

Discussion

Nous remarquons ici, une amélioration de l'accuracy pour deux jeux de données traitant des problèmes similaires: la reconnaissance d'action squelettiques pour des niveaux différents (la main et le corps), pour des dimensions différentes (2d et 3d) et pour des dimensions de convolutions variantes.

L'apport de la normalisation semble plus marqué sur SHREC que sur JHMDB. Une explication serait que le niveau d'abstraction et de précision n'est pas le même entre ces jeux de données. Le nombre de keypoints dans SHREC est supérieur à JHMDB et par conséquent l'organisation de ceux-ci pourrait apporter plus d'information.

En choisissant un modèle complexe pour la convolution 1d et un modèle trivial pour la convolution 2d, nous émettons l'hypothèse que cette transformation peut-être bénéfique quelque soit le niveau de complexité du modèle. Cependant, celle-ci modifie significativement la taille de l'entrée du réseau et par conséquent la taille en paramètres des approches augmentera. Cette augmentation n'est que peu visible sur DD-Net car il n'y a pas de layer dense. Pour Lenet-5 la taille du modèle augmente du simple au double, et ce à cause de la partie fully connected de l'architecture. Il conviendra donc d'éviter le plus possible l'utilisation de layer dense pour cette approche ou de travailler sur des réseaux de taille réduite.

Finalement, cette normalisation nécessite une connaissance à priori de l'organisation de la structure des données en entrée (coude: keypoint 1, tête: keypoint 2, ...). Dans le cadre de la thèse, cela ne pose pas de contraintes spécifiques. Cependant, il sera par exemple impossible de réaliser ce travail sur des jeux de données dont la structure des données n'est pas précisée.

3.2 Autoencodeur semi-supervisé pour l'action recognition

3.2.1 Introduction

Les auto encodeurs étant une composition de transformation non linéaires en esperant trouver une représentation dans l'espace adaptée au format des données et conservant un maximum de sémantique de celui-ci.

Je me suis intéressé à cette question de représentation pour plusieurs raisons:

- Tout est une question d'embedding et de normalisation en machine learning. Le rôle premier des couches cachées est de transformer cet embedding en espérant qu'il soit porteur d'information. Le jour où l'on trouve un moyen de normaliser et de représenter les données de manière plus adéquate, n'importe quel classifieur pourra obtenir

de bons résultats pour peu que les données en entrée soient porteuses d'information. Le jour où l'on arrivera à obtenir une représentation utilisable plus rapidement au lieu de stacker des layers au sein d'un réseau de neurones, d'autres problématiques d'apprentissage apparaîtront mais le gain serait intéressant: (Selon le théorème d'approximation universelle, n'importe quel Shallow Network peut supposément approximer une fonction donnée [Cybenko, 1989, Scarselli and Tsoi, 1998]).

- En s'intéressant à la question de la représentation des données, on évite un apprentissage de cette représentation à "l'aveugle": en optimisant un réseau précis sur un jeu de données précis et en testant un maximum d'hyper-paramètres on est assurés de faire un bon résultat. Plus la représentation des données est fiable, plus l'architecture utilisée pour la classification peut-être triviale. On peut donc potentiellement réduire la taille de notre réseau et donc par définition réduire le temps d'inférence de celui-ci, ce qui peut être intéressant dans le cadre d'un sujet en temps réel.
- Peu de travaux de recherche en reconnaissance d'actions se focalisent sur la représentation des données, la majorité des travaux actuels translatent la connaissance de domaines connexes tels que la computer vision ou le text-mining et s'inspirent des méthodes de l'état de l'art dans ces domaines. Il n'existe à ce jour à ma connaissance aucune publication sur l'utilisation des auto-encodeurs pour la reconnaissance d'actions squelettiques.

L'idée étant d'entraîner un auto-encodeur avec une fonction de coût modifiée en rajoutant une contrainte sur la séparation linéaire des classes dans l'espace latent (LDA / QDA / FDA). Il a été montré que la représentation optimale interne d'un MLP s'obtient en faisant une analogie avec une analyse discriminante non linéaire [Webb and Lowe, 1990]. Parallèlement, [Brigato and Iocchi, 2020] montrent que l'ajout d'une fonction de coût non-supervisée lors d'un problème de classification avec peu de données permet une meilleure généralisation des réseaux.

Les approches factorielles citées plus haut s'apparentent à la projection de données dans l'espace comme celle d'une ACP mais tandis que l'ACP maximise la variance du jeu de données, celles-ci se focalisent sur la maximisation de la séparabilité des classes dans l'espace, ce qui en fait des méthodes supervisées. On obtiendrait donc un Auto-encodeur "semi-supervisé" dans le sens où celui-ci se focalise sur deux informations complètement différentes dans les données, l'une de manière non-supervisée, l'autre de manière supervisée:

- La structure inhérente des données capturée de manière non supervisée grâce à la reconstruction de l'Auto-encodeur et sa capacité d'abstraction. On conserverait

alors une partie de l'information importante et discriminante du jeu de données.
(feature extraction)

- La séparabilité des classes dans l'espace grâce à l'analyse discriminante linéaire. Ce qui permettrait de réduire le nombre de layers.

Nous pouvons imaginer deux utilisations à cette approche:

- Récupérer cet espace latent combinant les informations "data driven" grâce à l'apprentissage non-supervisé de l'auto encodeur classique et la première ébauche de séparabilité linéaire grâce à la seconde partie de la fonction de coût. Envoyer l'espace latent à un réseau de neurones très peu profond pour la classification. Finetuner le réseau de bout en bout et voir ce que cela apporte quand à la capacité de discrimination de l'approche.
- Générer des données en réalisant une génération d'instances au niveau de l'espace latent grâce à des approches type SMOTE [Chawla et al., 2002] ou ADASYN [He et al., 2008] et utiliser la partie décodeur pour faire de la data génération.

