Atelier Apprentissage Profond : Théorie et Applications (APTA)

Atelier organisé dans le cadre de la conférence EGC 2021

26 janvier 2021 Montpellier (France)

Organisateurs

- Maxime Devanne, MCF, Université de Haute-Alsace, IRIMAS
- Camille Kurtz, MCF, Université Paris Descartes, LIPADE
- Jonathan Weber, MCF, Université de Haute-Alsace, IRIMAS
- Cédric Wemmert, PU, Université de Strasbourg, ICUBE
- Germain Forestier, PU, Université de Haute-Alsace, IRIMAS

ICUBE IRIMAS UNIVERSITÉ HAUTE-ALSACE Institut de Recherche en Informatique, Mathématiques, Automatique et Signal LIPADE Laboratoire d'Informatique PAris DEscartes

L'ATELIER APTA

L'apprentissage profond révolutionne depuis quelques années l'apprentissage machine. Si les premiers résultats marquants ont été obtenus principalement en analyse d'images, les travaux actuels en apprentissage profond (deep learning) s'intéressent à présent à tous les types de données et presque tous les types de traitement (classification de séries temporelles, augmentations de données, analyse de texte, etc.). Son impact dans le domaine de la science des données et l'extraction de connaissances est considérable

Nous avons souhaité proposer dans le cadre de la conférence EGC 2021 un espace d'échanges autour de ce domaine, permettant d'aborder les défis théoriques et les possibilités applicatives offertes à notre discipline de l'extraction et de la gestion des connaissances. Dans le cadre de cet atelier, nous nous focalisons sur les applications de l'apprentissage profond dans différents domaines (analyse ou génération d'images, classification de données temporelles, extraction d'informations à partir de données hétérogènes, etc.) mais également permettre la présentation de travaux plus théoriques (nouvelles architectures, nouvelles fonctions de coût, interprétabilité des modèles, etc).

L'objectif de cet atelier est d'offrir un espace d'échange entre d'une part, des experts de l'apprentissage profond, développant de nouveaux modèles et éventuellement à la recherche de domaines d'application pour les valider, et d'autre part, des utilisateurs avec moins d'expérience et souhaitant appliquer les méthodes d'apprentissage profond à leurs données. Cet espace peut permettre également aux potentiels nouveaux utilisateurs, curieux de ces approches d'en comprendre les avantages et les limites.

CONTENU SCIENTIFIQUE DE L'ATELIER

5 propositions originales ont été retenues par le comité de programme de l'atelier. Nous avons été en mesure de proposer pour l'ensemble des propositions deux rapports d'experts afin d'offrir un processus scientifique constructif aux auteurs.

Les articles qui vous sont proposés cette année dans les actes qui suivent explorent une grande variété de thématiques relatives à l'apprentissage profond, aussi bien dans les données que dans les processus méthodologiques proposées.

REMERCIEMENTS

Les responsables de l'atelier souhaitent remercier vivement toutes les personnes ayant contribué à la tenue de cet atelier. En particulier :

- les auteurs pour la qualité de leurs contributions constituant la base essentielle de discussions fructueuses ;
- les membres du comité de programme et plus généralement tous les relecteurs de cet atelier dont le travail d'évaluation était crucial pour assurer la qualité de l'atelier ;
- les organisateurs d'EGC 2021 ainsi que les responsables du comité de programme qui ont mis en place l'environnement et les moyens

Table des matières

Segmentation d'images à l'aide d'annotations faibles par combinaison de critères sémantiques et géométriques, Deléarde Robin [et al.] **1**

Prédiction d'efficience d'algorithme de placement, Lallier Corentin [et al.] **15**

Visualisation d'un ensemble d'images et de sa représentation interne dans un réseau de neurones convolutionnel, Serres Barthélémy [et al.] **25**

Auto-encodeur multi-tâches pour le calcul de moyennes de séries temporelles, Terefe Tsegamlak [et al.] **35**

Segmentation non-supervisée d'images histopathologiques à l'aide d'auto-encodeurs et d'ensemble clustering, Rmouque Ilias [et al.] **50**

Liste des auteurs **60**

Segmentation d'images à l'aide d'annotations faibles par combinaison de critères sémantique et géométriques

Robin Deléarde[*,**], Camille Kurtz[*]
Philippe Dejean[**], Laurent Wendling[*]

[*]LIPADE - Université de Paris
prenom.nom@u-paris.fr
[**]Magellium (France)
prenom.nom@magellium.fr

Résumé. Nous proposons une chaîne de traitement permettant d'extraire automatiquement des objets segmentés dans une image, à partir des labels ou des rectangles englobants. Lorsqu'un label est fourni, notre système recherche le label le plus proche dans la liste des sorties, en utilisant une mesure de similarité sémantique. Et lorsqu'un rectangle englobant est fourni, il recherche l'objet en sortie avec la meilleure couverture, en fonction de plusieurs critères géométriques. Associé à un modèle de segmentation sémantique entraîné sur un jeu de données similaires, ou à un bon algorithme de proposition de régions, ce système offre une solution simple pour segmenter efficacement un jeu de données sans nécessiter d'apprentissage spécifique. Une étude expérimentale menée sur PASCAL VOC 2012 atteste que de tels critères permettent de sélectionner la proposition avec le meilleur score de segmentation (IoU) dans la plupart des cas, et ainsi de tirer le meilleur parti de la pré-segmentation. Une implémentation en python est disponible sur: `https://github.com/RobinDelearde/SegMyO`.

1 Introduction

La segmentation est une tâche incontournable dans l'analyse d'images depuis plusieurs décennies, du traitement d'images avec des solutions fondées sur la détection de contours et de régions, à l'apprentissage automatique avec des modèles (profonds) entraînés sur de grands ensembles d'images annotées. Les images segmentées sont utiles pour de nombreuses tâches de vision par ordinateur de plus haut niveau, car elles fournissent une délimitation des objets présents dans la scène, ou des parties constituant un objet. Par ailleurs des informations pertinentes pour leur reconnaissance ou leur interprétation (texture, forme, pose, configuration spatiale, etc.) peuvent être calculées à partir des objets segmentés.

Initialement, la segmentation d'images avait pour objectif de partitionner le contenu de l'image en régions homogènes, en attribuant chaque pixel à une région. Actuellement, beaucoup de travaux font référence au concept de segmentation sémantique, dont le but est de trouver le meilleur label pour la région ou le pixel. Les sorties sont appelées « segments » et sont constituées d'un label et d'un ensemble de pixels de l'image, qui peuvent être donnés sous forme de « masques de segmentation ». Une variante est la segmentation d'instances, où différentes instances de la même classe sont représentées dans différents segments.

En parallèle, la segmentation d'objets consiste à extraire un objet par rapport au fond. Des approches spécifiques existent également pour cette tâche, par exemple en partant de l'hypothèse qu'il n'y a qu'un seul objet dans l'image. D'autres hypothèses sur l'objet peuvent aussi être exploitées, comme sa position (généralement au centre), son étendue (donnée par son rectangle englobant typiquement), etc. Les solutions reposant sur le rectangle englobant et le label sont détaillées dans la suite de cet article.

La segmentation sémantique est apparue avec l'essor de l'apprentissage automatique, et plus récemment l'apprentissage profond, avec des modèles de réseaux de neurones convolutionnels (CNN) comme R-CNN ou FCN. L'idée sous-jacente s'appuie sur une supervision avec des annotations denses au niveau des pixels, ce qui est fastidieux à produire sur de grands ensembles de données. Plusieurs jeux de données ont été mis à disposition de la communauté, comme PASCAL VOC ou COCO, mais ceux-ci sont encore limités à des applications spécifiques. Par ailleurs, il a été évalué expérimentalement que « collecter des rectangles englobants autour de chaque objet de l'image est environ 15 fois plus rapide que d'étiqueter des images au niveau pixellique » (Lin et al., 2014). Dans ce contexte, les solutions faiblement supervisées sont apparues comme une alternative moins coûteuse, n'utilisant que des annotations faibles comme des rectangles englobants ou des légendes d'images (*captions*) pour l'apprentissage. Et lorsqu'elles sont disponibles à l'inférence, ces annotations peuvent être exploitées pour assister la segmentation (« segmentation à l'aide d'annotations »). Il résulte de ces différents cas des tâches de difficulté variable et des solutions spécifiques. Enfin, il est intéressant de remarquer que la segmentation est étroitement liée à la proposition de régions, qui consiste à extraire des régions cohérentes de l'image.

Bénéficiant du développement de grands jeux de données panoptiques, il est désormais possible de trouver de nombreux modèles de segmentation pré-entraînés et de les utiliser directement sur de nouvelles données avec de bonnes performances, pour de nombreuses applications. La sortie peut être une seule image segmentée avec différents labels, ou un masque de segmentation par classe avec recouvrement possible entre les classes. Dans les deux cas, il peut être utile d'extraire automatiquement un objet particulier parmi tous les segments produits, à partir d'indices à son sujet, de manière à intégrer cette étape dans un processus plus large. C'est ce problème que nous considérons ici. La segmentation faiblement supervisée est une extension naturelle de ce problème, et une solution simple consiste à combiner proposition de région et sélection, de façon à générer une supervision au niveau des pixels à partir d'images faiblement annotées.

Dans cet article, nous proposons des critères complémentaires pour extraire une sortie particulière parmi plusieurs propositions, en fonction de son rectangle englobant et/ou de son label. En ce qui concerne le rectangle englobant, nous utilisons une combinaison de critères géométriques sur la couverture de celui-ci par chaque proposition de segmentation. Et concernant le label, nous utilisons la sémantique pour trouver le label proposé le plus proche de celui recherché. Nous proposons d'utiliser cette sélection en aval d'un modèle de segmentation pour générer les propositions, ce qui permet d'obtenir une chaîne de traitement complète pour segmenter des ensembles de données contenant des annotations faibles. De plus, cette chaîne peut être facilement intégrée comme première étape de certains modèles de segmentation faiblement supervisée, de manière à segmenter de nouvelles données sans aucune annotation disponible à l'étape d'inférence, en utilisant les annotations faibles pour l'apprentissage.

2 État de l'art

2.1 Transfert de segmentation

Les modèles modernes de segmentation, quel que soit leur niveau de supervision, sont entraînés sur des données représentatives, les rendant spécifiquement adaptés à ces dernières, avec une capacité de généralisation variable. Utiliser un tel modèle pour un autre jeu de données ou une autre tâche est possible, mais cela nécessite quelques adaptations pour le rendre efficace, en exploitant les connaissances disponibles sur le nouveau jeu de test. Cette solution générale appelée « apprentissage par transfert » est déclinée en « adaptation de domaine » lorsque la tâche est modifiée, et « *fine-tuning* » lorsque le modèle est uniquement adapté à de nouvelles classes (en modifiant la dernière couche) et ré-entraîné sur les nouvelles données. Le *fine-tuning* est très utilisé pour la détection et la reconnaissance d'objets, mais beaucoup moins pour la segmentation sémantique qui requiert des annotations denses.

Dans ce contexte, (Hong et al., 2016) propose d'utiliser l'apprentissage par transfert pour la segmentation sémantique faiblement supervisée, en transférant les connaissances des catégories avec des annotations fortes vers des catégories inconnues avec des annotations faibles, grâce à une architecture encodeur-décodeur associée à un modèle d'attention visuelle. (Sun et al., 2019) propose une solution pour bénéficier à la fois de données réelles et synthétiques, en utilisant avec le réseau de segmentation un deuxième réseau dédié à l'apprentissage de la similarité des pixels synthétiques avec les pixels réels. Enfin, (Pascal et al., 2019) propose d'utiliser la similarité sémantique, en réutilisant la dernière couche du réseau d'origine lorsque les labels sont sémantiquement proches. Il n'est évalué que sur une tâche de classification, mais l'idée peut également être utilisée pour la segmentation. Nous considérons cette approche ici, mais uniquement comme un post-traitement étant donné que nous n'avons pas accès aux annotations au niveau des pixels pour apprendre un nouveau modèle.

2.2 Segmentation à l'aide d'annotations faibles

Nous n'utilisons pas l'expression de « segmentation faiblement supervisée » pour cette tâche car elle se réfère généralement à l'étape d'entraînement du modèle uniquement, avec des solutions basées sur un apprentissage faiblement supervisé, alors qu'ici nous considérons que les annotations faibles sont disponibles à l'inférence. En fait, il s'agit souvent de la première étape de solutions de segmentation faiblement supervisées, exploitant des annotations faibles pour générer une vérité terrain au niveau du pixel, qui est ensuite utilisée pour entraîner un modèle de segmentation classique, de la même manière que les méthodes auto-supervisées.

Deux principales catégories d'approches coexistent : celles exploitant le rectangle englobant de l'objet et celles exploitant des labels donnés au niveau image (*captions*). Typiquement, ces indices proviennent d'une première étape de détection ou de sous-titrage (*captioning*), ou bien d'annotations manuelles. Quelques solutions exploitent une combinaison de labels au niveau image et de rectangles englobants : (Papandreou et al., 2015) utilise l'un ou l'autre en fonction de ce qui est disponible, tandis que (Li et al., 2018) utilise des rectangles englobants pour les objets d'intérêt et des labels au niveau image pour l'arrière-plan (*stuff objets*). Mais à notre connaissance et étonnamment, aucune solution n'exploite la combinaison d'un rectangle englobant directement avec le label de l'objet concerné.

Segmentation d'objet à l'aide d'un rectangle englobant : Connaître le rectangle englobant d'un objet est évidemment un indice pertinent pour en déduire son étendue spatiale réelle, d'autant plus si celui-ci est limité à l'objet ciblé. Le cas le plus simple est la segmentation objet/fond, où l'objet est seul dans l'image ou le rectangle englobant. Mais dans de nombreux cas plusieurs objets peuvent être présents et choisir le plus pertinent peut être délicat. Alors que les premières solutions reposaient sur une interaction plus importante avec l'utilisateur, comme de la sélection de points ou des griffonnages à l'intérieur et à l'extérieur de l'objet, GrabCut (Rother et al., 2004) était la première solution à ne s'appuyer que sur un rectangle englobant. Cette méthode bien connue se fonde sur deux modèles d'apparence (arrière et premier plans) et sur une approche itérative par graph-cut pour obtenir la segmentation finale. Des variantes comme (Lempitsky et al., 2009) exploitent l'hypothèse d'étroitesse du rectangle englobant, en ajoutant des contraintes dans la minimisation de l'énergie.

Une autre solution courante consiste à générer plusieurs propositions de segments, avec une pré-segmentation ou un algorithme de proposition de régions, et de sélectionner ou de générer la sortie finale grâce à un vote ou une combinaison. Dans la littérature, la plupart des solutions reposent sur la proposition de régions, en particulier dans le cadre faiblement supervisé où elle est utilisée comme première étape pour générer une vérité terrain au niveau pixellique. Par exemple, BoxSup (Dai et al., 2015) repose sur l'approche MCG (Arbeláez et al., 2014) et évalue plusieurs autres solutions. (Khoreva et al., 2017) est également basé sur MCG mais le combine avec GrabCut, et permet en plus de gérer plusieurs instances d'un même objet. Dans notre solution, nous suggérons d'utiliser une pré-segmentation avec un modèle de segmentation complet entraîné sur un jeu de données similaire, en utilisant par exemple le modèle Mask R-CNN (He et al., 2017).

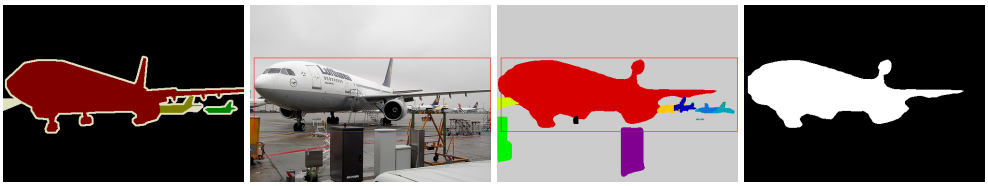
D'autres pistes ont également été explorées, notamment pour la segmentation faiblement supervisée. (Papandreou et al., 2015) propose d'utiliser un champ aléatoire conditionnel (CRF) et de le contraindre à considérer la zone centrale du rectangle englobant comme premier plan et les pixels à l'extérieur du rectangle comme arrière-plan. (Hsu et al., 2019) utilise quant à lui le modèle MIL (Multiple Instance Learning) pour gérer plusieurs instances, en y intégrant l'hypothèse d'étroitesse du rectangle englobant. Et plus récemment encore, BB-UNet (Jurdi et al., 2020), qui est basé sur U-Net, exploite des hypothèses sur la forme en introduisant une nouvelle couche de convolution, pour segmenter des images médicales.

Segmentation d'image à l'aide de labels : De nombreuses solutions exploitant les labels au niveau de l'image existent dans la littérature, en particulier pour la segmentation faiblement supervisée à nouveau (Kolesnikov et Lampert, 2016; Zhou et al., 2018; Ahn et al., 2019; Wang et al., 2020). Ces solutions sont généralement basées sur des cartes d'activation par classe (CAM) et un modèle d'attention visuelle pour générer la vérité terrain au niveau des pixels. L'approche de (Guillaumin et al., 2014) peut également être mentionnée ici, puisque son objectif est de segmenter un jeu de données complet contenant des annotations faibles comme nous (ImageNet dans leur cas), mais avec une idée totalement différente. Celle-ci repose sur une approche gloutonne originale, en segmentant progressivement les objets dont les labels sont sémantiquement proches de ceux déjà segmentés, avec un modèle d'apparence pour les pixels de premier plan et un autre pour l'arrière-plan. Le modèle est initialisé avec les annotations niveau pixels de PASCAL VOC, et exploite également les annotations de rectangles englobants lorsqu'elles sont disponibles.

