, ISBN: 978-1-4503-2174-7. DOI: 10.1145/2487575.2487700. [Online]. Available: https://doi.org/10.1145/2487575.2487700.

[CV95] C. Cortes and V. Vapnik, "Support-vector networks", *Machine Learning*, vol. 20, no. 3, pp. 273–297, 1995. DOI: 10.1007/BF00994018. [Online]. Available: https://doi.org/10.1007/BF00994018.

[CW17] N. Chawla and W. Wang, Eds., *Proceedings of the 2017 SIAM International Conference on Data Mining, Houston, Texas, USA, April 27-29, 2017*, SIAM, 2017, ISBN: 978-1-61197-497-3. DOI: 10.1137/1.9781611974973. [Online]. Available: https://doi.org/10.1137/1.9781611974973.

[DGSZ03] V. Dohnal, C. Gennaro, P. Savino, and P. Zezula, "D-index: distance searching index for metric data sets", *Multimedia Tools Appl.*, vol. 21, no. 1, pp. 9–33, 2003. DOI: 10.1023/A:1025026030880. [Online]. Available: https://doi.org/10.1023/A:1025026030880.

[DKG+13] I. S. Dhillon, Y. Koren, R. Ghani, T. E. Senator, P. Bradley, R. Parekh, J. He, R. L. Grossman, and R. Uthurusamy, Eds., *The 19th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD 2013, Chicago, IL, USA, August 11-14, 2013*, ACM, 2013, ISBN: 978-1-4503-2174-7. [Online]. Available: http://dl.acm.org/citation.cfm?id=2487575.

[dSRdE+11] S. F. da Silva, M. X. Ribeiro, J. do E. S. Batista Neto, C. T. Jr., and A. J. M. Traina, "Improving the ranking quality of medical image retrieval using a genetic feature selection method", *Decision Support Systems*, vol. 51, no. 4, pp. 810–820, 2011. DOI: 10.1016/j.dss.2011.01.015. [Online]. Available: https://doi.org/10.1016/j.dss.2011.01.015.

[DTS+08] H. Ding, G. Trajcevski, P. Scheuermann, X. Wang, and E. J. Keogh, "Querying and mining of time series data: experimental comparison of representations and distance measures", *PVLDB*, vol. 1, no. 2, pp. 1542–1552, 2008. DOI: 10.14778/1454159.1454226. [Online]. Available: http://www.vldb.org/pvldb/1/1454226.pdf.

[DWL+06] U. Dayal, K. Whang, D. B. Lomet, G. Alonso, G. M. Lohman, M. L. Kersten, S. K. Cha, and Y. Kim, Eds., *Proceedings of the 32nd International Conference on Very Large Data Bases, Seoul, Korea, September 12-15, 2006*, ACM, 2006, ISBN: 1-59593-385-9. [Online]. Available: http://dl.acm.org/citation.cfm?id=1182635.

[EA12] P. Esling and C. Agón, "Time-series data mining", *ACM Comput. Surv.*, vol. 45, no. 1, 12:1–12:34, 2012. DOI: 10.1145/2379776.2379788. [Online]. Available: http://doi.acm.org/10.1145/2379776.2379788.

[EBC+00] A. El Abbadi, M. L. Brodie, S. Chakravarthy, U. Dayal, N. Kamel, G. Schlageter, and K. Whang, Eds., *VLDB 2000, Proceedings of 26th International Conference on Very Large Data Bases, September 10-14, 2000, Cairo, Egypt*, Morgan Kaufmann, 2000, ISBN: 1-55860-715-3.

[ES18] J. Euzenat and F. Schwarzentruber, Eds., *Actes de la Conférence Nationale d'Intelligence Artificielle et Rencontres des Jeunes Chercheurs en Intelligence Artificielle (CNIA+RJCIA 2018), Nancy, France, 4-6 Juillet 2018*, vol. 2133, CEUR Workshop Proceedings, CEUR-WS.org, 2018. [Online]. Available: http://ceur-ws.org/Vol-2133.

[EUCG06] T. Eliassi-Rad, L. H. Ungar, M. Craven, and D. Gunopulos, Eds., *Proceedings of the Twelfth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Philadelphia, PA, USA, August 20-23, 2006*, ACM, 2006, ISBN: 1-59593-339-5.

[Fis19] R. A. Fisher, "Xv.—the correlation between relatives on the supposition of mendelian inheritance.", *Transactions of the royal society of Edinburgh*, vol. 52, no. 02, pp. 399–433, 1919.

[FNV18] V. S. S. Fotso, E. M. Nguifo, and P. Vaslin, "Découverte des u-shapelets basée sur la corrélation pour le clustering de séries temporelles incertaines(frobenius correlation based u-shapelets discovery for time series clustering)", *in Actes de la Conférence Nationale d'Intelligence Artificielle et Rencontres des Jeunes Chercheurs en Intelligence Artificielle (CNIA+RJCIA 2018), Nancy, France, 4-6 Juillet 2018.*, J. Euzenat and F. Schwarzentruber, Eds., ser. CEUR Workshop Proceedings, vol. 2133, CEUR-WS.org, 2018, pp. 1–9. [Online]. Available: http://ceur-ws.org/Vol-2133/rjcia-paper1.pdf.

[FRM94] C. Faloutsos, M. Ranganathan, and Y. Manolopoulos, "Fast subsequence matching in time-series databases", *in Proceedings of the 1994 ACM SIGMOD International Conference on Management of Data, Minneapolis, Minnesota, USA, May 24-27, 1994.*, R. T. Snodgrass and M. Winslett, Eds., ACM Press, 1994, pp. 419–429. DOI: 10.1145/191839.191925. [Online]. Available: https://doi.org/10.1145/191839.191925.

[Fu11] T. Fu, "A review on time series data mining", *Eng. Appl. of AI*, vol. 24, no. 1, pp. 164–181, 2011. DOI: 10.1016/j.engappai.2010.09.007. [Online]. Available: https://doi.org/10.1016/j.engappai.2010.09.007.

[FU94] U. M. Fayyad and R. Uthurusamy, Eds., *Knowledge Discovery in Databases: Papers from the 1994 AAAI Workshop, Seattle, Washington, USA, July 1994. Technical Report WS-94-03*, AAAI Press, 1994, ISBN: 0-929280-73-3.

[GAD+02] M. Ghil, M. Allen, M. Dettinger, K. Ide, D. Kondrashov, M. Mann, A. W. Robertson, A. Saunders, Y. Tian, F. Varadi, *et al.*, "Advanced spectral methods for climatic time series", *Reviews of geophysics*, vol. 40, no. 1, pp. 3–1, 2002.

[Gam17] J. C. B. Gamboa, "Deep learning for time-series analysis", *CoRR*, vol. abs/1701.01887, 2017. arXiv: 1701.01887. [Online]. Available: http://arxiv.org/abs/1701.01887.

[GB01] D. Georgakopoulos and A. Buchmann, Eds., *Proceedings of the 17th International Conference on Data Engineering, April 2-6, 2001, Heidelberg, Germany*, IEEE Computer Society, 2001, ISBN: 0-7695-1001-9. [Online]. Available: `http://ieeexplore.ieee.org/xpl/mostRecentIssue.jsp?punumber=7304`.

[Gha07] Z. Ghahramani, Ed., *Machine Learning, Proceedings of the Twenty-Fourth International Conference (ICML 2007), Corvallis, Oregon, USA, June 20-24, 2007*, vol. 227, ACM International Conference Proceeding Series, ACM, 2007, ISBN: 978-1-59593-793-3.

[GHK+02] R. L. Grossman, J. Han, V. Kumar, H. Mannila, and R. Motwani, Eds., *Proceedings of the Second SIAM International Conference on Data Mining, Arlington, VA, USA, April 11-13, 2002*, SIAM, 2002, ISBN: 978-0-89871-517-0. DOI: `10.1137/1.9781611972726`. [Online]. Available: `https://doi.org/10.1137/1.9781611972726`.

[Gio+09] T. Giorgino *et al.*, "Computing and visualizing dynamic time warping alignments in r: the dtw package", *Journal of statistical Software*, vol. 31, no. 7, pp. 1–24, 2009.

[GL00] O. Günther and H. Lenz, Eds., *Proceedings of the 12th International Conference on Scientific and Statistical Database Management, Berlin, Germany, July 26-28, 2000*, IEEE Computer Society, 2000, ISBN: 0-7695-0686-0. [Online]. Available: `http://ieeexplore.ieee.org/xpl/mostRecentIssue.jsp?punumber=6993`.

[GNZ01] N. Golyandina, V. Nekrutkin, and A. A. Zhigljavsky, *Analysis of time series structure: SSA and related techniques.* Chapman and Hall/CRC, 2001.

