Réseaux de Neurones à Convolution Spatio-Temporelle pour l'analyse et la reconnaissance précoce d'actions et de gestes

William Mocaër

COLLEGE MATHS, TELECOMS
DOCTORAL INFORMATIQUE, SIGNAL
BRETAGNE SYSTEMES, ELECTRONIQUE

INSA INSTITUT NATIONAL DES SCIENCES APPLIQUÉES RENNES

THÈSE DE DOCTORAT DE

L'INSTITUT NATIONAL DES SCIENCES APPLIQUÉES DE RENNES

ÉCOLE DOCTORALE N° 601
Mathématiques, Télécommunications, Informatique, Signal, Systèmes, Électronique
Spécialité : « *Informatique* »

Par

« William MOCAËR »

« Réseaux de Neurones à Convolution Spatio-Temporelle pour l'analyse et la reconnaissance précoce d'actions et de gestes »

Thèse présentée et soutenue à Rennes, le 15 décembre 2023
Unité de recherche : « IRISA »
Thèse N° : 23ISAR 41 / D23 - 41

Rapporteurs avant soutenance :

Harold MOUCHÈRE — Professeur des Universités, Nantes Université
Jenny BENOIS-PINEAU — Professeure des Universités, Université de Bordeaux

Composition du Jury :

Présidente :	Catherine ACHARD	Professeure des Universités, Sorbonne Université
Examinateurs :	Harold MOUCHÈRE	Professeur des Universités, Nantes Université
	Jenny BENOIS-PINEAU	Professeure des Universités, Université de Bordeaux
Dir. de thèse :	Eric ANQUETIL	Professeur des Universités, INSA Rennes
Co-dir. de thèse :	Richard KULPA	Professeur des Universités, Université de Rennes 2

Remerciements

Je tiens à exprimer ma sincère gratitude à toutes les personnes qui ont contribué à la réalisation de cette thèse.

Tout d'abord, je tiens à remercier mes directeurs de thèse. Eric, dans un premier temps pour m'avoir encouragé à faire de la recherche lors de discussions, notamment à l'occasion du projet de 4e année de l'INSA. Dans un second temps pour son investissement sincère et sans faille tout au long de ma thèse ainsi que pour la qualité des discussions scientifiques que nous avons eues. Je remercie Richard de m'avoir permis de réaliser cette thèse via son intégration précoce dans le projet DIGISPORT. Je le remercie en particulier pour ses encouragements et pour sa reconnaissance du travail accompli et des progrès réalisés.

Je souhaite également exprimer ma profonde reconnaissance envers les rapporteurs Harold Mouchère et Jenny Benois-Pineau pour avoir accepté de lire et d'évaluer ce travail. Je remercie également Catherine Achard et Pierre-François Marteau, qui ont accepté de faire partie du jury de thèse ainsi que pour les discussions lors des réunions du CSI.

J'ai également une pensée pour Dalila Tamzalit et Joost Noppen qui m'ont permis de découvrir la recherche à l'occasion de mon stage à l'Université d'East Anglia, et qui ont été les premiers à me parler de doctorat.

Un grand merci à toute l'équipe de recherche Intuidoc/Shadoc pour leur collaboration, les discussions scientifiques, et l'ambiance de travail conviviale. Je souhaite remercier particulièrement mon collègue doctorant Killian, présent dès mes premiers pas dans l'équipe lors du stage de master.

Je remercie Pauline Chabaud, mon amie et ancienne collègue de DUT, pour ses nombreuses relectures et corrections d'articles et de documents importants, sans lesquelles la qualité de mes écrits n'aurait pas été la même.

Je remercie mes amis du master et de l'INSA, Gaël, Julien, Antoine, Taha, Olivier et Enzo pour les discussions et le partage de nos expériences de doctorat. Je remercie également Tsiry et Romain pour les discussions sans filtres que nous avons autour de la recherche.

Enfin, je tiens à exprimer ma profonde gratitude envers Camille, pour son soutien infaillible et pour son intérêt sincère pour mes travaux.

Table des Matières

Chapitre 1

Introduction

L'**interaction humain-machine** basée sur les gestes constitue une approche intuitive et naturelle pour communiquer avec les ordinateurs. Cette forme d'interaction tire parti des mouvements corporels et gestuels pour permettre aux utilisateurs de contrôler, de communiquer et de collaborer avec des applications de manière plus fluide et immersive. Contrairement aux interfaces traditionnelles, telles que les claviers et les souris qui peuvent être contraignantes et nécessiter un apprentissage, les gestes humains sont une forme d'expression universelle qui ne nécessite pas toujours d'apprentissage préalable.

Un geste peut être effectué par les doigts ou un crayon sur un dispositif tactile, ou bien effectué par un corps complet. De nombreuses applications exploitent déjà l'interaction humain-machine basée sur les gestes. Du côté de l'interaction sur des écrans tactiles, les gestes de navigations sont très répandus, à l'image du geste de pincement pour appliquer un zoom. Dans le domaine du sport, certaines applications utilisent des systèmes de capture de mouvement et permettent aux joueurs de contrôler les actions de leurs avatars en effectuant des gestes spécifiques.

L'interaction humain-machine basée sur les gestes offre un potentiel considérable pour repousser les limites de la communication entre l'humain et la machine. Cependant, malgré les avantages indéniables, il reste des défis à relever pour garantir une reconnaissance précise, robuste et rapide des gestes afin de permettre une interaction fluide.

En particulier, la **reconnaissance précoce du geste**, en plus de s'assurer de la bonne réactivité du système, est parfois nécessaire afin de garantir l'utilisation d'une application de façon naturelle. Par « reconnaissance précoce de geste », nous entendons le fait de reconnaître un geste **le plus tôt possible** durant son déroulement. Dans le cadre de cette thèse, nous visons principalement les **systèmes interactifs**. Ces systèmes ont donc besoin de réagir rapidement à l'action de l'utilisateur afin de permettre une interaction fluide. Par exemple, dans le cadre d'une application de boxe qui met à disposition un adversaire virtuel, la reconnaissance du geste de l'utilisateur doit être effectuée le plus

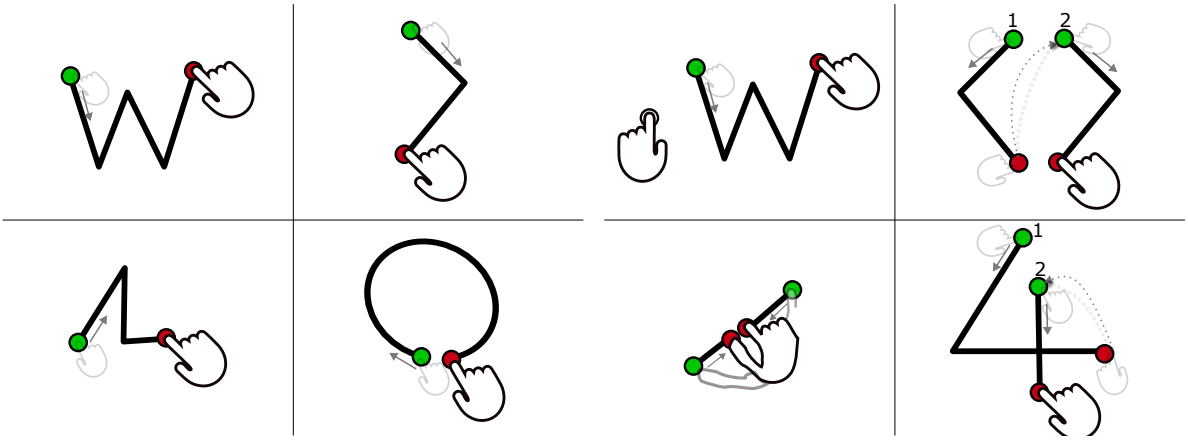
rapidement possible afin de permettre une réaction adaptée de l'adversaire virtuel. La reconnaissance précoce permet de répondre à ce **besoin de réactivité**.

Il y a également un deuxième problème théorique auquel la reconnaissance précoce vient répondre, il s'agit de celui de la coexistence entre deux types de gestes, les gestes que nous appellerons **« symboliques » ou « abstraits »** et les gestes de **manipulations directes**. Ce problème théorique a été discuté par Kurtenbach et Buxton [KB91] ainsi que par Petit et Maldivi [PM13]. L'exemple classique d'un geste de manipulation directe est le geste que nous utilisons traditionnellement pour « zoomer », c'est-à-dire un geste écartant deux doigts initialement proches sur la surface, un autre exemple est le geste de rotation. De manière générale, les gestes de manipulation directe permettent de manipuler l'interface via l'amplitude du geste effectué. Pour le geste du zoom, plus les doigts s'écartent, plus le zoom sera intense. Les gestes dits « symboliques » sont l'ensemble des autres gestes qui produisent un *feedback* généralement à la fin de la réalisation. La coexistence de ces deux types de gestes dans un contexte interactif n'est possible que si nous sommes capables de prédire très tôt l'intention de l'utilisateur, avant que le geste ne soit terminé. Plus précisément, s'il s'agit d'un geste en prise directe, le système doit obligatoirement le reconnaître dès son initiation afin de permettre la manipulation en direct.

La thèse va donc s'intéresser à cette problématique de la reconnaissance précoce de gestes. Cette problématique se pose pour des dispositifs comme les tablettes et les gestes 2D associés ou encore des systèmes immersifs tels que la réalité virtuelle en interaction gestuelle complète avec la capture du mouvement en 3D. Il est donc nécessaire de concevoir **une approche générique qui répond aux exigences des gestes 2D et 3D**.

Reconnaissance de gestes 2D et 3D

Le **geste en deux dimensions (2D)** se caractérise par le mouvement d'un point ou d'un ensemble de points dans un plan. Il existe différentes natures de gestes 2D. Il est généralement décrit et illustré par la trajectoire qu'il produit par son déroulement. Concernant la nature du tracé, les gestes peuvent être *mono-stroke* (figure 1.1a) : un seul tracé continu représente le geste ou bien *multi-stroke* (figure 1.1b) : plusieurs tracés sont utilisés pour former un geste complet. Regardant le nombre de contacts simultanés avec l'interface, les gestes multi-strokes peuvent être *mono-touch* ou *multi-touch*.



(a) Mono-stroke (b) Multi-stroke multi et mono-touch

FIGURE 1.1 – Exemples de gestes a) mono-stroke b) multi-stroke multi-touch (colonne de gauche) et mono-touch (colonne de droite).

À l'inverse, le **geste 3D** caractérise un mouvement effectué dans un environnent en 3D. Le plus souvent le corps complet est considéré, mais il est possible de ne considérer qu'une partie du corps, comme les mains.

L'étude pionnière de Johansson [Joh73] a démontré la capacité humaine à reconnaître les mouvements à partir de signaux simples tels que des points lumineux positionnés sur les articulations d'un acteur filmé en noir et blanc. Cette expérience visuelle a montré que les humains sont capables de reconnaître des actions en se basant uniquement sur les informations de mouvement fournies par les articulations, indépendamment des détails comme la couleur et l'arrière-plan. Cette représentation simple des mouvements articulaires présente des avantages en éliminant les biais liés aux couleurs et aux éléments non pertinents.

Ainsi, nous estimons que connaître les coordonnées des articulations dans l'espace est suffisant afin de permettre de reconnaître les gestes. C'est le choix qu'a fait une grande partie de la communauté de la reconnaissance de gestes 3D. Nous observons du côté des approches qui se basent sur les vidéos RGB (images en couleurs filmées avec une caméra standard) que trouver le squelette de l'utilisateur afin de l'exploiter dans leurs systèmes est de plus en plus fréquent.

Dans cette thèse, nous avons fait le choix d'exploiter uniquement la modalité « squelette », c'est-à-dire lorsque de nous disposons des coordonnées 3D des articulations de l'utilisateur. La manière d'obtenir des coordonnées importe peu, mais nous avons particu-

lièrement privilégié les systèmes de type *Kinect* qui mettent à disposition un algorithme de restitution du squelette [Sho+11] à partir de la vidéo RGB+D (couleurs et profondeur) qu'ils obtiennent.

Cette notion de gestes 3D à partir d'articulations présente des similitudes avec la reconnaissance de gestes 2D sur tablette, où les mouvements des doigts sont capturés en fonction des coordonnées spatiales. Comme pour le geste 2D, nous pouvons regarder les trajectoires générées par les différentes articulations du corps humain, un exemple est donné en figure 1.2.

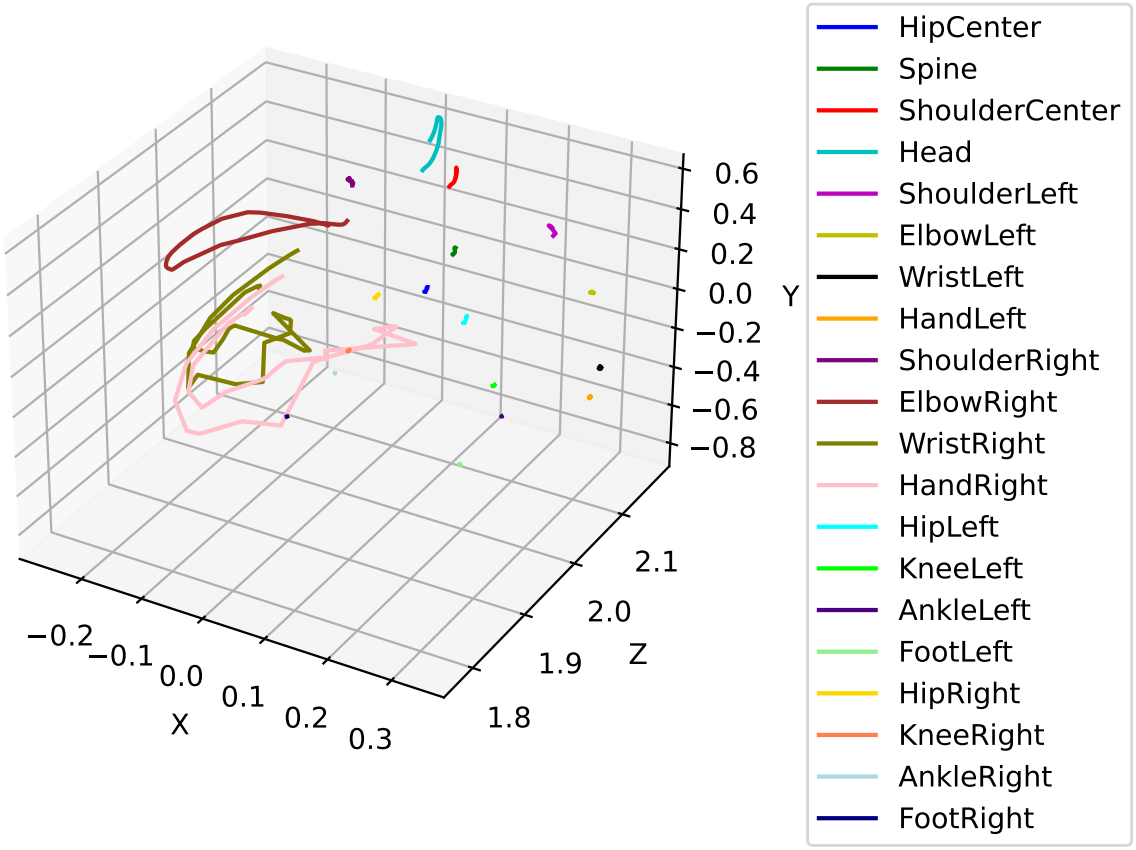


FIGURE 1.2 – Illustration de trajectoires 3D issues d'un geste effectué par un corps complet. Exemple issu de la base *Chalearn* [Esc+13] geste « sonostufo ».

Le geste 3D peut parfois plus facilement s'illustrer avec une séquence de postures, comme visible en figure 1.3.

Cependant, il existe un lien entre les gestes en 2D et 3D. Il est possible de considérer un geste 3D comme une extension d'un geste 2D. Le geste 3D est un cas particulier des gestes

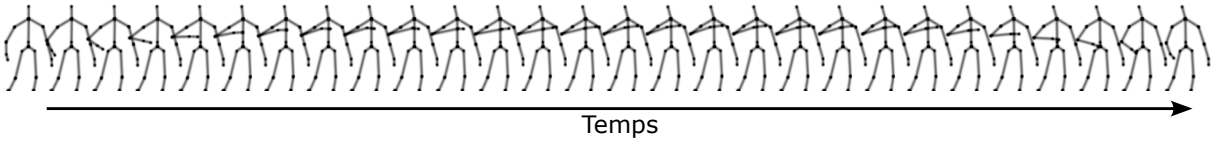


FIGURE 1.3 – Une séquence correspondant au geste « sonostufo » de la base *Chalearn* [Esc+13]

multi-stroke multi-touch, où chaque articulation à son propre tracé. La figure 1.4 illustre ce lien en montrant un tracé en fonction du temps dans un espace 2D. Un tracé peut être associé à la fois à un geste 2D, ou à un tracé d'une articulation constituant un geste 3D. Le geste 3D n'a pas d'espace de « contact », les articulations sont toujours captées. Déterminer celles qui expriment réellement le geste n'est pas systématiquement trivial. Cette continuité dans la capture des articulations en 3D présente des défis supplémentaires pour la reconnaissance des gestes. Une différence majeure est que les articulations sont identifiables (main, tête, pied...) contrairement aux contacts captés sur un appareil tactile. De manière générale, le geste 3D sera beaucoup plus bruité que le geste 2D, d'une part du fait du nombre d'articulations captées, d'autre part par les limites en termes de précision du système de capture.

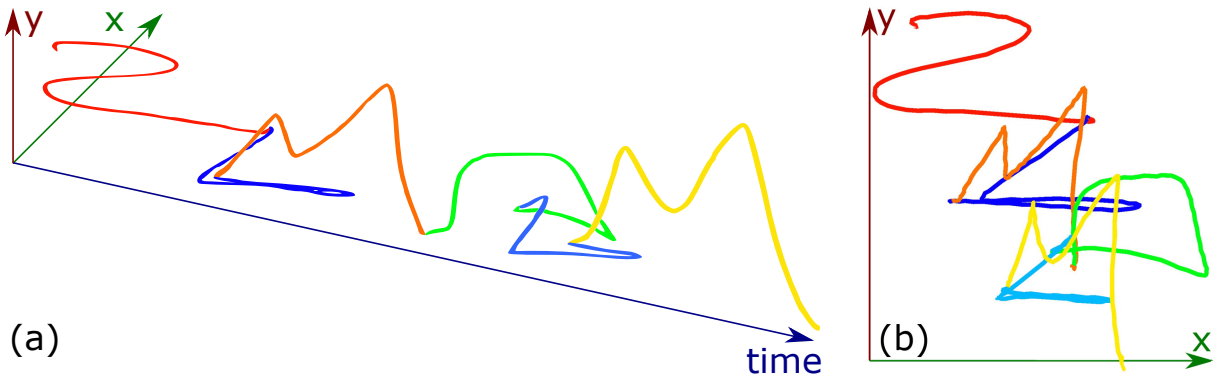


FIGURE 1.4 – Illustration du contexte non segmenté pour des gestes 2D mono-stroke sans levers. a) Vue perspective « spatio-temporelle ». b) Vue « spatiale » uniquement.

En nous appuyant sur les similitudes entre les gestes 2D et 3D, mais en prenant aussi en compte leurs différences, nous allons travailler sur une approche **générique** qui inclut la prise en compte de ces deux types de gestes. À cette fin, cette thèse est née de la collaboration entre deux équipes de recherche, **Shadoc** (anciennement Intuidoc) et **MimeTIC/M2S**. Shadoc est spécialiste de l'apprentissage machine appliqué aux documents dont les gestes « 2D » effectués sur dispositifs interactifs tactiles. MimeTIC est

spécialiste du mouvement humain, en particulier appliqué au sport. De plus, cette thèse est partiellement financée par l'EUR **DIGISPORT**, une école universitaire de recherche dédiée à l'utilisation du numérique dans le domaine du sport.

Segmentation des gestes

Enfin, que les gestes soient en 2D ou 3D, ils peuvent être **pré-segmentés** ou **non segmentés**. Dans un cadre pré-segmenté (figure 1.5) un seul geste est considéré à la fois, alors qu'en non segmenté (figure 1.6) les gestes s'enchaînent dans un flux continu de données sans aucune information a priori sur leurs localisations temporelles. D'un point de vue applicatif, l'usage de gestes segmentés se traduit par des situations où la segmentation des gestes est triviale. Par exemple, lorsque seuls des gestes mono-strokes sont utilisés, n'importe quel « lever » (action de lever le doigt ou le stylet de la surface tactile) permet de segmenter les gestes de manière très simple.

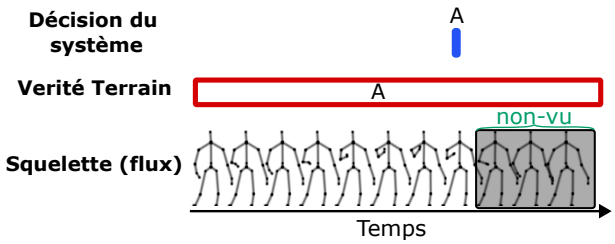


FIGURE 1.5 – Dans le contexte de la détection du geste segmenté, notre objectif est de détecter un geste unique (détection en bleu) le plus tôt possible, même si cette détection peut survenir à un stade avancé du geste en fonction des confusions possibles avec d'autres gestes.

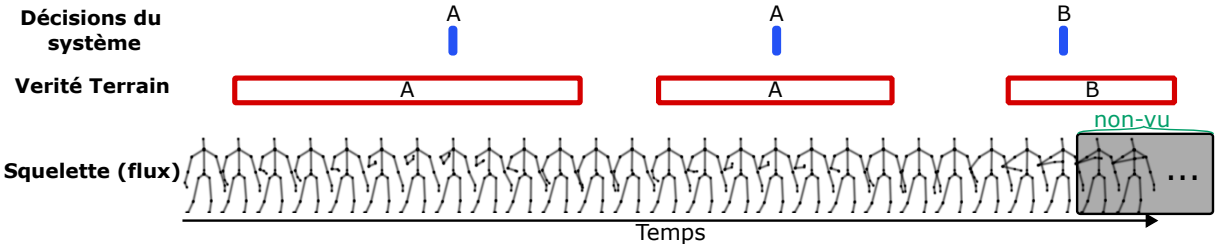


FIGURE 1.6 – Dans le contexte de la détection non segmentée, nous cherchons à détecter et identifier chaque geste individuellement dès que possible. Plusieurs gestes peuvent être effectués à la suite, avec des instants de pause ou non.

Bien que les gestes 2D sont plus souvent considérés dans un contexte segmenté, la segmentation devient un problème dès lors que l'on peut utiliser des gestes à la chaîne. D'un point de vue théorique, la segmentation des gestes 2D n'est plus triviale dans **deux situations**. **La première** est lorsque plusieurs tracés sont effectués séquentiellement et que le nombre de tracés varie entre les différents gestes. Par exemple, si 6 tracés séquentiels sont effectués, 3 gestes pourraient être effectués : le premier avec 1 tracé, le deuxième avec 3 tracés, et le dernier avec 2 tracés. Mais il pourrait également y avoir 6 gestes de 1 tracé chacun. Le système doit donc réussir à différencier les instances de gestes. De plus, les gestes peuvent être superposés les uns aux autres (la superposition peut être renforcée par l'usage de petits écrans tactiles comme des montres), donc les localisations spatiales ou temporelles de ces tracés peuvent ne pas aider à segmenter le geste dans un cas général. En revanche le système dispose d'une sur segmentation à chaque lever, ce qui est une aide considérable pour les systèmes. **La deuxième** situation est lorsqu'il s'agit de gestes mono-strokes qui s'enchaînent sans aucun lever entre les gestes. Cette situation ressemble à un contexte d'écriture manuscrite, sauf que les gestes peuvent être superposés les uns aux autres. La figure 1.4 illustre ce cas. Cette fois-ci, le système ne dispose d'aucune information sur segmentation pour l'aider à trouver les gestes. Dans ces deux situations, les gestes 2D sont donc **non segmentés**.

Concernant le geste 3D, la problématique de reconnaissance dans le contexte segmenté a été beaucoup explorée dans la littérature. En 2D, effectuer des levers sur la surface tactile permet de produire des segmentations triviales pour des gestes mono-strokes. En 3D, il est impossible d'effectuer un « lever », ce concept n'existe pas dans ce contexte. La segmentation de gestes 3D n'est donc généralement pas triviale, et la considérer comme telle en adressant uniquement des gestes segmentés facilite beaucoup la tâche. Le contexte non segmenté est donc plus réaliste pour le geste 3D, nous n'adresserons que ce contexte dans cette thèse.

Dans cette thèse, nous adressons donc la **reconnaissance précoce de geste** dans **contexte segmenté** pour le geste 2D, ainsi que dans le **contexte non segmenté** pour les gestes 2D et 3D.

Objectifs de la thèse

L'objectif général de la thèse est de produire un système capable de **reconnaître les gestes le plus tôt possible, qu'ils soient en 2D ou 3D**. Le système devra notamment

être capable de repousser la décision jusqu'au moment où les gestes ne peuvent plus être confondus. **Dans le contexte segmenté**, que nous adressons pour le geste 2D, il s'agira d'émettre une unique décision de reconnaissance et de faire en sorte que celle-ci arrive le plus tôt possible. Pour les gestes **non segmentés** (2D ou 3D), le système doit être capable d'émettre une décision pour chaque instance de geste, toujours en visant à ce que ces décisions arrivent le plus tôt possible.

Contributions de la thèse

Au cours de cette recherche, nous avons apporté des contributions sur trois axes clés. Tout d'abord, des stratégies novatrices ont été développées pour **représenter les gestes**, mettant l'accent sur son exploitation par un réseau de neurones convolutif (CNN) spatio-temporel afin d'exploiter pleinement les avantages des convolutions en 3D. Ces représentations projettent les trajectoires du geste dans un espace euclidien, et utilisent le déplacement effectif de cette trajectoire afin d'y construire une représentation indépendante de la vitesse d'exécution. Ensuite, une **architecture CNN** a été conçue en adéquation avec la représentation. Sa conception efficiente inspirée de Wavenet [Oor+16] permet son inférence en temps réel et son utilisation en ligne, ainsi que l'intégration d'un contexte temporel suffisamment large pour considérer l'ensemble du début de geste réalisé. Afin de permettre une **prise de décision** robuste, des mécanismes sont intégrés. Dans le contexte segmenté, nous avons adapté celui de SelectiveNet [GE19] dans notre contexte en ligne. Pour les gestes non segmentés nous avons développé une méthode d'apprentissage basée sur la fonction CTC (Connectionist Temporal Classification) [Gra+06] afin d'obtenir une sortie à un niveau instance, produisant une sortie stable et fiable au cours des différents instants de la séquence. En guidant l'apprentissage avec un CTC guidé par la segmentation, et en permettant de contrôler l'équilibre entre précocité et précision, nous obtenons un système disposant de bonnes propriétés pour effectuer la tâche de détection précoce pour le contexte de l'interaction humain-machine.

Plan du manuscrit

Ce manuscrit de thèse se compose de quatre chapitres principaux. Dans le **chapitre 2**, nous effectuons un examen approfondi de l'état de l'art de la reconnaissance de gestes en ligne. Nous y étudierons les différentes tâches relatives à notre problématique, les mé-

triques associées, les différents mécanismes de prises de décisions exploitées, ainsi que différentes approches à base de réseaux récurrents, transformers, et de réseaux à convolutions. Le **chapitre 3** présente sur notre approche pour la reconnaissance précoce des gestes segmentés, détaillant nos contributions et les expérimentations menées pour évaluer cette tâche. Le **chapitre 4** détaille notre méthode dédiée à la détection des gestes non segmentés. Nous évaluerons cette tâche sur un total de huit bases de données (6 pour le geste 3D, 2 pour le geste 2D) avec des métriques pertinentes pour notre contexte applicatif. Cette évaluation en profondeur permettra de voir les impacts des différents modules, et de nous comparer avec les méthodes de l'état de l'art. Enfin, le **chapitre 5** présente nos conclusions et ouvre des perspectives pour les recherches futures.

Chapitre 2

État de l'art

2.1 Classification des tâches relatives à la reconnaissance de gestes en ligne

La classification des tâches en ligne dans la reconnaissance de gestes et la détection d'actions se caractérise par une diversité d'objectifs et de méthodes d'évaluation. La définition et le nommage de ces tâches n'étant pas uniforme dans tous les travaux, nous proposons ici une **classification en fonction de deux axes**, le **niveau de segmentation** (segmenté ou non segmenté) et le **niveau de compréhension** du geste (niveau frame ou niveau instance).

Plus précisément, les gestes peuvent être **segmentés**, c'est-à-dire que chaque séquence ne contient qu'un seul geste, il commence et se termine généralement au début et à la fin de cette séquence. Au contraire, les gestes peuvent être **non segmentés**, sans aucune information préalable sur leur localisation temporelle, dans une séquence où plusieurs actions s'enchaînent.

Le **niveau frame** s'oppose au **niveau instance**, ils caractérisent un niveau de compréhension de la séquence à la sortie du système, et une philosophie d'évaluation. Le **niveau frame** fait référence à une prédiction d'un geste à chaque frame, sans philosophie de regroupement de frames. La métrique d'évaluation compare les prédictions frame par frame, sans regarder la cohérence temporelle. À ce niveau, il est par exemple impossible de compter le nombre de gestes effectués dans une séquence. Le **niveau instance** désigne un plus haut niveau de compréhension, la sortie peut être des bornes de détection (début/fin), ou un instant de détection ponctuel. Les méthodes d'évaluation regardent les instances de gestes, elles peuvent tester la qualité de la segmentation en regardant par exemple le taux de chevauchement entre les bornes prédites et réelles, la distance entre l'instant de prédiction et un début de geste (précocité) ou d'autres métriques considérant un geste comme une unité. Ces deux niveaux de sorties sont illustrés en figure 2.1. Il

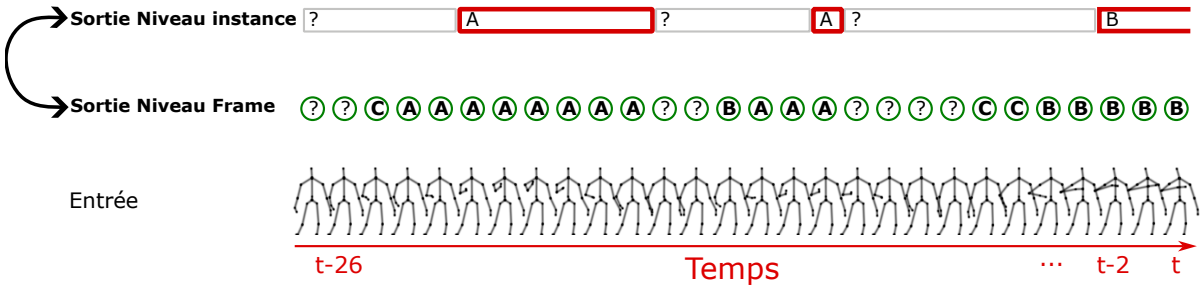


FIGURE 2.1 – Deux niveaux de sorties sont possibles, le niveau frame qui propose une classe par frame, ou le niveau instance qui regroupe une ou plusieurs frames afin de constituer une instance de geste.

est possible de passer d'une prédiction niveau frame à un niveau instance, par exemple en regroupant toutes les frames consécutives dont la classe prédite est la même, tout en conservant la contrainte en ligne (pas de requalification de frames passées). La pertinence de ce regroupement dépendra alors de l'approche.

Cette distinction de niveau a été explicitée dans certains travaux récents [Sho+18 ; KNK22] qui traitent de tâches annexes telles que l'*Online Temporal Action Localisation* (OnTal) ou l'*Online Detection of Action Start* (ODAS), où le niveau instance est requis pour effectuer la tâche. Nous détaillerons ces deux tâches dans les prochaines sections (sections 2.1.2.4 et 2.1.2.3).

Dans le domaine du geste 2D, la distinction de niveau n'est généralement jamais faite, car depuis le début les gestes 2D ont toujours été considérés au niveau instance. Ainsi, l'explicitation de ces différents niveaux dans cette communauté n'a jamais été nécessaire.

Comme cette thèse se situe à la frontières des communautés du geste 2D et 3D, nous avons besoin de faire des liens entre les tâches existantes dans ces deux domaines. Certains termes diffèrent entre ces communautés. Les principales différences sont visibles dans le tableau 2.1. On peut observer que certains termes en 3D n'ont pas d'équivalent en 2D. **En 2D**, le terme **hors ligne (*offline*)** qualifie le **signal capturé**, et est souvent représenté par une trace dont l'ordre de tracé n'est pas connu (comme une écriture sur papier). **En 3D** il n'y a pas d'équivalent pour ce terme, le signal est nécessairement séquencé. « Hors ligne » désigne autre chose dans la communauté 3D, cela qualifie un **traitement de la donnée** qui a lieu a posteriori (on parle de détection d'action *hors ligne*), où l'entièreté de la séquence est disponible, qui s'oppose avec le terme **en ligne** (à la volée, en temps réel).

TABLE 2.1 – Équivalence de vocabulaire entre les communautés 2D et 3D

Sujet	**2D**	**3D**
Signal	En ligne / Online	✗
Signal	Hors ligne / Offline	✗
Traitement	Temps réel, à la volée / Real-time, on-the-fly	En ligne / Online
Traitement	A posteriori	Hors ligne / Offline

Dans cette thèse, nous utiliserons principalement la terminologie 3D, sauf mention explicite. Ainsi, quand nous parlerons de méthode en ligne, nous désignerons la méthode de traitement (traitement en temps réel, à la volée). Pour le geste 3D, le mot « action » est plus souvent utilisé dans la littérature, dans ce manuscrit nous utiliserons les termes « geste » et « action » sans distinction.

Le tableau de classification des tâches (tableau 2.2) permet de clarifier les différentes tâches en fonction du niveau de traitement des gestes, et de la segmentation dans les approches **en ligne**. On remarque que certaines tâches ne sont pas adressées en 2D. Par exemple, aucune approche 2D ne semble s'intéresser au contexte non segmenté en ligne (à la volée). Pourtant, nous avons vu en introduction que le problème se posait dans deux situations distinctes. La première est le cas d'enchaînements de gestes multi-strokes au nombre variable de tracés, la deuxième est lorsque des gestes mono-strokes s'enchaînent sans lever de stylet.

Dans les sections suivantes, nous allons détailler les objectifs de ces différentes tâches. Dans une optique de clarté vis-à-vis de la littérature, nous conserverons les noms de tâches en anglais.

2.1.1 Tâches niveau frame

2.1.1.1 Action Prediction

La tâche de prédiction d'action ("action prediction") se déroule dans un contexte **segmenté**, où l'objectif est de pouvoir reconnaître un geste avant sa fin, avec une observation partielle du geste en cours. Pour évaluer cette tâche, une métrique courante consiste à mesurer le score de reconnaissance à différents pourcentages d'observation partielle. Par exemple, on peut évaluer la reconnaissance à 10%, 20%, 30%, etc. d'observation du geste complet.

TABLE 2.2 – Tableau récapitulatif des différentes tâches en ligne en lien avec notre tâche. * indique nos articles.

		Tâche	**Refs.**
Niveau Frame	Segmenté	Action Prediction Early Recognition	2D : [UA08 ; Car+14] 3D : [Ryo11 ; ZLS13 ; KKF14 ; LCS14 ; EMS16 ; Hu+16 ; Ali+17 ; SFH18 ; Hu+19 ; LL19 ; Wan+19 ; Sun+19 ; Cai+19 ; Ke+20 ; DWV19 ; Li+20 ; SY22 ; Foo+22 ; Wan+23a ; Wan+23b]
	Non Segmenté	Online Action Detection (OAD) Online Action Prediction	3D : [De +16 ; MSS16 ; JN17 ; Mle+19 ; Xu+19a ; Hou+20 ; Eun+20 ; Liu+20a ; YCL20 ; Eun+21 ; KNK21 ; Gao+21 ; Yan+21 ; VKM21 ; Wan+21 ; Xu+21 ; Zha+22 ; MM22 ; Che+22 ; Guo+22 ; ZK22 ; Cao+22 ; Cao+23 ; Gue+23]
		Action anticipation	3D :[JN17 ; KFS19 ; Sun+19 ; VKM21 ; Wan+21 ; Xu+21 ; MM22 ; Gue+23]
Niveau Instance	Segmenté	Early Gesture Recognition Real-time gesture Recognition	2D : [Che+17], [**MAK21**]* 3D : [Kaw+11 ; Ko+13]
	Non Segmenté	Online Action Detection (OAD) Early Action Detection Online Gesture Recognition	2D : [**MAK22a ; MAK22b**]* 3D : [Fot+12 ; Zha+13 ; Zha+14 ; Web+14 ; BMA14 ; Sha+15 ; MHT16 ; Mol+16 ; Li+16 ; BKK17 ; BAM17 ; Bou+18b ; Car+19 ; Liu+19],[**MAK23**]*
		Online Detection of Action Start (ODAS)	3D :[Sho+18 ; Gao+19 ; Gao+21]
		Online Temporal Action Loc. (OnTal) Online Action Segmentation	3D :[Gon+12 ; ZLS13 ; Hua+14 ; WS14 ; EMS16 ; Li+16 ; MSS16 ; Dev+17 ; Sin+17 ; BKK17 ; GK17 ; Bou+18a ; Liu+18a ; DWV19 ; Liu+19 ; YCL20 ; KNK22]

Cette tâche est particulièrement utile dans des applications où une réponse est nécessaire à un instant non contrôlé. Dans ce contexte une prédiction doit être émise dans tous les cas, même si le geste n'est pas clairement reconnaissable.

2.1.1.2 Online Action Detection (frame)

La tâche d'Online Action Detection (niveau frame) se déroule dans un contexte **non segmenté**. Elle a commencé avec l'article de De Geest et al. [De +16] en 2016, qui traite des vidéos RGB. Il décrit la tâche comme suit :

> « L'objectif de la détection d'action en ligne est de détecter une action au fur et à mesure qu'elle se produit, et idéalement même avant que l'action ne soit entièrement terminée. [...] Les défis de la détection d'action en ligne dans le monde réel sont les suivants. Premièrement, les actions doivent être détectées le plus tôt possible, idéalement après avoir observé seulement une partie de l'action. Deuxièmement, il est nécessaire de détecter les actions parmi une grande variété de données négatives non pertinentes. Troisièmement, en partant de données vidéo longues et non segmentées, la segmentation des gestes ne doit pas être triviale. »

Il s'agit d'une description plutôt claire, cependant, un choix de métrique afin d'évaluer les approches a fait basculer cette tâche au niveau frame, plutôt qu'au niveau instance comme la description pourrait le laisser entendre. Dans leur article suivant [DT18], la description de la tâche est réduite :

> « Notre objectif est d'identifier les actions effectuées à chaque frame de la vidéo en utilisant uniquement les observations passées et actuelles. »

Cette nouvelle description ne laisse plus de doute sur le niveau choisi. Ainsi, la quasi-totalité des approches adressant la tâche d'OAD a suivi cette philosophie. Du moins, les approches qui comparent leurs performances à celles de De Geest et al. sur les mêmes bases, des bases d'actions qui dispose de la modalité RGB.

En revanche, les approches qui se comparent entre elles sur des bases avec la modalité « squelette » ont tout de suite choisi des métriques au niveau instance, nous en reparlerons dans la section 2.1.2.2.

2.1.1.3 Action Anticipation

L'objectif de l'anticipation d'action (*action anticipation*), qui se déroule dans un contexte **non segmenté**, est de prédire quelle action sera effectuée dans un futur proche.

La métrique utilisée est similaire à celle de la tâche OAD par frame, mais le score est calculé sur les frames futures. Cette tâche peut être intéressante afin de prédire l'action suivante (dans un contexte où l'ordre des actions n'est pas aléatoire), ou afin d'estimer la fin de l'action en cours.

2.1.2 Tâches niveau instance

2.1.2.1 Early Gesture Recognition

Dans le contexte de la reconnaissance précoce de gestes (tâche *Early Gesture Recognition*), les gestes sont **segmentés** et l'objectif est de reconnaître un geste le plus tôt possible pendant sa réalisation. La décision de reconnaissance est irrévocable, ce qui signifie qu'une fois qu'un geste est classifié, la classe ne peut pas être modifié, même si le système reconnaît un geste différent peu après. La métrique utilisée pour évaluer cette tâche est la précocité, qui mesure à quel point le geste est reconnu tôt dans sa réalisation, ainsi qu'un score évaluant la qualité de la reconnaissance. Concernant l'évaluation de la qualité, on considère les trois cas possibles par geste : correctement classifié, mal classifié, ou rejeté (absence totale de décision).

Cette tâche de reconnaissance de gestes précoce trouve des applications dans divers domaines, notamment les applications interactives où la coexistence des gestes en prise directe et indirecte est nécessaire. Cela permet également de fluidifier l'interaction en fournissant une réponse rapide à l'utilisateur. Techniquement, n'importe quel système donnant une réponse par frame doté d'un mécanisme de rejet temporel peut être utilisé pour cette tâche. Seule la première décision de reconnaissance est importante, car un seul geste est traité à la fois dans cette tâche.

2.1.2.2 Online Action Detection (instance)

Cette tâche se déroule dans un contexte **non segmenté**. En 2012, Fothergill et al. [Fot+12] font partie des pionniers qui ont abordé la tâche d'OAD (*Online Action Detection*) au niveau instance. Ils ont également introduit le jeu de données MSRC-12 avec la modalité du squelette, doté du premier protocole d'évaluation au niveau de l'instance. L'objectif initial était de détecter l'action « au bon moment ». Pour cela, la base MSRC-12 a été annotée avec des *points d'action*. Sans réellement formaliser la tâche, il parle de « reconnaissance de gestes en ligne » (*Online Gesture Recognition*). Terme ensuite repris dans les articles de Zhao et al. [Zha+13 ; Zha+14] qui adressent la même tâche.

Nowozin et Shotton [NS12] ont défini le concept de « point d'action » comme suit :

> « Un point d'action d'une action est un moment unique où la présence de l'action est évidente et qui peut être identifié de manière unique pour toutes les instances de l'action. »

Ici, Nowozin et Shotton définissent ce concept afin de permettre l'annotation de ce point dans les bases de données. Il est sous-entendu que ce point doit être placé le plus tôt possible (car sinon il pourrait être placé systématiquement à la fin du geste), mais il n'est pas explicitement mentionné. Dans cette thèse, nous ferons parfois mention du point d'action *théorique* qui symbolise le point d'action le plus précoce possible sans risque d'ambiguïté, il diffère du point d'action classique qui désigne plutôt l'annotation faite par un humain, et qui n'est pas nécessairement le plus précoce possible.

Ce concept important peut être pertinent pour évaluer la performance de précocité d'un système. Connaître la frame du point d'action pour chaque échantillon de test permet d'évaluer l'exactitude de la prédiction en mesurant à quel point elle est proche de cette frame. C'est l'idée de la métrique "Latency-Aware" définie dans le travail de Fothergill et al., que nous décrirons en section 2.2.3.8. Cependant, définir les points d'action peut être difficile dans le contexte des actions humaines, car certaines peuvent être détectées tôt en raison d'actions précédentes, d'indices subtils ou de facteurs externes. La définition des points d'actions renforce la compréhension du problème.

La notion de précocité est présente dès le début, car l'objectif est d'établir des systèmes interactifs fluides. Elle se traduit également dans la métrique utilisée (latency-aware).

2.1.2.3 Online Detection of Action Start (ODAS)

La tâche ODAS (*Online Detection of Action Start*) vise à détecter le début d'un geste dans un flux vidéo **non segmenté**, ainsi que classifier l'action. Dans cette tâche, la détection doit être réalisée dans le contexte en ligne, c'est-à-dire que le début doit être localisé au moment où celui-ci s'effectue.

La tâche est introduite en 2018 dans le travail de Shou et al. [Sho+18]. La tâche ODAS se concentre uniquement sur le début de l'action. Ils ont ainsi proposé une métrique de performance Point-mAP (Point mean Average Precision) qui compare le point de début prédit avec le point annoté.

2.1.2.4 Online Temporal Action Localization (OnTAL)

La tâche de l'Online Temporal Action Localization (OnTal) a pour objectif de localiser temporellement les actions dans un flux vidéo **non segmenté**, en déterminant les bornes temporelles de début et de fin de chaque action. Cette tâche peut être réalisée soit avec un léger délai (qualifié de "*soft-online*"), en trouvant le début de l'action après avoir observé la fin de l'action, soit en temps réel sans requalification des détections ("hard-online").

Dans l'article de Kim et al. [KNK22], la tâche OnTal est explicitement définie et discutée, mais elle est finalement présente assez tôt dans la littérature, notamment avec les premières méthodes qui se sont évaluées sur la base MAD [Hua+14], qui appelaient parfois cette tâche "Online Action Detection". En effet, certaines approches disaient adresser la tâche d'OAD (au niveau instance), comme [Bou+18a] (pour l'évaluation sur la base MAD) ou [YCL20], alors qu'ils s'évaluaient dans un contexte "soft-online". En pratique, cela se traduisait souvent par la classification de l'ensemble des frames d'une fenêtre glissante (la classification de la première frame de la fenêtre a été réalisée notamment grâce à la dernière frame de la fenêtre, il y a donc requalification de frames passées). Mais dans ce contexte, les contraintes ne sont pas les mêmes, car selon notre conception de la tâche OAD, aucune requalification des frames passées n'est possible. Nous pourrions en revanche rassembler les tâches OAD et la tâche OnTal dans le contexte "hard-online", mais les objectifs restent différents, OnTal vise à effectuer une segmentation, alors que OAD vise à détecter une action le plus tôt possible.

La métrique couramment utilisée pour évaluer les approches sur la tâche OnTal est l'Intersection over Union (IoU) avec un seuil pour considérer la détection comme étant correcte (par exemple, IoU>0.5).

2.1.3 Conclusion sur les tâches de reconnaissance en ligne

Dans cette partie consacrée à la classification des tâches dans le contexte de la reconnaissance de gestes en ligne, nous avons identifié et décrit un ensemble de tâches qui couvrent un large éventail d'objectifs et d'applications. Ces tâches se répartissent en deux catégories principales : les tâches au **niveau frame** et les tâches au **niveau instance**.

Au niveau frame, les tâches visent à fournir une classification précise et en temps réel des gestes à chaque frame (Action Prediction et OAD) d'une séquence. La tâche d'anticipation a pour objectif de prédire la future action dans un nombre de frames futures. Ce qui caractérise ces tâches est qu'il n'est pas nécessaire d'avoir un mécanisme de décision,

car la stabilité des prédictions n'est pas rédhibitoire. D'ailleurs, les approches qui adressent ces tâches utilisent généralement des fonctions de coût par frame, sans objectif de stabilité. De même, les métriques qu'ils emploient sont des métriques au niveau frame. Ces tâches trouvent des applications dans un contexte où il faut impérativement donner une réponse à n'importe quel moment (par exemple le flux de données qui s'arrête brutalement), ce qui contraste avec le niveau instance où les systèmes peuvent ne pas émettre de décision à certains moments.

Au niveau instance, les gestes doivent être détectés comme des entités, des instances, à un niveau de compréhension supérieur à celui de la frame. Ici, la stabilité est un prérequis afin de construire des instances au fur et à mesure du flux de données. Les applications visées sont toutes celles qui nécessitent un plus haut niveau de compréhension, sans avoir une contrainte impérative de décision à chaque instant, on y trouve la plupart des applications existantes et imaginables utilisant des interactions gestuelles.

Dans le cadre de cette thèse, nous nous concentrons sur deux tâches spécifiques : la **Early Gesture Recognition** (segmenté, décrite en section 2.1.2.1,) et l'**Online Action Detection** (OAD, non segmenté, décrite en section 2.1.2.2), toutes les deux au niveau de l'**instance**. La reconnaissance précoce de gestes se rapporte davantage au domaine 2D, où l'objectif est de reconnaître les gestes le plus tôt possible. D'autre part, l'OAD est plus adaptée au domaine 3D, mais pourrait également être appliquée à de nouvelles applications 2D. Son objectif est de détecter et reconnaître les gestes dans un flux de gestes, également le plus tôt possible.

En somme, cette partie tente de mettre un peu d'ordre dans les tâches existantes dans la littérature. Elle a pour objectif de mieux cerner à quelles tâches notre objectif se rattache, et de comprendre les différences cruciales entre elles. Se rattacher à une tâche concrète permet de trouver des sources d'inspirations, des approches auxquelles se comparer, mais aussi des **métriques** intéressantes afin d'évaluer son système.

2.2 Métriques pour la détection de geste en ligne

2.2.1 Philosophie générale de l'évaluation de la détection en ligne

Le niveau frame et le niveau instance qualifient également les métriques pour l'évaluation sur les tâches de reconnaissance en ligne.

Au niveau frame, les métriques comparent la prédiction du modèle frame par frame, sans prendre en compte la cohérence temporelle entre les prédictions successives. Une métrique courante à ce niveau est l'*Accuracy par frame* (décrite en 2.2.2.1), qui mesure la proportion de frames correctement classifiées par rapport au nombre total de frames.

Au niveau instance, les métriques évaluent la performance du modèle en termes de détection des **instances** d'action. Par exemple, pour la tâche de segmentation, la métrique BOffD (décrite en 2.2.3.3) mesure la qualité des détections en termes de correspondance entre les détections proposées et les véritables instances d'action, en utilisant un seuil d'Intersection over Union (IoU) pour évaluer la précision de la localisation temporelle des actions.

En ce qui concerne la précocité, les métriques par frame mesurent généralement la capacité du modèle à reconnaître l'action après avoir observé une certaine proportion du geste. Par exemple, on peut évaluer la reconnaissance à 10%, 20%, 30% d'observation, etc. Au niveau instance, on peut évaluer la précocité en se basant sur la borne temporelle de début de l'action, c'est-à-dire en mesurant le délai entre le début réel de l'action et le moment où le modèle commence à la reconnaître. On peut aussi évaluer la précocité en se basant sur le point d'action.

Le tableau 2.3 résume les différentes métriques, ainsi que les tâches pour lesquelles elles sont généralement utilisées pour les tâches en ligne. Les prochaines sections détaillent les différentes métriques. Nous verrons que face aux limites des métriques existantes concernant la tâche d'OAD au niveau instance, nous en proposerons une nouvelle (*Bounded Online Detection* - BOD) que nous détaillerons dans le chapitre 4 dédié à nos travaux concernant le geste non segmenté, dans la section 4.4.4.2.

Les métriques en non segmenté doivent disposer des **annotations des bornes de début/fin pour chaque instance de gestes** pour la plupart, ou alors des **annotations des points d'actions** pour chaque instance pour les autres.

2.2.2 Métriques basées sur la performance au niveau frame

2.2.2.1 Per frame Accuracy et Fscore

L'Accuracy est la métrique la plus simple pour évaluer les performances des systèmes de détection en ligne. Elle mesure le nombre de frames correctement classifiées par rapport au nombre total de frames.

$$Accuracy = \frac{TP}{N} \tag{2.1}$$

TABLE 2.3 – Métriques utilisées dans le contexte de la reconnaissance de gestes en ligne. * indique nos contributions.

	Métrique	**Mesure Qualité**	**Tâches**
Niveau Frame	Smooth Ratio-Prediction[Liu+20a]	Reconnaissance	OAD (frame)
	Accuracy par frame	Reconnaissance	OAD (frame) Action Prediction
	DAP Simplifié[BAM17]	Reconnaissance et Précocité	OAD (frame)
	mAP (par frame)[De +16]	Reconnaissance	OAD (frame), Action Anticipation
	Instantaneous Accuracy [Bap+20]	Reconnaissance	OAD (frame)
Niveau instance	TAR, FAR, RR[Che+17]	Reconnaissance	Early Gesture Recognition
	IoU[Eve+10]	Segmentation	OnTAL
	BOffD[Li+16]	Détection et Segmentation	OnTAL, OAD (instance)
	Ap/mAP (instance)[Eve+10]	Détection et Segmentation	OnTAL, OAD (instance)
	P-MAP[Sho+18]	Détection et Précocité	ODAS
	Action-Based Score[BMA12]	Détection et Précocité	OAD (instance)
	Latency-Aware Score[Fot+12]	Détection et Précocité	OAD (instance)
	DAP[Bou+18b]	Détection et Précocité	OAD (instance)
	SL/EL Score[Li+16]	Précocité (SL) et Segmentation (SL/EL)	OAD (instance)
	NTtoD[HD12]	Précocité	OAD (instance), Early Gesture Recognition
	IoU$_{st}$ *	Segmentation partielle	OAD (instance), OnTAL
	BOD *	Détection	OAD (instance)

où *TP* désigne les frames *True Positive* (correctement classifiées) et N est le nombre total de frames.

Sur le même principe, il est possible de calculer la précision, le rappel et le F-score pour chaque classe d'action.

Précision :

$$\text{Précision} = \frac{\text{TP}}{\text{TP} + \text{FP}} \tag{2.2}$$

Rappel :

$$\begin{gathered} \text{Rappel} = \frac{\text{TP}}{\text{TP} + \text{FN}} \\ \text{ou} \\ \text{Rappel} = \frac{\text{TP}}{\text{Nombre d'unités (frames) appartenant à la classe}} \end{gathered} \tag{2.3}$$

F-score :

$$\text{F-score} = 2 \times \frac{\text{Précision} \times \text{Rappel}}{\text{Précision} + \text{Rappel}} \tag{2.4}$$

Ces métriques peuvent être utilisées pour évaluer les systèmes tant dans le contexte segmenté que non segmenté.

Elles peuvent être calculées soit en faisant la moyenne par séquence indépendamment de leur longueur (macro). Soit en comptant indépendamment des séquences en regardant le nombre total de frames dans toutes les séquences (micro). Pour faire une micro moyenne, cela revient à additionner tous les TP, FP et FN de toutes les séquences avant de faire les calculs de *précision/rappel/fscore*, dans le cas où il ne peut y avoir qu'un FP/TP ou FN par unité (comme pour une évaluation par frame), on obtiendra $précision = rappel = fscore = accuracy$, dans ce cas il est plus préférable de ne mentionner que l'accuracy. Ce n'est pas le cas si par exemple plusieurs FP sont possibles par unité (par instance de geste par exemple). Pour faire une macro-moyenne, il faut calculer les scores de *précision/rappel/fscore* pour chaque classe indépendamment, puis faire une moyenne non pondérée des *fscore* afin d'obtenir le *fscore* final. Également, une distinction peut être faite sur les frames n'appartenant à aucune action en les pondérant différemment, en les excluant de l'évaluation, ou en les considérant comme des négatifs (dans les calculs de précision/rappel). Il faut donc être prudent lors de l'interprétation et l'utilisation de ces métriques en fonction des différents cas. La façon exacte de calculer ces scores devrait être précisée à chaque fois.

Ces métriques peuvent également être utilisées pour évaluer la précocité de la détection des actions en utilisant un pourcentage d'observation du geste. Cela permet de mesurer la capacité du modèle à reconnaître l'action après avoir observé une certaine proportion du geste.

Leurs utilisations au niveau frame **ne permet pas de capter les instabilités** dans les prédictions du système. Par exemple, si le système prédit une frame sur deux la bonne classe, le score sera de 50% concernant cette action. Cependant, d'un point de vue applicatif dans un contexte interactif, ce genre d'instabilité rend le système inutilisable.

Ces métriques peuvent être utilisées avec des instances de gestes à condition que la nouvelle unité soit les instances (et non les frames) et que les critères de TP/FP soient définis très précisément, c'est le cas pour un grand nombre de métriques au niveau instance définies dans les sections suivantes.

2.2.2.2 Smooth Ratio-Prediction

La métrique "Smooth Ratio-Prediction" a été utilisée dans l'article de Liu et al. [Liu+20a], mais n'a jamais été explicitée dans le papier [1]. Cette métrique est calculée sur des portions du geste p à l'aide de la formule suivante.

$$SRP(p) = \frac{1}{N}\sum_{n=1}^{N}\frac{1}{F_{n,p}}\sum_{f=1}^{F_{n,p}}\begin{cases} 1, & if pred(f) = GT(f) \\ 0, & else \end{cases} \tag{2.5}$$

où N est le nombre de gestes effectués, p est le pourcentage d'observation, et $F_{n,p}$ est l'indice relatif (commençant à 1) de la frame pour l'instance du geste n qui correspond à p% d'achèvement du geste.

Ces différents scores obtenus pour chaque pourcentage p peuvent permettre d'établir une courbe. Celle-ci peut être difficile à interpréter, car le score à un pourcentage p dépend de l'ensemble de la performance de 0% à p%, un exemple de courbe est illustré en figure 2.2.

Tout comme l'*Accuracy par frame*, cette métrique **ne permet pas de capter correctement les instabilités** des prédictions du système, même si le lissage effectué permet d'avoir une idée de la performance globale du système sur toutes les frames du geste.

1. La formule de cette métrique a été obtenue à partir d'une conversation avec l'auteur

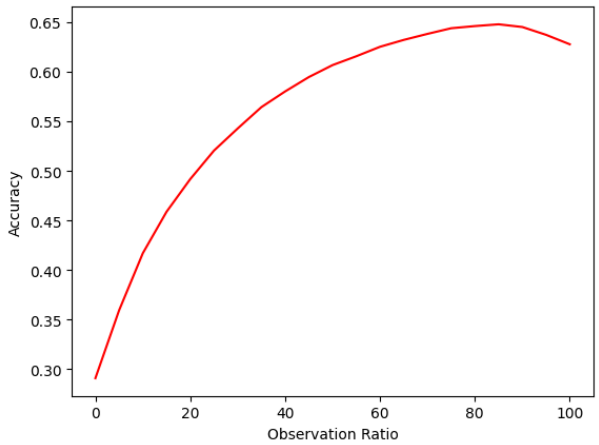


FIGURE 2.2 – Exemple de courbe que l'on peut obtenir avec la métrique "Smooth Ratio-Prediction" (SRP). L'axe des abscisses représente le pourcentage p d'observation, et l'axe des ordonnées représente le score.

2.2.2.3 Detection to Action Point - simplified (DAPs)

Proposée par Bloom et al. [BAM17], la métrique DAP simplifiée calcule l'accuracy à chaque frame avant un point d'action (AP - *Action Point*) de la vérité terrain :

$$DAPS(f) = Accuracy(AP - f) \forall f \in \{0..20\} \tag{2.6}$$

où f désigne l'indice relatif au point d'action d'une frame, Accuracy(AP-f) désigne l'Accuracy (telle que définie dans la section de Accuracy par frame) des frames situées f frames avant le point d'action annoté de chaque geste de l'ensemble de test.

Pour calculer cette métrique, il faut donc **disposer des annotations des points d'action**. Elle mesure la capacité d'un système à prédire avant le point d'action, dans les zones d'ambiguïtés.

Cette métrique produit une courbe avec un nouveau point pour chaque frame. Par rapport à la métrique précédente, l'interprétation est plus facile le score correspond à la performance de chaque frame indépendamment. Cependant, tout comme les métriques précédentes, **la cohérence temporelle du système n'est pas prise en compte**, les classes détectées pouvant être différentes d'une frame à l'autre.

2.2.2.4 Per-Frame AP, mAP et mcAP

La métrique "Average Precision" (AP) [De +16] est utilisée pour évaluer la performance des systèmes de détection d'actions au niveau frame (il existe une version au niveau instance que nous verrons en section 2.2.3.5). Étant données les probabilités prédites pour toutes les frames d'une classe d'action, on commence par les trier par ordre décroissant. Ensuite, à chaque étape, on calcule la précision en considérant les n premières frames triées. Puis, on calcule la moyenne de ces précisions pour obtenir l'Average Precision (AP).

$$AP_c = \frac{\sum_n Prec_c(n) I(n)}{P_c} \tag{2.7}$$

où $Prec_c(n) = \frac{TP(n)}{TP(n)+FP(n)}$. FP$(n)$ sont les frames prédites à tort comme appartenant à la classe d'action c parmi les n premières frames (triées par ordre de confiance). $I(n)$ est un indicateur qui vaut 1 si la frame n est un vrai positif et P_c est le nombre total de positifs (le nombre total de frames appartenant à la classe d'action c).

En calculant la moyenne de l'AP pour toutes les classes d'action, nous obtenons la mAP au niveau des frames.

$$mAP = \frac{1}{C} \sum_{c=1}^{C} AP_c \tag{2.8}$$

où C est le nombre total de classes d'action.

L'avantage principal de l'AP est qu'il permet de calculer un score indépendamment du seuil de confiance sélectionné.

La version « calibrée » a été proposée dans le but de pondérer l'importance des frames n'appartenant à aucune action (*background*). En effet, sur certaines bases ces frames sont prépondérantes. On la calcule en utilisant $cPrec$ à la place de $Prec$:

$$cPrec = \frac{TP}{TP + \frac{FP}{w}} = \frac{w * TP}{w * TP + FP} \tag{2.9}$$

w correspond au ratio du nombre de frame du *background* sur le nombre de frame appartenant à un geste. w est parfois calculé en prenant toute frame n'appartenant pas à la classe considérée, dans le but d'équilibrer toutes les classes.

