Clustering contraint par apprentissage profond appliqué aux séries temporelles d'images satellites

Baptiste Lafabregue[*,**] Jonathan Weber[*]
Pierre Gançarski[**], Germain Forestier[*]

[*]IRIMAS, University of Haute-Alsace, Mulhouse, France
<prenom>.<nom>@uha.fr,
https://www.irimas.uha.fr
[**]ICube, University of Strasbourg, Strasbourg, France
<nom>@unistra.fr
https://icube.unistra.fr

Résumé. Les avancées dans l'imagerie satellitaire ont généré un nombre sans précédent d'images de télédétection. Les derniers satellites fournissent des images avec une haute fréquence de revisite et très facilement accessibles. Les séries d'images ainsi acquises, sur une même région, peuvent être vues comme des séries temporelles. L'analyse de telles données permet de faire une observation continue et à grande échelle de la terre, avec des applications très diverses, allant de l'occupation du sol en agriculture, au suivi de catastrophe environnementales. Cependant, le manque d'une large quantité de données labellisée empêche d'utiliser directement des méthodes supervisées. A l'opposé, les méthodes non-supervisées ne requièrent aucune connaissance de l'expert, mais donnent souvent des résultats mitigés, ou qui ne correspondent pas aux attentes de l'expert. Dans ce contexte, le clustering contraint, qui est une forme d'algorithme d'apprentissage semi-supervisé, est une alternative offrant un compromis intéressant. Dans cet article, nous explorons l'utilisation de contraintes au sein de méthodes de clustering basées sur l'apprentissage profond appliquées aux séries temporelles d'images satellites. Notre étude expérimentale repose sur la méthode Deep Embedded Clustering et son adaptation qui intègre des contraintes par paires (must-link et cannot-link). Les tests conduits sur un jeu de données composé de 11 images satellites montrent des résultats encourageants et ouvrent de nombreuses perspectives dans l'application aux séries temporelles d'images satellites du clustering contraint basé sur de l'apprentissage profond.

1 Introduction

Les méthodes d'apprentissage profond sont largement utilisées dans un grand nombre de domaines, et font l'objet d'un intérêt grandissant dans la communauté de la télédétection Zhu et al. (2017). Ces méthodes donnent de très bons résultats, mais elles sont fortement dépendantes de la quantité de données disponibles, et plus particulièrement de données labellisées.

La télédétection produit un grand nombre de données, qui sont cependant rarement annotées. Ce manque d'annotation est encore plus marqué pour les séries d'images satellites. Des images satellites peuvent être librement obtenues tous les 5 jours, cependant, de par la complexité des données et le manque de typologie bien définie, il est difficile d'utiliser des approches supervisées. De ce fait, de nombreuses méthodes de clustering sont appliquées dans ce domaine Khiali et al. (2019); Rey et al. (2019). Cependant, même en l'absence de données annotées, l'expert a très souvent une forte connaissance thématique qu'il peut mettre à disposition. Les contraintes permettent d'intégrer une partie de cette connaissance dans le processus d'apprentissage, nos travaux se concentrent uniquement sur les contraintes par paires, les must-link (ML) et les cannot-link (CL). Ces contraintes indiquent que deux instances doivent être assignées au même cluster (must-link) ou à des clusters différents (cannot-link). Les contraintes par paires sont largement utilisées et ont déjà fait l'objet de nombreuses études Basu et al. (2008), ce qui nous permet de facilement obtenir une base de comparaison, ce type de contraintes étant supportées par de nombreuses méthodes.

Dans cet article, nous voulons étudier si on peut tirer avantage des avancées en apprentissage profond à travers une approche basée contrainte qui semble plus appropriée dans le domaine de la télédétection. Après avoir présenté les travaux antérieurs en clustering sur les séries temporelles et le clustering contraint, respectivement dans les sections 2.1 et 2.2, nous présenterons la méthode de clustering contraint d'apprentissage profond utilisée et son adaptation aux séries temporelles 3, puis nous comparerons les résultats à d'autres méthodes classiques de clustering contraint sur des données de télédétection dans la section 4. Enfin, dans la section 5 nous commenterons les résultats obtenus et les perspectives qui en résultent.