[GRO13] M. F. Ghalwash, V. Radosavljevic, and Z. Obradovic, "Extraction of interpretable multivariate patterns for early diagnostics", *in 2013 IEEE 13th International Conference on Data Mining, Dallas, TX, USA, December 7-10, 2013*, H. Xiong, G. Karypis, B. M. Thuraisingham, D. J. Cook, and X. Wu, Eds., IEEE Computer Society, 2013, pp. 201–210, ISBN: 978-0-7695-5108-1. DOI: `10.1109/ICDM.2013.19`. [Online]. Available: `https://doi.org/10.1109/ICDM.2013.19`.

[GSWS14] J. Grabocka, N. Schilling, M. Wistuba, and L. Schmidt-Thieme, "Learning time-series shapelets", *in The 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD '14, New York, NY, USA - August 24 - 27, 2014*, S. A. Macskassy, C. Perlich, J. Leskovec, W. Wang, and R. Ghani, Eds., ACM, 2014, pp. 392–401, ISBN: 978-1-4503-2956-9. DOI: `10.1145/2623330.2623613`. [Online]. Available: `https://doi.org/10.1145/2623330.2623613`.

[GSZ11] K. Golmohammadi, M. Smit, and O. R. Zaïane, "Learning actions in complex software systems", *in Data Warehousing and Knowledge Discovery - 13th International Conference, DaWaK 2011, Toulouse, France, August 29-September 2,2011. Proceedings*, A. Cuzzocrea and U. Dayal, Eds., ser. Lecture Notes in Computer Science, vol. 6862, Springer, 2011, pp. 367–381, ISBN: 978-3-642-23543-6. DOI: `10.1007/978-3-642-23544-3\_28`. [Online]. Available: `https://doi.org/10.1007/978-3-642-23544-3%5C_28`.

[Has07] H. Hassani, "Singular spectrum analysis: methodology and comparison", 2007.

[HCBV10] F. P. Holgado–Tello, S. Chacón–Moscoso, I. Barbero–García, and E. Vila–Abad, "Polychoric versus pearson correlations in exploratory and confirmatory factor analysis of ordinal variables", *Quality & Quantity*, vol. 44, no. 1, p. 153, 2010.

[HCN05] X. He, D. Cai, and P. Niyogi, "Laplacian score for feature selection", *in Advances in Neural Information Processing Systems 18 [Neural Information Processing Systems, NIPS 2005, December 5-8, 2005, Vancouver, British Columbia, Canada]*, 2005, pp. 507–514. [Online]. Available: `http://papers.nips.cc/paper/2909-laplacian-score-for-feature-selection`.

[HDZ+12] Q. He, Z. Dong, F. Zhuang, T. Shang, and Z. Shi, "Fast time series classification based on infrequent shapelets", *in 11th International Conference on Machine Learning and Applications, ICMLA, Boca Raton, FL, USA, December 12-15, 2012. Volume 1*, IEEE, 2012, pp. 215–219, ISBN: 978-1-4673-4651-1. DOI: `10.1109/ICMLA.2012.44`. [Online]. Available: `https://doi.org/10.1109/ICMLA.2012.44`.

[HH14] K. E. Hoganson and J. (He, Eds., *Proceedings of the 2014 ACM Southeast Regional Conference, Kennesaw, GA, USA, March 28 - 29, 2014*, ACM, 2014, ISBN: 978-1-4503-2923-1. [Online]. Available: `http://dl.acm.org/citation.cfm?id=2638404`.

[HK04] T. Hagerup and J. Katajainen, Eds., *Algorithm Theory - SWAT 2004, 9th Scandinavian Workshop on Algorithm Theory, Humlebaek, Denmark, July 8-10, 2004, Proceedings*, vol. 3111, Lecture Notes in Computer Science, Springer, 2004, ISBN: 3-540-22339-8. DOI: `10.1007/b98413`. [Online]. Available: `https://doi.org/10.1007/b98413`.

[HKZ16] L. Hou, J. T. Kwok, and J. M. Zurada, "Efficient learning of timeseries shapelets", *in Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence, February 12-17, 2016, Phoenix, Arizona, USA.*, D. Schuurmans and M. P. Wellman, Eds., AAAI Press, 2016, pp. 1209–1215, ISBN: 978-1-57735-760-5. [Online]. Available: `http://www.aaai.org/ocs/index.php/AAAI/AAAI16/paper/view/12382`.

[HLB+14] J. Hills, J. Lines, E. Baranauskas, J. Mapp, and A. J. Bagnall, "Classification of time series by shapelet transformation", *Data Min. Knowl. Discov.*, vol. 28, no. 4, pp. 851–881, 2014. DOI: `10.1007/s10618-013-0322-1`. [Online]. Available: `https://doi.org/10.1007/s10618-013-0322-1`.

[HTF13] T. Hastie, R. Tibshirani, and J. Friedman, *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*, ser. Springer Series in Statistics. Springer New York, 2013, ISBN: 9780387216065. [Online]. Available: `https://books.google.at/books?id=yPfZBwAAQBAJ`.

[IFFZ09] J. F. E. IV, F. Fogelman-Soulié, P. A. Flach, and M. J. Zaki, Eds., *Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Paris, France, June 28 - July 1, 2009*, ACM, 2009, ISBN: 978-1-60558-495-9.

[IKM00] P. Indyk, N. Koudas, and S. Muthukrishnan, "Identifying representative trends in massive time series data sets using sketches", *in VLDB 2000, Proceedings of 26th International Conference on Very Large Data Bases, September 10-14, 2000, Cairo, Egypt*, A. El Abbadi, M. L. Brodie, S. Chakravarthy, U. Dayal, N. Kamel, G. Schlageter, and K. Whang, Eds., Morgan Kaufmann, 2000, pp. 363–372, ISBN: 1-55860-715-3. [Online]. Available: `http://www.vldb.org/conf/2000/P363.pdf`.

[Ita75] F. Itakura, "Minimum prediction residual principle applied to speech recognition", *IEEE Trans. Signal Processing*, vol. 23, no. 1, pp. 67–72, 1975.

[JC16] I. T. Jolliffe and J. Cadima, "Principal component analysis: a review and recent developments", *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences*, vol. 374, no. 2065, p. 20 150 202, 2016.

[JCD+97] M. Jarke, M. J. Carey, K. R. Dittrich, F. H. Lochovsky, P. Loucopoulos, and M. A. Jeusfeld, Eds., *VLDB'97, Proceedings of 23rd International Conference on Very Large Data Bases, August 25-29, 1997, Athens, Greece*, Morgan Kaufmann, 1997, ISBN: 1-55860-470-7.

[JJO11] Y. Jeong, M. K. Jeong, and O. A. Omitaomu, "Weighted dynamic time warping for time series classification", *Pattern Recognition*, vol. 44, no. 9, pp. 2231–2240, 2011. DOI: `10.1016/j.patcog.2010.09.022`. [Online]. Available: `https://doi.org/10.1016/j.patcog.2010.09.022`.

[JM09] D. Jurafsky and J. H. Martin, *Speech and language processing: an introduction to natural language processing, computational linguistics, and speech recognition, 2nd Edition*, ser. Prentice Hall series in artificial intelligence. Prentice Hall, Pearson Education International, 2009, ISBN: 9780135041963. [Online]. Available: `http://www.worldcat.org/oclc/315913020`.

[Jol02] I. Jolliffe, *Principal component analysis*. New York: Springer Verlag, 2002.

[JTSF00] C. T. Jr., A. J. M. Traina, B. Seeger, and C. Faloutsos, "Slim-trees: high performance metric trees minimizing overlap between nodes", *in Advances in Database Technology - EDBT 2000, 7th International Conference on Extending Database Technology, Konstanz, Germany, March 27-31, 2000, Proceedings*, C. Zaniolo, P. C. Lockemann, M. H. Scholl, and T. Grust, Eds., ser. Lecture Notes in Computer Science, vol. 1777, Springer, 2000, pp. 51–65, ISBN: 3-540-67227-3. DOI: `10.1007/3-540-46439-5\_4`. [Online]. Available: `https://doi.org/10.1007/3-540-46439-5%5C_4`.

[KCHP04] E. Keogh, S. Chu, D. Hart, and M. Pazzani, "Segmenting time series: a survey and novel approach", *in Data mining in time series databases*, World Scientific, 2004, pp. 1–21.

[KCPM01] E. J. Keogh, K. Chakrabarti, M. J. Pazzani, and S. Mehrotra, "Dimensionality reduction for fast similarity search in large time series databases", *Knowl. Inf. Syst.*, vol. 3, no. 3, pp. 263–286, 2001. DOI: `10.1007/PL00011669`. [Online]. Available: `https://doi.org/10.1007/PL00011669`.

