



Université ABBES LAGHROUR Khenchela
Faculté des Sciences et de la Technologie
Département de Génie Electrique
جامعة عباس لغرور خنشلة
كلية العلوم والتكنولوجيا
قسم الهندسة الكهربائية



N° Série :

Mémoire de fin d'étude

Pour l'obtention du diplôme de Master

Filière : Télécommunications

Spécialité : Systèmes des Télécommunications

THEME

**Conception et réalisation d'un système
intelligent de télésurveillance des
personnes âgées basé sur la carte Nvidia
Jetson Nano et une application Android**

Réalisé par : LATRECHE Manal

Soutenu le : 14/06/2026

Dr. DOUAK Fouzi

Dr. MALLEM Ali

Dr. KHEZZAR. Zaki Aissam

Encadreur

Président

Examineur

Université Abbes Laghrour-Khenchela

Université Abbes Laghrour-Khenchela

Université Abbes Laghrour-Khenchela

Promotion 2025/2026

À mes très chers parents,

À mon cher père, pour son soutien inconditionnel, ses sacrifices constants et ses précieux conseils qui ont éclairé mon chemin. Que Dieu lui accorde santé et longue vie. À ma tendre mère, source de tendresse, d'amour et de prières bienveillantes. Ses encouragements m'ont donné la force de persévérer dans les moments les plus difficiles. Aucune dédicace ne saurait exprimer mon amour et ma reconnaissance. Que Dieu la protège.

À mes chers frères et sœurs,

Pour leur présence constante, leur affection, leur soutien moral et les moments de partage qui m'ont toujours redonné de l'énergie. Je leur souhaite un avenir radieux et plein de succès.

À mes fidèles amis,

Ceux qui ont partagé avec moi les bancs de l'université, les moments de stress et les joies de la réussite. À mes compagnons de route pour leur fraternité et leur soutien indéfectible.

À tous ceux qui me sont chers,

Et à tous ceux qui portent dans leur cœur un amour pour la science et le savoir.

Manal

Remerciements

Avant tout, je remercie Dieu le Tout-Puissant de m'avoir accordé la santé, la force, la patience et le courage nécessaires pour mener à bien ce modeste travail d'études.

Je tiens à exprimer ma profonde gratitude, ma reconnaissance la plus sincère ainsi que ma plus haute considération à mon honorable encadrant, **Monsieur Fouzi Douak**, pour avoir accepté d'assurer la direction de ce mémoire avec tant de bienveillance, de patience et de rigueur scientifique.

Je le remercie chaleureusement pour la confiance qu'il m'a accordée dès le premier jour, ainsi que pour sa vision éclairée, ses conseils hautement judicieux et ses orientations précieuses, qui ont constitué le fil conducteur de ce mémoire. Malgré ses nombreuses obligations académiques, il a toujours su se rendre disponible et faire preuve d'une écoute attentive à chaque étape de la rédaction. Travailler sous sa direction a été pour moi un immense honneur et une expérience académique particulièrement enrichissante. Qu'il trouve dans ce témoignage l'expression de mon profond respect et de ma gratitude éternelle.

J'adresse également mes sincères remerciements au Laboratoire des Technologies Innovantes, de la Maîtrise de l'Énergie et des Énergies Renouvelables (LTIMER) de l'Université Abbes Laghrour Khenchela, dirigé par Monsieur le Professeur Yassine Beddiaf.

Mes remerciements les plus respectueux s'adressent également aux honorables membres du jury qui ont bien voulu accepter d'examiner ce travail, de l'évaluer et d'enrichir par leurs remarques constructives et leurs critiques avisées.

Enfin, je tiens à remercier chaleureusement tous les enseignants qui ont contribué à ma formation tout au long de mon parcours universitaire, ainsi que tous mes proches, mes amis et toutes les personnes qui ont contribué, de près ou de loin, à la réalisation de ce projet.

Résumé

Ce mémoire présente la conception et le développement d'un système intelligent de télésurveillance destiné aux personnes âgées, basé sur la carte Nvidia Jetson Nano et une application Android dédiée. L'objectif principal est de détecter les chutes en temps réel grâce à l'intégration de techniques d'intelligence artificielle, notamment le modèle de détection d'objets YOLOv5, capable d'analyser les flux vidéo avec rapidité et précision.

Le système proposé combine le traitement embarqué des données et la surveillance à distance afin d'assurer une détection fiable des situations à risque tout en préservant la confidentialité des données. Une application mobile développée sous Android permet la réception instantanée des alertes ainsi que la visualisation des flux vidéo en temps réel. Elle exploite également les fonctionnalités de communication des smartphones, notamment les réseaux GSM et les communications Wi-Fi, afin de garantir la transmission efficace des alertes et des informations de surveillance.

Les résultats expérimentaux obtenus démontrent la faisabilité d'une solution embarquée performante, capable d'assurer une détection fiable des chutes avec une interaction fluide avec l'utilisateur via l'application mobile. Ce travail contribue ainsi au domaine de la télésurveillance médicale en proposant une solution hybride combinant l'intelligence artificielle embarquée (Edge AI) et les technologies mobiles. La solution développée constitue une base prometteuse pour l'amélioration de la sécurité, de l'autonomie et de la qualité de vie des personnes âgées vivant seules à domicile.

Mots clés : Télésurveillance, Jetson Nano, YOLOv5, Deep Learning, Application Android, Détection de chutes.

Abstract

This thesis presents the design and development of an intelligent remote health monitoring system for elderly people, based on the Nvidia Jetson Nano platform and a dedicated Android application. The main objective is to detect falls in real time through the integration of artificial intelligence techniques, particularly the YOLOv5 object detection model, which is capable of analyzing video streams quickly and accurately.

The proposed system combines edge data processing and remote monitoring to ensure reliable detection of risky situations while preserving data privacy. An Android mobile application enables instant alert notifications and real-time video streaming. It also leverages smartphone communication capabilities, including GSM networks and Wi-Fi connectivity, to ensure efficient transmission of alerts and monitoring information.

The experimental results demonstrate the feasibility of a high-performance embedded solution capable of providing reliable fall detection with low latency while ensuring smooth user interaction through the mobile application. This work contributes to the field of medical monitoring by proposing a hybrid solution that combines Edge Artificial Intelligence (Edge AI) with mobile technologies. The developed system represents a promising foundation for improving the safety, autonomy, and quality of life of elderly people living independently at home.

Keywords: Monitoring, Jetson Nano, YOLOv5, Deep Learning, Android Application, Fall Detection.

ملخص

تُقدّم هذه المذكرة تصميم وتطوير نظام ذكي للمراقبة الصحية عن بُعد موجه لكبار السن، يعتمد على منصة Nvidia Jetson Nano وتطبيق مخصص لنظام Android. ويتمثل الهدف الرئيسي في الكشف عن حالات السقوط في الزمن الفعلي من خلال دمج تقنيات الذكاء الاصطناعي، ولا سيما نموذج الكشف YOLOv5 القادر على تحليل تدفقات الفيديو بسرعة ودقة عاليتين.

يجمع النظام المقترح بين المعالجة الطرفية للبيانات (Edge Computing) والمراقبة عن بُعد لضمان الكشف الموثوق عن الحالات الخطرة مع الحفاظ على خصوصية البيانات. كما يتيح التطبيق المحمول المطور على نظام Android استقبال التنبيهات بشكل فوري وعرض بث الفيديو في الزمن الفعلي. ويستفيد كذلك من إمكانيات الاتصال المتوفرة في الهواتف الذكية، بما في ذلك شبكات GSM واتصالات Wi-Fi، لضمان النقل الفعال للتنبيهات ومعلومات المراقبة.

أظهرت النتائج التجريبية إمكانية تنفيذ حل مدمج عالي الأداء قادر على توفير كشف موثوق لحالات السقوط بزمن استجابة منخفض، مع ضمان تفاعل سلس للمستخدم عبر التطبيق المحمول. ويساهم هذا العمل في مجال المراقبة الطبية عن بُعد من خلال تقديم حل هجين يجمع بين الذكاء الاصطناعي الطرفي (Edge AI) والتقنيات المحمولة. ويمثل النظام المطور أساساً واعدًا لتحسين سلامة كبار السن واستقلاليتهم وجودة حياتهم، خاصةً لأولئك الذين يعيشون بمفردهم في منازلهم.

الكلمات المفتاحية: المراقبة عن بُعد، Jetson Nano، YOLOv5، التعلّم العميق، تطبيق أندرويد، رصد السقوط.

Liste des tableaux

Tableau I. 1. Les facteurs de chute chez la personne âgée [16].	15
Tableau I. 2. Fonction d'activation classique.....	29
Tableau I. 3. Fonctions d'activations rectifiées.	29
Tableau I. 4. Fonctions d'activation pour les sorties.	30
 Tableau II. 1. Spécifications techniques de la carte Nvidia Jetson Nano.	39
 Tableau III. 1. Évolution des différentes versions du modèle YOLO [8].	66
Tableau III. 2. Matrice de confusion.	73
Tableau III. 3. Performances des modèles YOLOv5 sur les ensembles d'entraînement, de validation et de test, basées sur les différentes métriques d'évaluation : précision (P), rappel (R), mAP50 et mAP50-95.	88

Liste des figures

Figure I. 1. Présentation des 3 états de la fragilité chez le sujet âgé [13].	13
Figure I. 2. Classification générale des causes de chutes [15].	14
Figure I. 3. Les différentes phases de la chute [17].	16
Figure I. 4. Test « Get up and Go » [18].	17
Figure I. 5. Surveillance des personnes âgées [17].	18
Figure I. 6. Catégories de capteurs pour la détection des activités de la personne âgée ou de sa chute [18].	19
Figure I. 7. Système de détection des chutes basé sur les capteurs portables [22].	20
Figure I. 8. Système de détection des chutes basé sur la vision [22].	21
Figure I. 9. Intelligence Artificielle [27].	23
Figure I. 10. Apprentissage supervisé.	24
Figure I. 11. Apprentissage non supervisé.	24
Figure I. 12. Apprentissage semi-supervisé.	25
Figure I. 13. Apprentissage par renforcement [29].	25
Figure I. 14. Techniques d'apprentissage en profondeur avec leurs différents types [30].	26
Figure I. 15. Comparaison entre un neurone biologique et un Neurone Artificiel [29].	26
Figure I. 16. Architecture typique du CNN [30].	28
 Figure II. 1. Vision pour la conception d'un système permettant le maintien à domicile des personnes âgées.	 36
Figure II. 2. Carte Nvidia Jetson Nano.	37
Figure II. 3. (a) Nvidia Jetson Nano A02, (b) Nvidia Jetson NanoB01.	38
Figure II. 4. Schéma descriptif des différents composants de Jetson Nano de Nvidia [5].	40
Figure II. 5. Caméra Havit (Hv-N 5080).	41
Figure II. 6. Caméra Imou (IPC-G26EP-IMOU).	42
Figure II. 7. Les différentes technologies sans fil.	43
Figure II. 8. Connexion sans fil.	45
Figure II. 9. Le modèle TCP/IP.	45
Figure II. 10. Encapsulation et la décapsulation du modèle TCP/IP.	47

