Reconnaissance de gestes et d'actions de la main, combinant vision par ordinateur et technologies de réalité augmentée

Theo Voillemin

Année 2021

Université de Lille

Ecole Doctorale SPI-MADIS

Laboratoire CRIStAL (umr cnrs 9189)

THÈSE

pour obtenir le grade de

Docteur,

spécialité Informatique

soutenu par

Théo Voillemin

le 29/10/2021

Reconnaissance de gestes et d'actions de la main, combinant vision par ordinateur et technologies de réalité augmentée

COMPOSITION DU JURY

Mme. Catherine Achard	Professeure, Sorbonne Université	Rapporteur
M. Hedi Tabia	Professeur, Paris Saclay	Rapporteur
M. Fabien Moutarde	Professeur, Mines ParisTech	Président du jury
Mme. Sylvie Gibet	Professeure, Université Bretagne Sud	Examinateur
M. Jean-Philippe Vandeborre	Professeur, IMT Nord Europe	Co-directeur
M. Hazem Wannous	Professeur, IMT Nord Europe	Co-directeur

IMT Nord Europe
École Mines-Télécom
IMT-Université de Lille

CRIStAL
Centre de Recherche en Informatique, Signal et Automatique de Lille

Remerciements

Même si ces trois années de thèse ont été principalement une période de travail personnelle et solitaire, il est tout de même important de préciser qu'il aurait été difficile de les surmonter sans le soutien de mon entourage.

Je remercie bien évidemment ma famille, mes parents, Laurence et Philippe ainsi que ma soeur Philippine. Ils sont accompagnés de mes amis proches, Corentin, Solen, Vincent, Pauline, Grégoire, Thomas, Anaëlle et Nicolas. Ces personnes ont su m'écouter, me supporter et me remotiver dans les moments les plus durs de ma thèse et sans elles, ce mémoire n'aurait probablement jamais été rédigé.

Je souhaite aussi remercier et montrer ma gratitude pour l'engagement des examinateurs sur ce manuscript qui ont accepté d'accomplir cet important travail, à savoir les professeurs Catherine Achard et Hedi Tabia.

Je remercie aussi mes directeurs de thèse, Jean-Philippe Vandeborre, Hazem Wannous ainsi que Laurent Grisoni pour leur patience et suivie amical et professionnel durant ces trois années. Leur compréhension et leur patience m'ont certainement aussi permis d'arriver au bout de ces trois années de thèse. Je me souviendrais aussi de nos intéressantes discussions autour de ce travail de recherche dont une pensée s'accompagne aussi vers toute l'équipe de travail de recherche MINT au sein du laboratoire CRIStAL.

Je remercie enfin mes collègues et compagnons de thèse, Vivien Bernard, Victor Delvigne, Sami Barchid ainsi que Oriane Lanseman qui représentent de fabuleuses rencontres et sans qui l'issue de ces trois années aurait été probablement bien différente. Ils ont représenté un soutien moral incontestable et m'ont offert de bien intéressantes discussion ainsi que des moments de rire inoubliable. Je remercie aussi

Antoine Facchini.

Résumé

Les gestes de la main constituent le médium de communication non verbal le plus naturel et intuitif pour utiliser un ordinateur, et les efforts de recherche relatifs en ont récemment stimulé l'intérêt. De sucroît, l'analyse et l'interprétation du comportement humain à partir de signaux visuels est l'un des domaines les plus animés et recherchés de la vision par ordinateur. Afin de contribuer à ce champ de recherche, notre travail s'articule autour de la technologie de l'apprentissage automatique, plus particulièrement autour de l'apprentissage profond. Depuis peu, les réseaux de neurones profonds ont récemment prouvé leur remarquable efficacité dans de nombreux domaines de recherche et ont ainsi permis aux chercheurs de faire de considérables avancées en terme d'efficacité et de robustesse pour résoudre le problème de reconnaissance de gestes et d'actions de la main.

Le principal objectif de cette thèse est de proposer un système d'assistance à l'utilisateur durant des activités orientées vers des objectifs précis, par exemple médical avec un assistant d'opérations ou d'auto-rééducation, ou encore dans l'industrie automobile avec un système d'assistance avancée pour la conduite, le tout sous la forme la plus intuitive et discrète possible. Ainsi, ce système observera les mains de l'utilisateur pour générer des commentaires contextuels en rapport avec le système de reconnaissance de gestes intégré. Cette thèse combine donc des techniques des domaines de recherche de la vision par ordinateur, avec la reconnaissance de gestes et les objets manipulés par l'utilisateur, et de réalité augmentée pour proposer un outil d'intervention et de correction. Pour cela, ces travaux explorent la récente architecture de réseau neuronal nommée Capsule Network qui n'a encore jamais été

utilisée dans un problème de reconnaissance de gestes malgré la proposition de résultats prometteurs dans d'autres domaines. Une base de donnée extraite à l'aide du casque de réalité augmentée Microsoft HoloLens pour le problème de reconnaissance d'actions orientée vers l'apprentissage de l'instrument du piano et à des fins purement applicatif est aussi proposée, ainsi que des expérimentations pour prouver qu'il est possible d'entraîner une méthode dessus ainsi que pour continuer à démontrer l'efficacité de notre architecture neuronale.

Mots clés : Vision par ordinateur, apprentissage profond, interaction homme-machine

Abstract

Hand gestures are the most natural and intuitive non-verbal communication medium while using a computer, and related research efforts have recently boosted interest. Additionally, the analysis and the interpretation of human behavior from visual input is one of the trendiest computer vision fields of research. To contribute to this field of research, our work revolves around the technology of machine learning, particularly around deep learning. Recently, deep neural networks have proven their outstanding effectiveness in many areas of research and they allowed researchers to make a significant jump in robustness and efficiency to solve the problem of hand gestures and actions recognition.

The main goal of this thesis is to propose an assistance system to the user during activities oriented towards very specific objectives, for example as medical with assisted surgery or self-rehabilitation, or in automobile industry as advanced assistance to driver. The system should also be develop so the user could interact with it in an intuitive and discrest way. Hence, this system will observe the hands of the user and will generate contextual comments according to the integrated gesture recognition system. This thesis then combines research techniques in computer vision on the recognition of gestures and manipulation of objects by the user, and augmented reality to propose a corrective intervention tool. For that, our works explore a recent architecture of neural network named Capsule Network, whose use has not yet been explored in hand gestures and actions recognition but which has already shown promising results in other areas. A database extract with the help of the Microsoft HoloLens augmented reality device for the problem of action recognition focused towards piano instrument learning and for purely application

purposes is also proposed as well as some experimentations to prove that the dataset can be use to train an algorithm but also to still prove the efficiency of our neural network.

Keywords: Computer vision, deep learning, human-machine interaction

Table des matières

Table des figures

Liste des tableaux

Introduction

"Les attitudes, gestes et mouvements du corps humain sont risibles dans l'exacte mesure où ce corps nous fait penser à une simple mécanique."

Henri Bergson, Le Rire.

Contexte

Par ces mots, Henri Bergson nous décrit une réalité. Le geste est une simple mécanique qui pourrait dès lors être analysée de manière concrète par un algorithme qui s'efforcera d'en extraire et comprendre ses composantes spatiale et temporelle. Malheureusement, l'outil primordial de l'Homme qu'est sa main, revêt une complexité difficilement appréhendable de part la nature même de sa forme. En effet, la main est composée de cinq doigts pouvant se mouvoir indépendamment les uns des autres. De surcroît, selon l'angle de celle-ci, des occlusions peuvent se produire et sa position exacte peut dès lors devenir difficilement perceptible. Or, si l'oeil et le cerveau humain, fort de plusieurs années à constamment observer des mains effectuer divers gestes et actions, que ce soit par le biais de ses propres mains ou celles des autres, est capable de reconnaître exhaustivement la position de n'importe quelle main dans n'importe quelle position ; il est plus difficile de développer un algorithme bénéficiant d'une telle érudition.

De plus, même si l'Homme, de part son expérience, peut décrire précisément la nature du geste ou de l'action qu'il observe, analyser et reproduire à l'exactitude ces gestes peut se relever beaucoup plus complexe selon la difficulté de la tâche. Souvent, pour des actions relatives aux sports, à l'art ou à diverses professions manuelles,

l'intervention et les conseils avisés d'un expert extérieur, qui connaît et maîtrise le geste, peut être indispensable à la maîtrise de celui-ci. S'il est possible, à force de nombreuses erreurs et répétitions de l'action, de finir par la dompter, il serait intéressant de pouvoir développer de nouveaux systèmes d'interaction personnalisés et automatiques qui seraient capables de fournir en temps réel des indications sur les gestes observés.

L'arrivée de nouvelles technologies liées à la réalité augmentée (RA) et virtuelle (RV) permettent déjà d'obtenir un contexte où un utilisateur peut se retrouver en présence d'un appareil qui le suit partout, portatif et avec qui il peut être possible d'interagir avec les mains libres. Pour analyser leur environnement, ces appareils sont la plupart du temps équipés de caméra tournées vers l'extérieur et qui peuvent dès lors observer les mains de l'utilisateur. De plus, que ce soit par le biais d'une sur-couche transparente dans le cas de la RA, ou d'un écran opaque dans le cas de la RV, un médium de retour est possible afin de communiquer avec l'utilisateur en temps réel.

Avec la motivation de pouvoir épauler et compléter les conseils d'un expert et l'arrivée de ces technologies RA et RV, il devient intéressant de considérer à apporter des solutions et des méthodes d'évaluation automatique qui permettraient, par la suite, de développer de telles applications fournissant des retours en temps réel.

Problématique

Si l'estimation de la position, segmentation ou extraction de squelettes, corps humain ou encore de mains, à tendance à s'automatiser depuis plusieurs années, notamment grâce à divers appareils et technologies comme les caméras Microsoft Kinect, Leap Motion ou encore Intel RealSense, ce n'est pas encore tout à fait le cas de l'estimation et l'analyse de gestes et d'actions. Même si certaines solutions sont déjà apportées, notamment dans le casque de réalité augmentée Microsoft HoloLens, elles ne considèrent encore que des gestes simples et sans précision dans l'action. Des solutions extrêmement précises peuvent être retrouvées dans le milieu du cinéma, avec la motion capture, où nous sommes aujourd'hui capable de recopier

la gestuelle d'un acteur humain vers celle d'un personnage fictif, elles requièrent cependant un coût matériel immense, notamment en terme de nombre de caméras infrarouges demandées et supposent que le comédien porte une combinaison possédant plusieurs traqueurs. Cependant, les avancées technologiques et matérielles, notamment en terme de puissance de calcul, ont récemment permis aux méthodes d'apprentissage statistiques de considérablement s'améliorer et offrent des résultats de plus en plus satisfaisants.

Parmi ces méthodes statistiques, nous retrouvons celles de l'apprentissage automatique et notamment de l'apprentissage profond. Ce sont ces dernières méthodes qui ont le plus profité de ces avancées technologiques car, dites basées sur les données, elles peuvent traiter de plus en plus d'exemples et ainsi apprendre des modèles de plus en plus performants. L'utilisation de l'apprentissage profond a ainsi permis de nombreuses percées dans maints domaines de recherche, notamment celui de la vision par ordinateur. Or si aujourd'hui, certains problèmes de ce champ comme la reconnaissance d'images semble être de plus en plus maîtrisés, l'analyse de gestes et d'actions, que ce soit du corps humain ou juste de sa main connaît encore quelques difficultés. En effet, outre la complexité spatiale de l'étude de la main résultant de sa forme et des innombrables positions qu'elle peut prendre, l'étude d'un geste ou d'une action rajoute une dimension temporelle qui demande à être analysée. Cependant, un même geste peut être par exemple exécuté à des vitesses variables, avec des pauses, ralentissements ou accélérations en plein milieu de l'action. La composante temporelle ajoute donc une vraie complexité à la tâche.

De plus, l'ensemble des gestes réalisables, même contraint à un simple et unique contexte, est extrêmement varié et il peut dès lors être difficile, même pour un être humain, de savoir à partir de quel degré de différence d'exécution un geste peut être discriminé d'un autre. Lorsqu'un musicien joue de son instrument, ses mains peuvent jouer une même note ou un même accord avec une gestuelle totalement différente selon l'enchaînement futur qu'il devra effectuer. A l'inverse, en fonction de son exécution précédente, ou même juste par habitude selon son apprentissage et ses répétitions, il jouera une même note ou un même accord dans un même contexte

avec des gestuelles différentes.

Par ailleurs, au delà du fait que bon nombre de bases de données de gestes ou d'actions peuvent manquer de données à cause du côté laborieux de l'acquisition et empêche ainsi souvent leur obtention en grande quantité dans un temps et une mise en oeuvre raisonnable, ce qui ne favorise pas la qualité de l'entraînement de la méthode développée. Nous retrouvons aussi très peu de bases qui proposent un ensemble de gestes ou d'actions dans un contexte précis et cadré qui permettrait, une fois une méthode entraînée dessus, de pouvoir être utilisée dans un cas d'application en temps réel qui puisse avoir une réelle utilité autre que de prouver le bon fonctionnement de l'algorithme.

Contributions

Durant les travaux de recherches menés au cours de cette thèse, nous avons apporté des solutions au problème de l'analyse de gestes et d'actions que ce soit par le biais d'une nouvelle méthode de classification ou bien par l'élaboration d'une nouvelle base de données spécifique à être appliquée en cas d'utilisation en temps réel.

La motivation principale concernant la méthode de classification proposée est d'avoir une méthode légère mais performante qui puisse être appliquée en temps réel sur des appareils à la configuration technologique et à la puissance de calcul modeste. De plus, cette méthode se devait d'accepter en entrée des images, que ce soit en couleur, niveau de gris ou encore de profondeur afin de nécessiter le moins de pré-traitements possible des données acquises au fur et à mesure et ainsi économiser du temps de calcul. Cette motivation s'explique aussi par le fait que la grande majorité des casques de réalité augmentée ou virtuelle sont tous équipés d'un capteur RGB ou de profondeur frontal. L'architecture proposée, basée sur des méthodes d'apprentissage profond a été testée et validée sur des bases de données de gestes et d'actions exigeantes de l'état de l'art comme First Person Hand Action de Garcia *et al.* [1] ou encore Dynamic Hand Gesture de De Smedt *et al.* [2] et, outre une amélioration des résultats de classification, promet aussi un nombre de calculs

nécessaire à chaque inférence et analyse d'image bien moindre que celles des autres méthodes de l'état de l'art.

En ce qui concerne la contribution de la base de données, la motivation maîtresse était d'acquérir un ensemble de vidéos d'actions effectuées par l'utilisateur à la première personne et dans un contexte bien précis pour pouvoir ensuite être utilisée de manière intéressante et pertinente dans le cas réel d'une application de système d'assistance personnalisée sur casque de réalité augmentée. Pour pouvoir être appliquée en temps réel, les données ont été récoltées directement à l'aide des capteurs présents sur le casque de réalité augmentée Hololens, à savoir des vidéos en couleur, niveau de gris ainsi que de profondeur permettant ainsi de posséder une extraction de données qui sera identique en cas d'application développée sur un casque AR similaire.

Ces deux contributions associées, en appliquant la méthode de classification de gestes ou d'actions sur notre base de données, nous permettent ainsi de prouver qu'il est possible de créer un système d'assistance personnalisée. L'objectif final étant la reconnaissance d'un geste en temps réel et, par la suite éventuellement informer via des retours visuels sur l'écran transparent d'un casque de réalité augmentée, les éventuelles erreurs ou améliorations qui pourraient bénéficier à la gestuelle de l'utilisateur.

Plan du mémoire

Le mémoire est organisé et chapitré en 5 parties :

— **Chapitre 1** : Ce chapitre se concentre sur la description, l'analyse et la critique de l'état de l'art en ce qui concerne tout particulièrement l'analyse de gestes et d'actions effectués avec la main, à savoir les différentes bases de données existantes ainsi que les méthodes et solutions déjà explorées et proposées, notamment dans le domaine de l'apprentissage profond. Nous verrons aussi les recherches de cas plus applicatifs de la reconnaissance de gestes, notamment pour des systèmes d'interaction homme-machine ou de systèmes d'assistance.

— **Chapitre 2** : Ce chapitre explique en profondeur le fonctionnement des méthodes d'apprentissage profond, notamment les architectures particulières à la résolution de notre problème, à savoir celles qui permettent l'analyse d'images et celles qui permettent l'étude de données temporelles comme les vidéos. Nous décrivons notamment en détail la base, les principes et le fonctionnement d'un réseau neuronal à capsules, architecture fondatrice de la solution que nous détaillons dans le chapitre suivant.

— **Chapitre 3** : Ce chapitre détaille une nouvelle méthode d'analyse et de classification pour la reconnaissance de gestes et d'actions de la main que nous proposons. Elle est développée à l'aide d'un algorithme d'apprentissage profond basé sur l'architecture de réseau neuronal à capsule, pour l'extraction et l'analyse de caractéristiques spatiales, ainsi que de l'implémentation d'un module de décalage temporel appliqué aux capsules pour l'analyse et l'extraction de caractéristiques temporelles. La description de cette méthode est suivie d'une description des expérimentations menées, que ce soit sur différentes bases de données ou bien pour prouver les bénéfices du module de décalage temporel.

— **Chapitre 4** : Ce chapitre propose une toute nouvelle base de données de reconnaissance d'actions de la main. Il s'agit de la première base de ce type à avoir un caractère purement applicatif en proposant une base contenant des vidéos de gammes musicales bien effectuées ou mal effectuées au piano et à pour ambition, après avoir développé et entraîné une méthode de classification dessus, de pouvoir utiliser cette dernière directement dans un casque de réalité augmentée et ainsi proposer une base de système d'assistance gestuelle personnalisée à l'apprentissage du piano. Des expérimentations sont aussi effectuées dessus pour continuer à démontrer l'efficacité de notre méthode proposée au chapitre précédente.

— **Conclusion** : Ce dernier chapitre termine ce mémoire en rappelant la problématique principale ainsi que les réponses et contributions apportées pour y répondre. Ce chapitre et le mémoire se conclut sur les perspectives en vue de poursuivre ce travail de recherche.

Publications

Durant ces travaux de recherches, des solutions et contributions ont été apportées à divers problèmes de reconnaissance de gestes et d'actions qui ont ainsi donné lieu à plusieurs publications :

— F.M. Caputo, S. Burato, G. Pavan, T. Voillemin, H. Wannous, J.P. Vandeborre, M. Maghoumi, E.M. Taranta, A. Razmjoo, J.J. LaViola Jr, F. Manganaro, S. Pini, G. Borghi, R. Vezzani, R. Cucchiara, H. Nguyen, M.T. Tran, A. Giachetti *Online gesture recognition*, Eurographics Workshop on 3D Object Retrieval, 3D Shape Retrieval Contest - 2019 [3]

— T. Voillemin, H. Wannous, J.P. Vandeborre, *2D Deep Video Capsule Network with Temporal Shift for Action Recognition*, ICPR - 2020 [4]

Ainsi que des travaux en cours de publication ou de soumission :

— T. Voillemin, H. Wannous, J.P. Vandeborre *Deep Video Capsule Network avec Décalage Temporel pour la Reconnaissance d'Action*, admis à CORESA - 2021 [5]

— H. Wannous, T. Voillemin, J.P. Vandeborre *Continuous Hand Gesture Detection and Recognition using Deep Coarse and Fine Hand Features*, en soumission à 3DV - 2021 [6]

— T. Voillemin, H. Wannous, J.P. Vandeborre *FirstPiano : A new dataset oriented towards AR application*, en soumission à ICIAP - 2021 [7]

Etat de l'art

1.1 Introduction

La reconnaissance de gestes constitue un domaine de la vision par ordinateur qui a pour objectif la reconnaissance de l'évolution temporelle et spatiale du mouvement de la main humaine dans son ensemble, son déplacement global ainsi que de chacun des doigts la composant. Ce domaine s'étend vers la reconnaissance d'actions qui combine la reconnaissance de gestes avec la manipulation d'objets comme par exemple dans le cadre de pratiques de sports ou de différentes professions. Il s'agit de champs de recherche en pleine expansion depuis plusieurs années notamment par son aspect applicatif qui se fait de plus en plus ressentir avec l'arrivée émergente de technologies comme les casques de réalitée augmentée (RA) qui viennent redéfinir la conception d'interaction homme-machine. En effet, ces nouvelles interfaces sont motivées et utiles à des utilisations autonomes, sans l'aide de clavier, souris ou quelconque manette si ce n'est les mains de l'utilisateur pour notamment naviguer dans les menus. C'est pourquoi de nombreuses recherches se sont penchées sur ces domaines et leurs défis respectifs, nous nous appliquerons donc, dans ce chapitre, à discuter et définir ces différents défis ainsi que d'exposer et d'étudier les approches principales de l'état de l'art pour les résoudre. Au sein de ce chapitre, après avoir expliqué en profondeur le problème et les défis de la reconnaissance de gestes et d'actions et exposé ses différentes applications possibles, nous étudierons dans un premier temps les bases de données existantes avant d'observer les différentes méthodes existantes pour résoudre ce problème. Nous finirons sur les différentes applications déjà

développées.

1.1.1 Reconnaissance de gestes et d'actions

Dans le cadre de cette thèse, nous nous concentrons sur l'analyse et la reconnaissance de gestes et d'actions effectués avec une ou deux mains. L'objectif de ces reconnaissances est d'étudier la composante spatiale des mains et des éventuels objets manipulés ainsi que la dynamique de leur évolution temporelle afin de déterminer le geste ou l'action effectué. Il s'agit donc d'un problème de classification. Deux différents types de données sont couramment utilisés pour analyser ces séquences. D'un côté les flux vidéos, qu'ils soient de couleur, de profondeur ou optique [8], constituent les données les plus facilement accessibles, ou nécessitant un très faible pré-traitement dans le cas du flux optique, notamment par n'importe quel type d'utilisateur, les caméras RGB composant la très grande majorité des smartphones et les caméras de profondeur étant, de nos jours, du matériel très peu coûteux et accessible. De l'autre côté, les séquences du squelette de la main constituent une information beaucoup plus fine et précise, permettant d'obtenir l'évolution temporelle de chacune des articulations de la main, cependant, ces séquences demandent, soit des méthodes de pré-traitements lourdes et coûteuses, demandant un temps de calcul supplémentaire à l'algorithme de classification à l'instar de DeepPrior [9] qui extrait le squelette de la main à partir d'un flux de profondeur, soit du matériel coûteux ou peu accessible comme les caméras Leap Motion qui extrait le squelette à partir de capteurs infrarouges ou encore des capteurs magnétiques attachés à chaque articulation pour traquer leurs déplacements.

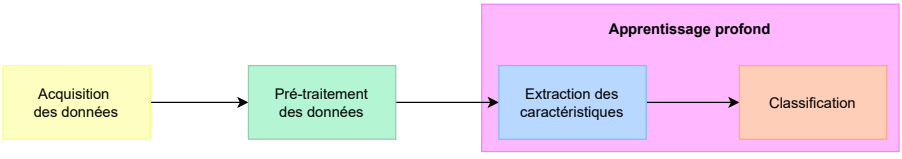

FIGURE 1.1 – Illustration des différentes étapes pour un processus générique de reconnaissance de gestes ou d'actions

Ces données sont ensuite analysées puis classifiées, généralement à l'aide de méthodes d'apprentissage automatique (voir Figure 1.1). Notons que, lors de l'utilisation de méthodes dites "handcrafted", la phase d'extraction des caractéristiques est bien distincte de la phase de classification. La partie classification est généralement une méthode d'apprentissage automatique comme les machines à vecteurs de support (SVM) [10] ou des forêts d'arbres décisionnels [11]. A l'inverse, dans le cas d'utilisation de méthodes d'apprentissage profond, ces deux phases d'extraction et de classification sont regroupées au sein du réseau de neurones utilisé qui s'occupe de ces deux phases en même temps, l'inconvénient étant que dans la majorité des cas, la partie extraction du réseau agit comme une boîte noire. La classification peut être correcte *in fine*, il est compliqué de savoir comment et quelles caractéristiques ont été extraites par le réseau pour y arriver. Outre la motivation de trouver des algorithmes qui proposent de meilleurs résultats de classification, la recherche de solutions plus légères et moins coûteuses en puissance de calcul peut être observée, notamment pour justifier l'application de ces algorithmes dans des cas d'utilisations réelles. De plus, nous pouvons aussi différencier les algorithmes qui sont capables de traiter une séquence, vidéo ou de squelettes, en temps réel, donnée après donnée, et qui proposent ainsi un résultat de classification à chacune de ces données, des algorithmes qui traitent les séquences dans leur entierté. Cette différenciation est primordiale dans des cas d'applications, notamment d'interaction homme-machine, où l'algorithme en temps réel pourra, non seulement éventuellement proposer une classification anticipée, mais pourra fonctionner continuellement à mesure du flux de données extrait là où un algorithme hors ligne nécessitera que l'utilisateur lui indique le début et la fin de la séquence à analyser.

1.1.2 Applications

La reconnaissance de gestes et d'actions de la main présente de nombreuses applications motivées pour redéfinir l'interaction homme-machine, que ce soit, par exemple, par l'arrivée de nouvelles technologies ou encore la volonté d'accessibilité à des personnes en situation de handicap qui ne peuvent pas manipuler aisément, voire

pas du tout, les actuelles interfaces d'interactions avec leurs appareils. Du point de vue des nouvelles technologies, on retrouve des fantasmes de ce genre d'interaction depuis déjà plusieurs dizaines d'années dans divers médias comme le cinéma et ses films de science-fiction où les personnages peuvent se retrouver à manipuler des hologrammes ou écrans verticaux simplement avec leurs mains et divers gestes comme pincer, pointer ou balayer. Aujourd'hui, ces fantasmes commencent à se concrétiser notamment dans le domaine des casques de réalité augmentée où la manipulation des menus se fait principalement via des gestes simples d'une ou des deux mains de l'utilisateur (voir Figure 1.2). Des applications peuvent aussi se voir être développées dans le cadre de la réalité virtuelle où la reconnaissance de main de l'utilisateur en face du masque serait utilisée. En effet, même si la plupart des systèmes proposent des manettes pour manipuler le monde virtuel, ces applications se verrait plus immersives et naturelles avec une manipulation exclusivement par les mains.

FIGURE 1.2 – Exemple d'utilisation du casque RA Hololens (`https://onetech.vn/service/hololens`)

La plupart des casques RA ont actuellement un système de reconnaissance de gestes très basique, uniquement destiné à la manipulation d'interface et qui ne reconnait en général qu'un très faible nombre de gestes. Il y a donc un réel besoin à implémenter un système plus complexe pour exploiter plus profondément la technologie de réalité augmentée et proposer des applications de manipulation d'objets et d'interfaces virtuels ou d'assistance personnalisée en fonction des gestes reconnus et ainsi offrir un médium d'interaction beaucoup plus subtil, discret et immersif.

FIGURE 1.3 – Exemple de gestes de langue des signes [12]

En ce qui concerne l'accessibilité, nous pouvons trouver la volonté de mettre au point de nouveaux moyens d'interaction homme-machine afin de permettre à des personnes en situations de handicap, que ce soit sensoriel ou moteur, d'utiliser un ordinateur ou de communiquer plus facilement avec tout le monde, comme par exemple via la reconnaissance de la langue des signes pour les personnes non-voyantes (voir Figure 1.3).

Enfin, la manipulation d'objets et l'utilisation de gestes manuels précis sont souvent les composantes principales de bons nombres de pratiques professionnelles ou de loisirs. C'est pourquoi nous pouvons aussi trouver de nombreuses possibilités d'applications dans ces domaines, en proposant des interfaces d'assistances personnalisées pour certaines professions, comme la chirurgie par exemple, où le professionnel pourrait se voir afficher et indiquer des informations sur la suite de l'opération à effectuer en fonction des gestes ou actions actuellement effectués et reconnus. Dans la vie de tous les jours, avec la potentielle popularisation des technologies RA ouvertes au grand public, des applications similaires pourraient être bénéfiques comme une assistance durant l'élaboration d'une recette de cuisine ou encore de l'aide et des corrections, par exemple, dans le cas d'apprentissage d'instruments de musique.

1.2 Bases de données

Avec l'émergence et la plus grande accessibilité des capteurs RGB ou de profondeur au début des années 2010, de nombreuses recherches se sont penchées de plus en plus activement sur les problèmes de reconnaissance de gestes et d'actions. Jusque là, la plupart des bases de données étaient principalement constituées de

simples images à reconnaître et classifier, comme ImageNet [13] ou NTU Hand Digit [14]. Ces nouveaux défis de reconnaissance introduisent des bases de données à des dimensions beaucoup plus importantes notamment parce qu'elles sont composées de vidéos. De plus, notons que leur arrivée se joint avec la démocratisation des méthodes d'apprentissage profond, notamment avec les impressionnants résultats du réseau neuronal convolutif AlexNet [15] sur la base ImageNet. Or, ces méthodes sont dites basées sur les données, c'est-à-dire que leur principale source d'entraînement est constituée par le nombre et la diversité de données envoyées au réseau durant son entraînement. C'est pourquoi on remarque aussi en cette période l'arrivée de bases de données à la taille de plus en plus conséquente pour cause d'une demande plus importante de données pour les méthodes d'apprentissage profond.

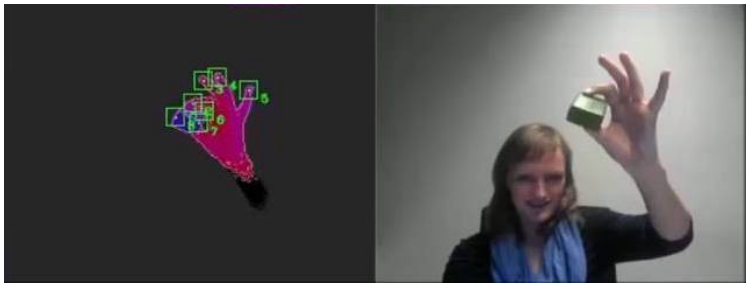

FIGURE 1.4 – Exemple d'estimation de la position de la main [16]

1.2.1 Reconnaissance de gestes de la main

Les bases de données de gestes se sont démocratisées et multipliées à partir des années 2010 avec l'arrivée de capteurs RGB et de profondeur bon marché. Précisons qu'il ne faut pas confondre le problème de reconnaissance de gestes avec celui d'estimation de position de la main (voir Figure 1.4), comme le propose par exemple les bases Dexter [16] ou encore UCI-EGO [17]. Même si, structurellement, les bases sont extrêmement similaires, à savoir des images ou séquences vidéos enregistrant la ou les mains d'une ou plusieurs personnes effectuant divers actions, le problème d'estimation de position de la main est un défi de segmentation, à savoir déterminer un masque qui permet d'isoler la main dans l'image ou déterminer la position du

squelette des différentes articulations de la main alors que le problème de reconnaissance de gestes est un défi de classification, à savoir attribuer à l'image ou la séquence d'images, une classe représentant un geste précis. Nous présentons ici les bases de données de reconnaissance de gestes les plus populaires (voir Tableau 1.1) dont nous allons expliciter les quelques différences et particularités.

Contexte : Trois contextes différents peuvent être rencontrés dans les bases de données de reconnaissance de gestes. La plupart des premières bases développées se penchent exclusivement sur le problème de reconnaissance de la **langue des signes** [18, 19, 20, 21, 22, 23, 24]. Composées en général de 10 à 30 labels à identifier, elles n'ont pas pour motivation de contenir exhaustivement le lexique de la langue des signes mais permettent d'entraîner des méthodes à reconnaître la gestuelle pour représenter l'alphabet et/ou la numérique. Le deuxième contexte observé est celui de la **reconnaissance de gestes pure** [14, 25, 26, 27, 28, 2, 29, 30]. Il s'agit presque exclusivement de gestes enregistrés dans un point de vue à la troisième personne, si ce n'est pour EgoGesture [29] qui propose un point de vue à la première personne. Il s'agit de proposer une base de données constituée de gestes qui pourrait être utilisée dans le cadre de nouvelles interactions homme-machine modulées par le geste à l'aide d'une caméra RGB ou de profondeur, comme par exemple un balayage de la main, pincer avec les doigts ou encore tracer une forme simple. Le dernier contexte est très proche du précédent mais dans un cadre très précis, à savoir la reconnaissance de gestes dans le cadre de **conduite de véhicule** [31, 32]. Il s'agit des premières bases très orientées applications puisque dans un contexte et un environnement très précis, à savoir de courtes séquences de conduite où le conducteur effectue un mouvement de la main droite en face d'une caméra placée sur le tableau de bord.

Temporalité : La temporalité d'une base constitue un élément important quant à l'approche que la solution devra avoir. Elle peut être **statique**, dans ce cas, une donnée de la base est une image contenant la main au moment où le geste est effectué, il s'agit dans ce cas d'étudier uniquement spatialement la position de la main et de ses doigts et est surtout retrouvée dans le cas de bases de données de la langue des

signes car ces dernières se concentrent sur la reconnaissance de lettres ou de chiffres uniquement. Dans ce cas de figure, ce n'est pas l'évolution du mouvement de la main qui est important mais la position finale de celle-ci qui représente la lettre ou le chiffre à identifier. Cependant, la plupart des bases récentes ont une approche **dynamique** et sont ainsi constituées de séquences vidéo. il s'agit d'une approche primordiale si l'évolution du mouvement de la main est indispensable pour différencier deux gestes proches mais différents, par exemple un balayage de la gauche vers la droite serait indifférenciable d'un balayage de la droite vers la gauche dans une base à temporalité statique. Dans ce cadre, la méthode proposée doit pouvoir étudier l'aspect spatial de la position et la forme de la main ainsi que l'aspect temporel de son déplacement dans le temps. Par exemple, dans la base DHG [2] (voir Figure 1.5), nous pouvons retrouver deux classes de gestes identiques mais qui sont effectués, dans un cas avec la main ouverte, et dans l'autre avec uniquement l'index. Notons que dans la grande majorité des cas, ces bases proposent une séquence par geste, or, nous pouvons aussi retrouver plus rarement des variantes où le problème est complexifié et où une séquence est un enchaînement de gestes comme dans la base ODHG [33]. Dans ce cas précis, il faut proposer une méthode qui soit capable de labéliser l'entièreté des images de la séquence vidéo pour différencier les différents gestes effectués au cours d'une même vidéo.