3.2.2 Formalisation

La pipeline complète de notre approche est présentée en figure 3.6. Nous définissons le problème comme ceci:

$$\min_{\theta_1, \theta_2} \| \mathbf{X} - g_{\theta_2}(f_{\theta_1}(\mathbf{X})) \|^2$$

Fonction de reconstruction habituelle d'un auto-encodeur avec θ_1, θ_2 les paramètres des blocs encodeur et décodeur de l'AE.

$$\min_{\theta_1, \theta_2, \mathbf{S}} \| \mathbf{X} - g_{\theta_2}(f_{\theta_1}(\mathbf{X})) \|^2 + \lambda \| f_{\theta_1}(\mathbf{X}) - \mathbf{S} f_{\theta_1}(\mathbf{X}) \|^2$$

Ajout d'une variation dans la fonction de coût: avec \mathbf{S} la matrice de projection des individus dans l'espace latent obtenue avec une analyse discriminante linéaire (LDA) et λ un régularisateur.

Une fois, l'entraînement de l'auto-encodeur réalisé, récupération des poids de la partie encodeur: f_{θ_1} , ajout d'une fonction classifieur softmax:

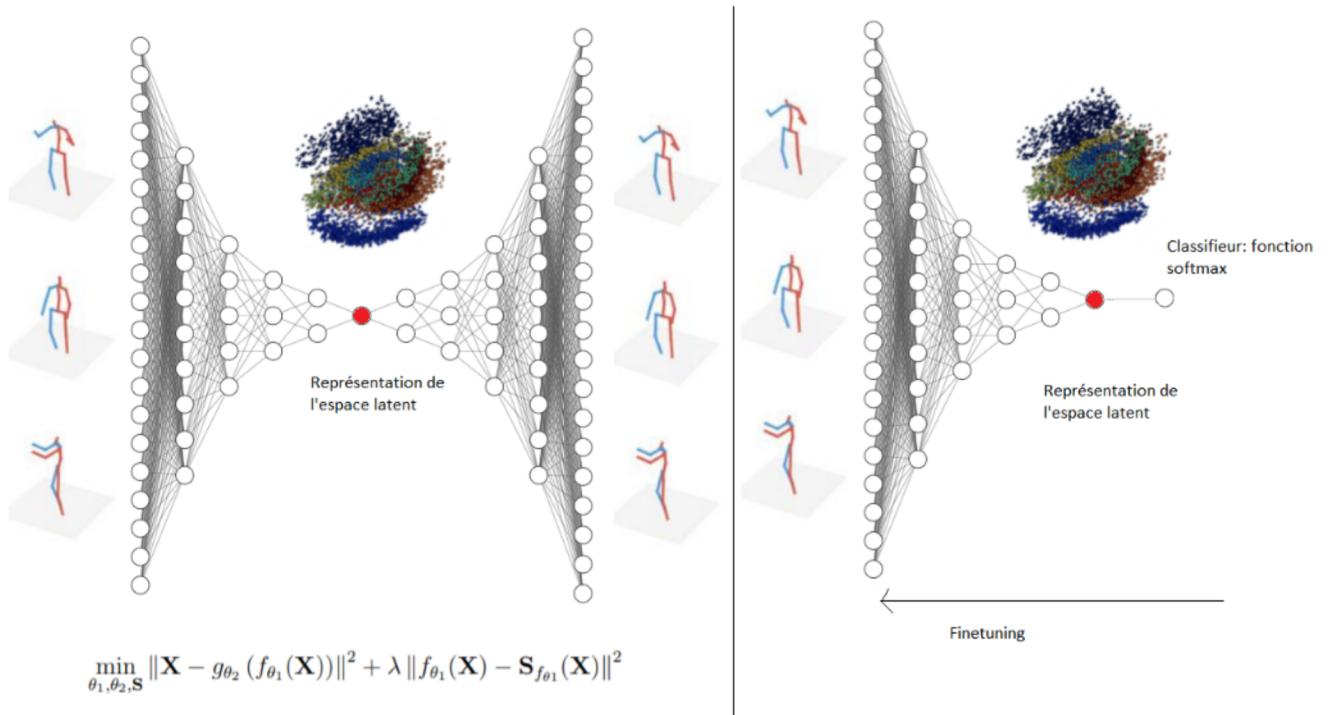


Figure 3.6: Pipeline de l’approche: (1) nous entraînons un auto-encodeur pour reconstruire une séquence représentant une action en y ajoutant une contrainte spécifique à la séparabilité des classes dans l’espace latent. (2) extraction des poids de la partie encodeur jusqu’au bottleneck représenté en rouge et ajout d’une softmax, transformant alors la partie encodeur en un réseau pré-entraîné sur les données pour la classification des actions.

Nous souhaitons dans un premier temps évaluer l’apport de cette représentation pour la classification.

3.2.3 Tests et évaluation

Evaluation datasets

Par soucis d’économie de temps, j’ai réalisé l’évaluation de cette approche sur les mêmes jeux de données que ceux présentés en 3.1.3, à savoir SHREC [De Smedt et al., 2017] et JHMDB [Jhuang et al., 2013].

Détails d’implémentation

Afin d’évaluer la qualité de représentation, nous avons choisi de prendre une architecture d’autoencodeur la plus facile possible: semblable à celle d’un perceptron multicouches. Naturellement, il est possible de réaliser le même travail sur des auto-encodeurs convolutifs ou séquentiels afin de capturer la séquentialité de l’action ou de minimiser la taille du