Figure II. 11. Communication en temps réel entre le serveur (Jetson Nano) et le client (smartphone) via WebSocket.	48
Figure II. 12. Architecture du réseau GSM [6].	50
Figure II. 13. Logo Python [16].	52
Figure II. 14. Logo Pytorch [18].	53
Figure II. 15. Logo OpenCV [17].	53
Figure II. 16. Logo Google colab [18].	54
Figure II. 17. Environnement de Google Colab [19].	54
Figure II. 18. Android studio [20].	55
Figure II. 19. Logo de Kotlin	56
Figure III. 1. Réseaux de neurones convolutifs basés sur les régions (R-CNN) [2].	64
Figure III. 2. Fast R-CNN [3].	64
Figure III. 3. Faster R-CNN [4].	64
Figure III. 4. Principe de l'algorithme de détection d'objets «YOLO» [6].	65
Figure III. 5. Architecture du modèle YOLOv5 [10].	68
Figure III. 6. PANet. (a) FPN. (b) voie ascendante. (c) mise en commun adaptative des fonctionnalités. (d) branche de détection. (e) couche de fusion.	69
Figure III. 7. Structure du SPPF / Régression de la boîte englobante.	69
Figure III. 8. Différentes modèles proposées par YOLOv5 [15].	71
Figure III. 9. Version initiale de YOLOv5 montre la promesse d'une détection d'objets de pointe [15].	72
Figure III. 10. Connexion du notebook Google Colab à Google Drive.	80
Figure III. 11. Structure des dossiers du jeu de données.	80
Figure III. 12. (a) Répartition des classes, (b) superposition des boîtes englobantes, (c) distribution des centres des objets (x, y) et (d) distribution des dimensions des boîtes (width, height).	81
Figure III. 13. Clonage et installation du dépôt YOLOv5.	82
Figure III. 14. Mise en œuvre du processus d'entraînement.	82
Figure III. 15. Progression de l'entraînement sur 100 époques.	83
Figure III. 16. Processus d'entraînement enregistré dans le dossier «Fall_Detection».	83
Figure III. 17. Mise en œuvre du processus de détection.	84

Figure III. 18. Performances des modèles YOLOv5, basées sur les différentes métriques d'évaluation : la perte de classification (a–b), la perte de localisation des boîtes (c–d), ainsi que la perte d'objectivité (e–f).....	85
Figure III. 19. Performances des modèles YOLOv5 sur l'ensemble de validation, basées sur les différentes métriques d'évaluation : (a) précision, (b) rappel, (c) mAP50, (d) mAP50-95.	87
Figure III. 20. Application du modèle YOLOv5n sur des images de test.....	89
Figure III. 21. Application du modèle YOLOv5s sur des images de test.	90
Figure III. 22. Application du modèle YOLOv5m sur des images de test.	91
Figure III. 23. Application du modèle YOLOv5l sur des images de test.	92
Figure III. 24. Application du modèle YOLOv5x sur des images de test.....	93
Figure III. 25. Matrice de confusion du modèle YOLOv5s après 100 époques d'entraînement.	94
Figure III. 26. Courbe de précision du modèle YOLOv5s après 100 époques d'entraînement.	94
Figure III. 27. Courbe de rappel du modèle YOLOv5s après 100 époques d'entraînement.	95
Figure III. 28. Courbe F1 du modèle YOLOv5s après 100 époques d'entraînement.	95
 Figure IV. 1. Système proposé de surveillance en temps réel.....	 101
Figure IV. 2. Architecture globale du système de surveillance et de détection de chute.	102
Figure IV. 3. Chaîne de traitement des données de système de détection la chute.	104
Figure IV. 4. Configuration de la carte Nvidia Jetson Nano.	106
Figure IV. 5. Utilisation d'une clé USB avec la Jetson.	106
Figure IV. 6. Application du terminal.	108
Figure IV. 7. Schéma du serveur de diffusion et communication.	109
Figure IV. 8. Logo de l'application «AI Safety Monitor».....	110
Figure IV. 9. Interface de l'application «AI Safety Monitor» (a) Connexion en langue arabe et (b) Connexion en langue anglais.	110
Figure IV. 10. Interface et saisie des données : (a) en arabe et (b) en anglais.	111
Figure IV. 11. Interface du tableau de bord de l'application mobile: version en arabe (à gauche) et version en anglais (à droite).	112
Figure IV. 12. Les champs de «Hardware Link» : version en arabe (à gauche) et version en anglais (à droite).	113

Figure IV. 13. Détection de chute: version en arabe (à gauche) et version en anglais (à droite).	113
Figure IV. 14. Écran de gestion des contacts d'urgence de l'application : interface en arabe (à gauche); interface en anglais (à droite).....	114
Figure IV. 15. Intégration des contacts d'urgence dans l'application : a) en arabe et b) en anglais.....	115
Figure IV. 16. Les champs de Emergency Times: a) en arabe et b) en anglais.	115
Figure IV. 17. Les informations personnelles de personne à détecter.....	116
Figure IV. 18. Exemple de remplir les champs des informations personnelles de personne chuté : a) en arabe et b) en anglais.....	117
Figure IV. 19. Prototype expérimental du système de détection de chutes proposé : (a) vue de dessus ; (b) vue de face.....	118
Figure IV. 20. Connexion Bureau à distance vers la carte Jetson Nano.	119
Figure IV. 21. (a) Interface d'authentification de la carte Jetson Nano, (b) fenêtre apparaît, indiquant l'accès à la carte Jetson Nvidia Nano.	120
Figure IV. 22. Répertoire du projet Fall_Detection_System.....	121
Figure IV. 23. Contenu de dossier Fall_Detection_System.	121
Figure IV. 24. Code Python créé pour la détection de chute.....	122
Figure IV. 25. Exécution du système avec caméra IP.....	123
Figure IV. 26. Affichage de l'alerte de détection de chute d'une personne âgée à domicile sur l'application AI Safety Monitor.	124
Figure IV. 27. Détection d'une activité normale (non-Fall) d'une personne âgée à domicile par l'application AI Safety Monitor.	125

Liste des symboles

- AAE** : Adversarial Autoencoder (Autoencodeur antagoniste).
- ADAM** : Adaptive Moment Estimation.
- AlexNet** : AlexNet Architecture (Architecture AlexNet).
- AP** : Average Precision.
- Attention-Net** : Attention Network.
- BiFPN** : Bi-directional Feature Pyramid Network.
- BSS** : Base Sub-System.
- CBS** : Convolution + Batch Normalization + SiLU (activation).
- CMOS** : Complementary Metal-Oxide Semiconductor .
- CNN** : Convolutional Neural Network (Réseau de neurones convolutifs).
- CPU** : Central Processing Unit.
- CRT** : Cathode Ray Tube.
- CSI** : Camera Serial Interface.
- CSP** : Cross Stage Partial.
- CUDA** : Compute Unified Device Architecture.
- cuDNN** : CUDA Deep Neural Network library.
- Darknet53** : Réseau de neurones convolutionnel composé de 53 couches.
- DCGAN** : Deep Convolutional Generative Adversarial Network (Réseau antagoniste génératif convolutionnel profond).
- Deep RNN** : Deep Recurrent Neural Network (Réseau de neurones récurrents profond).
- DL** : L'apprentissage profond (Deep Learning).
- DNN** : Deep Neural Network.
- DNS** : Domain Name System.
- DOD** : Departement Of Defense
- EfficientDet** : Scalable and Efficient Object Detection.
- Exponential Linear Unit (ELU)** : Exponential Linear Unit (Unité Linéaire Exponentielle).
- FC** : **Fully Connected** (Couche entièrement connectée).
- FDMA** : Frequency Division Multiple Access.
- fps** : frames per second.
- FTP** : File Transfer Protocol.

Full HD : Full High Definition.

GAN : Generative Adversarial Network (Réseau antagoniste génératif).

G-CNN : Group Equivariant Convolutional Neural Network.

GoogleNet : Réseau neuronal convolutif GoogleNet.

GPIO : General Purpose Input/Output (désigne des entrées/sorties).

GPRS : General Packet Radio Service.

GPU : Graphics Processing Unit (Le GPU est le processeur graphique chargé des calculs liés à l’affichage et au traitement d’images).

GRU : Gated Recurrent Unit (Unité récurrente à portes).

GSM : Global System for Mobile Communication.

HDMI : High Definition Multimedia Interface.

HPST : Hôpital, Patients, Santé et Territoires.

HTML : HyperText Markup Language.

HTTP : HyperText Transfer Protocol.

IA : L’intelligence artificielle.

IMOU : Marque commerciale spécialisée dans les caméras de sécurité intelligentes (pas une abréviation technique, mais un nom de marque).

INT8 : Integer 8-bit.

IoT : Internet of Things (Objets connectés et périphériques intelligents).

IoU : L’intersection par rapport à l’union.

IP : Internet Protocol.

JetPack : Kit de développement NVIDIA JetPack.

Jetson Nano A02 : La première édition de Nvidia.

Jetson Nano B01 : Révision améliorée de Nvidia.

JSON : JavaScript Object Notation.

LCD : Liquid Crystal Display.

Le2i : Laboratoire d’Électronique, Informatique et Image

Leaky ReLU : **Leaky Rectified Linear Unit** (Unité Linéaire Rectifiée Fuyante).

LOOS : Perte.

LPDDR4 : Low Power Double Data Rate 4.

LSTM : Long Short-Term Memory (Mémoire à long court terme).

lux : L’unité de mesure de l’éclairement lumineux.

Mac OS : Macintosh Operating System.

mAP : Mean Average Precision.

mAP@0.5 : moyenne de la précision moyenne calculée avec un seuil IoU de 0.5.

mAP@0.5:0.95 : moyenne des AP calculée sur plusieurs seuils IoU (de 0.5 à 0.95).

MIPI CSI : Mobile Industry Processor Interface – Camera Serial Interface.

MIPI-CSI-(x2) : Mobile Industry Processor Interface.- Camera Serial Interface-deux connecteurs CSI disponible.

MJPEG : Motion Joint Photographic Experts Group.

ML : L'apprentissage automatique (Machine Learning).

MP : Megapixels

MS : Mobile Station.

NPE : Null-Pointer-Exception.

NSS : Network Sub-System.

OMS : Organisation mondiale de la Santé.

ONU : Organisation des Nations Unies.

OpenCV : Open Source Computer Vision Library.

OSS : Opération Sub-System.

P : Precision.

PAN : Path Aggregation Network.

PANet : Path Aggregation Network.

Parametric ReLU (PReLU) : Parametric Rectified Linear Unit (Unité Linéaire Rectifiée Paramétrique).

Pascal VOC : Pattern Analysis, Statistical Modelling and Computational Learning-Visual Object Classes.

POOL : Pooling (une technique de sous-échantillonnage utilisé dans le CNN).

QoS : La **qualité de service**.

R : Recall.

R-CNN : Region-based Convolutional Neural Network.

ReLU fuyante : Leaky Rectified Linear Unit.

ReLU : Rectified Linear Unit (Unité Linéaire Rectifiée).

ResNet : Residual Neural Network (Réseau neuronal résiduel).

RGB : Rouge – Vert – Bleu.

RJ45 : Registered Jack 45 (un connecteur réseau standard utilisé pour les câbles Ethernet).

RNIS : Réseau Numérique à Intégration de Services.

RNN : Recurrent Neural Network (Réseau de neurones récurrents).

RPN : Réseau de propositions de régions.

RTCP : Réseau Téléphonique Commuté Public.

S : Specifity.

SD : Secure Digital.

SDK : Software Development Kit.

SGD : Stochastic Gradient Descent.

SMS : Short Message Service (Service de messages courts).

Softmax : Soft Maximum Function (fonction maximum adoucie).

SPPF : Spatial Pyramid Pooling - Fast.

SSD : Single Shot Detector.

StyleGAN : Style-based Generative Adversarial Network (Réseau antagoniste génératif basé sur le style).

Tanh: Fonction d'activation hyperbolique.