[KDD06] A. KDD, *Data mining curriculum: a proposal*, `https://www.kdd.org/curriculum/index.html`, Accessed: 2018-09-30, 2006.

[Keo02] E. J. Keogh, "Exact indexing of dynamic time warping", *in VLDB 2002, Proceedings of 28th International Conference on Very Large Data Bases, August 20-23, 2002, Hong Kong, China*, Morgan Kaufmann, 2002, pp. 406–417. [Online]. Available: `http://www.vldb.org/conf/2002/S12P01.pdf`.

[KG01] V. Kumar and R. L. Grossman, Eds., *Proceedings of the First SIAM International Conference on Data Mining, SDM 2001, Chicago, IL, USA, April 5-7, 2001*, SIAM, 2001, ISBN: 978-0-89871-495-1. DOI: `10.1137/1.9781611972719`. [Online]. Available: `https://doi.org/10.1137/1.9781611972719`.

[KGP01] K. Kalpakis, D. Gada, and V. Puttagunta, "Distance measures for effective clustering of ARIMA time-series", *in Proceedings of the 2001 IEEE International Conference on Data Mining, 29 November - 2 December 2001, San Jose, California, USA*, N. Cercone, T. Y. Lin, and X. Wu, Eds., IEEE Computer Society, 2001, pp. 273–280, ISBN: 0-7695-1119-8. DOI: `10.1109/ICDM.2001.989529`. [Online]. Available: `https://doi.org/10.1109/ICDM.2001.989529`.

[KJ97] R. Kohavi and G. H. John, "Wrappers for feature subset selection", *Artif. Intell.*, vol. 97, no. 1-2, pp. 273–324, 1997. DOI: `10.1016/S0004-3702(97)00043-X`. [Online]. Available: `https://doi.org/10.1016/S0004-3702(97)00043-X`.

[KJF97] F. Korn, H. V. Jagadish, and C. Faloutsos, "Efficiently supporting ad hoc queries in large datasets of time sequences", *in SIGMOD 1997, Proceedings ACM SIGMOD International Conference on Management of Data, May 13-15, 1997, Tucson, Arizona, USA.*, J. Peckham, Ed., ACM Press, 1997, pp. 289–300. DOI: `10.1145/253260.253332`. [Online]. Available: `https://doi.org/10.1145/253260.253332`.

[KK03] E. J. Keogh and S. Kasetty, "On the need for time series data mining benchmarks: A survey and empirical demonstration", *Data Min. Knowl. Discov.*, vol. 7, no. 4, pp. 349–371, 2003. DOI: `10.1023/A:1024988512476`. [Online]. Available: `https://doi.org/10.1023/A:1024988512476`.

[KL05] E. J. Keogh and J. Lin, "Clustering of time-series subsequences is meaningless: implications for previous and future research", *Knowl. Inf. Syst.*, vol. 8, no. 2, pp. 154–177, 2005. DOI: `10.1007/s10115-004-0172-7`. [Online]. Available: `https://doi.org/10.1007/s10115-004-0172-7`.

[KL83] J. B. Kruskal and M. Liberman, "The symmetric time-warping problem: from continuous to discrete", *Time warps, string edits and macromolecules: The theory and practice of sequence comparison*, pp. 125–161, 1983.

[KP00] E. J. Keogh and M. J. Pazzani, "Scaling up dynamic time warping for datamining applications", *in Proceedings of the sixth ACM SIGKDD international conference on Knowledge discovery and data mining, Boston, MA, USA, August 20-23, 2000*, R. Ramakrishnan, S. J. Stolfo, R. J. Bayardo, and I. Parsa, Eds., ACM, 2000, pp. 285–289, ISBN: 1-58113-233-6. DOI: 10.1145/347090.347153. [Online]. Available: https://doi.org/10.1145/347090.347153.

[KP01] ——, "Derivative dynamic time warping", *in Proceedings of the First SIAM International Conference on Data Mining, SDM 2001, Chicago, IL, USA, April 5-7, 2001*, V. Kumar and R. L. Grossman, Eds., SIAM, 2001, pp. 1–11, ISBN: 978-0-89871-495-1. DOI: 10.1137/1.9781611972719.1. [Online]. Available: https://doi.org/10.1137/1.9781611972719.1.

[KPA15] I. Karlsson, P. Papapetrou, and L. Asker, "Multi-channel ECG classification using forests of randomized shapelet trees", *in Proceedings of the 8th ACM International Conference on PErvasive Technologies Related to Assistive Environments, PETRA 2015, Corfu, Greece, July 1-3, 2015*, F. Makedon, Ed., ACM, 2015, 43:1–43:6, ISBN: 978-1-4503-3452-5. DOI: 10.1145/2769493.2769520. [Online]. Available: https://doi.org/10.1145/2769493.2769520.

[KPC01] S. Kim, S. Park, and W. W. Chu, "An index-based approach for similarity search supporting time warping in large sequence databases", *in Proceedings of the 17th International Conference on Data Engineering, April 2-6, 2001, Heidelberg, Germany*, D. Georgakopoulos and A. Buchmann, Eds., IEEE Computer Society, 2001, pp. 607–614, ISBN: 0-7695-1001-9. DOI: 10.1109/ICDE.2001.914875. [Online]. Available: https://doi.org/10.1109/ICDE.2001.914875.

[KPP99] M. Kitsuregawa, M. P. Papazoglou, and C. Pu, Eds., *Proceedings of the 15th International Conference on Data Engineering, Sydney, Australia, March 23-26, 1999*, IEEE Computer Society, 1999, ISBN: 0-7695-0071-4. [Online]. Available: http://ieeexplore.ieee.org/xpl/mostRecentIssue.jsp?punumber=6088.

[KR05] E. J. Keogh and C. (Ratanamahatana, "Exact indexing of dynamic time warping", *Knowl. Inf. Syst.*, vol. 7, no. 3, pp. 358–386, 2005. [Online]. Available: http://www.springerlink.com/index/10.1007/s10115-004-0154-9.

[KR13] E. J. Keogh and T. Rakthanmanon, "Fast shapelets: A scalable algorithm for discovering time series shapelets", *in Proceedings of the 13th SIAM International Conference on Data Mining, May 2-4, 2013. Austin, Texas, USA.*, SIAM, 2013, pp. 668–676, ISBN: 978-1-61197-262-7. DOI: 10.1137/1.9781611972832.74. [Online]. Available: https://doi.org/10.1137/1.9781611972832.74.

[KRGI11] V. Kurbalija, M. Radovanović, Z. Geler, and M. Ivanović, "The influence of global constraints on dtw and lcs similarity measures for time-series databases", *in Third International Conference on Software, Services and Semantic Technologies S3T 2011*, D. Dicheva, Z. Markov, and E. Stefanova, Eds., Berlin, Heidelberg: Springer Berlin Heidelberg, 2011, pp. 67–74, ISBN: 978-3-642-23163-6.

[Kru83] J. B. Kruskal, "An overview of sequence comparison: time warps, string edits, and macromolecules", *SIAM review*, vol. 25, no. 2, pp. 201–237, 1983.

[KS05] M. W. Kadous and C. Sammut, "Classification of multivariate time series and structured data using constructive induction", *Machine Learning*, vol. 58, no. 2-3, pp. 179–216, 2005. DOI: 10.1007/s10994-005-5826-5. [Online]. Available: https://doi.org/10.1007/s10994-005-5826-5.

[KSKG05] H. Kargupta, J. Srivastava, C. Kamath, and A. Goodman, Eds., *Proceedings of the 2005 SIAM International Conference on Data Mining, SDM 2005, Newport Beach, CA, USA, April 21-23, 2005*, SIAM, 2005, ISBN: 978-0-89871-593-4. DOI: 10.1137/1.9781611972757. [Online]. Available: https://doi.org/10.1137/1.9781611972757.

[KSM16] T. Kocyan, K. Slaninová, and J. Martinovic, "Flexible global constraint extension for dynamic time warping", *in Computer Information Systems and Industrial Management - 15th IFIP TC8 International Conference, CISIM 2016, Vilnius, Lithuania, September 14-16, 2016, Proceedings*, K. Saeed and W. Homenda, Eds., ser. Lecture Notes in Computer Science, vol. 9842, Springer, 2016, pp. 389–401, ISBN: 978-3-319-45377-4. DOI: 10.1007/978-3-319-45378-1_35. [Online]. Available: https://doi.org/10.1007/978-3-319-45378-1%5C_35.

[KWX+06] E. J. Keogh, L. Wei, X. Xi, S. Lee, and M. Vlachos, "Lb_keogh supports exact indexing of shapes under rotation invariance with arbitrary representations and distance measures", *in Proceedings of the 32nd International Conference on Very Large Data Bases, Seoul, Korea, September 12-15, 2006*, U. Dayal, K. Whang, D. B. Lomet, G. Alonso, G. M. Lohman, M. L. Kersten, S. K. Cha, and Y. Kim, Eds., ACM, 2006, pp. 882–893, ISBN: 1-59593-385-9. [Online]. Available: http://dl.acm.org/citation.cfm?id=1164203.