FIGURE 1.5 – Exemple de gestes de la base DHG et ses données de profondeur et de squelettes de la main associées [2]

Données & Taille : Si les capteurs et données RGB et de profondeur constituent les types de données les plus couramment utilisés et proposés dans les bases de reconnaissance de gestes, nous pouvons aussi noter que certaines proposent des types de données plus spécifiques permettant, si l'approche le permet en entrée, d'avoir des informations plus précises afin d'aider à la classification. La possibilité d'utiliser un masque indiquant la position du corps [21], des doigts [22] ou de la main [26] permet d'extraire précisément ces parties de l'image ou de la séquence vidéo et de s'abstenir d'informations inutiles. D'autres bases proposent aussi le squelette de la main sous forme de la position XYZ des différentes articulations [28, 2] ce qui permet d'avoir directement d'extrait les caractéristiques spatiales de la main et de se concentrer sur la relation des différentes articulations les unes par rapport aux autres ainsi que de leur évolution temporelle. Le problème de ces types ajoutés est que, dans un cas d'utilisation réelle de la méthode par la suite, il faut pouvoir extraire de nouveau ces types de données à la volée à l'inverse d'un simple flux RGB ou de profondeur qui peut directement être utilisé. Concernant la taille de ces bases, notons dans un premier temps qu'il faut bien différencier la taille des bases statiques de celle dynamique, les premières étant composées d'images il est naturel qu'elles soient de plus grande dimensionnalité que les secondes, composées de vidéos, plus difficile à extraire mais où chacune d'entre elles est composée de dizaines voire centaines d'images. Dernièrement, nous pouvons aussi noter l'arrivée de bases de données dynamiques de très grande taille [29, 30], motivées par la demande des méthodes d'apprentissage profond qui sont basées sur les données. La base de données Jester [30] et ses centaines de milliers de séquences vidéo possible grâce à l'aspect collaboratif d'internet en demandant à n'importe quel utilisateur d'envoyer une ou plusieurs vidéos à l'aide de leur webcam est presque trop grande puisque l'utilisation de toutes ces données au moment de l'entraînement demande, soit une énorme puissance de calcul, soit un temps d'entraînement extrêmement long, ce qui peut la rendre inutilisable avec du matériel à la puissance de calcul modeste.

Base	Contexte	Nb classes	Temporalité	Données	Taille données	Année
ASL finger-spelling [18]		24	Statique	RGB, Profondeur	48,000	2011
MSR Gesture 3D [19]		12	Dynamique	Profondeur	336	2012
UESTC-ASL [20]		10	Statique	RGB, Profondeur	1,100	2013
ChaLearn [21]	Langue des signes	20	Dynamique	RGB, Profondeur, Pos. corps	7,754	2014
Marin *et al.* [22]		10	Statique	Profondeur, Pos. doigts	1,400	2014
HUST-ASL [23]		34	Statique	RGB, Profondeur	5,440	2017
Gesture [24]		14	Dynamique	RGB	126	2009
NTU Hand Digit [14]		10	Statique	RGB, Profondeur	1000	2011
SKIG [25]		10	Dynamique	RGB, Profondeur	2,160	2013
Xu *et al.* [26]		10	Dynamique	Pos. main	1,600	2014
Wang *et al.* [27]		10	Statique	RGB, Profondeur	1000	2015
Handicraft [28]	Gestes de la main	10	Dynamique	Squelette main	300	2016
DHG14/28 [2]		28	Dynamique	Profondeur, Squelette main	2,800	2016
ODHG [33]		28	Dynamique	Profondeur, Squelette main	280	2017
Egogesture [29]		83	Dynamique	RGB, Profondeur	24,161	2018
Jester [30]		27	Dynamique	RGB	148,092	2019
Ohn-Bar *et al.* [31]		19	Dynamique	RGB, Profondeur	886	2014
NVGesture [32]	Scénario voiture	25	Dynamique	RGB, Profondeur	1,532	2016

TABLE 1.1 – Bases de données de gestes les plus populaires. Pour la taille des données, une base dynamique contient des séquences tandis qu'une base statique contient uniquement des images

1.2.2 Reconnaissance d'actions

FIGURE 1.6 – Exemple de reconnaissance d'activités du corps humain [34]

A l'instar de la reconnaissance de gestes, la reconnaissance d'actions est un problème de classification. Précisons que l'on parle généralement d'activité pour le corps entier tandis que l'on utilise plutôt le terme d'action lorsqu'il s'agit uniquement de la main. Ce problème est cependant plus complexe car il fait intervenir l'utilisation d'objets ou d'interactions plus complexes qui demandent une plus grande compréhension spatiale et temporelle notamment en rajoutant, par exemple, des problèmes d'occlusions de la main. Cependant, ce problème propose aussi de possibles cas d'applications réelles plus complexes et prometteurs. De la même manière, les bases de données de reconnaissance d'actions les plus populaires sont présentées (voir Tableau 1.2). Notons aussi que nous nous concentrons ici sur les bases de données de reconnaissance d'actions effectuées avec les mains mais que le problème est plus large et qu'il existe des bases de reconnaissance d'actions du corps humain, par exemple, dans des cadres sportifs (voir Figure 1.6) ou d'instruments de musiques comme par exemple les bases HMDB51 [34] ou encore UCF101 [35]. Ajoutons aussi qu'il est intéressant de noter que, à l'inverse des bases de reconnaissances de gestes, celles de reconnaissances d'actions de la main sont presque exclusivement du point de vue de la première personne, prouvant la motivation et la volonté d'utiliser et d'appliquer ces différentes bases et contextes à des cas d'utilisation réelles en utilisant des casques de réalité augmentée ou virtuelle qui proposent naturellement ce point de

vue. Enfin, étant donné que la manipulation d'objets est au cœur du problème de reconnaissance d'actions, il est aussi naturel que toutes les bases trouvables bénéficient d'une temporalité dynamique.

FIGURE 1.7 – Exemple de données d'actions de la base GTEA [36]

Contexte : Deux contextes différents d'actions effectuées avec la main peuvent être observés dans les différentes bases de données. Le premier est le cas simple d'**attraper des objets** [37, 38, 39] où la forme de l'objet est importante et où il est intéressant d'étudier la position que la main prend pour attraper les différents objets, par exemple du bout des doigts pour attraper un crayon ou la main en pince pour attraper une canette. Le deuxième contexte est plus populaire, il s'agit des *actions de l'activité quotidienne*, que ce soit des actions relatives à la cuisine (voir Figure 1.7), à l'interaction avec d'autres personnes ou encore avec des objets d'entretiens [36, 40, 41, 42, 43, 44, 1, 45]. L'un des manques que l'on peut trouver actuellement dans ces bases est l'absence de concentration sur un sujet et un contexte précis à l'instar des bases de reconnaissances de gestes qui pourraient directement se voir être appliquées dans des cas d'utilisation réelle d'interactions homme-machine. Les bases de reconnaissances d'actions sont généralement trop diversifiées et ne présentent pas

de volonté ou motivations à enregistrer un contexte particulier qui pourrait être appliqué réellement dans un système d'interaction personnalisée au sein d'un casque RA ou RV.

Données & Taille : De la même manière que pour les bases de reconnaissance de gestes, les données RBG ou de profondeur sont les données les plus communes que nous pouvons retrouver dans toutes les bases. Le squelette de la main [1] ou la direction du regard et position de la main [45] peuvent aussi être retrouvés et proposent les mêmes avantages et inconvénients que ceux exposés plus tôt dans le cadre de la reconnaissance de gestes. Le problème de reconnaissance d'actions étant plus récent que celui de gestes, nous retrouvons beaucoup plus de bases à la dimensionnalité importante, à savoir avec au moins plusieurs milliers de données d'entraînement, et, à l'instar de la base Jester [30], ADL [40], Something-Something [44] ou encore EGTEA [45] sont des bases qui posent plusieurs dizaines voire centaines d'heures de données d'entraînement, utile pour entraîner une méthode stable à la plupart des petites variations rencontrées mais qui proposent aussi des difficultés à être entraînées sur des configurations modestes, demandant des mois d'entraînement à chaque essai.

1.3 Méthodes de reconnaissance de gestes et d'actions

Avec l'apparition progressive de ces différentes bases de données et l'arrivée de technologies prometteuses pour diverses potentielles applications, de nombreux travaux et recherche se sont penchés sur la résolution de ces défis.

Dans cette section, nous nous concentrons principalement sur l'utilisation de méthodes d'apprentissage profond, cependant, évoquons tout d'abord qu'il existe des méthodes dites hand-crafted c'est à dire des méthodes où la partie d'extraction des caractéristiques, spatiales ou temporelles, est différenciée de la partie classification. A l'inverse des méthodes d'apprentissage profond, où la partie d'extraction agit la plupart du temps comme une boîte noire, l'algorithme des méthodes hand-crafted

Base	Contexte	Nb classes	Données	Taille données	Année
Yale [37]		33	RGB	18,210	2015
UT Grasp [38]	Attraper objet	17	RGB	20	2015
GUN-71 [39]		71	RGB, Profondeur	12,000	2015
GTEA [36]		61	RGB	525	2011
ADL [40]		18	RGB, Profondeur	10 heures	2012
WCVS [41]		10	RGB, Profondeur	5	2014
EgoHands [42]	Activité quotidienne	4	RGB	48	2015
THU-READ [43]		40	RGB, Profondeur	1,920	2017
Something-Something [44]		174	RGB	220,847	2017
FPHA [1]		45	RGB, Profondeur, Squelette main	1,175	2018
EGTEA [45]		106	RGB, Regard, Masque main	28 heures	2018

TABLE 1.2 – Bases de données d'actions les plus populaires. Les bases étant toutes de temporalité dynamique, elles contiennent toutes des séquences

est connu et maîtrisé, l'inconvénient étant qu'elles sont souvent bien moins performantes. Les deux types de méthodes partagent cependant un même défaut, à savoir que lors d'essais d'utilisation sur des données qui s'éloignent un peu trop de celles utilisées pour développer l'algorithme, les performances sont largement détériorées. Par exemple, De Smedt *et al.* [46] proposent d'utiliser la position 3D des 22 articulations de la main extraits directement par la caméra Intel RealSense. Ces données, ainsi que l'évolution de leur rotation et position, sont utilisées pour représenter un geste de la main en les encodant dans un vecteur Fisher à l'aide de modèles de mélanges Gaussien. La partie temporelle est encodée dans ce qu'ils nomment une pyramide temporelle et le tout est classifié à l'aide de machines à vecteur de support (voir Figure 1.8).

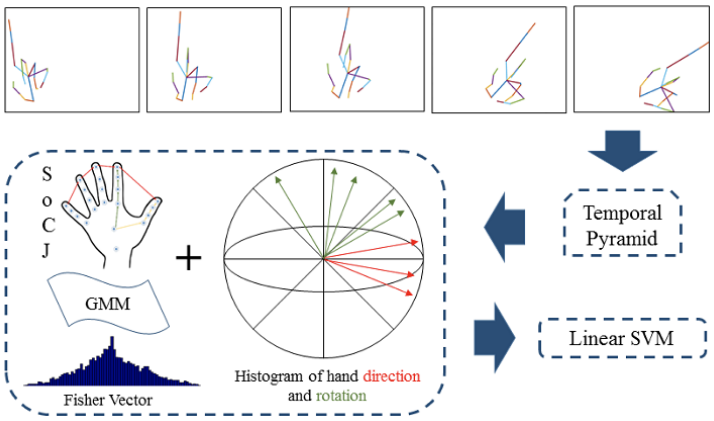

FIGURE 1.8 – Aperçu de la méthode de De Smedt *et al.* [46], les articulations sont récupérées et encodées de deux manières, temporelles et spatiales puis sont classifiées à l'aide d'un classifieur SVM

Plus récemment, Fang *et al.* [47] proposent une solution similaire, en utilisant uniquement des images de profondeur ou de couleur. L'image est segmentée afin de ne garder que la forme de la main sous forme de masque noir et blanc. Les contours de ce masque sont ensuite étudiés, de la même manière que le centre de la paume ou encore le centre de gravité de la main. Ces données sont aussi encodées sous forme de vecteur Fisher pour étudier les caractéristiques spatiales suivantes : les angles ainsi que la courbe de la main et les distances entre les différentes données extraites.

Un classifieur SVM est ensuite aussi utilisé pour la partie de classification.

Malgré des résultats intéressants sur les bases de données appliquées, ces méthodes sont cependant très dépendantes du format des données accessibles et requiert souvent des ajustements au moindre changement, ce qui les rend souvent performantes, mais très peu portables et souples aux changements. Nous pouvons d'ailleurs observer que les méthodes hand-crafted pures sont de moins en moins étudiées et recherchées. Cependant la partie extraction manuelle de caractéristiques reste tout de même étudiée, notamment pour servir de pré-traitements des données pour ensuite les classifier par des méthodes d'apprentissage profond.

Pour rappel, l'apprentissage profond constitue un ensemble de sous-méthodes de l'apprentissage automatique. Elles sont caractérisées par deux particularités, l'enchaînement de couches regroupant plusieurs opérations non linéaires. Une couche reçoit en entrée les résultats de la couche précédente, et elles permettent de modéliser différents niveaux d'abstractions, de bas niveaux à hauts niveaux selon l'enchaînement des couches. Ces méthodes sont dites basées sur les données car c'est en étudiant un grand nombre d'exemples différents qu'elles peuvent instancier les paramètres de leurs couches afin de converger vers une classification ou segmentation de plus en plus précise sur ces données (voir Chapitre 2).

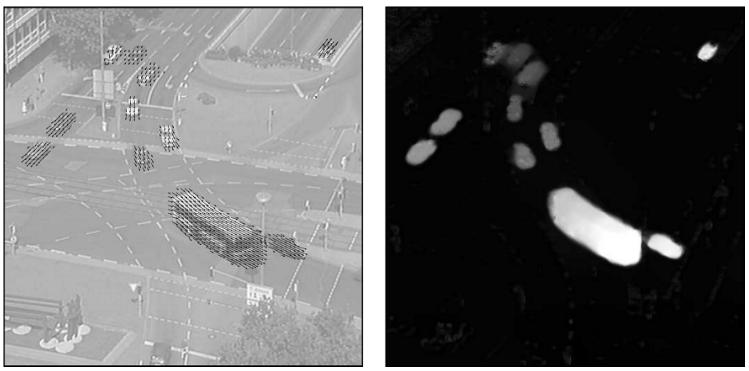

FIGURE 1.9 – Exemple de flux optique extrait d'images de flux routier [48]

1.3.1 Pré-traitement des données

Avant de nous attarder en profondeur sur ces méthodes, regardons les différents algorithmes de pré-traitement de données qui peut être indispensables pour certaines solutions d'apprentissage profond. Deux objectifs différents peuvent être observés pour ces algorithmes. Le premier consiste à générer, à partir d'une donnée d'entrée, une toute nouvelle donnée au formalisme complètement différent contenant une information différente. Parmi ces méthodes, on retrouve par exemple l'extraction de flux optique [48], qui, à partir de deux images consécutives d'une vidéo, en analysant le déplacement les pixels d'objets en mouvement, et en contrebalançant l'éventuel mouvement de caméra, permet d'extraire des images en niveau de gris où les pixels tendant vers le blanc représentent les objets en déplacement (voir Figure 1.9). Le flux optique est une donnée qui peut être intéressante, souvent en complément des images originales car perdant toutes formes d'informations spatiales statiques mais qui permet à la méthode qui le prendra en entrée de pouvoir se concentrer sur l'évolution temporelle des objets ou encore la main d'un utilisateur effectuant un geste, en mouvement. De plus, il s'agit d'une rare méthode où le traitement est stable et où il est sûr de bien extraire l'information recherchée.

Ce n'est pas forcément le cas d'une autre tâche, celle de l'estimation de la position de la main. Si elle permet d'extraire un squelette de celle-ci, à savoir la position 2D voire 3D des différentes articulations des doigts, du poignet et du centre de la paume, les méthodes les plus efficaces à l'heure actuelle sont celles basées aussi sur l'apprentissage profond. Nous pouvons par exemple retrouver la méthode de Oberweger *et al.* [9] qui base cette extraction à partir d'image de profondeur et dans des cas de point de vue à la troisième personne où la main est l'objet le plus proche de la caméra afin de pouvoir extraire un cube ne comprenant que celle-ci et ainsi pouvoir appliquer dessus des réseaux de convolutions (voir Figure 1.10). Zhou *et al.* [49] utilisent une méthode similaire en introduisant cette fois en fin de réseau une couche spécialement développée pour le problème d'estimation de position de la main puisqu'elle prend en compte la géométrie très particulière de celle-ci est s'assure donc que la sortie du réseau soit au moins cohérente avec la forme d'une vraie main.

Une approche différente peut être observée avec les travaux de Ge *et al.* [50], qui, au lieu d'utiliser des images de profondeur en entrée, prennent plutôt des nuages de points 3D de mains, ces données sont moins courantes mais contiennent déjà une information plus complète qu'une simple image ce qui leur permet de développer une méthode beaucoup plus légère simplement basée sur quelques couches entièrement connectées afin d'extraire le squelette.

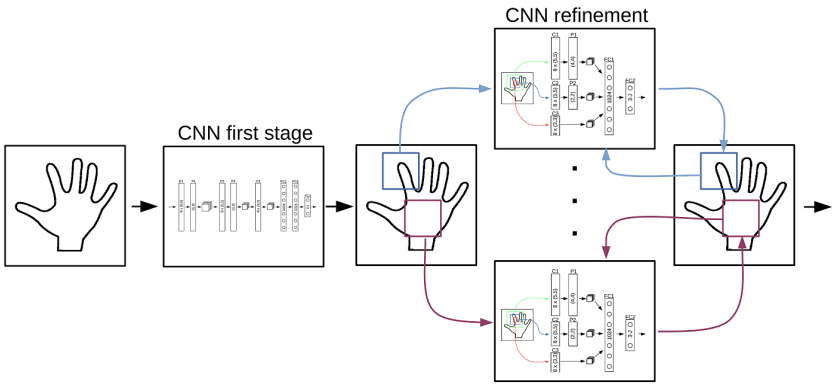

FIGURE 1.10 – Illustration du réseau utilisé par Oberwerger *et al.* dans leur réseau DeepPrior [9]

Même si l'information 2D ou 3D du squelette de la main peut être une information beaucoup plus complète et puissante qu'une simple image, nous pouvons observer dans l'état de l'art que, pour pouvoir l'extraire sans utilisation de technologie spéciale comme la caméra Leap Motion, les méthodes proposées reposent entièrement sur des méthodes d'apprentissage profond, qui, même si très performante, sont encore loin d'être exhaustives et de pouvoir être appliquées sans risque sur n'importe quelles données, pouvant ainsi induire à extraire de fausses informations. De plus, elle rajoute forcément une phase de calcul et un réseau neuronal supplémentaire à charger, espace et temps de calcul qui peut être très précieux pour des cas d'applications en temps réel.

Le deuxième objectif consiste, non pas à générer une nouvelle information, mais extraire une sous-partie de la donnée d'entrée. A l'instar du flux optique, la segmentation de la main consiste, à partir d'une image contenant une ou plusieurs mains, de

créer une image en masque noir et blanc où les pixels blancs décrivent l'emplacement de la main. Cette segmentation peut être utile dans le cadre de la reconnaissance de gestes de la main puisqu'elle permet d'extraire cette dernière à chaque image de la séquence et ainsi permettre à la méthode de ce concentrer uniquement sur elle en s'abstenant de toutes autres informations inutiles. Peu de méthodes d'apprentissage profond se retrouve à la résolution de ce défi, même si nous pouvons retrouver les travaux de Vodopivec *et al.* [51] qui utilisent un réseau neuronal convolutif pour segmenter la main à partir d'images RBG. Cependant, des méthodes d'apprentissage automatique plus classiques sont encore utilisées, notamment par l'utilisation de forêts d'arbres décisionnels sur du sous-échantillonnage de pixels [52, 53]. Cependant, ces méthodes perdent en intérêt pour des problèmes de reconnaissance d'actions où le contexte de la scène générale peut être très important pour la bonne identification de l'action.

Cependant, malgré les bénéfices qu'offrent ces algorithmes, notamment de permettre l'accès à de l'information plus fine qui pourrait être par la suite plus facilement interprétable par le réseau de neurones, elles rajoutent nécessairement une phase de calcul supplémentaire, une demande qui peut être impossible dans des cas d'applications réelles sur des configurations modestes comme celles des casques de réalité augmentée.

1.3.2 Méthodes d'apprentissage profond

Depuis les années 2010 et les résultats impressionnants du réseau AlexNet [15] sur la base de données ImageNet [13], notamment par rapport aux méthodes handcrafted habituellement utilisées, les méthodes d'apprentissage profond se sont largement popularisées dans un grand nombre de domaines, notamment la reconnaissance de gestes et d'actions.

Convolution 2D

Une méthode populaire est l'utilisation de convolution 2D, qui sera présentée dans le Chapitre 2.3, et qui prend un entrée une donnée sous forme de matrice 2D,

la plupart du temps, une image. Avec l'utilisation de ce type de réseau, on obtient généralement une solution qui fonctionne aussi bien pour la reconnaissance de gestes comme d'actions. En effet, une image contient directement des informations à la fois sur la main effectuant le geste ou l'action, mais aussi de l'éventuel objet manipulé. Ce n'est pas le cas, par exemple comme nous le verrons plus tard, des architectures de réseaux qui prennent en entrée le squelette de la main. Dans ce cas, même si la forme qu'elle prend peut donner de légers indices sur l'objet manipulé, cela est rarement suffisant pour de la reconnaissance d'actions.

L'utilisation de convolutions 2D pour résoudre ces problèmes reste assez rare à cause de la limitation de l'opération de convolution 2D à ne prendre en compte naturellement que les informations spatiales. Néanmoins, nous pouvons retrouver des travaux qui tentent de résoudre ces problèmes tout en bénéficiant de la légèreté de cette opération.

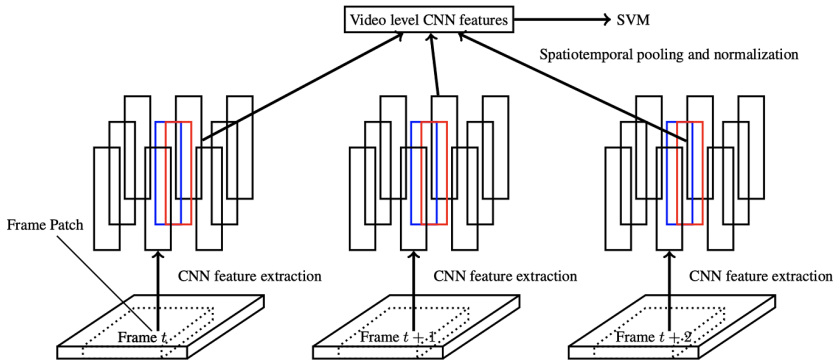

FIGURE 1.11 – Méthode proposée par Zha *et al.* [54] utilisant des CNNs 2D à des fins de reconnaissance d'actions

Ainsi, en 2015, Zha *et al.* [54] proposent d'extraire les caractéristiques spatiales de chaque image d'une vidéo à l'aide de CNN avant de les regrouper ensemble et, étant de bien plus faible dimensionnalité que les images d'origine, de pouvoir ensuite appliquer un SVM à des fins de classifications (voir Figure 1.11).

Sur le problème de reconnaissance de la langue des signes, Neverova *et al.* [55] proposent d'utiliser une architecture multi-modale qui utilise différents points de vue et informations de la vidéo, étudiés indépendamment les uns des autres avant d'être

regroupés. Ainsi, pour chaque image de la vidéo, un CNN est utilisé pour analyser la partie haute du corps, un autre pour chacune des mains de la personne analysée ainsi qu'un dernier pour étudier le flux audio. La classification se fait par vote de majorité pour labeliser la vidéo.

Wang *et al.* [56] proposent une approche similaire pour la reconnaissance d'actions en utilisant, pour une même vidéo, un flux spatial composé d'un CNN qui analyse une fenêtre d'images consécutives de la vidéo RGB. Un second flux temporel, composé d'un autre CNN qui analyse la même fenêtre mais sur le flux optique, permet à l'architecture de classifier efficacement une activité en faisant la fusion de chaque classification de chaque fenêtre de la vidéo (voir Figure 1.12). Zhou *et al.* [57] proposent des travaux relativement similaires où pour étudier l'aspect temporel d'une vidéo, décident de regrouper de différentes façons les images de la vidéos en la segmentant en paquet de 2, 3 ou 4 images. Ces différents paquets sont envoyés à des CNNs parallèles dont les résultats sont regroupés avant d'être classifiés.

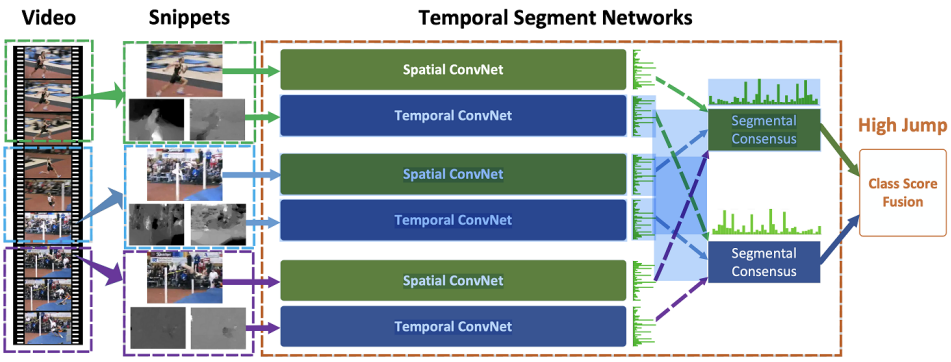

FIGURE 1.12 – Méthode proposée par Wang *et al.* [56] qui fusionne deux CNNs, l'un étudiant la spatialisation, l'autre la temporalité d'une vidéo d'action

Luvizon *et al.* [58] développent une architecture profonde à base de convolution 2D et ayant une ambition multi-tâches, à savoir l'estimation de la positon 2D et 3D du corps humain ainsi que la classification d'actions. Basée sur Inception-V4 [59], leur architecture permet d'extraire les caractéristiques visuelles à bas niveau du clip vidéo couleur passé en entrée ainsi que les cartes de probabilités. Ces deux données

agissent comme un mécanisme d'attention et permettent ainsi de se concentrer sur une partie précise de l'image. Leur réseau permet aussi, par régression de déterminer les coordonnées 2D ou 3D du squelette humain observé, ces deux données sont finalement assemblées pour inférer sur une classification du clip.

Tran *et al.* [60] proposent d'aller plus loin en observant les avantages de la convolution 3D pour l'étude de composantes temporelles et spatiales mais en conservant la légèreté d'une convolution 2D. Pour cela, ils décomposent l'opération 3D en deux opérations, une 2D pour l'analyse spatiale, l'autre 1D pour l'analyse temporelle. Plus récemment, Lin *et al.* [61] ont développé le module de décalage temporel qui permet, en faisant passer les images à un CNN, indépendamment les unes des autres, et en décalant au fur et à mesure les résultats des convolutions calculées pour une frame t à une frame $t+1$, permettant ainsi l'échange d'informations temporelles sans ajout de temps de calcul supplémentaire (voir Figure 1.13).

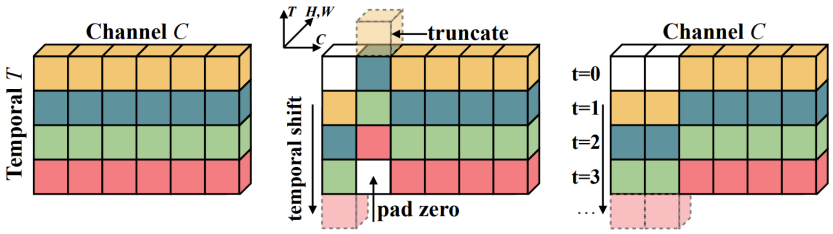

FIGURE 1.13 – Module de décalage temporel pour convolution développé par Lin *et al.* [61]

Nous retrouvons aussi plus rarement des travaux qui appliquent l'opération de convolution à des données de squelettes plutôt que d'images. C'est par exemple le cas de Weng *et al.* [62] qui proposent de mixer les opérations de convolution et de récurrence (voir Chapitre 2.5) pour pouvoir analyser les articulations d'un squelette avec les articulations directement reliées ou proches les unes des autres. Nous pouvons aussi retrouver plus récemment les travaux de Li *et al.* [63] qui utilisent une combinaison de convolution 1D et 2D sur des données de squelettes. Une première branche du réseau étudie l'évolution des coordonnées XYZ, indépendamment les

unes des autres tandis qu'une seconde branche, plus classique étudie de manière graphique l'ensemble des squelettes.

Réseau neuronal récurrent

L'utilisation de réseaux neuronaux récurrents (RNN) (voir Chapitre 2.5) a aussi été explorée. Comme nous l'évoquions plus tôt, ces réseaux prennent généralement en entrée des données de faibles dimensionnalités, dans notre cas, souvent le squelette de la main. Or, le squelette est rarement suffisant dans le cas de la reconnaissance d'actions, c'est pourquoi la plupart des méthodes suivantes se basent sur la reconnaissance de gestes, notamment quand la main est le sujet.

En effet, pour des reconnaissances d'actions du corps humain, notamment pour des activités sportives, la position et la dynamique du corps peuvent presque suffire pour identifier le sport pratiqué par le sujet. C'est ce sur quoi se reposent les travaux de Du *et al.* [64] qui utilisent une suite de couches de RNNs bi-directionnels en parallèle sur chaque partie du corps, jambe, bras, torse, avant de les regrouper successivement jusqu'à assembler le corps entier, toujours à l'aide de RNNs (voir Figure 1.14).

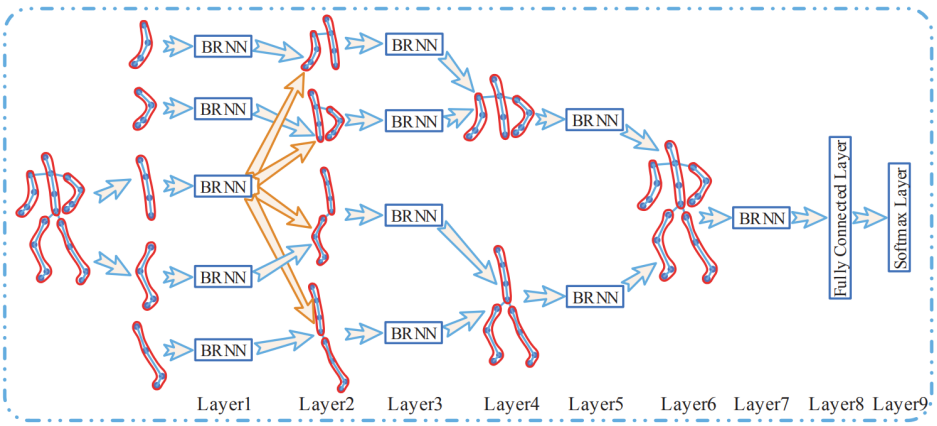

FIGURE 1.14 – Fonctionnement de la méthode de Du *et al.* d'assemblage progressive de couche de Bi-RNN qui travaillent au départ sur chaque partie du corps [64]

Quand le sujet est la main, les travaux sur les réseaux de neurones récurrents se concentrent principalement sur la reconnaissance de gestes pour les raisons que nous évoquions plus haut. Ainsi, Chen *et al.* [65] basés sur les travaux précédents, proposent un réseau similaire pour la reconnaissance des gestes de la main où une partie RNN étudie le mouvement des doigts tandis qu'une autre analyse le mouvement global de la main. Pour la résolution du même problème, De Smedt *et al.* [33] travaillent à partir des données extraites par la méthode de Oberweger *et al.* [9], à savoir le squelette à partir d'une image de profondeur avant de le passer à un RNN. La particularité vient du fait qu'ils récupèrent les données d'une couche avant la sortie du squelette extrait. Les résultats de cette couche sont ensuite envoyés à un RNN en parallèle dont la sortie est fusionnée avec celle du premier.

Nous pouvons aussi retrouver des travaux sur la reconnaissance d'actions à l'aide de couches récurrentes. Vaudaux-Ruth *et al.* [66] développent un réseau qui utilise dans un premier temps un réseau neuronal convolutif prenant en entrée une image ou un segment de vidéo afin d'extraire les caractéristiques spatiales et ainsi les compresser dans un vecteur de bien plus faible dimensionnalité que l'image d'origine. Ce vecteur est ensuite envoyé à une couche récurrente GRU qui l'associe au résultat de cette même couche appliqué à l'image ou clip précédente et dont la sortie sera renvoyée au suivant. La sortie est cependant aussi analysée par quatre opérations, une pour savoir si l'image couramment analysée doit être ignorée ou non, une autre pour savoir quelle devrait être la prochaine à analyser. Les deux dernières opérations servent à estimer la classification et à favoriser la bonne convergence du réseau. L'objectif principal est de proposer une méthode capable de repérer une action particulière au sein d'une séquence vidéo où plusieurs plans et actions se déroulent.

Les RNN ont un autre avantage, celui de pouvoir fournir une sortie à chacune des données de la série temporelle là où la plupart des autres architectures ne retournent qu'un label pour la globalité de la série. Profitant de cette particularité Guo *et al.* [67] proposent une architecture ayant pour but de traduire une vidéo passée en entrée en une phrase la décrivant.

Nous pouvons noter de derniers travaux qui se penchent sur un contexte plus applicatif, à savoir celui de la reconnaissance de gestes dans le domaine de la chirurgie. Van *et al.* [68] n'utilisent cette fois pas les squelettes de la main comme entrée de leur RNN mais la cinématique de deux petits outils qui servent au chirurgien à effectuer des points de suture. La fin du réseau est séparée en deux branches entièrement connectées, l'une pour déterminer la progression de chaque étape de l'opération, l'autre pour déterminer l'enchaînement des étapes (voir Figure 1.15).

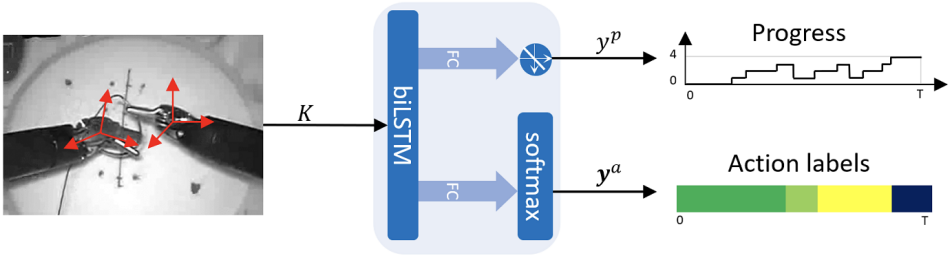

FIGURE 1.15 – Réseau développé par Van *et al.* [68] visant à la reconnaissance de gestes chirurgicaux

L'inconvénient de l'utilisation de réseaux neuronaux récurrents est qu'ils prennent en entrée des données de faible dimensionnalité et que, dans le cas de la reconnaissance de gestes et d'actions, seules les données de squelettes répondent à ces critères. Or, sauf matériel très particulier, comme la caméra Microsoft Kinect ou Leap Motion, il est obligatoire d'utiliser une méthode de pré-traitement qui va extraire ledit squelette. Or, ces méthodes rajoutent du temps de calcul supplémentaire qui pourrait être utilisé directement par celle de reconnaissance, de plus, ces méthodes ne sont pas encore assez fiables pour que l'extraction repose sur elles en toutes circonstances.