2 État de l'art

2.1 Le clustering pour les séries temporelles

Différentes approches de clustering pour les séries temporelles ont été proposées dans la littérature, essentiellement basées sur des méthodes de représentation tel que la transformée en ondelettes discrète Chan et Fu (1999) ou des mesures de similarité tel que Dynamic Time Warping Sakoe et Chiba (1978). Ces méthodes sont ensuite généralement intégrées comme prétraitement, le résultat étant alors utilisé pour alimenter une méthode standard de clustering, tel que les familles de méthodes k-means, k-medoid, clustering spectral ou encore clustering hiérarchique, comme illustré dans Aghabozorgi et al. (2015). Dans ce domaine, de nouvelles propositions sont faites, e.g., la méthode k-shape Paparrizos et Gravano (2015), qui est basée sur une procédure de raffinement itératif qui utilise une version normalisée de la corrélation croisée. Une des difficultés, lors de l'utilisation de séries temporelles, est l'hétérogénéité des sujets abordés et du type des données traitées, allant du nombre d'attributs ou de la longueur de la séquence, au type de corrélation entre éléments, basée sur la forme ou la structure, avec à chaque fois des amplitudes différentes. Cette problématique est largement traitée en apprentissage supervisé par l'utilisation de l'apprentissage de représentation à travers des réseaux de neurones profonds. Récemment, des approches de clustering utilisant ces méthodes d'apprentissage ont été proposées. Elles sont essentiellement basées sur des architectures *end-to-end* qui apprennent simultanément une représentation des données et un clustering, en utilisant un autoencodeur et une couche de clustering attachée à la sortie de l'encodeur Xie et al. (2016);

Guo et al. (2017). Une architecture dérivée de cette dernière a été développée pour les séries temporelles, elle utilise une couche convolutive 1-D suivie par un Bi-LSTM comme autoencodeur, afin de préserver la dimension temporelle dans l'encodage de la représentation, un clustering est alors appliqué sur cette représentation en utilisant une métrique de similarité comme couche de clustering Madiraju et al. (2018).

2.2 Clustering contraint

De nombreuses méthodes ont été proposées pour l'intégration de contraintes en clustering. La plupart d'entre elles consistent en l'extension d'algorithmes standards de clustering tel que k-means Wagstaff et al. (2001) ou du spectral clustering Li et al. (2009), mais il existe aussi des méthodes dédiées, comme par exemple la programmation par contrainte Duong et al. (2017). Une étude comparative a d'ailleurs été menée sur ce sujet Lampert et al. (2018). Dans le domaine de l'apprentissage profond, la plupart des méthodes semi-supervisées font référence à des méthodes d'auto-apprentissage ou à d'autres moyens d'intégrer des connaissances dans une tâche supervisée. Mais récemment, des méthodes ont été proposées pour inclure des contraintes par paires dans du clustering basé sur l'apprentissage profond Zhang et al. (2019); Ren et al. (2019). Ces deux articles utilisent les contraintes au niveau de leur fonction de coût qui va alors maximiser la similarité entre les encodages d'instances d'une contrainte must-link et respectivement la minimiser pour une contrainte cannot-link. Notre travail se base sur les travaux de Zhang et al. (2019). Nos contributions consistent en l'adaptation de cette approche aux séries temporelles et à l'étude des résultats sur des séries temporelles d'images satellites. La méthode proposée par Zhang et al. (2019) supporte plusieurs types de contraintes, mais, comme précisé précédemment, nos travaux se concentrent sur les contraintes ML et CL.

3 La méthode Deep Constrained Clustering et son adaptation aux séries temporelles

La méthode Deep Constrained Clustering (DCC) proposée par Zhang et al. (2019) est basée sur une méthode de clustering par apprentissage profond, Deep Embedded Clustering (DEC) (Xie et al., 2016) et sa version améliorée (IDEC) (Guo et al., 2017). Dans un premier temps, nous présenterons la méthode IDEC, puis son extension pour l'intégration de contraintes par la méthode DCC et finalement les adaptations apportées pour l'utilisation sur des séries temporelles.