TCP : Transmission Control Protocol.

TDMA : Time Division Multiple Access.

TensorRT : Tensor Runtime (NVIDIA TensorRT).

TPH : Téléphone mobile (portable).

TPU : Tensor Processing Unit.

Train : training dataset.

Ubuntu : Distribution Linux.

UDP : User Datagram Protocol.

UE : Union européenne.

UMTS : Universal Mobile Telecommunications System.

USB : Universal Serial Bus.

USB-C : Universal Serial Bus Type-C.

VAE : Variational Autoencoder (Autoencodeur variationnel).

Valid : validation dataset.

VGA : Video Graphics Array.

VGGNet : Visual Geometry Group Network (Réseau du groupe de géométrie visuelle).

Weights : Model weights (poids du modèle).

WGAN : Wasserstein Generative Adversarial Network (Réseau antagoniste génératif de Wasserstein).

WHO : World Heath Organization.

WLAN : Wireless Local Area Network.

WMAN : Wireless Metropolitan Area Network.

WPAN : Wireless Personal Area Network.

WS : le protocole WebSocket.

WWAN : Wireless Wide Area Network.

XML : eXtensible Markup Language (Langage de balisage extensible).

YAML = YAMl Ain't Markup Language.

YOLO : You Only Look Once.

YOLOv5 : You Only Look Once, version 5.

YOLOv5l : You Only Look Once version 5 Large.

YOLOv5m : You Only Look Once version 5 Medium.

YOLOv5n : You Only Look Once version 5 Nano.

YOLOv5s : You Only Look Once version 5 Small.

YOLOv5x : You Only Look Once version 5 Extra-Large.

Table des matières

Introduction Générale.....	1
Références citées dans l'introduction générale	5
 Chapitre I	
 État de l'art sur la télésurveillance et la détection de chute	
I.1 Introduction	8
I.2 Télémédecine et télésurveillance des patients.....	8
I.2.1 Organisation mondiale de la Santé (OMS)	8
I.2.2 Télémédecine	9
I.2.3 Télésurveillance médicale	10
I.2.3.1 Différents types de télésurveillance médicale	10
I.2.3.2 Principe de la télésurveillance médicale	11
I.2.3.3 Enjeux de la télésurveillance médicale	11
I.2.3.4 Domaines d'application de télésurveillance.....	12
I.3 Application médicale chez les personnes âgées	12
I.3.1 Types de chute.....	15
I.3.2 Caractéristiques de la chute.....	16
I.3.3 Système de télésurveillance pour le maintien à domicile des personnes âgées	17
I.4 Méthodes de détection de chute	18
I.4.1 Capteurs portatifs	19
I.4.2 Capteurs ambiants	20
I.4.3 Approches basées sur la vision	21
I.5 Méthodes d'intelligence artificielle.....	23
I.5.1 Apprentissage automatique	24
I.5.2 Apprentissage profond	25
I.5.2.1 Réseaux de neurones artificiels	26
I.6 Réseaux de neurones convolutif	27
I.6.1 Fonctions d'activation	28
I.7 Conclusion.....	30
I.8 Références citées dans le Chapitre I.....	31

Chapitre II

Plateforme matérielle, logicielle et technologies de communication

II.1 Introduction.....	35
II.2 Choix du matériel.....	35
II.3 Plateforme matérielle	37
II.3.1 Carte Nvidia Jeston Nano	37
II.3.1.1 Présentation de la carte Nvidia Jeston Nano.....	37
II.3.1.2 Caractéristiques principales de la carte Nvidia Jeston Nano	38
II.3.1.3 Applications de Nvidia Jetson Nano.....	40
II.3.2 Caméra	41
II.4 Technologies de communication sans fil.....	43
II.4.1 Réseau local sans fil (WLAN) basé sur la technologie Wi-Fi.....	44
II.4.2 Protocoles de communication TCP/IP	45
II.4.3 WebSocket	47
II.5 Technologie GSM.....	48
II.5.1 Réseau GSM	49
II.5.1.1 Architecture du réseau GSM.....	49
II.5.1.2 Services offerts par un réseau GSM.....	50
II.5.1.3 Principe de fonctionnement du GSM.....	51
II.6 Plateforme logicielle	52
II.6.1 Python	52
II.6.2 Pytorch	52
II.6.3 OpenCV	53
II.6.4 Google Colab	53
II.6.5 Android Studio.....	55
II.6.6 Kotlin	56
II.7 Conclusion	57
II.8 Références citées dans le Chapitre II	57

Chapitre III

Méthodes d'intelligence artificielle utilisées

III.1 Introduction	61
III.2 Modèles de la détection d'objet.....	61
III.2.1 Modèles modernes de détection basés sur les réseaux neuronaux convolutifs (CNN)	

.....	62
III.2.1.1 Algorithmes de détection d'objet par CNN	63
III.2.2 YOLO (You Only Look Once).....	65
III.2.2.1 Evolution de YOLO.....	66
III.2.2.2 Architecture de YOLOv5	68
III.2.2.3 Différentes parties de YOLOv5.....	68
III.2.2.4 Modèles de YOLOv5.....	71
III.2.2.5 Entraînement.....	72
III.2.2.6 Métriques d'évaluation	73
III.3 Système intelligent pour la télésurveillance des personnes âgées	77
III.3.1 Base de données	78
III.3.2 Entraînement sur YOLOv5.....	79
III.3.3 Résultats et discussions	84
III.4 Conclusion.....	96
III.5 Références citées dans le Chapitre III	96

Chapitre IV

Conception et réalisation du système proposé

IV.1 Introduction	100
IV.2 Architecture globale du système	100
IV.2.1 Chaîne de traitement de données.....	104
IV.3 Système de détection sur Jetson	105
IV.3.1 Configuration de la carte Nvidia	105
IV.3.2 Principaux étapes de la configuration de la carte Nvidia	107
IV.3.3 Terminal	108
IV.3.4 Serveur de diffusion et communication.....	108
IV.4 Développement de l'Application Android	109
IV.4.1 Connexion à l'application	110
IV.4.2 Remplir les champs	111
IV.4.3 Accéder à l'application et ouvrir l'interface du tableau de bord.....	111
IV.5 Validation expérimentale	117
IV.6 Résultats	123
IV.7 Conclusion.....	126
Conclusion Générale	129

Introduction générale

Introduction générale

La télésurveillance médicale est devenue un axe stratégique dans le domaine de la santé numérique, en réponse aux défis posés par le vieillissement de la population et la prévalence croissante des maladies chroniques. Parmi les problématiques figure l'augmentation du nombre de chutes chez les personnes âgées, qui constitue une cause majeure de complications physiques, psychologiques et économiques.

Selon les statistiques de l'Organisation mondiale de la Santé (OMS), entre 28% et 35% des personnes âgées de plus de 60 ans sont victimes d'au moins une chute chaque année [1]. Cette proportion augmente significativement avec l'âge, atteignant entre 32% et 42% chez les personnes de plus de 70 ans [2]. Les chutes sont responsables d'environ 90% des fractures de la hanche et du poignet, ainsi que de 60% des traumatismes crâniens observés chez les personnes âgées [3].

Au-delà des blessures, les chutes peuvent entraîner une situation de « long séjour au sol », où la personne reste immobilisée sans assistance. Cette situation peut provoquer des complications graves telles que l'hypothermie, la déshydratation, des troubles cardiovasculaires, voire le décès. Elle peut également engendrer une peur de tomber, réduisant l'autonomie et la qualité de vie des personnes âgées [4].

Dans ce contexte, le développement de systèmes automatiques de détection de chute fiables, précis et réactifs apparaît comme une nécessité. Ces systèmes permettent d'assurer une assistance rapide en cas d'accident, en particulier lorsque la personne reste immobilisée au sol pendant une période prolongée, contribuant ainsi à réduire les complications médicales et à améliorer la qualité de vie des personnes âgées [5]. Les technologies intelligentes offrent ainsi une solution prometteuse pour renforcer la sécurité des personnes âgées, favoriser leur maintien à domicile et améliorer leur prise en charge.

Chez les personnes âgées vivant seules, la détection des chutes représente un enjeu critique. Les chutes figurent parmi les principales causes de perte d'autonomie et de mortalité, et leur prise en charge rapide est indispensable pour limiter les complications [6]. Les systèmes de télésurveillance permettent ainsi une intervention immédiate en cas d'urgence, tout en réduisant les coûts d'hospitalisation et en améliorant la qualité de vie des patients ainsi que la tranquillité de leurs proches [6], [7].

Les techniques contemporaines de détection automatique des chutes peuvent être regroupées en trois grandes catégories : les systèmes basés sur les capteurs portatifs, les systèmes basés sur les capteurs ambiants et les systèmes de vision.

La première catégorie regroupe les systèmes utilisant des capteurs portatifs, tels que les accéléromètres et les gyroscopes, intégrés dans des dispositifs portés sur le corps. Ces systèmes exploitent les signaux cinématiques générés lors des mouvements de l'utilisateur afin d'identifier les événements de chute. Bien qu'ils présentent plusieurs avantages, notamment en matière de respect de la vie privée et de simplicité de traitement, ils nécessitent le port permanent d'un dispositif, ce qui peut être perçu comme contraignant par certaines personnes âgées et limiter leur acceptabilité à long terme [8].

La deuxième catégorie repose sur l'utilisation de capteurs ambiants installés dans l'environnement de vie de la personne âgée, que ce soit à domicile ou dans des établissements spécialisés. Les données collectées sont transmises à une unité de traitement chargée d'analyser les événements et de déclencher des alertes en cas de situation anormale. Parmi les capteurs les plus couramment utilisés figurent les capteurs de pression ou de vibration intégrés au sol, les capteurs infrarouges fixés aux murs ou au plafond, ainsi que les capteurs acoustiques capables de détecter les bruits associés à une chute [9].

La troisième catégorie est constituée des systèmes de vision, qui reposent sur l'utilisation de caméras pour surveiller les déplacements des individus dans leur environnement. Grâce aux avancées récentes en vision par ordinateur et en intelligence artificielle, ces systèmes permettent d'analyser automatiquement les mouvements et les postures des personnes afin de détecter les chutes avec une précision élevée. Ils présentent l'avantage de ne nécessiter aucun dispositif porté par l'utilisateur et offrent une surveillance continue dans les zones couvertes par les caméras [8].

Ces systèmes analysent les images issues d'une ou plusieurs caméras pour détecter une chute. Ils peuvent traiter plusieurs événements simultanément et déterminer facilement les postures [10]. Les caractéristiques exploitées incluent la posture [11], le changement de forme pendant l'activité [12], [13], la trajectoire 3D de l'articulation de la tête [14] et la distribution verticale de la silhouette en 3D [15]. Ils fonctionnent en temps réel à l'aide de caméras 3D ou de capteurs de mouvements. La caméra de profondeur, telle que Kinect dans [16], utilise des LED infrarouges indépendantes de l'éclairage ambiant et opère efficacement en faible luminosité. Les informations de profondeur préservent la vie privée de la personne filmée et sont faciles à synthétiser, permettant ainsi de constituer un ensemble d'apprentissage volumineux et fiable pour la détection de mouvement sans marqueurs [17].

Dans le domaine de la vision par ordinateur, l'apprentissage profond (Deep Learning) a profondément transformé les approches traditionnelles grâce à sa capacité à apprendre

automatiquement des représentations complexes et hiérarchiques à partir des données. Contrairement aux méthodes classiques basées sur des capteurs portés, les architectures telles que les réseaux de neurones convolutifs (CNN) et les réseaux à mémoire longue et courte durée (LSTM) offrent une robustesse accrue face aux environnements variés et aux postures imprévisibles [18]. Ces modèles permettent de préserver les détails structuraux et d'adapter les algorithmes aux variations des mouvements humains, ce qui les rend particulièrement adaptés à la détection de chutes dans des conditions réelles [18], [19].