Là encore, cette métrique, comme l'ensemble des métriques par frame, **ne permet pas de capter les instabilités du système**.

2.2.3 Métriques basées sur la performance au niveau instance

2.2.3.1 TAR, FAR et RR

Développés pour le **contexte segmenté**, le TAR, FAR et RR sont des métriques pertinentes pour des tâches de reconnaissance au **niveau instance** [Che+17].

Le taux d'acceptation correct (TAR - *True Acceptance Rate*) mesure la précision du classifieur lorsque la prédiction est acceptée. Le taux d'acceptation erronée (FAR - *False Acceptance Rate*) mesure le taux d'erreur lorsque la prédiction est acceptée. Pour mesurer le rejet final, le taux de rejet (RR - *Reject Rate*) représente le nombre d'échantillons qui ne sont jamais acceptés. En référence à la notation de la Tableau 2.4, le TAR, le FAR et le RR sont définis comme suit :

$$TAR = \frac{N_A^T}{N}; FAR = \frac{N_A^F}{N}; RR = \frac{N_R}{N} = 1 - TAR - FAR \tag{2.10}$$

Dans le contexte de la reconnaissance précoce segmentée, seule la classification de **la première décision** est utilisée pour calculer le TAR et le FAR. Si le geste n'est jamais accepté, il est pris en compte dans le RR. Notez que $TAR + FAR + RR = 1$.

Le Tableau 2.4 présente les notations utilisées pour la mesure des systèmes de rejet, telles que celles présentées dans [Che+17]. N_A^F correspond aux échantillons qui sont mal classés, mais acceptés par le système de rejet, tandis que les échantillons N_A^T sont correctement classés et acceptés.

TABLE 2.4 – Notations utilisées pour la mesure des systèmes basés sur le rejet, utilisées dans [Che+17]. N_A^F représente les échantillons de test qui sont mal classés, mais acceptés par le système de rejet, tandis que les échantillons N_A^T sont correctement classés et acceptés.

Ensemble d'échantillons (N)	Option de rejet	
	Accepter (N_A)	Rejeter (N_R)
Correctement classés (N_{cor})	Acceptation correcte $\left(N_A^T\right)$	Rejet erroné $\left(N_R^F\right)$
Mal classés (N_{err})	Acceptation erronée $\left(N_A^F\right)$	Rejet correct $\left(N_R^T\right)$

Le TAR, FAR et RR sont généralement utilisés pour caractériser le résultat général, mais il est possible de regarder la progression du TAR, FAR et RR en fonction de la complétion du geste. Il est possible de construire la courbe de la figure 2.3 en regardant pour chaque pourcentage de complétion quels sont les gestes qui ont été acceptés ou rejetés avant ce pourcentage. Le score au moment complétion de 100% correspond au TAR, FAR

et RR finaux. Des comparaisons entre approche sur ce type de courbe sont possibles, attention cependant à ce que signifie la complétion, elle peut être calculée en fonction du temps ou du déplacement. Pour le 2D, c'est généralement plus pertinent d'utiliser le déplacement comme unité de complétion.

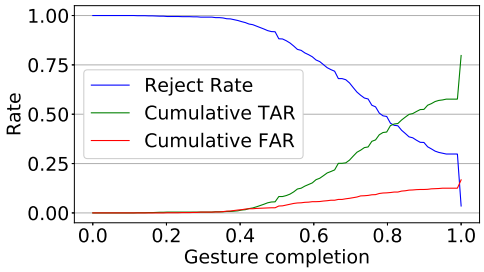


FIGURE 2.3 – Exemple de courbes TAR, FAR et RR progressive.

Les métriques NTtoD ou NDtoD (détaillées en section 2.2.3.10) peuvent être utilisées en complément de ces valeurs pour avoir une évaluation de la précocité du système, qui vient compléter l'observation de la courbe. Elle n'est calculée que sur les gestes correctement classifiés (TAR).

Ces métriques nous semblent **pertinentes** pour notre tâche sur **l'aspect segmenté**, notamment du fait de la **prise en compte de la possibilité de ne pas émettre de décision** pour une instance de geste. L'unité de cette métrique est l'instance du geste, cette instance peut être bien classifiée (TAR), mal classifiée (FAR) ou rejetée (RR, absence de décision). La précocité est mesurée avec une autre métrique indépendante.

En revanche, elle n'est **pas pertinente sur l'aspect non segmenté**. Elle ne prendrait pas en compte les décisions intervenues entre deux gestes, et elle n'a pas été conçue afin d'évaluer plusieurs décisions par geste.

2.2.3.2 Intersection Over Union (IoU)

L'Intersection over Union (IoU) [Eve+10] est une métrique d'évaluation utilisée à l'origine pour la détection d'objets. Elle calcule la qualité de la segmentation en mesurant le degré de recouvrement entre la prédiction et la vérité terrain. Elle est utilisée pour la segmentation de gestes en se basant sur les bornes temporelles, comme illustrée dans la figure 2.4, en divisant l'aire de l'union par l'aire de l'intersection :

$$IoU = \frac{\text{Aire de l'intersection}}{\text{Aire de l'union}} \tag{2.11}$$

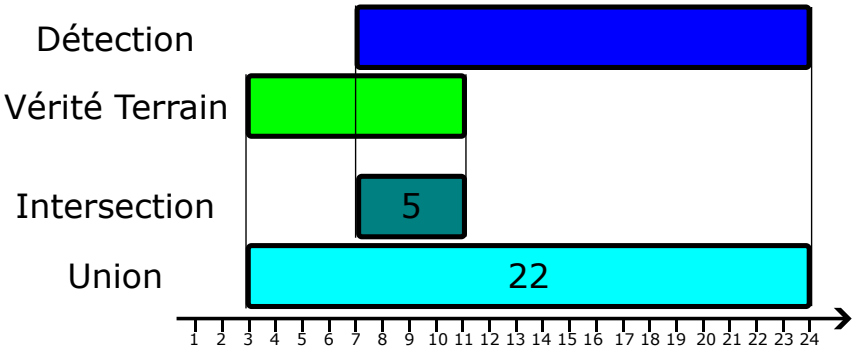


FIGURE 2.4 – Calcul de l'*Intersection Over Union*. Dans cet exemple IoU $= 5/22 \approx 0.23$.

L'IoU est particulièrement pertinente lorsque l'annotation est fiable, car elle permet de quantifier précisément la qualité de la segmentation.

L'IoU est souvent utilisée comme indicateur dans d'autres métriques, telles que BOffD (décrite dans la section suivante) et mAP (instance) (décrite en section 2.2.3.5). Dans ces métriques, on regarde si l'IoU dépasse une certaine valeur pour considérer une détection comme correcte. Par exemple, dans le cas de la métrique BOffD, on considère une détection comme correcte si l'IoU entre la prédiction et la vérité terrain dépasse un seuil Δ donné.

L'IoU est donc un indicateur important pour évaluer la qualité de la segmentation d'actions en ligne, en mesurant à quel point les prédictions du modèle correspondent aux vérités terrain annotées par des humains.

L'IoU mesure la **qualité de la segmentation** et donc n'est pas directement pertinente pour notre tâche, en revanche son utilisation avec d'autres métriques disposant d'un seuil $\Delta = 0$ permet d'évaluer la qualité de la détection. Nous en reparlerons dans les métriques concernées.

2.2.3.3 Bounded Offline Detection (BOffD)

La métrique Bounded Offline Detection (BOffD) a été utilisée dans l'article de Li et al. [Li+16]. Bien que cette métrique n'ait pas de nom spécifique dans la littérature, elle est souvent désignée de manière générique sous le nom de "F1-Score". Pour plus de clarté, nous proposons de l'appeler BOffD ici.

La métrique BOffD est conçue pour évaluer la qualité des détections en utilisant l'Intersection over Union (IoU). Elle permet de définir un critère stable pour dire si une segmentation est correcte en comparant l'IoU entre la prédiction et la vérité terrain avec un seuil prédéfini Δ. Si l'IoU est supérieur à Δ, la détection est considérée comme cor-

recte (TP). Un fscore est ensuite calculé. Δ peut être mis à 0 pour évaluer une détection, indépendamment du recouvrement. Un exemple d'application de cette métrique se trouve dans la figure 2.5.

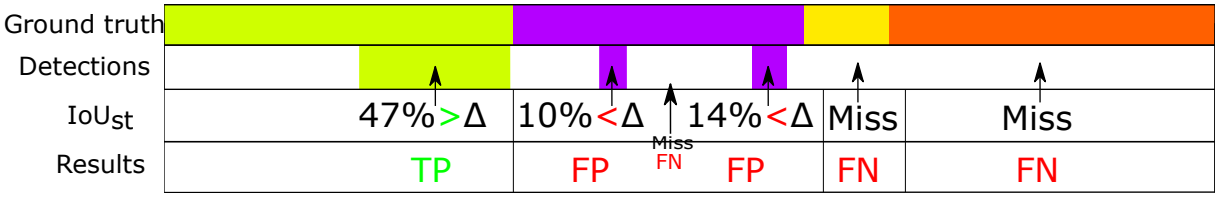


FIGURE 2.5 – Exemple d'application de la métrique Bounded Offline Detection (BOffD) avec $\Delta = 30\%$. Précision=$\frac{TP}{TP+FP} = \frac{1}{3} \approx 0.33$; Rappel = $\frac{TP}{TP+FN} = \frac{1}{4} = 0.25$; Fscore = $2 \times \frac{\text{Précision} \times \text{Rappel}}{\text{Précision} + \text{Rappel}} = 2 * \frac{0.33 * 0.25}{0.33 + 0.25} \approx 0.28$. Le Fscore pour la métrique BoffD pour cette séquence est donc de 0.31. Pour évaluer sur plusieurs séquences, il est possible de faire une micro ou macro-moyenne comme détaillé en 2.2.2.1.

Une notion qui n'est pas souvent explicitée dans les articles est la gestion des **doubles détections**. Dans les implémentations disponibles dans d'autres domaines, il semble que détecter une nouvelle instance sur un geste déjà correctement détecté soit considéré comme un faux positif. En revanche si l'instance avait été mal classifiée, et que la nouvelle détection est une bonne classification, cette dernière est considérée comme une bonne détection.

La métrique BOffD est une bonne mesure de la qualité des détections, mais **elle peut poser des problèmes lorsqu'elle est utilisée en ligne avec un Δ non nul**. En effet, du fait de l'exigence de chevauchement, la précocité est implicitement considérée, car un grand chevauchement implique de commencer à prédire l'action suffisamment tôt. De plus, certaines prédictions de gestes pourraient ne jamais excéder Δ. Par exemple, un geste qui n'est reconnaissable que sur la dernière frame du fait d'une grande partie commune avec un autre geste, aurait une valeur d'IoU très faible, même avec un système parfait. Dans de tels cas, il peut être préférable d'utiliser la métrique Bounded Online Detection (BOD), qui sera présentée ultérieurement dans nos contributions (cf. section 4.4.4.2).

2.2.3.4 SL-Score/EL-Score

Le SL et EL Score [Li+16] sont des métriques utilisées pour évaluer la qualité de segmentation des détections en ligne. Le SL-Score compare le début prédit avec le début

réel. Le EL-score est calculé de la même manière, mais pour la borne de fin du geste.

$$SL_{Score} = e^{-\frac{|t_s - t_{start}|}{t_{end} - t_{start}}} \tag{2.12}$$

$$EL_{Score} = e^{-\frac{|t_e - t_{end}|}{t_{end} - t_{start}}} \tag{2.13}$$

t_{start} est l'index de la frame du début du geste (vérité terrain), t_{end} est l'index correspondant à la fin du geste, t_s et t_e sont respectivement les instants prédits comme étant le début et la fin du geste.

Le SL-Score peut être utilisé pour mesurer la précocité, comme alternative au NTtoD. Les SL et EL-Scores sont liés à un critère de vrais positifs (TP), de faux positifs (FP) et de faux négatifs (FN), généralement utilisé dans les articles avec la métrique Bounded Offline Detection (BOffD). Lorsqu'une détection est correcte (TP), la valeur de la métrique est calculée en fonction de la distance entre le moment de détection et le moment réel de début ou de fin de l'action. En revanche, si la détection est incorrecte, la valeur de la métrique est mise à 0, pénalisant fortement le score (le meilleur score est 1). Pour le SL-Score, ce dernier point constitue la différence majeure avec le NTtoD qui ne prend pas en compte les détections incorrectes dans son calcul.

2.2.3.5 AP et mAP (instance)

La métrique Average Precision (AP) et la mean Average Precision (mAP) [Eve+10] sont des métriques couramment utilisées pour évaluer les systèmes de détection d'actions **hors ligne**. Il ne faut pas les confondre avec leur version par frame.

À l'origine, ces métriques ont été développées pour la détection d'objets dans des images. Elles permettent de prendre en compte plusieurs bornes de début/fin prédites avec des chevauchements possible, et s'évaluent en se basant sur le score de confiance de chaque prédiction.

Les étapes pour calculer l'AP sont les suivantes :

— classer les prédictions par ordre de confiance décroissante ;

— considérer les prédictions TP (True Positive) si l'Intersection over Union (IoU) est supérieur à un seuil Δ, souvent noté mAP@Δ (par exemple, mAP@0.5) ;

— calculer la précision et le rappel au fur et à mesure, dans l'ordre classé précédemment, en considérant seulement les prédictions faites avec une confiance supérieure ou égale ; ;

— calculer une précision interpolée à l'aide d'une interpolation, dont la plus courante est donnée par l'équation 2.14 ;
— enfin, on calcule le mAP en utilisant la précision interpolée sur différentes valeurs de rappel (équation 2.15)

$$p_{int}(r, \theta) = \max_{r' \geq r} p\left(r', \theta\right) \tag{2.14}$$

$$mAP(\theta) = \frac{1}{C} \sum_{c=1}^{C} \frac{1}{m_c} \sum_{k=1}^{m_c} p_{int}\left(r_{ck}, \theta\right) \tag{2.15}$$

où C est le nombre de classes, θ est le seuil d'IoU, r_{ck} est le rappel à la prédiction de rang k (dans l'ordre de confiance), m_c est le nombre d'échantillons dans la classe c.

Cette métrique est intéressante, car elle permet de s'abstraire du seuil de confiance choisi pour la détection. Cependant, elle suppose que plusieurs bornes peuvent être prédites au même endroit.

Ces métriques ne sont **pas adaptées au contexte en ligne**. En effet, pour l'OAD, on cherche à avoir une compréhension à chaque instant de ce qui se passe dans la vidéo, que ce soit une absence d'action, une action en cours ou une action qui vient de se terminer. Il faudrait donc que le système soit capable de générer des bornes de début et de fin à différents niveaux de confiance à chaque instant, ce qui **n'est pas toujours réalisable**.

Ces métriques peuvent être plus facilement applicables pour la tâche OnTAL (Online Temporal Action Localization) dans un contexte "soft-online" où une requalification est effectuée pour ajuster les bornes de début et de fin prédites.

2.2.3.6 P-mAP

Le Point-mAP (P-mAP) [Sho+18] est une métrique utilisée spécifiquement pour la tâche ODAS (Online Detection of Action Start).

Le P-mAP est basé sur le même principe que le mAP décrite juste avant (cf. section 2.2.3.5). Cependant, la différence réside dans le critère de positivité. Alors que le mAP utilise un seuil d'Intersection over Union (IoU) pour considérer une prédiction comme positive, le P-mAP utilise un seuil de distance entre le début de l'action prédit et le début réel.

2.2.3.7 Action Based Score

Le Action-Based score est une métrique proposée par Bloom [BMA12] pour évaluer la qualité de prédiction et la précocité.

Cette métrique se concentre sur la précocité de la détection, en évaluant si une action est prédite au niveau du début annoté. Le critère de précocité est binaire : soit la détection est précoce (prédite avant 4 frames après le début de l'action), soit elle ne l'est pas.

Les détections doubles sont pénalisées dans cette métrique. Cela signifie que si le système détecte plusieurs fois une même action, seule la première détection précoce est comptée comme TP, et les détections supplémentaires sont comptées comme FP.

Cette métrique **manque de nuance dans la mesure de la qualité de la détection**. En effet, il est facile d'imaginer que certains gestes ne soient reconnaissables que dans les dernières frames du geste.

2.2.3.8 Latency Aware Score

Le Latency-Aware F-score [Fot+12] est une métrique qui évalue la précocité des systèmes de détection d'action en ligne en tenant compte d'une fenêtre de latence autour du point d'action annoté. Cette métrique nécessite des annotations très fiables du point d'action, car elle évalue les détections dans une fenêtre de $\pm\ \Delta$ frames autour du point d'action réel. Δ est fixé à 10 dans toutes les expérimentations effectuées dans la littérature.

Le critère de précocité du Latency-Aware F-score est binaire : une détection est considérée comme correcte si elle est détectée autour du point d'action annoté dans la fenêtre de latence de ± 10 frames. Dans ce cas, la détection est comptée comme un vrai positif (TP). Si la détection se produit en dehors de cette fenêtre de latence, elle est considérée comme un faux positif (FP). Le Fscore final s'obtient à partir de la formule du Fscore donnée en 2.2.2.1 (equation 2.4), mais en considérant l'unité instance plutôt que frame.

Contrairement à certaines autres métriques, le Latency-Aware F-score **ignore les doubles détections**. Cela signifie que si le système détecte plusieurs fois une même action dans la fenêtre de latence, seule la première détection est comptée comme TP, et les bonnes détections supplémentaires ne seront pas considérées (ni FP, ni TP). Un exemple d'application de la métrique est illustré en figure 2.6

Il est important de noter que le Latency-Aware F-score est **sensible à la qualité de l'annotation du point d'action**, car il évalue la précocité des détections par rapport à ce point. Les systèmes de détection doivent donc être évalués sur des données avec

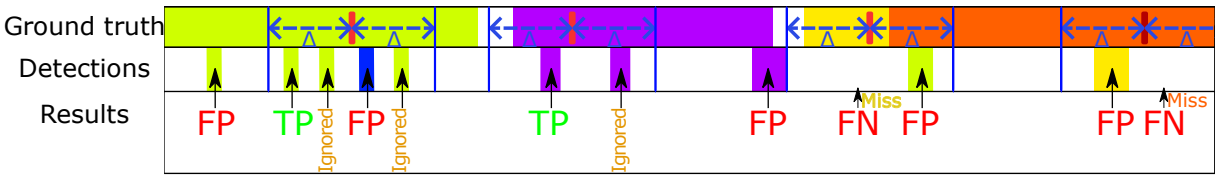


FIGURE 2.6 – Exemple d'application de la métrique latency aware. Les traits rouges indiquent les points d'action annotés. Ici on a précision $= \frac{2}{7} \approx 0.29$; Rappel $= \frac{2}{4} = 0.5$; Fscore ≈ 0.36. Le latency aware f-score pour cette séquence est donc de 0.36. Le score prenant en compte plusieurs séquences peut être une micro ou macro-moyenne comme expliqué en 2.2.2.1.

des annotations du point d'action fiables pour obtenir des résultats significatifs avec cette métrique. Certains travaux se sont affranchis de cette contrainte en évaluant les détections dans une fenêtre de latence autour de la borne de début de l'action plutôt que du point d'action exact.

2.2.3.9 Detection to Action Point (DAP)

La métrique Detection to Action Point (DAP) a été proposée dans l'étude de Boulahia et al. [Bou+18b] pour évaluer spécifiquement la détection précoce et mesurer le degré de précocité des systèmes de détection d'action en ligne. Bien qu'elle n'ait pas reçu de nom spécifique dans l'article original, nous l'appelons DAP ici pour plus de clarté.

La philosophie générale derrière la métrique DAP est la suivante :

— si le système se trompe dans ses détections, il a la possibilité de se corriger jusqu'à ce qu'il trouve la bonne réponse. Cependant, toutes les erreurs commises sont comptées, ce qui signifie que les fausses détections sont considérées comme des faux positifs (FP) ;

— toutes les erreurs sont comptées à chaque instant d'évaluation, même celles qui se produisent après l'instant d'évaluation. Cela permet d'évaluer la qualité des détections à différents moments dans le temps ;

— la première détection correcte, qui se produit avant l'instant d'évaluation, est comptabilisée comme un vrai positif (TP). Les autres détections correctes après cet instant sont également comptées comme TP. Avant d'atteindre la première détection correcte dans l'instant d'évaluation, il y a une fausse négative (FN) supplémentaire.

En utilisant cette approche, la métrique DAP permet de construire une courbe qui représente l'évolution du nombre de TP, FP et FN en fonction du temps. Cette courbe permet de visualiser la précocité des détections et d'évaluer la capacité du système à

détecter les actions avant leur point d'action annoté. La logique [2] permettant de construire la courbe est visible dans l'*algorithme* 1.

Cette métrique est pertinente pour la tâche d'OAD, mais elle nécessite **une annotation du point d'action**.

2.2.3.10 NTtoD/NDtoD

La métrique Normalized Time to Detection (NTtoD) [HD12] ou sa variante NDtoD (D pour Distance) a pour objectif d'évaluer la précocité d'un système. Cette métrique est une alternative au SL-Score plus adapté pour mesurer la précocité, notamment car la valeur est plus facilement interprétable.

NTtoD mesure le temps normalisé (par rapport à la longueur de l'action) entre le moment où l'action débute (selon l'annotation) et le moment où elle est détectée par le système. Cette métrique n'est comptabilisée que pour les vrais positifs, elle est doit donc être liée à une autre métrique qui mesure la performance de classification (comme BOD, ou les TAR, FAR, RR).

$$\text{NTtoD} = \frac{start(pred) - start(GT)}{end(GT) - start(GT) + 1} \tag{2.16}$$

Elle a été utilisée dans la littérature notamment dans les travaux de Molchanov et al. [Mol+16] et Chen et al. [Che+17].

2.2.4 Conclusion sur les métriques

Pour la tâche de reconnaissance précoce dans un contexte segmenté, les métriques utilisées dans la littérature nous semblent pertinentes, le TAR, FAR et RR, couplé à une métrique afin d'évaluer la précocité comme la NTtoD.

Du côté des métriques pour la tâche d'OAD, le choix est plus difficile. D'un côté, il y a les métriques de détection avec précocité « binaire », Latency-Aware Score et Action-Based Score. D'un point de vue qualité de détection, ces métriques ne sont pas utilisables pour toutes les bases de données. Concernant la Latency-Aware Score, il faut l'annotation du point d'action, qui n'est généralement pas annoté sur les bases, de plus les doubles détections positives sont ignorées alors que d'un point de vue applicatif il serait plus cohérent de les considérer comme un faux positif. Concernant le Action-Based Score, le

2. Algorithme obtenu grâce à une conversation avec l'auteur

Algorithm 1 Fonction d'évaluation "Detection to Action Point" (DAP), métrique utilisée dans [Bou+18b].

```
size(detections) # le nombre de détection
classe(detection) # l'identifiant de la classe prédit
time(detection) # l'instant de la detection

#entrée:
AP: int # le point d'action
begin: int # le début de l'action
end: int # la fin de l'action
classeID_GT :int # id de la classe, vérite terrain
allDetections :Liste de Detection # toutes les detections sur la
    séquence


#init
nbFrameApresAP = (end-AP) # le nombre de frame entre end et AP
TP = [0,0,0...0] # size(TP)==20+nbFrameApresAP
FP = [0,0,0...0] # size(FP)==20+nbFrameApresAP
FN = [0,0,0...0] # size(FN)==20+nbFrameApresAP

#coeur de l'algo
#detections: toutes les détections qui ont lieu entre le debut et la
    fin du geste
detections = allDetections[begin:end]
for (t = 20 ; t>=-nbFrameApresAP ; t--):
 indext = 20-t #juste pour le positionnement dans les tableaux TP/FP/FN

 if size(detections)==0:
  FN[indext]++
 else: #size(detections)>=1
     # si la bonne classe n'a pas encore predite entre debut et AP-t =>
        FN
  if classeID_GT not in classes(detection[:AP-t]):
   FN[indext]++

  for (i=0 ; i<size(detections) ; i++) :
   if classe(detection[i])==classeID_GT AND
               (classeID_GT not in classes(detection[:i])) AND
               time(detection[i])<=AP-t:
    TP[indext]++
   else:
    FP[indext]++

#calcul final du score
for each time t:
 Precision[t] = TP[t]/(TP[t]+FP[t])
 Recall[t] = TP[t]/(TP[t]+FN[t])
 FMeasure[t] = (2*Precision[t]*Recall[t])/(Precision[t]+Recall[t])
```

choix est fait de considérer la détection comme positive à partir du moment où elle est faite suffisamment tôt par rapport au **début** de l'action annotée (ex : 4 frames après le début du geste). Cependant, l'ambiguïté entre des gestes sur certaines bases rend cette métrique inadéquate et non équitable entre les différents gestes, car certains gestes seront reconnaissable plus tard que d'autres. Dans ces deux métriques, la précocité est considérée comme un prérequis pour valider une détection. Bien que notre objectif soit de détecter les gestes le plus tôt possible, il nous semble important de devoir dissocier qualité de détection et précocité. En effet, l'ambiguïté entre les gestes à leur début peut être très variable suivant les bases, sur certains gestes il faudra attendre le dernier moment avant que celui-ci soit discriminable, pour d'autres il pourra être reconnu dès les premières frames. Pour ces mêmes raisons, la métrique P-MAP n'est pas pertinente. La métrique DAP, avec sa courbe associée, est plutôt pertinente pour notre objectif, cependant elle nécessite l'annotation du point d'action. De plus l'interprétation de la courbe peut être difficile étant donné que toutes les erreurs sont considérées dès le début de la courbe. La métrique BOffD semble finalement la plus pertinente pour l'évaluation de la tâche. Comme la tâche d'OAD n'a pas besoin de segmentation particulière, nous pouvons l'utiliser avec un seuil d'IoU égal à zéro. Cependant, nous avons relevé deux problèmes avec cette métrique. La première est le manque de clarté lors de ses premières utilisations pour le geste dans l'article de Li et al. [Li+16], en effet la façon dont sont considérées les doubles détections n'est notamment pas claire. Deuxièmement, cette métrique, originellement développée pour la détection d'objet dans une image est davantage pertinente pour les tâches hors ligne. En effet, lorsque le seuil d'IoU est supérieur à zéro, cela implique implicitement d'avoir une précocité minimale afin de couvrir suffisamment l'annotation du geste. Même si nous comptons utiliser cette métrique avec un seuil d'IoU à zéro, où la deuxième remarque n'est plus valable, il est dommage d'utiliser une métrique qui devient non pertinente en changeant le seuil. Pour ces deux raisons, nous avons développé une métrique fortement inspirée de BOffD qui a pour objectif d'être explicite avec un paramètre dédié à la gestion des doubles détections. De plus, cette nouvelle métrique reste pertinente lorsque l'exigence d'IoU augmente, en considérant une version modifiée de l'IoU. Cette métrique, nommée BOD (Bounded Online Detection) sera présentée dans les contributions dans la section 4.4.4.2.

2.3 Prise de décision et mécanismes de rejet pour une sortie niveau instance

2.3.1 Philosophie de la prise de décision

Nous souhaitons adresser le problème de la reconnaissance de gestes au niveau instance, il faut donc des mécanismes de décision permettant de gérer ce niveau d'analyse. Nous présentons ici une modélisation originale du problème et du fonctionnement général des systèmes en ligne pour cette tâche. Cette modélisation permet d'expliquer l'ensemble des approches réalisées jusqu'à présent.

Expression du système. Dans un premier temps, il faut déjà que le système soit en capacité de « s'exprimer », afin qu'il puisse décider de quelque chose. Son expression va passer par la sortie du système. Étant donné l'aspect temporel en ligne de la tâche, il faut qu'il puisse émettre plusieurs sorties, en lien avec différents instants d'entrée (frames), afin de potentiellement décider d'une détection à chaque étape. Classiquement, des systèmes dits « **séquence-à-séquence** » (au sens large) sont utilisés afin de produire une sortie pour chaque entrée dans un contexte en ligne.

Instants de prise de décision L'élément primordial qu'il faudra considérer dans les deux tâches que nous adressons est l'ambiguïté entre les gestes. Lorsque des gestes ont des débuts communs, il est impossible pour un système de reconnaître le geste sur ces sections. Dans l'exemple de la figure 2.7, il est impossible de discriminer les gestes parmi l'ensemble représenté dès le début du tracé. Il faut attendre que certaines portions du geste soient passées, ces portions dépendent des similarités avec les autres gestes de l'ensemble considéré. **Tous les gestes ne seront pas discriminables au même moment**, par exemple le geste E sera reconnaissable dès les premiers instants, car c'est le seul geste qui commence vers le bas. En revanche pour reconnaître le geste A ou B, il faudra quasiment attendre le dernier moment.

Un cas particulier qui est souvent oublié est le cas du geste jaune D dans la figure. Ce geste constitue une portion d'un autre geste (en l'occurrence de plusieurs dans l'exemple). Pour discriminer ce geste par rapport aux autres, il est obligatoire de savoir qu'il est terminé. Si reconnaître la fin dans un contexte segmenté n'est pas problématique (quoiqu'il faut permettre au système de s'exprimer après la fin du geste), cela pose un problème important dans le cadre du non segmenté : il n'est possible de savoir s'il s'agit du sous

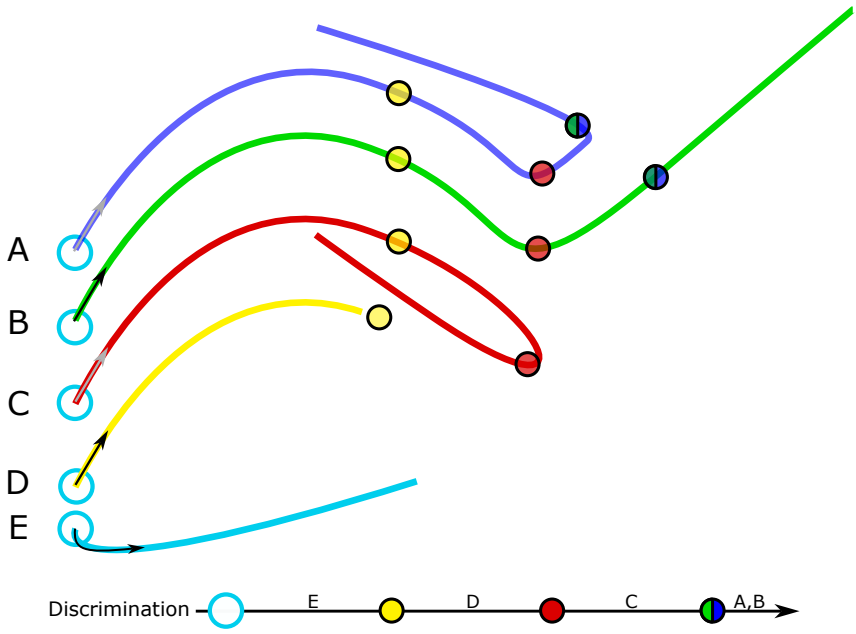


FIGURE 2.7 – Illustration des instants de discrimination, assimilable aux points d'actions théoriques, dans un ensemble de gestes (A,B,C,D,E). Tous les gestes ne peuvent pas être reconnus aux mêmes moments.

geste uniquement avec la suite du tracé. D'un point de vue applicatif, si l'application a pour objectif d'offrir une interaction fluide, alors il vaut mieux tout simplement éviter de choisir des gestes qui sont des sous-parties d'autres gestes, car de toute façon ces gestes seront reconnus au minimum à 100% de leur complétion.

Un autre cas qui pourrait poser problème est celui du geste vert *B*. Il sera toujours un minimum risqué de tenter de prédire le geste B sur la dernière portion du geste, car un changement de direction (vers la gauche) pourrait faire basculer ce geste en catégorie *A*. En effet, du fait que le geste B ne change pas de direction après la bifurcation entre A et B provoque ce problème. Il est cependant possible pour un système de considérer que le tracé a duré suffisamment longtemps dans la même direction pour dire que le geste soit le B, où « suffisamment » dépendra de la variabilité des exemples d'apprentissage.

Bien que ces deux cas semblent répondre à des situations peu fréquentes, ils sont pourtant présents dans les bases 2D existantes (telles que ILGDB et MTGSetB que nous décrirons dans la section liée aux expérimentations). Il a en effet été montré [Li+12] que des gestes qui disposent de parties communes (par exemple liées à un type de commande) sont plus facilement mémorisables par l'utilisateur. Disposer d'un début commun permet de rassembler des commandes liées à la même nature d'action, par exemple le geste qui

produit l'action « copier » pourrait avoir un début commun avec les gestes des actions « coller » et « couper ».

Nature de la sortie Étant donnée une expression de type séquence-à-séquence, le système émettra une sortie à chaque étape d'entrée (voir la première étape de la figure 2.8). Cependant, nous avons vu qu'il est parfois impossible de prédire correctement la catégorie du geste avant une certaine étape. Il faudra donc que le système soit en capacité d'exprimer cette absence de reconnaissance.

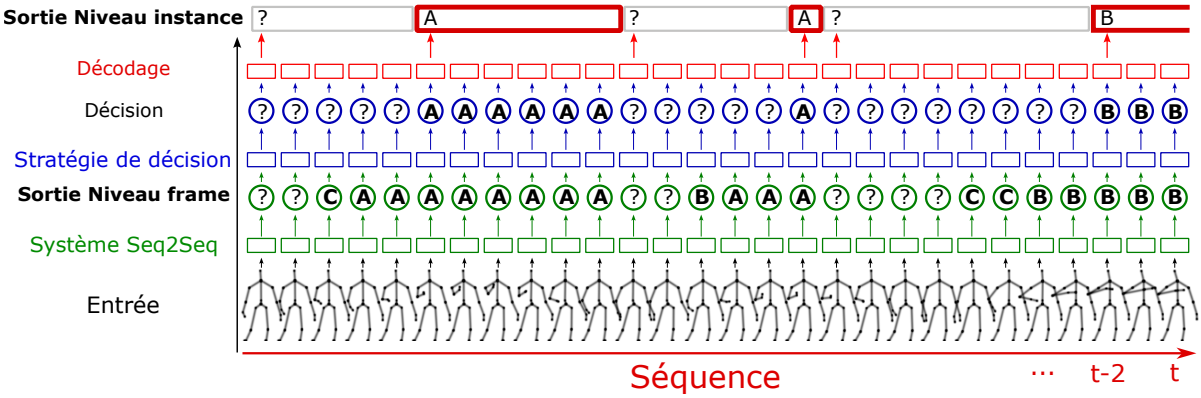


FIGURE 2.8 – Modélisation d'un système pour la reconnaissance de geste en ligne, ici dans un contexte non segmenté. La stratégie de décision utilisée dans cet exemple est la *répétition* de trois gestes identiques.

Dans le cadre non segmenté, il faut en plus considérer qu'il y a des instants de « non-geste », particulièrement du côté du geste 3D. Ces instants peuvent être courts, comme des mouvements de transitions entre deux gestes, ou longs comme des phases de repos. Étant donnée la tâche de détection de gestes que nous adressons, ces instants peuvent être exprimés de la même manière que les instants d'ambiguïté par le système.

Afin d'émettre l'absence de reconnaissance, il peut être nécessaire d'utiliser une **stratégie de décision** qui se basera sur la sortie du système séquence-à-séquence. Cette stratégie aura pour objectif de décider pour chaque frame si elle est rejetée (absence de reconnaissance) ou si elle est reconnue (et précisera la classe du geste). La stratégie peut être de complexité variable en fonction de la nature du système séquence-à-séquence, ces deux éléments doivent être en adéquation. Par exemple, le système séquence-à-séquence peut être entraîné pour prédire une classe spéciale au moment des non-geste ou des ambiguïtés, ce qui simplifie le travail de la stratégie. Celle-ci peut être triviale en considérant la même instance de geste tant que la sortie niveau frame est la même, la détection com-

mencera donc dès le premier instant où le système a prédit une catégorie. La stratégie peut également être plus complexe, comme se baser sur des scores de confiance, et sur un nombre de répétitions. Nous pouvons distinguer deux natures de stratégies de décision, celles de type **récurrentes** qui se servent des résultats de la stratégie à l'étape précédente, et celles de type **instantanées**, qui effectuent chaque décision de manière indépendante.

D'un point de vue purement applicatif, une compréhension à un niveau plus élevé est primordiale pour être utilisable. En effet, l'application va par exemple effectuer une commande par rapport au geste effectué par l'utilisateur. Pour pouvoir le faire, le geste ne doit être détecté qu'une seule fois. Il faut donc un mécanisme de décodage afin de traduire la sortie au niveau frame en une sortie au niveau instance. Cette étape de décodage est parfois implicite dans les approches, mais son explicitation permet d'y voir plus clair quant à la philosophie des approches et de leurs évaluations. Le passage par ce décodage est obligatoire pour permettre l'évaluation avec les métriques de niveau instances. Un diagramme récapitulatif est présent dans la figure 2.8.

En résumé, deux endroits peuvent donc principalement intervenir pour obtenir la décision finale, le système séquence-à-séquence, et la stratégie de décision.

Pour étudier l'existant concernant les mécanismes de décision, distinguons les deux tâches que nous adressons dans cette thèse : la reconnaissance précoce de gestes segmentés (*Early Gesture Recognition*), et la détection précoce de gestes non segmentés (*Online Action/Gesture Detection*).

2.3.2 Décision pour le contexte segmenté (Early Gesture Recognition)

Pour rappel, la tâche de reconnaissance précoce de gestes a pour objectif de reconnaître le geste le plus tôt possible dans un contexte segmenté.

Dans un contexte segmenté, on suppose que l'application peut segmenter les gestes de manière triviale. Par exemple, sur une surface 2D, lorsque seuls des gestes mono-strokes sont utilisés, la segmentation est faite à chaque lever de doigts/crayon. Ainsi, dès la première décision qui sera émise par le système de reconnaissance, la commande associée pourra être effectuée et l'application pourra se mettre en attente du prochain geste.

Étant donné le problème d'ambiguïté précédemment mentionné, un système doit notifier la classe du geste seulement à partir du moment où celui-ci est reconnu. Dans ce cadre segmenté, le système pourra employer n'importe quelle technique de rejet classique

sans besoin d'avoir une cohérence temporelle étant donné que seule la première décision d'acceptation est importante. Dans les techniques de rejet classiques on retrouve [Mou07] les classes de rejet, les classifieurs spécialisés, les classifieurs spécialisés basés sur les scores de confiance et l'application de seuils sur les scores de confiance.

Le tableau 2.5 résume les méthodes traitant spécifiquement de la tâche de reconnaissance précoce de gestes segmentés ainsi que leurs choix de stratégie de décision.

Méthode	**Geste**	**Type de Seq2Seq**	**Type de sortie Par frame**	**Stratégie de décision**
[Kaw+11]	3D	Template Matching	Confiance par Classe	Seuils relatifs
[Ko+13]	3D	Pose Matching	Confiance par Classe	Seuils absolus
[Che+17]	2D	Template Matching	Confiance par Classe Distance aux prototypes	Seuils relatifs Seuils absolus Répétitions

TABLE 2.5 – Résumé des méthodes traitant de la reconnaissance précoce de gestes segmentés au niveau instance. « Seuils relatifs » signifie que des seuils sont appliqués sur les différences de scores entre les classes, par exemple entre les deux classes les plus probables. Les seuils absolus sont eux appliqués directement sur un score. Ces seuils peuvent être appris de manière automatique.

L'aspect que nous traitons dans cette sous-partie est la notion de stratégie de décision dans un contexte segmenté. Nous allons d'abord, en section 2.3.2.1, voir un exemple de classifieur qui émet un score de confiance, le SVM. Le SVM a longtemps été un classifieur privilégié de par ses performances et sa bonne capacité de généralisation. Les réseaux de neurones, bien que nécessitant généralement plus de données pour apprendre, ont fait preuve de meilleures performances au cours des dernières années. Le problème est qu'il a été démontré [Guo+17] que les scores retournés par les réseaux de neurones récents ne reflètent pas naturellement un indice de confiance. Il est cependant possible de les calibrer afin d'obtenir un score qui reflète davantage un score de confiance, c'est ce que nous verrons dans la section 2.3.2.2. Enfin, dans la section 2.3.2.3 nous verrons qu'il est possible d'apprendre directement une stratégie de décision au sein même du réseau.

2.3.2.1 Classifieurs avec mesure de confiance possible

Les SVM, ou machines à vecteurs de support, sont utilisés pour résoudre des problèmes de classification et de régression. L'idée principale des SVM est de trouver la meilleure séparation entre différentes classes de données dans un espace multidimensionnel. En

général, des caractéristiques sont extraites des données puis projetées dans un espace plus pertinent.

L'objectif des SVM est de trouver les hyperplans (séparateur dans l'espace) qui séparent au mieux les classes. Pour cela, le SVM cherche à maximiser les marges qui séparent chaque hyperplan avec les données de chaque classe. Le principe est illustré dans la figure 2.9.

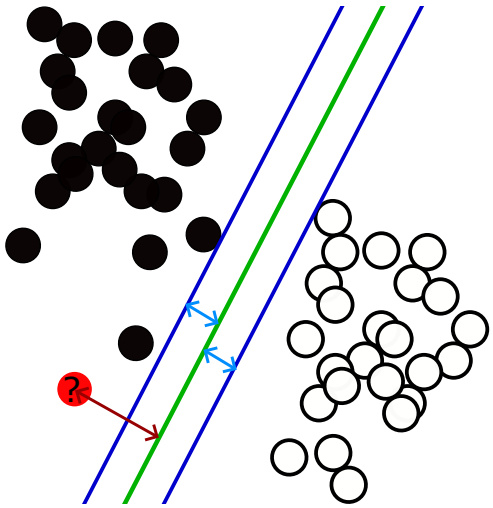


FIGURE 2.9 – Séparation des classes avec marge maximale dans les SVM.

Une fois que l'hyperplan est trouvé, il peut être utilisé pour prédire à quelle classe appartient un nouveau point de données en regardant de quel côté de l'hyperplan il se trouve.

De plus, la distance par rapport à l'hyperplan peut être utilisée pour calculer un indice de confiance, plus la donnée est loin de l'hyperplan, moins il semble y avoir un risque qu'elle soit confusive. La distance peut être alors directement utilisée comme score de confiance.

Chen et al. [Che+17] et Boulahia et al. [Bou+18a] utilisent ce principe afin d'évaluer le risque de *confusion* entre deux classes. Chen et al. calculent également des clusters (cf. figure 2.10), puis utilise la distance à leurs centres pour estimer un deuxième indice de confiance, permettant d'évaluer la *distance* par rapport aux classes. Ce deuxième indice permet notamment de savoir si l'exemple correspond à l'une des classes possibles.

Dans le cadre du *template matching*, il est également possible d'utiliser les distances aux *templates* comme indicateur de confiance, comme le fait Kawashima et al. [Kaw+11] et Ko et al. [Ko+13].

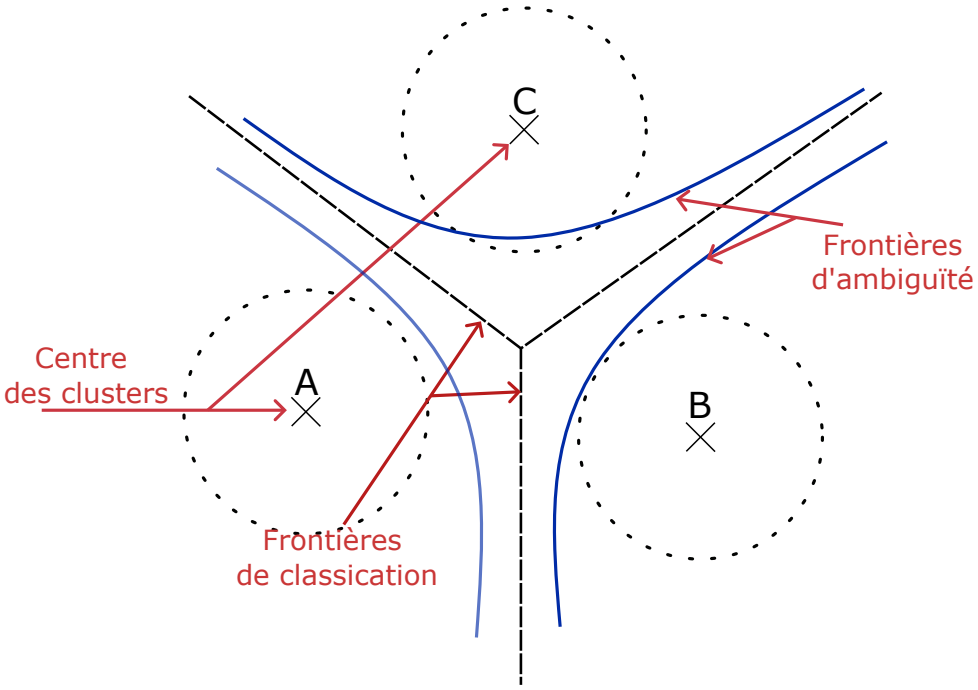


FIGURE 2.10 – Afin d'élaborer un rejet de distance, des clusters peuvent être calculés afin de désigner la zone de l'espace qui est plausible pour les exemples. En calculant la distance séparant le centre du clusters avec les exemples, il est possible de voir à quel point l'exemple est proche des autres [Che+17].

2.3.2.2 Calibrer les classifieurs neuronaux

Guo et al. [Guo+17] ont montré dans une étude que la plupart des réseaux neuronaux modernes n'étaient pas « calibrés ». En effet, lorsque le réseau prédit un score de 90% pour une classe d'action, il ne se trompe pas neuf fois sur dix. D'autant plus que les différences observées entre le score de prédiction et l'erreur effective sont souvent très variables. Par exemple, Guo et al. montrent (pour une architecture et une base de donnée précise, mais qui reflète un cas général) que les prédictions faites avec un score entre 80 et 90% sont erronées 50% du temps.

Les scores émis par les réseaux de neurones modernes ne sont donc pas directement utilisables comme indice de confiance. Cependant, il est possible de calibrer les réseaux, ce qui revient à appliquer un post-traitement sur les scores reçus. Les paramètres de ce post-traitement doivent être calculés sur un ensemble de validation afin de retomber sur des scores qui sont de nouveau calibrés. Parmi ces méthodes, on peut trouver la méthode de calibration "temperature scaling" [Guo+17] qui est une approche simple à mettre en place, et efficace en termes de résultats.

Ainsi, en appliquant une calibration sur des réseaux modernes, il serait possible d'avoir un indice de confiance qui serait plus fiable pour effectuer du rejet.

2.3.2.3 Apprendre la stratégie de décision au sein de classifieurs neuronaux

Plutôt que d'utiliser une stratégie de décision en se basant sur les scores et distances émis par un classifieur, il est également possible de faire apprendre cette stratégie au sein même de ce classifieur. Le classifieur aura alors une sortie spéciale pour signifier la présence ou l'absence de la décision.

Geifman et al. ont présenté **SelectiveNet** [GE19] afin d'avoir un mécanisme de rejet directement intégré dans leur réseau profond. Ils l'appliquent dans un contexte sans notion de temporalité, où chaque image peut être acceptée ou refusée.

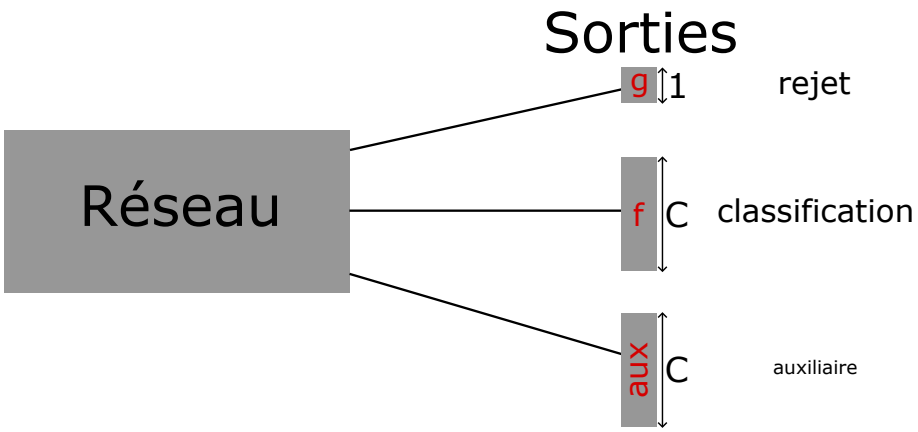


FIGURE 2.11 – Illustration des sorties pour l'application du rejet dans SelectiveNet [GE19].

Deux nouvelles sorties sont ajoutées au réseau, comme illustré dans la figure 2.11 : la tête de rejet (g) et la tête auxiliaire (aux, qui n'a pour objectif que de faciliter l'apprentissage). La tête de rejet se termine par un seul neurone. La fonction d'activation Sigmoid est utilisée pour cette tête afin de fixer la sortie entre 0 et 1. L'objectif de cette sortie est d'accepter ou de rejeter la prédiction. La prédiction est considérée comme rejetée si la sortie de la tête de rejet est inférieure à un paramètre τ (en pratique mis à 0.5), et acceptée si elle est supérieure. La décision de sortie finale par rapport à g est définie comme suit :

$$(f, g_\tau)(x) = \begin{cases} f(x), & \text{if } g(x) \geq \tau \\ ? & \text{sinon.} \end{cases} \tag{2.17}$$

où $f(x)$ correspond à la sortie de classification.

La fonction de coût principale est définie comme suit :

$$\mathcal{L}(f,g) = \frac{1}{m}\sum_{i=1}^{m} \ell\left(f\left(x_i\right), y_i\right) * g\left(x_i\right) + \lambda\Psi\left(c - \hat{\phi}\left(g\right)\right) \tag{2.18}$$

où l est une fonction de coût pour la classification, (typiquement cross-entropy), c est la couverture cible, λ est un hyperparamètre relatif à l'importance de la contrainte de couverture et $\Psi(a) = \max(0,a)^2$. $\hat{\phi}\left(g\right)$ est la couverture empirique, c'est-à-dire la valeur moyenne de g.

La première partie de la fonction $\ell\left(f\left(x_i\right), y_i\right) * g\left(x_i\right)$ est donc le coût de la fonction de classification classique, à laquelle est multipliée la sortie de sélection/rejet g. Ainsi, le réseau peut minimiser l'impact de l'erreur de classification en prédisant une petite valeur g. Cependant, g doit en moyenne être proche de c afin de minimiser la deuxième partie de la fonction de coût $\Psi\left(c - \hat{\phi}\left(g\right)\right)$, qui revient à minimiser l'écart entre c et le g moyen. À noter que du fait de la nature de la fonction Ψ, la valeur de $\hat{\phi}\left(g\right)$ peut dépasser c sans être pénalisée (c'est-à-dire avoir une quantité d'acceptation supérieure à l'exigence c).

Nous adapterons cette approche dans nos travaux afin d'avoir un rejet sur la dimension temporelle, et non pas par échantillon (section 3.2.3).

2.3.3 Décision pour le contexte non segmenté (Online Action Detection)

La tâche d'Online Action Detection consiste à détecter les gestes le plus tôt possible dans un flot de gestes continu.

Dans ce contexte, les gestes ne peuvent pas être segmentés de manière triviale, il faut donc que le système réussisse à distinguer les différentes instances de gestes. Pour cela il lui faudra une certaine stabilité au cours du temps. En effet il faut que d'une frame à l'autre le système soit cohérent en termes de décision. Pour obtenir cette stabilité, le système séquence-à-séquence doit être robuste, ou alors la stratégie de décision doit permettre cette stabilité. Dans l'exemple de la figure 2.8, le système séquence-à-séquence n'était pas particulièrement robuste, car il se trompait en début de geste, ce qui est un comportement plutôt classique étant donné le peu d'information disponible. En revanche la stratégie de décision très sécurisée (répétition de trois fois la prédiction) permet de ne pas prendre en compte les erreurs du système séquence-à-séquence dans la décision finale. Les différentes

stratégies de décision présentes dans les approches traitant de l'OAD (niveau instance) sont résumées dans la dernière colonne du tableau 2.6.

TABLE 2.6 – Résumé des méthodes traitant de la tâche OAD (niveau instance). La colonne « type de Seq2Seq » indique la méthode appliquée afin d'obtenir un système capable de faire du séquence-à-séquence. La colonne « non-geste (*background*) » indique la manière de gérer les instants de non-geste. La colonne « sortie par step » indique ce que le système émet comme sortie(s) à chaque instant. Les approches de [Mol+16 ; Bou+18b ; Li+16 ; Liu+19] seront détaillées dans les sections suivantes.

Méthode	**Type de Seq2Seq**	**Non-Geste (*background*)**	**Sortie par Step**	**Stratégie de décision**
[Fot+12]	Fenêtre Glissante	Classe dédiée	Confiance par classe	Seuil de confiance absolu
[Zha+13] [Zha+14]	Template Matching	Ident. rejet	Confiance par classe	Seuil de confiance absolu
[Web+14]	RNN	Classe dédiée	Confiance par classe	Répetition
[BMA14]	Template Matching	Ident. rejet	Distance aux templates	Matching avec un template
[Sha+15]	Fenêtre Glissante	Ident. rejet	Confiance par classe	Seuil de confiance absolu
[MHT16]	Par frame	Ident. rejet	Score par classe	Cumul temporel et seuils de score
[Mol+16]	RNN	Classe dédiée (blank)	Confiance par classe	
[BKK17]	Fenêtre Glissante	Classe dédiée	Classe, Confiance début et fin	Seuils de confiance début/fin, États
[BAM17]	Fenêtre Glissante	Ident. rejet	Distance aux templates	Matching avec un template
[Bou+18b]	Fenêtre Glissante	Ident. rejet	Confiances par classe	Cumuls de confiances relatives, Seuils de confiances
[Car+19]	RNN	Ident. rejet	Confiances par classe	Seuils de confiance absolu
[Li+16] [Liu+19]	RNN	Classe dédiée	Classe, Confiance début et fin	Seuils de confiance début/fin, Attente du maximum États

Les instants de *non-geste*, souvent appelés instants d'arrière-plan (*background*) (ou bien inter-gestes, de repos, de transitions, ou encore gestes "perturbateurs") sont définis par tous les instants qui ne sont pas des gestes qui appartiennent à l'ensemble des gestes que l'on souhaite classifier. Bien qu'elle puisse être assimilée à la notion de rejet, elle est souvent considérée comme une classe à part entière qui englobe tout ce qui n'est pas un vrai geste. Cependant, cela peut dépendre de la nature spécifique de ces instants dans les différents cas d'application. Dans la 3^{e} colonne du tableau 2.6, on peut voir comment les méthodes gèrent cette notion. Dans la littérature, elle a été gérée soit via le mécanisme de rejet, soit via une classe dédiée. Dans la plupart des approches listées dans le tableau, les systèmes séquence-à-séquence ne possèdent pas de mécanisme dédié à l'ambiguïté, celui-ci est présent dans la stratégie de décision.

Le tableau 2.7 divise les stratégies utilisées en deux catégories : les **stratégies instantanées** et celles **récurrentes**. Les stratégies instantanées permettent la prise de décisions pour chaque frame, **indépendamment des résultats des frames précédentes**. Elles permettent une prise de décision plus rapide que les stratégies récurrentes. Ces dernières permettent généralement une plus grande robustesse lors d'une prise de décision en **considérant les résultats précédents**.

TABLE 2.7 – Les stratégies de décision peuvent être réparties en deux catégories, instantanées et récurrentes

Stratégie de Décision	
Instantanée	**Récurrente**
Seuil de confiance absolu, Matching avec un template	Cumul temporel de score + seuil, Cumul de confiance relatives + seuil, Seuils de confiance début/fin (maximum) + États, Répétition
[Fot+12 ; Zha+13 ; Zha+14 ; BMA14] [Sha+15 ; BAM17 ; Car+19]	[Web+14 ; MHT16 ; Li+16] [BKK17 ; Bou+18b]

Concernant les approches qui utilisent des stratégies de décisions instantanées, ils s'évaluent systématiquement avec une méthode où la constance n'est pas un problème pour la métrique. La plupart s'évaluent avec la métrique Latency-Aware Score (cf. section 2.2.3.8) qui ne pénalise pas les doubles détections positives. L'absence probable de robustesse liée à la stratégie instantanée n'est donc pas entièrement visible avec la métrique utilisée. Cependant, les approches utilisant des systèmes séquence-à-séquence récurrents (RNN) [Car+19] sont probablement plus stables dans leurs prédictions d'un instant à l'autre du fait de la nature du réseau, ce qui peut justifier l'utilisation d'une stratégie instantanée.

Les approches utilisant des stratégies de décisions récurrentes utilisent plutôt des métriques plus sévères comme la BOffD (cf. section 2.2.3.3) ou la métrique DAP (cf. section 2.2.3.9), où les doubles détections positives sont comptées comme des faux positifs.

L'approche de Molchanov et al. [Mol+16] est cas particulier intéressant, car **elle n'utilise pas de stratégie de décision**. En effet, ils utilisent la fonction de coût CTC [Gra+06] (Connectionist Temporal Classification, détaillé en 2.3.3.3) pour apprendre leur système séquence-à-séquence. CTC étant construite afin de produire naturellement une sortie au niveau instance, elle rend une stratégie de décision supplémentaire inutile.

Le *blank* qui est le caractère spécial du CTC, sert à la fois pour les instants de non-geste, et pour les instants d'ambiguïté.

Dans les sous-sections suivantes, nous allons étudier les stratégies de décision récurrentes. Plus particulièrement nous allons voir une stratégie de cumul de confiance relative en section 2.3.3.1, une stratégie qui localise les instants de début/fin de gestes en section 2.3.3.2, et enfin nous détaillerons l'utilisation de la fonction CTC pour la prise de décision en section 2.3.3.3.

2.3.3.1 Cumul de confiance et seuils

Boulahia et al. [Bou+18b] dans leur approche E-CuDi3D (Early Curvilinear Distance 3D), mettent en place un processus de décision à partir d'une combinaison de classifieurs et de cumul de scores de confiance.

Trois modèles curvilignes distincts sont utilisés (cf. figure 2.12). Ces modèles traitent le flux d'entrée à court, moyen et long terme. Chaque modèle intègre autant de classifieurs que de classes d'actions. La taille de la fenêtre pour chaque classifieur est déterminée en moyennant les déplacements curvilignes des instances de sa classe d'action respective. Le premier modèle (à long terme) utilise des fenêtres équivalant à 100% de ces valeurs moyennes. Le modèle intermédiaire (à moyen terme) emploie des fenêtres de 50%, tandis que le dernier modèle (à court terme) utilise des fenêtres de 10%. Ces trois modèles fonctionnent en parallèle sur le flux de frames, chacun ayant son propre système de décision.

Le cumul des scores s'exprime avec la fonction suivante :

$$His_i(j) = \begin{cases} His_i(j) + \beta, & \text{si } j = Predicted_i \\ His_i(j) - \gamma, & \text{sinon} \end{cases} \tag{2.19}$$

où β et γ correspondent à des différentes de scores entre les différentes classes. L'idée est de qualifier l'intensité de la confiance pour la prédiction, et de mettre à jour l'histogramme en fonction de celle-ci.

Concernant la combinaison des résultats, une décision finale est émise lorsqu'un des trois modèles émet une décision. Les modèles émettent des décisions dès qu'un des seuils est dépassé. Dans l'exemple de la figure 2.12, la décision sera prise, car l'histogramme bleu dépasse son seuil sur le classifieur 2 du modèle à long terme (un seuil par modèle, par classifieur et par classe). Si plusieurs décisions sont prises en même temps par les différents

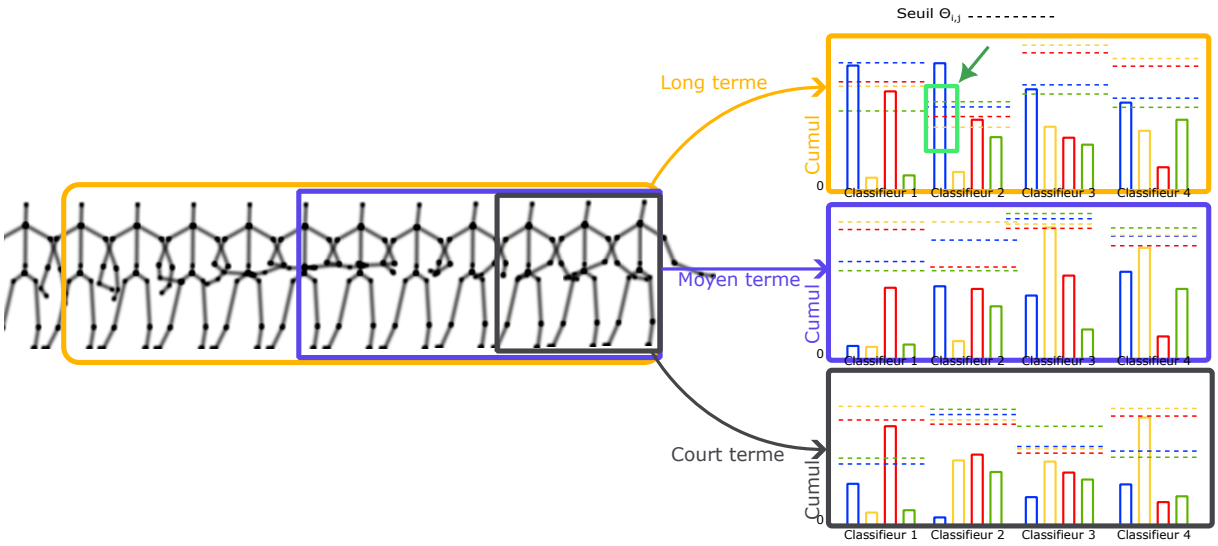


FIGURE 2.12 – Schéma de l'approche E-CuDi3D [Bou+18b]. Trois modèles (long, moyen et court termes) sont en compétition. Chaque modèle dispose d'autant de classifieurs que de classe. Un mécanisme de cumul de score de confiance est mis en place afin de mettre à jour des histogrammes. Les décisions locales des différents modèles sont ensuite combinées afin de permettre une décision finale.

modèles, alors la décision est mise en attente jusqu'à que l'ambiguïté soit levée. Les scores cumulés sont réinitialisés après une décision pour éviter les répétitions de détection.