3.1 Improved Deep Embedded Clustering

Deep Embedded Clustering (DEC), lors de la phase initiale, entraîne un autoencodeur ($x_i = g(f(x_i))$) puis supprime le décodeur. Le réseau ainsi obtenu, qui consiste en l'encodeur ($z_i = f(x_i)$), est affiné en optimisant la divergence de Kullback-Leiber entre deux distributions Q et P. Q est un *soft cluster assignment* , où pour chaque instance i on calcule un vecteur q_i de longueur k, k étant le nombre de clusters souhaités, où q_{ij} est le degré de confiance que l'instance i appartienne au cluster j. P est la distribution cible qui est définie en fonction de Q pour renforcer la prédiction faite pour chaque cluster, comme défini ci-dessous. Nous obtenons

donc la fonction de coût de clustering L_c :

$$L_c = KL(P|Q) = \sum_i \sum_j p_{ij} log \frac{p_{ij}}{q_{ij}} \tag{1}$$

où q_{ij} est la similarité entre l'encodage de x_i, z_i et le centroïde du cluster j, μ_j, mesurée par une t-distribution de Student (Maaten et Hinton, 2008) :

$$q_{ij} = \frac{(1 + ||z_i - \mu_j||^2)^{-1}}{\sum_j (1 + ||z_i - \mu_j||^2)^{-1}} \tag{2}$$

et p_{ij} est la distribution cible, tel que :

$$p_{ij} = \frac{q_{ij}^2 / \sum_i q_{ij}}{\sum_j (q_{ij}^2 / \sum_i q_{ij})} \tag{3}$$

L'ensemble des centroïdes μ est initialisé avec les centres d'un k-means exécuté sur la représentation z. L'amélioration apportée par IDEC est de garder le décodeur et la fonction de coût de reconstruction de l'autoencodeur L_r même après la phase initiale. L'intuition derrière cela est que la fonction de coût du clustering, en distordant l'espace de représentation, peut altérer la représentativité de l'encodage généré et de ce fait la performance du clustering. Ainsi, la fonction de coût de clustering augmente la séparabilité des clusters, tandis que la fonction de coût de reconstruction maintient la représentativité de l'encodage appris par l'autoencodeur. La fonction de coût de reconstruction correspond à l'erreur quadratique moyenne entre l'instance en entrée et sa reconstruction faite par l'autoencodeur. On a donc la fonction de coût global qui est définie comme suit :

$$L = L_r + \gamma * L_c \tag{4}$$

où $\gamma > 0$ est le coefficient qui contrôle le degré de distorsion de l'espace de représentation.

3.2 Intégration des contraintes

L'extension de DEC pour incorporer les contraintes est basée sur la méthode Deep Constrained (DCC). Ils proposent quatre types de contraintes, mais nous n'avons pris en considération que les contraintes par paires, car elles sont supportées par de nombreux autres méthodes de clustering contraint.

La fonction de coût utilisée pour l'ensemble ML des contraintes must-link est définie par :

$$l_{ML} = L_r - \gamma_{ML} * \sum_{(a,b) \in ML} log \sum_j q_{aj} * q_{bj} \tag{5}$$

De manière équivalente, la fonction de coût pour l'ensemble CL des contraintes cannot-link est définie par :

$$l_{CL} = - \sum_{(a,b) \in CL} log(1 - \sum_j q_{aj} * q_{bj}) \tag{6}$$

De manière intuitive, la fonction de coût pour les ML va favoriser des instances avec le même *soft assignment* et celle pour les CL qui ont le *soft assignment* opposé. La fonction de coût pour les ML est régularisée par l'ajout de la fonction de coût de reconstruction L_r pondérée par un facteur $\gamma_{ML} > 0$, de manière identique à L_c, afin d'éviter de tomber dans une solution triviale qui serait d'assigner toutes les instances concernées à un seul cluster.