Convolution 3D

Nous avons observé plus tôt les inconvénients de l'application de méthodes de convolution 2D, à savoir l'impossibilité structurelle d'analyser à la fois des informations spatiale et temporelle. Une solution naturelle est l'utilisation de convolution 3D (voir Chapitre 2.5) dont l'opération est agrandie sur une dimension, à savoir celle de la temporalité pour l'analyse de vidéos. Il est d'ailleurs intéressant de noter que

l'extension de cette opération a été développée pour le problème de reconnaissance d'action du corps humain par Ji *et al.* [69].

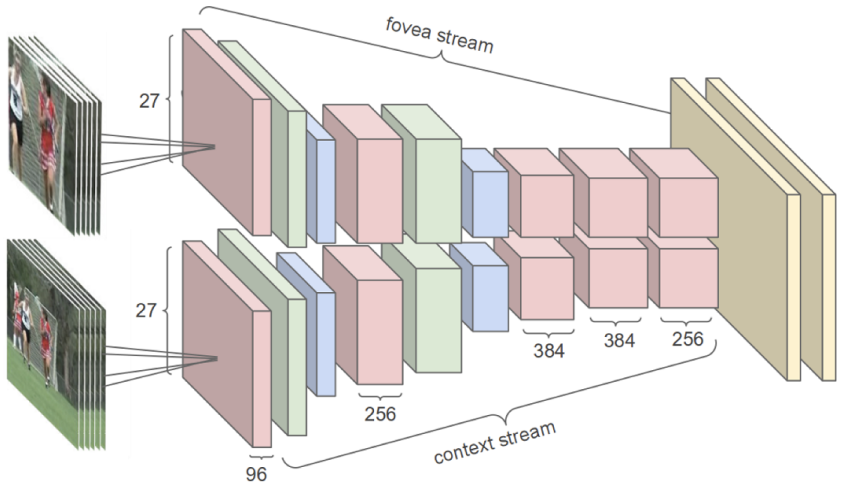

FIGURE 1.16 – Illustration de la méthode à deux flux proposée par Karpathy *et al.* [70]

Karpathy *et al.* [70] proposent d'utiliser un réseau neuronal convolutif 3D à deux flux qui prennent chacun en entrée la même vidéo à des résolutions différentes, l'un pour se concentrer sur l'action effectuée, l'autre contenant la totalité du contexte de la vidéo (voir Figure 1.16).

Molchanov *et al.* [71] implémentent une solution similaire cette fois-ci pour la reconnaissance des gestes de la main, notamment en vue à la première personne, accompagnée de techniques d'augmentation du jeu de données. La même équipe développe une nouvelle méthode plus spécialisée, le R3DCNN [32] (voir Figure 1.17). Le principe est de fragmenter l'entièreté de la vidéo en clips d'images consécutives sans chevauchement. Chacun de ces clips est envoyé à un même enchaînement de couches de convolution 3D. Les sorties de ce CNN 3D appliquées à chacun des clips sont ensuite envoyées à un RNN qui va analyser plus en profondeur la dimension temporelle de la vidéo avant d'aboutir à une classification finale de la vidéo originale.

A l'instar d'autres méthodes vues précédemment, Carreira *et al.* [72] utilisent un réseau à deux flux de convolutions 3D, un prenant les données RGB en entrée, l'autre prenant le flux optique de la même vidéo les deux flux étant entraînés à

part. La particularité étant que, comparés aux autres méthodes, ils n'ont pas besoin d'échantillonner la vidéo et peuvent la passer en entier au réseau.

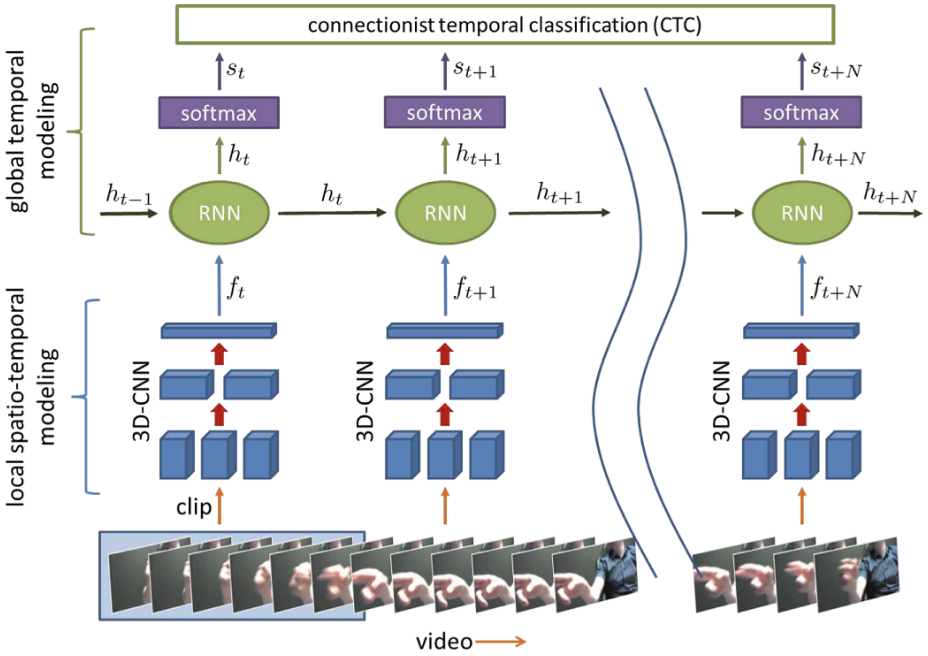

FIGURE 1.17 – Méthode R3DCCN de classification de gestes de la main développée par Molchanov *et al.* [32]

Zang *et al.* [73] utilisent cette fois-ci trois flux différents, l'un RGB, l'autre flux optique pur et le dernier flux optique aussi mais en compensant l'éventuel mouvement de caméra. Ils ajoutent en plus à cela un mécanisme d'attention qui va rassembler la sortie des trois flux et réagir au moment où les trois flux se sont activés au même moment afin d'extraire des données de vecteur d'attention pour chacun des flux, données plus précises qui seront ensuite envoyées à une simple couche entièrement connectée pour procéder à la labellisation de la vidéo (voir Figure 1.18).

Hu *et al.* [74] proposent quant à eux des travaux visant à développer une méthode à la taille modeste pour qu'elle puisse être appliquée sur des appareils autonomes et limités sans grande perte de précision en proposant d'utiliser une architecture de CNN 3D dont la profondeur a pu être réduite en utilisant des connections séparées pour ainsi non pas former un réseau purement linéaire dans l'enchaînement de ses

couches mais avec des connections sautées qui vont d'une couche à une couche bien plus profonde et ainsi rentabiliser plus efficacement une profondeur bien moindre.

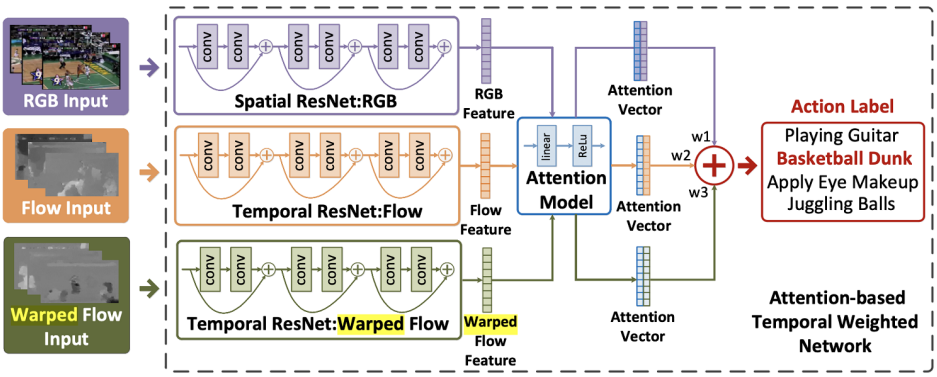

FIGURE 1.18 – Méthode proposée par Zang *et al.* [73] qui vient ajouter un mécanisme d'attention à une architecture CNN 3D à plusieurs flux d'entrée

Une amélioration de l'analyse de la composante temporelle des CNNs 3D est proposée par Yu *et al.* [75] en modifiant directement l'opération de convolution 3D de trois manières différentes. Les trois ajoutent des opérations entre une image t et ses deux images $t-1$ et $t+1$ pour calculer la différence ou la moyenne des régions locales spatio-temporelles extraites de chaque image pour renforcer l'analyse de l'évolution du mouvement.

L'architecture de convolution 3D, malgré quelques rares recherches pour l'alléger, reste néanmoins beaucoup plus lourde que son homologue 2D. De plus, ces deux architectures souffrent structurellement de deux défauts, leur incapacité à l'invariance, si ce n'est à la translation, et leur impossibilité de mettre en relation les différentes caractéristiques spatiales extraites au fil des couches.

Réseau neuronal à graphe

Une architecture que nous pouvons aussi retrouver dans l'état de l'art est celle des réseaux neuronaux à graphe. Tout comme les réseaux neuronaux récurrents dans notre cas d'identification de gestes ou d'activités et d'actions, l'architecture de graphe travaille principalement sur des données de squelettes. L'idée principale

est de regrouper les coordonnées des différents articulations du squelette, éventuellement en série temporelle, sous forme de matrice pour y effectuer des opérations de convolutions. Nous pouvons retrouver l'utilisation de cette architecture pour l'étude des gestes de la main. Zhang *et al.* [76] proposent par exemple d'utiliser des graphes se concentrant sur l'aspect spatial de la main, la relation entre toutes les articulations la composant, couplés à des graphes temporels se concentrant quand à eux sur l'évolution et le déplacement de ces articulations.

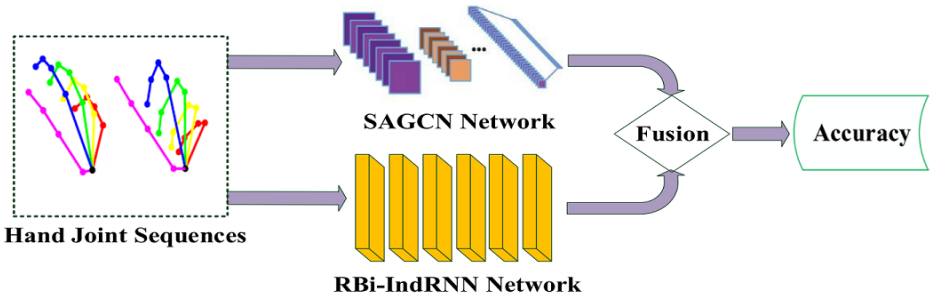

FIGURE 1.19 – Approche proposée par Li *et al.* [77] pour la reconnaissance de geste à partir de squelette de la main et à l'aide de deux branches, l'un pour analyser la composante spatiale, le second pour analyser la temporalité

Li *et al.* [77] développe quant à eux une approche où l'étude de l'aspect temporel du squelette se fait cette fois-ci à l'aide d'un réseau récurrent bi-directionnel tandis que l'analyse de la composante spatiale du squelette se fait à l'aide d'une architecture de graphe en parallèle (voir Figure 1.19). Un mécanisme d'attention est ajouté à l'architecture de graphe afin d'explorer les relations spatiales des articulations entre elles.

Cependant, la main est un objet assez petit dont les relations entre les articulations sont assez directes. Or, ces méthodes ne proposent pas d'approche particulière pour considérer les articulations, leur emplacement ainsi que leur déplacement, les uns par rapport aux autres. A l'inverse, des études plus approfondies en ce sens se retrouvent dans l'étude de la reconnaissance d'action du corps humain. Par exemple,

Yan *et al.* [78] proposent l'utilisation de graphe spatio-temporel qui analyse l'évolution de la position d'une articulation avec celles de ses voisines pour déterminer l'action effectuée par un corps humain (voir Figure 1.20). La méthode présente aussi différentes façons de considérer les relations entre les articulations du squelette. Une première en les considérant toutes de la même manière, une deuxième en considérant la distance entre deux articulations et donc en apportant plus d'attention sur les clusters proches. Enfin, une troisième en considérant aussi la distance avec le centre de gravité du corps humain pour s'intéresser plus précisément aux articulations présentes aux extrémités car ce sont celles qui se déplacent le plus.

Une autre approche est celle de Si *et al.* [79] qui propose d'étudier à l'aide de graphe convolutif, et indépendamment les unes des autres, différentes parties du corps (bras, jambes et buste). Les résultats sont regroupés afin d'étudier leur mouvement ainsi que leur position dans deux branches parallèles contenant des couches LSTM.

Li *et al.* [80], afin de complexifier la méthode et donner l'opportunité à l'architecture d'encore mieux appréhender les relations structurelles entre les articulations du squelette, proposent d'utiliser en amont un auto-encodeur afin de déterminer les meilleures relations en fonction des actions effectuées. Cet auto-encodeur permettra par exemple de déterminer une forte relation entre les articulations de la main ainsi que celles des pieds pour l'action de la marche.

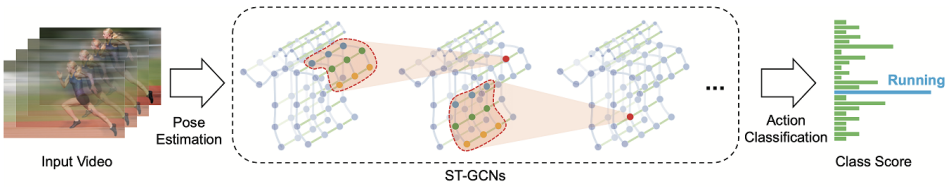

FIGURE 1.20 – Méthode de graphe spatio-temporel proposée par Yan *et al.* [78] pour de l'identification d'actions

Cependant, si les réseaux neuronaux à graphes possèdent une qualité propre à la convolution, à savoir la légèreté de l'opération et donc de l'architecture produite, ils

ont le même défaut que les réseaux neuronaux récurrents, à savoir qu'ils fonctionnent à partir de données squelettes qui restent encore difficiles à extraire.

Réseau neuronal à capsules

Pour remédier à ces problèmes, Sabour *et al.* [81] développent en 2017 une évolution de l'architecture convolutive, le réseau neuronal à capsules [81]. Leur architecture a pour but de réunir les opérations de convolutions d'une même couche en vecteurs qui vont capturer ensemble une caractéristique spatiale et ses possibles transformations (rotation, translation, taille et autre). A cela ils ajoutent l'algorithme de routing par accord qui permet de relier les capsules d'une même couche, qui s'activent aux mêmes données, aux mêmes capsules de la couche suivante afin de mieux lier les représentations spatiales apprises (voir Chapitre 2.6).

Même si l'architecture est récente, peu de travaux l'exploitant sont finalement trouvables dans l'état de l'art, notamment dans la reconnaissance de gestes et d'actions. Srivastava *et al.* [82] proposent d'utiliser un réseau neuronal à capsules pour l'analyse de textes et de commentaires vidéos en lui envoyant les sorties de couches LSTM qui elles, reçoivent les données textes en entrée. Le procédé inverse est appliqué par Ma *et al.* [83], qui, pour le problème d'analyse de trafic routier, utilisent en amont les capsules pour analyser les images de trafic avant d'envoyer leur sortie à des couches récurrentes spécialement développées pour recevoir en entrée des sorties de capsules.

Un des rares travail de recherche exploitant les capsules pour la reconnaissance d'action se retrouve dans les travaux de Duarte *et al.* [84] qui proposent, à l'instar de l'architecture de convolution 3D, de développer les capsules 3D à des fins d'analyse de vidéos. Cependant, leur architecture est conçue plutôt à des fins de segmentation, avec une branche de reconstruction qui a pour but de reconstruire l'image d'origine en y appliquant un masque permettant d'identifier où se situe l'action reconnue, plutôt que de classification (voir Figure 1.21). Ils ont par la suite approfondi leurs travaux et développé les TextCapsules [85] qui permettent de prendre en compte,

toujours pour de la segmentation d'actions, une phrase en entrée décrivant brièvement la vidéo à étudier.

L'architecture de capsule est prometteuse, et malgré un manque de travaux de recherches permettant de l'exploiter pleinement, les problèmes qu'elle résout vis à vis des limitations intrinsèques à la structure des opérations de convolution 2D, offrent des opportunités qu'il serait intéressant d'explorer, notamment pour la reconnaissance de gestes et d'actions.

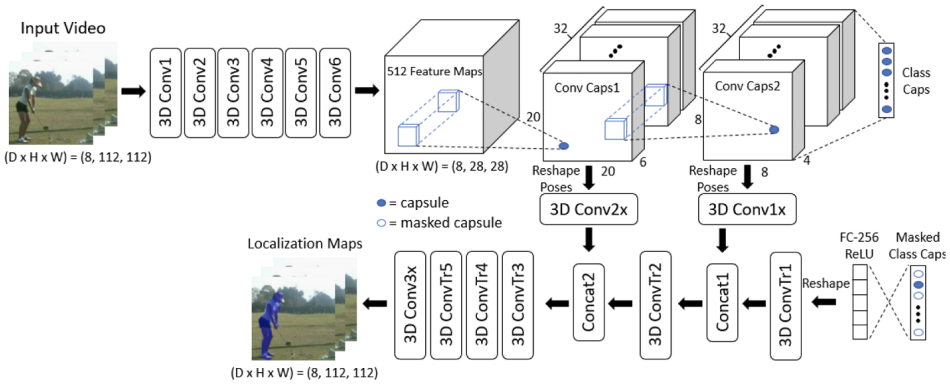

FIGURE 1.21 – Méthode de capsules 3D proposée par Duarte *et al.* [84] pour de la segmentation et classification de vidéo d'actions du corps humain

1.4 Applications pour l'interaction homme-machine et casque RA

Concernant la partie applicative, notamment en réalité augmentée, nous pouvons retrouver différents travaux qui essaient de proposer des méthodes d'interactions utilisant cette récente technologie. Nous pouvons remarquer que les premiers travaux accompagnent la sortie des premières tablettes grand publique au début des années 2000. Ainsi, en 2004, Zhu *et al.* [86] proposent par exemple un système d'interaction personnalisée pour les utilisateurs de e-commerce. La reconnaissance d'objets n'étant pas encore maîtrisée à cette époque, la position de la tablette de chaque utilisateur est traquée et en fonction de sa position et de son orientation vis-à-vis d'un produit

dans une vitrine, différentes informations sont affichées autour de l'objet sur le retour vidéo de la tablette de l'utilisateur qui peut ainsi interagir pour zoomer ou faire défiler les informations. Une proposition similaire est faite par Vinh *et al.* [87] qui développent un système d'interaction RA qui, à l'aide de la caméra d'un smartphone ou d'une tablette, peut reconnaître des marqueurs semblables à de petits QR codes permettant ainsi, à partir de données stockées dans une base de données externe, d'entourer l'objet reconnu à partir de son marqueur associé et d'ajouter une couche d'informations à côté que l'utilisateur pourra observer via son écran.

FIGURE 1.22 – Vision du projet ADAMAAS de lunettes intelligentes pour des interactions en réalité augmentée proposée par Essig *et al.* [88]

C'est à partir de 2013 que l'on peut observer les premiers travaux d'interaction RA à partir de reconnaissance de gestes, très simples au début, Kane *et al.* [89] utilisent une caméra externe reliée à un ordinateur et surplombant une carte en papier de l'Europe pour développer une application permettant à l'utilisateur de pouvoir "cliquer", avec le bout de son doigt, sur la carte pour que l'ordinateur prononce le nom du pays touché, pareil pour la couleur la plus proche du doigt. La solution intègre aussi une reconnaissance vocale qui reconnaît des mots simples où encore la

possibilité d'afficher la liste des mots reconnus par la caméra sur le document apposé devant elle.

Il faut attendre 2016 pour que les premières applications de reconnaissance de gestes et d'actions complexes ne soit recherchées. Essig *et al.* [88] développent ADAMAAS une paire de lunette intelligente en réalité augmentée et équipée d'une caméra frontale RGB et de profondeur ainsi que d'un système de suivi oculaire (voir Figure 1.22). Leur vision sur le long terme est de proposer un système équipé d'une reconnaissance de gestes et d'actions et de créer une analyse de la représentation mentale de l'utilisateur pour pouvoir lui afficher des retours pertinents sur l'action qu'il est en train de pratiquer directement sur son champ de vision grâce aux écrans transparents des lunettes. Schröder *et al.* [90] proposent un système similaire où, à partir des données RGB et de profondeur extraites des capteurs de lunettes de réalité augmentée, développent une architecture d'apprentissage profond permettant de reconnaître les actions effectuées par l'utilisateur pour une tâche précise et connue en amont du réseau et ainsi pouvoir afficher des retours sur la suite des actions ou encore les corrections à effectuer sur le champ de vision de l'utilisateur (voir Figure 1.23). Le réseau proposé est découpé en deux parties. Une première analyse la partie interaction en segmentant la position des mains et de l'objet manipulé. La seconde analyse les flux d'images entier pour avoir le contexte nécessaire à l'identification de l'action. Le problème ici étant que l'architecture qu'ils développent ne s'exécute pas directement sur les lunettes de réalité augmentée utilisées mais sur un ordinateur externe équipé d'une bonne configuration, elle n'est donc pas autonome.

D'un point de vue plus concret en terme d'applications, nous pouvons récemment retrouver d'autres travaux de recherches. Galea *et al.* [91] proposent par exemple d'utiliser les technologies d'AR et de reconnaissance de gestes afin d'aider des personnes victimes de déficiences visuelles de pouvoir suivre les indications d'un chef d'orchestre au sein d'une chorale. La partie réalité augmentée est limitée à des retours haptiques, via les vibrations d'un smartphone ou d'indications orales via un écouteur. Pour le suivi de gestes du chef d'orchestre, deux caméras Leap Motion sont

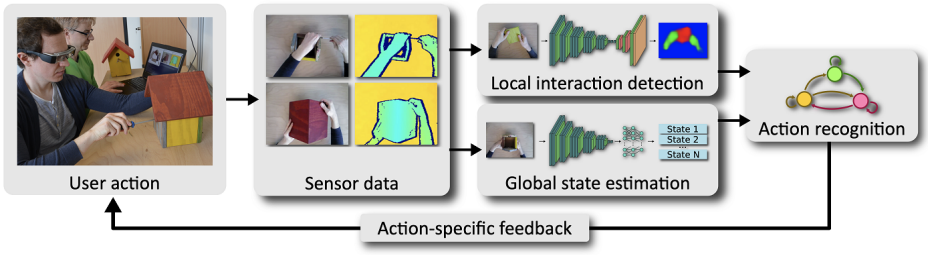

FIGURE 1.23 – Système de reconnaissance d'actions pour retour contextuelle en fonction de l'action effectuée reconnue proposée par Schröder *et al.* [90] en AR

placées sur une estrade sous les bras du chef afin de pouvoir déterminer facilement les indications produites par les mouvements effectués par chacun de bras et mains de la personne.

1.4.1 Traitement en temps réel

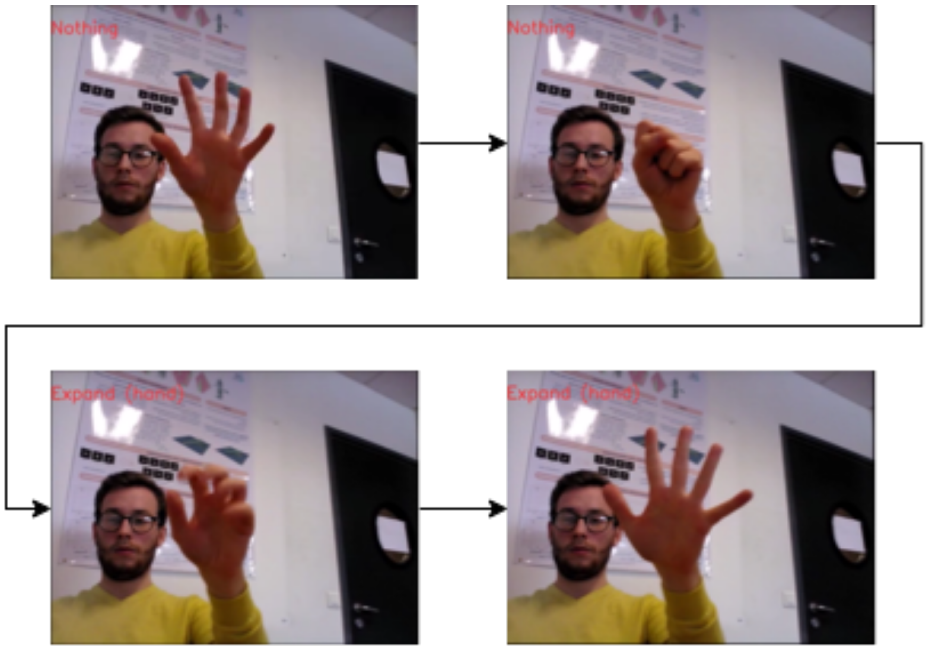

FIGURE 1.24 – Reconnaissance de geste en temps réel. Démonstration de notre application en temps réel sur la base ODHG à l'aide de la méthode RNN de De Smedt *et al.* [33]. L'approche développée permet de reconnaître un geste avant la fin du mouvement répondant aux exigences des applications en temps réel

Avant de conclure cet état de l'art, abordons un dernier point assez spécifique à la volonté de réaliser une application, il s'agit de l'utilisation et de l'application de méthodes en temps réel. En effet, dans la plupart des travaux, les méthodes proposées sont très souvent entraînées et testées hors ligne, c'est-à-dire que l'entièreté, ou tout du moins une segmentation de la séquence est analysée par la méthode et une classe unique lui est attribuée *in fine*. Cependant, dans des cas d'applications en temps réel, il est souvent nécessaire, par exemple pour de l'interaction homme-machine ainsi que de la manipulation d'interface, que la méthode puisse prendre la vidéo au fur et à mesure et ainsi proposer continuellement une classification sur chaque image récupérée pour pouvoir anticiper la reconnaissance et proposer une application réactive. Comme nous l'avions vu, les réseaux neuronaux récurrents peuvent naturellement être employés pour ce genre d'application car, et ce même durant l'entraînement, il s'agit d'une architecture qui retourne une classification pour chaque donnée de la séquence qui lui est envoyée. Pour illustrer, nous avons développé au début de nos travaux de recherches une démonstration basée sur les gestes appris sur la base ODHG ainsi qu'à l'aide de la méthode développée par De Smedt *et al.* basée sur un RNN [33]. Grâce à la caméra Intel Real Sense et à la légèreté de la méthode, il nous est possible de traiter en temps réel le squelette de la main extrait par la caméra sur un ordinateur portable à la configuration modeste. L'idée de l'application est de déterminer pour chaque geste, notamment car ils ont un nombre d'images moyen différent, à partir de combien d'images consécutives classifiées au label d'un même geste, nous pouvons être le plus sûr de pouvoir s'assurer que le geste est effectué même s'il n'est pas fini et pour ainsi pouvoir anticiper l'interaction et la réaction de l'ordinateur. Pour cela, nous nous sommes aidés des courbes ROC [92] et du temps normalisé pour détecter ou Normalized Time to Detect en anglais (NTtD) [93], le but étant, à partir des données gestuelles de la base d'entraînement, de trouver le seuil de la courbe ROC de chaque geste permettant de minimiser leur NTtD, à savoir de pouvoir les détecter le plus tôt possible. Une fois ce seuil trouvé pour chacun des gestes, il nous suffit en temps réel de compter le nombre d'images consécutives étant classifiées sous le même label et au dessus du seuil déterminé pour ensuite

valider le geste en cours et, dans notre cas, afficher le geste reconnu selon le nombre d'images restantes que prend en moyenne le geste à être effectué dans les données d'entraînement (voir Figure 1.24).

D'autres méthodes peuvent être trouvées, par exemple Köpüklü *et al.* [94] proposent une méthode similaire mais en utilisant un réseau neuronal convolutif. La séquence vidéo à classifier est segmentée en des clips de quelques images qui sont traités un à un par le CNN pour déterminer à partir de quel moment l'on peut s'assurer de la classification sur une segmentation plus grande. Une fois ce clip atteint, il sert de point de départ à l'extraction d'un clip plus grand qui servira dès lors à la classification finale de la séquence (voir Figure 1.25).

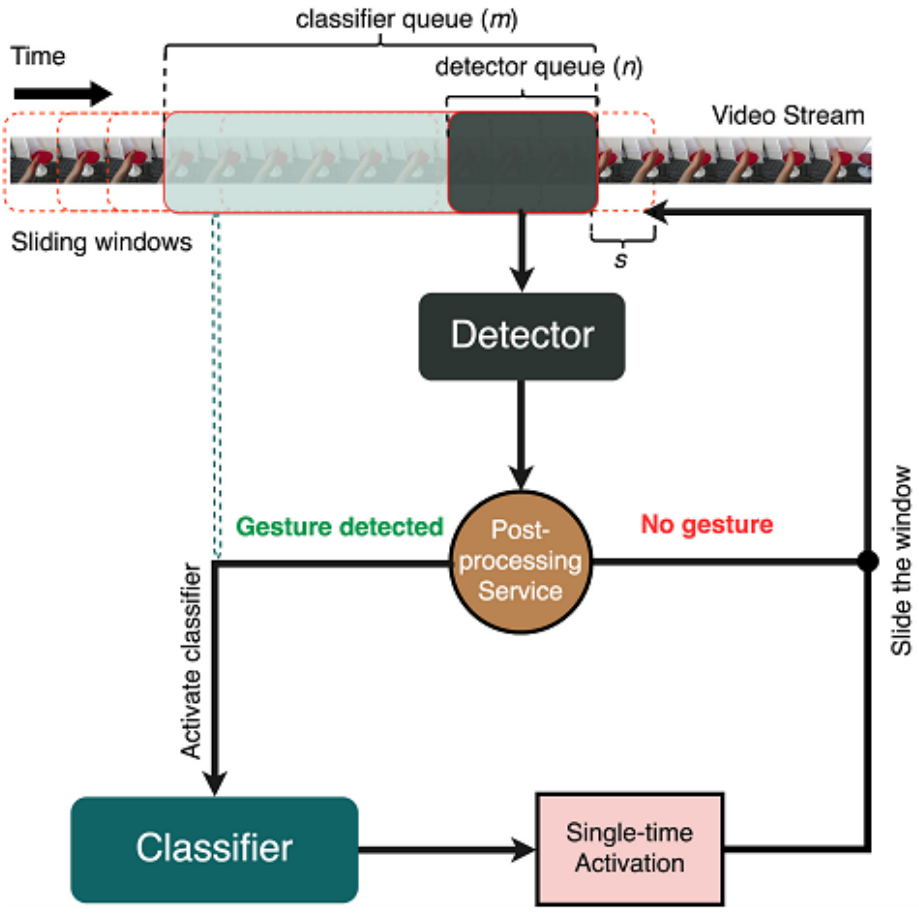

FIGURE 1.25 – Proposition de détection de gestes anticipée à partir de CNN par Köpüklü *et al.* [94]

1.5 Conclusion

L'arrivée des capteurs de couleurs et de profondeur bon marché a motivé la recherche à se confronter aux problèmes de reconnaissance de gestes et d'actions. Si les méthodes d'apprentissage profond se sont rapidement imposées comme les solutions les plus performantes et robustes à la résolution de ces défis, la grande majorité cherche encore à améliorer sans cesse la qualité de classification sans prendre en compte le côté applicatif que l'utilisation de ces reconnaissances pourrait permettre. En effet, leur utilisation en temps réel, souvent avec une nécessité de gérer un flux à 24 images par secondes et plus pour obtenir une information temporelle suffisante, demande de développer une solution au coût de calcul modeste. Cela est d'autant plus vrai lorsque l'application en temps réel ne doit pas se faire sur une machine externe puissante mais directement sur un appareil autonome comme un casque ou des lunettes de réalité augmentée. Or, très peu de recherches s'efforcent de trouver des méthodes qui certes, n'offrent pas une grande amélioration de la qualité de classification, mais qui promettent des temps de calcul raisonnables pour une inférence voire même juste qui pourraient être chargées sur un GPU limité en espace mémoire. C'est pour cela que nous pensons qu'il serait intéressant de proposer une solution répondant à ces limitations, notamment en utilisant la récente puissance des réseaux neuronaux à capsules qui permettent d'obtenir des résultats similaires à ceux d'un réseau convolutif 2D tout en ayant une profondeur et donc un nombre de paramètres bien moindre. La structure particulière des capsules permet aussi de reconnaître des motifs spatiaux complexes ainsi que d'analyser comment elles peuvent être mises en relation les unes par rapport aux autres. De plus, nous pouvons remarquer, que ce soit dans les bases de données comme dans le développement de méthodes proposées, qu'il y a très peu de travaux ayant pour objectif de proposer un exemple concret d'application de réalité augmentée exploitant la reconnaissance d'actions. Nous pouvons notamment le remarquer car aucune base de données existante ne permet de développer une application intéressante puisqu'elle ne sont pas assez focalisée sur un objectif et un contexte précis et limité. Ces bases contiennent souvent un ensemble

de gestes et d'actions sans forcément de lien les uns par rapport aux autres et sans réelle substance applicative. C'est pourquoi nous pensons qu'il serait aussi intéressant de proposer une première base de données qui serait principalement orientée vers ce genre d'applications.

Apprentissage profond et vision par ordinateur

2.1 Introduction

Depuis plusieurs années, le domaine de l'apprentissage automatique et plus précisément son sous-domaine, l'apprentissage profond, réalise les meilleurs résultats dans de nombreux champs de recherche, notamment de la vision par ordinateur. Ces méthodes datant du début des années 40 et inspirées par le fonctionnement du neurone biologique [95] sont dites basées sur les données, c'est-à-dire que leur principale source d'apprentissage provient d'un grand nombre de données (images, vidéos, textes, etc...) afin d'inférer un modèle ayant pour but de classifier, segmenter ou autre, ces données. La popularisation récente de ces méthodes s'explique par l'accumulation de bases de données ainsi que l'augmentation exponentielle de la puissances de calcul des ordinateurs récents qui limitait auparavant fortement la possibilité d'apprentissage de ces méthodes. De plus, les algorithmes d'apprentissage profond ont aussi développé des avantages comparés aux méthodes dites "handcrafted", comme la robustesse à l'encontre de données encore jamais rencontrées durant l'entraînement.

