Apprentissage non supervisé de séries temporelles à l'aide des k-Means et d'une nouvelle méthode d'agrégation de séries

Rémi Gaudin, Nicolas Nicoloyannis

LABORATOIRE ERIC 3038 Université Lumière - Lyon2
Batiment L 5 av. Pierre Mendès-France 69676 BRON cedex FRANCE
Remi.Gaudin@univ-lyon2.fr, Nicolas.Nicoloyannis@univ-lyon2.fr

Résumé. L'utilisation d'un algorithme d'apprentissage non supervisé de type *k-Means* sur un jeu de séries temporelles amène à se poser deux questions : Celle du choix d'une mesure de similarité et celle du choix d'une méthode effectuant l'agrégation de plusieurs séries afin d'en estimer le centre (*i.e.* calculer les k moyennes). Afin de répondre à la première question, nous présentons dans cet article les principales mesures de similarité existantes puis nous expliquons pourquoi l'une d'entre elles (appelée *Dynamic Time Warping*) nous paraît la plus adaptée à l'apprentissage non supervisé. La deuxième question pose alors problème car nous avons besoin d'une méthode d'agrégation respectant les caractéristiques bien particulières du *Dynamic Time Warping*. Nous pensons que l'association de cette mesure de similarité avec l'agrégation Euclidienne peut générer une perte d'informations importante dans le cadre d'un apprentissage sur la "forme" des séries. Nous proposons donc une méthode originale d'agrégation de séries temporelles, compatible avec le *Dynamic Time Warping*, qui améliore ainsi les résultats obtenus à l'aide de l'algorithme des *k-Means*.

Mots-clés : Fouille de données et Apprentissage non supervisé, Séries temporelles, K-Means, Dynamic Time Warping

1 Introduction

Les séries temporelles sont des données ordonnées dans le temps et cet ordonnancement a une signification que l'on ne peut ignorer. Ainsi, on ne peut pas leur appliquer des méthodes de fouille de données classiques mais bien des méthodes spécialement adaptées, qui respectent la temporalité de ce type de donnée. Nous nous intéresserons ici uniquement à l'apprentissage non supervisé à partir des séries temporelles.

L'utilisation d'un algorithme d'apprentissage non supervisé de type "moyenne mobile" (le plus connu étant les *k-Means*) sur un jeu de séries temporelles amène à se poser les questions du choix d'une mesure de distance entre deux séries temporelles et celle du choix d'une méthode effectuant l'agrégation de plusieurs séries temporelles afin d'en estimer le centre (*i.e.* calculer les k moyennes). Afin de répondre à la première question, nous allons dresser l'état des lieux des principales méthodes de comparaison de séries temporelles déjà existantes (paragraphe 2), puis nous allons discuter l'intérêt de chacune d'entre elles dans le cadre d'un apprentissage non supervisé (paragraphe 2.4).

Pour répondre à la seconde question, nous expliciterons l'inconvénient d'associer la mesure de similarité *Dynamic Time Warping* (qui est la mesure que nous avons retenu) avec l'agrégation Euclidienne afin de recalculer les centres de chaque cluster (paragraphe 3.1). Nous présenterons ensuite notre nouvelle méthode d'agrégation dans le paragraphe 3.2. Afin de valider notre apport, nous avons effectué des tests à partir d'un jeu de six types de séries temporelles différentes (paragraphe 4). Pour finir, nous avons énuméré les perspectives d'approfondissement de notre travail et les différentes pistes de recherches futures qui nous ont semblées intéressantes (paragraphe 5).

2 Les principales mesures de similarité

2.1 Mesure de similarité p-normée

Cette mesure de similarité est couramment utilisée car elle a le mérite d'être simple à mettre en œuvre. La similarité $Sim(Q, C)$ entre les séries $Q = q_1, q_2, ..., q_m$ et $C = c_1, c_2, ..., c_m$ est égale à :

$$Sim(Q, C) = \frac{1}{L_p(Q, C)} = \frac{1}{(\sum_{i=1}^{m}(q_i - c_i)^p)^{\frac{1}{p}}} \quad (1)$$

Si $p = 1$ alors on utilise la distance de Manhattan : $L_1(Q, C) = \sum_{i=1}^{m}(q_i - c_i)$
Si $p = 2$ alors on utilise la distance Euclidienne : $L_2(Q, C) = (\sum_{i=1}^{m}(q_i - c_i)^2)^{\frac{1}{2}}$

2.2 Dynamic Time Warping

La particularité de la méthode *Dynamic Time Warping* (DTW) est de savoir gérer les décalages temporels qui peuvent éventuellement exister entre deux séries (Berndt et Clifford 1994). Au lieu de comparer chaque point d'une série avec celui de l'autre série qui intervient au même instant t, on permet à la mesure de comparer chaque point d'une série avec un ou plusieurs points de l'autre série, ceux-ci pouvant être décalés dans le temps (Fig. 1).