[KWX+09] E. J. Keogh, L. Wei, X. Xi, M. Vlachos, S. Lee, and P. Protopapas, "Supporting exact indexing of arbitrarily rotated shapes and periodic time series under euclidean and warping distance measures", *VLDB J.*, vol. 18, no. 3, pp. 611–630, 2009. DOI: 10.1007/s00778-008-0111-4. [Online]. Available: https://doi.org/10.1007/s00778-008-0111-4.

[LB12] J. Lines and A. J. Bagnall, "Alternative quality measures for time series shapelets", *in Intelligent Data Engineering and Automated Learning - IDEAL 2012 - 13th International Conference, Natal, Brazil, August 29-31, 2012. Proceedings*, H. Yin, J. A. F. Costa, and G. D. A. Barreto, Eds., ser. Lecture Notes in Computer Science, vol. 7435, Springer, 2012, pp. 475–483, ISBN: 978-3-642-32638-7. DOI: 10.1007/978-3-642-32639-4_58. [Online]. Available: https://doi.org/10.1007/978-3-642-32639-4%5C_58.

[LCW+18] J. Li, K. Cheng, S. Wang, F. Morstatter, R. P. Trevino, J. Tang, and H. Liu, "Feature selection: A data perspective", *ACM Comput. Surv.*, vol. 50, no. 6, 94:1–94:45, 2018. DOI: 10.1145/3136625. [Online]. Available: https://doi.org/10.1145/3136625.

[LDHB12] J. Lines, L. M. Davis, J. Hills, and A. J. Bagnall, "A shapelet transform for time series classification", *in The 18th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD '12, Beijing, China, August 12-16, 2012*, Q. Yang, D. Agarwal, and J. Pei, Eds., ACM, 2012, pp. 289–297, ISBN: 978-1-4503-1462-6. DOI: 10.1145/2339530.2339579. [Online]. Available: https://doi.org/10.1145/2339530.2339579.

[Lee08] R. Y. Lee, Ed., *Software Engineering, Artificial Intelligence, Networking and Parallel/Distributed Computing*, ser. Studies in Computational Intelligence. Springer, 2008, vol. 149, ISBN: 978-3-540-70559-8.

[Lew92] D. D. Lewis, "Feature selection and feature extraction for text categorization", *in Proceedings of the workshop on Speech and Natural Language*, Association for Computational Linguistics, 1992, pp. 212–217.

[Lia05] T. W. Liao, "Clustering of time series data - a survey", *Pattern Recognition*, vol. 38, no. 11, pp. 1857–1874, 2005. DOI: 10.1016/j.patcog.2005.01.025. [Online]. Available: https://doi.org/10.1016/j.patcog.2005.01.025.

[LKL12] J. Lin, R. Khade, and Y. Li, "Rotation-invariant similarity in time series using bag-of-patterns representation", *J. Intell. Inf. Syst.*, vol. 39, no. 2, pp. 287–315, 2012. DOI: 10.1007/s10844-012-0196-5. [Online]. Available: https://doi.org/10.1007/s10844-012-0196-5.

[LKWL07] J. Lin, E. J. Keogh, L. Wei, and S. Lonardi, "Experiencing SAX: a novel symbolic representation of time series", *Data Min. Knowl. Discov.*, vol. 15, no. 2, pp. 107–144, 2007. DOI: 10.1007/s10618-007-0064-z. [Online]. Available: https://doi.org/10.1007/s10618-007-0064-z.

[LLS08] Y. Li, B. Liu, and S. Sarawagi, Eds., *Proceedings of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Las Vegas, Nevada, USA, August 24-27, 2008*, ACM, 2008, ISBN: 978-1-60558-193-4.

[LMTA17] A. Lods, S. Malinowski, R. Tavenard, and L. Amsaleg, "Learning dtw-preserving shapelets", *in Advances in Intelligent Data Analysis XVI - 16th International Symposium, IDA 2017, London, UK, October 26-28, 2017, Proceedings*, N. M. Adams, A. Tucker, and D. J. Weston, Eds., ser. Lecture Notes in Computer Science, vol. 10584, Springer, 2017, pp. 198–209, ISBN: 978-3-319-68764-3. DOI: 10.1007/978-3-319-68765-0_17. [Online]. Available: https://doi.org/10.1007/978-3-319-68765-0%5C_17.

[Lom93] D. B. Lomet, Ed., *Foundations of Data Organization and Algorithms, 4th International Conference, FODO'93, Chicago, Illinois, USA, October 13-15, 1993, Proceedings*, vol. 730, Lecture Notes in Computer Science, Springer, 1993, ISBN: 3-540-57301-1. DOI: 10.1007/3-540-57301-1. [Online]. Available: https://doi.org/10.1007/3-540-57301-1.

[LVVC11] S. Lhermitte, J. Verbesselt, W. W. Verstraeten, and P. Coppin, "A comparison of time series similarity measures for classification and change detection of ecosystem dynamics", *Remote sensing of environment*, vol. 115, no. 12, pp. 3129–3152, 2011.

[LZ04] Y. Liu and Y. F. Zheng, "Fs_sfs: a novel feature selection method for support vector machines", *in 2004 IEEE International Conference on Acoustics, Speech, and Signal Processing, ICASSP 2004, Montreal, Quebec, Canada, May 17-21, 2004*, IEEE, 2004, pp. 797–800, ISBN: 0-7803-8484-9. DOI: 10.1109/ICASSP.2004.1327231. [Online]. Available: https://doi.org/10.1109/ICASSP.2004.1327231.

[Mac67] J. MacQueen, "Some methods for classification and analysis of multivariate observations", *in Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability, Volume 1: Statistics*, Berkeley, Calif.: University of California Press, 1967, pp. 281–297. [Online]. Available: https://projecteuclid.org/euclid.bsmsp/1200512992.

[Mak15] F. Makedon, Ed., *Proceedings of the 8th ACM International Conference on PErvasive Technologies Related to Assistive Environments, PETRA 2015, Corfu, Greece, July 1-3, 2015*, ACM, 2015, ISBN: 978-1-4503-3452-5. [Online]. Available: http://dl.acm.org/citation.cfm?id=2769493.

[Mar18] L. Mary, *Searching Speech Databases*, ser. SpringerBriefs in Speech Technology Series. Springer, 2018, ISBN: 9783319977614. [Online]. Available: https://books.google.fr/books?id=kmBwDwAAQBAJ.

[McL04] G. McLachlan, *Discriminant Analysis and Statistical Pattern Recognition*, ser. Wiley Series in Probability and Statistics. Wiley, 2004, ISBN: 9780471691150. [Online]. Available: https://books.google.com.ph/books?id=O%5C_qHDLaWpDUC.

[MJP97] M. Mozer, M. I. Jordan, and T. Petsche, Eds., *Advances in Neural Information Processing Systems 9, NIPS, Denver, CO, USA, December 2-5, 1996*, MIT Press, 1997. [Online]. Available: http://papers.nips.cc/book/advances-in-neural-information-processing-systems-9-1996.

[MK10] A. Mueen and E. J. Keogh, "Online discovery and maintenance of time series motifs", *in Proceedings of the 16th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Washington, DC, USA, July 25-28, 2010*, B. Rao, B. Krishnapuram, A. Tomkins, and Q. Yang, Eds., ACM, 2010, pp. 1089–1098, ISBN: 978-1-4503-0055-1. DOI: 10.1145/1835804.1835941. [Online]. Available: https://doi.org/10.1145/1835804.1835941.

[MKY11] A. Mueen, E. J. Keogh, and N. E. Young, "Logical-shapelets: an expressive primitive for time series classification", *in Proceedings of the 17th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, San Diego, CA, USA, August 21-24, 2011*, C. Apté, J. Ghosh, and P. Smyth, Eds., ACM, 2011, pp. 1154–1162, ISBN: 978-1-4503-0813-7. DOI: 10.1145/2020408.2020587. [Online]. Available: https://doi.org/10.1145/2020408.2020587.

[MMSS99] C. Manning, C. Manning, H. Schütze, and H. SCHUTZE, *Foundations of Statistical Natural Language Processing*, ser. Mit Press. MIT Press, 1999, ISBN: 9780262133609. [Online]. Available: `https://books.google.fr/books?id=YiFDxbEX3SUC`.

