Auto-encodeur multi-tâches pour le calcul de moyennes de séries temporelles

Tsegamlak Terefe[*,**], Maxime Devanne[*] Jonathan Weber[*],
Dereje Hailemariam[**], Germain Forestier[*]

[*]IRIMAS, Université Haute Alsace, Mulhouse, France
prénom.nom@uha.fr,
[**]Addis Ababa Institute of Technology, Addis Ababa University
prénom.nom@aait.edu.et

Résumé. L'estimation de moyennes de séries temporelles a beaucoup été étudiée au cours des dernières décennies. Les distorsions temporelles entre les séries rendent cette tâche très difficile. De précédents travaux ont principalement abordé ce défi en utilisant des algorithmes d'alignement comme Dynamic Time Warping (DTW). Cependant, la complexité de calcul quadratique de DTW et son incapacité à aligner plus de deux séries simultanément compliquent fortement l'estimation de moyennes. Dans cet article, nous suivons une approche différente en considérant le problème du calcul de moyennes comme un problème génératif. Ainsi, nous proposons un auto-encodeur multi-tâches pour extraire des caractéristiques latentes à partir de séries temporelles appartenant à la même classe mais avec des distorsions temporelles variables. Nous considérons ensuite la moyenne arithmétique des caractéristiques latentes comme une estimation de la moyenne latente. De plus, nous projetons ces estimations dans le domaine temporel afin de vérifier leur cohérence. Nous avons évalué l'approche proposée à travers une classification basée sur le plus proche centroïde à partir de 85 jeux de données issus de l'archive UCR. Les résultats expérimentaux montrent que l'approche proposée obtient des performances de classification dans l'espace latent compétitives avec les approches de l'état de l'art. Cela montre que l'espace latent appris est pertinent pour calculer des moyennes de séries temporelles. Aussi, la projection des moyennes dans le domaine temporel offre des résultats supérieurs par rapport aux moyennes arithmétiques directement calculées dans ce domaine.

1 Introduction

Aujourd'hui, il existe de nombreux jeux de données de séries temporelles, issus de domaines différents comme l'acoustique, la finance, le trafic internet, les images satellitaires, etc. (Wei, 2006). Une définition formelle de ces jeux de données considère une série temporelle comme une séquence ordonnée selon le temps $X : \{x_1, x_2, ..., x_T\}$, définie par un ensemble d'attributs λ, $x_i \in \lambda$ (Jain et Schultz, 2016). Dans cette étude, nous nous intéressons aux séries

temporelles uni-variées avec $\forall x_i \in \lambda : x_i \in \mathbb{R}$. En général, les techniques de fouilles de données doivent être adaptées aux spécificités des données analysées (Bagnall et al., 2012). Dans le cas de séries temporelles, l'aspect majeur qui doit être considéré concerne les désalignements ou distorsions temporelles. Ces distorsions temporelles apparaissent au sein des jeux de données pour différentes raisons comme la différence de fréquences d'acquisition des équipements de mesure (Bagnall et al., 2017). Par exemple, si nous prenons le cas de la consommation d'énergie de différents appareils ménagers dans plusieurs résidences, nous ne pouvons pas attendre que les mesures soient parfaitement alignées car chacune d'elles dépend des réglages des utilisateurs. Ainsi, si l'on souhaite identifier les appareils ménagers par l'intermédiaire de modèles représentatifs comme les moyennes, les techniques classiques de calcul de similarité comme la distance euclidienne ne vont pas être performantes à cause des distorsions (Bagnall et al., 2017). Ce problème est illustré en Figure 1 où sont présentées les séries du jeu de données BeetleFly représentant les contours extraits d'images de coléoptères et de mouches. La variation de taille et d'orientation des insectes dans l'image résulte en des séries temporelles non alignées. Le calcul de la moyenne arithmétique des séries correspond à la série en bleu dans la Figure 1. Comme mis en valeur dans les zones rouges, une telle moyenne n'est pas cohérente et n'est pas capable de capturer la forme générale des séries du jeu de données. Pour faire face

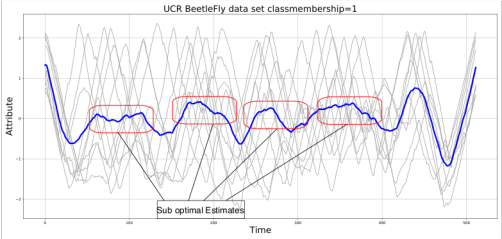


FIG. 1: Impact de la distorsion temporelle sur la moyenne arithmétique.

à ce problème, plusieurs techniques emploient des algorithmes d'alignement temporel comme première étape (Gupta et al., 1996; Niennattrakul et Ratanamahatana, 2009; Petitjean et Gançarski, 2012; Cuturi et Blondel, 2017). Ces algorithmes minimisent le décalage de phase entre deux ou plusieurs séries temporelles mais présentent souvent des inconvénients. Par exemple, l'algorithme Dynamic Time Warping (DTW), très utilisé pour l'alignement temporel, n'est pas adapté pour l'alignement de plusieurs séries simultanément (Petitjean et Gançarski, 2012). Cela complique fortement le calcul de moyenne d'un ensemble de séries temporelles (Brill et al., 2019).

Nous pouvons définir le problème de calcul de moyenne comme : soit un ensemble de séries temporelles $X = \{X_1, X_2, ..., X_T\}$ où $X_i \in \mathbb{R}^m$, générer une série $\mu \in \mathbb{R}^n$ où $n \geq m$. Nous générons μ afin de minimiser :

$$D = \frac{1}{T} \sum_{i=1}^{T} d(\mu, X_i), \tag{1}$$

où d est une distance, DTW dans la plupart des cas. Les approches heuristiques existantes transforment l'ensemble X en une représentation alignée de celui-ci. Ensuite, la moyenne de l'ensemble est estimée en calculant la moyenne arithmétique. Dans ce contexte, si DTW est

utilisé pour l'alignement, (1) devient une fonction de coût non convexe et non lisse (Schultz et Jain, 2018). Cela a pour effet majeur d'empêcher l'utilisation de DTW comme fonction de coût au sein d'un réseau de neurones.