3.3 Application aux séries temporelles d'images satellites

L'objectif principal de ces expériences est d'évaluer si ce nouveau type de clustering contraint est pertinent sur des séries temporelles d'images satellites, et de le comparer aux méthodes de l'état de l'art. Nous avons testé la version originale de DCC, qui est composée de couches totalement connectées. Pour cette version, l'architecture est identique mais les séries temporelles multivariées sont réduites à une dimension. Nous proposons également une version modifiée de DCC avec des convolutions 1D, ces dernières ayant montré leur efficacité pour les séries temporelles en classification supervisée Fawaz et al. (2019). Dans cette nouvelle version, nous gardons la dimension originale des séries temporelles et le réseau est composé uniquement de couches convolutionnelles 1D suivies à chaque fois d'une couche de *batch-normalisation*, une couche de *global average pooling* est placée à la fin juste avant la couche finale d'encodage. Cette dernière reste une couche totalement connectée.

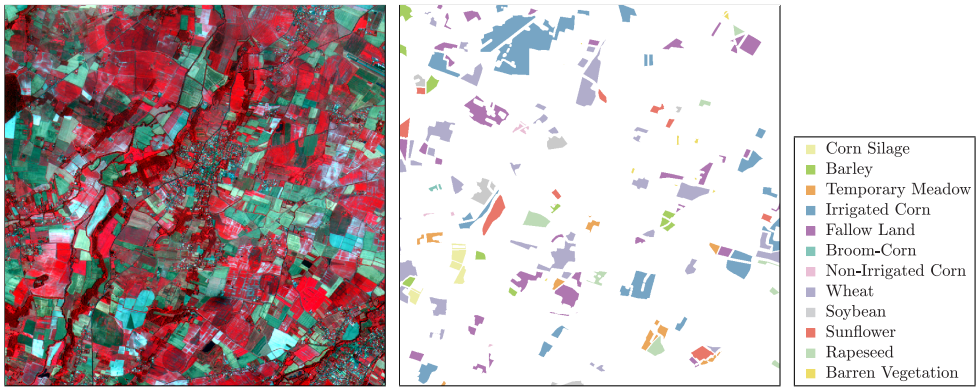
4 Expériences et résultats

Pour ces expérimentations, nous avons appliqué ces méthodes sur un problème de classification de cultures agricoles qui est un champ de recherche important en télédétection et qui fait l'objet de nombreuses études (Sicre et al., 2014; Garnot et al., 2019).

4.1 Jeu de données et paramètres expérimentaux

Le jeu de données est composé de 12 classes de cultures agricoles (blé, maïs irrigué, etc. voir Fig. 1c), situées près de Toulouse (Sud-Est de la France). Les images d'origine [1] sont composées de 11 images multi-spectrales (vert, rouge, proche-infrarouge) de 1000×1000 pixels réparties de manière non-uniforme entre le 15/02/07 et le 20/10/07 et acquises par le satellite Formosat-2. Une des images est présentée dans la Fig. 1a. Le jeu de données est composé de pixels sélectionnés aléatoirement dans les régions annotées (voir Fig. 1b) qui ont été ensuite répartis en jeu d'entraînement et de test, composés de respectivement 1974 et 9869 série temporelles de pixels. Les algorithmes de clustering sont entraînés et évalués uniquement sur le jeu de test. Le jeu d'entraînement est seulement utilisé pour fixer les hyperparamètres des méthodes si besoin. Les contraintes sont générées depuis le jeu de test en sélectionnant aléatoirement des paires de pixels et en créant une contrainte ML ou CL en fonction de leurs labels. La donnée de référence est basée sur la déclaration des agriculteurs à l'AEE pour la Politique Agricole Commune. Pour tester la sensibilité des méthodes au nombre de contraintes, nous avons défini trois niveaux de taille de jeu de contraintes : 5%, 15% et 50% de la cardinalité du jeu de test $N = 9869$ (une très petite fraction de l'ensemble des contraintes possibles, $\frac{1}{2}N[N-1]$). Pour l'évaluation nous avons utilisé la mesure d'Adjusted Rand Index (ARI) et le taux de satisfaction de contraintes par le résultat du clustering (Sat.) moyennés sur 10 exécutions. Pour chaque niveau de contraintes, 10 ensembles ont été générés aléatoirement, un par exécution. Les mêmes 10 ensembles sont utilisés pour chaque méthode, pour s'assurer que chaque méthode bénéficie des mêmes contraintes.

1. Mises à disposition par le *Centre d'Études Spatiales de la Biosphère (CESBIO) Unité Mixte de Recherche CNES-CNRS-IRD-UPS*, Toulouse, France.