2.3.3.2 Supervision de la détection des bornes de début et fin pour le niveau instance

Li et al. ont proposé JCR-RNN [Li+16], un réseau séquence-à-séquence pour la tâche d'OAD. L'objectif est d'apprendre au réseau à localiser les instants de début et fin, tout en classifiant chaque frame. Ainsi, à chaque frame le système prédit trois sorties : la classe, la probabilité que cela soit le début de l'action, et la probabilité que cela soit la fin de l'action. Avec ces trois sorties, il est possible d'obtenir une sortie au niveau instance. Les deux sorties qui prédisent les bornes sont supervisées par une fonction de coût de régression de type erreur moyenne quadratique (MSE), où la vérité terrain est représentée par une gaussienne centrée sur les débuts et fins du geste. Ce système a été repris dans leur approche plus récente [Liu+19].

Comme il s'agit d'un contexte en ligne « strict » (notre contexte par défaut), l'action prédite au moment où le début est détecté (détection d'un pic après un seuil θ) sera celle choisie pour tout l'intervalle de détection. Ce processus est illustré en figure 2.13.

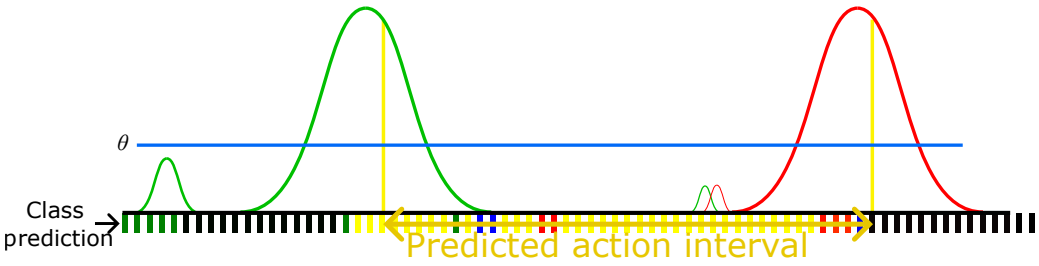


FIGURE 2.13 – Illustration de la stratégie de décision de JCR-RNN [Li+16]. Les courbes vertes (resp. rouges) indiquent les scores de sorties dédiées à la détection du début (resp. fin). Les traits verticaux indiquent les différentes frames, leurs couleurs représentent les classes prédites. Il est possible de prendre la décision de détection en observant les scores de prédiction de débuts et de fins, notamment en observant lorsque les scores commencent à redescendre après avoir passé un certain seuil θ.

On peut remarquer que ce fonctionnement permet de lisser l'ensemble de prédiction dans l'intervalle, ce qui est plutôt intéressant dans le but de l'utiliser pour un système interactif. En revanche, le gros point noir de cette approche est qu'elle suppose que le geste est reconnaissable dès son début. Elle pourrait difficilement marcher dans le cas où plusieurs gestes ont des débuts communs.

En relâchant la contrainte stricte du en ligne afin de faire de la segmentation qualifiée de « en ligne souple/flexible » (tâche OnTAL), il serait possible d'agréger les résultats obtenus pendant tout l'intervalle afin de classifier dans son intégralité.

Baek et al. [BKK17] utilisent également un processus de détection de début et de fin d'actions qui est assimilable à celui de Liu et al. et utilisable de la même manière.

2.3.3.3 La fonction CTC comme outil de prise de décision

La fonction de coût de Classification Temporelle Connexionniste (CTC, *Connectionist Temporal Classification*) [Gra+06] est généralement utilisée pour entraîner des systèmes lorsque la segmentation temporelle de la séquence d'entrée n'est pas annotée et que seul l'ordre des classes d'action est disponible. CTC est utilisé pour entrainer un modèle à produire un séquencement d'action (ex : $A, B, A, C...$) à partir des données d'entrée, conduisant naturellement à une sortie au niveau de l'instance. CTC est une fonction de coût qui est beaucoup utilisé pour la reconnaissance d'écriture, où l'annotation n'est généralement pas au niveau de la lettre (par exemple : [Gra+06 ; Sou+19 ; GZL21]). Il a rarement été utilisé dans la littérature pour la tâche OAD, cependant nous trouvons le travail de Molchanov et al.[Mol+16]. Selon une étude de Zeyer et al. [ZSN21], un système entrainé

avec cette fonction a tendance à prédire un « pic » de probabilité de classe dans très peu de frames, les autres frames sont qualifiées du label spécial "blank". De plus, les pics ne sont pas nécessairement situés sur les frames correspondant aux actions. Ils peuvent se produire après, voire avant si l'on utilise des systèmes hors ligne. Ce deuxième point découle du fait que la fonction CTC vise à optimiser tous les chemins possibles dans le graphe de la séquence conduisant à la séquence finale correcte de label (cf. figure 2.14). L'utilisation de CTC garantira la production de sorties au niveau instance.

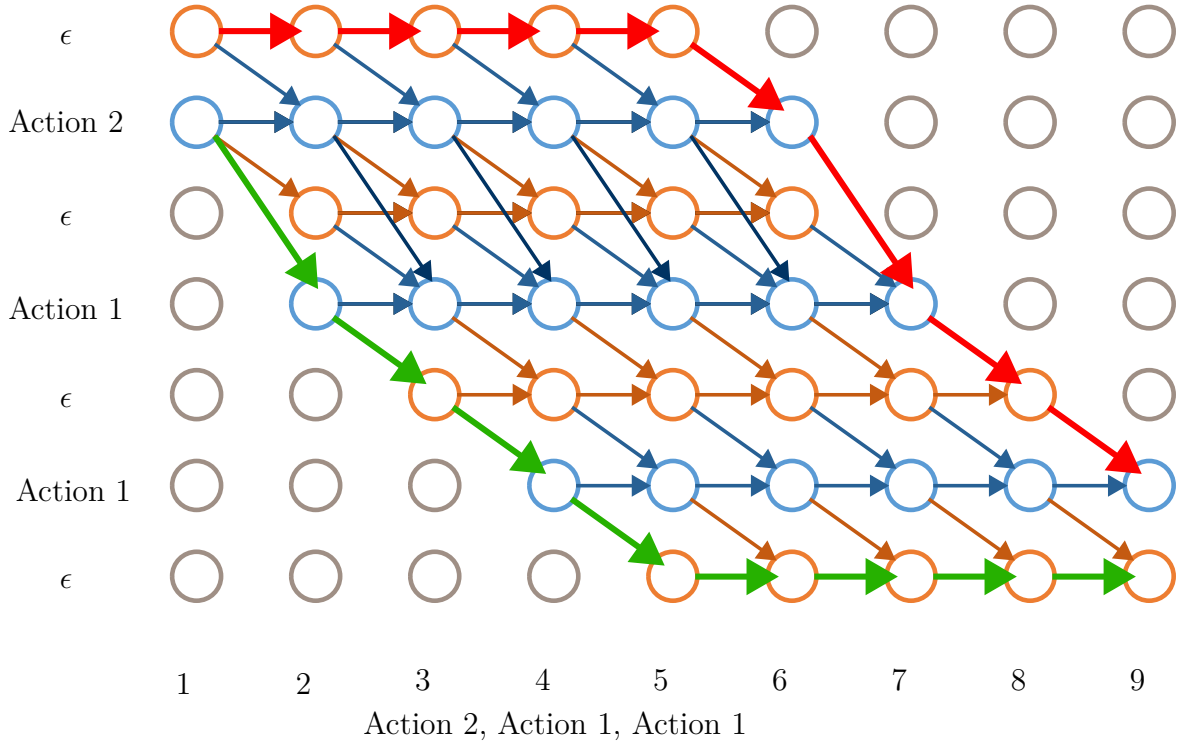


FIGURE 2.14 – Graphe du Connectionist Temporal Classification (CTC). Les deux chemins mis en évidence en rouge et en vert désignent les deux chemins extrêmes qui conduisent à la prédiction correcte de la séquence. Le chemin vert est la séquence 21ϵ1ϵϵϵϵ, tandis que le chemin rouge est ϵϵϵϵϵ21ϵ1.

Le CTC se formalise comme suit : soit $x = (\boldsymbol{E}^1, \boldsymbol{E}^2, ..., \boldsymbol{E}^F)$ la séquence d'entrée où F est le nombre de frames pour une séquence donnée. $l = \{1, ..., L\}$ est l'ensemble des labels de sortie possible et L est le nombre de labels. Soit $y = (y_1, y_2, ..., y_U)$ l'étiquette pour une séquence donnée où chaque $y_u \in l \cup \epsilon$, et ϵ est le symbole *blank*. y est composé des labels de classe ordonnés dans le temps avec des blank insérés avant chaque étiquette et à la fin. L'algorithme CTC apprend à prédire la séquence de sortie $\hat{y}$ étant donnée la séquence d'entrée x. Pour entraîner le réseau, nous devons d'abord construire un graphe

représentant tous les alignements possibles entre x et y. Par exemple, pour reconnaître la séquence « action 2, action 1, action 1 » en 9 frames, le modèle peut prédire « $\texttt{21}\epsilon\texttt{1}\epsilon\epsilon\epsilon\epsilon$ » (chemin vert dans la figure 2.14), « $\epsilon\epsilon\epsilon\epsilon\epsilon\texttt{21}\epsilon\texttt{1}$ » (chemin rouge) ou tous les chemins intermédiaires représentés dans la figure 2.14. Chaque nœud dans le graphe correspond à un alignement possible où le f-ième élément d'entrée est aligné à l'étiquette de sortie u-ième. Les arêtes entre les nœuds représentent les transitions valides entre les étiquettes. Une arête peut aller d'un nœud $n_{f,u}$ à un nœud $n_{f+1,u'}$ si la condition $\mathcal{C}$ est vérifiée :

$$\mathcal{C}(n_{f,u}, n_{f+1,u'}) = \begin{cases} y_{u'} \in \{y_u, y_{u+1}\} \\ \textbf{ou} \\ y_{u'} \in \{y_{u+2}\} \textbf{ et } y_{u'} \neq \epsilon \\ \qquad\qquad \textbf{et } y_{u'} \neq y_u \end{cases} \tag{2.20}$$

La fonction de coût $\mathcal{L}_{CTC}$ représente la somme des probabilités de tous les chemins corrects, elle est définie comme suit :

$$\mathcal{L}_{CTC} = -\log \sum_{\pi \in paths|\mathcal{C}} \prod_{f:F} p(n_{f,\pi_f}) \tag{2.21}$$

où $paths|\mathcal{C}$ représente tous les chemins conduisant à la séquence finale correcte de labels, où les transitions vérifient la condition $\mathcal{C}$. π_f est le label dans le chemin π à l'étape f. $p(n_{f,\pi_f})$ est la probabilité de sortie du modèle pour le label π_f à l'étape f.

2.3.4 Conclusion sur la prise de décision

Concernant la reconnaissance précoce de gestes segmentés, les stratégies de prise de décision sont plus simples que celles du non segmenté, car la notion de stabilité et de doubles détections n'est pas présente. Dans la littérature, ce sont plutôt l'utilisation de seuils qui permettent de prendre une décision. Bien que les scores émis par les réseaux profonds ne sont naturellement pas utilisés comme score de confiance, nous avons vu qu'il était possible de calibrer les réseaux afin d'obtenir des scores plus significatifs. Une fois la calibration faite, il est alors possible d'utiliser une technique de seuils comme certaines approches précédentes. Cependant, nous avons vu qu'il était possible d'intégrer un mécanisme de rejet directement dans le réseau, via le biais de SelectiveNet, mais aussi

indirectement via un apprentissage CTC (qui sera cependant plus pertinent pour le non segmenté).

Concernant la reconnaissance de gestes non segmentés, les stratégies de décision sont plus originales, notamment parce qu'une contrainte de stabilité s'ajoute dans ce contexte. Le tableau 2.8 résume les avantages/inconvénients des trois stratégies détaillées. Les stratégies qui reposent sur un seuil de cumul de confiance émis par différents classifieurs au fil du temps offrent un mécanisme explicite et contrôlable pour la détection précoce. Cependant, la mise en place de mécanismes de cumul et de choix de seuils peut être complexe et nécessite une calibration minutieuse. Les stratégies qui localisent le début et la fin des gestes encouragent fortement la précocité en signalant le début d'un geste. Cependant, cela implique que les gestes doivent être reconnaissables dès leur commencement et donc qu'il n'y a aucune ambiguïté entre les gestes. L'apprentissage avec la fonction de coût CTC est une approche qui permet de produire des sorties au niveau de l'instance. Elle est intégrable naturellement aux architectures de réseaux profonds. Cependant, contrairement aux deux autres stratégies, elle ne vise pas spécifiquement la détection précoce. Elle est plutôt adaptée pour les systèmes où la localisation précise des gestes n'est pas cruciale.

TABLE 2.8 – Récapitulatifs des avantages/inconvénients des différentes stratégies de décisions pour le contexte non segmenté.

Stratégie	**Avantages**	**inconvénient**
Cumul de confiance et seuils	Explicite, contrôlable	Complexité des mécanismes de cumul, combinaison et du choix de seuils
Localisation débuts et fins	Sortie niveau instance, Encourage fortement la précocité	Nécessite que les gestes soient reconnaissables dès leurs débuts
Apprentissage avec CTC	Sortie niveau instance, Intégrable aux réseaux profonds	Pas d'objectif de précocité

Nous avons fait le choix dans cette thèse d'explorer les stratégies de décision apprises au sein des réseaux profonds. Pour le segmenté, nous adapterons SelectiveNet afin de permettre son utilisation dans un contexte séquentiel. Pour le non segmenté, nous modifierons le CTC afin d'atteindre un objectif de précocité.

2.4 Approches récurrentes pour la reconnaissance de gestes en ligne

Auparavant, la reconnaissance de gestes et la détection d'actions étaient réalisées en utilisant des techniques qui nécessitaient l'extraction manuelle de caractéristiques à partir des données. Cependant, ces approches étaient souvent limitées par leur dépendance à des caractéristiques spécifiques et leur capacité à capturer les relations temporelles complexes dans les séquences vidéo. Avec l'émergence des réseaux de neurones profonds, notamment les RNN (Réseaux de Neurones Récurents), ces limitations ont été surmontées. Les RNN ont la capacité de modéliser naturellement les séquences de données, ce qui en a fait un choix privilégié pour la reconnaissance de geste.

Les RNN ont été largement exploités dans le domaine de la reconnaissance de gestes et la tâche de détection d'actions en ligne (OAD). De nombreux travaux ont utilisé des RNN pour la reconnaissance de gestes basée sur les squelettes et ont montré des résultats prometteurs en exploitant les relations temporelles entre les articulations. Bien que les RNN aient été couramment utilisés dans ces tâches, les récentes avancées en matière de Transformers montrent un grand potentiel pour améliorer encore davantage les performances, tout en répondant aux principaux problèmes des réseaux RNN.

2.4.1 Approches à base de réseaux récurrents

2.4.1.1 Fonctionnement des RNN

Les réseaux de neurones récurrents (RNN) sont un type de réseau de neurones conçus pour traiter directement des données séquentielles 1D (un vecteur de caractéristiques à chaque étape). Contrairement aux réseaux de neurones de type MLP ou CNN (dits *feed-forward*), les RNN ont des connexions récurrentes, ce qui signifie que la sortie à un certain instant est réinjectée comme entrée à l'instant suivant.

Un état interne (ou « mémoire ») est conservé et mis à jour à chaque étape. Cette mémoire permet au réseau de conserver des informations sur les étapes précédentes de la séquence, ce qui est essentiel pour comprendre le contexte temporel des données.

Cependant, les RNN traditionnels ont tendance à rencontrer des problèmes lors de l'apprentissage de séquences à long terme en raison du phénomène de "vanishing gradient" [HS97] (le gradient tend à disparaître au fur et à mesure qu'il se propage vers l'arrière dans le réseau), ce qui limite leur capacité à capturer des dépendances à long terme

dans les données. Pour surmonter ce problème, des variantes plus avancées de RNN, telles que les Long Short-Term Memory (LSTM) et les Gated Recurrent Unit (GRU), ont été développées. Ces architectures intègrent des mécanismes de portes qui régulent le flux d'information à travers le réseau, permettant ainsi de capturer des dépendances à long terme de manière plus efficace. L'inférence de ces réseaux doit se faire de manière séquentielle, ce qui rend l'entraînement généralement plus long que pour les réseaux de type *feed-forward*, surtout dans les contextes où il est difficile d'appliquer des stratégies de type *teacher-forcing* [Lam+16].

2.4.1.2 Architectures des réseaux récurrents

Les RNN ont été largement utilisés pour la détection d'actions en ligne en raison de leur capacité à modéliser la dynamique temporelle des séquences et à fournir une mémoire au système [Li+16 ; Mol+16 ; JN17 ; DT18 ; Xu+19a ; Gao+19 ; Liu+19 ; Eun+20 ; MM22].

Concernant les approches qui utilisent en entrée le squelette [Li+16 ; Car+19 ; Liu+19], elles prennent en entrée généralement une représentation 1D où les coordonnées (x, y et z) des articulations sont concaténées en un seul vecteur de taille $J * 3$, où J est le nombre d'articulations. Ce vecteur est donné en entrée du réseau à chaque instant.

Concernant les approches qui utilisent des vidéos RGB(D), une architecture type CRNN est souvent employée [De +16 ; Mol+16 ; MSS16 ; JN17 ; DT18 ; Gao+19 ; Mle+19 ; Xu+19a ; Eun+21 ; KNK21], où de premières couches de type CNN traitent les images afin d'extraire des caractéristiques tout en réduisant les dimensions. Des réseaux CNN pré-entrainés comme VGG [SZ15] ou C3D [Tra+15] peuvent être alors utilisés.

2.4.2 Approches à base de Transformers

2.4.2.1 Fonctionnement des Transformers

Les réseaux de type Transformers ont été introduits pour des tâches de traitement du langage naturel. Ils viennent améliorer les RNN en termes de performance globale et de facilité d'apprentissage.

Le fonctionnement des réseaux Transformers repose sur deux principaux blocs de construction : l'**attention multi-tête** et les couches feedforward. L'attention multi-tête (*Multi-Head Attention*) réside dans des mécanismes d'attention « propre » (*self-attention*). L'entrée est transformée plusieurs fois avant d'appliquer une pondération des caractéristiques, pondération elle-même calculée à partir de l'entrée transformée.

Pour les approches de type séquence-à-séquence, une partie « décodeur » est utilisée. Elle prend en entrée le vecteur de représentation encodée généré par l'encodeur, ainsi que la séquence cible (les mots ou symboles à prédire). Le fait de prendre en entrée la séquence cible permet un entraînement totalement en parallèle, car le réseau considère la séquence cible (partiellement masquée) en entrée comme la prédiction précédente du réseau (*teacher-forcing*), ne nécessitant pas de récurrence pendant l'apprentissage. En revanche, lors de l'inférence en test, une récurrence est nécessaire afin d'utiliser la sortie réelle du réseau en entrée du décodeur.

Les réseaux transformers sont largement utilisés dans le traitement du langage naturel pour des tâches telles que la traduction automatique, la génération de texte et la compréhension de texte. Les transformers sont à la base des LLM (*Large-Langage Models*) récents (GPT, Llama, LaMDA, BLOOM...).

Des approches ont été proposées pour pouvoir traiter des images avec les transformers, comme les *Vision Transformer* [Dos+21] ou encore les *ConvNeXt* [Liu+22 ; Woo+23].

2.4.2.2 Transformers pour la reconnaissance d'action

Depuis 2020, des transformers commencent a être utilisés pour la reconnaissance d'actions segmentées [Cho+20 ; Shi+20 ; PCM21 ; Zha+21 ; KAK22 ; Ahn+23 ; Sun+23], et pour la tâche d'OAD (niveau frame) [Wan+21 ; Xu+21 ; Che+22 ; Guo+22 ; ZK22 ; Cao+23 ; Gue+23].

Comme pour les approches basées RNN, la plupart des approches utilisent d'abord un CNN pré-entrainé comme TSN [Wan+16] ou I3D [CZ17a] pour extraire des caractéristiques et réduire la dimension afin d'obtenir un vecteur de caractéristique par frame.

Wang et al. ont développés LSTR [Xu+21], un Transformer modélisant une mémoire à long terme et à court terme. Dans la continuité de certaines approches récentes se basant sur des RNN ou CNN [Xu+19a ; Xu+19b ; Li+19 ; Wan+19 ; YCL20 ; KNK21 ; FH21 ; Zha+22 ; Wan+23a] qui tentent de prédire le futur pour mieux prédire l'action courante, OadTR [Wan+21] combine les prédictions des futures frames avec les caractéristiques calculées à partir des frames passées afin identifier l'action en cours. GateHub [Che+22] a introduit un mécanisme de filtrage pour éliminer les informations redondantes et les bruits dans les frames passées. Enfin, Guo et al. [Guo+22] exploite une notion d'incertitude afin d'améliorer l'efficacité de l'attention.

Les réseaux Transformers ont démontré leur efficacité dans de nombreuses tâches de traitement du langage naturel et ont également montré des performances prometteuses

dans le domaine de la vision, notamment avec des modèles tels que VIT et ConvNexts. Cependant, leur potentiel pour la détection d'actions en ligne n'a pas encore été pleinement exploité. Une approche prometteuse serait de les entraîner de manière non supervisée en prédisant les poses futures à partir de grandes quantités de données vidéo. En utilisant ces prédictions futures, les Transformers pourraient être en mesure de capturer les relations temporelles à long terme nécessaires pour la détection d'actions en ligne.

2.5 Approches à base de réseaux à convolution spatio-temporelle

2.5.1 Fonctionnement général des réseaux à convolution

2.5.1.1 Opération de convolution

L'opération de convolution consiste à appliquer un filtre (aussi appelé noyau) à travers les données d'entrée. Le filtre est une petite matrice de poids qui glisse sur l'image en effectuant des multiplications élément par élément avec les pixels de l'image. Ces produits sont ensuite sommés pour obtenir une nouvelle valeur qui représente la présence d'une caractéristique particulière dans cette région de l'image.

Convolution multi-canaux Plusieurs filtres sont généralement appliqués sur la même image (générant plusieurs nouvelles images appelées carte de caractéristiques), de même que plusieurs images sont données en entrée via les canaux. Les canaux d'entrée sont utilisés pour représenter différentes informations dans les images, souvent associées à des composantes de couleur. Typiquement, les images RVB (Rouge, Vert, Bleu) sont représentées par trois canaux d'entrée distincts. Par conséquent, un filtre de convolution doit être adapté : si trois canaux sont présents sur une image 2D alors le filtre sera un volume 3D $n \times m \times \mathbf{3}$. Un filtre de dimension 2×2 contiendra alors 12 valeurs distinctes.

Étant donné le filtre positionné à l'emplacement (x, y) de l'image, le résultat de la convolution correspondante est la suivante :

$$\text{Convolution(x,y)} = \sum_{c=1}^{C} \sum_{i=1}^{n} \sum_{j=1}^{m} (\text{Filtre}[i][j][c] \times \text{Données}[x+i][y+j][c]) + b \qquad (2.22)$$

où :

— Convolution représente la valeur résultante de la convolution à un emplacement donné ;
— Filtre[i][j][c] est la valeur du filtre à la position (i, j) du canal c ;
— Données[x+i][y+j][c] est la valeur des données d'entrée à l'emplacement $(x+i, y+j)$ du canal c ;
— n et m correspondent aux dimensions du filtre ;
— C est le nombre de canaux dans les données d'entrée et le filtre ;
— b est le biais, il y a une valeur de biais par filtre.

Taille de la sortie L'application de la convolution réduit généralement la taille spatiale des données. La taille de la sortie dépend de la taille du filtre, du padding (ajout de valeurs nulles autour des bords de l'image) et du stride (décalage entre chaque position de la convolution). Le padding permet de conserver la taille spatiale de l'image, tandis que le stride est un paramètre permettant de réduire davantage la taille de l'image de sortie.

La formule pour calculer la taille de sortie d'une couche de convolution est la suivante :

$$\text{Taille de sortie} = \left(\frac{\text{Taille d'entrée} - \text{Taille du filtre} + \text{Padding Total}}{\text{Stride}} \right) + 1 \tag{2.23}$$

— **Taille de sortie** : correspond à la taille spatiale de la sortie de la couche de convolution ;
— **Taille d'entrée** : est la taille spatiale de l'entrée de la couche de convolution ;
— **Taille du filtre** : représente la taille spatiale du filtre utilisé dans la convolution ;
— **Padding** : est le nombre de valeurs nulles ajoutées à l'image d'entrée, de manière symétrique ou non ;
— **Stride** : correspond au décalage entre chaque position de la convolution, généralement fixé à 1.

Par exemple, avec une image d'entrée de taille 32×32 pixels, un filtre de taille 3x3 pixels, aucun padding et un stride de 1, la sortie serait de taille 30×30 :

$$\text{Taille de sortie} = \left(\frac{32 - 3 + 0}{1} \right) + 1 = 30 \tag{2.24}$$

À noter que cette taille de sortie n'a pas d'impact sur le nombre de paramètres de la couche de convolution suivante.

Le volume 3D résultant de l'opération de convolution sur l'entrée complète est de dimension Taille de sortie$_x$ × Taille de sortie$_y$ × Nombre de filtres.

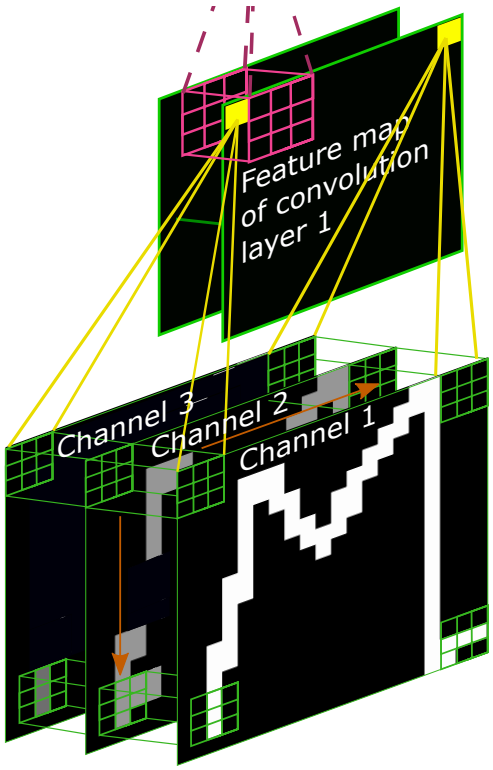


FIGURE 2.15 – Convolution 2D avec 3 canaux. 2 cartes de caractéristiques sont obtenues (2 filtres 3 × 3 utilisés).

2.5.1.2 Opération de *Pooling*

Le pooling joue un rôle très important dans la capacité des réseaux à convolution à généraliser. Il vise à réduire la dimension spatiale des cartes de caractéristiques obtenues à partir des couches de convolution précédentes. Il peut permettre de diminuer le nombre de paramètres (si opération d'aplatissement, cf. 2.5.1.3) et de calculs nécessaires tout en conservant les informations importantes. Le pooling permet également de rendre les cartes de caractéristiques plus invariantes aux translations des caractéristiques détectées. La plupart des fonctions d'agrégation peuvent être utilisées pour faire du pooling.

La fenêtre du pooling circule sur l'image de la même manière que le filtre de convolution, mais généralement avec un stride plus grand, permettant une réduction plus drastique de la taille de l'image. Il s'applique sur chaque carte de caractéristique (résultante de l'application d'un filtre) indépendamment.

Le max pooling et le average pooling sont les deux types de pooling les plus couramment utilisés. Le max pooling sélectionne la valeur maximale dans une région donnée et l'utilise comme représentation de cette région. Cela permet de conserver les caractéristiques les plus saillantes et les plus pertinentes. D'autre part, l'average pooling calcule la moyenne des valeurs dans une région donnée et utilise cette moyenne comme représentation de la région. Cela permet de réduire le bruit et de conserver une information plus globale.

$$\text{Average Pooling(x,y)} = \frac{1}{n \times m} \sum_{i=1}^{n} \sum_{j=1}^{m} \text{Données}[x+i][y+j] \tag{2.25}$$

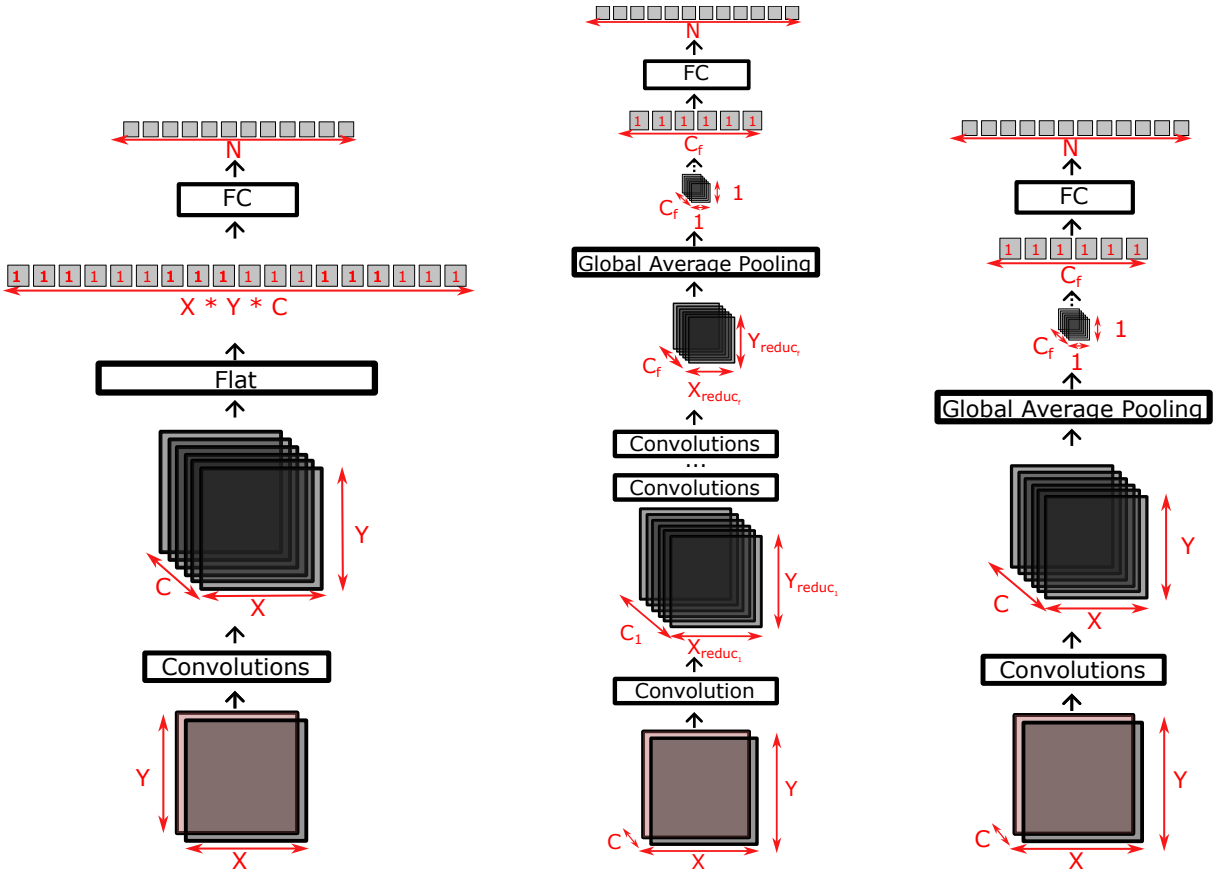
où :

— Average Pooling représente la valeur résultante de l'average pooling à une position donnée.
— Données$[x+i][y+j]$ est la valeur des données d'entrée à l'emplacement $(x+i, y+j)$.
— n et m correspondent aux dimensions de la fenêtre de pooling.

2.5.1.3 Production d'une sortie pour la classification

Dispense des axes : aplatissement ou agrégation Un réseau à convolution est généralement constitué d'un enchaînement de couches de convolution et de pooling. À l'issue de ces opérations, on veut généralement produire une sortie. La sortie peut être de différentes natures suivant la tâche (images, valeur unique pour régression, score par classe pour classification...). Pour produire des scores de classification, l'objectif est de passer d'un volume issu des couches de convolution/pooling $X \times Y \times C$ (où C est le nombre de cartes de caractéristiques), à un vecteur de dimension N, correspondant au nombre de classes. Plusieurs stratégies sont envisageables pour y arriver.

La stratégie la plus simple (figure 2.16a) est de réorganiser le volume 3D en volume 1D, en aplatissant toutes les valeurs, formant un vecteur de taille $X * Y * C$. Une couche complètement connectée (fully-connected, FC) permet ensuite d'adapter ce nombre au nombre de classes. Dans cette stratégie, la taille de l'image d'entrée peut avoir un fort impact sur le nombre de paramètres total, car il jouera sur le nombre de poids sur la couche FC suivant l'aplatissement. Au lieu d'aplatir, il est aussi possible de réduire, au fil des convolutions et des poolings, les dimensions X et Y afin de n'avoir plus qu'un seul pixel spatial à la fin $(1 \times 1 \times C)$ ou que quelques pixels puis d'utiliser un Global Average Pooling (GAP) (cf. figure 2.16b). En utilisant cette approche, il ne sera pas possible d'utiliser les



(a) Aplatissement (b) Réductions et GAP (c) GAP

FIGURE 2.16 – Pour transformer les caractéristiques extraites d'une image en vecteurs compatibles avec la classification, trois approches sont utilisées. La première (a) consiste à aplatir le volume 3D en un vecteur 1D, mais cela peut entraîner un grand nombre de paramètres. La deuxième (b) réduit progressivement les dimensions spatiales en utilisant des convolutions et des poolings, tandis que la troisième (c) applique un pooling global pour obtenir une seule valeur.

branches résiduelles car les dimensions à chaque couche seront différentes. Il est également possible d'utiliser un *GAP* sur la dernière couche sans réduire progressivement la taille (cf. figure 2.16c). Le GAP est un *pooling* qui dispose d'une taille égale aux dimensions spatiales ($X \times Y$), réduisant directement ces dimensions à une unique valeur (devient $1 \times 1 \times N$).

Conservation des axes Il est possible de conserver les dimensions originales afin de produire une sortie différente selon les axes. Par exemple, en restant au niveau volume

$X \times Y \times C$, et appliquant la couche FC de manière identique pour chaque « pixel » de la carte de caractéristique (x,y) $\in X \times Y$ (aucun mélange de valeur entre ces pixels), il est possible d'obtenir une sortie de dimension $X \times Y \times N$. Au lieu d'utiliser une couche FC, il est équivalent d'appliquer une couche de convolution avec N filtres 2D de taille 1×1. Pour une image, l'intérêt est de permettre la classification de chaque pixel. Pour une séquence, cela permettrait de prédire une classe par instant. Avec un nombre plus grand de dimension, il serait possible d'aplatir certains axes et d'en conserver d'autres.

Combiner Aplatissement/Agrégation et conservation des axes, dans un contexte en ligne Il est également possible de combiner ces deux principes afin de conserver certaines dimensions et d'en aplatir ou agréger d'autres, y compris dans un **contexte en ligne**. Il faut en revanche faire attention que les opérations effectuées ne permettent pas d'utiliser des informations futures. Nous avons exploité cette combinaison pour conserver l'axe temporel et agréger les dimensions spatiales, comme nous le verrons en détail en section 3.2.2.

Tous les principes vus dans les sections précédentes pour le cas de convolution 2D sont généralisables pour les convolutions 1D, 3D et au-delà.

2.5.1.4 Bilan sur la mécanique des réseaux à convolution

Nous avons vu que les réseaux à convolution exploitaient des filtres matriciels, et que ces filtres « glissent » sur toute l'image afin de produire une sortie par position.

Comparativement, les CNN se distingue du *Perceptron Multi-couche* (Multi Layer Perceptron, MLP) par son fondement conceptuel. Le CNN repose sur un principe fondamental inspiré des mécanismes de perception humaine [LB95]. En exploitant l'idée du glissement sur l'image, il identifie l'intérêt de capturer des caractéristiques similaires à différents endroits de l'image à travers des groupes de pixels. Ce mécanisme de convolution permet au système de généraliser davantage et offre une efficacité supérieure par rapport aux MLP lors du traitement des images. Contrairement aux MLP, qui attribuent un poids à chaque pixel de l'image, les CNN exploitent efficacement les **relations spatiales** entre les pixels grâce à leurs opérations de convolution, ce qui les rend **particulièrement adaptés à l'analyse des données « visuelles »**.

C'est cette analyse des réseaux à convolution qui nous a orienté dans notre choix d'utiliser une représentation euclidienne des gestes dans notre méthode, mais nous en reparlerons.

2.5.2 Représentations des données d'entrée

Plusieurs travaux ont montré que la manière de présenter les données brutes au réseau, que nous appelons « représentation » des données de squelette, peut avoir un impact décisif sur les performances [Ke+17 ; Yan+20 ; LAM19 ; Cae+19 ; CBS19 ; Liu+20a ; Li+21 ; MND22 ; Qin+22 ; Dua+22], et particulièrement pour les méthodes basées sur les CNN.

Les représentations des données de squelette peuvent être divisées en deux principales catégories : celles basées sur les **coordonnées** des articulations (résumé dans le tableau 2.9) et celles basées sur un **espace euclidien** (tableau 2.10).

2.5.2.1 Représentations matricielles des coordonnées des articulations

La première catégorie consiste à organiser les coordonnées des articulations dans une matrice 2D ou 3D. Un résumé des méthodes concernées est fourni dans le tableau 2.9. Hou et al. [Hou+18] concatènent toutes les coordonnées de toutes les articulations dans

Table 2.9 – Résumé des représentations matricielles utilisées dans les approches basées sur le squelette avec les CNN. J : nombre d'articulations considérées, *time* : nombre de frames, c : nombre de canaux de sortie, dépend des approches et des variantes. μ et τ : dépendent des approches, τ dépend toujours de *time*. X, Y et Z désignent ici la taille d'une image. *views* représente le nombre de points de vue différents intégrés dans les représentations.

Dim.	**Description courte**	**Dimension de sortie**	**Approches**
1D	Temps, concaténation d'articulations	$time \times (J * 3)$	[Hou+18], [Mar+21] (troisième branche)
1D	Temps, Composante	$3 * (time \times J)$	[Li+21] (première branche)
1D	Temps, Articulation-Composante	$3 * J * (time \times 1)$	[Dev+18]
1D	Temps, Distances relatives	$time \times (\frac{(J*J)}{2} + 2 * J * 3)$	[Yan+20]
1D	Articulations, Temps	$time * (J \times 3)$	[Liu+20a]
2D	Temps and Articulations	$time \times J \times 3$	[DFW15 ; Ke+17 ; Lar+17 ; Zha+19 ; Núñ+18 ; Ke+20 ; Cae+19 ; CBS19 ; Cao+19a ; RBA21 ; MND22 ; Qin+22]
2D	Temps/Articulations Arrangement en grille	$(\sqrt{J} * \mu) \times (\sqrt{J} * \tau) \times 3$	[LAM19], [Li+21] (deuxième branche)

un vecteur pour chaque frame, comme illustré dans la figure 2.17a, même si cette représentation est très compacte et simple, elle obtient de bons résultats lorsqu'elle est utilisée avec un CNN 1D. Une disposition différente utilisant la même quantité de données que Hou et al. a été introduite par Du et al. [DFW15]. Elle consiste à placer les coordonnées dans une matrice 3D : temps × articulations × position de l'articulation

(valeurs x, y, z), un CNN 2D peut alors être appliqué à cette structure, comme illustré dans la figure 2.17b. Cette représentation et ses variantes ont été largement utilisées dans la littérature [Ke+17 ; Lar+17 ; Núñ+18 ; Cao+19a ; CBS19 ; Cae+19 ; Ke+20 ; Zha+19 ; RBA21 ; MND22 ; Qin+22]. Un aspect crucial à considérer est que l'ordre des articulations revêt une importance dans cette représentation, car le noyau de la première couche de convolution perçoit exclusivement les articulations voisines dans la matrice. De plus, il y a une limitation dans sa capacité à capturer les relations spatiales entre les articulations qui sont proches dans l'espace euclidien à un moment donné, mais pas consécutives dans l'ordre de la matrice.

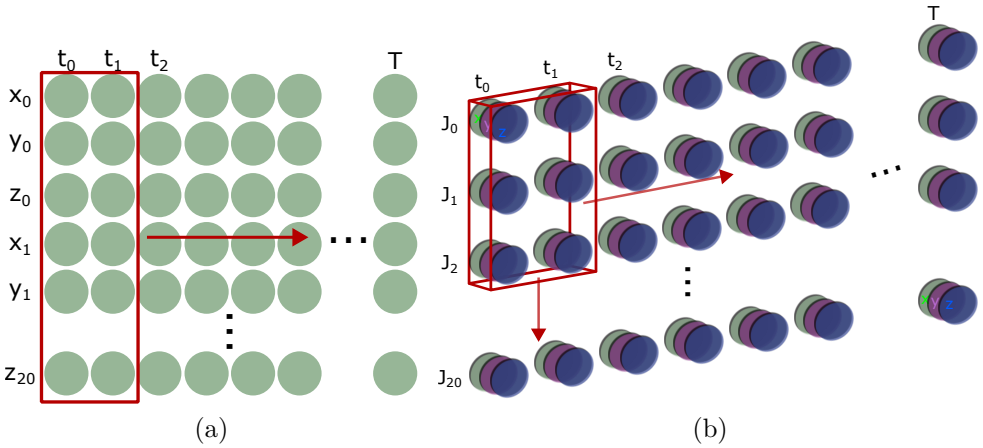


FIGURE 2.17 – a) Illustration de l'application d'une convolution 1D (ici filtre de dimension 2, 60 canaux) sur la représentation de Hou et al.[Hou+18]. t_i représente les frames, T est le nombre de frame. x_i, y_i, z_i sont les coordonnées de l'articulation i (en canaux). b) Illustration de l'application d'une convolution 2D (ici filtre de taille 2×3, 3 canaux) sur la représentation 2D très utilisée dans la littérature. t_i représente les frames, T est le nombre de frames. J_i représente l'articulation i, avec ses coordonnées x, y, z sur la troisième dimension (en canaux).

2.5.2.2 Représentations euclidiennes

Certains travaux ont tenté de représenter les gestes dans un espace euclidien en 2D, 3D ou 4D [TLL18 ; Wan+18a ; Shi+20 ; Dua+22]. Un résumé est donné dans le tableau 2.10. Par exemple, Duan et al. [Dua+22] construisent des images 2D du squelette. Plusieurs images sont générées par frame pour localiser l'emplacement de chaque articulation et os. Il est à noter que ces images sont très clairsemées, mais sont construites sous forme de cartes de chaleur (ou cartes thermiques - *heatmap*). L'intensité des pixels représente la

confiance de la présence des articulations ou os dans cette zone (car la pose est estimée à partir du RGB dans l'approche originale). Ces images au fil du temps sont utilisées comme entrée directe pour un CNN 3D dans les différents canaux. Le CNN extrait ensuite des caractéristiques en utilisant des filtres se déplaçant dans trois dimensions : temps, dimension X et dimension Y, comme l'illustre la figure 2.18. Cette approche améliore les résultats précédents de l'état de l'art sur plusieurs benchmarks.

TABLE 2.10 – Résumé des représentations euclidiennes utilisées dans les approches basées sur le squelette avec les CNN. J : nombre d'articulations considérées, *time* : nombre de frames, c : nombre de canaux de sortie, dépend des approches et des variantes. μ et τ : dépendent des approches, τ dépend toujours de *time*. X, Y et Z désignent ici la taille d'une image. *views* représente le nombre de points de vue différents intégrés dans les représentations.

Dim.	**Description courte**	**Dimension de sortie**	**Approches**
2D/3D	Spatial, temps cumulé	$views * (X \times Y[\times Z] \times c)$	[Wan+18a] (2D), [TLL18] (3D)
3D/4D	Spatio-temporel	$views * (time \times X \times Y[\times Z] \times c)$	[Shi+20] (4D, deuxième branche), [Dua+22] (3D)

2.5.3 Architecture des réseaux à convolution

2.5.3.1 CNN 1D : Temporal Convolutional Network

WaveNet [Oor+16] est une architecture de réseau à convolution introduite en 2016. L'une des caractéristiques clés de WaveNet est l'utilisation de convolutions dilatées, qui permettent d'élargir le champ réceptif du réseau sans augmenter le nombre de paramètres. Contrairement aux réseaux de neurones récurrents, WaveNet utilise uniquement ces convolutions dilatées pour capturer les dépendances à long terme dans les données séquentielles. Cela lui permet d'être plus efficace en termes de calcul et de parallélisation. Une autre caractéristique clé de ce réseau est qu'il puisse être utilisé en mode séquence-à-séquence, à chaque frame d'entrée, il produit une sortie. De plus, ses convolutions sont causales, pour générer la sortie d'un instant donné il ne se sert que des frames passées. La causalité s'obtient pendant l'apprentissage grâce à un padding au début de la séquence. La taille du padding est calculée de sorte à combler la réduction de la dimension provoquée par l'opération de convolution.

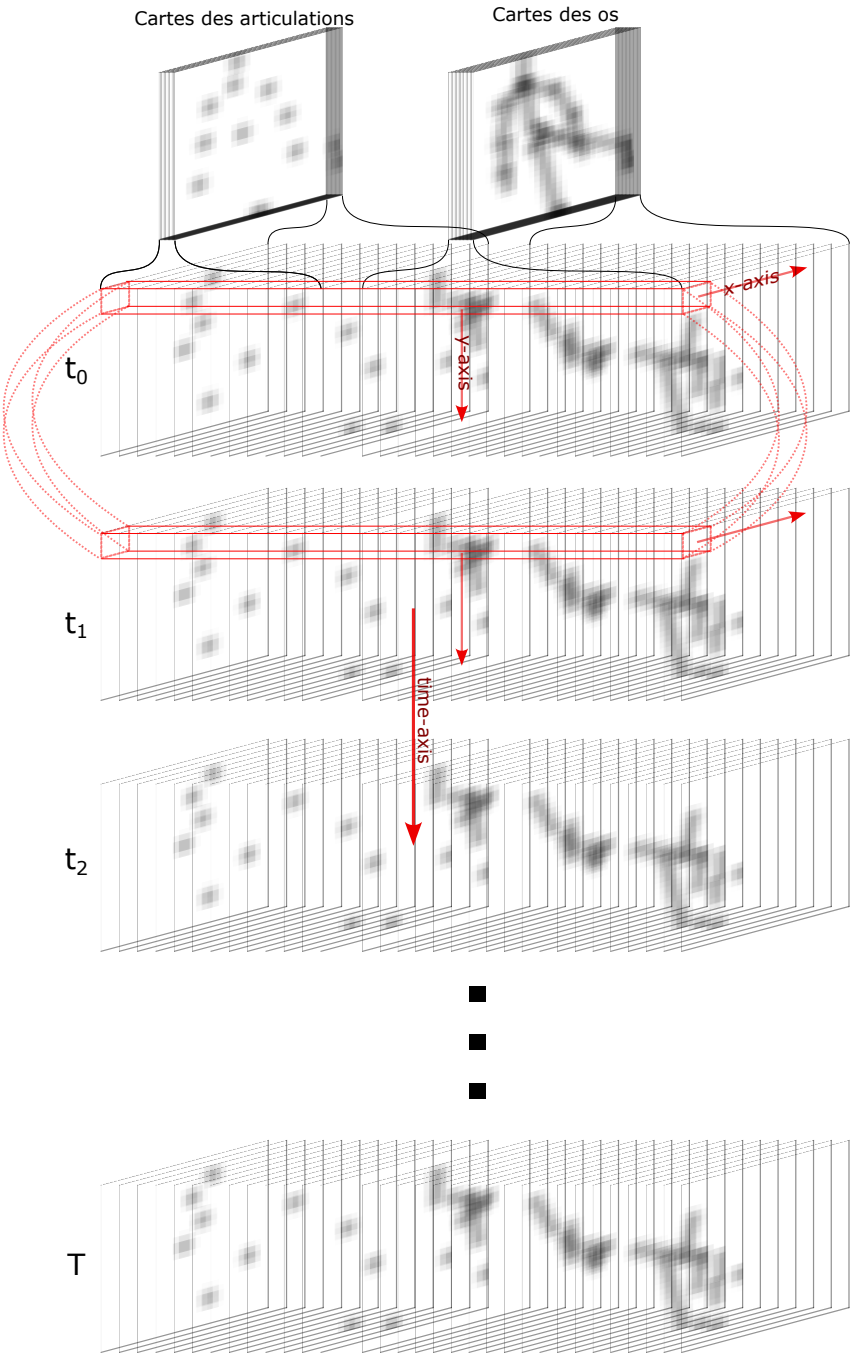
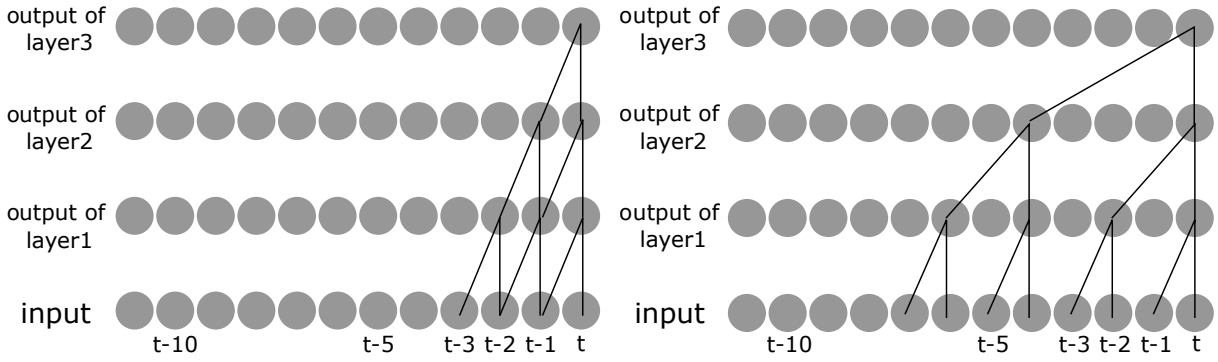


FIGURE 2.18 – Illustration de l'application d'une convolution 3D sur la représentation de Duan et al. [Dua+22]. Ici la taille du filtre est de 2 sur l'axe temporel.

Une illustration d'architecture avec des convolutions causales classiques et dilatées est visible en figure 2.19.



(a) Convolutions causales classiques (b) Convolutions causales et dilatées

FIGURE 2.19 – Illustration de la différence en termes de champs réceptifs suivant le type de convolution utilisée. L'utilisation de convolutions dilatées permet d'agrandir grandement le champ réceptif et donc d'intégrer davantage de contexte temporel.

Cette architecture a inspiré plusieurs approches dédiées à la reconnaissance de gestes [Lea+17 ; Liu+20a ; Dai+21].

En particulier Liu et al. [Liu+20a] ont développé une approche dédiée à la tâche d'OAD (par frame). Une première partie du réseau extrait une représentation 1D à partir du squelette. Cette représentation est ensuite fournie en entrée du réseau séquence-à-séquence de type wavenet, avec des convolutions causales et dilatées. Une particularité de leur approche est que le réseau prédit, en plus de la classe courante, le nombre de frames depuis lequel l'action a commencé. Cette prédiction est ensuite utilisée afin de sélectionner une couche du réseau dont le champ réceptif est le mieux adapté, dans le but de voir tout le geste, mais sans trop d'information superflue. En effet, en observant la figure 2.19b, on peut voir que sélectionner une couche en particulier permet d'avoir un champ réceptif précis : jusqu'à t-3 pour la deuxième couche, jusqu'à t-7 pour la troisième. Ainsi, si le geste a commencé à t-3, la deuxième couche couvre l'ensemble de ce début de geste en minimisant la quantité d'information antérieure au geste, jugée superflue pour la reconnaissance. Les caractéristiques sélectionnées sont alors passées dans des couches FC afin de prédire la classe. Il est intéressant de voir que cette couche réussit à interpréter correctement les caractéristiques malgré le fait qu'elles peuvent venir de couches différentes. On suppose alors une sorte d'uniformité dans les caractéristiques extraites à chaque couche.

Zhao et al. [Zha+22] a également utilisé un réseau de convolution temporelle 1D. Le vecteur de caractéristiques 1D en entrée est extrait à partir d'un réseau CNN 3D à deux

flux. Ils ont utilisé une stratégie de « distillation de connaissances » (*Knowledge Distillation*, KD) pour entraîner un modèle « élève », entraîné à l'aide des frames passées, à partir d'un modèle « enseignant » capable d'utiliser des informations futures. L'élève essaie d'adapter sa représentation à partir de celle de l'enseignant en utilisant moins d'informations. C'est une manière implicite de prédire les informations futures à partir des frames passées. Une procédure d'apprentissage particulière est utilisée pour entraîner progressivement l'élève en utilisant des enseignants intermédiaires. Ces enseignants intermédiaires utilisent progressivement plus d'informations provenant du futur.

2.5.3.2 CNN 2D pour représentation à base de coordonnées

Les CNN 2D ont été beaucoup utilisées pour la reconnaissance dans un contexte segmenté, notamment pour exploiter les représentations à base de coordonnées mentionnées précédemment. Certaines approches [Lar+17 ; RBA21 ; MND22] convertissent les valeurs dans un format RGB afin d'exploiter des modèles pré-entrainés dans ce format, ce qui permet d'améliorer les résultats par rapport à un entraînement *from-scratch*.

Zhang et al. [Zha+19] font apprendre au réseau à réorienter le squelette de sorte à toujours le voir sous le même angle, et de faciliter la tâche de reconnaissance.

Dans l'approche de Cao et al [Cao+19a], une permutation est apprise afin d'être moins dépendant de la disposition initiale des articulations dans la représentation.

Les stratégies employées par les différentes approches afin de répondre au problème de l'ordre des articulations dans la matrice de la représentation n'étant pas totalement satisfaisantes et commençant à s'épuiser, une nouvelle architecture de réseau a vu le jour à base de graphe, afin de prendre en compte au maximum les relations entre les articulations.

2.5.3.3 GCN, l'évolution pour les représentations à base de coordonnées

Une autre façon de résoudre le problème de corrélation spatiale dans la représentation de Du et al. consiste à concevoir un mécanisme spécial pour appliquer des convolutions sur le graphe du squelette, en exploitant sa structure sémantique. Le premier Réseau de Neurones Convolutifs sur Graphe Spatio-Temporel (ST-GCN) a été introduit par Yan et al. [YXL18] en utilisant cette idée. Les approches basées sur les GCN ont suscité un intérêt considérable dans le domaine, avec de nombreuses publications explorant son utilisation ces dernières années [Li+19 ; Shi+19 ; Che+20 ; Liu+20b ; Che+21 ; Zho+22].

2.5.3.4 CNN 3D pour l'extraction de caractéristiques spatio-temporelles sur des vidéos

Du côté des vidéos RGB pour la reconnaissance de gestes segmentés, le réseau C3D (*Convolutional 3D* [Tra+15]) fut parmi les premiers à exploiter des CNN 3D afin d'extraire des caractéristiques pour la reconnaissance d'actions. À l'origine, C3D est entraîné sur des clips de 16 frames, et donne une prédiction pour caractériser ces 16 frames. Pour classifier une vidéo entière, plusieurs clips sont extraits aléatoirement de la vidéo, et une moyenne des scores est faite afin de classifier la vidéo. En 2017, I3D [CZ17a] fut présenté afin d'améliorer les performances de C3D, notamment en utilisant le « flux optique » (*optical-flow*) qui indique comment chaque pixel se déplace d'une image à l'autre. De plus, I3D bénéficie de poids pré-entrainé en exploitant les poids de réseaux pré-appris sur des bases d'images statiques RGB. Depuis, d'autres évolutions de ce type de réseaux sont apparues, comme les réseau SlowFast [Fei+19] ou TSM [LGH19], qui se sont fait récemment dépassés en termes de performances par des architectures de type Transformers [Arn+21].

En ce qui concerne la reconnaissance de gestes non segmentés en ligne, les CNN ont été peu exploités, au profit de réseaux récurrents qui sont naturellement plus propices au mode séquence-à-séquence. Cependant, comme mentionné dans la section dédiée au RNN, un extracteur de caractéristique de type C3D est souvent exploité avant les couches RNN afin d'extraire des caractéristiques visuelles et de réduire la dimensionnalité (pour obtenir un vecteur par frame). Cependant, on trouve l'approche de Zhao et al. [Zha+22] qui utilise des couches de convolution 1D à la place de couches de RNN.

2.5.3.5 Technique d'élargissement du contexte temporel passé pour les CNN

Poussées par la limitation de la visibilité du réseau original C3D, plusieurs techniques ont émergé afin d'intégrer plus de contexte temporel dans les réseaux de type **CNN 3D**.

3D Temporal Convolutional Networks Dans un contexte segmenté hors ligne, Varol et al. [VLS18] ont montré que l'augmentation du nombre d'images d'entrée pour obtenir une visibilité à plus long terme a un impact significativement positif sur le score final. Du fait de la diminution drastique de la dimension temporelle via les couches de pooling, l'impact de cette dimension sur le nombre de paramètres total du réseau est très limité. Liu et al. [Liu+18b] ont proposé une stratégie pour un 3D-CNN pour prendre en compte l'ensemble de la vidéo. Tout d'abord, la vidéo est divisée en S parties. Quelques images sont échantillonnées de manière aléatoire à partir de chaque partie, formant un clip. Chaque clip

est passé indépendamment dans le réseau 3D CNN les cartes de caractéristiques obtenues sont agrégées. Ce vecteur est ensuite passé dans des couches denses pour classer l'action. La fonction de coût est calculée après la dernière couche en considérant l'ensemble de la vidéo.

Ces approches ont été appliquées sur des gestes segmentés. Si l'on souhaitait les appliquer sur un flot de données de gestes, la première approche pourrait être envisagée avec un système de fenêtre glissante. Il faudrait cependant revoir la complexité du réseau afin de permettre une exécution en temps réel. La deuxième est moins adaptée à notre contexte puisque l'ensemble de la vidéo est considérée, au détriment d'information plus récente.

Non-local neural network Les approches à base de réseau non local [Wan+18b ; Cao+19b ; Wan+22] visent à capturer les interactions à longue distance dans les données. Contrairement aux réseaux convolutifs traditionnels qui se concentrent sur les interactions locales, les réseaux non locaux permettent de modéliser les relations entre des entités éloignées dans une séquence ou une image. Ils utilisent des modules spéciaux pour calculer les similarités entre différentes parties de l'entrée, ce qui leur permet de saisir les interactions globales. Les réseaux non locaux peuvent être plus complexes en termes de calcul et de mémoire. Ces techniques utilisent souvent des mécanismes d'attentions, les approches à base de transformer pourraient être assimilés à cette catégorie.

Convolutions Dilatées Une autre manière d'intégrer du contexte dans un réseau CNN 3D est d'utiliser des convolutions dilatées. Quelques convolutions dilatées sur la dimension temporelle, couplées avec un module non-local, ont été utilisées dans le cadre de la reconnaissance d'actions segmentées dans les travaux de Xu et al. [Xu+20]. Cependant, l'utilisation des convolutions dilatées pour étendre le champ réceptif de réseaux CNN 3D spatio-temporel a été très peu explorée.

2.6 Conclusion de l'état de l'art

Dans ce panorama qui concerne la reconnaissance de gestes en ligne, segmentés et non segmentés, nous avons exploré les multiples facettes de cette problématique complexe. Nous avons catégorisé et analysé les différentes tâches, et répertorié les métriques adoptées ainsi que les mécanismes de prise de décision couramment utilisés. De plus, nous avons

examiné les systèmes d'apprentissage reposant sur les réseaux profonds, les approches récurrentes telles que les RNN et les transformers, et les architectures basées sur les réseaux à convolution.