3 Pipeline de segmentation

3.1 Principe

Nous proposons une solution à la tâche de segmentation d'objets à l'aide d'un rectangle englobant et/ou d'un label, sur une image contenant plusieurs objets et pas seulement un arrière-plan. Il repose sur un pipeline en deux étapes appelé SegMyO (Segment My Object) : tout d'abord un ensemble de régions est extrait avec une solution classique de segmentation d'images, ou avec un algorithme de proposition de régions, puis la meilleure sortie est sélectionnée pour le rectangle englobant donné, et le label dans le cas de la segmentation sémantique. Concernant l'étape de segmentation, l'utilisation d'un modèle pré-entraîné suppose que ce dernier a été entraîné sur des objets et des labels similaires à ceux d'intérêt. Pour cela, il existe maintenant de bons modèles de segmentation pré-entraînés sur des ensembles de données importants et variés, pour de nombreuses applications. Afin d'étendre le modèle appris à d'autres labels similaires, nous proposons en plus d'utiliser la similarité sémantique pour rechercher un label similaire parmi ceux du modèle. Cela peut aussi être vu comme un transfert d'un modèle entraîné sur un ensemble de labels vers un autre ensemble ne contenant pas exactement les mêmes labels, comme cela est fait dans (Pascal et al., 2019).



(a) (b) (c) (d)

FIG. 1: Segmentation d'une image (PASCAL VOC 2012) : (a) Vérité terrain (3 instances de la classe “aeroplane"), (b) image d'entrée et rectangle englobant avec le label “aeroplane", (c) masques produits par Mask R-CNN (limités au rectangle englobant - 12 objets), (d) masque sélectionné par SegMyO (de la classe “airplane").

Notre système prend en entrée :

— une image I de dimension $W \times H$, avec W la largeur et H la hauteur de l'image ;
— un label $l \in L^{target}$ d'un objet présent dans l'image (*e.g.*, $L^{target} = \{\text{Man}, \text{Car}, \ldots\}$) ;
— un rectangle englobant B autour de cet objet, défini par les coordonnées de son coin supérieur gauche (x_1, y_1) et de son coin inférieur droit (x_2, y_2), avec (x_1, y_1) et (x_2, y_2) $\in [1, W] \times [1, H]$;
— une segmentation de l'image I constituée d'un ensemble de régions $\{R_i\}_{i=1..N}$, avec recouvrement possible, chacune détectée avec un score de confiance r_i, et chacune potentiellement accompagnée d'un label $l_i \in L^{init}$ (*e.g.*, $L^{init} = \{\text{Person}, \text{Automobile}, \ldots\}$).

Pour chaque région candidate R_i, un score est calculé à partir de la couverture du rectangle englobant, et de la similarité sémantique entre le label attendu l et le label prédit l_i si on utilise la segmentation sémantique, grâce aux critères définis dans la section 3.2. Ensuite, le système retient comme segmentation de l'objet la région avec le meilleur score parmi toutes les régions candidates, comme illustré sur la figure 1.

3.2 Critères de sélection

3.2.1 Critères géométriques

Tout d'abord, deux critères visant à exploiter le fait que le rectangle englobant est limité à l'objet sont considérés, étant donné que ce sont des critères nécessaires et universels. Ils reposent sur l'hypothèse que le rectangle englobant contient tout l'objet et pas plus d'autres éléments que nécessaire. Ainsi, plus le rectangle englobant est précis, meilleure sera la segmentation. Cette hypothèse est utilisée dans (Lempitsky et al., 2009; Hsu et al., 2019) et peut également être dérivée des hypothèses sur l'étendue de l'arrière-plan et de l'objet de (Khoreva et al., 2017). Les critères proposés sont les suivants :

C1. la distance relative maximale de la région aux bords du rectangle englobant, qui doit être proche de zéro pour les 4 côtés ;

C2. l'étendue relative de la région, c'est-à-dire la partie de la région qui se trouve dans le rectangle englobant, en supposant que tout l'objet doit être dans celui-ci.

D'autres critères optionnels peuvent être définis en fonction des connaissances a priori sur la forme des objets, comme :

C3. la couverture du rectangle englobant par l'objet, qui vise à donner plus de poids aux grands objets (c'est ce critère qui est utilisé dans BoxSup (Dai et al., 2015)) ;

C4. la présence au centre du rectangle englobant, pour les objets « barycentriques » ou pour éviter les objets présents uniquement sur les bords.

Dans nos tests nous utilisons les critères C1, C2 et C3. Ils sont définis comme suit dans l'intervalle $[0, 1]$, avec $R_{i,B} = R_i \cap B$ la restriction de la région R_i au rectangle englobant B (cf. figure 2) :

$$c_1(R_i, B) = 1 - \max\left[\frac{d_x(R_{i,B}, B)}{W}, \frac{d_y(R_{i,B}, B)}{H}\right]$$

$$c_2(R_i, B) = \frac{area(R_{i,B})}{area(R_i)} \qquad c_3(R_i, B) = \frac{area(R_{i,B})}{area(B)}$$

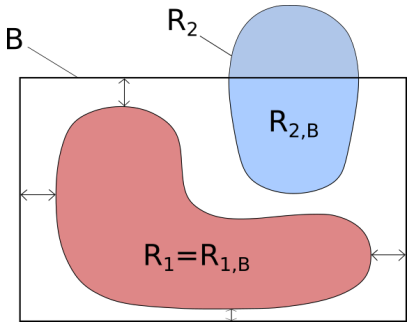


FIG. 2: Deux régions candidates au sein de la même boite englobante.

3.2.2 Critère sémantique

Lors de l'utilisation de la segmentation sémantique, chaque segment se voit attribuer un label à partir de l'ensemble de labels appris par le modèle. Il est donc utile de connaître celui de l'élément recherché, cependant il peut ne pas faire partie de cet ensemble. En effet, un problème majeur réside dans la spécificité et le faible nombre de classes dans la plupart des bases de données académiques usuelles. Les modèles de segmentation pré-entraînés sur ces bases de données sont donc limités à la segmentation des classes d'objets existantes dans ces bases de données. Par exemple, si la classe « chat » est apprise dans le jeu de données initial, il sera difficile de fournir comme label d'entrée « lynx » dans le jeu de données cible. Pourtant, le modèle de segmentation du chat peut être utilisé pour segmenter un lynx puisque ces objets sont visuellement proches et sémantiquement liés. C'est pourquoi nous proposons d'utiliser la correspondance sémantique entre labels pour étendre le vocabulaire aux labels en dehors de l'ensemble initial.

Pour cela, nous utilisons des outils de traitement automatique des langues, permettant de calculer une similarité sémantique entre le label prédit et le label attendu :

$$c_{sem}(l_i, l) = semantic\_similarity(l_i, l)$$

Deux approches principales peuvent être envisagées : utiliser la similarité entre mots dans une taxonomie comme WordNet, ou la similarité entre des représentations de ces mots (*word embeddings*) apprises sur un corpus, avec un modèle comme Word2Vec. Cependant, cela nécessite que les deux labels soient présents dans le vocabulaire, ce qui nécessite un contrôle lors du choix du modèle, ou d'effectuer des corrections manuelles.

Plusieurs méthodes sont disponibles pour calculer la similarité sémantique dans une taxonomie, comme les similarités de Rada, Resnik, Li, etc. La plupart d'entre elles sont basées soit sur des mesures structurelles entre les concepts de la taxonomie (par exemple, la longueur et la profondeur du chemin), soit sur le contenu d'information (IC). Nous suggérons d'utiliser la similarité *wpath* (Zhu et Iglesias, 2016), qui combine les deux approches en utilisant le contenu d'information pour pondérer la longueur de chemin la plus courte entre les concepts, et donne les meilleures performances en général. Cette mesure est définie comme :

$$c_{sem,wpath}(l_i, l) = \left(1 + length(l_i, l) * k^{IC(lcs(l_i,l))}\right)^{-1}$$

avec lcs le plus petit ancêtre commun (*least common subsumer*), et k un paramètre indiquant la contribution du contenu d'information de celui-ci.

3.3 Combinaison de critères

Un critère global peut être calculé à partir de l'ensemble des critères précédents et du score de confiance de segmentation r_i. Nous proposons ici de le calculer simplement avec une somme pondérée des différents scores, en fixant nous-mêmes les poids de chaque critère. D'autres solutions sont possibles, par exemple en apprenant les poids, cependant nous avons constaté expérimentalement que la combinaison proposée était satisfaisante (cf. section 4).

Comme mentionné précédemment, les critères C1 et C2 sont particulièrement importants, nous suggérons donc de leur donner un poids plus important, alors que le critère C3 est plus

litigieux et devrait donc avoir un poids plus petit, à moins d'avoir des connaissances a priori sur la forme des objets recherchés. Le critère sémantique est également particulièrement significatif, nous suggérons donc de lui affecter un poids important.

Dans nos expérimentations, nous utilisons le score suivant :

$$score_1 = \begin{cases} (r_i + 2 * c_1 + 2 * c_2 + c_3 + 4 * c_{sem})/10 \\ \quad \text{quand } l_i \in L^{init} \text{ et } l \in L^{target} \\ (r_i + 2 * c_1 + 2 * c_2 + c_3)/6 \\ \quad \text{sinon} \end{cases}$$

Pour aller encore plus loin, les critères C1 et C2 sont des conditions nécessaires, nous suggérons donc de ne conserver que la valeur minimale de C1, C2 et le score global :

$$score_2 = min(c_1, c_2, score_1)$$

Il est à noter que tous ces critères peuvent être impactés si plusieurs instances de l'objet sont présentes dans l'image et que la segmentation ne parvient pas à les séparer. Ils doivent donc être utilisés uniquement avec la segmentation d'instances ou avec des ensembles de données comprenant une seule instance de chaque objet.

4 Expérimentations

4.1 Données et protocole expérimental

Nous évaluons notre chaîne de traitement sur le jeu de données de segmentation PASCAL VOC 2012, en utilisant le sous-ensemble de validation (1 449 images, 3 427 objets, 20 classes). Pour générer les régions candidates, nous utilisons le modèle *torchvision* Mask R-CNN ResNet-50 FPN [1] (He et al., 2017), entraîné sur COCO 2017 (80 classes, dont les 20 classes de PASCAL VOC mais avec des noms pouvant être différents). Le résultat de la pré-segmentation est une liste de propositions d'objets composées d'un label, d'un score et d'un masque de segmentation avec des valeurs dans $[0, 1]$. Nous ne conservons que les propositions dont le score de segmentation est supérieur à un seuil (fixé à 0,25). L'utilisabilité d'une solution de proposition de régions par groupement combinatoire multi-échelle (MCG) (Arbeláez et al., 2014) est aussi évaluée, à partir des propositions précalculées disponibles en ligne [2].

Les critères décrits précédemment ainsi que leur combinaison sont ensuite calculés. Pour la similarité sémantique, nous avons utilisé la taxonomie WordNet et la méthode *wpath* (Zhu et Iglesias, 2016), avec le framework python *sematch* [3]. Comme métrique d'évaluation de la segmentation, le score d'intersection sur l'union (IoU) est calculé pour chaque objet en utilisant les annotations de segmentation d'instance fournies (après une binarisation de l'image - seuil à 0,3). La moyenne sur l'ensemble des objets est donnée dans le Tableau 1. Afin de comparer avec d'autres méthodes, nous calculons également le score (IoU) moyen pour chaque classe d'objets, après avoir transformé nos sorties pour se conformer à la tâche de segmentation d'image. Pour ce faire, nous avons ajouté une étape de fusion des segments sélectionnés

1. https://pytorch.org/docs/stable/torchvision/models.html
2. https://www2.eecs.berkeley.edu/Research/Projects/CS/vision/grouping/mcg/
3. https://github.com/gsi-upm/sematch

individuellement, en les insérant dans l'image de sortie du plus grand au plus petit, de façon à gérer le cas des segments qui se chevauchent. Nous nous comparons également à GrabCut (Rother et al., 2004), en tant que solution non supervisée de l'état de l'art pour notre tâche.

4.2 Résultats et discussion

Les résultats de nos expériences sur la segmentation à l'aide d'annotations faibles sont indiqués dans le Tableau 1, pour plusieurs solutions et plusieurs critères de sélection, tandis que le Tableau 2 fournit une comparaison avec plusieurs modèles récents de segmentation faiblement et entièrement supervisés. Pour chaque solution, nous indiquons à la fois le niveau d'annotations utilisé pour entraîner le modèle (supervision) et le niveau d'annotations disponibles à l'inférence. Il peut s'agir de « pixels » lorsque les annotations au niveau des pixels sont utilisées, de « légende » lorsque des labels au niveau de l'image sont considérés et de « b.box » ou « b.box + label » lorsque des annotations au niveau du rectangle englobant sont utilisées.

Dans nos tests (Tableau 1), nous distinguons les solutions totalement non supervisées, c'est-à-dire n'utilisant aucune donnée pour construire le modèle, et celles n'utilisant pas les données d'apprentissage de PASCAL VOC mais un jeu de données similaire (ici COCO), dans un mode d'apprentissage par transfert. Comme méthode comparative simpliste pour les annotations de rectangles englobants, une solution basée sur un remplissage partiel du rectangle est également évaluée (solution « rectangle englobant »). Enfin, concernant les solutions basées sur notre pipeline avec une pré-segmentation et une sélection de sortie, une borne supérieure pour le modèle de pré-segmentation choisi peut également être obtenue, en utilisant pour la sélection de sortie le score IoU comme critère au lieu du nôtre (ce qui suppose d'avoir accès aux annotations au niveau pixellique, donc à la vérité terrain).

méthode	supervision	annotation à l'inférence	mIoU	mIoU$_{obj}$
non supervisée				
rectangle englobant	\	b.box	56.50%	**53.16%**
GrabCut (Rother et al., 2004)	\	b.box	**58.82%**	44.29%
MCG + SegMyO$_{C3}$	\	b.box	47.64%	44.39%
MCG + SegMyO$_{mIoU}$	\	pixels (VT)	51.08%	47.50%
transfert d'apprentissage (depuis COCO)				
MR-CNN + SegMyO$_{mIoU}$	COCO	pixels (VT)	**74.68%**	**70.67%**
MR-CNN + SegMyO$_{C1}$	COCO	b.box	70.55%	65.04%
MR-CNN + SegMyO$_{C2}$	COCO	b.box	53.19%	51.95%
MR-CNN + SegMyO$_{C3}$	COCO	b.box	**71.02%**	**66.85%**
MR-CNN + SegMyO$_{sem}$	COCO	caption	68.71%	58.19%
MR-CNN + SegMyO$_{C3*sem}$	COCO	b.box+label	73.16%	68.63%
MR-CNN + SegMyO$_{score1}$	COCO	b.box	73.08%	68.97%
MR-CNN + SegMyO$_{score2}$	COCO	b.box	73.04%	68.74%
MR-CNN + SegMyO$_{score1}$	COCO	b.box+label	**73.62%**	**69.35%**
MR-CNN + SegMyO$_{score2}$	COCO	b.box+label	73.30%	68.93%
MR-CNN+ SegMyO$_{score1}$ / rectangle englobant	COCO	b.box+label	**73.99%**	**69.88%**

TAB. 1: Scores de segmentation sur l'ensemble de validation de PASCAL VOC, en utilisant des annotations lors de l'inférence, pour des modèles non supervisés ou entraînés sur un autre ensemble de données (Mask R-CNN entraîné sur COCO) (mIoU : moyenne sur les 21 classes, mIoU$_{obj}$: moyenne sur l'ensemble des objets sans la classe d'arrière-plan).

			scores (mIoU)	
méthode	supervision	annotation à l'inférence	validation	test
faiblement supervisé				
BoxSup (Dai et al., 2015)	b.box	\	62.0%	64.2%
SEAM (Wang et al., 2020)	caption	\	64.5%	65.7%
supervisé sans données additionnelles				
ResNet-38 (Wu et al., 2019)	pixels	\	n.c.	82.5%
DeepLabv3+ (Chen et al., 2018)	pixels	\	81.63%	n.c.
supervisé avec données additionnelles (COCO)				
ResNet-38 (Wu et al., 2019)	pixels	\	80.84%	84.9%
DeepLabv3+ (Chen et al., 2018)	pixels	\	84.56%	89.0%

TAB. 2: Scores de segmentation issus de la littérature sur les ensembles de validation/test de PASCAL VOC, pour des modèles entraînés sur l'ensemble d'apprentissage de PASCAL VOC (mIoU : moyenne sur les 21 classes).

Ces résultats montrent qu'une segmentation totalement non supervisée à l'aide d'annotations faibles ne donne pas de résultats satisfaisants, et ne permet pas segmenter efficacement un jeu de données malgré l'information du rectangle englobant. En revanche, l'utilisation d'un modèle pré-entraîné sur un jeu de données similaire permet d'atteindre une bonne performance, se classant entre les modèles faiblement supervisés et entièrement supervisés, sans nécessiter d'entraînement sur le nouveau jeu de données. Cela montre qu'il est possible de segmenter sans entraînement spécifique, grâce à un usage approprié des annotations faibles lors de l'inférence.

Concernant les différents critères proposés, ceux-ci donnent des résultats variables lorsqu'ils sont pris individuellement, avec un bon comportement des critères C3 et C1, et dans une moindre mesure du critère sémantique (qui montre un écart important entre les deux modes de calcul du mIoU). Combiner plusieurs de ces critères permet alors de se rapprocher de la borne supérieure donnée par le mIoU, par exemple en combinant le critère sémantique au critère C3. Le meilleur score moyen est alors obtenu avec le $score_1$, en utilisant à la fois les rectangles englobants et les labels. Cependant ajouter les labels n'apporte par un gain significatif contrairement aux rectangles englobants, ce qui atteste de l'intérêt supérieur d'annoter des rectangles englobants. Enfin, le $score_2$ ne semble pas améliorer la performance ici. Enfin, une dernière solution consiste à remplir le rectangle englobant (à 90%) lorsque le score calculé est inférieur à un seuil (fixé à 0,5), ce qui permet d'atteindre la meilleure performance, très proche de la borne supérieure donnée par le mIoU.