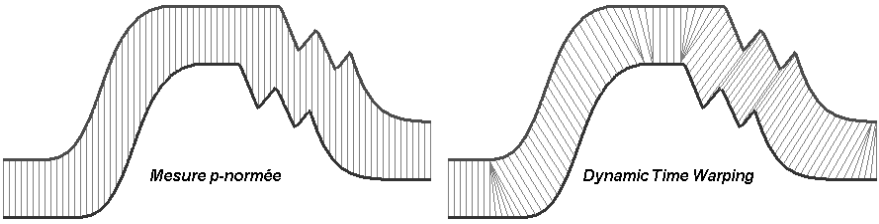

FIG. 1 – Comparaison entre la mesure *p-normée* et le DTW

Le DTW possède une définition récursive qui calcule la similarité entre les séries $Q = q_1, q_2, ..., q_m$ et $C = c_1, c_2, ..., c_n$ de la manière suivante :

Soit $D(i,j)$ la distance entre les sous-séquences $q_1, q_2, ..., q_i$ et $c_1, c_2, ..., c_j$ (avec $1 \leq i \leq m$ et $1 \leq j \leq n$) :

$$D(i,j) = \begin{cases} |q_1 - c_1|, & \text{si } i = j = 1; \\ |q_i - c_j| + min\{D(i-1,j), D(i-1,j-1), D(i,j-1)\}, & \text{sinon.} \end{cases} \quad (2)$$

Soit $Sim(Q,C)$ la mesure de similarité DTW entre les séries Q et C :

$$Sim(Q,C) = \frac{1}{D(m,n)} \quad (3)$$

On peut aussi paramétrer l'écart temporel maximum permis à la mesure pour comparer deux points. On définit ainsi une fenêtre temporelle (appelée *delta*) que l'on fera "glisser" sur chacune des séries à comparer. Par exemple, si on fixe la taille de la fenêtre $delta = 3$, chaque point d'une série qui intervient à un instant t ne pourra être comparé qu'avec les points de l'autre série qui interviennent aux instants $t-3, t-2, t-1, t, t+1, t+2$ et $t+3$. Le choix de la taille de cette fenêtre peut avoir une influence sur le résultat et il convient de la déterminer avec soin (Ratanamahatana et Keogh 2004 a).

2.3 Longest Common Subsequence

L'idée de la méthode *Longest Common Subsequence* ($LCSS$) est de comparer uniquement les portions les plus similaires de chacune des séries. Plus les sous-séquences communes sont nombreuses, plus on considèrera les deux séries comme similaires. Cette méthode effectue donc elle aussi une déformation des séries dans le temps, mais contrairement au DTW qui compare chaque point d'une série avec k points de l'autre série ($1 \leq k \leq delta$ si on utilise une fenêtre temporelle), le $LCSS$ compare chaque point d'une série avec 0 ou 1 point de l'autre.[1]

Afin d'affiner la sélection des sous-séquences communes, on peut paramétrer une fenêtre temporelle de la même manière qu'avec le DTW. On doit aussi fixer une fenêtre spatiale (appelée *epsilon*) qui servira à définir quel est l'écart maximum toléré pour pouvoir considérer deux sous-séquences comme communes. Cette méthode possède, elle aussi, une définition récursive (Yazdani et al. 1997) :

Soit $D(i,j)$ la distance entre les sous-séquences $q_1, q_2, ..., q_i$ et $c_1, c_2, ..., c_j$ (avec $1 \leq i \leq m$ et $1 \leq j \leq n$) :

$$D(i,j) = \begin{cases} 1 + D(i-1,j-1), & \text{si } |q_i - c_j| < epsilon; \\ max\{D(i-1,j), D(i,j-1)\}, & \text{sinon.} \end{cases} \quad (4)$$

Soit $Sim(Q,C)$ la mesure de similarité $LCSS$ entre les séries Q et C :

$$Sim(Q,C) = \frac{D(m,n)}{min\{m,n\}} \quad (5)$$

[1] On rappellera que la méthode *p-normée* compare chaque point de la première série avec obligatoirement un seul point de la seconde : celui qui intervient au même instant t.

