République Algérienne Démocratique et Populaire
Ministère de l'Enseignement Supérieur et de la Recherche Scientifique
Université Saad Dahlab de Blida 1

Faculté des sciences

Département d'Informatique

En vue d'obtenir le diplôme de master

Domaine : Mathématique et informatique

Filière : Informatique

Spécialité : Ingénierie des logiciels

Application De L'apprentissage Machine Pour La Reconnaissance Des Activités Humaines

Présenté par :

- Moussa Ikram
- Anine Safaa

Encadreur : Dr. Kahlouche Souhila

Promoteur: Dr. Bacha Sihem

Soutenu le :01/10/2019

Devant le jury :

-Mme Yekhlef	**Président**
-Mme Bey	**Examinateur**
-Mme.Kahlouche Souhila	**Encadrant**
-Mme.Bacha Sihem	**Promoteur**

Année universitaire : 2018/2019

Remerciement

Nous tenons à remercier notre encadreur Mme KAHLOUCHE qui nous a guidé et fait profiter de ses expériences tout au long de notre travail.

Un grand merci à notre promotrice Mme BACHA a sa disponibilité, ces recommandations, sa confiance et son aide dans la correction de ce mémoire, ainsi nos enseignantes.

Je tiens aussi à remercier l'ensemble des enseignants de département Informatique pour les efforts qu'ils ont déployé pour assurer notre formation.

Nous remercions les membres du jury pour avoir accepté d'évaluer notre modeste travail.

Enfin, nous adressons nos remerciements les plus sincères à nos familles, tous nos amis ainsi que toute personne ayant contribué de près ou de loin à la réalisation de ce projet.

En espérant que ce modeste travail soit à la hauteur et reflète ce que nous avons pu acquérir pendant ce travail.

Dédicace

Je dédie ce modeste travail :

À mes étoiles qui ont éclair ma vie depuis ma naissance jusqu'à aujourd'hui et ils resteront.

*À ma tendre mère **FOUZIA** et mon cher père **DJAAFER** pour leurs aides, leurs conseils, leur éternel soutien moral et financier pour m'avoir toujours montré le meilleur d'existante, je leurs souhaites une longue vie.*

*À mon très cher frère **KARIM**, À ma très chère sœur **RANIA**, sont le pivot de ma vie.*

*À toutes mes amies d'enfance et du long parcours scolaire et universitaire surtout mon binôme :**Irkam***

*À mes deux chers **Lyna** et **Z.RIADH** qu'ils ont été toujours à mes côtés.*

*À tous mes collègues de la promotion surtout : **Hadjer, Habiba, Sara, Imed, Chakib.***

*À tous mes oncles et tantes, mes cousins et cousines. En particulier ma cousine : **Fatiha.***

À tous ceux et celles que j'ai oublié de les citer mais ils sont toujours dans ma mémoire.

À tous mes enseignants /enseignantes de l'école primaire jusqu'à l'université.

SAFAA

Dédicace

A ***mon père*** *sans qui je ne serai jamais là où je suis maintenant. A* ***ma mère*** *qui a sacrifié ses ambitions pour notre réussite. A* ***Mes frères*** *et* ***ma sœur*** *qui n'ont jamais cessé de me supporter et de croire en moi. A mes chers amis* ***Sara,Habiba*** *pour la meilleure compagnie qui puisse être. A mes amis d'enfance,* ***Rabiaa*** *et* ***Houria****. A ma cherie* ***Soumia*** *qui m'a aidé dans plusieurs choses*
.A mon binôme ***Safa****, pour une amitié éternelle inch'Allah.*

Ikram

Résumé

La reconnaissance des activités humaines(RAH)constitue une tâche clé pour une Interaction Homme robot intelligente et intuitive. En effet, dans un scénario d'interaction sociale entre un robot et un humain, le robot doit pouvoir détecter les humains, les poursuivre, les guider, et aussi reconnaitre leurs gestes et leurs actions liés à cette interaction.

Le travail demandé concerne l'application des techniques d'apprentissage machine (Machine learning) pour reconnaitre les activités des humains afin d'inférer leurs intentions et de les aider à atteindre leurs objectifs.

Pour cela nous avons implémenté une solution de classification basée essentiellement sur des méthodes existantes, connues pour leur efficacité en reconnaissance des activités humaines

Cette dernière est basée sur les classifieurs suivants Machine a vecteur support (SVM), K plus proches voisins (KNN) et Arbre de décision (Decision Tree). Ces algorithmes ont été implémentés et testés sur plusieurs DataSets dans le but de les intégrer en temps réel sur le robot B12r du CDTA. Ensuite, pour permettre une meilleure généralisation des modèles générées par les algorithmes cités ci-dessus, nous avons implémenté un descripteur discriminant basé sur l'histogramme des vecteurs de déplacement, permettent de classifier les activités de manière plus correcte.

Nous allons présenter en temps réel sur le robot B12r l'implémentation des algorithmes de Machine Learning.

Mots-clés : Reconnaissance d'activités humaines, Apprentissage machine, Robotique, Classification de données.

Abstract

Human Activity Recognition (RAH) is a key task for intelligent and intuitive robot human interaction. Indeed, in a scenario of social interaction between a robot and a human, the robot must be able to detect humans, pursue them, guide them, and recognize their actions and activities related to this interaction.

The work involved is the application of machine learning techniques to recognize human activities in order to infer their intentions and help them to achieve their goals.

For this, we have implemented a classification solution based essentially on existing methods, known for their effectiveness in recognizing human activities

The latter is based on the following classifiers Support Vector Machine (SVM), K nearest neighbors (KNN) and DecisionTree (TREE).These algorithms have been implemented and tested on several datasets for the purpose of integrated in real time on the CDTA robot.

Then, to allow a bette rgeneralization of the models generated by the algorithms mentioned above, we implemented a discriminant descriptor to classify the activities in a more correct way

We will present in real time on the robot B12r the algorithms of Machine Learning.

Keywords: Recognition of human activities, Machine learning, Robotics, Data classification.

Table des matières

Liste des Figures

Liste des tableaux

Liste des abréviations

CDTA	Centre de Développement des Technologies avancées
RAH	Reconnaissance des Activités Humaines
IHM	Interaction Homme-Machine
SVM	Support Vector Machine
KNN	k Nearest Neighbors

Introduction générale

Introduction générale

1. Contexte et Motivations

La reconnaissance automatique des activités physiques - communément appelée reconnaissance de l'activité humaine (RAH) - est devenue un domaine de recherche clé en interaction homme-machine (IHM) et en informatique mobile. L'objectif principal de la reconnaissance d'activité consiste à fournir des informations sur le comportement d'un utilisateur ; ce qui permet aux systèmes informatiques d'assister les utilisateurs de manière proactive dans leurs tâches [1]. Un grand nombre de chercheurs ont enquêté sur la reconnaissance automatique des gestes et des activités à partir d'images fixes et d'environnements vidéos non encombrés ou fixes.

Par ailleurs la reconnaissance de l'activité a récemment fait ses débuts en tant que composant clé de plusieurs produits de consommation. Par exemple, les consoles de jeux telles que la Nintendo Wii et le Microsoft Kinect s'appuient sur la reconnaissance de gestes ou même de mouvements de tout le corps pour changer radicalement l'expérience de jeu. Bien qu'ils aient été développés à l'origine pour le secteur des loisirs, ces systèmes ont trouvé d'autres applications, telles que l'entrainement personnel et la rééducation, et ont également stimulé de nouvelles recherches sur la reconnaissance des activités [2]. Enfin, certains produits sportifs tels que les chaussures de course Philips Direct Life ou les chaussures de course Nike + intègrent des capteurs de mouvement et offrent aux athlètes amateurs et professionnels un retour d'informations sur leurs performances. Tous ces exemples soulignent l'importance de la reconnaissance de l'activité humaine dans les milieux universitaires et industriels, Il faut donc fournir des robots cognitifs, qui revêtent une grande importance aujourd'hui en raison du large éventail d'applications induites et des défis scientifiques et technologiques fondamentaux. Alors que la première génération de robots avait été conçue pour des tâches industrielles répétitives et spécifiques, la nouvelle génération présente les robots comme des partenaires aux humains dotés de capacités cognitives, dont l'objectif n'est pas de remplacer les humains, mais plutôt de les assister de manière collaborative. Ces robots peuvent, par exemple, communiquer et faire des tâches similaires aux humains. Cette tendance a été accentuée par la prolifération spectaculaire des capteurs innovants et à bas cout. Par ailleurs l'augmentation des capacités des processeurs, l'amélioration des techniques de traitement du signal, des images, et la maturité du domaine de reconnaissance de formes et d'apprentissage statistique ont permis une amélioration de la prise de décisions sous différents niveaux d'incertitude.

Pour améliorer l'interaction homme-robot, il est fondamental de doter le robot de capacités sensorielles naturelles : voix, vision, toucher, etc. Alors que plusieurs travaux sur les robots interagissant par la voix ont été effectués, les efforts dédiés à la robotique par vision ont été bien moins nombreux et restreints à des tâches très limitées. Ce décalage peut être expliqué par les capacités de processeurs de la génération précédente qui étaient plus adaptées à des données de type voix, bien moins gourmandes que les images et surtout les séquences vidéos. Les algorithmes d'apprentissage automatique étaient (et sont toujours) aussi bien plus matures pour la reconnaissance de parole que pour la reconnaissance d'objets et d'activités humaines dans des scènes complexes. La robotique par la vision permet d'autres applications importantes telles que la communication par le geste pour des personnes ayant des troubles de la parole et du langage, ou l'interaction homme-robot pour la stimulation d'exercices physiques, la réhabilitation et le divertissement.

Toutefois, ce type de solutions nécessite le passage par plusieurs phases partant de la collecte de données à la prise de décision. Une étape très importante et critique dans ce processus est la « Classification des données », elle exige une grande précision afin d'assurer l'identification de la vraie activité et non pas une autre, et ceci vu la sensibilité du système et l'impact des résultats sur les prises de décisions.

Notre part de contribution dans ce projet est de jouer sur des combinaisons d'algorithmes de l'apprentissage automatique reconnus par leur efficacité dans la classification et d'implémenter la ou les meilleures solutions, cette dernière sera le cœur d'un système complet de reconnaissance des activités humaines pour un robot.

2. Problématique

Malgré des progrès considérables dans la déduction d'activités à partir de capteurs et le déploiement de systèmes de reconnaissance d'activités, la tâche de la reconnaissance des activités humaines reste problématique, il n'existe pas de définition commune, de langage ou de structure d'activités humaines nous permettant de formuler un énoncé de problème clair et commun (quelle activité doit être reconnue ?, comment une activité spécifique est caractérisée ?, etc.). L'un des objectifs majeurs de la RAH est de reconnaitre et de comprendre le comportement humain, et notamment de permettre la classification des activités. Cependant, la classification des données repose sur des modèles d'apprentissage automatique. Les types de modèles d'apprentissage varient en fonction de plusieurs facteurs, par exemple, les capteurs

utilisés pour la collecte des données. De ce fait, un nombre considérable d'algorithmes d'apprentissage supervisé et non supervisé de l'état de l'art ont été développés.
Dans ce travail, nous avons tenté d'identifier un certain nombre d'activités humaines (Arrêter, Appeler, Saluer, Venir, Partir, Pointer vers) dans le but d'exploiter la capacité du robot à apprendre ces activités pour interagir avec les personnes.

3. Objectifs

Notre travail, au sein du Centre de Développement des Technologies Avancées – CDTA, a pour objectif de développer des méthodes de classification pour le processus de reconnaissance d'activité humaine pour un robot dans le but d'améliorer les performances d'interaction avec les humains. Pour ce faire, nous avons procédé comme suit :

- Étude bibliographique sur l'apprentissage machine pour la reconnaissance des activités humaines.
- Proposition d'une technique intelligente pour la reconnaissance des activités humaines.
- Comparaison entre plusieurs algorithmes de classification.
- Implémentation d'un descripteur discriminant permettent de classifier les activités de manière plus correcte.
- Évaluation des performances des algorithmes proposés.
- Déploiement du système de reconnaissance sur le robot B21R.

4. Plan du document

Nous avons organisé notre document comme suit :

— Chapitre 1 : Présentant des généralités sur la reconnaissance d'activités humaines.

— Chapitre 2 : est consacré à l'apprentissage machine pour la reconnaissance des activités humaines et les déférents types utilisés pour la classification.

— Chapitre 3 : dédié à la présentation de notre solution conceptuelle

— Chapitre 4 : Dédié à l'évaluation des performances, ainsi que les différents résultats obtenus Enfin, nous conclurons par un rappel des principaux résultats et les perspectives en vigueur pour le projet.

Généralités sur la reconnaissance des activités humaines

Chapitre 1 : Généralités sur la reconnaissance d'activités humaines

1.1 Introduction

La reconnaissance d'activité est un domaine de recherche très actif qui peut être appliqué à de nombreux problèmes de la vie réelle, telle que l'assistance aux personnes âgées, la vidéo surveillance et le gaming, …

La reconnaissance des activités humaines peut aller de l'action simple (comme Salut), jusqu'aux activités plus complexes (composées de plusieurs actions), jusqu'aux activités d'interaction entre plusieurs personnes ou avec des objets.

1.2 Activité humaine

Une activité est une tâche de la vie quotidienne que la personne effectue sur un intervalle de temps donné [3]. Dans la littérature, sept groupes d'activités peuvent être distingués selon le domaine. Ils sont résumés dans le tableau 1. Il est à noter que les ensembles ne sont pas disjoints. C'est-à-dire une activité peut faire partie de deux ensembles à la fois.

Certaines de ces activités sont simples (par exemple marcher), d'autres sont complexes (par exemple faire du shopping). Ces dernières sont une agrégation d'une suite de tâches élémentaires. On distingue également les activités relatives à la santé qui comprennent : l'exercice physique, la chute, la rééducation/réhabilitation et le suivi des régimes/routines.

Groupe	Activités
Ambulation	Marcher, courir, s'assoir, être debout, être allongé, monter et descendre un escalier, un escalier mécanique ou un ascenseur
Transportation	Monter un bus, faire du vélo et conduire.
Utilisation du téléphone	Envoyer un message, passer un appel.
Activités quotidiennes	Manger, boire, travailler sur PC, regarder la TV, lire, se brosser les dents, s'étirer, frotter, passer l'aspirateur.
Exercice physique	Faire de l'aviron, soulever des poids, tourner, marche nordique, faire des pompes.
Militaire	Ramper, s'agenouiller, évaluer la situation et ouvrir une porte.
Haut du corps	Mâcher, parler, avaler, soupirer et bouger la tête.

Tableau 1:Types d'activités reconnues dans la littérature [4]

1.3 La reconnaissance d'activités humaines

La reconnaissance d'activités humaines est une branche de l'informatique visant à définir des méthodes et des algorithmes, afin de modéliser les activités de la vie quotidienne des êtres humains. Ces modèles permettent la conception de systèmes capables d'observer et de comprendre de manière automatique et autonome, l'activité réalisée par une personne dans un environnement surveillé afin de répondre au mieux à ses besoins et/ou lui apporter assistance de manière proactive à travers la connaissance relative à son comportement [5].

La RAH est aujourd'hui un axe de recherche en pleine effervescence en raison des nombreux domaines d'application potentiels. Un axe de recherche étant toujours mieux décrit par la question à laquelle il essaye de répondre [6], on pourrait formuler la question correspondant à la reconnaissance d'activités comme suit :

« Étant donné une séquence d'images (ou un ensemble d'informations capturées dans un environnement) avec une ou plusieurs personnes effectuant une activité, peut-on concevoir un système qui pourrait automatiquement reconnaitre quelle activité est ou était réalisée ? » [7].

Ainsi la reconnaissance d'activités est le processus par lequel le comportement d'un acteur et l'environnement dans lequel il évolue sont surveillés et analysés afin de déduire et de reconnaitre les activités réalisées. Il comprend plusieurs tâches relatives à la modélisation de

l'activité, la surveillance et le suivi de l'environnement, le traitement des données et la reconnaissance des schémas1 [8]. Selon « Liming Chen » et « Ismail Khalil » dans [8]. Afin de réaliser un système de reconnaissance d'activité, il est nécessaire de :

1. Créer un modèle informatique d'activités pour permettre à des agents logiciels de conduire un raisonnement et manipuler les informations recueillies.
2. Surveiller et capturer le comportement de l'utilisateur ainsi que les changements opérant sur son environnement.
3. Traiter les informations perçues à travers la fusion et l'agrégation des données pour générer un haut niveau d'abstraction du contexte ou de la situation.
4. Décider de l'approche et de l'algorithme de reconnaissance d'activités à utiliser.
5. Effectuer une classification afin de déterminer l'activité effectuée.

1.4 Processus de reconnaissance d'Activité humaine

Les approches guidées par les données peuvent être divisées en deux catégories : les approches non supervisées et les approches supervisées. Les approches non supervisées telles que les méthodes d'extraction de motifs fréquents ou périodiques nécessitent une longue phase de collecte des données et ont un temps d'exécution long qui les rendent difficilement utilisables dans la pratique.

Les approches supervisées sont basées sur des techniques d'apprentissage utilisant les données des capteurs comme entrée.

Le problème de reconnaissance d'activité peut être ainsi réduit à un problème de classification, avec des caractéristiques extraites à partir des données de capteurs comme données d'entrée, et les activités déduites comme données de sortie de l'algorithme d'apprentissage. Le processus partant de l'extraction des données à partir des capteurs et allant jusqu'à l'entrainement d'un modèle pour reconnaitre des activités est appelé la chaine de Reconnaissance d'Activité (voir figure 1). Ce processus est classiquement divisé en différentes étapes qui sont toute d'une importance cruciale : acquisition des données, prétraitement, segmentation, extraction de caractéristiques et classification. [35]

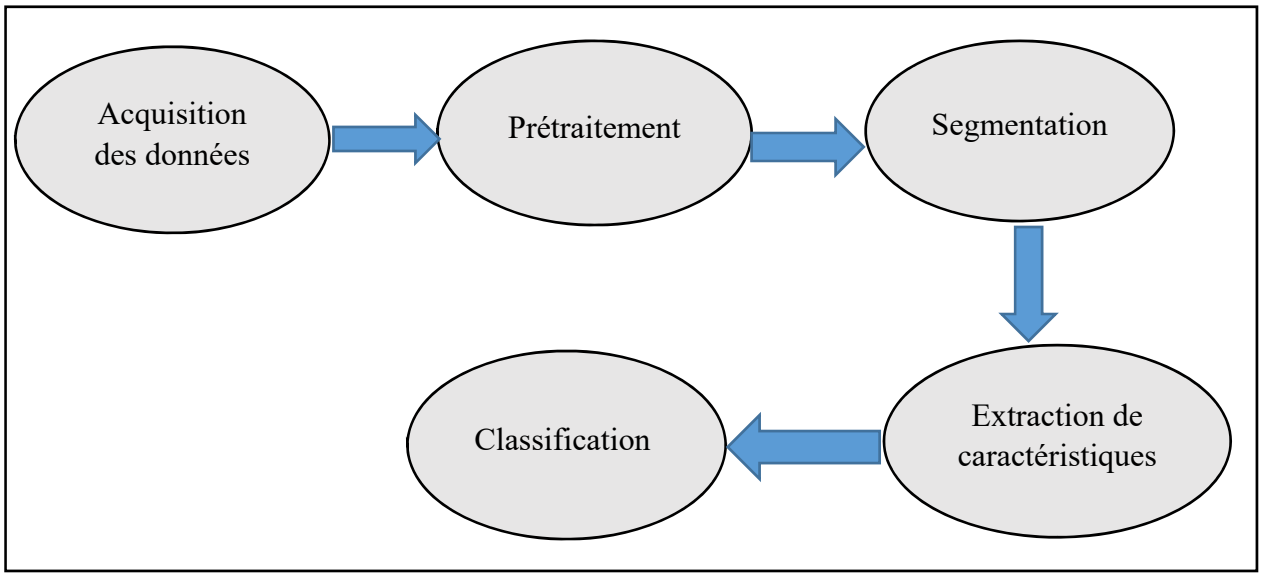

Figure 1 : Processus de reconnaissance d'activités humaines

Une fois que les données d'entrée sont recueillies à partir de tous les capteurs (acquisition de données), le processus de reconnaissance commence par nettoyer les données brutes (prétraitement). Le sous-ensemble de données sur lequel le processus de reconnaissance est appliqué est alors extrait (segmentation). Cela peut être réalisé à l'aide de deux techniques : la technique de fenêtre glissante ou l'analyse de changement d'activité. La technique de la fenêtre glissante est la plus utilisée à cause de sa simplicité (aucun prétraitement n'est nécessaire) et des bons résultats qu'elle permet d'obtenir. Chaque fenêtre glissante se chevauche avec la précédente pour garantir une consistance temporelle. Les données contenues dans chaque fenêtre sont transformées en un vecteur de caractéristiques (extraction de caractéristiques). Des caractéristiques communes (représentant la moyenne, la variance, l'entropie, etc.) sont alors utilisées comme données d'entrée pour un classifieur. Il existe de nombreux classifieurs supervisés : Support Vector Machines (SVM), k plus proches voisins (k-NN), classifieur basé sur le plus proche cluster (NCC), arbres de décision, réseaux de neurones, etc. Chaque classifieur supervisé doit être entrainé avec des données spécifiques (appelées données d'entrainement) pour pouvoir reconnaitre des classes (les activités de l'utilisateur dans notre cas).

1.5 Travaux récents du domaine de la reconnaissance des activités humaines

Dans le but de proposer une technique interactive pour le robot B21R, nous allons passer en revue des méthodes existantes de reconnaissance d'action par la vision dans le but d'exploiter la capacité du robot à percevoir des actions spécifiques afin d'interagir avec les personnes.

• **(Yu et al. 2010)** [27] proposent une méthode automatique de détection et de suivi des parties du corps humain à l'aide d'une caméra monoculaire. Après avoir détecté la zone en mouvement dans la séquence vidéo, le corps humain est segmenté en plusieurs parties (tête, pieds, bras, etc.) et représenté par un modèle de squelette humain en 2 dimensions. Les actions reconnues par ces systèmes sont : marcher, s'assoir, se lever, courir, sauter et tomber. La reconnaissance de l'action effectuée s'effectue à l'aide d'un apprentissage des vecteurs de caractéristiques (angles entre les membres) extraits à partir du modèle articulé pour chaque silhouette. À l'issue de la classification, la méthode montre une très bonne précision pour la reconnaissance de ces actions.