C'est parce que les travaux de ce mémoire se basent principalement sur les méthodes d'apprentissage profond qu'il sera fait, dans ce chapitre, une explication claire

du fonctionnement des architectures et opérations utilisées dans le cadre de ces algorithmes, de la base du fonctionnement du perceptron jusqu'à l'une de ses plus récentes architectures, le réseau neuronal à capsule.

Notons aussi qu'il ne s'agit pas ici de faire un compte rendu exhaustif de tous les types d'architectures ayant été développés mais de se concentrer sur ceux utiles dans le cadre du sujet de ce mémoire. Nous souhaitons que le lecteur puisse avoir une compréhension claire et profonde du fonctionnement de ces algorithmes pour appréhender au mieux la méthode de reconnaissance de gestes et d'actions développée dans le cadre de cette thèse et expliquée plus en avant dans ce mémoire.

2.2 Neurone formel

Le neurone formel (voir Figure 2.1), constitue la base de l'apprentissage profond [95]. Son fonctionnement se découpe en trois parties :

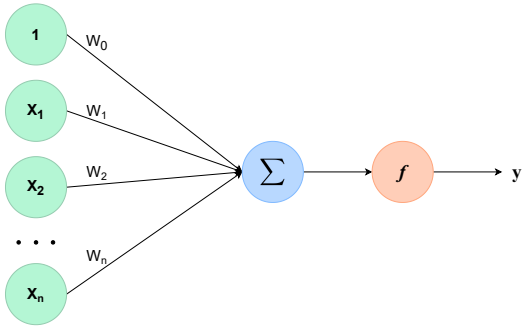

FIGURE 2.1 – Schéma du neurone formel

— **Entrées** : Les entrées constituent les données reçues par le perceptron, souvent normalisées entre 0 et 1. Chaque entrée est associée à un poids afin de les pondérer. Les entrées peuvent aussi contenir une valeur supplémentaire, toujours égale à 1 et nommée *biais* permettant au perceptron de plus grandes opportunités d'apprentissage.

— **Noyau** : Partie centrale du neurone formel, il consiste à effectuer la somme pondérée des entrées avec leur poids associé :

$$w_1x_1 + ... + w_nx_n = \sum_{i=1}^{n} w_ix_i \tag{2.1}$$

— **Sortie** : La sortie est la valeur retournée par le neurone formel. Il s'agit de la valeur finale du noyau passée à une fonction d'activation, souvent :

— Fonction sigmoïde :

$$f_{sig}(x) = \frac{1}{1+e^{-x}} \tag{2.2}$$

— Fonction tangente hyperbolique :

$$tanh(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}} \tag{2.3}$$

— Unité de rectification linéaire (ReLU) :

$$ReLU(x) = \begin{cases} 0 & si \quad x \leq 0 \\ x & si \quad x > 0 \end{cases} \tag{2.4}$$

Ces exemples de fonctions d'activations ont en points communs de ne pas être des polynômes afin de ne pas ralentir le calcul [96] mais aussi d'être dérivables, indispensable pour calculer le gradient de l'erreur commise par le neurone et ainsi permettre son apprentissage.

Le perceptron [97], dérivé du neurone formel mais accompagné d'une règle d'apprentissage, peut être vu comme le premier et plus petit réseau de neurones. Il s'agit d'apprentissage supervisé, à savoir que chaque donnée d'entrée est associée à une valeur que la sortie du perceptron doit essayer d'approcher, sachant que le perceptron agit comme un classifieur binaire. Pour cela, la règle d'apprentissage associée au perceptron vise à modifier la valeur du poids associé à chaque entrée en s'aidant de l'erreur observée en sortie par rapport à la valeur attendue :

$$W_i^{'} = W_i + \alpha * (S - y)X_i \tag{2.5}$$

— W_i : valeur actuelle du poids i

— $W_i^{'}$: valeur corrigée du poids i

— α : pas d'apprentissage

— S : valeur de sortie attendue

— y : valeur de sortie calculée

— X_i : valeur de l'entrée i

Cependant, des travaux ont rapidement mis en lumière les limitations théoriques du perceptron simple, à savoir son incapacité à traiter des problèmes non linéaires. Pour répondre à ce problème, le premier réseau de neurones profond est développé, à savoir le perceptron multicouche [98] (voir Figure 2.2).

Cette nouvelle architecture introduit deux nouveaux concepts, celui de couche, représentant un certain nombre de perceptrons recevant les mêmes entrées et dont les calculs des valeurs de sortie sont effectués en parallèle car indépendant les uns des autres, ainsi que celui de profondeur de réseau, à savoir le nombre de couches cachées que celui-ci contient. D'une couche à l'autre, les perceptrons sont, en général, tous connectés les uns aux autres, mais pas au sein d'une même couche. Il s'agit aussi du premier réseau pouvant effectuer une classification multiple étant donné que le nombre de perceptrons à la couche de sortie peut être multiple.

C'est aussi grâce au développement de la rétropropagation [99] que ces nouveaux réseaux peuvent désormais résoudre des problèmes non linéaires en permettant, de la fin jusqu'au début du réseau, de corriger les poids des différentes couches en fonction de l'erreur calculée en sortie par la fonction d'évaluation. La fonction d'évaluation la plus utilisée est en général l'erreur quadratique moyenne (MSE) mais peut dépendre du problème à résoudre (classification ou segmentation) :

$$MSE(S_i, y_i) = \frac{1}{n}\sum_{i=0}^{n}(y_i - S_i)^2 \tag{2.6}$$

— S_i : valeur attendue en sortie du neurone i

— y_i : valeur calculée en sortie du neurone i

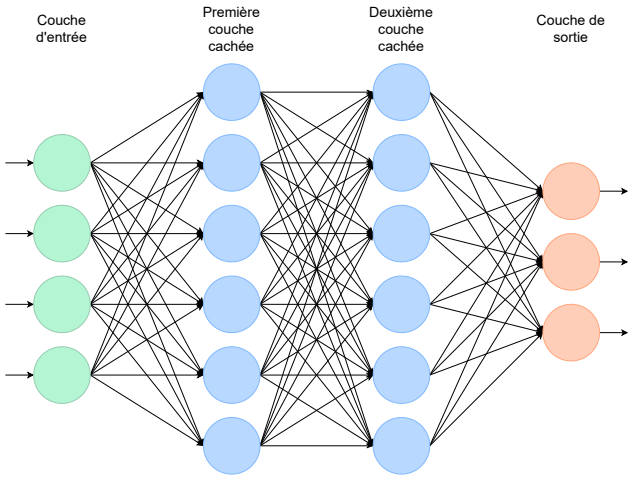

FIGURE 2.2 – Exemple de perceptron multicouche avec 2 couches cachées

L'objectif étant de minimiser le résultat de la fonction d'évaluation, l'algorithme de rétropropagation permet de calculer les modifications de poids de chaque perceptron de chaque couche de la sorte :

$$W_i^{j'} = W_i^j - \alpha * \frac{\partial F}{\partial W_i^j} \tag{2.7}$$

— W_i^j : valeur actuelle du poids i de la couche j

— $W_i^{j'}$: valeur corrigée du poids i de la couche j

— α : pas d'apprentissage

— F : valeur calculée en sortie du réseau par la fonction d'évaluation

Plusieurs itérations en passant plusieurs fois les mêmes données au réseau sont nécessaires pour qu'une convergence des poids s'opère et qu'un bon apprentissage apparaisse. Les poids modifiés dans le réseau au cours de l'apprentissage via la rétropropagation constituent les paramètres du réseau tandis que le reste (nombre de couches cachées, nombre de perceptrons par couche, pas d'apprentissage, fonction d'évaluation, fonction d'activation des perceptrons et autres) constituent les hyper-paramètres du réseau.

Le perceptron multicouche constitue le point d'entrée de l'apprentissage profond

avant que de nouveaux types de réseaux ne soit développés, notamment pour s'acquitter de tâches plus spécifiques comme le traitement et la reconnaissance d'image.

2.3 Augmentation de données et phase d'apprentissage

Avant de passer à des architectures plus complexes et plus spécifiques à certains problèmes, évoquons comment la phase d'apprentissage d'un réseau neuronal, quel qu'il soit, se déroule. Comme nous l'avions déjà précisé, les méthodes d'apprentissage profond sont dites, basées sur les données, et doivent donc étudier un très grand nombre de données différentes afin de faire converger ses poids vers une solution de plus en plus précise sur ces données.

Or, de base, la rétropropagation du gradient ordinaire a été pensée pour être appliquée sur toutes les données disponibles, elles sont toutes passées au travers du réseau, le gradient est calculé puis rétropropagé pour corriger tous les poids. Malheureusement, sur des bases de données de très grande taille, il est impossible de faire converger un réseau dans un temps raisonnable si celui-ci doit étudier toutes les données une par une avant de pouvoir se rectifier. Pour éviter cela, diverses solutions ont été apportées notamment celle la plus utilisée de nos jours, à savoir l'utilisation de mini-lots (ou batch).

La méthode des batchs consiste à regrouper les données de la base en petits paquets qui seront par la suite envoyés un par un au réseau qui effectuera une rétropropagation du gradient à chacun d'entre eux. Ainsi, là où une unique étape de convergence est effectuée dans le cas de la rétropropagation ordinaire, des dizaines voire des centaines sont effectuées dans le cas de la méthode des batchs, cela dépend de la taille de la base ainsi que de la taille des batchs. Les batchs sont regroupés aléatoirement et le tirage est généralement changé à chaque epoch, nom qui est donné à l'étape représentant l'entraînement sur tous les batchs d'une base. Cette notion d'aléatoire permet deux choses, la première est de rassembler des données de classes différentes dans un même batch, et l'autre de diversifier le gradient à chaque epoch puisque la répartition des données dans les lots aura changée. Ainsi, même si sur une epoch et par mégarde, l'aléatoire fait qu'un ensemble de données de même classe

se retrouve dans un même mini-lot, même si au moment de la rétropropagation du gradient de celui-ci, le réseau aura tendance à faire converger ses poids pour favoriser cette classe commune, les chances que cette erreur soit répétée sur plusieurs batchs et sur plusieurs epochs sont tellement minimes que rencontrer quelques uns de ces cas durant l'entraînement devient négligeable. Cette méthode a tout de même quelques défauts notamment celui de pouvoir se retrouver plus facilement bloqué dans un minimum local par rapport aux autres solutions. Enfin, notons qu'à chaque epoch, et ce, généralement, quelle que soit la méthode utilisée, le pas d'apprentissage est légèrement réduit au fur et à mesure. En effet, plus la convergence vers un minimum se fait, et plus le pas de correction doit être petit pour se rapprocher au plus du minimum sans s'en éloigner et risquer de converger vers un minimum local voisin mais moins performant. Cependant, ce pas doit tout de même rester assez grand pour éviter de converger dans un minimum local et de s'y retrouver coincé à cause d'un pas d'apprentissage trop faible.

Évoquons enfin une phase qui s'exécute en amont d'une inférence d'une donnée ou d'un batch, il s'agit de la phase d'augmentation de données. Non obligatoire, notamment sur des bases de données complètes et de grande taille, elle permet dans beaucoup de cas d'ajouter de la diversité aux données et d'améliorer la robustesse du réseau entraîné ainsi que possiblement ses résultats de classification au moment de la phase de test. L'augmentation de données consiste plus précisément à modifier les données d'entrées, par exemple, pour des images, en modifiant leur résolution, en ajoutant du bruit, en effectuant un renversement horizontal ou vertical ou encore en appliquant une légère rotation. Ces modifications sont généralement de l'ordre de l'aléatoire, que ce soit pour savoir quelles modifications appliquer ou non et à quel pourcentage de modification. De plus, ces valeurs et diversifications ne sont pas les mêmes d'une epoch à l'autre ce qui permet au réseau de lui donner une constante impression de rencontrer des données différentes, renforçant son apprentissage et son adaptabilité à des données inconnues en démultipliant le nombre de cas possibles rencontrés pour chaque classe.

Ces deux notions étant communes et indifférenciées à l'ensemble des architectures, nous allons désormais pouvoir les explorer.

2.4 Apprentissage profond et données spatiales

Toujours basée sur le fonctionnement du neurone biologique, la nouvelle architecture de réseau neuronal convolutif (CNN), développé fin des années 90 par LeCun *et al.* [100], permet l'application de méthodes d'apprentissage profond sur des images notamment par l'extraction et la reconnaissance de motifs via des opérations de convolutions. L'architecture générale d'un CNN consiste en l'alternance des deux types de couches :

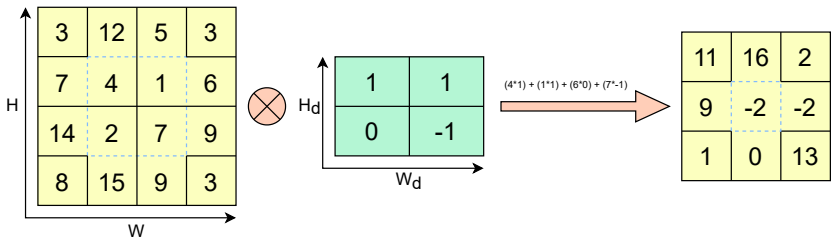

FIGURE 2.3 – Exemple d'une opération de convolution sans padding sur la matrice d'entrée et avec un pas de déplacement de 1

— **Couche de convolution** : idée centrale de ce type de réseau de neurones, ces couches sont composées de filtres de caractéristiques spatiales qui servent à extraire des motifs au sein de l'image passée en entrée. Une couche de convolution est composée de D_n filtres de dimensions $W_d * H_d$ qui effectuent chacun une opération de convolution sur les C entrées de dimensions $W * H$ (voir Figure 2.3). Une opération de padding peut être effectuée en amont sur les entrées en augmentant sa dimensionnalité et en rajoutant des valeurs à zéro de sorte que le résultat de l'opération de convolution ait la même dimensionnalité que l'entrée (avant padding). Il s'agit d'un hyper-paramètre à prendre en compte à l'instar du nombre de filtres, de leur taille et du pas de déplacement lors du calcul de la convolution. L'enchaînement de couches de convolution permet de reconnaître et d'assembler des motifs de plus en plus complexes et haut

niveau. Il s'agit aussi de l'une des rares architectures qui permet de visualiser ce que le réseau produit comme apprentissage et ayant la possibilité d'afficher les filtres intermédiaires et ainsi observer les motifs appris.

— **Couche de pooling** : Cette couche consiste à effectuer une opération de pooling sur la sortie des filtres d'une couche de convolution. Il s'agit une nouvelle fois d'utiliser une fenêtre, de la déplacer selon un certain pas sur l'image et pour chaque position, selon l'opération de pooling, récupérer la valeur la plus haute dans la fenêtre (Max pooling), la moyenne des valeurs (Average pooling) ou autre (voir Figure 2.4). Un padding sur l'entrée peut aussi être nécessaire si la taille de la fenêtre associée à son pas de déplacement ne correspond pas exactement à la dimensionnalité de l'entrée. La principale raison de cette couche, en plus de réduire la taille des données au fur et à mesure de leur application et donc de minimiser le nombre de calculs à effectuer. L'autre raison est de permettre aussi de l'invariance à la translation en généralisant la présence d'un motif à une zone générale et plus large plutôt qu'à un endroit précis de l'image.

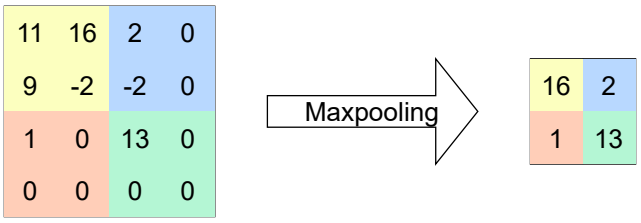

FIGURE 2.4 – Exemple d'une opération de maxpooling avec padding sur la matrice de sortie de la couche de convolution

La fin d'un CNN consiste généralement en l'enchaînement de couches entièrement connectées où la dernière possède un nombre de neurones égal au nombre de classes nécessaires au problème de classification (voir Figure 2.5). On dénote aussi deux caractéristiques sur lesquelles reposent le développement des CNNs :

— **Partage de poids** : contrairement au perceptron multicouche, les neurones d'une même couche d'un CNN partagent tous les mêmes poids. En plus de limiter le nombre de paramètres à corriger durant l'apprentissage, cette particularité permet aussi, à l'instar des couches de pooling, une invariance à la

translation. Ce partage des poids permet en effet de produire des valeurs de sortie similaires pour des entrées où l'objet à reconnaître est identique mais à des positions différentes dans l'image.

— **Connectivité partielle** : de même, contrairement au percreptron multicouche qui est entièrement connecté, l'avantage des opérations de convolution et de pooling qui utilisent une fenêtre glissante est de limiter encore plus le nombre de poids et de réduire le nombre de calculs. En effet, si une image était passée en entrée d'un percreptron multicouche, tous les pixels seraient connectés à tous les neurones de la couche suivante, dans le cas du CNN, un poids est relié à un groupement de pixels voisins.

Malgré la théorie, les CNNs n'ont pas connus immédiatement de succès faute de puissance de calcul. Dans les années 2010, une première implémentation d'apprentissage de réseau neuronal sur GPU fut proposée par Ciresan *et al.* [101], permettant, en 2012, le développement du réseau AlexNet par Krizhevsky *et al.* [15] qui permit une forte amélioration des résultats de classification par rapport aux méthodes hand-crafted sur la base de données ImageNet [13]. Ce réseau déclare le point de départ de la démocratisation de l'utilisation de méthodes d'apprentissage profond dans de nombreux autres domaines. Pour ce qui est toujours du traitement d'image, on peut noter les architectures de VGG [102] qui proposent une architecture extrêmement profonde, ResNet [103] qui utilise notamment des opérations résiduelles, à savoir transférer des informations d'une couche à une autre plus loin dans le réseau plutôt qu'immédiatement la suivante, ou encore Inception [59] qui combine ces deux idées.

Jusqu'ici, les architectures étudiées ne permettent d'étudier et d'apprendre que sur des données simples et statiques, or, l'étude de vidéos ou de tout autre donnée possédant une dimensionnalité temporelle a attirée l'attention de se voir appliquer des méthodes d'apprentissage profond à leur tour. Même s'il est tout à fait possible de passer des données temporelles à un CNN, en regroupant par exemple les données de la suite temporelle sous forme de matrice semblable, formellement, à une image, de

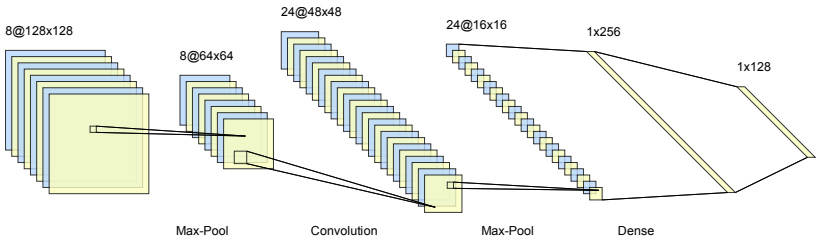

FIGURE 2.5 – Exemple d'un réseau neuronal convolutif

nouveaux *types* de réseaux neuronaux ont cependant été développés pour répondre au mieux à ce nouveau problème.

2.5 Apprentissage profond et données temporelles

Plusieurs solutions pour répondre au problème de l'étude et la compréhension de flux vidéos ont été apportées dans le domaine de l'apprentissage profond, deux des plus populaires et très différentes dans l'approche sont le réseau neuronal récurrent (RNN) et le réseau neuronal convolutif 3D (3D CNN).

2.5.1 Réseau neuronal récurrent

L'architecture de RNN est un modèle conçu pour les séquences temporelles à faible dimensionnalité, comme l'étude de textes, de musiques ou de séquences de squelettes de corps humain. La nomination de récurrence vient de la particularité des neurones à boucler sur eux-mêmes et de se renvoyer l'information d'une donnée du temps t à celle du temps $t + 1$ [104] (voir Figure 2.6).

Le neurone récurrent possède un noyau dit *caché* possédant deux points d'entrées, l'un pour la donnée au temps t et l'autre pour l'activation de ce noyau calculé au temps $t - 1$, et possède deux sorties, une qui sera envoyée à la couche de neurones récurrents suivante, l'autre qui est le résultat du calcul de ce noyau caché au temps

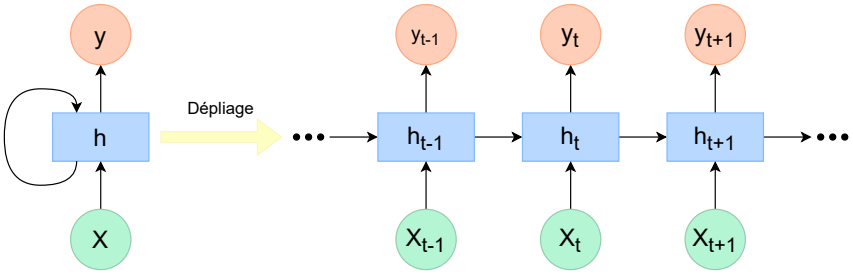

FIGURE 2.6 – Neurone récurrent et son dépliage temporelle

t et qui lui sera renvoyé en entrée au moment du traitement de la donnée associée au temps $t+1$. Le calcul des deux sorties du noyau sont :

$$a^t = g_1(W_{aa} * a^{t-1} + W_{ax} * X^t + b_a) \tag{2.8}$$

$$y^t = g_2(W_{sa} * a^t + b_s) \tag{2.9}$$

— W_{aa}, W_{ax} et W_{sa} : matrices de poids

— b_a et b_s : vecteurs de poids

— g_1 et g_2 : fonctions d'activation

— X^t : entrée du neurone au temps t

— a^t : entrée et sortie du neurone renvoyée à lui-même au temps t

— y^t : sortie du neurone envoyée à la couche suivante au temps t

Notons que les différents paramètres du neurone récurrent W et b sont indépendants du temps et ont donc la même valeur pour toutes les données de la série temporelle (voir Figure 2.7). Cette indépendance est nécessaire dans le cas où les données d'entrées sont des séries temporelles de tailles différentes, peu importe le nombre de dépliages du réseau récurrent. En effet, on ne souhaite pas qu'il y ait un certains nombre de paramètres moins entraînés que d'autres s'il y avait un paramètre associé à la dimension temporelle avec des séquences d'entraînement de tailles différentes. A l'inverse, les calculs de l'erreur et des opérations de rétropropagation sont calculés en fonction du temps, ce qui demande aussi d'avoir une sortie attendue

pour chaque donnée de la série temporelle, mais si elle peut être commune à l'ensemble des donnés de la série si seule la classification de la série dans son ensemble nous intéresse :

$$F(S, y) = \sum_{t=1}^{T_y} F(S_t, y_t) \tag{2.10}$$

$$\frac{\partial F_T}{\partial W} = \sum_{t=1}^{T} \frac{\partial F_T}{\partial W}\bigg|_t \tag{2.11}$$

— S_t : valeur de sortie attendue au temps t

— y_t : valeur de sortie calculée au temps t

— F : valeur calculée en sortie du réseau par la fonction d'évaluation

On remarque ainsi que pour une série temporelle de taille T où chaque élément possède une classe distincte, l'erreur est calculée en sommant la somme des fonctions d'évaluation F de chaque sortie avec leur sortie attendue. La rétropropagation et la correction des poids se fait de la même manière. La correction globale d'un poids, étant donné qu'il est indépendant du temps, se fait par la somme des corrections à chaque temps t.

En plus de traiter activement des données ayant une dimension temporelle en permettant aux calculs des sorties des neurones récurrents de prendre en compte les résultats des données antérieures, les RNNs ont aussi l'avantage de pouvoir prendre en entrée des séries temporelles de n'importe quelle taille, le tout sans que cela n'augmente la taille du réseau, même en terme de nombre de paramètres à apprendre. Il s'agit aussi d'une des rares architectures qui peut analyser des données en temps réel, simplement en sauvegardant les sorties d'activations des cellules au temps t pour les charger au moment de l'inférence de l'information reçue au temps $t + 1$. Cependant, le noyau d'un neurone récurrent effectue beaucoup plus d'opérations qu'un perceptron, et sans partage de paramètres comme dans le cas des CNNs, ce qui induit des temps de calcul relativement longs et coûteux. Dans sa forme de base, le RNN a aussi du mal à prendre en compte des informations d'un passé lointain et ne peux pas prendre en compte des données futures.

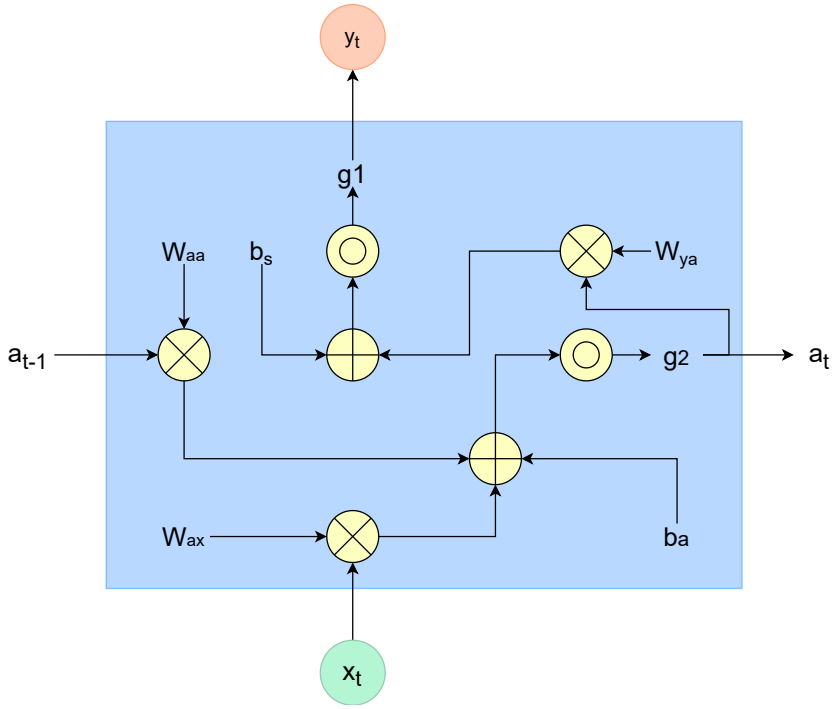

FIGURE 2.7 – Fonctionnement détaillé d'un neurone récurrent

Pour répondre à ces problèmes, des alternatives ont été développées, que ce soit au sein des noyaux des neurones récurrents, comme dans les cellules Long Short Term Memory (LSTM) [105] ou les cellules Gated Recurrent Unit (GRU) [106] qui visent à ajouter des paramètres capturant des informations comme la pertinence des données reçues à un temps t ou encore la nécessité d'oublier l'information accumulée jusqu'à ce temps. Ces différentes architectures peuvent aussi être implémentées dans un réseau neuronal récurrent bi-directionnel [107] permettant cette fois-ci d'avoir des sorties d'activation qui envoient l'information analysée d'un temps t à la fois au temps $t + 1$ et au temps $t - 1$, enlevant cependant la possibilité d'utiliser ce genre d'architecture dans des cas en temps réel, puisqu'il est impossible d'avoir accès directement à des données futures.

2.5.2 Réseau neuronal convolutif 3D

Une autre architecture pour étudier les séries temporelles a aussi été développée, il s'agit d'une évolution du CNN à savoir le réseau neuronal convolutif 3D (3D CNN)

[69]. Tout à fait similaire au CNN dans sa structure, le principale changement s'effectue au niveau de l'opération de convolution. En effet, dans ce type d'architecture, les données n'était plus des images mais des séquences d'images, une donnée d'entrée d'un 3D CNN est de dimensionnalité : $t*w*h*c$ (t nombre d'images dans la séquence, $w*h$ dimensions des images et c le nombre de canaux de chaque image). Comme pour le CNN, le nombre de canaux en entrée d'une couche équivaut aux nombre de filtres de sorties de la couche de convolution précédentes, les calculs de convolution étant effectués en parallèle sur chacun de ces canaux, ou quatrième dimension, c'est donc bien uniquement sur les trois premières dimensions que s'effectue l'opération de convolution 3D (voir Figure 2.8).

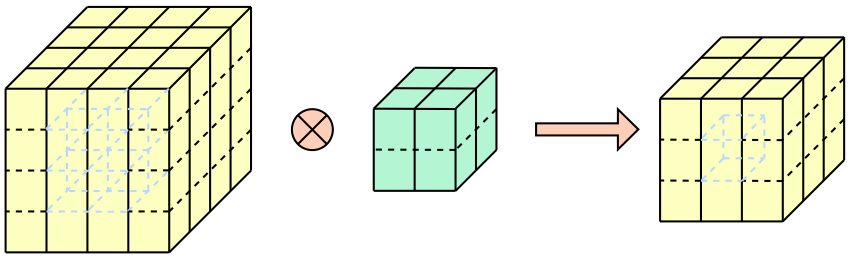

FIGURE 2.8 – Exemple d'opération de convolution 3D

La nouvelle dimension des entrées étant la temporalité, l'opération de convolution 3D, en plus de reconnaître des motifs spatiaux va aussi essayer de reconnaître et d'apprendre leur évolution, modification, translation, rotation, ou autre, au cours du temps. L'opération de pooling 3D est aussi équivalente à celle présente dans le 2D CNN mais étendue à la troisième dimension. L'architecture du 3D CNN est aussi similaire, à savoir une alternance de couches de convolution 3D et de pooling 3D finissant par un enchaînement de couches entièrement connectées pour la classification.

L'architecture 3D CNN est naturellement plus lourde que celle d'un 2D CNN, de par la taille des données passées en entrée, du nombre de paramètres à gérer ou encore de la plus grande complexité de l'opération de convolution 3D. Elle reste néanmoins plus rapide à entraîner qu'une architecture de RNN à profondeur égale car moins d'opérations à effectuer dans la convolution que dans un noyau récurrent. Cependant,

si les données envoyées à un RNN sont de faible dimensionnalité, un 3D CNN reçoit quant à lui des flux vidéos, ce qui provoque souvent une limitation technique réelle si nous souhaitons envoyer une vidéo complète à ce type de réseau. Effectivement, cette vidéo n'est pas envoyée image par image au réseau mais est regroupée sous forme de matrice pouvant ainsi être de dimension immense en fonction de la dimension $W * H$ des images ainsi que de la longueur de la vidéo. Il est donc très souvent nécessaire d'effectuer, en plus d'un redimensionnement des images, un échantillonage à pas régulier, ou non, et ainsi se passer de plusieurs images pouvant contenir de précieuses informations. De plus, le fait de devoir passer toute la vidéo échantillonnée d'un coup au réseau empêche aussi son utilisation dans des cas de traitements en temps réel car impossible d'envoyer la vidéo frame par frame au réseau et d'obtenir des retours progressivement comme pourrait le faire un RNN.

Précisons aussi que nous avons décrit ces différentes architectures principalement dans les cas de traitement d'images ou de vidéos mais elles peuvent recevoir en entrée n'importe quel type de données ou d'informations. Il s'agit généralement juste d'un prétraitement et d'une mise en forme de ces données pour être acceptées par la première couche de l'architecture, il est ainsi tout à fait possible, par exemple, de mettre sous forme de matrice une série temporelle de graphes de squelette humain pour l'envoyer à un 3D CNN.

Si du CNN au 3D CNN, l'architecture fondamentale n'a pas beaucoup évoluée et consiste à traiter les flux d'images de la même manière, ce n'est pas le cas du dernier type de réseau que nous allons décrire.

2.6 Réseau neuronal à capsule

Derniers travaux majeurs dans la conception fondamentale d'un nouveau type de réseau, Sabour *et al.* [81] développent le réseau neuronal à capsules (CapsNet). Cette architecture vise à améliorer celle du CNN et à corriger ses défauts. En effet, si nous avions précisé l'invariance à la translation de cette architecture, ce n'est pas le cas, par exemple, pour la rotation sans augmentation de données. Un motif reconnu dans un sens ne le sera pas dans un autre à moins que ce ne soit à l'aide d'un autre filtre.

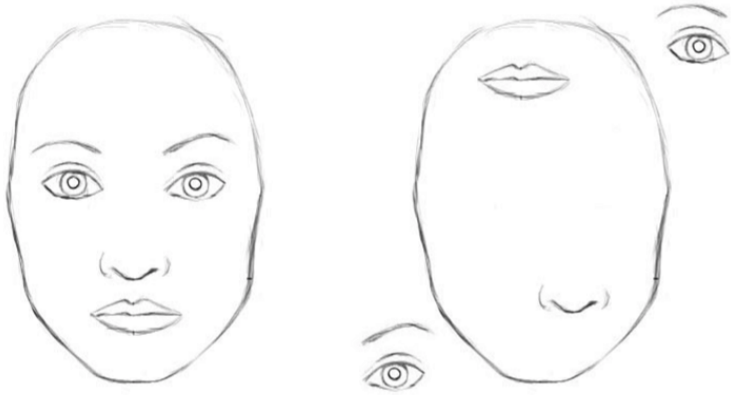

FIGURE 2.9 – A l'inverse d'un CapsNet, un CNN classique classifie ces deux images comme étant un visage

De plus, pour reconnaître une classe, seule l'activation des filtres nécessaires à cette reconnaissance est prise en compte et non la relation qu'ils entretiennent les uns par rapport aux autres. Par exemple, afin de reconnaître un visage, il sera probable que les filtres de haut niveau d'un CNN reconnaissent, pour l'un un oeil, l'autre un nez et une dernier une bouche, mais à aucun moment le CNN n'apprendra que les deux yeux reconnus doivent se trouver, sur l'image, au dessus du nez reconnu, lui-même au dessus de la bouche (voir Figure 2.9).

Pour répondre à ce problème, le réseau neuronal à capsules propose, structurellement au sein d'une couche de convolution, de regrouper les différents filtres de convolution en capsules. En effet, en supposant qu'une couche entraîne $n * m$ filtres, ceux-ci vont être regroupés en n capsules de m convolutions chacune (voir Figure 2.10). Ainsi, et contrairement à une couche standard de convolution qui a pour sortie un ensemble de scalaires, la sortie d'une couche de capsules est un ensemble de vecteurs. Là où la présence d'un motif était justifiée par une valeur plus haute pour le scalaire, l'activation d'un vecteur se décrit par sa norme ($||\vec{s}||$). Les scalaires composant le vecteur vont quant à eux décrire divers attributs du motif associé, par exemple un scalaire pour l'emplacement, un autre pour la taille ou pour l'orientation et autres caractéristiques.