Dans ce travail, nous avons choisi une approche différente en étudiant l'espace latent appris par des auto-encodeurs pour estimer la moyenne μ. Pour cela, nous proposons deux architectures d'auto-encodeurs, une première effectuant la reconstruction des séries d'origines, et une seconde, multi-tâches, effectuant en plus une tâche de classification. Dans cette dernière architecture, la classification est imposée pour forcer l'encodeur à apprendre des caractéristiques latentes par classes qui soient séparables et compactes. Dans ce travail, nous nous concentrons sur l'utilisation de caractéristiques latentes pour estimer une moyenne cohérente pour une raison principale :

— Soit un ensemble de séries $X = \{X_1, X_2, X_3, \ldots, X_N\} | X_i \in \mathbb{R}^m$, un alignement multiple de l'ensemble est sensé transformer les séries pour les rapprocher les unes des autres, c'est à dire obtenir une représentation compacte dans $\mathbb{R}^n$, où $n \geq m$ (Petitjean et al., 2011; Shapira Weber et al., 2019a). Ainsi, l'objectif est d'étudier si les auto-encodeurs sont capables d'extraire des caractéristiques compactes et séparables et ainsi de simuler un alignement temporel multiple par classe dans $\mathbb{R}^l$, où $m > l$.

De plus, l'objectif est de tirer partie de la nature symétrique d'un auto-encodeur pour projeter les moyennes latentes dans le domaine temporel et observer leur performance en comparaison à l'état de l'art. A notre connaissance, les approches génératives n'ont jamais été utilisées pour le calcul de moyennes de séries temporelles.

2 Etat de l'art

Comme mentionné auparavant, le calcul de moyennes de séries temporelles se base sur des algorithmes d'alignement comme DTW ou la considération d'un difféomorphisme (Gupta et al., 1996; Niennattrakul et Ratanamahatana, 2009; Petitjean et Gançarski, 2012; Shapira Weber et al., 2019a).

La première approche est le filtre de calcul de moyenne et d'alignement non linéaire (NLAAF) (Gupta et al., 1996). Dans NLAAF, les séries d'un ensemble sont alignées par paire avant le calcul de moyenne. Pour cela, étant donné N séries temporelles, NLAAF génère $N/2$ estimations à la première itération, en calculant la moyenne arithmétique de chaque paire de séries alignées avec DTW. Le processus itératif se poursuit jusqu'à ce qu'une seule estimation soit générée. Une des limites de l'approche NLAAF est qu'elle suppose un nombre pair de séries (Niennattrakul et Ratanamahatana, 2009). En outre, différentes estimations sont générées pour différentes sélections initiales de paires de séries. Enfin, la dimension de la moyenne estimée augmente à chaque itération en raison du calcul de moyenne pour chaque point associé par DTW (Petitjean et Gançarski, 2012).

Pour surmonter ces limitations, deux heuristiques de calcul de moyennes ont été proposées : la moyenne de forme prioritaire (PSA) (Niennattrakul et Ratanamahatana, 2009) et la moyenne barycentrique DTW (DBA) (Petitjean et Gançarski, 2012). PSA a surmonté la dépendance à l'initialisation en utilisant le clustering hiérarchique et la moyenne par paire (Niennattrakul et Ratanamahatana, 2009). Cependant, PSA n'est pas capable d'éviter le problème de dimension croissante de la moyenne (Petitjean et Gançarski, 2012). Par conséquent, DBA a proposé d'aligner les membres de la série sur un modèle sélectionné dans l'ensemble ou initialisé aléa-

toirement (Petitjean et Gançarski, 2012). De plus, DBA a proposé de prendre le barycentre des points associés par DTW (Petitjean et Gançarski, 2012). Cette approche offre deux avantages. Tout d'abord, DBA a simulé un alignement multiple en alignant la série sur un modèle. Par conséquent, il a fourni des résultats améliorés par rapport à son prédécesseur. Deuxièmement, la dimension de la moyenne estimée est fixée à la dimension du modèle. Néanmoins, DBA présente un inconvénient majeur car l'algorithme doit toujours minimiser une fonction de coût non lisse et non convexe (Schultz et Jain, 2018).

Pour cela, une version dérivable de DTW, appelée softDTW, a été proposée (Cuturi et Blondel, 2017). Ces travaux proposent de lisser la fonction de coût et améliorent ainsi la probabilité que DBA identifie les minimums globaux, résultant en l'approche softDBA. Malgré tout, la méthode étant basée sur DTW, elle nécessite un nouveau calcul complet de la moyenne dès qu'une nouvelle série devient disponible (Shapira Weber et al., 2019a).

Pour faire face à cette observation, une approche récente a été proposée : "Diffeomorphic temporal alignment net" (DTAN) (Shapira Weber et al., 2019a). Elle utilise des champs de vitesse affines, continus et par morceau basés sur un difféomorphisme. DTAN estime ces champs de vitesses via un réseau de neurones à convolutions à partir du jeu de données d'entrée. Ensuite, les champs estimés sont utilisés pour inférer la transformation difféomorphe qui minimise la somme des carrés intra-groupes des séries temporelles appartenant à la même classe. Après l'application de cette transformation, les séries temporelles d'une même classe sont alignées et la moyenne arithmétique des séries est considérée comme cohérente (Shapira Weber et al., 2019b). En plus d'être moins coûteuse, l'approche DTAN aborde le problème d'alignement multiple en transformant des groupes de séries temporelles simultanément. Ainsi, cette approche obtient des résultats supérieurs en comparaison des approches précédentes (Shapira Weber et al., 2019b).