(a) Image (b) Donnée de référence (c) Noms des classes

FIG. 1 – *Une image de la série temporelle :* 12 *classes, et* 11 *dates (*t_4 *affiché).*

4.2 Méthodes de comparaisons et paramétrisation

Afin d'avoir une base de comparaison, nous avons ajouté à DCC et DCC-Conv, quatre méthodes standards de clustering par contraintes. Nous avons utilisé une variation de k-means contraint (COP-KMeans) Wagstaff et al. (2001), un algorithme de clustering spectral contraint (Spec) Li et al. (2009), une méthode déclarative de programmation par contraintes (CPClustering) Duong et al. (2017) et une méthode de clustering collaboratif contraint (SAMARAH) Forestier et al. (2010) (utilisant 3 k-means). Pour illustrer la variabilité induite par le choix de la métrique sélectionnée, nous avons utilisé la distance Euclidienne et la métrique DTW Sakoe et Chiba (1978) [2]. Mis à part CPClustering, qui ne requiert aucun paramètre, les autres méthodes nécessitent au moins le nombre de clusters, sinon les paramètres par défaut ont été utilisés. La seule exception est la méthode Spec qui nécessite l'apprentissage d'hyperparamètres appris par grid search sur le jeu d'entraînement. Pour les méthodes de clustering par apprentissage profond, nous avons suivi le paramétrage proposé dans DEC et IDEC mais comme les résultats n'étaient pas stables (voir section 4.3 pour plus de détails) nous avons fait des modifications mineures. Pour la dimension de la couche d'encodage nous l'avons fixé à 2 au lieu de 10, car cela semblait donner plus de stabilité lors de l'apprentissage. Pour DCC, l'encodeur est fixé aux dimensions $d-500-500-2000-2$, où $d = l*f$, l étant la longueur de la série en entrée et f le nombre d'attributs de la série. Pour DCC-Conv les dimensions sont $l*t-128-256-128-2$, avec respectivement des filtres 1D de dimension $8-5-3$, suivant ainsi les recommandations dans Wang et al. (2017). Pour les deux versions, les dimensions des décodeurs sont en miroir de celles de l'encodeur. Pour les deux, la fonction d'optimisation utilisée est SGD avec un *momentum* de 0.9 et une valeur de *decay* de $1e-6$, pour compenser la variabilité mentionnée précédemment. γ et γ_{ML} sont tous deux fixés à 0.1 comme décrit dans les articles d'origine. [3]

2. Les méthodes utilisées pour la comparaison sont disponibles sur `https://icube-forge.unistra.fr/lampert/TSCC`

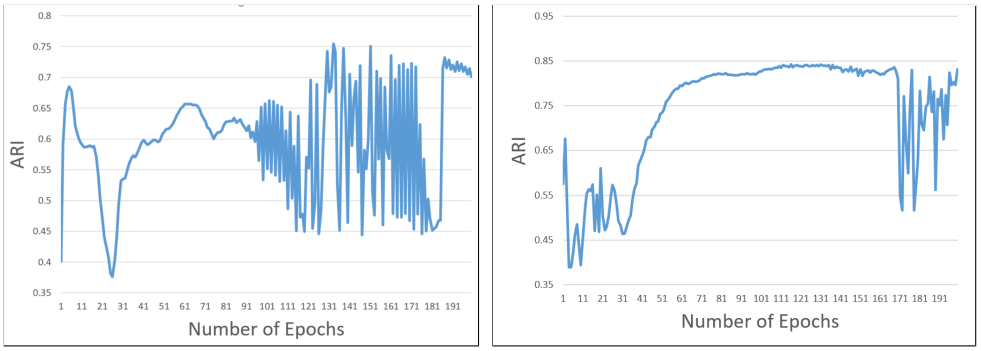
3. Le code utilisé pour cet article peut être trouvé sur : `https://github.com/blafabregue/DeepConstrainedClustering`

TAB. 1 – *ARI et satisfaction des contraintes, avec et sans contraintes. La meilleur performance par pourcentage de contraintes et métrique est mise en gras. La Sat., dans le cas sans contrainte, est mesurée par une moyenne sur les ensembles de contraintes à 50%.*