2.4 Quelle mesure utiliser en apprentissage non supervisé de séries temporelles ?

2.4.1 Temps de calculs

La complexité de la mesure *p-normée* est linéaire ($O(n)$). Ses temps de calculs sont donc excellents. L'algorithme récursif du DTW est beaucoup trop lent et nécessite d'avoir recours à une méthode d'implémentation appelée *programmation dynamique* (Bellman 1957, Berndt et Clifford 1996). Celle-ci permet de réduire la complexité du DTW pour la ramener à l'ordre du quadratique ($O(m*n)$ si on n'utilise pas de fenêtre temporelle et $O(m*delta)$ si on en utilise une). Le DTW reste toutefois beaucoup moins rapide que la mesure *p-normée* et cela peut être handicapant lorsque l'on doit analyser un gros jeu de séries temporelles où chacune d'entre elles est de taille conséquente. Le $LCSS$ peut être lui-aussi amélioré à l'aide de la *programmation dynamique*. Sa complexité est donc identique à celle du DTW.

2.4.2 Sensibilité au bruit

Les performances de la mesure *p-normée* s'effondrent en présence de données bruitées. Etant donné que tous les points de chacune des séries sont systématiquement comparés, tout point anormalement éloigné (un *outlier*) influencera considérablement le résultat final de la mesure. Le DTW est plus souple mais il reste toutefois sensible au bruit et nécessite un prétraitement des données pour résoudre partiellement ce problème. Le $LCSS$ est très robuste au bruit. Contrairement aux deux autres mesures de similarité, il a le droit de ne pas mesurer tous les points mais seulement ceux qui sont proches entre eux d'une série à l'autre. Toutes les données anormalement éloignées sont donc automatiquement ignorées.

2.4.3 Paramétrage

La mesure *p-normée* ne possède aucun paramétrage[2], ce qui est idéal dans le cadre d'un apprentissage non supervisé. Le DTW possède le paramétrage de la fenêtre temporelle (*delta*) afin de limiter la comparaison de points trop éloignés. Celui-ci n'est pas trop handicapant dans la mesure où une fenêtre dont la taille correspond à 10% de la taille des séries à analyser donne en moyenne des résultats satisfaisants. Le $LCSS$ possède lui aussi le paramétrage de la fenêtre temporelle. Mais on doit en plus effectuer le paramétrage de la fenêtre spatiale (*epsilon*), celui-ci étant très important et pouvant modifier considérablement le résultat final en fonction de sa valeur. Si on ne possède pas de connaissances préalables sur le jeu de données, ce qui est toujours le cas en apprentissage non supervisé, il faut fixer *epsilon* par tâtonnement. En plus de faire perdre un temps non négligeable pendant la phase d'apprentissage, ce paramètre rend les résultats incertains.

[2]Si ce n'est le paramêtre p lui-même, mais celui-ci ne sert qu'à normaliser les distances entre les séries et n'influence pas les résultats de l'apprentissage

2.4.4 Résultats

La mesure *p-normée* est extrêmement sensible aux écarts sur l'axe du temps. Deux séries très similaires dans leurs formes mais qui seraient légèrement décalées l'une à l'autre dans le temps obtiendraient une faible similarité. Le DTW obtient des résultats bien plus satisfaisants. Deux séries très similaires dans leurs formes mais qui interviennent à des instants différents dans le temps obtiendront tout de même une forte valeur de similarité. Le $LCSS$ fournit, lui aussi, des bons résultats en privilégiant les portions similaires aux deux séquences. Mais ceux-ci sont généralement inférieurs aux résultats du DTW sauf si les données sont bruitées. Les deux mesures DTW et $LCSS$ présentent l'inconvénient d'être non métriques, et donc de ne pas forcément respecter l'inégalité triangulaire $Sim(Q, C) \geq Sim(Q, A) + Sim(A, C)$. Cette caractéristique peut être fortement contre-intuitive dans certains cas d'apprentissage.