3 Approche proposée

Afin d'estimer des moyennes de séries temporelles et de répondre au défi de l'alignement, notre but est d'obtenir une représentation compacte des séries originales au sein d'un espace latent appris par un réseau de neurones. Par exemple, si nous considérons un réseau de neurones entraîné pour une tâche de classification, nous attendons que celui-ci soit capable d'extraire des caractéristiques cohérentes pour chaque classe à travers ses couches cachées (Vasilev et al., 2019; Fawaz et al., 2019). Ainsi, nous attendons que ces caractéristiques spécifiques à chaque classe dans l'espace latent soient plus compactes et séparables que leur contrepartie originale dans le domaine temporel. En d'autres mots, nous attendons qu'un réseau de neurones apprenne à simuler l'alignement temporel au sein de son espace latent. Dans ce travail, notre but est d'identifier une architecture et une tâche d'apprentissage adéquates pour transformer les séries temporelles en représentations latentes. Contrairement à la majorité des techniques de transformation, nous voulons que cette transformation ait également une interprétation sensée dans le domaine temporel. En effet, il est souhaitable de pouvoir observer et évaluer la capacité des moyennes estimées dans l'espace latent à préserver les formes des séries observées dans le domaine temporel. Dans ce but, nous avons choisi d'étudier les auto-encodeurs pour deux raisons principales :

1. Un auto-encodeur est par nature utilisé pour obtenir une représentation compressée d'un ensemble de données au sein d'un espace latent. Il est donc adapté à notre problématique d'apprentissage de caractéristiques.
2. Les auto-encodeurs permettent de projeter les représentations latentes apprises dans l'espace d'origine, le domaine temporel.

Dans un premier temps, nous étudions les performances d'un auto-encodeur à convolutions pour l'extraction de caractéristiques, comme illustré dans la figure 2 (partie haute de l'architecture). Dans un deuxième temps, nous proposons d'apprendre également des caractéristiques permettant de discriminer les classes entre elles en ajoutant une tâche de classification à celle de reconstruction. Le modèle devient ainsi un auto-encodeur multi-tâches, comme illustré en figure 2. Avec cette seconde architecture, notre objectif est de forcer l'encodeur à apprendre des représentations latentes compactes et spécifiques à chaque classe. Cependant, cette modification transforme la tâche d'apprentissage des caractéristiques en un problème semi-supervisé puisque les étiquettes des classes sont requises durant l'apprentissage. Néanmoins, notre approche reste une approche de calcul de moyenne entraînable contrairement à la majorité des méthodes heuristiques à l'exception de DTAN (Shapira Weber et al., 2019a).

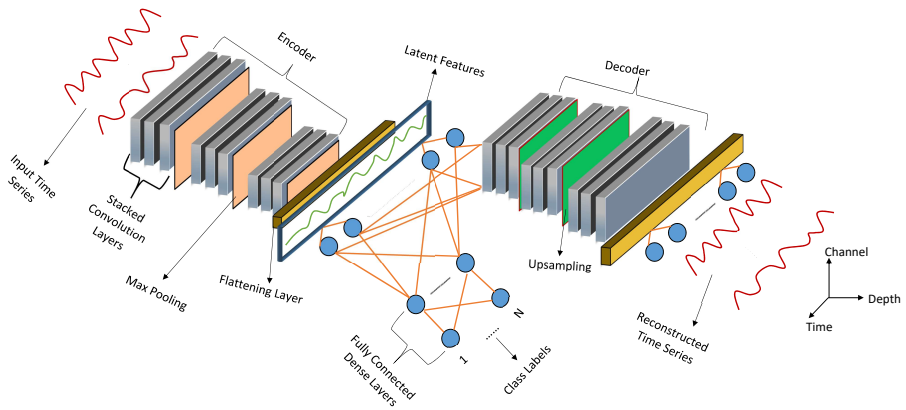


FIG. 2: Architecture de l'auto-encodeur multi-tâches proposé.

3.1 Auto-encodeurs à convolutions

3.1.1 Modèles d'auto-encodeur

Un auto-encodeur utilise une architecture symétrique incluant un encodeur apprenant une représentation latente des données d'entrée et un décodeur apprenant la re-projection dans l'espace d'origine (Baldi, 2012; Dong et al., 2018). Pour cela, un auto-encodeur a pour but de minimiser une fonction de coût de reconstruction des données originales à partir de leurs représentations latentes. Soit une série temporelle $X = \{x_0, x_1, ..., x_T\}$ et sa version reconstruite $R = \{r_0, r_1, ..., r_T\}$, un auto-encodeur minimise la fonction de coût :

$$L_{reconstruction} = \frac{1}{T}\sum_{i=0}^{T}(r_i - x_i)^2. \quad (2)$$

3.1.2 Auto-encodeurs à convolutions

Au sein d'un auto-encodeur, afin de transformer les données d'entrée en représentation latente puis de reconstruire les données d'origine, différentes architectures peuvent être envisagées (couches de neurones entièrement connectés, couches de neurones à convolutions ou couches de neurones récurrents). Un type de couche influence fortement le type des caractéristiques latentes extraites (Vasilev et al., 2019). Dans ce travail, nous avons sélectionné une architecture basée sur des couches de neurones à convolutions pour deux raisons principales :

1. Les couches de neurones à convolutions sont moins coûteuses en temps calcul.
2. Les couches de neurones à convolutions permettent d'apprendre des caractéristiques locales et de les identifier par une analyse globale de la série.

Une couche de convolution permet d'extraire des caractéristiques en calculant le produit scalaire entre des données d'entrées et un filtre de convolution. Ainsi, pour une série temporelle d'entrée $X = \{x_0, x_1, x_2, ..., x_T\}$ et les poids à apprendre d'un filtre de taille j, $W = \{w_0, w_1, ..., w_j\}$, la sortie de la couche de convolution sera calculée comme suit :

$$C(i) = \sum_{k=0}^{j} w_k x_{i+k} \tag{3}$$

Nous pouvons considérer chaque élément de la transformation $C(i)$ comme la réponse d'un filtre de convolution appliqué sur un segment de la série temporelle d'entrée. Le filtre de convolution est glissé le long de la série temporelle selon un certain pas et une nouvelle réponse est calculée pour chaque segment. La plupart des couches de convolutions apprennent M filtres en parallèle. Ainsi, la sortie d'une couche de convolution avec un pas de 1 est un dictionnaire de réponses de taille $M \times T$, où T est la longueur des séries du jeu de données d'origine.