Comme expérience supplémentaire « qualitative », nous avons utilisé ce pipeline pour segmenter SpatialSense (Yang et al., 2019), un jeu de données contenant des annotations de relations spatiales entre des objets représentés par leurs rectangles englobants. Ce jeu de données est composé de 2 162 images avec des objets de la vie quotidienne, des animaux, des personnes (avec plusieurs labels "homme", "femme", "fille"...), mais aussi des classes d'objets de type *stuff* (comme "ciel", "sol", "mur"...). La figure 3 présente quelques exemples de segmentations obtenues sur ces données, en utilisant comme pré-segmentation un modèle HRNet entraîné sur le jeu de données ADE20K [4]. De tels résultats mettent en évidence l'intérêt de notre pipeline dans un cas réel, puisque SpatialSense est fourni sans aucune annotation de segmentation. Une telle segmentation peut être utile pour diverses tâches de vision par ordinateur, comme le calcul de descripteurs de position relative (Deléarde et al., 2021).

4. https://github.com/CSAILVision/semantic-segmentation-pytorch

FIG. 3: Exemples de segmentations obtenues sur SpatialSense, pré-segmentation avec HRNet entraîné sur ADE20K et sélection avec SegMyO : (1^{e} ligne) entrées : images + annotations (rectangle englobant + label) ; (2nde ligne) sorties : objets segmentés sélectionnés.

5 Conclusion

Nous avons introduit un pipeline pour extraire automatiquement des objets segmentés dans des images à l'aide de labels et/ou de rectangles englobants. Il s'appuie sur des critères simples pour sélectionner le meilleur segment parmi plusieurs propositions pour un objet donné, grâce à la similarité sémantique pour le label et à plusieurs mesures géométriques pour le rectangle englobant. Sa capacité à segmenter et sélectionner facilement et automatiquement tout objet dont on dispose de la classe et/ou du rectangle englobant en fait une solution pertinente pour segmenter un jeu de données sans nécessiter d'annotation dense ni d'entraînement spécifique. Par ailleurs, il s'agit également d'une amélioration intéressante pour les modèles de segmentation faiblement supervisés, que nous envisageons d'évaluer en l'intégrant dans un modèle dédié. Enfin, nous envisageons également d'évaluer plus précisément la complémentarité de nos critères et d'optimiser leur agrégation sur d'autres ensembles de données, où les différences sémantiques entre les classes pourraient être plus importantes.

Remerciements

Ce travail a été mené au LIPADE et financé par Magellium, avec le soutien de l'Agence de l'Innovation de Défense (AID).

Références

Ahn, J., S. Cho, et S. Kwak (2019). Weakly supervised learning of instance segmentation with inter-pixel relations. In *CVPR*, pp. 2209–2218.

Arbeláez, P., J. Pont-Tuset, J. T. Barron, F. Marques, et J. Malik (2014). Multiscale combinatorial grouping. In *CVPR*, pp. 328–335.

Chen, L.-C., Y. Zhu, G. Papandreou, F. Schroff, et H. Adam (2018). Encoder-decoder with atrous separable convolution for semantic image segmentation. In *ECCV*, pp. 801–818.

Dai, J., K. He, et J. Sun (2015). Boxsup : Exploiting bounding boxes to supervise convolutional networks for semantic segmentation. In *ICCV*, pp. 1635–1643.

Deléarde, R., C. Kurtz, P. Dejean, et L. Wendling (2021). Force banner for the recognition of spatial relations. In *ICPR 2020*, pp. 6065–6072.

Guillaumin, M., D. Küttel, et V. Ferrari (2014). Imagenet auto-annotation with segmentation propagation. *Int J Comput Vis 110*(3), 328–348.

He, K., G. Gkioxari, P. Dollár, et R. Girshick (2017). Mask R-CNN. In *ICCV*, pp. 2961–2969.

Hong, S., J. Oh, H. Lee, et B. Han (2016). Learning transferrable knowledge for semantic segmentation with deep convolutional neural network. In *CVPR*, pp. 3204–3212.

Hsu, C.-C., K.-J. Hsu, C.-C. Tsai, Y.-Y. Lin, et Y.-Y. Chuang (2019). Weakly supervised instance segmentation using the bounding box tightness prior. In *NIPS*, pp. 6586–6597.

Jurdi, R. E., C. Petitjean, P. Honeine, et F. Abdallah (2020). BB-UNet : U-Net with bounding box prior. *IEEE J Sel Top Signal Process*.

Khoreva, A., R. Benenson, J. Hosang, M. Hein, et B. Schiele (2017). Simple does it : Weakly supervised instance and semantic segmentation. In *CVPR*, pp. 876–885.

Kolesnikov, A. et C. H. Lampert (2016). Seed, expand and constrain : Three principles for weakly-supervised image segmentation. In *ECCV*, pp. 695–711.

Lempitsky, V., P. Kohli, C. Rother, et T. Sharp (2009). Image segmentation with a bounding box prior. In *ICCV*, pp. 277–284.

Li, Q., A. Arnab, et P. H. Torr (2018). Weakly-and semi-supervised panoptic segmentation. In *ECCV*, pp. 102–118.

Lin, T.-Y., M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, et C. L. Zitnick (2014). Microsoft COCO : Common objects in context. In *ECCV*, pp. 740–755.

Papandreou, G., L.-C. Chen, K. P. Murphy, et A. L. Yuille (2015). Weakly-and semi-supervised learning of a deep convolutional network for semantic image segmentation. pp. 1742–1750.

Pascal, L., X. Bost, et B. Huet (2019). Semantic and visual similarities for efficient knowledge transfer in cnn training. In *CBMI*, pp. 1–6.

Rother, C., V. Kolmogorov, et A. Blake (2004). "GrabCut" interactive foreground extraction using iterated graph cuts. *ACM Trans Graph 23*(3), 309–314.

Sun, R., X. Zhu, C. Wu, C. Huang, J. Shi, et L. Ma (2019). Not all areas are equal : Transfer learning for semantic segmentation via hierarchical region selection. In *CVPR*, pp. 4360–4369.

Wang, Y., J. Zhang, M. Kan, S. Shan, et X. Chen (2020). Self-supervised equivariant attention mechanism for weakly supervised semantic segmentation. In *CVPR*, pp. 12275–12284.

Wu, Z., C. Shen, et A. Van Den Hengel (2019). Wider or deeper : Revisiting the resnet model for visual recognition. *Pattern Recognit 90*, 119–133.

Yang, K., O. Russakovsky, et J. Deng (2019). SpatialSense : An adversarially crowdsourced benchmark for spatial relation recognition. In *ICCV*, pp. 2051–2060.

Zhou, Y., Y. Zhu, Q. Ye, Q. Qiu, et J. Jiao (2018). Weakly supervised instance segmentation using class peak response. In *CVPR*, pp. 3791–3800.

Zhu, G. et C. A. Iglesias (2016). Computing semantic similarity of concepts in knowledge graphs. *IEEE Trans Knowl Data Eng 29*(1), 72–85.

Summary

We propose a pipeline to automatically extract segmented objects in images based on given labels or bounding boxes. When providing the expected label, our system looks for the closest label in the list of outputs, using a measure of semantic similarity. And when providing the bounding box, it looks for the output object with the best coverage, based on several geometric criteria. Associated with a semantic segmentation model trained on a similar dataset, or a good region proposal algorithm, this pipeline provides a simple solution to segment efficiently a dataset without requiring specific training. An experimental study conducted on the PASCAL VOC 2012 dataset shows that such criteria allow to select the proposal with the best segmentation score (IoU) in most cases, and so to get the best of the pre-segmentation. A python implementation of the proposed pipeline is available at: `https://github.com/RobinDelearde/SegMyO`.

Apprentissage profond : prédiction d'efficience de placement pour la découpe industrielle de textiles

Corentin Lallier*,**, Bruno Pinaud*, Marc Palyart**, Laurent Vézard**, Guillaume Blin*

*LaBRI UMR CNRS 5800 – Université de Bordeaux, Bât A30,
351 cours de la Libération, 33405 Talence Cedex
{prénom}.{nom}@u-bordeaux.fr,
https://www.labri.fr
**Lectra - 23 Chemin de Marticot, 33610 Cestas
{initiale prénom}.{nom}@lectra.com
https://www.lectra.com

Résumé. Dans un processus industriel de découpe de matières textiles, les minimisations des pertes de matières ainsi que des temps de coupe sont des préoccupations majeures. Le temps peut être optimisé en découpant plusieurs couches de matières en même temps. Optimiser la matière est en revanche un problème complexe. Afin d'être découpées, les pièces sont regroupées par lots. Obtenir les regroupements les plus efficients en matière nécessite le calcul de la consommation de matière pour chaque lot possible. L'efficience matière d'un lot est obtenue grâce à un algorithme de placement qui positionne les pièces sur la matière. Cet algorithme est très coûteux en temps et financièrement, rendant impossible le calcul de l'ensemble des lots. Actuellement, le problème est résolu à l'aide de tables de références approximatives conçues empiriquement qui donnent le taux d'utilisation matière pour chaque lot. Dans cet article, nous présenterons les travaux de ma première année de thèse : une première approche en apprentissage profond permettant d'estimer l'efficience du placement d'un lot à partir d'une base de données construites sur des placements existants. Nous proposons une combinaison de techniques de modélisation des pièces à l'aide de réseaux de neurones convolutionels et de modélisation d'un lot de pièces grâce à des méthodes de type *Multiple Instance Learning*. Nous présentons aussi les résultats préliminaires liés à cette approche et nos plans pour la suite.

1 Introduction

L'utilisation de textile est prépondérante dans de nombreux domaines tels que la mode (vêtements, accessoires, chaussures), l'automobile (sièges, intérieurs, airbags) ou encore l'ameublement (canapés, fauteuils). La minimisation des pertes de matières textiles lors de la fabrication industrielle de ces articles est une préoccupation majeure ;

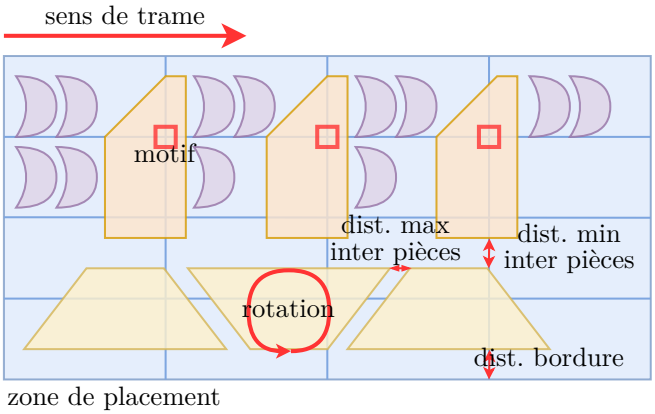


FIG. 1 – *Schéma d'un placement. La zone bleue représente la surface rectangulaire du matériau à découper (aussi appelée zone de placement). Les polygones sont les pièces à découper. Les contraintes sont représentées en rouge.*

autrefois pour des raisons purement économiques et plus récemment aussi pour des raisons environnementales (Rissanen et McQuillan, 2016; Henninger et al., 2016).

Au-delà de la conception elle-même du produit (forme et composition), deux facteurs permettent de réduire les chutes de textile : la façon dont les pièces des articles sont regroupées par lot (couramment appelé plan de sections) pour être découpées sur des rouleaux ou des matelas de matériaux (empilement de couches de tissus) et la façon dont ces pièces sont disposées sur la matière pour la découpe (appelé placement). La réalisation d'un plan de sections nécessite de connaître l'efficience (proportion de matière réellement utilisée) de chaque groupe possible pour déterminer les groupements les plus efficients. Malheureusement, la réalisation d'un placement pour chaque combinaison de regroupement possible n'est pas envisageable car l'opération de placement est trop coûteuse tant en temps de calcul que financièrement. À la place on se base sur des tables de références approximatives conçues empiriquement via la réalisation de quelques placements types. Par exemple, on va décider qu'un placement mélangeant des pantalons basiques de taille S et M va aboutir à une efficience de 80%. Cette approximation conduit souvent à ne pas faire les regroupements optimaux. Dans cette optique, nos travaux visent à développer un algorithme capable de prédire rapidement l'efficience attendue d'un placement sans le réaliser afin d'améliorer la performance de la construction des plans de sections.

Le plan de cet article est le suivant : la partie 2 présente le problème du placement plus en détails, puis dans la partie 3 nous présentons notre approche de la modélisation du problème. Ensuite, la partie 4 détaille l'architecture du modèle qui répond à cette modélisation et en particulier les premiers résultats dans la partie 4.2. Enfin, nous concluons (partie 5) sur nos travaux futurs.

2 Problématique

Un placement consiste à positionner convenablement les pièces sur le rouleau ou matelas de matériau à découper en respectant les contraintes imposées, tout en minimisant la perte de matière.

La figure 1 présente les principaux éléments d'un placement. La zone de placement, ici la zone bleue, représente la surface rectangulaire du matériau à découper, par exemple un matelas de tissu. Les pièces représentent les formes issues du patron CAO (conception assistée par ordinateur). Ces formes sont représentées par des polygones. Par exemple, dans un patron de chemise, une manche est représentée par un trapèze (pièce plus large à l'épaule qu'au niveau des poignets).

L'algorithme de placement doit répondre à un certain nombre de contraintes liées aux pièces ou à la zone de placement. Sur la figure 1, on peut voir certains exemples de contraintes en rouge. Généralement les contraintes sont liées à la machine de découpe ou à la matière à découper. Si la matière à découper est un tissu, alors les pièces vont devoir respecter la trame du tissu (contrainte **sens de trame**) afin de répondre à des critères de résistance et de qualité de découpe et seules certaines rotations respectant la trame sont autorisées (contraintes **rotation**).

De plus, dans les rouleaux de tissus à découper, les bordures sont généralement à éviter, car moins résistantes et les colorations ne sont pas uniformes sur toute la longueur du rouleau. Sur ce type de matériau, les pièces proches dans l'article final, par exemple deux parties d'une manche d'une chemise, vont devoir être placées sur des zones de couleur ou de motifs similaires. Ceci est modélisé par les contraintes de **distances maximum entre pièces**, de **distance à la bordure** et de **motifs**.

De même, si deux pièces sont trop proches, le découpage d'une pièce peut dégrader la pièce adjacente du fait de contraintes physiques de la machine de découpe comme la largeur de la lame. Cette contrainte est nommée **distance minimale entre pièces**.

L'efficience est donnée par le rapport entre la somme des surface des pièces sur la surface de la zone de placement. Notre problématique générale est de pouvoir **estimer rapidement** l'efficience d'un placement, donnée par un algorithme de placement, **sans exécuter cet algorithme**, en fonction de la laize, des pièces à y placer et des contraintes qui s'appliquent.

3 Modélisation de l'estimation de l'efficience

Pour estimer l'efficience d'un placement sans le réaliser, nous utilisons une base d'apprentissage de placements sur laquelle nous appliquons des techniques de régression en apprentissage automatique. Nous décomposons le problème d'estimation d'efficience ainsi : 1) modélisation des pièces, 2) modélisation d'ensembles de pièces, et 3) modélisation des relations et contraintes entre pièces.

Cet article traite des deux premiers points. Le troisième est abordé dans la discussion finale (partie 5) pour nos futurs travaux.

3.1 Modélisation haut-niveau

Pour répondre à ces deux premiers points, nous proposons l'architecture présentée dans la figure 2. Elle se base sur un modèle multi-entrées et combine plusieurs modules spécifiques. Le premier module, illustré en figure 2 (a), est dédié à la modélisation des pièces. Il est spécialisé dans la reconnaissance de formes et repose sur une succession de convolutions. Son rôle est d'extraire les caractéristiques saillantes de chaque image, afin

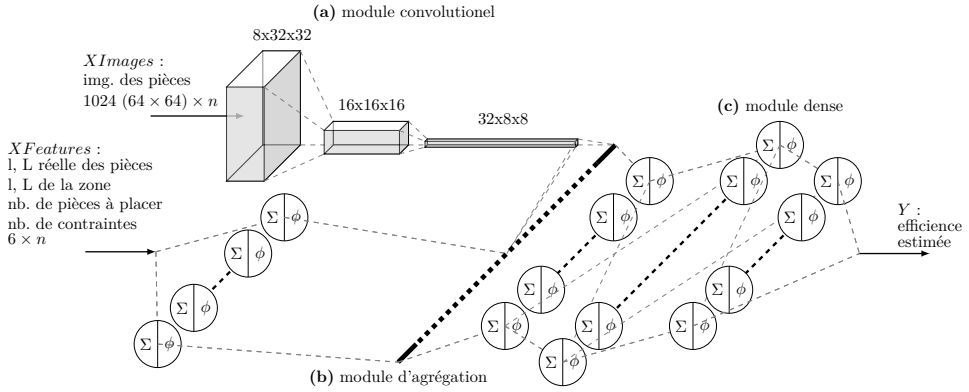


FIG. 2 – *Schéma du modèle.* (a) *: module de convolutions, dédié à la modélisation des pièces.* (b) *: module d'agrégation MIL Pool, dédié à la modélisation d'ensembles de pièces.* (c) *: module dense, dédié à la décision. Ici* n *est le nombre de pièces d'un placement.*

d'en créer une représentation simplifiée, sous la forme d'un vecteur basse dimension. Il prend en entrée, pour chaque placement, la liste des images rasterisées des pièces du placement (notées $XImages$) et fournit en sortie un vecteur pour chaque image appelé *embeddings* (voir figure 3). Par la suite, nous y ferons référence comme "module de convolutions".

Le second module, illustré en figure 2 (b), est dédié à la modélisation d'ensembles de pièces. Son rôle est de créer un vecteur unique à faible dimension à partir des listes de vecteurs fournies par le module précédent. Il prend en entrée les représentations construites par le module de convolutions, ainsi que les vecteurs de caractéristiques associés à chaque image des pièces (notées $XFeatures$) et fournit en sortie un vecteur pour chaque placement. Par la suite, nous y ferons référence comme "module d'agrégation".

Le dernier module est un module dense, illustré en figure 2 (c), dédié à la décision. Il prend en entrée le vecteur de sortie du module d'agrégation, représentant le placement et effectue la régression vers une valeur scalaire. Cette valeur représente l'efficience prédite du placement. La fonction de coût est calculée sur cette valeur. Par la suite, nous y ferons référence comme "module dense".

