

École Centrale de Nantes - Avisia

Option INFO-IA

Rapport de stage (STING)

Application de l'estimation de position humaine pour analyser le mouvement de la course d'un sportif



Atys PANIER

Encadrant de stage : Christophe SAWADOGO

Version du 01/09/2023

Table des matières

1 Remerciements	3
2 Contexte	3
3 Introduction	3
4 Présentation détaillée de l'entreprise	5
5 État de l'art de l'estimation de pose humaine	7
5.1 Fondements théoriques	7
5.1.1 Définition de la pose estimation	7
5.1.2 Présentation des différents types de pose estimation (2D, 3D, multipersonnes)	7
5.1.3 Présentation des différents domaines d'application de la pose estimation	7
5.2 Les différentes approches de la “pose estimation”	8
5.2.1 Comprendre la classification classique d'images	8
5.2.2 Exemple de modèle classique de Human Pose Estimation.	10
5.2.3 Comprendre les réseaux de neurones convolutifs	10
5.2.4 Comment fonctionne un RCCN ?	11
5.2.5 Comparaison des avantages et des inconvénients de ces différentes approches	12
5.3 Les techniques d'annotation de données	12
5.3.1 Présentation des différents types de données utilisés pour entraîner les modèles de pose estimation	12
5.3.2 Présentation des techniques d'annotation manuelle et semi-automatique de ces données	12
5.4 Bases de données	13
5.4.1 Analyse des caractéristiques de ces bases de données	13
5.4.2 Présentation des métriques utilisées pour évaluer ces performances	13
5.4.3 Analyse des performances des modèles de pose estimation récents	14
6 Implémentation d'un algorithme de classification d'image	15
6.1 Architecture du réseau de neurones convolutif VGG16	15
6.2 Combinaison du VGG16 et de notre classifier	15
6.3 Résultats	16
7 Les spécificités de l'estimation de pose appliquée au sport	17
7.1 Présentation et enjeux	17
7.2 Les 3 algorithmes de human pose estimation pour sport	17
7.3 Présentation d'une API qui propose des modèles de HPE	17
8 Phase 1 : Mise en place de l'environnement	18
9 Phase 2 : Présentation générale de l'utilisation des algorithmes de HPE	19
9.1 Un premier exemple de Human Pose Estimation sur un joueur de rugby	19
9.1.1 Présentation de chaque algorithme (inferencer) disponible	21
10 Phase 3 : Utilisation de Google Cloud Platform	21
11 Phase 4 : Inférence sur des vidéos de sportifs avec extraction des données de position des points clés du corps humain pour l'analyse des mouvements	22
12 Phase 5 : Présentation des résultats	22
13 Phase 6 : Vers une application web	26
14 Le projet LOL	28

14.1 Drafting	28
14.2 Scouting	29
15 Conclusion	36
Annexes	38
A Documentation et liens	38

1 Remerciements

Je voudrais commencer par remercier Avisia, sa direction et ses employés. Je tiens en particulier à exprimer ma gratitude à Xavier Gamarre, fondateur et PDG d'Avisia, pour avoir développé une si bonne culture de travail. Je suis également particulièrement reconnaissant à Christophe Sawadogo pour m'avoir donné l'opportunité de travailler dans un endroit aussi formidable et à Elie N'kaoua, pour m'avoir présenté Avisia et m'avoir encouragé à postuler pour ce stage.

Ensuite, je tiens à souligner l'aide apportée par Christophe Sawadogo, mon maître de stage, qui m'a fourni des conseils techniques et professionnels inestimables. Au cours de nos nombreuses réunions hebdomadaires, il a énormément contribué à mon projet, en me montrant de nombreuses méthodes statistiques possibles pour chaque problème auquel j'étais confronté, ainsi que des conseils professionnels et de carrière, en m'orientant sur les meilleures façons de gérer les tâches quotidiennes.

Je suis aussi particulièrement reconnaissant à Julien Legrave pour l'aide qu'il m'a apportée dans le cadre de mon stage, notamment en m'aidant à développer mes compétences techniques.

Enfin, j'aimerais exprimer ma gratitude à tous les professeurs de l'option Informatique et IA et en particulier à Didier Lime, qui a toujours fait preuve d'un soutien essentiel à tous les étudiants et d'un engagement en faveur de la qualité globale du programme.

2 Contexte

Dans le cadre de ma deuxième année dans l'option Informatique et Intelligence Artificielle à l'École Centrale de Nantes, je suis actuellement en stage chez Avisia, une société de conseil basée à Paris. Ce stage m'a permis de travailler sur un projet interne. L'objectif de ce projet était de comprendre et de mettre en œuvre des algorithmes d'estimation de pose humaine pour l'analyse des mouvements sportifs. Le projet s'est déroulé en trois parties distinctes. Dans la première partie, j'ai effectué une revue de l'état de l'art sur l'estimation de pose humaine, également connue sous le nom de "human pose estimation" (HPE). Ensuite, dans la deuxième partie, j'ai eu l'opportunité de mettre en place un algorithme d'estimation de pose humaine dans un notebook, permettant d'appliquer cette méthode à une vidéo. Enfin, la troisième partie consistait à créer une application web dédiée. Bien que la majeure partie de mon temps de travail ait été consacrée à ce projet, Avisia m'a également offert la possibilité de me former à diverses compétences. Certaines formations étaient directement liées au projet, tandis que d'autres étaient axées sur la science des données et les statistiques en général. J'ai ainsi suivi plusieurs formations sur Dataiku, qui m'ont permis d'obtenir deux certifications : le "Core Designer Certificate" et l'"Advanced Designer Certificate". En plus de ces cours, j'ai aussi participé à des ateliers en entreprise portant sur divers sujets, tels que la sécurité des données. J'ai également eu l'opportunité de participer à un hackathon sur la visualisation de données avec PowerBI.

3 Introduction

Ce rapport présente le travail accompli lors d'un stage de 5 mois chez Avisia, une entreprise de conseil française spécialisée dans les sciences des données et l'analyse, comptant 4 bureaux en France et environ 180 consultants. Pendant ce stage, j'ai travaillé quatre jours par semaine en présentiel chez Avisia et un jour par semaine en télétravail. Chaque semaine, j'ai eu en moyenne une à deux réunions avec mon maître de stage pour discuter de l'avancement du projet. De plus, j'ai eu des échanges hebdomadaires avec Julien pour répondre aux besoins spécifiques des clients. En effet, ce projet est né de l'opportunité pour Avisia de collaborer avec le Stade Rochelais, un club de rugby actuellement champion d'Europe, afin d'analyser les données de leurs joueurs. Dans le cadre de cette collaboration, l'une des pistes d'analyse de données porte sur l'estimation de la pose humaine, permettant ainsi de caractériser et de mieux comprendre les mouvements des joueurs selon leur poste. Au sein d'Avisia, j'ai donc travaillé sur la mise en place de cette analyse de la position humaine, en commençant par un cadre restreint afin de tester certaines hypothèses, présenter des résultats simples et m'assurer que ces algorithmes produisent des réponses pertinentes pour notre client.

Le projet HPE (Human Pose Estimation) s'est déroulé en suivant les étapes suivantes :

- **État de l'art de l'estimation de pose humaine** : Au début du projet, une revue de l'état de l'art dans le domaine de l'estimation de pose humaine a été réalisée. Cela a permis de comprendre les techniques et les méthodes existantes pour estimer la pose d'un individu à partir d'images ou de vidéos.
- **Implémentation d'un algorithme de classification d'image** : Un algorithme de classification d'image a été développé pour identifier les différentes couches d'un réseau de neurones convolutif. Cette étape a impliqué la collecte et le prétraitement d'un ensemble de données approprié, ainsi que l'entraînement d'un modèle de classification.
- **Comprendre les spécificités de l'estimation de pose appliquée au sport** : Une attention particulière a été portée à l'application de l'estimation de pose humaine dans le contexte sportif. Les caractéristiques et les défis liés à l'estimation de pose dans des activités sportives ont été étudiés afin d'adapter les techniques et les modèles utilisés.
- **Compréhension de l'utilisation de l'API MMPose** : L'API MMPose, une bibliothèque populaire pour l'estimation de pose humaine, a été explorée et comprise. Cela a permis d'utiliser les fonctionnalités et les modèles de l'API dans le projet.
- **Mise en place d'un environnement d'inférence** : Un environnement d'inférence a été configuré sur une machine virtuelle (VM) sur Google Cloud Platform (GCP). Cela a permis d'exécuter les modèles d'estimation de pose sur des vidéos de sportifs pour obtenir les données de position des points clés du corps humain.
- **Inférence sur des vidéos de sportifs** : Les modèles d'estimation de pose ont été utilisés pour effectuer des inférences sur des vidéos de sportifs. Les coordonnées des points clés du corps humain (par exemple, les mains, les pieds, les coudes) ont été extraites à partir de ces vidéos.
- **Extraction des données de position des points clés du corps humain pour l'analyse des mouvements** : Les données de position des points clés extraites ont été utilisées pour l'analyse des mouvements. Différentes mesures et métriques ont été appliquées pour comprendre et évaluer les performances et les caractéristiques des mouvements des sportifs.
- **Comparaison et analyse des mouvements** : Les données de position des points clés ont été comparées et analysées pour identifier des schémas de mouvement, des erreurs ou des anomalies. Cela a permis de mieux comprendre les performances des sportifs et de fournir des informations utiles pour les entraîneurs et les analystes.
- **Création d'une application web généralisant ce travail** : Enfin, une application web a été développée pour généraliser le travail réalisé. Cette application permet aux utilisateurs d'importer des vidéos de sportifs, d'obtenir des estimations de pose et des analyses de mouvements, et de visualiser les résultats de manière conviviale.

L'ensemble du travail réalisé est disponible sur un dépôt Github privé et open-source (Apache V2) : <https://github.com/atysp/HPE>.

En parallèle de mon travail sur le projet HPE, au cours du dernier mois de mon stage, j'ai également eu l'opportunité de participer au lancement d'un deuxième projet passionnant. Ce projet, baptisé projet LOL, était axé sur l'esport et visait à établir une collaboration avec une équipe spécialisée dans le jeu League of Legends. L'objectif principal de cette collaboration était d'explorer les opportunités liées à l'analyse des données dans le domaine de l'esport. Cette expérience m'a permis d'acquérir de nouvelles compétences et de contribuer à un projet prometteur dans un domaine en pleine croissance.

Mon travail s'est orienté autour de 2 points :

- **Drafting** : L'objectif était de développer une assistance pour la phase de construction des équipes. Il s'agit de fournir des outils permettant d'obtenir la meilleure composition d'équipe possible, en prenant en compte celle de l'adversaire, afin d'optimiser les chances de victoire.
- **Scouting** : Le but était de fournir des indicateurs de performance qui puissent servir de révélateurs du niveau de chaque joueur professionnel, facilitant ainsi le processus de recrutement.

4 Présentation détaillée de l'entreprise

Avisia est une société de conseil française basée au cœur de Paris, juste à côté de l'Arc de Triomphe. Fondée en 2007 dans une philosophie "Data Centric", elle dispose aujourd'hui de bureaux dans toute la France, à Paris, Lyon, Nantes et Bordeaux, et emploie plus de 180 consultants de 14 nationalités différentes. Aujourd'hui, la structure de l'entreprise est basée sur trois axes : Data, Digital et Technology.

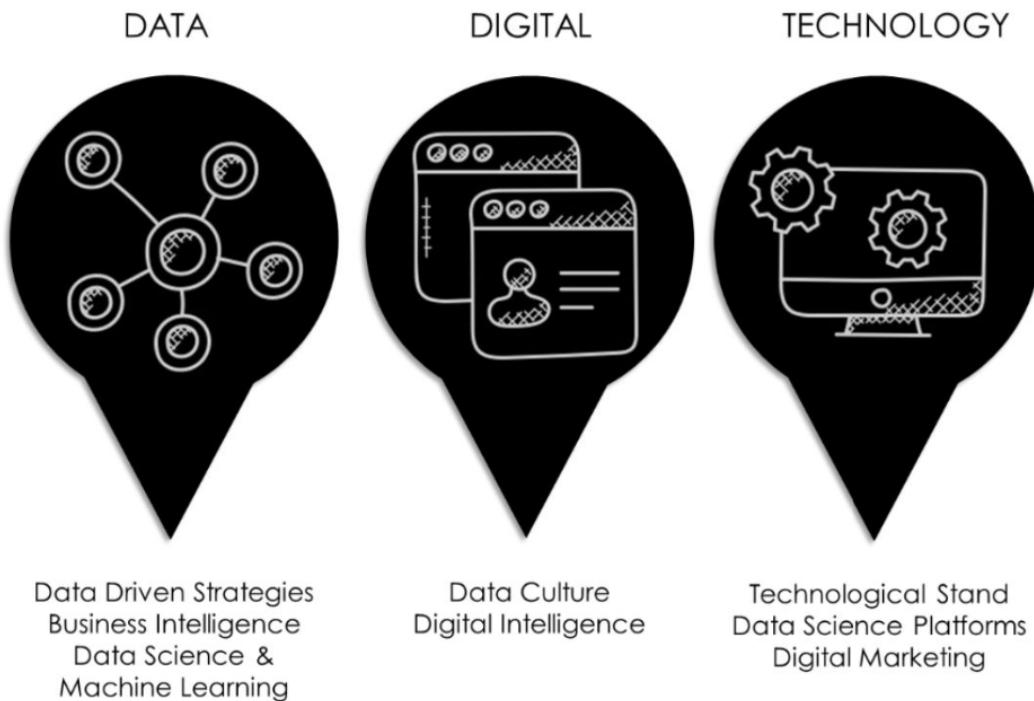


FIGURE 1 – Les 3 axes de travail chez Avisia

J'ai été affecté à la division Data, mais j'ai souvent interagi avec la division Digital. Entre autres tâches, la division Data utilise des algorithmes de science des données pour répondre aux besoins de ses clients, en particulier dans les domaines de la finance et du risque, mais aussi dans le domaine du marketing. En ce qui concerne plus particulièrement le marketing, par exemple, elle offre des services tout au long du processus de traitement des données, tels que la collecte et la réconciliation des données, l'intelligence et la modélisation des données, ainsi que l'activation des campagnes de marketing.

En 2016, Avisia a étendu son expertise avec la création d'un Digital hub, qui se concentre sur l'analyse des données spécifiques au web et à la navigation des utilisateurs, comme la mise en œuvre des plans de tracking afin de collecter les données de navigation, l'analyse des chemins de navigation des utilisateurs, et enfin comment présenter ces données d'une manière conviviale au client. Ces processus sont réalisés à l'aide de nombreuses plates-formes différentes et d'un grand nombre d'outils.



FIGURE 2 – Logiciels principalement utilisés durant mon stage

Plusieurs logiciels, notamment Google Cloud Platform (BigQuery, ComputeEngine, DataStore, etc.),

Google Marketing Platform (Google Analytics, Data Studio, Google Tag Manager, Optimize, etc.) et Adobe Analytics Cloud (Adobe Analytics, Adobe Campaigns, Audience Manager, etc.) sont présents dans le travail des consultants. De nos jours, il y a toujours beaucoup de données provenant de l'utilisation du web, de sorte que cette intégration entre les départements de données et numériques devient de plus en plus nécessaire.