Méthode	Distance	Sans-contraintes		5%		15%		50%	
		ARI	Sat.	ARI	Sat.	ARI	Sat.	ARI	Sat.
COP-KMeans	DTW	0.426	0.812	0.416	**1.00**	0.407	**1.00**	0.436	**1.00**
Wagstaff et al. (2001)	Eucl.	0.420	0.807	0.406	**1.00**	0.443	**1.00**	0.369	**1.00**
Spec	DTW	**0.531**	**0.840**	**0.683**	0.867	**0.725**	0.888	0.786	0.911
Li et al. (2009)	Eucl.	**0.737**	**0.885**	0.671	0.854	0.702	0.875	0.781	0.916
CPClustering	DTW	0.437	0.803	0.469	**1.00**	0.510	**1.00**	0.589	**1.00**
Duong et al. (2017)	Eucl.	0.681	0.413	0.650	**1.00**	0.542	**1.00**	0.510	**1.00**
SAMARAH	DTW	0.406	0.802	0.597	0.870	0.637	0.867	0.681	0.878
Forestier et al. (2010)	Eucl.	0.463	**0.817**	**0.691**	0.884	**0.714**	0.890	0.702	0.885
DCC Zhang et al. (2019)		0.703	0.885	0.550	0.852	0.448	0.816	0.615	0.862
DCC-Conv		0.508	0.833	0.497	0.844	0.491	0.819	**0.820**	0.936

4.3 Résultats

Les résultats, avec et sans contraintes, sont présentés dans la Table 1. Spec donne globalement les meilleurs résultats, mais cela doit être relativisé, comme mentionné précédemment, par le fait que cette méthode nécessite l'apprentissage d'hyperparamètres (sans contraintes, le résultat moyen est de 0.367 d'ARI, contre 0.737 pour les paramètres retenus). On peut aussi remarquer que les résultats varient fortement selon la métrique utilisée. C'est également le cas pour quasiment toutes les autres méthodes basées métrique. Les versions par apprentissage profond ne font mieux que dans le cas où le nombre de contraintes est très élevé et uniquement pour la version convolutionnelle. Mais le plus surprenant est le comportement quasiment opposé des deux versions. DCC donne de relativement bons résultats sans-contraintes, mais les contraintes ont un fort effet négatif. Cet effet négatif a déjà pu être étudié en clustering contraint et il peut être observé également pour les autres méthodes, à l'exception de SAMARAH. Il a été observé dans Lampert et al. (2018) que si une méthode capture déjà bien la structure de la donnée sans contrainte, elle ne va pas bénéficier de l'ajout de contraintes, ceci peut être mesuré par la Sat. sur l'exécution sans-contrainte. Cela semble être le cas ici, ce qui va en opposition des observations faites par Zhang et al. (2019), qui concluaient en l'absence de cet effet. Dans notre cas, cela peut s'expliquer par la forte présence de bruit dans les contraintes (route à travers les champs, pixels en frontière des champs, ...). DCC-Conv, de son côté, obtient de bons résultats avec les contraintes, mais seulement quand leur nombre est assez élevé. En effet, la manière dont les contraintes sont utilisées lors de la descente de gradient, ne force pas l'algorithme à respecter les contraintes, il semble cependant qu'un grand nombre de contraintes permet de bien faire redescendre l'information dans les poids du réseau. Dans Lampert et al. (2018), il a été observé que les méthodes ne bénéficient pas de manière significative d'un nombre croissant de contraintes, mais plutôt de contraintes plus informatives et cohérentes. Dans le cas de DCC-Conv, nous pouvons légitimement nous demander si le réseau ne commence pas à apprendre le jeu de données lui-même et non la structure de la donnée, l'algorithme étant entraîné et testé sur le même jeu de données (celui de test). Cependant en



(a) DCC avec 50% de contraintes (b) DCC-COnv avec 50% de contraintes

FIG. 2 – *Évolution de l'ARI au cours du processus d'apprentissage de DCC et DCC-Conv.*

utilisant le modèle appris sur le jeu d'entraînement nous avons remarqué que nous obtenons un score très proche (0.81 d'ARI contre 0.82 sur le test), ce qui relativise cette explication.