3.2 Modélisation des pièces

La reconnaissance de formes est généralement effectuée par des réseaux de neurones convolutionels (CNNs, LeCun et al. (1999)). Ce sont des réseaux hiérarchiques à multiples couches. Chaque couche transforme les données d'entrée via un ensemble de noyaux de convolutions, appelés filtres, qui vont être optimisés pendant l'apprentissage. Ces réseaux se basent sur les concepts de **champs réceptifs locaux**, de **partage des poids** et de **sous-échantillonage spatial** pour dégager une invariance spatiale partielle entre les pièces. Les champs réceptifs locaux sont définis par le fait que chaque

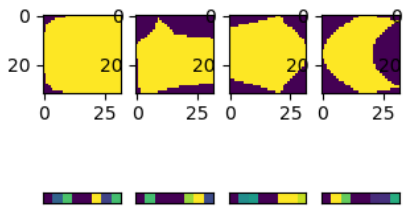


FIG. 3 – *Exemple de données du module de convolutions de la Figure 2-(a). En haut, les images correspondant aux formes des pièces rasterisées, en entrée du module. En bas, les embeddings de sortie, générés par le module.*

unité, ou neurone, d'une couche prend en entrée les valeurs de sortie des unités de la couche précédente situées dans son voisinage. Cette opération a pour but d'extraire des descripteurs visuels élémentaires locaux tels que des bordures ou des coins. Ces descripteurs sont ensuite combinés dans les couches plus profondes pour répondre à des caractéristiques de plus haut niveau (combinaisons de bordures et de coins). Cette idée d'invariance spatiale est inspirée par la découverte de neurones réagissant localement à des stimulus visuels précis dans le cortex visuel du chat (Hubel et Wiesel, 1959). Les unités d'une couche partagent le même ensemble de poids. Elles sont ainsi contraintes à apprendre et appliquer les mêmes filtres par couche. Le sous-échantillonage (ou pooling) est une méthode pour réduire la résolution spatiale des représentations intermédiaires (appelées cartes de caractéristiques). À chaque couche du réseau, les unités appliquent une opération d'agrégation sur les entrées de leur champ réceptif pour réduire la résolution spatiale. Cela permet, de plus, de construire des relations spatiales entre les descripteurs comme le modèle prototypique de ces architectures, baptisé LeNet (LeCun et al., 1999; Lecun et al., 1998).

L'architecture moderne des CNN a été proposée avec Alexnet (Krizhevsky et al., 2017) sous la forme d'une alternance de couches de convolutions et de couches d'agrégation *maximum*, appelées max-pooling. Via un balayage par fenêtrage des unités de la carte de caractéristiques, cet opérateur considère le maximum des valeurs du voisinage de chaque unité comme nouvelle représentation. Ce fenêtrage est appliqué avec un pas pour réduire la taille de la carte de caractéristiques et ainsi compresser l'information. Chacune de ces couches est suivie d'une fonction d'activation, dont le but est de filtrer la sortie de chaque unité, généralement en fonction d'un seuil. Cette architecture est, elle aussi, suivie d'un petit nombre de couches denses, dans lesquelles chaque unité est connectée aux unités de la couche précédente, sans notion de voisinage (Springenberg et al., 2015; Khan et al., 2020).

Nous allons maintenant présenter notre approche de modélisation des ensembles de tailles variables de ces pièces.

3.3 Modélisation d'ensembles de pièces

Plusieurs familles de méthodes classiques sont utilisables pour modéliser des ensembles, par exemple, les LSTM -*Long Short-Term Memory* (Hochreiter et Schmidhuber, 1997) ou les *GRU Gated Recurrent Units* (Cho et al., 2014). Ces méthodes sont issues de champs de recherches particuliers (traitement du langage, séries temporelles) et nécessitent généralement que les données soient ordonnées.

Dans cet article, nous nous intéressons aux méthodes du type *Multiple Instance Learning* ou MIL qui sont des formes d'apprentissages semi-supervisés. Nous pensons que ces méthodes sont plus adaptées aux données d'ensembles de pièces, qui n'ont pas de notion d'ordre. L'hypothèse standard en MIL est qu'un label est fourni pour un ensemble d'instances. Chacun de ces ensembles est appelé *bag*. Cette méthode est semi-supervisée, car les labels ne sont connus qu'au niveau des bags et seulement partiellement au niveau des instances (les données d'entrées). Un bag est considéré comme positif si a minima une de ses instances est positive. Un bag est considéré comme négatif si l'ensemble des instances qu'il contient sont négatives (Carbonneau et al., 2018).

Nous considérons le MIL comme une approche permettant de comparer des ensembles de tailles variables non ordonnés : le principe est d'agréger un ensemble d'instances (un ensemble de vecteurs) en un unique vecteur. Le label est associé à cette représentation agrégée pour permettre une classification ou une régression.

Dans les réseaux de neurones, on considère que les instances sont des caractéristiques extraites de couches de réseaux. On peut les voir comme des représentations basse dimension des vecteurs d'entrées, sur lesquelles on applique une agrégation de type MIL, appelé MIL Pool. Cette opération d'agrégation doit être invariante à l'ordre dans lequel les instances sont présentées (Ilse et al., 2018).

4 Architecture

Notre modèle est basé sur une architecture classique en apprentissage profond, dans laquelle on peut voir une partie extraction de caractéristiques suivie d'une partie de compression par agrégation (pooling), et enfin une partie décision (Lecun et al., 1998).

En se basant sur les architectures des CNN présentées précédemment, nous avons choisi pour le module de convolution de s'inspirer du modèle LeNet en y apportant les fonctions d'activations proposées par Alexnet et en remplaçant les couches de sous-échantillonnages par un pas de fenêtrage (ou *stride*) ayant une plus basse complexité computationelle. Le module de convolution est composé de trois couches de convolutions, utilisant chacune une fonction d'activation de type ReLU (Rectified Linear Unit). Chacune de ces couches a une taille de voisinage de deux et un pas de fenêtrage (ou stride) de deux permettant la réduction d'informations à chaque couche.

4.1 Implémentation

L'implémentation, basée sur Pytorch (Paszke et al., 2019), prend en entrée des lots de placements où chaque placement est décrit à l'aide de $XImages$ et de $XFeatures$. Les $XImages$ sont des images rasterisées des pièces du placement en résolution 64×64 pixels. Les $XFeatures$ sont des caractéristiques de ces pièces : largeur et longueur réelle de chaque pièce, taille de la zone de placement, nombre de pièces à placer et nombre de contraintes. Les dimensions pour $XImages$ sont de $1024 \times n$ et pour $XFeatures$ sont de $6 \times n$. Avec n le nombre de pièces du placement.

Les tailles des pièces et la taille de la zone du placement sont fournies avec une précision de l'ordre du dix-millième de mètre.

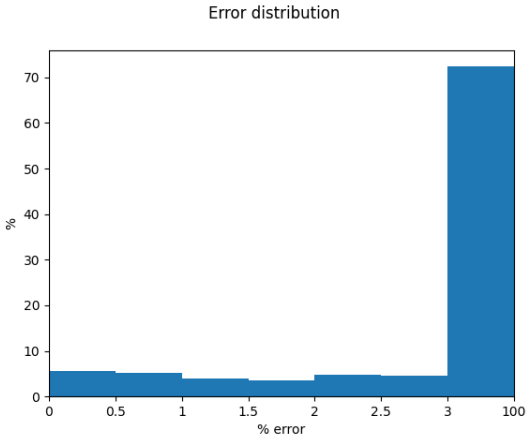


FIG. 4 – *Distribution de l'erreur absolue du modèle, par pas de 0.5% en dessous de 3% puis au-delà de 3%*

Une étude antérieure relative à l'identification de pièces de patrons montre qu'il y a un compromis à faire pour les dimensions des images des pièces utilisées en entrée entre la qualité et les temps d'entraînement : plus ces images sont grandes, plus la qualité de la reconnaissance augmente, ainsi que la complexité, le temps et l'espace des calculs. Au-delà d'une résolution de 64×64 pixels, la reconnaissance de pièces ne bénéficie plus de l'apport d'informations fournies (Jouanneau, 2019). La sortie du module de convolution, appelée embedding, est un vecteur à 16 dimensions.

4.2 Résultats préliminaires

La base de données utilisée comporte 100000 placements. Nous entraînons notre modèle sur 90000 placements dont 10% servent (soit 9000) à la validation à chaque fin de cycle d'apprentissage (ou *epochs*). Les 10000 placements restants sont utilisés pour la phase d'évaluation. L'apprentissage est fait sur cinq epochs, car l'amélioration des résultats de l'étape de validation stagnait. Nous utilisons Adam *Adaptative Moment Estimation* (Kingma et Ba, 2014) comme méthode d'optimisation de descente de gradient avec un taux d'apprentissage de 0.01 et la distance $L1$ (moyenne des différences absolues entre les prédictions et les efficiences réelles) comme fonction de coûts.

Pour l'évaluation, nous utilisons trois métriques courantes (Botchkarev, 2019) et une métrique métier de distribution des erreurs. Le modèle produit une erreur quadratique moyenne (Mean Squared Error, MSE) faible (1.7%), mais une erreur moyenne (Mean Absolute Error, MAE) élevée ($7,75\%$ d'erreur). L'erreur maximale (soit 82.6% dans le pire cas) est simplement inacceptable. La distribution de l'erreur absolue en fonction des placements évalués est représentée dans la figure 4. Cet histogramme montre la répartition d'erreur par pas de 0.5% entre 0 et 3% d'erreur, puis au-delà de 3%. Cette répartition est utilisée comme métrique de comparaison métier. Notre but est d'avoir une erreur moyenne en dessous de 3% et idéalement autour de 1% pour être utilisable en pratique dans l'optimisation des plans de sections.

5 Conclusion et travaux futurs

Cet article est une première approche de modélisation et d'implémentation d'un problème d'estimation d'efficience de processus de découpe industrielle de matières textiles. Ces premiers résultats sont plutôt décevants, mais ouvre la voie a de futures expérimentations.

Tel que présenté, l'erreur moyenne des prédictions du modèle est de $7,8\%$, notre cible étant en dessous de 3% pour l'optimisation des plans de sections. Nous pensons qu'un des problèmes principaux est lié à la distribution des exemples d'apprentissage. En effet, notre base d'apprentissage est créée à partir de données réelles et est donc biaisée, les placements globalement efficients étant sur-représentés par rapport aux autres. La distribution des efficiences des placements a une moyenne supérieure à 70% et un écart-type faible (environ 10). Une solution serait de ré-échantillonner de manière plus homogène notre base d'apprentissage pour être plus représentatif des cas plus rares en particulier une faible efficience ($< 70\%$).

Un autre point d'amélioration serait la prise en compte des contraintes, comme présenté dans la partie 3. Elles ne sont pas utilisées actuellement. On peut voir le problème de la représentation des contraintes comme un problème de représentation de graphe, dans lequel les contraintes sont des relations entre les pièces à placer. Dans cette optique, les ensembles formes-contraintes sont représentées par un graphe dans lequel les pièces sont les sommets et les contraintes sont les arêtes du graphe. Prédire une efficience donnée revient à un problème de comparaison de graphes (par exemple en classification ou en régression). Le champ de recherche lié à ce domaine est celui de l'apprentissage géométrique. Les modèles d'apprentissage profond sont performants dans les traitements du langage, des images ou des vidéos pour lesquels les données ont une structure euclidienne. Récemment, une nouvelle famille de techniques est apparue, appelée apprentissage géométrique profond (Bronstein et al., 2017) et essayant de généraliser l'application des modèles d'apprentissages profonds aux données non-euclidiennes comme les graphes (ex : réseaux sociaux, réseaux génétiques, conception de molécules (Duvenaud et al., 2015), systèmes de recommandation) ou des géométries à plus de deux dimensions (modèles de déformations 3D, simulations physiques 3D (Sanchez-Gonzalez et al., 2020)) Deux approches principales sont utilisées pour ces modèles, soit de créer une représentation ou *embedding* pour chaque sommet (ex : Graph Convolution Network, GraphSage, Gated Graph Neural Networks), soit de créer des représentations de sous-graphes. Nous proposons d'explorer ces méthodes pour nos futurs travaux.

Le troisième point concerne les variations autour du modèle actuel. Particulièrement autour de la partie MIL Pool. Le MIL est un champ de recherche large et très documenté. Son application aux réseaux de neurones est assez récente et certains opérateur d'agrégation pourraient améliorer les performances de notre modèle, par exemple l'approche attentionelle présentée dans Ilse et al. (2018). Une autre approche intéressante serait de décomposer notre modèle en sous-modèles puis de les pré-entraîner pour une tâche spécifique, comme la reconnaissance de forme, puis de recombiner ces modèles, à la manière du *transfer learning*.

Références

Botchkarev, A. (2019). Performance metrics (error measures) in machine learning regression, forecasting and prognostics : Properties and typology. *Interdisciplinary Journal of Information, Knowledge, and Management 14*, 045–076.

Bronstein, M. M., J. Bruna, Y. LeCun, A. Szlam, et P. Vandergheynst (2017). Geometric deep learning : going beyond euclidean data. *IEEE Signal Processing Magazine 34*(4), 18–42.

Carbonneau, M.-A., V. Cheplygina, E. Granger, et G. Gagnon (2018). Multiple instance learning : A survey of problem characteristics and applications. *Pattern Recognition 77*, 329–353.

Cho, K., B. van Merriënboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, et Y. Bengio (2014). Learning phrase representations using RNN encoder–decoder for statistical machine translation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, Doha, Qatar, pp. 1724–1734. Association for Computational Linguistics.

Duvenaud, D., D. Maclaurin, J. Aguilera-Iparraguirre, R. Gómez-Bombarelli, T. Hirzel, A. Aspuru-Guzik, et R. P. Adams (2015). Convolutional networks on graphs for learning molecular fingerprints. In C. Cortes, N. D. Lawrence, D. D. Lee, M. Sugiyama, et R. Garnett (Eds.), *Advances in Neural Information Processing Systems*, Volume 28, pp. 2224–2232. Curran Associates, Inc.

Henninger, C. E., P. J. Alevizou, et C. J. Oates (2016). What is sustainable fashion ? *Journal of Fashion Marketing and Management : An International Journal 20*(4), 400–416.

Hochreiter, S. et J. Schmidhuber (1997). Long short-term memory. *Neural computation 9*(8), 1735–1780.

Hubel, D. H. et T. N. Wiesel (1959). Receptive fields of single neurones in the cat's striate cortex. *The Journal of Physiology 148*(3), 574–591.

Ilse, M., J. Tomczak, et M. Welling (2018). Attention-based deep multiple instance learning. In J. Dy et A. Krause (Eds.), *Proceedings of Machine Learning Research*, Volume 80, pp. 2127–2136. PMLR.

Jouanneau, W. (2019). Identification de patrons de vêtements et de pièces par machine learning. Rapport confidentiel interne, EINSEIRB-MATMECA / LECTRA.

Khan, A., A. Sohail, U. Zahoora, et A. S. Qureshi (2020). A survey of the recent architectures of deep convolutional neural networks. *Artificial Intelligence Review 53*, 5455–5516.

Kingma, D. et J. Ba (2014). Adam : A method for stochastic optimization. In *International Conference on Learning Representations*.

Krizhevsky, A., I. Sutskever, et G. E. Hinton (2017). ImageNet classification with deep convolutional neural networks. *Communications of the ACM 60*(6), 84–90.

Lecun, Y., L. Bottou, Y. Bengio, et P. Haffner (1998). Gradient-based learning applied to document recognition. *Proceedings of the IEEE 86*(11), 2278–2324. Conference

Name : Proceedings of the IEEE.

LeCun, Y., P. Haffner, L. Bottou, et Y. Bengio (1999). Object recognition with gradient-based learning. In D. A. Forsyth, J. L. Mundy, V. di Gesú, et R. Cipolla (Eds.), *Shape, Contour and Grouping in Computer Vision*, Lecture Notes in Computer Science, pp. 319–345. Springer.

Paszke, A., S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, A. Desmaison, A. Kopf, E. Yang, Z. DeVito, M. Raison, A. Tejani, S. Chilamkurthy, B. Steiner, L. Fang, J. Bai, et S. Chintala (2019). PyTorch : An imperative style, high-performance deep learning library. 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. 8026–8037. Curran Associates, Inc.

Rissanen, T. et H. McQuillan (2016). *Zero waste fashion design*, Volume 57. Bloomsbury Publishing.

Sanchez-Gonzalez, A., J. Godwin, T. Pfaff, R. Ying, J. Leskovec, et P. W. Battaglia (2020). Learning to simulate complex physics with graph networks. In *Thirty-seventh International Conference on Machine Learning (ICML)*.

Springenberg, J., A. Dosovitskiy, T. Brox, et M. Riedmiller (2015). Striving for simplicity : The all convolutional net. In *International Conference on Learning Representations (ICLR, workshop track)*.

Summary

In an industrial process of textile materials cutting, the minimization of both material losses and cutting times are major concerns. Time can be minimized by cutting a stack of material. However, optimizing the material is a complex problem. In order to be cut, parts have to be grouped into sets. Obtaining the most efficient material groupings requires the calculation of material consumption for each possible set. Efficiency is given by a nesting algorithm that positions parts on the material. This algorithm is expensive in time and money, making it impossible to calculate all the possible sets. Currently, the problem is solved using empirically designed reference tables which approximate the material utilization rate for each set.

In this paper, we present the work of my first year of PHD: a first deep learning approach to estimate the efficiency of the nesting of a set from a database built on existing nestings. We propose a combination of techniques from modeling the parts using convolutional neural networks and modeling a set of parts using *Multiple Instance Learning*. We also present the preliminary results related to this approach and our futur work.