Avisia offre également de nombreuses possibilités de développement professionnel et d'apprentissage, telles que des cours et des certifications. Personnellement, j'en ai profité pour suivre des cours, ateliers et passer des certifications pendant mon stage, tous liés au sujet sur lequel je travaillais. Je pense que c'est une bonne façon de progresser dans une matière.

Chez Avisia, les principaux intervenants dans mon projet étaient moi-même, Christophe Sawadogo et Julien Legavre. Mon rôle, en tant que stagiaire, était de comprendre le projet dans son ensemble, de réfléchir aux approches et techniques à adopter à chaque étape et d'écrire le code approprié pour implémenter chacune d'entre elles. Christophe S. était mon superviseur de stage, responsable de garantir que le projet était aligné sur les directives, que mon travail était pertinent dans le cadre de mon programme de Master, et que les méthodes algorithmiques que j'utilisais étaient appropriées. Il m'a également guidé dans le choix des meilleures approches pour chaque tâche. Julien Legavre, quant à lui, est un consultant en data science chargé de superviser l'ensemble des projets internes liés au sport. Son rôle était de nous fournir des retours sur l'avancement du projet tout en adoptant une perspective orientée client, afin de pouvoir ensuite promouvoir le projet auprès d'éventuels clients.



FIGURE 3 – Photographie mettant en valeur l'open space d'Avisia

5 État de l'art de l'estimation de pose humaine

Les premières semaines de mon stage m'ont permis de faire des recherches autour de l'estimation de position. J'ai pu passer en revue les différentes approches et techniques d'estimation de position d'objets et d'humains.

5.1 Fondements théoriques

5.1.1 Définition de la pose estimation

La vision par ordinateur est un domaine scientifique et une branche de l'intelligence artificielle qui traite de la façon dont les ordinateurs peuvent acquérir une compréhension de haut niveau à partir d'images ou de vidéos numériques. Du point de vue de l'ingénierie, il cherche à comprendre et à automatiser les tâches que le système visuel humain peut effectuer.

L'estimation de position est une technique de vision par ordinateur qui permet de déterminer la position et l'orientation d'un objet ou d'un corps dans l'espace en utilisant des données provenant de capteurs ou de caméras.

5.1.2 Présentation des différents types de pose estimation (2D, 3D, multipersonnes)

Il existe plusieurs types d'estimation de position, chacun avec ses propres avantages et applications.

La “pose estimation” 2D se concentre sur la détection des points clés du corps humain sur une image en deux dimensions. Il s'agit essentiellement d'un moyen de capturer un ensemble de coordonnées pour chaque articulation (bras, tête, torse, etc.) pouvant décrire la pose d'une personne. La connexion significative entre deux points forme une paire. L'objectif de la “Human Pose Estimation” (HPE) est de former une représentation du corps humain sous la forme d'un squelette, puis de la traiter pour des applications spécifiques. La pose estimation multipersonnes implique la détection et la localisation simultanées des points clés du corps de plusieurs personnes dans une scène. Enfin, l'estimation de position d'objet est utilisée pour estimer la pose d'objets autres que les corps humains

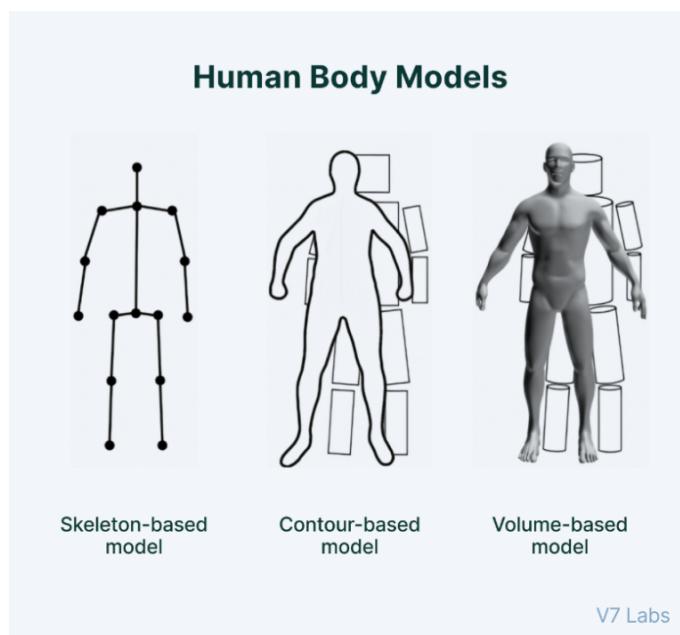


FIGURE 4 – Les différents modèles pour l'estimation de position humain

5.1.3 Présentation des différents domaines d'application de la pose estimation

Elle trouve de nombreuses applications dans des domaines tels que la réalité virtuelle, l'animation de personnages, la reconnaissance de gestes, la sécurité, la robotique, la modélisation 3D et la santé. Dans le

domaine de la réalité virtuelle, l'estimation de pose est utilisée pour permettre aux utilisateurs d'interagir avec des environnements virtuels de manière naturelle et intuitive. Dans l'animation de personnages, elle permet de créer des mouvements fluides et réalistes pour les personnages animés. La reconnaissance de gestes en est une autre application importante, utilisée dans les interfaces homme-machine, les jeux vidéo et la surveillance de la sécurité. La robotique utilise également la "pose estimation" pour permettre aux robots de se déplacer et d'interagir avec leur environnement de manière autonome. Enfin, elle est utilisée dans le domaine de la santé pour l'analyse de mouvements et la réadaptation physique.

5.2 Les différentes approches de la "pose estimation"

Présentation des approches basées sur les caractéristiques descriptives (features-based)

Les approches basées sur les caractéristiques descriptives, ou features-based en anglais, sont des méthodes de vision par ordinateur qui visent à extraire des caractéristiques significatives d'une image ou d'une séquence vidéo pour identifier des objets, des personnes ou des actions. Ces caractéristiques peuvent inclure des points clés, des contours, des textures ou des couleurs. Les algorithmes de features-based peuvent utiliser des techniques de détection de points clés, comme les coins de Harris ou les blobs de Laplacian, pour extraire les caractéristiques de l'image. Ces caractéristiques sont ensuite utilisées pour estimer la pose, la position ou le mouvement de l'objet ou de la personne. Les approches features-based ont l'avantage d'être robustes aux variations d'éclairage et de perspective, et sont utilisées dans de nombreuses applications telles que la reconnaissance faciale, la surveillance de la sécurité, la navigation autonome, la réalité augmentée et la modélisation 3D. Cependant, ces approches peuvent être limitées dans des environnements complexes ou en mouvement rapide, où les caractéristiques peuvent être difficilement détectées ou suivies.

5.2.1 Comprendre la classification classique d'images

Les filtres sont souvent utilisés pour retrouver des motifs particuliers dans une image. Ces motifs sont représentés par de petites images, appelées templates. La tâche de template matching a pour but de retrouver des templates dans une image. Le template matching réalisé avec des filtres utilise l'opérateur de corrélation croisée (cross-correlation), noté (\otimes).

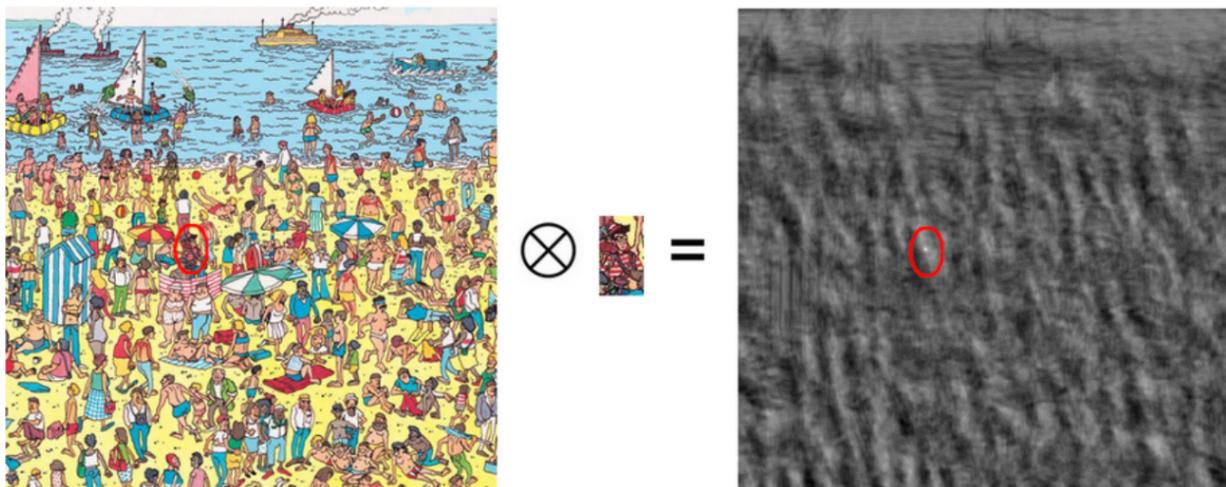


FIGURE 5 – Exemple d'utilisation de l'opérateur de corrélation croisée

H est une petite image représentant le template à retrouver. Concrètement, cette opération revient à faire glisser H sur l'image X , à multiplier les pixels qui se superposent et à sommer ces produits. Ainsi, le template matching consiste à calculer la corrélation croisée entre une image X et un filtre dont le noyau H représente un template que l'on souhaite retrouver dans X .

En vision par ordinateur, le terme de features désigne des zones intéressantes de l'image numérique. Ces zones peuvent correspondre à des contours, des points ou des régions d'intérêt. À chaque feature détectée est associé un vecteur, appelé descripteur (feature descriptor ou feature vector), qui, comme son nom l'indique,

décrit la zone concernée. La résolution du problème d'image matching se fait alors en deux étapes : Déetecter et décrire les features dans chaque image Trouver les paires de features qui se correspondent dans les deux images (features matching)

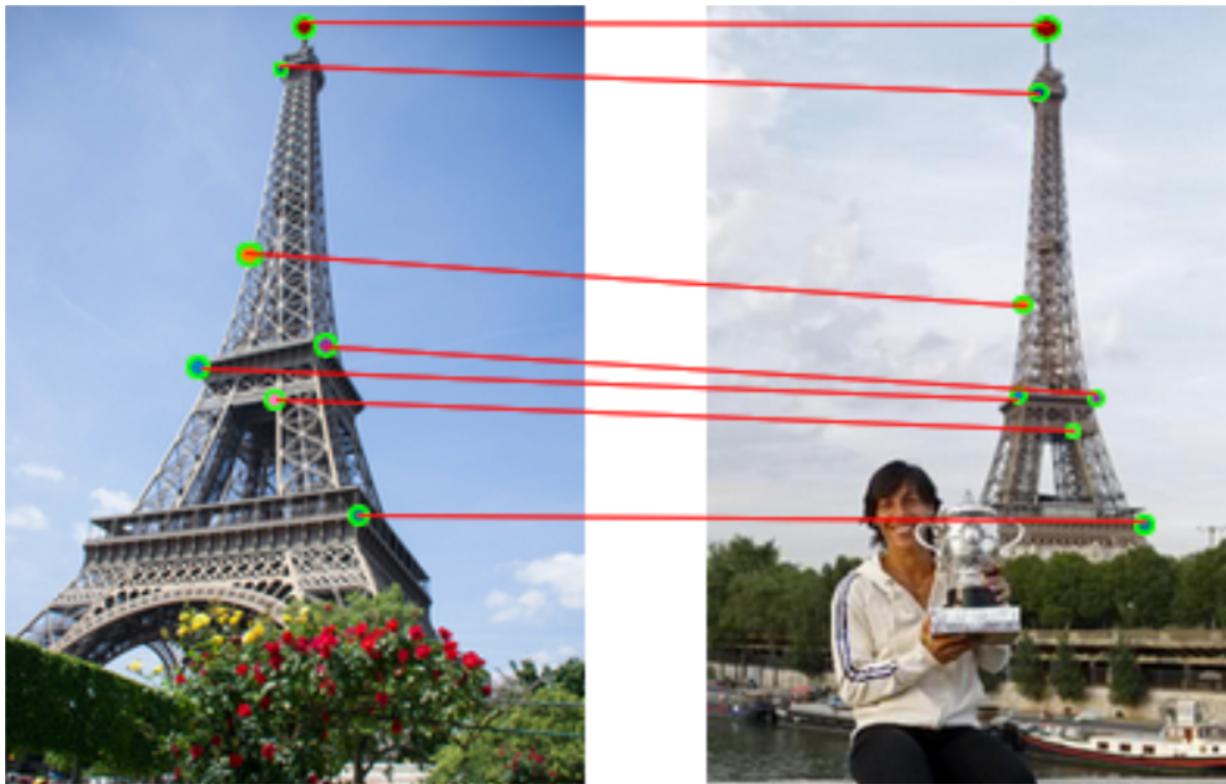


FIGURE 6 – Exemple de "features matching"

Une zone constitue un bon choix de features si elle est :

- Répétable : une feature doit se retrouver dans les images représentant la même scène malgré les différences géométriques et photométriques. Une feature doit donc présenter des propriétés d'invariance à ces transformations.
- Distinctive : une feature doit être suffisamment unique et non ambiguë au sein d'une image pour faciliter le matching. Ce sont les informations contenues dans son descripteur qui doivent mettre en valeur sa particularité.
- Locale : une feature doit correspondre à une zone suffisamment petite, et elle est décrite selon son voisinage uniquement. Cela permet d'éviter les difficultés de matching dues aux phénomènes d'occlusion et de background cluster.

Pour détecter des features classiques : le filtre de Canny pour les bords, puis le détecteur de Harris-Stephens pour les coins. Les coins localisés sont invariants par rotation, mais pas par changement d'échelle ! Cela signifie que l'on parvient à détecter les mêmes coins (avec des orientations différentes) lorsqu'on applique une rotation à l'image, mais pas lorsqu'on change le zoom. Il a fallu attendre l'algorithme SIFT pour caractériser le contenu visuel d'une image de la façon la plus indépendante possible de l'échelle.

Une fois que l'on extrait les features, il faut les quantifier. Pour cela, on a recours à un bag-of-features représentant une image par un "sac" dans lequel on a mis ses features en vrac. Mathématiquement, c'est un vecteur créé en deux temps : On crée un dictionnaire de "visual words" en appliquant un algorithme de clustering aux descripteurs de features construits à l'étape 1, comme le k-means. Les "visual words" correspondent alors aux centres des clusters trouvés. On construit un histogramme qui indique la fréquence d'apparition de chaque "visual word" dans l'image.

La classification supervisée est la dernière étape de notre méthode de résolution : l'objectif est d'apprendre les règles de décision permettant d'assigner correctement une représentation bag-of-features à une classe.

Cela signifie que l'on va entraîner un algorithme d'apprentissage supervisé (k-NN, Régression logistique, SVM, SVM à noyau) sur les bag-of-features construits à l'étape 2.