Au fil de notre exploration, nous avons clarifié nos objectifs et les nuances des différentes tâches de reconnaissance de gestes en ligne, en les regroupant en deux catégories distinctes : tâche au niveau frame et tâche au niveau instance. Nos objectifs de détection dans un contexte d'interaction nécessitent une interprétation au **niveau instance**. Ainsi, nous nous sommes donc rattachés à deux tâches existantes : la **reconnaissance précoce de gestes segmentés** (*Early Action Detection*) et l'**OAD** (*Online Action Detection*).

Notre analyse des métriques utilisées pour les tâches en ligne a révélé leurs avantages et leurs limitations, soulignant les défis inhérents à la mesure de la performance dans ce contexte complexe. Ainsi, pour le contexte segmenté, nous avons identifié que les métriques **TAR, FAR, RR**, couplées au **NTtoD** pour mesurer la précocité, étaient adaptées à la tâche. Pour le contexte non segmenté, les problèmes soulevés (section 2.2.4) pour les métriques candidates nous poussent à conclure qu'il n'existe pas de métrique vraiment adaptée et pertinente. Ainsi, nous proposerons une nouvelle métrique "***Bounded Online Detection***" (BOD) qui constitue une contribution pour répondre à un contexte applicatif (détaillée en section 4.4.4.2).

Nous avons également détaillé des mécanismes de prise de décision existants, en les expliquant via un modèle en trois étapes : l'expression via un modèle séquence-à-séquence, la décision avec une stratégie de décision, et le décodage pour obtenir une sortie au niveau instance. Nous espérons apporter par ce biais une meilleure compréhension des stratégies existantes, et une meilleure structuration des approches à venir. Nous avons exploré plusieurs stratégies existantes pour nos deux tâches. En particulier, les approches permettant d'apprendre comment effectuer la décision au sein d'un réseau de neurones nous ont semblé particulièrement intéressantes. Pour le contexte segmenté, **SelectiveNet** permet d'avoir une sortie de type rejet, en revanche une adaptation est nécessaire afin de permettre **un rejet sur la dimension temporelle** pour permettre une détection en ligne. Pour le contexte non segmenté, la fonction de coût **"Connectionist Temporal Classification" (CTC)** a été construite initialement pour produire des sorties au niveau instance sur des séquences. Partir de cette fonction nous semble donc une bonne idée. En revanche, elle n'est pas adaptée pour localiser précisément un geste et encore moins pour permettre une détection précoce. Nous allons donc **modifier cette fonction** pour atteindre nos objectifs de précocité.

Nous avons exploré différentes architectures de réseaux de neurones : réseaux récurrents, Transformers, CNN et GCN. Bien que les *Transformers* commencent à être compétitifs sur certaines tâches de reconnaissance de gestes (en particulier reconnaissance hors ligne segmenté et OAD par frame), leur montée en puissance s'est faite trop récemment pour les adapter pour nos tâches dans le cadre de cette thèse. En effet, pour pouvoir l'exploiter sur des bases de données qui possèdent souvent peu d'exemples annotés, il faudrait par exemple pré-entrainer le réseau de manière auto-supervisée sur un grand nombre de données non annotées. Les réseaux CNN sont aujourd'hui encore compétitifs sur certaines tâches, et il nous semble encore pertinent de continuer à les étudier. Nous avons identifié des architectures intéressantes telles que **Wavenet et C3D**, dont nous nous inspirerons dans le développement de notre approche. Nous avons détaillé les différentes représentations couramment utilisées, nous avons vu que les **représentations euclidiennes** dans le contexte des CNN étaient particulièrement prometteuses.

Chapitre 3

Reconnaissance précoce de gestes segmentés

3.1 Introduction

Pour être réactives, certaines applications ont besoin de connaître le plus tôt possible l'intention de l'utilisateur. Dans un environnement tactile où l'on peut zoomer, faire défiler avec une manipulation directe, nous devrions également être en mesure, dans le même contexte interactif, d'effectuer des actions plus complexes associées à des gestes tels que des symboles (commande abstraite), comme illustrés dans la figure 3.1. La coexistence de la manipulation directe et de la commande abstraite (ou indirecte) n'est possible dans un contexte interactif que si nous sommes capables de prédire très tôt l'intention de l'utilisateur, avant que le geste ne soit effectué. Très peu de travaux ont abordé ce problème de coexistence, mais on peut citer ceux de Petit & Maldivi [PM13] et de Kurtenbach & Buxton [KB91].

La plupart des travaux existants se concentrent sur la reconnaissance des gestes une fois qu'ils sont terminés, seuls quelques-uns couvrent le problème de la reconnaissance précoce qui est un nouveau défi complexe pour la communauté de la reconnaissance des gestes manuscrits en 2D, mais aussi pour la communauté de la reconnaissance des gestes 3D.

Dans la plupart des cas, il est possible de distinguer le geste avant qu'il ne soit terminé. Nous adressons dans ce chapitre la tâche de reconnaissance précoce dans un contexte segmenté (*early gesture recognition*), telle que définie en section 2.1.2.1. Pour éviter les erreurs, le système doit être capable de reporter la décision si davantage d'informations sont nécessaires, ce qui est lié à la confiance. La reconnaissance précoce des gestes ouvre un large champ de nouvelles applications avec la coexistence de commandes directes et abstraites dans le même contexte.

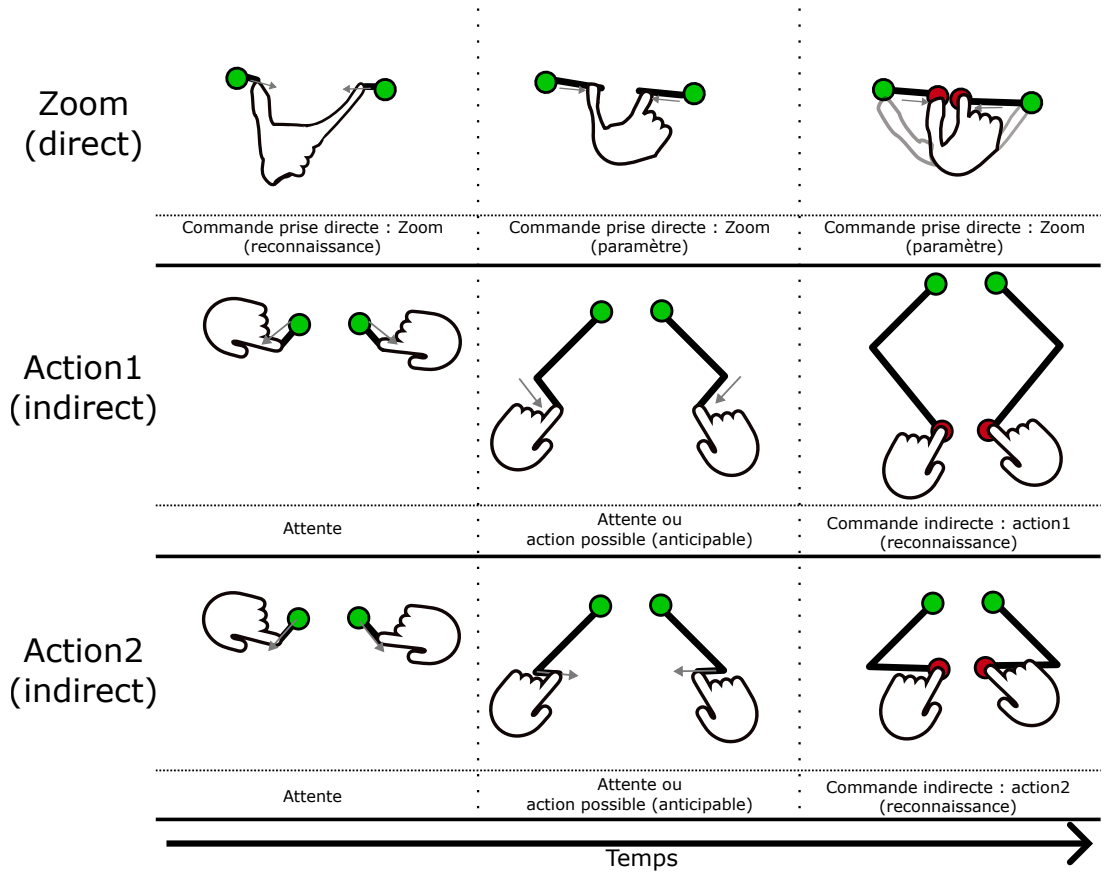


FIGURE 3.1 – Illustration du problème de la coexistence du geste en prises direct et des commandes abstraites. Afin de pouvoir effectuer le feedback (action de l'application) en temps voulu, il est important de reconnaître le plus tôt possible les gestes de manipulation qui sont en prise directe. Pour les gestes de commandes indirectes, il n'y a pas besoin d'un feedback instantané. Cependant, il est possible de reconnaître ces gestes plus tôt (par anticipation) et donc d'effectuer l'action plus tôt pour permettre une interaction plus fluide.

Pour relever ces défis, nous proposons un nouveau réseau que nous avons nommé « OLT-C3D » pour « Online Long-Term Convolutional 3D », couplé à un système intégré d'option de rejet temporel. Inspirés par les récents CNN spatio-temporels dans le domaine de la vision par ordinateur [CZ17b ; Tra+15], nous proposons une nouvelle architecture basée sur un CNN 3D. Le signal d'entrée du geste est traduit en une représentation euclidienne via une séquence d'images dans le temps de la trace cumulative. Ensuite, les images sont passées dans le CNN 3D qui donne une prédiction à chaque nouvelle image. Nous fournissons au réseau OLT-C3D une capacité de décision grâce au système de rejet temporel : la prédiction peut être soit acceptée pour confirmer la prédiction, soit

rejetée si le réseau a besoin d'attendre plus d'informations pour prendre une décision. Les principales contributions qui vont être détaillées dans les sections suivantes sont résumées ci-dessous :

— Nous avons conçu une **représentation euclidienne** originale pour traduire le signal d'entrée en ligne dans un *contexte libre* en une séquence d'images. Représenter le geste dans un espace euclidien permet d'exploiter pleinement les propriétés d'un réseau CNN.
— Nous proposons le réseau **OLT-C3D**, **un réseau de neurones à convolution 3D** conçu pour traiter en ligne des images 2D au fil du temps.
— Nous avons ajouté un système d'**option de rejet temporel** au réseau OLT-C3D pour reporter la décision dans le temps si davantage d'informations sont nécessaires.
— Le réseau est entraînable de bout en bout et ne nécessite pas d'étalonnage a posteriori pour le système de rejet.
— Notre méthode permet d'obtenir des performances supérieures pour la tâche de reconnaissance précoce en ce qui concerne la précision et la précocité. Des expériences ont été menées sur deux ensembles de données complémentaires et librement accessibles : ILGDB [Ren+12] (gestes mono-touch) et MTGSetB [Che+15] (gestes multi-touch).

3.2 Reconnaissance précoce de gestes 2D segmentés avec OLT-C3D

Tout d'abord, inspirés par les méthodes basées sur la trajectoire [BAM17 ; Bou+18b ; UA08], nous proposons une **représentation spatio-temporelle** représentant l'accomplissement du geste dans le temps. Le signal en ligne du geste est traduit en une séquence d'images contenant la trajectoire, chaque nouvelle image de la séquence est incrémentée par un nouveau morceau de la trajectoire. Ensuite, les images sont passées dans notre **réseau OLT-C3D**, ce réseau original est principalement inspiré par les récents CNN spatio-temporels [CZ17b ; Tra+15] qui ont prouvé leurs capacités à apprendre des caractéristiques spatio-temporelles. OLT-C3D fournit une prédiction à chaque nouvelle image. Enfin, le **système d'option de rejet temporel** est capable de reporter la décision en rejetant les prédictions.

3.2.1 Stratégie de représentation des gestes spatio-temporels

Dans ce travail, nous présentons une représentation euclidienne du geste. Le signal d'entrée est une trajectoire en ligne, à chaque instant nous connaissons la position des doigts/du stylo sur l'appareil. Nous traduisons ce signal d'entrée en une séquence d'images, chaque nouvelle image contenant les nouvelles positions des doigts avec ses trajectoires précédentes. Cette représentation est l'entrée du CNN spatio-temporel. À chaque instant, nous alimentons le CNN avec les nouvelles informations reçues afin que l'entrée soit toujours à jour. Au début, le réseau ne voit qu'un petit morceau du geste, à la fin, il voit la trace complète du geste avec tout l'historique de sa réalisation. L'historique est très important pour voir dans quel ordre le geste est effectué : deux gestes peuvent avoir la même forme finale, mais ne sont pas effectués dans le même ordre, comme l'illustre la figure 3.2.

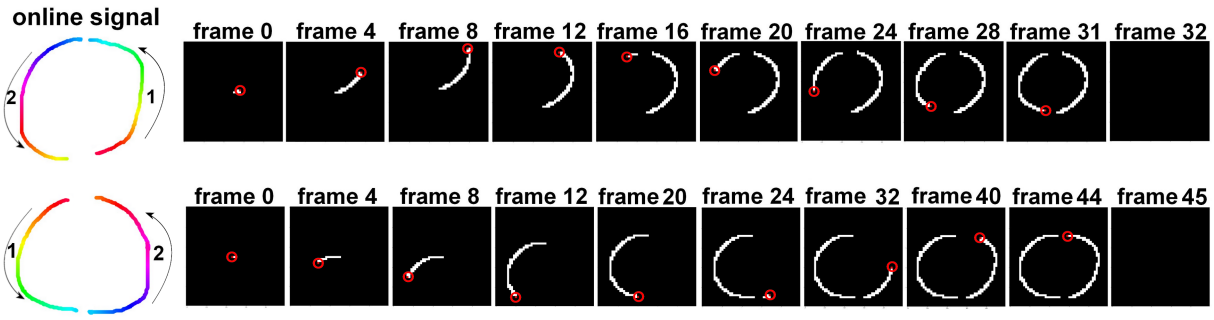


FIGURE 3.2 – Si l'on considère la forme finale, deux classes de gestes peuvent être impliquées en ce qui concerne l'ordre du trait. Pour le second geste, l'ordre des traits est inversé par rapport au premier geste. Notre représentation prend en compte l'ordre des traits grâce à l'historique. La dernière position du doigt dans le segment est représentée par un pixel rouge (entouré d'un cercle rouge dans les images pour des raisons de visibilité) dans notre représentation. Si le geste atteint le bord de l'image, toute la trace du geste est déplacé dans la direction opposée dans les nouvelles images (ceci est visible dans les deux gestes entre les images 8 et 12). La fin du geste est symbolisée par une image noire.

Une stratégie naïve consisterait à alimenter le réseau neuronal avec une nouvelle image à chaque fois que nous disposons de nouvelles informations provenant de l'appareil utilisé (à chaque frame), la plupart des travaux sur la reconnaissance des gestes en 3D utilisent cette stratégie. Néanmoins, il peut n'y avoir qu'une très petite quantité de nouvelles informations si le geste est effectué lentement, ou même aucune nouvelle information si le geste est en pause. En traduisant le signal d'entrée en une nouvelle image à chaque instant, nous

obtiendrions un grand nombre d'images avec une très petite quantité de nouvelles informations effectives entre les images. Si nous choisissons de réduire le taux d'échantillonnage pour réduire le nombre d'images, un geste effectué lentement et le même geste effectué rapidement ne produiraient pas la même quantité d'images au fil du temps. Ce qui pose des problèmes de **robustesse de représentation** et de **capacité d'expression du système**. Pour le geste rapide, nous ne pourrions pas reconnaître le geste aussi rapidement (en termes de quantité d'informations) que nous pourrions le reconnaître si le système avait été interrogé plus tôt. Pour résoudre ces problèmes, nous avons utilisé, comme dans les études précédentes, la quantité d'information (déplacement) au lieu du temps pour quantifier le déplacement effectif et générer une séquence d'images **indépendante de la vitesse**. Chaque nouvelle image est incrémentée du même déplacement θ. Un θ plus petit conduit à un plus grand nombre d'images pour obtenir le geste complet. Cette stratégie présente trois avantages principaux : **moins d'images dans la séquence**, une **différence plus significative entre chaque image** et une **représentation invariante de la vitesse d'exécution**.

Le réseau possède tout l'historique de la trace, mais il ne peut pas faire la différence entre un simple toucher puis un doigt levé et un toucher constant. Comme cette différence peut être discriminante, notamment pour certains gestes multi-touch, nous avons besoin d'une stratégie pour la faire apparaître. À cette fin, nous ajoutons un deuxième canal à nos images, contenant une valeur « 1 » si un doigt est actif dans cette coordonnée à la fin du déplacement actuel, une valeur nulle dans le cas contraire. On obtient ainsi deux images : l'une contenant la trajectoire, et l'autre, très peu dense, contenant les coordonnées où un doigt est actif. Ces deux images seront utilisées comme canaux par le CNN. Ce deuxième canal n'a été utilisé que pour l'ensemble de données multi-touch MTGSetB [Che+15]. Pour les ensembles de données mono-touch tels que ILGDB [Ren+12], il ne présente théoriquement aucun avantage. La présence d'une information dans ce deuxième canal est illustrée dans la figure 3.2 par le cercle rouge qui entoure un pixel.

Une autre difficulté est le *contexte libre* : nous ne savons pas à l'avance quelle sera la taille du geste complet, nous ne pouvons donc pas garantir que le geste entrera correctement dans l'image par rapport à la résolution dont nous disposons au début du tracé. Une solution consiste à redimensionner le geste à chaque nouvelle image pour l'adapter à la taille du tracé, mais nous pensons que cela briserait la spatio-temporalité de l'information qui sera utilisée par le CNN, avec des difficultés à percevoir quel nouveau morceau vient d'être ajouté entre deux images. Pour conserver la même échelle du début à la fin, nous

avons choisi de « suivre » le mouvement en déplaçant tout le geste dans la direction opposée au mouvement lorsqu'il atteint le bord de l'image. Nous perdons potentiellement un morceau du geste à chaque fois que nous le décalons, mais le réseau n'en a plus besoin car cette partie est encore dans son historique. De plus, ces petits décalages devraient être absorbés grâce aux propriétés des couches de pooling qui permettent une certaine robustesse vis-à-vis des translations. Cette stratégie est visible dans la figure 3.2, particulièrement entre les images 8 et 12.

Enfin, dans une base de données où certains gestes sont des sous-parties d'autres gestes, le réseau a besoin de savoir quand le geste est terminé. Pour ce faire, nous ajoutons une image noire à la fin des gestes. Dans une application réelle reposant sur une approche de reconnaissance de gestes segmentés, une stratégie doit être établie pour déterminer quand un geste est terminé, il peut s'agir d'un lever de crayon, d'une confirmation explicite, d'un temps sans action... Si aucune stratégie n'est possible de manière triviale, alors il faudra utiliser une approche construite pour un contexte non segmenté, comme nous le ferons dans le chapitre suivant. La représentation finale est illustrée dans la figure 3.2.

On remarque que cette stratégie de représentation fonctionne dans le contexte segmenté, mais qu'elle ne serait pas applicable dans un contexte non segmenté. En effet, si l'on garde la trace complète des gestes effectués alors l'image finira complètement marquée au bout de quelques gestes, conservant les traces dans anciens gestes rendant toute reconnaissance difficile.

3.2.2 CNN 3D spatio-temporel en ligne avec système de rejet temporel

Récemment, les CNN ont prouvé leur capacité à apprendre à partir de séries temporelles [Liu+20a ; Oor+16] et de séquences d'images [CZ17b ; Tra+15]. Inspirés par ces approches, nous proposons OLT-C3D (Online Long-Term Convolutional 3D), un CNN 3D spatio-temporel capable de traiter un flux de données en continu et de donner une réponse en temps réel.

Cette architecture est principalement inspirée par C3D [Tra+15] et WaveNet [Oor+16]. Le réseau C3D (Convolutional 3D) a été spécifiquement développé pour capturer des caractéristiques spatio-temporelles à partir de données vidéo RVB. Il a démontré une efficacité remarquable en tant qu'alternative aux réseaux récurrents pour accomplir des tâches impliquant des informations spatio-temporelles. Néanmoins, le principal inconvénient qui

est souvent reproché au réseau C3D est sa vision contextuelle limitée sur la dimension temporelle. Ce problème est abordé dans la littérature à l'aide de méthodes telles que les réseaux convolutifs temporels 3D [VLS18 ; Liu+18b], les réseaux neuronaux non locaux [Wan+18b ; Cao+19b] et les convolutions dilatées [Xu+20], mais celles-ci ne sont pas conçues pour le contexte en ligne. D'un autre côté, les réseaux convolutifs temporels 1D (TCN) ont émergé. Plus précisément, Wavenet [Oor+16] a été conçu pour traiter des séries temporelles 1D dans un contexte en ligne, avec des convolutions dilatées pour intégrer un contexte plus large. En outre, des convolutions causales sont utilisées pour produire une sortie à partir du passé uniquement. En utilisant cette architecture, SSNet [Liu+20a] a été présenté plus tard pour traiter la prédiction d'action basée sur le squelette, produisant une prédiction de classe pour chaque image. Cependant, comme Wavenet traite les entrées 1D, SSNet extrait d'abord une représentation spatiale à l'aide d'un CNN, puis utilise les caractéristiques résultantes comme entrée de SSNet, pour extraire les caractéristiques temporelles.

Le traitement séquentiel des informations spatiales puis temporelles ne permet pas d'extraire directement les caractéristiques spatio-temporelles, ce qui est un avantage clé des réseaux C3D. Notre réseau combine les avantages de ces deux catégories de réseaux, avec l'extraction directe des caractéristiques spatio-temporelles et le traitement en ligne.

Pour ce faire, nous avons développé un réseau causal avec des convolutions dilatées sur l'axe temporel. En utilisant des convolutions causales, le réseau peut produire une sortie uniquement à partir des images précédentes, ce qui garantit qu'aucune information des images futures n'est utilisée pour les prédictions. Les convolutions dilatées permettent au réseau d'augmenter rapidement son champ réceptif au fur et à mesure qu'il traverse les couches.

Un exemple de convolution 3D (première couche) d'OLT-C3D est fourni dans la figure 3.3.

Notre architecture est composée de 10 couches convolutives empilées. Ces couches sont divisées en deux blocs de 5 couches, avec un taux de dilatation égal à 2^i où $i \in \{0, 1, 2, 3, 4\}$ est l'indice de la couche dans le bloc, le taux de dilatation est de 16 pour les dernières couches des blocs. Ces taux de dilatation permettent d'augmenter le champ réceptif de manière optimale, chaque information d'entrée ne sera prise en compte qu'une seule fois pour calculer la sortie à un instant donné. Chaque couche de convolution est suivie d'une couche de max-pooling appliquée aux deux dimensions spatiales, il n'y a pas de pooling le long de la dimension temporelle.

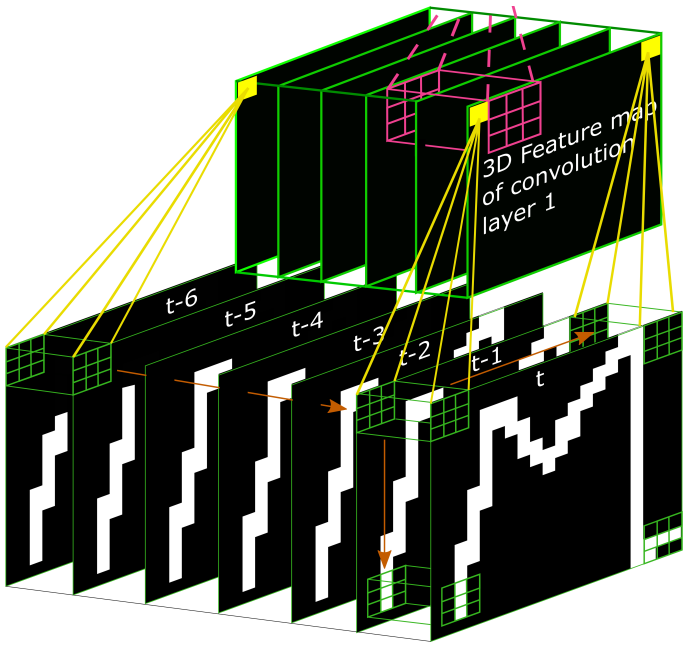


FIGURE 3.3 – Exemple de convolution spatio-temporelle 3D dans la première couche. La taille du filtre est de 2 (temps) × 3 (axe x) × 3 (axe y). Les filtres de convolutions circulent sur les trois axes, avec des convolutions causales sur la dimension du temps. Il n'y a pas de dilatation le long de l'axe temporel pour la première couche. Le filtre vert fait partie de la première couche. Le filtre rose fait partie de la deuxième couche, et possède un taux de dilatation égal à 2. Les filtres peuvent apprendre des motifs spatio-temporels grâce à la convolution 3D.

Comme le montre la figure 3.4, la couche convolutive inférieure a un champ réceptif très restreint puisque seules deux images sont utilisées pour calculer la sortie de cette couche, elle se concentre sur les deux dernières images. La deuxième couche utilise le résultat de la précédente, en utilisant indirectement quatre images. Le nombre d'images utilisées augmente avec le nombre de couches. La couche convolutive supérieure du réseau est capable de voir indirectement jusqu'à 63 images d'entrée (un bloc a une visibilité de 32 entrées, les superposer occasionne une intersection d'une frame). Le champ réceptif doit être accordé au paramètre θ défini dans la section 3.2.1 et à la longueur des gestes afin d'avoir l'historique suffisant permettant de voir l'ensemble du geste. Avec le paramètre θ que nous avons utilisé dans notre expérience, 63 images suffisent pour voir l'historique complet du geste dans la plupart des cas. Ces convolutions dilatées permettent à notre système d'éviter l'utilisation de mécanismes de cellules de mémoire comme les RNN utilisés dans certains travaux [Mol+16 ; Web+14] pour avoir une mémoire à long terme. Comme le réseau n'utilise que les cartes de caractéristiques correspondant à la dernière image, notre

réseau peut traiter n'importe quelle longueur de séquence et est capable de fournir une prédiction à chaque nouvelle image. Ce fonctionnement peut être assimilé à un principe de fenêtre glissante d'une taille de 63 images.

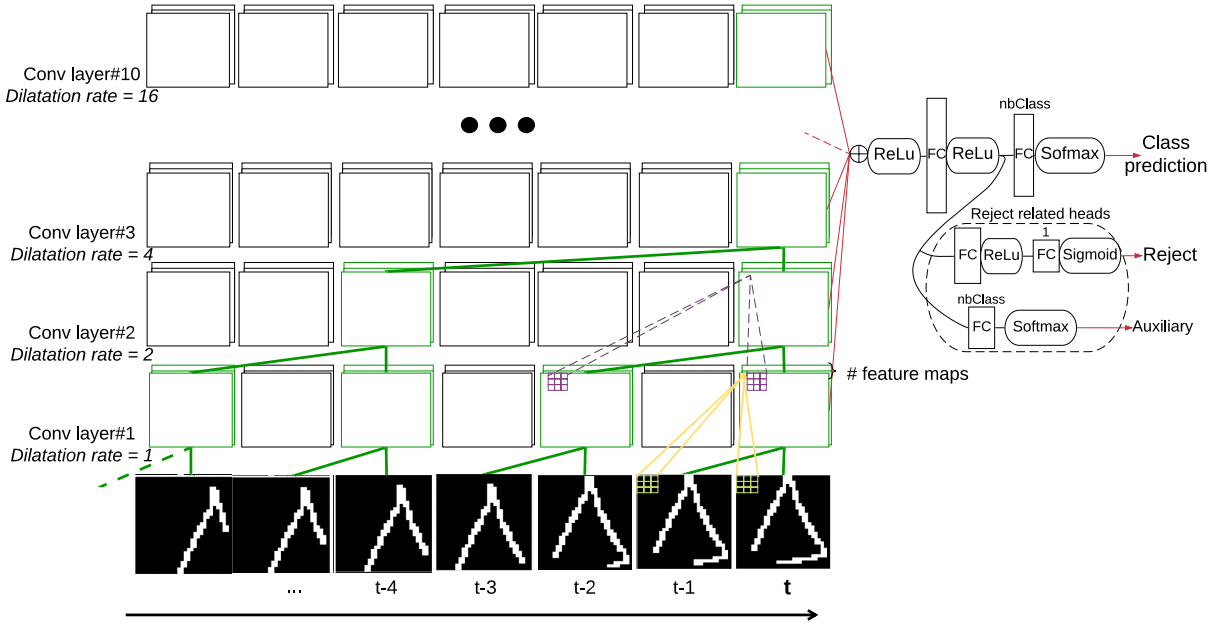


FIGURE 3.4 – L'architecture complète de l'OLT-C3D. Tout d'abord, le signal est transformé en images et introduit dans le réseau. Le réseau fait une prédiction à chaque instant. OLT-C3D est couplé à un système de rejet temporel.

L'objectif est de produire une nouvelle séquence correspondante avec une classification **par instant**, donc obtenir à la fin une matrice $T \times N$, où N est est le nombre de classe et T la longueur de la séquence[1]. La dimension originale de l'image est de dimension $T \times X \times Y \times C$, avec $C = 2$ (canaux de trajectoires et de contacts vus dans la section précédente). En appliquant une convolution 3D sans *padding* les trois dimensions X, Y et T commenceraient à se réduire. Comme on souhaite prédire une classe par instant t, il ne faut pas réduire cette dimension. On peut donc appliquer seulement le *padding* sur la dimension T. Ce *padding* doit se faire de façon à respecter la contrainte en ligne. Ainsi, les données paddées seront rajoutées à « gauche » (en référence à la figure 3.4), c'est-à-dire au début de la séquence. De cette manière les cartes de caractéristiques de n'importe quelle couche à l'instant t ne considèrent que les instants précédents. C'est ce padding qui fait que la convolution est **causale**. Des couches de pooling sont utilisées entre chaque couche, mais elles sont appliquées seulement sur les dimensions X et Y. En

1. lors de l'inférence en test, T correspond à la longueur de la séquence partielle observée

effet l'application d'une opération de pooling sur la dimension T avec un *stride* supérieur à 1 (généralement le cas pour un pooling) provoquerait un décalage qui causerait la perte de la causalité du réseau.

Afin d'utiliser des branches résiduelles entre les différents blocs, les dimensions X et Y doivent rester les mêmes pour toutes les couches. Il faut donc également appliquer un padding sur ces dimensions.

Une fois arrivé au bout des couches de convolutions, il faut réduire la dimension afin d'arriver à la fin à une matrice $T \times N$. Pour cette architecture nous avons aplati toutes les valeurs spatiales, nous obtenons donc une matrice de taille $T \times (X * Y * C)$. Juste avant, nous avons effectué une moyenne des caractéristiques sur l'ensemble des couches (ne change rien aux dimensions), de la même manière que le fait SSNet [Liu+20a]. Il s'agit d'une des différences majeures avec le réseau qui sera présenté dans le chapitre suivant, ce qui explique en partie la différence du nombre de paramètres (750K/420K ici, contre ≈150K dans le chapitre suivant pour le réseau pour le geste 2D). En effet, aplatir les cartes de caractéristiques augmente beaucoup le nombre de paramètres par rapport à l'application d'un *global average pooling*. Cette différence a été illustrée dans le chapitre sur l'état de l'art dans la figure 2.16.

L'axe T est entièrement conservé. Une couche FC est ensuite appliquée et permet d'obtenir une sortie de taille $T\times$ le nombre de neurones dans la couche FC. Ensuite, trois « têtes » sont créées (duplications des valeurs dans trois matrices distinctes, cf. figure 3.4). La première tête f est dédiée à la prédiction de classe, elle est composée d'une nouvelle couche FC avec le nombre de neurones de classe suivie d'une fonction d'activation softmax. La deuxième tête (Reject, g) est composée de deux couches FC successives et la troisième tête (Auxiliary, h) dispose de la même architecture que la première tête. Ces dernières têtes sont dédiées au système d'option de rejet, nous les détaillerons dans la section suivante. L'architecture complète est présentée dans la figure 3.4.

3.2.3 Système d'option de rejet temporel

L'un des principaux problèmes de la reconnaissance précoce est de ne prendre la décision de reconnaître un geste que lorsqu'il n'y a plus d'ambiguïté entre les gestes. Nous voulons laisser au classifieur la possibilité de reporter une décision lorsqu'il estime qu'il ne dispose pas de suffisamment d'informations. Par exemple, si nous avons des gestes avec des parties communes au début, nous voulons que le classifieur ne donne pas de réponse jusqu'à ce que la partie commune soit dépassée. Nous avons besoin d'un méca-

nisme permettant d'obtenir une sorte de score de confiance, afin de rejeter ou d'accepter la prédiction actuelle.

Pour résoudre ce problème, nous avons utilisé SelectiveNet [GE19], un mécanisme de rejet intégré dans un réseau que nous avons décrit en 2.3.2.3. Nous l'avons adapté de sorte à permettre un apprentissage dans un contexte temporel. Cela nous amène à ajouter deux nouvelles sorties : la tête de sélection/rejet et la tête auxiliaire, ces deux têtes sont incluses dans le bloc "Reject related heads" montré dans la figure 3.4. La tête de rejet est composée d'une couche FC avec activation ReLu. Elle est ensuite prolongée par une couche FC avec un seul neurone. Sigmoïde est la fonction d'activation finale utilisée dans cette tête, nous définissons cette sortie comme g. Le but de cette sortie est d'accepter ou de rejeter la prédiction. Nous considérons que la prédiction est rejetée si la sortie de la tête de rejet est inférieure à un paramètre τ, et qu'elle est acceptée si elle est supérieure. La sortie de prédiction finale par rapport à g est définie comme suit :

$$(f, g_\tau)(x) = \begin{cases} f(x), & \text{si } g(x) \geq \tau \\ ?, & \text{sinon.} \end{cases} \tag{3.1}$$

Nous avons utilisé $\tau = 0.5$ dans cet article, comme dans SelectiveNet. Enfin, la fonction de coût principale, en utilisant f et g, est définie comme suit :

$$\mathcal{L}_{(f,g)} = \frac{1}{m} \sum_{i=1}^{m} \ell\left(f\left(x_i\right), y_i\right) g\left(x_i\right) + \lambda \Psi\left(c - \hat{\phi}\left(g\right)\right) \tag{3.2}$$

où c est la couverture cible, λ est un hyperparamètre relatif à l'importance de la contrainte de couverture et $\Psi(a) = \max(0, a)^2$. Nous avons utilisé $\lambda = 32$. $\hat{\phi}(g)$ est la couverture empirique, c'est-à-dire la valeur moyenne de $g(x)$, et ℓ est l'entropie croisée. La moyenne de couverture $\hat{\phi}(g)$ et la fonction de coût ℓ sont calculées pour toutes les prédictions au cours du temps. Par conséquent, la couverture c est davantage liée à la précocité dans notre cas.

La tête auxiliaire est la même que la tête de prédiction, mais elle est optimisée avec une fonction de coût standard $\mathcal{L}_h$ (entropie croisé) pour l'entraîner à prédire la classe de chaque frame. Elle est utilisée pour optimiser la représentation CNN sans trop se concentrer sur la fonction de la tête de prédiction $\mathcal{L}_{(f,g)}$.

La fonction de coût finale est la suivante :

$$\mathcal{L} = \alpha \mathcal{L}_{(f,g)} + (1 - \alpha) \mathcal{L}_h \tag{3.3}$$

Dans nos expériences, nous avons fixé α à 0,5.

À chaque instant, OLT-C3D émet une prédiction de classe et le système de rejet temporel accepte ou rejette la prédiction s'il a besoin de plus d'informations. Le réseau peut également rejeter totalement le geste, même à la fin, si le geste est trop proche de deux classes, ou s'il ne correspond pas à une classe connue.

3.3 Bilan des contributions sur la reconnaissance précoce de gestes segmentés

Les contributions présentées dans cette section s'articulent en trois composantes.

D'abord, nous proposons une représentation euclidienne basée sur les trajectoires spatio-temporelles. Celle-ci a été conçue de manière à être indépendante de la vitesse d'exécution du geste. Avec le deuxième canal dédié à l'indication du contact des doigts avec la surface, le système dispose de toute l'information nécessaire pour effectuer la reconnaissance.

Ensuite, nous avons détaillé le réseau OLT-C3D, un CNN 3D que nous avons conçu pour extraire des caractéristiques spatio-temporelles. Ce réseau dispose d'une visibilité du passé suffisante pour permettre la captation du geste dans son ensemble. De plus, il est structurellement conçu pour ne pas exploiter les informations du futur.

Enfin, nous avons intégré au système un mécanisme de rejet directement intégré au réseau CNN. Ce mécanisme se base sur les têtes de sorties du réseau ainsi que les fonctions de coût de SelectiveNet. Nous l'avons adapté pour faire face au contexte temporel.

3.4 Expérimentations : évaluation sur la tâche de reconnaissance précoce de gestes 2D segmentés

Nous évaluons l'approche OLT-C3D sur deux ensembles de données librement disponibles : ILGDB [Ren+12] qui ne contient que des gestes **mono-stroke** et MTG-SetB [Che+15] qui contient des gestes **multi-touch**. Ces deux ensembles de données sont complémentaires en termes de nature des gestes (mono/multi-stroke, mono/multi-touch). Nous comparons nos scores à la méthode de l'état de l'art [Che+17] sur la tâche de reconnaissance précoce sur ces deux jeux de données.

3.4.1 Hyperparamètres et détails du réseau

La taille du pooling et du stride des couches de maxpooling sont de 3 pour les dimensions spatiales, 1 pour la dimension temporelle (pas de pooling). Nous avons utilisé un petit dropout pour les couches convolutives de 0.1, et 0.3 pour la première couche FC. ReLu est la fonction d'activation utilisée après chaque couche convolutive. Nous avons optimisé le réseau avec Adam [KB17] avec le taux d'apprentissage fixé à 0,003. 85 % des données d'entraînement sont utilisées pour l'entraînement, 15 % sont utilisées comme ensemble de validation. Les hyperparamètres spécifiques au jeu de données sont présentés dans le tableau 3.1. Pour ILGDB, nous avons fixé la dimension de l'image à 30 x 30. Les coordonnées des gestes sont mises à l'échelle par 0,2, et nous avons augmenté les données d'entraînement en mettant à l'échelle les coordonnées des gestes d'entraînement par 0,3, 0,4, 0,5 et 0,6. La quantité de déplacement θ est fixée à 4,5 pixels (une fois mise à l'échelle). En ce qui concerne le CNN, nous avons constaté que 10 filtres par couches sont suffisants pour cet ensemble de données, avec 300 neurones dans les couches FC. Nous avons utilisé un lot de 85 séquences complétées par des images noires à la fin. La couverture cible c est fixée à 0,6. Les hyperparamètres pour MTGSetB conduisent à un réseau plus important (770k paramètres pour MTGSetB et 420k pour ILGDB), ceci est dû au fait que ce jeu de données est plus complexe que ILGDB en raison d'un plus grand nombre de classes de gestes et de variétés de formes. Ces hyperparamètres ont été affinés à l'aide d'un ensemble de validation.

TABLE 3.1 – Hyperparamètres spécifiques aux bases de données

	Image dim.	Scale	Data aug. scale	filters	θ	FC neurons	Batch size	c
ILGDB	30x30	0.2	0.3, 0.4, 0.5, 0.6	10	4.5	300	85	0.6
MTGSetB	40x40	0.03	0.04	25	2	150	40	0.75

3.4.2 Métriques

Pour évaluer la tâche de reconnaissance précoce, nous utilisons le TAR (*True Acceptance Rate*) qui mesure la précision du classifieur lorsque la prédiction est acceptée. Nous utilisons également le FAR (*False Acceptance Rate*) qui mesure le taux d'erreur lorsque la prédiction est acceptée. Pour mesurer le rejet final, nous utilisons le RR (taux de rejet, *reject rate*) qui est le nombre d'échantillons totalement rejetés, même à la fin de la séquence. Ces mesures ont été définies dans la section 2.2.3.1.

Dans notre réseau, seule la classification de la première acceptation est utilisée pour calculer le TAR et le FAR. Si le geste n'est jamais accepté, il est pris en compte dans le RR. Pour évaluer la précocité, nous avons utilisé la Distance Normalisée de Détection (NDtoD).

3.4.3 Résultats de la reconnaissance précoce

3.4.3.1 Base ILGDB

La base de données ILG [Ren+12] est un **ensemble de données de gestes mono-stroke à base de stylo** réalisé par 38 utilisateurs. Elle contient 21 classes de gestes différentes avec un total de 1923 échantillons, dont 693 sont utilisés pour l'apprentissage et 1230 pour le test. La spécificité de cet ensemble de données est qu'il y a beaucoup de gestes qui ont des débuts communs, ou qui sont des sous-parties d'autres gestes. Les classes de la base sont illustrées dans la figure 3.5.

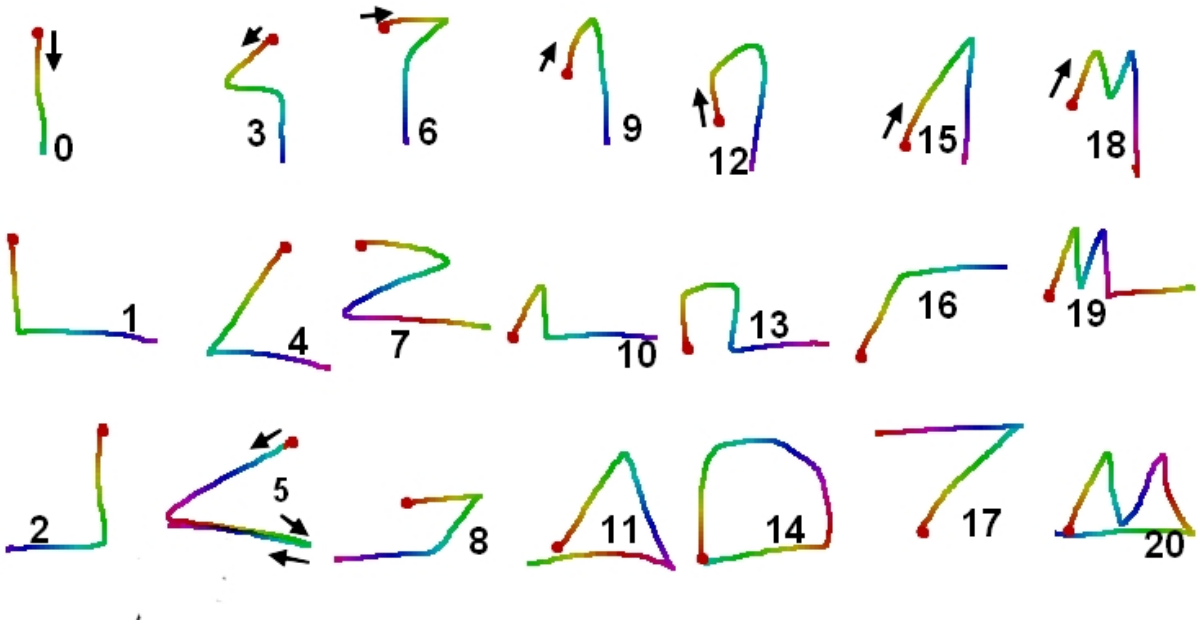
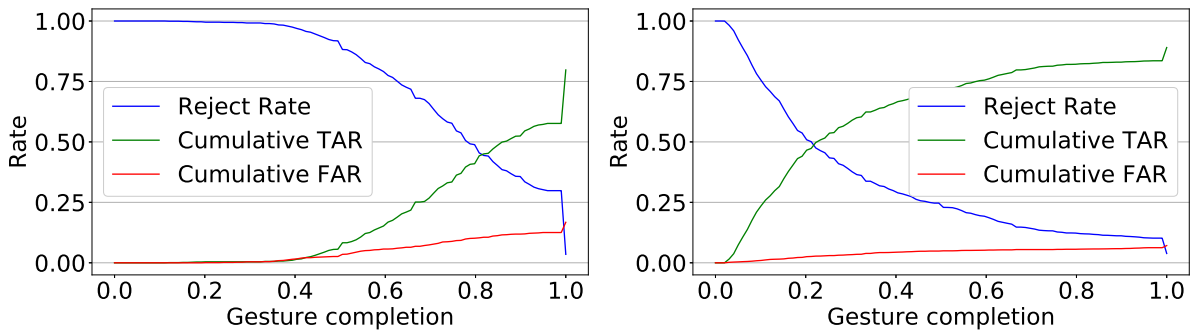


FIGURE 3.5 – Gestes de la base ILGDB. Les gestes sont construits par groupe de trois (arrangé par colonne ici), au sein d'un groupe les gestes possèdent un début commun, ce qui en fait une base très intéressante pour tester un système de reconnaissance précoce.

Le réseau doit donc rejeter jusqu'à ce que la trajectoire devienne discriminante, ce qui peut être très tardif. Nous avons comparé notre score à l'approche de Chen et al. [Che+17] en utilisant les ensembles d'apprentissage et de test prédéfinis fournis par le jeu de données.

Pour comparer équitablement nos résultats à ceux de l'état de l'art, nous avons utilisé le paramètre t qui est le nombre de fois que la même classe de prédiction doit être acceptée consécutivement par notre système de rejet pour être finalement acceptée. Il s'agit d'un moyen de renforcer la confiance, mais il conduit à des décisions retardées et à un taux de rejet plus élevé, ce qui peut être moins optimal que de régler le système de rejet. $t = 1$ est la valeur par défaut pour les autres expériences, sauf mention explicite. Les scores sont dans le tableau 3.2. On constate que pour une précocité équivalente ($t = 2$ pour Chen et al. et $t = 1$ pour nous), notre réseau est beaucoup plus précis avec 15 % de meilleure classification lorsque le geste est accepté, 10 % de moins de mauvaise classification, et moins de gestes rejetés. La décision est prise en moyenne à 76 % de la trajectoire totale du geste, ce qui est tardif, mais cohérent avec les gestes. Certains gestes ont des débuts communs, pour lesquels le réseau doit attendre que la partie commune soit passée. Dans la figure 3.6a, nous voyons que le réseau rejette la plupart des prédictions dans les premiers moments, attendant que le geste soit complété à au moins 40 % pour commencer à accepter les prédictions. Le FAR reste faible jusqu'à la fin. Nous pouvons observer un pic à 100 % de l'achèvement du geste, ce qui s'explique par le fait que certains gestes sont des sous-parties d'autres gestes, de sorte que la seule façon de reconnaître le geste est d'attendre que le geste soit complètement terminé, c'est notamment le cas du geste « 0 » visible dans la figure 3.5.



(a) ILGDB (b) MTGSetB

FIGURE 3.6 – Comportement de chaque taux (TAR cumulé, FAR cumulé, RR) sur les deux ensembles de données. Le TAR cumulé à $x\%$ d'achèvement est le nombre d'échantillons acceptés avant $x\%$ et correctement classés par rapport au nombre total d'échantillons.

TABLE 3.2 – Comparaison de l'approche OLT-C3D avec la méthode de Chen et al. [Che+17] sur les ensembles de données. TAR : taux d'acceptation réelle, FAR : taux d'acceptation erronée, RR : taux de rejet, NDtoD : distance de détection normalisée (précocité). t est le nombre d'acceptations consécutives de la même classe nécessaires pour qu'elle soit finalement acceptée.

Dataset	t	Chen et al. [Che+17]				**OLT-C3D**			
		TAR	FAR	RR	NDtoD	TAR	FAR	RR	NDtoD
ILGDB	1	30.65 %	67.15 %	2.20 %	34.81 %	**79.75** %	**16.74** %	**3.50** %	**76.81** %
	2	**64.15** %	**26.42** %	**9.43** %	**75.53** %	81.79 %	14.15 %	4.07 %	82.56 %
	3	73.98 %	11.22 %	14.80 %	92.24 %	83.25 %	12.03 %	4.72 %	87.44 %
	4	77.72 %	6.26 %	16.02 %	97.62 %	84.31 %	9.51 %	6.18 %	91.01 %
	5	77.80 %	4.88 %	17.32 %	99.19 %	85.20 %	7.40 %	7.40 %	93.71 %
	6	77.72 %	4.55 %	17.72 %	99.68 %	84.39 %	6.34 %	9.26 %	95.54 %
MTGSetB	1	**81.89** %	**14.56** %	**3.54** %	**37.04** %	89.25 %	7.24 %	3.51 %	30.77 %
	2	83.44 %	10.85 %	5.71 %	46.82 %	**90.52** %	**5.82** %	**3.66** %	**36.89** %
	3	82.38 %	8.85 %	8.77 %	55.89 %	90.94 %	5.08 %	3.98 %	42.78 %
	4	82.20 %	6.06 %	11.73 %	66.16 %	91.22 %	4.25 %	4.53 %	48.48 %
	5	80.35 %	4.60 %	15.05 %	71.03 %	91.49 %	3.53 %	4.98 %	53.92 %
	6	77.42 %	3.41 %	19.17 %	77.54 %	91.36 %	2.86 %	5.77 %	58.92 %

3.4.3.2 Base MTGSetB

L'ensemble de données MTGSetB est composé de 45 gestes **multi-touch** différents regroupés en 31 classes de gestes invariants en rotation, réalisés par 33 utilisateurs. L'ensemble des gestes est illustré dans la figure 3.7. Ces gestes sont divisés en trois catégories A, B et C, qui présentent différentes propriétés. Les gestes de catégorie A sont des gestes multi-touch, avec un doigt immobile sur la surface, et un ou deux autres doigts qui effectuent un ou deux tracés. Ceux de catégorie B sont composés de deux doigts ou plus qui effectuent des tracés en même temps. Les gestes appartenant à la catégorie C sont des gestes mono-touch multi-stroke, composés de deux tracés effectués successivement.

Comme pour ILGDB, nous avons comparé notre score à celui de Chen et al. [Che+17]. 50% des données sont utilisées pour l'entraînement. La comparaison est présentée dans le tableau 3.2. On constate que, pour une même précocité et un même taux de rejet ($t = 1$ pour Chen et al. et $t = 2$ pour nous), OLT-C3D a un TAR beaucoup plus élevé (+9%), un FAR beaucoup plus faible (−9%). L'ensemble de données MTGSetB contient des gestes multi-touch, et seuls quelques-uns d'entre eux sont des sous-parties d'autres gestes. En utilisant le nombre de doigts et les directions initiales, notre réseau est capable de prédire

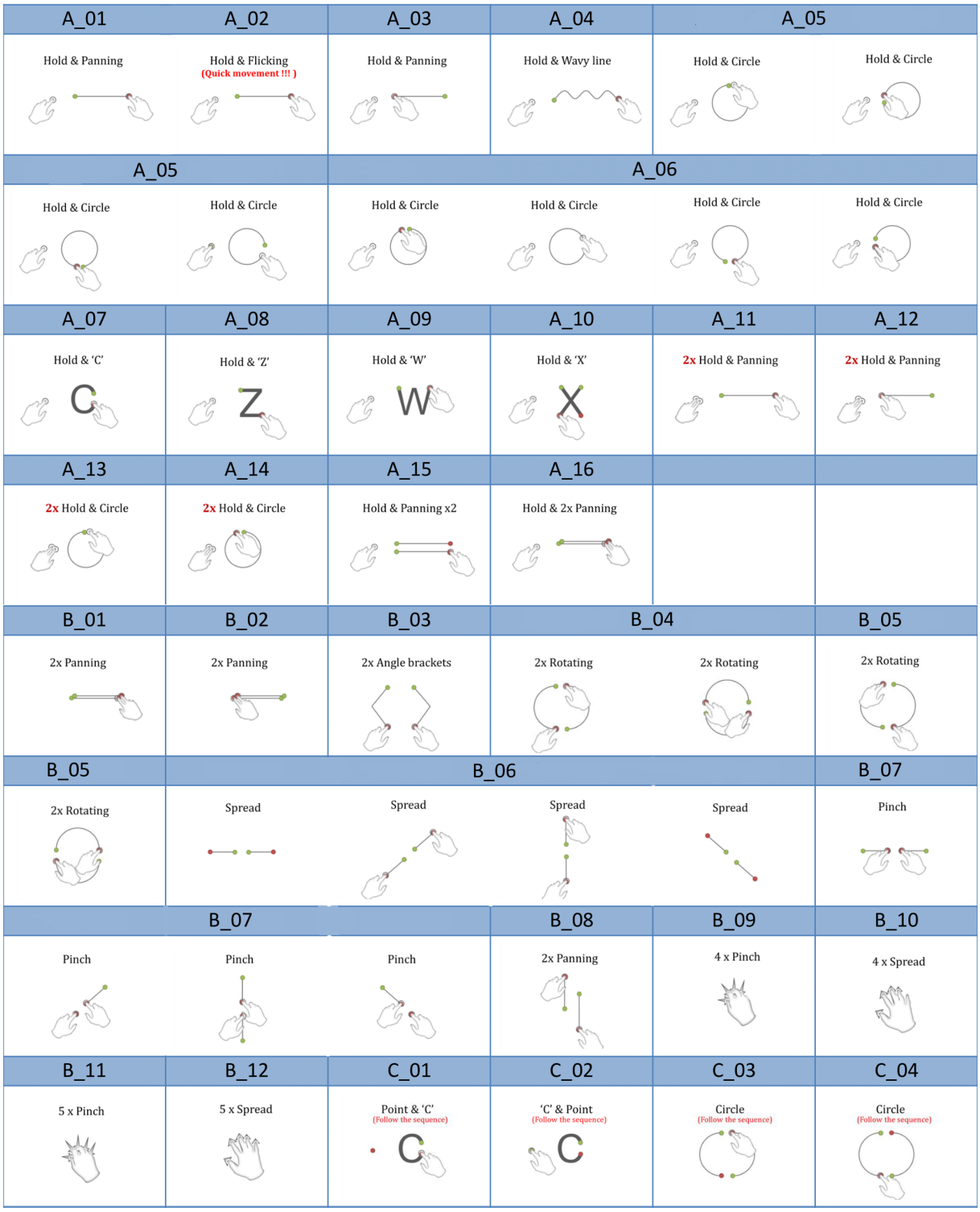
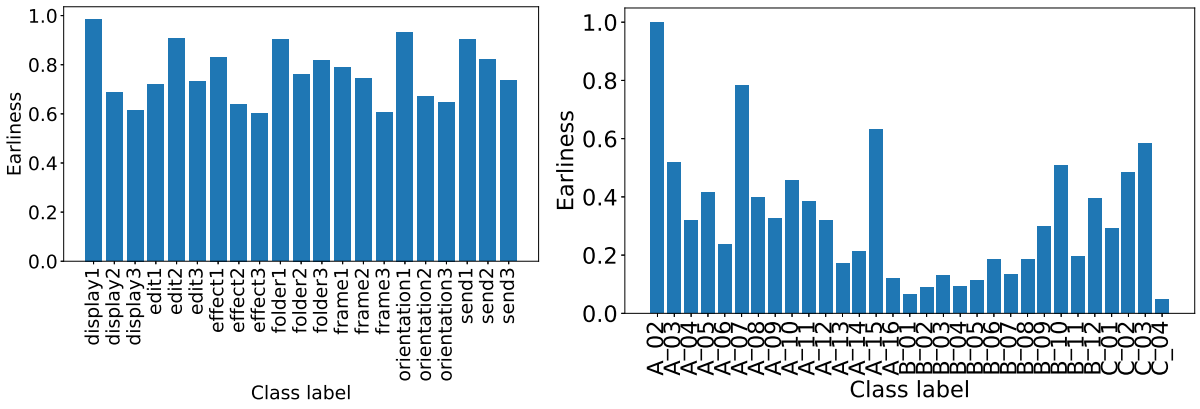


FIGURE 3.7 – Gestes de la base MTGSetB [Che+15]. Ces gestes sont divisés en trois catégories (*A*, *B* et *C*) qui disposent de propriétés différentes. Comme pour les expérimentations précédentes sur cette base [Che+17], A_01 et A_02 sont rassemblés dans la même classe.

les gestes très tôt, avec peu d'erreurs. Dans la figure 3.6b, nous voyons que le réseau commence à accepter très tôt, en gardant un faible FAR.

3.4.3.3 Évaluation de la précocité par classe.

La précocité par classe est présentée dans la figure 3.8. Pour ILGDB, nous observons des différences entre les classes, avec une précocité allant d'une moyenne de 60 % d'achèvement du geste pour la classe « effect3 » à 100 % pour la classe « display1 ». Notez que le geste correspondant à « display1 » est une sous-partie de deux autres classes de gestes, donc la seule façon de reconnaître le geste correctement est d'attendre que le geste soit complètement terminé. Dans MTGSetB, nous observons un cas similaire avec la classe « A_02 » qui est une sous-partie de plusieurs autres classes de gestes. Autrement, les gestes sont acceptés généralement tôt pour la plupart des classes. La classe prédite le plus tôt est la dernière, « C_04 », qui est acceptée en moyenne à 5 % de l'achèvement du geste. Le réseau est capable de prédire aussi tôt pour cette classe parce que c'est la seule avec un seul toucher qui commence par un déplacement vers la droite, donc il y a assez d'informations dans la première image dans la plupart des cas.



(a) ILGDB (b) MTGSetB

FIGURE 3.8 – Précocité par classe (NDToD).

3.4.4 Évaluation de la représentation spatio-temporelle

Nous comparons notre représentation discutée dans la section 3.2.1 avec deux variantes : l'une avec seulement la trajectoire, l'autre avec seulement la position du doigt. Les résultats sont présentés dans le tableau 3.3. Nous constatons que la trajectoire et la

position du doigt apportent toutes deux des informations significatives à la représentation. En particulier, le canal de la position du doigt apporte la différence entre un toucher constant et un toucher puis un relâchement.

TABLE 3.3 – Comparaison des différentes variantes de notre représentation sur MTGSetB.

Variant	TAR	FAR	RR	NDtoD
Trajectoire seule (premier canal)	84.1 %	9.73 %	6.17 %	**30.71** %
Seulement les positions des doigts (deuxième canal)	86.54 %	10.87 %	2.59 %	33.28 %
Les deux : trajectoire et positions des doigts	**89.25** %	**7.24** %	3.51 %	**30.77** %

3.4.5 Résultats qualitatifs

Dans les ensembles de données, certains gestes ont des parties communes. Le comportement attendu de notre réseau est de rejeter la prédiction jusqu'à ce que la partie commune soit passée. Dans cette section, nous analysons les résultats de notre système. Par exemple, ILGDB contient trois gestes qui commencent comme une lettre « M », la direction donnée après le trait « M » est décisive. La figure 3.9 montre un exemple du comportement de notre réseau sur ces trois étiquettes. On voit que le système de rejet

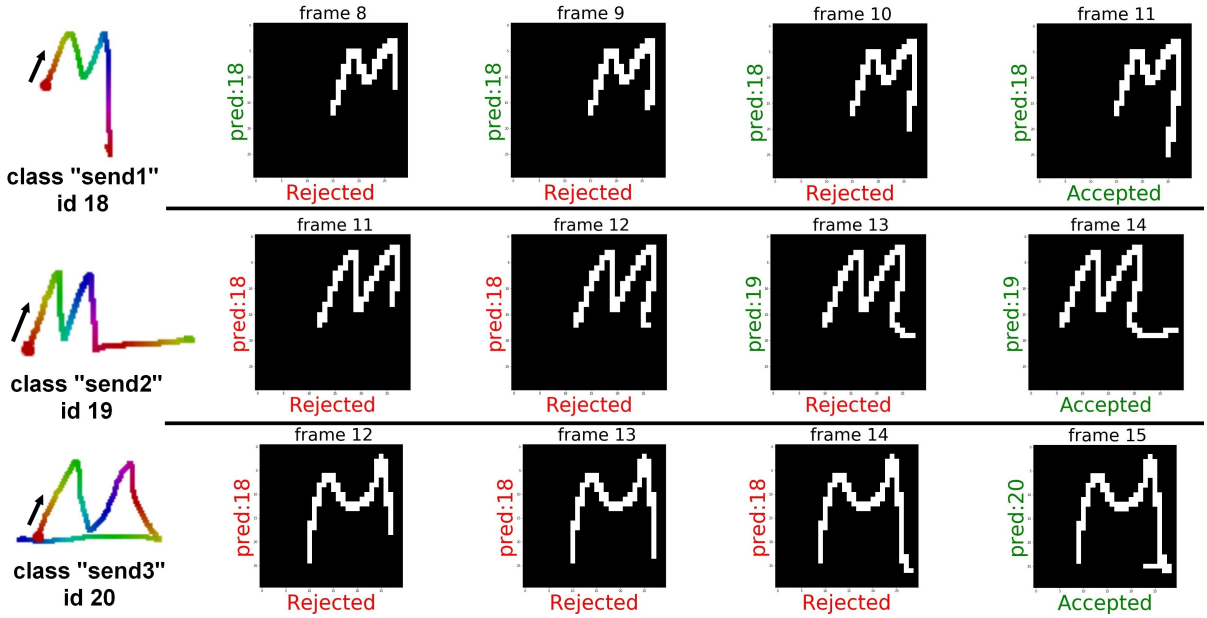


FIGURE 3.9 – Comportement sur les classes « M » jusqu'à la première acceptation. Le système rejette les prédictions jusqu'à l'instant décisif.

temporel attend l'instant décisif pour accepter la prédiction. Notons que cet exemple est très représentatif du comportement du réseau sur les classes « M ». Cela montre la capacité de notre approche à bien rejeter dans le temps la prédiction jusqu'à ce que la partie commune soit dépassée. Le cas de la classe « display1 » est également intéressant, ce geste est un simple trait vers le bas, comme illustré dans la figure 3.10. Ce geste est une sous-

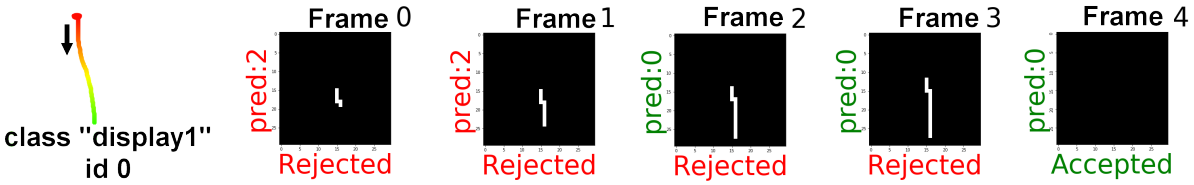


FIGURE 3.10 – La classe « display1 » est une sous-partie de deux autres classes, le système rejette les prédictions jusqu'à l'image noire (fin du geste). Une prédiction avec une étiquette verte à gauche signifie une bonne classification.

partie de deux autres gestes, dont l'un va vers la gauche et l'autre vers la droite après le trait vers le bas. Le seul moyen pour le réseau de savoir qu'il s'agit de la classe « display1 » est d'attendre la fin du geste (par exemple, le lever stylo pour les gestes mono-strokes). Comme expliqué dans la section 3.2.1, nous avons modélisé la fin du geste à l'aide d'une image noire. Pour cette classe de gestes, le réseau rejette la prédiction jusqu'à l'image noire.