[Mör06] F. Mörchen, "Algorithms for time series knowledge mining", *in Proceedings of the Twelfth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Philadelphia, PA, USA, August 20-23, 2006*, T. Eliassi-Rad, L. H. Ungar, M. Craven, and D. Gunopulos, Eds., ACM, 2006, pp. 668–673, ISBN: 1-59593-339-5. DOI: `10.1145/1150402.1150485`. [Online]. Available: `https://doi.org/10.1145/1150402.1150485`.

[MPL+14] S. A. Macskassy, C. Perlich, J. Leskovec, W. Wang, and R. Ghani, Eds., *The 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD '14, New York, NY, USA - August 24 - 27, 2014*, ACM, 2014, ISBN: 978-1-4503-2956-9. [Online]. Available: `http://dl.acm.org/citation.cfm?id=2623330`.

[MR10] O. Maimon and L. Rokach, Eds., *Data Mining and Knowledge Discovery Handbook, 2nd ed.* Springer, 2010, ISBN: 978-0-387-09822-7. [Online]. Available: `http://www.springerlink.com/content/978-0-387-09822-7`.

[MR15] P. Moradi and M. Rostami, "A graph theoretic approach for unsupervised feature selection", *Eng. Appl. of AI*, vol. 44, pp. 33–45, 2015. DOI: `10.1016/j.engappai.2015.05.005`. [Online]. Available: `https://doi.org/10.1016/j.engappai.2015.05.005`.

[MSC18] S. Marchand-Maillet, Y. N. Silva, and E. Chávez, Eds., *Similarity Search and Applications - 11th International Conference, SISAP 2018, Lima, Peru, October 7-9, 2018, Proceedings*, vol. 11223, Lecture Notes in Computer Science, Springer, 2018, ISBN: 978-3-030-02223-5. DOI: `10.1007/978-3-030-02224-2`. [Online]. Available: `https://doi.org/10.1007/978-3-030-02224-2`.

[MU04] K. Makino and T. Uno, "New algorithms for enumerating all maximal cliques", *in Algorithm Theory - SWAT 2004, 9th Scandinavian Workshop on Algorithm Theory, Humlebaek, Denmark, July 8-10, 2004, Proceedings*, T. Hagerup and J. Katajainen, Eds., ser. Lecture Notes in Computer Science, vol. 3111, Springer, 2004, pp. 260–272, ISBN: 3-540-22339-8. DOI: `10.1007/978-3-540-27810-8\_23`. [Online]. Available: `https://doi.org/10.1007/978-3-540-27810-8%5C_23`.

[Mül07a] M. Müller, "Dynamic time warping", *in Information Retrieval for Music and Motion.* Berlin, Heidelberg: Springer Berlin Heidelberg, 2007, pp. 69–84, ISBN: 978-3-540-74048-3. DOI: `10.1007/978-3-540-74048-3_4`. [Online]. Available: `https://doi.org/10.1007/978-3-540-74048-3_4`.

[Mül07b] ——, *Information Retrieval for Music and Motion.* Springer, 2007. DOI: `10.1007/978-3-540-74048-3`. [Online]. Available: `https://doi.org/10.1007/978-3-540-74048-3`.

[NÖK+04] M. A. Nascimento, M. T. Özsu, D. Kossmann, R. J. Miller, J. A. Blakeley, and K. B. Schiefer, Eds., *(e)Proceedings of the Thirtieth International Conference on Very Large Data Bases, VLDB 2004, Toronto, Canada, August 31 - September 3 2004*, Morgan Kaufmann, 2004, ISBN: 0-12-088469-0. [Online]. Available: `https://www.sciencedirect.com/book/9780120884698/`.

[NR07] V. Niennattrakul and C. A. Ratanamahatana, "Inaccuracies of shape averaging method using dynamic time warping for time series data", *in Computational Science - ICCS 2007, 7th International Conference Beijing, China, May 27-30, 2007, Proceedings, Part I*, Y. Shi, G. D. van Albada, J. J. Dongarra, and P. M. A. Sloot, Eds., ser. Lecture Notes in Computer Science, vol. 4487, Springer, 2007, pp. 513–520, ISBN: 978-3-540-72583-1. DOI: `10.1007/978-3-540-72584-8\_68`. [Online]. Available: `https://doi.org/10.1007/978-3-540-72584-8%5C_68`.

[NR09] ——, "Learning DTW global constraint for time series classification", *CoRR*, vol. abs/0903.0041, 2009.

[PAS82] K. K. Paliwal, A. Agarwal, and S. S. Sinha, "A modification over sakoe and chiba's dynamic time warping algorithm for isolated word recognition", *Signal Processing*, vol. 4, no. 4, pp. 329–333, 1982.

[Pec97] J. Peckham, Ed., *SIGMOD 1997, Proceedings ACM SIGMOD International Conference on Management of Data, May 13-15, 1997, Tucson, Arizona, USA*, ACM Press, 1997.

[PH74] T. Pavlidis and S. L. Horowitz, "Segmentation of plane curves", *IEEE Trans. Computers*, vol. 23, no. 8, pp. 860–870, 1974. DOI: 10.1109/T-C.1974.224041. [Online]. Available: https://doi.org/10.1109/T-C.1974.224041.

[PKG11] F. Petitjean, A. Ketterlin, and P. Gançarski, "A global averaging method for dynamic time warping, with applications to clustering", *Pattern Recognition*, vol. 44, no. 3, pp. 678–693, 2011. DOI: 10.1016/j.patcog.2010.09.013. [Online]. Available: https://doi.org/10.1016/j.patcog.2010.09.013.

[PKPP15] O. P. Patri, R. Kannan, A. V. Panangadan, and V. K. Prasanna, "Multivariate time series classification using inter-leaved shapelets", *in NIPS 2015 Time Series Workshop*, 2015.

[PT17] D. Precup and Y. W. Teh, Eds., *Proceedings of the 34th International Conference on Machine Learning, ICML 2017, Sydney, NSW, Australia, 6-11 August 2017*, vol. 70, Proceedings of Machine Learning Research, PMLR, 2017. [Online]. Available: http://proceedings.mlr.press/v70/.

[PW06] D. B. Percival and A. T. Walden, *Wavelet methods for time series analysis.* Cambridge university press, 2006, vol. 4.

[Qui86] J. R. Quinlan, "Induction of decision trees", *Machine Learning*, vol. 1, no. 1, pp. 81–106, 1986. DOI: 10.1023/A:1022643204877. [Online]. Available: https://doi.org/10.1023/A:1022643204877.

[RCM+12] T. Rakthanmanon, B. J. L. Campana, A. Mueen, G. E. A. P. A. Batista, M. B. Westover, Q. Zhu, J. Zakaria, and E. J. Keogh, "Searching and mining trillions of time series subsequences under dynamic time warping", *in The 18th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD '12, Beijing, China, August 12-16, 2012*, Q. Yang, D. Agarwal, and J. Pei, Eds., ACM, 2012, pp. 262–270, ISBN: 978-1-4503-1462-6. DOI: 10.1145/2339530.2339576. [Online]. Available: https://doi.org/10.1145/2339530.2339576.

[RK04a] C. Ratanamahatana and E. Keogh, "Everything you know about dynamic time warping is wrong", *Third WMSD*, pp. 22–25, 2004. DOI: 10.1097/01.CCM.0000279204.24648.44.

[RK04b] C. (Ratanamahatana and E. J. Keogh, "Making time-series classification more accurate using learned constraints", *in Proceedings of the Fourth SIAM International Conference on Data Mining, Lake Buena Vista, Florida, USA, April 22-24, 2004*, M. W. Berry, U. Dayal, C. Kamath, and D. B. Skillicorn, Eds., SIAM, 2004, pp. 11–22, ISBN: 978-0-89871-568-2. DOI: 10.1137/1.9781611972740.2. [Online]. Available: https://doi.org/10.1137/1.9781611972740.2.

[RK05] ——, "Three myths about dynamic time warping data mining", *in Proceedings of the 2005 SIAM International Conference on Data Mining, SDM 2005, Newport Beach, CA, USA, April 21-23, 2005*, H. Kargupta, J. Srivastava, C. Kamath, and A. Goodman, Eds., SIAM, 2005, pp. 506–510, ISBN: 978-0-89871-593-4. DOI: 10.1137/1.9781611972757.50. [Online]. Available: https://doi.org/10.1137/1.9781611972757.50.

[RKTY10] B. Rao, B. Krishnapuram, A. Tomkins, and Q. Yang, Eds., *Proceedings of the 16th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Washington, DC, USA, July 25-28, 2010*, ACM, 2010, ISBN: 978-1-4503-0055-1.

[RLG+10] C. A. Ratanamahatana, J. Lin, D. Gunopulos, E. J. Keogh, M. Vlachos, and G. Das, "Mining time series data", *in Data Mining and Knowledge Discovery Handbook, 2nd ed.* O. Maimon and L. Rokach, Eds., Springer, 2010, pp. 1049–1077, ISBN: 978-0-387-09822-7. DOI: `10.1007/978-0-387-09823-4\_56`. [Online]. Available: `https://doi.org/10.1007/978-0-387-09823-4%5C_56`.