Le noyau d'un neurone de couche de capsules procède à des calculs légèrement différents de ceux d'un noyau standard. Seule la première opération est similaire, à

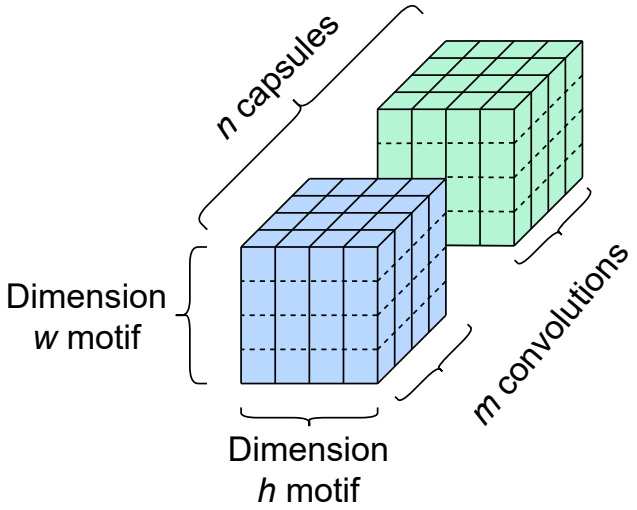

FIGURE 2.10 – Couche de capsules détaillée

savoir la multiplication des valeurs d'entrées avec les matrices de poids entraînables de la couche, à la différence prêt que les entrées sont des vecteurs et non des scalaires.

$$\widehat{u}_{j|i} = W_{ij} * u_i \tag{2.12}$$

— u_i : vecteur i d'entrée

— W_{ij} : matrice de poids entre l'entrée i et la capsule j

La deuxième phase est très particulière au réseau neuronal à capsules et constitue la deuxième idée centrale de cette architecture, le routage dynamique [81]. L'idée derrière cet algorithme est toujours de mettre en relation les différents motifs reconnus par une couche de capsules bas niveau en liant ceux en communs dans les capsules de la couche plus haut niveau. Pour cela, un nouveau poids entraînable est ajouté qui aura pour fonction de savoir quelle quantité d'informations laisser passer d'une capsule à une autre. Ces poids sont entraînés par l'algorithme de routage dynamique de sorte que les motifs ayant une activation similaire sur la donnée d'entrée soient transférés et liés à la même capsule suivante. L'algorithme de routage dynamique remplace la rétropropagation traditionnelle au moment de l'entraînement et seulement au niveau des couches de capsules, et il a aussi pour but de remplacer

la traditionnelle couche de pooling sans la perte d'informations provoquée par le redimensionnement des données.

Algorithm 1 Algorithme de routage dynamique

procedure ROUTAGE($\widehat{u}_{j|i}, r, l$)
∀ capsules i dans couche l et ∀ capsules j dans couche $(l+1)$: $b_{ij} \leftarrow 0$.
for r itérations **do**
∀ capsules i dans couche l : $c_i \leftarrow softmax(b_{ij})$
∀ capsules j dans couche $(l+1)$: $s_j \leftarrow \sum_i (c_{ij} * \widehat{u}_{j|i})$
∀ capsules j dans couche $(l+1)$: $v_j \leftarrow squash(s_j)$
∀ capsules i dans couche l et ∀ capsules j dans couche $(l+1)$: $b_{ij} \leftarrow b_{ij} + \widehat{u}_{j|i} \cdot v_j$
return v_j

$$softmax(b_{ij}) = \frac{exp(b_{ij})}{\sum_k exp(b_{ik})} \tag{2.13}$$

On remarquera que cet algorithme est calculé sur r itérations, cela permet aux coefficients c_{ij} de mieux configurer et lier les capsules. Cependant, un trop grand nombre d'itérations a tout de même tendance à conduire à un surapprentissage sur les données d'entraînement, il est en général plutôt recommandé d'avoir 3 itérations. Enfin, étant donné que les valeurs d'une capsule sont des vecteurs et non des scalaires, les fonctions d'activation habituelles ne peuvent pas être appliquées telles quelles. Le dernier apport de cette nouvelle architecture est donc la fonction d'activation squash [81] qui a aussi pour fonction de faire en sorte que la longueur du vecteur soit comprise entre 0 et 1 pour éviter une explosion et divergence de ces valeurs au cours de l'entraînement.

$$squash(s_j) = \frac{||s_j||^2}{1 + ||s_j||^2} * \frac{s_j}{||s_j||} \tag{2.14}$$

Dans sa globalité, l'architecture de réseau neuronal à capsules se comporte comme un auto-encodeur et est donc composée de deux phases. La première, l'encodeur (voir Figure 2.11), commence par une ou plusieurs couches de convolution 2D afin d'extraire les motifs basiques mais aussi de réduire la dimensionnalité de l'entrée. Elles sont suivies par une ou plusieurs couches de PrimaryCaps qui sont des couches de

capsules effectuant des opérations de convolutions et qui ont pour but de combiner les motifs extraits. La dernière couche de capsule, DigitCaps, agit comme une couche entièrement connectée et possède n vecteurs pour n classes à reconnaître par le réseau, le vecteur i ayant la plus grande norme indique que l'entrée a été classifiée à la classe correspondante i.

En ce qui concerne l'entraînement, la fonction d'évaluation est elle aussi découpée en deux parties, la première dans le cas où la classification est bonne mais où la probabilité de classification est tout de même inférieure à une valeur arbitraire fixée m^+ (généralement 0.9). L'autre dans le cas où la mauvaise capsule a été choisie et avec un probabilité supérieure à m^- (généralement 0.1).

$$\mathcal{F}_k = T_k * max(0, m^+ - ||v_k||)^2 + \alpha * (1 - T_k) * max(0, ||v_k|| - m^-)^2 \tag{2.15}$$

— T_k : 1 si bonne classification, 0 sinon

— v_k : vecteur de la capsule associée à la classification

— α : pas d'apprentissage

— $\mathcal{F}$: valeur calculée en sortie du réseau par la fonction d'évaluation

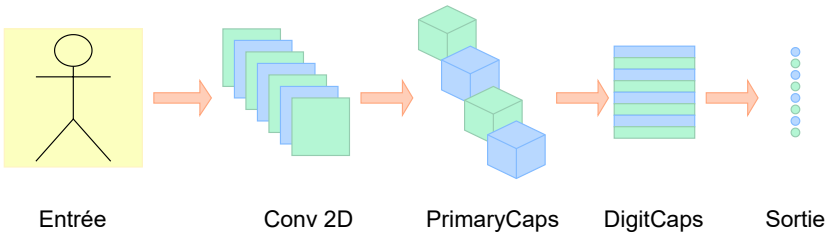

FIGURE 2.11 – Exemple d'architecture capsule network - partie classification

La deuxième partie du réseau fait office de décodeur et a pour but de reconstruire l'image d'entrée uniquement à partir du vecteur de la capsule choisie de la couche DigitCaps (voir Figure 2.12). Durant l'entraînement, étant donné que l'on connaît la classe, même si la capsule associée n'a pas la plus grande norme et n'aura pas été choisie pour la classification, c'est tout de même son vecteur que l'on prendra

pour le décodeur. Dans le cas d'une inférence en phase de test, nous choisissons forcément le vecteur de la capsule associée à la classification étant donné que nous ne sommes pas censé être au courant de la bonne classe. La partie décodeur a un objectif de régularisation en permettant de mieux entraîner la capsule associée à la classe ainsi que toutes les capsules antérieures qui lui sont liées. Cette phase est composée d'un enchaînement, soit de couches entièrement connectées, soit de couches de déconvolution 2D tant que la dimensionnalité de la sortie finale est similaire à celle de l'entrée initiale du réseau. Concernant la fonction d'évaluation de cette partie, il s'agit de la distance euclidienne entre l'image reconstruite et celle d'entrée. La fonction d'évaluation de l'architecture dans se globalité est dès lors une somme pondérée de la fonction de chacune des deux parties, en privilégiant la fonction de la partie encodeur qui sert à la classification, but principal de l'architecture.

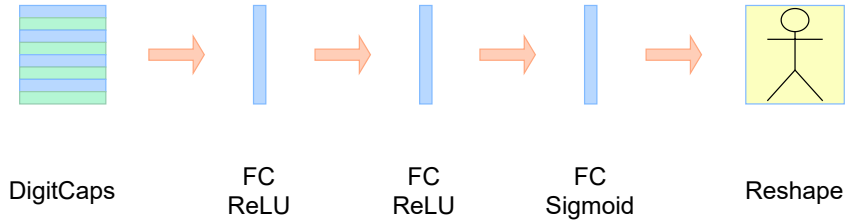

FIGURE 2.12 – Exemple d'architecture capsule network - partie reconstruction

2.7 Conclusion

L'apprentissage profond a énormément évolué depuis ses fondements et a apporté d'énormes progrès dans de nombreux domaines notamment celui de la vision par ordinateur, que ce soit pour l'analyse d'images avec la convolution et les capsules, ou celui de l'analyse vidéo avec la récurrence et la convolution 3D. Les capsules justement, même si assez récentes, ont finalement été plutôt sous-exploitées et se retrouvent rarement dans la littérature. C'est pourquoi nous allons désormais présenter notre principale contribution, à savoir la première application d'un réseau neuronal à capsules 2D à l'étude d'un flux vidéo avec l'architecture Deep 2D Video Capsule Network avec module de décalage temporel.

Reconnaissance des gestes et d'actions de la main par un réseau neuronal à capsules

3.1 Introduction

La reconnaissance de gestes et la compréhension de vidéos sont des défis demandant généralement un coût de calcul énorme, ce qui peut rendre dès lors difficile le traitement de vidéos en continue. De plus, beaucoup d'applications ont besoin de solutions efficaces et précises à ces problèmes comme la reconnaissance de gestes de la main pour de nouvelles interactions en réalité augmentée ou virtuelle.

Les méthodes d'apprentissage profond sont devenues l'une des techniques les plus performantes pour la compréhension d'images et de vidéos, particulièrement depuis le développement des réseaux neuronaux convolutifs [100] (voir section 2.4). Ces méthodes ont aussi prouvé leur avantage à être déployées en temps réel sur des applications, tout du moins pour de relativement petits réseaux, puisque seule la phase d'apprentissage est principalement longue et coûteuse car elle rajoute à l'inférence le calcul du gradient et sa rétropropagation sur tous les poids du réseau. Cependant, l'analyse d'une vidéo, ainsi que sa dimension temporelle en plus à prendre en compte, complexifie le problème.

3.1.1 Challenges

En effet, si les architectures de convolutions 3D ont le bénéfice de prendre en compte les informations spatiales et temporelles, elles sont cependant généralement trop lourdes pour être appliquées à des cas d'utilisation en temps réel surtout avec une puissance de calcul raisonnable. Inversement, les CNNs 2D, grâce notamment au partage des poids propre à leur architecture (voir section 2.4), demande peu de puissance de calcul, sont parfaits pour un entraînement rapide et de l'inférence en temps réel, mais leur implémentation basique ne permet pas l'extraction et l'analyse d'informations temporelles puisqu'ils ne prennent qu'une unique image en entrée. Les réseaux de neurones récurrents, quant à eux, sont extrêmement performants pour analyser des données temporelles mais ne prennent en entrée que des données de faible dimensionnalité, donc pas directement de flux de vidéos continus, une image seule étant déjà presque trop grande pour ce type de réseau.

En résumé, les contraintes et défis principaux que nous nous sommes fixés sont donc :

— Puisque la solution doit pouvoir être déployée en temps réel sur un appareil autonome à la puissance de calcul limité, l'architecture développée doit être de taille raisonnable, à l'inférence rapide, tout en restant au maximum performante pour des cas d'application réelle.

— A cela s'ajoute que cette architecture doit prendre en entrée des données brutes ou avec le minimum de pré-traitement, un flux vidéo étant le plus facile à extraire et une modalité commune à la grande majorité des appareils de réalité augmentée, elle doit donc pouvoir prendre en entrée des images.

— Puisqu'elle doit prendre en entrée un flux vidéo et que la dimension temporelle et sa compréhension sont indispensables dans le cas de reconnaissance de gestes ou d'actions, l'architecture doit intégrer ces contraintes et pouvoir traiter cette dimensionnalité à moindre coût pour ne pas impacter ses performances en terme de puissance de calcul.

Pour répondre à ces problèmes, deux travaux nous serviront de base et de point de départ. Le premier est le réseau neuronal à capsules développé par Sabour *et al.* [81] et le second est une approche qui utilise un modèle CNN en lui appliquant un module de décalage temporel, développé par Lin *et al.* [61]. Sabour *et al.* [81] proposent une nouvelle architecture aux résultats comparables à ceux d'un CNN 2D mais avec un nombre de paramètres à entraîner bien moindre, la contrepartie étant un temps d'entraînement plus long à cause de l'algorithme de routing (voir section 2.6) mais qui n'a aucune incidence au moment de l'inférence. Étant donné que cette nouvelle architecture, à l'instar du CNN 2D, n'extrait que des informations spatiales, nous nous sommes intéressés aux travaux de Lin *et al.* [61] et de leur module de décalage temporel qui permet, au sein d'un CNN 2D, d'apporter du traitement d'informations temporelles en décalant les résultats des opérations d'une couche de convolution appliquées à une image de la vidéo vers la même couche mais au moment du traitement de l'image suivante et/ou précédente, le tout sans incidence sur le nombre de paramètres à gérer et donc sur la puissance de calcul nécessaire.

3.1.2 Approche proposée

Nous proposons ainsi dans ce chapitre une nouvelle architecture combinant le réseau neuronal profond à capsules (DeepCaps) [108] avec le module de décalage temporel [61] pour la compréhension efficace de vidéos et de reconnaissance d'actions, le 2D Deep Video Capsule Network avec module de décalage temporel (2D DVCN). Nous avons ainsi développé et étudié trois implémentations différentes d'opérations de décalage sur les capsules basées sur l'approche de Li *et al.* [61] ; à savoir, le décalage des premiers filtres des premières capsules d'une couche (voir Figure 3.2), le décalage des premiers filtres de toutes les capsules d'une couche (voir Figure 3.3) et enfin le décalage de l'entièreté des filtres des premières capsules d'une couche (voir Figure 3.4). Nous avons ainsi testé cette nouvelle architecture sur des bases de données exigeantes comme FPHA [1] et DHG14/28 [46] et ainsi démontré que notre méthode approche ou surpasse les résultats de l'état de l'art par rapport à des méthodes utilisant des images de couleurs ou de profondeurs, le tout avec

une architecture ayant 10 à 40 fois moins de paramètres que la plupart des autres méthodes récurrentes ou de convolution 2D/3D.

3.1.3 Motivations

L'utilisation d'une architecture de réseau neuronal à capsules est motivée par le fait que, outre un temps d'apprentissage plus long à cause de l'algorithme de routing, cette architecture s'est montrée tout aussi efficace qu'un réseau neuronal convolutif 2D tout en étant beaucoup moins profonde. Ceci nous permet de justifier d'un réseau avec beaucoup moins de paramètres à gérer qui, une fois l'entraînement effectué sur une machine externe, peut être déployé et exécuté sur des appareils autonomes et à la puissance de calcul limitée comme c'est le cas pour les casques de réalité augmentée.

L'implémentation du module de décalage temporel appliqué à un réseau neuronal à capsules est motivée par la force de capsules à extraire et comprendre les relations entre les caractéristiques spatiales extraites par l'opération de convolution. Notre objectif est de renforcer les informations temporelles décalées et partagées d'une image à l'autre de la vidéo en liant ces convolutions décalées grâce à l'architecture des capsules et leur algorithme de routing. L'utilisation de ce module nous permet ainsi d'avoir un réseau incluant la dimension temporelle dans sa structure sans avoir à l'alourdir à l'instar d'une architecture de convolution 3D.

Nous avons aussi déterminé comment éviter la divergence de poids lorsque des données de grandes dimensions sont envoyées en entrée à un réseau neural à capsules en modifiant l'architecture originale de DeepCaps [108] permettant ainsi d'envoyer des images dix fois plus grande que la plupart des images de base de données utilisées habituellement pour tester les autres architectures à capsules.

3.2 2D Deep Video Capsule Network avec décalage temporel

Le développement d'architectures de réseau neuronal à capsules 2D pour le traitement et l'analyse de flux vidéo n'a encore presque pas été exploré. Les seuls travaux explorant le traitement d'informations temporelles à l'aide de capsules se basent sur l'utilisation de cellules récurrentes LSTM [109] qui ne peuvent être dès lors utilisées que sur des données de faible dimensionnalité, ce qui n'est pas le cas du traitement de vidéos. Cependant, les réseaux neuronaux à capsules ont déjà prouvé leur potentiel et efficacité pour l'extraction de caractéristiques spatiales [81], [108] ainsi que leur mise en relation. C'est pourquoi nous avons décidé d'explorer comment il serait désormais possible d'incorporer à cette architecture, une capacité d'extraction et d'analyse d'informations et de caractéristiques temporelles. Commençons par étudier comment nous avons implémenté le module de décalage temporel au sein de capsules.

3.2.1 Décalage temporel sur capsule

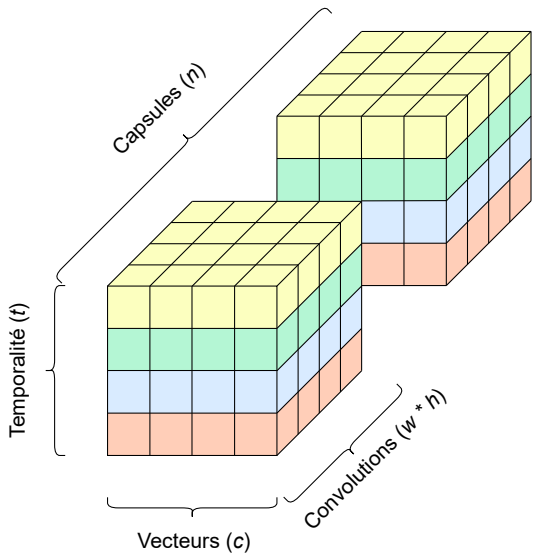

FIGURE 3.1 – Représentation d'une couche de capsules appliquée à plusieurs images d'un flux vidéo

Représentons la forme d'une sortie d'une couche de capsules convolutives comme $R^{(w,h,c,n)}$ avec c la taille du vecteur d'une capsule, sachant qu'un élément de ce vecteur est le résultat d'une convolution appliquée à l'entrée de la couche, w et h, respectivement la largeur et la hauteur de chacun de ces filtres et enfin n, le nombre de capsules dans la couche (voir section 2.6). Rappelons ainsi que, structurellement, une couche de capsules est simplement constituée de $c*n$ opérations de convolution de taille $w*h$ regroupées en n capsules de c filtres chacune. Ce regroupement permet d'extraire et d'analyser des modularités à un motif reconnu (taille, rotation, emplacement, etc.) et c'est au moment de l'algorithme de routing que ces motifs sont mis en relation les uns par rapport aux autres via les capsules des couches supérieures.

A l'instar de l'application du module de décalage temporel appliqué à une architecture de convolution 2D, les opérations menées au sein d'une couche de capsules sont exécutées indépendamment pour chaque image de la vidéo passée en entrée, c'est a posteriori de l'entrée de chaque couche que le module de décalage temporel va s'appliquer en décalant les résultats de la couche précédente appliquée à l'image de temps t à celle de temps $t+1$ afin de partager l'information au moment du calcul de la couche suivante. On peut donc ainsi dans notre cas représenter la sortie d'une couche de capsules en ajoutant une dimension t représentant le nombre d'images composant la vidéo (voir Figure 3.1) : $R^{(t,w,h,c,n)}$.

Précisons que le décalage peut être effectué dans les deux sens si nous souhaitons faire une application pour une utilisation hors ligne, dans ce cas de figure, puisque nous possédons l'entièreté de la vidéo a priori, il est possible d'effectuer une partie du décalage de l'image t vers l'image $t+1$ mais aussi de l'image t vers l'image $t-1$ afin de transmettre l'information temporelle dans les deux sens. Pour un cas d'utilisation en temps réel, puisque la vidéo est capturée image par image et si nous souhaitons récupérer les classifications au fur et à mesure, dans ce cas, l'opération de décalage n'est effectuée que dans un sens, du temps t vers le temps $t+1$, en ce cas, puisque la vidéo est traitée au fur et à mesure, les sorties de chaque couche sont sauvegardées à chaque inférence afin d'être récupérées pour le décalage au moment de l'inférence suivante.

Avant d'étudier les trois différentes implémentations développées du module de décalage temporel appliqué aux capsules, rappelons le fonctionnement de ce module appliqué à un CNN [61]. Soit une couche de convolution 2D effectuant c opérations de convolutions indépendamment sur les t images de la vidéo, une proportion m est fixée de sorte que les $\frac{c}{m}$ premières convolutions soient décalées d'une étape dans le futur et les $\frac{c}{m}$ convolutions suivantes d'une étape dans le passé pour le cas d'une application hors ligne, pour une application en temps réelle, que les $\frac{2*c}{m}$ premières convolutions soient décalées d'une étape dans le futur. Cette proportion m est choisie a priori, est identique pour toutes les couches sur lesquelles sont appliquées le module et est constante durant toute la phase d'entraînement, il s'agit donc d'un nouvel hyperparamètre à gérer.

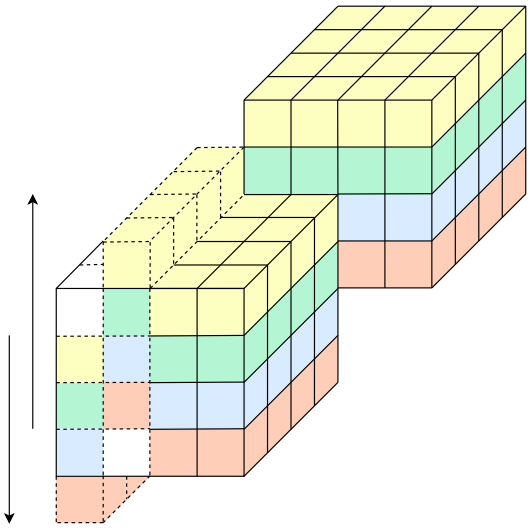

FIGURE 3.2 – Décalage temporel appliqué aux premiers filtres des premières capsules

Observons dès à présent les trois implémentations développées dans le cadre de l'application du module de décalage temporel appliqué aux capsules :

— **Décalage des premiers filtres des premières capsules** : Dans cette implémentation, une capsule de la couche sur laquelle est appliquée le module est considérée de la même manière qu'un filtre de convolution dans sa couche dans le cas de l'application du module à un CNN 2D (voir Figure 3.2). Soit une couche comprenant n capsules, il est choisi une première proportion m de telle

sorte que les $\frac{n}{m}$ premières capsules soient décalées. Par la suite, chacune de ces capsules sont à leur tour considérées comme une couche de convolution sur laquelle serait appliquée le module de décalage temporel. Ainsi, si chaque capsule contient c filtres de convolution, il est choisi une nouvelle proportion m' de sorte que les $\frac{2*c}{m'}$ premiers filtres soient décalés, une première moitié dans le futur et la seconde dans le passé dans le cas d'une implémentation hors ligne, ou tous décalés dans le futur dans le cas en temps réel. Dans cette configuration, les capsules sont bien considérées comme représentant chacune une caractéristique spatiale à l'instar de chaque convolution d'une couche d'un CNN. Décaler une partie des vecteurs de ces capsules revient donc à transmettre aux images suivantes ou précédentes, une partie d'informations temporelles quant à la valeur de certaines modularités de cette caractéristique capturée par la capsule décalée.

— **Décalage des premiers filtres de toutes les capsules** : Pour cette implémentation, nous considérons les caractéristiques spatiales capturées par chaque capsule indépendamment des autres capsules (voir Figure 3.3). Ainsi, chaque capsule est considérée comme une couche de convolution à part entière. Toutes les capsules subissent donc un décalage de leur $\frac{2*c}{m'}$ premiers filtres de convolution de la même manière que dans le cas de l'implémentation précédente que nous soyons dans une application hors ligne ou en temps réel. Notre motivation pour cette implémentation est de permettre à une couche de capsules de pouvoir capturer de l'information temporelle pour chaque caractéristique spatiale complexe capturée par chaque capsule, notamment par le fait qu'elles soient indépendantes les unes par rapport aux autres, le partage de ces informations est interne à chaque capsule. Dans cette configuration, le décalage temporel est dès lors une partie intégrante du fonctionnement et de l'architecture d'une capsule puisque commune à toutes.

— **Décalage de l'entièreté des filtres des premières capsules** : Cette dernière implémentation représente plus une implémentation naïve du décalage temporel originalement implémenté pour les CNNs (voir Figure 3.4). En effet,

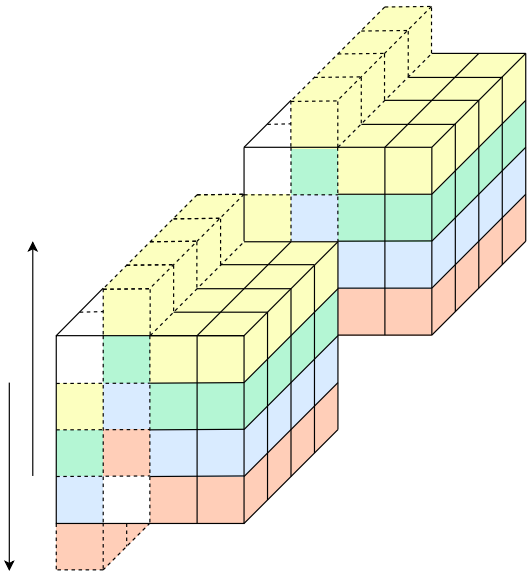

FIGURE 3.3 – Décalage temporel appliqué aux premiers filtres de toutes les capsules

rappelons qu'une couche de capsules est avant tout constituée d'une couche effectuant des opérations de convolutions générant $c * n$ filtres regroupés en n capsules de c filtres chacune. Dès lors, à l'instar de l'application du module de décalage temporel sur CNN, si on effectue le module sur une proportion des $\frac{2*c*n}{m'}$ premiers filtres avant leur regroupement en capsule, une fois assemblés, nous pouvons remarquer que nous avons effectué un décalage de l'entièreté des filtres composant les premières capsules, il suffit juste de s'assurer que la proportion choisie permet de décaler suffisamment de filtres pour décaler obligatoirement des capsules entières. D'un autre point de vue, on considère ici les capsules comme des filtres de convolutions d'une couche de CNN, nous ne nous intéressons dès lors plus vraiment à la composition d'une capsule et de l'information temporelle que l'on fait circuler à l'intérieur, mais nous considérons plutôt la couche de capsules dans sa globalité, nous partageons ici des caractéristiques spatiales complexes ainsi que toutes leurs modularités dans le temps.

Ces différentes implémentations, telles que précédemment décrites, partagent cependant toutes le même inconvénient, le décalage des différents filtres composant les

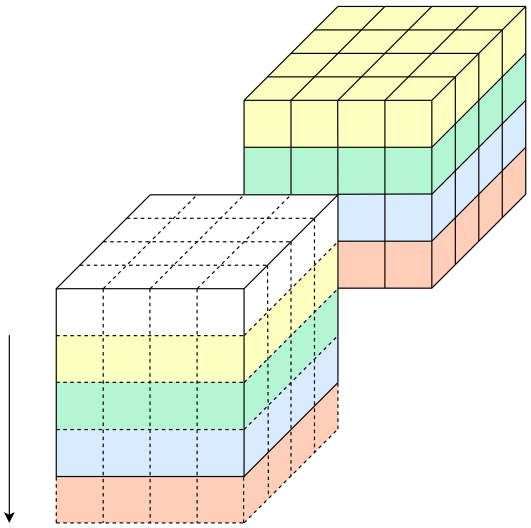

FIGURE 3.4 – Décalage temporel appliqué à tous les filtres des premières capsules

capsules fait forcément perdre de l'informations à propos des caractéristiques spatiales extraites de l'image courante. Pour éviter cette perte, chaque implémentation a la même structure dans ce que nous appelons une ShiftConvCaps (voir Figure 3.5). A l'instar des travaux originels du module de décalage temporel appliqué sur CNN, nous ajoutons une branche résiduelle à l'intérieur de chaque capsule convolutive où les opérations de décalage et de convolution sont effectuées. Une branche résiduelle consiste à diviser une branche de calcul en plusieurs sous-branches effectuant des calculs indépendamment les unes des autres avant de les rassembler plus tard dans le réseau et permettre de prendre en compte chacun des calculs effectués. Ici, l'entrée d'une couche de capsules reste à la fois intacte dans une branche, tandis que dans l'autre, l'un des trois modules de décalage temporel est appliqué dessus ainsi que l'opération de convolution habituel d'une capsule, ces deux branches sont ensuite sommées avant de terminer par l'habituelle fonction d'activation de squeeze de la capsule, cela permet, en même temps que de transmettre de l'information temporelle via une branche, de garder aussi les caractéristiques spatiales de l'image actuelle intactes.

Maintenant que nous avons étudié en détails les différentes implémentations du module de décalage temporel appliqué aux capsules, observons désormais comment

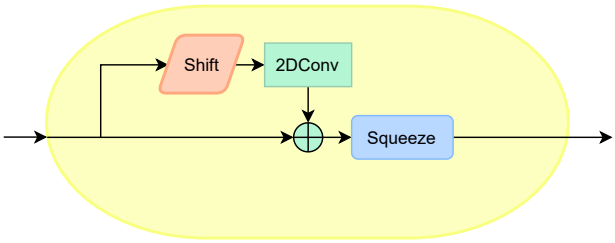

FIGURE 3.5 – Déroulement des opérations d'une capsule temporelle

ce module a été utilisé afin de développer l'architecture 2D DVCN.

3.2.2 Architecture - encodeur

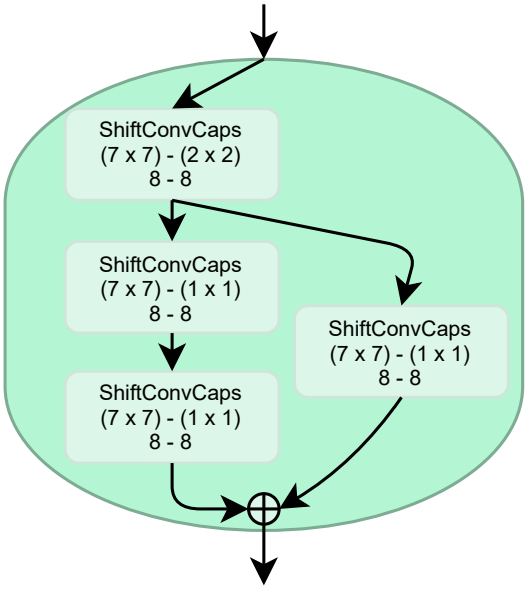

FIGURE 3.6 – Illustration d'une CapsCell qui constitue la composante principale de notre architecture en regroupant plusieurs couches de capsules convolutives sur lesquelles est appliqué le module de décalage temporel. Pour une ShiftConvCaps, le premier (n x n) représente la taille du filtre de convolution 2D, le second représentant le pas. Le deux derniers chiffres représentent le nombre de capsules dans la couche ainsi que le nombre de convolutions dans chacune de ces capsules

Notre architecture 2D DVCN avec module de décalage temporel est inspiré par le modèle DeepCaps [108]. L'architecture globale reste similaire à une architecture de réseau neuronal à capsules basique, à savoir une première étape de couches de

convolutions, afin de réduire la dimensionnalité des entrées et d'extraire les premières caractéristiques spatiales basiques, une seconde étape de couches de capsules convolutives, pour extraire et mettre en relation les caractéristiques spatiales plus complexes et enfin une dernière étape de classification suivie d'une étape de reconstruction. Avant de détailler l'architecture dans sa globalité, attardons nous sur la composition des CapsCell constituant l'étape de couches de capsules convolutives. En effet, à l'instar de l'architecture de DeepCaps, nous regroupons quatre couches de capsules convolutives avec module de décalage temporel (ou ShiftConvCaps) (voir Figure 3.6). Une CapsCell est composée d'une branche principale de 3 couches ShiftConvCaps, une connexion détournée est ajoutée entre la première et la dernière couche, elle a pour but, à la fois d'éviter une divergence du gradient au moment de l'entraînement [108], mais aussi d'ajouter une connexion intermédiaire entre les capsules haut niveau et bas niveau.

Ces CapsCells constituent la principale composante de la partie encodeur et classifieur de notre architecture (voir Figure 3.7). Si nous souhaitons augmenter la profondeur du réseau à mesure que la base de données se complexifie, il sera privilégié d'ajouter des CapsCells complètes plutôt que d'ajouter de simples et uniques couches de ShiftConvCaps. En détails, la partie encodeur reçoit en entrée l'image qui doit être classifiée, celle-ci subit une première étape d'opérations de convolutions 2D afin d'extraire les premiers motifs basiques et de réduire la dimensionnalité des entrées pour les couches de capsules. La deuxième étape est constituée d'un enchaînement de CapsCells qui auront pour but d'extraire les caractéristiques spatiales complexes et de les mettre en relation les unes par rapport aux autres grâce aux capsules mais aussi de les étudier dans le temps grâce au module de décalage temporel. Cette étape se termine par deux couches de capsules convolutives dont l'une récupère la sortie de la dernière CapsCells tandis que l'autre la sortie de la CapsCell précédente. Les deux sorties sont ainsi collectées et concaténées afin d'être passées en entrée de la dernière étape composée d'une couche de FlattenCaps effectuant de simples opérations entièrement connectées et possédant un nombre de capsules égal

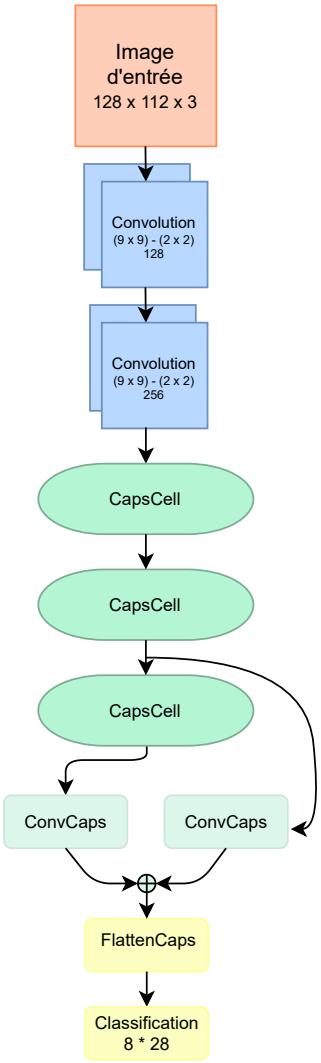

FIGURE 3.7 – Architecture de notre réseau 2D DVCN - Partie classification

au nombre de classes que le réseau doit gérer. Les changements apportés à l'architecture DeepCaps, en plus d'y ajouter le module de décalage temporel, figurent sur la taille du pas des deux premières couches de convolution. Là où leur architecture a surtout été testée sur des bases de données peu complexes comme MNIST [110] et pouvait donc se permettre d'utiliser des images d'entrées de très faible dimensionnalité (64 * 64), dans notre cas, le problème de reconnaissance de gestes et d'actions de la main étant bien plus complexe, nous avons préféré proposer plus d'informations à notre architecture avec des images de plus grande taille (128 * 112). Or, après

plusieurs tests, il se trouve que les architectures de réseau neuronal à capsules n'acceptent pas de recevoir des données de trop grandes dimensionnalités au niveau des couches de capsules, entraînant irrémédiablement une rapide explosion du gradient et donc un réseau qui ne converge plus. Pour répondre à ce problème nous arrivons à garder des capsules de dimension raisonnablement grande dans les CapsCells en réduisant le pas de convolution de la plupart des couches ShiftConvCaps sauf celles d'entrée. Nous avons aussi décidé d'augmenter la taille des filtres de convolutions de ces couches pour mieux se rapprocher du comportement des capsules des travaux originaux de Sabour *et al.* [81].