• Certains travaux se sont concentrés sur la reconnaissance de gestes effectués par la main. C'est le cas de **(Elmezain et al. 2009)** [10] qui mettent en œuvre un système automatique de reconnaissance de gestes en temps réel basé sur les Modèles de Markov Cachés. Les caractéristiques sont extraites à partir de séquences d'images stéréo en couleur. Les mains sont détectées à l'aide d'une segmentation de la couleur de la peau par mélange de gaussiennes puis un descripteur contenant les informations d'orientation est calculé à partir du chemin du geste. Cette méthode a été testée dans le cadre de la reconnaissance de trajectoires de la main représentant les chiffres arabes et montre un bon taux de reconnaissance (environ 95%). Il est également possible de reconnaitre les gestes de la main grâce à un système basé sur une méthode de soustraction de fond [11]. La caméra utilisée est une webcam et les gestes sont reconnus en temps réel à l'aide d'OpenCV.

• **(Chen et al. 2008)** [1] utilisent une reconnaissance en deux temps à l'aide d'une webcam. La première approche est une approche « bas niveau » et correspond à la reconnaissance des postures de la main par une méthode statistique basée sur les caractéristiques pseudo-Haar et un algorithme d'apprentissage en cascade AdaBoost. La deuxième de plus haut niveau

correspond à la reconnaissance d'un geste considéré comme une suite de postures à l'aide d'une analyse syntaxique basée sur une grammaire stochastique à contexte libre. Les postures détectées lors de l'approche « bas niveau » sont converties en séquence de phrases. Pour une phrase donnée, un geste peut être reconnu par association à la règle ayant la plus forte probabilité. Les taux de reconnaissance sont très satisfaisants pour cette méthode (> 97%) mais il reste difficile de garantir le fait que chaque utilisateur effectue le geste avec la bonne orientation.

• L'utilisation des caractéristiques pseudo-Haar est aussi mise en oeuvre par **(Agarwal et al. 2012)** [12] pour détecter les mains et l'algorithme de segmentation d'images couleur CamShift (Continuously Adaptive Mean Shift) est utilisé pour la reconnaissance des gestes. L'utilisation des caractéristiques pseudo-Haar permet une reconnaissance de gestes en temps réel et est avantageuse par rapport aux méthodes basées sur la détection de couleurs, car elle est robuste aux changements de luminosité. Cependant, cette méthode demande un grand nombre et une grande diversité d'exemples et l'entrainement d'un classifieur du type Haar demande beaucoup de temps et de ressources de calcul.

• (El-Yacoubi et al. 2014) [13] proposent une méthode de reconnaissance d'activités humaines par le robot humanoïde Nao. Les contraintes liées à la mobilité du robot (changements de fonds de scène, conditions d'éclairage, angles de vue et distances entre le robot et la personne effectuant les gestes) sont prises en compte. Le système développé est basé sur une extraction de trajectoires de points d'intérêt spatio-temporels et les vecteurs obtenus sont convertis par k-moyennes en un « Sac de Mots » qui est ensuite classifié par un SVM (« Support Vector Machine » ou « Séparateur à Vastes Marges »). Les points d'intérêt détectés sont caractérisés par des descripteurs appelés Histogrammes de Gradients (HOG) et Histogrammes de Flux Optique (HOF). La force de cette méthode est de permettre la reconnaissance d'activités effectuées à vitesses variables et/ou par des personnes différentes. L'utilisation d'un classifieur SVM est justifiée pour l'application de la reconnaissance d'activités par le robot Nao, car il ne nécessite qu'une représentation globale des données, est plutôt rapide et consomme une quantité limitée de mémoire, car il ne stocke qu'un ensemble réduit de vecteurs supports. Afin d'obtenir un temps de reconnaissance raisonnable, la qualité 20 des images capturées par la caméra du robot est réduite, car les ressources de Nao en stockage, en mémoire et en vitesse d'exécution sont limitées. Pour une séquence vidéo d'une durée de 2 secs, l'algorithme prend 15 secs pour reconnaitre l'activité effectuée.

Cet état de l'art des méthodes de reconnaissance de gestes par la vision montre une grande diversité d'approches. Les techniques de reconnaissance de gestes ou d'activité sont variées et

nécessitent pour la plupart de reconnaitre la partie du corps concernée ou de reconstruire le squelette avant de passer à la reconnaissance du geste effectué.

1.6 Conclusion

Nous avons introduit dans ce chapitre les concepts relatifs à la reconnaissance d'activité humaine, le problème formel et les approches existantes dans la littérature avec une emphase sur le processus de RAH à base de capteurs.

Dans le chapitre suivant, nous allons introduire une vue générale sur les techniques utilisées pour la classification des activités humaines.

Apprentissage automatique

Chapitre 2 : Apprentissage automatique

2.1 Introduction

Ce chapitre à caractère théorique, comprend des généralités relatives à l'apprentissage automatique ou machine learning en anglais. Nous y retrouvons notamment des notions de base sur les modèles de classifications. Ces notions nous seront utiles par la suite pour concevoir notre technique de reconnaissance d'activités humaines.

Dans un premier temps, nous présentons les notions de base sur l'apprentissage automatique. Par la suite, nous présentons des modèles d'apprentissage machine SVM multi-classe, TREE et KNN qui traite l'information issue d'un capteur de nouvelle génération du type RGB-D pour reconnaitre l'activité humaine.

2.2 Apprentissage automatique

2.2.1 Définitions

L'apprentissage automatique (Machine Learning en anglais) est un domaine de l'intelligence artificielle qui s'intéresse aux modèles qui peuvent apprendre à partir des données et sans être explicitement programmés (Figure 2). Il fait référence au développement, à l'analyse et à l'implémentation de méthodes qui permettent à une machine d'évoluer grâce à un processus d'apprentissage, et ainsi de remplir des tâches qu'il est difficile ou impossible de remplir par des moyens algorithmiques plus classiques [33].

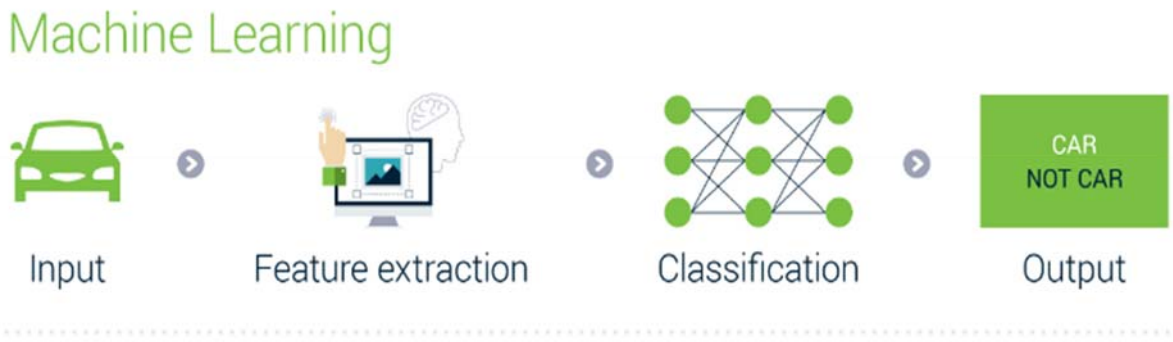

Figure 2:Méthode de catégorisation de véhicules de l'apprentissage automatique (Machine Learning) [42]

Une solution d'apprentissage automatique repose donc sur deux étapes principales :

• **L'apprentissage dans un premier temps** : Les données d'apprentissage sont souvent, réparties en 3 catégories [36]:

- **L'ensemble d'apprentissage** ou **Ensemble d'entrainement** : constitue l'ensemble des candidats ou exemples (images, attributs, DB, …) utilisés pour générer le modèle d'apprentissage.

- **l'ensemble de Tests** est constitué des candidats sur lesquels sera appliqué le modèle d'apprentissage (pour tester et corriger le model).

- **L'ensemble de validation** peut être utilisé lors de l'apprentissage (comme sous population de l'ensemble d'apprentissages) afin de valider (intégrer) le modèle et d'éviter le sur-apprentissage.

• **La prédiction dans un deuxième temps :** elle utiles le modèle généré lors de l'apprentissage, et une ou plusieurs instances de données, pour prédire la classe d'appartenance des instances de données.

Nous ne connaissons pas les classes de résultats pour les nouvelles données. C'est pourquoi nous avons besoin du modèle en premier lieu.

Nous pouvons prédire la classe de nouvelles instances de données à l'aide de notre modèle de classification finalisé dans scikit-learn à l'aide de la fonction ***Predict ().***

Par exemple, nous avons une ou plusieurs instances de données dans un tableau appelé Xnew. Ceci peut être passé à la fonction ***Predict ()*** sur notre modèle afin de prédire les valeurs de classe pour chaque instance du tableau.

2.3 Type d'apprentissage automatique

Il existe plusieurs manières de faire apprendre un algorithme à une machine. Chaque approche est différente des autres et convient à des problématiques différentes selon les données à disposition et l'objectif fixé de l'apprentissage, On cite les deux plus pertinent : apprentissage supervisé, apprentissage non-supervisé et apprentissage par renforcement (Figure 3). Nous allons nous attarder sur l'apprentissage supervisé car c'est l'approche que nous allons utiliser, contrairement aux autres qui ne serons traiter que de manière générale

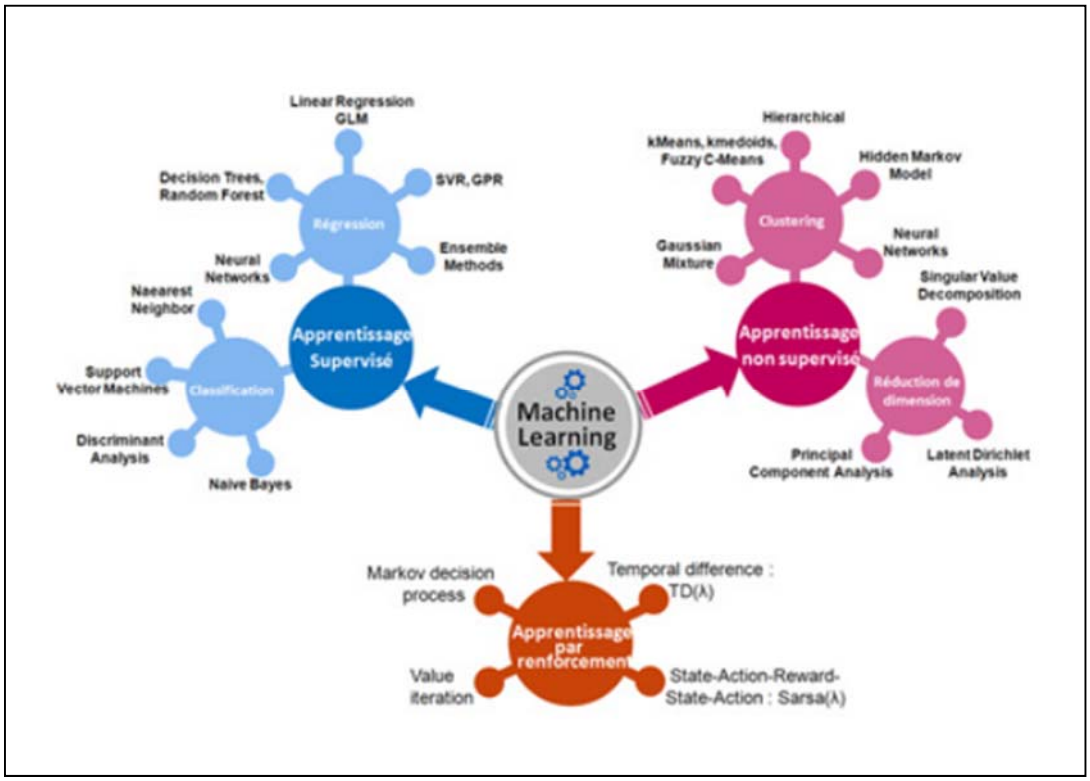

Figure 3 : Les 3 types d'apprentissage du machine learning [43]

2.3.1 L'apprentissage non supervisé

Dans l'apprentissage non supervisé, les données sont non étiquetées, de sorte que l'algorithme d'apprentissage trouve tout seul des points communs parmi ses données d'entrée. Les données non étiquetées étant plus abondantes que les données étiquetées, les méthodes d'apprentissage automatique qui facilitent l'apprentissage non supervisé sont particulièrement utiles.

L'objectif de l'apprentissage non supervisé peut être aussi simple que de découvrir des modèles cachés dans un ensemble de données, mais il peut aussi avoir un objectif d'apprentissage des caractéristiques, qui permet à la machine intelligente de découvrir automatiquement les représentations nécessaires pour classer les données brutes [28].

L'apprentissage non supervisé est couramment utilisé pour les données transactionnelles. Vous pouvez avoir un grand ensemble de données sur les clients et leurs achats, mais en tant qu'être humain, vous ne serez probablement pas en mesure de comprendre quels attributs similaires peuvent être tirés des profils de clients et de leurs types d'achats. Avec ces données introduites dans un algorithme d'apprentissage non supervisé, on peut déterminer que les femmes d'une

certaine tranche d'âge qui achètent des savons non parfumés sont susceptibles d'être enceintes, et donc une campagne de marketing liée à la grossesse et aux produits pour bébés pour augmenter leur nombre d'achats.

Sans une réponse « correcte », les méthodes d'apprentissage non supervisées peuvent examiner des données complexes, plus expansives et apparemment sans point commun, afin de les organiser de manière potentiellement significative. L'apprentissage non supervisé est souvent utilisé pour la détection d'anomalies, y compris pour les achats frauduleux de cartes de crédit et les systèmes de recommandation qui conseille sur les produits à acheter ensuite. Dans l'apprentissage non supervisé, les photos non marquées des chiens peuvent être utilisées comme données d'entrée pour l'algorithme afin de trouver des similitudes et de classer les photos de chiens ensemble.

2.3.1.1 Les méthodes de classification non supervisées

Les méthodes de classification non supervisée sont des méthodes qui cherchent à identifier, ou à partitionner un ensemble de données à un certain nombre de classes distinctes à partir d'un fichier de description, tout en essayant d'optimiser un critère qui vise à regrouper les données les plus homogènes dans chaque classe.

- **K-moyennes**

C'est une méthode itérative qui a été proposée par MacQueen en 1967. Le principe de l'algorithme k-means est le suivant :

- On définit un nombre k de nuages à priori.
- Chaque nuage est initialisé par un centre. Le centre est tiré d'une façon aléatoire de l'espace de l'individu.
- On alloue chaque individu à un nuage *i* en basant sur une mesure de similarité.
- En faisant la moyenne des éléments dans chaque nuage *i* afin de produire les nouveaux centres.
- On réitère jusqu'à ce qu'aucun individu ne change de nuage.

Un des défauts de l'algorithme k-means est le choix des conditions initiales qui peuvent affecter les résultats de classement, ça signifie que la partition d'un groupe d'individus dépend largement des centres initiaux et du nombre de nuages.

- **Classification hiérarchique**

Les méthodes de classification hiérarchique [29] sont des méthodes itératives fondées sur des mesures de similarité. Ces méthodes visent à construire des regroupements en classes

homogènes d'un ensemble d'individus. On cite par exemple la méthode de classification ascendante qui consiste à calculer une matrice de similarité entre chaque paire d'objets. Ensuite à chaque itération, un nouveau cluster est formé par la fusion de deux clusters les plus proches en basant sur la matrice trouvée. La matrice de similarité est mise à jour par le calcul de la ressemblance entre le nouveau cluster et les clusters existants. La mise à jour se répète jusqu'à la fusion de deux derniers clusters. La qualité de clustering par cette méthode dépend largement de la métrique utilisée. Une application de la classification hiérarchique est la recherche d'image [30] (voir figure 4).

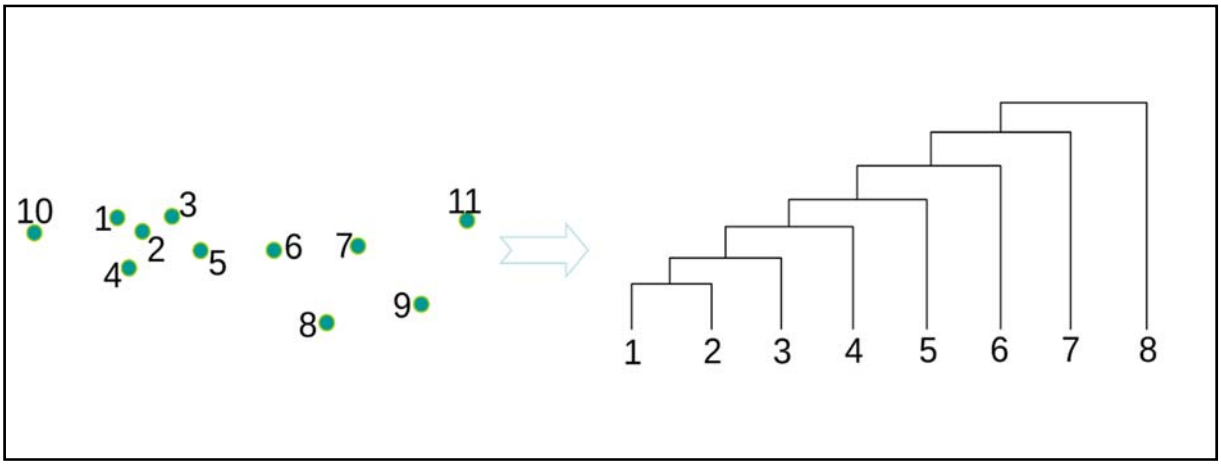

Figure 4 : Processus de fusion par une méthode hiérarchique [30]

2.3.2 L'apprentissage supervisé

Dans ce cas, l'ensemble de données d'apprentissage (observations) **X** utilisé lors de cette phase, est labélisé par les solutions correctes **Y** au problème donné. Pour chaque individu dans cet ensemble, l'algorithme essaye de résoudre le problème posé, puis compare la solution trouvée **Y'** à la solution correcte **Y** (donnée) [16].

Il s'agit donc d'optimiser une fonction **f** tel que **Y'** se rapproche le plus possible de **Y**, et donc de minimiser l'erreur de prédiction. Cette fonction **f** est ensuite utilisée pour résoudre le problème posé avec n'importe quel autre ensemble de données d'entrée (observations) **X**. L'optimisation de cette fonction peut se faire de différentes manières et par divers algorithmes. Le plus souvent, la fonction gradient est utilisée pour optimiser des probabilités. L'algorithme le plus connu est celui du « gradient descent » qui utilise les dérivées partielles afin de minimiser

l'erreur en fonction des paramètres de la fonction **f** [17]. On peut distinguer deux grands types d'apprentissages supervisés : la classification et la régression [32].

2.3.2.1 Classification

Lorsqu'on fait de la classification, l'entrée est l'instance d'une classe et l'étiquette est la classe correspondante. La classification consiste donc à apprendre une fonction fclass de X= Rd dans Y= N qui associe à un vecteur sa classe. Si le nombre de classe est égal à 2, on parle alors de la classification binaire [31].

2.3.2.2 Régression

Dans le cas de la régression, l'entrée n'est pas associée à une classe mais à une ou plusieurs quantités continues. Ainsi, l'entrée pourrait être les caractéristiques d'une personne (son âge, son sexe, son niveau d'études) et l'étiquette son revenu. La régression consiste donc à apprendre Daune fonction f_{regr} de X =R_d dans Y=R_k qui associe à un vecteur sa valeur associée [31].

2.3.3.1 Les méthodes de classification supervisées

Dans ces méthodes, l'ensemble d'apprentissage est constitué d'un ensemble de couples entrées-sorties dans lequel l'entrée représente le vecteur de caractéristique et la sortie représente l'étiquette correspondant à l'entrée. Le problème de classification supervisé cherche à identifier la classe d'appartenance d'une nouvelle entrée qui n'appartient pas à l'ensemble d'apprentissages, tout en essayant d'apprendre une fonction de classification à partir d'un ensemble d'entrainement. La fonction de classification apprise permet d'associer une classe à une nouvelle entrée à l'aide de certaines caractéristiques qui décrivent l'entrée.

A. Les k plus proches voisins (KNN)

L'algorithme KNN (en anglais k - NN : k Nearest Neighbors) est un algorithme de classification supervisé. Chaque observation de l'ensemble d'apprentissages est représentée par un point dans un espace à n dimensions ou n est le nombre de variables prédictives. Pour prédire la classe d'une observation, on cherche les k points les plus proches de cet exemple (Figure 5). La classe de la variable cible, est celle qui est la plus représentée parmi les k plus proches voisins. Il existe des variantes de l'algorithme ou on pondère les k observations en fonction de leur distance à l'exemple dont on veut classer [19], les observations les plus éloignées de notre exemple seront considérées comme moins importantes.

Une variante de l'algorithme est utilisée par NetFlix [20] pour prédire les scores qu'un utilisateur attribuera à un film en fonction des scores qu'il a attribués à des films similaires

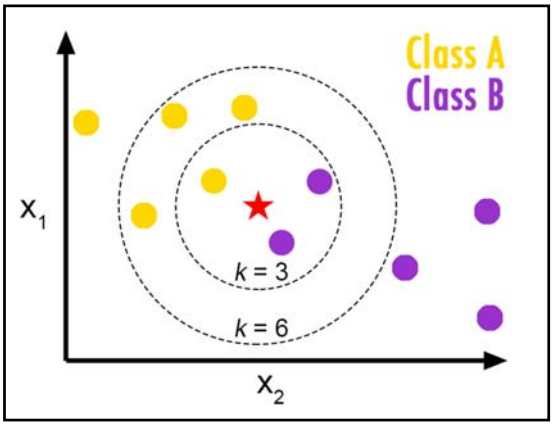

Figure 5 : Pour k = 3 la classe majoritaire du point central est la classe B, mais si on change la valeur du voisinage k = 6 la classe majoritaire devient la classe A [34]

- **Avantages :**

— simple à concevoir

- **Inconvénients**

— Sensible aux bruits

— Pour un nombre de variables prédictives très grand, le calcul de la distance devient très couteux.