Par ailleurs, le modèle YOLOv5 s'est imposé comme une référence en détection d'objets en temps réel grâce à son compromis entre rapidité et précision. Son architecture optimisée permet un traitement efficace des flux vidéo avec une faible latence, ce qui est essentiel pour les applications de sécurité [20]. Il est également adapté aux systèmes embarqués et à l'Edge Computing, notamment sur des plateformes comme la Nvidia Jetson Nano [6]. De nombreuses études ont montré que YOLOv5 surpasse les modèles antérieurs en termes de précision et de robustesse, ce qui justifie son choix dans le cadre de ce travail [20].

Ce mémoire se concentre sur la troisième catégorie, les systèmes de vision. Grâce aux avancées récentes en intelligence artificielle et en vision par ordinateur, ces systèmes permettent une détection automatique et précise des chutes à partir d'images vidéo. Par ailleurs, l'intégration de technologies embarquées, telles que la carte Nvidia Jetson Nano, associée à une application Android dédiée à la surveillance et à la gestion des alertes, ouvre de nouvelles perspectives pour le développement de solutions compactes, autonomes et sécurisées. Ces solutions sont capables d'effectuer un traitement des données en temps réel directement à la périphérie du réseau (Edge AI, Intelligence Artificielle en périphérie), réduisant ainsi la latence, préservant la confidentialité des données et améliorant la réactivité du système.

Notre travail est structuré autour de trois volets complémentaires. Le premier concerne l'intelligence artificielle et vise à déterminer les paramètres optimaux permettant aux modèles de détecter et de classer efficacement les chutes à partir de séquences vidéo enregistrées dans un contexte de simulation. Cette phase comprend également l'évaluation des performances des différents modèles afin de sélectionner la solution la plus adaptée en termes de précision, de robustesse et de rapidité d'exécution.

Le deuxième volet consiste à déployer le modèle retenu sur une plateforme embarquée de type Nvidia Jetson Nano, utilisée comme unité de traitement en temps réel. Ce choix permet

de bénéficier des capacités d'accélération matérielle offertes par le GPU (Graphics Processing Unit) tout en garantissant une exécution locale et rapide des algorithmes de détection.

Le troisième volet, particulièrement important, porte sur le développement d'une application Android dédiée. Cette application assure la supervision du système, l'affichage des flux vidéo et des alertes, ainsi que la gestion des notifications en cas de détection d'une chute. Elle constitue l'interface principale entre le système de détection et les utilisateurs, facilitant ainsi la surveillance à distance et la prise en charge rapide des situations d'urgence. Son importance réside dans sa capacité à rendre le système accessible et pratique pour les familles et le personnel médical, tout en garantissant une interaction fluide et sécurisée avec la plateforme embarquée. De plus, l'utilisation d'Android, système largement répandu, permet une adoption facile et une compatibilité avec une grande variété de dispositifs mobiles [7].

Les objectifs scientifiques de ce mémoire sont multiples :

- Développer un système embarqué de détection de chutes basé sur l'IA ;
- Intégrer et optimiser le modèle YOLOv5 sur la Jetson Nano ;
- Concevoir une application Android intuitive pour la réception des alertes ;
- Évaluer les performances du système en conditions réelles.

Ces objectifs visent à contribuer à la recherche en télésurveillance intelligente et à proposer une solution à la fois efficace sur le plan technique et pertinente sur le plan social.

Enfin, le mémoire est structuré en quatre chapitres principaux :

- Le premier chapitre présente l'état de l'art sur la télésurveillance et les méthodes de détection de chutes ;
- Le deuxième chapitre décrit la plateforme matérielle et les technologies de communication utilisées ;
- Le troisième chapitre expose les méthodes d'intelligence artificielle retenues, notamment l'utilisation de YOLOv5 ;
- Le quatrième chapitre détaille la conception, la réalisation et la validation expérimentale du système proposé.

La conclusion générale met en évidence les contributions scientifiques et techniques, les limites rencontrées et les perspectives futures.

Références citées dans l'introduction générale

- [1] “World Health Organization. Good health adds life to years: Global brief for World Health Day 2012,” *No. WHO/DCO/WHO/2012.2. World Health Organization*, 2012.
- [2] C. Todd and D. Skelton, “What are the main risk factors for falls amongst older people and what are the most effective interventions to prevent these falls?,” *World Health Organization*, 2004.
- [3] S. N. Robinovitch, F. Feldman, Y. Yang, R. Schonnop, P. M. Leung, T. Sarraf, J. Sims-Gould, M. Loughin, “Video capture of the circumstances of falls in elderly people residing in long-term care: an observational study,” *The Lancet*, vol. 381, no. 9860, pp. 47–54, 2013.
- [4] S. R. Lord, C. Sherrington, H. B. Menz, J. C. Close, “Falls in older people: risk factors and strategies for prevention,” *Cambridge University Press*, 2007.
- [5] Y. S. Delahoz, M. A. Labrador, “Survey on fall detection and fall prevention using wearable and external sensors,” *Sensors*, vol. 14, no. 10, pp. 19806–19842, 2014.
- [6] N. Corporation, “Jetson Nano Developer Kit,” *Nvidia official documentation*, 2020.
- [7] J. Bravo, R. Hervás, J. Fontecha, and F. Navarro, “Smartphone-based fall detection system for elderly people,” *Sensors*, vol. 15, n°12, pp. 29113–29129, 2015.
- [8] R. Igual, C. Medrano, I. Plaza, “Challenges, issues and trends in fall detection systems,” *Biomedical engineering online*, vol. 12, no. 1, p. 66, 2013.
- [9] I. Halima, “Suivi de l’activité d’une personne à partir de capteurs multi-modalités préservant l’anonymat dans un cadre de détection et prévention des chutes chez les personnes âgées,” *Thèse de Doctorat, Université de Rennes I*, 2021.
- [10] Z. A. Mundher, J. Zhong, “A real-time fall detection system in elderly care using mobile robot and kinect sensor,” *Int J Mater Mech Manuf*, vol. 2, no. 2, pp. 133–38, 2014.
- [11] D. Brulin, Y. Benezeth, E. Courtial, “Posture recognition based on fuzzy logic for home monitoring of the elderly,” *IEEE transactions on information technology in biomedicine*, vol. 16, no. 5, pp. 974–982, 2012.
- [12] H. Foroughi, B. S. Aski, H. Pourreza, “Intelligent video surveillance for monitoring fall detection of elderly in home environments,” *11th international conference on computer and information technology*, pp. 219–224, 2008.
- [13] C. Rougier, J. Meunier, A. St-Arnaud, J. Rousseau, “Fall detection from human shape and motion history using video surveillance,” *In 21st International Conference on*

- Advanced Information Networking and Applications Workshops (AINAW'07)*, vol. 2, pp. 875–880, 2007.
- [14] C. Rougier, J. Meunier, A. St-Arnaud, J. Rousseau, “3D head tracking for fall detection using a single calibrated camera,” *Image and Vision Computing*, vol. 31, no. 3, 2013.
- [15] E. Auvinet, F. Multon, A. Saint-Arnaud, J. Rousseau, J. Meunier, “Fall detection with multiple cameras: An occlusion-resistant method based on 3-d silhouette vertical distribution,” *IEEE transactions on information technology in biomedicine*, vol. 15, no. 2, pp. 290–300, 2010.
- [16] “Microsoft. [En ligne]. Disponible sur : <http://www.microsoft.com/en-us/kinectforwindows/>,” 2012.
- [17] J. Shotton *et al.*, “Real-Time Human Pose Recognition in Parts from Single Depth Images,” *CVPR 2011*, pp. 1297–1304, 2011.
- [18] H. Gjoreski, M. Ciliberto, L. Wang, D. M. Gjoreski, and F. J. Ordóñez, “Deep learning for human activity recognition: A review,” *Pattern Recognition*, vol. 120, pp. 108–122, 2021.
- [19] M. R. Yacchirema, J. A. García, E. E. Cruz, and C. H. Sandoval, “Fall detection system for elderly people using IoT and mobile technologies,” *IEEE Access*, vol. 6, pp. 55568–55577, 2018.
- [20] G. Jocher, A. Stoken, J. Borovec, and A. Chaurasia, “YOLOv5: Object detection at scale,” *Github repository*, 2020.

Chapitre I

*Etat de l'art sur la
télésurveillance et la détection
de chute*

I.1 Introduction

Face à l'évolution démographique marquée par le vieillissement de la population et l'augmentation des pathologies chroniques, les systèmes de télésurveillance médicale connaissent un intérêt croissant. Ces technologies permettent d'assurer un suivi continu et à distance des patients, tout en optimisant les ressources médicales et en améliorant la qualité des soins.

Dans ce contexte, la détection de chute représente une problématique critique, notamment chez les personnes âgées, en raison des conséquences physiques et psychologiques qu'elle peut engendrer. Plusieurs approches ont été proposées dans la littérature, allant des systèmes basés sur les capteurs aux méthodes fondées sur la vision par ordinateur et l'intelligence artificielle.

L'objectif de ce chapitre est de présenter un état de l'art sur les différentes méthodes utilisées pour la télésurveillance et la détection de chute. Nous décrivons les approches basées sur les capteurs, la vision par ordinateur et les techniques d'intelligence artificielle, ainsi que les limites des travaux existants.

I.2 Télémédecine et télésurveillance des patients

I.2.1 Organisation mondiale de la Santé (OMS)

Fondée le 7 avril 1948, l'Organisation mondiale de la Santé (OMS), en anglais World Health Organization (WHO), est l'institution spécialisée de l'Organisation des Nations Unies (ONU) pour la santé. Son siège est à Genève dans le quartier des Nations. L'OMS est dirigée, depuis le 8 novembre 2006, par le Dr. Margaret Chan et par ses 193 Etats membres [1].

Sa structure comporte deux volets: d'une part le Siège à Genève, et d'autre part les six Régions et leurs Bureaux régionaux, avec un effectif total de plus de 7000 personnes. La gouvernance de l'Organisation est assurée par l'Assemblée mondiale de la santé (les délégués des 194 États membres), le Conseil exécutif et le Secrétariat. La Région européenne regroupe 53 pays, et le Bureau régional pour l'Europa se trouve à Copenhague. Les grands domaines d'activité de l'OMS sont les maladies transmissibles, les maladies non transmissibles, la promotion de la santé à toutes les étapes de la vie, les systèmes de santé ainsi que la préparation, la surveillance et la riposte aux situations d'urgence. La collaboration de l'Organisation avec l'Union européenne (UE) s'étend sur plusieurs niveaux [2].

Le but de l'OMS est d'amener tous les peuples au niveau de santé le plus élevé possible. L'Organisation, pour atteindre son but, l'Organisation exerce les fonctions suivantes [3]:

- a) Agir en tant qu'autorité directrice et coordonnatrice, dans le domaine de la santé, des

- travaux ayant un caractère international;
- b) Aider les gouvernements, sur leur demande, à renforcer leurs services de santé;
 - c) Fournir l'assistance technique appropriée et, dans les cas d'urgence, l'aide nécessaire, à la requête des gouvernements ou sur leur acceptation;
 - d) Stimuler et guider la recherche dans le domaine de la santé ;
 - e) Favoriser l'amélioration des normes de l'enseignement et de celles de la formation du personnel sanitaire, médical et apparenté.