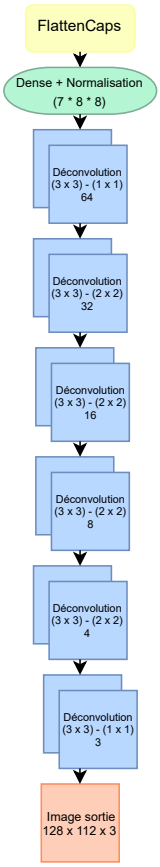

FIGURE 3.8 – Architecture de notre réseau 2D DVCN - Partie reconstruction

3.2.3 Architecture - décodeur

La seconde partie du réseau fait office de décodeur et de reconstructeur (voir Figure 3.8). Elle prend en entrée le vecteur d'une des capsules de la couche de FlattenCaps. Il s'agit, soit de la capsule associée au bon label de l'image d'entrée durant la phase d'entraînement, soit de la capsule qui possède la plus grande norme, et qui décrit donc le label de l'image, durant une phase d'inférence. Les valeurs de ce vecteur sont d'abord passées en entrée d'une couche entièrement connectée avant un enchaînement de couches de déconvolution 2D. Le nombre de neurones de la couche entièrement connectée et le nombre de couches de déconvolution dépend de la taille du vecteur d'entrée et de la dimension de l'image à reconstituer pour qu'elle soit égale à la dimension de l'image d'entrée du réseau.

Les fonctions de pertes utilisées sont les mêmes que pour l'architecture de réseau neuronal à capsules traditionelle (voir section 2.6), à savoir la margin loss pour la partie encodeur et classification du réseau :

$$\mathcal{F}_k = T_k * max(0, m^+ - ||v_k||)^2 + \alpha * (1 - T_k) * max(0, ||v_k|| - m^-)^2 \tag{3.1}$$

avec les valeurs habituelles de 0,9 pour m^+ et 0,1 pour m^- et de 0.5 pour α. De même, la partie décodeur et reconstruction utilise l'habituelle erreur quadratique :

$$MSE = \frac{1}{n} * \sum_{i=1}^{n} (P_i' - P_i)^2 \tag{3.2}$$

— n : nombre de pixels dans les images d'entrée et reconstruite

— P_i' : valeur du pixel i de l'image reconstruite

— P_i : valeur du pixel i de l'image d'entrée

Ces deux fonctions de pertes sont sommées et pondérées de sorte que la perte de la partie de reconstruction du réseau représente 0,5% de la perte du réseau globale.

3.2.4 Expérimentations

Étudions dès à présent les différentes expérimentations et évaluations de notre approche sur deux bases de données contenant divers défis. Notre méthode sera aussi comparée aux autres méthodes existantes de l'état de l'art et prenant aussi en entrée des données de flux vidéo couleur ou de profondeur.

Précisons que les deux évaluations ont été menées dans les même conditions d'un point de vue purement technologique. La bibliothèque Keras [111] a été utilisée avec Tensorflow [112] comme backend sur une machine équipée d'une configuration de 32 Go de RAM, d'un Intel i9 9900k et d'une Nvidia RTX Titan. Pour toutes les expériences, l'algorithme d'optimisation stochastique Adam [113] a été utilisé avec un taux d'apprentissage (learning rate) de l'ordre de 0,001 et un entraînement sur 500 itérations dans une configuration du module de décalage temporel hors ligne.

D'un point de vue de préparation des données, notons qu'une difficulté de notre architecture, commune aux travaux de TSM sur CNN 2D, est qu'elle reçoit en entrée non pas une vidéo mais une image, c'est au moment de l'application de module de décalage temporel que les images d'une même vidéo sont regroupées puis décalées. Or, étant donné que, durant une phase d'entraînement, les bibliothèques d'apprentissage profond effectuent les inférences sur chaque image en parallèle afin d'accélérer le processus, puisqu'il nous faut, à chaque étape du module de décalage temporel les résultats de l'application de ShiftConvCaps précédentes sur l'ensemble des images de la vidéo, nous ne pouvons pas nous permettre d'avoir des vidéos trop longues au risque de surcharger le GPU au moment de l'entraînement. Pour éviter cela, les séquences des différentes bases sont échantillonnées en un même nombre d'images.

Des opérations d'augmentation des données sont aussi effectuées durant l'apprentissage. Au moment de l'échantillonnage de la vidéo, des images peuvent être sautées ou non afin de générer des séquences légèrement différentes. D'un point de vue de l'image en elle-même, des opérations de légère rotation, redimensionnement ou de retournement sur l'axe vertical peuvent être effectuées. Nous nous assurons juste que toutes les images d'une même séquence subissent les mêmes opérations

d'augmentation de données afin de garder une vidéo cohérente. La dernière augmentation consiste, à partir de la vidéo de dimension 160*120, de découper une sous partie de taille 128*112 qui sera envoyée en entrée du réseau. Au moment de la phase de test ou pour une inférence post-entraînement, aucune de ces opérations n'est appliquée, le découpage se fait toujours au centre de l'image et l'échantillonnage ne saute aucune image et est simplement effectué à pas régulier.

First Person Hand Action

La première base testée est First Person Hand Action Dataset (FPHA) [1]. Il s'agit d'une base de données d'actions effectuées à la première personne dans un cadre de manipulation de différents objets avec les mains dans le contexte d'une cuisine (voir Figure 3.9). La base contient 1 175 séquences réparties sur 45 classes où le sujet manipule un des 28 objets. Ces séquences sont disponibles selon trois modalités différentes, une première constitue un flux vidéo couleur, la seconde un flux vidéo de profondeur et la dernière est constituée d'une série de positions du squelette de la main effectuant l'action. Dans notre cas, comme expliqué plus tôt, étant donné que le flux vidéo de couleurs est disponible, nous n'avons utilisé que cette modalité pour entraîner notre architecture.

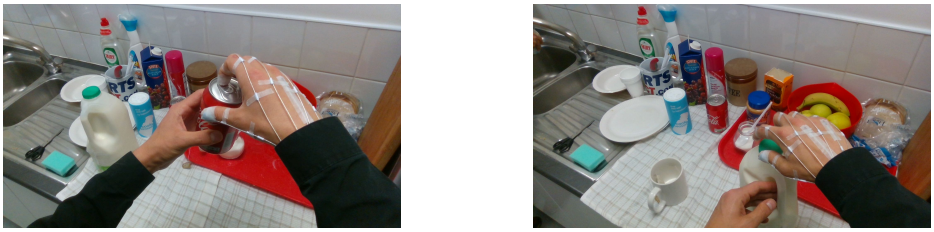

FIGURE 3.9 – Exemple de deux actions de la base FPHA [1]

Comparé à d'autres base de données, FPAH est relativement petite. En effet, avec ses 1 175 séquences pour 45 classes, on dénombre une moyenne de 26 séquences par action, ce qui est assez peu comparé, par exemple, au premier ImageNet avec ses 200 000 images pour 1000 classes soit 200 exemples par label. Le risque avec une base de données avec aussi peu d'exemples par classe est d'avoir un réseau trop profond qui va effectuer un sur-apprentissage, c'est à dire faire converger tous ses

poids de sorte que le réseau s'active parfaitement aux données d'entraînement mais qu'il n'apprenne cependant aucune généralisation et provoque donc une classification aléatoire au moment d'une inférence. Afin d'éviter ce problème, nous avons décidé de considérablement réduire la profondeur de notre architecture par rapport à celle d'un DeepCaps et de ne garder que 2 CapsCells. Une telle configuration nous permet d'obtenir un très petit réseau avec seulement 4 millions de paramètres à gérer tout en améliorant les résultats par rapport à un réseau plus profond.

La présence de manipulation d'objets dans la base apporte une facilité dans la reconnaissance de l'action puisque le réseau n'a pas à se baser uniquement sur l'évolution temporelle de la main pour estimer l'action effectuée mais peut aussi s'aider de l'information spatiale qu'est l'objet manipulé et reconnu. Cependant, puisqu'il y a plus de classes (45) que d'objets manipulés (28), se baser uniquement sur l'objet est impossible puisque certaines classes ont le même objet en commun, manipulé cependant différemment. Ainsi puisque nous avons cet avantage d'avoir un objet en plus de la main dans les séquences vidéo, nous avons décidé que pour cette base, un échantillonage des séquences sur 16 images serait suffisant.

Avant de comparer nos résultats avec les autres méthodes de l'état de l'art sur cette base, et afin de prouver les bénéfices du module de décalage temporel appliqué sur un réseau neuronal à capsule, nous avons aussi conduit des expérimentations en testant l'exact même architecture avec les mêmes hyper-paramètres et dans les mêmes conditions d'entraînement mais avec ou sans l'une des différentes implémentations du module de décalage temporel. Nous pouvons ainsi prouver qu'il y a une réelle et directe amélioration des résultats en appliquant deux des trois implémentations du module que nous avons implémentées et que le partage d'information temporelle a un impact positif sur la qualité de classification avec une amélioration d'au moins 4% de la précision. Cependant, nous pouvons observer un dégradation significative des résultats lorsque nous appliquons le module de décalage temporel sur les premiers filtres de premières capsules (voir Table 3.1).

Nous pouvons expliquer cette dégradation étant donné qu'il s'agit de la seule implémentation où un décalage totalement partiel est effectué, que ce soit sur le

Décalage temporel	Taux de reconnaissance (%)
Sans décalage	70.01
Décalage premiers filtres premières capsules	64.14
Décalage tous les filtres premières capsules	74.33
Décalage premiers filtres toutes capsules	**76.72**

TABLE 3.1 – Comparaison des différentes implémentations de module de décalage temporel sur capsule sur la base FPHA [1]

nombre de capsules ou sur le nombre de filtres décalés au sein de ces capsules. Dans cette configuration, d'une couche à l'autre, des capsules s'associent à la fois avec des capsules où aucun décalage n'a été effectué mais aussi avec des capsules contenant des informations sur le passé, et éventuellement futur dans un cas hors ligne, mais aussi sur le présent. Ce mélange d'informations de toute temporalité ne se retrouve pas dans le cas de décalage de tous les filtres des premières capsules où, au moment du routing, les capsules de la couche suivante reçoivent soit une capsule contenant entièrement de l'information présente, soit entièrement de l'information d'une temporalité futur et/ou passée. De même, dans l'implémentation du décalage des premiers filtres de toutes les capsules, celles-ci agissent et se comportent dès lors toutes de la même manière ce qui peut aider le réseau à mieux instancier et converger ses poids de manière cohérente. Nous pouvons cependant aussi remarquer que sur la partie de la base de données réservée à l'entraînement, l'évolution de la classification n'est pas dégradée (voir Figure 3.10). Nous pouvons donc ainsi conclure à un surapprentissage du réseau lorsque le décalage des premiers filtres des premières capsules est l'implémentation appliquée. Concernant les deux autres implémentations du module, nous pouvons observer qu'elles conduisent vers des améliorations similaires en surpassant les résultats de classification par rapport à l'absence de décalage. Cette amélioration peut être expliquée par le fait que permettre, à toutes étapes du réseau et de l'algorithme de routing, d'avoir une cohérence générale du partage d'informations temporelles, que ce soit à l'intérieur d'une couche selon le nombre de capsules décalées où à l'intérieur même d'une capsule avec le nombre de motifs décalés, est le moyen le plus logique de partager de l'information temporelle au sein d'un réseau

neuronal à capsule. En effet, étant donné qu'une capsule représente des informations spatiales complexes en l'encodant au sein d'un vecteur, il est logique, soit de décaler entièrement certaines de ces caractéristiques spatiales, comme lorsque nous décalons l'entièreté des filtres d'une capsule, soit de décaler dans le temps les même attributs spécifiques, comme l'orientation, la dimension ou la position, encodés chacun dans une cellule du vecteur, pour toutes les capsules, ce que nous faisons dans le cas du décalage des premiers filtres de toutes les capsules.

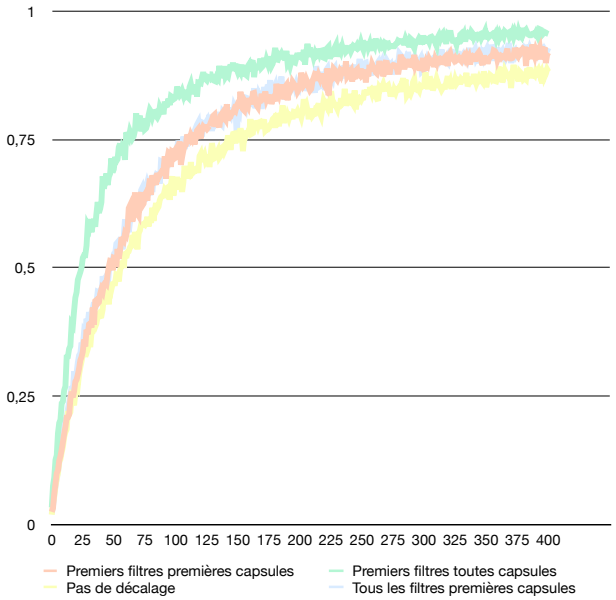

FIGURE 3.10 – Évaluation de la précision d'entraînement selon le décalage appliqué

Pour la suite des expérimentations, c'est ainsi le décalage de premiers filtres de toutes les capsules qui a toujours obtenu les meilleurs résultats et qui sont ainsi présentés. Ainsi, concernant la comparaison face aux autres méthodes sur la base FPHA, nous observons ainsi que nous obtenons une meilleure précision de classification que toutes les autres méthodes de l'état de l'art qui utilisent aussi le flux vidéo couleur ou de profondeur, le tout, avec 6 à 45 fois moins de paramètres à gérer et à entraîner pour le réseau (voir Table 3.2). Nous avons aussi appliqué la méthode TSM sur la base FPHA afin de comparer le bénéfice de notre développement du module de décalage temporel sur capsule plutôt que sur convolution simple, bénéfice prouvé

par les 5% supplémentaires de bonnes classifications. Nous pouvons observer que, même si l'amélioration n'est pas aussi notable, de l'ordre de 1%, face à la méthode de Feichtenhofer *et al.* [114] qui propose de combiner le résultat de deux CNN 2D parallèles s'exécutant sur deux modalités différentes, l'une de couleur et l'autre de profondeur, non seulement notre architecture n'utilise qu'une seule de ces deux modalités, la couleur, mais en plus, c'est ici que la différence du nombre de paramètres est la plus notable avec notre architecture demandant 45 fois moins de paramètres à entraîner.

Méthode	Nombre paramètres	Taux de reconnaissance (%)
Two stream-color [114]	46M	61.56
Two stream-flow [114]	46M	69.91
Two stream-all [114]	181M	75.30
TSM [61]	24M	71.57
2D DVCNN	**4M**	**76.72**

TABLE 3.2 – Comparaison du 2D DVCNN sur la base de données FPHA [1] avec l'état de l'art

Dynamic Hand Gesture

La deuxième base de données testée est Dynamic Hand Gesture (DHG) [46]. Il s'agit cette fois-ci d'une base de gestes de la main en vue à la troisième personne (voir figure 3.11). Nous avons notamment choisi cette base de données pour démontrer que notre méthode n'est pas seulement efficace dans un unique contexte de reconnaissance d'action à la première personne et que le fait qu'elle accepte tout type d'image en entrée la rend portable et adaptable à bon nombre de problématiques. Cette base contient 2800 séquences réparties sur 28 classes. Parmi ces 28 classes, on retrouve 14 gestes distincts chacun effectué avec un seul doigt et le reste de la main fermée ou effectué avec la main complètement ouverte. Deux modalités sont fournies pour chacune des séquences, soit un flux vidéo de profondeur que nous utiliserons pour notre réseau, soit une séquence de positions du squelette de la main effectuant le geste.

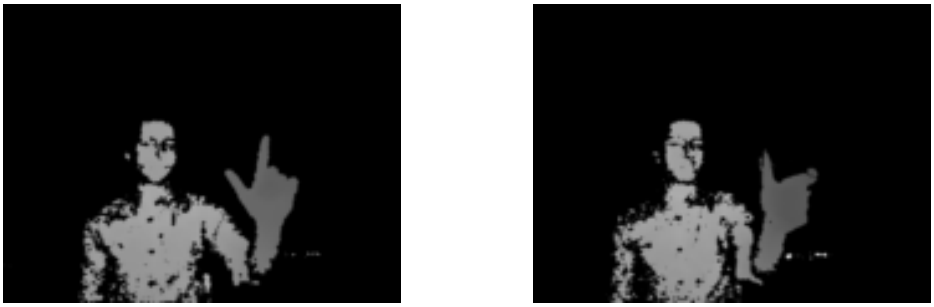

FIGURE 3.11 – Exemple de deux gestes de la base DHG14/28 [46]

Avec ces 2800 séquences pour 28 classes, soit 100 exemples par classe, nous pouvons nous permettre d'augmenter la taille du réseau avec 3 CapsCells tout en évitant un éventuel surapprentissage. D'autant que le problème ici est plus complexe puisque les séquences donnent beaucoup moins d'indices spatiaux que sur la base FPHA. Pas de manipulation d'objet dans la base DHG, la composante spatiale est même d'autant plus difficile que les 28 classes fonctionnent par paire et que le réseau va devoir déterminer les mêmes gestes effectués avec un unique doigt de ceux effectués avec la main complètement ouverte. Étant donné ainsi que la composante temporelle est d'autant plus importante car seule l'évolution du mouvemement du poignée compte pour différencier les 14 différents gestes, nous avons aussi décidé d'opter pour un échantillonage à 32 frames des séquences. Nous obtenons ainsi ici une architecture comprenant environ 7 millions de paramètres à gérer.

De la même manière que pour FPHA, nous avons comparé notre méthode aux autres de l'état de l'art. Ainsi, sur la base DHG, si nous ne battons pas l'état de l'art, à savoir la méthode de Abadi *et al.* [112] qui propose cette fois-ci de fusionner le résultat de deux CNN, l'un 2D et l'autre 3D, nous nous approchons tout de même de leur résultat de classification avec une architecture proposant 20 fois moins de paramètres à entraîner. Nous avons aussi, une fois de plus, testé le module TSM sur CNN 2D sur cette base et montrons, de la même manière que pour FPHA, que l'application du module de décalage temporel appliqué sur des capsules plutôt que sur de simples convolutions offre des bénéfices avec ici un apport de l'ordre de 10% de bonnes classifications supplémentaires (voir Table 3.3).

Méthode	Nombre paramètres	Taux de reconnaissance (%)
TSM [61]	24M	58.66
2D-3DCNN Fusion [115]	140M	**74.41**
2D DVCNN	**7M**	**68.98**

TABLE 3.3 – Comparaison de 2D DVCNN sur la base DHG14/28 [46] avec l'état de l'art

3.3 Conclusion

Nous avons proposé dans ce chapitre une nouvelle architecture pour la compréhension de vidéos et pour la reconnaissance de gestes ou d'actions de la main. Notre réseau 2D DVCN avec module de décalage temporel peut, pour la première fois pour une architecture de réseau neuronal à capsules traiter des entrées de hautes dimensionnalités sans explosion du gradient, que ce soit dans la taille des images passées en entrée, que la dimension temporelle ajoutée pour le traitement de vidéos. Nous avons aussi prouvé que l'intégration et l'implémentation du module de décalage temporel appliqué aux couches de capsules injecte des informations temporelles pertinentes et bénéfiques pour le traitement de séquences vidéos sans ajouter aucun coût de calcul supplémentaire. Nous avons enfin démontré que décaler et partager les premiers filtres de toutes les capsules d'une couche, et dans une moindre mesure, de décaler et partager tous les motifs des premières capsules d'une couche permettait à notre architecture d'obtenir de meilleurs résultats. Ceci est possible grâce à un partage d'informations temporelles cohérent en partageant tout, ou en partie, les caractéristiques spatiales complexes extraites par les capsules. Ceci nous permet ainsi de dépasser l'état de l'art sur la base FPHA et de s'en approcher sur la base DHG, dans les deux cas avec au moins 20 fois moins et jusqu'à 45 fois moins de paramètres à gérer par notre architecture. Grâce à ce réseau, nous répondons ainsi à tous les défis que nous nous étions posés. Même si un réseau neuronal à capsules est plus long à entraîner qu'une architecture de convolution 2D à cause de l'algorithme de routing, celui-ci n'a plus aucune incidence au moment d'une inférence. En outre, cette architecture nous permet de proposer un réseau possédant un très faible

nombre de paramètres à gérer, permettant ainsi son déploiement sur des appareils autonomes à la puissance de calcul limité. De plus, 2D DVCN accepte des vidéos en entrées, de toutes modalités, couleur comme de profondeur, car il s'agit des flux les plus couramment obtenu à partir des capteurs équipés sur la plupart de ces appareils autonomes et ne nécessitent aucun pré-traitement à l'inverse de l'extraction du squelette de la main par exemple. Enfin, le tout est couplé au module de décalage temporel qui permet d'extraire le maximum d'informations de la vidéo d'entrée, que ce soit spatial grâce au capsule, et temporel grâce au module de décalage temporel, le tout, sans impact sur la puissance de calcul nécessaire. Grâce à cette solution, nous allons pouvoir ainsi tester son efficacité et son adaptabilité sur des cas en temps réel.

Application de la reconnaissance d'actions en réalité augmentée - scénario égocentrique

4.1 Introduction

Nous l'avions souligné en regardant les différentes bases de données en rapport avec la reconnaissance d'actions de la main, de nombreuses bases sont centrées sur le point de vue à la première personne ce qui permettrait en théorie leur application dans des cas réels sur des appareils de réalité virtuelle ou augmentée. Cependant, aucune n'a été pensée et développée dans le but d'un cas utile d'application concrète et proposant un ensemble d'actions contraint autour d'un contexte précis et limité.

En effet, ces bases concentrent plutôt leurs efforts pour proposer une diversité de cas de manipulations d'objets ou d'interactions avec l'environnement ou encore avec d'autres personnes plutôt que de se concentrer sur une tâche précise. Néanmoins, les technologies de réalité augmentée, même si elles ne sont pas encore axées grand public, mais plutôt du côté des entreprises, sont déjà présentes sur le marché, comme le casque Microsoft HoloLens ou encore du casque Magic Leap. Il suffit d'expérimenter ses technologies pour comprendre leur potentiel et s'interroger à comment développer un nouveau moyen d'interagir avec, plus qu'avec de simples gestes pour de la manipulation de menus comme c'est déjà le cas sur HoloLens ou plus qu'avec

l'aide d'une manette comme sur Magic Leap [116].

4.1.1 Challenges

Le défi étant de proposer une base qui puisse être facilement utilisée à des fins d'applications en temps réel sur des appareils autonomes de réalité augmentée ou virtuelle, cette base doit répondre à différentes conditions :

— **Type de données** : Les données collectées pour former la base doivent être d'un type (couleur, profondeur, squelette, etc...) qui puisse être facilement accessible en temps réel par un casque de réalité augmentée ou virtuelle et de manière autonome afin d'éviter au maximum l'utilisation d'outils externes qui alourdiraient irrémédiablement l'application développée derrière.

— **Point de vue** : Comme la plupart des autres bases de données d'actions de la main, mais surtout motivée par l'éventualité d'une application, le point de vue de la base se doit d'être à la première personne pour correspondre au point de vue des capteurs disponibles sur les casques de réalité augmentée et virtuelle.

— **Contexte** : Le contexte de la base doit être clair, précis et limité, notamment dans les gestes et actions qu'elles possèdent. Ces derniers doivent être tous centrés autour d'un même et unique objectif afin que l'application proposée soit cohérente, centralisée et utile.

— **Occlusion** : Du fait du point de vue à la première personne associé au domaine de la reconnaissance d'actions, l'interaction entre les mains et l'objet au sein de l'ensemble d'actions à capturer doit faire en sorte que les mains puissent être visibles la majorité du temps, sans occlusion trop présente qui pourrait rendre trop difficile l'exercice d'identification de l'action.

4.1.2 Approche proposée

La mise en place de ces différents défis ainsi que leur analyse afin de les résoudre nous a conduits au développement d'une toute nouvelle base de données que nous avons nommée **FirstPiano**. Cette base, de la façon dont nous l'envisageons dans sa représentation globale et finale, a pour ambition l'apprentissage d'un instrument de

musique, le piano. Pour cela, elle contient, pour le moment, un ensemble de séquences capturées représentant un joueur, plus ou moins expérimenté, effectuant un exercice de gamme. Cet exercice consiste à jouer un ensemble précis de 8 notes, dans un ordre exact et avec une gestuelle particulière. Afin de permettre une réelle application grâce à cette base, les séquences peuvent montrer soit une bonne exécution de la gamme, soit une mauvaise qui est caractérisée par une ou plusieurs mauvaises notes jouées, oubliées ou en trop, ou bien par une mauvaise gestuelle. Nous obtenons ainsi une base enregistrée à l'aide de 4 personnes différentes et contenant 672 séquences réparties entre 2 labels, si nous souhaitons simplement déterminer si une gamme, n'importe laquelle, est bien ou mal jouée, et jusqu'à 16 labels si nous souhaitons en même temps différencier l'une des 7 gammes tout en sachant si elle a été jouée de la main gauche ou de la main droite et s'il s'agit juste d'un aller ou d'un aller-retour ainsi que si nous souhaitons connaître exactement le genre d'erreurs effectuées.

4.1.3 Motivations

Le choix du contexte de l'apprentissage du piano nous permet de répondre aux défis explicités plus tôt. Le piano est l'un des rares instruments qui permette, à partir du point de vue de l'utilisateur, d'observer ses mains de manière naturelle et sans que l'instrument ne vienne obstruer la vue et provoquer de l'occlusion au niveau des mains. De plus, les flux des capteurs intégrés au casque HoloLens nous sont accessibles, nous avons donc décidé d'enregistrer les séquences composant la base directement à partir de ceux-ci nous permettant ainsi de proposer des données qui seront en tout point similaire à celles qui seraient extraites dans un cas d'utilisation réelle du casque sans avoir à utiliser du matériel supplémentaire qui encombrerait l'utilisation.

Enfin, le contexte de l'apprentissage nous offre la possibilité de produire et développer de réels et utiles cas d'applications en réalité augmentée. L'apprentissage d'instruments de musique est un exercice complexe qui se pratique la plupart du temps seul, souvent en parallèle de cours avec l'aide d'un professeur. Avoir une application qui étudie les gestes que nous effectuons afin d'offrir, en complément des

conseils de l'enseignant, des retours en temps réel sur ce que nous effectuons de bien ou de mal, a un réel potentiel. Si notre base contient actuellement uniquement des exercices sur la pratique de gammes majeures, il est tout à fait envisageable de la compléter avec nombres d'autres exercices similaires voire associatifs, comme l'entraînement du rythme ou encore d'exercices d'indépendance des mains gauche et droite.

4.2 FirstPiano : base pour l'apprentissage du piano

Étudions dès à présent la base FirstPiano, du contexte d'apprentissage musical de l'instrument et de l'exercice de gamme à son acquisition à l'aide des capteurs directement intégrés au casque de réalité augmentée HoloLens.

Commençons par étudier et comprendre ce qu'est une gamme dans le domaine musical et comment elle est jouée au piano afin de pouvoir observer ce que sera une séquence enregistrant une gamme bien jouée d'une gamme mal jouée.

4.2.1 Contexte musical

Dans le langage musical occidental, développé durant le Moyen-Âge et aujourd'hui étant le système d'écriture le plus répandu, nous retrouvons 7 notes (ou hauteurs) dites naturelles : *Do Ré Mi Fa Sol La* et *Si*. L'avantage des instruments à clavier, comme le piano, comparés aux autres, est que les touches de ce clavier forment un motif permettant d'identifier aisément les différentes notes (voir Figure 4.1). Le *Do* est souvent pris en référence de point de départ et il s'agit toujours de la touche blanche se trouvant directement avant le paquet de deux touches noires, la suite des notes se déroule ensuite sur les touches blanches suivantes puis le motif des touches blanches et noires se répète ensuite.

Les touches noires représentent quant à elles les altérations de ces 7 différentes notes. Nous pouvons retrouver deux formes d'altérations, *Dièse* ♯ et *Bémol* ♭. Une touche noire directement au dessus, respectivement en dessous, d'une note blanche représente son altération en dièse, respectivement en bémol. Nous pouvons cependant dénoter que dans les couples de notes *Si, Do* et *Mi, Fa*, les notes naturelles ne

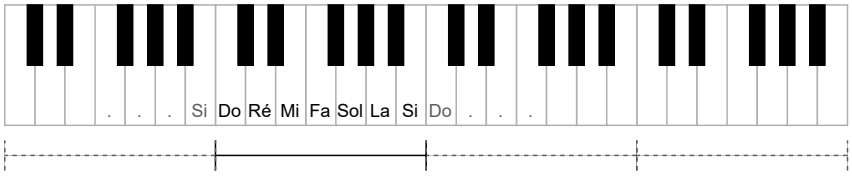

FIGURE 4.1 – Illustration du motif des touches et leurs notes le clavier d'un piano

sont pas séparées par une touche noire, ainsi, *Do*, respectivement *Fa*, est l'altération dièse de *Si*, respectivement *Mi* et vice-versa pour l'altération bémol.

Afin de comprendre comment sont construites les gammes, il est important de comprendre ce que sont un demi-ton et un ton. Dans le système musicale occidental, un demi-ton est le plus petit écart qu'il puisse être trouvé entre deux hauteurs, par exemple entre *Do* et *Do* ♯, entre *La* ♭ et *La* ou encore entre *Si* et *Do*. Sur le clavier d'un piano, les demi-tons apparaissent naturellement, il s'agit des couples de deux touches consécutives, noires comme blanches (voir Figure 4.2).

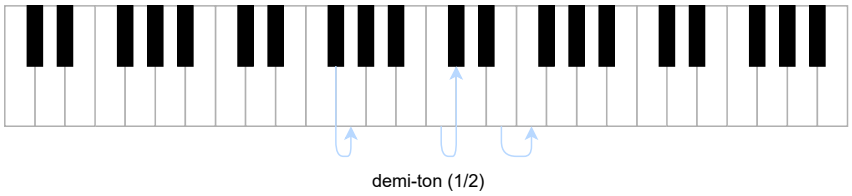

FIGURE 4.2 – Illustration du demi-ton sur le clavier d'un piano

Un ton équivaut quant à lui logiquement à deux demi-tons. Il s'agit souvent de l'écart entre deux notes naturelles (touches blanches), comme entre *Do* et *Ré*, ou altérées (touches noires), comme entre *Sol* ♭ et *La* ♭ par exemple. Autour des cas particuliers des couples de notes naturelles *Si, Do* et *Mi, Fa* qui ne sont séparées que par un demi-ton, nous retrouvons des cas plus particuliers et plus délicat à identifier, même sur un clavier, comme *Mi* ♭ et *Fa* ou encore *Mi* et *Fa* ♯ (voir Figure 4.3).

C'est à partir des notes, naturelles (blanches) comme altérées (noires), et en connaissant le principe de demi-ton et de ton que nous pouvons facilement déterminer comment construire une gamme dans le système musical occidental. Sachant qu'il existe différent type de gamme, nous illustrerons ici uniquement les gammes

dites *Majeures*, car c'est seulement ce type de gamme que nous avons enregistré pour construire notre base.

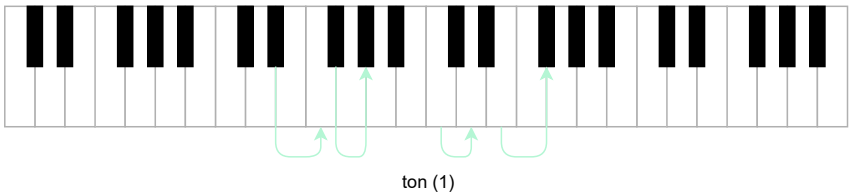

FIGURE 4.3 – Illustration du ton sur le clavier d'un piano

Commençons par prendre la gamme Majeure la plus connue, même par les non-initiés, sans qu'ils ne le sachent forcément. Il s'agit de la gamme Do Majeur : *Do Ré Mi Fa Sol La Si Do*. Comme nous pouvons l'observer avec cet exemple, une gamme est composée de 8 notes, ici naturelles mais pouvant tout aussi bien être altérées, et le nom de la gamme est donné à partir de la première de 8 notes. En étudiant la construction de cette gamme Do Majeur, notamment les demi-tons et tons séparants les différentes notes, nous pouvons extraire facilement la règle et motif de construction d'une gamme majeur.

Ainsi, nous pouvons remarquer que pour construire une gamme, il suffit de prendre n'importe quelle première note sur le piano, naturelle comme altérée, et de déterminer ensuite les 7 notes suivantes qui composeront sa gamme majeure. Ces 7 notes sont déterminées en suivant la suite suivante : 1 ton, 1 ton, 1/2 ton, 1 ton, 1 ton, 1 ton et 1/2 ton, en commençant bien sûr par la note choisie au départ et en appliquant les intervalles au fur et à mesure des notes extraites, nous pouvons observer ce motif sur la gamme de Do Majeur (voir Figure 4.4).

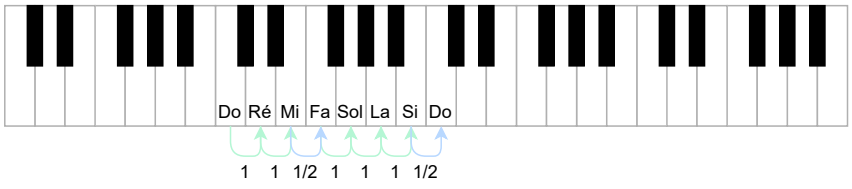

FIGURE 4.4 – Illustration de la gamme Do Majeur

Ainsi, si nous souhaitons construire la gamme Ré Majeur, il suffit de partir de la note *Ré* et en appliquant la suite d'intervalle exposée plus haut, nous obtenons la

gamme Ré Majeur : *Ré Mi Fa♯ Sol La Si Do♯ Ré* (voir Figure 4.5 pour l'illustration des intervalles).

