

# Projet d'apprentissage automatique

## Reconnaissance des actions humaines à l'aide de capteurs inertIELS

### 1 Présentation du projet

L'objectif de ce projet est de réaliser un système de reconnaissance des actions humaines à l'aide de capteurs inertIELS. La figure I schématise la reconnaissance d'actions avec ces capteurs implantés dans un smartphone. Les capteurs inertIELS embarqués à l'intérieur du smartphone porté par la personne, à savoir, l'accéléromètre, le gyroscope et le magnétomètre permettent de collecter les signaux inertIELS. Ces derniers seront ensuite mis à l'entrée d'un système de reconnaissance d'actions afin d'identifier l'action exécutée en temps réel. Dans l'exemple de la figure I, le système est censé émettre l'étiquette de l'action "la marche", autrement cela sera considéré comme une erreur de classification.

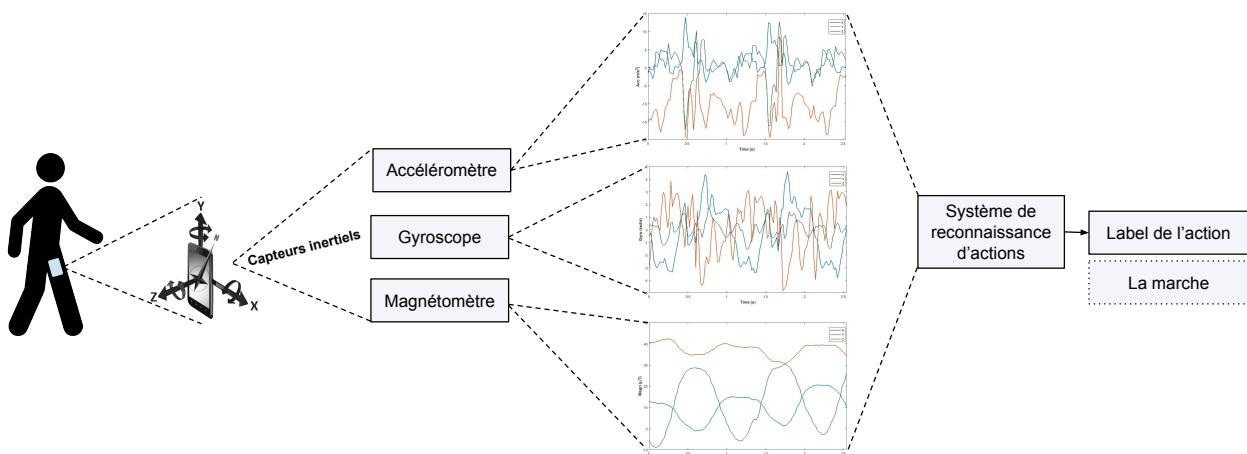


FIGURE I – Reconnaissance de l'action humaine à l'aide d'un smartphone.

### 2 Travail à réaliser

Le travail à réaliser se décompose en plusieurs étapes qui sont caractéristiques d'un projet d'apprentissage automatique à savoir : chargement et compréhension des données, extraction d'informations pertinentes (attributs), préparation des données pour la phase d'apprentissage, entraînement d'un modèle de classification, test du modèle afin de mesurer et analyser

ces performances. Après avoir mené à bien ces différentes étapes, le modèle sauvegardé est prêt à être déployé.

Les étapes sont décrites dans les sections suivantes. Il est conseillé de suivre l'ordre indiqué. Pour chaque étape, des explications et des indications de fonctions à utiliser dans différentes bibliothèques sont données. Votre travail consiste à comprendre les explications, à explorer la documentation des fonctions mentionnées et à écrire le code pour effectuer chaque étape. Pensez à vérifier que votre code produit bien le résultat attendu.

En fin de projet, il faudra fournir vos codes commentés ainsi qu'un rapport clair et concis rédigé à la manière d'un compte-rendu d'étude.

### 3 Environnement de développement

Ce projet sera réalisé en Python, un langage de programmation très populaire en intelligence artificielle et apprentissage automatique. Si vous n'avez pas encore installer Python sur votre machine, vous pouvez installer Anaconda, une distribution libre et open source du langage Python ([rendez vous sur la page d'installation d'Anaconda](#)). À son installation, Anaconda installera Python ainsi qu'une multitude d'IDE et packages (voir [la liste de packages](#)).