Sur MTGSetB, la première acceptation est faite très tôt en moyenne. En analysant le nombre de contacts et le début de la trajectoire, le réseau est capable de faire des prédictions précises et très précoces. Un exemple est montré dans la figure 3.11. Dans cet exemple, nous voyons que le seul élément qui fait la différence entre les deux gestes est le canal de la position du doigt. Sans ce canal, le réseau ne serait pas capable de discriminer ces deux gestes.

3.4.6 Vitesse d'exécution

OLT-C3D est capable de traiter des données en continu à raison de **46** images par seconde, en considérant que le temps d'exécution de l'extraction de la représentation de l'image est négligeable. Notez que notre système attend d'acquérir suffisamment de déplacement pour soumettre la nouvelle image au réseau, dans ce cas il donnera une réponse en **≈22ms**. Ce délai est suffisant pour être utilisé dans une application en temps réel. Les expériences ont été menées sur une Quadro RTX 3000. Le temps de réponse peut être amélioré en utilisant la stratégie de partage d'activation utilisé dans SSNet [Liu+20a].

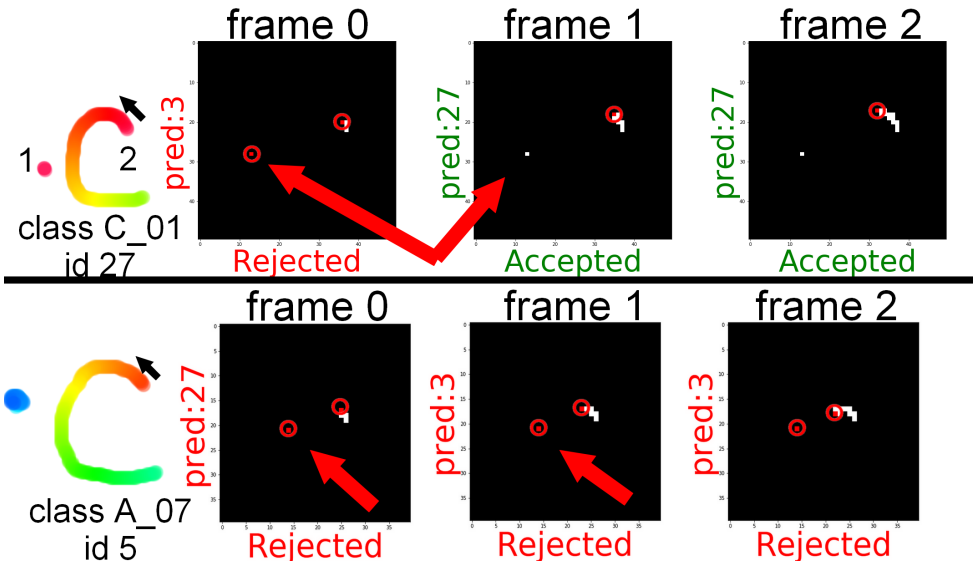


FIGURE 3.11 – Comportement des classes C_01 et A_07 de MTGSetB. Pour le geste de gauche, le réseau est capable d'accepter la prédiction dans l'image 1 parce qu'il peut voir que le doigt du trait gauche a été relâché, et c'est le seul geste qui commence ainsi. Pour le geste de droite, le doigt du trait gauche est continuellement pressé, ce qui peut représenter plusieurs gestes à cet instant, et le réseau rejette donc les prédictions pour ces images.

3.5 Conclusion

Dans cette section, nous avons proposé un système complet composé de trois parties principales. Tout d'abord, une **représentation spatio-temporelle** originale du geste, adaptée pour représenter le geste en ligne, quelle que soit sa nature (mono/multi-stroke, mono/multi-touch). Ensuite, **OLT-C3D**, un CNN 3D original capable d'extraire des caractéristiques spatio-temporelles en ligne. Enfin, un **système de rejet temporel** pour reporter la décision si nécessaire. Notre réseau couplé au système de rejet temporel est entraînable de bout en bout et fonctionne en temps réel. Nous avons montré que notre méthode est capable de faire des prédictions très tôt, avec des performances très intéressantes sur deux bases de données qui présentent une grande variété de typologie de gestes (mono-stroke pour ILGDB[Ren+12] et mono/multi-touch, multi-stroke pour MTGSetB [Che+15]). Ces résultats ont été publiés dans la conférence ICDAR 2021 [MAK21].

Cependant, cette approche **n'est pas directement utilisable dans un contexte non segmenté** du fait de trois éléments principaux. Premièrement, **la stratégie de représentation n'est pas applicable telle quelle** dans un contexte non segmenté. En

effet, si l'on garde la trace complète des gestes effectués alors l'image finira complètement marquée au bout de quelques gestes, conservant les traces dans anciens gestes rendant toute reconnaissance difficile voir impossible. Ensuite, le réseau est conçu pour exploiter une donnée 2D en entrée, en plus du temps. **Les trois dimensions de la convolution 3D sont déjà complètement exploitées**. Résoudre ce problème sera un des défis à aborder. De plus, nous avons cependant remarqué que la décision donnée par **Selective-Net n'est pas toujours stable dans le temps**. En effet, il peut arriver que la décision soit de nouveau rejetée après avoir été acceptée. Bien que cela n'ait aucune incidence dans le contexte segmenté, puisque seule la première acceptation est importante, cela rend difficile son utilisation dans un contexte non segmenté. Il va donc falloir trouver une autre technique afin de gérer le rejet dans le cadre non segmenté.

Nous allons détailler dans le prochain chapitre notre méthode pour répondre aux défis soulevés dans le cadre de la **reconnaissance précoce de gestes non segmentés**.

Chapitre 4

Détection Précoce de Gestes Non-Segmentés

4.1 Introduction

L'interaction gestuelle est devenue une composante essentielle de nombreux systèmes interactifs humain-machine. Ces systèmes nécessitent la détection et la reconnaissance en temps réel des gestes effectués par les utilisateurs. Dans cette étude, nous nous attaquons à la tâche de détection d'action en ligne (Online Action Detection - OAD), qui implique l'analyse d'un flux vidéo non segmenté pour détecter et reconnaître les gestes en temps réel.

Dans le contexte des systèmes interactifs, il est essentiel de disposer d'un niveau élevé de compréhension et de capacité de décision lorsqu'il s'agit de détecter des actions humaines.

Cependant, la plupart des approches OAD produisent un résultat au *niveau frame* sans vision d'ensemble de l'action réalisée. Cela conduit à une compréhension de bas niveau de la séquence et offre peu de garanties quant à la cohérence de la prédiction de l'action dans le temps.

Il est donc essentiel de développer des systèmes OAD qui produisent des résultats au *niveau de l'instance*. La sortie au niveau de l'instance peut être représentée à l'aide des bornes de détection ou d'un point de décision ponctuel. Le choix de la représentation de la sortie dépend des exigences spécifiques du système interactif.

Pour éclaircir à quel moment un système devrait prendre une décision de reconnaissance, Nowozin et al. [NS12] ont défini le concept de « point d'action ». Un point d'action dans une action fait référence à un moment spécifique où la présence de l'action est sans ambiguïté et peut être identifiée de manière cohérente dans toutes les instances de l'action. Un exemple est donné dans la figure 4.1, deux actions ayant un début similaire ne peuvent pas être clairement identifiées avant une certaine frame, à partir duquel le geste

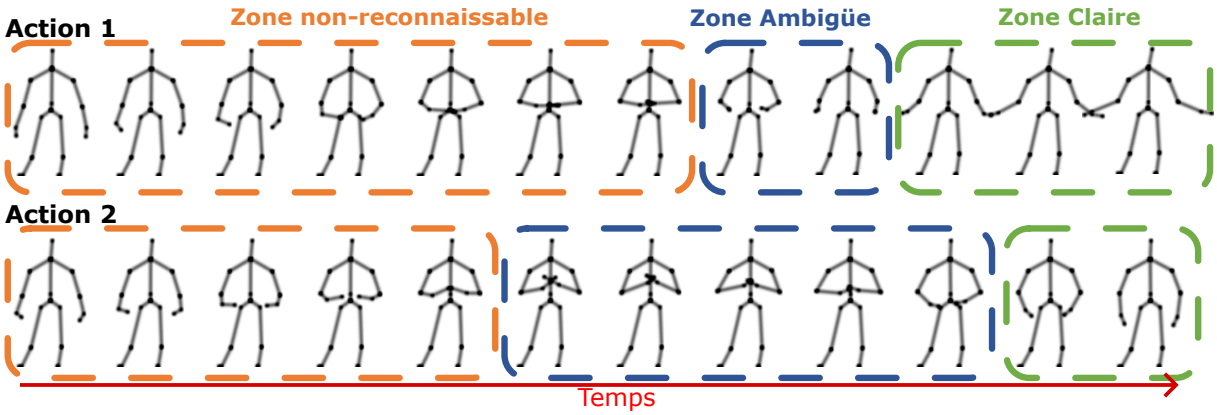


FIGURE 4.1 – Différentes zones d'un geste étant donné deux actions. Les premières frames contiennent les mêmes mouvements entre les deux actions considérées, les actions ne peuvent pas être différenciées. Une zone ambiguë intermédiaire peut être définie à l'endroit où de petits indices pourraient permettre d'identifier l'action. Dans la dernière zone, la classe est clairement identifiable.

devient clairement reconnaissable. Nous pouvons associer le point d'action à la première image de la zone claire. Ce point peut être difficile à définir puisqu'il implique toutes les classes de l'ensemble considéré.

Le niveau de précocité requis pour la reconnaissance des gestes dans les applications interactives varie en fonction des exigences et du contexte spécifiques de l'application. Si l'on se réfère à la figure 4.1, tenter de prédire les gestes dans la zone ambiguë peut permettre d'obtenir des systèmes de reconnaissance plus précoces, mais probablement au détriment de la précision globale. Par exemple, une application d'interface utilisateur basée sur des commandes qui utilise des gestes pour créer, modifier et déplacer des objets dans un espace graphique nécessiterait un haut niveau de précision pour avoir une interaction fluide. En revanche, la précocité n'est pas essentielle dans cette application. Par contre, une application d'entraînement sportif, telle que la boxe, où un entraîneur ou un adversaire virtuel doit réagir aux actions de l'utilisateur, exige que le système reconnaisse très rapidement l'attaque de l'utilisateur pour produire la bonne défense. Le système de reconnaissance doit donc être capable d'atteindre différents niveaux de précocité pour faire des compromis entre la précocité et la précision en fonction des exigences et des contraintes spécifiques de l'application.

Dans ce chapitre, nous présentons le système de détection d'action en ligne conçu pendant la thèse et qui est basé sur le squelette. Nous nous appuyons sur l'étude de Johansson [Joh73] qui a montré que la perception et la compréhension du mouvement du

corps humain est possible juste avec le mouvement des articulations du corps, c'est-à-dire le squelette. En outre, des capteurs comme la Kinect peuvent fournir efficacement des données sur le squelette à partir de cartes de profondeur [Sho+11]. Ainsi, nous estimons que les informations du squelette sont suffisantes pour reconnaître les gestes, et qu'elles fournissent une approche robuste et légère pour la détection d'actions en ligne.

Voici le résumé de nos contributions concernant la tâche d'OAD :

— Nous avons imaginé *E-SIM*, une représentation euclidienne du geste et indépendante de la vitesse d'exécution. Cette représentation construite à partir du squelette préserve les relations spatiales et temporelles entre les articulations, faisant d'elle une représentation bien adaptée au CNNs. Sa variante pour les gestes 2D, E-SI, dispose de propriétés similaires ;
— Nous avons construit le réseau *Dual-stream Online Long-term Convolutional 3D* (*DOLT-C3D*), un CNN 3D imaginé pour adresser les défis de l'OAD en extrayant des caractéristiques spatio-temporelles avec un contexte suffisant ;
— Nous avons élaboré deux stratégies d'apprentissage guidées par la segmentation des gestes. Basées sur le CTC, elles permettront de mieux localiser le geste dans le flux non segmenté.
— Nous avons proposé le *label prior* pondéré, une régularisation du CTC permettant de régler le ratio précision-précocité pour répondre aux différents besoins applicatifs.

4.2 Détection précoce de gestes non segmentés

4.2.1 Représentation Euclidienne Indépendante de la Vitesse

Dans un premier temps nous allons détailler notre méthode permettant de représenter le geste 3D (E-SIM) et 2D (E-SI). Cette méthode est basée sur une représentation du geste dans un espace euclidien afin de permettre une bonne exploitation par le réseau qui suivra. Nous distinguons dans les deux prochaines sections la représentation pour le geste 3D et pour le geste 2D, car même si les deux sont basées sur une représentation euclidienne, chacune dispose de ses propres spécificités liées aux natures de gestes.

4.2.1.1 E-SIM : Représentation à base de Cartes Euclidiennes Indépendantes de la Vitesse pour le geste 3D

Inspirés des travaux précédents sur les représentations des gestes indépendantes de la vitesse [Che+17 ; Bou+18a], et poussés par les limites des représentations compactes du squelette des approches exploitant des CNN, nos premiers essais de représentations ont été tournés vers une représentation euclidienne. Après avoir tenté de voxeliser le squelette en 3D, nous avons remarqué que la représentation était un peu lourde et généralement difficile à apprendre par les systèmes avec peu de données. Après avoir tenté de la simplifier, nos essais ont finalement convergé avec le travail de Duan et al. [Dua+22] publié dans la conférence CVPR en juin 2022, dédié à la reconnaissance de gestes segmentés hors ligne. Nous avons alors conçu une nouvelle représentation pour les systèmes basés sur le squelette, en exploitant les paramètres de la représentation de Duan et al. en l'adaptant avec l'idée d'indépendance de la vitesse et pour le contexte en ligne.

En utilisant directement les positions 3D estimées par les dispositifs tels que kinect, nous produisons des cartes thermiques (*heatmap*) 2D pour chaque étape temporelle. Les cartes thermiques générées sont des projections du squelette 3D dans un espace euclidien 2D, en ne conservant que les axes X et Y. Cette projection est effectuée une seconde fois avec les axes Y et Z car le déplacement le long de l'axe Z est généralement discriminant. Duan et al. estiment les positions des articulations à partir de la vidéo RVB, il serait tout à fait envisageable de faire de même dans notre cas.

Tout d'abord, nous devons normaliser le squelette. Comme nous sommes dans un contexte en ligne, nous ne pouvons pas normaliser en utilisant la boîte de délimitation globale pour toutes les images. Au lieu de cela, nous utilisons le squelette dans chaque image avec une distance invariable dans le temps, comme la longueur du bras. Nous normalisons le squelette avec une marge suffisante pour étirer les bras dans toutes les directions. Les coordonnées résultantes sont comprises entre 0 et W pour l'axe X, et entre 0 et H pour l'axe Y. La racine du squelette est centrée pour chaque image à la position $(W/2, H/2)$.

Deuxièmement, des « chunks » de frames (un chunk est un ensemble d'une ou plusieurs frames) sont constitués afin de créer une représentation indépendante de la vitesse. À l'intérieur de chaque nouveau chunk, une quantité équivalente de déplacement θ a été réalisée. Le déplacement de toutes les articulations concernées $k \in J$ est pris en compte pour calculer la quantité de déplacement. Nous pouvons obtenir le chunk c à partir de la liste des frames disponibles F, de taille V, qui n'ont pas encore été prises en compte et

qui sont classées par ordre chronologique de la manière suivante :

$$c = \left\{ f_v \in F \middle| \sum_v \sum_{J_k \in J} \| J_{k,f_{v-1}} - J_{k,f_v} \| \leq \theta \right\} \tag{4.1}$$

où $\|x\|$ est la norme euclidienne de x, J_{k,f_v} est le vecteur des coordonnées 3D de l'articulation k à la frame f_v. Un exemple de séquence décomposée en chunks est illustré dans la figure 4.2.

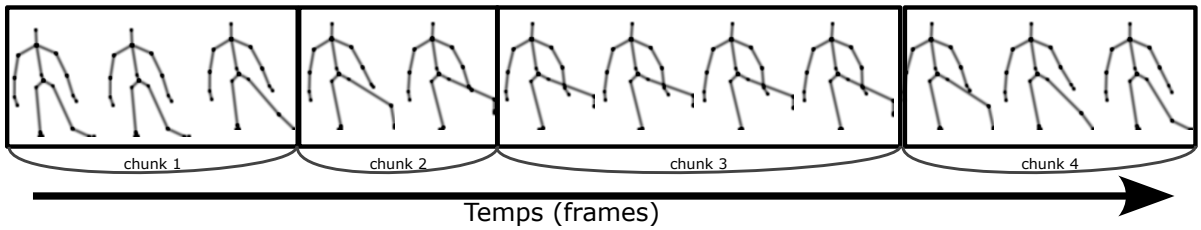


FIGURE 4.2 – Une séquence est décomposée en plusieurs chunks. Chaque chunk contient la même quantité de déplacement θ, et peuvent donc contenir un nombre différent de frames. Une représentation est extraite à partir de chaque chunk.

En plus d'être indépendante de la vitesse, ce qui rend le flux d'entrée plus cohérent pour notre réseau, cette stratégie de découpage est également un moyen très intéressant d'augmenter l'efficacité de notre système. Lors de l'apprentissage, beaucoup moins de données sont introduites dans notre modèle, ce qui permet un apprentissage beaucoup plus rapide. De plus, dans le contexte de test, le système attendra d'avoir suffisamment de déplacements pour faire une prédiction, ce qui évite de saturer le système.

Troisièmement, des cartes thermiques sont générées à partir de chaque chunk. Trois types de cartes thermiques seront dessinés pour chaque chunk : les cartes thermiques d'articulations $|J|$, les cartes thermiques d'os $|B|$ et une carte de trace de trajectoire. Nous produisons une carte par articulation et par os afin de toujours pouvoir les identifier séparément (par exemple la carte de l'articulation « épaule » sera toujours dans le même canal du réseau). En ce qui concerne les cartes thermiques des articulations, nous avons choisi de ne traiter que les pixels situés à une distance d de la position normalisée de l'articulation, afin d'optimiser le processus puisqu'il est destiné à un traitement en temps réel. L'intensité des pixels dépend de la distance aux articulations en utilisant la même formule que [Dua+22] :

$$\boldsymbol{E}^c_{k,i,j} = e^{-\frac{(i-x_k)^2+(j-y_k)^2}{2*\sigma^2}}, \forall k \in \{1, ..., |J|\}, \forall (i,j) \in \mathcal{I}^d_{c,k} \tag{4.2}$$

où $\boldsymbol{E}^c$ est l'ensemble final de cartes thermiques pour un chunk donné, chaque carte thermique est une image de taille $W \times H$, x_k et y_k est la coordonnée (x, y) de l'articulation k à la dernière image du chunk c. $\mathcal{I}^d_{c,k}$ est l'ensemble des coordonnées discrètes (pixels) à une distance d autour de (x_k, y_k), σ est le paramètre de variance. Tout comme l'usage initial, cette façon de « flouter » les positions des articulations a pour objectif de réduire l'impact des approximations d'estimation de la pose. En effet, du fait de la discrétisation en image du squelette, les positions précises peuvent passer d'un pixel à l'autre rien qu'avec un petit décalage. L'ordre des articulations n'est pas important. Ces cartes sont illustrées au centre de la figure 4.3.

Pour les os, en utilisant une idée similaire, l'intensité du pixel dépend de la distance au segment osseux :

$$\boldsymbol{E}^c_{|J|+b,i,j} = e^{-\frac{\mathcal{D}((i,j),[b_0,b_1])^2}{2*\sigma^2}}, \forall b \in \{1, ..., |B|\}, \forall (i,j) \in \beta^d_{c,k} \tag{4.3}$$

où $[b_0, b_1]$ est le segment déterminé par les articulations b_0 et b_1, $\mathcal{D}$ est une fonction calculant la distance entre un point et un segment. $\beta^d_{c,k}$ représente l'ensemble des coordonnées discrètes à l'intérieur de la boîte de délimitation donnée par les deux articulations de l'os à la dernière image du chunk c, avec des marges de d-unité. Les cartes dédiées aux os sont visibles à droite de la figure 4.3.

Une carte supplémentaire est ajoutée à la représentation. L'objectif est d'incorporer l'information temporelle de la trajectoire dans la représentation. Comme les cartes thermiques des articulations et des os ne prennent en compte que la dernière image du chunk, cette dernière carte fait le lien entre les deux derniers chunks en dessinant toutes les positions des articulations de toutes les frames du chunk dans la même image. L'intensité des pixels reflète la séquence temporelle. Ceux dont l'intensité maximale est de « 1 » représentent la position finale du squelette à l'intérieur du chunk. Cette stratégie permet de reconstruire les trajectoires en une seule image tout en préservant l'information sur l'ordre temporel. Cette carte est illustrée à gauche de la figure 4.3.

Pour chaque chunk, **la vue frontale** (VF, axes X et Y) produit $|J|+|B|+1$ cartes de taille $W \times H$, comme illustré dans la figure 4.3. En procédant de la même manière pour la projection de la **vue latérale** (VL, axes Y et Z), nous obtenons deux fois ce nombre de cartes.

Les deux flux produits, les vues frontales et latérales, seront introduits dans notre réseau DOLT-C3D.

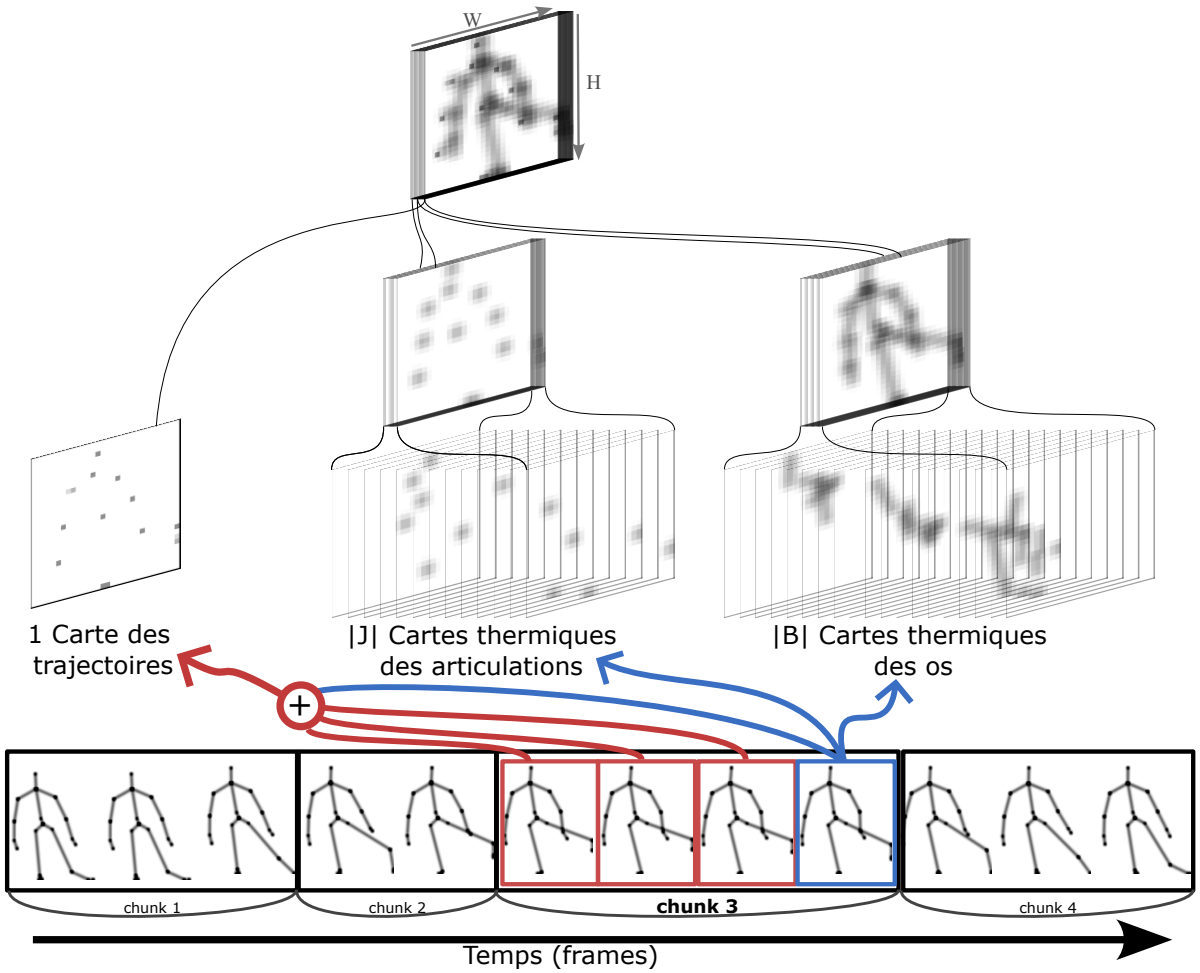


FIGURE 4.3 – Les cartes thermiques générées sont composées de 3 groupes, les cartes thermiques des articulations et les cartes thermiques des os représentant leur localisation spatiale, et la carte de la trace de la trajectoire qui représente le déplacement local de chaque articulation dans le même espace. Ici, le chunk 3 est représenté. Seule la dernière frame du chunk est utilisée afin de produire les cartes des articulations et des os, tandis que l'ensemble des frames du chunk sont utilisées pour construire la carte des trajectoires.

4.2.1.2 E-SI : Représentation Euclidienne Indépendante de la Vitesse pour le geste 2D

Pour représenter le geste 2D, nous utilisons une représentation similaire mais davantage adaptée au type de geste.

Pour déterminer notre méthode de représentation des gestes 2D, nous devons prendre en compte deux scénarios d'application différents qui exigent une reconnaissance précoce dans un contexte non segmenté. Dans le premier scénario, nous examinons des **gestes**

multi-touch effectués consécutivement. Entre deux tracés, tous les doigts peuvent être retirés du dispositif, et cela est également possible entre deux gestes distincts. Dans le second scénario, le doigt n'est jamais retiré de l'appareil et effectue uniquement des **gestes mono-stroke sans aucun lever**, similaire à l'écriture d'un mot avec des lettres.

Dans un premier temps, nous construisons des chunks de la même manière que pour la représentation E-SIM, afin d'avoir une représentation indépendante de la vitesse.

Une nouvelle difficulté, moins présente avec le geste 3D car le squelette peut être normalisé par frame, réside dans l'incapacité de prévoir la taille du geste 2D à l'avance. L'utilisateur peut effectuer le geste à n'importe quelle échelle, tandis que notre image possède une résolution spatiale fixe. Pour surmonter ce défi, nous avons préalablement défini une échelle, et si le geste atteint le bord de l'image, nous décalons l'image dans la direction opposée pour créer de l'espace. Ce choix se justifie du fait de l'utilisation d'un CNN dans la suite de l'approche, ceux-ci étant particulièrement robustes à la translation (plus que d'un changement d'échelle en cours de séquence).

Pour capturer la dynamique dans une image statique (dans quel sens va le geste), nous avons ajouté un deuxième canal à l'image pour indiquer la présence d'un doigt sur le dispositif. Ce canal supplémentaire est très clairsemé, ne contenant qu'une valeur (un « 1 ») dans les positions où les doigts se trouvent à chaque instant. Grâce à ce canal, le réseau peut déduire la direction dans laquelle le trait est dessiné. Il permet également de capturer le nombre de doigts en contact avec la surface, même si ceux-ci sont immobiles.

Dans le contexte segmenté (section 3.2.1), nous tracions toute la trace de la trajectoire depuis le début du geste. Cependant, dans un contexte non segmenté, cette approche n'est pas possible, car le début du geste est inconnu. Conserver l'intégralité des trajectoires des gestes précédents entraînerait un chevauchement des traces, rendant impossible toute reconnaissance. Nous avons donc adopté une stratégie de représentation compatible avec une séquence de gestes, en fonction du contexte des scénarios décrits ci-dessus. Pour les gestes multi-touch, nous traçons toute la trajectoire, mais celle-ci est complètement réinitialisée (image remise en noir) lorsque tous les doigts sont brièvement retirés du dispositif, que ce soit entre deux tracés ou deux gestes distincts. Cette approche permet d'accumuler la trajectoire du geste jusqu'à ce que tous les tracés simultanés soient terminés. Lorsque cela se produit, nous ajoutons une image noire pour indiquer explicitement cet événement au réseau. Cette stratégie assure au réseau la possibilité de prédire quelque chose sur cette image noire tout en étant certain que le trait est terminé, ce qui est crucial pour détecter les gestes qui sont des sous-parties d'autres gestes. Cependant, cette stratégie ne

s'applique pas aux gestes mono-stroke puisque le doigt n'est jamais retiré de l'appareil. Pour les gestes mono-stroke, étant donné l'absence de ruptures identifiables, la trajectoire du geste est accumulée dans une fenêtre glissante de taille ω. Chaque image résultante contient un déplacement égal à $\omega \times \theta$. Une fenêtre de glissement trop grande génère des images bruitées avec des traces potentiellement superposées à des parties de gestes antérieurs [1]. Un exemple de cette représentation est illustré dans la figure 4.4.

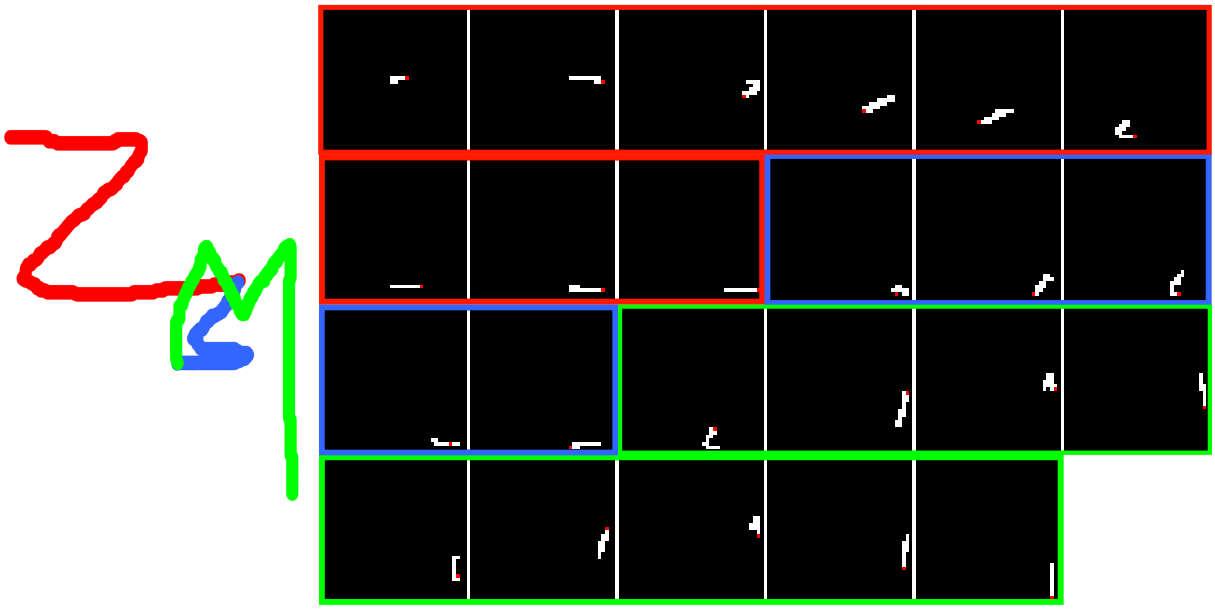


FIGURE 4.4 – Représentation E-SI ($\omega = 2$) d'une séquence mono-stroke sans lever de crayon comportant trois gestes. Chaque image intègre un nouvel élément d'information.

4.2.2 Réseau de neurones à convolution spatio-temporel : DOLT-C3D

Le réseau 3D convolutif à long terme à double flux en ligne (DOLT-C3D - Dual-stream Online Long-Term Convolutional 3D network) que nous présentons ici est principalement inspiré de notre réseau OLT-C3D décrit dans la section 3.2.2, où il a été appliqué à la reconnaissance des gestes segmentés 2D. Cependant, dans cette section, nous étendons son application à la reconnaissance des gestes en 3D en introduisant la capacité de traiter **deux flux en ligne simultanément**, permettant l'observation à partir de deux points de vue différents : la vue frontale (VF) et la vue latérale (VL). Étant donné que le geste

1. Pour le geste 3D, ω était toujours fixé à 1 du fait du nombre élevé de trajectoires.

se fait en réalité en 3D, il est essentiel de ne pas perdre la dimension « Z ». L'usage de deux points de vue projetés en 2D nous permet de conserver l'usage des convolutions 3D, en ne perdant que peu d'informations.

Notre réseau est composé de blocs OLT-C3D. Chaque bloc OLT-C3D comporte quatre couches convolutives 3D avec un taux de dilatation égal respectivement à 1, 2, 4 et 8 sur l'axe temporel. Un bloc est représenté dans la figure 4.5. Avec un noyau temporel de taille 2, le bloc ne prend qu'une seule fois chaque chunk disponible dans son champ réceptif, ce qui rend le traitement efficace. Chaque couche convolutive est suivie d'une couche de max-pooling appliquée aux deux dimensions spatiales. Comme nous devons toujours avoir une sortie par chunk qui respecte la contrainte en ligne, aucun pooling n'est appliqué le long de la dimension temporelle.

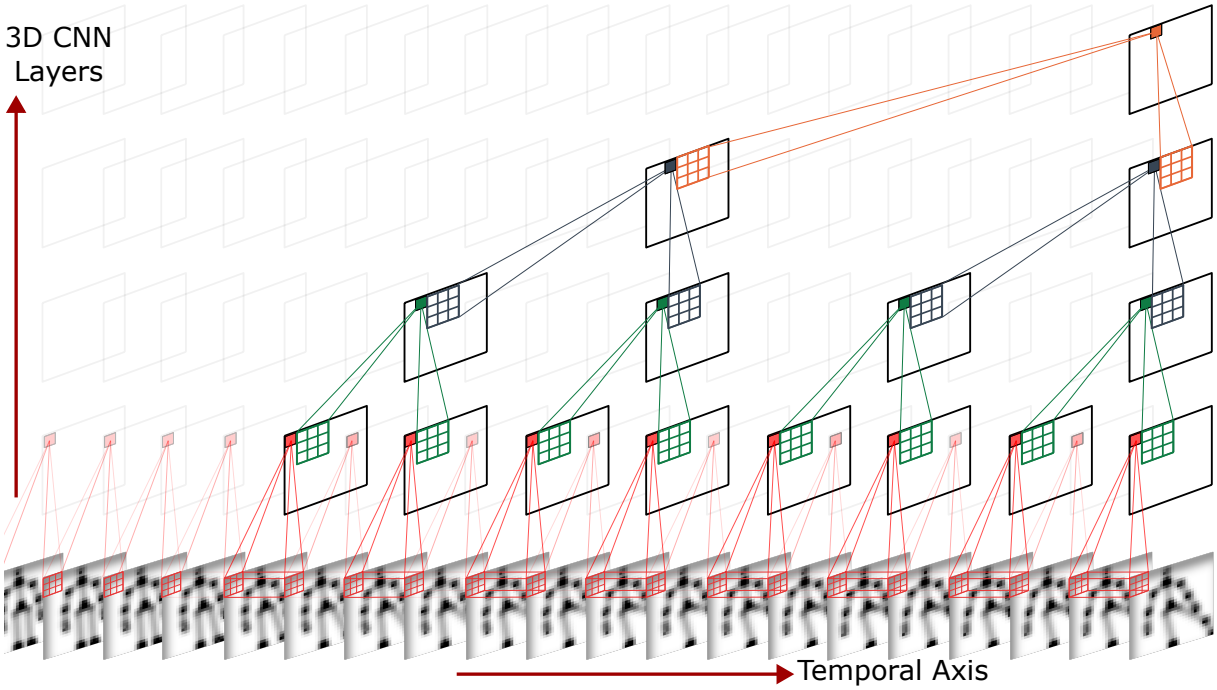


FIGURE 4.5 – Le bloc OLT-C3D est composé de 4 couches de convolution 3D. Chaque couche augmente son champ réceptif en utilisant un taux de dilatation plus important sur l'axe temporel. Les convolutions sont causales pour respecter la contrainte en ligne.

Le réseau complet, représenté dans la figure 4.6, comprend quatre blocs OLT-C3D. Cependant, il peut être ajusté en fonction de la complexité souhaitée du réseau et du champ réceptif requis. En effet, chaque bloc ayant une visibilité de 16 entrées du fait des 4 couches de convolution, superposer les blocs augmente la visibilité maximale, avec 4

blocs, la visibilité est de 61 chunks [2] $(16 + 15 * 3{=}61)$, ce qui est largement suffisant dans notre contexte. Le nombre de couches à l'intérieur d'un bloc peut également être ajusté.

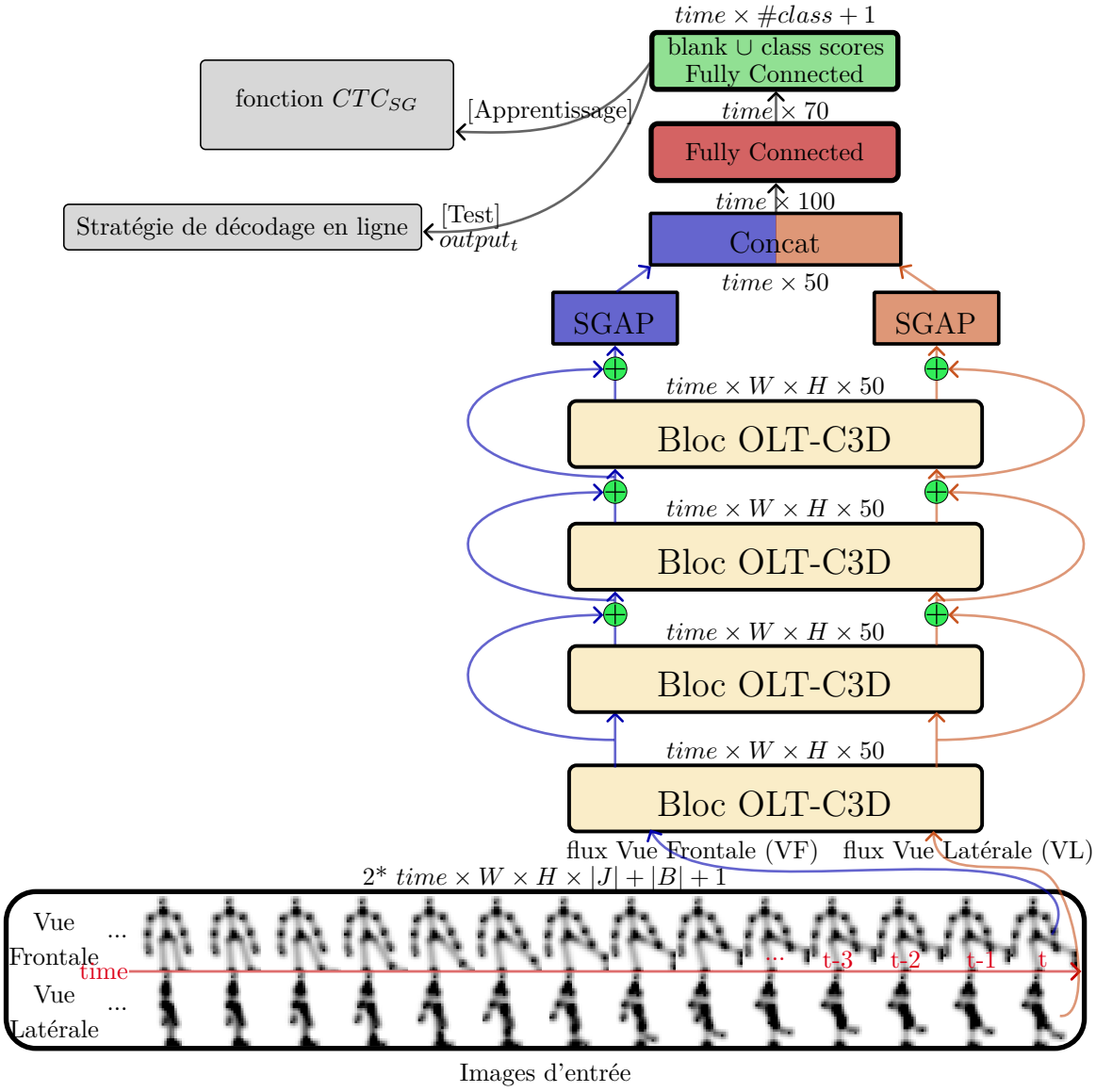


FIGURE 4.6 – Le réseau DOLT-C3D est composé de 4 blocs OLT-C3D et se termine par des couches entièrement connectées pour produire le nombre de classes + 1 scores.

Comme indiqué précédemment, notre réseau est à double flux. Il reçoit des entrées de la vue frontale (VF) et de la vue latérale (VL) par le biais de deux flux distincts. Les deux flux

2. et non 64, car la frame utilisée la plus ancienne d'un bloc est réutilisé une fois par le bloc supérieur. Il y a une intersection d'une frame par bloc rajouté, donc 3 frames sont utilisées deux fois, donc pour trois blocs : $16 * 4 - 3 = 61$.

partagent les mêmes blocs OLT-C3D avec les mêmes poids pour optimiser l'apprentissage des caractéristiques. Les caractéristiques de sortie de chaque flux sont ensuite concaténées après une couche *global average pooling* sur les dimensions *spatiales* (SGAP), de sorte que la couche entièrement connectée (FC) suivante puisse constamment identifier le flux d'où proviennent les caractéristiques. Les caractéristiques sont ensuite transmises à deux couches entièrement connectées pour produire la sortie. Le réseau émet une prédiction de classification pour chaque nouveau chunk. Notre architecture peut prendre en charge n'importe quelle longueur de séquence pendant l'apprentissage.

Pendant la phase de test, une stratégie de décodage en ligne est nécessaire pour réaliser une détection au niveau de l'instance sur la base des prédictions par chunk. Cette stratégie est illustrée dans la figure 4.7. À chaque image, nous déterminons la classe la plus probable comme prédiction. La borne de début de détection est émise lorsqu'une classe différente de la précédente est détectée sur un chunk. Par la suite, nous considérons que l'action se poursuit jusqu'à ce qu'une prédiction de classe différente soit rencontrée pour un chunk. Pour traiter les images qui ne correspondent à aucun geste ou celles dont la prédiction de classe est incertaine, nous utilisons une classe « Aucun geste » (appelée *blank*).

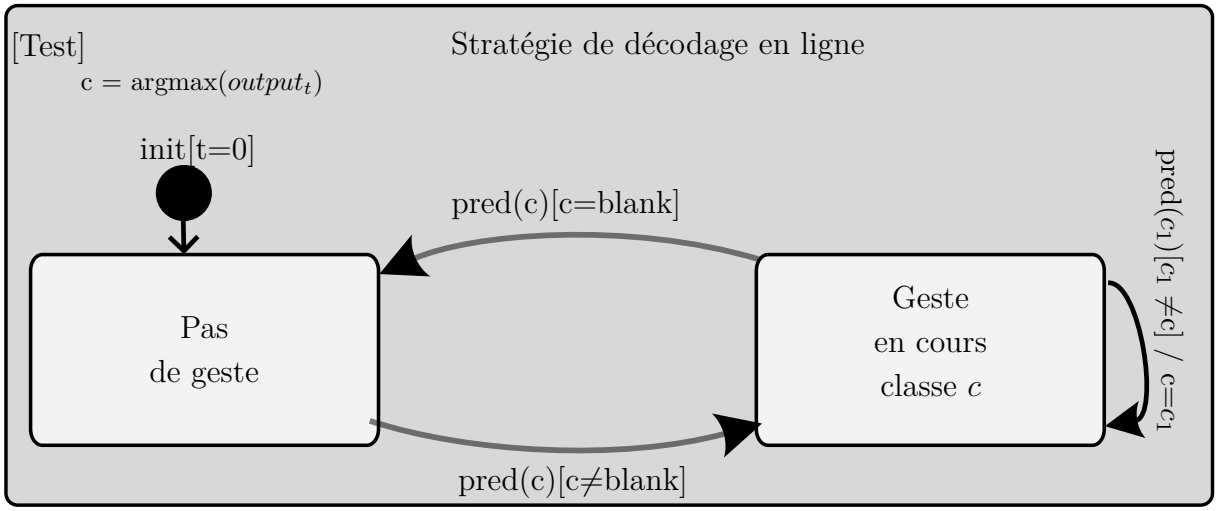


FIGURE 4.7 – Stratégie de décodage en ligne utilisée. Le blank est utilisé comme délimiteur, l'action détectée reste la même instance tant la prédiction reste la même. Il s'agit de la même stratégie que le décodage classique « glouton » du CTC.

Pendant l'apprentissage, nous avons besoin d'une fonction de coût qui soit cohérente avec l'objectif de sortie au niveau de l'instance et avec la stratégie de décodage en ligne. Nous choisissons d'utiliser la fonction CTC [Gra+06] qui répond à cette exigence.

4.2.3 Apprentissage via un CTC guidé par la segmentation pour une meilleure localisation des gestes

La fonction de coût CTC [Gra+06], décrite en 2.3.3.3, va nous servir de base de mécanisme de décision. La CTC est mise en œuvre pour entraîner un modèle à générer une séquence d'étiquettes à partir d'une séquence de données d'entrée, ce qui génère une sortie au niveau de l'instance. Une étude réalisée par Zeyer et al. [ZSN21] confirme que les systèmes appris avec cette fonction de coût ont tendance à prédire un « pic » de probabilité de classe dans un nombre restreint d'images, tandis que les autres frames sont qualifiées du label spécial "blank". De plus, les pics ne sont pas nécessairement alignés avec les frames correspondant aux actions. L'effet est encore accentué du fait du traitement en ligne où le CTC aura tendance à prédire en fin de geste ou après la fin, comme cela est illustré avec l'exemple de la figure 4.8.

L'utilisation du CTC garantira la production de résultats au niveau de l'instance. Cependant, le comportement de localisation temporelle imprécis mentionné précédemment entrave notre objectif de détection précoce. Dans cette section, nous abordons le problème de la **localisation des pics** en ajoutant des contraintes aux chemins appris par la fonction CTC. Pour cela, pendant l'apprentissage, nous utilisons un **CTC guidé par la connaissance de la segmentation temporelle** (CTC_{SG}, SG : *Segmentation Guided*), que nous avons avec l'annotation de la vérité terrain. L'objectif est de faire en sorte que le pic se produise entre le début et la fin du geste, comme l'illustre la figure 4.8.

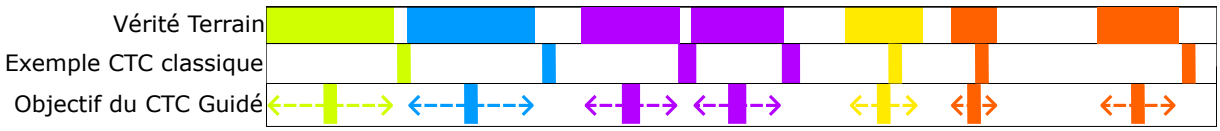


FIGURE 4.8 – Le CTC classique a tendance à effectuer des « pics » de prédictions mal localisés par rapport au geste, en ligne le pic se fera en moyenne plutôt à la fin du geste, ou même après celui-ci. L'objectif du CTC guidé est d'apprendre au système à localiser le pic à l'intérieur des bornes du geste, entre le début et la fin.

Soit $x = (\boldsymbol{E}^1, \boldsymbol{E}^2, ..., \boldsymbol{E}^C)$ notre séquence d'entrée où C est le nombre de chunks pour une séquence donnée. $l = \{1, ..., L\}$ représente l'ensemble des étiquettes de sortie possible et L est le nombre total d'étiquettes. Soit $y = (y_1, y_2, ..., y_U)$ l'étiquette pour une séquence donnée où chaque $y_u \in l \cup \epsilon$, et ϵ représente l'étiquette vide. La séquence y est composée des étiquettes de classe ordonnées dans le temps, avec des étiquettes vides insérées avant chaque étiquette et à la fin. L'algorithme CTC apprend à prédire la séquence de sortie

$\hat{y}$ en fonction de la séquence d'entrée x. Pour l'apprentissage, il est d'abord nécessaire de construire un graphe représentant tous les alignements possibles entre x et y. Chaque nœud dans le graphe correspond à un alignement possible où le c-ème élément d'entrée est aligné avec la u-ème étiquette de sortie. Les arêtes entre les nœuds représentent les transitions valides entre les étiquettes. Une arête peut aller d'un nœud $n_{c,u}$ à un nœud $n_{c+1,u'}$ si la condition $\mathcal{C}$ est vérifiée :

$$\mathcal{C}(n_{c,u}, n_{c+1,u'}) = \begin{cases} y_{u'} \in \{y_u, y_{u+1}\} \\ \textbf{ou} \\ y_{u'} \in \{y_{u+2}\} \textbf{ et } y_{u'} \neq \epsilon \\ \qquad\qquad \textbf{et } y_{u'} \neq y_u \end{cases} \tag{4.4}$$

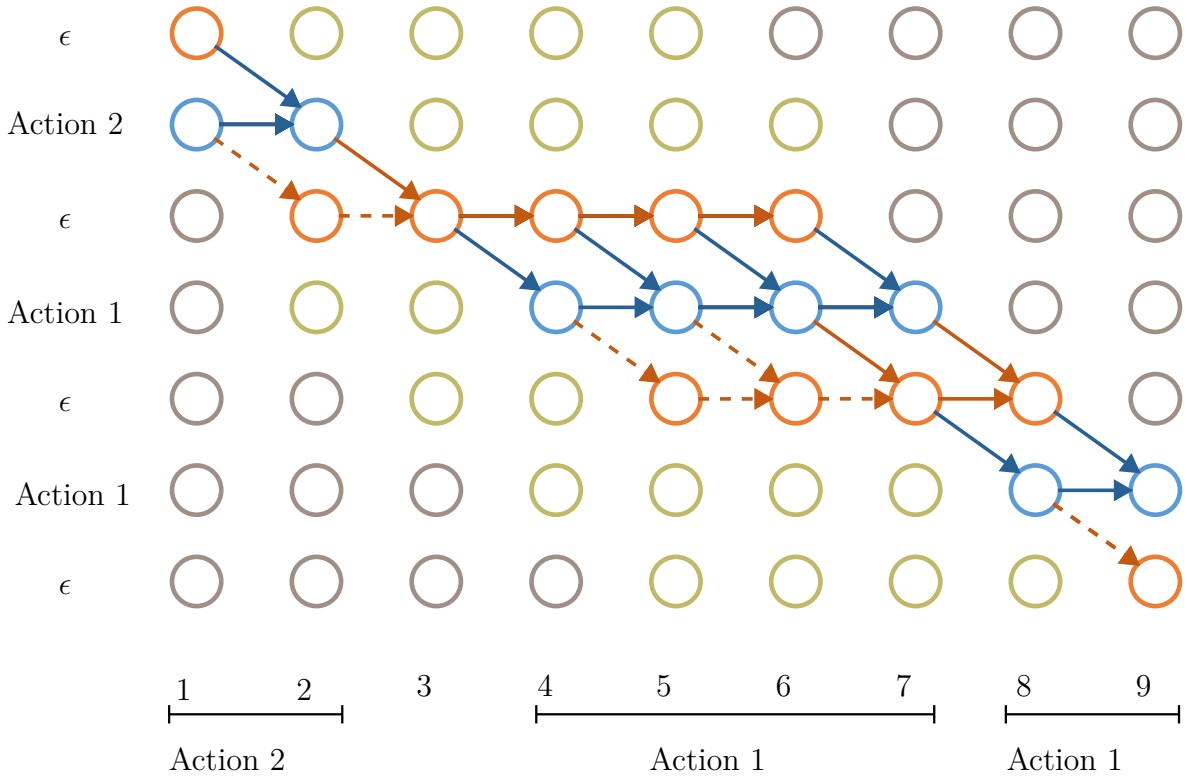


FIGURE 4.9 – Graphe du CTC, le graphe élagué après l'élagage SSG et l'élagage HSG (sans les transitions en pointillés). Les nœuds jaunes sont ceux qui sont totalement supprimés du graphe CTC classique à l'aide des deux stratégies d'élagage.

Jusqu'à présent, toutes les approches qui font de l'OAD exploitent les annotations de segmentation (début/fin) de gestes durant l'apprentissage. Le CTC classique n'utiliserait

que l'ordre dans laquelle les actions sont effectuées, indépendamment de leurs localisations. Ainsi, nous proposons deux versions d'élagage du graphe CTC guidé par la segmentation afin d'optimiser uniquement les chemins pertinents pendant l'apprentissage. L'objectif est de faire en sorte que les pics se produisent pendant les images des actions. Nous présentons deux versions d'élagage de la fonction de coût CTC : l'élagage *Soft Segmentation-Guided (SSG)* et l'élagage *Hard Segmentation-Guided (HSG)*.

L'*élagage SSG* permet l'optimisation de tout chemin passant par au moins une frame classée comme action **pendant l'action**. Cet élagage est suffisamment souple pour permettre la prédiction des blanks jusqu'à l'avant-dernière frame de l'action (prédiction tardive) et à partir de la deuxième frame de l'action (pic de détection précoce). L'élagage SSG ressemble au "Local-CTC" utilisé dans [ZMV20]. Nous pouvons écrire la condition de transition pour l'élagage SSG comme suit :

$$\mathcal{C}_{SSG}(n_{c,u}, n_{c+1,u'}) = \begin{cases} \mathcal{C}(n_{c,u}, n_{c+1,u'}) \\ \textbf{et} \\ y_{u'} \neq \epsilon \implies s_{u'} \leq c+1 \leq e_{u'} \end{cases} \tag{4.5}$$

où $s_{u'}$ et $e_{u'}$ sont respectivement les frames de début et fin de l'action u'. Le graphe élagué qui en résulte est illustré à la figure 4.9.

Le *HSG pruning* supprime les chemins de la version SSG allant des actions aux blanks pendant l'action (pic de détection précoce). L'objectif est d'encourager le réseau à continuer à prédire cette classe jusqu'à la fin de l'action. Cet élagage est plus pertinent avec le label prior présenté dans la section suivante.

$$\mathcal{C}_{HSG}(n_{c,u}, n_{c+1,u'}) = \begin{cases} \mathcal{C}_{SSG}(n_{c,u}, n_{c+1,u'}) \\ \textbf{et} \\ y_{u'} = \epsilon \implies s_{u'} \leq c+1 \leq e_{u'} \\ \qquad\qquad \textbf{ou} \ (y_{u+2} = y_u \ \textbf{et} \ c+2 = s_{u+2}) \end{cases} \tag{4.6}$$

Notez que la condition $s_{u'} \leq c+1 \leq e_{u'}$ est toujours fausse lorsqu'il n'y a pas de blank entre deux actions. Comme deux actions consécutives avec la même étiquette de classe ont besoin d'un blank pour décoder efficacement deux actions différentes, la condition $(y_{u+2} = y_u \ \textbf{et} \ c+2 = s_{u+2})$ assure d'avoir un chemin passant par la première frame de la seconde action puisqu'elle permet à la dernière frame de la première action d'être un

blank. Le graphe résultant est présenté dans la figure 4.9 (chemins sans les transitions en pointillés).

Notre fonction de coût, CTC guidée par la segmentation $\mathcal{L}_{CTC_{SG}}$, est définie comme suit :

$$\mathcal{L}_{CTC_{SG}} = -\log \sum_{\pi \in paths|\mathcal{C}_{SG}} \prod_{c:C} p(n_{c,\pi_c}) \tag{4.7}$$

où $paths|\mathcal{C}_{SG}$ sont tous les chemins menant à la séquence d'étiquettes correcte, où les transitions vérifient la condition guidée par la segmentation $\mathcal{C}_{SG}$. $p(n_{c,\pi_c})$ est la probabilité de sortie du modèle pour l'étiquette π_c à l'étape c.

4.2.4 Ajout d'une pondération des scores de classes a priori pour le réglage de la précocité

Dans le contexte en ligne, la précocité d'un système fait référence à sa capacité à détecter et à reconnaître avec précision un geste dès que possible au cours de son exécution. Généralement, pendant les dernières frames d'un geste, il est plus facile de classer l'action car le système a accès à toutes les informations à ce moment-là. Nous nous attendons donc à ce que les pics de CTC se produisent en moyenne dans les dernières frames des gestes. Cependant, notre objectif est de détecter les gestes le plus tôt possible.

Une approche possible pourrait consister à ajouter davantage de contraintes au graphe. Cependant, comme nous ne disposons pas de connaissances préalables sur la localisation temporelle du point d'action théorique des gestes (c'est-à-dire l'image à partir de laquelle le geste devient reconnaissable), l'ajout de contraintes supplémentaires pourrait éliminer des chemins pertinents. De plus, compte tenu du fait que l'annotation des bornes de début/fin des gestes dans la vérité de terrain n'est pas toujours cohérente en raison des variations entre les annotateurs, il est nécessaire de permettre une certaine flexibilité dans les chemins.

Ainsi, au lieu d'essayer de déplacer le pic de prédiction d'action plus tôt, nous présentons ici un moyen d'aplanir progressivement les pics en prédisant un plus grand nombre de frames dans un geste. En combinaison avec l'élagage guidé par la segmentation présenté dans la section précédente, nous nous attendons à ce que la première image classée comme étant l'action arrive plus tôt en aplanissant le pic. Cependant, comme il est risqué de prendre une décision avec moins d'informations, le système fera probablement plus de fausses détections à partir d'un certain point. Il y a donc un compromis précision/pré-

cocité que nous voulons étudier afin d'être cohérents avec l'application utilisée. Pour y remédier, nous aplanissons les pics en utilisant un *label prior pondéré* dans nos versions élaguées de la CTC, sur la base de l'étude de Zeyer et al. [ZSN21].

En introduisant le terme du *label prior*, la contribution des étiquettes fréquemment prédites, telles que l'étiquette *blank*, est effectivement pondérée à la baisse dans la fonction de coût. Cela encourage le modèle à produire des distributions d'étiquettes de sortie qui sont plus uniformément distribuées, en mettant moins l'accent sur les étiquettes surreprésentées qui tendent à être prédites plus fréquemment. En effet, lorsque l'on regarde la construction du graphe du CTC (décris dans l'état de l'art en section 2.3.3.3), on peut voir que le blank est surreprésenté par rapport aux autres classes. Celui-ci est systématiquement présent entre chaque classe annotée. Le terme *softmax label prior* est défini comme suit :

$$P_{prior}(\pi_c) = \frac{1}{C} \sum_{d:C} p(n_{d,\pi_c}) \tag{4.8}$$

qui est la probabilité moyenne de l'étiquette π_c sur la séquence. Ce terme peut également être estimé sur l'ensemble des données d'apprentissage [ZSN21].

Prédire une étiquette *blank* pendant une action **n'est pas une erreur**, car l'action peut être ambiguë à cette étape. Les blanks sont nécessaires, en particulier dans les premiers instants. La fonction de coût CTC classique (guidée par la segmentation ou non) a l'avantage de produire des séquences de prédictions très stables avec moins d'erreurs, car l'étiquette blank est prédite fréquemment et est considérée comme sûre. En divisant le score de la prédiction par son *label prior* (comme le montre l'équation 4.9), les étiquettes deviennent parfaitement équilibrées, ce qui fait que les actions et le blank sont aussi probables les uns que les autres. Cela conduira également à plus d'erreurs car le blank sera moins prédit. Pour permettre d'ajuster cet équilibre entre les étiquettes, nous ajoutons un poids Ψ sur le *label prior*. $\Psi = 0$ conduirait à la fonction de coût classique du CTC, $\Psi = 1$ équilibre totalement les étiquettes. L'ajustement de ce poids permettra d'affiner l'équilibre entre la précision et précocité. Ce *label prior* peut être appliqué soit sur le CTC classique, soit sur le CTC guidé. Dans notre cas nous allons le coupler avec la version guidée afin de contraindre la localisation de l'aplanissement.