2.4.5 Conclusion

Nous pensons que la distance DTW est la plus indiquée dans le cadre de l'apprentissage non supervisé. Ses résultats sont très satisfaisants, malgré sa sensibilité au bruit, et son paramétrage n'est pas trop contraignant. La mesure *p-normée* est simple, rapide, sans paramètre, mais elle obtient souvent des résultats catastrophiques qui la disqualifient automatiquement. La principale force du $LCSS$ est sa robustesse au bruit. Ses résultats sont satisfaisants mais nous considérons que le paramétrage de *epsilon* est trop lourd en apprentissage non supervisé.

Nous allons donc utiliser le DTW en tant que mesure de similarité dans notre algorithme d'apprentissage non supervisé de type "moyenne mobile" (en l'occurence nous utiliserons ici les *k-Means*) adapté aux séries temporelles.

3 Les *k-Means* adaptés aux séries temporelles

La méthode des *k-Means* adaptée aux séries temporelles consiste à classer le jeu de séries en k classes (appelées *clusters*) disjointes (Lin et al. 2003). Son algorithme est le suivant :

1. Choisir la valeur de k.
2. Initialiser les centres des k clusters (aléatoirement si nécessaire).
3. Affecter chaque série au cluster dont le centre lui est le plus proche.
4. Ré-estimer les centres des k clusters, en supposant que toutes les affectations des séries sont correctes.
5. Si aucune des séries n'a changé de cluster, alors fin de l'algorithme. Sinon, retour à l'étape 3.

3.1 Problème de l'agrégation des séries temporelles dans le cadre d'un apprentissage sur la "forme" des séries

L'algorithme des *k-Means*, avec des données classiques (*i.e.* non temporelles), ne pose pas de problème d'implémentation. La ré-estimation du centre de chaque cluster

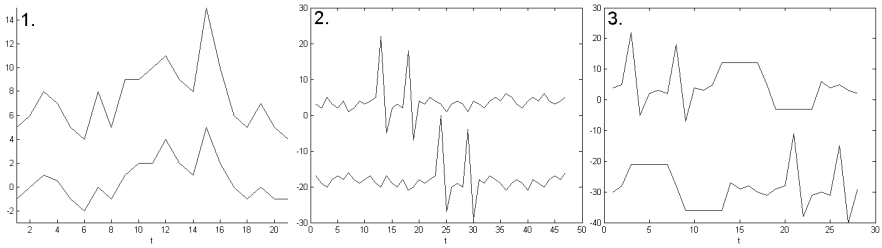

FIG. 2 – Exemples de comparaisons de séries par leurs formes

se fait habituellement en calculant la moyenne Euclidienne de tous les objets affectés à ce cluster (étape 4 de l'algorithme). Par contre, dans le cadre d'un apprentissage basé sur la classification des séries en fonction de leurs "formes", nous pensons que cette agrégation Euclidienne entraîne une perte d'informations importante.

Nous entendons par forme d'une série les différentes variations relatives qu'elle effectue au cours du temps. Cette notion est difficile à définir et assez subjective, mais nous pouvons néanmoins en dégager les deux caractéristiques suivantes :

1. Cette similarité est relative vis-à-vis de l'espace : Les valeurs absolues des points d'une série ne nous intéressent pas. Seules les variations de ces valeurs au cours du temps ont de l'importance (cela implique généralement une normalisation des séries avant tout apprentissage). De même, les amplitudes de chacune de ces variations ne doivent avoir que peu d'influence sur la mesure. Dans le premier exemple de la figure 2, les deux séries sont considérées comme très similaires au niveau de leur forme malgré leurs écarts spatiaux et leurs variations d'amplitudes.
2. Cette similarité est relative vis-à-vis du temps : Les variations des valeurs de deux séries peuvent intervenir à différents instants tout en préservant la similarité des séries (2^{eme} exemple de la figure 2). Par contre, l'ordonnancement dans le temps des différentes variations est important : deux séries possédant les même formes mais dans un ordre différent ne pourront être considérées comme similaires (3^{eme} exemple de la figure 2).