La complexité et les capacités de description des caractéristiques apprises peuvent être améliorées en empilant les couches de convolutions afin d'apprendre différentes formes de manière hiérarchique en augmentant la "vue" de chaque filtre de convolution (Vasilev et al., 2019; Simonyan et Zisserman, 2014; Fawaz et al., 2019). De plus, la plupart des réseaux de neurones à convolutions affinent les caractéristiques apprises par l'intermédiaire de couches de pooling. Ces couches permettent de filtrer les caractéristiques bruitées et/ou redondantes en sélectionnant soit les caractéristiques dominantes (Max Pooling), soit les caractéristiques moyennes (Average Pooling). Ainsi, une opération de pooling permet également de réduire la dimension des données d'entrée par un facteur $\frac{1}{K}$, où K est la taille du filtre de pooling. Enfin, la majorité des couches de neurones à convolutions intègrent des fonctions d'activation comme la fonction ReLU. Elle permet de tronquer les réponses négatives des neurones afin de ne garder uniquement des réponses positives. De plus, l'activation ReLU permet de surmonter le problème de disparition du gradient lors de l'optimisation des poids par rétro-propagation et ainsi d'assurer une convergence la plupart du temps (Vasilev et al., 2019).

3.1.3 Auto-encodeur à convolutions proposé

Comme décrit auparavant, nous pouvons considérer un auto-encodeur à convolutions comme une fonction de transformation non linéaire. La transformation apprend les fonctions de base (caractéristiques) d'un jeu de données. La plupart du temps, ces fonctions ont la capacité à

séparer l'information de phase des séries d'un jeu de données. Par exemple, si nous considérons une série temporelle $X = sin(t + 30), t = 1, \ldots, T$, elle peut aussi être écrite comme $sin(t)cos(30) + cos(t)sin(30)$. Ainsi, nous pouvons écrire X comme :

$$sin(t + 30) = [sin(t)\ cos(t)][cos(30)\ sin(30)]^T. \tag{4}$$

De plus, toute variante de $sin(t)$ et $cos(t)$ peut également être écrite de cette manière, faisant ainsi de $sin(t)$ et $cos(t)$ le dictionnaire de base. De manière générale, pour un dictionnaire D de taille $M \times T$ et un facteur de combinaison $q \in \mathbb{R}^n$, une série temporelle $X \in \mathbb{R}^m$ est calculée comme suit :

$$X = Dq^T, \tag{5}$$

où les fonctions de base du dictionnaire sont arrangées en colonne.

Avec cette définition, nous pouvons à présent considérer les sorties de l'encodeur, illustré en Figure 2, comme les fonctions de base apprises (D). Nous pouvons aussi considérer les facteurs de combinaison (q) comme les poids appris. Si les entrées de l'encodeur appartiennent à des classes similaires, alors les couches de convolutions, les activations linéaires et les couches de pooling vont extraire des caractéristiques similaires. De plus, les transformations combinées doivent permettre de filtrer les distorsions de phase mineures. Si un auto-encodeur répond à ces deux caractéristiques, nous pouvons considérer les représentations latentes comme compactes. Ainsi, la moyenne arithmétique dans cet espace latent peut être vue comme une moyenne cohérente des séries temporelles étudiées.

Néanmoins, dans beaucoup de cas pratiques, plusieurs moyennes par classe sont nécessaires. Dans de tels scénarios, il serait ambitieux d'attendre d'un auto-encodeur basique de générer des caractéristiques latentes compactes et séparables. Les approches existantes basées sur les heuristiques utilisent l'information de classe directement ou indirectement. Par exemple, DBA calcule les moyennes pour chaque classe séparément, tandis que DTAN utilise les étiquettes des classes pour apprendre un alignement par classe (Shapira Weber et al., 2019a; Petitjean et Gançarski, 2012). Dans notre cas, nous avons identifié trois solutions possibles :

1. Pour C classes, entraîner C auto-encodeurs.
2. Forcer un simple auto-encodeur à apprendre plusieurs dictionnaires par classe.
3. Intégrer l'information de classe dans l'auto-encodeur pour en faire un modèle multi-tâches

La première solution n'a pas été retenue car trop coûteuse en temps de calcul. Nous avons donc privilégié les deux autres propositions. Pour le modèle multi-tâches, nous proposons un auto-encodeur effectuant à la fois une tâche de classification et une tâche de reconstruction avec un encodeur partagé, comme illustré en Figure 2.

3.2 Auto-encodeur multi-tâches

La qualité des représentations latentes apprises est dépendante de l'objectif donné à l'auto-encodeur à travers sa fonction de coût. Dans un auto-encodeur basique, la fonction de coût va permettre de forcer l'encodeur à apprendre des caractéristiques minimisant l'erreur de reconstruction. Dans le cas d'entrées multi-classes, il n'y a pas de contrôle de la séparabilité des caractéristiques par classe dans l'espace latent. Cela peut augmenter les chances d'obtenir des moyennes latentes mélangées entre les classes. Pour faire face à ce problème, il est également

possible de forcer l'auto-encodeur à effectuer une tâche supplémentaire. Afin d'obtenir une représentation latente compacte, nous avons choisi comme seconde tâche la classification pour deux raisons :

1. Le classifieur permet de forcer l'encodeur à apprendre des caractéristiques discriminantes entre les classes (Vasilev et al., 2019).
2. Le classifieur permet aussi une combinaison compacte des caractéristiques discriminantes après la couche d'aplanissement de l'encodeur. Cela est dû au fait que dans l'architecture proposée, le classifieur est attaché au modèle après la couche de neurones entièrement connectés. En effet, afin de classifier correctement les séries temporelles, le classifieur doit d'abord combiner les caractéristiques apprises.

De plus, cette caractéristique multi-tâches doit également permettre d'apprendre de meilleures caractéristiques à partir d'ensembles d'apprentissage limités. L'approche multi-tâche de l'auto-encodeur modifie la fonction de coût 2 et devient :

$$L_{multi}(x, r, h, P) = \frac{1}{T}\sum_{i=0}^{T}(r_i - x_i)^2 - \sum_{c=0}^{C} h_{o,c} \log p_{o,c}, \tag{6}$$

où r et x sont respectivement la série reconstruite et la série originale, h et p les représentations catégorielles et les valeurs d'activation Softmax pour chaque catégorie c.