Notre fonction de coût finale est :

$$\mathcal{L}_{CTC_{SG,\Psi}} = -\log \sum_{\pi \in paths | \mathcal{C}_{SG}} \prod_{c:C} \frac{p(n_{c,\pi_c})}{SGrad(P_{prior}(\pi_c)^{\Psi})} \tag{4.9}$$

où SGrad (Stop Gradient) signifie que cette partie n'est pas optimisée par le réseau.

En résumé, cette méthode de pondération permet aux étiquettes de devenir plus ou moins équiprobable a priori. En rendant le blank moins probable, celui-ci sera moins prédit, et donc les pics de prédictions seront aplanis. Combiné à la contrainte de localisation du CTC guidé, l'apprentissage va encourager à prédire les « pics aplatis » entre le début et la fin du geste. Par effet un peu indirect, plus le pic sera aplati, plus il aura tendance à commencer tôt et donc à prédire précocement. Ceci, tout en conservant l'objectif classique du CTC qui est de mener à la bonne séquence de gestes à la fin, et donc un niveau de compréhension au **niveau instance**.

4.3 Bilan des contributions sur la détection précoce de gestes non segmentés (OAD)

Les contributions de ce chapitre se décomposent en quatre éléments.

Tout d'abord, nous proposons une **représentation euclidienne** du geste. Si la représentation euclidienne du 2D peut sembler plutôt naturelle du fait de son utilisation classique dans les méthodes de reconnaissance d'écriture manuscrite, son utilisation dans le domaine du geste 3D est relativement originale. En effet, comme nous l'avons vu dans l'état de l'art (section 2.5.2), les représentations matricielles compactes ont été principalement utilisées, même si quelques approches récentes utilisent des représentations euclidiennes [TLL18 ; Shi+20 ; Dua+22]. Utiliser une représentation euclidienne nous semble particulièrement intéressant lorsqu'elle est exploitée par un réseau CNN. En plus de l'utilisation d'un espace de représentation euclidien, notre approche vise à s'abstraire de la vitesse d'exécution du geste en composant des regroupements de frames (chunks) qui contiennent une même quantité de déplacement. Cette approche réduit aussi le nombre de données à traiter par le réseau. Ensuite, en ne remplissant les cartes thermiques que localement autour des articulations et des os, il est possible d'optimiser encore un peu plus la vitesse d'exécution du système.

Deuxièmement, nous avons conçu **DOLT-C3D**, une architecture CNN basée sur OLT-C3D, qui permet de traiter deux flux de données en parallèle. Du fait du partage des poids entre les deux flux dans les couches de convolution, le réseau peut ainsi réutiliser des informations apprises sur un flux pour l'appliquer sur l'autre.

Ensuite, des contraintes sur le graphe de la fonction de coût CTC sont utilisées afin de produire un **CTC guidé par la segmentation**. Deux versions sont proposées : SSG

(Soft-Segmentation Guided) et HSG (Hard-Segmentation Guided). SSG est plus souple et « autorise » le réseau à prédire des blanks après avoir prédit une classe, même s'il n'a prédit qu'une seule fois la classe au début du geste. La version HSG est plus contraignante, cette fois dès que le système commence à prédire la classe le réseau apprend à continuer la prédiction de la classe jusqu'à la fin du geste.

Pour finir, nous avons ajouté un **label prior pondéré**, qui a pour objectif d'aplatir les pics du CTC. Sa combinaison avec le CTC guidé permettra une prédiction plus précoce. De plus, en faisant varier l'intensité de la pondération, il est possible d'ajuster le compromis précision/précocité.

4.4 Expérimentations

4.4.1 Détails d'implémentation

Pour réduire le bruit dans les positions des articulations, un filtre de Butterworth léger et en ligne est appliqué avant de calculer la représentation. Pour normaliser le squelette, nous utilisons la distance tête-root (tête-ventre) comme distance invariante. Nous considérons que ces articulations sont détectées avec plus de précision par les dispositifs Kinect que les articulations des bras. Nous avons construit notre implémentation du CTC basée sur l'implémentation CTC de Liu et al. [LJZ18]. Nous avons intégré une fonction de coût de lissage en calculant l'entropie croisée entre la prédiction actuelle au moment t et la prédiction précédente au moment $t-1$, multipliée par un poids de 10 pour s'assurer qu'elle soit du même ordre de grandeur que la fonction CTC. Les hyperparamètres de notre réseau sont les mêmes pour tous les ensembles de données : les dimensions de l'image de sortie utilisées pour E-SIM sont $15 \times 15 \times 15$ avec une distance $d = 2$ (sauf mention explicite dans l'étude d'ablation) et θ est fixé à 3. 13 articulations ont été sélectionnées pour être représentées dans les cartes thermiques des articulations, mais tous les os sont utilisés. La variance de la carte thermique σ est fixée à 1,3. En ce qui concerne le réseau, nous avons utilisé quatre blocs OLT-C3D de quatre couches CNN 3D avec 50 filtres chacune, ReLu est utilisé après les convolutions et la couche entièrement connectée (FC). Le Dropout est ajouté après chaque couche de convolution (0,2) et après la couche FC (0,3). 70 neurones sont utilisés dans la couche FC. Le maxpooling spatial est utilisé après chaque couche de convolution ($1 \times 3 \times 3$) avec un padding pour garder les mêmes dimensions. La taille du batch est fixée à 4 séquences de ≈ 200 chunks (40 pour

G3D car les séquences sont beaucoup plus petites). Pour les gestes et les bases qui s'y prêtent (ceux qui sont symétriques), nous avons augmenté les données en faisant une mise en miroir de la séquence. L'entraînement est effectué avec l'optimiseur Adam. Le réseau a un total de ≈715K paramètres entraînables. Notre implémentation est disponible sur `https://gitlab.inria.fr/intuidocenlignepublic/OLT-C3D_OAD`. Pour évaluer notre système, nous avons utilisé notre framework d'évaluation que nous avons rendu public à l'adresse suivante : `https://gitlab.inria.fr/intuidocenlignepublic/evaluation-framework-OAD` pour de futures expériences dans le domaine.

Concernant les expérimentations sur le geste 2D, nous avons utilisé la représentation E-SI présentée en section 4.2.1.2, avec $\omega = 2$. Le réseau est identique à celui présenté dans ce chapitre, mais il est utilisé en simple flux. En revanche, seuls deux blocs OLT-C3D de 5 couches de convolutions sont utilisés. Le Dropout est appliqué dans toutes les couches convolutionnelles et denses, avec un taux de 0,1 pour les couches convolutionnelles et de 0,2 pour les couches denses. Chaque couche convolutive apprend 30 filtres. Après les couches convolutionnelles, une couche dense de 100 unités est utilisée, dont toutes les sorties sont partagées. Le réseau dispose d'environ 150K paramètres. Pendant l'apprentissage, une rotation aléatoire est appliquée à la séquence pour augmenter les données (toutes les images de la séquence subissent la même rotation), suivant une distribution normale avec $\mu = 0$ et $\sigma = 15°$, afin d'améliorer la généralisation. L'entraînement est réalisé avec un batch de 5 séquences.

4.4.2 Base de données 3D

Pour évaluer les performances de notre système, nous avons réalisé des expériences sur six ensembles de données couramment utilisés dans la littérature pour la détection d'actions non segmentées. Dans cette section, nous décrivons chacun de ces ensembles de données. Un résumé des ensembles de données est donné dans le tableau 4.1.

4.4.2.1 L'ensemble de données MSRC-12

L'ensemble de données Microsoft Research Cambridge-12 (MSRC-12) [Fot+12], conçu pour la détection d'actions, comprend 594 séquences non segmentées mettant en scène 30 sujets effectuant 12 gestes. 20 articulations du squelette sont capturées à 30 images par seconde à l'aide de la Kinect. Ces gestes sont divisés en catégories iconiques et métaphoriques. Le même geste est répété dans chaque séquence. L'ensemble de données a été

TABLE 4.1 – Résumé des ensembles de données de gestes 3D utilisés dans cette thèse. Le nombre de séquences désigne le décompte initial ; un nombre alternatif est mentionné si différent (sous-ensemble ou ensemble étendu). Les détails sont donnés dans la section correspondante de l'ensemble de données.

Nom	**#classes**	**#sequences /utilisées**	**Kinect**	**Annotation débuts/fins**	**Annotation Point d'action**
MSRC-12 [Fot+12]	12	594	V1	✓	✓
MSRC6-Iconic-C4 [Fot+12]	6	58	V1	✓	✓
OAD [Li+16]	10	59	V2	✓	✗
G3D [BMA12] (Fighting)	5	30/33	V1	✓	✓
MAD [Hua+14]	35	40/107	V1	✓	✗
Chalearn [Esc+13]	20	680	V1	✓	✗
PKU-MMD [Liu+17]	43 (1pers.)	1076/860	V2	✓	✗

collecté avec cinq modalités d'instruction et annoté à l'origine avec des points d'action. Plus tard, un autre travail a proposé une annotation avec l'annotation début et fin de chaque geste [Hus+13] . Bien que la répétition des gestes de la même classe dans les séquences ne reflète pas les scénarios du monde réel, cet ensemble de données est utile pour comparer notre méthode avec les approches précédentes.

Un sous-ensemble de cet ensemble de données est également couramment utilisé, avec 58 séquences (provenant de la modalité d'instruction « C4 Video+Text ») avec les 6 classes iconiques. Nous donnons le nom de « MSRC6-Iconic-C4 » à ce sous-ensemble pour plus de clarté. Un exemple de séquence est visible en vidéo à cette adresse : lien [3].

4.4.2.2 L'ensemble de données OAD

L'ensemble de données OAD (Online Action Detection) [Li+16] se compose de 59 séquences non segmentées enregistrées dans un environnement intérieur quotidien à l'aide de la Kinect v2 à 8 images par seconde. Il comprend 10 actions d'activités quotidiennes, telles que manger, boire et écrire, avec des ordres et des durées variables. Ce petit ensemble de données représente un défi pour les méthodes d'apprentissage profond en raison de sa petite taille. De plus, l'ordre aléatoire des gestes le rend plus difficile et plus proche des scénarios du monde réel. Un exemple de séquence est visible à cette adresse : lien [4].

3. Lien temporaire avant migration vers le site de Shadoc : MSRC12 `https://drive.google.com/file/d/17MOa1MLejpFM56NLODUDW9NScWVxPhze/view`

4. OAD : `https://drive.google.com/file/d/17MbfjR7gy2Cu1PeGcHv-i7qbsYA_ObYg/view`

4.4.2.3 L'ensemble de données G3D

L'ensemble de données G3D, présenté par Bloom et al. [BMA12], est divisé en sept groupes d'actions, notamment les combats, le golf, le tennis, le bowling, les FPS, la conduite et les actions diverses. G3D dispose d'annotations au niveau frame (début/fin) et des points d'action. Pour permettre une comparaison avec les approches précédentes, nous n'avons utilisé que la catégorie combat (5 classes) où chacune des 30 séquences contient ces 5 actions dans le même ordre. Un exemple de séquence est visible à cette adresse : lien [5].

Nous introduisons un ensemble de test étendu (désigné comme « réarrangé » dans nos résultats) dans lequel les séquences de test originales sont divisées en séquences plus courtes et certaines des sous-séquences sont mises en miroir, ce qui modifie les classes d'action. L'ordre des actions dans ce nouvel ensemble de test est donc plus varié et peut commencer par n'importe quel geste. Cela permet d'éviter que les systèmes apprennent l'ordre des gestes et soient complètement biaisés. L'ensemble de tests utilisé est disponible à l'adresse suivante : `http://www-intuidoc.irisa.fr/oad-datasets`.

4.4.2.4 L'ensemble de données MAD

MAD (Multi-Modal Action Detection) est un ensemble de données introduit par Huang et al. [Hua+14]. Il se compose de 40 longues séquences réalisées par 20 sujets (2 séquences par sujet) à l'aide de la Kinect, où chaque sujet effectue 35 actions dans chaque séquence, dans le même ordre. L'ensemble de données contient un large éventail de classes d'actions avec 35 gestes différents. Elle dispose des annotations au niveau frame (débuts/fins). Un exemple de séquence est visible à cette adresse : lien [6].

Comme pour G3D, nous introduisons un ensemble de test « réarrangé » en suivant la même stratégie, c'est-à-dire la division et la mise en miroir. L'ensemble de tests utilisé est également disponible.

4.4.2.5 L'ensemble de données Chalearn

L'ensemble de données Chalearn Gesture **2013** [Esc+13] (il semble qu'elle a été renommée « Montalbano V1 » pour ne plus la confondre avec la version 2014) est une collection à grande échelle de vidéos Kinect conçues pour l'analyse des actions humaines,

5. G3D : `https://drive.google.com/file/d/17MimwAkj7QmmlMEQbD0DbcLWgRrm-ubP/view`
6. MAD : `https://drive.google.com/file/d/17MHw9DfJGV3XUh2fZJiD9w037HnvXYHe/view`

avec un accent particulier sur le langage corporel. Elle comprend 27 sujets effectuant 20 actions différentes, dans un ordre variable au sein de chaque séquence. L'ensemble de données comprend 680 séquences non segmentées avec des annotations au niveau frame (débuts/fins). Il s'agit d'un ensemble de données difficile, car certaines classes de gestes sont très similaires. Un exemple de séquence est visible à cette adresse : lien [7].

4.4.2.6 L'ensemble de données PKU-MMD

Le jeu de données PKU-MMD [Liu+17] est un jeu de données à grande échelle pour la compréhension des actions humaines. Il contient 1076 séquences non segmentées de 51 catégories d'actions, réalisées par 66 sujets. Trois caméras avec des points de vue différents filment la scène. Les séquences durent environ 3 à 4 minutes et contiennent approximativement 20 actions. 8 des 51 actions sont effectuées par deux personnes de manière interactive [8], nous n'avons pas utilisé les séquences correspondantes dans nos expériences. Ce jeu dispose des annotations au niveau frame (débuts/fins). Un exemple de séquence est visible à cette adresse : lien [9].

4.4.3 Base de données 2D

Pour nous évaluer sur le geste 2D dans le contexte non segmenté, nous avons construit deux bases de données synthétiques à partir des bases de gestes 2D segmentés ILGDB [Ren+12] et MTGSetB [Che+15] détaillées en sections 3.4.3.1 et 3.4.3.2.

4.4.3.1 L'ensemble de données ILGDB_Untrimmed

Pour générer la nouvelle base ILGDB_Untrimmed, nous avons créé des séquences composées de 4 à 8 gestes choisis de manière aléatoire. Chaque séquence est conçue de manière à ce que le dernier point d'un geste soit le même que le premier point du geste suivant. En suivant la répartition originale entre les données d'entraînement et de test, nous utilisons 119 séquences pour l'apprentissage et 210 pour les tests. Ce jeu de données présente un défi particulier car les séquences ne comportent aucune pause, rendant ainsi difficile la détermination du début et de la fin des gestes. De plus, il y a peu d'exemples d'entraînement disponibles. Pour pallier ce manque de données, nous avons créé un ensemble

7. Chalearn : `https://drive.google.com/file/d/17MdA-vScAgRkfSRIauaqP2YUdLpNse23/view`
8. classes avec les identifiants 2,14,16,18,21,24,26,27
9. PKU-MMD : `https://drive.google.com/file/d/17MrGeUtnirF7ckyM1jnWk19VP3vYHrh6/view`

de données augmenté en utilisant différentes échelles de gestes (5 échelles différentes) et en réutilisant le même geste dans plusieurs séquences (chaque geste apparaît dans 5 séquences distinctes). En fin de compte, chaque geste d'entraînement est répété 25 fois dans les séquences, ce qui se traduit par un total de 2621 séquences d'entraînement. Un exemple de séquence générée est illustré dans la figure 1.4.

4.4.3.2 L'ensemble de données MTGSetB_Untrimmed

Pour la base MTGSetB_Untrimmed, nous avons construit des séquences de 4 à 8 gestes aléatoires de sorte à ne pas être en mesure de différencier trivialement un lever de crayon au sein d'un geste et un lever de crayon entre les gestes. Chaque geste a été recentré par rapport au geste précédent dans la séquence, éliminant ainsi toute possibilité de segmentation spatiale entre les gestes. Selon la répartition originale entre l'ensemble d'entraînement et de test, qui était basée sur les utilisateurs, nous avons obtenu 607 séquences de gestes pour l'entraînement et 672 pour le test. Nous avons également créé une version augmentée de l'ensemble de données en utilisant différentes échelles (3 échelles différentes) et en incluant chaque geste dans 2 séquences distinctes. Cela a abouti à un total de 3076 séquences d'entraînement.

4.4.4 Métriques

4.4.4.1 Latency-Aware, DAP et NTtoD

Nous utilisons dans ces expérimentations les métriques Latency-Aware Score, DAP et NTtoD décrites en section 2.2.3.

4.4.4.2 Métrique BOD (Bounded Online Detection) pour une évaluation de l'OAD au niveau instance

Afin d'évaluer au mieux notre système, nous avons développé une nouvelle métrique inspirée de BOffD utilisée dans [Li+16], dont les problèmes ont été décrits en section 2.2.3.3.

L'idée principale de cette métrique, appelée *« Bounded Online Detection (BOD) »*, est de n'autoriser qu'une seule détection par borne de vérité terrain, en conditionnant la détection à une certaine quantité de chevauchement entre la borne de vérité terrain et la borne de détection. Toute détection supplémentaire sera considérée comme un faux positif. L'algorithme permettant de calculer la métrique est l'algorithme 2.

Algorithm 2 Algorithme permettant de calculer la métrique BOD, avec la NTtoD

Inputs : Predictions bounds, Labels bounds (GT) ; *Parameters* : canCorrect, Δ.
Sort Predictions and Labels by starting bound.
for all pred **in** Predictions **do**
  $GT^* \leftarrow \underset{GT \in Labels}{\text{argmax}}\ IoU_{st}(pred, GT)$
  if flag(GT^*) = 0 **and** class(GT^*) = class($pred$)
  and $IoU_{st}(pred, GT^*) > \Delta$ **then**
    Add a True Positive ; flag(GT^*) $\leftarrow$ 1
    earliness $\leftarrow \frac{start(pred) - start(GT^*)}{end(GT^*) - start(GT^*) + 1}$
  else
    Add a False Positive
    if not canCorrect **then** flag(GT^*) $\leftarrow$ 1 **end if**
  end if
end for
Precision $\leftarrow \frac{TP}{TP+FP}$; Recall $\leftarrow \frac{TP}{length(Labels)}$; NTtoD $\leftarrow$ average(earliness)

Notez que pour calculer cette métrique, nous avons besoin des bornes des prédictions obtenues dans le contexte en ligne **sans requalification des frames passées**, c'est-à-dire que nous ne devons pas utiliser les prédictions futures pour estimer une borne de début et de fin. Dans le cas où la méthode prédit une frame ponctuelle, il suffira de créer une borne de fin à la frame suivante.

Nous avons conçu IoU_{st} [10] qui est une variante de la mesure *Intersection Over Union* (IoU) pour le contexte en ligne. Comme nous ne voulons pas pénaliser la prédiction tardive sur ce critère, nous calculons le chevauchement à partir de la borne de départ de la prédiction : $IoU_{st} = \frac{Intersection}{Union_{st}}$ où $Union_{st}$ est calculé à partir de la borne de départ de la prédiction, comme illustré dans la figure 4.10. Un IoU_{st} élevé caractérise une détection qui correspond bien aux bornes de la vérité terrain à partir de la borne de départ de la prédiction.

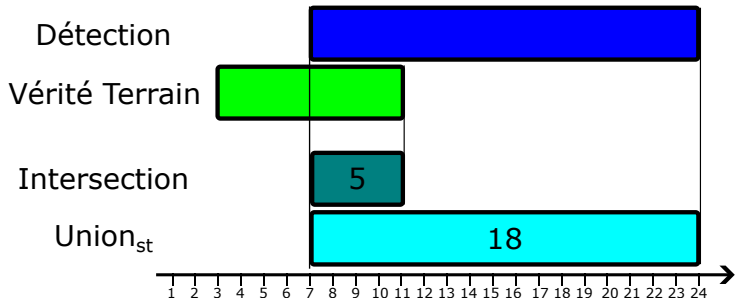


FIGURE 4.10 – Exemple de calcul de l'IoU_{st}. Ici $IoU_{st} = 5/18 \approx 0.28$.

10. $_{st}$ signifie *start* pour désigner le début de la détection, et ne doit pas être confondue avec l'*IoU* **S**patio-**T**emporelle qui est parfois utilisée dans les articles traitant de la tâche de localisation de geste.

Pour justifier davantage cette variante de l'IoU, prenons pour exemple la première détection de la figure 4.11. Celle-ci arrive plutôt tardivement (à plus de 50% de complétion), ce qui lui procure un $IoU = 47\%$, alors que son $IoU_{st} = 98\%$. Il est possible que du fait de l'ambiguïté entre les gestes, celui-ci ne soit pas reconnaissable avant 50% de sa complétion, le système ici a fait une détection au mieux possible, il n'est pas donc pas juste de le pénaliser sur le critère de la détection. De plus, si nous impliquons un deuxième geste qui lui n'est reconnaissable qu'à sa dernière frame (du fait de l'ambiguïté), il disposera d'un IoU systématiquement très faible ne dépassant jamais un certain seuil. L'IoU_{st} permet donc de ne pas abaisser le critère de détection en fonction de la tardiveté de sa détection, qui peut varier en fonction des gestes. Cette métrique est plus équitable entre les classes. De plus, elle n'évalue **que** la qualité de la détection, quel que soit le seuil Δ choisi. La précocité sera évaluée à part, via la métrique NTtoD.

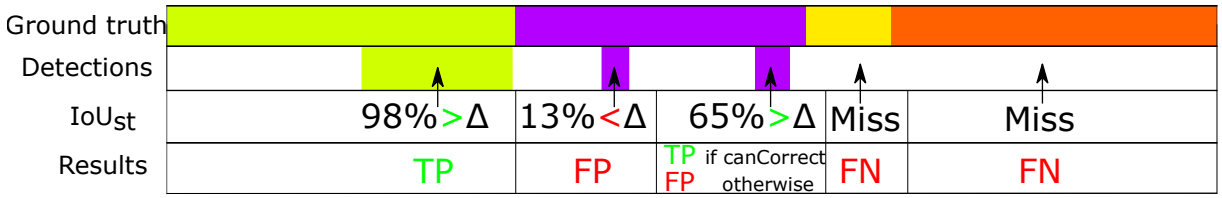


FIGURE 4.11 – Exemple de l'application de la métrique Online Detection (BOD) Metric avec Δ=50 %. Les FN sont pris en compte via le calcul du rappel.

La métrique a deux paramètres : *canCorrect* et Δ. *canCorrect* est un booléen, s'il est vrai, il permet au modèle de se corriger s'il a fait une erreur de détection sur un geste donné. Δ est une valeur comprise entre 0% et 100%, et correspond à la valeur minimale de IoU_{st} autorisée pour considérer le geste comme un vrai positif (TP). Pour les applications qui ne nécessitent pas de conserver la prédiction pendant le geste (juste un pic), Δ peut être fixé à 0%. Une valeur de 100% signifierait que la borne de fin détectée devrait correspondre exactement à la borne de fin de la vérité terrain pour être un TP.

La détection est considérée comme un TP si la vérité terrain avec le maximum de IoU_{st} n'a pas déjà été correctement détectée (ou même faussement détectée si $canCorrect = False$), s'il s'agit de la prédiction de classe correcte, et si le IoU_{st} est strictement supérieur à Δ. Dans le cas contraire, elle est considérée comme un *faux positif* (FP). Un exemple d'application de la métrique est présenté dans la figure 4.11.

Une fois que la *Precision* et le *Rappel* sont calculés, nous pouvons calculer le *FMeasure* micro-moyen global final en utilisant le calcul traditionnel vu en section 2.2.2.1 : $FMeasure = \frac{2*Recall*Precision}{Precision+Recall}$. À chaque fois dans nos expérimentations, la micro-moyenne

a été utilisée pour cette métrique. En revanche pour calculer le score final sur plusieurs folds, c'est la macro-moyenne qui est utilisée.

Dans notre contexte d'interaction gestuelle, l'utilisation de $\Delta = 0$ est le plus pertinent. De plus, nous avons choisi de mettre *canCorrect* à *False* afin d'avoir le contexte le plus difficile. Un exemple d'application de BOD avec ces paramètres est disponible dans la figure 4.12.

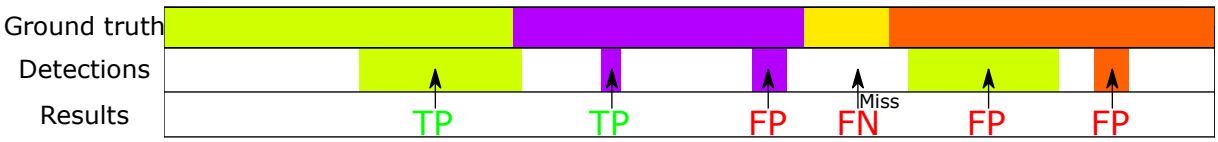


FIGURE 4.12 – Application de la métrique Bounded Online Detection (BOD) avec Δ=0 et $canCorrect = False$ sur une séquence de 4 actions. Les détections manquées sont prises en compte dans le calcul du rappel. Le Fscore est ensuite calculé à partir du rappel et de la précision. Le premier faux positif (FP) est considéré comme incorrect car le geste a déjà été correctement classé. Le dernier faux positif est incorrect parce que nous avons mis *canCorrect* à faux, le système ne peut pas corriger sa première erreur de classification.

4.4.5 Études par ablation

Avant d'effectuer une comparaison avec les approches de l'état de l'art, nous réalisons d'abord une étude par ablation pour évaluer les performances des différents composants de notre méthode. Nous évaluons ici notre stratégie de représentation présentée en section 4.2.1, puis l'efficacité de notre fonction de coût basée sur la CTC (présentée en section 4.2.3 et section 4.2.4) en comparant avec des variantes.

4.4.5.1 Efficacité de la stratégie de représentation E-SIM

Pour démontrer l'efficacité de la stratégie de représentation E-SIM, nous avons mené des expériences sur l'ensemble de données Chalearn. Nous avons d'abord évalué notre représentation par rapport à celle introduite par Duan et al. [Dua+22], qui transforme chaque frame en un ensemble de cartes thermiques temporelles. Le tableau 4.2 montre que notre approche traite les séquences beaucoup plus rapidement que la méthode des cartes thermiques temporelles. Cela s'explique par le fait que notre stratégie de regroupement (façon de construire les chunks) réduit le nombre d'images à traiter, en plus d'être indépendante de la vitesse d'exécution. Pour un ensemble de données comportant en moyenne 1 370 images par séquence, notre stratégie de regroupement produit environ 400 images

lorsque la quantité de déplacement θ est fixée à 3. En phase d'exécution, cela signifie qu'aucune nouvelle image n'est créée ou traitée tant qu'une quantité suffisante de déplacement ne s'est pas accumulée. Cette approche évite la surcharge du système et le rend bien adapté aux applications du monde réel. De plus, notre approche définit les valeurs d'intensité localement autour des points clés des cartes thermiques du squelette (à une distance maximale d du point clé), au lieu d'itérer sur l'ensemble de la carte thermique, ce qui réduit considérablement le temps de traitement. Le temps d'inférence passe ainsi de 115 ms par chunk à 25-53 ms, en fonction de la valeur choisie pour d.

TABLE 4.2 – Résultats des différentes variantes de représentation, avec notre DOLT-C3D + $\text{CTC}_{SG,\Psi=0.1}$. Base Chalearn, BOD FScore $\Delta = 0, canCorrect = False$. VF est le flux en vue frontale, VL est en vue latérale. E-TM est la représentation sans la stratégie indépendante de la vitesse, elle est équivalente à la représentation utilisée par Duan et al. [Dua+22]. Carte complète signifie que d n'est pas limité. "Traitement/seq." est le temps de traitement moyen (s) pour une séquence. "Taille Sortie" est le nombre moyen d'images de sortie par séquence. TI est le temps d'inférence (ms) par chunk (pour E-SIM) ou par image (pour E-TM).

Représentation	**FScore ↑**	**NTtoD ↓**	**Traitement /seq. (s)**	**Taille Sortie**	**TI (ms)**
E-SIM $d = 2$, VF Seulement	77.8 ± 0.8	41.3 ± 0.7	**7**	**400**	**17.5**
E-SIM $d = 2$, VL seulement	78.7 ± 0.5	41.2 ± 0.9	**7**	**400**	**17.5**
E-SIM $d = 1$	80.7 ± 0.4	40.0 ± 0.8	10	**400**	25.0
E-SIM $d = 2$	81.9 ± 0.4	**39.6** ± 1.2	15	**400**	37.5
E-SIM $d = 3$	**82.4** ± 0.6	**39.5** ± 0.8	21	**400**	52.5
E-SIM Carte complète	82.2 ± 0.6	39.7 ± 0.8	46	**400**	115.0
E-TM Carte complète [Dua+22]	**84.2** ± 0.6	**37.6** ± 0.6	158	1370	115.3

Nous avons également expérimenté le fait de ne conserver qu'une seule projection dans notre représentation, soit l'axe X et Y (Vue Frontale, VF), soit l'axe Z et Y (Vue Latérale, VL). Bien que le temps de traitement ait été réduit de moitié, les performances en termes de Fscore et de précocité sont réduites.

Bien que les cartes thermiques temporelles (E-TM, représentation de [Dua+22]) soient les plus performantes, une exécution à près de 9 images par seconde avec un grand nombre d'image à traiter (3.5 fois plus que E-SIM), n'est pas adaptée aux applications en temps réel. Par conséquent, la stratégie par chunks avec $d = 2$ ou $d = 3$ est un bon compromis entre les performances et la vitesse, dans les prochaines expérimentations nous utiliserons E-SIM avec $d = 2$.

4.4.5.2 Impact du CTC guidé et du *Label Prior* pondéré

Dans cette étude, nous avons analysé l'impact de nos stratégies d'élagage et du *Label Prior* pondéré sur les performances de notre système. Pour réaliser cette analyse, nous nous appuyons principalement sur la base de gestes 3D Chalearn, mais aussi sur 6 autres bases de données, y compris les deux bases de gestes 2D. Tous les Fscores mentionnés dans cette section correspondent à la métrique BOD, avec $\Delta = 0$ et $canCorrect = False$.

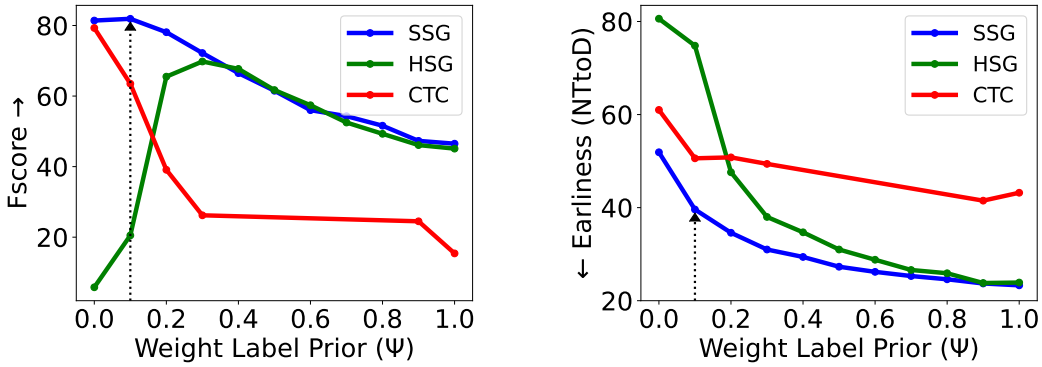
CTC guidé par la segmentation comparé au CTC Classique. Tout d'abord, l'élagage guidé par la segmentation donne des résultats très intéressants par rapport au CTC classique, en particulier pour la version SSG. Le tableau 4.3 montre que même sans aucun *Label Prior* (lorsque $\Psi = 0$), le système appris avec l'élagage SSG produit des détections beaucoup plus précoces (-9%) que le système appris avec le CTC classique, en plus d'un petit gain dans le Fscore (+2%). Ces performances confirment que l'utilisation de la connaissance de la segmentation pendant l'apprentissage est une stratégie efficace pour améliorer la performance du système, pour la précocité, mais aussi pour le Fscore. En revanche, sans label prior la version HSG ne permet pas d'obtenir des résultats satisfaisants (de l'ordre de 15% de Fscore), de plus nous avons observé que le réseau a de grandes difficultés à converger avec cette version. Laisser une souplesse au CTC semble donc préférable dans ces conditions. Nous reparlerons de HSG combiné au label prior ensuite.

TABLE 4.3 – Comparaison du système entraîné avec différentes fonctions de coût. Le BOD Fscore avec $\Delta = 0, canCorrect = False$ est montré avec NTtoD (précocité). Ensemble de données Chalearn.

Loss	**FScore ↑**	**NTToD ↓**
E-SIM + DOLT-C3D + CTC classique	79.3 ± 2.2	61.0 ± 4.1
E-SIM + DOLT-C3D + $CTC_{SSG,\Psi=0}$	81.4 ± 0.7	51.9 ± 0.7
E-SIM + DOLT-C3D + $CTC_{SSG,\Psi=0.1}$	**81.9 ± 0.4**	**39.6 ± 1.2**

CTC guidé couplé au label prior faiblement pondéré ($\Psi = 0.1$). Ensuite, concernant le CTC guidé avec la version d'élagage SSG couplé au Label Prior, on observe dans les figures 4.13 (pour Chalearn) et 4.14 (pour les autres bases) systématiquement un gain significatif de Fscore entre $\Psi = 0$ et $\Psi = 0.1$ (+15% pour ILGDB, +7% pour MTG-SetB, +20% pour PKU-MMD$_{cv}$, +11% pour OAD, +0.5% pour Chalearn, +2% pour

MAD, +9% pour G3D). Ce gain montre que, de manière générale, le label prior pondéré faiblement ($\Psi = 0.1$) améliore les résultats du CTC guidé.



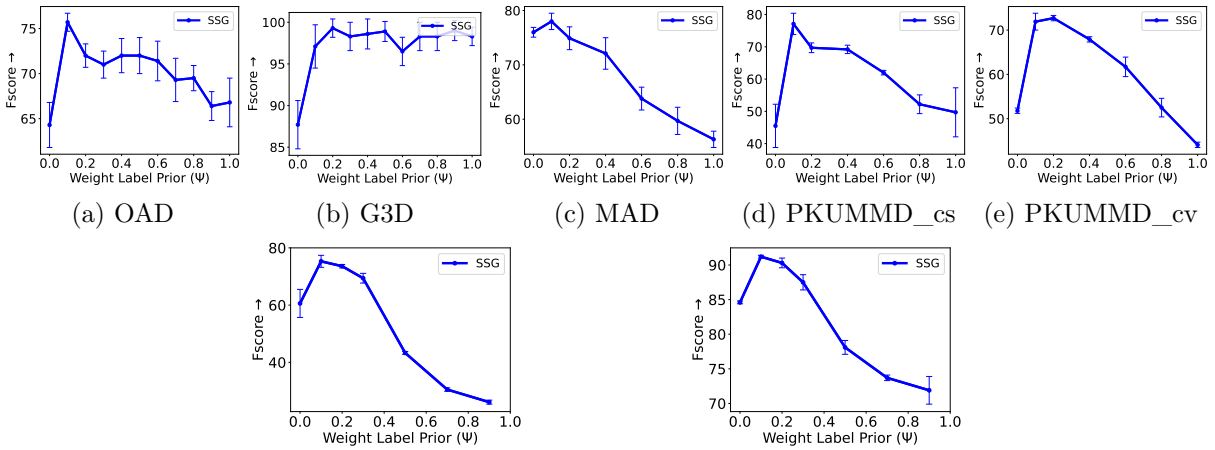
(a) Fscore vs. Ψ

(b) Précocité vs. Ψ

FIGURE 4.13 – Évolution : a) du Fscore de BOD ($\Delta = 0$) et b) de la précocité en fonction du poids du label prior Ψ. L'augmentation du poids conduit à un système qui détecte les gestes plus tôt. Dans le même temps, le Fscore se dégrade avec le poids, mais certains points sont plus optimaux que d'autres. Par exemple, pour la version SSG, le meilleur Fscore est obtenu lorsque $\Psi = 0.1$ avec une performance de précocité intéressante. Expériences réalisées sur l'ensemble de données Chalearn.

Cela peut s'expliquer notamment par le faible rappel lorsque $\Psi = 0$. Sans le label prior, l'impact du blank dans l'apprentissage est tellement prépondérant que l'on peut n'aboutir à aucune prédiction de gestes dans certains cas. Par exemple pour PKU-MMD$_{cv}$, le rappel passe de 35.5% avec $\Psi = 0$ à 58.9% avec $\Psi = 0.1$. En plus de l'augmentation du Fscore, on observe un gain significatif en termes de précocité (de 51,9 % à 39,6 % pour Chalearn) pour les bases de gestes 3D, cela est visible dans la figure 4.15. Ce résultat couplé à l'analyse des résultats qualitatifs (figure 4.16b et annexe A), nous permet d'en déduire que ce point est particulier. En effet, cela montre que des détections peuvent être **plus précoces sans pour autant diminuer la qualité de la reconnaissance**. Ce résultat est très important car il signifie que les détections se produisent plus proches des « points d'action » théoriques, ces points étant l'instant théorique où les gestes se distinguent les uns des autres et qui dépend de chaque classe de geste.

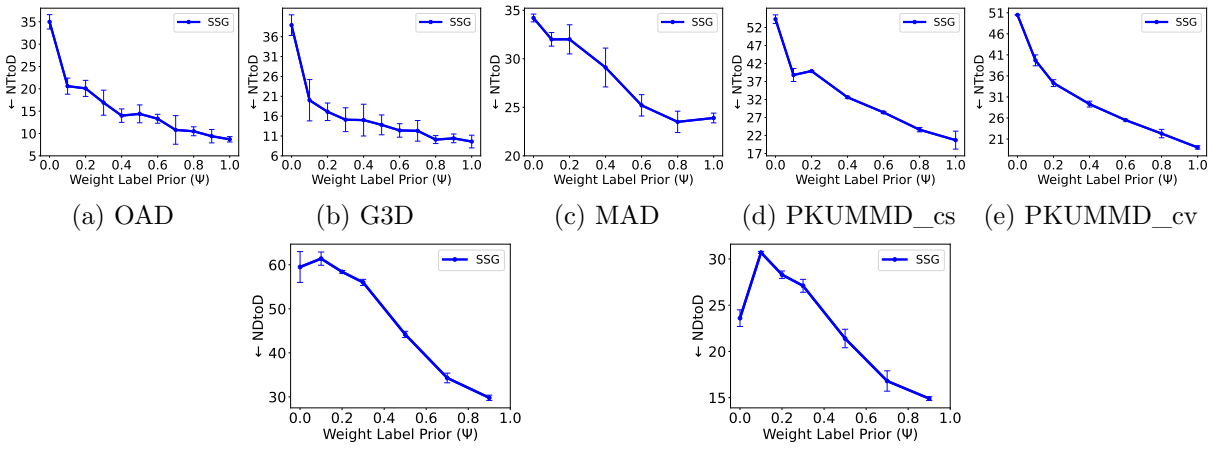
Optimisation du compromis entre précocité et précision ($\Psi \geq 0.1$). Nous pouvons observer dans les figures 4.14 et 4.15 que la variation du poids Ψ du *label prior* nous permet ainsi d'ajuster efficacement l'équilibre entre le Fscore et la précocité (pour SSG avec $\Psi \geq 0.1$).



(a) OAD (b) G3D (c) MAD (d) PKUMMD_cs (e) PKUMMD_cv

(f) ILGDB_U. (2D) (g) MTGSetB_U. (2D)

FIGURE 4.14 – Fscore (BOD $\Delta = 0, canCorrect = False$) en fonction de la pondération du label prior (Ψ) sur les bases de données de gestes 3D : a) OAD b) G3D c) MAD d) PKUMMD_cs e) PKUMMD_cv et les bases de gestes 2D : f) ILGDB_Untrimmed g) MTGSetB_Untrimmed.



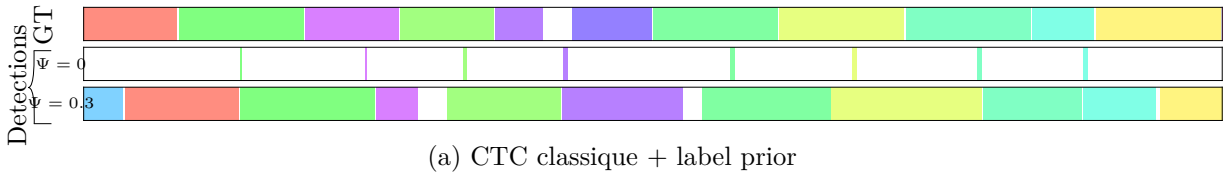
(a) OAD (b) G3D (c) MAD (d) PKUMMD_cs (e) PKUMMD_cv

(f) ILGDB_U. (2D) (g) MTGSetB_U. (2D)

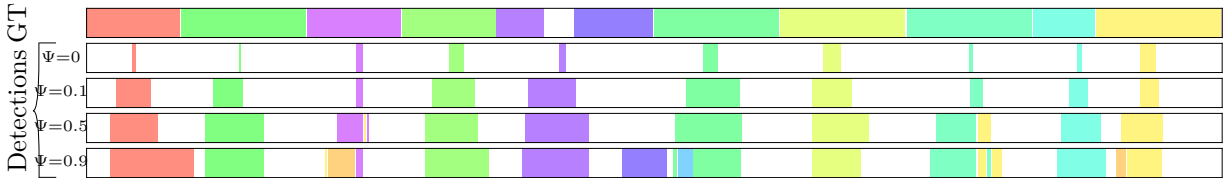
FIGURE 4.15 – Précocité en fonction de la pondération du label prior (Ψ) sur les bases de données de gestes 3D : a) OAD b) G3D c) MAD d) PKUMMD_cs e) PKUMMD_cv et bases de gestes 2D : f) ILGDB_Untrimmed g) MTGSetB_Untrimmed.

Nous avons constaté qu'un poids plus élevé permet au système de produire des détections plus précoces, mais avec un Fscore généralement plus faible. Ce constat est le même pour toutes les bases de données de gestes 3D et 2D. Remarquons que le Fscore de G3D ne diminue pas vraiment étant donné que le score est très proche de 100%, avec de grands écarts-types.

L'effet du label prior permettant de calibrer le ratio précocité/précision est lié à la réduction des prédictions de classe *blank*, comme le montre l'exemple qualitatif de la figure 4.16b.



(a) CTC classique + label prior



(b) CTC guidé (SSG) + label prior

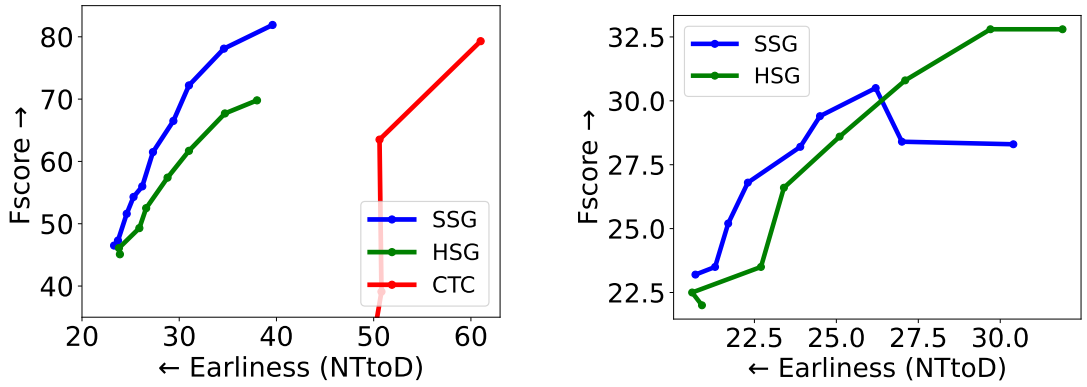
FIGURE 4.16 – Exemple de détections de gestes avec différentes valeurs du poids du label prior (Ψ) pour a) le CTC classique et b) le CTC guidé (SSG). La première ligne (GT) est l'annotation (vérité terrain), chaque couleur désignant une étiquette de classe. Exemple tiré de la base de test de Chalearn (Sample4). Lorsque le poids (Ψ) du label prior est plus élevé, moins de prédictions de blanks sont faites, ce qui permet une détection plus rapide des gestes pour le CTC guidé. Cependant, une valeur Ψ plus élevée augmente le risque d'erreurs. Pour le CTC classique, la détection n'arrive pas forcément plus tôt. Davantage d'exemples sont visibles en annexe A. Exemple visualisable en vidéo ici : ***lien*** [11]

La présence du blank permet au système d'attendre que le geste devienne plus clairement reconnaissable. Prédire moins de blanks dans les premières étapes d'un geste comporte un risque plus élevé. Lorsque $\Psi = 0.9$, le système commet plus d'erreurs que lorsque les valeurs de Ψ sont plus faibles. En ce qui concerne le CTC classique, lorsqu'il est combiné avec le *label prior* moins de blanks sont prédits, mais les détections sont décalées dans le temps, ce qui ne conduit pas à des détections plus précoces comme le montre la figure 4.16a. C'est notamment dû au fait que l'apprentissage des chemins n'est pas contraint par les bornes de l'action dans cette version. De plus, lorsque les actions sont temporellement proches, il reste peu d'espace pour la détection de l'action suivante. Par conséquent, la performance Fscore est faible et des difficultés de convergence apparaissent. On peut remarquer en figure 4.16a que les gestes prédits avec $\Psi = 0$ ne sont pas vraiment localisés à la fin des gestes annotés (vérité terrain, ligne GT) comme nous pourrions nous y attendre avec le CTC classique. Cela s'explique par le fait que les annotations sur cet

11. `https://drive.google.com/file/d/17LpohgT2CfOmhUver6C5fpezcpr4H3g7/view?usp=sharing`

exemple de la base Chalearn sont très « larges », elles englobent le geste avec des marges, surtout sur la fin du geste.

En ajustant le compromis entre la précocité et la précision à l'aide du poids Ψ du label prior, il est donc possible d'obtenir des systèmes avec différents niveaux d'équilibres entre précision et précocité. Dans la figure 4.17, chaque point correspond à un état du système. Ces systèmes disposent de performances variables de manière à privilégier la précision ou la précocité, ou bien à faire un compromis entre les deux. Le choix du poids Ψ permettra de répondre aux exigences spécifiques d'une application donnée.



(a) BOD Fscore with $\Delta = 0$

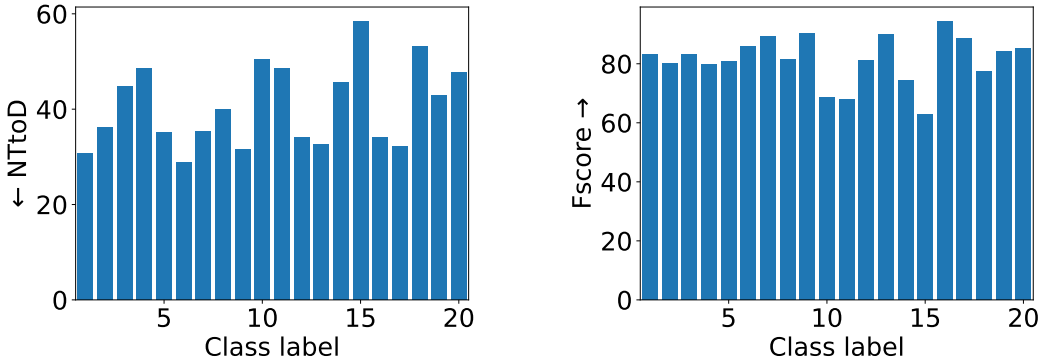
(b) BOD Fscore with $\Delta = 0.8$

FIGURE 4.17 – L'utilisation de différentes valeurs de poids pour le label prior permet de créer des systèmes avec différents degrés de précision et de précocité. (a) Nos méthodes d'élagage SSG sont plus performantes que l'élagage HSG et le CTC classique pour notre tâche de détection en ligne. (b) HSG montre des résultats intéressants lorsqu'une prédiction précise des bornes de fin est requise (BOD avec $\Delta = 0.8$). Expériences réalisées sur l'ensemble de données Chalearn avec $\text{CTC}_{SSG,\Psi\geq 0.1}$ et $\text{CTC}_{HSG,\Psi\geq 0.3}$.

Nous avons également évalué les performances de l'élagage HSG, nous avons trouvé qu'il avait généralement des résultats inférieurs à SSG avec le Fscore BOD ($\Delta = 0$) pour notre tâche, comme le montre la figure 4.13. Avec un Ψ faible ($\leq 0,2$), les résultats sont médiocres par rapport à la version SSG. Avec un Ψ plus élevé, les deux versions ont des performances similaires, mais l'élagage HSG a généralement moins de détections précoces pour un Fscore similaire. Cependant, pour une application qui souhaite détecter avec précision la fin du geste (score de BOD avec Δ élevé), elle peut être plus adaptée que l'élagage SSG dans certaines configurations, comme le montre la figure 4.17b.

Il est intéressant de constater que la précocité n'est pas la même en fonction des différentes classes, comme le montre la figure 4.18. De grandes variations sont observables (de 58.5% à 28.9%), ce qui montre que le système est capable de s'adapter en fonction de la

classe d'action (certaines classes sont effectivement reconnaissables plus tôt que d'autres).



(a) NTtoD par classe

(b) BOD Fscore ($\Delta = 0$) per class

FIGURE 4.18 – Résultats par classe sur l'ensemble de données de Chalearn avec $\text{CTC}_{SSG,\Psi=0.1}$. (a) La valeur de la précocité dépend fortement de la classe, allant de 28.9% à 58.5%. (b) Les Fscores sont plutôt stables autour de ≈80%.

Un résultat qualitatif à un niveau plus fin qu'illustré précédemment est visible dans la figure 4.19.

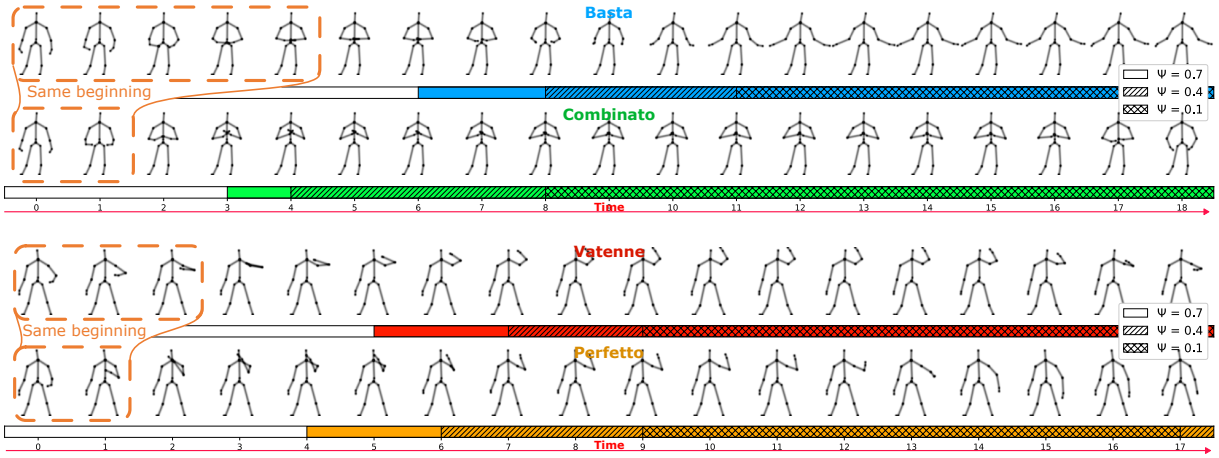


FIGURE 4.19 – Résultats qualitatifs des prédictions sur des séquences démontrant l'impact des différentes valeurs de Ψ sur le timing des prédictions. Les gestes illustrés de haut en bas proviennent de l'ensemble de données de Chalearn : *Basta, Combinato, Vatenne* et *Perfetto*. Trois valeurs différentes de Ψ (0,1, 0,4, 0,7) sont représentées. Avec $\Psi = 0, 1$, nous observons que les prédictions sont faites lorsque les gestes sont généralement clairement identifiables, ce qui implique une décision plus tardive. En revanche, des valeurs Ψ plus élevées introduisent en moyenne un niveau de risque plus important, les prédictions étant effectuées à des stades plus précoces des gestes.

Comparaison avec une fonction de coût par frame. Pour montrer l'efficacité de notre fonction de coût, nous la comparons à la fonction de coût classique par frame (entropie croisée) utilisée dans la plupart des méthodes de l'état de l'art. Comme le montre le tableau 4.4, l'utilisation de la fonction de coût par frame est très peu performante pour la tâche exigeante que l'on s'est fixé, évaluée avec la métrique BOD. Cette fonction de coût ne tient compte d'aucune forme de rejet puisqu'elle classe chaque frame sans aucun objectif de cohérence temporelle au niveau instance.

TABLE 4.4 – Comparaison du système entraîné avec une fonction de coût classique par frame (entropie croisée) et avec notre fonction de coût basée sur le CTC. BOD Fscore ($\Delta = 0$) sur la base Chalearn.

Loss	**FScore** ↑	**NTToD** ↓
E-SIM + DOLT-C3D + Per frame loss	36.7 ± 2.5	**22.7** ± 0.3
E-SIM + DOLT-C3D + $CTC_{SSG,\Psi=0.1}$	**81.9 ± 0.4**	**39.6 ± 1.2**

4.4.6 Comparaison avec l'état de l'art

Nous avons comparé notre méthode avec celles de l'état de l'art utilisant la modalité squelette qui se concentrent sur la détection d'actions au niveau de l'instance dans un contexte en ligne. Pour cela, en cohérence avec l'étude d'ablation, nous avons utilisé : E-SIM avec $d = 2$ et le CTC_{SSG}, la valeur de Ψ sera précisée pour chaque expérimentation.

4.4.6.1 Expérimentations : évaluation de la détection de gestes 3D en ligne sur MSRC-12 et G3D

Tout d'abord, nous comparons notre méthode aux approches précédentes en utilisant le Latency-Aware Score. Cette mesure permet d'évaluer la détection en ligne, mais sans aucune mesure spécifique sur la précocité de la détection. Cependant, comme elle utilise le point d'action comme référence pour calculer la mesure, celle-ci a du sens dans notre contexte. Deux petits ensembles de données sont évalués avec cette métrique, G3D (table 4.5) et MSRC-12 (table 4.6).

En ce qui concerne G3D (table 4.5), nous avons obtenu des résultats au niveau de l'état de l'art [BMA14 ; Bou+18a], proche de 100%. L'évaluation a porté sur les catégories « fighting » suivant un protocole leave-subject-out sur 10 folds. Chacun des 10 jeux de test est composé de 3 séquences de 5 actions réalisées par un utilisateur.

TABLE 4.5 – Latency aware Fscore ($\Delta = 10$ frames) sur la base G3D. Notre méthode obtient des résultats similaires à ceux des dernières approches.

Méthode	**Année**	**FScore**
DFS [BAM13], Random Forest	2013	91.9
RTMS [Sha+15], SVM	2015	92.1
RF [BKK17], Random Forest	2017	94.8
CAM [BMA14], Manifolds + DTW	2014	97.8
CuDi3D [Bou+18a], SVMs	2018	**98.9**
E-SIM + DOLT-C3D + CTC$_{SG,\Psi=0.2}$	2023	98.3 ± 2.8

En ce qui concerne MSRC-12 (table 4.6), la différence avec les approches précédentes est significative. Nous obtenons des résultats supérieurs dans toutes les catégories, ce qui se traduit par un gain de 7,4 % dans le score moyen pour toutes les modalités d'instruction par rapport à l'état de l'art précédent [MHT16 ; WYY19 ; Bou+18a] dans cette expérience. Le protocole utilisé est une évaluation croisée de 10 sujets. Chaque modalité d'instruction est entraînée et testée séparément. Un ensemble de test minimum est créé pour chaque *fold* en prenant des sujets aléatoires parmi les 30 personnes jusqu'à ce que toutes les classes de gestes soient présentes dans l'ensemble de test.

TABLE 4.6 – MSRC-12, Latency-Aware Fscore. Les différentes modalités d'instruction sont : V-Video, I-Images, T-Text.

Methode	ELS [MHT16]	IELS [WYY19]	CuDi3D [Bou+18a]	E-SIM + DOLT-C3D + CTC$_{SSG,\Psi=0.2}$
	SVMs	SVMs	SVMs	
Année	2016	2019	2018	2023
V	72.6	79.5	84.5 ± 8.0	**90.2** ± 4.9
I	67.0	69.2	73.1 ± 12.0	**84.4** ± 5.7
T	62.2	63.8	67.3 ± 10.0	**77.2** ± 9.8
V+T	79.0	82.3	85.4 ± 7.0	**90.1** ± 3.5
I+T	71.1	73.8	75.3 ± 9.0	**80.6** ± 9.9
Moyenne	70.4	74.1	77.1	**84.5**

4.4.6.2 Expérimentations : évaluation de détection précoce de gestes 3D

Peu de travaux se sont intéressés à l'aspect précocité dans un contexte de détection en ligne au **niveau instance**. En effet, les approches de comparaison dans la section précé-

dente se sont évaluées avec des métriques qui ne permettaient pas d'évaluer correctement la précocité des systèmes. Néanmoins, l'approche E-CuDi3D [Bou+18b] s'est évaluée avec une métrique qui permet d'évaluer la précocité dans ce contexte, nous allons donc nous comparer directement à son score. Afin de permettre des comparaisons supplémentaires avec des métriques qui nous semble plus pertinentes (BOD et NTtoD), nous avons implémenté la dernière approche qui adresse l'OAD au niveau instance, SM-MT [Liu+19].

Concernant le geste 2D, nous comparons cette approche à notre ancienne approche que nous avions publiée à ICPR [MAK22a] mais que nous n'avons pas détaillé dans ce manuscrit. Cette approche était basée sur une combinaison de CTC et SelectiveNet.

Détection au point d'action sur MSRC6-Iconic-C4 Dans cette expérimentation, nous comparons notre méthode avec l'évaluation faite par Boulahia et al. avec la méthode ECuDi3D [Bou+18b]. Le résultat est présenté dans la figure 4.20. Nous pouvons voir que notre méthode a une bien meilleure performance. Dans les premiers instants, ECuDi3D a un Fscore un peu plus élevé, mais notre système le rattrape rapidement. Notre méthode atteint un score final de 92%, surpassant le score de l'ECuDi de 78% (métrique DAP). Notre système montre une bonne capacité à classer les gestes au bon moment. Afin d'avoir

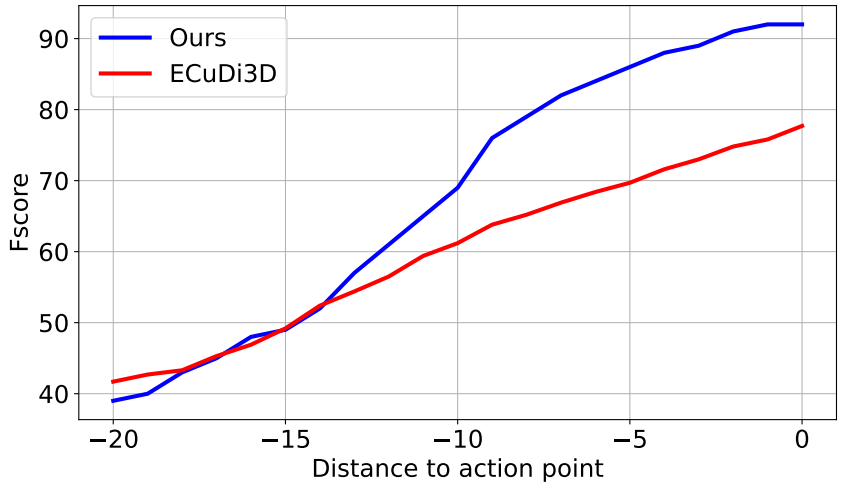


FIGURE 4.20 – Évaluation de la détection de la précocité sur le jeu de données MSRC6-Iconic-C4. La métrique utilisée est le score de détection au point d'action (DAP) utilisé dans [Bou+18b]. Notre système, E-SIM + DOLT-C3D + $\text{CTC}_{SSG,\Psi=0.4}$, montre une meilleure performance globale que les travaux précédents.

un niveau de risque similaire aux points les plus précoces de ECuDi3D, nous avons fixé le paramètre $\Psi = 0,4$ dans cette expérience.