B. Les Arbres de Décision

Les Arbres de Décision sont des modèles de ML supervisés, pouvant être utilisés pour la classification que pour la régression.

Un arbre de décision représente une fonction qui prend comme entrée un vecteur d'attributs et retourne une décision qui est une valeur unique. Les entrées et les sorties peuvent être discrètes ou continues.

Un arbre de décision prend ses décisions en exécutant une séquence de test, chaque nœud interne de l'arbre correspond à un test de la valeur d'un attribut et les branches qui sortent du nœud sont les valeurs possibles de l'attribut. La classe de la variable cible est alors déterminée par la feuille dans laquelle parvient l'observation à l'issue de la séquence de test.

La phase d'apprentissage consiste à trouver la bonne séquence de test. Pour cela, on doit décider des bons attributs à garder. Un bon attribut divise les exemples en ensembles homogènes c.à. d

qu'ils ne contiennent que des observations appartenant à la même classe, alors qu'un attribut inutile laissera les exemples avec presque la même proportion de valeur pour la variable cible.

Ce dont on a besoin c'est d'une mesure formelle de "bon" et "inutile". Pour cela, il existe des métriques standards homogénéisées avec lesquels on peut mesurer l'homogénéité d'un ensemble. Les plus connus sont l'indice de diversité de Gini et l'entropie [23]

En général l'entropie d'une variable aléatoire ***V*** avec des valeurs ***vk*** chacune avec une probabilité ***P(vk).*** (Voir Figures (6,7 et 8))

Example	Input Attributes										Goal
	Alt	*Bar*	*Fri*	*Hun*	*Pat*	*Price*	*Rain*	*Res*	*Type*	*Est*	*WillWait*
$\mathbf{x}_1$	*Yes*	*No*	*No*	*Yes*	*Some*	*$$$*	*No*	*Yes*	*French*	*0–10*	$y_1 = Yes$
$\mathbf{x}_2$	*Yes*	*No*	*No*	*Yes*	*Full*	*$*	*No*	*No*	*Thai*	*30–60*	$y_2 = No$
$\mathbf{x}_3$	*No*	*Yes*	*No*	*No*	*Some*	*$*	*No*	*No*	*Burger*	*0–10*	$y_3 = Yes$
$\mathbf{x}_4$	*Yes*	*No*	*Yes*	*Yes*	*Full*	*$*	*Yes*	*No*	*Thai*	*10–30*	$y_4 = Yes$
$\mathbf{x}_5$	*Yes*	*No*	*Yes*	*No*	*Full*	*$$$*	*No*	*Yes*	*French*	*>60*	$y_5 = No$
$\mathbf{x}_6$	*No*	*Yes*	*No*	*Yes*	*Some*	*$$*	*Yes*	*Yes*	*Italian*	*0–10*	$y_6 = Yes$
$\mathbf{x}_7$	*No*	*Yes*	*No*	*No*	*None*	*$*	*Yes*	*No*	*Burger*	*0–10*	$y_7 = No$
$\mathbf{x}_8$	*No*	*No*	*No*	*Yes*	*Some*	*$$*	*Yes*	*Yes*	*Thai*	*0–10*	$y_8 = Yes$
$\mathbf{x}_9$	*No*	*Yes*	*Yes*	*No*	*Full*	*$*	*Yes*	*No*	*Burger*	*>60*	$y_9 = No$
$\mathbf{x}_{10}$	*Yes*	*Yes*	*Yes*	*Yes*	*Full*	*$$$*	*No*	*Yes*	*Italian*	*10–30*	$y_{10} = No$
$\mathbf{x}_{11}$	*No*	*No*	*No*	*No*	*None*	*$*	*No*	*No*	*Thai*	*0–10*	$y_{11} = No$
$\mathbf{x}_{12}$	*Yes*	*Yes*	*Yes*	*Yes*	*Full*	*$*	*No*	*No*	*Burger*	*30–60*	$y_{12} = Yes$

Figure 6 : L'ensemble d'apprentissages contient 12 observations décrites par 10 variables prédictives et une variable cible. [18]

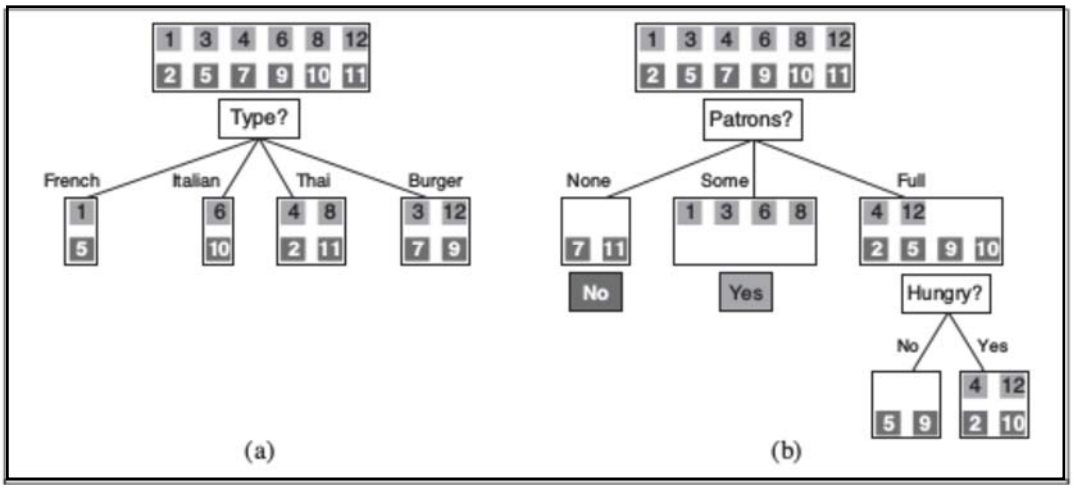

Figure 7: Les exemples positifs sont représentés par des cases claires

Dans cette figures les exemples négatifs sont représ entés par des cases sombres. (a) montre que la division par l'attribut Type n'aide pas à avoir une distinction entre les positifs et les négatifs

exemples. (b) montre qu'avec la division par l'attribut Patrons on obtient une bonne séparation entre les deux classes. Après division par Patrons, Hungry est un bon second choix. [18]

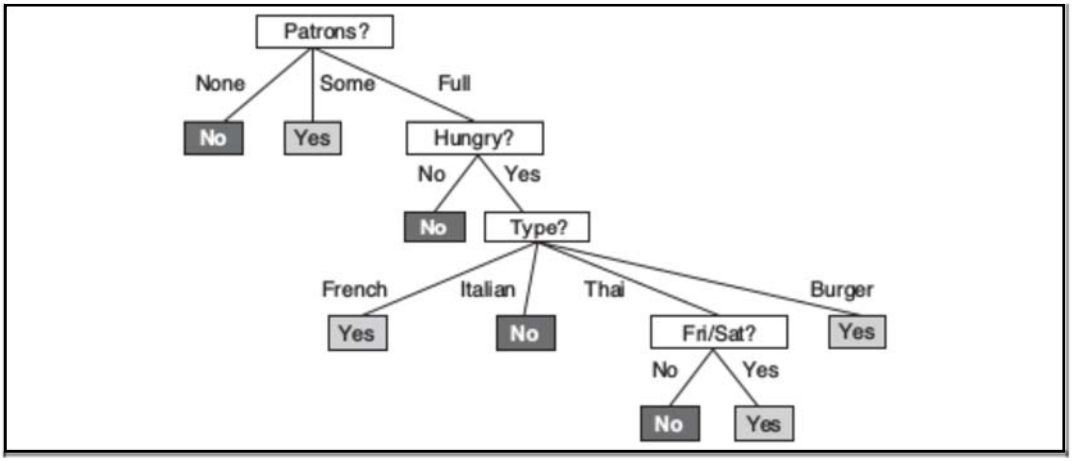

Figure 8 : L'arbre de décision déduit à partir des 12 exemples d'apprentissage. [18]

- **Avantages :**

— C'est un modèle boite blanche, simple à comprendre et à interpréter

— Peu de préparation des données.

— Les variables prédictives en entrée peuvent être aussi bien qualitatives que quantitatives.

— Performant sur de grands jeux de données

- **Inconvénients :**

— L'existence d'un risque de sur-apprentissage si l'arbre devient très complexe. On utilise des procédures d'élagage pour contourner ce problème.

C. Machines à vecteurs de support (SVM)

Les machines à vecteurs de support ou séparateurs à vaste marge (en anglais SVM : Support Vector Machines) font partie des techniques d'apprentissage supervisé, destinées à résoudre des problèmes de classification. Si les données d'entrainement sont linéairement séparables, il existe généralement plusieurs hyperplans qui peuvent les séparer. Ce type de classifieur tente de trouver une frontière de décision permettant de séparer linéairement les exemples de la première classe des exemples de la deuxième classe dans l'ensemble d'apprentissages.

L'objectif du SVM est de choisir l'hyperplan qui donnera le modèle de classification qui se généralisera le mieux à d'autres données que celles de l'ensemble d'entrainement. Afin d'arriver à cela, SVM utilise la notion de la marge. Une marge d'un hyperplan est la distance

entre ce dernier et la donnée la plus proche (Figure 9). Du point de vue mathématique, il a été démontré que l'hyperplan optimal est celui qui possède la marge maximale [37]. L'algorithme SVM peut être utilisé pour classifier les données non linéairement séparables par l'application d'une technique dite " noyau " (polynomial, gaussien) permettant de projeter les caractéristiques initiales dans un nouvel espace à grande dimension. Cette projection vise à rendre les données linéairement séparables dans le nouvel espace. Les méthodes SVM ont été appliquées avec succès dans les problèmes de catégorisation des textes [38], de reconnaissances des actions humaines [39], de reconnaissance d'objets [40], etc.

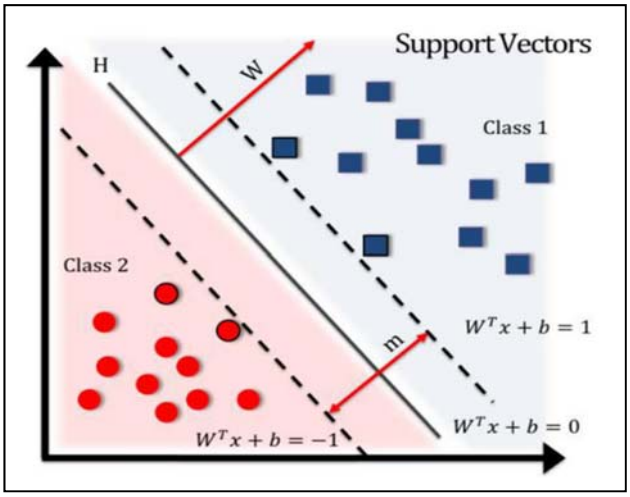

Figure 9 : Frontière de décision linéaire d'un classifieur SVM. Les échantillons qui se trouvent sur la marge s'appellent les vecteurs de support [41].

- **Avantages :**

— Il permet de traiter des problèmes de classification non linéaire complexe.

— Les SVM constituent une alternative aux réseaux de neurones, car plus faciles à entrainer.

- **Inconvénients :**

— Les SVM sont souvent moins performants que les forets aléatoires.

D. Naïves de Bayes

Les méthodes naïves de Bayes sont un ensemble d'algorithmes d'apprentissage supervisé basé sur l'application du théorème de Bayes en supposant que les caractéristiques sont indépendantes

(dite naïve) les unes des autres [45]. L'apprentissage par cette méthode consiste à entrainer un estimateur de densité de probabilité sur chaque classe afin de concevoir un modèle probabiliste. Pour une nouvelle entrée, le modèle construit permet d'estimer la probabilité de vraisemblance de chaque classe. La classification d'une nouvelle observation est basée sur l'estimation des probabilités a posteriori des classes en retenant la classe pour laquelle la probabilité à posteriori est maximale. La probabilité à posteriori d'une classe est estimée à partir la connaissance de la probabilité à priori la probabilité de vraisemblance par la règle de Bayes.
L'avantage du classifieur bayésien naïf est qu'il requiert relativement peu de données d'entraînement pour estimer les paramètres nécessaires à la classification, à savoir moyennes et variances des différentes variables.

Dans notre travail nous avons utilisé ces types d'algorithmes (SVM, KNN, Arbre de Décision)

2.4 La validation croisée

La validation croisée fait probablement partie des techniques les plus importantes utilisées par les scientifiques, car il est toujours nécessaire de valider la stabilité du modèle d'apprentissage automatique. La validation croisée est une technique de validation de modèle permettant d'évaluer la manière dont les résultats d'une analyse statistique (modèle) se généraliseront à un ensemble de données indépendant. Elle est principalement utilisée pour estimer la précision avec laquelle un modèle prédictif fonctionnera, dans des contextes où l'objectif est la prédiction.

La validation croisée a pour objectif de définir un ensemble de données permettant de tester le modèle en phase d'apprentissage (c'est-à-dire un ensemble de données de validation) afin de limiter les problèmes de sur-apprentissage, et d'obtenir un aperçu de la manière dont le modèle se généralisera en un ensemble de données indépendant.

— **Les stratégies de la validation croisée** [44] : En règle générale, différentes stratégies de validation existent en fonction du nombre de divisions effectuées dans l'ensemble de données.

(a) **testset validation** : nous divisons l'échantillon de taille n en deux sous-échantillons, le premier dit d'apprentissage (communément supérieur à 60% de l'échantillon) et le second dit de test. Le modèle est entraîné sur l'échantillon d'apprentissage et validé sur

l'échantillon de test. L'erreur est estimée en calculant un test, une mesure ou un score de performance du modèle sur l'échantillon de test.

(b)**k-fold** : nous divisons l'échantillon original en K échantillons, puis nous sélectionnons un des K échantillons comme ensemble de validation et les K-1 autres échantillons constitueront l'ensemble d'apprentissage. Nous calculons comme dans la première méthode le score de performance, puis nous répétons l'opération en sélectionnant un autre échantillon de validation parmi les K-1 échantillons qui n'ont pas encore été utilisés pour la validation du modèle. L'opération se répète ainsi K fois pour qu'en fin de compte chaque sous-échantillon ait été utilisé exactement une fois comme ensemble de validation. La moyenne est enfin calculée pour estimer l'erreur de prédiction (voir figure 10).

(c) **leave-one-out** : Il s'agit d'un cas particulier de Kfold lorsque K est égal au nombre d'échantillons de notre jeu de données. Cela signifie qu'il parcourra chaque échantillon de notre ensemble de données à chaque fois en utilisant un objet k-1 en tant qu'échantillons d'apprentissage et un seul objet en tant qu'ensemble de test.

Cette méthode peut être utile si nous avons trop peu de données et si le modèle est assez rapide pour se recycler.

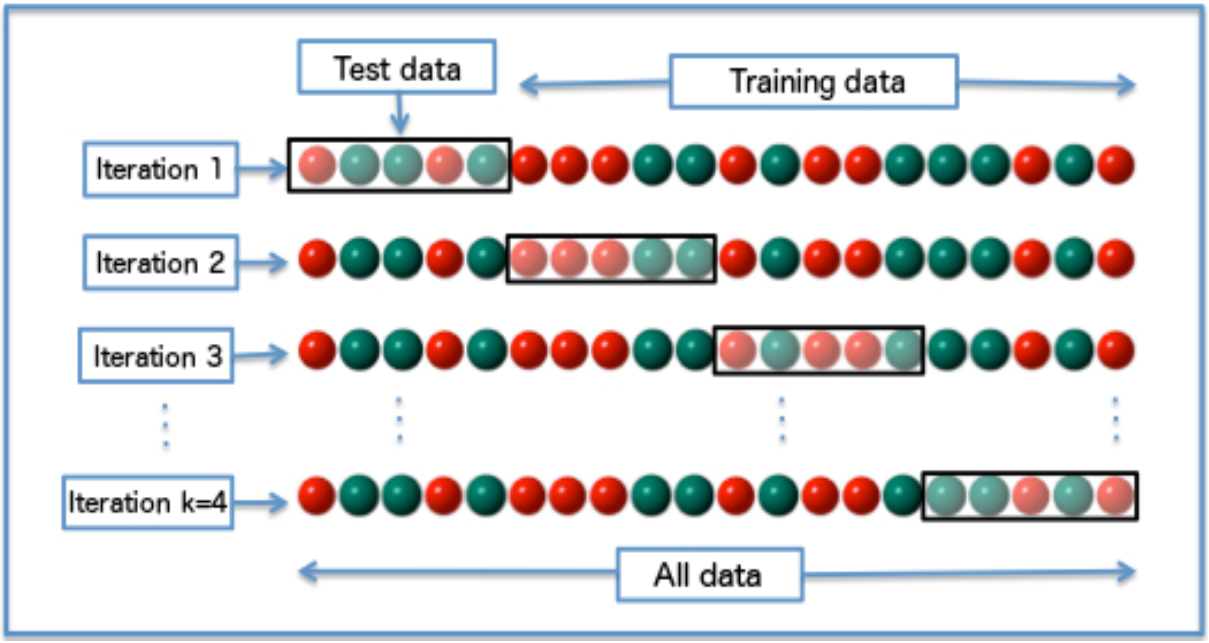

Figure 10 : fonctionnement de la méthode k_fold.

2.5 Conclusion

Dans ce chapitre, nous avons présenté les différents concepts théoriques relatifs à l'apprentissage automatique e rencontrés dans la littérature de la reconnaissance d'activités humaines. Nous avons abordé les différents algorithmes pour la classification en détail afin de faciliter la comp

réhension de la suite du PFE

Conception de la solution

Chapitre 3 : Conception de la solution

3.1 Introduction

Rappelons que notre projet s'inscrit dans le cadre d'un projet de recherche qui a pour objectif principal l'amélioration de l'interaction entre un robot et un humain, pour cela une étude de faisabilité de l'application de techniques d'apprentissage automatique pour la reconnaissance d'activités humaines a été réalisée. Dans ce qui suit, nous abordons la conception et la réalisation qui démontrent et appuient les résultats des études et qui permettent aussi de faire des essais sur les données.

Dans le présent chapitre, nous présenterons une conception de haut niveau où nous identifierons les éléments composant notre solution. Par la suite, nous détaillerons la conception de chaque élément en présentant l'architecture détaillée et les algorithmes mis en jeu.

3.2 Conception architecturale

La figure 11 représente un schéma de l'architecture de la solution proposée pour la classification de données d'activités humaines, en mettant en avant le découpage modulaire par la suite, nous détaillerons la conception de chaque module en présentant son architecture détaillée et les algorithmes utilisés

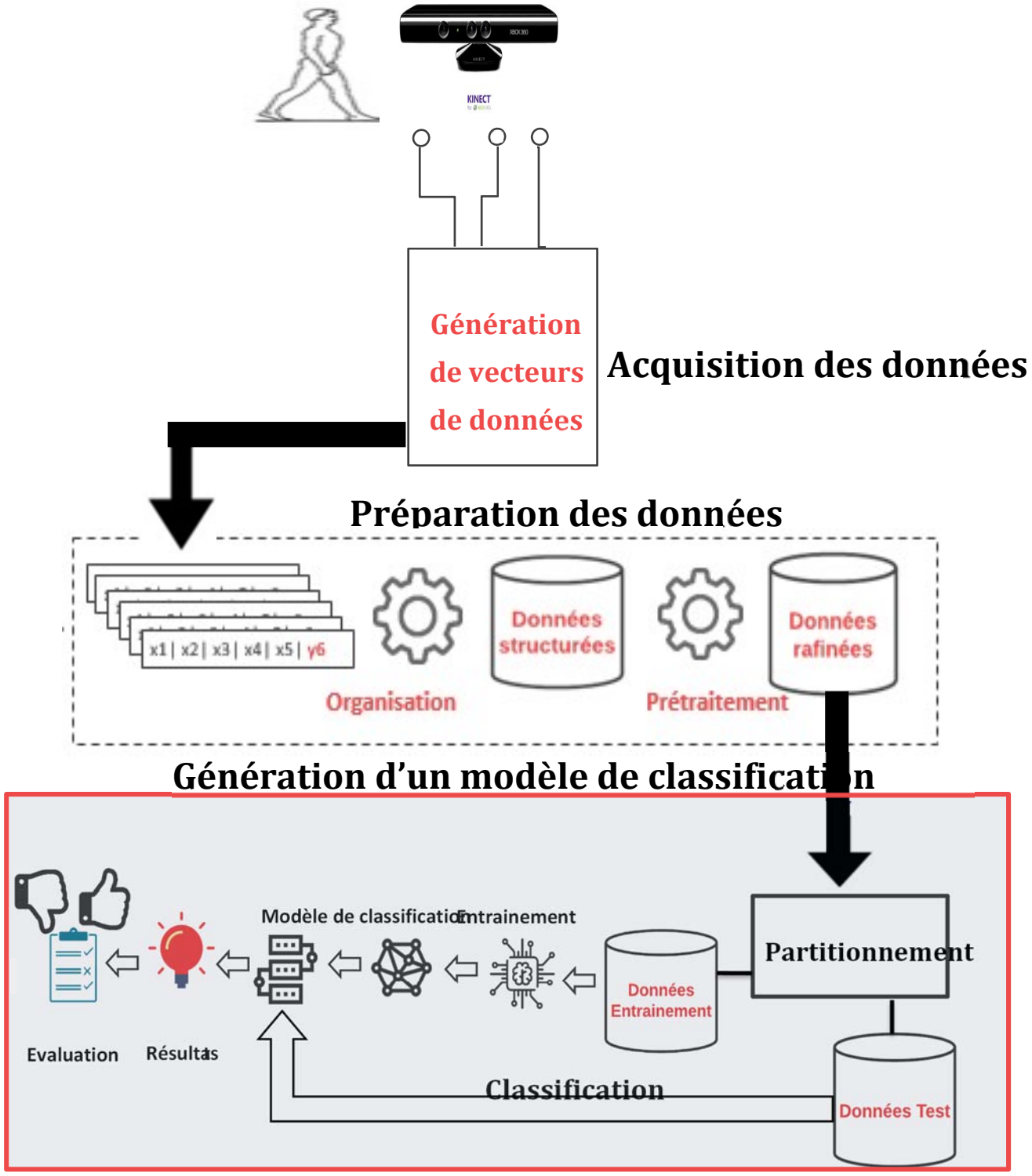

Figure 11 : Architecture de haut niveau de solution proposée

Notre système est composé principalement de trois modules dont les fonctionnalités sont les suivantes :

- Acquisition et préparation des données.
- Classification des données.
- Evaluation des résultats obtenus de la classification.