4 Évaluation expérimentale

4.1 Architectures d'auto-encodeur proposées

Pour les portions d'encodeur et de décodeur du réseau, nous avons employé une architecture similaire à celle proposée par l'équipe VGG (Visual Geometry Group), le modèle VGG16, comme illustré dans la partie haute de la Figure 2. Ainsi, pour l'encodeur, nous avons utilisé neuf couches de convolutions divisées en trois groupes. Chaque groupe utilise respectivement 128, 64 et 32 filtres de convolution. Après chaque groupe de convolution, une couche de maximum pooling est utilisée (illustrées en orange dans la Figure 2). Une architecture similaire est utilisée pour le décodeur avec le remplacement du pooling par des couches de sur-échantillonage (illustrées en vert dans la Figure 2). De plus, nous avons fixé la dimension D_l de l'espace latent à $T/4$, où T est la longueur d'une série temporelle. Enfin, pour l'auto-encodeur multi-tâches, le classifieur est construit à partir de trois couches de neurones entièrement connectés avec respectivement $D_l/2$, $D_l/4$ et C neurones, C correspondant au nombre de classes.

4.2 Procédures d'évaluation et jeux de données

En pratique, la qualité de l'estimation d'une moyenne est mesurée par la somme des carrés intra-groupes (WGSS), une somme de Fréchet donnée en équation (1) (Petitjean et Gançarski, 2012). Une alternative est d'effectuer une classification basée sur le plus proche centroïde. En effet, si les moyennes estimées permettent d'obtenir une bonne précision de classification via le plus proche centroïde, il est ainsi fort probable qu'elles minimisent leur distance

WGSS (Shapira Weber et al., 2019b). Pour cela, nous avons évalué notre proposition en utilisant la classification via le plus proche centroïde sur un ensemble de 85 jeux de données issus de l'archive UCR (Chen et al., 2015).

Nous avons effectué la tâche de classification à la fois dans l'espace latent et dans le domaine temporel. Pour la classification dans l'espace latent, nous avons utilisé la moyenne arithmétique des caractéristiques latentes extraites à partir de l'ensemble d'apprentissage. Nous avons ensuite projeté les séries temporelles de l'ensemble de test dans l'espace latent grâce à l'auto-encodeur appris, puis avons calculé la similarité entre ces projections et les moyennes par classe grâce à la distance euclidienne. Cela est effectué pour évaluer la séparabilité et la compacité des caractéristiques latentes. Au contraire, comme le domaine temporel est toujours impacté par de possibles distorsions, la qualité des moyennes projetées dans ce domaine est évalué à travers la distance DTW. Afin de comparer les résultats de notre approche, nous avons extrait les résultats dans le domaine temporel des autres techniques de calcul de moyennes reportés dans (Shapira Weber et al., 2019b). Les résultats de classification pour DBA et softDBA ont été calculés grâce à la distance DTW sur 84 jeux de données. Bien que les auteurs n'aient pas calculé les résultats de classification pour le jeu de données "StarlightCurves", nous avons fait nos expérimentation sur l'ensemble des 85 jeux de données pour permettre des comparaisons futures. De plus, Shapira Weber et al. (2019b) calcule également le taux de classification en utilisant les moyennes arithmétiques dans le domaine temporel. Cela n'est pas très juste car la moyenne arithmétique est fortement impactée par les distorsions temporelles. De manière différente, nous avons comparé notre approche aux moyennes arithmétiques dans le domaine temporel calculées avec la distance DTW. Nous avons utilisé les implémentations de DTW, DBA et SDBA fournies dans la bibliothèque Tslearn(Tavenard et al., 2020).

Nous avons effectué plus de 580 expérimentations. Le code a été implémenté en Keras avec un backend Tensorflow. Notre implémentation de l'auto-encodeur ainsi que l'ensemble des résultats expérimentaux sont disponibles en ligne [1]. Lors des expérimentations, nous avons utilisé 80% des données d'apprentissage pour entraîner le modèle et 20% pour la validation. Les auto-encodeurs simples et multi-tâches ont été entraînés durant 2500 époques avec un taux d'apprentissage fixé à 10^{-4}.

4.3 Résultats expérimentaux

4.3.1 Analyse des résultats

Comme le montre la Table 1, dans l'espace latent, l'auto-encodeur multi-tâches obtient le meilleur taux de classification pour 32.14% des jeux de données (MT.Enc.Lat). Cependant, la projection des moyennes dans le domaine temporel ne permet d'obtenir le meilleur taux de classification que sur 2.38% des jeux de données (MT.Enc.Time). L'auto-encodeur simple obtient un meilleur taux sur 2.38% des jeux de données dans l'espace latent (Enc.Lat) et sur 1.19% des jeux de données dans le domaine temporel (Enc.Time), se comportant ainsi comme une moyenne arithmétique, comme illustré en Figure 6. Les approches de l'état de l'art, DTAN, DBA, SDBA et la moyenne arithmétique obtiennent le meilleur taux de classification dans le domaine temporel sur $35.71\%, 2.38\%, 19.05\%$ et 1.19% des jeux de données, respectivement.

Les résultats de classification de la Table 1 montrent que notre auto-encodeur multi-tâches est capable d'extraire des caractéristiques compactes permettant d'obtenir des résultats compa-

1. https://github.com/tsegaterefe/Time-Series-Averaging-Using-Multi-Tasking-Autoencoder

TAB. 1: Résultats comparatifs de classification basée sur le plus proche centroïde.