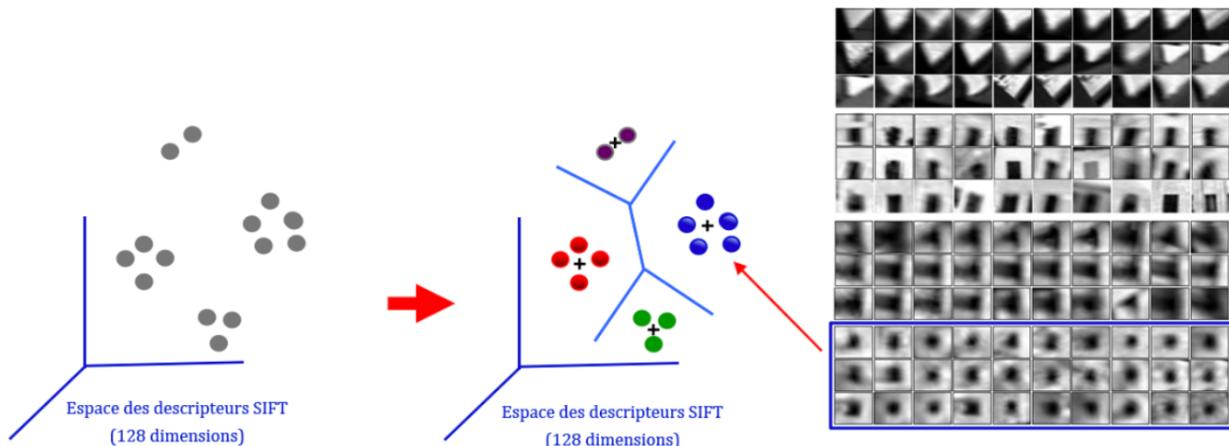


FIGURE 7 – Schéma de fonctionnement de la classification supervisée

5.2.2 Exemple de modèle classique de Human Pose Estimation.

Un exemple d'approches basées sur les caractéristiques descriptives (features-based en anglais) utilise une méthode appelée "Pictorial Structure Framework" (PSF) en combinaison avec un algorithme de Random Forest pour prédire les positions des articulations du corps humain. Supposons que vous avez une image d'un humain debout, et que vous souhaitez prédire les positions des keypoints de son corps, tels que les coudes, les genoux, les épaules, etc. Tout d'abord, vous devez extraire des caractéristiques descriptives à partir de l'image. Ces caractéristiques peuvent inclure des informations telles que les bords, les textures, les couleurs et les formes des parties du corps humain. Le rôle du premier composant du discriminateur utilise ces caractéristiques descriptives pour modéliser la probabilité de la présence d'une certaine partie du corps humain à un emplacement particulier de l'image. Ensuite, le Prior modélise la distribution de probabilité de la posture à partir de la sortie du Discriminateur. L'objectif du PSF est de représenter le corps humain comme une collection de coordonnées pour chaque partie du corps dans une image donnée. Pour prédire les positions des articulations, un algorithme de Random Forest est utilisé. Ce dernier permet d'entraîner des régresseurs non-linéaires pour chaque articulation, avec des caractéristiques de l'image en entrée comme entrée. Les avantages de cette approche sont une clarté dans la représentation des membres et articulations du corps humain. Cependant, cette approche peut échouer lorsqu'une partie du corps est cachée et donc non visible à l'image. Présentation des approches basées sur les réseaux de neurones (deep learning) Les approches basées sur les réseaux de neurones, également connues sous le nom de deep learning, sont des méthodes de vision par ordinateur qui utilisent des réseaux de neurones artificiels pour apprendre des modèles à partir de données. Les réseaux de neurones profonds peuvent apprendre des caractéristiques de manière autonome à partir de données d'entraînement, ce qui leur permet d'identifier et de suivre les objets ou les personnes de manière efficace.

5.2.3 Comprendre les réseaux de neurones convolutifs

En 2012, une révolution se produit : lors de la compétition annuelle de vision par ordinateur ILS VRC, un nouvel algorithme de Deep Learning pulvérise les records ! Il s'agit d'un réseau de neurones convolutif appelé AlexNet. Les réseaux de neurones convolutifs ont une méthodologie similaire à celle des méthodes traditionnelles d'apprentissage supervisé : ils reçoivent des images en entrée, détectent les features de chacune d'entre elles, puis entraînent un classifieur dessus. Cependant, les features sont apprises automatiquement ! Les CNN réalisent eux-mêmes tout le boulot fastidieux d'extraction et description de features : lors de la phase d'entraînement, l'erreur de classification est minimisée afin d'optimiser les paramètres du classifieur ET les features ! De plus, l'architecture spécifique du réseau permet d'extraire des features de différentes complexités, des plus simples aux plus sophistiquées. L'extraction et la hiérarchisation automatiques des features, qui s'adaptent au problème donné, constituent une des forces des réseaux de neurones convolutifs :

plus besoin d'implémenter un algorithme d'extraction "à la main", comme SIFT ou Harris-Stephens. Il existe quatre types de couches pour un réseau de neurones convolutif : la couche de convolution, la couche de pooling, la couche de correction ReLU et la couche fully-connected.

5.2.4 Comment fonctionne un RCCN ?

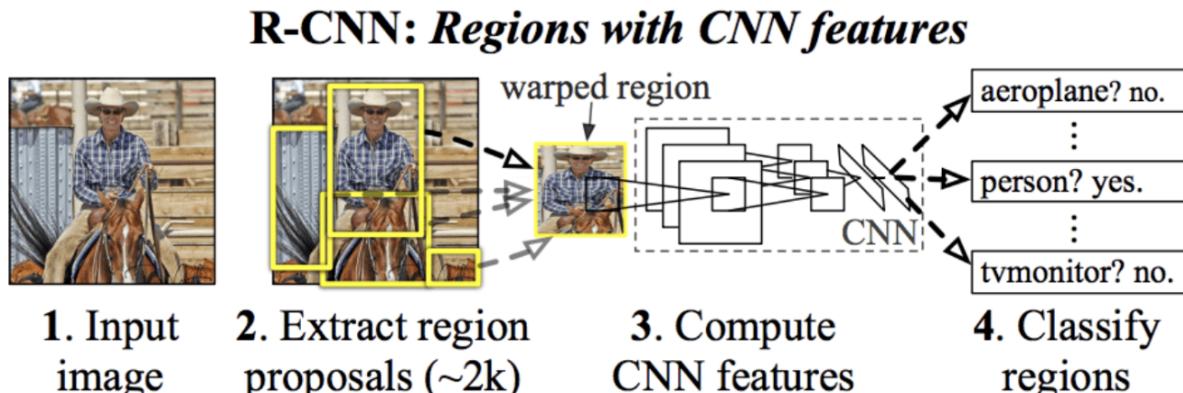


FIGURE 8 – Fonctionnement d'un R-CNN

Ils fonctionnent suivant ce schéma : Proposition de régions d'intérêt (RoI), Extraction de caractéristiques, Classification des régions, Régression des boîtes englobantes, Non-maximum suppression

La première étape du fonctionnement d'un RCNN est la proposition de régions d'intérêt (RoI). Cette étape consiste à extraire les régions d'intérêt de l'image en utilisant une méthode de recherche de zones intéressantes. Prenons par exemple une image d'un chat qui se trouve sur une chaise et entouré d'autres objets. Pour détecter le chat, le modèle doit d'abord trouver les régions d'intérêt de l'image, c'est-à-dire les zones qui pourraient contenir l'objet d'intérêt. Cette étape est souvent réalisée à l'aide d'un algorithme de recherche de régions prometteuses comme Selective Search ou EdgeBoxes. Ces algorithmes examinent l'image à plusieurs échelles et proposent des régions d'intérêt en fonction de critères tels que la texture, la couleur, la forme et la proximité avec d'autres régions.

Une fois que les régions d'intérêt ont été extraites, chaque région est redimensionnée en une image de taille fixe et passée à travers un réseau de convolution pré-entraîné, tel que VGG 16 ou ResNet. Prenons l'exemple de la région d'intérêt qui correspond au chat sur la chaise. Cette région sera redimensionnée en une image de taille fixe (par exemple 224x224 pixels) et passée à travers le réseau de convolution pré-entraîné pour extraire des caractéristiques visuelles telles que des bords, des textures, des couleurs, etc.

Les caractéristiques extraites de chaque région sont ensuite utilisées pour classer les objets présents dans chaque région. Cette étape peut être réalisée avec une couche entièrement connectée qui produit un score pour chaque classe d'objet possible. Par exemple, si la région d'intérêt correspondant au chat sur la chaise est passée à travers une couche entièrement connectée, le modèle pourrait produire des scores pour différentes classes d'objet, comme "chat", "chaise", "table", etc. Le modèle pourrait alors classer la région d'intérêt comme étant un chat. En plus de prédire la classe de chaque région d'intérêt, on cherche également à déterminer les coordonnées de la boîte englobante de chaque objet. Cette étape consiste à ajuster les dimensions et la position de la boîte englobante pour mieux s'adapter à l'objet présent dans la région. Prenons l'exemple de la région d'intérêt correspondant au chat sur la chaise. Le modèle pourrait prédire les coordonnées de la boîte englobante qui entoure le chat (par exemple, les coordonnées de l'angle supérieur gauche et de l'angle inférieur droit de la boîte). Cela permet de localiser précisément l'objet d'intérêt dans l'image.

L'ensemble du modèle est ensuite entraîné en minimisant une fonction de coût qui prend en compte à la fois la classification et la régression des boîtes englobantes pour chaque région d'intérêt. En général, plus le modèle est entraîné sur un grand nombre d'images annotées, plus il est performant dans la détection des objets dans des nouvelles images.

Pour éviter d'avoir des détections en double ou des boîtes englobantes qui se chevauchent, une étape de

suppression de non-maximum est souvent réalisée. Cette étape consiste à filtrer les détections qui ont un score inférieur à un seuil prédéfini et à supprimer les boîtes englobantes qui se chevauchent significativement.

5.2.5 Comparaison des avantages et des inconvénients de ces différentes approches

Les approches de “pose estimation” peuvent être basées sur des caractéristiques descriptives ou des réseaux de neurones profonds. Les approches basées sur des caractéristiques descriptives ont l'avantage de fournir une compréhension intuitive de la nature des caractéristiques utilisées pour la détection de la pose. Elles sont également plus rapides à exécuter et nécessitent moins de données d'entraînement que les approches basées sur les réseaux de neurones. Cependant, elles peuvent être sensibles aux changements d'éclairage, aux occlusions et à la variation de l'apparence. Les approches basées sur les réseaux de neurones ont l'avantage de pouvoir apprendre automatiquement les caractéristiques les plus pertinentes pour la détection de la pose, ce qui leur permet d'être plus robustes à la variation de l'apparence. Elles peuvent également être entraînées sur une grande quantité de données pour améliorer leur précision. Cependant, elles nécessitent une grande quantité de données d'entraînement et des ressources informatiques importantes pour l'entraînement et l'inférence.

5.3 Les techniques d'annotation de données

5.3.1 Présentation des différents types de données utilisés pour entraîner les modèles de pose estimation

Les données utilisées pour entraîner les modèles de pose estimation peuvent être de différents types, notamment des images 2D, des images 3D, des vidéos et des données de capteurs inertielles. Les images 2D sont souvent utilisées pour la pose estimation des personnes, car elles sont faciles à obtenir à partir de caméras standard. Les images 3D sont utiles pour la “pose estimation” d'objets en trois dimensions, telles que des véhicules ou des bâtiments. Les vidéos peuvent être utilisées pour suivre la pose d'un objet ou d'une personne dans le temps, ce qui permet d'obtenir des informations sur le mouvement. Les données de capteurs inertiel, telles que les capteurs de mouvement portés sur le corps, peuvent également être utilisées pour la “pose estimation”, en particulier pour les applications de réalité augmentée ou virtuelle. Les données d'entraînement doivent être représentatives de la tâche de pose estimation souhaitée, afin que le modèle puisse généraliser à de nouvelles données. Les données d'entraînement doivent également être suffisamment diverses pour couvrir différentes conditions d'éclairage, de fond et de variation de l'apparence. L'utilisation de grandes quantités de données d'entraînement est souvent nécessaire pour obtenir des modèles de pose estimation de haute qualité.

5.3.2 Présentation des techniques d'annotation manuelle et semi-automatique de ces données

Les données utilisées pour entraîner les modèles de pose estimation doivent être annotées avec des étiquettes décrivant la position et l'orientation de l'objet ou de la personne dans l'image ou la vidéo. Les techniques d'annotation manuelle sont souvent utilisées pour produire des données d'entraînement de haute qualité. Ces techniques consistent à faire annoter les données par des annotateurs humains qui indiquent manuellement les positions des différentes parties de l'objet ou de la personne dans l'image ou la vidéo. Cette technique peut être coûteuse et chronophage, mais elle permet d'obtenir des données d'entraînement précises et fiables. Les techniques d'annotation semi-automatique permettent de réduire le coût et le temps nécessaires à l'annotation manuelle en utilisant des algorithmes pour assister les annotateurs humains. Par exemple, des algorithmes peuvent être utilisés pour détecter automatiquement certaines parties de l'objet ou de la personne dans l'image ou la vidéo, puis les annotateurs humains peuvent ajuster manuellement les positions. Cette technique peut permettre d'obtenir des données d'entraînement de qualité comparable à l'annotation manuelle, tout en réduisant le coût et le temps nécessaires. Cependant, les résultats peuvent dépendre de la qualité de l'algorithme utilisé pour assister l'annotation semi-automatique. En général, le choix de la technique d'annotation dépendra des ressources disponibles et de la qualité des données d'entraînement requise pour la tâche de pose estimation souhaitée.

5.4 Bases de données

Présentation des bases de données les plus utilisées pour entraîner les modèles de pose estimation Il existe plusieurs bases de données utilisées pour entraîner les modèles de “pose estimation”. Voici une présentation des plus courantes :

- **COCO (Common Objects in Context)** : c'est une base de données d'images annotées qui contient plus de 200 000 images avec plus de 250 000 annotations de pose pour plus de 1 000 catégories d'objets. Elle est souvent utilisée pour la détection de pose multi-personnes.
- **MPII Human Pose** : c'est une base de données d'images de personnes en action, avec des annotations de pose pour 24 parties du corps humain. Elle contient environ 25 000 images d'entraînement et 3 000 images de test.
- **Human3.6M** : c'est une base de données d'images de personnes réalisant des actions spécifiques en 3D, avec des annotations de pose pour 32 parties du corps. Elle contient environ 3,6 millions de poses annotées et est souvent utilisée pour l'entraînement de modèles de pose estimation en 3D.
- **LSP (Leeds Sports Pose)** : c'est une base de données d'images de personnes pratiquant des sports, avec des annotations de pose pour 14 parties du corps. Elle contient environ 2 000 images d'entraînement et 1 000 images de test.
- **Penn Action** : c'est une base de données d'images d'actions humaines, avec des annotations de pose pour 13 parties du corps. Elle contient environ 2 400 séquences vidéo, avec plus de 232 000 cadres annotés.

Ces bases de données sont souvent utilisées pour entraîner des modèles de pose estimation en raison de leur qualité, de leur taille et de leur diversité. Cependant, il est important de noter que chaque base de données peut être adaptée à une tache de “pose estimation” spécifique en fonction des besoins.