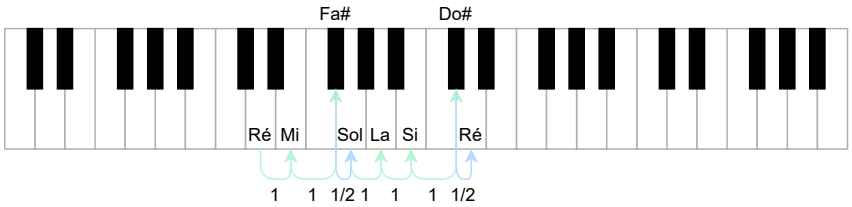

FIGURE 4.5 – Illustration de la gamme Ré Majeur

Regardons dès à présent comment la gestuelle est importante dans l'exécution d'une gamme au piano.

4.2.2 Gestuelle d'exécution des gammes

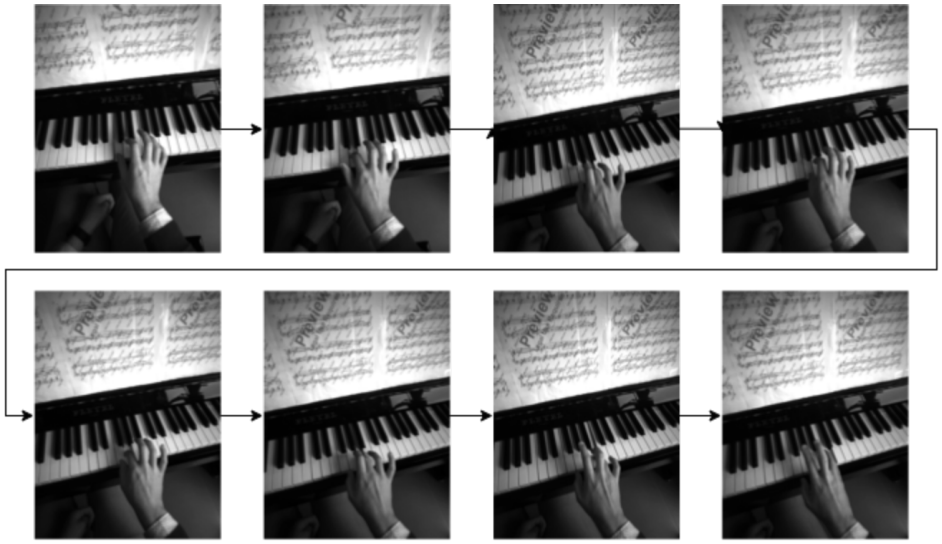

FIGURE 4.6 – Gestuelle gamme Do Majeur main droite

En effet, l'exercice de gamme demande une gestuelle, d'un point de vue académique, très précise du joueur, et différente selon la main effectuant la gamme. Si nous nous plaçons dans le cas où l'utilisateur joue la gamme de manière ascendante, du plus grave au plus aiguë (donc de la gauche vers la droite sur le clavier), pour le cas de la main droite, les notes sont jouées dans cet ordre avec les doigts : pouce - index - majeur - pouce - index - majeur - annulaire - auriculaire. Nous pouvons

remarquer que lors de la 4ème note le pouce revient jouer pour pouvoir ensuite dérouler le reste de la main sur les dernières notes. Dans le cas où nous jouerions la gamme de manière descendante, ce serait exactement l'ordre inverse qui serait dès lors appliqué (voir Figure 4.6 pour un exemple de la gamme Do Majeur jouée de la main droite).

Pour ce qui est de la main gauche, la gestuelle est similaire mais différentes dans l'ordre des doigts, toujours dans le cas ascendant : auriculaire - annulaire - majeur - index - pouce - majeur - index - pouce. Dans ce cas, la main est d'abord déroulée avant de ramener le majeur pour finir les trois dernières notes. De la même manière, dans le cas descendant, l'ordre est exactement inverse (voir Figure 4.7 pour un exemple de la gamme Ré Majeur jouée avec la main gauche).

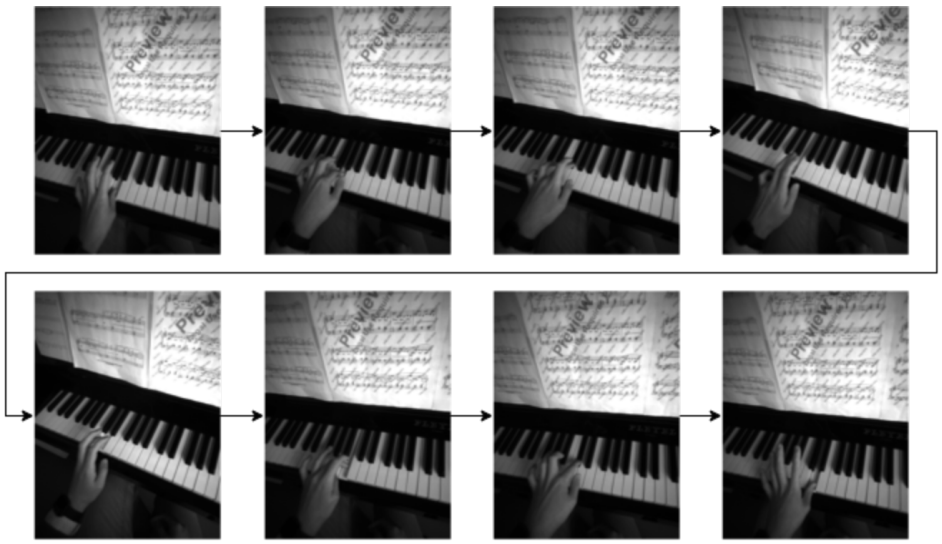

FIGURE 4.7 – Gestuelle gamme Ré Majeur main gauche

Étudions désormais comment nous avons pu acquérir les séquences composant notre base ainsi que les différentes classes qui puissent leur être associées.

4.2.3 Acquisition et composition

En ce qui concerne l'acquisition pure des séquences vidéos composant la base FirstPiano, nous avons décidé de profiter du fait qu'il soit possible de directement extraire les données des différents capteurs composants le casque HoloLens. Ceci est

motivé par le fait que notre base a pour but d'entraîner un algorithme qui pourrait, par la suite, servir à une application en temps réel et en réalité augmentée. Ainsi, extraire les données directement depuis ces capteurs nous permet d'avoir exactement le même type, le même format et la même fréquence de données, que ce soit dans la base mais aussi plus tard leur d'extraction et de traitement en cas réel.

En tout, le casque HoloLens est équipé de 8 capteurs vidéos frontaux, à savoir un capteur couleur et un capteur de lumière ambiante en son centre, au dessus se trouve deux capteurs de profondeur, l'un à courte portée l'autre à longue et enfin, le capteur RGB est aussi entouré de 4 capteurs en niveau de gris, deux de chaque côté, permettant d'avoir une vision un peu plus périphérique (voir Figure 4.8).

FIGURE 4.8 – Description des capteurs présents sur le casque HoloLens (`https://docs.microsoft.com/fr-fr/hololens/hololens1-hardware`)

Dans le cadre de notre base et le contexte d'apprentissage du piano que nous avons choisi, nous ne pouvons pas utiliser le capteur couleur qui propose un champ de vision beaucoup trop réduit et qui empêche d'avoir un point de vue correct sur les mains à moins d'extrêmement baisser la tête lors de l'acquisition et ainsi opter pour une position qui n'est pas du tout naturelle pour un joueur du piano, expérimenté ou non. Cependant, afin de maximiser les informations que nous pouvons obtenir grâce au casque HoloLens, nous avons décidé de garder tous les autres capteurs à savoir le flux de profondeur courte portée, le flux de lumière ambiante et les 4 flux de niveau de gris périphériques. Notons que de ces 4 derniers flux, les deux capteurs les plus proches du centre permettent tout de même d'obtenir un point de vue intéressant du champ de vision, celui excentré à gauche permettant d'avoir un meilleur point de vue sur la main gauche et identiquement pour le capteur droit et

la main droite. Ces captures nous permettent ainsi, pour chacune des données de la base, de posséder 6 flux de données différents (voir Figure 4.9), indépendant les uns des autres, notamment le flux de profondeur qui pourrait se voir aussi éventuellement être utilisé à des fins d'extractions des squelettes des mains.

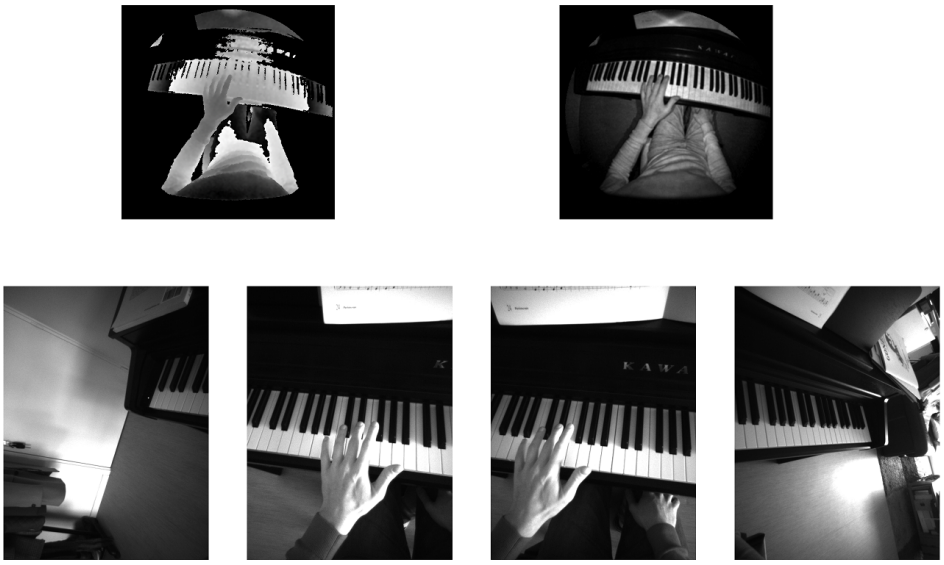

FIGURE 4.9 – Démonstration d'une même scène capturée par les 6 capteurs utilisés pour la base FirstPiano, dans l'ordre, le flux de profondeur courte portée, le flux de lumière ambiante et les 4 flux de niveau de gris périphériques

Nous avons évoqué plusieurs fois que les enregistrements pouvaient contenir soit une gamme bien exécutée, soit mal exécutée. Les gestuelles des séquences jouées correctement sont toutes identiques dans leur exécution comme précisé plus haut. Cependant, le contexte visuel (piano utilisé, main et vêtements de l'utilisateur, pièce dans laquelle l'enregistrement est fait, etc...), l'évolution du geste (rapidité et aisance) et le positionnement sur le piano (octave sur laquelle la gamme est jouée) peuvent varier afin d'ajouter de la diversité dans la base.

Concernant les gammes mal exécutées, en plus des mêmes variations que celles explicitées précédemment, elles ont été enregistrées de sorte que différentes erreurs peuvent apparaître :

— **Mauvaise gestuelle** : l'exécution peut être effectuée via un mauvais geste, notamment par l'ordre des doigts pour jouer les notes de la gamme.

— **Mauvaises notes** : la gamme peut être jouée avec la bonne gestuelle et le bon nombre de notes mais avec certaines ne faisant pas partie de la gamme majeure identifiée par la première note jouée, notamment via des écarts de demi-tons ou de tons trop grands ou trop petits entre certaines notes.

— **Notes en trop et/ou manquantes** : la gamme peut aussi être exécutée avec le bon geste mais cette fois-ci avec des notes jouées en trop ou en moins.

Ces trois différentes erreurs peuvent être associées au sein d'une même séquence permettant ainsi de produire une plus grande diversité du jeu de données et même possiblement de permettre à un algorithme de déterminer plus exactement quelle erreur précisément a été produite par le joueur et ainsi pouvoir avoir une application pouvant informer l'utilisateur, à l'aide de retours visuels, exactement quelle erreur il a effectué et comment la corriger.

Notons enfin que pour chacune des gammes et pour chacun des deux cas de bonne et mauvaise interprétation, 4 différentes gestuelles sont proposées. Pour chacune de la main gauche et de la main droite nous avons enregistré une pratique ascendante de la gamme ainsi que ascendante puis descendante (voir Figure 4.10 pour la hiérarchie de la base FirstPiano).

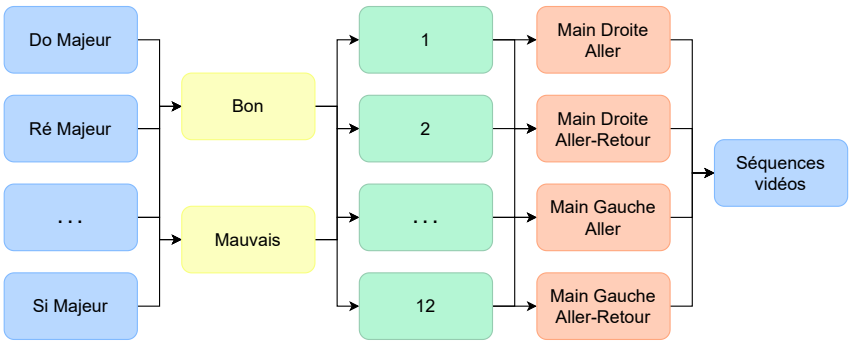

FIGURE 4.10 – Arbrorescence actuelle de notre base FirstPiano

Enfin, nous proposons ainsi, pour chaque gamme majeure, 12 enregistrements contenant chacun une interprétation bonne et une interprétation mauvaise de la gamme effectuée avec la main gauche et la main droite de manière ascendante ainsi que ascendante puis descendante pour un total de 672 données contenant chacune 6 vidéos des différents capteurs extraits du casque HoloLens. Le tout est enregistré à

l'aide de 4 personnes différentes dans 2 environnements différents. En ce qui concerne la labellisation, chacune des données se voit attribuer un nombre qui peut être converti en un vecteur composé de 5 éléments (zéro ou un) où le premier élément représente si la gamme est jouée correctement, et les autres éléments représentent les différentes erreurs possibles explicitées plus tôt. Plus précisément, le deuxième élément est à 1 si la mauvaise gestuelle est exécutée, le troisième si de mauvaises notes sont jouées, le quatrième si des notes sont oubliées et enfin le cinquième si des notes sont jouées en trop. Nous proposons ainsi en tous jusqu'à 16 labels différents en fonction de la juxtaposition d'erreurs possible (voir Figure 4.11).

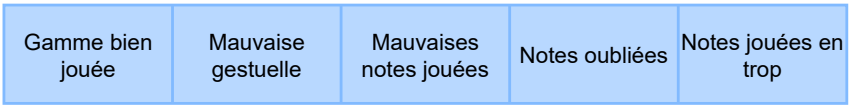

FIGURE 4.11 – Vecteur de labellisation utilisée, si la condition d'une case est remplie, alors la valeur attribuée est 1, sinon 0

4.3 Expérimentations sur la reconnaissance d'action

Afin de démontrer l'applicabilité de notre base mais aussi de prouver que les enregistrements qu'elle contient sont cohérents, nous avons décidé d'appliquer, avec différentes configurations et protocoles d'entraînements, notre méthode 2D Deep Video Capsule Network. Ceci nous permet aussi d'explorer un peu plus loin le module de décalage implémenté pour les capsules de notre réseau et d'étudier plus en profondeur ses éventuels bénéfices. Divers expérimentations ont été menées, mais nous souhaitons tout d'abord exposer les conditions et le pré-traitement des données d'entraînement et de tests car ils sont communs à toutes les configurations. Précisons que nous avons découpé notre base de données de sorte qu'environ 25% des séquences soient regroupées dans le groupe de test pour un total de 176 séquences contre 496 séquences d'entraînement. Nous fournissons bien évidemment ce découpage dans les méta-données de notre base afin que tout entraînement et test d'algorithmes sur celle-ci soient fait dans des conditions similaires afin que les résultats soient comparables.

A l'instar des entraînements effectués sur les bases First Person Hand Action [1] et Dynamic Hand Gesture [46], les séquences vidéos de notre base FirstPiano ont subi différentes opérations de pré-traitement dessus. Les données subissent d'abord une étape d'échantillonnage nous permettant de les réduire à des séquences de 32 images. Pour les données d'entraînement, l'échantillonnage est produit de manière aléatoire et irrégulière ne prenant pas en compte certaines images et en ayant un pas différent de sorte que, d'une epoch à l'autre, les échantillonnages d'une même séquence soient différents et ainsi offrir plus de diversité à l'algorithme pour apprendre et converger vers un modèle plus stable. Toujours dans cette optique, des opérations d'augmentation de données sont aussi effectuées, sachant que toutes les images d'une même séquence échantillonnée subissent exactement les mêmes modifications comme par exemple un changement de définition, une rotation ou encore une légère translation. Au moment des phases de test, et afin que les conditions soient exactement les mêmes pour pouvoir ainsi comparer de la même manière les différentes architectures entraînées, les données de test subissent un échantillonage régulier et similaire et ne se voient pas être transformées via les opérations d'augmentation. Durant l'entraînement, chacune des images de la séquence se voit attribuer une classe, sachant que toutes les images d'une même séquence sont sous le même label, mais toutes sont considérées au moment du calcul du gradient et de la rétropropagation afin de faire converger au mieux le réseau. Au moment de la phase de test, ce qui nous importe, c'est avant tout la classification de la séquence complète et non de chaque image. Cela ne doit pas être un problème si, au milieu d'une séquence parfaitement classifiée, une seule ou quelques images n'ont pas eu le bon label d'attribué, c'est pour cela que durant la phase de test, nous avons choisi de classifier seulement les séquences complètes en lui attribuant le label que la majorité des images composant la séquence se sont vues attribuées.

Nous avons ainsi utilisé ces conditions d'entraînement et de test afin d'évaluer différentes expérimentations sur notre base FirstPiano à l'aide de notre méthode 2D Deep Video Capsule Network (voir Chapitre 3).

4.3.1 Reconnaissance binaire de gammes

Utilisation d'un unique flux de niveau de gris

Notre première idée a été de simplifier au maximum le problème en décidant de ne considérer que deux classes au sein de notre base et ainsi demander à notre méthode d'apprentissage profond de ne discerner uniquement les séquences où une gamme était bien jouée de celles où l'exécution est mauvaise, peu importe la gamme jouée et peu importe les erreurs effectuées. Une telle approche est motivée par le fait que, dans ce cas de figure, l'architecture peut se concentrer uniquement sur la gestuelle et sur le nombre de notes jouées et sur le motif de celles-ci. Cette approche permet au réseau de se concentrer sur un objectif plus restreint et il serait possible, après avoir classifié une séquence comme étant une gamme bien jouée, d'envoyer cette même séquence à un autre réseau qui aura dès lors, quant à lui, uniquement appris à différencier les gammes bien jouées. Cela permettrait ainsi de pouvoir coder des méthodes moins profondes car concentrées sur un objectif plus restreint et moins compliqué. Elles n'ont pas besoin de calculer en parallèle car le traitement de l'une se déroule après celle de l'autre et ainsi, elles pourraient être déployées ensemble sur un casque de réalité augmentée.

Utilisation de deux flux de niveau de gris

Lors de nos tous premiers tests, afin de réduire au maximum la complexité du problème et afin d'observer s'il était seulement possible d'obtenir des résultats acceptables sur notre base, nous avions décidé d'envoyer au réseau le flux du bon capteur en niveau de gris, respectivement droite lorsque la gamme était jouée par la main droite et capteur de gauche lorsque jouée par la main gauche. Les résultats se sont directement trouvés être très encourageants donc, afin de proposer une méthode plus généralisable qui pourrait analyser la gamme jouée peu importe la main, les premières réelles expérimentations ont été effectuées en envoyant au réseau les deux flux côte à côté (voir Figure 4.12).

FIGURE 4.12 – Visualisation des données envoyées au réseau lors de l'utilisateur de deux capteurs de la base

Bien évidemment, là aussi, les deux flux subissent le même échantillonage et les mêmes transformations pour rester dans un contexte cohérent que pourrait retrouver le réseau dans un cas réel. Les résultats obtenus dans cette configuration sont cohérents avec ceux exposés au moment des tests de notre méthode 2D Deep Video Capsule Network sur d'autres bases (voir Chapitre 3.2.4). C'est-à-dire que l'application du module de décalage temporel appliqué sur les premiers filtres de toutes les capsules améliore de manière significative les résultats par rapport à la non application d'un tel module (voir Tableau 4.1). Cela s'explique naturellement par le caractère temporel de l'évolution du geste lors de l'exécution d'une gamme mais aussi et surtout par l'enchaînement des notes qui sont de caractère particulier à chaque gamme et qui permet ainsi aux réseaux à distinguer une bonne gamme, qui a toujours le même enchaînement de notes parmi l'une des 7 gammes majeures retrouvées, par rapport à une gamme mal jouée.

Utilisation du flux de profondeur courte portée

Des résultats similaires peuvent être observés lorsque nous envoyons cette fois à notre architecture uniquement le flux de profondeur. Généralement plus informatif, car, en plus de pouvoir être interprété visuellement à la manière d'une image de niveau de gris, la valeur de ces pixels renseignent non pas un niveau de gris permettant

Décalage	Taux de reconnaissance (%)
1^{ers} filtres des $1^{ères}$ capsules	75.71
Tous les filtres des $1^{ères}$ capsules	76.26
1^{ers} filtres de toutes capsules	**78.56**
Sans décalage	76.13

TABLE 4.1 – Comparaison de notre méthode 2D Deep Video Capsule Network selon les différents décalages utilisés sur notre base First-Piano en considérant 2 labels et en utilisant 2 flux vidéos

la cohérence visuelle, mais une valeur plus ou moins grande indiquant la profondeur de l'objet associé au pixel au moment de la capture vidéo et par rapport à l'origine du capteur. Cette donnée peut être théoriquement utile pour analyser plus précisément si une touche du clavier de l'instrument est enfoncée ou non. En effet, le niveau de gris ne profite pas de la même profondeur de contraste que peut offrir une image en couleur, notamment autour des touches noires qui permettent très difficilement de proposer un ombrage différent lorsque la touche est abaissée ou non. Le flux de profondeur permet théoriquement de s'affranchir de cette contrainte en permettant à la méthode d'apprentissage profond d'avoir une connaissance plus précise de la distance de la touche par rapport au capteur et par rapport à ces touches voisines pour identifier si une touche est actuellement abaissée ou non. Les résultats sont similaires mais un peu moins précis que lors de l'utilisation des deux capteurs en niveau de gris. Le cas de l'implémentation du décalage sur les premiers filtres de toutes les capsules offrent encore une amélioration considérable des résultats (voir Tableau 4.2).

Décalage	Taux de reconnaissance (%)
1^{ers} filtres des $1^{ères}$ capsules	69.14
Tous les filtres des $1^{ères}$ capsules	69.91
1^{ers} filtres de toutes capsules	**73.33**
Sans décalage	70.57

TABLE 4.2 – Comparaison de notre méthode 2D DVCN selon les différents décalages utilisés sur notre base FirstPiano en considérant 2 labels et en utilisant le flux de profondeur uniquement

Utilisation des trois flux

Enfin, et toujours dans le cas où nous demandons à notre architecture de ne classifier que deux labels, nous avons décidé d'assembler ces trois flux de données en une seule image (assemblés pour chacun des instants de la séquence donc) pour ainsi offrir un maximum d'informations, cohérentes entre elles, à notre réseau (voir 4.13). Dans ce cas de figure, nous pouvons remarquer que les résultats sont extrêmement similaires à ceux du cas où nous n'avions envoyé que le flux de profondeur au réseau, c'est-à-dire avec de moins bonnes précisions que lors de l'envoi des deux vidéos en niveau de gris (voir Tableau 4.3). Étant donné que nous envoyons aussi ces informations à notre réseau dans ce cas, en plus de l'information de profondeur, nous pouvons déduire que les trois flux assemblés donnent une dimensionnalité trop grande de données d'entrée pour le réseau. Que ce soit d'un point de vue du nombre de paramètres à gérer par le réseau, possiblement trop faible pour pouvoir converger correctement vers une solution convenable ou bien plus naturellement le fait que l'architecture des capsules ait du mal, via leurs opérations de convolution, à correctement analyser les trois images non pas envoyées en parallèle au réseau mais réunies côte à côte. Les applications du module de décalage temporel offrent toujours une amélioration de la précision de classification, cependant, c'est cette fois-ci le décalage de l'entièreté des filtres des premières capsules qui propose la meilleure classification.

Décalage	Taux de reconnaissance (%)
1^{ers} filtres des $1^{ères}$ capsules	70.14
Tous les filtres des $1^{ères}$ capsules	**73.56**
1^{ers} filtres de toutes capsules	72.54
Sans décalage	69.14

TABLE 4.3 – Comparaison de notre méthode 2D DVCN selon les différents décalages utilisés sur notre base FirstPiano en considérant 2 labels et en utilisant 3 flux vidéos

FIGURE 4.13 – Visualisation des données envoyées au réseau lors de l'utilisateur de trois des capteurs de la base

4.3.2 Reconnaissance multiclasse de gammes

Utilisation de deux flux

Suite à ces premières expérimentations concluantes, nous avons décidé de complexifier le problème en demandant cette fois-ci à notre méthode d'apprentissage profond de classifier huit labels au lieu de deux. Parmi ces huit labels, nous retrouvons toujours un label qui représente une gamme mal jouée, peu importe sa nature, tandis que les sept autres classes sont associées à chacune des gammes majeures correctement interprétées. La non-dissociation des gammes mal jouées est motivée par le fait que, pour une gamme très mal exécutée, il peut être difficile voire inutile d'essayer d'identifier la gamme, le plus facile étant sûrement de simplement identifier la première note jouée car différente et unique entre toutes les gammes majeures et servant de point de repère fixe. Il serait dès lors plus facile, une fois la mauvaise interprétation de gamme identifiée, d'extraire l'image contenant la première note jouée et de simplement l'analyser, grâce au motif du piano, pour l'identifier, retrouver la gamme associée et ainsi indiquer des retours d'informations à l'utilisateur en conséquence dans le cas d'une application en réalité augmentée. Par exemple en affichant en modèle 3D démontrant la bonne gestuelle à effectuer pour jouer cette gamme.

A l'instar du cas binaire précédemment étudié, nous avons commencé par une série d'expérimentations en envoyant à notre réseau 2D DVCN les deux images de niveau de gris périphériques centre gauche et centre droit. Dans la continuité des

résultats précédemment obtenus c'est cette fois aussi l'implémentation où le décalage est appliqué sur tous les filtres des premières capsules qui nous offre la meilleur classification. Cependant, l'implémentation du décalage des premiers filtres de toutes les capsules propose des résultats extrêmement similaires. Nous pouvons dénoter bien sûr une dégradation conséquente de la qualité de la classification, de l'ordre de 18% par rapport aux meilleurs résultats dans les conditions de classification binaire. Cette dégradation s'explique naturellement par le fait que le problème est considérablement complexifié car la méthode doit désormais se concentrer obligatoirement sur les notes jouées et non plus uniquement sur la gestuelle pratiquée pour pouvoir différencier les différentes gammes bien jouées de la base.

Décalage	Taux de reconnaissance (%)
1^{ers} filtres des $1^{ères}$ capsules	59.09
Tous les filtres des $1^{ères}$ capsules	**61.93**
1^{ers} filtres de toutes capsules	61.36
Sans décalage	60.20

TABLE 4.4 – Comparaison de notre méthode 2D DVCN selon les différents décalages utilisés sur notre base FirstPiano en considérant 8 labels et en utilisant 3 flux vidéos

Utilisation des trois flux

Lorsque nous essayons de classifier les séquences en une des 8 classes en envoyant au réseau les trois flux vidéos en niveau de gris et de profondeur, nous obtenons des résultats similaires à ceux du cas de deux labels dans le rapport entre l'utilisation de décalage temporel ou non, mais avec, naturellement, une précision bien moindre. En effet, notre architecture s'approche dans ce cas précis difficilement des 60% de bonnes classifications (voir Tableau 4.5), un résultat qui reste cependant tout à fait honorable compte tenu de la réelle difficulté du contexte de la base où la méthode doit analyser une gestuelle extrêmement fine et précise et où elle ne peut pas s'aider d'indices visuels et spatiaux, notamment vis-à-vis de l'environnement, pour pouvoir différencier les classes entres elles, à l'inverse de beaucoup d'autres bases d'actions

où, par le sport pratiqué ou encore l'objet manipulé, le contexte associé offre de nombreux indices visuels autres que la gestuelle pratiquée par les mains ou le corps humain.

Décalage	Taux de reconnaissance (%)
1^{ers} filtres des $1^{ères}$ capsules	57.23
Tous les filtres des $1^{ères}$ capsules	**59.45**
1^{ers} filtres de toutes capsules	56.46
Sans décalage	56.89

TABLE 4.5 – Comparaison de notre méthode 2D DVCN selon les différents décalages utilisés sur notre base FirstPiano en considérant 8 labels et en utilisant 3 flux vidéos

4.4 Conclusion

Nous avons proposé, au sein de ce chapitre, une description détaillée de notre nouvelle base de données d'actions de la main, FirstPiano, où les actions évoluent dans un contexte purement musical d'apprentissage du piano. Notre base FirstPiano offre pour la première fois un contexte bien précis qui propose une utilisation réelle et concrète des méthodes entraînées dessus dans des cas d'application en temps réel, notamment sur casque de réalité augmentée ou virtuelle, afin de développer des systèmes d'assistance gestuelle personnalises, dans ce cas précis, d'aide à l'apprentissage du piano. FirstPiano propose pour cela, pour le moment, un ensemble de séquences vidéos permettant d'identifier et de différencier des exercices de gammes majeures jouées sur le clavier. Ces données sont de plus directement extraites à partir des capteurs intégrés au casque de réalité augmentée HoloLens ce qui permet de garder une cohérence lors d'une utilisation en cas réel où les données récupérées seront identiques. Cela permet ainsi de proposer une base native sans avoir à développer des systèmes complexes et très particulier, où des caméras externes doivent être accrochées de manière très peu pratique au niveau du front de l'utilisateur et reliée à une ordinateur externe. Dans notre base FirstPiano, tout est inclus et peut être extrait du casque RA.

Cette base a aussi pour ambition de se voir progressivement être agrandie et améliorée, que ce soit en rajoutant de nouvelles données dans les exercices déjà existants et pour ajouter de la diversité, notamment dans les erreurs possibles à exécuter. Néanmoins, notre vision sur le long terme a pour ambition de proposer aussi tout une panoplie de nouveaux exercices comme ceux de dissociation de jeux de la main gauche et de la main droite, que ce soit dans le rythme, les différents doigts utilisés ou encore la force de la pression exercée sur les touches. Une autre amélioration plus intéressante pour un futur proche serait de proposer aussi, pour les séquences actuellement enregistrées, d'ajouter des degrés dans la labellisation afin de préciser exactement, pour les gammes mal interprétées, où se situe exactement l'erreur et ainsi permettre l'entraînement de méthodes encore plus précises qui, une fois associées à une application en temps réel, pourrait proposer des retours beaucoup plus instructifs à l'utilisateur, comme par exemple lui indiquer où, quand et quelle erreur il a commise afin de l'aider au mieux dans son apprentissage de l'instrument.

Conclusion

Résumé

Au cours de cette thèse, nous nous sommes penchés sur le problème de développement de système d'assistance gestuelle personnalisée, en combinant les technologies de vision par ordinateur et celles de réalité augmentée.

Pour cela, nous avons tout d'abord réfléchi à comment ces solutions et méthodes de reconnaissance et d'identification de gestes et d'actions, plus particulièrement de la main doivent être développées. En effet, les systèmes de réalité augmentée sont autonomes et demandent dès lors de nouveaux systèmes d'interactions, dans un premier temps au moins pour naviguer dans les menus. Leur caractère autonome se joint à leur motivation de vouloir être portatif et, à terme, devenir tellement léger et discret qu'ils puissent être emmenés partout et ainsi pouvoir proposer des systèmes d'assistances dans grands nombres de cas de la vie de tous les jours. Dès lors, il devient logique de se dire qu'il est déjà absurde de s'imaginer devoir transporter et ainsi communiquer avec ces casques à l'aide de souris ou clavier à la manière d'un ordinateur, et même si nous retrouvons aujourd'hui de petites manettes ou cliqueurs, ces dispositifs seront aussi déjà trop invasifs et peu pratiques pour un futur proche où la réalité augmentée se voudra totalement immersive. C'est pourquoi la volonté de développer des algorithmes de reconnaissance des gestes et d'actions de la main se fait de plus en plus impérieux et motive de nombreuses recherches. Cependant, la plupart des méthodes proposées ne prennent pas en compte une contrainte fondamentale de ces dispositifs de réalité augmentée, si nous ne souhaitons pas utiliser un moyen d'interaction externe autre que ses propres mains, il serait absurde de se

laisser convaincre qu'il faille aussi la puissance d'un ordinateur externe pour pouvoir exécuter ses algorithmes de reconnaissance. Or, la grande majorité des recherches se concentrent, à raison aussi, sur l'amélioration de la qualité et précision de reconnaissance, mais bien moins sur la recherche de méthodes plus légères qui pourraient être appliquées telles quelles sur ces appareils autonomes.

C'est dans cette optique que les recherches menées durant cette thèse ont permis de développer une nouvelle méthode de reconnaissance de gestes et d'actions, 2D Deep Video Capsule Network. Ainsi, en plus de proposer pour la première fois un réseau qui utilise l'architecture de réseau neuronal à capsules 2D [81] à des fins d'analyse vidéos, la méthode permet en plus de cela de proposer une solution légère et portative avec des résultats au dessus ou au moins comparables à ceux de l'état de l'art. Un autre bénéfice étant que cette méthode demande jusqu'à quarante cinq fois moins de paramètres à gérer. L'architecture de capsules et sa puissance structurelle nous permet de produire une analyse spatiale plus complète et complexe avec bien moins de paramètres qu'un réseau de convolution classique en compensant par un algorithme de routing dont l'impact ne se produit qu'au moment de l'entraînement. L'analyse temporelle de la vidéo se fait via l'implémentation du module de décalage temporel [61] appliqué aux capsules, le tout, sans ajout de calculs si ce n'est le déplacement et la recopie de données au moment de l'étude d'une image à l'autre. Notre méthode répond ainsi aux contraintes de pouvoir proposer une solution de reconnaissance de gestes et d'actions de la main qui pourrait être déployée sur un appareil autonome de réalité augmentée grâce à sa légèreté.