Anaconda propose un outil de gestion de packages appelé Conda, ce dernier permettra de mettre à jour et installer facilement les librairies dont on aura besoin dans ce projet. Conda vous permet aussi de créer un nouvel environnement virtuel : répertoire contenant une collection spécifique de packages conda que vous avez installé. Pour cela, allez dans l'onglet “Environments”, il suffit de créer un nouvel environnement virtuel en cliquant sur “Create”, puis d'installer ou de mettre à jour les packages dont vous avez besoin.

Liste des bibliothèques dont vous aurez besoin pour ce projet : numpy, pandas, matplotlib, SciPy, Scikit-Learn.

### 4 Base de données

La base de données utilisée dans ce projet (voir la figure II) contient 27 différentes actions : (1) balayage du bras droit vers la gauche, (2) balayage du bras droit vers la droite, (3) faire signe de la main droite, (4) applaudissement, (5) lancer du bras droit, (6) croix bras dans la poitrine, (7) tir de basket-ball, (8) dessiner un “x”, (9) dessiner un cercle de la main droite (sens horaire), (10) dessiner un cercle de la main droite(sens antihoraire), (11) dessiner un triangle, (12 ) bowling (main droite), (13) boxe, (14) swing Baseball , (15) tennis swing, (16)

flexion des bras (deux bras), (17) service de tennis, (18) pousser, (19) frapper à la porte, (20) attraper un objet, (21) ramasser et lancer, (22) courir sur place, (23) marcher, (24) se lever, (25) s'asseoir, (26) pied gauche en avant, (27) Squat. 8 sujets ont participé à cette expérimentation dont 4 hommes et 4 femmes. Chaque sujet a répété chaque action 4 fois. Après avoir supprimé trois séquences corrompues, la base de données comprend en total 861 séquences. Les données ont été stockées en utilisant l'environnement de développement MATLAB.

## 5 Protocole de collecte de données

Afin de collecter les données, des capteurs portables type microsystèmes électromécaniques (Microelectromechanical systems - MEMS) ont été utilisés, plus spécifiquement des MEMS inertIELS comme il est présenté dans la figure III1. Ce dispositif contient deux capteurs de mouvements : l'accéléromètre qui permet de calculer les accélérations statiques et dynamiques d'un dispositif, le gyroscope qui calcule la vitesse angulaire. Les signaux sont collectés suivant trois axes perpendiculaires ( $x, y, z$ ). La fréquence d'échantillonage pour les deux capteurs est de  $50Hz$ .

Dans la base de données, le MEMES inertiel était porté sur le poignet droit du sujet pour les actions qui sollicitent plus les bras (actions 1 à 21), ou sur la cuisse droite pour les actions qui sollicitent les jambes (actions 22 à 27).

## 6 Chargement des données

Téléchargez la base de données au format zip [IMU.zip](#) dans le dossier [Projet](#) sur MyLearningSpace. Vous trouverez dans ce dossier des fichiers avec l'extension “mat” (pour MATLAB). Les fichiers sont nommés suivant le format “ai\_sj\_tk\_inertial.mat”, où “ai” représente l'action numéro i, “sj” représente le sujet numéro j et “tk” signifie l'essai numéro k.

→ Après avoir téléchargé la base de données, écrivez une fonction “load\_data” qui permet de charger les données sur python (vous pouvez utiliser la fonction [loadmat](#) de la bibliothèque [Scipy](#)), la fonction doit renvoyer un “dataframe” organisé comme suite :

- Colonne 0-2 : contiennent les données de l'accéléromètre suivant les trois axes.
- Colonne 3-5 : contiennent les données du gyroscope suivant les trois axes.
- Colonne 6 : contient l'identifiant du sujet (1 à 8).
- Colonne 7 : contient l'identifiant de l'essai (1 à 4).

- Colonne 8 : contient l'identifiant (l'étiquette) de l'action (1 à 27).