5.4.1 Analyse des caractéristiques de ces bases de données

Les bases de données utilisées pour l'entraînement des modèles de pose estimation ont des caractéristiques différentes qui les rendent plus ou moins adaptées à certaines tâches.

- Taille : La taille de la base de données est importante car elle détermine le nombre d'exemples d'entraînement disponibles pour le modèle. Les bases de données telles que COCO ou Human3.6M contiennent des centaines de milliers de données d'entraînement, tandis que des bases de données plus petites, telles que LSP, contiennent seulement quelques milliers.
- Diversité : la diversité des données est importante car elle permet aux modèles de généraliser à différents scénarios et situations. Les bases de données telles que COCO ou MPII Human Pose contiennent des images de personnes dans des contextes variés, tandis que des bases de données plus spécifiques, telles que Penn Action, se concentrent sur des actions spécifiques.
- Qualité de l'annotation : la qualité de l'annotation est importante, car elle détermine la précision des annotations de pose utilisées pour l'entraînement du modèle. Les bases de données telles que COCO ou Human3.6M ont été annotées manuellement par des experts, tandis que d'autres bases de données, telles que MPI-INF-3DHP, utilisent des annotations automatiques qui peuvent être moins précises.
- Disponibilité de la base de données : la disponibilité de la base de données est importante, car elle détermine si la base de données peut être utilisée pour un projet de “pose estimation” spécifique. Certaines bases de données, telles que COCO, sont largement utilisées et disponibles publiquement, tandis que d'autres, telles que MPI-INF-3DHP, sont plus difficiles à obtenir.

5.4.2 Présentation des métriques utilisées pour évaluer ces performances

Les métriques d'évaluation les plus couramment utilisées pour mesurer les performances des modèles de pose estimation sont le PCK (Percentage of Correct Keypoints) et l'AP (Average Precision). Le PCK mesure le pourcentage de points clés correctement détectés dans une image. Les points clés sont considérés

comme correctement détectés si leur distance avec les points clés annotés manuellement est inférieure à un seuil prédéfini. Le seuil dépend généralement de la taille de l'objet dans l'image. Plus le seuil est faible, plus la tâche est difficile. L'AP mesure la précision moyenne de la détection des points clés dans toutes les images d'une base de données. L'AP est calculée en prenant en compte la précision et le rappel du modèle. Le rappel mesure la proportion de points clés correctement détectés par rapport au nombre total de points clés dans la base de données. La précision mesure la proportion de points clés correctement détectés par rapport au nombre total de points clés détectés par le modèle. En plus du PCK et de l'AP, il existe d'autres métriques d'évaluation telles que le PCKh (Percentage of Correct Keypoints, Head), le PCKt (Percentage of Correct Keypoints, Torso) et le PCKm (Percentage of Correct Keypoints, Mean). Ces métriques mesurent le pourcentage de points clés correctement détectés pour différentes parties du corps, telles que la tête, le torse et les membres.

5.4.3 Analyse des performances des modèles de pose estimation récents

Ces dernières années, de nombreux modèles de pose estimation ont été proposés pour résoudre les problèmes de reconnaissance de la pose humaine en temps réel. Les réseaux de neurones convolutifs sont devenus la méthode de choix pour la plupart des travaux récents. Les modèles de pose estimation les plus récents ont montré des performances impressionnantes en termes de précision, de vitesse d'inférence et de robustesse aux variations de pose et d'occlusion. Parmi les modèles de "pose estimation" récents, on peut citer OpenPose, HRNet, SimpleBaseline, HigherHRNet, et bien d'autres. OpenPose est un modèle populaire qui utilise un réseau de neurones en cascade pour détecter les points clés du corps humain. HRNet utilise une architecture de réseau en parallèle avec des blocs résiduels pour capturer des informations à différentes échelles spatiales. SimpleBaseline est un modèle basé sur ResNet qui prédit les coordonnées 2D des points de repère du corps en utilisant une approche de régression. HigherHRNet est une version améliorée de HRNet, avec une architecture de réseau plus profonde et une meilleure représentation de la structure hiérarchique du corps humain.

6 Implémentation d'un algorithme de classification d'image

Afin de compléter ma compréhension autour de la computer vision, j'ai décidé de suivre un tutoriel afin d'utiliser un modèle de classification déjà entraîné. Le modèle VGG 16 que nous allons utiliser est un modèle préentraîné sur des millions d'images de la base imangenet. Les avantages sont multiples : cela permet de répondre à un problème sur un dataset spécifique sans partir de zéro et d'autre part, les temps d'entraînement sont également bien plus courts qu'un modèle complet. Dans l'exemple présenté, le modèle pourrait réussir sans doute très bien pour la reconnaissance de chien/chat sans entraînement spécifique. Cependant, j'ai décidé de reprendre ce type de réseau et de mettre en œuvre un entraînement sur un classifieur spécifique à nos jeux de données. À noter que j'utilise les librairies Keras de tensorflow.

6.1 Architecture du réseau de neurones convolutif VGG16

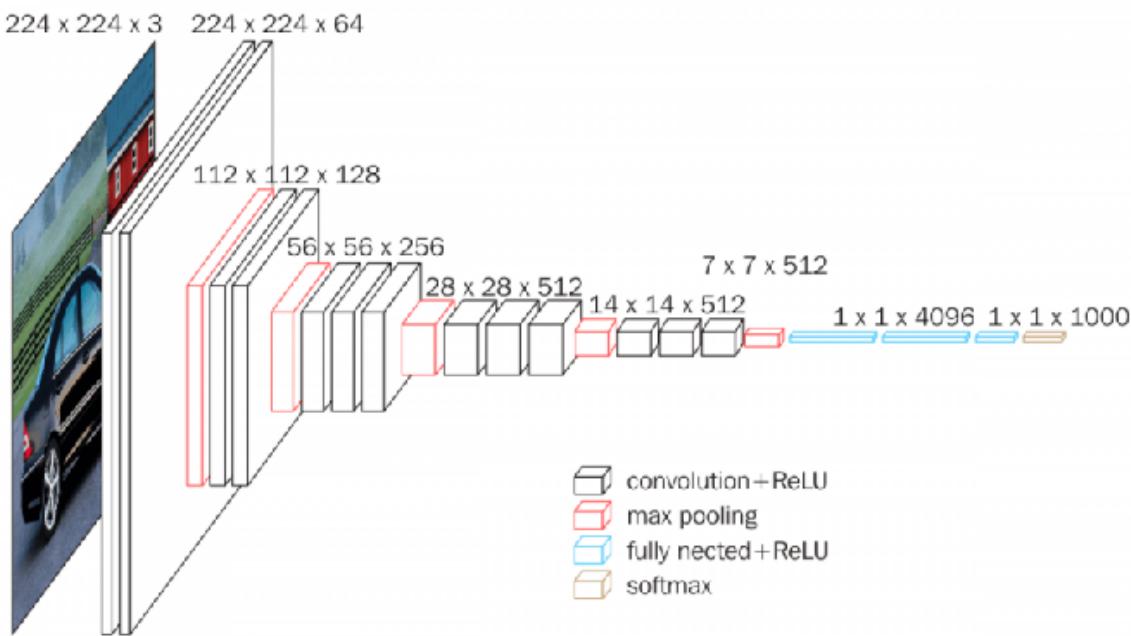


FIGURE 9 – Architecture du modèle VGG16

Le modèle de réseau CNN présenté a été proposé par K. Simonyan et A. Zisserman de l'université d'Oxford. Dans ce modèle nous n'allons pas utiliser la couche de classification du VGG16, mais nous allons ajouter et entraîner notre propre classifieur, spécifique à la réponse au problème de nos propres données. L'image en entrée à une taille de 224 224 3(RVB). Les 3 dimensions étant les couches RVB constituant les images. Ensuite, les layers de convolution grâce à leur mécanisme créent un ensemble de couches supplémentaires (features) qui "augmentent" les dimensions de l'image (épaisseur) en ajoutant les résultats des filtres de convolution. La couche de pooling réduit la largeur/hauteur de l'image. Au final, avant la couche entièrement connectée, nous avons une image très épaisse, mais très petite. Un rapide calcul montre qu'il y a $224 \times 224 \times 3 = 150\,528$ pixels en entrée pour 1 835 008 en sortie ($7 \times 7 \times 512$). En passant cette partie d'extraction de features du réseau, "l'image" résultante s'est donc enrichie en information.

6.2 Combinaison du VGG16 et de notre classifieur

Afin de résoudre mon problème de classification d'images de chien et de chats, j'utilise le réseau VGG16 en ajoutant ma propre couche de classification. Cette couche FC (entièlement connectée) est un réseau de neurones classique qui va permettre à partir de la sortie du VGG16 de faire une prédiction. Sa couche de sortie contient n neurones (n représentant le nombre de classes que nous souhaitons prédire) dans ce cas : 2.

Pour définir le Classifieur, j'ajoute la couche de classification (Top Layer) sur le réseau de Convolution :

- aplatissement de l'image de sortie en un vecteur

- ajout de 2 couches de 128 neurones
 - ajout d'une couche de régularisation pour éviter l'overfitting
 - ajout de la couche de sortie à deux neurones pour la prédiction (probabilistique)
- Cette dernière effectue la descente de gradient pour le calcul numérique de l'optimisation de la fonction de coût
- le feed forward des input vers la couche de sortie
 - la backpropagation en fonction de l'erreur de prédiction

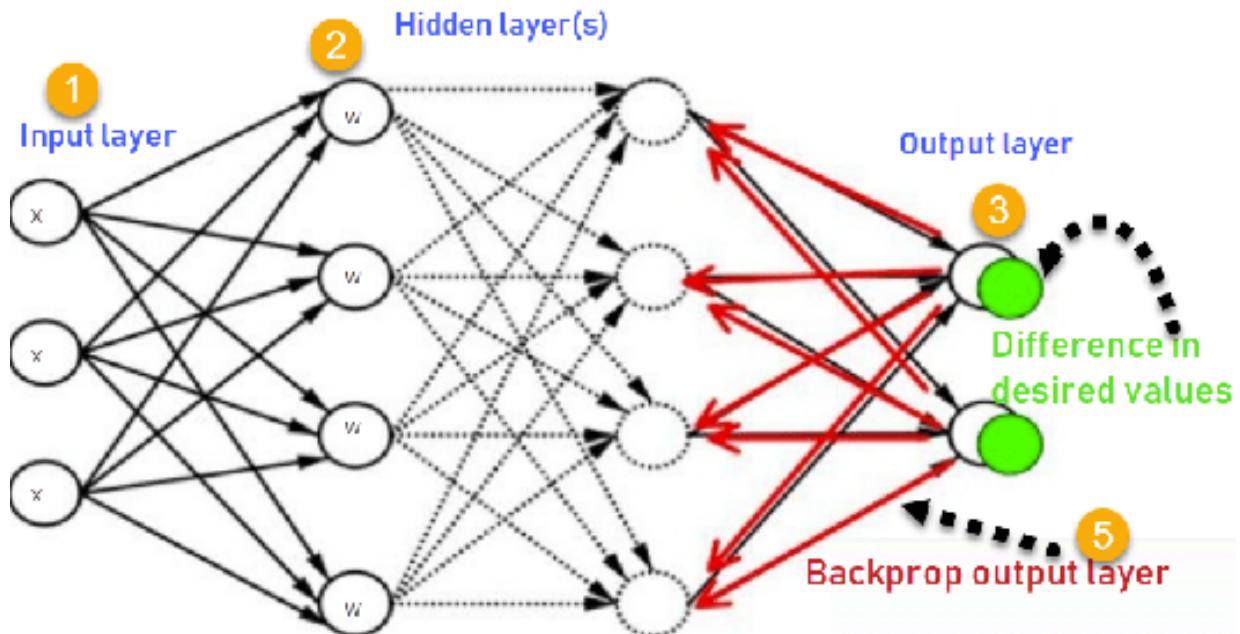


FIGURE 10 – Schéma du modèle VGG16

La figure 10 représente le classifieur. L'entrée du classifieur est un vecteur aplati issu de la sortie du modèle de convolution pour chaque image.

6.3 Résultats

Après assemblage du modèle final et la phase d'entraînement du modèle, il nous reste à l'évaluer et lui demander de faire les prédictions sur nos images de tests

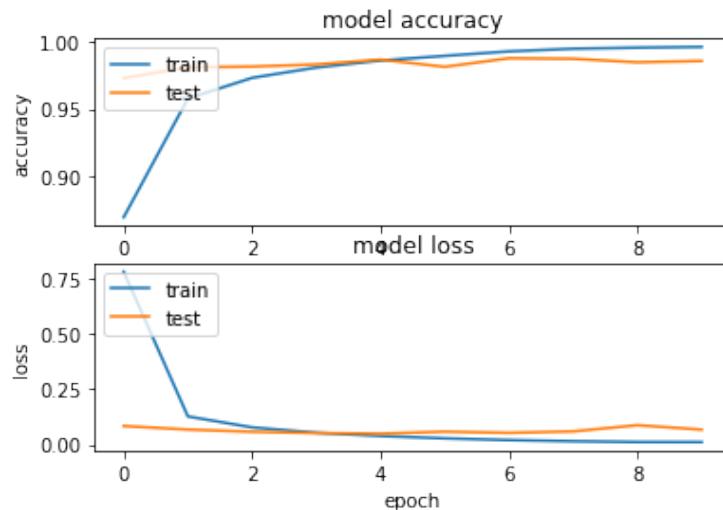


FIGURE 11 – Entrainement sur plusieurs epochs

7 Les spécificités de l'estimation de pose appliquée au sport

Après avoir examiné de manière approfondie les travaux existants dans le domaine de l'estimation de position, j'ai dirigé mes efforts vers l'estimation de la position humaine, en mettant principalement l'accent sur les activités physiques.

7.1 Présentation et enjeux

La “pose estimation” 2D peut être utilisée pour extraire des informations de mouvement des sportifs, telles que la vitesse, la direction, la longueur de la foulée, etc. Elle permet également de détecter les erreurs de technique de course ou les risques de blessure. En combinant la pose estimation 2D avec d'autres données telles que la vitesse de déplacement ou la fréquence cardiaque (fusion multimodale), il est possible d'obtenir une analyse encore plus fine des mouvements des sportifs lors de la course.

La “pose estimation” 2D en sport est encore confrontée à plusieurs défis. L'un des défis les plus importants est la complexité des mouvements des sportifs, qui peuvent être rapides, saccadés ou imprévisibles. Cela peut rendre la détection et la localisation des parties du corps plus difficile, en particulier en cas d'occlusions ou de déformations. La variabilité inter-individuelle des poses peut rendre l'apprentissage des modèles de pose estimation 2D plus difficile, en particulier pour les sportifs ayant des morphologies ou des styles de mouvement différents. Il est donc important de disposer de jeux de données de pose variés et de techniques d'apprentissage robustes pour obtenir des modèles généralisables. (Un autre défi est la nécessité de traiter des vidéos en temps réel pour permettre une analyse en temps réel des mouvements des sportifs. Cela nécessite des algorithmes de pose estimation 2D rapides et efficaces.)

7.2 Les 3 algorithmes de human pose estimation pour sport

Trois algorithmes se démarquent dans le domaine de l'estimation de la pose humaine.