No.	Méthodes de calcul de moyennes	Victoires	Égalités
1.	MT.Enc.Lat	27	1
2.	MT.Enc.Time	2	1
3.	Enc.Lat	2	0
4.	Enc. Time	1	0
5.	DTAN	30	4
6.	DBA	2	1
7.	SDBA	16	1
8.	Moyenne arithmétique	1	0

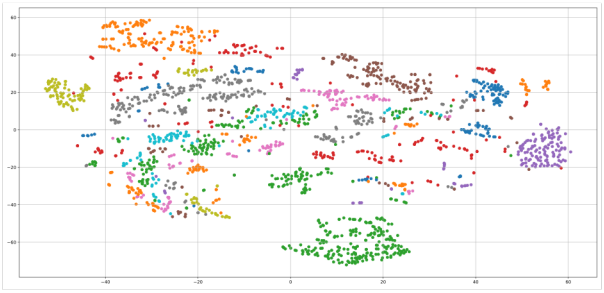
rables à l'état de l'art (DTAN). Les résultats montrent également que l'auto-encodeur simple ne parvient pas à capturer de telles caractéristiques séparables. Cela est démontré par la Figure 3 représentant la projection t-SNE des séries du jeux de données FaceUCR avec 14 classes différentes. La Figure 3a montre la projection pour l'ensemble de test d'origine, tandis que la Figure 3b et la Figure 3c montrent la projection pour l'ensemble de test dans l'espace latent des auto-encodeurs simples et multi-tâches, respectivement. Ces projections t-SNE démontrent que l'auto-encodeur apprend des caractéristiques par classe qui sont séparables et compactes. Ainsi, avec ces caractéristiques, il est possible de calculer des moyennes arithmétiques par classe cohérentes et obtenant des meilleures performances à la fois dans l'espace latent et dans le domaine temporel.

En outre, afin de permettre une analyse visuelle des projections dans le domaine temporel, nous avons illustré quelques exemples de moyennes estimées avec la moyenne arithmétique, DBA, SDBA et notre approche d'auto-encodeur multi-tâches en Figure 4. Nous avons sélectionné les jeux de données ECG200 (haut) et ECGFiveDays (bas).

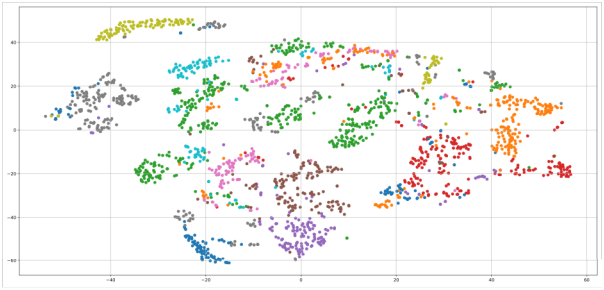
Pour l'estimation de la moyenne avec l'auto-encodeur multi-tâches, nous avons utilisé un réseau entraîné pour projeter les données de l'ensemble d'apprentissage dans l'espace latent. Nous avons ensuite calculé les moyennes comme les moyennes arithmétiques dans cet espace latent. Enfin, le décodeur a été utilisé pour projeter les estimations des moyennes dans le domaine temporel. La Figure 4d montre l'efficacité de l'auto-encodeur multi-tâches pour capturer les valeurs aux pics et les formes générales observées sur les séries originales, contrairement aux moyennes arithmétiques directement calculées dans le domaine temporel illustrées en Figure 4a. Pour calculer les moyennes avec DBA et SDBA, nous avons exécuté leur algorithme respectif pendant 100 époques. La Figure 4b et la Figure 4c montrent comment ces deux approchent sont capable de mieux capturer l'allure générale des séries en comparaison à l'auto-encodeur multi-tâches dans le domaine temporel.

4.3.2 Test d'hypothèse

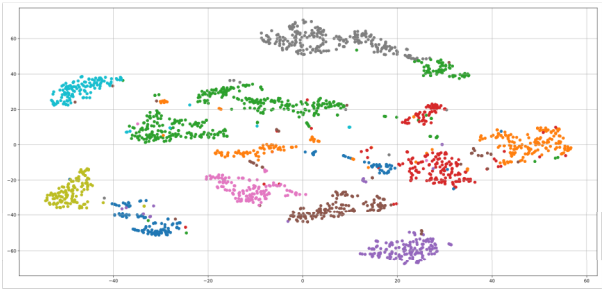
En plus des représentations visuelles et des comparaisons des taux de classification, nous avons effectué un test des rangs signés de Wilcoxon, synthétisé par le diagramme de différence critique en Figure 6. Le diagramme montre que la classification dans l'espace latent (MT.Enc.Lat) en utilisant l'auto-encodeur multi-tâches est une hypothèse similaire à DTAN.



(a) Séries temporelles originales



(b) Caractéristiques latentes de l'auto-encodeur simple



(c) Caractéristiques latentes de l'auto-encodeur multi-tâches

FIG. 3: Projections t-SNE pour le jeux de données FacesUCR. Les données d'entrée sont les séries temporelles originales (a), les séries encodées avec l'auto-encodeur simple (b) et les séries encodées avec l'auto-encodeur multi-tâches (c).

En effet, l'auto-encodeur permet de simuler l'alignement temporel et le calcul de moyennes arithmétiques dans l'espace latent permet alors d'obtenir des moyennes cohérentes. De plus, le diagramme montre que les moyennes latentes projetées dans le domaine temporel (MT.Enc.Time) permettent une meilleur classification qu'avec les moyennes arithmétiques directement calculées dans le domaine temporel avec DTW. Cependant, comme l'auto-encodeur simple n'est pas capable de capturer des caractéristiques latentes compactes et séparables, comme illustré en Figure 3, ses performances de classification sont inférieures en comparaison des autres approches heuristiques de calcul de moyenne.