I.2.2 Télémédecine

Dans un rapport de 1997, l'Organisation Mondiale de la Santé (OMS) donne une première définition de la télémédecine:

- La télémédecine est la pratique de la médecine au moyen de techniques interactives de communication des données (audiovisuelles notamment); cela comprend la fourniture de soins médicaux, la consultation, le diagnostic et le traitement, ainsi que la formation et le transfert de données médicales [4].

La télémédecine est une forme de pratique médicale à distance utilisant les technologies de l'information et de la communication [5].

En Algérie, il n'existe pas encore de définition juridique clairement établie de la télémédecine. A titre d'exemple, on peut citer l'expérience française. En France, la télémédecine est définie légalement par l'article 78 de la loi n° 2009-879 du 21 juillet 2009 dite «HPST» (Hôpital, Patients, Santé et Territoires). Elle englobe cinq actes principaux, définis par le décret n° 2010-1229 du 19 octobre 2010.

Les actes sont :

- a) La téléconsultation :** qui a pour objet de permettre à un professionnel médical de donner une consultation à distance à un patient. Un professionnel de santé peut être présent auprès du patient et, le cas échéant, assister le professionnel médical au cours de la téléconsultation.
- b) La télé-expertise :** qui a pour objet de permettre à un professionnel médical de solliciter à distance l'avis d'un ou de plusieurs professionnels médicaux en raison de leurs formations ou de leurs compétences particulières, sur la base des informations médicales liées à la prise en charge d'un patient ;
- c) La télésurveillance médicale :** qui a pour objet de permettre à un professionnel médical d'interpréter à distance les données nécessaires au suivi médical d'un patient et, le cas échéant, de prendre des décisions relatives à la prise en charge de ce patient. L'enregistrement et la transmission des données peuvent être automatisés ou réalisés par

le patient lui-même ou par un professionnel de santé ;

d) La téléassistance médicale : qui a pour objet de permettre à un professionnel médical d'assister à distance un autre professionnel de santé au cours de la réalisation d'un acte;

e) La réponse médicale qui est apportée dans le cadre de **la régulation médicale** [6].

L'OMS a mentionné deux formes de la télémédecine à savoir [7] :

 **La télémédecine clinique :** La télémédecine clinique est une activité professionnelle qui met en œuvre des moyens de télécommunication numériques permettant à des médecins et d'autres membres du corps médical de réaliser à distance des actes médicaux pour des malades.

 **La télémédecine informative :** La télémédecine informative est un service de communication audiovisuelle interactif qui organise la diffusion du savoir médical et des protocoles de prise en charge des malades et des soins dans le but de soutenir et d'améliorer l'activité médicale.

I.2.3 Télésurveillance médicale

La télésurveillance comprend la collecte de données cliniques et la transmission de ces données entre un patient à un endroit éloigné et un fournisseur de soins de santé par des moyens électroniques et des technologies de traitement de l'information.

I.2.3.1 Différents types de télésurveillance médicale

Les fonctions activées par un système de télésurveillance médicale peuvent être classées en trois catégories principales, chacune jouant un rôle essentiel dans le suivi à distance de la santé des patients :

a) Le télé-test et la télémaintenance : Les dispositifs d'assistance médicale, équipés de capteurs, font l'objet d'une surveillance en temps réel grâce aux systèmes informatiques installés au domicile du patient. Cette infrastructure permet de tester, à distance, le bon fonctionnement des appareils et des capteurs. Dès qu'une anomalie est détectée, une alerte est envoyée immédiatement au centre serveur, qui déclenche une intervention technique pour résoudre ou réparer l'anomalie, garantissant ainsi la continuité de la surveillance et de l'assistance.

b) La téléalarme : Lorsque les capteurs détectent une situation critique, soit en raison d'une défaillance de l'appareil, soit liée à l'état du patient, une alarme est envoyée instantanément au centre serveur. Ce dernier, après avoir analysé la situation, intervient à distance pour prendre le contrôle des appareils et activer l'alarme auprès du médecin traitant, du service des urgences ou du personnel médical du centre, selon la nature de l'alerte. Cette réactivité permet d'assurer une prise en charge rapide et

efficace du patient.

c) Le télé-monitoring : Le télé-monitoring consiste à collecter, en temps réel, des données relatives au fonctionnement des dispositifs médicaux et à l'état de santé du patient. Cette surveillance continue permet d'identifier toute déviation par rapport aux prescriptions médicales, telles que le non-respect des durées de traitement ou des paramètres de télésurveillance. En cas de non-conformité, le système alerte immédiatement le médecin prescripteur, qui peut alors ajuster le traitement ou intervenir rapidement pour prévenir toute complication [8].

1.2.3.2 Principe de la télésurveillance médicale

La télésurveillance médicale d'un patient s'appuie sur un système d'information global comprenant les éléments suivants :

- a) Un ensemble de capteurs de différents types** (physiologie, environnement, activité) installés dans l'habitat ou portés par la personne, reliés en réseaux pour la collecte en temps réel de données, et d'appareillages automatiques (domotique) pour adapter l'environnement de vie de la personne à ses capacités personnelles, motrices et cognitives .
- b) Une unité locale de traitement**, au niveau de chaque habitat, responsable du stockage et du traitement des signaux reçus des capteurs, de la gestion d'une base de connaissances relative à la personne télé-surveillée, et de l'émission de messages et d'alarmes.
- c) Un centre de télé-vigilance** pour le traitement des messages et alarmes reçus des patients. Un ensemble d'acteurs (personnel médical, personne télé-surveillée) peuvent accéder à tout moment, après authentification et selon leurs privilèges, aux données du système, au niveau de l'unité locale de traitement [9].

1.2.3.3 Enjeux de la télésurveillance médicale

Les principales fonctionnalités nécessaires à la mise en place de systèmes de télésurveillance médicale sont la perception, l'analyse, le stockage et la transmission de données et d'informations relatives au patient télé-surveillé. On identifie alors d'après cinq sous-systèmes clés du développement des systèmes d'information pour les services de télésurveillance [9] :

- a) Système de surveillance local :** Il s'agit d'un réseau local pour l'enregistrement téléométrique de données relatives à un patient par l'intermédiaire de capteurs physiologiques, d'environnement et d'activité.
- b) Système d'analyse de données :** La grande quantité de données collectées nécessite

la conception d'assistants intelligents pour l'extraction d'informations pertinentes permettant la génération de messages et d'alarmes, l'aide au diagnostic et à la décision.

- c) Système de base de données :** Les données collectées ou les informations extraites doivent être stockées et accessibles pour leur consultation ou leur mise à jour.
- d) Système d'interfaces :** Les données et informations issues de la télésurveillance et de l'analyse des données collectées doivent être facilement accessibles aux différents acteurs du système.
- e) Système de communication :** Il s'agit de permettre l'interopérabilité des quatre sous systèmes précédents à travers un réseau médical qui relie les patients, les centres hospitaliers.

I.2.3.4 Domaines d'application de télésurveillance

La télésurveillance est employée dans de nombreuses situations, généralement pour des raisons de sécurité :

- Dans le cadre de la sécurité routière, au moyen de caméras spécialisées ou des capteurs à proximité voire même noyés dans la chaussée permettent d'évaluer la densité du trafic, les ralentissements qui peuvent en découler, la présence de personnes sur les bandes d'arrêt d'urgence, etc.
- Pour la surveillance des machines : divers capteurs permettent d'évaluer l'état de la machine, ces informations peuvent alors être envoyées à un poste de surveillance.
- L'épuisement de consommables, une anomalie de fonctionnement ou même un acte de malveillance serait alors détecté à distance.
- Dans le cadre de la prévention de la délinquance (avec notamment la vidéosurveillance)
- Pour la surveillance de lieux sensibles (banques, centrales nucléaires, etc.) et d'habitations.
- Dans le cadre de la télémédecine, et en particulier pour la surveillance des patients à distance [10].

I.3 Application médicale chez les personnes âgées

La personne âgée est définie par l'Organisation Mondiale de la Santé (OMS) comme étant une personne de plus de 60 ans. Il existe une diversité de capacités et de besoin en termes de santé chez ces personnes [11]. Par ailleurs, la notion de personne «âgée» demeure relative subjective, car elle dépend de nombreux facteurs tels que le contexte social, la situation

géographie et les caractéristiques culturelles [12].

Les syndromes gériatriques correspondent à une situation clinique retrouvée chez la personne âgée dont la cause est multifactorielle. Ces syndromes ne sont pas des pathologies à part entière mais ils correspondent à une accumulation de déficits organiques qui peut mener à un risque de dépendance fonctionnelle voire d'entrée en institution. Leur prise en charge est multidisciplinaire.

Dans les paragraphes suivants, nous présenterons deux syndromes gériatriques les plus fréquents : la fragilité et la chute.

La fragilité : La fragilité est un concept qui existe depuis des années en tant que syndrome gériatrique. L'hétérogénéité de la population âgée conduit à distinguer trois profils, comme l'illustre la Figure I.1:

- les «robustes» : personnes âgées qui sont globalement en bonne santé ;
- les «fragiles» : personnes âgées présentant des limitations fonctionnelles et une baisse des capacités d'adaptation au stress
- les «dépendants» : personnes âgées qui ne peuvent plus réaliser certaines activités de la vie quotidienne (ex. : s'habiller, se déplacer)

Une personne âgée «fragile» peut revenir à l'état de robuste, une personne âgée «dépendante» ne pourra plus revenir à l'état précédent. Elles sont à risque d'institutionnalisation voire de décès [13].

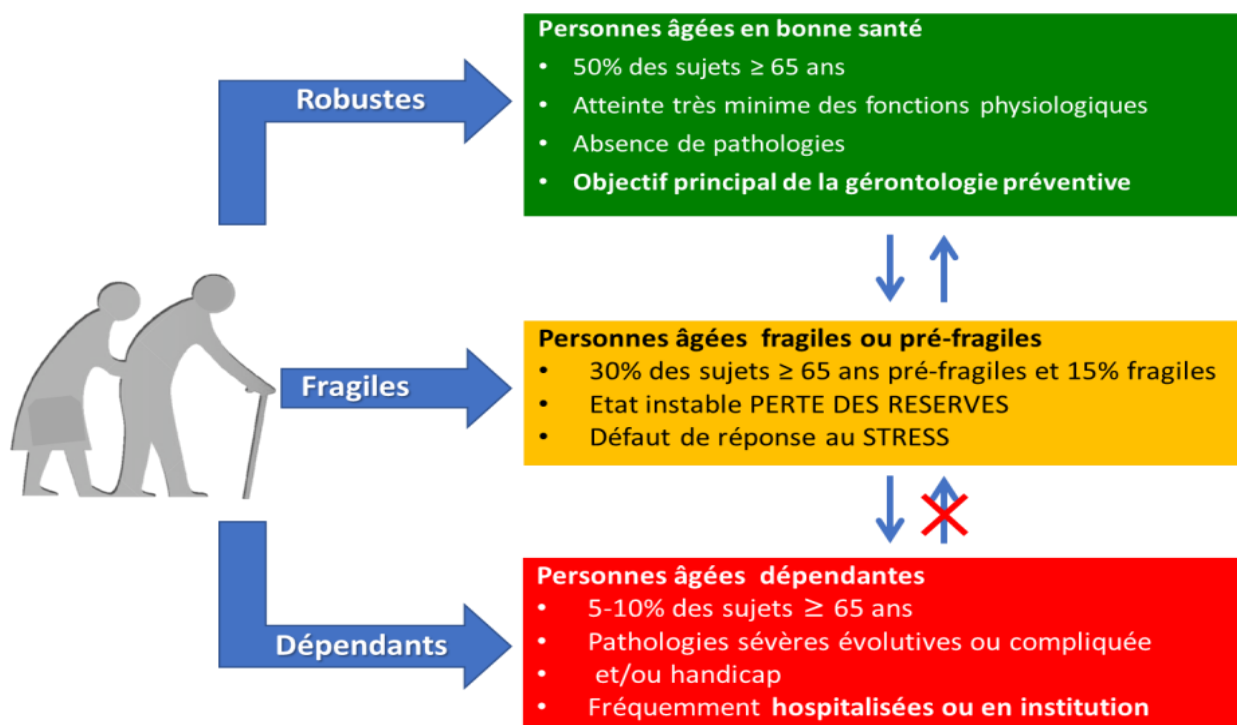


Figure I. 1. Présentation des 3 états de la fragilité chez le sujet âgé [13].