- **OpenPose** : algorithme très populaire pour la pose estimation humaine. Il a une très bonne précision et est capable de détecter plusieurs personnes en même temps, même lorsqu'elles se chevauchent. Il possède également une API open source facile à utiliser et des exemples de codes disponibles pour les débutants. Cependant, OpenPose nécessite un matériel performant pour fonctionner efficacement. Il peut être relativement lent sur des ordinateurs moins puissants.
- **AlphaPose** : algorithme de pose estimation humaine basé sur des réseaux de neurones. Il est rapide, précis et peut détecter plusieurs personnes simultanément. Il est également open source et dispose d'une API facile à utiliser. Mais AlphaPose a tendance à perdre en précision lorsque les personnes sont proches les unes des autres. De plus, sa mise en place peut être plus compliquée que celle d'OpenPose.
- **HRNet** : algorithme de pose humaine qui utilise une approche de réseau résiduel haute résolution. Il est capable de détecter les articulations de manière très précise et est moins sensible aux occlusions que d'autres algorithmes. Il est également open source et possède une API facile à utiliser. Il peut être lent à s'exécuter sur des ordinateurs moins puissants. Il peut également avoir des difficultés avec les poses qui ne sont pas courantes ou qui ne sont pas incluses dans les données d'entraînement.

7.3 Présentation d'une API qui propose des modèles de HPE

MMpose est une API open source développée par OpenMMLab qui propose des modèles d'estimation de pose humaine précis et rapides. Elle offre une large gamme d'algorithmes basés sur les réseaux de neurones, tels que HRNet, SimpleBaseline et High-Resolution Net, qui ont démontré une grande précision dans la détection des poses humaines en temps réel. MMpose a été développé en utilisant PyTorch et est capable de traiter des images en temps réel sur des GPU haut de gamme. Cette API est conçue pour être facilement intégrable dans des applications de vision par ordinateur grâce à son interface simple et ses fonctions prédéfinies, notamment l'inférence de pose, la visualisation de pose et l'entraînement de modèles personnalisés. Les modèles de MMpose sont pré-entraînés sur des bases de données de grande taille telles que COCO, MPII, et PoseTrack, ce qui permet d'obtenir des résultats de haute qualité pour diverses tâches de pose estimation.



FIGURE 12 – Logo de MMpose

En outre, MMpose dispose d'une communauté active de contributeurs et de développeurs, ce qui garantit une amélioration constante de la qualité des modèles proposés.



FIGURE 13 – Exemple d'inférence avec MMpose

8 Phase 1 : Mise en place de l'environnement

Pour commencer à faire de l'analyse de mouvements, j'ai donc choisi d'utiliser l'API MMpose. Grâce à leur documentation, et en me rendant sur leur site : <https://mmpose.readthedocs.io/en/latest/>, j'ai installé les différentes bibliothèques requises. Voici un exemple de commandes bash pour pouvoir créer un environnement fonctionnel (avec CPU).

```
1 conda create --name openmmlab python=3.8 -y  
2 conda activate openmmlab
```

```

3 conda install pytorch torchvision cpuonly -c pytorch
4 pip install -U openmim
5 mim install mmengine
6 mim install "mmcv>=2.0.0"
7 mim install "mmdet>=3.0.0"

```

Une fois que notre environnement est configuré, on peut installer le dépôt GitHub :

```

1 git clone https://github.com/open-mmlab/mmpose.git
2 cd mmpose
3 pip install -r requirements.txt
4 pip install -v -e .

```

9 Phase 2 : Présentation générale de l'utilisation des algorithmes de HPE

Grâce à l'API, il est possible d'utiliser facilement les algorithmes avec le module MMPOSEInferencer.

9.1 Un premier exemple de Human Pose Estimation sur un joueur de rugby

```

1 from mmpose.apis import MMPOSEInferencer
2
3 img = 'datas/dulin.jpeg' # replace this with your own image path
4 file = 'configs/body_2d_keypoint/topdown_heatmap/mpii/td-hm_hrnet-w48_8xb64-210e_mpii-256x256.
5 py'
6 checkpoints = 'https://download.openmmlab.com/mmpose/top_down/hrnet/hrnet_w48_mpii_256x256-92
7 cab7bd_20200812.pth'
8
9
10 inferencer = MMPOSEInferencer(pose2d = file,
11                                 pose2d_weights= checkpoints)
12
13 result_generator = inferencer(img, return_vis = True, out_dir = 'vis_results/')
14 result = next(result_generator)

```

L'algorithme prend en entrée une image aux différents formats possibles.

La variable file est chemin d'accès vers un fichier python de configuration du modèle, tandis que la variable checkpoints peut-être soit un chemin d'accès vers fichier de points de contrôle (ou poids) pré-entraînés pour le modèle, soit un URL de téléchargement de ce fichier.

Le fichier de configuration de l'algorithme contient plusieurs sections avec des commentaires pour spécifier les paramètres de l'exécution de l'algorithme. On y retrouve l'algorithme d'optimisation utilisé pour ajuster les poids du modèle pendant l'entraînement, le taux d'apprentissage utilisé pour ces ajustements, les paramètres du modèle lui-même tels que l'architecture du réseau, la taille des entrées et la profondeur du réseau. Il spécifie également les paramètres du jeu de données de base utilisé pour l'entraînement, les étapes de traitement des données, les augmentations de données et les tâches d'entraînement/inférence à effectuer dans l'algorithme. Enfin, il inclut la spécification des métriques d'évaluation à utiliser pour mesurer la performance du modèle.

Les résultats sont stockés dans la variable result. C'est un dictionnaire qui contient les clefs 'visualization' et 'predictions' dont les valeurs sont respectivement un tableau numpy représentant l'image prédite, une liste contenant pour chaque boîte englobante un dictionnaire avec les scores de détections de la boîte, ses coordonnées, les scores de détections des keypoints et leurs coordonnées.

Pour afficher la figure 14



FIGURE 14 – Exemple d'estimation de position sur Brice Dulin (joueur du Stade Rochelais)

```

1 import matplotlib.pyplot as plt
2 plt.imshow(result['visualization'])
3 plt.show()

```

Et, lorsqu'on affiche le premier élément de 'predictions' on obtient les informations sur l'estimation de pose humaine relative à Brice Dulin.

```

1 result['predictions'][0][0]

1 { 'keypoints': [[517.8267517089844, 382.3274230957031],
2 [432.2948913574219, 382.3274230957031],
3 [422.7913513183594, 254.02963256835938],
4 [479.8125915527344, 239.77432250976562],
5 [446.5502014160156, 372.8238830566406],
6 [451.3019714355469, 486.8663635253906],
7 [451.3019714355469, 249.27786254882812],
8 [418.0395812988281, 116.22830200195312],
9 [408.5360412597656, 87.71768188476562],
10 [389.5289611816406, 11.689361572265625],
11 [370.5218811035156, 216.01547241210938],
12 [365.7701110839844, 192.25662231445312],
13 [370.5218811035156, 120.98007202148438],
14 [465.5572814941406, 106.72476196289062],
15 [494.0679016113281, 187.50485229492188],
16 [427.5431213378906, 211.26370239257812]],
17 'keypoint_scores': [0.9143447279930115,
18 0.8326352834701538,
19 0.9141951203346252,
20 0.8946501016616821,
21 0.8945473432540894,
22 0.8945529460906982,
23 0.9398229122161865,
24 0.9117014408111572,
25 0.9552825689315796,
26 0.9198225736618042,

```

```

27 0.8265993595123291,
28 0.8483145236968994,
29 0.9045183658599854,
30 0.9282854795455933,
31 0.9183376431465149,
32 0.9159461855888367],
33 'bbox' : ([346.9585266113281,
34 8.363122940063477,
35 550.8936157226562,
36 494.9443664550781],),
37 'bbox_score' : 0.8811495}
38

```

Pour donner un ordre d'idée du temps de calcul, avec 4 CPU disponibles sur une VM et en utilisant un fichier de poids de 110 Mo, l'inférence prend 3,4 secondes par image. Si on découpe une vidéo en 400 frames, on devra attendre au moins 1360 secondes soit un peu plus d'une vingtaine de minutes

9.1.1 Présentation de chaque algorithme (inferencer) disponible

Tableau récapitulatif des algorithmes de HPE

MPII				Coco Dataset			
Architecture	Input size	Mean	Fichier de poids	Architecture	Input Size	AP	Fichier de poids
pose_hrnet_w48	256x256	0.901	ckpt	ViTPose-L	256x192	0.782	ckpt
pose_resnet_152	256x256	0.889	ckpt	pose_hrnet_w48_dark	384x288	0.772	ckpt
pose_hrnet_w48_dark	256x256	0.905	ckpt	rtmpose-l	256x192	0.758	ckpt
rtmpose-m	256x256	0.907	ckpt				

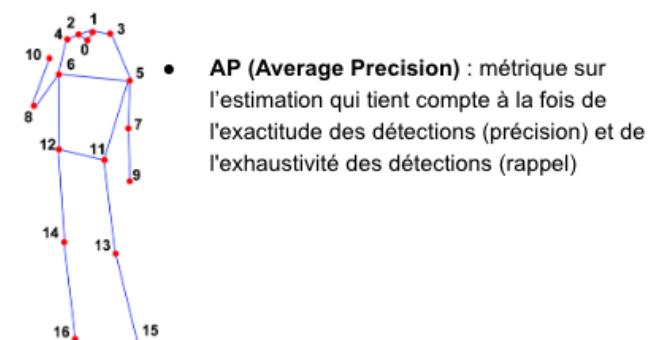
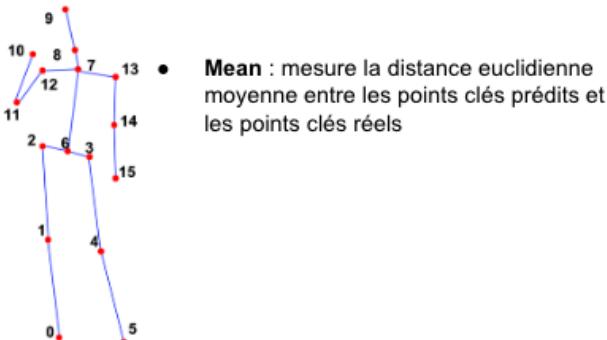


FIGURE 15 – Tableau récapitulatif des algorithmes de HPE

La figure 15 nous donne un récapitulatif des meilleurs algorithmes d'estimation de position entraîné sur le dataset MPII (annotation avec 16 points clefs), ou sur le dataset Coco (annotation avec 17 points clefs).

10 Phase 3 : Utilisation de Google Cloud Platform

Après avoir acquis une compréhension globale de l'estimation de pose humaine et à maîtriser l'utilisation de l'API de MMpose, notre objectif était de mettre en place un environnement de travail sur Google Cloud Platform (GCP). Il devait être à la fois accessible à distance et convivial pour tous les utilisateurs potentiels voulant s'essayer à l'HPE. Pour ce faire, nous avons configuré une machine virtuelle (VM) sur GCP où étaient préalablement installées les bibliothèques essentielles.

Ce travail, qui semble simple à première vue, s'est révélé plus complexe qu'il n'y paraît, en grande partie en raison des problèmes de dépendances et de gestion des versions de l'API MMpose. De plus, cette API a connu

une mise à jour pendant cette période, passant de la version 1.9 à la version 2.0, ce qui a ajouté un niveau de complexité supplémentaire. Cette expérience a été une opportunité précieuse pour moi, car elle m'a permis de consolider mes compétences, notamment en ce qui concerne la gestion des environnements, la configuration des fichiers, l'installation des bibliothèques, etc. Elle m'a également familiarisé avec les connexions SSH pour l'utilisation de la machine virtuelle depuis mon ordinateur, des compétences qui avaient été abordées lors de mes cours sur le GPGPU. En outre, elle m'a permis de me familiariser davantage avec l'utilisation de la plateforme GCP, enrichissant ainsi mon bagage professionnel.

11 Phase 4 : Inférence sur des vidéos de sportifs avec extraction des données de position des points clés du corps humain pour l'analyse des mouvements

Mon objectif à ce stade du projet était de construire une fonction qui, prenant en entrée une vidéo d'un sportif dans son mouvement, est capable de fournir en sortie une analyse de celui-ci. Cette analyse peut être simplement descriptive comme la visualisation du mouvement du pied du sportif ou comparative en essayant de montrer les anomalies du mouvement par rapport à une norme.

J'ai cherché à maintenir des conditions d'observation constantes en élargissant mon champ d'étude au-delà du domaine du rugby. J'ai rencontré des difficultés à obtenir suffisamment de vidéos de cours similaires dans ce domaine, ce qui m'a conduit à m'orienter vers l'athlétisme. Les plans de fin de course dans l'athlétisme offrent une vision globale des mouvements des sportifs, ce qui facilite l'analyse.

L'algorithme que j'ai développé se divise en trois parties distinctes. Tout d'abord, il découpe la vidéo en un certain nombre de frames. Ensuite, il utilise l'API MMpose pour effectuer l'inférence, permettant ainsi d'obtenir la position des points clés du corps humain sur chaque image. Enfin, la troisième partie de l'algorithme se concentre sur l'analyse de la position de ces points clés. Il convient de noter que, étant donné que l'algorithme effectue des prédictions sur chaque image de manière indépendante, des valeurs aberrantes peuvent apparaître dans les prédictions. C'est pourquoi nous avons opté pour une approche de moyennage, par exemple en calculant la position moyenne sur 4 frames, afin d'atténuer ces valeurs aberrantes et d'obtenir des résultats plus fiables.

Les deux premières étapes de l'algorithme, bien que moins passionnantes, ont constitué une part essentielle de mon travail, impliquant principalement des aspects de développement informatique. Toutefois, la partie la plus complexe et réfléchie de mon travail était centrée sur l'analyse de la position des points clés. Cette étape présentait un défi particulier, car l'algorithme ne fournissait que la position brute des points clés sur chaque image, nécessitant donc une transformation pour une analyse plus approfondie. Pour résoudre ce problème, j'ai opté pour une approche consistant à centrer, réduire et normaliser les positions de chaque point clé, permettant ainsi une analyse plus précise et significative des données. Cette phase de traitement des données a été cruciale pour obtenir des résultats fiables et exploitables dans notre étude.

12 Phase 5 : Présentation des résultats

Voici quelques graphiques que j'ai choisis pour illustrer les caractéristiques du mouvement de la course d'Usain Bolt lors de la fin d'un 100 mètres.

La figure 16 nous permet d'avoir une idée du mouvement du coureur que l'on peut illustrer davantage avec la figure 17.

On peut ainsi tracer l'évolution de la position des points clés qui nous intéressent.

On a aussi des indicateurs qui nous permettent de caractériser la course d'Usain Bolt, comme on peut le voir avec la figure 20 et la figure 21.