L'agrégation Euclidienne ne permet pas le respect de ces caractéristiques. Prenons par exemple les deux séries suivantes : $Q_1 = \{1, 2, 4, 3, 1, 1, 2, 1\}$ et $Q_2 = \{2, 1, 1, 2, 4, 3, 1, 1\}$. Si nous essayons d'agréger les séries Q_1 et Q_2 en une troisième série à l'aide de la moyenne Euclidienne, nous obtenons la série suivante : $S = \{1.5, 1.5, 2.5, 2.5, 2.5, 2, 1.5, 1\}$. La figure 3 présente le résultat graphique de cette agrégation.

On constate en observant la figure 3 que l'agrégation Euclidienne ne conserve pas les formes, pourtant très similaires, des deux séries originelles. Les deux séries Q_1 et Q_2 possèdent chacune un pic de valeur 4 (le 3^{eme} point pour Q_1 et le 5^{eme} pour Q_2) qu'on ne retrouve pas dans la série S. Cela est du au fait que les deux sommets interviennent à des endroits différents dans l'axe du temps et que l'agrégation Euclidienne ne gère pas les décalages temporels. L'agrégation Euclidienne conduit donc dans cet exemple à

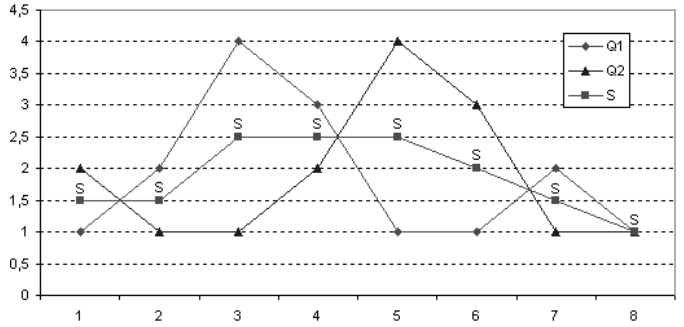

FIG. 3 – Agrégation Euclidienne de deux séries temporelles

une perte importante d'information qui nous semble dommageable dans le cadre d'un apprentissage basé sur les formes des séries. C'est pour cette raison que nous avons créé une nouvelle méthode d'agrégation qui sache gérer les éventuels décalages temporels entre deux séries.

3.2 Elaboration d'une nouvelle méthode d'agrégation

Nous avons vu que la mesure DTW associe chaque point d'une des deux séries avec un ou plusieurs points de l'autre afin d'évaluer la distance qui les sépare. Nous appellerons chacune de ces associations un *arc*. Si cet arc ne possède aucun point en commun avec d'autres arcs, on l'appellera *arc simple*. Si cet arc possède un point en commun avec un ou plusieurs autres arcs, on appellera alors l'ensemble de ces arcs groupés un *groupement d'arcs* (fig. 4).

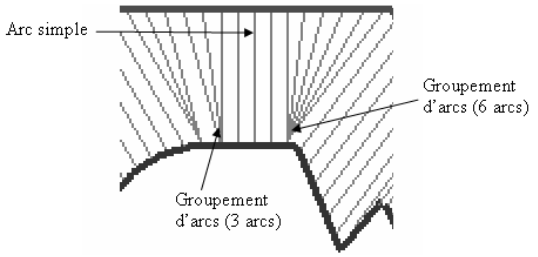

FIG. 4 – Arcs issus de la distance DTW

Nous allons à présent énoncer les règles d'agrégation suivantes :
– Soit Q_1 et Q_2 les deux séries à agréger et S la série résultante de cet agrégation.