La chute : Le terme de «chute» est défini comme un événement à l'issue duquel un individu se trouve par inadvertance sur le sol ou sur toute autre surface située à un niveau inférieur de celui d'origine.

Chaque année, environ deux millions de chutes sont responsables de plus de 130000 hospitalisations et de 10000 décès chez les personnes âgées de plus de 65 ans. Par ailleurs, 40% des personnes âgées hospitalisées pour une chute sont orientées vers une institution [13]. Selon la littérature, la chute correspond à une perte brutale et involontaire d'équilibre, entraînant la personne au sol ou à un niveau inférieur, sous l'effet de la gravité. Il s'agit donc d'un événement survenant indépendamment de la volonté, où le corps, sous l'action de son propre poids, est attiré vers le sol. Ainsi, la chute se caractérise par la survenue imprévue d'un déséquilibre menant à une posture inférieure à celle initialement occupée [6].

En l'absence d'une intervention rapide et adaptée, une chute peut entraîner des conséquences graves, pouvant aller jusqu'au décès. C'est dans ce contexte que s'inscrit l'objectif de ce mémoire, visant à prévenir, détecter ou prendre en charge efficacement ces situations.

L'organisme des personnes âgées vieillit et ses performances diminuent :

- **Au niveau des os et des articulations :** La force musculaire diminue tandis que l'amplitude des gestes est diminuée par l'arthrose. L'architecture du pied se modifie, compromettant la marche. L'ostéoporose entraîne des fractures du col du fémur, ...etc.
- **Au niveau du système nerveux :** La précision des gestes peut être altérée, notamment par des pathologies comme la maladie de Parkinson, etc.
- **Au niveau sensoriel :** Les troubles visuels (presbytie, cataracte, dégénérescence maculaire liée à l'âge, etc.) et auditifs (presbycusie) diminuent l'aptitude à éviter les obstacles. Par ailleurs, avec l'âge, le système responsable de l'équilibre fatigue, ... etc [14].

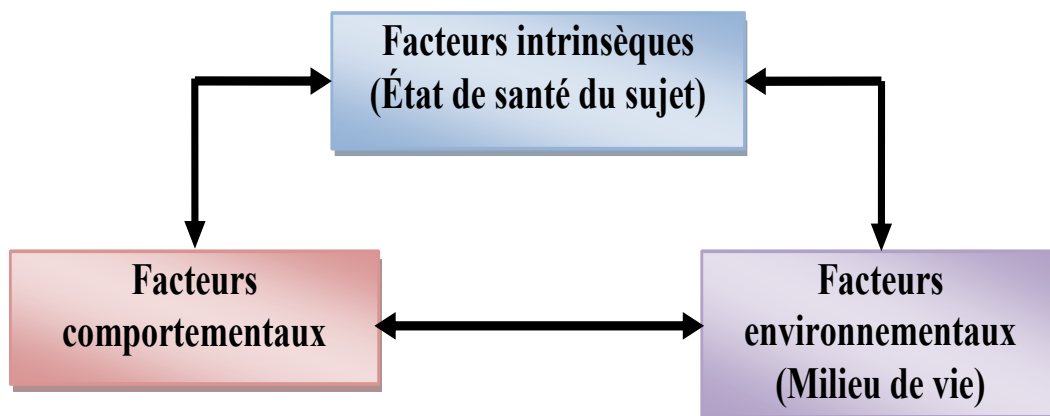


Figure I. 2. Classification générale des causes de chutes [15].

Le mécanisme de la chute est complexe chez la personne âgée en raison de son caractère multifactoriel. Dans la démarche de diagnostic après une chute, il convient de distinguer les facteurs prédisposants souvent multiples (facteurs de risque qui sont les conséquences des maladies chroniques et du vieillissement), des éléments précipitants (intrinsèques et extrinsèques). Cette distinction, illustrée par la Figure I.2, permet une meilleure compréhension des causes de la chute et une prise en charge plus adaptée [15].

Les raisons favorisant la chute des sujets âgés sont complexes. Néanmoins, hormis la fragilité liée à l'âge (altérations cardiovasculaires, neuromusculaires et métaboliques), six séries de facteurs, classés en facteur intrinsèque ou extrinsèque, peuvent favoriser la chute de la personne âgée (voir Tableau I.1) [16].

Tableau I. 1. Les facteurs de chute chez la personne âgée [16].

Facteur intrinsèque	Facteur extrinsèque
• Le processus de vieillissement. • Les sujets sont encore actifs et prennent, parfois des risques. • La peur de tomber. • L'atteinte par une maladie. • La consommation médicamenteuse.	• L'environnement de vie.

I.3.1 Types de chute

En général, il existe plusieurs types de chutes selon plusieurs causes comme suivants :

- La chute dite lourde perte de verticalité rapide associée à un choc .
- La chute molle : lorsque la personne se retient à un meuble par exemple.
- La chute syncopale (maladie): lors de la perte de conscience.

Chaque type parmi ces derniers types de chute peut être divisé en plusieurs catégories en se basant sur différents critères : l'orientation, l'amplitude du mouvement et aussi la transition de la posture. Plus le temps passé au sol est long, plus les conséquences de la chute seront graves. Il est donc primordial de pouvoir détecter une chute pour le maintien de l'autonomie des personnes âgées. Cependant, il est difficile de définir avec précision toutes les chutes, d'autant qu'il convient d'identifier également les fausses alertes. Afin de minimiser les fausses d'alarme, les chercheurs ont essayé d'analyser le comportement de la personne au cours du temps afin de distinguer entre deux types d'activités, les activités anormales qui sont considérées comme des activités non-contrôlées et des activités normales qui sont considérés comme des activités contrôlées [17].

I.3.2 Caractéristiques de la chute

Dans cette section, nous présentons les différentes phases permettant l'analyser et la reconnaissance d'une chute. Avant d'aborder cette analyse, les chercheurs ont essayé de définir une chute et de classifier les différentes méthodes de détection de chute. L'événement de chute est généralement décomposé en quatre phases comme l'illustre la Figure I.3 [17].

- 1) **Phase avant la chute (Pré-chute):** Durant cette phase, la personne exerce ses activités quotidiennes. Elle peut effectuer des mouvements normaux, parfois brusques, tels que s'asseoir ou s'accroupir. Ces activités ne doivent pas générer d'alarme par le détecteur de chutes.
- 2) **Phase critique :** Cette phase, très courte, correspond à un mouvement soudain du corps en direction du sol, terminé par un impact au sol. Elle peut être exploitée des méthodes directes de détection de chutes, notamment : la détection précoce de la chute (par exemple à partir de la vitesse du corps) ou la détection de l'impact au sol.
- 3) **Phase après la chute (post-chute):** Juste après la chute, la personne reste le plus souvent immobile, incapable de bouger et allonger sur le sol. Cette phase peut être exploitée par des méthodes indirectes de détection de chute : la présence d'une personne en position allongée ; ou l'absence de mouvement.
- 4) **Phase de rétablissement :** Éventuellement, la personne peut réussir à se relever seule ou avec l'aide d'une autre personne [17].

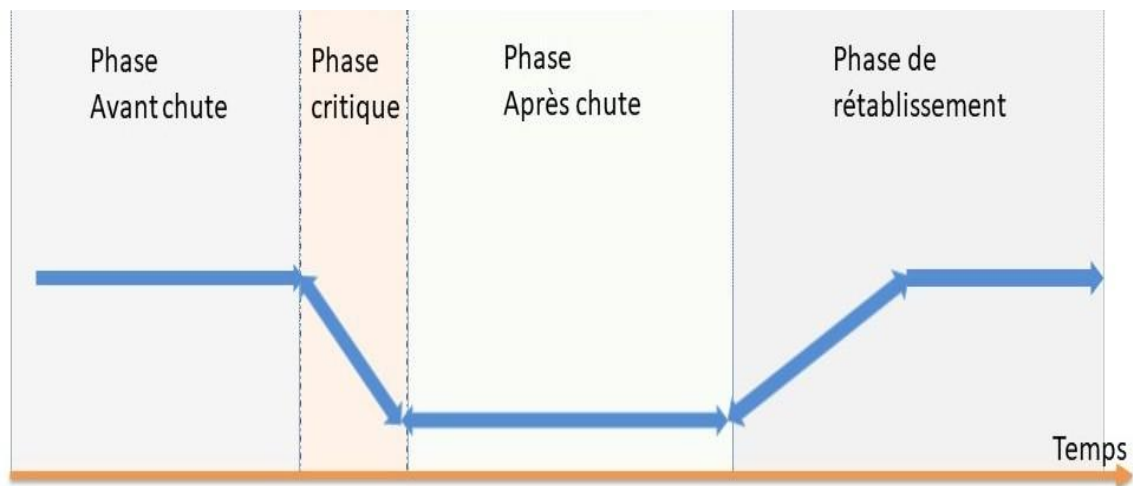


Figure I. 3. Les différentes phases de la chute [17].

Les chutes peuvent avoir des conséquences physiques, sociales, psychologiques, financières et médicales importantes, ainsi que des conséquences sur les coûts de santé. La chute provoque des traumatismes physiques de degrés différents, souvent classés en trois catégories [18]:

- Les traumatismes mineurs qui apparaissent au niveau de la peau tels que les hématomes sous cutanés ou les légères écorchures de la peau.
- Les traumatismes modérés correspondant à des fractures (des côtes ou/et des vertèbres) avec des symptômes légers et une douleur modérée.
- Les traumatismes sévères correspondant aux fractures fémorales, des hématomes cérébraux, des hématomes extra ou sous-duraux, des traumatismes crâniens et des déchirures cutanées de grande taille ou/et profondes. Cette catégorie de traumatismes conduit constamment à une hospitalisation et met en jeu le pronostic vital des personnes.

Pour minimiser les conséquences de la chute, il existe des solutions pour évaluer le risque des chutes afin de les prévenir. Dans leurs cabinets, les médecins évaluent le risque de la chute par la réalisation de deux tests simples : Le test de Tinetti et le test «Get up and Go» [18] .