[RM02] T. M. Rath and R. Manmatha, "Lower-bounding of dynamic time warping distances for multivariate time series", *University of Massachusetts Amherst, Tech. Rep. MM-40*, 2002.

[RSBP00] R. Ramakrishnan, S. J. Stolfo, R. J. Bayardo, and I. Parsa, Eds., *Proceedings of the sixth ACM SIGKDD international conference on Knowledge discovery and data mining, Boston, MA, USA, August 20-23, 2000*, ACM, 2000, ISBN: 1-58113-233-6. [Online]. Available: `http://dl.acm.org/citation.cfm?id=347090`.

[RW08] C. A. Ratanamahatana and D. Wanichsan, "Stopping criterion selection for efficient semi-supervised time series classification", *in Software Engineering, Artificial Intelligence, Networking and Parallel/Distributed Computing*, ser. Studies in Computational Intelligence, R. Y. Lee, Ed., vol. 149, Springer, 2008, pp. 1–14, ISBN: 978-3-540-70559-8. DOI: `10.1007/978-3-540-70560-4\_1`. [Online]. Available: `https://doi.org/10.1007/978-3-540-70560-4%5C_1`.

[SBK16] D. F. Silva, G. E. A. P. A. Batista, and E. J. Keogh, "Prefix and suffix invariant dynamic time warping", *in IEEE 16th International Conference on Data Mining, ICDM 2016, December 12-15, 2016, Barcelona, Spain*, F. Bonchi, J. Domingo-Ferrer, R. A. Baeza-Yates, Z. Zhou, and X. Wu, Eds., IEEE Computer Society, 2016, pp. 1209–1214. DOI: `10.1109/ICDM.2016.0161`. [Online]. Available: `https://doi.org/10.1109/ICDM.2016.0161`.

[SC07] S. Salvador and P. Chan, "Toward accurate dynamic time warping in linear time and space", *Intell. Data Anal.*, vol. 11, no. 5, pp. 561–580, 2007. [Online]. Available: `http://content.iospress.com/articles/intelligent-data-analysis/ida00303`.

[SC78] H. Sakoe and S. Chiba, "Dynamic programming algorithm optimization for spoken word recognition", *IEEE Trans. Signal Processing*, vol. 26, no. 1, pp. 43–49, 1978, ISSN: 0096-3518. DOI: `10.1109/TASSP.1978.1163055`.

[SGGS06] C. Saunders, M. Grobelnik, S. R. Gunn, and J. Shawe-Taylor, Eds., *Subspace, Latent Structure and Feature Selection, Statistical and Optimization, Perspectives Workshop, SLSFS 2005, Bohinj, Slovenia, February 23-25, 2005, Revised Selected Papers*, vol. 3940, Lecture Notes in Computer Science, Springer, 2006, ISBN: 3-540-34137-4. DOI: `10.1007/11752790`. [Online]. Available: `https://doi.org/10.1007/11752790`.

[SH16] K. Saeed and W. Homenda, Eds., *Computer Information Systems and Industrial Management - 15th IFIP TC8 International Conference, CISIM 2016, Vilnius, Lithuania, September 14-16, 2016, Proceedings*, vol. 9842, Lecture Notes in Computer Science, Springer, 2016, ISBN: 978-3-319-45377-4. DOI: `10.1007/978-3-319-45378-1`. [Online]. Available: `https://doi.org/10.1007/978-3-319-45378-1`.

[Sha01] C. E. Shannon, "A mathematical theory of communication", *Mobile Computing and Communications Review*, vol. 5, no. 1, pp. 3–55, 2001. DOI: `10.1145/584091.584093`. [Online]. Available: `https://doi.org/10.1145/584091.584093`.

[Shi10] J. Shieh, "Time series retrieval: indexing and mining large datasets", PhD thesis, University of California, Riverside, USA, 2010. [Online]. Available: `http://www.escholarship.org/uc/item/7gk6t3b8`.

[SHJ+17] M. Shokoohi-Yekta, B. Hu, H. Jin, J. Wang, and E. J. Keogh, "Generalizing DTW to the multi-dimensional case requires an adaptive approach", *Data Min. Knowl. Discov.*, vol. 31, no. 1, pp. 1–31, 2017. DOI: `10.1007/s10618-016-0455-0`. [Online]. Available: `https://doi.org/10.1007/s10618-016-0455-0`.

[SIL07] Y. Saeys, I. Inza, and P. Larrañaga, "A review of feature selection techniques in bioinformatics", *Bioinformatics*, vol. 23, no. 19, pp. 2507–2517, 2007. DOI: 10.1093/bioinformatics/btm344. [Online]. Available: https://doi.org/10.1093/bioinformatics/btm344.

[SJdPG15] R. C. Sperandio, Z. K. G. P. Jr., H. B. de Paula, and S. J. F. Guimarães, "An efficient access method for multimodal video retrieval", *Multimedia Tools Appl.*, vol. 74, no. 4, pp. 1357–1375, 2015. DOI: 10.1007/s11042-014-1917-2. [Online]. Available: https://doi.org/10.1007/s11042-014-1917-2.

[SK08] J. Shieh and E. J. Keogh, "*i*sax: indexing and mining terabyte sized time series", *in Proceedings of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Las Vegas, Nevada, USA, August 24-27, 2008*, Y. Li, B. Liu, and S. Sarawagi, Eds., ACM, 2008, pp. 623–631, ISBN: 978-1-60558-193-4. DOI: 10.1145/1401890.1401966. [Online]. Available: https://doi.org/10.1145/1401890.1401966.

[SMAT18] R. C. Sperandio, S. Malinowski, L. Amsaleg, and R. Tavenard, "Time series retrieval using dtw-preserving shapelets", *in Similarity Search and Applications - 11th International Conference, SISAP 2018, Lima, Peru, October 7-9, 2018, Proceedings*, S. Marchand-Maillet, Y. N. Silva, and E. Chávez, Eds., ser. Lecture Notes in Computer Science, vol. 11223, Springer, 2018, pp. 257–270, ISBN: 978-3-030-02223-5. DOI: 10.1007/978-3-030-02224-2_20. [Online]. Available: https://doi.org/10.1007/978-3-030-02224-2%5C_20.

[Smy96] P. Smyth, "Clustering sequences with hidden markov models", *in Advances in Neural Information Processing Systems 9, NIPS, Denver, CO, USA, December 2-5, 1996*, M. Mozer, M. I. Jordan, and T. Petsche, Eds., MIT Press, 1996, pp. 648–654. [Online]. Available: http://papers.nips.cc/paper/1217-clustering-sequences-with-hidden-markov-models.

[SvADS07] Y. Shi, G. D. van Albada, J. J. Dongarra, and P. M. A. Sloot, Eds., *Computational Science - ICCS 2007, 7th International Conference Beijing, China, May 27-30, 2007, Proceedings, Part I*, vol. 4487, Lecture Notes in Computer Science, Springer, 2007, ISBN: 978-3-540-72583-1. DOI: 10.1007/978-3-540-72584-8. [Online]. Available: https://doi.org/10.1007/978-3-540-72584-8.

[SvASD05] V. S. Sunderam, G. D. van Albada, P. M. A. Sloot, and J. J. Dongarra, Eds., *Computational Science - ICCS 2005, 5th International Conference, Atlanta, GA, USA, May 22-25, 2005, Proceedings, Part III*, vol. 3516, Lecture Notes in Computer Science, Springer, 2005, ISBN: 3-540-26044-7. DOI: 10.1007/b136575. [Online]. Available: https://doi.org/10.1007/b136575.

[SW16] D. Schuurmans and M. P. Wellman, Eds., *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence, February 12-17, 2016, Phoenix, Arizona, USA*, AAAI Press, 2016, ISBN: 978-1-57735-760-5. [Online]. Available: http://www.aaai.org/Library/AAAI/aaai16contents.php.

[SW94] R. T. Snodgrass and M. Winslett, Eds., *Proceedings of the 1994 ACM SIGMOD International Conference on Management of Data, Minneapolis, Minnesota, USA, May 24-27, 1994*, ACM Press, 1994.

[SWK15] M. Shokoohi-Yekta, J. Wang, and E. J. Keogh, "On the non-trivial generalization of dynamic time warping to the multi-dimensional case", *in Proceedings of the 2015 SIAM International Conference on Data Mining, Vancouver, BC, Canada, April 30 - May 2, 2015*, S. Venkatasubramanian and J. Ye, Eds., SIAM, 2015, pp. 289–297, ISBN: 978-1-61197-401-0. DOI: 10.1137/1.9781611974010.33. [Online]. Available: https://doi.org/10.1137/1.9781611974010.33.