Visualisation d'un ensemble d'images et de sa représentation interne dans un réseau de neurones convolutionnel

Barthélémy Serres[***], Fatma Bouali[*,**], Christiane Guinot[*],
Maïwenn Colin[*], Florian Le Meur[*], Gilles Venturini[*]

[*]LIFAT, Université de Tours
[**]IUT C, Université de Lille
[***]CETU ILIAD3, Université de Tours

Résumé. Nous présentons dans cet article une approche préliminaire pour explorer visuellement certaines propriétés des données (images) et de leur représentation interne apprise par un réseau de neurones convolutionnel. Nous observons les caractéristiques extraites par le réseau juste avant la couche de classification. Nous construisons un graphe de voisinage à partir de cet espace vectoriel en connectant les données proches topologiquement. Nous définissons quelques exemples de propriétés topologiques à détecter (points isolés, points erronés, frontières des classes). Ensuite nous proposons une visualisation de ce graphe soulignant ces informations et offrant une vue d'ensemble du graphe (groupes de données). Cette visualisation inclut une représentation des images afin de permettre à l'utilisateur de comprendre ce qui peut causer une erreur (images mal acquises, erreurs dans le pré-traitement ou l'étiquetage, choix du réseau, résultat de l'apprentissage etc). Nous avons réalisé des tests sur le réseau VGG16 et sur des extraits d'ensembles de données classiques.

1 Introduction

Le point de départ de ce travail a été une expérience en cours avec un jeu de $n = 1072$ images et le réseau VGG19. Il s'agit d'un problème de détermination d'espèces animales très similaires entre elles (de type "fine-grained recognition" (Gebru et al., 2017)) et qui jusqu'à présent est résolu de manière coûteuse par des experts humains. Cette tâche de classification est difficile (86% de bonne prédiction avec VGG19, en utilisant du "transfer learning" pour les couches convolutionnelles et un ré-apprentissage de la partie classification). Pour comprendre ce "mauvais" résultat, nous avons cherché à visualiser les images selon les descripteurs internes appris par ce réseau. En utilisant les caractéristiques que VGG19 extrait de ces images (en dimension p = 4096, valeurs produites par la dernière couche convolutionnelle, juste avant les couches de classification), nous avons construit une première visualisation 2D des données, en combinant des approches topologiques (graphes de voisinage) avec une visualisation de graphe incluant une représentation des images elles-mêmes. Nous avons alors trouvé des erreurs d'étiquetage, ainsi que d'autres problèmes dans les données d'apprentissage (des éléments parasites qui se retrouvaient sur des images de différentes classes, et qui augmentaient inutilement la similarité inter-classe), erreurs confirmées par l'expert une fois qu'elles lui ont été soulignées.

Nous avons alors cherché à étudier plus en avant la problématique de la visualisation pour l'analyse des propriétés d'un CNN et des données traitées. Notre intérêt s'est renforcé aussi du fait que 1) les réseaux convolutionnels (CNNs) se démocratisent et sont maintenant mis en oeuvre par des utilisateurs "non experts", ces derniers ayant besoin d'analyser les difficultés rencontrées (notamment celles liées à leur données), et 2), la communauté scientifique exprime le besoin de comprendre les représentations internes complexes découvertes par ces réseaux de neurones profonds (Ras et al., 2018).

Notre travail est préliminaire et se focalise sur ce point : peut-on évaluer, visuellement si possible, la qualité des images qui servent à l'apprentissage et l'efficacité de leur représentation interne dans un CNN ? La suite de cet article est organisée de la manière suivante : la section 2 présente un état de l'art sur les approches qui essaient de visualiser l'espace de représentation interne appris par un CNN. Dans la section 3, nous détaillons notre approche qui s'appuie sur la construction d'un graphe de voisinage et la détection puis la visualisation de certaines propriétés topologiques dans ce graphe. Dans la section 4 nous présentons des résultats sur quelques jeux de données classiques (des sous-ensemble de taille réduite) et nous concluons dans la section 5 en présentant les perspectives de ce travail.

2 Etat de l'art

Nous présentons ici quelques références parmi un ensemble croissant de travaux dans le domaine de la visualisation pour l'analyse des CNNs (voir un survol dans (Hohman et al., 2018)). Les visualisations peuvent être utilisées à différentes étapes de l'apprentissage. Par exemple il est possible de représenter visuellement l'architecture du réseau comme dans (Wongsuphasawat et al., 2017). Concernant les approches qui cherchent à analyser les propriétés des caractéristiques extraites (et donc de la représentation interne), Summit (Hohman et al., 2019) propose un outil interactif (fredhohman.com/summit) qui permet d'expliciter quelles caractéristiques sont utilisées pour une classe donnée. Il propose une vue de type projection sur les classes plutôt que les images fournies en entrée. Cette vue permet de voir quelles classes se retrouvent les unes à coté des autres, selon les caractéristiques qu'elles partagent. Dans (Rauber et al., 2016), une projection des données utilisant l'algorithme t-SNE permet de comprendre les relations et groupes qui peuvent se former dans les données selon les caractéristiques internes du CNN. Notre travail se rapproche de cette étude, cependant les auteurs ne cherchent pas à inclure les images dans la visualisation ni à détecter automatiquement des propriétés topologiques (car t-SNE peut commettre des erreurs de projection, alors qu'un graphe de voisinage construit des relations selon une propriété topologique fixe). Cette caractéristique des graphes de voisinage a d'ailleurs été utilisée dans (Ide et Uchida, 2017), permettant ainsi de savoir visuellement si un CNN fusionne ou non les représentations des données de sources différentes (et dans quelle couche interne cette fusion peut avoir lieu). Tensorflow dispose également d'un outil de visualisation des données selon les caractéristiques internes du CNN, en incluant des imagettes (voir projector.tensorflow.org).

Pour situer notre travail par rapport à l'existant, nous souhaitons montrer qu'un graphe de voisinage permet d'aller plus loin que les approches utilisant d'autres types de projection de données : en calculant des propriétés topologiques sur les données, il est possible non seulement de les visualiser mais aussi de détecter automatiquement certaines propriétés, afin de ne pas surcharger l'utilisateur dans sa tâche visuelle. Un autre point qui nous distingue de certains

travaux est le fait de vouloir visualiser conjointement les images et les propriétés détectées, une étape nécessaire si l'on veut que l'expert puisse analyser les propriétés détectées et les relier à son domaine. Enfin, peu de travaux s'intéressent finalement à un point important pour les utilisateurs non spécialisés : la qualité des images collectées en les observant par le prisme de la représentation interne du CNN.

3 Approche proposée

Notre approche consiste à plonger les données dans l'espace de représentation interne d'un CNN, à construire des relations topologiques entre les données de cet espace, à détecter automatiquement certaines propriétés topologiques, et enfin à visualiser l'ensemble (images + propriétés) dans un outil interactif. Nous pensons d'après notre première expérience que la découverte de ces relations topologiques, associée à des visualisations, peut permettre de détecter certaines informations et problèmes sur cet espace : formation de groupes de données, relations et séparabilité entre ces groupes, situations anormales concernant les données, données isolées ou "entourées" de données d'autres classes. Par ailleurs, cette approche va permettre à l'utilisateur de faire le lien entre ces informations topologiques et les images correspondantes. Si deux images sont proches topologiquement et si, dans l'oeil de l'utilisateur expert, elles correspondent à des concepts très différents dans son domaine, cela peut révéler soit une erreur dans la collecte des images (images erronées, concept sous-représenté dans l'ensemble d'apprentissage), dans leur étiquetage ou plus généralement leur pré-traitement (erreur humaine, pré-traitement non adapté), soit une erreur dans le réseau (mauvais choix, mauvais apprentissage). Un des points clés de cette approche est donc la capacité à représenter visuellement et conjointement l'espace interne du CNN et les images correspondantes.

Pour construire cette visualisation, nous proposons la stratégie suivante :

1. Calcul de la représentation interne des n images (soit p la dimension de la dernière couche convolutionnelle[1]),
2. Construction d'un graphe de proximité sur cette représentation,
3. Détection de propriétés topologiques particulières sur les données pour faire ressortir les erreurs dans les données et la représentation,
4. Visualisation du graphe de proximité avec les propriétés détectées et les images.

Pour le point 2), la structure topologique que nous avons choisie de construire est un graphe de proximité (Bose et al., 2012). Les graphes de proximité ou graphes de voisinage permettent de construire un graphe à partir de données en reliant entre elles les données qui sont proches dans l'espace de représentation. Cette notion de proximité est liée à une distance, ici la distance euclidienne puisque l'espace de représentation interne du CNN est numérique. Parmi ces graphes, on peut citer certains qui sont très connus comme le graphe des k plus proches voisins (k-NNG), le graphe ϵ (ϵ-G), le graphe de Gabriel (GG), ou encore le graphe des voisins relatifs (RNG) (Toussaint, 1980) (Jaromczyk et Toussaint, 1992). k-NNG est intéressant cependant pour notre problématique il pose deux difficultés : d'une part il faut déterminer la bonne valeur

[1] On pourrait utiliser aussi une autre couche intermédiaire.

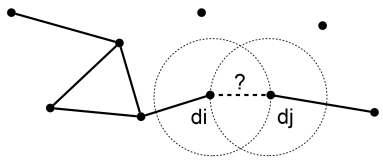


FIG. 1 – *Illustration de la construction d'un graphe de type voisins relatifs sur un ensemble de données. Une arête est créée entre deux données d_i et d_j si la zone d'exclusion (lunule) définie par l'intersection des sphères centrées en chaque donnée et de rayon $Dist(d_i, d_j)$ ne contient aucune autre donnée.*

de k (si k est trop petit, le graphe sera constitué de nombreuses composantes non connectées entre elles ; si k est trop grand toutes les données seront connectées et la structure ne va pas extraire d'informations intéressantes). D'autre part, k-NNG ne construit pas un graphe connexe, et dans l'affichage du graphe, s'il existe des composantes indépendantes, l'algorithme de dessin de graphe ne saura pas les positionner les unes par rapport aux autres et donc il ne donnera pas d'informations sur la proximité entre ces composantes. Le deuxième graphe considéré est ϵ-G qui consiste à élaguer la matrice de distances (les données qui sont à une distance inférieure à ϵ sont connectées). Cela peut être intéressant dans certains cas mais pour nos objectifs ϵ-G pose le problème du choix de ϵ et ne garantit pas la connexité du graphe. Pour le graphe de Gabriel, nous savons qu'en théorie il engendre plus d'arêtes que les autres, ce qui pourra créer une surcharge visuelle (voir la section Tests). Nous avons choisi par la suite le graphe des voisins relatifs. RNG a pour avantage de calculer toujours la même propriété topologique locale sur les données (voir Figure 1). Il construit un graphe connexe ce qui va faciliter l'affichage des données. Par ailleurs, il a d'autres propriétés intéressantes. Ppar exemple, il inclut par construction 1-NNG (Bose et al., 2012). Il a cependant un inconvénient qui ne sera pas abordé dans cet article (voir les perspectives et nos travaux précédents avec des algorithmes parallélisés sur GPU (Liu et al., 2014)) : sa construction est en $O(n^3)$. Par le passé nous avons montré par exemple comment RNG permet de détecter des groupes et autres propriétés dans un ensemble de documents (Liu et al., 2014) ou d'images (Rayar et al., 2018).

Pour le point 3), nous avons décidé d'extraire un premier ensemble de propriétés topologiques. Nous avons cherché à obtenir les données qui semblent non correctes : les points isolés connectés à un seul autre point et à une distance supérieure à un seuil (nous avons pris la somme de la moyenne des distances + l'écart type), et les points "erronés" qui sont entourés de points d'une autre classe (au moins 4 données connectées à ce point et de classes différentes). Pour les points isolés, ces données sont topologiquement à la périphérie et donc plus éloignées du centre de la distribution. Pour les points erronés, ces données ne sont pas topologiquement à leur place puisque leurs voisins sont de classes différentes. Pour terminer, afin de représenter la frontière entre les classes, nous étiquetons les arêtes selon qu'elles connectent des données de même classe ou de classes différentes.

Pour le point 4), il s'agit de représenter visuellement pour l'utilisateur toutes les informations détectées : l'espace de représentation interne dans sa globalité, les structures topologiques détectées et les problèmes associés, la relation entre ces informations et les images. Nous souhaitons que l'utilisateur puisse interagir avec cette visualisation pour explorer cette représen-

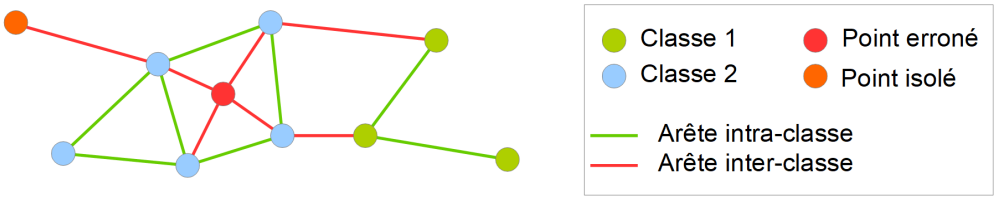


FIG. 2 – *Choix de représentation visuelle des propriétés extraites sur le graphe de proximité. Par soucis de clarté, dans cette figure nous n'avons pas représenté d'images, qui seront visualisées sous la forme de vignettes à l'intérieur des noeuds (voir la section Tests).*

tation visuelle. Nous avons adopté une représentation de type noeud-lien (voir Figure 2) :

- un noeud représente une donnée. Un noeud contient une vignette de l'image qu'il représente. La bordure d'un noeud est colorée selon la classe. Si un noeud est un point isolé, sa bordure est d'une autre couleur (orange), et si c'est un point erroné, sa bordure est rouge. Le principe utilisé ici est de permettre une perception pré-attentive de ces points particuliers, et d'utiliser des codes couleurs aux propriétés connues (voir www.colorbrewer2.org). Nous avons testé d'autres choix qui se sont révélés moins efficaces (comme la forme des noeuds par exemple). Un label texte peut être utilisé pour indiquer le nom de la classe d'un noeud,
- une arête représente une relation topologique. Les arêtes intra-classes sont en vert, les arêtes inter-classes sont en rouge.

Nous utilisons ensuite le logiciel Tulip (tulip.labri.fr) pour représenter ce graphe avec l'algorithme FM^3 (Hachul et Jünger, 2005).

4 Tests

Nous avons utilisé le jeu CIFAR-10 (Krizhevsky et al., 2009) en prenant 6000 images issues de trois classes (Car, Airplane, Ship). Compte tenu des performances assez basses en classification, nous avons choisi trois classes avec des chevauchement modérés par rapport aux autres. Nous avons choisi le réseau VGG16 ($p = 4096$) (Simonyan et Zisserman, 2014). En construisant le graphe (13182 arêtes) et les informations que nous avons proposées, on obtient une représentation peu lisible du fait du grand nombre d'arêtes (voir Figure 3(a)). Nous avons donc enlevé l'affichage des arêtes dans la vue globale pour obtenir une représentation où la structure globale ainsi que les données non correctes se détectent nettement plus facilement (voir Figure 3(b)). Ensuite nous avons observé des détails sur les données non correctes. Pour cela, Tulip dispose d'une fonction permettant de visualiser localement le voisinage d'un noeud, et d'explorer ce voisinage de proche en proche. Sur cette vue de détail, on voit les images, ce qui va aider l'utilisateur à comprendre les problèmes qui se posent. Par exemple, Figure 3(c)), nous nous sommes focalisés sur une donnée erronée selon nos critères, et en observant son voisinage, on peut constater qu'une nette composante rouge apparaît de manière commune

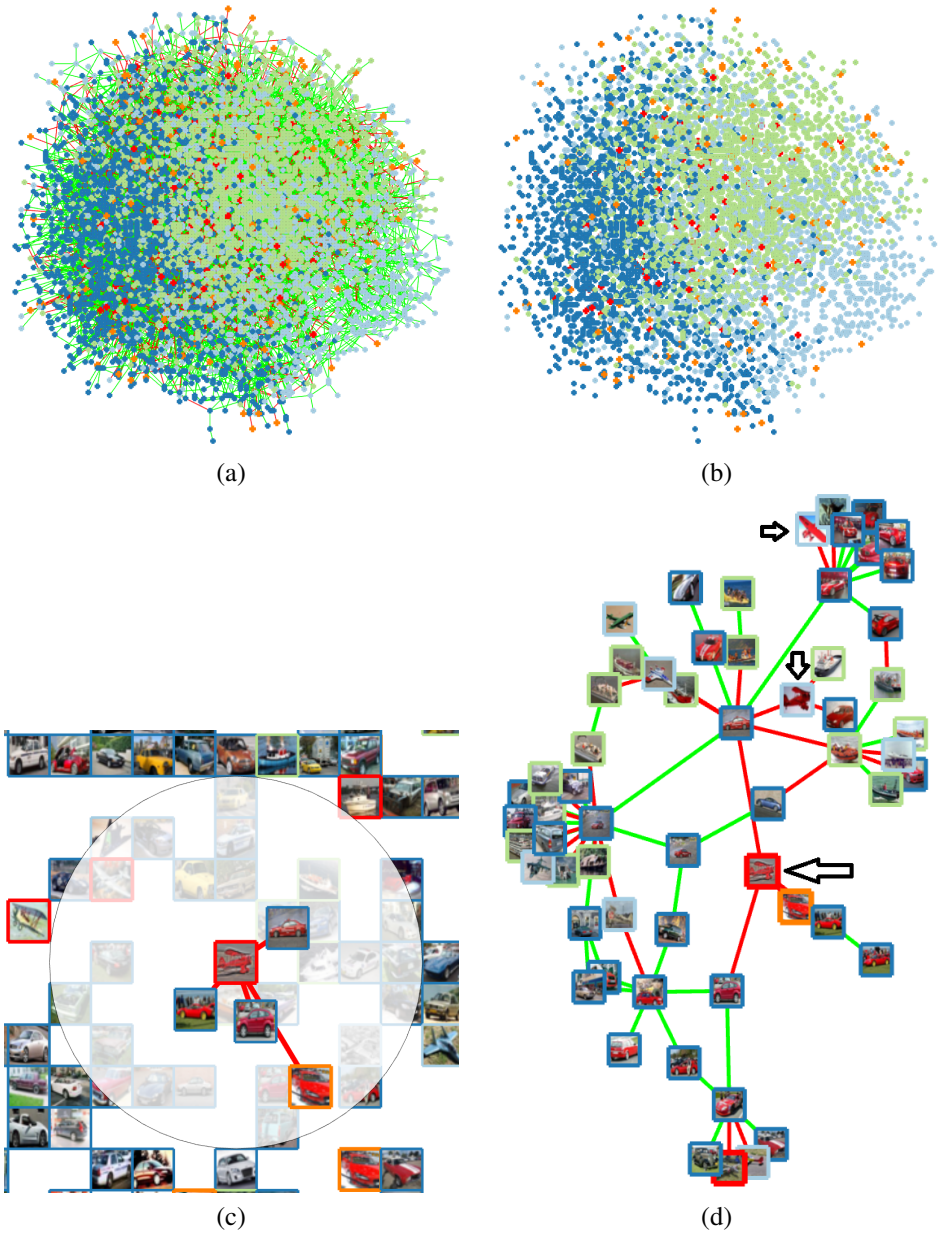


FIG. 3 – *Vues globales (6000 images CIFAR-10) avec (a) et sans arête (b), avec les données non correctes en rouge et orange. Focalisation sur un point erroné en (c), dont l'image partage une même couleur rouge avec ses voisines d'autres classes. Exploration du graphe au voisinage de ce point en (d) (indiqué par la grande flèche), où l'on trouve d'autres avions rouges mélangés avec des voitures rouges (petites flèches).*