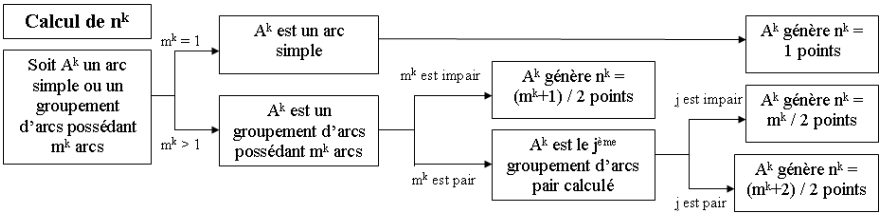

FIG. 5 – Algorithme 1

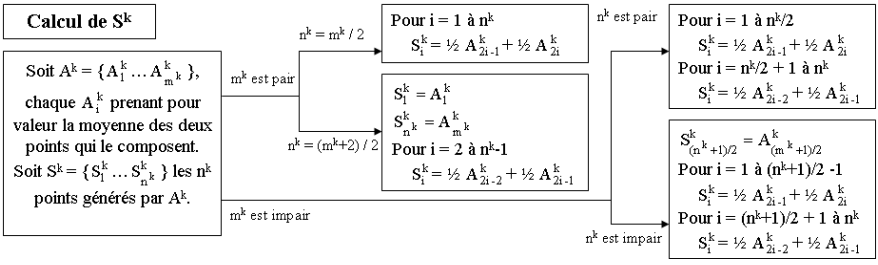

FIG. 6 – Algorithme 2

- Soit $A = \{A^1, A^2, ..., A^k, ..., A^p\}$ l'ensemble des arcs simples ou groupement d'arcs issus de la mesure de similarité DTW entre Q_1 et Q_2.
- Soit m^k le nombre d'arcs que possède A^k (si A^k est un arc simple alors $m^k = 1$, si A^k est un groupement d'arcs alors $m^k > 1$).
- Soit n^k le nombre de points de S générés par A^k.
- Soit $S^k = \{S^k_1, S^k_2, ..., S^k_{n^k}\}$ l'ensemble des n^k points de S générés par A^k.

- Pour chaque A^k :
  - Soit $A^k = \{A^k_1, A^k_2, ..., A^k_i, ..., A^k_{m^k}\}$ l'ensemble des m^k arcs qui composent A^k. A^k_i est le i^{eme} arc de A^k et prend pour valeur la moyenne des deux points qui le composent.
  - Etablir le nombre n^k de points que générera A^k dans S à l'aide de l'algorithme 1 (fig. 5).
  - Calculer la valeur des n^k points de S^k générés par les m^k arcs de A^k à l'aide de l'algorithme 2 (fig. 6).

- On obtient alors $S = \{S^1, S^2, ..., S^k, ..., S^p\}$ la série générée par l'agrégation de Q_1 et Q_2.

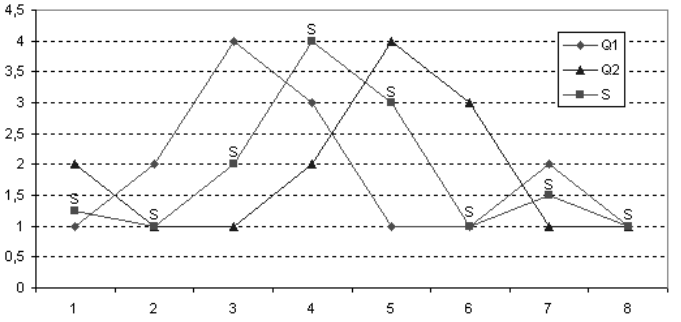

FIG. 7 – Nouvelle agrégation basée sur les arcs associatifs DTW

Si à présent on calcule l'agrégation S des séries Q_1 et Q_2 définies plus haut à l'aide de cet algorithme, on trouve le résultat suivant : $S = \{1.25, 1, 2, 4, 3, 1, 1.5, 1\}$. La figure 7 présente le résultat graphique de cette agrégation.

On constate en observant la figure 7 que la série S obtenue à l'aide de notre nouvelle méthode conserve les formes de chacune des deux séries originelles. Cette série ressemble à l'agrégation qu'aurait naturellement dessiné un être humain à la vue des deux séries Q_1 et Q_2. Notre méthode sait gerer les décalages temporels de la même manière que le DTW, elle nous semble donc plus appropriée que l'agrégation Euclidienne pour effectuer la ré-estimation des centres de chaque cluster dans l'algorithme des *k-Means* adaptés aux séries temporelles (étape 4 de l'algorithme).

4 Résultats obtenus

Afin de tester cette nouvelle méthode d'agrégation (que nous avons appellée "Agrégation basée sur les Arcs Dynamic Time Warping" ($AADTW$)), nous avons implémenté les trois méthodes d'apprentissage non supervisé suivantes :

1. *k-Means* avec mesure de similarité *p-normée* + agrégation Euclidienne
2. *k-Means* avec mesure de similarité DTW + agrégation Euclidienne
3. *k-Means* avec mesure de similarité DTW + $AADTW$

Pour effectuer les tests, nous avons utilisé six échantillons de cent séries temporelles chacun (librement mis à disposition par Eammon Keogh (Keogh et Folias 2002)). Chaque série possède soixante valeurs. Chaque échantillon est constitué d'un seul type de séries spécifique : Séries aléatoires générées artificiellement, séries cycliques, séries croissantes régulières, séries croissantes par seuil, séries décroissantes régulières et séries décroissantes par seuil. Les séries ont été préalablement normalisées et lissées à l'aide de la moyenne mobile (le lissage améliore sensiblement les résultats dans de nombreux cas d'apprentissage sur la forme).