Du point de vue applicatif pur, une nouvelle limitation s'est offerte à nos yeux, celle de l'absence totale de bases de données qui proposent un contexte d'actions clair, précis et limité et qui permettrait ainsi d'offrir des cas d'utilisation réel un minimum intéressant à développer. En effet, toutes les bases de données actuelles sur la reconnaissance d'actions de la main, la très grande majorité en vue à la première personne, proposent toutes une grande diversité d'actions, souvent orientées autour de la manipulation d'objets du quotidien. Cependant, ces contextes ne permettent d'afficher en retour sur les écrans transparents du casque de réalité augmentée que

porte l'utilisateur, uniquement une phrase indiquant quel objet il manipule, ce qui n'a pas grand intérêt et qui revient simplement à tester la méthode entraînée sur la base. Elles n'ont ainsi pas pour motivation de permettre le développement d'une réelle application qui saurait s'en détacher.

C'est pourquoi nous proposons, toujours dans le cadre de cette thèse, le développement de notre base FirstPiano, composée de séquences vidéos capturant des interprétations d'exercices de gammes majeures jouées au piano. La pratique et l'apprentissage d'instruments constituent un domaine riche en possibilités d'application mais pour le moment encore sous-exploité dans la littérature. Fort d'être en cohérence avec l'utilisation d'un casque de réalité augmentée qui propose naturellement un point de vue à la première personne, le piano s'impose comme l'un des rares instruments qui permet, depuis ce point de vue particulier, de pouvoir observer très clairement et précisément les mains interagir avec l'instrument, sans occlusion et sans demander une position peu naturelle comme le demanderait une guitare par exemple. C'est à l'aide des capteurs directement intégrés au casque de réalité augmentée HoloLens que nous avons ainsi pu extraire 6 flux vidéos synchronisés, que ce soit en niveau de gris ou de profondeur et avec un point de vue plus ou moins panoramique du champ de vision de l'utilisateur. Un total de 672 enregistrements sur 4 personnes différentes sont proposés et qui sont répartis sur un maximum de 16 labels qui permettent de classifier les 7 gammes majeures bien pratiquées mais aussi les différentes combinaisons d'erreurs possibles lorsqu'elles sont mal jouées. Ce contexte très précis offre une base de données au problème de reconnaissance d'actions relativement difficile, car l'environnement et l'interaction avec l'unique objet qu'est le clavier du piano ne propose que très peu d'indices visuels associés à chaque classe. Cela permet cependant de pouvoir développer des applications utilisant la méthode entraînée sur FirstPiano pour proposer un système d'assistance à l'apprentissage du piano en pouvant indiquer à l'utilisateur s'il pratique correctement ou non ses gammes, et pouvoir aussi lui proposer des aides de corrections et de bonnes pratiques en temps réel.

Travaux futurs

Bien évidemment, le domaine du développement d'assistance gestuelle personnalisée en réalité augmentée et à l'aide de vision par ordinateur est extrêmement jeune et complexe et demandera encore de nombreuses années de recherches avant de pouvoir proposer des solutions fiables, précises et stables. La partie reconnaissance de gestes et d'actions est un domaine relativement ancien, fort de déjà plusieurs dizaines d'années de recherches et qui commence à avoir des résultats réellement prometteurs, notamment avec l'arrivée récente des méthodes d'apprentissage profond. Cependant, il faut que de plus en plus de recherches se penchent vers le développement de solutions légères et portables comme nous pouvons le retrouver dans d'autres domaines à l'instar de la reconnaissance vocale sur smartphone. En effet, ces solutions peuvent aujourd'hui être directement déployées et utilisées localement sur les appareils de l'utilisateur sans avoir à envoyer la requête d'identification de la séquence vocale à un serveur externe. Cependant, l'apprentissage profond est un domaine prolifique et en constante évolution, proposant régulièrement de nouvelles architectures combinant les recherches et avancées des uns et des autres et un peu plus rarement de nouvelles architectures au fonctionnement structurelle profond radicalement différent mais proposant une nouvelle approche dans la compréhension de ces algorithmes à la résolution de problèmes bien particulier. L'un des derniers en date, le réseau neuronal à capsules [81] pour l'étude d'images en est une démonstration récente. Si ces méthodes sont encore loin de se comparer à la performance d'un être humain, la puissance exponentielle de calculs des ordinateurs, la proposition et l'arrivée de bases de données de plus en plus diversifiées et importantes en nombre de données ainsi que la multitude de solutions proposées, que ce soit directement par l'étude d'images et de flux vidéos avec les opérations de convolution ou de capsules, ou encore l'étude du squelette avec les réseaux récurrents ou d'analyse de graphes, devraient bientôt permettre de surpasser les êtres humains. Actuellement, l'étude de séquences d'actions à partir de flux vidéos semble être l'approche la plus raisonnable à pouvoir être appliquée localement sur un appareil de réalité augmentée, que

ce soit par le fait que ce soit les données les plus facilement accessible et qui requiert un minimum de pré-traitement, que par le fait que les architectures travaillant sur ces données sont naturellement légères. Cependant, l'arrivée récente de technologie portable permettant d'extraire des données beaucoup plus précises et informatives ont fait leur apparition dernièrement, comme les caméras Intel Real Sense ou Magic Leap. Si ces technologies pouvaient se voir embarquer directement au sein de casques RA et RV, elles permettraient d'avoir un accès immédiat à ce type de données, sans pré-traitement lourd sur des données de couleurs ou de profondeur comme actuellement, et pourraient ainsi réorienter la recherche vers des architectures utilisant ce type de données en entrée, souvent plus précises que celles utilisant simplement des flux vidéos.

L'étape d'après, celle d'application d'interaction homme-machine va naturellement et abondamment suivre. Elle est déjà explorée mais encore jeune, et si nous proposons une toute nouvelle base principalement orientée et motivée vers cette idée d'interaction, notamment via la possibilité de développer des systèmes d'assistance gestuelle personnalisés dessus, elle mérite encore un certain approfondissement, que ce soit dans sa labellisation pour faciliter voire permettre aux méthodes de produire des retours très précis sur les erreurs effectuées. Mais que ce soit aussi de compléter notre base pour, en restant dans le contexte d'apprentissage du piano, de diversifier considérablement le problème en proposant plus que la reconnaissance du simple exercice de pratique de gamme majeure, en espérant que notre apport motive par la suite d'autres chercheurs à produire des travaux similaires et ainsi diversifier, complexifier et compléter le domaine de la reconnaissance de gestes et d'actions orienté vers la réalité augmentée. La partie application se mêlera naturellement avec le champ de recherche de l'interaction homme-machine afin de déterminer quels moyens et interfaces de communication gestuelle ainsi que de retours d'informations mettre en place pour offrir les expériences les plus naturelles et intuitives possible pour l'utilisateur. Les recherches et avis de ces experts seront d'une aide précieuse, notamment pour orienter au mieux la complétion et amélioration de notre base de

donnée pour qu'elle soit le plus utile possible à l'élaboration d'algorithmes de classification performant qui seront à leur tour utile à leurs recherches d'interactions homme-machine.

Références

[1] G. Garcia-Hernando, S. Yuan, S. Baek, and T.-K. Kim, "First-person hand action benchmark with rgb-d videos and 3d hand pose annotations," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 409–419, 2018. [Cité en pages 4, 20, 21, 22, 73, 87, 89, 91 et 107]

[2] Q. De Smedt, H. Wannous, J.-P. Vandeborre, J. Guerry, B. L. Saux, and D. Filliat, "3d hand gesture recognition using a depth and skeletal dataset : Shrec'17 track," in *Proceedings of the Workshop on 3D Object Retrieval*, 3Dor '17, (Goslar, DEU), p. 33–38, Eurographics Association, 2017. [Cité en pages 4, 15, 16, 17 et 18]

[3] F. M. Caputo, S. Burato, G. Pavan, T. Voillemin, H. Wannous, J.-P. Vandeborre, M. Maghoumi, E. Taranta, A. Razmjoo, J. LaViola Jr, *et al.*, "Online gesture recognition," in *Eurographics Workshop on 3D Object Retrieval*, The Eurographics Association, 2019. [Cité en page 7]

[4] T. Voillemin, H. Wannous, and J.-P. Vandeborre, "2d deep video capsule network with temporal shift for action recognition," in *2020 25th International Conference on Pattern Recognition (ICPR)*, pp. 3513–3519, IEEE, 2021. [Cité en page 7]

[5] T. Voillemin, H. Wannous, and J.-P. Vandeborre, "Deep video capsule network avec décalage temporel pour la reconnaissance d'action," in *admis à COmpression et REprésentation des Signaux Audiovisuels (CORESA)*, CORESA, 2021. [Cité en page 7]

[6] H. Wannous, T. Voillemin, and J.-P. Vandeborre, "Continuous hand gesture

detection and recognition using deep coarse and fine hand features," in *en soumission à International Conference on 3D Vision (3DV)*, 3DV, 2021. [Cité en page 7]

[7] T. Voillemin, H. Wannous, and J.-P. Vandeborre, "Firstpiano : A new dataset oriented towards ar application," in *en soumission à International Conference on Image Analysis And Processing (ICIAP)*, ICIAP, 2021. [Cité en page 7]

[8] A. F. Bobick and J. W. Davis, "The recognition of human movement using temporal templates," *IEEE Transactions on pattern analysis and machine intelligence*, vol. 23, no. 3, pp. 257–267, 2001. [Cité en page 10]

[9] M. Oberweger, P. Wohlhart, and V. Lepetit, "Hands deep in deep learning for hand pose estimation," *arXiv preprint arXiv :1502.06807*, 2015. [Cité en pages 10, 25, 26 et 32]

[10] B. E. Boser, I. M. Guyon, and V. N. Vapnik, "A training algorithm for optimal margin classifiers," in *Proceedings of the fifth annual workshop on Computational learning theory*, pp. 144–152, 1992. [Cité en page 11]

[11] T. K. Ho, "Random decision forests," in *Proceedings of 3rd international conference on document analysis and recognition*, vol. 1, pp. 278–282, IEEE, 1995. [Cité en page 11]

[12] V. Athitsos, C. Neidle, S. Sclaroff, J. Nash, A. Stefan, Q. Yuan, and A. Thangali, "The american sign language lexicon video dataset," in *2008 IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops*, pp. 1–8, IEEE, 2008. [Cité en page 13]

[13] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "Imagenet : A large-scale hierarchical image database," in *2009 IEEE conference on computer vision and pattern recognition*, pp. 248–255, Ieee, 2009. [Cité en pages 14, 27 et 58]

[14] Z. Ren, J. Yuan, J. Meng, and Z. Zhang, "Robust part-based hand gesture recognition using kinect sensor," *IEEE transactions on multimedia*, vol. 15, no. 5, pp. 1110–1120, 2013. [Cité en pages 14, 15 et 18]

[15] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in neural information processing systems*, pp. 1097–1105, 2012. [Cité en pages 14, 27 et 58]

[16] S. Sridhar, A. Oulasvirta, and C. Theobalt, "Interactive markerless articulated hand motion tracking using rgb and depth data," in *Proceedings of the IEEE international conference on computer vision*, pp. 2456–2463, 2013. [Cité en page 14]

[17] G. Rogez, M. Khademi, J. Supančič III, J. M. M. Montiel, and D. Ramanan, "3d hand pose detection in egocentric rgb-d images," in *European Conference on Computer Vision*, pp. 356–371, Springer, 2014. [Cité en page 14]

[18] N. Pugeault and R. Bowden, "Spelling it out : Real-time asl fingerspelling recognition," in *2011 IEEE International conference on computer vision workshops (ICCV workshops)*, pp. 1114–1119, IEEE, 2011. [Cité en pages 15 et 18]

[19] A. Kurakin, Z. Zhang, and Z. Liu, "A real time system for dynamic hand gesture recognition with a depth sensor," in *2012 Proceedings of the 20th European signal processing conference (EUSIPCO)*, pp. 1975–1979, IEEE, 2012. [Cité en pages 15 et 18]

[20] H. Cheng, Z. Dai, and Z. Liu, "Image-to-class dynamic time warping for 3d hand gesture recognition," in *2013 IEEE International Conference on Multimedia and Expo (ICME)*, pp. 1–6, IEEE, 2013. [Cité en pages 15 et 18]

[21] L. Bretzner, I. Laptev, and T. Lindeberg, "Hand gesture recognition using multi-scale colour features, hierarchical models and particle filtering," in *Proceedings of fifth IEEE international conference on automatic face gesture recognition*, pp. 423–428, IEEE, 2002. [Cité en pages 15, 17 et 18]

[22] G. Marin, F. Dominio, and P. Zanuttigh, "Hand gesture recognition with leap motion and kinect devices," in *2014 IEEE International conference on image processing (ICIP)*, pp. 1565–1569, IEEE, 2014. [Cité en pages 15, 17 et 18]

[23] B. Feng, F. He, X. Wang, Y. Wu, H. Wang, S. Yi, and W. Liu, "Depth-projection-map-based bag of contour fragments for robust hand gesture recognition," *IEEE Transactions on Human-Machine Systems*, vol. 47, no. 4, pp. 511–523, 2016. [Cité en pages 15 et 18]

[24] Z. Lin, Z. Jiang, and L. S. Davis, "Recognizing actions by shape-motion prototype trees," in *2009 IEEE 12th international conference on computer vision*, pp. 444–451, IEEE, 2009. [Cité en pages 15 et 18]

[25] L. Liu and L. Shao, "Learning discriminative representations from rgb-d video data," in *Twenty-third international joint conference on artificial intelligence*, 2013. [Cité en pages 15 et 18]

[26] Y. Xu, Q. Wang, X. Bai, Y.-L. Chen, and X. Wu, "A novel feature extracting method for dynamic gesture recognition based on support vector machine," in *2014 IEEE International Conference on Information and Automation (ICIA)*, pp. 437–441, IEEE, 2014. [Cité en pages 15, 17 et 18]

[27] C. Wang, Z. Liu, and S.-C. Chan, "Superpixel-based hand gesture recognition with kinect depth camera," *IEEE transactions on multimedia*, vol. 17, no. 1, pp. 29–39, 2014. [Cité en pages 15 et 18]

[28] W. Lu, Z. Tong, and J. Chu, "Dynamic hand gesture recognition with leap motion controller," *IEEE Signal Processing Letters*, vol. 23, no. 9, pp. 1188–1192, 2016. [Cité en pages 15, 17 et 18]

[29] Y. Zhang, C. Cao, J. Cheng, and H. Lu, "Egogesture : A new dataset and benchmark for egocentric hand gesture recognition," *IEEE Transactions on Multimedia*, vol. 20, no. 5, pp. 1038–1050, 2018. [Cité en pages 15, 17 et 18]

[30] J. Materzynska, G. Berger, I. Bax, and R. Memisevic, "The jester dataset : A large-scale video dataset of human gestures," in *Proceedings of the IEEE/CVF International Conference on Computer Vision Workshops*, pp. 0–0, 2019. [Cité en pages 15, 17, 18 et 21]

[31] E. Ohn-Bar and M. M. Trivedi, "Hand gesture recognition in real time

for automotive interfaces : A multimodal vision-based approach and evaluations," *IEEE transactions on intelligent transportation systems*, vol. 15, no. 6, pp. 2368–2377, 2014. [Cité en pages 15 et 18]

[32] P. Molchanov, X. Yang, S. Gupta, K. Kim, S. Tyree, and J. Kautz, "Online detection and classification of dynamic hand gestures with recurrent 3d convolutional neural network," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 4207–4215, 2016. [Cité en pages 15, 18, 34 et 35]

[33] Q. De Smedt, *Dynamic hand gesture recognition-From traditional handcrafted to recent deep learning approaches*. PhD thesis, Université de Lille 1, Sciences et Technologies ; CRIStAL UMR 9189, 2017. [Cité en pages 16, 18, 32, 43 et 44]

[34] H. Kuehne, H. Jhuang, E. Garrote, T. Poggio, and T. Serre, "Hmdb : a large video database for human motion recognition," in *2011 International conference on computer vision*, pp. 2556–2563, IEEE, 2011. [Cité en page 19]

[35] K. Soomro, A. R. Zamir, and M. Shah, "Ucf101 : A dataset of 101 human actions classes from videos in the wild," *arXiv preprint arXiv :1212.0402*, 2012. [Cité en page 19]

[36] A. Fathi, X. Ren, and J. M. Rehg, "Learning to recognize objects in egocentric activities," in *CVPR 2011*, pp. 3281–3288, IEEE, 2011. [Cité en pages 20 et 22]

[37] I. M. Bullock, T. Feix, and A. M. Dollar, "The yale human grasping dataset : Grasp, object, and task data in household and machine shop environments," *The International Journal of Robotics Research*, vol. 34, no. 3, pp. 251–255, 2015. [Cité en pages 20 et 22]

[38] M. Cai, K. M. Kitani, and Y. Sato, "A scalable approach for understanding the visual structures of hand grasps," in *2015 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 1360–1366, IEEE, 2015. [Cité en pages 20 et 22]

[39] G. Rogez, J. S. Supancic, and D. Ramanan, "Understanding everyday hands in action from rgb-d images," in *Proceedings of the IEEE international conference on computer vision*, pp. 3889–3897, 2015. [Cité en pages 20 et 22]

[40] H. Pirsiavash and D. Ramanan, "Detecting activities of daily living in first-person camera views," in *2012 IEEE conference on computer vision and pattern recognition*, pp. 2847–2854, IEEE, 2012. [Cité en pages 20, 21 et 22]

[41] M. Moghimi, P. Azagra, L. Montesano, A. C. Murillo, and S. Belongie, "Experiments on an rgb-d wearable vision system for egocentric activity recognition," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pp. 597–603, 2014. [Cité en pages 20 et 22]

[42] S. Bambach, S. Lee, D. J. Crandall, and C. Yu, "Lending a hand : Detecting hands and recognizing activities in complex egocentric interactions," in *Proceedings of the IEEE International Conference on Computer Vision*, pp. 1949–1957, 2015. [Cité en pages 20 et 22]

[43] Y. Tang, Y. Tian, J. Lu, J. Feng, and J. Zhou, "Action recognition in rgb-d egocentric videos," in *2017 IEEE International Conference on Image Processing (ICIP)*, pp. 3410–3414, IEEE, 2017. [Cité en pages 20 et 22]

[44] R. Goyal, S. Ebrahimi Kahou, V. Michalski, J. Materzynska, S. Westphal, H. Kim, V. Haenel, I. Fruend, P. Yianilos, M. Mueller-Freitag, *et al.*, "The" something something" video database for learning and evaluating visual common sense," in *Proceedings of the IEEE International Conference on Computer Vision*, pp. 5842–5850, 2017. [Cité en pages 20, 21 et 22]

[45] Y. Li, M. Liu, and J. M. Rehg, "In the eye of beholder : Joint learning of gaze and actions in first person video," in *Proceedings of the European Conference on Computer Vision (ECCV)*, pp. 619–635, 2018. [Cité en pages 20, 21 et 22]

[46] Q. De Smedt, H. Wannous, and J.-P. Vandeborre, "Skeleton-based dynamic hand gesture recognition," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pp. 1–9, 2016. [Cité en pages 23, 73, 91, 92, 93 et 107]

[47] L. Fang, N. Liang, W. Kang, Z. Wang, and D. D. Feng, "Real-time hand posture recognition using hand geometric features and fisher vector," *Signal Processing : Image Communication*, vol. 82, p. 115729, 2020. [Cité en page 23]

[48] T. Brox, A. Bruhn, N. Papenberg, and J. Weickert, "High accuracy optical flow estimation based on a theory for warping," in *European conference on computer vision*, pp. 25–36, Springer, 2004. [Cité en pages 24 et 25]

[49] X. Zhou, Q. Wan, W. Zhang, X. Xue, and Y. Wei, "Model-based deep hand pose estimation," *arXiv preprint arXiv :1606.06854*, 2016. [Cité en page 25]

[50] L. Ge, Y. Cai, J. Weng, and J. Yuan, "Hand pointnet : 3d hand pose estimation using point sets," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 8417–8426, 2018. [Cité en page 26]

[51] T. Vodopivec, V. Lepetit, and P. Peer, "Fine hand segmentation using convolutional neural networks," *arXiv preprint arXiv :1608.07454*, 2016. [Cité en page 27]

[52] G. Serra, M. Camurri, L. Baraldi, M. Benedetti, and R. Cucchiara, "Hand segmentation for gesture recognition in ego-vision," in *Proceedings of the 3rd ACM international workshop on Interactive multimedia on mobile & portable devices*, pp. 31–36, 2013. [Cité en page 27]

[53] B. Kang, K.-H. Tan, N. Jiang, H.-S. Tai, D. Tretter, and T. Nguyen, "Hand segmentation for hand-object interaction from depth map," in *2017 IEEE Global Conference on Signal and Information Processing (GlobalSIP)*, pp. 259–263, IEEE, 2017. [Cité en page 27]

[54] S. Zha, F. Luisier, W. Andrews, N. Srivastava, and R. Salakhutdinov, "Exploiting image-trained cnn architectures for unconstrained video classification," *arXiv preprint arXiv :1503.04144*, 2015. [Cité en page 28]

[55] N. Neverova, C. Wolf, G. Taylor, and F. Nebout, "Moddrop : adaptive multi-modal gesture recognition," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 38, no. 8, pp. 1692–1706, 2015. [Cité en page 28]

[56] L. Wang, Y. Xiong, Z. Wang, Y. Qiao, D. Lin, X. Tang, and L. Van Gool, "Temporal segment networks : Towards good practices for deep action recognition," in *European conference on computer vision*, pp. 20–36, Springer, 2016. [Cité en page 29]

[57] B. Zhou, A. Andonian, A. Oliva, and A. Torralba, "Temporal relational reasoning in videos," in *Proceedings of the European Conference on Computer Vision (ECCV)*, pp. 803–818, 2018. [Cité en page 29]

[58] D. C. Luvizon, D. Picard, and H. Tabia, "2d/3d pose estimation and action recognition using multitask deep learning," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 5137–5146, 2018. [Cité en page 29]

[59] C. Szegedy, S. Ioffe, V. Vanhoucke, and A. Alemi, "Inception-v4, inception-resnet and the impact of residual connections on learning," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 31, 2017. [Cité en pages 29 et 58]

[60] D. Tran, H. Wang, L. Torresani, J. Ray, Y. LeCun, and M. Paluri, "A closer look at spatiotemporal convolutions for action recognition," in *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, pp. 6450–6459, 2018. [Cité en page 30]

[61] J. Lin, C. Gan, and S. Han, "Tsm : Temporal shift module for efficient video understanding," in *Proceedings of the IEEE International Conference on Computer Vision*, pp. 7083–7093, 2019. [Cité en pages 30, 73, 77, 91, 93 et 118]

[62] J. Weng, M. Liu, X. Jiang, and J. Yuan, "Deformable pose traversal convolution for 3d action and gesture recognition," in *Proceedings of the European Conference on Computer Vision (ECCV)*, pp. 136–152, 2018. [Cité en page 30]

[63] Y. Li, D. Ma, Y. Yu, G. Wei, and Y. Zhou, "Compact joints encoding for skeleton-based dynamic hand gesture recognition," *Computers & Graphics*, vol. 97, pp. 191–199, 2021. [Cité en page 30]

[64] Y. Du, W. Wang, and L. Wang, "Hierarchical recurrent neural network for skeleton based action recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 1110–1118, 2015. [Cité en page 31]

[65] X. Chen, H. Guo, G. Wang, and L. Zhang, "Motion feature augmented recurrent neural network for skeleton-based dynamic hand gesture recognition," in

2017 IEEE International Conference on Image Processing (ICIP), pp. 2881–2885, IEEE, 2017. [Cité en page 32]

[66] G. Vaudaux-Ruth, A. Chan-Hon-Tong, and C. Achard, "Actionspotter : Deep reinforcement learning framework for temporal action spotting in videos," in *2020 25th International Conference on Pattern Recognition (ICPR)*, pp. 631–638, IEEE, 2021. [Cité en page 32]

[67] Z. Guo, L. Gao, J. Song, X. Xu, J. Shao, and H. T. Shen, "Attention-based lstm with semantic consistency for videos captioning," in *Proceedings of the 24th ACM international conference on Multimedia*, pp. 357–361, 2016. [Cité en page 32]

[68] B. van Amsterdam, M. J. Clarkson, and D. Stoyanov, "Multi-task recurrent neural network for surgical gesture recognition and progress prediction," in *2020 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 1380–1386, IEEE, 2020. [Cité en page 33]

[69] S. Ji, W. Xu, M. Yang, and K. Yu, "3d convolutional neural networks for human action recognition," *IEEE transactions on pattern analysis and machine intelligence*, vol. 35, no. 1, pp. 221–231, 2012. [Cité en pages 34 et 63]

[70] A. Karpathy, G. Toderici, S. Shetty, T. Leung, R. Sukthankar, and L. Fei-Fei, "Large-scale video classification with convolutional neural networks," in *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, pp. 1725–1732, 2014. [Cité en page 34]

[71] P. Molchanov, S. Gupta, K. Kim, and J. Kautz, "Hand gesture recognition with 3d convolutional neural networks," in *Proceedings of the IEEE conference on computer vision and pattern recognition workshops*, pp. 1–7, 2015. [Cité en page 34]

[72] J. Carreira and A. Zisserman, "Quo vadis, action recognition ? a new model and the kinetics dataset," in *proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 6299–6308, 2017. [Cité en page 34]

[73] J. Zang, L. Wang, Z. Liu, Q. Zhang, G. Hua, and N. Zheng, "Attention-based temporal weighted convolutional neural network for action recognition," in *IFIP International Conference on Artificial Intelligence Applications and Innovations*, pp. 97–108, Springer, 2018. [Cité en pages 35 et 36]

[74] Z. Hu, Y. Hu, J. Liu, B. Wu, D. Han, and T. Kurfess, "3d separable convolutional neural network for dynamic hand gesture recognition," *Neurocomputing*, vol. 318, pp. 151–161, 2018. [Cité en page 35]

[75] Z. Yu, B. Zhou, J. Wan, P. Wang, H. Chen, X. Liu, S. Z. Li, and G. Zhao, "Searching multi-rate and multi-modal temporal enhanced networks for gesture recognition," *arXiv preprint arXiv :2008.09412*, 2020. [Cité en page 36]

[76] W. Zhang, Z. Lin, J. Cheng, C. Ma, X. Deng, and H. Wang, "Sta-gcn : two-stream graph convolutional network with spatial–temporal attention for hand gesture recognition," *The Visual Computer*, vol. 36, no. 10, pp. 2433–2444, 2020. [Cité en page 37]

[77] C. Li, S. Li, Y. Gao, X. Zhang, and W. Li, "A two-stream neural network for pose-based hand gesture recognition," *arXiv preprint arXiv :2101.08926*, 2021. [Cité en page 37]

[78] S. Yan, Y. Xiong, and D. Lin, "Spatial temporal graph convolutional networks for skeleton-based action recognition," in *Thirty-second AAAI conference on artificial intelligence*, 2018. [Cité en page 38]

[79] C. Si, Y. Jing, W. Wang, L. Wang, and T. Tan, "Skeleton-based action recognition with spatial reasoning and temporal stack learning," in *Proceedings of the European Conference on Computer Vision (ECCV)*, pp. 103–118, 2018. [Cité en page 38]

[80] M. Li, S. Chen, X. Chen, Y. Zhang, Y. Wang, and Q. Tian, "Actional-structural graph convolutional networks for skeleton-based action recognition," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 3595–3603, 2019. [Cité en page 38]

[81] S. Sabour, N. Frosst, and G. E. Hinton, "Dynamic routing between capsules," in *Advances in neural information processing systems*, pp. 3856–3866, 2017. [Cité en pages 39, 64, 66, 67, 73, 75, 84, 118 et 120]

[82] S. Srivastava, P. Khurana, and V. Tewari, "Identifying aggression and toxicity in comments using capsule network," in *Proceedings of the First Workshop on Trolling, Aggression and Cyberbullying (TRAC-2018)*, pp. 98–105, 2018. [Cité en page 39]

[83] X. Ma, Y. Li, Z. Cui, and Y. Wang, "Forecasting transportation network speed using deep capsule networks with nested lstm models," *arXiv preprint arXiv :1811.04745*, 2018. [Cité en page 39]

[84] K. Duarte, Y. Rawat, and M. Shah, "Videocapsulenet : A simplified network for action detection," in *Advances in Neural Information Processing Systems*, pp. 7610–7619, 2018. [Cité en pages 39 et 40]

[85] B. McIntosh, K. Duarte, Y. S. Rawat, and M. Shah, "Visual-textual capsule routing for text-based video segmentation," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 9942–9951, 2020. [Cité en page 39]

[86] W. Zhu, C. B. Owen, H. Li, and J.-H. Lee, "Personalized in-store e-commerce with the promopad : an augmented reality shopping assistant," *Electronic Journal for E-commerce Tools and Applications*, vol. 1, no. 3, pp. 1–19, 2004. [Cité en page 40]

[87] N. V. Vinh and H.-S. Jun, "Ar-based message annotation system for personalized assistance," *The KIPS Transactions : PartB*, vol. 16, no. 6, pp. 435–442, 2009. [Cité en page 41]

[88] K. Essig, B. Strenge, and T. Schack, "Adamaas : towards smart glasses for mobile and personalized action assistance," in *Proceedings of the 9th ACM International Conference on PErvasive Technologies Related to Assistive Environments*, pp. 1–4, 2016. [Cité en pages 41 et 42]

[89] S. K. Kane, B. Frey, and J. O. Wobbrock, "Access lens : a gesture-based screen reader for real-world documents," in *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pp. 347–350, 2013. [Cité en page 41]

[90] M. Schröder and H. Ritter, "Deep learning for action recognition in augmented reality assistance systems," in *ACM SIGGRAPH 2017 Posters*, pp. 1–2, 2017. [Cité en pages 42 et 43]

[91] C. Galea and C. Porter, "Accessible choral ensembles for visually impaired singers," in *Proceedings of the 32nd International BCS Human Computer Interaction Conference 32*, pp. 1–10, 2018. [Cité en page 42]

[92] A. P. Bradley, "The use of the area under the roc curve in the evaluation of machine learning algorithms," *Pattern recognition*, vol. 30, no. 7, pp. 1145–1159, 1997. [Cité en page 44]

[93] M. Hoai and F. De la Torre, "Max-margin early event detectors," *International Journal of Computer Vision*, vol. 107, no. 2, pp. 191–202, 2014. [Cité en page 44]

[94] O. Köpüklü, A. Gunduz, N. Kose, and G. Rigoll, "Real-time hand gesture detection and classification using convolutional neural networks," in *2019 14th IEEE International Conference on Automatic Face & Gesture Recognition (FG 2019)*, pp. 1–8, IEEE, 2019. [Cité en page 45]

[95] W. S. McCulloch and W. Pitts, "A logical calculus of the ideas immanent in nervous activity," *The bulletin of mathematical biophysics*, vol. 5, no. 4, pp. 115–133, 1943. [Cité en pages 49 et 50]

[96] M. Leshno, V. Y. Lin, A. Pinkus, and S. Schocken, "Multilayer feedforward networks with a nonpolynomial activation function can approximate any function," *Neural networks*, vol. 6, no. 6, pp. 861–867, 1993. [Cité en page 51]

[97] F. Rosenblatt, "The perceptron : a probabilistic model for information storage and organization in the brain.," *Psychological review*, vol. 65, no. 6, p. 386, 1958. [Cité en page 51]

[98] J. J. Hopfield, "Neural networks and physical systems with emergent collective computational abilities," *Proceedings of the national academy of sciences*, vol. 79, no. 8, pp. 2554–2558, 1982. [Cité en page 52]

[99] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, "Learning internal representations by error propagation," tech. rep., California Univ San Diego La Jolla Inst for Cognitive Science, 1985. [Cité en page 52]

[100] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998. [Cité en pages 56 et 71]

[101] D. C. Ciresan, U. Meier, L. M. Gambardella, and J. Schmidhuber, "Deep big simple neural nets excel on handwritten digit recognition," *CoRR*, vol. abs/1003.0358, 2010. [Cité en page 58]

[102] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *arXiv preprint arXiv :1409.1556*, 2014. [Cité en page 58]

[103] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778, 2016. [Cité en page 58]

[104] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, "Learning representations by back-propagating errors," *nature*, vol. 323, no. 6088, pp. 533–536, 1986. [Cité en page 59]

[105] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997. [Cité en page 62]

[106] K. Cho, B. Van Merriënboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio, "Learning phrase representations using rnn encoder-decoder for statistical machine translation," *arXiv preprint arXiv :1406.1078*, 2014. [Cité en page 62]

[107] M. Schuster and K. K. Paliwal, "Bidirectional recurrent neural networks," *IEEE transactions on Signal Processing*, vol. 45, no. 11, pp. 2673–2681, 1997. [Cité en page 62]

[108] J. Rajasegaran, V. Jayasundara, S. Jayasekara, H. Jayasekara, S. Seneviratne, and R. Rodrigo, "Deepcaps : Going deeper with capsule networks," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 10725–10733, 2019. [Cité en pages 73, 74, 75, 81 et 82]

[109] X. Ma, H. Zhong, Y. Li, J. Ma, Z. Cui, and Y. Wang, "Forecasting transportation network speed using deep capsule networks with nested lstm models," *IEEE Transactions on Intelligent Transportation Systems*, 2020. [Cité en page 75]

[110] Y. LeCun, "The mnist database of handwritten digits. nec research institute," 1998. [Cité en page 83]

[111] C. François *et al.*, "Keras : The python deep learning library," *keras. io*, 2015. [Cité en page 86]

[112] M. Abadi, P. Barham, J. Chen, Z. Chen, A. Davis, J. Dean, M. Devin, S. Ghemawat, G. Irving, M. Isard, *et al.*, "Tensorflow : A system for large-scale machine learning," in *12th {USENIX} Symposium on Operating Systems Design and Implementation ({OSDI} 16)*, pp. 265–283, 2016. [Cité en pages 86 et 92]

[113] D. P. Kingma and J. Ba, "Adam : A method for stochastic optimization," *arXiv preprint arXiv :1412.6980*, 2014. [Cité en page 86]

[114] C. Feichtenhofer, A. Pinz, and A. Zisserman, "Convolutional two-stream network fusion for video action recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 1933–1941, 2016. [Cité en page 91]

[115] E. Zhang, B. Xue, F. Cao, J. Duan, G. Lin, and Y. Lei, "Fusion of 2d cnn and 3d densenet for dynamic gesture recognition," *Electronics*, vol. 8, no. 12, p. 1511, 2019. [Cité en page 93]

[116] A. Swaminathan, "Perception at magic leap," 2019. [Cité en page 96]