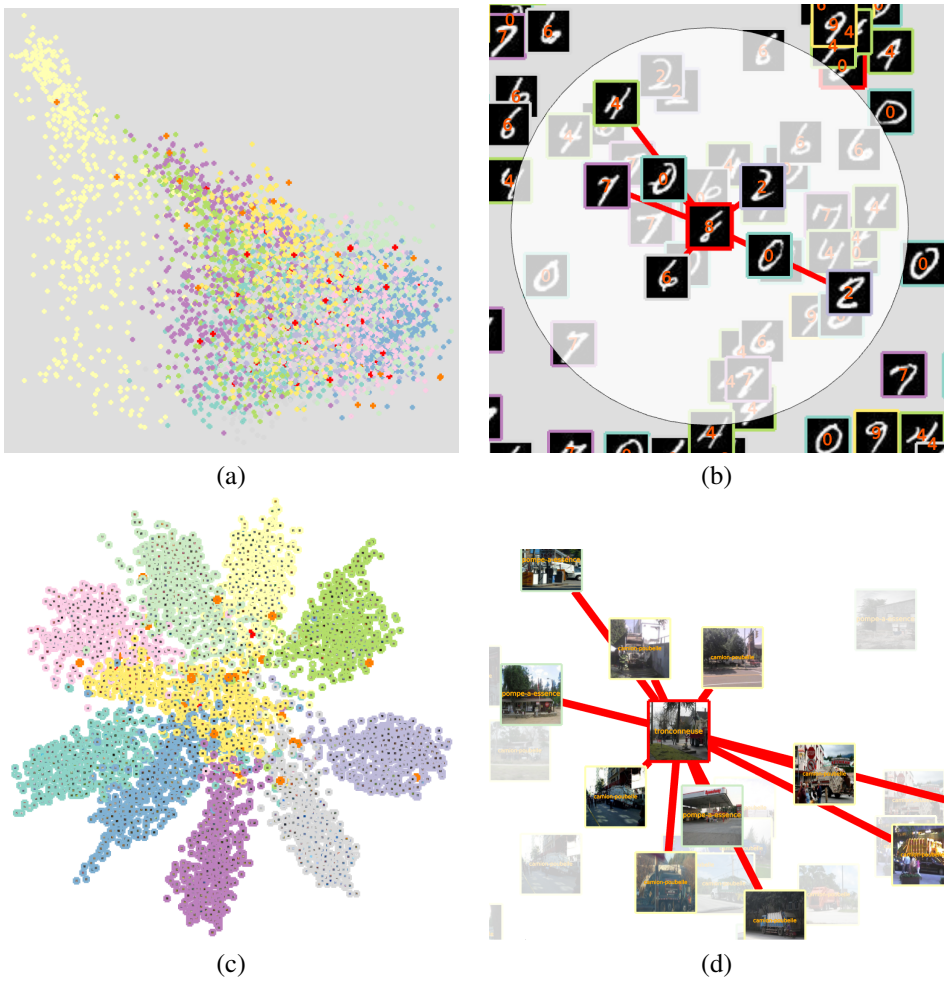


FIG. 4 – *Vue globale (5000 images MNIST) en (a) et détails sur une image détectée comme erronée (b) où l'on vérifie que l'image est effectivement difficile à interpréter (chiffre 8). Même représentation pour un extrait d'ImageNet en (c) avec un exemple d'image erronée (d) "tronçonneuse" entourée de 7 images "camion-poubelle" (les roues et la remorque présentes dans l'image en sont peut-être la cause).*

dans les images de cette partie du graphe. Cela peut donc amener l'utilisateur à s'interroger sur l'influence de cette caractéristiques dans cette erreur. En explorant le voisinage de cette donnée (Figure 3(d)), on s'aperçoit que d'autre avions rouges se confondent autour des voitures rouges.

Dans un deuxième test, nous avons utilisé la base MNIST en sélectionnant 5000 images. Après la construction du graphe (12372 arêtes), 62 données erronées ont été détectées, et 23 données isolées (voir Figure 4). Avec cette base, l'interprétation des propriétés découvertes visuellement est plus simple que pour CIFAR-10 du fait que les images soient des chiffres manuscrits. Un inspection visuelle des 62 données erronées montre qu'au moins 21 d'entre elles sont visuellement et sans équivoque des chiffres très mal formés. On observe une très bonne correspondance entre les propriétés topologiques détectées et la qualité des images. Le fait que des données non correctes soient détectées et soulignées automatiquement facilite l'exploration visuelle, l'utilisateur n'ayant pas à chercher ces données.

Dans un dernier test nous avons utilisé la base Imagenette (github.com/fastai/imagenette), avec 3925 images 320×320. Il s'agit d'une sélection facilement séparable issue de 10 classes de la base ImageNet. Le graphe construit a 5959 arêtes, et sa représentation dans VGG16 fait nettement apparaître la séparabilité (voir Figure 4(c)). On trouve 36 noeuds isolés, et très peu de noeuds erronés (5) selon notre critère. Pour l'un de ces noeuds de classe "tronçonneuse" (voir Figure 4(d)), on constate qu'une remorque (ressemblant à un camion) est présente, ce qui peut expliquer que cette image soit par exemple entourée de 7 images de camion-poubelle. Un utilisateur expert dans un domaine métier pourrait par exemple décider que cette image est trop atypique et la supprimer de l'ensemble d'apprentissage, ou inversement, que cette combinaison arrive assez fréquemment et essayer d'obtenir plus d'images de ce type pour mieux la séparer des autres.

5 Conclusions et perspectives

Nous avons présenté une approche visuelle aidant à comprendre comment la représentation interne apprise par un CNN peut structurer un jeu de données images. Pour cela nous avons choisi de construire un graphe de proximité connexe sur les données (RNG) afin d'expliciter la topologie locale autour de chaque donnée. Ensuite, pour alléger la tâche de l'utilisateur, nous avons détecté automatiquement certaines propriétés topologiques locales susceptibles de révéler des erreurs à différents niveaux (pré-traitement des données, étiquetage, configuration et apprentissage du réseau). Notre outil permet ensuite de représenter visuellement l'ensemble des données en corrélant les informations détectées avec les images, cela dans le but d'aider l'utilisateur à comprendre d'où peuvent venir les propriétés observées. Nous avons testé notre outil sur des sous-ensembles d'images, avec un nombre réduit d'images.

Concernant les perspectives, nous allons proposer la détection de nouvelles propriétés topologiques. Pour mieux comprendre qu'elles propriétés rechercher, nous allons définir une interface pour filtrer dynamiquement les données en fonction de leurs propriétés topologiques. Il serait possible également d'inclure les classes prédites parmi les critères de filtrage, si l'utilisateur souhaite analyser les erreurs de classifications. Ensuite, l'approche proposée devra traiter plus de données. Nous avons montré dans (Rayar et al., 2018) qu'un graphe de voisinage peut être construit hiérarchiquement sur un grand ensemble d'images (ImageNet, 15 millions d'images). Nous allons chercher à étendre ces travaux pour visualiser plus d'images, sans perdre le détail des relations topologiques. Enfin, notre outil peut suggérer des explica-

tions, qui auront un sens ou non dans l'oeil de l'expert qui observe les images. Une approche comme (Hohman et al., 2019) permettrait d'aller plus loin et de voir si ces suggestions extraites par le graphe et qui intéressent l'expert peuvent être interprétées en terme de caractéristiques d'images.

Références

Bose, P., V. Dujmović, F. Hurtado, J. Iacono, S. Langerman, H. Meijer, V. Sacristán, M. Saumell, et D. R. Wood (2012). Proximity graphs : E, δ, δ, χ and ω. *International Journal of Computational Geometry & Applications 22*(05), 439–469.

Gebru, T., J. Hoffman, et L. Fei-Fei (2017). Fine-grained recognition in the wild : A multi-task domain adaptation approach. In *Proceedings of the IEEE International Conference on Computer Vision*, pp. 1349–1358.

Hachul, S. et M. Jünger (2005). Large-graph layout with the fast multipole multilevel method. *Spring, V (December)*, 1–27.

Hohman, F., M. Kahng, R. Pienta, et D. H. Chau (2018). Visual analytics in deep learning : An interrogative survey for the next frontiers. *IEEE transactions on visualization and computer graphics 25*(8), 2674–2693.

Hohman, F., H. Park, C. Robinson, et D. H. P. Chau (2019). Summit : Scaling deep learning interpretability by visualizing activation and attribution summarizations. *IEEE transactions on visualization and computer graphics 26*(1), 1096–1106.

Ide, S. et S. Uchida (2017). How does a cnn manage different printing types ? In *2017 14th IAPR International Conference on Document Analysis and Recognition (ICDAR)*, Volume 1, pp. 1004–1009. IEEE.

Jaromczyk, J. W. et G. T. Toussaint (1992). Relative neighborhood graphs and their relatives. *Proceedings of the IEEE 80*(9), 1502–1517.

Krizhevsky, A., G. Hinton, et al. (2009). Learning multiple layers of features from tiny images.

Liu, T., F. Bouali, et G. Venturini (2014). Exod : A tool for building and exploring a large graph of open datasets. *Computers & graphics 39*, 117–130.

Ras, G., M. van Gerven, et P. Haselager (2018). Explanation methods in deep learning : Users, values, concerns and challenges. In *Explainable and Interpretable Models in Computer Vision and Machine Learning*, pp. 19–36. Springer.

Rauber, P. E., S. G. Fadel, A. X. Falcao, et A. C. Telea (2016). Visualizing the hidden activity of artificial neural networks. *IEEE transactions on visualization and computer graphics 23*(1), 101–110.

Rayar, F., S. Barrat, F. Bouali, et G. Venturini (2018). A viewable indexing structure for the interactive exploration of dynamic and large image collections. *ACM Transactions on Knowledge Discovery from Data (TKDD) 12*(1), 1–26.

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

Toussaint, G. T. (1980). The relative neighbourhood graph of a finite planar set. *Pattern recognition 12*(4), 261–268.

Wongsuphasawat, K., D. Smilkov, J. Wexler, J. Wilson, D. Mane, D. Fritz, D. Krishnan, F. B. Viégas, et M. Wattenberg (2017). Visualizing dataflow graphs of deep learning models in tensorflow. *IEEE transactions on visualization and computer graphics 24*(1), 1–12.

Summary

In this paper, we present a preliminary approach to visually explore some properties of the internal representation learned by a convolutional neural network. We observe the characteristics extracted by the network just before the classification layer. We build a neighborhood graph from this vector space by connecting the topologically close data. We define typical examples of topological properties to be detected (isolated points, erroneous points, class boundaries). Then we propose a visualization of this graph highlighting this information and offering an overview of the graph (groups of data). This visualization includes a representation of the images in order to allow the user to understand what can cause an error (errors during image acquisition, errors in pre-processing or labeling images, choice of network, result of learning, etc.). We performed our tests on the VGG16 network and on small extracts of classic datasets.

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

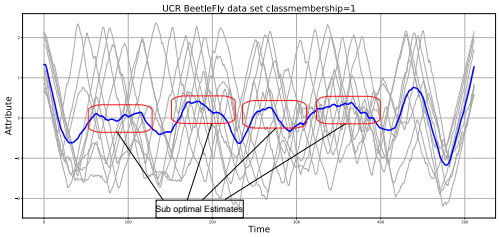


FIG. 1: Impact de la distorsion temporelle sur la moyenne arithmétique.

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).

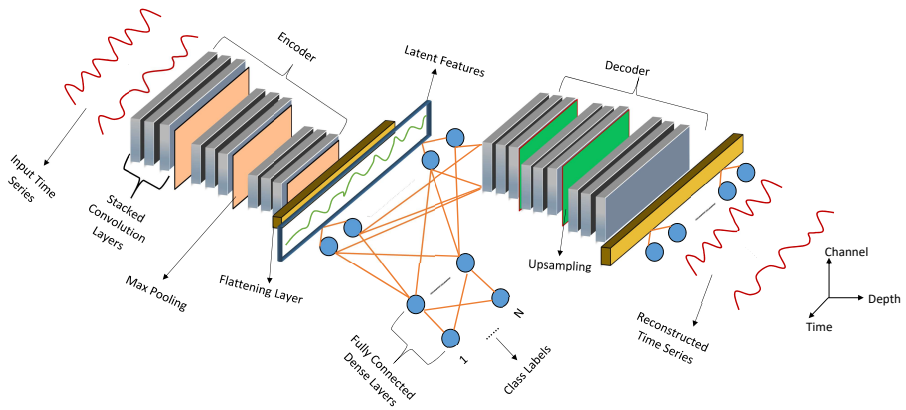


FIG. 2: Architecture de l'auto-encodeur multi-tâches proposé.

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

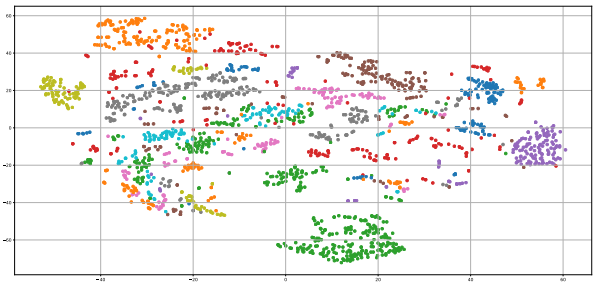
rables à l'état de l'art (DTAN). Les résultats montrent également que l'auto-encodeur simple ne parvient pas à capturer de telles caractéristiques séparables. Cela est démontré par la Figure 3 représentant la projection t-SNE des séries du jeux de données FaceUCR avec 14 classes différentes. La Figure 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).

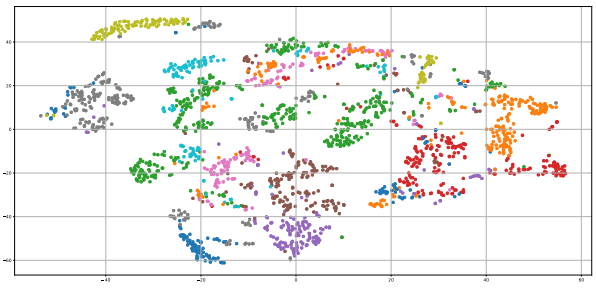
Pour l'estimation de la moyenne avec l'auto-encodeur multi-tâches, nous avons utilisé un réseau entraîné pour projeter les données de l'ensemble d'apprentissage dans l'espace latent. Nous avons ensuite calculé les moyennes comme les moyennes arithmétiques dans cet espace latent. Enfin, le décodeur a 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

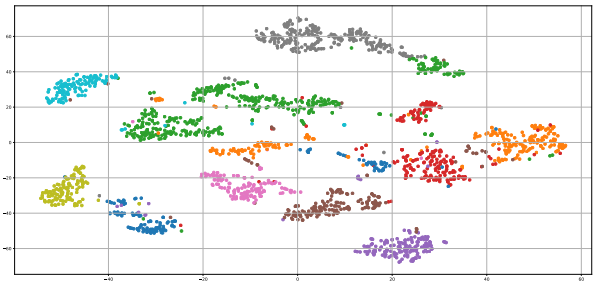
En plus des représentations visuelles et des comparaisons des taux de classification, nous avons effectué un test des rangs signés de Wilcoxon, synthétisé par le diagramme de différence critique en Figure 6. Le diagramme montre que la classification dans l'espace latent (MT.Enc.Lat) en utilisant l'auto-encodeur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