3.3 Conception détaillée

Dans ce qui suit, nous présentons la conception détaillée des modules du système. Nous donnons leurs principales composantes et nous détaillons leur fonctionnement.

3.3.1 Module d'acquisition et préparation de données

Ce module se charge de l'acquisition du flux de données 3D de l'utilisateur. Pour se faire, on a utilisé le capteur Kinect (figure 12).

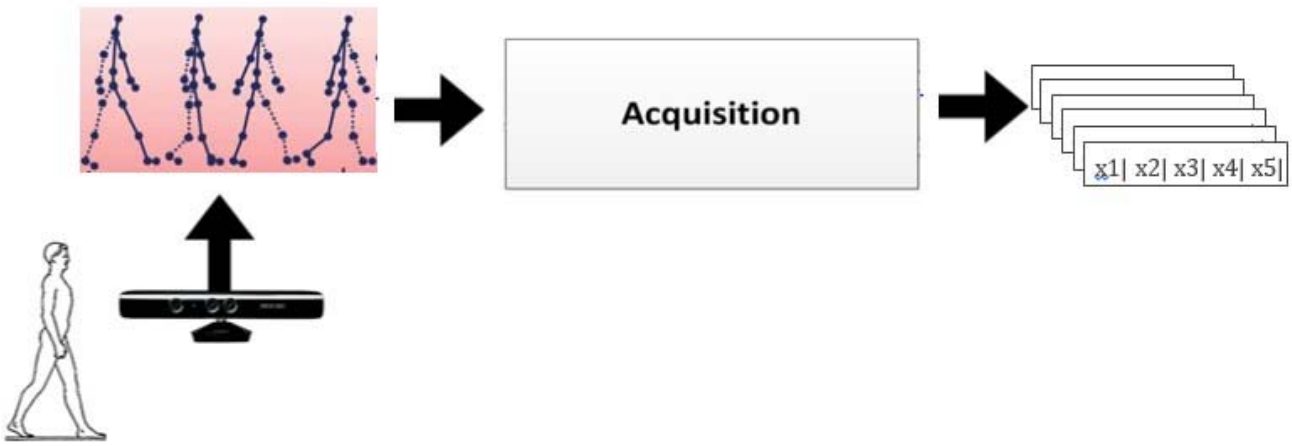

Figure 12 : Module d'acquisition de données.

3.3.1.1 Kinect :

La Kinect est un capteur équipé d'une camera RGB et une caméra infrarouge (figure 13), elle a été développée en 2010 par la société multinationale de l'informatique Microsoft, pour la console de jeu XBOX 360. Récemment ce capteur a attiré l'attention de plusieurs chercheurs pour son prix raisonnable en le comparant aux autres capteurs disponibles sur le marché. Il est utilisé dans différents domaines ; l'imagerie, les scanners3D, réalité virtuelle...etc.

Les données de profondeur obtenues nous fournissent les informations à propos de la profondeur des pixels, mais pour localiser et suivre les mouvements, il faut réaliser un module, de plus qui puisse le faire. La puissance du suivi des squelettes (SkeletonTracking), est la fonctionnalité la plus impressionnante que le SDK Kinect offre. Cette plateforme aide à comprendre la position du corps humain en 3D, et avoir une très bonne idée sur la position des articulations du corps à chaque instant de temps en X, Y, et Z.

Auparavant, l'acquérir de ce genre de données, nécessitait un matériel spécial comprenant plusieurs caméras, et des unités de traitements adéquats. Actuellement la Kinect SDK peut suivre les mouvements de quatre personnes simultanément, et 15 articulations par squelette.

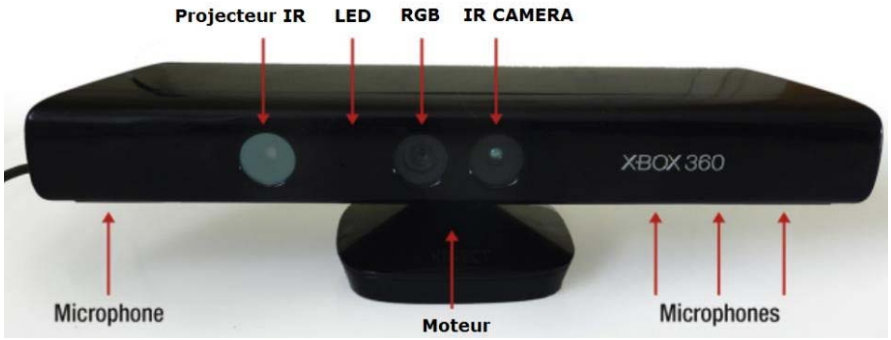

Figure 13 : Hardware de la Kinect.

Le SDK de la Kinect fournit le modèle de squelette, et permet le suivi de déplacement des 15 articulations du corps humain (voir figure 14). Cette propriété a rendu les systèmes de RAH plus efficaces en concentrant sur la partie reconnaissance (classification) pour améliorer les méthodes de classification.

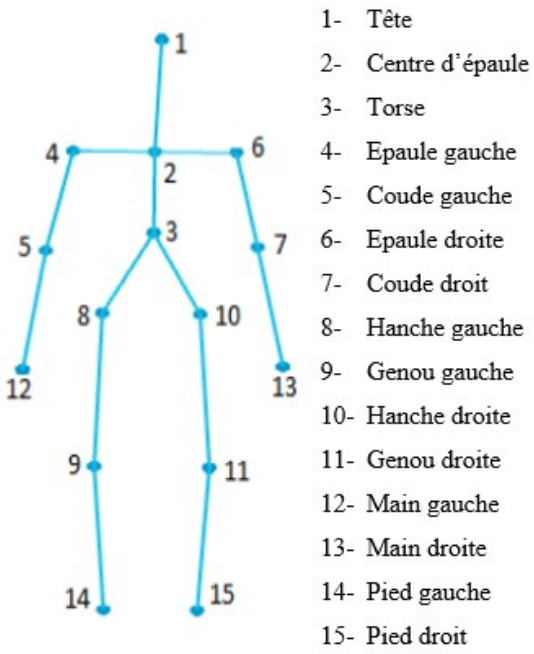

Figure 14 : Points articulaires squelettiques fournis par le capteur Kinect.

Généralement pour une séquence gestuelle en 2D P ($p1, p2, \ldots, p_n$) tel que pi est un point (xi, yi) dans l'espace et n la langueur de la séquence, nous obtiendrons une séquence d'orientations ($a1, \ldots .an$ - 1).

Dans notre cas les activités sont faites dans un enivrement 3D donc il est nécessaire d'extraire d'un autre paramètre pour représenter la troisième dimension. Après une étude qu'on a faite et dans le but de faciliter les étapes ultérieures de classification et améliorer l'interprétation des résultats, on a utilisé que 9 articulations **(**Tête, centre d'épaule, le torse, épaule droite, épaule gauche, coude droit, coud gauche, main droite, main gauche.**)** du squelette capter par la Kinect, parce que les activités que nous avons choisies nécessitent que la partie supérieure du corps humaines.

3.3.2 Identification de l'ensemble des activités humaines

Il existe d'innombrables activités humaines dans la vie quotidienne qui peuvent être incluses dans notre ensemble de données. La sélection d'une activité particulière à inclure dans notre DataSet a été contraint aux informations émises par le capteur utilisé. Vu que notre approche de classification des activités est basée sur le suivi de la position des articulations squelettiques, nous avons sélectionné les activités qui impliquent le mouvement des articulations du squelette,

dans notre projet six activités ont été jugées nécessaires pour vérifier notre approche. Les activités sont: 1- Arrêter, 2- Appeler, 3- Saluer, 4- Venir, 5- Aller, 6-Pointer vers.

Nous pouvons rajouter d'autres activités à l'ensemble de données dans l'avenir. La description des chaque activité est donnée dans le tableau 2

Activité	Description
1- Arrêter	La personne utilise les deux mains au même temps pour faire empêcher d'avancer, d'aller plus loin.
2- Appeler	La personne utilise sa main droite/gauche pour appeler le robot à distance.
3- Saluer	La personne salue avec sa main droite/gauche.
4- Venir	La personne se déplace de manière à aboutir dans un lieu proche (S'approcher de la Kinect).
5- Partir	La personne se déplace de manière à aboutir dans un lieu éloigné (s'éloigner de la Kinect).
6- Pointer vers	La personne pointe vers un endroit avec sa main droite/gauche.

Tableau 2 : Description des activités de notre DataSet

3.3.2.1 Structure de notre DataSet :

Il est primordial de rendre toutes les données émises par la Kinect sous un format unique pour les utiliser comme entrée dans chacun des classificateurs. L'objectif final de cette réorganisation des données est d'aboutir à une structure commune, standardization ou normalisation. Le schéma ci-dessous montre la structure de chaque vecteur de notre DataSet comme illustrée dans la Figure 15. (Les informations émises à un instant t sont composées de neuf mesure de l'articulation, 4 itérations suffit pour avoir l'information sur une activité, mais

plus le nombre d'itération est plus élevé nous aurons l'information complète sur l'activité, chaque mesure de l'articulation est composée des coordonnées spatiales X, Y, Z et les mesures d'orientation p, w, r.)

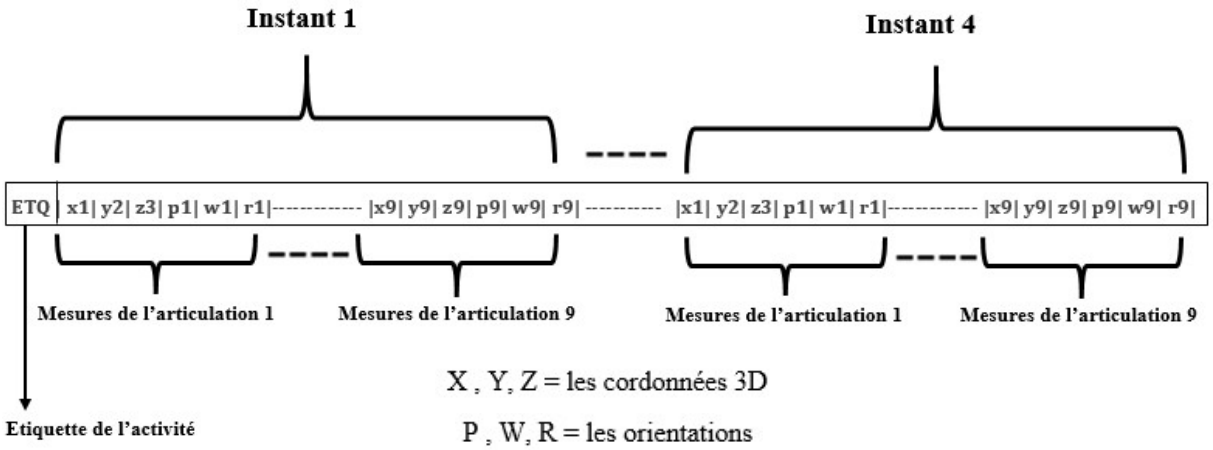

Figure 15 : Structure de données de notre DataSet.

3.3.2.2 Organisation des données :

La phase d'acquisition des données, nous devons les répartir pour avoir une partie destinée à l'entraînement des modèles de classification et une autre partie pour effectuer le test et évaluer les performances des classificateurs Après. Selon plusieurs études, il a été recommandé de prendre 88% pour l'entraînement partie et 12% pour l'évaluation des performances.
(Figure 16)

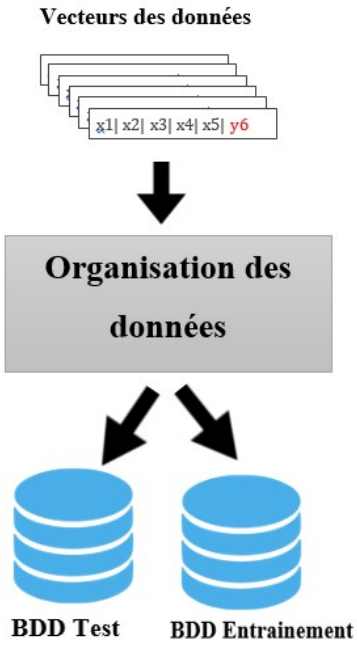

Figure 16 : Organisation des données

Ainsi, ce module permet d'avoir une forme commune à toutes les DataSets afin de pouvoir les injecter dans les classificateurs par la suite.

3.3.3 Classification des données

Cette étape consiste à entrainer le classificateur afin de construire et générer des règles sous forme de modèle pour une classification ultérieure de nouveaux échantillons.

Dans cette partie nous détaillons les différentes méthodes d'apprentissage automatique utilisées pour générer les modèles de classification.

3.3.3.1 Les k plus proches voisins:

Dans un premier temps nous allons utiliser l'algorithme k plus proches voisins pour la tâche de reconnaissance d'activité. Six activités différentes sont présentes dans notre DataSet. Le nombre de voisins représente l'un des paramètres les plus importants dans cette algorithme. Il n'existe pas de règles ou critères générales pour choisir ce paramètre. Par conséquent, nous sommes amenés à faire plusieurs expérimentations afin de pouvoir choisir le bon nombre de voisins. Cette phase est la plus contrariante de la méthode. Pour cela, nous avons implémenté une méthode qui donne à K plusieurs valeurs et retourne le taux d'erreur de chaque valeur.

Pour chaque vecteur d'attributs à classer.

Ce paramètre est très important pour cette algorithme est la distance de similarité utilisée. Nous avons opté pour la distance euclidienne. Elle permet de calculer la similarité entre les vecteurs des instances à classer comme suit

$$D_{euclid} = \sqrt{\sum_{i=1}^{d}(x_i - y_i)^2}$$

Une fois les distances de similarité sont calculées entre l'instance en question (requête) et les celles de le DataSet, nous retenons la liste des k vecteurs les plus proches comme suit :

$$x_{i^*} = \underset{i\in\{1,\ldots..N\}}{argmin}\, d(x_i, x)$$

Enfin, la classe assignée à l'instance en question est la classe majoritaire parmi les k instances les plus proches déterminées par l'algorithme k plus proches voisins

3.3.3.2 Les arbres de décision

L'idée de base de cet algorithme est la suivante:

- Sélectionnez le meilleur attribut à l'aide de mesures de sélection d'attribut (ASM) pour fractionner les enregistrements.
- Faites de cet attribut un nœud de décision et divisez le jeu de données en sous-ensembles plus petits.
- Démarre la construction de l'arborescence en répétant ce processus de manière récursive pour chaque enfant jusqu'à ce que l'une des conditions corresponde: (Figure 17)
- Tous les tuples appartiennent à la même valeur d'attribut.
- Il ne reste plus d'attributs.
- Il n'y a plus d'instances.

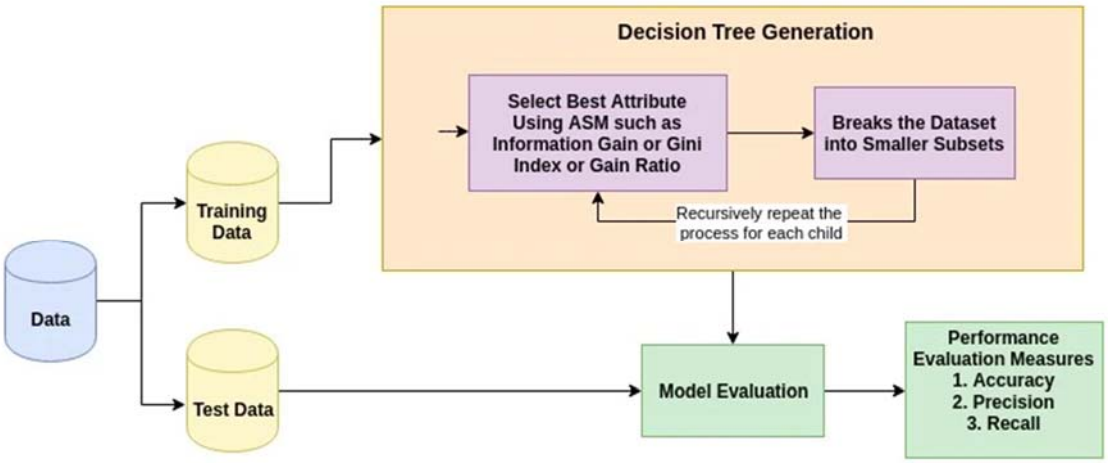

Figure 17 : Architecture globale du modèle de classification avec Arbre de Décision

3.3.3.3 Machine à vecteur support :

Dans cette approche nous avons entrainé notre SVM sur nos données d'entrainement en utilisant la bibliothèque Scikit-Learn qui contient des classes intégrées pour différents algorithmes SVM. Comme nous allons effectuer une tâche de multi classification, nous utiliserons la classe de classificateur de vecteur de support, qui est écrite comme SVC dans la svm bibliothèque de Scikit-Learn. Cette classe utilise un paramètre très important, qui est type de noyau. Dans le cas d'un SVM simple, nous définissons simplement ce paramètre comme "linéaire" car les SVM simples ne peuvent classer que des données séparables linéairement. (Figure 18)

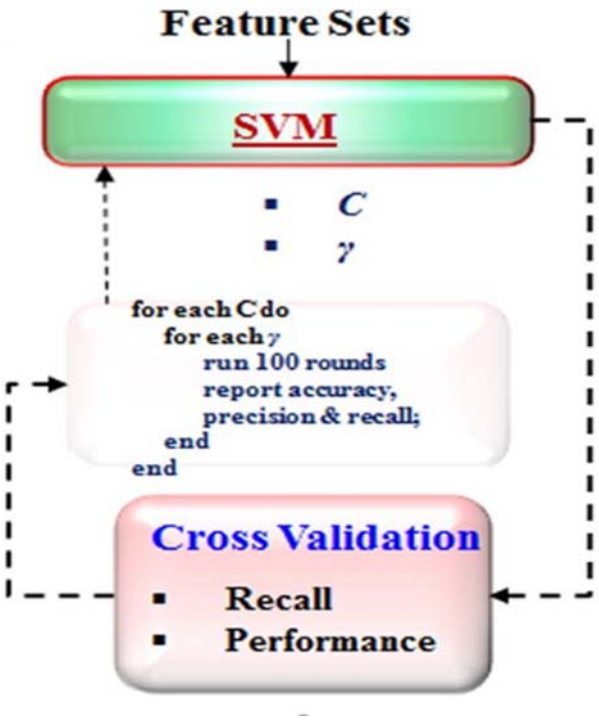

Figure 18 : Architecture globale du modèle de classification avec SVM.

Les algorithmes de Machine Learning cités plus haut ont été appliqué sur les données brutes, fournies par le capteur, ce type d'algorithme donne un résultat satisfaisant sur des DataSet où les donnes sont simplement séparable, et le nombre de classe est restreint, ce qui n'est pas toujours le cas. Pour cela les chercheurs se sont intéressés à proposer des descripteurs discriminant permettent de mieux séparer les données.

3.3.4 Histogramme de vecteurs de déplacement :

En se basant sur les travaux de [9], qui propose un descripteur basé sur l'histogramme du vecteur de déplacement, nous avons implémenté cette méthode pour créer un nouveau DataSet, ensuite nous avons appliqué le SVM pour classer les différentes activités. (Voir figure 19)

L'algorithme se résume en ce qui suit :

Pour chaque frame f

Pour chaque joint $i \in \{1, \dots \dots \dots, 15\}$

1-Discretiser l'espace euclidien au tours de chaque joint sur une grille de taille 27

$Q[3 * 3 * 3]$. Initialement tous les clusters de $Q^f = 0$

2-calculer le déplacement $d_f^i = [p_f^i - p_{f+T}^i], \forall\, f \in [1,2, \dots \dots . f_{max} - T]$

3-chercher l'indice de la direction q^*satisfaisant :

$q^* = \text{argmin}\left\|d_f^i - \sigma_q\right\| \forall\, q \in [1,2,\ldots\ldots,27]$ avec σ_q l'indice de q dans la grille

$$Q_p^F = \begin{cases} 1, & if\ q=q^* \\ 0, & sinon \end{cases}$$

5- Calculer l'histogramme de la fréquence de chaque direction $h^* = \sum_f Q^r$

6- La concaténation des h_i est le vecteur d'attribut $H = [h_1, h_2, \ldots\ldots\ldots, h_i]$

7_Classification classique

Fin

Fin

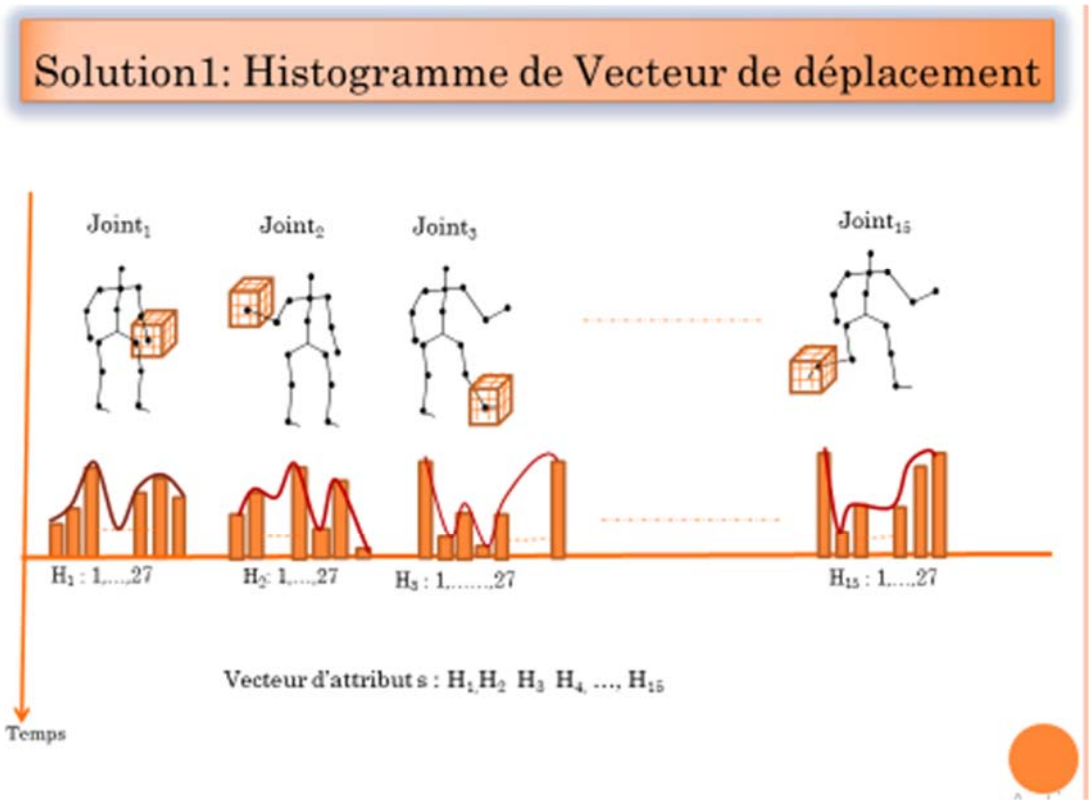

Figure 19 : Le cadre général de l'approche proposée.