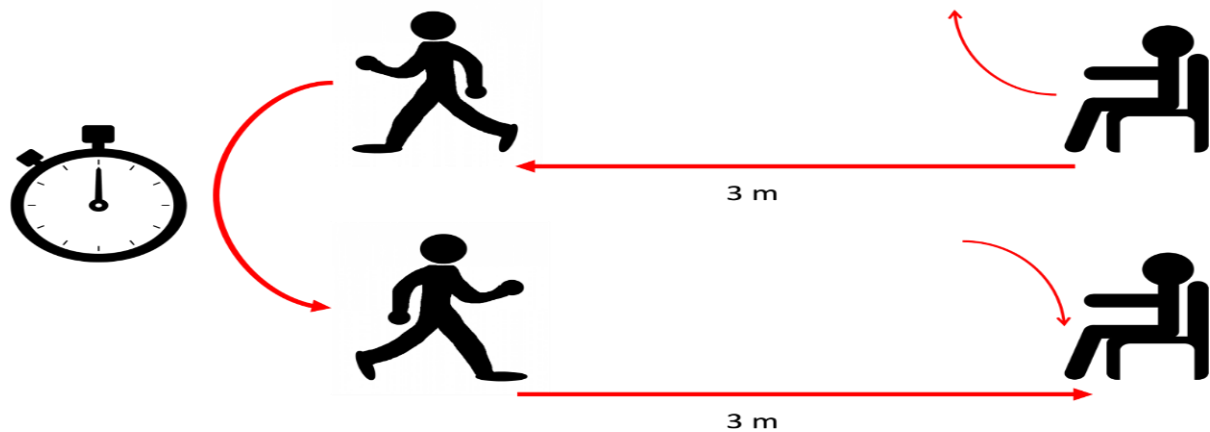


Figure I. 4. Test « Get up and Go » [18].

L'évaluation du risque de chute peut également être réalisée en analysant le comportement de la personne dans son environnement, notamment à travers le suivi de sa routine quotidienne. Par exemple, une diminution ou une absence d'exécution des activités de la vie quotidienne peut être un signe de fragilité et semble exposer la personne à la chute.

Les systèmes existants jouent ainsi un rôle essentiel pour déclencher des alertes en situation d'urgence, permettre une intervention rapide, préserver l'autonomie de la personne et améliorer ainsi les chances de survie [18].

I.3.3 Système de télésurveillance pour le maintien à domicile des personnes âgées

Il devient primordial de disposer d'infrastructures adéquates pour assurer la prise en charge des personnes âgées, qui constituent une population particulièrement exposée aux chutes

accidentelles. Pour résoudre ce problème, le maintien à domicile de la personne âgée semble être la solution privilégiée.

Dans un contexte d'amélioration des conditions de vie, il est donc nécessaire d'adapter les domiciles afin de permettre à la personne âgée de rester le plus longtemps possible dans son logement tout en lui offrant un maximum d'autonomie. Il s'agira donc de surveiller son état de santé via la détection d'une chute ou d'un accident cardio-vasculaire, via la détection de la fragilité, etc. L'installation de moyens technologiques (voir Figure I.5) peut ainsi être une aide significative dans l'accomplissement des actes de la vie quotidienne. Ces solutions sont souvent proposées dans les maisons intelligentes [18].

C'est dans ce cadre que s'inscrit l'objectif de ce travail : la réalisation d'un système intelligent permettant de détecter les chutes et d'intervenir dans un délai réduit afin de sauver la vie des personnes âgées, tout en respectant leur vie privée.

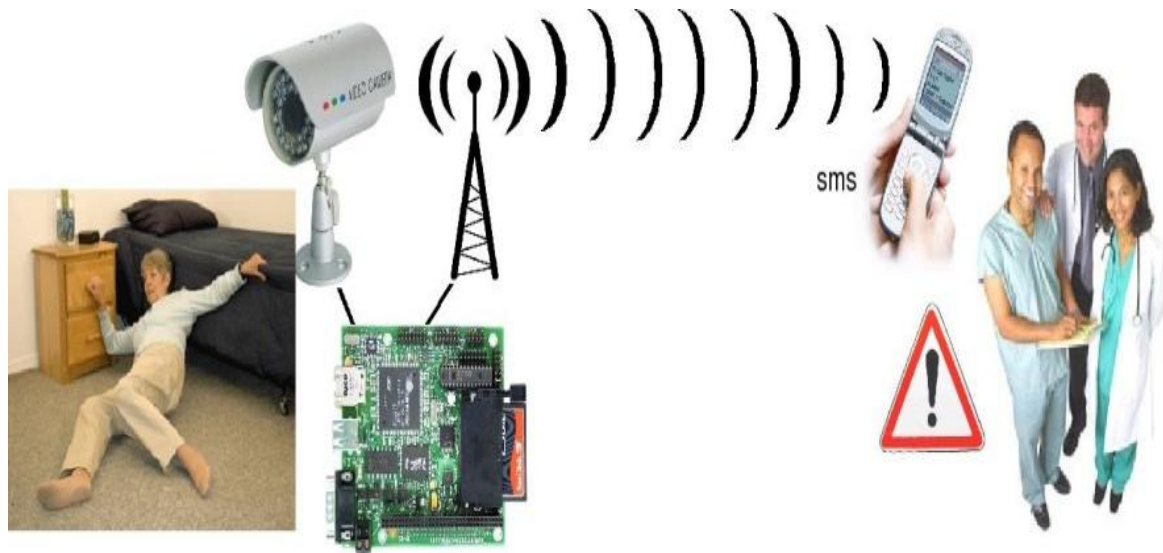


Figure I. 5. Surveillance des personnes âgées [17].

I.4 Méthodes de détection de chute

Dans la littérature scientifique, le domaine de la surveillance des personnes âgées est devenu indispensable dans le monde. Les chercheurs ont profité de cette pandémie mondiale pour développer des systèmes de surveillance robustes et faisables pour sauver la vie des personnes âgées.

La Figure I. 6, montre les trois catégories des systèmes de détection de chute dont on trouve les systèmes à base des capteurs portables, les systèmes à base de caméra et les systèmes qui regroupent les deux derniers systèmes [17], [18].

Le choix d'un capteur dépend généralement de la personne âgée et la position où il sera installé selon l'infrastructure de la maison. Généralement, ces capteurs peuvent être divisés en

plusieurs groupes selon leurs positionnements où ils sont installés. On trouve des capteurs qui sont portés par la personne sous forme de bracelet, avec les vêtements ou à travers les chaussures. Il y a aussi des capteurs qui sont installés dans la maison comme le sol [17].

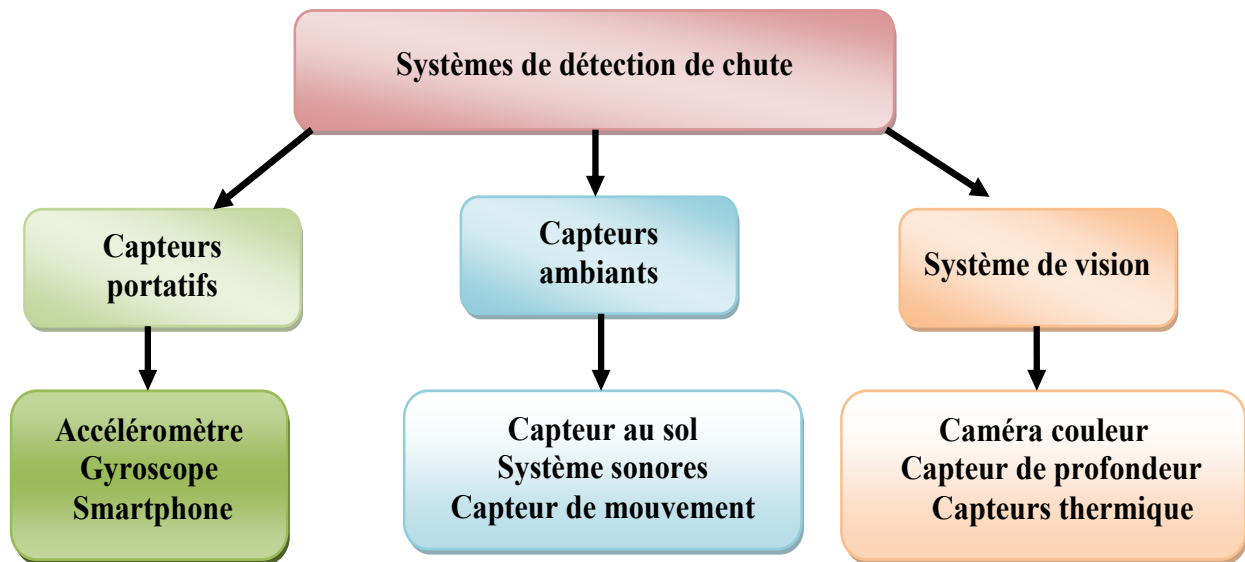


Figure I. 6. Catégories de capteurs pour la détection des activités de la personne âgée ou de sa chute [18].

I.4.1 Capteurs portatifs

Les méthodes classiques de détection de chutes reposent principalement sur des capteurs ou boutons poussoirs portés par la personne. Les boutons d'alerte permettent à la personne âgée d'appeler de l'aide en cas de problèmes. Cependant, dans le cas d'une chute, cette technologie n'est efficace que si la personne est consciente après l'événement et capable d'actionner le bouton, ce qui n'est pas toujours le cas, notamment en cas d'immobilisation. Les capteurs automatiques de détection de chutes [19], [20] présentent ainsi un intérêt plus important, car ils ne nécessitent aucune d'intervention humaine. Certains systèmes reposent sur des accéléromètres, capables de mesurer la magnitude et la direction de l'accélération; tandis que d'autres utilisent des gyroscopes qui mesurent l'orientation du corps. Plusieurs produits commerciaux exploitent déjà ces technologies.

Un capteur portable est un dispositif autonome en énergie, disposant de sa propre source d'énergie (pile ou batterie). Il intègre une unité de traitement ainsi qu'une unité de communication. Il transforme les grandeurs physiques mesurées (température, mouvement, etc.) en données numériques exploitables par l'unité de traitement. Cette dernière constitue le cœur du système : elle gère les périphériques, à programmer l'ordonnancement des tâches, à

traiter les données et à gérer la consommation énergétique (mise en veille et réveil du capteur selon la situation). Le traitement peut être effectué localement, ce qui nécessite des ressources mémoire importantes, ou déporté vers un serveur. L'unité de communication assure la transmission des données vers un serveur via des technologies sans fil telles que le Wi-Fi ou le Bluetooth. En termes de consommation énergétique, la technologie Wi-Fi consomme beaucoup d'énergie, contrairement au standard Bluetooth [21].

La Figure I.7 illustre le flux d'information dans un système basé sur des dispositifs portables, un smartphone et un serveur.

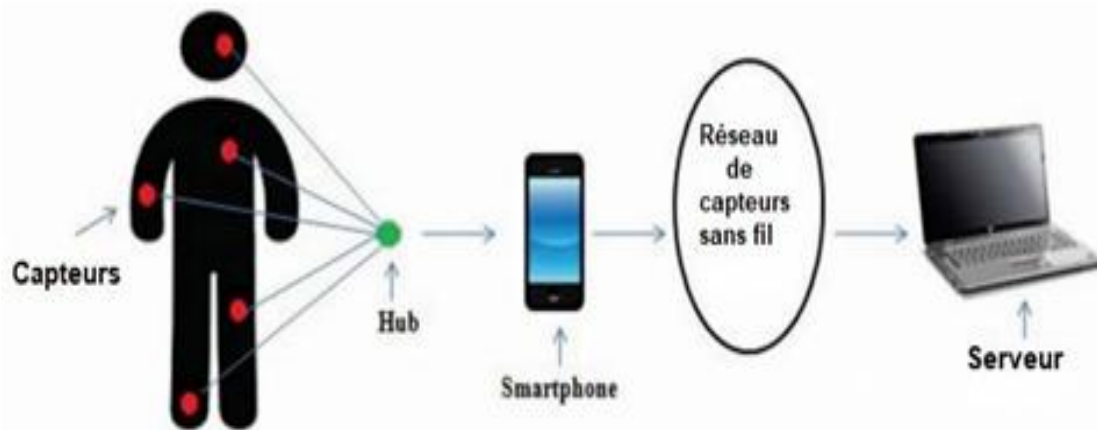


Figure I. 7. Système de détection des chutes basé sur les capteurs portables [22].