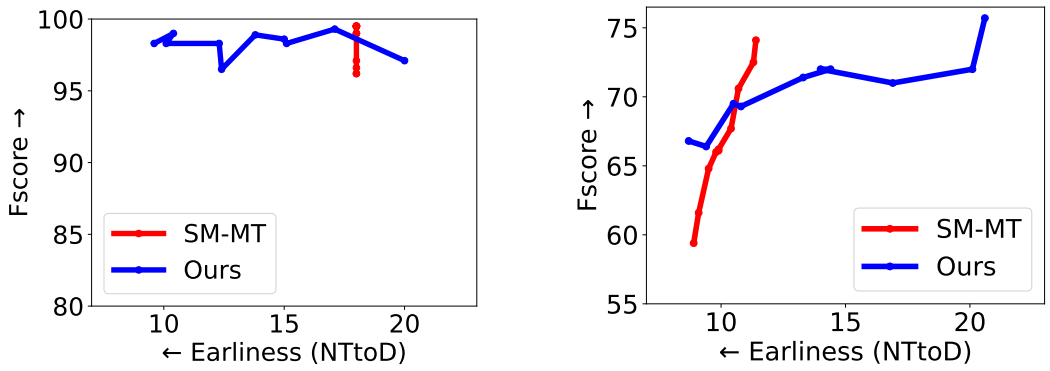
Comme dans [Bou+18b], les expériences ont été réalisées sur le sous-ensemble MSRC6-Iconic-C4. Le protocole est un 10-fold cross subject, avec la même contrainte que celle mentionnée précédemment.

Détection et précocité Nous comparons également notre méthode à une approche récente de l'état de l'art pour les tâches OAD basées sur des instances.

L'approche Skeleton-Modality Multi-Task (SM-MT) a été introduite par Li et al. [Liu+19] dans leur extension de l'approche JCR-RNN [Li+16]. Le réseau basé sur les LSTM a appris conjointement deux tâches : la classification et la régression. La régression est dédiée à la prédiction de la confiance des bornes de début et de fin de l'action (supervisée avec une courbe gaussienne centrée sur les frames de début/fin) en plus de la classe, elle permet d'extraire les gestes au niveau de l'instance. Cependant, essayer de prédire la borne de début avec la bonne classe est très difficile dans le contexte en ligne puisque plusieurs actions peuvent avoir un début similaire. Au contraire, avec notre conception du graphe CTC, nous permettons au système de commencer à prédire la classe lorsqu'il le juge approprié. Pour permettre au système SM-MT d'agir plus ou moins tôt, nous avons modifié les seuils de confiance de début et de fin pour détecter les gestes.

Nous allons évaluer notre approche en comparaison à SM-MT sur 5 bases de données : G3D, OAD, Chalearn, MAD et PKU-MMD.

Tout d'abord, nous avons évalué les performances de notre approche par rapport à l'approche SM-MT sur **deux petits ensembles de données, G3D et OAD,** avec la métrique BOD (*canCorrect*=False, $\Delta = 0$) couplée à la NTtoD. L'ensemble de données G3D comprend 5 classes avec une faible similarité interclasses au début des gestes (coup de poing gauche et droit, coup de pied gauche et droit, défense). La base OAD comporte 10 classes qui sont également identifiables très tôt. Dans ce contexte, notre approche a démontré des performances similaires à celles de SM-MT, comme l'illustre la figure 4.21. Sur **G3D**, nous pouvons atteindre des points plus précoces grâce au *label prior* pondéré. Cependant, la précision reste la même : prédire plus tard n'est pas clairement associé à un meilleur score, qui présente une grande variance (la variance moyenne est d'environ 2,5 %). Comme les gestes peuvent être détectés très tôt en raison de la nature des actions, le fait de disposer de plus ou moins d'informations pour prendre une décision n'a pas d'impact sur le score final, qui est davantage lié à la capacité de reconnaissance globale. Notre système atteint des zones un peu plus précoce que SM-MT (≈10% de précocité contre 17% pour SM-MT), pour un Fscore très proche (autour de 99%). Sur l'ensemble de

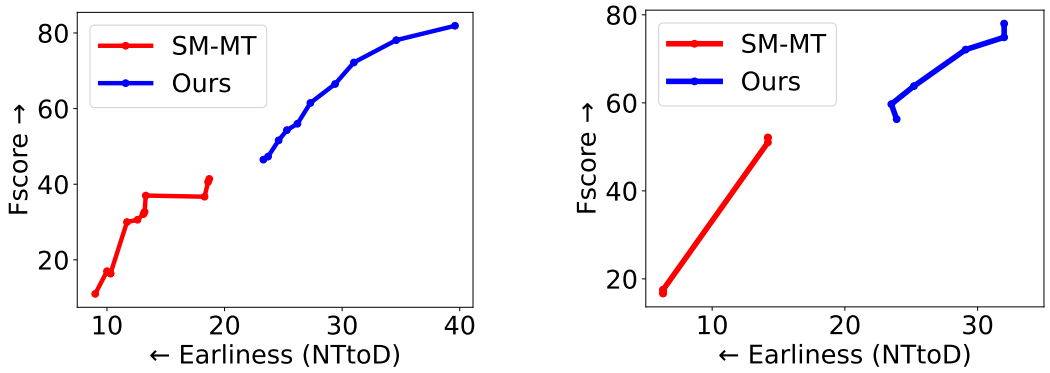


(a) G3D (b) OAD

FIGURE 4.21 – (a) Comparaison des performances sur l'ensemble de données G3D : Notre approche (E-SIM + DOLT-C3D + $\text{CTC}_{SSG,\Psi}$) présente des performances globalement similaires à celles de l'approche SM-MT, avec la capacité d'atteindre des points plus précoces tout en conservant une précision proche. (b) Comparaison des performances sur l'ensemble de données OAD : SM-MT concurrence notre approche sur les points antérieurs, mais nous obtenons le meilleur Fscore maximum avec une précocité raisonnable. Fscore est le Fscore BOD ($\Delta = 0$, $canCorrect = False$).

données **OAD**, le comportement est légèrement différent. Nous pouvons observer l'impact du *label prior* pondéré puisque le Fscore diminue globalement avec la précocité. Au point du meilleur Fscore, notre approche obtient un Fscore de **75.7%** pour une précocité de 20.6% alors que SM-MT parvient à un Fscore de 74.1% pour une précocité de **11.4%**. À valeur de précocité égale, deux points sont intéressants, autour de 11% de précocité, SM-MT dispose d'un avantage de 4.6% de Fscore (69.5% vs 74.1%), mais autour de 9% notre approche affiche un gain de 7.4% (66.8% vs 59.4%). En résumé, sur cette base de donnée il est difficile de départager les deux systèmes.

Lorsque les différences entre les gestes sont moins évidentes au début, notre méthode surpasse SM-MT, comme nous pouvons le voir sur les jeux de données **Chalearn et MAD** sur la figure 4.22. Sur les deux ensembles de données, SM-MT atteint des zones où le système répond de manière très précoce, mais avec un faible Fscore ($\leq 60\%$). Notre méthode obtient des Fscore ($\approx$**82%** pour Chalearn, **78%** pour MAD) que SM-MT ne peut pas atteindre. La stratégie de détection de SM-MT est conçue pour donner la priorité aux détections précoces. Par exemple, face à un début similaire de deux gestes, SM-MT les classe a priori comme une classe donnée pendant la zone ambiguë. Pour ces deux bases, les gestes ne peuvent pas être différenciés dès leur commencement du fait de leur plus grand nombre (20 et 35 classes) et des débuts communs entre les gestes. Il faut donc un système

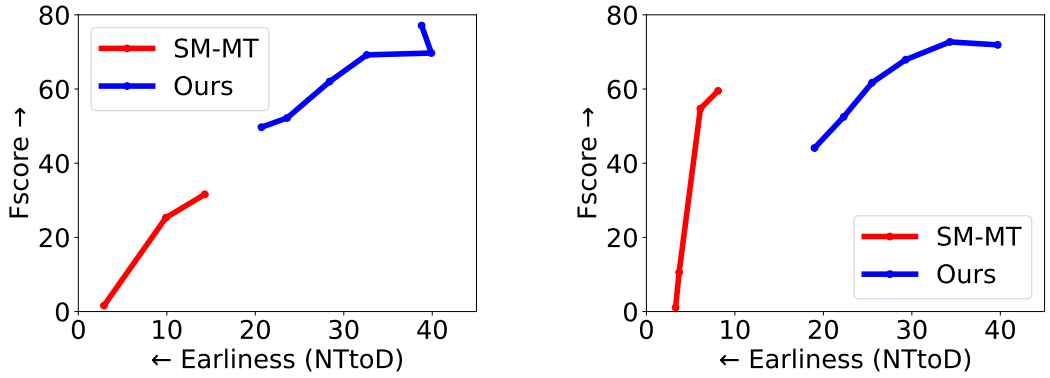


(a) Chalearn (b) MAD

FIGURE 4.22 – Comparaison sur les ensembles de données (a) Chalearn et (b) MAD. Notre méthode (E-SIM + DOLT-C3D + $\text{CTC}_{SSG,\Psi}$) surpasse SM-MT dans les cas où les différences de gestes sont moins évidentes, obtenant un Fscore plus élevé avec moins de précocité. Fscore est le Fscore BOD ($\Delta = 0$, $canCorrect = False$).

qui soit capable de ne pas décider dans certaines circonstances. De plus, il faut garder à l'esprit que la métrique NTtoD, qui exprime la précocité, ne prend en compte que les classifications correctes. Il faut donc toujours privilégier le score de détection (Fscore de BOD) lorsque l'écart est significatif. De manière générale, il est en effet peu utile d'avoir un système donnant des réponses de manière très précoce, mais qui n'est pas fiable.

Pour l'ensemble de données le plus large, **PKU-MMD** (43 classes considérées), notre méthode est également plus performante que SM-MT (figure 4.23). En ce qui concerne le Fscore, nous obtenons respectivement **77,1 %** et **72,7 %** pour les protocoles inter-sujets et inter-vues aux points les plus élevés, alors que SM-MT n'obtient que 31,6 % et 59,6 %, qui ne sont pas des valeurs suffisamment intéressantes pour permettre son utilisation dans un système interactif. La bonne performance de notre approche sur le protocole inter-vue, malgré l'utilisation de projections 2D du squelette dans notre représentation (VF et VL), peut être liée de l'utilisation de filtre de convolution dont les poids sont partagés dans les deux branches liées aux deux flux. Les filtres peuvent donc être utilisés pour extraire des caractéristiques communes, ce qui permet probablement d'être moins sensible au changement de point de vue. De plus, la variation du point de vue dans cette base de données est 45° (trois vues distinctes à -45°, 0 et 45°), ce qui ne modifie pas fondamentalement les projections.



(a) PKU-MMD Cross-Subject

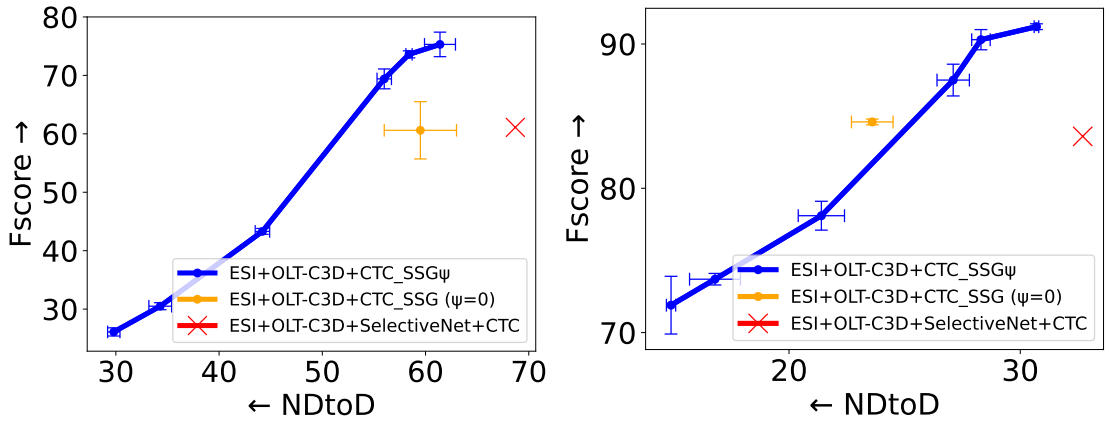
(b) PKU-MMD Cross-View

FIGURE 4.23 – Comparaison sur l'ensemble de données PKU-MMD (séquences d'une personne) (a) Protocole à sujets croisés (b) Protocole à vues croisées. Notre méthode (E-SIM + DOLT-C3D + $\text{CTC}_{SSG,\Psi}$) atteint les scores Fscore les plus élevés. Fscore est le Fscore BOD ($\Delta = 0$, $canCorrect = False$).

4.4.6.3 Expérimentations : évaluation de détection précoce de gestes 2D

En comparaison avec notre approche publiée dans ICPR [MAK22a], qui utilise une combinaison SelectiveNet + CTC (SelectiveNet pour adresser précocité, et le CTC classique pour la stabilité des décisions) notre nouvelle approche ($\text{CTC}_S SG + \Psi$) améliore encore très significativement les performances sur les deux bases de données, ILGDB_Untrimmed et MTGSetB_Untrimmed, comme le montre la figure 4.24. Nous obtenons ici à la fois un bien meilleur Fscore (**75,3 %** contre 61.1 % pour ILGDB, **91.2 %** contre 83.6 %) mais également une meilleure précocité (**61.4 %** contre 68.7 % pour ILGDB, **30.7 %** contre 32.7 % pour MTGSetB), pour $\Psi = 0.1$ (qui donne le meilleur Fscore et la moins bonne précocité).

La figure 4.25 montre quelques résultats qualitatifs. Pour ILGDB (figure 4.25a), trois gestes s'enchaînent sans lever de crayon. Si l'on se réfère à l'ensemble des gestes de cette base présentés en figure 3.5, le premier geste correspond à celui avec l'identifiant « 13 ». Il peut se confondre avec les gestes « 12 » et « 14 » durant les premiers instants, jusqu'au moment décisif (frame encadré en rouge). Selon les différents poids du label prior, le réseau ne réagit pas de la même manière. Pour $\Psi = 0.1$ et $\Psi = 0.2$ le système attend que le geste ait effectué le tournant à droite avant de prendre sa décision. Pour $\Psi = 0.3$ et $\Psi = 0.9$, il prend sa décision avant le tournant, et le confond avec le geste « 12 ». Sur les autres gestes, le comportement est similaire.



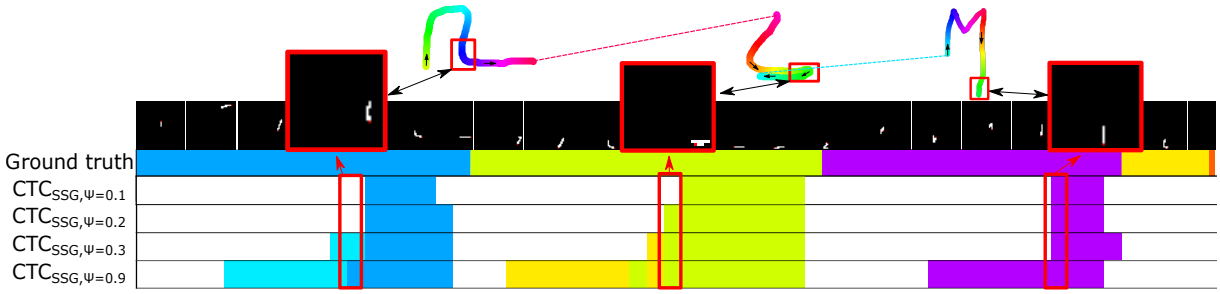
(a) ILGDB_Untrimmed (b) MTGSetB_Untrimmed

FIGURE 4.24 – Comparaison sur l'ensemble de données a) ILGDB_Untrimmed et b) MTGSetB_Untrimmed. Notre nouvelle méthode avec le $\text{CTC}_{SSG,\Psi}$ est bien plus intéressante que la méthode combinant CTC et SelectiveNet [MAK22a], tant sur l'aspect précocité que sur la qualité des détections. ESI (pour "Euclidean Speed-Independant") est la représentation utilisée dans [MAK22a]. Le Fscore est calculé avec la métrique BOD $\Delta = 0, canCorrect = False$. Les barres horizontales et verticales représentent l'écart type de la NDToD et du Fscore.

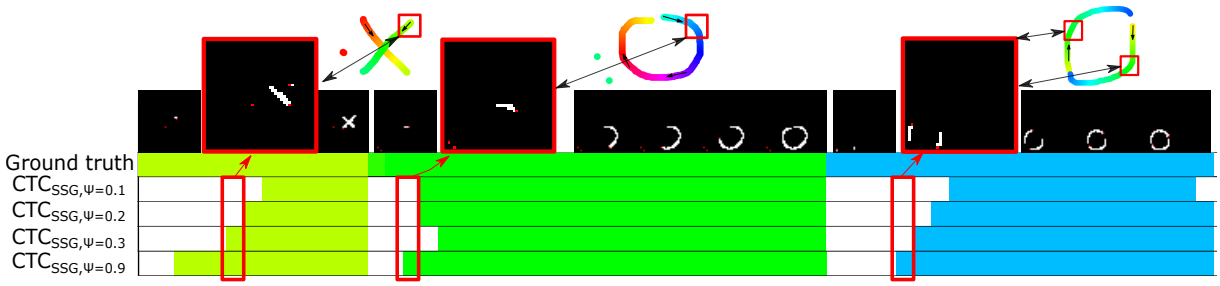
Sur MTGSetB, pour le premier geste, on constate que le système retarde la décision jusqu'au début du deuxième tracé du geste « X » pour $\Psi \leq 0.3$, afin de garantir qu'il ne soit pas confondu avec le geste « W », également présent dans le jeu de données. Pour $\Psi = 0.9$, il arrive à le prédire plus tôt, il est possible qu'il ait utilisé la pente de la première barre, qui aurait été probablement plus penchée si il s'agissait d'un « W », mais la prédiction était risquée. Pour les deux gestes suivants, le réseau attend également que les parties communes avec d'autres gestes soient dépassées avant de faire des prédictions, ce qui arrive généralement tôt sur cette base (NDToD entre 20% et 30% suivant le Ψ utilisé). Sur cette base, il est possible d'augmenter le Ψ a des valeurs plus importantes sans dégrader de manière importante les résultats de détection.

4.5 Conclusion

Notre travail souligne l'importance de prendre en compte les exigences spécifiques des systèmes interactifs lors de la conception de méthodes de détection d'actions en ligne. Très peu de travaux dans la littérature ont abordé la tâche de détection d'action en ligne



(a) ILGDB_Untrimmed



(b) MTGSetB_Untrimmed

FIGURE 4.25 – Résultats qualitatifs sur a) ILGDB et b) MTGSetB. Le système prend généralement sa décision aux instants décisifs (mis en évidence en rouge). Pour des raisons de visibilités, toutes les images (représentation) des gestes ne sont pas visibles.

en mettant l'accent sur la précocité et la prise de décision, ce qui est particulièrement important dans les systèmes interactifs.

Dans ce chapitre, nous avons présenté une approche pour la détection d'actions en ligne dans les systèmes interactifs. Nous avons introduit **E-SIM/E-SI**, une représentation indépendante de la vitesse qui représente le geste dans un espace euclidien. Nous avons présenté le réseau **Dual-stream Online Long-Term Convolutional 3D (DOLT-C3D)**, qui utilise efficacement les informations temporelles et spatiales fournies par la représentation pour améliorer la précision du processus de détection des gestes. Notre nouvelle fonction de coût, le **CTC** guidé par la segmentation, a montré sa capacité à localiser correctement les gestes dans le temps, tout en améliorant la précocité du système. Nous avons également présenté un **Label Prior** pondéré pour ajuster efficacement le compromis entre la précision et la précocité, ce qui le rend adapté à une large gamme d'applications interactives qui nécessitent différents niveaux de latence et de précision. L'évaluation sur six ensembles de données disponibles publiquement a démontré que notre approche surpasse les méthodes de l'état de l'art en termes de précision et de précocité. Ces trois éléments, la représentation **E-SIM/E-SI**, le réseau **DOLT-C3D**, le **CTC guidé**

avec ou sans le label prior pondéré sont indépendants, ils peuvent être utilisés dans d'autres systèmes de manière indépendante. En particulier, le CTC guidé, notamment avec le label prior faiblement pondéré ($0.1 \leq \Psi \leq 0.3$), est particulièrement prometteur et peut être facilement utilisé sur n'importe quel système à base de réseau de neurones qui cherche à faire de la reconnaissance (en ligne ou hors ligne) dans une séquence non segmentée, à condition d'avoir une information sur la localisation des éléments à reconnaître pendant l'apprentissage.

Ces récents résultats font l'objet d'un article en cours de soumission à la revue "Pattern Recognition".

Chapitre 5

Conclusion et Perspectives

5.1 Conclusion

Dans ce manuscrit, nous avons exploré le domaine de la reconnaissance précoce de gestes dans le contexte de l'interaction humain-machine. Nous avons cherché à développer une approche qui puisse être appliquée à une variété de gestes, qu'ils soient en deux dimensions sur une tablette ou en trois dimensions, réalisés par le corps humain. Pour résoudre ce défi complexe, nous avons choisi de nous concentrer sur l'utilisation de réseaux de neurones convolutifs (CNN) spatio-temporels, qui présentent des propriétés intrinsèquement adaptées à notre problématique.

Au travers de notre revue de l'état de l'art, nous avons étudié les multiples aspects de la reconnaissance de gestes en ligne, segmentés et non segmentés. Nous avons catégorisé et analysé les différentes tâches et métriques, examiné les mécanismes de prise de décision ainsi que les architectures de réseaux profonds.

En structurant nos recherches, nous avons clarifié nos objectifs en distinguant les tâches de reconnaissance de gestes au niveau frame et au niveau instance. Notre analyse approfondie des métriques et des mécanismes de prise de décision a mis en lumière les enjeux complexes de la mesure de performance dans ce contexte.

Ensuite, nous avons présenté nos contributions pour la reconnaissance précoce segmentée du geste 2D. Notre approche propose une représentation spatio-temporelle euclidienne du geste, un CNN 3D original nommé OLT-C3D, et un système de rejet temporel. Nous avons démontré l'efficacité de notre méthode à prédire très tôt les gestes avec des performances intéressantes.

De plus, nous avons exploré la **reconnaissance précoce de gestes 2D et 3D non segmentés**. Fondés sur nos travaux effectués en segmenté 2D, nous avons amélioré notre méthode pour l'adapter au cas du non segmenté. Une nouvelle fonction de coût reposant sur un CTC guidé a montré sa capacité à localiser correctement les gestes dans le temps,

et notre Label Prior pondéré permet d'**ajuster le compromis entre précision et précocité**.

En conclusion, ce manuscrit contribue à la fois à la compréhension théorique et aux applications pratiques de la reconnaissance précoce de gestes dans le domaine de l'interaction humain-machine. Nos travaux apportent une vision structurée et approfondie de ce domaine complexe, et nos approches originales ouvrent des perspectives pour de futures améliorations et extensions. Nous espérons que ces contributions contribueront à façonner les futures avancées de la reconnaissance précoce de gestes et à améliorer l'interaction entre les humains et les machines.

5.2 Perspectives

Plusieurs pistes de recherche restent à explorer dans de futurs travaux. L'une d'entre elles consiste à améliorer les performances de notre approche en matière d'apprentissage semi-supervisé, lorsque l'accès à de grandes quantités de données annotées par image est limité. Cet objectif peut être atteint en explorant davantage et en affinant les capacités de la fonction de coût CTC, permettant l'apprentissage sans annotation par frame.

En termes d'architecture de réseau, l'intégration de transformers appris de manière non supervisée est prometteuse pour améliorer la détection d'actions en ligne. En entraînant le réseau à prédire les futures poses du squelette, il peut apprendre à capturer la dynamique temporelle et les dépendances des mouvements de manière plus efficace. L'utilisation de transformers pourrait améliorer les performances globales et la précision de notre système. Cependant, ces architectures sont généralement plus lourdes en termes de poids et de temps d'apprentissage. En effet, les réseaux que nous avons proposés dans cette thèse ont un nombre de poids inférieur à 1 million de paramètres ($\approx$150K pour le geste 2D, $\approx$ 700K pour le geste 3D), ce qui est fait d'eux des réseaux de taille très raisonnables.

En relevant les défis liés à l'apprentissage semi-supervisé et en explorant des architectures de réseau innovantes, nous pourrons vraisemblablement continuer à améliorer les capacités et les performances de la détection d'action en ligne, améliorant ainsi l'expérience des utilisateurs dans divers domaines interactifs.

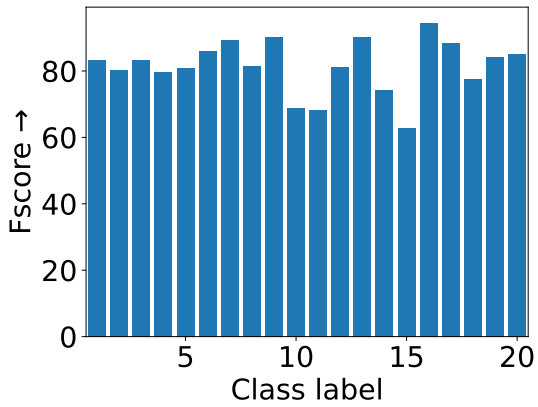
Annexe A

Résultats complets pour la tâche d'OAD, gestes 2D et 3D

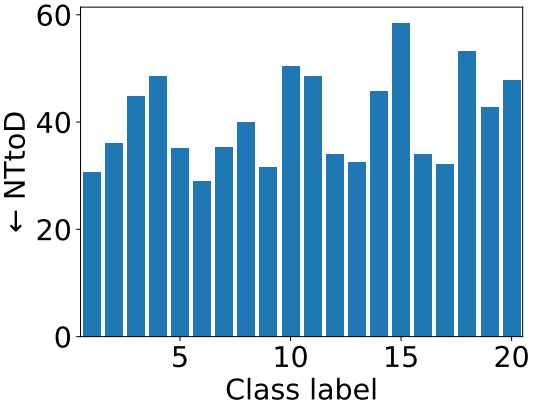
A.1 Base de données Chalearn

Ψ	Fscore ↑	Précision ↑	Rappel ↑	NTToD ↓
0.0	81.4 ± 0.7	91.7 ± 1.4	73.3 ± 1.7	51.9 ± 0.7
0.1	81.9 ± 0.4	86.0 ± 2.3	78.3 ± 2.2	39.6 ± 1.2
0.2	78.1 ± 1.3	77.2 ± 2.9	79.2 ± 0.8	34.6 ± 0.9
0.3	72.2 ± 1.2	67.8 ± 3.0	77.4 ± 1.4	31.0 ± 0.8
0.4	66.5 ± 2.1	59.2 ± 2.7	75.8 ± 1.2	29.4 ± 0.9
0.5	61.5 ± 0.9	52.4 ± 1.1	74.4 ± 0.5	27.3 ± 0.4
0.6	56.0 ± 1.6	46.2 ± 2.1	71.1 ± 0.5	26.2 ± 0.5
0.7	54.3 ± 1.1	44.6 ± 1.3	69.4 ± 1.1	25.3 ± 0.7
0.8	51.6 ± 1.7	41.3 ± 1.8	68.5 ± 0.9	24.6 ± 0.2
0.9	47.3 ± 1.9	36.8 ± 2.1	65.9 ± 1.4	23.7 ± 0.5
1.0	46.5 ± 1.7	36.3 ± 1.7	64.8 ± 1.1	23.3 ± 0.6

Table A.1 – Score complets de notre approche (E-SIM + DOLT-C3D + $CTC_{SSG,\Psi}$) sur la base de données Chalearn. Fscore de la métrique BOD ($\Delta = 0$ et $canCorrect = False$) et NTToD.

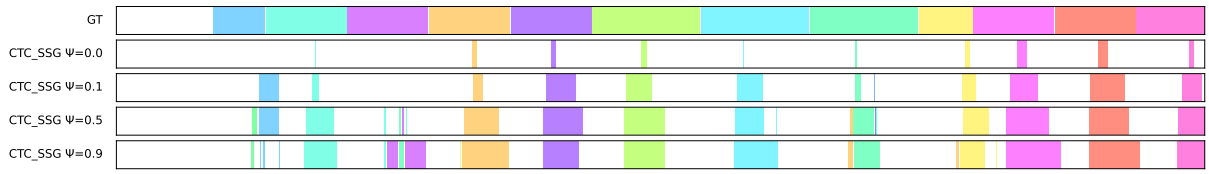


(a) Fscore par classe pour $\Psi = 0.1$

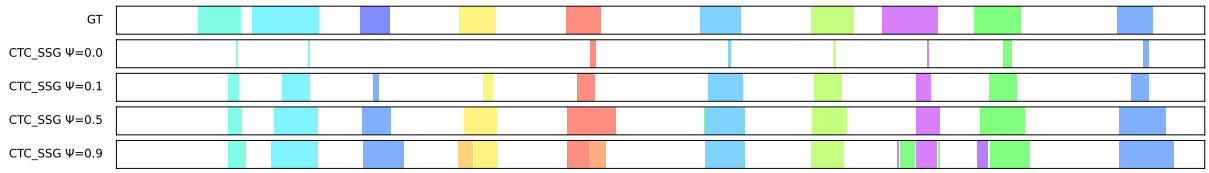


(b) Précocité par classe pour $\Psi = 0.1$

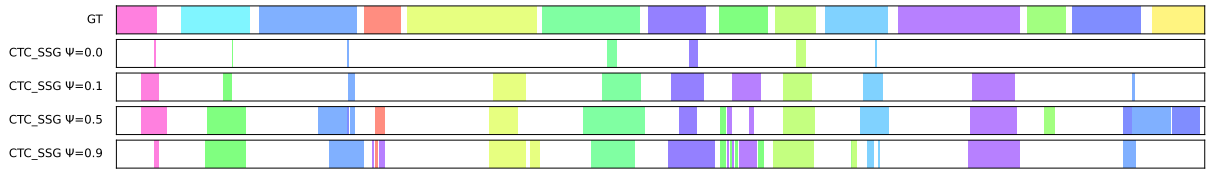
FIGURE A.1 – Fscores par classe (BOD Fscore $\Delta = 0$, $canCorrect = False$) et précocité par classe sur la base de données Chalearn pour E-SIM + DOLT-C3D + $CTC_{SSG,\Psi=0.1}$.



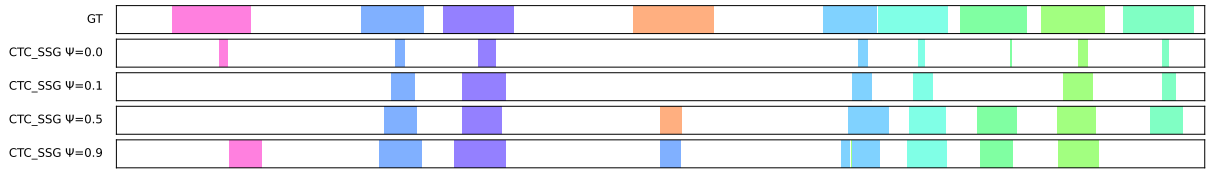
(a) Sample00137



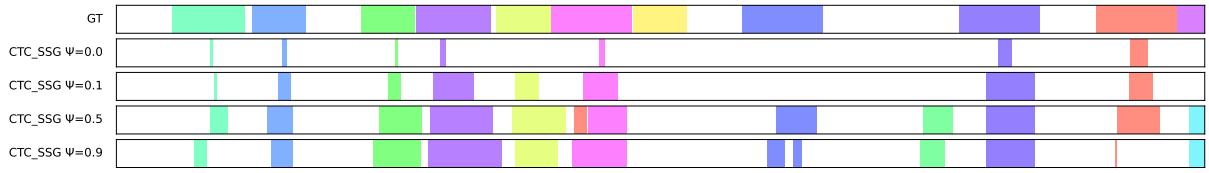
(b) Sample00044



(c) Sample00375



(d) Sample00555



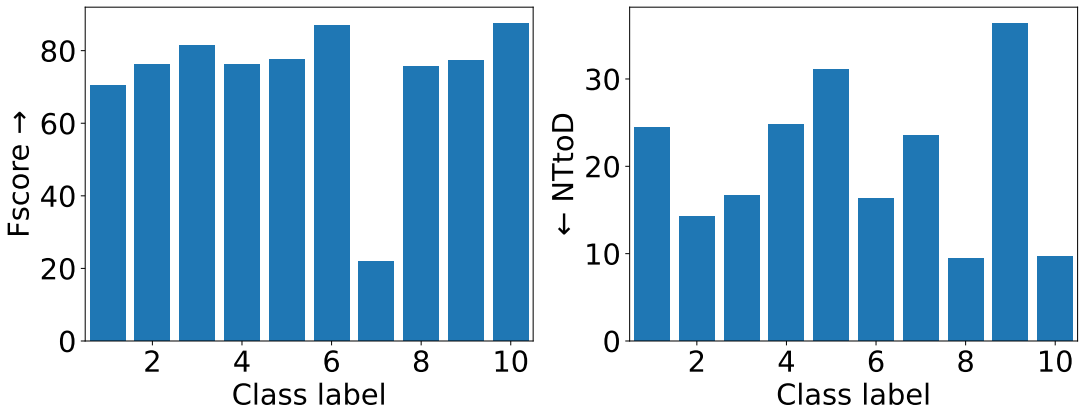
(e) Sample00710

FIGURE A.2 – Résultats qualitatifs de notre approche (E-SIM + DOLT-C3D + $CTC_{SSG,\Psi}$) sur la base de données Chalearn pour différentes séquences.

A.2 Base de données OAD

Ψ	Fscore ↑	Précision ↑	Rappel ↑	NTToD ↓
0.0	64.3 ± 2.5	88.1 ± 3.1	50.8 ± 4.0	35.0 ± 1.6
0.1	75.7 ± 1.0	77.0 ± 2.4	74.6 ± 3.6	20.6 ± 1.8
0.2	72.0 ± 1.3	69.8 ± 3.1	74.9 ± 5.0	20.1 ± 1.8
0.3	71.0 ± 1.5	64.2 ± 5.2	80.3 ± 4.7	16.9 ± 2.8
0.4	72.0 ± 1.9	63.6 ± 2.9	83.0 ± 1.8	14.0 ± 1.5
0.5	72.0 ± 2.0	62.7 ± 2.0	84.7 ± 2.1	14.4 ± 2.0
0.6	71.4 ± 2.2	61.9 ± 2.9	84.6 ± 1.8	13.3 ± 1.0
0.7	69.3 ± 2.4	58.3 ± 4.1	85.8 ± 2.4	10.8 ± 3.2
0.8	69.5 ± 1.4	58.4 ± 1.9	86.1 ± 1.8	10.5 ± 1.0
0.9	66.4 ± 1.6	54.3 ± 1.9	85.7 ± 1.9	9.4 ± 1.5
1.0	66.8 ± 2.7	54.7 ± 3.0	85.9 ± 1.9	8.7 ± 0.6

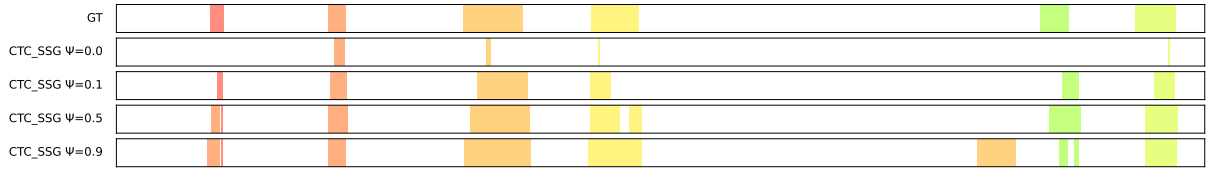
TABLE A.2 – Score complets de notre approche (E-SIM + DOLT-C3D + $CTC_{SSG,\Psi}$) sur la base de données OAD. Fscore de la métrique BOD ($\Delta = 0$ et $canCorrect = False$) et NTToD.



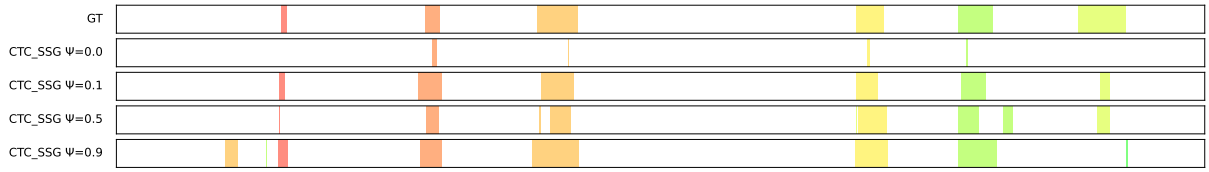
(a) Fscore par classe

(b) Précocité par classe

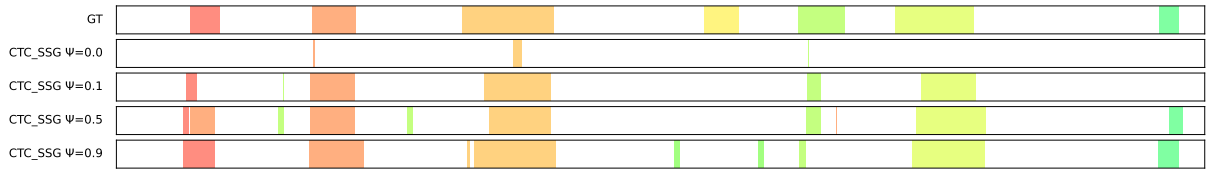
FIGURE A.3 – Fscores par classe (BOD Fscore $\Delta = 0$, $canCorrect = False$) et précocité par classe sur la base de données OAD, avec E-SIM + DOLT-C3D + $CTC_{SSG,\Psi=0.1}$).



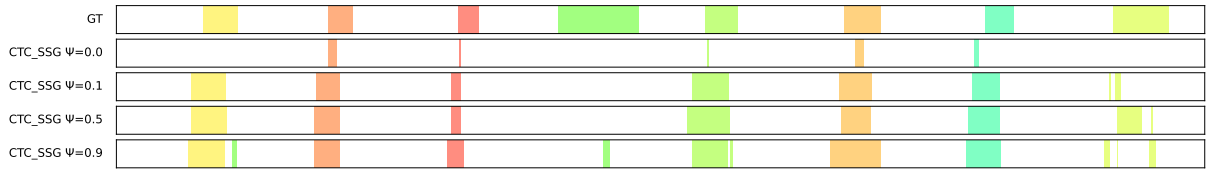
(a) Séquence 0



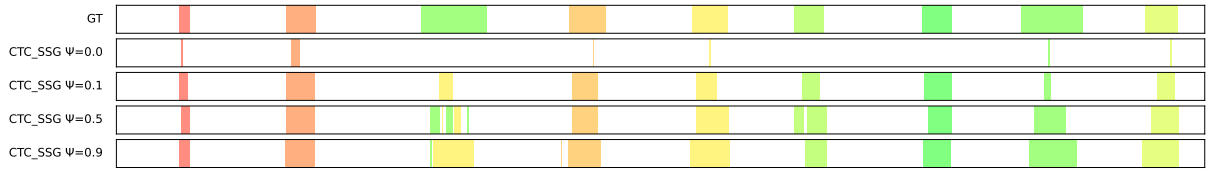
(b) Séquence 17



(c) Séquence 36



(d) Séquence 40



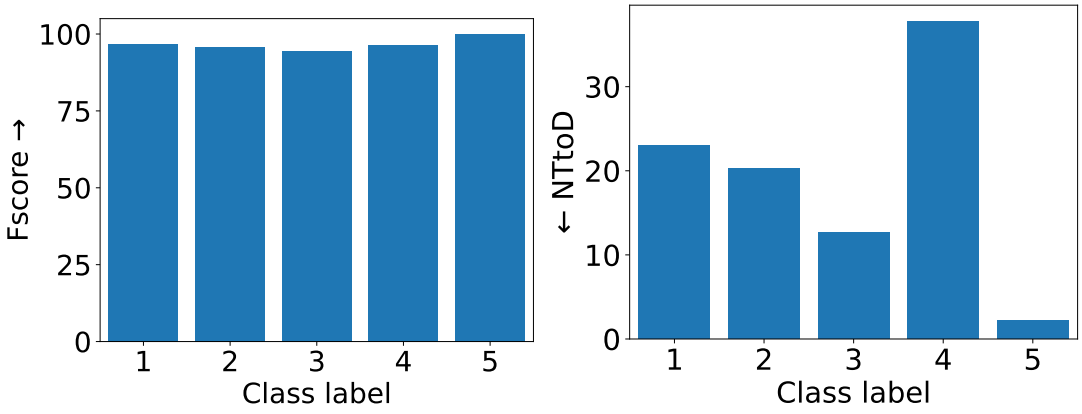
(e) Séquence 52

FIGURE A.4 – Résultats qualitatifs de notre approche (E-SIM + DOLT-C3D + $CTC_{SSG,\Psi}$) sur la base de données OAD pour différents exemples.

A.3 Base de données G3D

Ψ	Fscore ↑	Précision ↑	Rappel ↑	NTToD ↓
0.0	87.7 ± 2.9	98.4 ± 2.6	79.3 ± 4.9	38.9 ± 2.6
0.1	97.1 ± 2.6	98.6 ± 2.3	95.7 ± 4.2	20.0 ± 5.2
0.2	99.3 ± 1.1	98.6 ± 2.2	100.0 ± 0.0	17.1 ± 2.2
0.3	98.3 ± 1.7	96.7 ± 3.2	100.0 ± 0.0	15.1 ± 3.0
0.4	98.6 ± 1.8	98.6 ± 2.2	98.6 ± 2.3	15.0 ± 4.0
0.5	98.9 ± 1.2	98.6 ± 2.2	99.3 ± 1.7	13.8 ± 2.5
0.6	96.5 ± 1.7	94.0 ± 2.9	99.3 ± 1.7	12.4 ± 1.7
0.7	98.3 ± 1.7	96.7 ± 3.2	100.0 ± 0.0	12.3 ± 2.6
0.8	98.3 ± 1.7	96.7 ± 3.2	100.0 ± 0.0	10.1 ± 1.0
0.9	99.0 ± 1.2	98.0 ± 2.4	100.0 ± 0.0	10.4 ± 1.1
1.0	98.3 ± 1.1	96.6 ± 2.2	100.0 ± 0.0	9.6 ± 1.6

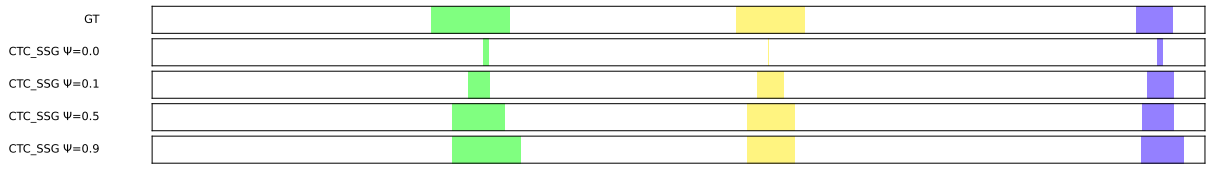
TABLE A.3 – Score complets de notre approche (E-SIM + DOLT-C3D + $CTC_{SSG,\Psi}$) sur la base de données G3D. Fscore de la métrique BOD ($\Delta = 0$ et $canCorrect = False$) et NTToD.



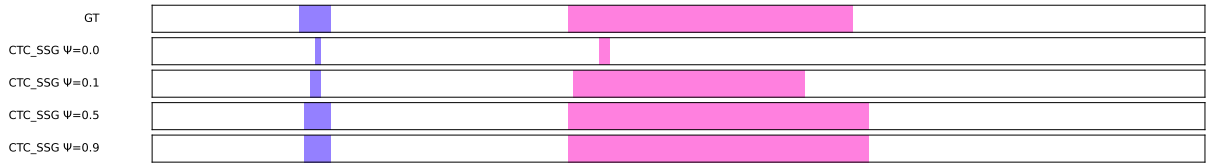
(a) Fscore par classe

(b) Précocité par classe

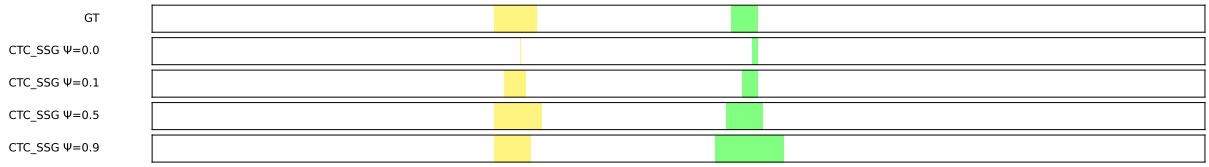
FIGURE A.5 – Fscores par classe (BOD Fscore $\Delta = 0$, $canCorrect = False$) et précocité par classe sur la base de données G3D, pour E-SIM + DOLT-C3D + $CTC_{SSG,\Psi}$.



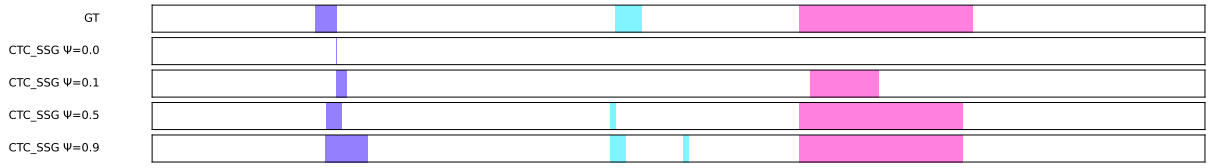
(a) test_modif65



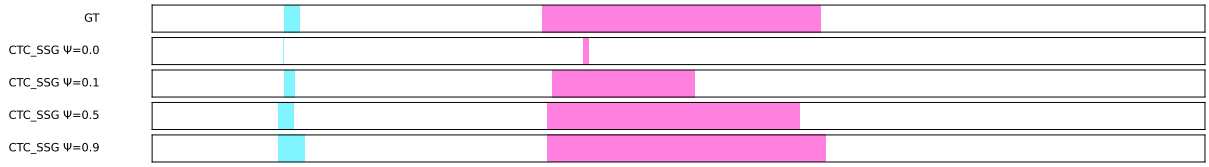
(b) test2_modif65



(c) test_modif150



(d) test2_modif150



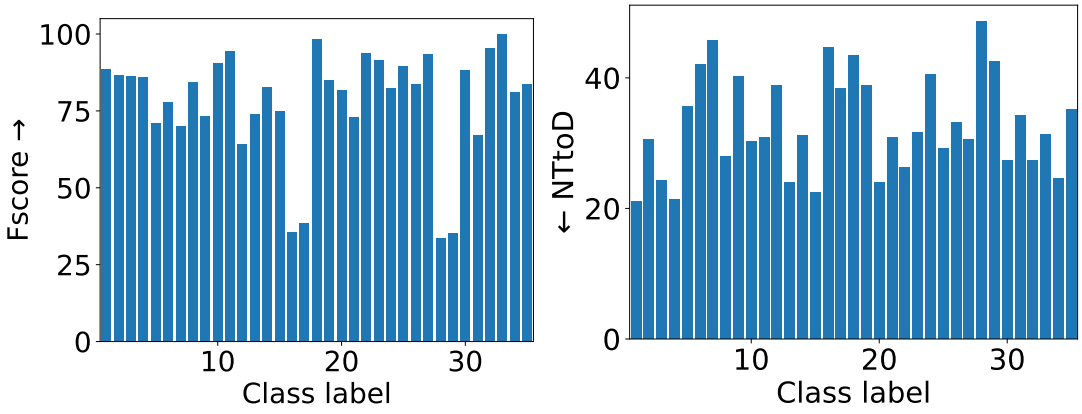
(e) test2_modif169

FIGURE A.6 – Résultats qualitatifs de notre approche (E-SIM + DOLT-C3D + $CTC_{SSG,\Psi}$) sur la base de données G3D pour différentes séquences.

A.4 Base de données MAD

Ψ	Fscore ↑	Précision ↑	Rappel ↑	NTToD ↓
0.0	76.0 ± 0.9	84.6 ± 1.8	68.9 ± 1.1	34.2 ± 0.4
0.1	78.0 ± 1.5	74.9 ± 2.5	81.4 ± 1.3	32.0 ± 0.7
0.2	74.9 ± 2.1	68.9 ± 2.9	82.2 ± 2.2	32.0 ± 1.5
0.4	72.1 ± 2.9	63.6 ± 3.8	83.3 ± 1.2	29.1 ± 2.0
0.6	63.8 ± 2.1	53.4 ± 2.7	79.4 ± 1.2	25.2 ± 1.1
0.8	59.7 ± 2.5	48.7 ± 2.6	77.3 ± 1.7	23.5 ± 1.1
1.0	56.3 ± 1.5	44.9 ± 1.8	75.4 ± 0.9	23.9 ± 0.5

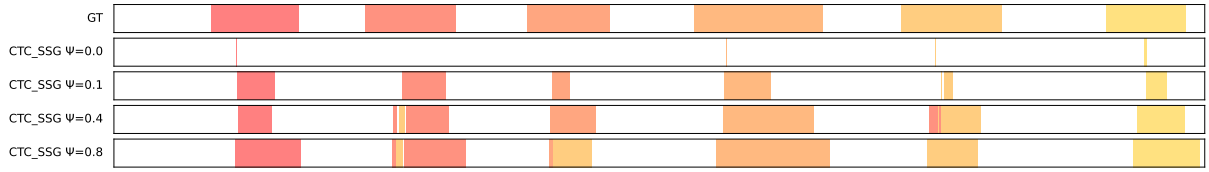
TABLE A.4 – Score complets de notre approche (E-SIM + DOLT-C3D + $CTC_{SSG,\Psi}$) sur la base de données MAD. Fscore de la métrique BOD ($\Delta = 0$ et $canCorrect = False$) et NTToD.



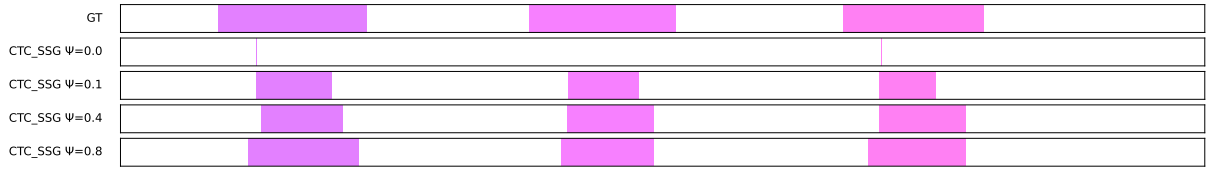
(a) Fscore par classe

(b) Précocité par classe

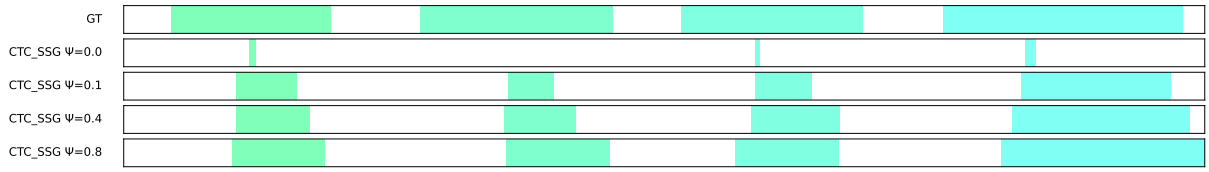
FIGURE A.7 – Fscores par classe (BOD Fscore $\Delta = 0$, $canCorrect = False$) et précocité par classe sur la base de données MAD, pour E-SIM + DOLT-C3D + $CTC_{SSG,\Psi=0.1}$



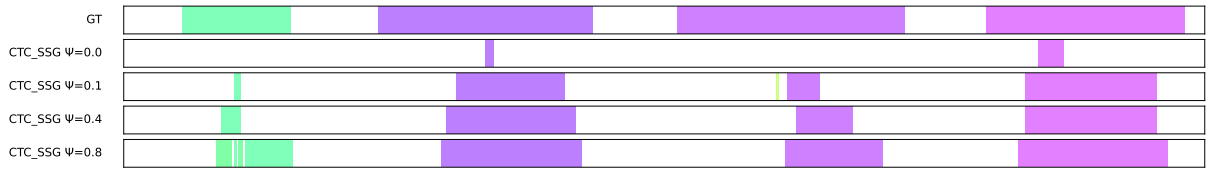
(a) test0_modifsub05_01



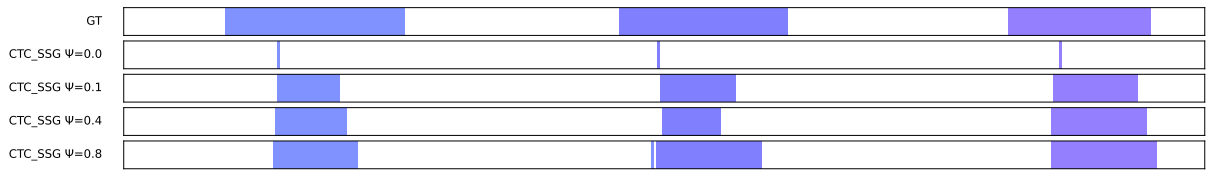
(b) test10_modifsub05_02



(c) test4_modifsub07_01



(d) test7_modifsub10_01



(e) test8_modifsub05_02

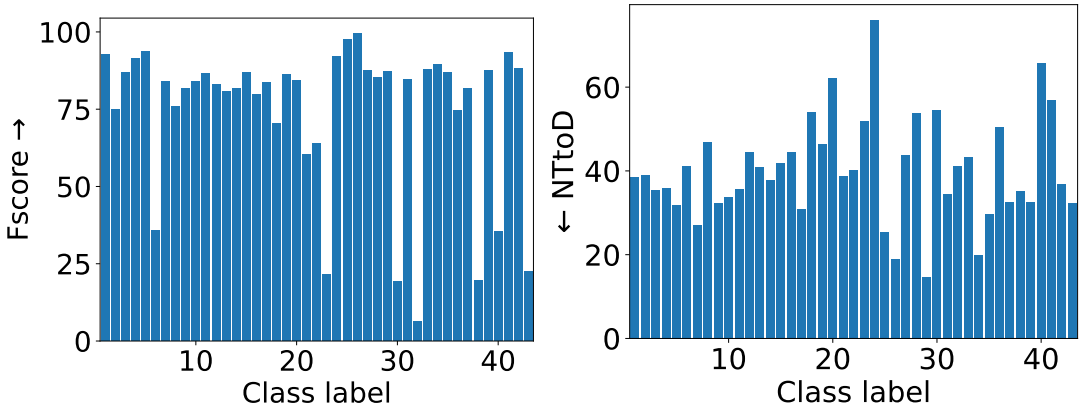
FIGURE A.8 – Résultats qualitatifs de notre approche (E-SIM + DOLT-C3D + $CTC_{SSG,\Psi}$) sur la base de données MAD pour différentes séquences.

A.5 Base de données PKUMMD

A.5.1 Protocole cross-suject

Ψ	Fscore ↑	Précision ↑	Rappel ↑	NTToD ↓
0.0	45.5 ± 6.7	95.4 ± 1.5	30.2 ± 6.1	54.3 ± 1.2
0.1	77.1 ± 3.3	92.7 ± 3.5	66.0 ± 3.0	38.8 ± 1.8
0.2	69.7 ± 1.5	81.3 ± 1.2	61.1 ± 2.8	39.9 ± 0.1
0.4	69.2 ± 1.3	64.6 ± 2.2	74.6 ± 0.5	32.6 ± 0.3
0.6	62.0 ± 0.7	52.6 ± 1.3	75.5 ± 0.5	28.4 ± 0.3
0.8	52.2 ± 2.9	40.4 ± 2.8	73.6 ± 2.3	23.6 ± 0.6
1.0	49.7 ± 7.6	37.8 ± 7.1	73.1 ± 6.2	20.7 ± 2.5

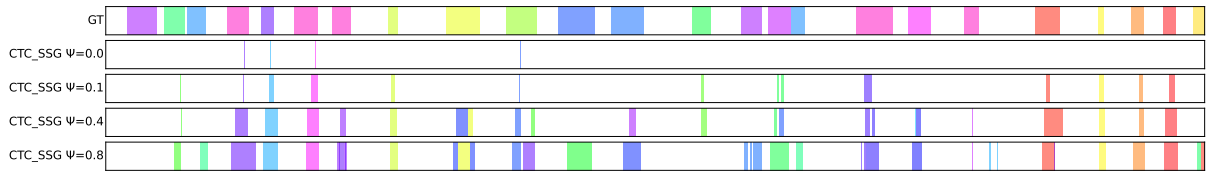
TABLE A.5 – Score complets de notre approche (E-SIM + DOLT-C3D + $CTC_{SSG,\Psi}$) sur la base de données PKUMMD (protocole cross-subject). Fscore de la métrique BOD ($\Delta = 0$ et $canCorrect = False$) et NTToD.



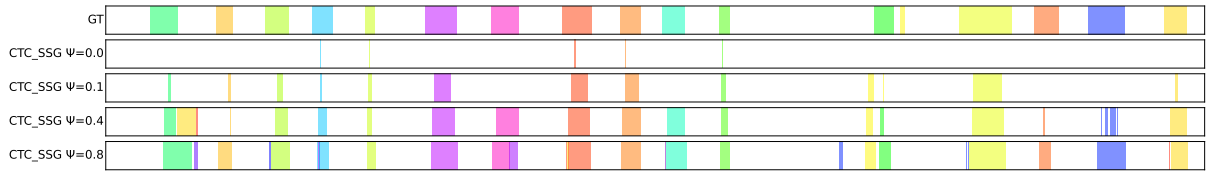
(a) Fscore par classe

(b) Précocité par classe

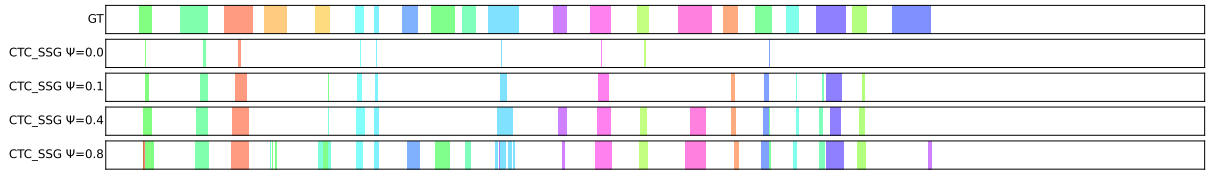
FIGURE A.9 – Fscores par classe (BOD Fscore $\Delta = 0$, $canCorrect = False$) et précocité par classe sur la base de données PKUMMD_cs pour E-SIM + DOLT-C3D + $CTC_{SSG,\Psi=0.1}$



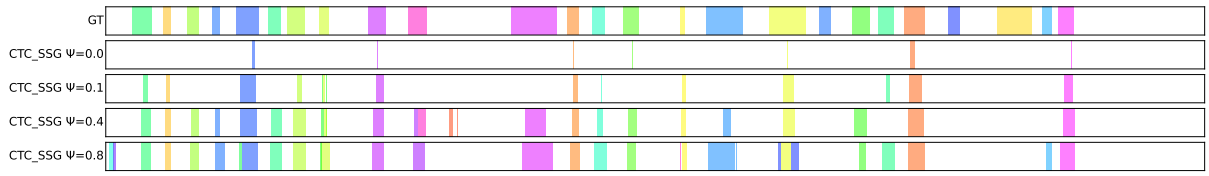
(a) 0293-M



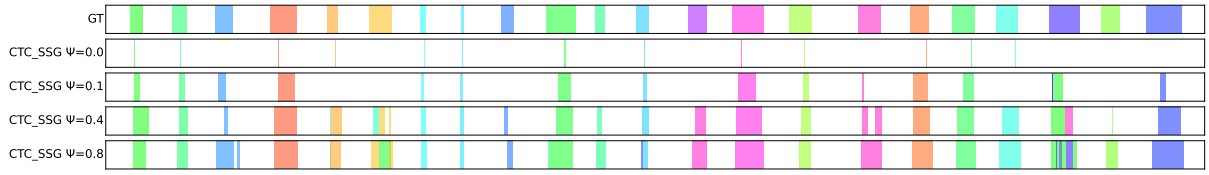
(b) 0295-M



(c) 0306-L



(d) 0311-L



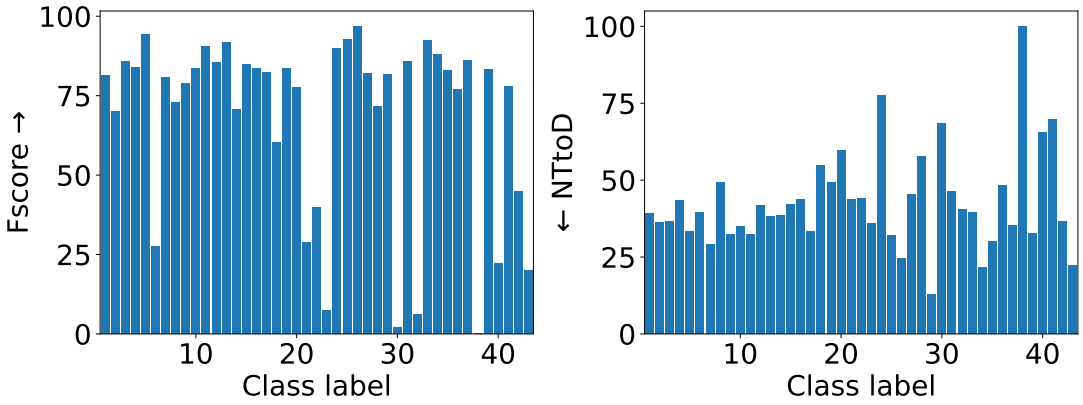
(e) 0334-R

FIGURE A.10 – Résultats qualitatifs de notre approche (E-SIM + DOLT-C3D + $CTC_{SSG,\Psi}$) sur la base de données PKUMMD_cs pour différentes séquences.