Le résultat doit ressembler à la figure IV. La méthode Python **listdir(path)** du module os



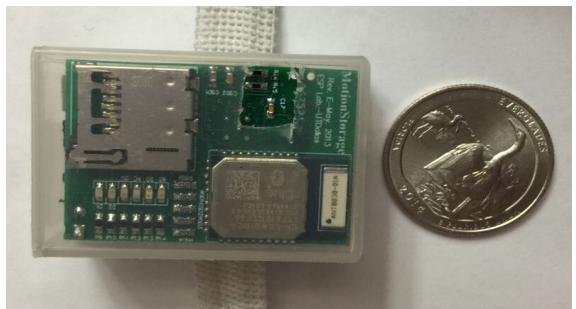
FIGURE II – Actions humaines contenues dans la base de données.

renvoie une liste contenant les noms des fichiers du répertoire donné par path.

→ Écrivez une fonction “tracer\_signal” qui permet de tracer les trois signaux ( $x, y, z$ ) d'un capteur correspondant à une action. Cette fonction prend 5 entrées :

- Le dataframe (envoyer par la fonction “load\_data”).
- Le capteur (1 : accéléromètre, 2 : gyroscope).
- Le numéro de l'action.
- Le numéro du sujet.
- Le numéro de l'essai.

Vous aurez besoin de la bibliothèque “matplotlib” pour tracer un signal. [Cliquez ici](#) pour



(1) MEMES inertiel.



(2) Position des capteurs

FIGURE III – Capteurs

	0	1	2	3	4	5	subject	experience	action
0	-0.95166	-0.41626	-0.09106	-18.19847	-8.67176	8.61069	1	1	10
1	-0.94849	-0.42554	-0.11865	-18.71756	-7.84733	6.44275	1	1	10
2	-0.94043	-0.42602	-0.14087	-21.40458	-5.67939	2.22901	1	1	10
3	-0.94482	-0.50757	-0.19141	-23.38931	-2.22901	-2.83969	1	1	10
4	-0.95459	-0.63696	-0.30737	-25.12977	2.13741	-6.16794	1	1	10
5	-0.94824	-0.70337	-0.37012	-31.81679	8.42748	-11.69466	1	1	10
6	-0.96387	-0.73828	-0.36133	-37.61832	16.82443	-23.11450	1	1	10
7	-0.96045	-0.79248	-0.38013	-33.49618	26.38168	-40.88550	1	1	10
8	-1.01392	-0.89014	-0.43823	-26.65649	35.14504	-61.92366	1	1	10
9	-1.01099	-0.97095	-0.46924	-29.67939	46.19847	-78.99237	1	1	10
10	-1.00513	-0.98682	-0.45288	-47.29771	58.07634	-96.42748	1	1	10
11	-1.02466	-0.94165	-0.43384	-60.27481	68.45802	-113.03817	1	1	10
12	-1.02978	-0.90478	-0.42822	-60.61069	76.67176	-127.93893	1	1	10
13	-1.03980	-0.92993	-0.44141	-58.59542	83.57252	-141.58779	1	1	10
14	-1.03662	-0.93164	-0.37598	-58.87023	89.92366	-151.05344	1	1	10
15	-1.02734	-0.93457	-0.33789	-61.31298	94.22901	-160.21374	1	1	10
16	-0.98340	-0.88989	-0.32324	-66.32061	96.70229	-169.34351	1	1	10
17	-0.92114	-0.82446	-0.31274	-67.87786	97.19084	-173.98473	1	1	10
18	-0.85425	-0.75073	-0.29565	-66.50382	96.15267	-173.70992	1	1	10
19	-0.77661	-0.67651	-0.25537	-60.45802	93.40458	-170.19847	1	1	10

FIGURE IV – Exemple des données chargées.

un exemple.

## 7 Extraction des attributs

Pour décrire chaque action humaine, une liste d'attributs est extraite des signaux. Dans ce projet, nous allons opter pour l'extraction des attributs statistiques. Voici une liste d'attributs qui peuvent être extraits : moyenne, moyenne quadratique, écart-type, médiane, quartiles, coefficient d'asymétrie, Kurtosis, entropie, énergie, maxima et minima, coefficient de corrélation, histogramme, passages par zéro, nombre d'occurrences de pics.