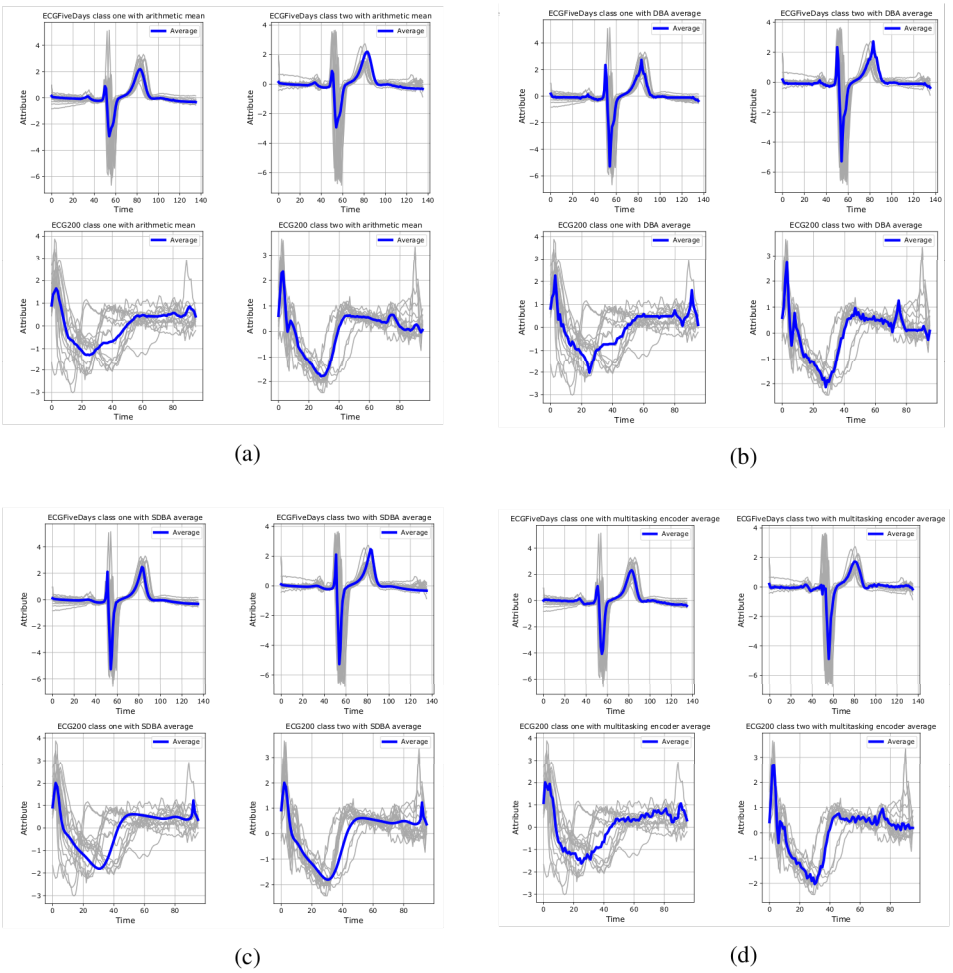
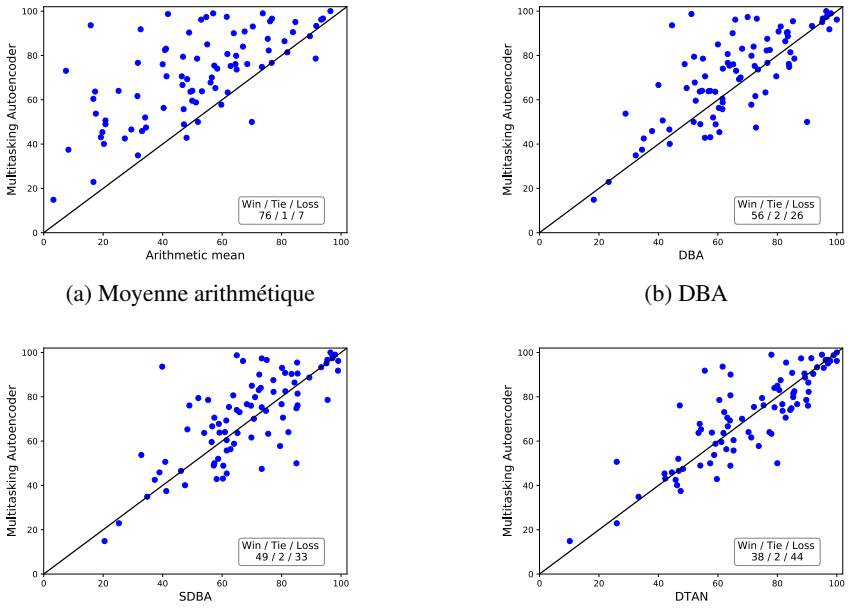


FIG. 4: Comparaison visuelle dans le domaine temporel des moyennes estimées avec la moyenne arithmétique (a), DBA (b), SDBA (c) et notre auto-encodeur multi-tâches (d).

5 Conclusion

Dans cet article, nous avons étudié l'utilisation de réseaux de neurones génératifs pour le calcul de moyennes de séries temporelles. Pour atteindre cet objectif, nous avons proposé de simuler l'alignement temporel de multiples séries temporelles en considérant l'espace latent d'auto-encodeurs simples et multi-tâches basés sur des couches de convolutions. Les représentations dans l'espace latent montrent qu'un auto-encodeur cherchant à optimiser une fonction de coût prenant en compte à la fois la reconstruction et la classification des séries permet d'obtenir des moyennes cohérentes. De plus, la projection des moyennes latentes dans le domaine temporel donne de meilleurs résultats de classification en comparaison aux moyennes arithmétiques directement calculées dans le domaine temporel. Cependant, ces résultats sont



(a) Moyenne arithmétique (b) DBA

(c) SDBA (d) DTAN

FIG. 5: Comparaison du taux de classification de notre approche avec les moyennes arithmétiques (a), DBA (b), SDBA (c) et DTAN (d).

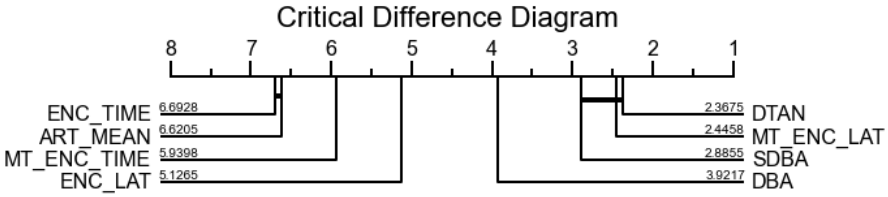


FIG. 6: Comparaison des méthodes par le test des rangs signés de Wilcoxon.

largement inférieurs à ceux obtenus pour les moyennes latentes puisque le décodeur n'observe qu'un nombre limité de séries pour optimiser la reconstruction. Pour faire face à ce problème, nous envisageons d'étendre notre approche en considérant des espace latents continus grâce aux auto-encodeurs variationnels et aux modèles de mélanges Gaussiens. De plus, nous souhaitons également effectuer des expérimentations étendues pour analyser l'impact des hyper-paramètres des auto-encodeurs (nombre de couches, taille des filtres, etc.) sur la qualité des moyennes calculées.