A.5.2 Protocole cross-view

Ψ	Fscore ↑	Précision ↑	Rappel ↑	NTToD ↓
0.0	51.8 ± 0.6	95.8 ± 0.1	35.5 ± 0.5	50.5 ± 0.0
0.1	71.9 ± 1.9	92.2 ± 3.5	58.9 ± 1.3	39.7 ± 1.3
0.2	72.7 ± 0.6	77.4 ± 0.5	68.6 ± 0.9	34.3 ± 0.8
0.4	67.9 ± 0.6	62.1 ± 1.3	74.9 ± 1.5	29.3 ± 0.7
0.6	61.7 ± 2.2	51.8 ± 2.4	76.5 ± 1.4	25.5 ± 0.2
0.8	52.5 ± 2.1	40.7 ± 2.1	73.9 ± 1.8	22.3 ± 1.0
1.0	44.1 ± 0.6	32.2 ± 0.3	69.8 ± 1.8	19.0 ± 0.4

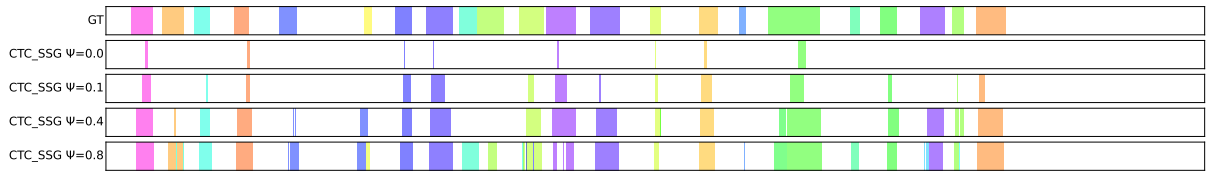
TABLE A.6 – Score complets de notre approche (E-SIM + DOLT-C3D + $CTC_{SSG,\Psi}$) sur la base de données PKUMMD_cv. Fscore de la métrique BOD ($\Delta = 0$ et $canCorrect = False$) et NTToD.



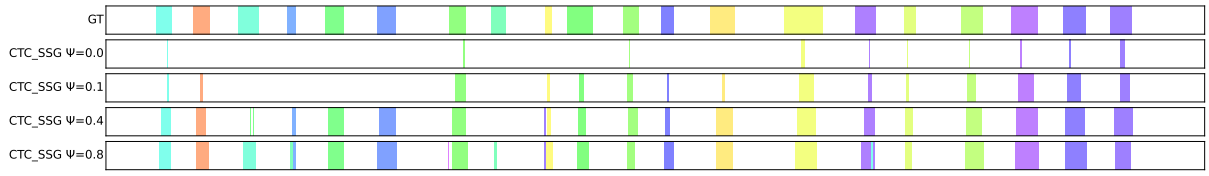
(a) Fscore par classe

(b) Précocité par classe

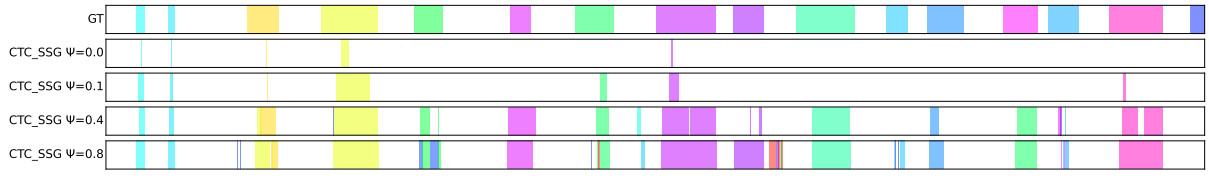
FIGURE A.11 – Fscores par classe (BOD Fscore $\Delta = 0$, $canCorrect = False$) et précocité par classe sur la base de données PKUMMD_cv pour E-SIM + DOLT-C3D + $CTC_{SSG,\Psi=0.1}$. Note : Pour la classe 38, la précocité est indiquée à 100, mais aucun geste de cette classe n'a été reconnu, donc il n'est pas possible de calculer la précocité avec la métrique NTtoD pour cette classe (calculée uniquement pour les TP).



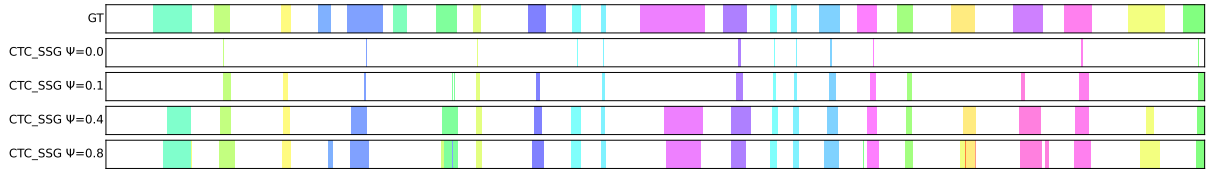
(a) 0010-M



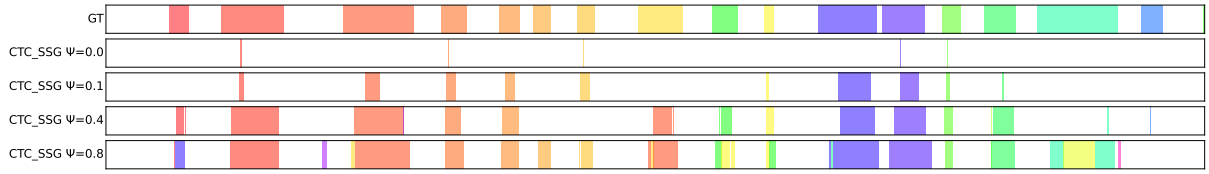
(b) 0070-M



(c) 0171-M



(d) 0290-M



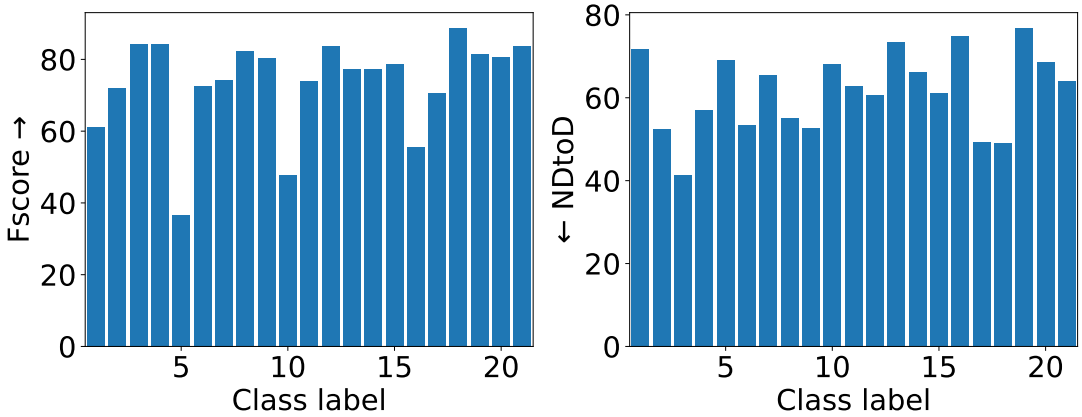
(e) 0355-M

FIGURE A.12 – Résultats qualitatifs de notre approche (E-SIM + DOLT-C3D + $CTC_{SSG,\Psi}$) sur la base de données PKUMMD_cv pour différentes séquences.

A.6 Base de données ILGDB_Untrimmed (2D)

Ψ	Fscore ↑	Précision ↑	Rappel ↑	NTToD ↓
0.0	60.6 ± 4.9	70.1 ± 6.3	53.3 ± 4.3	59.5 ± 3.5
0.1	75.3 ± 2.1	81.2 ± 2.7	70.2 ± 1.6	61.4 ± 1.5
0.2	73.6 ± 0.6	75.1 ± 1.4	72.2 ± 1.1	58.4 ± 0.3
0.3	69.4 ± 1.7	67.1 ± 3.3	71.9 ± 0.6	56.0 ± 0.7
0.5	43.3 ± 0.5	35.7 ± 0.5	55.0 ± 0.6	44.2 ± 0.7
0.7	30.5 ± 0.6	23.0 ± 0.5	45.5 ± 0.9	34.3 ± 1.1
0.9	26.1 ± 0.7	18.9 ± 0.6	42.1 ± 1.2	29.8 ± 0.6

TABLE A.7 – Score complets de notre approche (E-SI + OLT-C3D + $CTC_{SSG,\Psi}$) sur la base de données ILGDB_Untrimmed. Fscore de la métrique BOD ($\Delta = 0$ et $canCorrect = False$) et NTToD.



(a) Fscore par classe

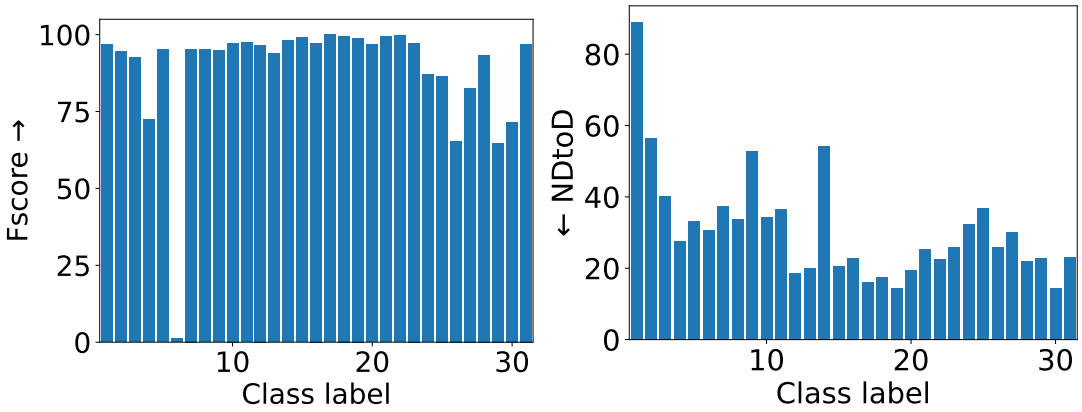
(b) Précocité par classe

FIGURE A.13 – Fscores par classe (BOD Fscore $\Delta = 0$, $canCorrect = False$) et précocité (NDToD) par classe sur la base de données ILGDB, pour E-SI + OLT-C3D + $CTC_{SSG,\Psi=0.1}$

A.7 Base de données MTGSetB_Untrimmed (2D)

Ψ	Fscore ↑	Précision ↑	Rappel ↑	NTToD ↓
0.0	84.6 ± 0.2	85.4 ± 0.3	83.7 ± 0.5	23.6 ± 0.9
0.1	91.2 ± 0.2	93.6 ± 0.3	88.9 ± 0.5	30.7 ± 0.1
0.2	90.3 ± 0.7	90.4 ± 1.4	90.2 ± 0.1	28.3 ± 0.4
0.3	87.5 ± 1.1	84.1 ± 1.6	91.2 ± 0.6	27.1 ± 0.7
0.5	78.1 ± 1.0	70.9 ± 1.4	87.0 ± 0.4	21.4 ± 1.0
0.7	73.7 ± 0.4	65.4 ± 0.3	84.6 ± 0.6	16.8 ± 1.1
0.9	71.9 ± 2.0	62.5 ± 2.4	84.6 ± 1.2	14.9 ± 0.2

TABLE A.8 – Score complets de notre approche (E-SI + OLT-C3D + $CTC_{SSG,\Psi}$) sur la base de données MTGSetB. Fscore de la métrique BOD ($\Delta = 0$ et $canCorrect = False$) et NDToD.



(a) Fscore par classe

(b) Précocité par classe

FIGURE A.14 – Fscores par classe (BOD Fscore $\Delta = 0$, $canCorrect = False$) et précocité (NDToD) par classe sur la base de données MTGSetB pour E-SI + OLT-C3D + $CTC_{SSG,\Psi=0.1}$.

BIBLIOGRAPHIE

Publications de l'auteur

Conférences Internationales avec comité de lecture

[MAK21] William MOCAËR, Eric ANQUETIL et Richard KULPA, « Online Spatio-temporal 3D Convolutional Neural Network for Early Recognition of Handwritten Gestures », in : *Document Analysis and Recognition – ICDAR 2021*, sous la dir. de Josep LLADÓS, Daniel LOPRESTI et Seiichi UCHIDA, Cham : Springer International Publishing, 2021, p. 221-236, ISBN : 978-3-030-86549-8.

[MAK22a] William MOCAËR, Eric ANQUETIL et Richard KULPA, « Early Recognition of Untrimmed Handwritten Gestures with Spatio-Temporal 3D CNN », in : *2022 26th International Conference on Pattern Recognition (ICPR)*, 2022, p. 1636-1642, DOI : 10.1109/ICPR56361.2022.9956529.

Conférences Nationales avec comité de lecture

[MAK22b] William MOCAËR, Eric ANQUETIL et Richard KULPA, « Réseau Convolutif Spatio-Temporel 3D pour la Reconnaissance Précoce de Gestes Manuscrits Non-Segmentés », in : *RFIAP 2022 - Congrès Reconnaissance des Formes, Image, Apprentissage et Perception*, Vannes, France, juill. 2022, p. 1-9, URL : https://hal.science/hal-03682604.

Journal International avec comité de lecture

En cours de soumission

[MAK23] William MOCAËR, Eric ANQUETIL et Richard KULPA, « Early Action Detection at instance-level driven by a controlled CTC-Based Approach », in : *Pattern Recognition (PR)* (2023), En cours de soumission.

Références

[Ahn+23] Dasom Ahn et al., « STAR-Transformer : A Spatio-Temporal Cross Attention Transformer for Human Action Recognition », in : *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, jan. 2023, p. 3330-3339.

[Ali+17] Mohammad Sadegh Aliakbarian et al., « Encouraging LSTMs to Anticipate Actions Very Early », in : *2017 IEEE International Conference on Computer Vision (ICCV)*, 2017, p. 280-289, doi : 10.1109/ICCV.2017.39.

[Arn+21] Anurag Arnab et al., « ViViT : A Video Vision Transformer », in : *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, oct. 2021, p. 6836-6846.

[BAM13] Victoria Bloom, Vasileios Argyriou et Dimitrios Makris, « Dynamic Feature Selection for Online Action Recognition », in : *Human Behavior Understanding*, sous la dir. d'Albert Ali Salah et al., Cham : Springer International Publishing, 2013, p. 64-76, isbn : 978-3-319-02714-2.

[BAM17] Victoria Bloom, Vasileios Argyriou et Dimitrios Makris, « Linear latent low dimensional space for online early action recognition and prediction », in : *Pattern Recognition* 72 (2017), p. 532-547, doi : 10.1016/j.patcog.2017.07.003.

[Bap+20] Marcos Baptista-Ríos et al., « Rethinking Online Action Detection in Untrimmed Videos : A Novel Online Evaluation Protocol », in : *IEEE Access* 8 (2020), p. 5139-5146, doi : 10.1109/ACCESS.2019.2961789.

[BKK17] S. Baek, K. I. Kim et T. Kim, « Real-Time Online Action Detection Forests Using Spatio-Temporal Contexts », in : *2017 IEEE Winter Conference on Applications of Computer Vision (WACV)*, mars 2017, p. 158-167.

[BMA12] V. Bloom, D. Makris et V. Argyriou, « G3D : A gaming action dataset and real time action recognition evaluation framework », in : *2012 IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops*, juin 2012, p. 7-12.

[BMA14] Victoria Bloom, Dimitrios Makris et Vasileios Argyriou, « Clustered Spatio-temporal Manifolds for Online Action Recognition », in : *2014 22nd International Conference on Pattern Recognition*, 2014, p. 3963-3968, DOI : 10.1109/ICPR.2014.679.

[Bou+18a] Said Yacine Boulahia et al., « CuDi3D : Curvilinear displacement based approach for online 3D action detection », in : *Computer Vision and Image Understanding* 174 (2018), p. 57-69, ISSN : 1077-3142, DOI : 10.1016/j.cviu.2018.07.003.

[Bou+18b] Said Yacine Boulahia et al., « Détection précoce d'actions squelettiques 3D dans un flot non segmenté à base de modèles curvilignes », in : *RFIAP 2018 Reconnaissance des Formes, Image, Apprentissage et Perception*, Paris, France, juin 2018, p. 1-8.

[Cae+19] Carlos Caetano et al., « SkeleMotion : A New Representation of Skeleton Joint Sequences based on Motion Information for 3D Action Recognition », in : *2019 16th IEEE International Conference on Advanced Video and Signal Based Surveillance (AVSS)*, 2019, p. 1-8, DOI : 10.1109/AVSS.2019.8909840.

[Cai+19] Yijun Cai et al., « Action Knowledge Transfer for Action Prediction with Partial Videos », in : *Proceedings of the AAAI Conference on Artificial Intelligence* 33.*01* (juill. 2019), p. 8118-8125, DOI : 10.1609/aaai.v33i01.33018118, URL : https://ojs.aaai.org/index.php/AAAI/article/view/4820.

[Cao+19a] Congqi Cao et al., « Skeleton-Based Action Recognition With Gated Convolutional Neural Networks », in : *IEEE Transactions on Circuits and Systems for Video Technology* 29.*11* (2019), p. 3247-3257, DOI : 10.1109/TCSVT.2018.2879913.

[Cao+19b] Yue Cao et al., « GCNet : Non-Local Networks Meet Squeeze-Excitation Networks and Beyond », in : *2019 IEEE/CVF International Conference on Computer Vision Workshop (ICCVW)*, 2019, p. 1971-1980, DOI : 10.1109/ICCVW.2019.00246.

[Cao+22] Shuqiang Cao et al., *A Circular Window-based Cascade Transformer for Online Action Detection*, 2022, arXiv : 2208.14209 [cs.CV].

[Cao+23] Shuqiang Cao et al., *E2E-LOAD : End-to-End Long-form Online Action Detection*, 2023, arXiv : 2306.07703 [cs.CV].

[Car+14] Baptiste Caramiaux et al., « Adaptive Gesture Recognition with Variation Estimation for Interactive Systems », in : *ACM Trans. Interact. Intell. Syst.* 4.*4* (déc. 2014), issn : 2160-6455, doi : 10.1145/2643204, url : https://doi.org/10.1145/2643204.

[Car+19] Fabio Carrara et al., « LSTM-based real-time action detection and prediction in human motion streams », in : *Multimedia Tools and Applications* 78.*19* (oct. 2019), p. 27309-27331, issn : 1573-7721, doi : 10.1007/s11042-019-07827-3.

[CBS19] Carlos Caetano, François Brémond et William Robson Schwartz, « Skeleton Image Representation for 3D Action Recognition Based on Tree Structure and Reference Joints », in : *2019 32nd SIBGRAPI Conference on Graphics, Patterns and Images (SIBGRAPI)*, 2019, p. 16-23, doi : 10.1109/SIBGRAPI.2019.00011.

[Che+15] Zhaoxin Chen et al., « Recognize multi-touch gestures by graph modeling and matching », in : *17th Biennial Conference of the International Graphonomics Society*, Drawing, Handwriting Processing Analysis : New Advances and Challenges, International Graphonomics Society (IGS) and Université des Antilles (UA), Pointe-a-Pitre, Guadeloupe, juin 2015.

[Che+17] Z. Chen et al., « Early Recognition of Handwritten Gestures Based on Multi-Classifier Reject Option », in : *14th IAPR International Conference on Document Analysis and Recognition (ICDAR)*, t. 01, 2017, p. 212-217, doi : 10.1109/ICDAR.2017.43.

[Che+20] Ke Cheng et al., « Skeleton-Based Action Recognition With Shift Graph Convolutional Network », in : *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, juin 2020.

[Che+21] Yuxin Chen et al., « Channel-Wise Topology Refinement Graph Convolution for Skeleton-Based Action Recognition », in : *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, oct. 2021, p. 13359-13368.

[Che+22] Junwen Chen et al., « GateHUB : Gated History Unit With Background Suppression for Online Action Detection », in : *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, juin 2022, p. 19925-19934.

[Cho+20] Sangwoo Cho et al., « Self-Attention Network for Skeleton-based Human Action Recognition », in : *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, mars 2020.

[CZ17a] Joao Carreira et Andrew Zisserman, « Quo Vadis, Action Recognition ? A New Model and the Kinetics Dataset », in : *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, juill. 2017.

[CZ17b] Joao Carreira et Andrew Zisserman, « Quo Vadis, Action Recognition ? A New Model and the Kinetics Dataset », in : *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, juill. 2017, doi : 10.1109/CVPR.2017.502.

[Dai+21] Rui Dai et al., « PDAN : Pyramid Dilated Attention Network for Action Detection », in : *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, jan. 2021, p. 2970-2979.

[De +16] Roeland De Geest et al., « Online Action Detection », in : *Computer Vision – ECCV 2016*, sous la dir. de Bastian Leibe et al., Cham : Springer International Publishing, 2016, p. 269-284, isbn : 978-3-319-46454-1.

[Dev+17] Maxime Devanne et al., « Motion segment decomposition of RGB-D sequences for human behavior understanding », in : *Pattern Recognition* 61 (2017), p. 222-233, issn : 0031-3203, doi : https://doi.org/10.1016/j.patcog.2016.07.041, url : https://www.sciencedirect.com/science/article/pii/S0031320316301996.

[Dev+18] Guillaume Devineau et al., « Deep Learning for Hand Gesture Recognition on Skeletal Data », in : *2018 13th IEEE International Conference on Automatic Face & Gesture Recognition (FG 2018)*, 2018, p. 106-113, doi : 10.1109/FG.2018.00025.

[DFW15] Y. Du, Y. Fu et L. Wang, « Skeleton based action recognition with convolutional neural network », in : *2015 3rd IAPR Asian Conference on Pattern Recognition (ACPR)*, 2015, p. 579-583.

[Dos+21] Alexey Dosovitskiy et al., *An Image is Worth 16x16 Words : Transformers for Image Recognition at Scale*, 2021, arXiv : 2010.11929 [cs.CV].

[DT18] Roeland De Geest et Tinne Tuytelaars, « Modeling Temporal Structure with LSTM for Online Action Detection », in : *2018 IEEE Winter Conference on Applications of Computer Vision (WACV)*, 2018, p. 1549-1557, DOI : 10.1109/WACV.2018.00173.

[Dua+22] Haodong Duan et al., « Revisiting Skeleton-based Action Recognition », in : *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022, p. 2959-2968, DOI : 10.1109/CVPR52688.2022.00298.

[DWV19] Quentin De Smedt, Hazem Wannous et Jean-Philippe Vandeborre, « Heterogeneous hand gesture recognition using 3D dynamic skeletal data », in : *Computer Vision and Image Understanding* 181 (2019), p. 60-72, ISSN : 1077-3142, DOI : https://doi.org/10.1016/j.cviu.2019.01.008, URL : https://www.sciencedirect.com/science/article/pii/S1077314219300153.

[EMS16] Hugo Jair Escalante, Eduardo F. Morales et L. Enrique Sucar, « A naïve Bayes baseline for early gesture recognition », in : *Pattern Recognition Letters* 73 (2016), p. 91-99, ISSN : 0167-8655, DOI : 10.1016/j.patrec.2016.01.013.

[Esc+13] Sergio Escalera et al., « Multi-modal Gesture Recognition Challenge 2013 : Dataset and Results », in : *Proceedings of the 15th ACM on International Conference on Multimodal Interaction*, ICMI '13, ACM, 2013, p. 445-452.

[Eun+20] Hyunjun Eun et al., « Learning to Discriminate Information for Online Action Detection », in : *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020, p. 806-815, DOI : 10.1109/CVPR42600.2020.00089.

[Eun+21] Hyunjun Eun et al., « Temporal filtering networks for online action detection », in : *Pattern Recognition* 111 (2021), p. 107695, ISSN : 0031-3203, DOI : https://doi.org/10.1016/j.patcog.2020.107695, URL : https://www.sciencedirect.com/science/article/pii/S0031320320304982.

[Eve+10] Mark Everingham et al., « The Pascal Visual Object Classes (VOC) Challenge », in : *International Journal of Computer Vision* 88.*2* (juin 2010), p. 303-338, ISSN : 1573-1405, DOI : 10.1007/s11263-009-0275-4, URL : https://doi.org/10.1007/s11263-009-0275-4.

[Fei+19] Christoph Feichtenhofer et al., « SlowFast Networks for Video Recognition », in : *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, oct. 2019.

[FH21] Basura Fernando et Samitha Herath, « Anticipating Human Actions by Correlating Past With the Future With Jaccard Similarity Measures », in : *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, juin 2021, p. 13224-13233.

[Foo+22] Lin Geng Foo et al., « ERA : Expert Retrieval and Assembly for Early Action Prediction », in : *Computer Vision – ECCV 2022*, sous la dir. de Shai Avidan et al., Cham : Springer Nature Switzerland, 2022, p. 670-688, ISBN : 978-3-031-19830-4.

[Fot+12] Simon Fothergill et al., « Instructing People for Training Gestural Interactive Systems », in : *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '12, Association for Computing Machinery, 2012, p. 1737-1746.

[Gao+19] Mingfei Gao et al., « StartNet : Online Detection of Action Start in Untrimmed Videos », in : *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, 2019, p. 5541-5550, DOI : 10.1109/ICCV.2019.00564.

[Gao+21] Mingfei Gao et al., « WOAD : Weakly Supervised Online Action Detection in Untrimmed Videos », in : *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, juin 2021, p. 1915-1923.

[GE19] Yonatan Geifman et Ran El-Yaniv, « SelectiveNet : A Deep Neural Network with an Integrated Reject Option », in : *Proceedings of the 36th International Conference on Machine Learning*, sous la dir. de Kamalika Chaudhuri et Ruslan Salakhutdinov, t. 97, Proceedings of Machine Learning Research, PMLR, sept. 2019, p. 2151-2159.

[GK17] Guillermo Garcia-Hernando et Tae-Kyun Kim, « Transition Forests : Learning Discriminative Temporal Transitions for Action Recognition and Detection », in : *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, juill. 2017.

[Gon+12] Dian Gong et al., in : *Computer Vision – ECCV 2012*, sous la dir. d'Andrew Fitzgibbon et al., Berlin, Heidelberg : Springer Berlin Heidelberg, 2012, p. 229-243, isbn : 978-3-642-33712-3.

[Gra+06] Alex Graves et al., « Connectionist Temporal Classification : Labelling Unsegmented Sequence Data with Recurrent Neural Networks », in : *Proceedings of the 23rd International Conference on Machine Learning*, ICML '06, Pittsburgh, Pennsylvania, USA : Association for Computing Machinery, 2006, p. 369-376, isbn : 1595933832, doi : 10.1145/1143844.1143891, url : https://doi.org/10.1145/1143844.1143891.

[Gue+23] Mohammed Guermal et al., *JOADAA : joint online action detection and action anticipation*, 2023, arXiv : 2309.06130 [cs.CV].

[Guo+17] Chuan Guo et al., « On Calibration of Modern Neural Networks », in : *Proceedings of the 34th International Conference on Machine Learning*, sous la dir. de Doina Precup et Yee Whye Teh, t. 70, Proceedings of Machine Learning Research, PMLR, juin 2017, p. 1321-1330, url : https://proceedings.mlr.press/v70/guo17a.html.

[Guo+22] Hongji Guo et al., « Uncertainty-Based Spatial-Temporal Attention for Online Action Detection », in : *Computer Vision – ECCV 2022*, sous la dir. de Shai Avidan et al., Cham : Springer Nature Switzerland, 2022, p. 69-86, isbn : 978-3-031-19772-7.

[GZL21] Likun Gao, Heng Zhang et Cheng-Lin Liu, « Handwritten Text Recognition with Convolutional Prototype Network and Most Aligned Frame Based CTC Training », in : *Document Analysis and Recognition – ICDAR 2021*, sous la dir. de Josep Lladós, Daniel Lopresti et Seiichi Uchida, Cham : Springer International Publishing, 2021, p. 205-220, isbn : 978-3-030-86549-8.

[HD12] Minh Hoai et Fernando De la Torre, « Max-margin early event detectors », in : *2012 IEEE Conference on Computer Vision and Pattern Recognition*, 2012, p. 2863-2870, doi : 10.1109/CVPR.2012.6248012.

[Hou+18] Jingxuan HOU et al., « Spatial-Temporal Attention Res-TCN for Skeleton-based Dynamic Hand Gesture Recognition », in : *The European Conference on Computer Vision (ECCV) Workshops*, sept. 2018.

[Hou+20] Jingyi HOU et al., « Confidence-Guided Self Refinement for Action Prediction in Untrimmed Videos », in : *IEEE Transactions on Image Processing* 29 (2020), p. 6017-6031, DOI : 10.1109/TIP.2020.2987425.

[HS97] Sepp HOCHREITER et Jürgen SCHMIDHUBER, « Long Short-Term Memory », in : *Neural Computation* 9.*8* (1997), p. 1735-1780, DOI : 10.1162/neco.1997.9.8.1735.

[Hu+16] Jian-Fang HU et al., « Real-Time RGB-D Activity Prediction by Soft Regression », in : *Computer Vision – ECCV 2016*, sous la dir. de Bastian LEIBE et al., Cham : Springer International Publishing, 2016, p. 280-296, ISBN : 978-3-319-46448-0.

[Hu+19] Jian-Fang HU et al., « Early Action Prediction by Soft Regression », in : *IEEE Transactions on Pattern Analysis and Machine Intelligence* 41.*11* (2019), p. 2568-2583, DOI : 10.1109/TPAMI.2018.2863279.

[Hua+14] Dong HUANG et al., « Sequential Max-Margin Event Detectors », in : *Computer Vision – ECCV 2014*, sous la dir. de David FLEET et al., Cham : Springer International Publishing, 2014, p. 410-424, ISBN : 978-3-319-10578-9.

[Hus+13] Mohamed HUSSEIN et al., « Human Action Recognition Using a Temporal Hierarchy of Covariance Descriptors on 3D Joint Locations », in : *International Joint Conference on Artificial Intelligence (IJCAI)*, Beijing, China, août 2013.

[JN17] Zhenheng Yang JIYANG GAO et Ram NEVATIA, « RED : Reinforced Encoder-Decoder Networks for Action Anticipation », in : *Proceedings of the British Machine Vision Conference (BMVC)*, sous la dir. de Gabriel Brostow TAE-KYUN KIM Stefanos Zafeiriou et Krystian MIKOLAJCZYK, BMVA Press, sept. 2017, p. 92.1-92.11, ISBN : 1-901725-60-X, DOI : 10.5244/C.31.92, URL : https://dx.doi.org/10.5244/C.31.92.

[Joh73] Gunnar Johansson, « Visual perception of biological motion and a model for its analysis », in : *Perception & Psychophysics* 14.*2* (juin 1973), p. 201-211, ISSN : 1532-5962, DOI : 10.3758/BF03212378.

[KAK22] Sangwon Kim, Dasom Ahn et Byoung Chul Ko, *Cross-Modal Learning with 3D Deformable Attention for Action Recognition*, 2022, DOI : 10.48550/ARXIV.2212.05638, URL : https://arxiv.org/abs/2212.05638.

[Kaw+11] M. Kawashima et al., « Adaptive template method for early recognition of gestures », in : *17th Korea-Japan Joint Workshop on Frontiers of Computer Vision (FCV)*, 2011, p. 1-6, DOI : 10.1109/FCV.2011.5739719.

[KB17] Diederik P. Kingma et Jimmy Ba, *Adam : A Method for Stochastic Optimization*, 2017, arXiv : 1412.6980 [cs.LG].

[KB91] Gordon Kurtenbach et William Buxton, « Issues in Combining Marking and Direct Manipulation Techniques », in : *Proceedings of the 4th Annual ACM Symposium on User Interface Software and Technology*, UIST '91, Hilton Head, South Carolina, USA : Association for Computing Machinery, 1991, p. 137-144, ISBN : 0897914511, DOI : 10.1145/120782.120797.

[Ke+17] Qiuhong Ke et al., « A New Representation of Skeleton Sequences for 3D Action Recognition », in : *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, juill. 2017.

[Ke+20] Qiuhong Ke et al., « Learning Latent Global Network for Skeleton-Based Action Prediction », in : *IEEE Transactions on Image Processing* 29 (2020), p. 959-970, DOI : 10.1109/TIP.2019.2937757.

[KFS19] Qiuhong Ke, Mario Fritz et Bernt Schiele, « Time-Conditioned Action Anticipation in One Shot », in : *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, juin 2019.

[KKF14] Yu Kong, Dmitry Kit et Yun Fu, « A Discriminative Model with Multiple Temporal Scales for Action Prediction », in : *Computer Vision – ECCV 2014*, sous la dir. de David Fleet et al., Cham : Springer International Publishing, 2014, p. 596-611, ISBN : 978-3-319-10602-1.

[KNK21] Young Hwi Kim, Seonghyeon Nam et Seon Joo Kim, « Temporally smooth online action detection using cycle-consistent future anticipation », in : *Pattern Recognition* 116 (2021), p. 107954, ISSN : 0031-3203, DOI : https://doi.org/10.1016/j.patcog.2021.107954, URL : https://www.sciencedirect.com/science/article/pii/S0031320321001412.

[KNK22] Young Hwi Kim, Seonghyeon Nam et Seon Joo Kim, « 2PESNet : Towards online processing of temporal action localization », in : *Pattern Recognition* 131 (2022), p. 108871, ISSN : 0031-3203, DOI : https://doi.org/10.1016/j.patcog.2022.108871, URL : https://www.sciencedirect.com/science/article/pii/S0031320322003521.

[Ko+13] Yoshiyasu Ko et al., « Hash-Based Early Recognition of Gesture Patterns », in : *Artif. Life Robot.* 17.*3–4* (fév. 2013), p. 476-482, ISSN : 1433-5298, DOI : 10.1007/s10015-012-0085-6, URL : https://doi.org/10.1007/s10015-012-0085-6.

[Lam+16] Alex M Lamb et al., « Professor Forcing : A New Algorithm for Training Recurrent Networks », in : *Advances in Neural Information Processing Systems*, sous la dir. de D. Lee et al., t. 29, Curran Associates, Inc., 2016, URL : https://proceedings.neurips.cc/paper_files/paper/2016/file/16026d60ff9b54410b3435b403afd226-Paper.pdf.

[LAM19] Jian Liu, Naveed Akhtar et Ajmal Mian, « Skepxels : Spatio-temporal Image Representation of Human Skeleton Joints for Action Recognition », in : *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, juin 2019.

[Lar+17] Sohaib Laraba et al., « 3D skeleton-based action recognition by representing motion capture sequences as 2D-RGB images », in : *Computer Animation and Virtual Worlds* 28.*3-4* (2017), e1782 cav.1782, e1782, DOI : https://doi.org/10.1002/cav.1782, eprint : https://onlinelibrary.wiley.com/doi/pdf/10.1002/cav.1782, URL : https://onlinelibrary.wiley.com/doi/abs/10.1002/cav.1782.

[LB95] Yann Lecun et Y. Bengio, « Convolutional Networks for Images, Speech, and Time-Series », in : jan. 1995.

[LCS14] Tian Lan, Tsung-Chuan Chen et Silvio Savarese, « A Hierarchical Representation for Future Action Prediction », in : *Computer Vision – ECCV 2014*, sous la dir. de David Fleet et al., Cham : Springer International Publishing, 2014, p. 689-704, isbn : 978-3-319-10578-9.

[Lea+17] C. Lea et al., « Temporal Convolutional Networks for Action Segmentation and Detection », in : *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017, p. 1003-1012.

[LGH19] Ji Lin, Chuang Gan et Song Han, « TSM : Temporal Shift Module for Efficient Video Understanding », in : *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, oct. 2019.

[Li+12] Pei Yu Li et al., « Semi-customizable Gestural Commands Approach and Its Evaluation », in : *2012 International Conference on Frontiers in Handwriting Recognition*, 2012, p. 473-478, doi : 10.1109/ICFHR.2012.267.

[Li+16] Yanghao Li et al., « Online Human Action Detection Using Joint Classification-Regression Recurrent Neural Networks », in : *Computer Vision – ECCV 2016*, sous la dir. de Bastian Leibe et al., Cham : Springer International Publishing, 2016, p. 203-220, isbn : 978-3-319-46478-7.

[Li+19] Maosen Li et al., « Actional-Structural Graph Convolutional Networks for Skeleton-Based Action Recognition », in : *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)* (2019), p. 3590-3598.

[Li+20] Tianjiao Li et al., « HARD-Net : Hardness-AwaRe Discrimination Network for 3D Early Activity Prediction », in : *Computer Vision – ECCV 2020*, sous la dir. d'Andrea Vedaldi et al., Cham : Springer International Publishing, 2020, p. 420-436, isbn : 978-3-030-58621-8.

[Li+21] Yangke Li et al., « Compact joints encoding for skeleton-based dynamic hand gesture recognition », in : *Computers & Graphics* 97 (2021), p. 191-199, issn : 0097-8493, doi : https://doi.org/10.1016/j.cag.2021.04.017, url : https://www.sciencedirect.com/science/article/pii/S0097849321000595.

[Liu+17] Chunhui Liu et al., « PKU-MMD : A Large Scale Benchmark for Continuous Multi-Modal Human Action Understanding », in : *CoRR* abs/1703.07475 (2017), arXiv : 1703.07475, url : http://arxiv.org/abs/1703.07475.

[Liu+18a] Bangli Liu et al., « Online action recognition based on skeleton motion distribution », English, in : *Proceedings of the British Machine Vision Conference*, 29th British Machine Vision Conference, BMVC 2018 ; Conference date : 03-09-2018 Through 06-09-2018, British Machine Vision Association, sept. 2018, URL : http://bmvc2018.org/,%20http://bmvc2018.org/.

[Liu+18b] Kun Liu et al., « T-C3D : Temporal Convolutional 3D Network for Real-Time Action Recognition », in : *Proceedings of the AAAI Conference on Artificial Intelligence* 32.*1* (avr. 2018), DOI : 10.1609/aaai.v32i1.12333, URL : https://ojs.aaai.org/index.php/AAAI/article/view/12333.

[Liu+19] J. Liu et al., « Multi-Modality Multi-Task Recurrent Neural Network for Online Action Detection », in : *IEEE Transactions on Circuits and Systems for Video Technology* 29.*9* (sept. 2019), p. 2667-2682, ISSN : 1558-2205, DOI : 10.1109/TCSVT.2018.2799968.

[Liu+20a] J. Liu et al., « Skeleton-Based Online Action Prediction Using Scale Selection Network », in : *IEEE Transactions on Pattern Analysis and Machine Intelligence* 42.*6* (2020), p. 1453-1467, DOI : 10.1109/TPAMI.2019.2898954.

[Liu+20b] Ziyu Liu et al., « Disentangling and Unifying Graph Convolutions for Skeleton-Based Action Recognition », in : *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, juin 2020.

[Liu+22] Zhuang Liu et al., « A ConvNet for the 2020s », in : *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, juin 2022, p. 11976-11986.

[LJZ18] Hu Liu, Sheng Jin et Changshui Zhang, « Connectionist Temporal Classification with Maximum Entropy Regularization », in : *Advances in Neural Information Processing Systems*, sous la dir. de S. Bengio et al., t. 31, Curran Associates, Inc., 2018, URL : https://proceedings.neurips.cc/paper/2018/file/e44fea3bec53bcea3b7513ccef5857ac-Paper.pdf.

[LL19] Dong-Gyu Lee et Seong-Whan Lee, « Prediction of partially observed human activity based on pre-trained deep representation », in : *Pattern Recognition* 85 (2019), p. 198-206, ISSN : 0031-3203.

[Mar+21] Pierre-Etienne MARTIN et al., « Three-Stream 3D/1D CNN for Fine-Grained Action Classification and Segmentation in Table Tennis », in : *Proceedings of the 4th International Workshop on Multimedia Content Analysis in Sports*, MMSports'21, Virtual Event, China : Association for Computing Machinery, 2021, p. 35-41, ISBN : 9781450386708, DOI : 10.1145/3475722.3482793, URL : https://doi.org/10.1145/3475722.3482793.

[MHT16] Moustafa MESHRY, Mohamed E. HUSSEIN et Marwan TORKI, « Linear-time online action detection from 3D skeletal data using bags of gesturelets », in : *2016 IEEE Winter Conference on Applications of Computer Vision (WACV)*, 2016, p. 1-9, DOI : 10.1109/WACV.2016.7477587.

[Mle+19] Molefe Vicky MLEYA et al., « Online Aggregated-Event Representation for Multiple Event Detection in Videos », in : *Advanced Data Mining and Applications*, sous la dir. de Jianxin LI et al., Cham : Springer International Publishing, 2019, p. 501-515, ISBN : 978-3-030-35231-8.

[MM22] Sunah MIN et Jinyoung MOON, « Information Elevation Network for Online Action Detection and Anticipation », in : *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, juin 2022, p. 2550-2558.

[MND22] Nassim MOKHTARI., Alexis NÉDÉLEC. et Pierre DE LOOR., « Human Activity Recognition : A Spatio-temporal Image Encoding of 3D Skeleton Data for Online Action Detection », in : *Proceedings of the 17th International Joint Conference on Computer Vision, Imaging and Computer Graphics Theory and Applications - Volume 5 : VISAPP,* INSTICC, SciTePress, 2022, p. 448-455, ISBN : 978-989-758-555-5, DOI : 10.5220/0010835800003124.

[Mol+16] Pavlo MOLCHANOV et al., « Online Detection and Classification of Dynamic Hand Gestures With Recurrent 3D Convolutional Neural Network », in : *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, juin 2016, DOI : 10.1109/CVPR.2016.456.

[Mou07] Harold MOUCHÈRE, « Etude des mécanismes d'adaptation et de rejet pour l'optimisation de classifieurs : Application à la reconnaissance de l'écriture manuscrite en-ligne », 2007ISAR0027, thèse de doct., 2007, 1 vol. 209 p. URL : http://www.theses.fr/2007ISAR0027/document.

[MSS16] Shugao Ma, Leonid Sigal et Stan Sclaroff, « Learning Activity Progression in LSTMs for Activity Detection and Early Detection », in : *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016, p. 1942-1950, doi : 10.1109/CVPR.2016.214.

[NS12] Sebastian Nowozin et Jamie Shotton, *Action Points : A Representation for Low-latency Online Human Action Recognition*, rapp. tech. MSR-TR-2012-68, Microsoft Research Cambridge, juill. 2012.

[Núñ+18] Juan C. Núñez et al., « Convolutional Neural Networks and Long Short-Term Memory for skeleton-based human activity and hand gesture recognition », in : *Pattern Recognition* 76 (2018), p. 80-94, issn : 0031-3203, doi : https://doi.org/10.1016/j.patcog.2017.10.033.

[Oor+16] Aäron van den Oord et al., « WaveNet : A Generative Model for Raw Audio », in : *CoRR* (2016), arXiv : 1609.03499.

[PCM21] Chiara Plizzari, Marco Cannici et Matteo Matteucci, « Spatial Temporal Transformer Network for Skeleton-Based Action Recognition », in : *Pattern Recognition. ICPR International Workshops and Challenges*, sous la dir. d'Alberto Del Bimbo et al., Cham : Springer International Publishing, 2021, p. 694-701, isbn : 978-3-030-68796-0.

[PM13] Eric Petit et Christophe Maldivi, « Unifying gestures and direct manipulation in touchscreen interfaces », in : (déc. 2013).

[Qin+22] Hushan Qin et al., « Structure-Preserving View-Invariant Skeleton Representation for Action Detection », in : *2022 26th International Conference on Pattern Recognition (ICPR)*, 2022, p. 3190-3196, doi : 10.1109/ICPR56361.2022.9956485.

[RBA21] Mohamed Lamine Rouali, Said Yacine Boulahia et Abdenour Amamra, « Simultaneous Temporal and Spatial Deep Attention for Imaged Skeleton-Based Action Recognition », in : PRIS '21, Bangkok, Thailand : Association for Computing Machinery, 2021, p. 77-80, isbn : 9781450390392, doi : 10.1145/3480651.3480668, url : https://doi.org/10.1145/3480651.3480668.

[Ren+12] N. Renau-Ferrer et al., « The ILGDB database of realistic pen-based gestural commands », in : *Proceedings of the 21st International Conference on Pattern Recognition (ICPR2012)*, 2012, p. 3741-3744.

[Ryo11] M. S. Ryoo, « Human activity prediction : Early recognition of ongoing activities from streaming videos », in : *2011 International Conference on Computer Vision*, 2011, p. 1036-1043, doi : 10.1109/ICCV.2011.6126349.

[SFH18] Yuge Shi, Basura Fernando et Richard Hartley, « Action Anticipation with RBF Kernelized Feature Mapping RNN », in : *Proceedings of the European Conference on Computer Vision (ECCV)*, sept. 2018.

[Sha+15] A. Sharaf et al., « Real-Time Multi-scale Action Detection from 3D Skeleton Data », in : *2015 IEEE Winter Conference on Applications of Computer Vision*, jan. 2015, p. 998-1005, doi : 10.1109/WACV.2015.138.

[Shi+19] L. Shi et al., « Two-Stream Adaptive Graph Convolutional Networks for Skeleton-Based Action Recognition », in : *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019, p. 12018-12027.

[Shi+20] Lei Shi et al., *What and Where : Modeling Skeletons from Semantic and Spatial Perspectives for Action Recognition*, 2020, doi : 10.48550/ARXIV.2004.03259, url : https://arxiv.org/abs/2004.03259.

[Sho+11] J. Shotton et al., « Real-time human pose recognition in parts from single depth images », in : *CVPR 2011*, juin 2011, p. 1297-1304, doi : 10.1109/CVPR.2011.5995316.

[Sho+18] Zheng Shou et al., « Online Detection of Action Start in Untrimmed, Streaming Videos », in : *Proceedings of the European Conference on Computer Vision (ECCV)*, sept. 2018.

[Sin+17] Gurkirt Singh et al., « Online Real-Time Multiple Spatiotemporal Action Localisation and Prediction », in : *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, oct. 2017.

[Sou+19] Yann Soullard et al., « Improving text recognition using optical and language model writer adaptation », in : *2019 International Conference on Document Analysis and Recognition (ICDAR)*, IEEE, 2019, p. 1175-1180.

[Sun+19] Chen Sun et al., « Relational Action Forecasting », in : *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019, p. 273-283, DOI : 10.1109/CVPR.2019.00036.

[Sun+23] Zehua Sun et al., « Human Action Recognition From Various Data Modalities : A Review », in : *IEEE Transactions on Pattern Analysis and Machine Intelligence* 45.*3* (2023), p. 3200-3225, DOI : 10.1109/TPAMI.2022.3183112.

[SY22] Mehrin Saremi et Farzin Yaghmaee, « Improved use of descriptors for early recognition of actions in video », in : *Multimedia Tools and Applications* 82 (juill. 2022), p. 1-17, DOI : 10.1007/s11042-022-13316-x.

[SZ15] Karen Simonyan et Andrew Zisserman, *Very Deep Convolutional Networks for Large-Scale Image Recognition*, 2015, arXiv : 1409.1556 [cs.CV].

[TLL18] J. Tu, M. Liu et H. Liu, « Skeleton-Based Human Action Recognition Using Spatial Temporal 3D Convolutional Neural Networks », in : *2018 IEEE International Conference on Multimedia and Expo (ICME)*, 2018, p. 1-6.

[Tra+15] D. Tran et al., « Learning Spatiotemporal Features with 3D Convolutional Networks », in : *IEEE International Conference on Computer Vision (ICCV)*, 2015, p. 4489-4497, DOI : 10.1109/ICCV.2015.510.

[UA08] S. Uchida et K. Amamoto, « Early recognition of sequential patterns by classifier combination », in : *19th International Conference on Pattern Recognition*, 2008, p. 1-4, DOI : 10.1109/ICPR.2008.4761137.

[VKM21] Vasiliki I. Vasileiou, Nikolaos Kardaris et Petros Maragos, « Exploring Temporal Context and Human Movement Dynamics for Online Action Detection in Videos », in : *2021 29th European Signal Processing Conference (EUSIPCO)*, 2021, p. 1431-1435, DOI : 10.23919/EUSIPCO54536.2021.9616092.

[VLS18] Gül Varol, Ivan Laptev et Cordelia Schmid, « Long-Term Temporal Convolutions for Action Recognition », in : *IEEE Transactions on Pattern Analysis and Machine Intelligence* 40.*6* (2018), p. 1510-1517, DOI : 10.1109/TPAMI.2017.2712608.

[Wan+16] Limin WANG et al., « Temporal Segment Networks : Towards Good Practices for Deep Action Recognition », in : *Computer Vision – ECCV 2016*, sous la dir. de Bastian LEIBE et al., Cham : Springer International Publishing, 2016, p. 20-36, ISBN : 978-3-319-46484-8.

[Wan+18a] Pichao WANG et al., « Action recognition based on joint trajectory maps with convolutional neural networks », in : *Knowledge-Based Systems* 158 (2018), p. 43-53, ISSN : 0950-7051, DOI : https://doi.org/10.1016/j.knosys.2018.05.029, URL : https://www.sciencedirect.com/science/article/pii/S0950705118302582.

[Wan+18b] Xiaolong WANG et al., « Non-local Neural Networks », in : *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2018, p. 7794-7803, DOI : 10.1109/CVPR.2018.00813.

[Wan+19] Xionghui WANG et al., « Progressive Teacher-Student Learning for Early Action Prediction », in : *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019, p. 3551-3560, DOI : 10.1109/CVPR.2019.00367.

[Wan+21] Xiang WANG et al., « OadTR : Online Action Detection With Transformers », in : *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, oct. 2021, p. 7565-7575.

[Wan+22] Wen WANG et al., « An Empirical Study on Temporal Modeling for Online Action Detection », in : *Complex & Intelligent Systems* 8.*2* (avr. 2022), p. 1803-1817, ISSN : 2198-6053, DOI : 10.1007/s40747-021-00534-3, URL : https://doi.org/10.1007/s40747-021-00534-3.

[Wan+23a] Rui WANG et al., « Dear-Net : Learning Diversities for Skeleton-Based Early Action Recognition », in : *IEEE Transactions on Multimedia* 25 (2023), p. 1175-1189, DOI : 10.1109/TMM.2021.3139768.

[Wan+23b] Wenqian WANG et al., « Magi-Net : Meta Negative Network for Early Activity Prediction », in : *IEEE Transactions on Image Processing* 32 (2023), p. 3254-3265, DOI : 10.1109/TIP.2023.3279991.

[Web+14] M. WEBER et al., « LSTM-Based Early Recognition of Motion Patterns », in : *2014 22nd International Conference on Pattern Recognition*, 2014, p. 3552-3557, DOI : 10.1109/ICPR.2014.611.

[Woo+23] Sanghyun Woo et al., « ConvNeXt V2 : Co-Designing and Scaling ConvNets With Masked Autoencoders », in : *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, juin 2023, p. 16133-16142.

[WS14] Di Wu et Ling Shao, « Leveraging Hierarchical Parametric Networks for Skeletal Joints Based Action Segmentation and Recognition », in : *2014 IEEE Conference on Computer Vision and Pattern Recognition*, 2014, p. 724-731, DOI : 10.1109/CVPR.2014.98.

[WYY19] Shiye Wang, Zhezhou Yu et Xiangchun Yu, « Real-time online action detection and segmentation using improved efficient linear search », in : *International Journal of Computing Science and Mathematics* 10.*2* (2019), p. 129-139, DOI : 10.1504/IJCSM.2019.098738.

[Xu+19a] Mingze Xu et al., « Temporal Recurrent Networks for Online Action Detection », in : *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, oct. 2019.

[Xu+19b] Wanru Xu et al., « Prediction-CGAN : Human Action Prediction with Conditional Generative Adversarial Networks », in : *Proceedings of the 27th ACM International Conference on Multimedia*, MM '19, Nice, France : Association for Computing Machinery, 2019, p. 611-619, ISBN : 9781450368896, DOI : 10.1145/3343031.3351073.

[Xu+20] Yongyang Xu et al., « Action Recognition Using High Temporal Resolution 3D Neural Network Based on Dilated Convolution », in : *IEEE Access* 8 (2020), p. 165365-165372, DOI : 10.1109/ACCESS.2020.3022407.

[Xu+21] Mingze Xu et al., « Long Short-Term Transformer for Online Action Detection », in : *Advances in Neural Information Processing Systems*, sous la dir. de M. Ranzato et al., t. 34, Curran Associates, Inc., 2021, p. 1086-1099, URL : https://proceedings.neurips.cc/paper_files/paper/2021/file/08b255a5d42b89b0585260b6f2360bdd-Paper.pdf.

[Yan+20] Fan Yang et al., « Make Skeleton-Based Action Recognition Model Smaller, Faster and Better », in : *Proceedings of the ACM Multimedia Asia*, MMAsia '19, Beijing, China : Association for Computing Machinery, 2020, ISBN : 9781450368414, URL : https://doi.org/10.1145/3338533.3366569.

[Yan+21] Zichen YANG et al., « Human-Aware Coarse-to-Fine Online Action Detection », in : *ICASSP 2021 - 2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2021, p. 2455-2459, DOI : 10.1109/ICASSP39728.2021.9413368.

[YCL20] Da-Hye YOON, Nam-Gyu CHO et Seong-Whan LEE, « A novel online action detection framework from untrimmed video streams », in : *Pattern Recognition* 106 (2020), p. 107396, ISSN : 0031-3203, DOI : https://doi.org/10.1016/j.patcog.2020.107396, URL : https://www.sciencedirect.com/science/article/pii/S0031320320301990.

[YXL18] Sijie YAN, Yuanjun XIONG et Dahua LIN, « Spatial Temporal Graph Convolutional Networks for Skeleton-Based Action Recognition », in : *Proceedings of the AAAI Conference on Artificial Intelligence* 32.*1* (avr. 2018), DOI : 10.1609/aaai.v32i1.12328, URL : https://ojs.aaai.org/index.php/AAAI/article/view/12328.

[Zha+13] Xin ZHAO et al., « Online Human Gesture Recognition from Motion Data Streams », in : *Proceedings of the 21st ACM International Conference on Multimedia*, MM '13, Barcelona, Spain : Association for Computing Machinery, 2013, p. 23-32, ISBN : 9781450324045, DOI : 10.1145/2502081.2502103, URL : https://doi.org/10.1145/2502081.2502103.

[Zha+14] Xin ZHAO et al., « Structured Streaming Skeleton – A New Feature for Online Human Gesture Recognition », in : *ACM Transactions on Multimedia Computing, Communications, and Applications* 11 (oct. 2014), p. 1-18, DOI : 10.1145/2648583.

[Zha+19] P. ZHANG et al., « View Adaptive Neural Networks for High Performance Skeleton-Based Human Action Recognition », in : *IEEE Transactions on Pattern Analysis and Machine Intelligence* 41.*8* (août 2019), p. 1963-1978, ISSN : 1939-3539, DOI : 10.1109/TPAMI.2019.2896631.

[Zha+21] Yuhan ZHANG et al., « STST : Spatial-Temporal Specialized Transformer for Skeleton-Based Action Recognition », in : *Proceedings of the 29th ACM International Conference on Multimedia*, MM '21, Virtual Event, China : Association for Computing Machinery, 2021, p. 3229-3237, ISBN : 9781450386517, DOI : 10.1145/3474085.3475473, URL : https://doi.org/10.1145/3474085.3475473.

[Zha+22] Peisen Zhao et al., « Progressive privileged knowledge distillation for online action detection », in : *Pattern Recognition* 129 (2022), p. 108741, ISSN : 0031-3203, DOI : https://doi.org/10.1016/j.patcog.2022.108741, URL : https://www.sciencedirect.com/science/article/pii/S0031320322002229.

[Zho+22] Chongyang Zhong et al., « Spatio-Temporal Gating-Adjacency GCN for Human Motion Prediction », in : *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, juin 2022, p. 6447-6456.

[ZK22] Yue Zhao et Philipp Krähenbühl, « Real-Time Online Video Detection with Temporal Smoothing Transformers », in : *Computer Vision – ECCV 2022*, sous la dir. de Shai Avidan et al., Cham : Springer Nature Switzerland, 2022, p. 485-502, ISBN : 978-3-031-19830-4.

[ZLS13] Mihai Zanfir, Marius Leordeanu et Cristian Sminchisescu, « The Moving Pose : An Efficient 3D Kinematics Descriptor for Low-Latency Action Recognition and Detection », in : *2013 IEEE International Conference on Computer Vision*, 2013, p. 2752-2759, DOI : 10.1109/ICCV.2013.342.

[ZMV20] Ting Zhang, Harold Mouchère et Christian Viard-Gaudin, « A tree-BLSTM-based recognition system for online handwritten mathematical expressions », in : *Neural Computing and Applications* 32.*9* (mai 2020), p. 4689-4708, ISSN : 1433-3058, DOI : 10.1007/s00521-018-3817-2, URL : https://doi.org/10.1007/s00521-018-3817-2.

[ZSN21] Albert Zeyer, Ralf Schlüter et Hermann Ney, « Why does CTC result in peaky behavior ? », in : *CoRR* abs/2105.14849 (2021), arXiv : 2105.14849, URL : https://arxiv.org/abs/2105.14849.

LISTE DES FIGURES

Liste des tableaux

COLLEGE MATHS, TELECOMS
DOCTORAL INFORMATIQUE, SIGNAL
BRETAGNE SYSTEMES, ELECTRONIQUE

INSA INSTITUT NATIONAL DES SCIENCES APPLIQUÉES RENNES

Titre : Réseaux de Neurones à Convolution Spatio-Temporelle pour l'Analyse et la Reconnaissance Précoce d'Actions et de Gestes.

Mot clés : Reconnaissance de gestes, Interaction homme-machine, Reconnaissance précoce de gestes, Détection en ligne, Décision, Réseaux à Convolution, CTC

Résumé : Ce travail de recherche se concentre sur la reconnaissance précoce de gestes dans le domaine de l'interaction homme-machine. Il s'inscrit dans le cadre d'une collaboration entre deux équipes de recherche : ShaDoc, spécialiste de la reconnaissance de document et du geste 2D et MimeTic, experte en analyse du mouvement humain. Il aborde un défi complexe consistant à développer une approche polyvalente pour la reconnaissance à la fois de gestes 2D effectués sur tablette et gestes 3D effectués par le corps humain. Pour des besoins de fluidité d'interaction et de réactivité dans les deux domaines, le défi principal est de parvenir à reconnaître ces gestes au plus tôt, si possible avant qu'ils ne soient terminés.

Les contributions s'inscrivent sur trois piliers : la représentation du geste, la réalisation d'un système de reconnaissance à base de réseaux profonds, et la conception d'un mécanisme de décision. Ces trois éléments sont coordonnés au sein d'un système capable de reconnaître un geste en cours de manière précoce, mais aussi de ne pas prendre de décision tant qu'un geste n'est pas reconnaissable du fait d'une ambiguïté entre plusieurs gestes.

Ces approches se sont avérées performantes lors des évaluations, à la fois dans le contexte segmenté sur le geste 2D, et dans le contexte non-segmenté sur le geste 2D et 3D. Les résultats et expérimentations de cette recherche démontrent la pertinence de ces approches pour les systèmes interactifs en temps réel.

Title: Convolutional Spatio-Temporal Neural Networks for the Analysis and Early Recognition of Actions and Gestures.

Keywords: Gesture recognition, Human-machine interaction, Early Gesture recognition, Online detection, Decision, Convolutional networks, CTC

Abstract: This research work focuses on the early recognition of gestures in the field of human-machine interaction. It is part of a collaboration between two research teams: ShaDoc, specializing in document and 2D gesture recognition, and MimeTic, experts in human motion analysis. The primary challenge addressed in this study is the development of a versatile approach for recognizing both 2D gestures performed on a tablet and 3D gestures executed by the human body. To ensure smooth interaction and responsiveness in both domains, the main goal is to recognize these gestures as early as possible, ideally before they are completed.

The contributions of this research are structured around three axes: gesture representation, the implementation of a deep learning-based recognition system, and the design of a decision mechanism. These three components work together within a system capable of recognizing a gesture in progress early, while also refraining from making a decision until a gesture becomes distinguishable due to ambiguity between multiple gestures.

These approaches proved to be effective in evaluations, both in the trimmed context for 2D gestures and in the untrimmed context for 2D and 3D gestures. The results and experiments of this research demonstrate the relevance of these approaches for real-time interactive systems.