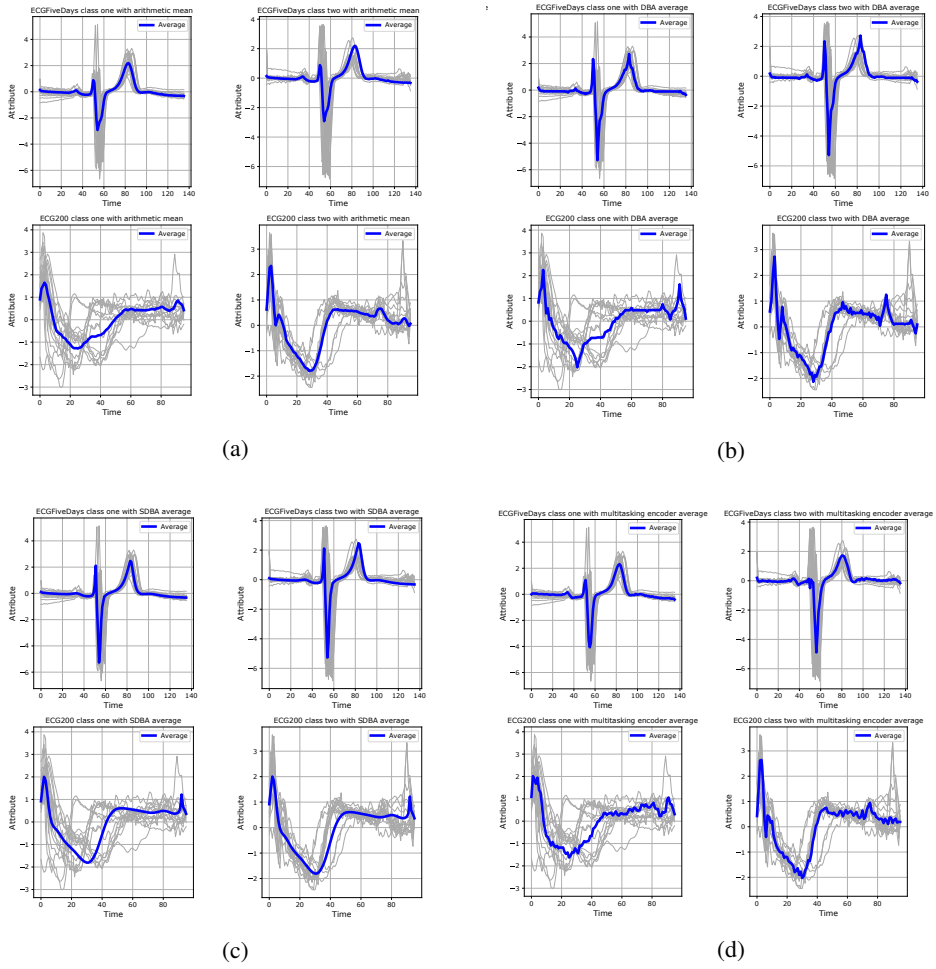
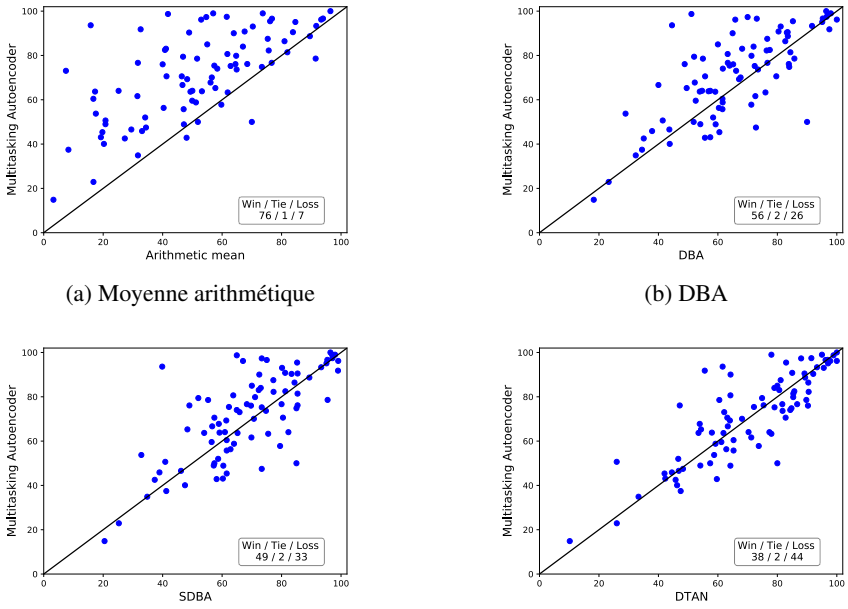


FIG. 4: Comparaison visuelle dans le domaine temporel des moyennes estimées avec la moyenne arithmétique (a), DBA (b), SDBA (c) et notre auto-encodeur multi-tâches (d).

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).

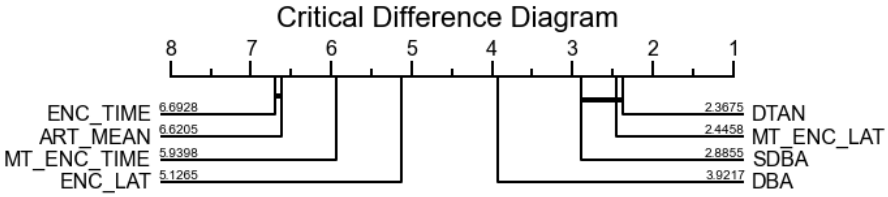


FIG. 6: Comparaison des méthodes par le test des rangs signés de Wilcoxon.

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.

Segmentation non-supervisée d'images histopathologiques à l'aide d'auto-encodeurs et d'ensemble clustering

I. Rmouque*, M. Devanne**, J. Weber**, G. Forestier**, C. Wemmert*

*ICube, Université de Strasbourg, France
wemmert@unistra.fr,
**IRIMAS, Université Haute Alsace, Mulhouse, France
prénom.nom@uha.fr

Résumé. L'apprentissage profond non-supervisé à partir d'auto-encodeurs permet d'obtenir d'excellents résultats en matière d'analyse d'images et de vision par ordinateur. Cependant, seules quelques études ont été présentées dans le domaine de la pathologie numérique, où l'étiquetage précis des objets d'intérêt est une tâche particulièrement coûteuse et difficile. Ainsi, le fait de disposer d'une première segmentation entièrement non supervisée pourrait grandement faciliter le processus d'analyse de ces images.

Dans cet article, de nombreuses architectures d'auto-encodeurs convolutifs ont été comparées afin d'étudier l'influence de trois principaux paramètres : 1) le nombre de couches convolutives, 2) le nombre de convolutions dans chacune de ces couches et 3) la taille de l'espace latent. Différents algorithmes de clustering sont également comparés et une nouvelle approche permettant d'obtenir des résultats plus précis en appliquant des techniques de clustering par ensemble est présentée.

1 Introduction

La pathologie est essentielle pour l'évaluation diagnostique et la compréhension de nombreux mécanismes biologiques et physiologiques sous-jacents. Il s'agit généralement d'une évaluation visuelle, par des pathologistes, d'un échantillon de tissu à l'aide d'un microscope pour en identifier les propriétés structurelles. Actuellement, l'évaluation visuelle des échantillons microscopiques est en grande partie un processus non assisté, et la précision du pathologiste est due à une formation approfondie, une analyse comparative, un contrôle de qualité par les pairs et l'expérience personnelle. Cependant, ce domaine a connu plusieurs révolutions technologiques ces dernières années avec l'avènement de la microscopie virtuelle (conversion de lames microscopique en images à haute résolution), souvent appelée "pathologie numérique". Ainsi, des efforts importants ont été faits pour concevoir des outils d'analyse d'images, par exemple pour identifier les structures biologiques de base (stroma, cellules immunitaires, tumeur, etc.), afin de faciliter l'interprétation (semi-)automatisée des lames par les médecins.

Entre-temps, les algorithmes d'analyse automatique d'images ont à nouveau fait des progrès extraordinaires et ce notamment, grâce aux méthodes d'apprentissage profond introduites

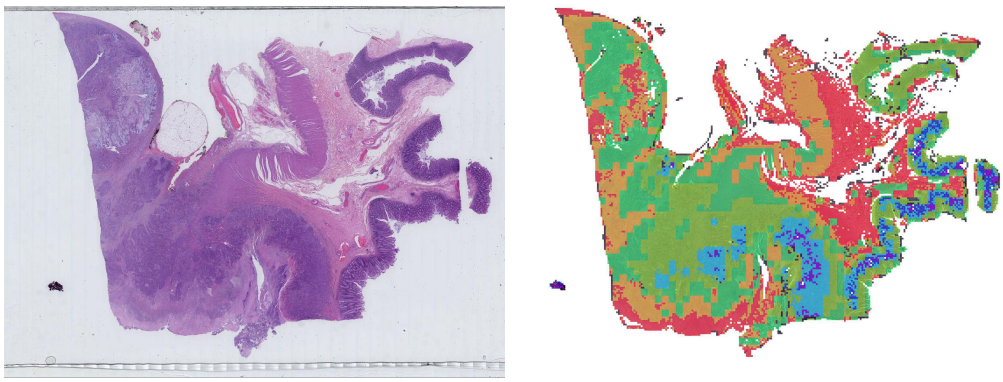
par LeCun et al. (2015). En effet, les performances de ces méthodes ont explosé ces dernières années permettant la détection, la classification et la segmentation d'objets d'intérêt dans les images avec une très grande précision. Mais la plupart de ces approches fonctionnent en mode supervisé, c'est-à-dire qu'elles nécessitent de nombreux exemples pour fournir un modèle efficace. En même temps, les approches non supervisées ont montré leur intérêt pour de nombreuses applications de l'analyse d'images, comme la télédétection (Liang et al., 2018; Mei et al., 2019). Récemment, elles ont également été appliquées à l'analyse d'images histopathologiques sur lames entières scannées (*Whole-Slide Images* - WSI). En particulier, dans Yamamoto et al. (2019), les auteurs décrivent une méthode d'extraction non supervisée d'informations intéressantes à partir d'images complètes de diapositives, qui se révèle plus précise que l'analyse humaine pour le pronostic de la récurrence du cancer de la prostate.

Nous nous intéressons ici à la segmentation automatique afin d'extraire rapidement les régions d'intérêt (les tumeurs par exemple) et d'effectuer une analyse plus précise pour ces seules régions (les WSI sont des images de très grande taille et l'analyse ne pourrait être effectuée que sur certaines régions d'intérêt).

Cependant, seuls quelques travaux sur la segmentation entièrement non supervisée des WSI ont été présentés. La première tentative de segmentation des régions d'intérêt des WSI sans aucune information ou exemple préalable a été réalisée dans Khan et al. (2013). Les auteurs ont étudié la morphologie des tissus dans les images histologiques du cancer du sein en calculant un ensemble de filtres de Gabor pour distinguer différentes régions. Dans Fouad et al. (2017), les auteurs utilisent la morphologie mathématique pour extraire des "cellules virtuelles" (par exemple des superpixels), pour lesquelles des caractéristiques morphologiques et de couleur sont calculées pour ensuite appliquer un algorithme d'ensemble clustering pour identifier les différents types de tissu dans l'image. Plus récemment, une approche similaire a été présentée par Landini et al. (2019), ajoutant aux techniques précédentes un classifieur auto-apprenant semi-supervisé qui améliore les résultats au prix de l'abandon de la non-supervision.

Toutes ces approches proposent d'effectuer un clustering sur les image en fonction de caractéristiques prédéfinies. Cependant, des approches d'apprentissage profond, notamment via des architectures d'auto-encodage, permettent d'éviter de définir manuellement les caractéristiques en calculant la représentation condensée de l'image dans un espace latent par l'application de filtres convolutifs. Malheureusement, comme indiqué dans Raza et Singh (2018), la plupart des applications des auto-encodeurs en pathologie numérique ont été développées pour effectuer la segmentation cellulaire ou la détection de noyaux (Xu et al., 2015; Hou et al., 2019), ou la normalisation des taches (Janowczyk et al., 2017).

Nous proposons donc ici d'étudier le potentiel de ces approches pour la segmentation des WSI. Dans cet article, nous présentons une étude sur les performances des auto-encodeurs convolutifs sur la segmentation des WSI en comparant différentes approches. Dans un premier temps, différentes architectures d'auto-encodeurs sont comparées afin de quantifier l'importance des paramètres d'intérêt (nombre de couches, taille de l'espace latent, nombre de convolutions par couche), puis, constatant que les méthodes de clustering conventionnelles donnent un résultat partiel, nous proposons une méthode multi-résolution utilisant des approches d'ensemble clustering.



(a) Exemple de WSI (b) Exemple de clustering à 8 clusters

FIG. 1 – *Un exemple de WSI et d'un clustering possible*

2 Jeu de données

Notre étude a été réalisée sur 8 WSI de tissus colorés à l'hématoxyline et à l'éosine (H&E) extraits d'une cohorte de patients construite dans le cadre du projet AiCOLO (INSERM/Plan Cancer) étudiant le cancer du côlon. Les images ont été fournies par le Centre Georges François Leclerc de Dijon (France). La coloration H&E permet de distinguer les noyaux des cellules en violet, de la matrice extracellulaire et du cytoplasme en rose.

Toutes les images ont été acquises à un grossissement de 40× ce qui correspond environ à 0.22μm/pixel mais elles sont stockées à plusieurs résolutions dans un format pyramidal. Les différentes résolutions sont désignées comme niveau de détail (LOD - *Level Of Details*), LOD0 correspondant à un grossissement de 40×, LOD1 à 20×, etc. La taille de chaque image est d'environ 90000×50000 pixels.

Pour entraîner l'auto-encodeur, 10000 zones de 128×128 pixels ont été extraites aléatoirement au LOD1 de toutes les images, car cela semble être la quantité minimale d'informations requise par un expert humain pour classer le tissu. Entre-temps, des pathologistes ont créé sur les images des annotations manuelles éparses pour les cinq classes de tissus, tumeur, stroma, muqueuse de la couche externe (cryptes de Lieberkuhn et tissu conjonctif), cellules immunitaires et nécrose, et deux classes pour le fond et les artefacts (marques de feutre, etc.), afin de pouvoir évaluer la pertinence du regroupement.

3 Auto-encodeurs convolutifs pour le clustering de WSI

Dans cette section, nous étudions le potentiel de l'utilisation d'auto-encodeurs convolutifs pour le clustering des images histopathologiques. Comme le montre la Figure 2, un *auto-encodeur convolutif* (AEC) est un réseau de neurones convolutif profond composé de deux parties : un encodeur et un décodeur. L'objectif principal de l'AEC est de minimiser une fonction de perte, c'est-à-dire une fonction évaluant la différence entre l'entrée et la sortie de l'AEC (généralement l'erreur quadratique moyenne). Une fois cette fonction minimisée, on peut sup-

poser que la partie encodeur établit un résumé approprié des données d'entrée dans l'espace latent, car la partie décodeur est capable d'en reconstruire une copie fortement similaire à partir de cette représentation encodée. L'encodeur est constitué d'une couche d'entrée (ayant la taille de l'image en entrée) qui est connectée à N couches de convolution de taille décroissante, jusqu'à un goulot d'information de taille Z, appelé espace latent. L'espace latent est connecté à une série de couches de N convolutions de taille croissante, jusqu'à atteindre la taille de l'image d'entrée. Cette seconde partie est appelée le décodeur. Chaque couche de convolution est composée de C convolutions et est suivie de trois autres couches : une normalisation, une fonction d'activation (ReLU) et un max pooling de taille (2,2).

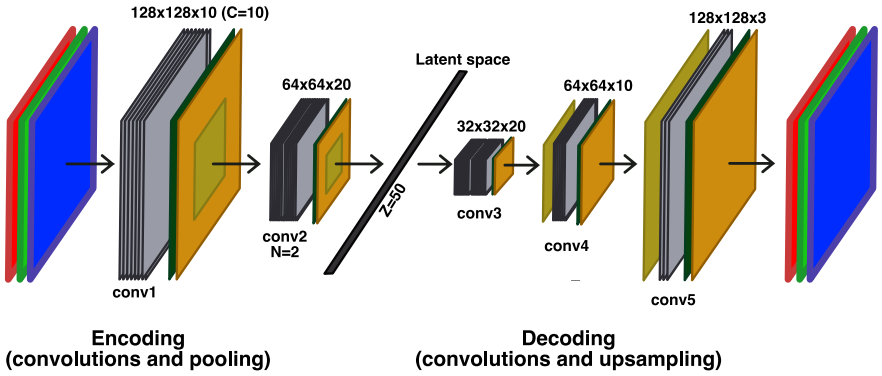


FIG. 2 – *Architecture de l'AEC pour* $N = 2$, $C = 10$ *et* $Z = 50$

Pour effectuer un clustering, un AEC entraîné est utilisé pour encoder chaque partie de l'image entière. Ensuite, la représentation encodée dans l'espace latent est donnée en entrée d'un algorithme de clustering qui lui attribue un cluster. Un exemple de clustering d'une WSI est montré sur la Figure 1(b). Nous avons décidé d'évaluer l'influence des trois principaux paramètres N, Z et C. Pour chacun d'eux, des valeurs différentes ont été testées tout en gelant les deux autres. Afin d'évaluer la qualité des résultats, l'indice de Rand ajusté (ARI) est calculé comme la normalisation de l'indice de Rand (RI) défini dans Eq.1.

$$RI = \frac{TP + TN}{TP + FP + TN + FN} \tag{1}$$

où TP, TN, FP et FN représentent respectivement les pixels vrais positifs, vrais négatifs, faux positifs et faux négatifs.

L'un des principaux défis du clustering non supervisé est de trouver la correspondance entre un cluster et une classe identifiée par l'expert. Afin de simplifier le problème, nous avons décider d'évaluer la capacité du clustering à détecter la classe tumeur uniquement (classe d'intérêt dans notre projet). Ainsi, le problème peut être exprimé comme un problème à deux classes : tumeur vs. non tumeur. Pour associer la classe de tumeur à un cluster, nous avons calculé sa densité (nombre de pixels de la tumeur / nombre total de pixels du cluster). Tous les clusters ayant une densité supérieure à 50% sont considérés comme "tumeur", les autres sont étiquetés comme "non tumeur". Chaque AEC a été entraîné sur un ensemble de 10000 extraits de WSI

sélectionnées aléatoirement. Comme les résultats du clustering et de l'entraînement des AEC sont non déterministes, en raison d'une sensibilité élevée aux conditions initiales, 10 auto-encodeurs ont été entraînés, et ont été conservés les résultats moyens pour chaque valeur de chaque paramètre. Nous avons également étudié les performances de plusieurs algorithmes de clustering, Kmeans, Agglomerative clustering (AggCl), Gaussian mixture (GM) et ainsi que la méthode "not too deep clustering" (N2D) de McConville et al. (2020).

Résultats

Comme le montre la Figure 3, il semble que le nombre de convolutions dans chaque couche de convolutions (paramètre C) n'affecte pas beaucoup la qualité de l'auto-encodeur, à part une légère baisse de la variabilité des résultats. Ceci s'explique par le fait que passé un certain seuil, un nombre supplémentaire de convolutions apporte peu d'informations.