Deux autres points doivent être abordés, qui ne sont pas directement visibles dans le tableau 1. Tout d'abord l'écart type augmente fortement avec l'ajout de contraintes et tend à diminuer quand le nombre de contraintes augmente, que ce soit pour DCC ou DCC-Conv (i.e. pour DCC-Conv, l'écart type est respectivement, sans contrainte, avec 5%, 15%, 50% de 0.005, 0.069, 0.010 et 0.015). L'effet des contraintes peut donc être une forte source de perturbation, cela peut s'expliquer par le bruit dans notre cas, mais quand ce nombre augmente cela régularise cet effet. Le second point est que le réseau n'est pas stable durant l'apprentissage, c'est le cas pour les deux versions, mais d'une manière plus amplifiée pour DCC. Même si nous avons pu réduire cette variabilité en intégrant un *decay* dans la fonction d'optimisation, celle-ci reste importante (i.e. pour DCC, l'ARI varie entre 0.55 et 0.74 avec un cas extrême à 0.45). Une illustration de cette instabilité peut être vu sur la figure 2. Cela se produit essentiellement quand des contraintes sont ajoutées, mais aussi sans-contrainte, mais dans une plus faible amplitude. Cela semble venir en partie du fait que la distribution cible est mise à jour à chaque époque, le réseau optimise donc vers une cible qui elle-même peut potentiellement fortement varier.

5 Conclusion

Le clustering par apprentissage profond démontre qu'il peut obtenir de bons résultats sur des séries temporelles en télédétection, avec ou sans contraintes. De plus, ces méthodes permettent de s'affranchir du choix d'une représentation ou d'une métrique, celle-ci étant apprise par l'autoencodeur, ce qui rend la tâche de l'expert du domaine plus simple. Cependant les résultats montrent que le choix des hyperparamètres (i.e. dimensions des couches, plus particulièrement celle de l'encodage, choix de la fonction d'optimisation) est important et requiert de plus amples investigations, ce qui vient contrebalancer en partie cet avantage. Il y a également deux points importants que nous souhaitons approfondir. Tout d'abord, DCC n'est pas aussi robuste que ne le laissaient penser les expériences précédentes. Second point, l'instabilité lors de la phase d'apprentissage et l'impact des contraintes dégradent fortement la moyenne

des résultats. Nous planifions d'étudier les facteurs qui entraînent ces problèmes dans notre cas, ainsi que l'effet du bruit dans les contraintes ou voir si cela se généralise à d'autres types de séries temporelles (domaine différent, taille du jeu de donnée, longueur des séries, ...).

Remerciements

Ces travaux ont été financés dans le cadre de l'ANR TIMES (financement ANR-17-CE23-0015) de l'Agence Nationale de la Recherche.

Références

Aghabozorgi, S., A. S. Shirkhorshidi, et T. Y. Wah (2015). Time-series clustering–a decade review. *Information Systems 53*, 16–38.

Basu, S., I. Davidson, et K. Wagstaff (2008). *Constrained clustering : Advances in algorithms, theory, and applications*. CRC Press.

Chan, K.-P. et A. W.-C. Fu (1999). Efficient time series matching by wavelets. In *Proceedings 15th International Conference on Data Engineering (Cat. No. 99CB36337)*, pp. 126–133. IEEE.

Duong, K.-C., C. Vrain, et al. (2017). Constrained clustering by constraint programming. *Artificial Intelligence 244*, 70–94.

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*, 1–47.

Forestier, G., P. Gançarski, et C. Wemmert (2010). Collaborative clustering with background knowledge. *Data & Knowledge Engineering 69*(2), 211–228.

Garnot, V. S. F., L. Landrieu, S. Giordano, et N. Chehata (2019). Time-space tradeoff in deep learning models for crop classification on satellite multi-spectral image time series. *arXiv preprint arXiv :1901.10503*.

Guo, X., L. Gao, X. Liu, et J. Yin (2017). Improved deep embedded clustering with local structure preservation. In *IJCAI*, pp. 1753–1759.