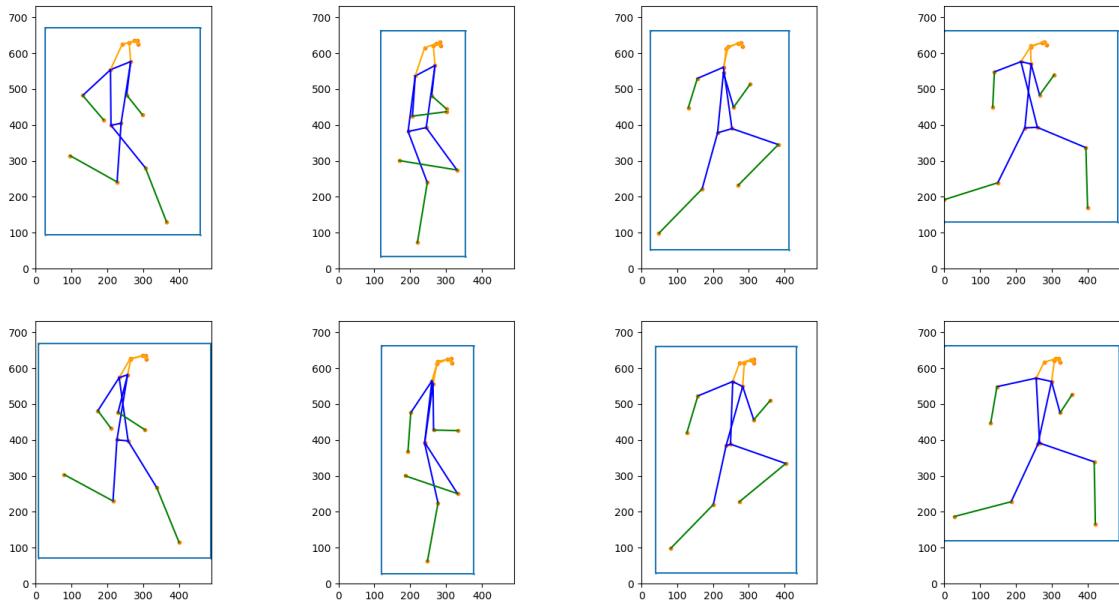


FIGURE 16 – illustration du mouvement

Visualisation de la première et de la dernière frame de la vidéo

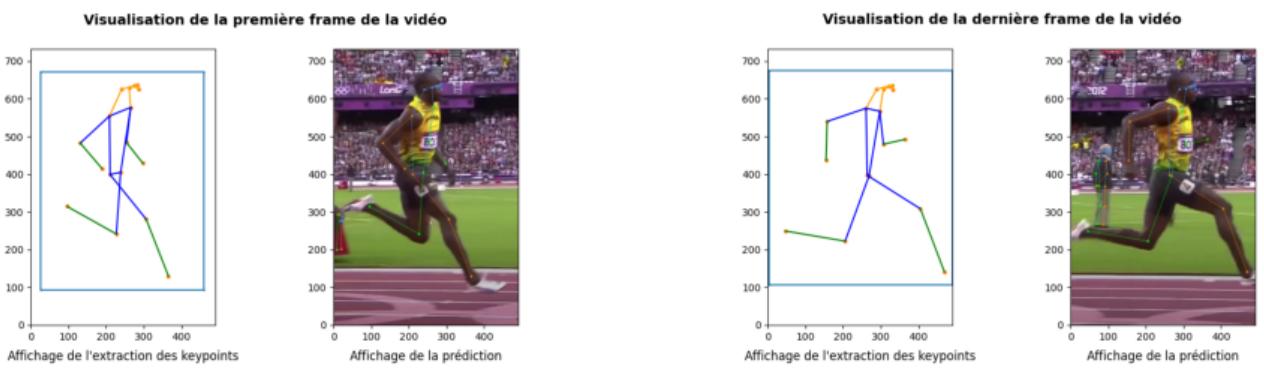


FIGURE 17 – illustration du mouvement 2

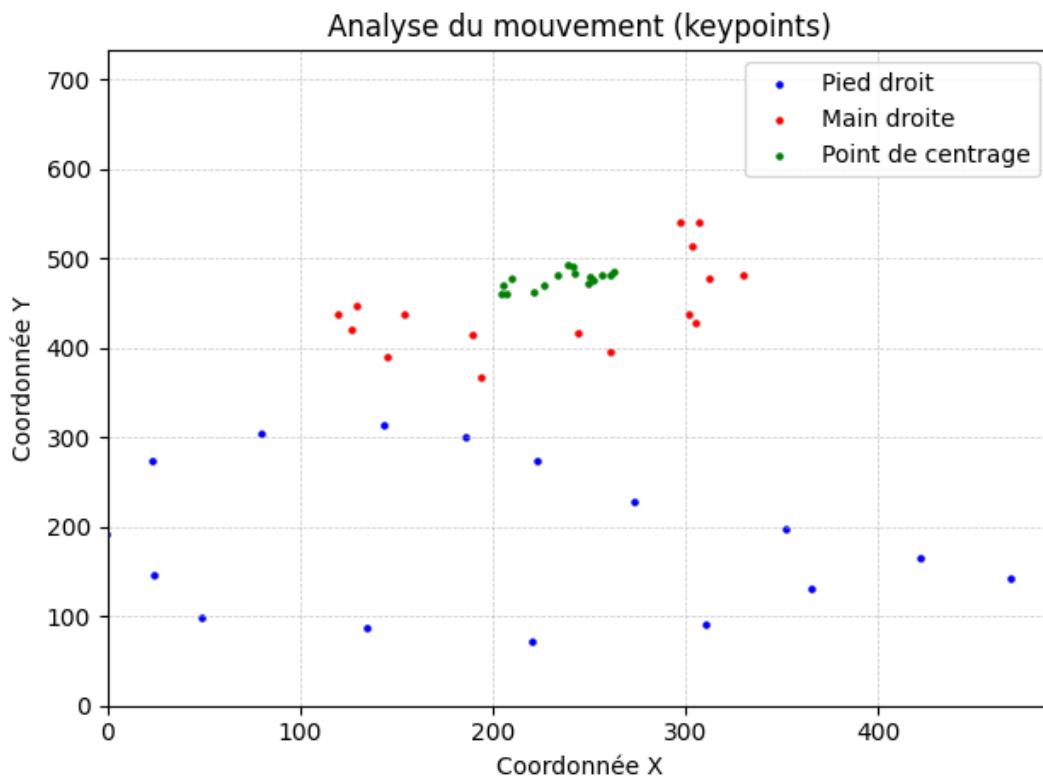


FIGURE 18 – Évolution de la position de la main droite, du pied droit et du centre du corps

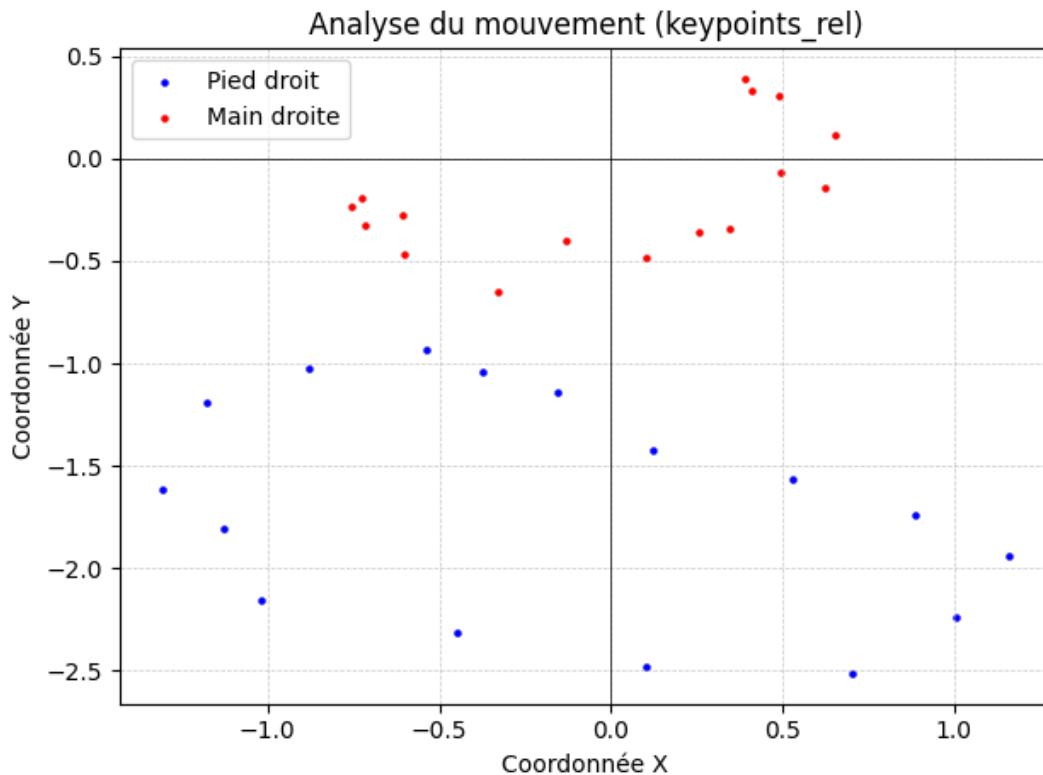


FIGURE 19 – Évolution de la position relative de la main droite et du pied droit par rapport au centre du corps

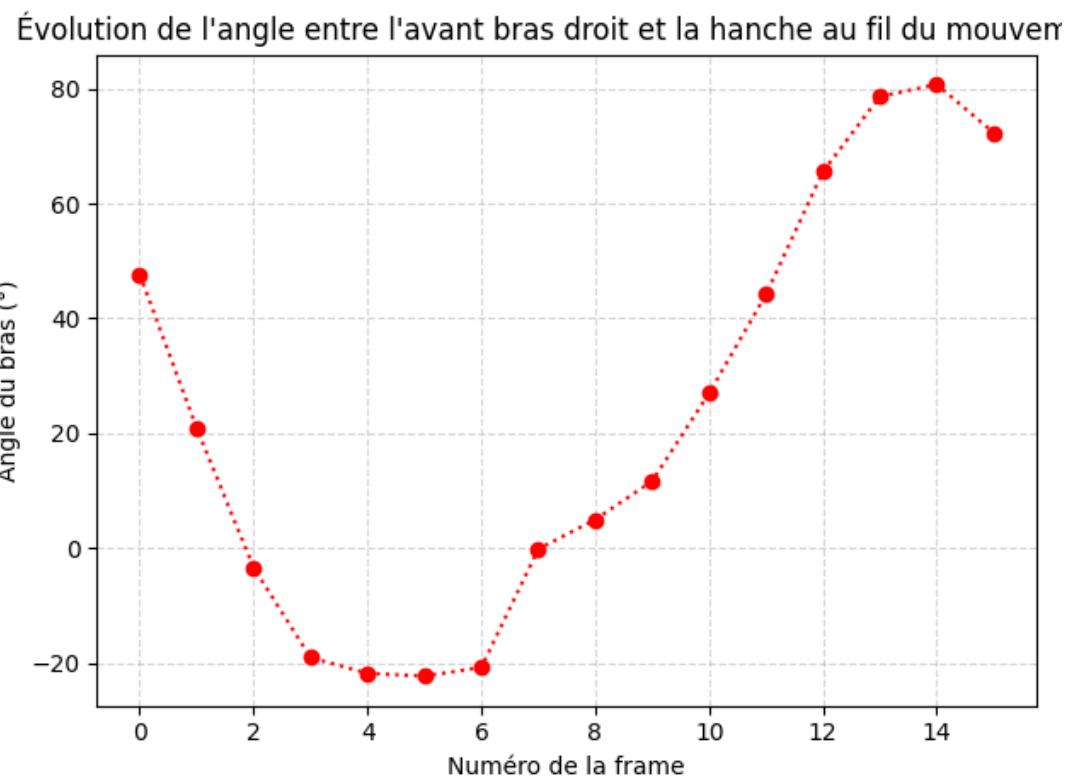


FIGURE 20 – Évolution de l'angle du bras droit

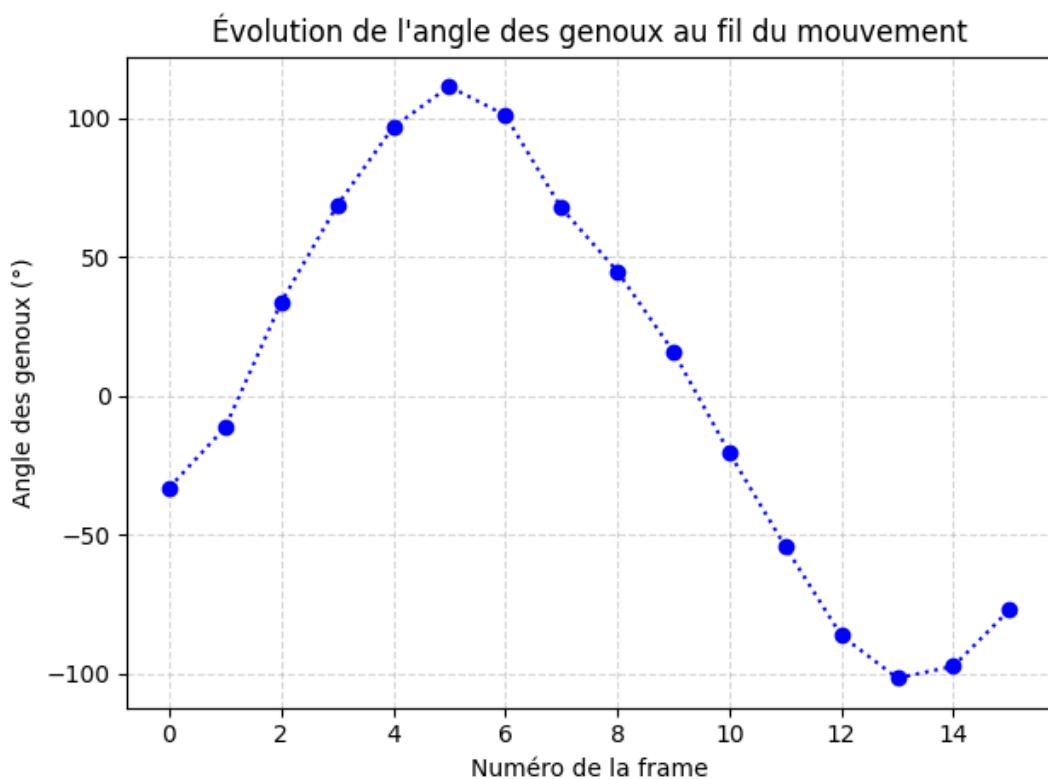


FIGURE 21 – Évolution de l'angle du genou droit

13 Phase 6 : Vers une application web

Dans le but de consolider l'ensemble de nos travaux jusqu'à présent et de continuer à acquérir des compétences en gestion de projets MLOps, nous avons entrepris le développement d'une application web. Cette application a pour objectif de permettre l'analyse de la position humaine à partir d'une vidéo en entrée. Initialement, la web app a été mise en place en local sur mon ordinateur en utilisant la bibliothèque Flask. Le développement de l'interface utilisateur a été réalisé en HTML, tandis que la gestion des fonctionnalités côté serveur a été implémentée en Python.

La figure 22 illustre la page d'accueil de l'application. Une fois la vidéo sélectionnée, on passe à l'analyse avec un bouton puis on obtient les informations présentées sur la page 23.