Nous allons expliquer le schéma ci-dessus :

- Le procédé commence par le calcul des vecteurs de directions. Pour chaque joint i, P (i, frame(T)) représente la position cartésienne tridimensionnelle du joint i au niveau du frame à l'instant T. Les vecteurs de directions sont alors déduits pour chaque joint(articulation) en calculant la différence entre les coordonnées des joints(articulations) du frame de l'instant T et celui de l'instant T + 0,1s (Dans le cas de notre dataset, la différence temporelle entre les frames est de 0,1s).
- Mathématiquement, les vecteurs des directions des joints pour chaque frame sont représentés

Comme suit :

$$d_f^i = [p_f^i - p_{f+\dot{T}}^i], \forall\, f \in [1,2, \dots \dots . f_{max} - T]$$

- Pour chaque frame, nous partitionnons la région locale entourant un joint i en une grille spatiale tridimensionnelle. Nous choisissons 27 directions primaires dans la grille. Les directions choisies pour la grille représentent les directions du monde réel, c'est à dire HAUT, BAS, HAUT-GAUCHE, BAS-DROITE, …etc. Ce qui résulte en un total de 27 directions.
- Le vecteur de direction de chaque joint i, est mappé à l'index de une des 27 directions primaires dans la grille, et ceci en estimant la distance euclidienne tridimensionnelle entre les direction σ_q de la grille et le vecteur de direction d_f^i du joint i dans le frame f. C'est à dire que le but est de trouver l'index q^*de la direction qui est à la distance euclidienne minimale du vecteur de direction du joint i.

$$\rightarrow q^* = \text{argmin}\left\|d_f^i - \sigma_q\right\| \forall\, q \in [1,2, \dots \dots ..,27]$$

- Soit Q^f le vecteur des directions. Q_p^F Représente la valeur du vecteur Q^f à l'index q.
- L'index q^*est alors utilisé pour mettre à jour le vecteur des directions. Avec

$$Q_p^F = \begin{cases} 1, & if\ q=q^* \\ 0, & sinon \end{cases}$$

- Pour obtenir le nombre total de fois qu'une direction particulière a été prise durant une activité (10 frames), nous performant une addition cumulative des vecteurs Q^f

Au niveau de chaque frame de cette dernière, comme le montre l'équation $\rightarrow h^* = \sum_f Q^r$

Ou h* représente le vecteur contenant le nombre de fois qu'une direction a été prise par un joint(articulation) i durant le cours d'une activité.

- Le vecteur h^* est alors normalisé pour calculer le vecteur h_i du joint i. la normalisation du vector h^* nous offre l'histogramme h_i représentant la probabilité de l'occurrence de chaque

Direction pour un joint(articulation) particulier durant le cours d'une activité ➔$h_i = \frac{h_i}{\|h^*\|}$

- En plus de cela, chaque histogramme h_i sera concaténé pour générer l'histogramme des vecteurs de directions finale ➔

$$H = [h_1, h_2, \ldots\ldots\ldots\ldots, h_i]$$

- Ainsi pour chaque activité, représentée par une ligne dans de notre DataSet initiale, nous obtiendrons un l'histogramme de vecteurs de directions

3.3.5 Module d'évaluation

Le choix du meilleur classificateur n'est pas toujours trivial. Les performances d'aussi complexes algorithmes dépendent de la nature des données capturées, des activités considérées et du jeu de données utilisé. Il s'agit donc d'un procédé expérimental consistant, ceci revient principalement à comparer le vecteur d'activités prédites Y' avec le vecteur réel des étiquettes de classes Y.

L'évaluation de la qualité d'un système de classification nécessite un certains calculs de comparaison basés sur des métriques de mesure de performances.

- **Métriques d'évaluation dans l'apprentissage automatique**

❖ **Matrice de confusion**

Les résultats de la classification sont organisés en une matrice de confusion $M_{n;n}$ pour un problème à n classes, tel que $M_{i;\,j}$ représente le nombre d'instances de classe i assignés à la classe j. A partir de la matrice M d'un problème binaire (voir Figure 20), les valeurs suivantes sont calculées :

- **Vrai positifs« VP» (True Positive)**: le nombre d'instances positives classifiées comme tel.

- **Faux positifs « FP» (False Positive**) : le nombre d'instances négatives classifiées comme positives.

- **Vrai négatifs « VN» (TrueNegative):** le nombre d'instances négatives classifiées comme tel.

- **Faux négatifs « FN» (False Negative):** le nombre d'instances positives classifiées comme négatives.

	Décision « Positif »	Décision « Négatif »	
Etiquette « Positif »	Vrai positif, VP	Faux négatif, FN	Total Positifs
Etiquette « Négatif »	Faux positif, FP	Vrai négatif, VN	Total Négatifs

Figure 20 : Matrice de confusion pour un problème binaire à 2 classes.

Dans le cas d'une classification n-aire, une instance peut être positive ou négative relativement à une certaine classe. Par exemple, les positifs peuvent être les instances où l'activité effectuée est Marcher. Autrement, ce sont des négatifs. (Voir figure 21)

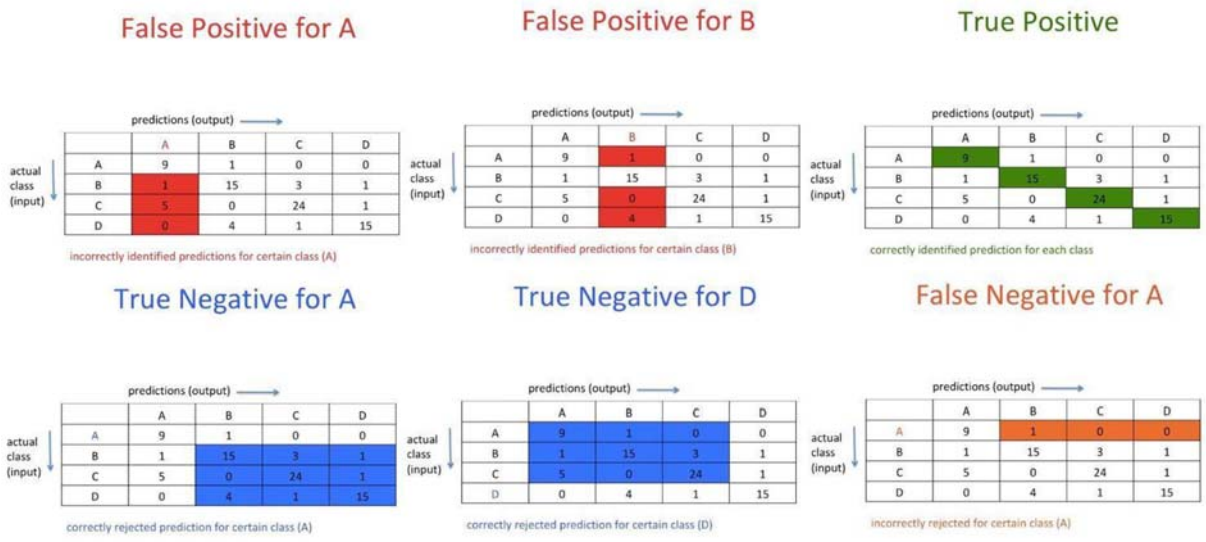

Figure 21 : Exemple des instances positives et négatives d'une classification n-aire

➢ Taux d'exactitude (accuracy)

Il s'agit de l'indicateur le plus évident permettant d'évaluer les performances d'un système de classification. Cette valeur, simple à calculer, correspond au nombre d'éléments correctement identifiés par le système. La définition du Taux d'exactitude est :

$$\textbf{Taux d'exactitude} = \frac{VP+VN}{Nombre\ d'élement\ totale} \qquad [13]$$

- **Précision**

La précision peut être considérée comme une mesure de l'exactitude ou de la qualité, c'est la fraction des instances récupérées qui sont pertinentes, i.e. le nombre d'exemples positifs correctement classés divisé par le nombre d'exemples étiquetés par le système comme positifs. Donc la précision peut être également considérée comme une mesure de la pertinence. Sa formule est :

$$\textbf{Précision} = \frac{VP}{VP+FP} \qquad [14]$$

- **Rappel ou taux de vrais positifs**

Rappel est utilisé pour l'évaluation d'une solution pour la détection positive. Il est égal à la fraction des instances pertinentes qui sont récupérées, i.e. le nombre d'exemples positifs correctement classés divisé par le nombre d'exemples positifs réellement.

Rappel peut être considère comme une mesure de la pertinence. Ce n'est rien d'autre que le vrai taux positif d'une classe L'expression du Rappel se présente comme suit :

$$\textbf{Rappel} = \frac{VP}{VP+FN} \qquad [15]$$

Rappel et Précision sont des mesures de performance importantes. Elles sont intimement liées (relation inverse). Plus la précision est élevée, moins d'efforts sont consacrés aux tests et à l'inspection, et plus le Rappel est important, plus les classes réelles sont détectées.

- **F-Mesure**

F-Measure est définie comme le nombre de cas de test requis pour détecter la première défaillance. Elle peut être interprétée comme une moyenne pondérée de la Précision et du Rappel.

$$\textbf{F-Mesure} = \frac{2.\text{Précision.Rappel}}{Précision+Rappel} \qquad [20]$$

3.4 Conclusion

A travers ce chapitre, nous avons exposé la conception de la solution que nous proposons au problème posé à savoir : réaliser un système de classification des activités humaines. Nous avons réorganisé notre DataSet selon nos besoins. Pour entrainer un classifieur (KNN, SVM ou Arbre de **D**écision**).** Ainsi l'histogramme de vecteurs de déplacement. Nous allons à la fin évaluer ces différentes méthodes afin de comparer les résultats obtenus.

Dans le chapitre qui suit nous allons présenter la réalisation de notre solution, les tests et les résultats obtenus, en utilisant différentes DataSets.

Réalisation de la solution et Test

Chapitre 4 : Réalisation de la solution et Test

4.1. Introduction

Après avoir décrit notre solution de façon conceptuelle dans le chapitre 3, nous exposons dans ce présent chapitre les technologies et les outils utilisés, nous décrivons le DataSet utilisé pour l'évaluation. Ensuite, nous présentons les résultats obtenus et évaluons les performances des méthodes de classification étudiées à travers les métriques d'évaluation choisies dans la phase conception. Enfin, nous allons présenter une étude comparative entre les différentes méthodes utilisées afin d'améliorer les performances de nos modèles.

4.2. Technologies utilisées

Pour réaliser les différentes étapes conçues, nous avons dû utiliser plusieurs technologies. Notre choix s'est porté sur des technologies réputées stables, que nous maitrisons.

4.2.1. Robot Operating System (ROS)

ROS est un méta-système d'exploitation open-source lancé la première fois en Janvier 2009, conçu pour être installé sur le système d'exploitait

Figure 22: : Logo de ROS

On Ubuntu (d'autres implémentations expérimentales pour l'installation sur d'autres systèmes d'exploitation existent). I1 fournit un Framework pour l'écriture des logiciels robotiques, avec plusieurs outils intégrés et d'autres publiés comme code open-source par la communauté des développeurs ROS, l'un des plus grands avantages de ROS.

La dernière version stable recommandée par les développeurs de ROS à la date d'aujourd'hui est ROS Indigo Igloo, la version précédente était ROS Hydro Medusa qui est devenue obsolète en Mai 2015.

Les logiciels ROS s'organisent dans des paquets 'Packages', où un paquet peut contenir des fichiers de configuration, des fichiers de descriptions, des définitions de types de messages ou de code qui peut contenir un ou plusieurs nœuds 'Nodes', qui est l'instance d'un exécutable sous ROS. Ces nœuds peuvent communiquer entre eux en utilisant plusieurs mécanismes

comme le système de transport d'information basé sur le système de l'abonnement/publication (en publiant ou en souscrivant à un ou plusieurs Topics'), ou l'appel d'un 'service' qui est un système de transport d'information basé sur la notion d'appel de procédure à distance (RPC). L'échange de l'information s'effectue soit de manière asynchrone via les 'Topics ou de manière synchrone via les 'Services'.

ROS Supporte plusieurs robots populaires tels que PR2 et TurtleBot. 11 fournit une couche d'abstraction matérielle et un modèle de description du robot ainsi que des paquets pour la simulation de ces robots. Pour les autres robots, leurs drivers et leur modèle doivent être fournis par les développeurs.

4.2.2. Python

Python est un langage orienté objet de haut niveau, il est l'un des langages de programmation les plus efficaces pour le développement Web. Python est utilisé dans différents domaines d'application (industrie, application de recherche et scientifique…), de plus il est facile, rapide et interprété (ne nécessite pas de compilation avant exécution), puissant, évolué et plus structuré (lisible). Il est connu aussi par sa simplicité syntaxique (proche du langage algorithmique et langage mathématique) et sa couverture de plusieurs bibliothèques surtout les bibliothèques scientifiques.

Figure 23: Logo de Python

4.2.3 eclipse

Eclipse est un environnement de développement intégré libre (le terme *Eclipse* désigne également le projet correspondant, lancé par IBM) extensible, universel et polyvalent, permettant potentiellement de créer des projets de développement mettant en œuvre n'importe quel langage de programmation. Eclipse IDE est principalement écrit en Java (à l'aide de la bibliothèque graphique SWT, d'IBM), et ce langage, grâce à des bibliothèques spécifiques, est également utilisé pour écrire des extensions.

Figure 24:Logo de Eclipce

4.3 Outils utilisés

Pour mettre en œuvre les technologies précédemment citées, il est nécessaire de maitriser une panoplie d'outils qui aident à les utiliser. Voici donc les différents outils que nous avons utilisés dans notre projet.

4.3.1 Scikit-learn

Scikit-learn est une bibliothèque libre python destinée à l'apprentissage automatique. Elle est développée par de nombreux contributeurs notamment dans le monde académique par des instituts français d'enseignement supérieur et de recherche comme **Inria** et **Télécom ParisTech**.

Figure 25 : Logo de Scikit-learn

Elle comprend notamment des fonctions pour estimer des forêts aléatoires, des régressions logistiques, des algorithmes de classification, et les machines à vecteurs de support. Elle est conçue

Pour s'harmoniser avec d'autres bibliothèques libres python, notamment NumPy et SciPy.

4.3.2 Rviz

Rviz est un outil de visualisation de ROS qui permet la visualisation de différents types de messages ROS, Nous utilisons cet outil sur le laptop pour la visualisation du squelette humain.

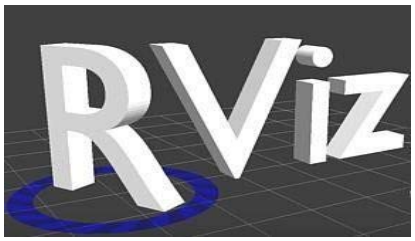

Figure 26: ***Logo de Rviz***

4.3.3 OpenNI

Le framework OpenNI est une couche abstraite capable de s'interfacer avec différents types de matériel et fournissant des fonctions pour le développement d'applications utilisant des interactions naturelles. OpenNI supporte quatre types de matériel :

- Des capteurs 3D tels que la Kinect.
- Des caméras RGB comme celle intégrée dans la Kinect.
- Des caméras infrarouges.
- Des appareils audio (microphone ou réseau de microphones).

OpenNI propose des fonctions correspondant à quatre types de traitements :

L'analyse du corps : récupération de la pose du corps sous forme d'une information sur la position et l'orientation des différentes articulations du corps humain.
- L'analyse du pointage avec la main : permet de suivre la position d'une main pointant vers la caméra.
- La détection de certains gestes particuliers comme, par exemple, deux mains qui applaudissent.
- L'analyse de la scène : extraction de l'avant-plan, segmentation du sol, etc.

Il n'existe pour le moment que l'implémentation fournie par le middleware propriétaire NITE.

4.4 Implemntation de la solution

4.4.1 Module d'acquisition et préparation de données

4.4.1.1 Notre DataSet

Dans cette partie nous allons expliquer les différentes étapes de la création des différents DataSets utilisés, que nous avons décrit dans la partie conception.

Nous avons utilisé le capteur Kinect de Microsoft XBOX 360 pour enregistrer nos vidéos. Chaque vidéo est enregistrée avec le framework OpenNI à la résolution de 640x480 avec un format de fichier (.bag).

Chaque vidéo commence par une pose psi (surrender) pour l'étalonnage dans OpenNI skeleton tracker. Le jeu de données comprend 6 activités différentes avec 100 vidéos enregistrées pour chaque activité.

Notre algorithme d'acquisition de données a été mis en œuvre à l'aide du système d'exploitation robot (ROS), nous avons utilisé le package OpenNI tracker pour fonctionner avec les vidéos qu'on a enregistrés afin d'obtenir la structure de donnée définie dans la partie conception. Le

traqueur peut détecter l'utilisateur instantanément, mais nécessite une pose en psi pour l'étalonnage, après il commence à suivre le squelette de l'utilisateur. Une fois calibré, il commence à publier des positions et des rotations 3D de 15 articulations du squelette par rapport à un cadre de référence fixe. Les squelettes publiés incluent les pieds, les genoux, les épaules, les mains, les coudes, les hanches, la tête, le cou et le torse. La grande importance du traqueur OpenNI s'apparaît dans le fait qu'il puisse suivre pratiquement toutes les articulations qui nous aident pour la reconnaissance des activités humaines.

- **Construction de vecteurs des données**

La construction de vecteurs de données est l'étape la plus cruciale de ce travail ; il s'agit de décider du choix des articulations ou parties du corps qui doivent être suivies pour chaque activité.

Le traqueur OpenNI publie des positions 3D et des rotations par rapport à la Kinect de 15 articulations, si nous suivons toutes les liaisons disponibles pour la construction de vecteurs de données, l'algorithme deviendra lourd en calculs et pourrait ne pas fonctionner en temps réel.

En outre, toutes les articulations ou parties du corps ne subissent pas de changement de position dans une aucune activité donnée. Il est donc naturel de sélectionner les articulations et les parties du corps qui doivent être suivi pour la construction de vecteurs de données pour chaque activité. L'OpenNI tracker publie les positions de toutes les articulations par rapport à un cadre de référence fixe. Si nous utilisons la position par rapport à un référentiel fixe, la même activité commise à différentes positions devant la caméra peut donner différents résultats. Nous avons donc utilisé la position commune par rapport à une autre position commune pour tenir compte des différentes positions de l'utilisateur devant la caméra. Pour cela nous avons choisi le torse un point de référence pour les activités Appeler, Arrêter, Saluer et Pointer et le centre de la kinect pour les activités Venir et Partir.

Dans tous ces activités nous avons utilisé les mêmes articulations (Tête, centre d'épaule, le

Torse, épaule droite, épaule gauche, coude droit, coud gauche, main droite, main gauche).(Voir tableau 3)

Activité	Point de référence
Arrêter	Le torse
Appeler	Le torse
Saluer	Le torse
Venir	Le centre de la Kinect
Partir	Le centre de la Kinect
Pointer vers	Le torse

Tableau 3 : résumé des articulations et leurs points de références suivies pour chaque activité.

4.4.1.2 DataSets utilisées

Nous avons utilisé 4 DataSet, parmi eux 3 sont enregistrées par nous même au niveau du Centre de développements de technologies avancées (CDTA) à l'aide du capteur Kinect, notamment : Dataset_4, Dataset_10, Dataset_20. Plus Data_Histo que nous avons obtenu par le descripteur Histogramme de vecteurs de déplacement que nous avons implémenté sur le Dataset_10.

4.4.1.3 Prétraitement de données

Nous avons presque ignoré cette étape, car nos données sont obtenu d'une même source, et sont de même type "réels".

Nous avons essayé de minimiser les bruits de notre DataSet, pour cela, nous avons visualiser notre DataSet à travers le plot, dans le but de montrer la conjonction et la disjonction entre les différents attributs, et d'éliminer celles qui constitue un bruit, ce bruit est dû par les attributs embrouillés.

Voici le schéma (figure 27) d'un seul exemple de plot, chaque classe est représentée par une couleur.