Khiali, L., M. Ndiath, S. Alleaume, D. Ienco, K. Ose, et M. Teisseire (2019). Detection of spatio-temporal evolutions on multi-annual satellite image time series : A clustering based approach. *International Journal of Applied Earth Observation and Geoinformation 74*, 103–119.

Lampert, T., B. Lafabregue, N. Serrette, G. Forestier, B. Crémilleux, C. Vrain, P. Gancarski, et al. (2018). Constrained distance based clustering for time-series : a comparative and experimental study. *Data Mining and Knowledge Discovery 32*(6), 1663–1707.

Li, Z., J. Liu, et X. Tang (2009). Constrained clustering via spectral regularization. In *2009 IEEE Conference on Computer Vision and Pattern Recognition*, pp. 421–428. IEEE.

Maaten, L. v. d. et G. Hinton (2008). Visualizing data using t-sne. *Journal of machine learning research 9*(Nov), 2579–2605.

Madiraju, N. S., S. M. Sadat, D. Fisher, et H. Karimabadi (2018). Deep temporal clustering : Fully unsupervised learning of time-domain features. *arXiv preprint arXiv :1802.01059*.

Paparrizos, J. et L. Gravano (2015). k-shape : Efficient and accurate clustering of time series. In *Proceedings of the 2015 ACM SIGMOD International Conference on Management of Data*, pp. 1855–1870. ACM.

Ren, Y., K. Hu, X. Dai, L. Pan, S. C. Hoi, et Z. Xu (2019). Semi-supervised deep embedded clustering. *Neurocomputing 325*, 121–130.

Rey, D. M., M. Walvoord, B. Minsley, J. Rover, et K. Singha (2019). Investigating lake-area dynamics across a permafrost-thaw spectrum using airborne electromagnetic surveys and remote sensing time-series data in yukon flats, alaska. *Environmental Research Letters 14*(2), 025001.

Sakoe, H. et S. Chiba (1978). Dynamic programming algorithm optimization for spoken word recognition. *IEEE transactions on acoustics, speech, and signal processing 26*(1), 43–49.

Sicre, C. M., F. Baup, et R. Fieuzal (2014). Determination of the crop row orientations from formosat-2 multi-temporal and panchromatic images. *ISPRS journal of photogrammetry and remote sensing 94*, 127–142.

Wagstaff, K., C. Cardie, S. Rogers, S. Schrödl, et al. (2001). Constrained k-means clustering with background knowledge. In *Icml*, Volume 1, pp. 577–584.

Wang, Z., W. Yan, et T. Oates (2017). Time series classification from scratch with deep neural networks : A strong baseline. In *2017 International joint conference on neural networks (IJCNN)*, pp. 1578–1585. IEEE.

Xie, J., R. Girshick, et A. Farhadi (2016). Unsupervised deep embedding for clustering analysis. In *International conference on machine learning*, pp. 478–487.

Zhang, H., S. Basu, et I. Davidson (2019). Deep constrained clustering-algorithms and advances. *arXiv preprint arXiv :1901.10061*.

Zhu, X. X., D. Tuia, L. Mou, G.-S. Xia, L. Zhang, F. Xu, et F. Fraundorfer (2017). Deep learning in remote sensing : A comprehensive review and list of resources. *IEEE Geoscience and Remote Sensing Magazine 5*(4), 8–36.

Summary

The advent of satellite imagery is generating an unprecedented amount of remote sensing images. Current satellites now achieve frequent revisits and high mission availability and provide series of images of the Earth captured at different dates that can be seen as time series. Analyzing satellite image time series allows to perform continuous wide range Earth observation with applications in agricultural mapping, environmental disaster monitoring, etc. However, the lack of large quantity of labeled data generally prevents from easily applying supervised methods. On the contrary, unsupervised methods do not require expert knowledge but sometimes provide poor results. In this context, constrained clustering, which is a class of semi-supervised learning algorithms, is an alternative and offers a good trade-off of supervision. In this paper, we explore the use of constraints with deep clustering approaches to process satellite image time series. Our experimental study relies on deep embedded clustering and the deep constrained framework using pairwise constraints (must-link and cannot-link). Experiments on a real dataset composed of 11 satellite images show promising results and open many perspectives for applying deep constrained clustering to satellite image time series.