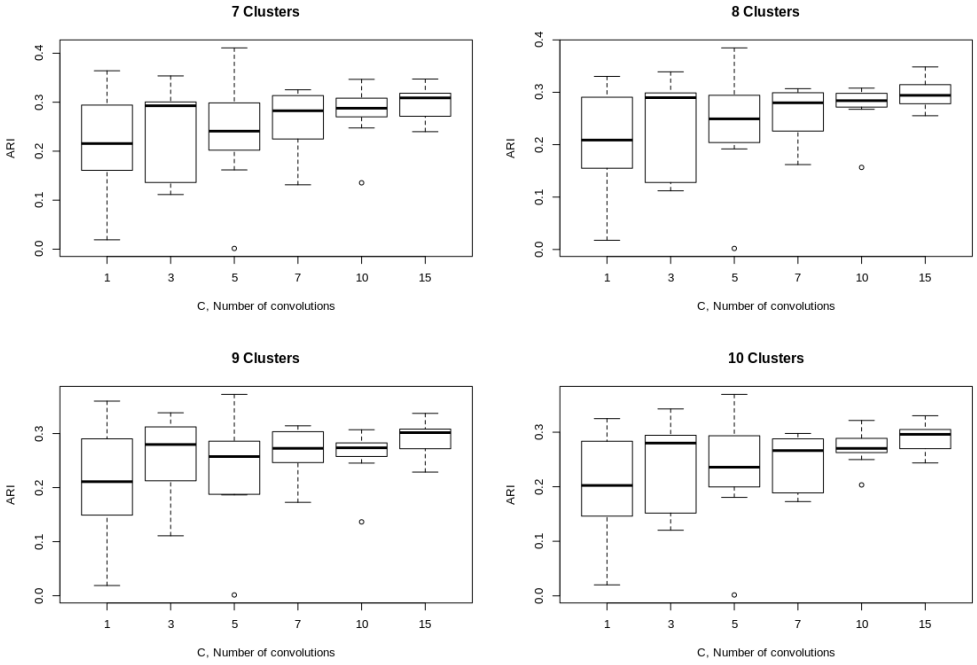


FIG. 3 – *Évaluation de l'ARI en fonction du nombre de convolutions C dans chaque couche de convolutions avec l'algorithme de clustering Kmeans pour 7, 8, 9 et 10 clusters.*

La Figure 4 montre l'évaluation de l'ARI avec un nombre différent de couches de convolutions dans l'architecture. On remarque une augmentation de l'indice de qualité jusqu'à 4 couches de convolution, puis une chute brutale pour 5, ce qui indique clairement qu'un trop grand nombre de convolutions (et les poolings qui réduisent la taille de l'information) affectent la qualité de l'information qui ne peut plus être traitée correctement.

Néanmoins, comme le montre la Figure 5, la taille de l'espace latent Z, semble avoir une grande influence sur la pertinence du AEC. En effet, l'ARI augmente nettement au fur et

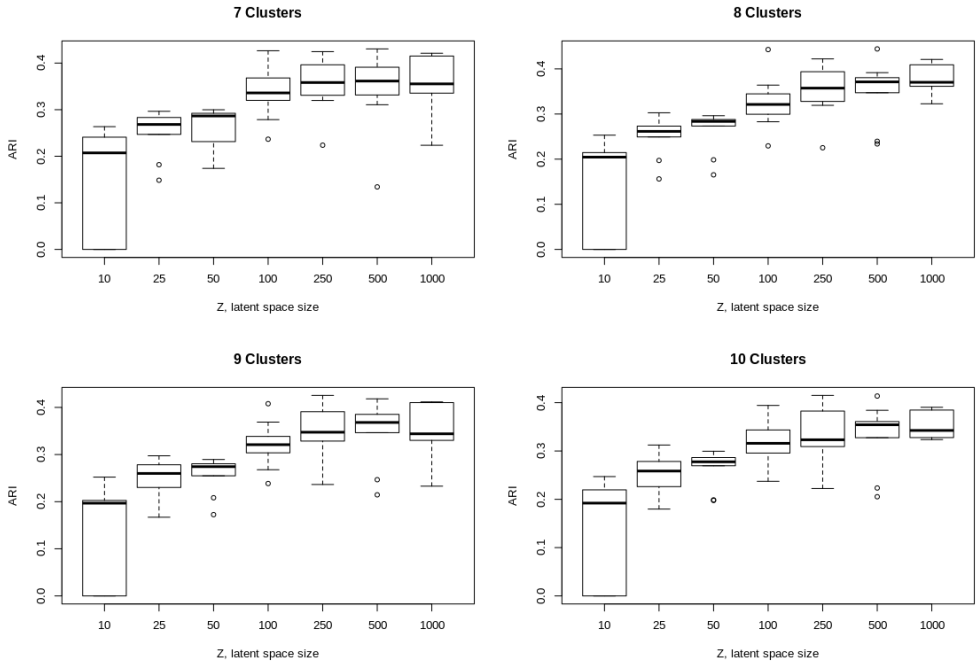


FIG. 4 – *Évaluation de l'ARI en fonction de la taille de l'espace latent avec l'algorithme de clustering Kmeans pour 7, 8, 9 et 10 clusters.*

à mesure que la taille de la représentation latente augmente. En effet, plus la représentation latente contient d'information, plus il est facile de différencier les classes, mais il est également évident qu'un espace latent trop grand ne pourra pas résumer efficacement les informations et n'aidera donc pas l'algorithme de clustering à distinguer les différents tissus. En outre, plus l'espace latent est grand, plus il faut de mémoire et de temps pour entraîner le réseau.

4 Ensemble clustering

Comme indiqué dans Yamamoto et al. (2019), les micro-structures et les macro-structures des objets anatomiques présents dans les WSI fournissent des informations différentes et complémentaires. Les pathologistes s'accordent également sur le fait qu'il est beaucoup plus difficile d'identifier une cellule sans son contexte environnant et ils regardent toujours la WSI à un grossissement plus faible (pour mieux saisir le contexte) avant de zoomer à un grossissement plus élevé.

Nous avons donc essayé d'améliorer nos résultats en utilisant un ensemble de méthodes de clustering, chacune se concentrant sur un niveau de détail différent. L'objectif est de fusionner les informations de bas niveau (contexte) avec les informations de haut niveau (forme des cellules, etc.). Pour ce faire, nous utilisons la méthode proposée dans Wemmert et Gançarski

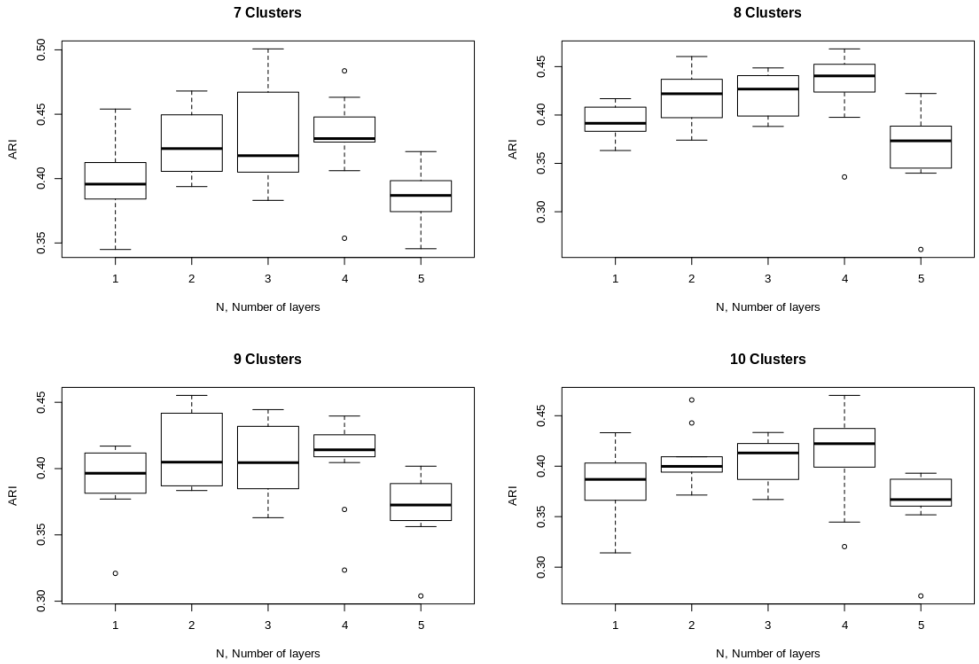


FIG. 5 – *Évaluation de l'ARI en fonction du nombre de couches de convolutions N avec l'algorithme Kmeans pour 7, 8, 9 et 10 clusters.*

(2002). De nombreuses configurations différentes ont été testées pour cette expérience, mais nous n'en présentons ici que deux, représentatives de l'amélioration de la qualité du résultat qui peut être obtenue en utilisant l'ensemble clustering.

La première, E_{multires}, est composée de trois algorithmes de clustering (Kmeans), utilisant la représentation dans l'espace latent construit par différents AEC entraînés à différentes résolutions : LOD2 avec 8 clusters, LOD3 avec 6 clusters et LOD3 avec 8 clusters.

Comme l'image reconstruite à partir de l'auto-encodeur semble se concentrer davantage sur l'intensité des couleurs que réellement sur les structures, une seconde configuration a été testée. Afin d'ajouter de la diversité et de forcer le résultat final à se concentrer davantage sur la structure des objets, un clustering travaillant sur une image binaire a été ajouté à l'ensemble. Cette image est construite en seuillant l'intensité de l'image initiale. La seconde configuration, E_{struct}, est donc composée de trois algorithmes de clustering (Kmeans) avec les paramètres suivants : LOD3 sur l'image binaire avec 6 clusters, LOD3 sur l'image binaire avec 8 clusters et LOD2 sur l'image RGB initiale avec 6 clusters.

Deux critères d'évaluation ont été calculés à partir des résultats et sont présentés dans le Tableau 1. L'ARI (voir Eq. 1) est complété par le $Fscore$ défini à partir de la précision et du rappel comme :

$$Fscore = \frac{TP}{TP + 1/2(FP + FN)} \tag{2}$$

	Données brutes		**Données encodées**											
	Kmeans		**Kmeans**		**AggCl**		**GM**		**N2D**		E_{multires}		E_{struct}	
	ARI	$FScore$	ARI	$FScore$	ARI	$FScore$	ARI	$FScore$	ARI	$FScore$	ARI	$FScore$	ARI	$FScore$
Image 1	0.39	0.89	**0.48**	0.96	0.38	0.94	0.27	0.73	0.43	**0.97**	0.47	0.96	0.42	0.88
Image 2	0.27	0.68	0.33	0.63	0.29	0.62	0.19	0.43	0.29	**0.68**	0.31	0.66	**0.46**	0.62
Image 3	0.25	0.76	0.39	0.87	0.35	0.85	0.22	0.76	0.31	0.88	0.37	0.87	**0.45**	**0.91**
Image 4	0.08	0.48	0.08	0.50	0.13	0.61	0.05	0.51	0.12	0.55	**0.08**	0.54	0.0	0.48
Image 5	0.11	0.65	0.11	0.64	0.10	0.60	0.10	0.62	0.11	0.65	0.12	0.65	**0.17**	**0.72**
Image 6	0.37	0.68	0.52	0.75	0.51	0.72	0.49	0.67	0.43	0.76	0.51	**0.77**	**0.57**	0.75
Image 7	0.28	0.68	0.35	0.73	0.33	0.75	0.14	0.61	0.37	0.74	**0.41**	0.76	0.36	**0.84**
Image 8	0.33	0.63	0.44	0.71	0.42	0.70	0.07	0.44	0.37	0.69	0.44	0.75	**0.45**	**0.75**

TAB. 1 – *Évaluations des résultats de clustering obtenus à l'aide des différentes méthodes*

Résultats

L'ensemble des résultats pour chacune des configurations est donné dans le Tableau 1. On y retrouve le résultat pour chacune des méthodes de clustering testées (simples et ensembles), ainsi que le résultat pour un clustering effectué avec l'algorithme Kmeans sur les données brutes (sans aucun traitement de l'AEC), qui a été calculé comme base de référence afin d'évaluer la pertinence de l'encodage des données avec un AEC.

Tout d'abord, on constate que le clustering appliqué directement sur les données brutes a obtenu des résultats de moins bonne qualité que toutes les autres configurations, ce qui confirme que l'encodage des données a un effet positif sur le clustering. Ensuite, les méthodes classiques appliquées sur la représentation de l'espace latent de l'AEC tendent à montrer des résultats acceptables.

Cependant, les deux configurations d'ensemble clustering montrent une efficacité plus grande pour trouver des clusters cohérents correspondant aux classes d'intérêt définies par les pathologistes. En effet, parmi toutes les méthodes proposées, E_{struct} semble donner les meilleurs résultats. Elle tend à confirmer l'importance, à la fois, d'introduire plus de variabilité dans l'ensemble pour améliorer les résultats, mais aussi de l'utilisation d'informations sur la structure des objets. En outre, cela montre que même si les auto-codeurs convolutifs visent à trouver automatiquement les meilleures caractéristiques pour coder l'image, ils peuvent également tirer parti de caractéristiques introduites manuellement par l'expert pour certaines tâches spécifiques.

5 Conclusion

Dans cet article, nous avons comparé différentes configurations d'auto-encodeurs convolutifs dans le domaine de l'apprentissage non supervisé pour la segmentation des images histopathologiques WSI. Pour cela, différentes architectures d'AEC ont été comparées pour essayer de trouver la meilleure configuration et d'étudier l'influence de chaque paramètre. Ensuite, nous avons proposé une nouvelle approche qui utilise la technique d'ensemble clustering pour tirer profit des informations multirésolution et des caractéristiques structurelles de l'image. Cela confirme l'importance de la diversité dans un cadre d'apprentissage profond et que le fait

de travailler à différentes résolutions en même temps peut réellement améliorer la qualité des résultats.

Remerciements

Ce travail est issu du projet AiCOLO financé par l'appel *Approches interdisciplinaires des processus oncogéniques et perspectives thérapeutiques : apports à l'oncologie des mathématiques et de l'informatique* Inserm / Plan Cancer (2014-2019).

Références

Fouad, S., D. Randell, A. Galton, H. Mehanna, et G. Landini (2017). Unsupervised morphological segmentation of tissue compartments in histopathological images. *PloS one 12*(11), e0188717.

Hou, L., V. Nguyen, A. B. Kanevsky, D. Samaras, T. M. Kurc, T. Zhao, R. R. Gupta, Y. Gao, W. Chen, D. Foran, et al. (2019). Sparse autoencoder for unsupervised nucleus detection and representation in histopathology images. *Pattern recognition 86*, 188–200.

Janowczyk, A., A. Basavanhally, et A. Madabhushi (2017). Stain normalization using sparse autoencoders (stanosa) : application to digital pathology. *Computerized Medical Imaging and Graphics 57*, 50–61.

Khan, A. M., H. El-Daly, E. Simmons, et N. M. Rajpoot (2013). Hymap : A hybrid magnitude-phase approach to unsupervised segmentation of tumor areas in breast cancer histology images. *Journal of Pathology Informatics 4*.

Landini, G., S. Fouad, D. Randell, et H. Mehanna (2019). Epithelium and stroma segmentation using multiscale superpixel clustering. *Journal of Pathology Informatics 10*, S45–S48.

LeCun, Y., Y. Bengio, et G. E. Hinton (2015). Deep learning. *Nature 521*(7553), 436–444.

Liang, P., W. Shi, et X. Zhang (2018). Remote sensing image classification based on stacked denoising autoencoder. *Remote Sensing 10*(1), 16.

McConville, R., R. Santos-Rodriguez, R. J. Piechocki, et I. Craddock (2020). N2d : (not too) deep clustering via clustering the local manifold of an autoencoded embedding.

Mei, S., J. Ji, Y. Geng, Z. Zhang, X. Li, et Q. Du (2019). Unsupervised spatial–spectral feature learning by 3d convolutional autoencoder for hyperspectral classification. *IEEE Transactions on Geoscience and Remote Sensing 57*(9), 6808–6820.

Raza, K. et N. K. Singh (2018). A tour of unsupervised deep learning for medical image analysis. *arXiv preprint arXiv :1812.07715*.

Wemmert, C. et P. Gançarski (2002). A multi-view voting method to combine unsupervised classifications. In *Artificial Intelligence and Applications*, pp. 447–452.

Xu, J., L. Xiang, Q. Liu, H. Gilmore, J. Wu, J. Tang, et A. Madabhushi (2015). Stacked sparse autoencoder (ssae) for nuclei detection on breast cancer histopathology images. *IEEE transactions on medical imaging 35*(1), 119–130.

Yamamoto, Y., T. Tsuzuki, J. Akatsuka, M. Ueki, H. Morikawa, Y. Numata, T. Takahara, T. Tsuyuki, K. Tsutsumi, R. Nakazawa, et al. (2019). Automated acquisition of explainable knowledge from unannotated histopathology images. *Nature communications 10*(1), 1–9.

Summary

Unsupervised deep learning with autoencoders have shown excellent results in image analysis and computer vision. However, only few studies have been presented in the field of digital pathology, where proper labelling of the objects of interest is a particularly costly and difficult task. Thus, having a first fully unsupervised segmentation could greatly help in the analysis process of such images. In this paper, many architectures of convolutional autoencoders have been compared to study the influence of three main hypermarameters: number of convolutional layers, number of convolutions in each layer and size of the latent space. Different clustering algorithms are also compared and we propose a new way to obtain more precise results by applying ensemble clustering techniques.

Liste des auteurs

Blin Guillaume, 15–24
Bouali Fatma, 25–34

Colin Maïwenn, 25–34

Dejean Philippe, 2–14
Deléarde Robin, 2–14
Devanne Maxime, 35–59

Forestier Germain, 35–59

Guinot Christiane, 25–34

Hailemariam Dereje, 35–49

Kurtz Camille, 2–14

Lallier Corentin, 15–24
Le Meur Florian, 25–34

Palyart Marc, 15–24
Pinaud Bruno, 15–24

Rmouque Ilias, 50–59

Serres Barthélémy, 25–34

Terefe Tsegamlak, 35–49

Venturini Gilles, 25–34
Vézard Laurent, 15–24

Weber Jonathan, 35–59
Wemmert Cédric, 50–59
Wendling Laurent, 2–14