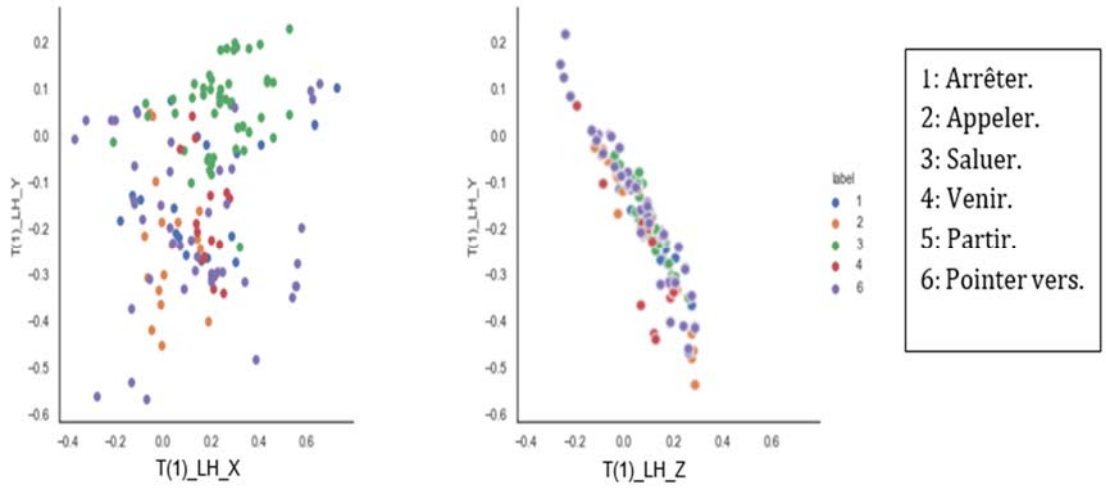

Figure 27: plot d'une tranche de notre DataSet

Nous avons terminé cette étape par supprimer quelques attributs qui sont embrouillés entre eux (T(3)_Lelm_rll, T(5)_Lelm_X), non seulement nous avons réduire le nombre de colonnes mais aussi nous avons remarqué que par la suite les taux de reconnaissance on était augmentés par 0.02.

4.4.2 Le module de génération des modèles

Après la collecte et la préparation de données, viens la phase qui consiste à entrainer le classificateur afin de construire et générer des règles sous forme de modèle pour une classification ultérieure de nouveaux échantillons.

4.4.2.1 Démarche de génération des modèles

La stratégie d'entrainement et de validation effectuée pour chaque classifieur (KNN, SVM, Arbre de Décision) est ainsi décrite :

- Diviser le DataSet en deux parties en utilisant la fonction ***train_test_split*** : une partie pour l'entraînement (Train_Data : la partie DataSet dédiée à l'entrainement du modèle) et l'autre pour la validation des performances (Validation_Data : la partie de DataSet dédiée au validation des performances de modèle).

  Nous n'avons pas choisi le pourcentage de validation aléatoirement. Mais, plutôt nous avons entrainé les modèles par plusieurs valeurs, le meilleur que nous avons trouvé c'est 12 % (***test_size=0.12***).

Aussi le paramètre ***shuffle*** qui a pour objectif de mélanger les données avant de les séparer, nous avons choisi " ***shuffle = False*** ", pour prendre le même pourcentage de test pour chaque classe équitablement.

Il y'a des paramètres qui ne participes pas à l'amélioration de notre taux, donc nous les avons gardés par défaut.

- Définir et initialiser le classifieur en utilisant des fonctions prédéfinies.
- Entraîner le modèle avec la fonction " ***Fit*** ", elle accepte en entrée le DataSet d'entraînement, l et retourne un modèle entrainé et prêt à subir des tests et par la suite déployer dans son environnement réel (voir figure 28).

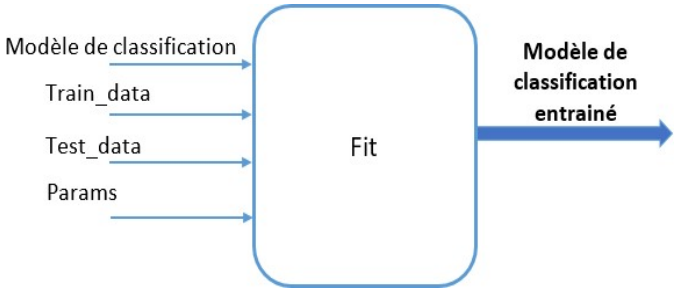

Figure 28 : La Fonction Fit .

- Obtenir les valeurs prédites des données testées.
- Calculer les différentes métriques de performance sur les DataSet.
- Sauvegarder le modèle avec ces paramètres si celui-ci a un meilleur taux de réussite que le précèdent.

Nous aurons à la fin de cette procédure :

- La sauvegarde du meilleur modèle aboutissant au meilleur taux de réussite.

Nous avons parlé dans le chapitre précédant de la phase la plus contraignante de la méthode de classification KNN, qui est le choix du paramètre ***K***.

Le schéma suivant montre les valeurs de ***K*** et le taux d'erreur qui leur sont associer. (Voir figure 29)

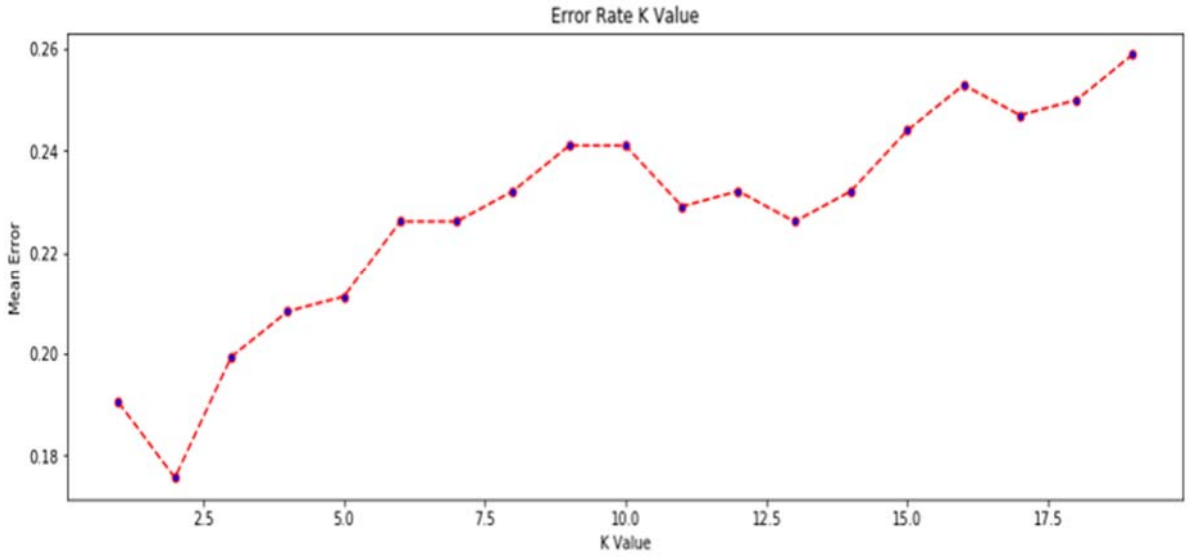

Figure 29 : courbe graphique des taux d'erreurs de l'algorithme de classification KNN.

Pour l'implémentation des modèles conçus nous avons utilisé la bibliothèque « Scikit-learn » qui fournit des fonctions dédiées à l'apprentissage automatique.

Après ce traitement, le modèle de classification entrainé est prêt pour être évalué et déployé dans un environnement réel pour reconnaître et classifier des activités humaines.

4.5 Test et évaluation

4.5.1 Critères d'évaluation

Afin d'évaluer notre système, nous avons fait appel aux différents critères de performances d'apprentissage cités dans le chapitre 3, donc la matrice de confusion serait interprétée comme suit :

- **Les vrais positifs (VP)** : Nombre d'activités bien prédites dans la classe à juste titre.
- **Les vrais négatifs (VN)** : Nombre d'activités prédites comme n'étant pas dans la classe à juste titre.
- **Les faux positifs (FP)** : Nombre d'activités prédites dans la classe alors qu'ils ne devraient pas en faire partie.

- **Les faux négatifs (FN)** : Nombre d'activités prédites comme étant de la classe alors qu'ils ne le sont pas en vrai.

Nous pouvons donc exploiter les différents critères d'évaluation définis dans le chapitre 3, à savoir :

- Le taux d'exactitude.
- La Précision.
- Le rappel.
- F-mesure.

4.5.2 Présentation des tests

Dans ce qui suit, nous présenterons les résultats des différents tests effectués, pour évaluer les performances des méthodes de classification, que nous avons proposées, sur les 4 DataSet, que nous les avons parlé précédemment (Data_04, Data_10, Data_20, Data_Histo).

Pour les trois méthodes de classification que nous avons défini dans le chapitre 2 (KNN, SVM, Arbre de Décision) la partie d'entrainement et la partie d'évaluation des performances ont été faite sur nos DataSet tel que :

- Train_Data = 88 % des données choisies aléatoirement pour chaque DataSet
- Validation_Data =12 % des données choisies aléatoirement pour chaque DataSet.

4.5.2.1 Data_10

Nous avons remarqué que dans le DataSet (Data_10) les différentes activités ont été très bien classés pour les deux méthodes de classification Arbre de Décision et SVM avec un taux de classification supérieur à 97% pour SVM (voir figure 30) et supérieur à 95% pour Arbre de Décision (voir figure 31), mais KNN a donné un taux moins élevé que ces deux derniers, des taux un peu bas pour 'Appeler' et 'Pointer' avec un taux de classification qui est égal à 81.25% et 89.65% respectivement (voir figure 32) , ceci est dû au fait que sa reconnaissance est très liée avec le déplacement de la main (confusion Pointer/Appeler).

Il est bien clair que pour les trois méthodes de classification les activités 'Saluer', 'Venir' et 'Partir' sont totalement bien classé, elles sont obtenus un taux de 100%. Dès le début nous avons attend cette situation, car la distance entre le capteur et le torse suffit pour reconnaitre l'activité.

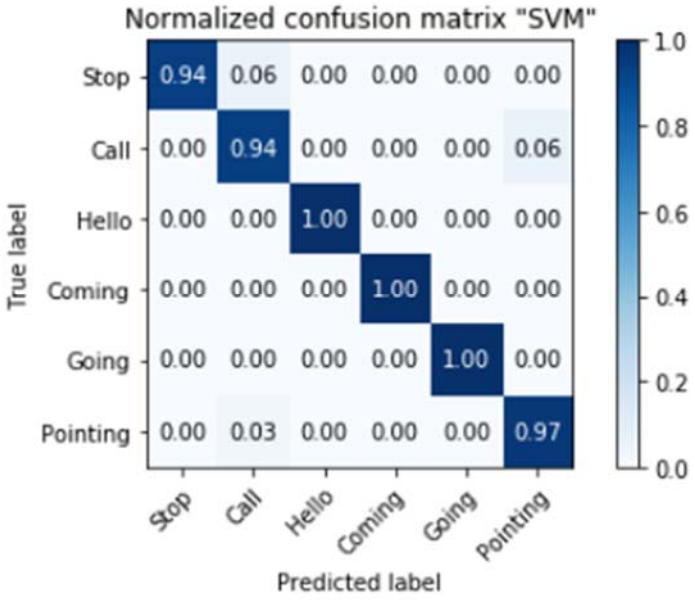

Figure 30 : Matrice de confusion des activités en utilisant SVM sur Data_10.

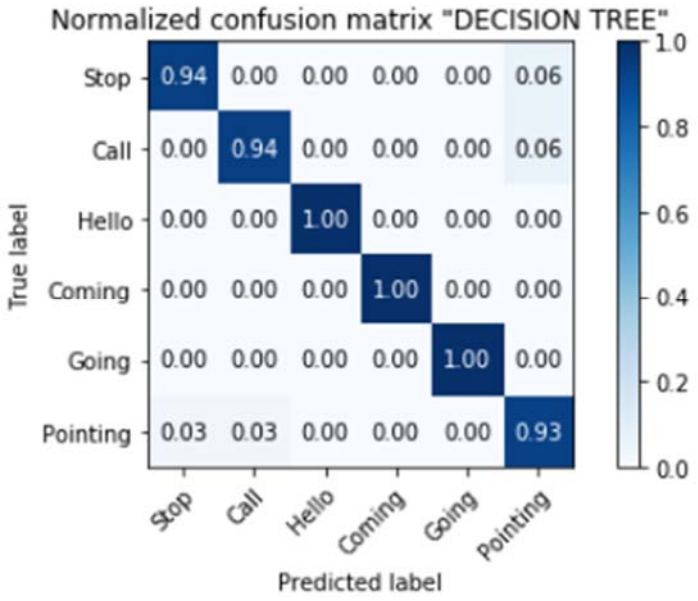

Figure 31 : Matrice de confusion des activités en utilisant Arbre de Decision sur Data_10.

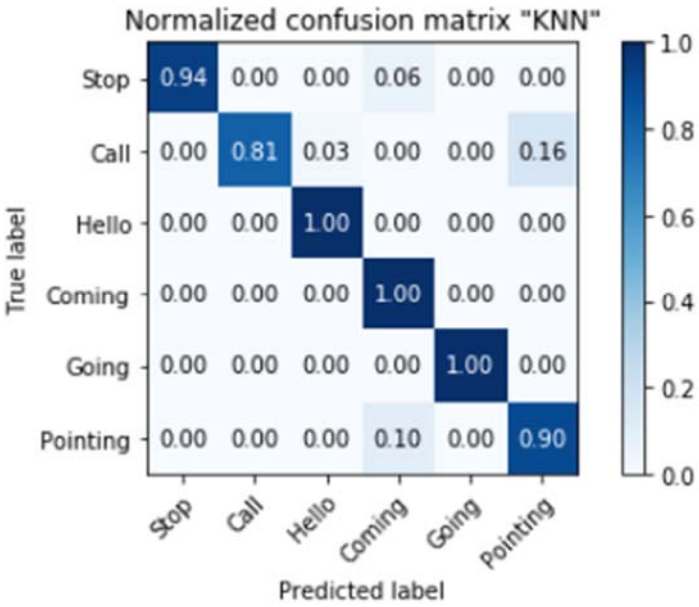

Figure 32: Matrice de confusion des activités en utilisant KNN sur Data_10.

4.5.2.2 Data_Histo :

Nous avons remarqué que dans ce DataSet (Data_Histo) les différentes activités ont été bien classés avec la méthode SVM avec un taux de classification supérieure à 93% pour SVM. Il (voir figure 33).

Les méthodes KNN et Arbre de Décision ont été un peu bien classé, avec un taux de 85% ,90% (respectivement). Ceci est bien détaillé dans les figures (34,35).

Concernant le KNN, il avait considéré l'activité "Pointer vers " comme "Arrêter ", " Appeler" et « Saluer", il fait une confusion entre eux.

Ce n'est pas les taux de reconnaissance souhaités car ça nécessite un nombre de Caractéristiques assez important.

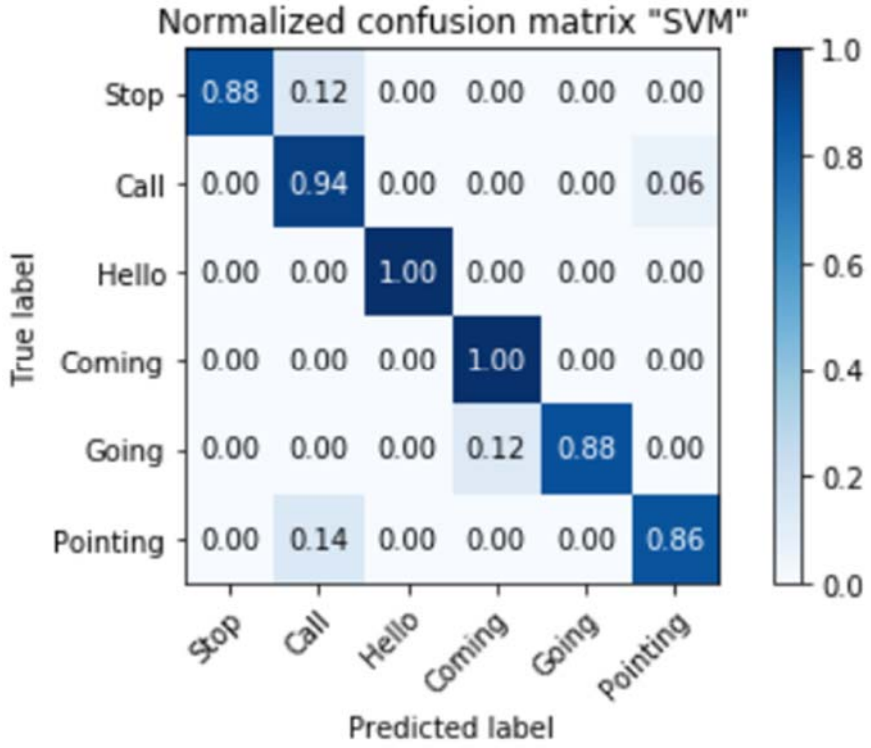

Figure 33 : Matrice de confusion des activités en utilisant SVM su Data_Histo.

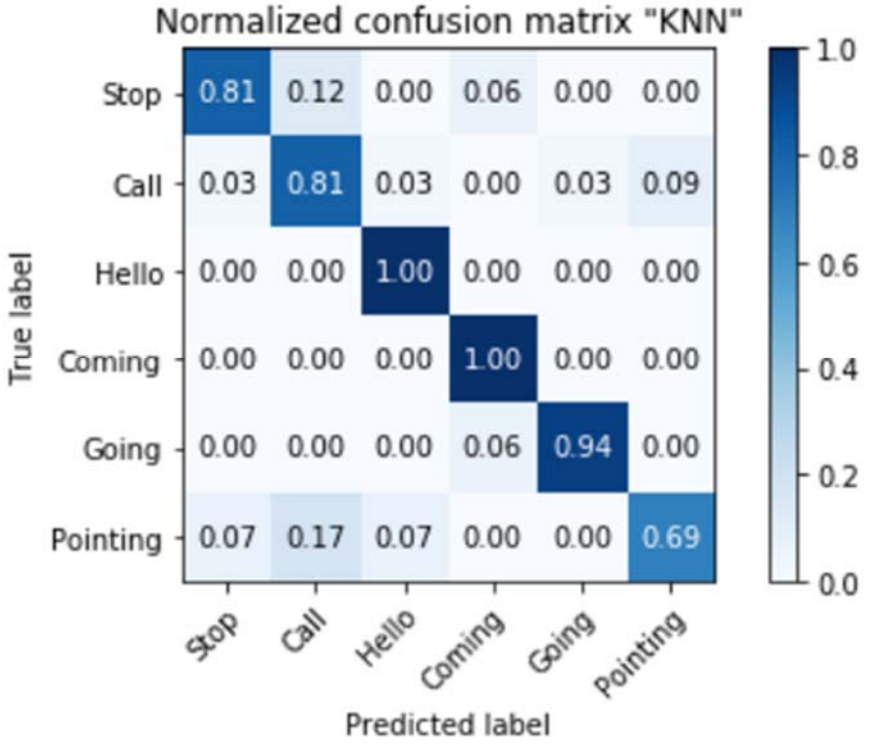

Figure 34 : Matrice de confusion des activités en utilisant KNN sur Data_Histo .

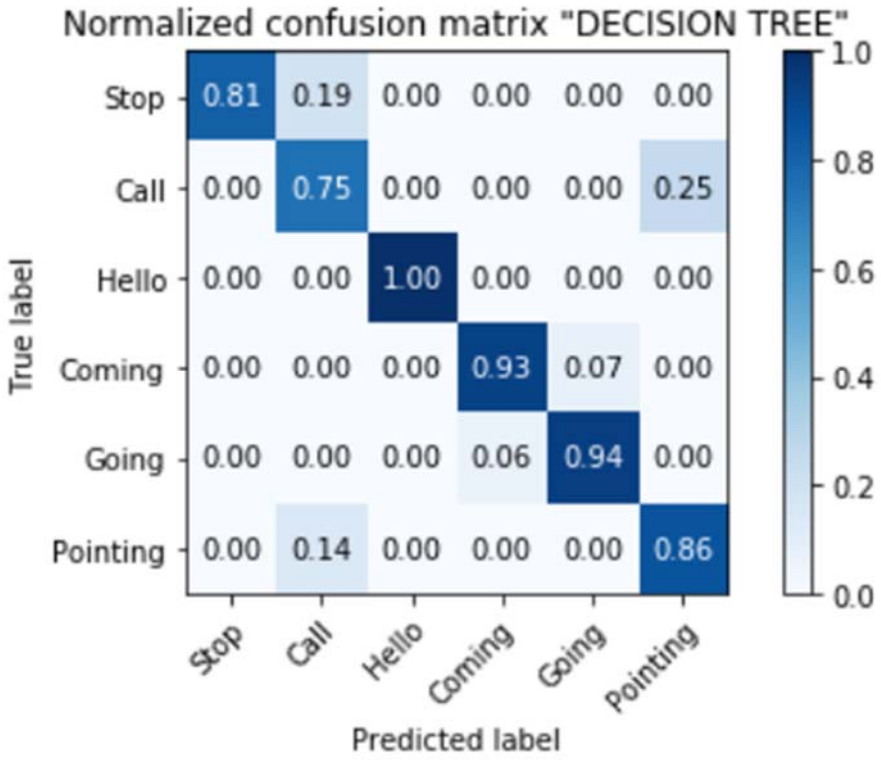

Figure 35 : Matrice de confusion des activités en utilisant KNN sur Data_Histo .

4.5.2.3 Data_20

Concernant cette DataSet nous n'avons rien à dire, il est bien clair que toutes les méthodes de classification sont bien classées, pour SVM et Arbre de Décision nous avons obtenus un taux de 94% avec SVM, et 98% avec Arbre de Décision (voir figure36 et 37).

Concernant KNN nous avons atteint le plafond avec une reconnaissance total 100% (voir figure 38), c'est la meilleure solution souhaitée, mais lors de test en temps réel ce n'est pas le cas, car ça a pris du temps pour faire 20 itérations.

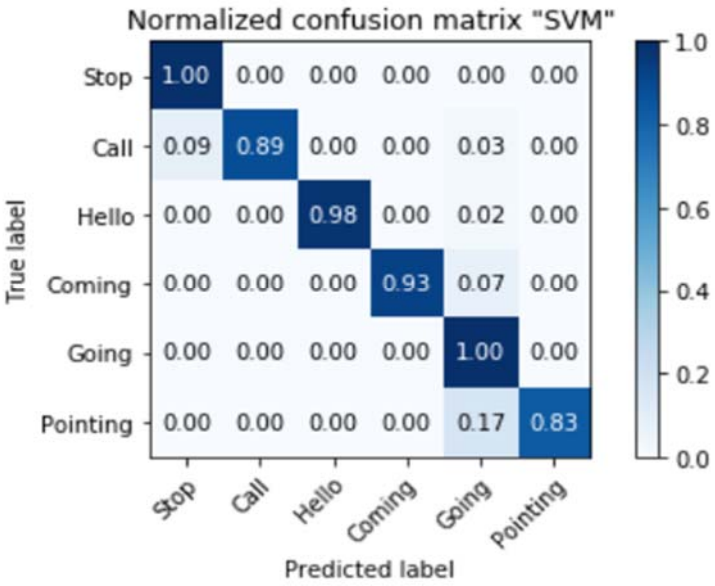

Figure 36 : Matrice de confusion des activités en utilisant SVM sur Data_20.