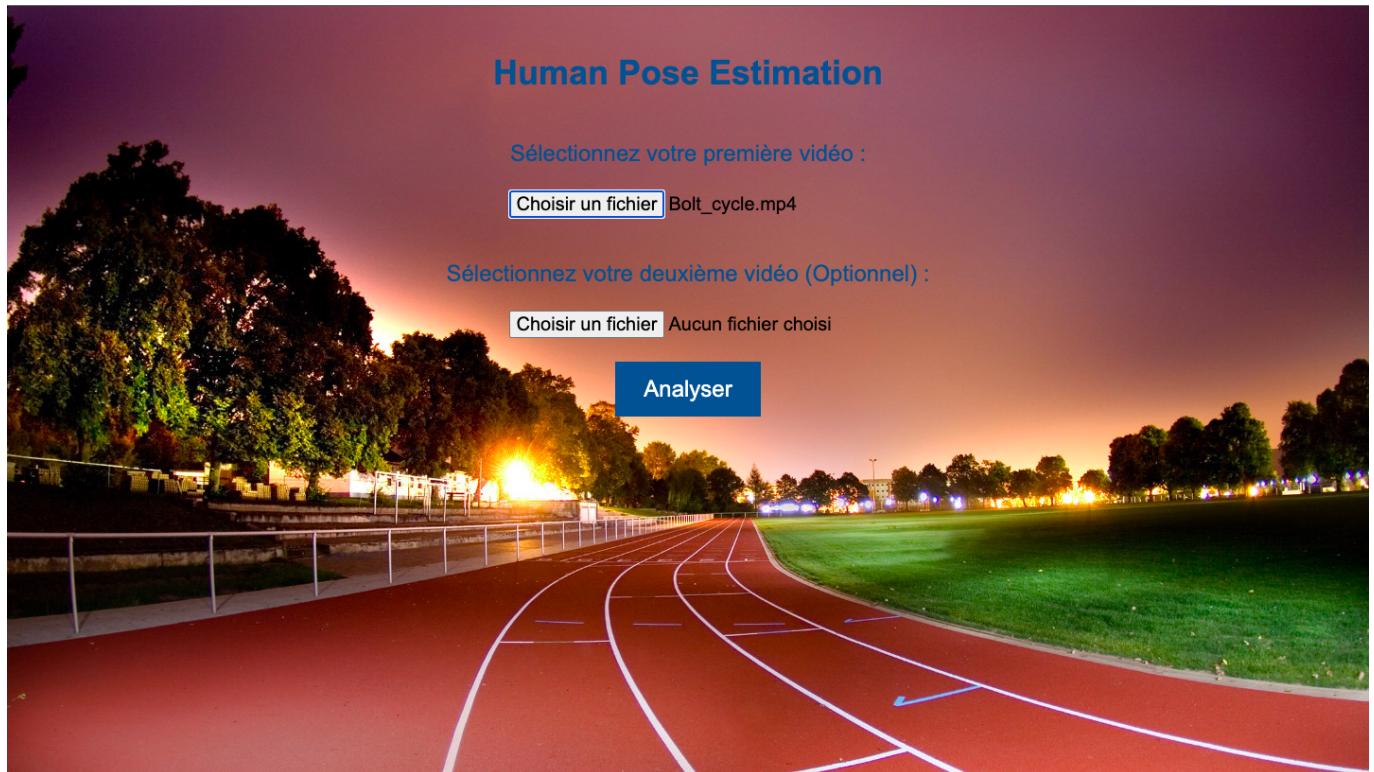


FIGURE 22 – Page d'accueil

Une fois que notre application fonctionnait en local, nous avons entrepris son déploiement sur la plate-forme GCP d'Avisia. Pour ce faire, nous avons opté pour la méthode de déploiement à partir d'une image Docker. Nous avons donc dockerisé l'ensemble de notre projet, y compris l'application, puis nous l'avons déployé via Docker. Ce processus s'est avéré être une tâche laborieuse et chronophage en raison de la non-optimisation de l'application, ainsi que des problèmes de dépendances qui ont surgi lors de l'intégration avec Docker. De plus, nous avons rencontré des difficultés liées à l'incompatibilité entre Docker et Anaconda, ce qui a entraîné des dysfonctionnements lors des appels à Conda depuis Docker. Cependant, après des efforts persévérandants, nous avons finalement réussi à surmonter ces obstacles pour parvenir au déploiement réussi de notre application.

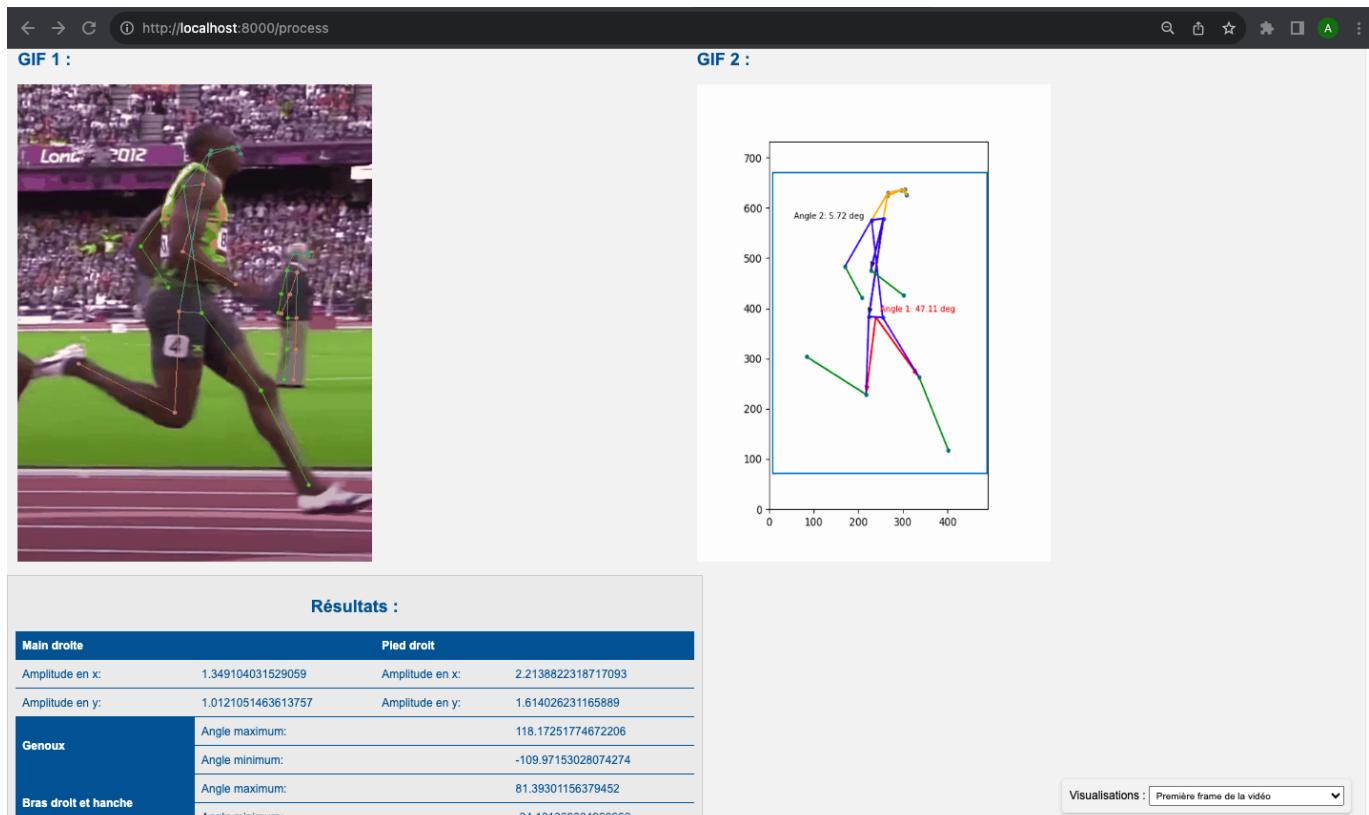


FIGURE 23 – Page de résultats

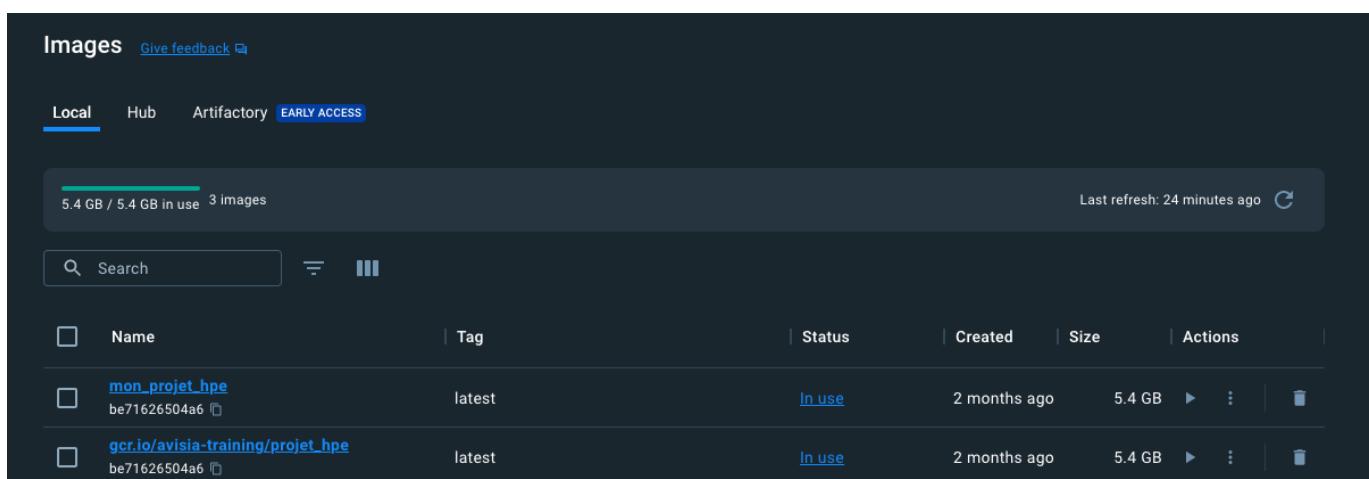


FIGURE 24 – Interface de docker avec l'image du projet hpe

14 Le projet LOL

14.1 Drafting

J'ai eu pour mission d'implémenter un algorithme de sélection de champions pour une équipe dans le jeu en ligne League of Legends (LoL). L'objectif de cet algorithme est d'assister les joueurs dans la phase de sélection des champions, appelée la "draft", en recommandant des champions en fonction de plusieurs critères tels que la synergie avec les champions alliés, le contre des champions ennemis, et le rôle du joueur dans l'équipe (Mid, Top, ADC, Support, Jungle).

L'algorithme commence par filtrer les champions éligibles en fonction du rôle du joueur et en excluant les champions déjà sélectionnés par l'équipe ou les adversaires. Ensuite, il calcule des scores de synergie et de contre pour les champions restants en se basant sur des matrices préalablement définies. Ces scores sont utilisés pour calculer une moyenne de performance globale pour chaque champion restant. L'algorithme choisit ensuite le champion avec la meilleure moyenne de performance parmi les champions éligibles.

Le code inclut également une fonction de sélection récursive qui permet de choisir plusieurs champions successivement en respectant l'alternance entre les joueurs de l'équipe et les joueurs adverses.

Chaque choix de champion est minutieusement effectué pour garantir qu'il représente la réponse optimale à la sélection précédente, en utilisant comme référence les matrices de synergie et de contre. L'algorithme, tout en restant simple, repose essentiellement sur les concepts étudiés en théorie des jeux au cours de cette année.

```

1 import numpy as np
2 import random as rd
3 role = ["Mid", "Top", "ADC", "Support", "Jungle"]
4
5 def pick_next_champion(ally_champions_chosen, enemy_champions_chosen, matrix_synergie,
6     matrix_counter, champions, player_role):
7     # Filter the champions for the given player's role
8     eligible_champions = [champion for champion in role_champions[player_role] if champion not
9         in ally_champions_chosen and champion not in enemy_champions_chosen]
10    champion_to_index = {champion: index for index, champion in enumerate(champions)}
11    eligible_champions_indices = [champion_to_index[champion] for champion in
12        eligible_champions]
13    print(eligible_champions_indices)
14
15    if len(ally_champions_chosen) == 0:
16        return np.random.choice(eligible_champions_indices)
17
18    ally_last_champion = ally_champions_chosen[-1]
19    enemy_last_champion = enemy_champions_chosen[-1]
20
21    # Calculer les scores de synergie pour les champions restants
22    synergie_scores = matrix_synergie[ally_last_champion, :]
23    for champ in ally_champions_chosen:
24        print(champ)
25        synergie_scores[champ] = 0 # Ignorer les champions alliés déjà choisis
26    for champ in enemy_champions_chosen:
27        print(champ)
28        synergie_scores[champ] = 0 # Ignorer les champions ennemis déjà choisis
29
30    # Calculer les scores de counter pour les champions ennemis
31    counter_scores = matrix_counter[enemy_last_champion, :]
32    for champ in ally_champions_chosen:
33        counter_scores[champ] = 100 # Ignorer les champions alliés déjà choisis
34    for champ in enemy_champions_chosen:
35        counter_scores[champ] = 100 # Ignorer les champions ennemis déjà choisis
36
37    # Calculer la moyenne counter/synergie pour chaque champion restant
38    average_scores = (100 - counter_scores) / (100 - synergie_scores)
```

```

36     # Trouver le champion avec la moyenne counter/synergie maximale parmi les éligibles
37     next_champion = eligible_champions_indices[np.argmax(average_scores[
38         eligible_champions_indices])]
39     return next_champion
40
41 def draft_algorithm(num_picks, ally_champions_chosen, matrix_synergie, matrix_counter,
42     champions, enemy_champions_chosen, role):
43     print()
44     print(num_picks)
45     # Determine the player's role based on the number of picks made so far
46     player_role = role[num_picks % len(role)]
47     print(player_role)
48     if num_picks == 5:
49         return ally_champions_chosen
50     else:
51         if num_picks % 2 == 0:
52             next_champion_chosen = pick_next_champion(ally_champions_chosen,
53                 enemy_champions_chosen[:num_picks], matrix_synergie, matrix_counter, champions, player_role
54             )
55         else:
56             next_champion_chosen = pick_next_champion(ally_champions_chosen,
57                 enemy_champions_chosen[:num_picks+1], matrix_synergie, matrix_counter, champions,
58                 player_role)
59             ally_champions_chosen.append(next_champion_chosen)
60             num_picks += 1
61     return draft_algorithm(num_picks, ally_champions_chosen, matrix_synergie,
62         matrix_counter, champions, enemy_champions_chosen, role)

```

14.2 Scouting

J'ai été chargé de lancer le projet de scouting avec l'aide de Lucas PRUTKI que je remercie aussi pour m'avoir éclairé lors de nos multiples échanges. Nous avons finalement pu construire un dossier contenant :

- 3 notebooks d'analyse
- plusieurs scripts python pour scraper les données des parties de LOL

Le script Python, "Creation_table_v2.py" est utilisé pour récupérer les données des 100 dernières parties d'un joueur à partir de son nom d'invocateur (summoner name) et le script "aggregation_dataset.py" a pour objectif de parcourir les fichiers pickle de données présents dans un dossier, de fusionner les ensembles de données et de supprimer les doublons. Ces scripts nous permettent de créer nos ensembles de données contenant les 100*N derniers matchs d'un joueur dont on connaît le nom d'invocateur.