Remerciements

Ce travail de recherche a été mené dans le cadre du programme doctoral Campus France Ethio-France, financé par l'Ambassade de France pour l'union Éthiopienne et Africaine et le Ministère Éthiopien des Sciences et de l'Enseignement supérieur (MoSHE). Nous tenons à remercier les deux organisations pour leur soutien financier.

Références

Bagnall, A., L. Davis, J. Hills, et J. Lines (2012). Transformation based ensembles for time series classification. In *Proceedings of the 2012 SIAM international conference on data mining*, pp. 307–318. SIAM.

Bagnall, A., J. Lines, A. Bostrom, J. Large, et E. Keogh (2017). The great time series classification bake off : a review and experimental evaluation of recent algorithmic advances. *Data Mining and Knowledge Discovery 31*(3), 606–660.

Baldi, P. (2012). Autoencoders, unsupervised learning, and deep architectures. In *Proceedings of ICML workshop on unsupervised and transfer learning*, pp. 37–49.

Brill, M., T. Fluschnik, V. Froese, B. Jain, R. Niedermeier, et D. Schultz (2019). Exact mean computation in dynamic time warping spaces. *Data Mining and Knowledge Discovery 33*(1), 252–291.

Chen, Y., E. Keogh, B. Hu, N. Begum, A. Bagnall, A. Mueen, et G. Batista (2015). The ucr time series classification archive. `www.cs.ucr.edu/~eamonn/time_series_data/`.

Cuturi, M. et M. Blondel (2017). Soft-dtw : a differentiable loss function for time-series. In *Proceedings of the 34th International Conference on Machine Learning*, Sydney, NSW, Australia, pp. 894–903.

Dong, G., G. Liao, H. Liu, et G. Kuang (2018). A review of the autoencoder and its variants : A comparative perspective from target recognition in synthetic-aperture radar images. *IEEE Geoscience and Remote Sensing Magazine 6*(3), 44–68.

Fawaz, H. I., G. Forestier, J. Weber, L. Idoumghar, et P.-A. Muller (2019). Deep learning for time series classification : a review. *Data Mining and Knowledge Discovery 33*(4), 917–963.

Gupta, L., D. L. Molfese, R. Tammana, et P. G. Simos (1996). Nonlinear alignment and averaging for estimating the evoked potential. *IEEE transactions on biomedical engineering 43*(4), 348–356.

Jain, B. J. et D. Schultz (2016). On the existence of a sample mean in dynamic time warping spaces.

Niennattrakul, V. et C. A. Ratanamahatana (2009). Shape averaging under time warping. In *2009 6th International Conference on Electrical Engineering/Electronics, Computer, Telecommunications and Information Technology*, Volume 2, pp. 626–629. IEEE.

Petitjean, F. et P. Gançarski (2012). Summarizing a set of time series by averaging : From steiner sequence to compact multiple alignment. *Theoretical Computer Science 414*(1), 76–91.

Petitjean, F., A. Ketterlin, et P. Gançarski (2011). A global averaging method for dynamic time warping, with applications to clustering. *Pattern Recognition 44*(3), 678–693.

Schultz, D. et B. Jain (2018). Nonsmooth analysis and subgradient methods for averaging in dynamic time warping spaces. *Pattern Recognition 74*, 340–358.

Shapira Weber, R. A., M. Eyal, N. Skafte, O. Shriki, et O. Freifeld (2019a). Diffeomorphic temporal alignment nets. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, et R. Garnett (Eds.), *Advances in Neural Information Processing Systems 32*, pp. 6574–6585. Curran Associates, Inc.

Shapira Weber, R. A., M. Eyal, N. Skafte, O. Shriki, et O. Freifeld (2019b). Diffeomorphic temporal alignment nets : Supplemantary material. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, et R. Garnett (Eds.), *Advances in Neural Information Processing Systems 32*, pp. 6574–6585. Curran Associates, Inc.

Simonyan, K. et A. Zisserman (2014). Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv :1409.1556*.

Tavenard, R., J. Faouzi, G. Vandewiele, F. Divo, G. Androz, C. Holtz, M. Payne, R. Yurchak, M. Rußwurm, K. Kolar, et E. Woods (2020). Tslearn, a machine learning toolkit for time series data. *Journal of Machine Learning Research 21*(118), 1–6.

Vasilev, I., D. Slater, G. Spacagna, P. Roelants, et V. Zocca (2019). *Python Deep Learning : Exploring deep learning techniques and neural network architectures with PyTorch, Keras, and TensorFlow, 2nd Edition*. Packt Publishing.

Wei, W. (2006). *Time Series Analysis : Univariate and Multivariate Methods*. Pearson Addison Wesley.

Summary

The estimation of time series averages has been studied for over three decades. The process is mainly challenging due to temporal distortion. Previous approaches mostly addressed this challenge by using alignment algorithms such as Dynamic Time Warping (DTW). However, the quadratic computational complexity of DTW and its inability to align more than two time series simultaneously complicate the estimation. In this paper, we follow a different path and state the averaging problem as a generative problem. To this end, we propose a multi-tasking convolutional autoencoder architecture to extract latent features of similarly labeled time series under the influence of temporal distortion. We then take the arithmetic mean of latent features as an estimate of the latent mean. Moreover, we project these estimations and investigate their performance in the time domain. We evaluated the proposed approach through one nearest centroid classification using 85 data sets obtained from the UCR archive. Experimental results show that, in the latent space, the proposed multi-tasking autoencoder achieves competitive accuracies as compared to the state-of-the-art, thus demonstrating that the learned latent space is suitable to compute time series averages. In addition, the time domain projection of latent space means provides superior results as compared to time domain arithmetic means.