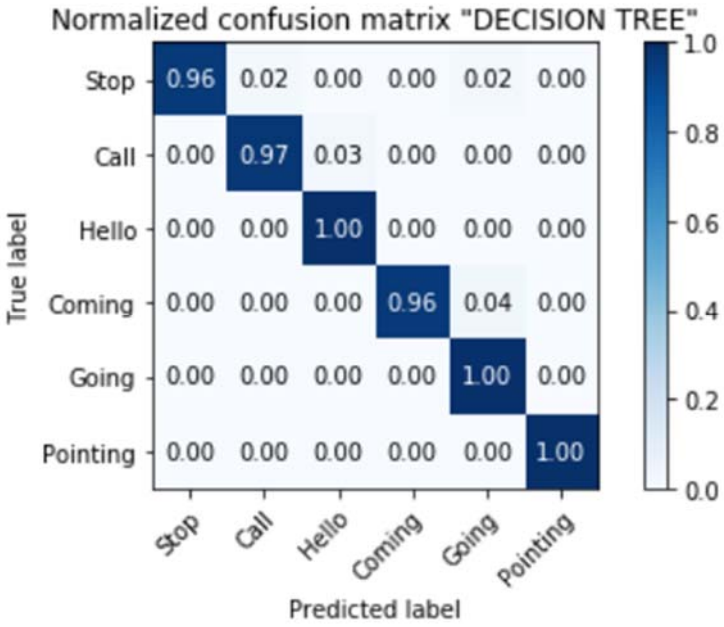

Figure 37 : Matrice de confusion des activités en utilisant Arbre de Décision sur Data_20.

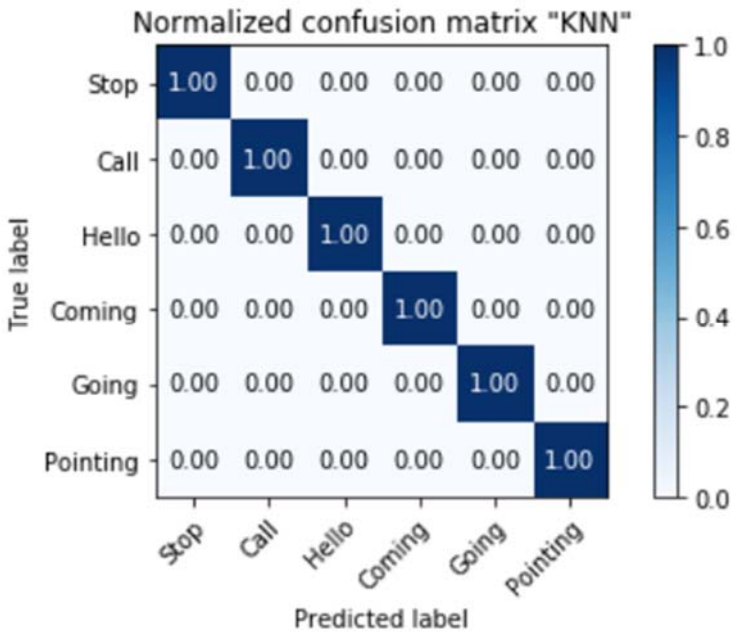

Figure 38 : Matrice de confusion des activités en utilisant KNN sur Data_20.

4.5.3 Tableau récapitulatif des critères de performance

On récapitule les différents résultats des tests dans les tableaux suivants :

	Méthode	Taux d'exactitude	Précision	Rappel	f-mesure
Data(04)	**SVM**	73.58	49.32	73.58	65.92
	KNN	75.47	81.83	75.47	74.03
	Arbre de Décision	**94.33**	**91.42**	**94.33**	**94.32**
Data(10)	**SVM**	**96.69**	**97.84**	**96.69**	**96.70**
	KNN	91.73	92.49	91.73	91.79
	Arbre de Décision	94.21	95.21	94.21	94.20

Data(20)	**SVM**	93.88	93.85	93.88	94.14
	KNN	**100**	**100**	**100**	**100**
	Arbre de Décision	98.25	97.90	98.25	98.26
Data_Histo	**SVM**	91.73	93.90	91.73	91.78
	KNN	**84.29**	**85.09**	**84.29**	**83.93**
	Arbre de Décision	85.95	89.96	85.95	86.04

Tableau 4 : Résultats des tests obtenus par l'application des algorithmes proposés sur les DataSet

Dans l'implémentation de l'algorithme de l'histogramme de vecteur de déplacement nous avons varié plusieurs valeurs pour le paramètre ***'pas'*** et appliqué ***ces*** dernières sur les 3 méthodes de classification SVM, KNN, Arbre de Décision pour avoir un bon résultat (voir Tableau 4 et figure 39)

	Taux de reconnaissance de KNN	**Taux de reconnaissance de SVM**	**Taux de reconnaissance d'Arbre de décision**	**PAS**
Data_10	38.%	57%	33%	0.001
	84%	25%	73%	0.005
	85%	25%	73%	0.008
	85%	25%	79%	0.01
	87%	**25%**	**86%**	**0.09**
	77%	25%	77%	0.5
Data_04	60%	36%	53%	0.001
	60%	36%	60%	0 .005

Tableau 5 : Résultats des tests obtenus par l'application d'histogramme et les algorithmes proposés sur Data_10.

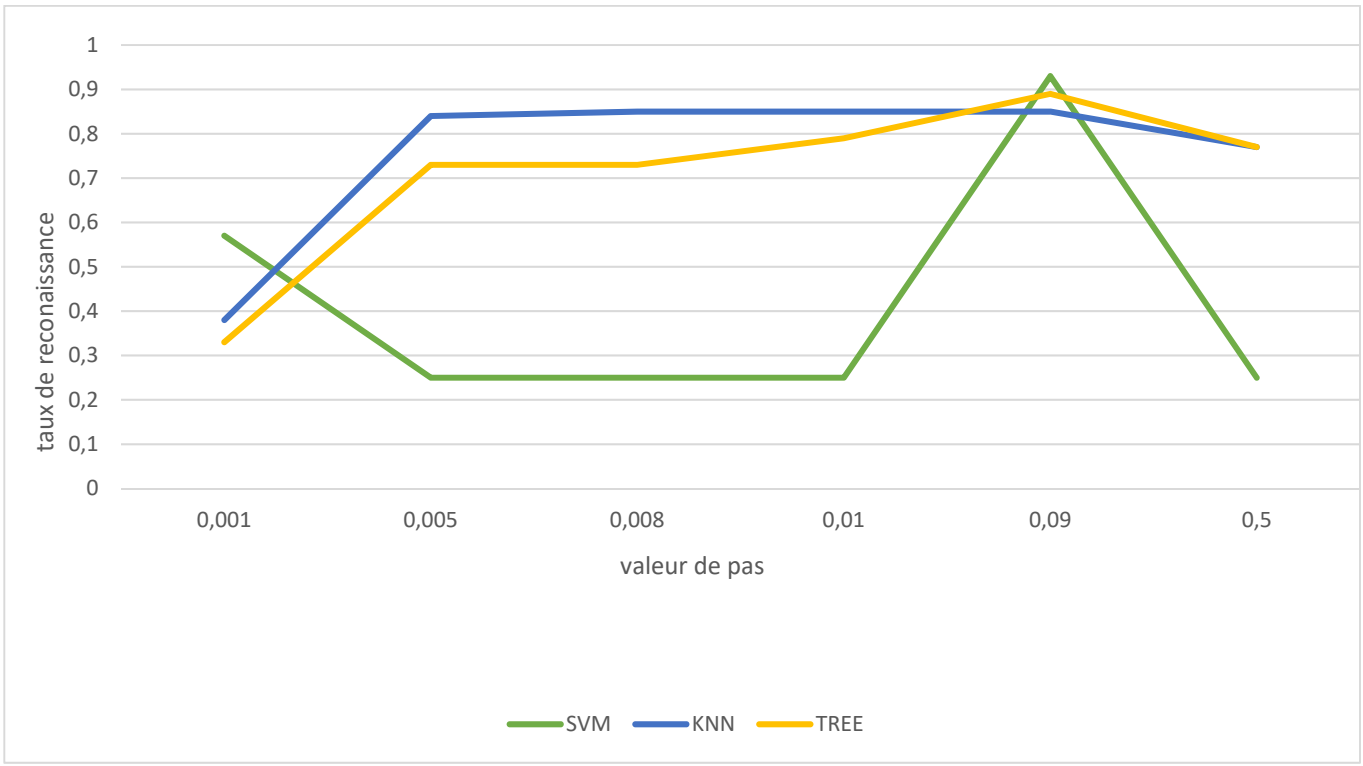

Figure 39 : courbe graphique de Résultats des tests obtenus par l'application d'histogramme et des algorithmes proposés sur Data_10 .

4.5.4 Graphe comparatif

Dans cette étape, nous présenterons un histogramme comparatif des méthodes de classification pour nos DataSet (Data_04, Data_10, Data_20, Data_Histo) en utilisant les mêmes métriques d'évaluation.(Voir figures 40,41 et 42)

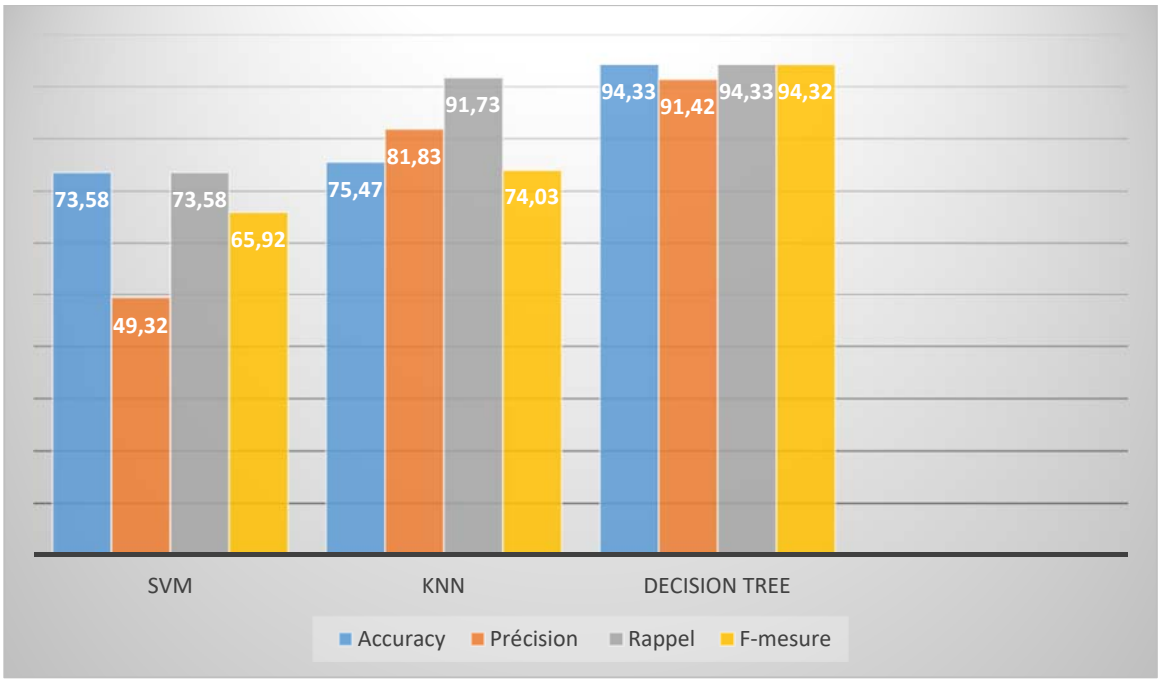

Figure 40 : Histogramme comparatif des méthodes de classification de Data_04 .

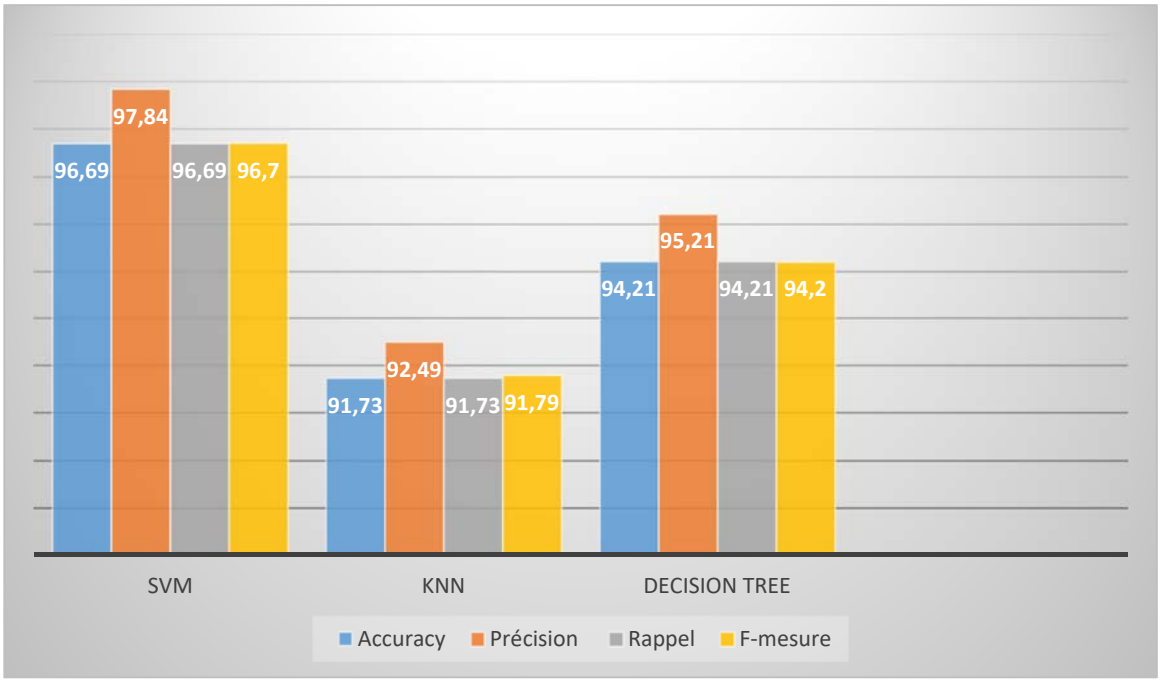

Figure 41 : Histogramme comparatif des méthodes de classification de Data_10.

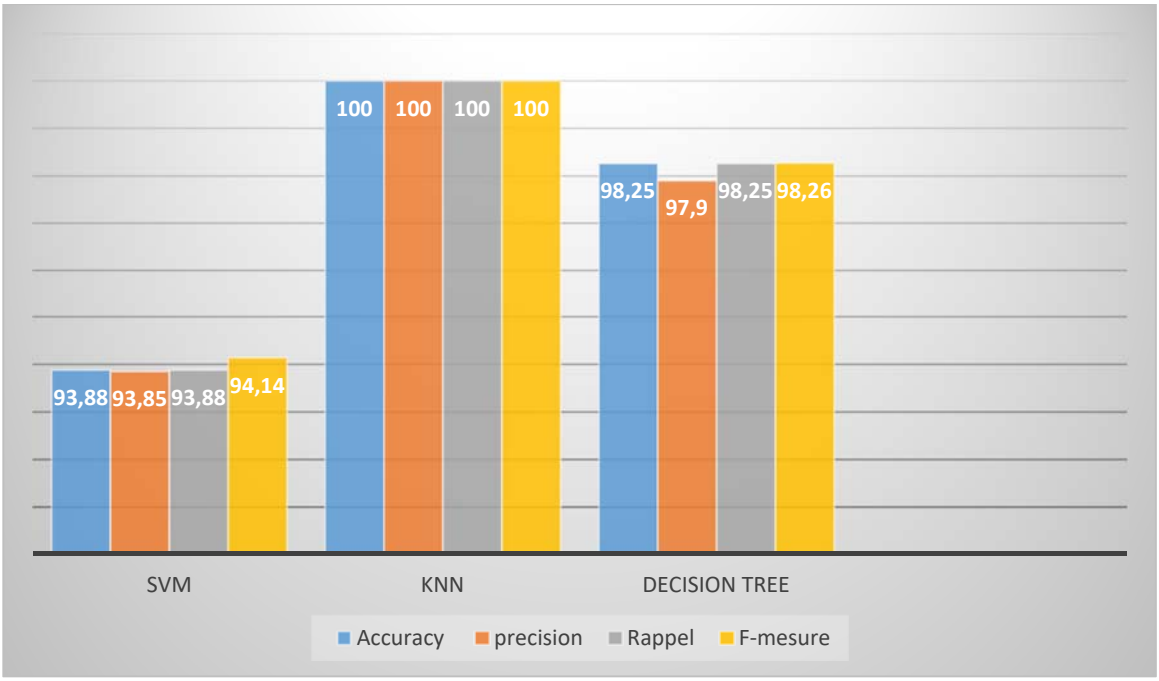

Figure 42 : Histogramme comparatif des méthodes de classification de Data_20 .

Après avoir comparé les histogrammes précédemment cités, qui démontre les résultats obtenus à partir de la classification des 3 méthodes (SVM, KNN, Arbre de Décision) sur les DataSets (Data_04, Data_10, Data_20) nous avons remarqué que les taux de reconnaissance de :

- Data_04 sont moins élevés.
- Data_10 sont élevés par rapport au taux de reconnaissance de la DataSet précédente.
- Data_20 ont atteint le taux de reconnaissance maximal.

En fin de cette comparaison nous avons constaté que les taux de reconnaissance dépendent du nombres d'itérations. (Voir figure 43)

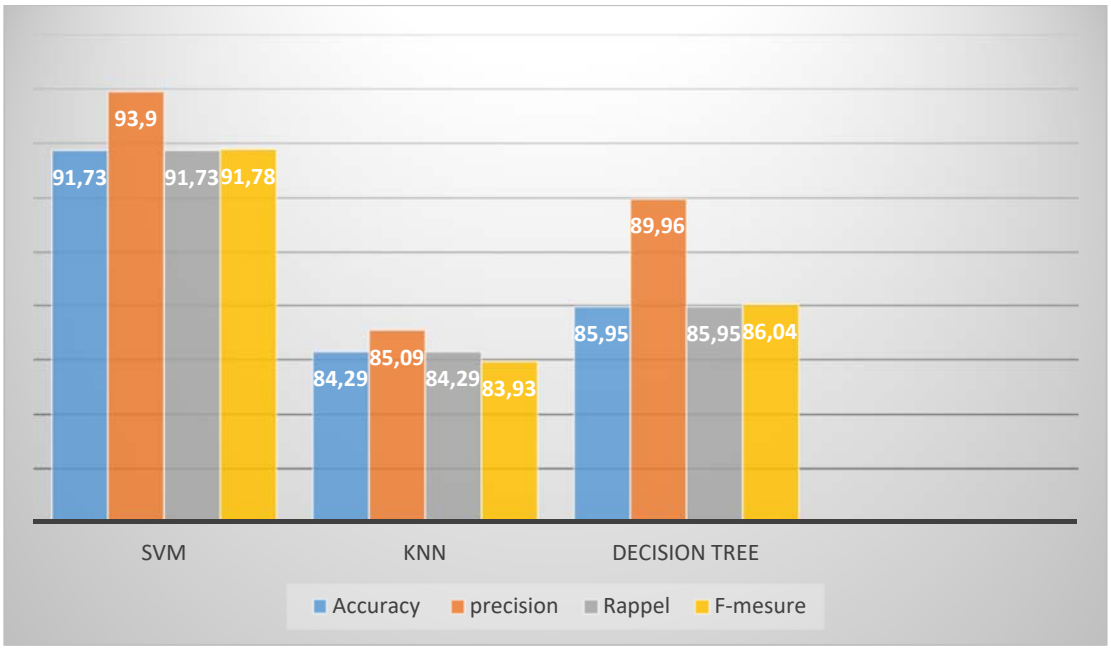

Figure 43 : Histogramme comparatif des méthodes de classification de Data_Histo.

Nous récapitulons l'évaluation établie avec un graphe comparatif du résultat de Précision obtenu en appliquant les méthodes de classification sur chacune des DataSets (voir figure 44).

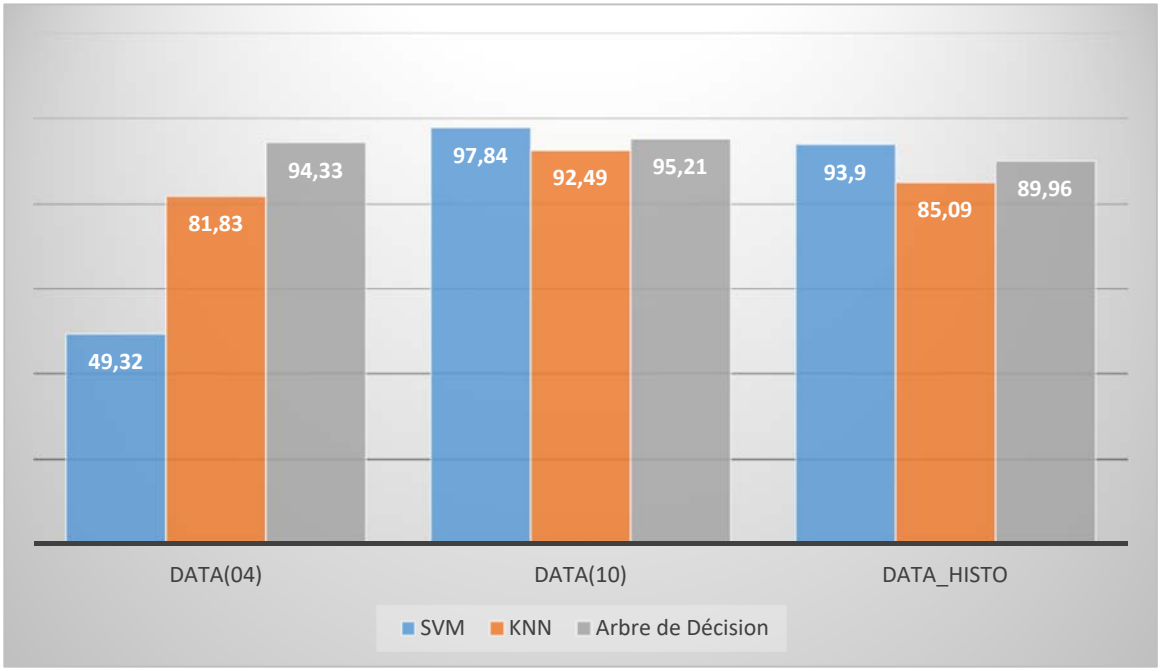

Figure 44 : Graphe comparatif de Precision.