Le dossier Scouting comporte aussi plusieurs éléments essentiels pour notre processus d'évaluation des performances des joueurs. Les notebooks d'analyse, tels que "analyse_v1.ipynb" et "analyse_v2.ipynb", sont conçus pour extraire des variables significatives afin de créer des méta-indicateurs, ou KPI, détaillant le style de jeu de chaque joueur. Ces KPI englobent divers aspects tels que les performances individuelles, la vision de jeu, le contrôle de la carte, la stratégie, l'adaptabilité avec différents champions, la communication et la coopération. Pour une représentation visuelle complète de ces KPI, nous utilisons des graphiques en forme de radar. Comme l'API de Riot permet d'avoir accès à énormément de données pendant la partie, nous avons essayé de trier les informations pour avoir un propos plus pertinent.

Cette procédure s'est avérée laborieuse et exigeait une expertise pointue du jeu, ainsi qu'une compréhension approfondie des concepts théorique de LOL, compétences que nous ne possédions pas nécessairement à l'origine.

Importation

```
In [27]: import os
import pandas as pd

# Chemin absolu du dossier courant
chemin_scouting = os.path.abspath('..')

print("Chemin du dossier parent : ", chemin_scouting)

chemin_data = chemin_scouting + "/datas/matches/equipes/GW/lol_df_Badlulu00.

data_lol = pd.read_pickle(chemin_data) # dataset

print(f"Le dataset a une taille de : {data_lol.shape}")

pd.set_option('display.max_columns', 163)
pd.set_option('display.max_rows', 5)

data_lol.head()
```

Chemin du dossier parent : /Users/atysp/Desktop/Avisia/projet_lol/scouting
Le dataset a une taille de : (100, 75)

	matchId	participants	platformId	q
0	EUW1_6528747599	KJLobfhi20s2V45keoOTQpX8CSmNGx15o6iM27wV2xUTk0...	EUW1	
1	EUW1_6528700594	KJLobfhi20s2V45keoOTQpX8CSmNGx15o6iM27wV2xUTk0...	EUW1	
2	EUW1_6528367649	KJLobfhi20s2V45keoOTQpX8CSmNGx15o6iM27wV2xUTk0...	EUW1	
3	EUW1_6528262584	ihr-O8AKXy2wTpa_RDzcTkGnd7qWyCuDzahwVIPpEw7iew...	EUW1	
4	EUW1_6527754806	KJLobfhi20s2V45keoOTQpX8CSmNGx15o6iM27wV2xUTk0...	EUW1	

```
In [28]: data_lol.dtypes
```

```
Out[28]: matchId          string
participants        string
...
ennemyTeamTotalLvl    Int64
ennemyTeamDamages     Int64
Length: 75, dtype: object
```

RADAR PLOT

Performances individuelles (Individual Performance)

Cette méta-variable peut être évaluée en utilisant des statistiques sur les performances pendant la phase de lane regroupant le gain de gold et d'expérience.

```
In [29]: import pandas as pd
import plotly.graph_objects as go

def calculate_means(df):
```

```

df = df[df["gameDuration"] > 0]
winRate = df["win"].mean()
goldPerMinute = df["goldPerMinute"].mean()
damagePerMinute = df["damagePerMinute"].mean()
csPerMinute = ((df["neutralMinionsKilled"] + df["totalMinionsKilled"]) /
kda = df["kda"].mean()
killParticipation = df["killParticipation"].mean()
return [winRate, goldPerMinute / 800, damagePerMinute / 1000, csPerMinute]

import os

# Chemin absolu du dossier courant
chemin_scouting = os.path.abspath('..')

print("Chemin du dossier parent :", chemin_scouting)

chemin_data_1 = chemin_scouting + "/datas/matches/equipes/GW/lol_df_Badlulu0
data_lol_1 = pd.read_pickle(chemin_data_1)
summoner_name_1 = data_lol_1["summonerName"][0]
data_lol_1 = data_lol_1[["win", "goldPerMinute", "damagePerMinute", "neutral
data_1 = calculate_means(data_lol_1)
data_1.append(data_1[0])

chemin_data_2 = chemin_scouting + "/datas/matches/equipes/GW/lol_df_zrh2.pkl
data_lol_2 = pd.read_pickle(chemin_data_2)
summoner_name_2 = data_lol_2["summonerName"][0]
data_lol_2 = data_lol_2[["win", "goldPerMinute", "damagePerMinute", "neutral
data_2 = calculate_means(data_lol_2)
data_2.append(data_2[0])

chemin_data_3 = chemin_scouting + "/datas/matches/equipes/GW/lol_df_Practice
data_lol_3 = pd.read_pickle(chemin_data_3)
summoner_name_3 = data_lol_3["summonerName"][0]
data_lol_3 = data_lol_3[["win", "goldPerMinute", "damagePerMinute", "neutral
data_3 = calculate_means(data_lol_3)
data_3.append(data_3[0])

# Ajouter les nouveaux dataframes
chemin_data_4 = chemin_scouting + "/datas/matches/equipes/GW/lol_df_GW_Villa
data_lol_4 = pd.read_pickle(chemin_data_4)
summoner_name_4 = data_lol_4["summonerName"][0]
data_lol_4 = data_lol_4[["win", "goldPerMinute", "damagePerMinute", "neutral
data_4 = calculate_means(data_lol_4)
data_4.append(data_4[0])

chemin_data_5 = chemin_scouting + "/datas/matches/equipes/GW/lol_df_ayekasia
data_lol_5 = pd.read_pickle(chemin_data_5)
summoner_name_5 = data_lol_5["summonerName"][0]
data_lol_5 = data_lol_5[["win", "goldPerMinute", "damagePerMinute", "neutral
data_5 = calculate_means(data_lol_5)
data_5.append(data_5[0])

categories = ['winRate', 'goldPerMinute / 800', 'damagePerMinute / 1000', '
categories.append(categories[0])

fig = go.Figure(
    data=[
        go.Scatterpolar(r=data_1, theta=categories, name=summoner_name_1),
        go.Scatterpolar(r=data_2, theta=categories, name=summoner_name_2),
        go.Scatterpolar(r=data_3, theta=categories, name=summoner_name_3),
        go.Scatterpolar(r=data_4, theta=categories, name=summoner_name_4),
    ],
)

```

```

        layout=go.Layout(
            title=go.layout.Title(text='Individual Performance'),
            showlegend=True
        )
    )

fig.show()

```

Chemin du dossier parent : /Users/atysp/Desktop/Avisia/projet_lol/scouting

Individual Performance



Vision et contrôle de la carte (Map Awareness)

Cette méta-variable peut être évaluée en utilisant des statistiques telles que le nombre de contrôleurs de vision placés, le nombre d'ennemis détectés par des contrôleurs de vision, le temps passé à la base pour acheter des objets, etc. Ces statistiques révèlent la capacité du joueur à comprendre et à contrôler la carte du jeu.

On pourrait simplement se baser sur le score de vision par minute pour cette caractéristique.

```

In [30]: import pandas as pd
import plotly.graph_objects as go

columns = ["visionScorePerMinute", "wardsPlaced", "controlWardsPlaced", "stealthWardsPlaced", "wardTakedowns", "wardsKilled", "wardsGuarded"]

# categories = [Vision score Per Minute, Ward Per Minute, Vision Ward Per Minute]
# visionScore
# wardsPlaced
# controlWardsPlaced
# stealthWardsPlaced
# wardTakedowns
# wardsKilled
# wardsGuarded

chemin_data_5 = chemin_scouting + "/datas/matches/equipes/GW/lol_df_ayekasia"
data_lol_5 = pd.read_pickle(chemin_data_5)
summoner_name_5 = data_lol_5["summonerName"][0]

data_lol = data_lol_5
summoner_name = summoner_name_5
data_lol = data_lol[data_lol["gameDuration"] > 0]
data = data_lol[columns].mean()

first_row = data.iloc[[0]]
data = pd.concat([data, first_row], ignore_index=True)

print(data)

```

```

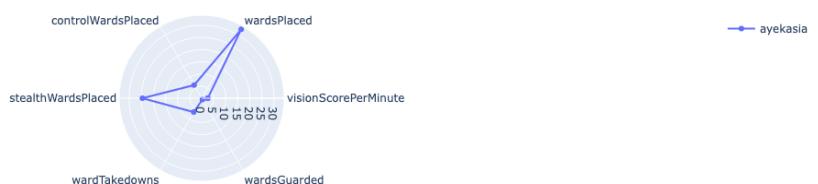
fig = go.Figure(
    data=[
        go.Scatterpolar(r=data, theta=columns, name=summoner_name),
    ],
    layout=go.Layout(
        title=go.layout.Title(text='Map Awarness'),
        showlegend=True
    )
)

fig.show()

0      2.527301
1      32.690000
...
6      2.527301
7      2.527301
Length: 8, dtype: float64

```

Map Awarness



Objectif et stratégie (Objective and Strategy)

Cette méta-variable peut être évaluée en utilisant des statistiques telles que le taux de participation aux objectifs (tours, dragons, barons, etc.). Ces statistiques montrent la capacité du joueur à prendre des décisions stratégiques et à coordonner son équipe.

On pourrait aussi rajouter le nombre de plaques éliminées ou le pourcentage de dommage infligé aux tours qui serait plus représentatif des compétences individuelles plutôt que collectives.

```

In [31]: import pandas as pd
import plotly.graph_objects as go

# - drake / herald / barron
# - turret / inhibitor / nexus
# 2 branches avec tourelles plates inib et baron drake etc
# dragonKills
# baronKills
# turretKills
# inhibitorKills

# dragonTakedowns          object
# baronTakedowns          object
# firstTowerAssist         object
# firstTowerKill           object
# turretKills              object
# turretTakedowns          object
# turretsLost               object
# inhibitorKills           object
# inhibitorTakedowns       object
# inhibitorsLost            object

```

```
# nexusKills          object
# nexusLost          object

# dragonTakedowns
# baronTakedowns
# inhibitorTakedowns
# turretTakedowns
# heraldTakedowns ??????
# plates ??

columns = [ "dragonTakedowns" , "baronTakedowns" , "turretTakedowns" , "inhibit
columns.append(columns[0])

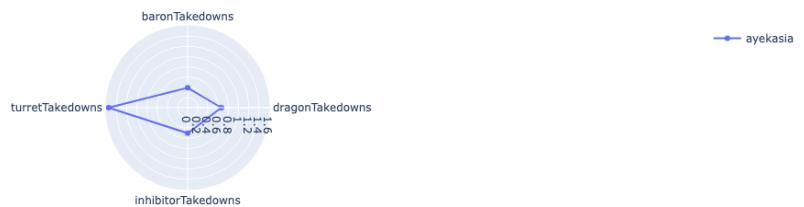
chemin_data_5 = chemin_scouting + "/datas/matches/equipes/GW/lol_df_ayekasia
data_lol_5 = pd.read_pickle(chemin_data_5)
summoner_name_5 = data_lol_5[ "summonerName" ][0]
data_lol = data_lol_5
summoner_name = summoner_name_5

data_lol = data_lol[data_lol[ "gameDuration" ]>0]
data = data_lol[columns].mean()

first_row = data.iloc[[0]]
data = pd.concat([data, first_row], ignore_index=True)

fig = go.Figure(
    data=[
        go.Scatterpolar(r=data, theta=columns, name=summoner_name),
    ],
    layout=go.Layout(
        title=go.layout.Title(text='Objective and Strategy'),
        showlegend=True
    )
)
fig.show()
```

Objective and Strategy



Adaptabilité (Adaptability)

Cette méta-variable peut être évaluée en utilisant des statistiques telles que la diversité des champions joués ainsi que le taux de victoire avec différents champions, etc. Ces statistiques indiquent la capacité du joueur à s'adapter à différentes situations de jeu.

On pourrait regarder le pourcentage de champions ayant un winrate > 50% parmi les champions joués les x derniers matchs. On pourrait aussi regarder le nombre de champions joués les x dernières semaines.

En ce qui concerne le notebook "analyse_player.ipynb", son objectif est d'établir une liste réduite de métriques, une dizaine en tout, qui reflètent le niveau de jeu d'un joueur au cours d'une partie. Pour ce faire, nous calculons un coefficient de corrélation pour chacune des 11 métriques par rapport à la victoire. Ensuite, nous standardisons ces métriques en les soustrayant de leur moyenne et en les divisant par leur écart-type. Les valeurs standardisées sont ensuite pondérées par les coefficients de corrélation correspondants, et la somme de ces valeurs est divisée par la somme des corrélations, fournissant ainsi un nombre représentatif de la performance relative du joueur dans la partie. Pour chaque partie, on a une note de la performance du joueur basé sur les indicateurs qu'on a sélectionné ce qui pourra aider dans le recrutement.

15 Conclusion

Pour conclure, ce stage a été une opportunité précieuse pour mettre en pratique de nombreuses notions que j'avais étudiées au cours de cette année. J'ai eu l'occasion d'affiner mes compétences en programmation Python et en analyse de données, tout en élargissant mes connaissances dans des domaines qui m'étaient moins familiers, tels que le web scraping et le développement web en HTML. Cette expérience m'a permis de développer un ensemble de compétences diversifié et de gagner en confiance dans ma capacité à aborder des projets multidisciplinaires et stimulants.

Liste des figures

1	Les 3 axes de travail chez Avisia	5
2	Logiciels principalement utilisés durant mon stage	5
3	Photographie mettant en valeur l'open space d'Avisia	6
4	Les différents modèles pour l'estimation de position humain	7
5	Exemple d'utilisation de l'opérateur de corrélation croisée	8
6	Exemple de "features matching"	9
7	Schéma de fonctionnement de la classification supervisée	10
8	Fonctionnement d'un R-CNN	11
9	Architecture du modèle VGG16	15
10	Schéma du modèle VGG16	16
11	Entrainement sur plusieurs epochs	16
12	Logo de MMPose	18
13	Exemple d'inférence avec MMPose	18
14	Exemple d'estimation de position sur Brice Dulin (joueur du Stade Rochelais)	20
15	Tableau récapitulatif des algorithmes de HPE	21
16	illustration du mouvement	23
17	illustration du mouvement 2	23
18	Évolution de la position de la main droite, du pied droit et du centre du corps	24
19	Évolution de la position relative de la main droite et du pied droit par rapport au centre du corps	24
20	Évolution de l'angle du bras droit	25
21	Évolution de l'angle du genou droit	25
22	Page d'accueil	26
23	Page de résultats	27
24	Interface de docker avec l'image du projet hpe	27

Annexe A Documentation et liens

Afin de mener à bien mes projets, je me suis appuyé sur des sites de documentation et projets existants. En voici les deux principaux :

- **Documentation de MMPOse** (<https://mmpose.readthedocs.io/en/latest>) : documentation principale de MMPOse, qui explique les différents composants sous-jacents et les technologies utilisées. C'est la documentation principale que j'ai utilisé pour prendre en main le projet.
- **Documentation de l'API Riot** (<https://developer.riotgames.com/apis>) : documentation principale de l'API de Riot, qui explique comment accéder aux données des parties de League of Legends