[SZ04] D. E. Shasha and Y. Zhu, *High Performance Discovery in Time Series - Techniques and Case Studies*, ser. Monographs in Computer Science. Springer, 2004, ISBN: 978-1-4419-1842-0. DOI: 10.1007/978-1-4757-4046-2. [Online]. Available: https://doi.org/10.1007/978-1-4757-4046-2.

[Tav11] R. Tavenard, "Indexation de séquences de descripteurs", In French, PhD thesis, Université Rennes 1, Rennes, France, Nov. 2011.

[Tav17] ——, *Tslearn: a machine learning toolkit dedicated to time-series data*, `https://github.com/rtavenar/tslearn`, 2017.

[TPKC10] S. Theodoridis, A. Pikrakis, K. Koutroumbas, and D. Cavouras, *Introduction to pattern recognition: a matlab approach.* Academic Press, 2010.

[TTT06] E. Tomita, A. Tanaka, and H. Takahashi, "The worst-case time complexity for generating all maximal cliques and computational experiments", *Theor. Comput. Sci.*, vol. 363, no. 1, pp. 28–42, 2006. DOI: `10.1016/j.tcs.2006.06.015`. [Online]. Available: `https://doi.org/10.1016/j.tcs.2006.06.015`.

[TWP17] C. W. Tan, G. I. Webb, and F. Petitjean, "Indexing and classifying gigabytes of time series under time warping", *in Proceedings of the 2017 SIAM International Conference on Data Mining, Houston, Texas, USA, April 27-29, 2017.*, N. Chawla and W. Wang, Eds., SIAM, 2017, pp. 282–290, ISBN: 978-1-61197-497-3. DOI: `10.1137/1.9781611974973.32`. [Online]. Available: `https://doi.org/10.1137/1.9781611974973.32`.

[UB98] S. D. Urban and E. Bertino, Eds., *Proceedings of the Fourteenth International Conference on Data Engineering, Orlando, Florida, USA, February 23-27, 1998*, IEEE Computer Society, 1998, ISBN: 0-8186-8289-2. [Online]. Available: `http://ieeexplore.ieee.org/xpl/mostRecentIssue.jsp?punumber=5268`.

[UBK15] L. Ulanova, N. Begum, and E. J. Keogh, "Scalable clustering of time series with u-shapelets", *in Proceedings of the 2015 SIAM International Conference on Data Mining, Vancouver, BC, Canada, April 30 - May 2, 2015*, S. Venkatasubramanian and J. Ye, Eds., SIAM, 2015, pp. 900–908, ISBN: 978-1-61197-401-0. DOI: `10.1137/1.9781611974010.101`. [Online]. Available: `https://doi.org/10.1137/1.9781611974010.101`.

[VY15] S. Venkatasubramanian and J. Ye, Eds., *Proceedings of the 2015 SIAM International Conference on Data Mining, Vancouver, BC, Canada, April 30 - May 2, 2015*, SIAM, 2015, ISBN: 978-1-61197-401-0. DOI: `10.1137/1.9781611974010`. [Online]. Available: `https://doi.org/10.1137/1.9781611974010`.

[WGS15] M. Wistuba, J. Grabocka, and L. Schmidt-Thieme, "Ultra-fast shapelets for time series classification", *CoRR*, vol. abs/1503.05018, 2015. arXiv: `1503.05018`. [Online]. Available: `http://arxiv.org/abs/1503.05018`.

[WMD+13] X. Wang, A. Mueen, H. Ding, G. Trajcevski, P. Scheuermann, and E. J. Keogh, "Experimental comparison of representation methods and distance measures for time series data", *Data Min. Knowl. Discov.*, vol. 26, no. 2, pp. 275–309, 2013. DOI: `10.1007/s10618-012-0250-5`. [Online]. Available: `https://doi.org/10.1007/s10618-012-0250-5`.

[WSH05] X. Wang, K. A. Smith, and R. J. Hyndman, "Dimension reduction for clustering time series using global characteristics", *in Computational Science - ICCS 2005, 5th International Conference, Atlanta, GA, USA, May 22-25, 2005, Proceedings, Part III*, V. S. Sunderam, G. D. van Albada, P. M. A. Sloot, and J. J. Dongarra, Eds., ser. Lecture Notes in Computer Science, vol. 3516, Springer, 2005, pp. 792–795, ISBN: 3-540-26044-7. DOI: `10.1007/11428862\_108`. [Online]. Available: `https://doi.org/10.1007/11428862%5C_108`.

[WW00] C. Wang and X. S. Wang, "Supporting content-based searches on time series via approximation", *in Proceedings of the 12th International Conference on Scientific and Statistical Database Management, Berlin, Germany, July 26-28, 2000*, O. Günther and H. Lenz, Eds., IEEE Computer Society, 2000, pp. 69–81, ISBN: 0-7695-0686-0. DOI: `10.1109/SSDM.2000.869779`. [Online]. Available: `https://doi.org/10.1109/SSDM.2000.869779`.

[XKT+13] H. Xiong, G. Karypis, B. M. Thuraisingham, D. J. Cook, and X. Wu, Eds., *2013 IEEE 13th International Conference on Data Mining, Dallas, TX, USA, December 7-10, 2013*, IEEE Computer Society, 2013, ISBN: 978-0-7695-5108-1. [Online]. Available: http://ieeexplore.ieee.org/xpl/mostRecentIssue.jsp?punumber=6724379.

[XY02] Y. Xiong and D. Yeung, "Mixtures of ARMA models for model-based time series clustering", *in Proceedings of the 2002 IEEE International Conference on Data Mining (ICDM 2002), 9-12 December 2002, Maebashi City, Japan*, IEEE Computer Society, 2002, pp. 717–720, ISBN: 0-7695-1754-4. DOI: 10.1109/ICDM.2002.1184037. [Online]. Available: https://doi.org/10.1109/ICDM.2002.1184037.

[YAMS17] D. E. Yagoubi, R. Akbarinia, F. Masseglia, and D. E. Shasha, "Radiussketch: massively distributed indexing of time series", *in 2017 IEEE International Conference on Data Science and Advanced Analytics, DSAA 2017, Tokyo, Japan, October 19-21, 2017*, IEEE, 2017, pp. 262–271, ISBN: 978-1-5090-5004-8. DOI: 10.1109/DSAA.2017.49. [Online]. Available: https://doi.org/10.1109/DSAA.2017.49.

[YAP12] Q. Yang, D. Agarwal, and J. Pei, Eds., *The 18th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD '12, Beijing, China, August 12-16, 2012*, ACM, 2012, ISBN: 978-1-4503-1462-6. [Online]. Available: http://dl.acm.org/citation.cfm?id=2339530.

[YCB12] H. Yin, J. A. F. Costa, and G. D. A. Barreto, Eds., *Intelligent Data Engineering and Automated Learning - IDEAL 2012 - 13th International Conference, Natal, Brazil, August 29-31, 2012. Proceedings*, vol. 7435, Lecture Notes in Computer Science, Springer, 2012, ISBN: 978-3-642-32638-7. DOI: 10.1007/978-3-642-32639-4. [Online]. Available: https://doi.org/10.1007/978-3-642-32639-4.

[YF00] B. Yi and C. Faloutsos, "Fast time sequence indexing for arbitrary lp norms", *in VLDB 2000, Proceedings of 26th International Conference on Very Large Data Bases, September 10-14, 2000, Cairo, Egypt*, A. El Abbadi, M. L. Brodie, S. Chakravarthy, U. Dayal, N. Kamel, G. Schlageter, and K. Whang, Eds., Morgan Kaufmann, 2000, pp. 385–394, ISBN: 1-55860-715-3. [Online]. Available: http://www.vldb.org/conf/2000/P385.pdf.

[YJF98] B. Yi, H. V. Jagadish, and C. Faloutsos, "Efficient retrieval of similar time sequences under time warping", *in Proceedings of the Fourteenth International Conference on Data Engineering, Orlando, Florida, USA, February 23-27, 1998*, S. D. Urban and E. Bertino, Eds., IEEE Computer Society, 1998, pp. 201–208, ISBN: 0-8186-8289-2. DOI: 10.1109/ICDE.1998.655778. [Online]. Available: https://doi.org/10.1109/ICDE.1998.655778.

[YK09] L. Ye and E. J. Keogh, "Time series shapelets: a new primitive for data mining", *in Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Paris, France, June 28 - July 1, 2009*, J. F. E. IV, F. Fogelman-Soulié, P. A. Flach, and M. J. Zaki, Eds., ACM, 2009, pp. 947–956, ISBN: 978-1-60558-495-9. DOI: 10.1145/1557019.1557122. [Online]. Available: https://doi.org/10.1145/1557019.1557122.