On peut voir que la mesure de Précision peut remplacer les autres mesures et être utilise comme critère de choix pour le meilleur model

Ce qui nous mene à introduire ce graph récapitulatif ou on peut voir que le SVM est plus performant que les deux autres

4.5.5 Analyse des résultats

Après avoir effectué les différents tests sur notre DataSet (Data_10) et comparé les résultats obtenus. Nous allons analyser l'ensemble de ces résultats et les discuter par la suite. Le Tableau récapitulatif (Tableau 4) montre que la méthode de classification SVM apporte de meilleurs résultats. Dans un premier lieu, en analysant la figure 30, nous pouvons remarquer que la méthode SVM est classée en premier avec un Précision supérieure à 97%. A la fin de cette phase on a choisi le modèles SVM compte tenu du taux élevé pour les déployer sur le robot B21R.

4.6 Test et expérience en temps réel sur le robot B21R

Nous avons déployé notre solution en temps real sur le robot et nous avons testé la solution basée sur les méthodes de machine Learning entre SVM, KNN et Arbre de Décision.

Cette étape a été réalisé pour déployer un système de reconnaissance d'activités fonctionnel dans le robot B21R du CDTA. Les tests expérimentaux pour une application en temps réel diffèrent un peu des tests expérimentaux sur les DataSets. Dans notre cas, le robot va acquérir 10 secondes (10 frames) de données de capteur RGB-D pour l'extraction et la classification des caractéristiques. Après les différents tests avec les modèles choisis, on en a conclu que le classifieur SVM était le meilleur en vue de ses résultats en temps réel. Après 10 secondes de classification, une décision finale est prise pour la reconnaissance de l'activité et pour déclencher une réaction appropriée du robot. Le cadre proposé est capable de reconnaître différentes transitions d'activités qui se produisent de manière séquentielle dans le cas où une personne passe d'une activité à l'autre (après 9 secondes de la reconnaissance d'une activité, il répète le même traitement).

4.6.1 Les réactions du robot B21R

Pour faire un scénario d'interaction homme-machine, nous avons définis des réactions pour chaque activité, Les tests en temps réel ont été effectués en utilisant un ordinateur portable avec un processeur intel i5-4210U 2.4GHz et 4 Go de RAM fonctionne sous Ubuntu 16.04 et ROS Kinetic.

Les réactions sont décrites dans le tableau 3 :

Activité	**Réaction**
Arrêter	Le robot demande à la personne un message de confirmation en disant en anglais « do you want me to stop ? », si la personne répond « yes » le robot redit « ok, i'm going to stop » et tourne 90 degrés.
appeler	Le robot demande à la personne un message de confirmation en disant en anglais « do you want me to come ? », si la personne répond « yes » le robot redit « ok, i'm coming » et va vers lui.
Saluer	Le robot dit en anglais « hello, welcome to our center»
Venir	Le robot recule et dit en anglais « how can i help you ? »
partir	Le robot dit en anglais « goodbye see you later » et tourne 180 degrès.
pointer	Le robot demande à la personne un message de confirmation en disant en anglais « do you want me to go there ? », si la personne répond « yes » le robot redit « ok, i'm going there » et va dans la direction où elle pointe.

Tableau 6 : réactions du robot pour chaque activité.

Les figures 45 et 46 représentent notre test en temps réel de la reconnaissance des activités « appeler » et « pointer » accompagnés du code affiché lors de leur exécution et des réactions du robot décrites dans le tableau ci-dessus.

La bonne solution proposée, c'est celle qui est généralisé, autrement dire la reconnaissance d'activité quel que soit la personne. Nous avons testé notre solution proposée sur des données qu'il n'a jamais vues, par testé les résultats sur deux autre personnes (voir figure 47,48).

Figure 45 : Test en temps réel de la reconnaissance de l'activité « Appeler » en utilisant le robot.

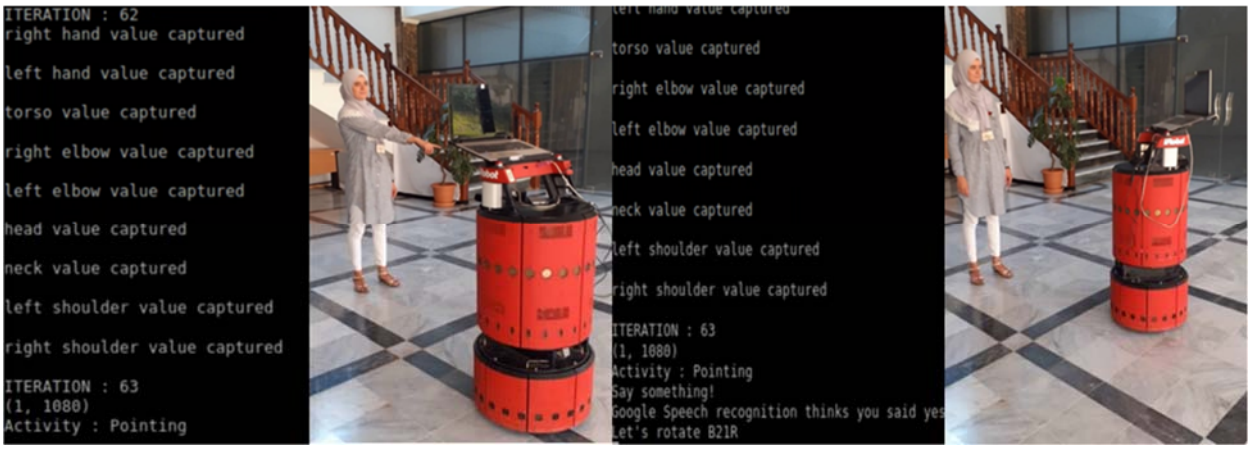

Figure 46 : Test en temps réel de la reconnaissance de l'activité « Pointer » en utilisant le robot .

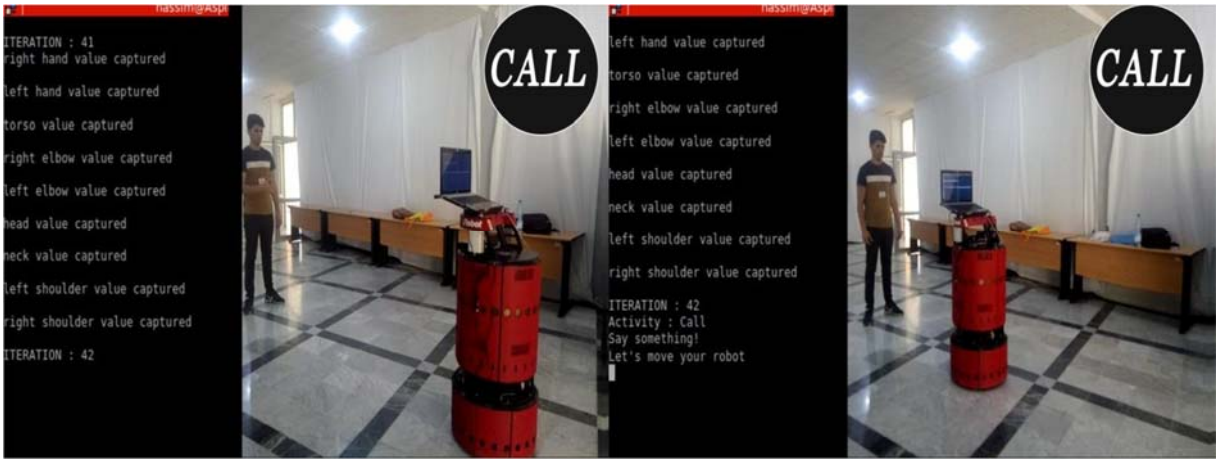

Figure 47 : Test en temps réel de la reconnaissance de l'activité « Appeler » en utilisant le robot.

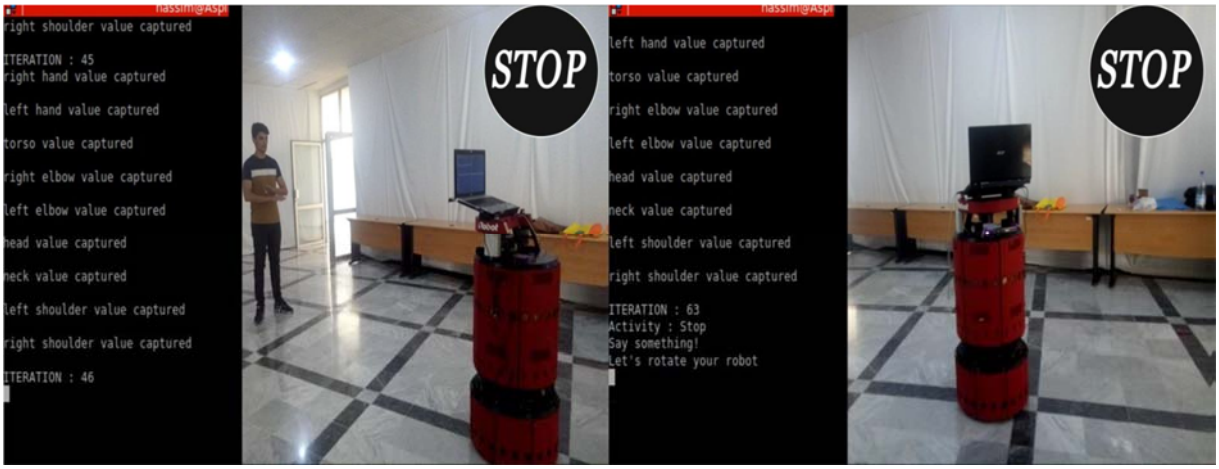

Figure 48 : Test en temps réel de la reconnaissance de l'activité « Arrêter » en utilisant le robot .

4.7 Conclusion

Dans ce dernier chapitre, nous avons présenté la réalisation de notre système de classification des activités humaines. Cette présentation inclut la structure modulaire du système, les outils utilisés pour le développement et l'implémentation de module d'acquisition des données ainsi que le module de génération du modèle de classification. Ensuite, nous avons présenté les résultats des tests d'évaluation des trois méthodes de classification KNN, SVM et Arbre de Décision.

Ce chapitre englobe la démarche suivie, et synthétise l'ensemble des résultats obtenus. Nous avons utilisé des DataSets pour l'évaluation des méthodes de classification choisies. Cette évaluation s'est faite en utilisant les métriques les plus utilisées dans le domaine de Machine Learning.

Nous avons synthétisé l'ensemble des résultats sous forme d'un tableau récapitulatif et d'un histogramme comparatif, à partir desquels nous avons tiré une synthèse. Celle-ci a révélé que le méthode SVM atteint un taux de classification très élevé par rapport aux autres méthodes. L'exécution en temps réel nous a permis d'en conclure que la méthode arbre de décision apporte les meilleurs résultats d'exactitude de reconnaissance des activités.

Conclusion et perspectives

Conclusion et perspectives

Ce mémoire présente le projet que nous avons réalisé ; dans le cadre de notre projet de fin d'études au sein du centre de développement des technologies avancées (CDTA). Celui-ci porte sur l'amélioration des performances de classification en proposant une nouvelle approche basée sur les méthodes d'apprentissage machine (Machine Learning) dans le domaine de la RAH. Ce stage s'inscrit dans le cadre d'un projet global de reconnaissance d'activités humaines basée sur des données issues d'une Kinect.

Nous avons commencé par l'élaboration d'une étude bibliographique sur la reconnaissance d'activités humaines. Nous avons exploré chacune des étapes du processus de près, à commencer par l'acquisition des données depuis le capteur jusqu'à la phase de classification, tout en citant quelques travaux dans le domaine. Ensuite, nous avons exposé l'état de l'art des méthodes d'apprentissage automatique utilisées pour la classification. A travers l'étude bibliographique conduite, nous avons constaté qu'il s'agit d'un domaine de recherche en plein essor, cible de nombreuses contributions scientifiques et industrielles.

Dans une deuxième partie, nous avons présenté notre solution proposée : un système classificateur basé sur l'apprentissage machine. Pour ce faire, nous avons défini l'architecture globale du système à développer qui s'articule autour de trois modules : acquisition des données, classification des données et enfin l'évaluation des résultats de classification. Pour démontrer l'efficacité de notre solution proposée, nous avons travaillé sur des différentes DataSets. Les résultats ont montré que les nouvelles approches sont notamment meilleures que les méthodes de base, et ceci dans la plupart des expérimentations (70% des expériences).

La dernière partie de ce mémoire est consacré à la réalisation de la solution :

Génération et déploiement d'un système de reconnaissance d'activités humaines fonctionnel dans le robot B21R du CDTA. Ce dernier a été conçu dans le but de montrer la faisabilité de l'intégration de notre solution de classification dans un cas réel. En revanche, une classification SVM est effectuée sur les données collectées afin de dégager un meilleur modèle.

L'objectif final de ce stage a été atteint du fait que nous avons réussi à implémenter une solution qui améliore les performances de classification d'une part, et l'intégration de cette dernière dans un système réel de RAH d'une autre part. Par ailleurs, les responsables du stage, pleinement convaincu de la nécessité de cette amélioration, ce sont déclarés satisfaits des résultats obtenus.

D’ailleurs, la solution proposée peut être incluse dans des études ultérieures du centre, ainsi dans des systèmes réels de RAH.

Les perspectives de ce travail concernent les aspects suivants :

• Création des DataSets riches avec la Kinect, contenant un nombre suffisant d’activités.

• Améliorer le taux de reconnaissance d’activités enchainés en temps réel.

• Prédiction des activités futures d’un être humain.

Enfin, cette expérience en laboratoire a été extrêmement gratifiante et enrichissante, tant sur le plan professionnel que personnel. En effet, ce stage a pleinement contribué à renforcer nos connaissances et nos compétences dans le domaine de reconnaissance d’activités humaines, ainsi que celui de Machine Learning. Il nous a également offert une bonne préparation à notre insertion professionnelle. Cette expérience a conforté notre désir d'exercer notre futur métier d'ingénieur en informatique.

Références bibliographies

Références bibliographies

[1] Chen, Q., N.D. Georganas, and E.M. Petriu (2008). “Hand Gesture Recognition Using Haar-Like Features and a Stochastic Context-Free Grammar”.

[2] M. Sung, C. Marci, and A. Pentland. (2005). Wearable feedback systems for rehabilitation. Journal of Neuro- Engineering and Rehabilitation 2, 1

[3] Miguel A Labrador & Oscar D Lara Yejas. (2013). Human activity recognition: Using wearable sensors and smartphones. CRC Press.

[4] Oscar D Lara & Miguel A Labrador. (2013). A survey on human activity recognition using wearable sensors. IEEE Communications Surveys and Tutorials, vol. 15, no. 3, pages 1192–1209.

[5]ABOWD, D, DEY, A K, ORR, R et BROTHERTON, J, (1998). Context-awareness in wearable and ubiquitous computing. In Virtual Reality [en ligne]. 1998. Vol. 3, n° 3, p. 200-211.DOI 10.1007/BF01408562. Disponible à l’adresse http://dx.doi.org/10.1007/BF01408562.

[6] MITCHELL, Tom M, (2006). The Discipline of Machine Learning. In : Machine Learning. (2006). Vol. 17, n° July, p. 1-7. DOI 10.1080/026404199365326. Disponible à l’adresse : http://www-cgi.cs.cmu.edu/~tom/pubs/MachineLearningTR.pdf.

[7] TURAGA, Pavan, CHELLAPPA, Rama, SUBRAHMANIAN, V S et UDREA, Octavian, (2008). Machine Recognition of Human Activities.

[8] CHEN, Liming et KHALIL, Ismail, (2011). Activity Recognition: Approaches, Practices and Trends. In : CHEN, Liming, NUGENT, Chris D, BISWAS, Jit et HOEY, Jesse (éd.), Activity Recognition in Pervasive Intelligent Environments.

[10] Elmezain, M. et al. (2009). “A Hidden Markov Model-Based Isolated and Meaningful Hand Gesture Recognition”.

[11] Bhatt, R., N. Fernandes, and A. Dhage (2013). “Vision Based Hand Gesture Recognition for Human Computer Interaction”.

[12] Agarwal, B., J. Desai, and S. Saha (2012). “A Fast Haar Classifier based Gesture Recognition using camShift algorithm and Curve Fitting Method”.

[13] El-Yacoubi, M. et al. (2014). “Reconnaissance d’activités humaines par un robot humanoïde à partir de séquences vidéos”.

[14] Goodfellow, I., Bengio, Y., Courville, A., & Bengio, Y. (2016). Deep learning (Vol. 1): MIT press Cambridge.

[15] Mitchell, Tom M. (Tom Michael). (1997). Machine Learning. 1st ed. McGraw-Hill.
[16] Mitchell, Tom M. (Tom Michael). (1997). Machine Learning. 1st ed. McGraw-Hill.

[17] Mason, Llew, Jonathan Baxter, Peter Bartlett, and Marcus Frean. (1999). "Boosting Algorithms as Gradient Descent." Nips: 512–18.

[18] S. Russell and P. Norvig, Artificial Intelligence : A Modern Approach. Upper Saddle River, NJ, USA : Prentice Hall Press, 3rd ed., 2009.

[19] K. Hechenbichler and K. Schliep, "Weighted k-nearest-neighbor techniques and ordinal classification," 2004.
[20] T. Hong and D. Tsamis, "Use of knn for the netflix prize," CS229 Projects, 2006.
[21] P. Domingos and M. Pazzani, "On the optimality of the simple bayesian classifier under zero-one loss," Machine learning, vol. 29, no. 2, pp. 103–130, 1997.
[22] I. Rish, "An empirical study of the naive bayes classifier," in IJCAI 2001 workshop on empirical methods in artificial intelligence, vol. 3, pp. 41–46, IBM New York, 2001.
[23] Shannon and Weaver 1949.
[24] B. E. Boser, I. M. Guyon, and V. N. Vapnik, "A training algorithm for optimal margin classifiers," in Proceedings of the fifth annual workshop on Computational learning theory, pp. 144–152, ACM, 1992.
[25] V. N. Vapnik and S. Kotz, Estimation of dependences based on empirical data, vol. 40. Springer-
Verlag New York, 1982.
[26] J. Mercer, "Functions of positive and negative type, and their connection with the theory of integral
equations," Philosophical transactions of the royal society of London. Series A, containing papers of a mathematical or physical character, vol. 209, pp. 415–446, 1909.

[27] Yu, C. et al. (2010). "Connectivity Based Human Body Modeling from Monocular Camera".

[28] Metomo JOSEPH BERTRAND RAPHAËL. (2017).machine-learning-introduction-apprentissage-automatique.

[29] P. Perner et A. Imiya, editeurs. (2005). Machine Learning and Data Mining in Pattern Recognition, 4th International Conference, MLDM'05, volume 3587 de Lecture Notes in Computer Science, Leipzig, Germany. Springer.

[30] S. Krishnamachari et M.A. Mottaleb. (1999). Image browsing using hierarchical clustering. Dans Proceedings ISCC'99.

[31] M^{elle} MAHDJANE Karima. Détection d'anomalies sur des données biologiques par SVM. Université Mouloud Mammeri de Tizi Ouzou, 14 Octobre 2012.

[32] Nicolas La Roux. Avancées théoriques sur la représentation et l'optimisation des réseaux de neurones, Université de Montréal, Mars, 2008.

[33] Cours sur l'intelligence artificiel, Université de Tlemcen : https://www.univtlemcen.dz/~benmammar/IA2.pdf 6041-machine-learning-introduction apprentissage automatique.[34] Moualek.D., Benazzouz. M. (2016). DeepLearning pour la classification des Images.78 :18-24.

[35] Luis .G., Christel .D., Stéphane .L., Jean.-Yves.T.(2016)Utilisation des Random Forests pour la reconnaissance d'activité humaine: une étude complète avec le jeu de données d'Opportunity.

[36] Mokhtar.T. INITIATION A L'APPRENTISSAGE AUTOMATIQUE.Université de Jijel

[37] M. A. Hearst, S. T. Dumais, E. Osman, J. Platt, and B. Scholkopf, "Support vector machines," IEEE Intelligent Systems and their Applications, vol. 13, no. 4, pp. 18 – 28, 1998.

[38] J.T.Y. Kwok. (1998). Automated text categorization using support vector machine. Dans Proceedings of the International Conference on Neural Information Processing.

[39] M. Pontil et A. Verri. (1998). Support vector machines for 3D object recognition. IEEE Trans. Pattern Anal. Mach. Intell., 20(6):637–646.

[40] S. Christian, L. Ivan, et C. Barbara. (2004). Recognizing human actions: A local svm approach. Dans Proceedings of the 17th International Conference on Pattern Recognition, ICPR'04,

[41]https://www.researchgate.net/figure/Figure-3-SVM-classification-scheme-H-is-the-classification-hyperplane-W-is-the-normal_fig3_286268965

[42] https://verhaert.com/difference-machine-learning-deep-learning/

[43] https://www.groupe-hli.com/machine-learning-dans-industrie/

[44] Georgios Drakos (2018). Cross Validation.

[45] Mario Augusto da Costa Vieira (2015). “Recognition of Daily Activities and Risk Situationstowards Robot-Assisted Living”.