Jeu de données	*p-normée*+Euclide	DTW+Euclide	DTW+$AADTW$
20×4 catégories[2]	46.25 %	1.25 %	1.25 %
10×6 catégories[3]	46.6 %	23.3 %	10 %
100×2 catégories[4]	42 %	48 %	38.5 %

TAB. 1 – Taux d'erreur de classification obtenus par chacune des trois variantes

Nous avons exécuté ces trois variantes des *k-Means* sur plusieurs jeux de tests différents et nous avons confronté les résultats obtenus par chacune d'entre-elles. Chaque variante bénéficiait de 100 essais par mesure, chaque essai bénéficiant d'un maximum de 20 itérations. Le paramétrage de la fenêtre temporelle *delta* pour les variantes 2 et 3 (*i.e.* "DTW + Euclide" et "DTW + $AADTW$") est de 6 (= 10% de la taille des séries).

Les résultats de ces tests sont présentés dans le tableau 1. La premier test montre tout d'abord la supériorité que peut avoir la mesure DTW vis-à-vis de la mesure Euclidienne dans le cadre d'un apprentissage non supervisé. Les deux tests suivants mettent en valeur l'amélioration des performances que peut apporter la variante "DTW + $AADTW$" vis-à-vis de la variante "DTW + Euclide". Ces améliorations sont statistiquement significatives (la p-value du second test est égale à 2.33 %, celle du troisième test est égale à 2.68 %).

5 Conclusions et perspectives

Nous avons développé une nouvelle méthode d'agrégation de séries temporelles basée sur la mesure de similarité *Dynamique Time Warping*. Cette méthode peut être associée à l'algorithme des *k-Means* afin d'amméliorer cette méthode d'apprentissage non supervisé adaptée aux séries temporelles.

Notre méthode d'agrégation est surtout efficace pour fusionner des séries qui sont déjà à l'origine très proches. Si celles-ci sont très différentes, l'agrégation perd en signification. Dans l'algorithme des *k-Means*, l'initialisation aléatoire des clusters conduit notre méthode à devoir agréger au début des séries très dissemblables et cela handicape le processus d'apprentissage. Nous pensons donc pouvoir améliorer notre méthode en résolvant les problèmes suivants :

- Effectuer un pré-apprentissage afin de repérer l'initialisation des clusters qui amènera notre méthode d'agrégation à fusionner les séries les plus pertinentes.
- Modifier notre méthode d'agrégation lorsqu'elle doit fusionner des séries très dissemblables en lui interdisant d'effectuer des écarts temporels trop grands.
- Adapter notre méthode d'agrégation à la mesure de similarité *Derivated Dynamic Time Warping* (Keogh et Pazzani 2001) qui est une version amméliorée de la mesure *Dynamic Time Warping*.

[2]On choisit aléatoirement : 20 séries aléatoires, 20 séries cycliques, 20 séries croissantes par seuil et 20 séries décroissantes par seuil.

[3]On choisit aléatoirement 10 séries de chacune des 6 catégories.

[4]100 séries croissantes régulières et 100 séries croissantes par seuil.

Références

Agrawal R., Lin K. I., Sawhney H. S. et Shim K. (1995), Fast Similarity Search in the Presence of Noise, Scaling, and Translation in Time Series Databases, In Proc. 21th International Conference on Very Large Database (VLDB-95) 1995.

Bellman R. (1957), Dynamic Programming, Princeton University Press, New Jersey, 1957.

Berndt Donald J. et Clifford James (1994), Using Dynamic Time Warping to Find Patterns in Time Series, KDD Workshop 1994.

Berndt Donald J. et Clifford James (1996), Finding Patterns in Time Series : A Dynamic Programming Approach, Advances in Knowledge Discovery and Data Mining 1996, pp 229-248.

Das G., Gunopulos D. et Manilla H. (1997), Finding Similar Time Series, In Principles of Data Mining and Knowledge Discovery in Databases (PKDD) Trondheim, Norway, 1997.