→ Écrivez une fonction "feature\_extraction" qui prend en entrée le "dataframe" crée et calcule pour chaque action un vecteur d'attributs. Remarque : si vous choisissez par exemple la moyenne et l'écart-type comme attributs, ces derniers doivent être calculés pour chaque signal (colonnes 0-5 dans la figure IV). Le vecteur d'attributs pour chaque action sera donc de dimension 12 (6 signaux  $\times$  2 attributs). Il est représenté comme suit :

$$V = [mean(A_x), std(A_x), mean(A_y), std(A_y), mean(A_z), std(A_z), mean(G_x), std(G_x), mean(G_y), std(G_y), mean(G_z), std(G_z)]$$

NB : La fonction **dataframe.groupby()** permet de diviser les données du dataframe en groupes en fonction de certains critères.

## 8 Préparation des données

Afin de préparer les données (attributs calculés) pour l'étape de classification, ces dernières doivent être divisées en deux groupes : données d'apprentissage, données de test. Dans ce projet, les données d'apprentissage doivent contenir les attributs calculés relatifs aux sujets [1, 3, 5, 7], et les données de test aux sujets [2, 4, 6, 8].

Une étape importante (mais pas nécessaire, elle dépend des attributs calculés) est la normalisation des données. Pour appliquer cette dernière, calculez le vecteur des moyennes et des l'écart-types sur les attributs d'apprentissage (la dimensions des vecteurs doit être égale aux nombre d'attributs calculés). Soustrayez ensuite le vecteur moyenne des données d'apprentissage et de test et divisez sur l'écart-type.

→ Créez une fonction qui permet de réaliser cette tâche. La fonction doit renvoyer quatre éléments : les données d'apprentissage, les étiquettes d'apprentissage, les données de test, les étiquettes de test.

## 9 Entrainement d'un modèle de classification

Afin d'entamer la classification des actions humaines, nous allons d'abord entraîner un modèle sur les données d'apprentissage. Pour cela, nous allons utiliser "Scikit-learn", une bibliothèque libre de Python destinée à l'apprentissage automatique.

- Installez "Scikit-learn" en utilisant la commande : `pip install scikit-learn`
- Appliquez quelques classifieurs (les plus connus : réseau de neurones, machine à vecteurs de support, k-plus proches voisins, arbres de décision, Naïve Bayes), pour cela consulter la page [scikit-learn](#).

Exemple d'application du classifieur arbre de décision :

```

1 from sklearn import tree           #Import descison tree
2 data = [[0, 0], [1, 1]]            #Training data (feature extracted)
3 labels = [0, 1]                   #labels (in our project action 1-27)
4 clf = tree.DecisionTreeClassifier() #Initialize our classifier
5 clf = clf.fit(data, labels)       #Train our classifier

```

- Affichez l'arbre de décision obtenu avec le classifieur d'arbre de décision (on pourra utiliser la fonction `plot_tree` du module `sklearn.tree`).
- Affichez l'arbre de décision obtenu avec le classifieur d'arbre de décision sous forme de règles imbriquées (on pourra utiliser la fonction `export_text`).

## 10 Test du modèle

Pour tester notre modèle, nous allons utiliser les données de test (qui sont indépendantes des données utilisées dans la phase d'apprentissage).

Exemple d'application pour l'arbre de décision :

```
1 clf.predict([[2., 2.]])  #Predict action label from feature vector [2, 2]
```

- Écrivez le code qui permet d'utiliser les données de test à l'entrée du classifieur (ou des classifieurs si vous avez entraîné plusieurs). Le résultat sera stocké sous forme d'un vecteur contenant les étiquettes des actions.

## 11 Analyse des données

Afin d'estimer les performances de reconnaissances globales de notre modèle, des mesures peuvent être calculées, les plus populaires sont : *précision*, *rappel*, *F-score* et *exactitude* (*accuracy* en anglais).

→ Écrivez une fonction qui permet de calculer ces mesures, vous pouvez utiliser “classification\_report” disponible sur “sklearn” (voir [classification report](#)). Commentez les résultats obtenus.

Les mesures de qualité du système citées ci-dessus délivrent un score global qui permet de savoir rapidement si notre système est performant ou non. Afin d'avoir plus de détails sur la qualité du système (score obtenu pour chaque action, les actions les plus confondues, etc.), la matrice de confusion est calculée.

→ Déterminez la matrice de confusion en utilisant “confusion\_matrix” disponible sur “sklearn” (voir [confusion matrix](#)) Commentez les résultats obtenus.