[YK11] ——, "Time series shapelets: a novel technique that allows accurate, interpretable and fast classification", *Data Min. Knowl. Discov.*, vol. 22, no. 1-2, pp. 149–182, 2011. DOI: 10.1007/s10618-010-0179-5. [Online]. Available: https://doi.org/10.1007/s10618-010-0179-5.

[YS04] K. Yang and C. Shahabi, "A pca-based similarity measure for multivariate time series", *in Proceedings of the Second ACM International Workshop on Multimedia Databases, ACM-MMDB 2004, Washington, DC, USA, November 13, 2004*, S. Chen and M. Shyu, Eds., ACM, 2004, pp. 65–74, ISBN: 1-58113-975-6. DOI: 10.1145/1032604.1032616. [Online]. Available: https://doi.org/10.1145/1032604.1032616.

[YZU+18] C. M. Yeh, Y. Zhu, L. Ulanova, N. Begum, Y. Ding, H. A. Dau, Z. Zimmerman, D. F. Silva, A. Mueen, and E. J. Keogh, "Time series joins, motifs, discords and shapelets: a unifying view that exploits the matrix profile", *Data Min. Knowl. Discov.*, vol. 32, no. 1, pp. 83–123, 2018. DOI: 10.1007/s10618-017-0519-9. [Online]. Available: https://doi.org/10.1007/s10618-017-0519-9.

[ZAB+05] Y. Zhang, F. N. Abu-Khzam, N. E. Baldwin, E. J. Chesler, M. A. Langston, and N. F. Samatova, "Genome-scale computational approaches to memory-intensive applications in systems biology", *in Proceedings of the ACM/IEEE SC2005 Conference on High Performance Networking and Computing, November 12-18, 2005, Seattle, WA, USA, CD-Rom*, IEEE Computer Society, 2005, p. 12, ISBN: 1-59593-061-2. DOI: 10.1109/SC.2005.29. [Online]. Available: https://doi.org/10.1109/SC.2005.29.

[ZL07] Z. Zhao and H. Liu, "Spectral feature selection for supervised and unsupervised learning", *in Machine Learning, Proceedings of the Twenty-Fourth International Conference (ICML 2007), Corvallis, Oregon, USA, June 20-24, 2007*, Z. Ghahramani, Ed., ser. ACM International Conference Proceeding Series, vol. 227, ACM, 2007, pp. 1151–1157, ISBN: 978-1-59593-793-3. DOI: 10.1145/1273496.1273641. [Online]. Available: https://doi.org/10.1145/1273496.1273641.

[ZLSG00] C. Zaniolo, P. C. Lockemann, M. H. Scholl, and T. Grust, Eds., *Advances in Database Technology - EDBT 2000, 7th International Conference on Extending Database Technology, Konstanz, Germany, March 27-31, 2000, Proceedings*, vol. 1777, Lecture Notes in Computer Science, Springer, 2000, ISBN: 3-540-67227-3. DOI: 10.1007/3-540-46439-5. [Online]. Available: https://doi.org/10.1007/3-540-46439-5.

[ZMK12] J. Zakaria, A. Mueen, and E. J. Keogh, "Clustering time series using unsupervised-shapelets", *in 12th IEEE International Conference on Data Mining, ICDM 2012, Brussels, Belgium, December 10-13, 2012*, M. J. Zaki, A. Siebes, J. X. Yu, B. Goethals, G. I. Webb, and X. Wu, Eds., IEEE Computer Society, 2012, pp. 785–794, ISBN: 978-1-4673-4649-8. DOI: 10.1109/ICDM.2012.26. [Online]. Available: https://doi.org/10.1109/ICDM.2012.26.

[ZMKY16] J. Zakaria, A. Mueen, E. J. Keogh, and N. E. Young, "Accelerating the discovery of unsupervised-shapelets", *Data Min. Knowl. Discov.*, vol. 30, no. 1, pp. 243–281, 2016. DOI: 10.1007/s10618-015-0411-4. [Online]. Available: https://doi.org/10.1007/s10618-015-0411-4.

[ZSY+12] M. J. Zaki, A. Siebes, J. X. Yu, B. Goethals, G. I. Webb, and X. Wu, Eds., *12th IEEE International Conference on Data Mining, ICDM 2012, Brussels, Belgium, December 10-13, 2012*, IEEE Computer Society, 2012, ISBN: 978-1-4673-4649-8. [Online]. Available: http://ieeexplore.ieee.org/xpl/mostRecentIssue.jsp?punumber=6412852.

Titre : Recherche de Séries Temporelles à l'aide de *DTW-Preserving Shapelets*

Mots clés : séries temporelles, apprentissage automatique, représentation, sous-séquences

Résumé : L'établissement de la similarité entre séries temporelles est au cœur de nombreuses tâches d'analyse de données. Les mesures permettant d'établir des similitudes entre les séries temporelles sont spécifiques en ce sens qu'elles doivent pouvoir prendre en compte les différences entre les valeurs constituant la série, ainsi que les distorsions selon l'axe du temps. La mesure de similarité la plus répandue est la mesure *Dynamic Time Warping (DTW)*. Cependant, son calcul est coûteux et son application à des séries temporelles nombreuses et/ou très longues est difficile en pratique. Malgré de nombreuses contributions visant l'accélération de la DTW, réussir son passage à l'échelle de la DTW reste une difficulté majeure. Le travail présenté dans cette thèse s'appuie sur l'idée de transformer les séries temporelles à l'aide de *shapelets*. Il montre comment des *shapelets* préservant les mesures DTW peuvent être utilisées dans le contexte spécifique de la recherches de séries temporelles similaires à une série utilisée comme requête, et cela dans un contexte grande échelle. Il s'agit de plonger les séries temporelles dans un espace euclidien construit de telle manière que les distances entre les séries selon la métrique DTW s'y trouvent préservées. Ce manuscrit apporte des contributions majeures : (1) il explique comment les *shapelets* préservant la DTW peuvent être utilisées dans le contexte spécifique de la recherche de séries temporelles similaires ; (2) il propose des stratégies de sélection de ces *shapelets* pour faire face à l'échelle, c'est-à-dire pour traiter une collection extrêmement vaste de séries temporelles ; (3) il explique en détail comment gérer les séries temporelles univariées et multivariées, couvrant ainsi tout le spectre des problèmes de recherches et facilitant la moise au point d'applications très diverses. Le cœur de la contribution présentée dans ce manuscrit permet de compenser facilement la complexité du processus de plongement par un jeu sur la précision de la recherche. Des expérimentations utilisant les jeux de données UCR et UEA démontrent l'amélioration considérable des performances par rapport aux techniques de pointe.

Title: Time Series Retrieval Using DTW-Preserving Shapelets

Keywords: time-series, shapelets, machine-learning, paa, lower-bounding, retrieval

Abstract: Establishing the similarity of time series is at the core of many data mining tasks such as time series classification, time series clustering, time series retrieval, among others. Metrics to establish similarities between time series are specific in the sense that they must be able to take into account the differences in the values making the series as well as distortions along the timelines. The most popular similarity metric is the Dynamic Time Warping (DTW) measure. However, it is costly to compute, and using it against numerous and/or very long time series is difficult in practice. There has been numerous attempts to accelerate the DTW, yet, scaling DTW remains a major difficulty. An elegant research direction proposes to change the representation of time series such that it is much cheaper to establish similarities. This typically relies on an embedding process where vectorial representations of time series are constructed, allowing then to estimate their similarity using e.g. L_2 distances, much faster to compute than DTW. Naturally, the quality of this representation largely depends on the embedding process, and the family of contributions relying on the concept of shapelets prove to work particularly well. Shapelets, and the transform operation materializing the embedding process, were originally proposed for time series classification. Shapelets are independent subsequences extracted or learned from time series to form discriminatory features. Shapelets are used to transform time series in high dimensional (Euclidean) vectors. Recently, it was proposed to embed time series into an Euclidean space such that the distance in this embedded space well approximates the true DTW. This contribution targets time series clustering. The work presented in this Ph.D. manuscript builds on the idea of transforming time series using shapelets. It shows how shapelets that preserve DTW measures can be used in the specific context of large scale time series retrieval. This manuscript is making major contributions: (1) it explains how DTW-preserving shapelets can be used in the specific context of time series retrieval; (2) it proposes some shapelet selection strategies in order to cope with scale, that is, in order to deal with extremely large collection of time series; (3) it details how to handle both univariate and multivariate time series, hence covering the whole spectrum of time series retrieval problems. The core of the contribution presented in this manuscript allows to easily trade-off the complexity of the transformation against the accuracy of the retrieval. Experiments using the UCR and the UEA datasets demonstrate the vast performance improvements compared to state of the art techniques.