Keogh Eamonn, Lonardi Stefano et Chiu Bill (2002), Finding Surprising Patterns in a Time Series Database in Linear Time and Space, In the 8th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. July 23 - 26, 2002. Edmonton, Alberta, Canada. pp 550-556.

Keogh Eamonn et Pazzani Michael J. (1998), An enhanced representation of time series which allows fast and accurate classification, clustering and relevance feedback, In 4th International Conference on Knowledge Discovery and Data Mining. New York, NY, Aug 27-31. pp 239-243.

Keogh Eamonn et Pazzani Michael J. (2001), Derivative Dynamic Time Warping, In First SIAM International Conference on Data Mining (SDM'2001), Chicago, USA.

Keogh E. et Folias T. (2002), The UCR Time Series Data Mining Archive [http :// www.cs.ucr.edu/~eamonn/TSDMA/index.html], Riverside CA, University of California - Computer Science & Engineering Department.

Lin Jessica, Keogh Eamonn, Lonardi Stefano et Patel Pranav (2002), Finding Motifs in Time Series, In proceedings of the 2nd Workshop on Temporal Data Mining, at the 8th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. Edmonton, Alberta, Canada. July 23-26, 2002.

Lin Jessica, Vlachos Michail, Keogh Eamonn et Gunopulos Dimitrios (2004), Iterative Incremental Clustering of Time Series, In proceedings of the IX Conference on Extending Database Technology (EDBT 2004). Crete, Greece. March 14-18, 2004.

Ratanamahatana Chotirat Ann et Keogh Eamonn (2004 a), Making Time-series Classification More Accurate Using Learned Constraints, In proceedings of SIAM International Conference on Data Mining (SDM '04), Lake Buena Vista, Florida, April 22-24, 2004. pp. 11-22.

Ratanamahatana Chotirat Ann et Keogh Eamonn (2004 b), Using Relevance Feedback in Multimedia Databases, In proceedings of the International Conferences of VISual Information System (VIS '04) San Francisco, CA, Sept 8-10, 2004.

Salvador Stan (2004), Learning States for Detecting Anomalies in Time Series, Master's thesis submitted to the College of Engineering at Florida Institute of Technology, Pas encore publié.

Vlachos Michail, Kollios Georges, Gunopulos Dimitrios (2002 a), Discovering Similar Multidimensional Trajectories, In Proc. of 18th International Conference on Data Engineering (ICDE), pp. 673-684, San Jose, CA, 2002.

Vlachos Michail, Kollios Georges, Gunopulos Dimitrios (2002 b), Robust Similarity Measures for Mobile Object Trajectories, In Proc. of 13th Database and Expert Systems Applications (DEXA), pp. 721-726, 5th International Workshop "Mobility in Databases and Distributed Systems" (MDDS), Aix-en-Provence, France, 2002.

Vlachos Michail, Hadjieleftheriou Marios, Gunopulos Dimitrios, Keogh Eamonn (2003), Indexing MultiDimensional TimeSeries with Support for Multiple Distance Measures, In Proc. of 9th International Conf. on Knowledge Discovery & Data Mining (SIGKDD), Washington, DC, 2003.

Yazdani N., Bozkaya T. et Ozsoyoglu Z.M. (1997), Matching and Indexing Sequences of Different Lengths, Proc. 1997 ACM CIKM, Sixth International Conference on Information and Knowledge Management, Las Vegas, Nevada, Nov. 1997.

Summary

To use unsupervised clustering algorithm such *k-Means* algorithm on a time series dataset, we need to ask two questions : which time series distance measures may we choose and which time series merging method can we use to estimate the k cluster center. To answer the first question, we present here the main existing distance measures and we explain why one of them (called *Dynamic Time Warping*) seems more efficient than others for time series unsupervised clustering. The second question is more difficult because we need a merging method that respect the so specific characteristics of *Dynamic Time Warping*. We think that the use of such a sophisticated distance measure as *Dynamic Time Warping* with such a "basic" merging method as *Euclidian* merging can disturb the results of a clustering operation based on series' shape. So we propose in this paper an original time series merging method which is compatible with *Dynamic Time Warping* and which improves the results obtained with *k-Means* algorithm.

Keywords : Unsupervised learning and clustering, Time series, k-Means, Dynamic Time Warping