I.4.2 Capteurs ambiants

Pour éviter les problèmes des capteurs portables, les chercheurs ont développé une autre solution de surveillance et de détection des chutes. Ce type de systèmes repose sur des capteurs ambiants, installés directement chez la personne âgée ou dans les maisons de retraite. Les données recueillies à partir des capteurs sont envoyées à un ordinateur pour les traiter et déclencher des alarmes dans les cas d'urgence. Ces capteurs peuvent être des capteurs de pression et de vibration au sol, des capteurs infrarouges muraux ou placés au plafond ou des capteurs acoustiques [23].

L'un des principaux avantages de ces dispositifs est leur discrétion : étant intégrés dans l'environnement, les personnes âgées peuvent les oublier facilement et ne se sentent pas surveillées en permanence. Par exemple, les capteurs placés au sol permettent de détecter les pressions plantaires exercées lors des déplacements. Ainsi, les algorithmes associés peuvent en déduire certaines informations, telles que la localisation de la personne dans le logement au cours de la journée ou la détection d'une position anormale, comme une personne allongée au sol.

Contrairement aux autres techniques, les problèmes de confidentialité sont éliminés en utilisant ces capteurs ambiants. Mais, ce type de capteurs est sensible au bruit. Il est facilement perturbé par l'environnement et coûteux.

Le développement des technologies de détection des chutes suit une progression notable. Par ailleurs, plusieurs idées d'utilisation de capteurs ambiantes restent restreintes à la recherche scientifique sans avoir recours à la commercialisation.

I.4.3 Approches basées sur la vision

Dans n'importe quel système de vidéosurveillance et spécialement de détection de chute, le type de caméra joue un rôle important dans le résultat final. Le système de vision extrait l'information des images brutes pour les traiter, comme illustré dans la Figure I.8.

Il existe différents types de caméras de vidéosurveillance, à domicile, dans une entreprise ou un commerce. Elles se distinguent par leurs fonctionnalités mais aussi par leur niveau de gamme. Plusieurs travaux ont été proposés pour la vidéosurveillance des personnes âgées en utilisant différents types de caméra où le choix d'un type de caméra est dépendu de plusieurs critères; la méthode à utiliser, l'environnement à surveiller et la position de la caméra, le champ de vue et n'oubliant pas le cout du système [17].



Figure I. 8. Système de détection des chutes basé sur la vision [22].

Parmi les performances de caméras, nous distinguons les critères suivantes :

- **Luminosité** : elle s'exprime en lux. Certaines caméras permettant de filmer aussi bien dans l'obscurité totale (0 lux) qu'en présence de forte luminosité.
- **Qualité de l'image** : plus le nombre de lignes est élevé, meilleure est la qualité de l'image.
- **Taille de l'objectif** : plus l'objectif est large (de 2,5 mm à 100 mm), plus le champ de vision est large; à l'inverse, un objectif plus petit permet de zoomer sur des zones plus éloignées.

Un système de télésurveillance peut associer différents types de caméras de surveillance pour une meilleure efficacité [24].

Aussi, un autre paramètre très important concerne le type de caméra. Sur le marché, on distingue notamment :

- a) **Caméra omnidirectionnelle** : qui offrent un champ de vue jusqu'à 360 degrés. Cependant, elle est rarement utilisée dans le domaine de détection de chute en raison de la déformation de l'image obtenue et de son coût élevé par rapport aux autres solutions.
- b) **Caméra fisheye** : ce type de caméra offre également un champ de vision de 360 degrés et aussi se base sur une seule lentille. Cependant, ce type de caméra produit des images de moins de qualité.
- c) **Caméra RGB** : Ce type de caméra est le plus utilisé dans la littérature car elle offre un champ de vue supérieur à 120 degrés, une meilleure qualité d'image et elle est moins chère. Cependant, les chercheurs doivent faire un effort pour étudier le contenu des images RGB où ils sont obligés de développer des algorithmes de prétraitement robustes aux changements de l'éclairage et les problèmes de bruit d'acquisition [17].
- d) **Caméra de surveillance IP** : La caméra de surveillance IP est utilisée pour les installations de surveillance de réseau, également appelée surveillance à distance IP (Internet Protocol). Caméra IP connectée au serveur central (par liaison WiFi ou câble RJ45) pouvant gérer de nombreux sites distants. Chaque caméra IP possède sa propre adresse IP [25].

Il existe deux types de caméras IP: La première est la caméra Simple : elle capture les images et les transfère directement sur le réseau informatique, où elles sont ensuite traitées, enregistrées et les stockées. La deuxième est la caméra IP autonome : toutes les opérations de traitement d'image sont effectuées directement par la caméra IP. Elle peut également être équipée de capteurs intégrés (détection de mouvement, fonction de comptage, détection d'anomalies, etc.) et déclencher une alarme en cas d'anomalie.

La vidéosurveillance IP permet aux utilisateurs d'obtenir à tout instant et en tout lieu des informations sur une opération en cours, et de la suivre en temps réel [26].

- e) **La caméra Microsoft Kinect** : qui est particulièrement utilisé au début dans les jeux et elles deviennent de plus en plus importantes dans le domaine de la vidéosurveillance. Parmi les avantages d'utiliser ce type de caméra dans la vidéosurveillance est qu'elles sont insensibles au changement de la luminosité car

elles utilisent un capteur infrarouge et elles offrent aussi 3 flux d'informations qui aident à la construction des modèles d'objets 3D [17].

D'autres méthodes reposent sur la reconnaissance des chutes à partir de flux vidéo. Certaines approches utilisent des techniques classiques, telles que le seuillage, qui consiste à appliquer un seuil sur l'image afin d'en extraire des informations pertinentes.

D'autres, en revanche, s'appuient sur des méthodes d'intelligence artificielle, beaucoup plus performantes, permettant d'obtenir de meilleurs résultats en termes de précision et de robustesse.

Dans la section suivante, nous présenterons un aperçu des principales méthodes d'intelligence artificielle utilisées dans ce domaine.

I.5 Méthodes d'intelligence artificielle

L'amélioration des algorithmes et des techniques d'intelligence artificielle a permis de perfectionner les systèmes de détection de chute, en réduisant les fausses alertes et en améliorant la précision de détection des chutes réelles.

L'intelligence artificielle (IA) est une branche de l'informatique qui vise à créer des systèmes capables de reproduire les fonctions cognitives humaines, telles que l'apprentissage, la reconnaissance des formes, la prise de décision et l'adaptation à des environnements complexes. Fondée sur des modèles mathématiques et algorithmiques avancés, l'IA permet aux machines d'exécuter des tâches nécessitant une intelligence humaine, notamment l'analyse d'images, la compréhension du langage naturel et la prise de décision automatisée.

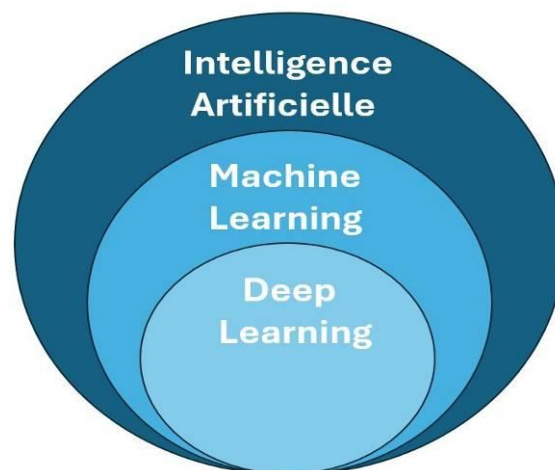


Figure I. 9. Intelligence Artificielle [27].

Parmi les sous-domaines les plus développés de l'IA figurent l'apprentissage automatique (Machine Learning, ML) et l'apprentissage profond (Deep Learning, DL), qui permettent

aux machines d'apprendre à partir de données et d'améliorer leurs performances sans nécessiter de programmation explicite, comme l'illustre la Figure I.9 [27].

I.5.1 Apprentissage automatique

L'apprentissage automatique ou Machine Learning (ML) est une branche de l'IA qui consiste à donner aux machines l'aptitude d'apprendre et d'améliorer leurs performances automatiquement en fonction des données qu'ils traitent sans être explicitement programmés [28], [29]. Il existe plusieurs types d'apprentissages, comme le montrent les sous-sections suivantes.

a) Apprentissage supervisé

L'apprentissage supervisé est la méthode la plus couramment utilisée en IA. Dans cette approche, comme l'illustre la Figure I.10, les algorithmes d'apprentissage sont entraînés sur un ensemble de données étiquetées, où chaque exemple comprend des entrées associées à une sortie attendue, appelée "étiquette". L'objectif de l'apprentissage supervisé est de construire un modèle capable de prédire correctement la sortie de nouvelles entrées, en se basant sur les exemples d'entraînement. Pour obtenir les meilleures performances, les paramètres du modèle sont ajustés en minimisant l'erreur de prédiction au cours du processus d'apprentissage.

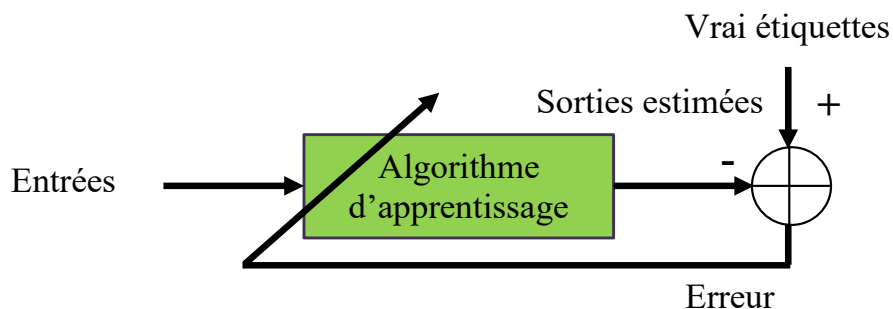


Figure I. 10. Apprentissage supervisé.

b) Apprentissage non supervisé

L'apprentissage non supervisé comme montre la Figure I.11 implique l'entraînement d'algorithmes sur un ensemble de données non étiqueté, où les sorties correctes ne sont pas fournies. Le but de cette méthode est de découvrir la structure sous-jacente des données, en identifiant des relations et des modèles qui ne sont pas immédiatement évidents.

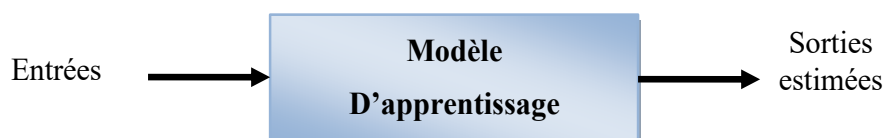


Figure I. 11. Apprentissage non supervisé.

c) Apprentissage semi-supervisé

L'apprentissage semi-supervisé est une méthode d'apprentissage qui combine les caractéristiques de l'apprentissage supervisé et non supervisé. Dans cette approche (Figure I.12), les algorithmes sont entraînés sur un ensemble de données partiellement étiqueté, où certaines entrées ont des étiquettes tandis que d'autres n'en ont pas. L'objectif de l'apprentissage semi-supervisé est d'utiliser les informations disponibles dans les données étiquetées et non étiquetées pour créer un modèle performant.

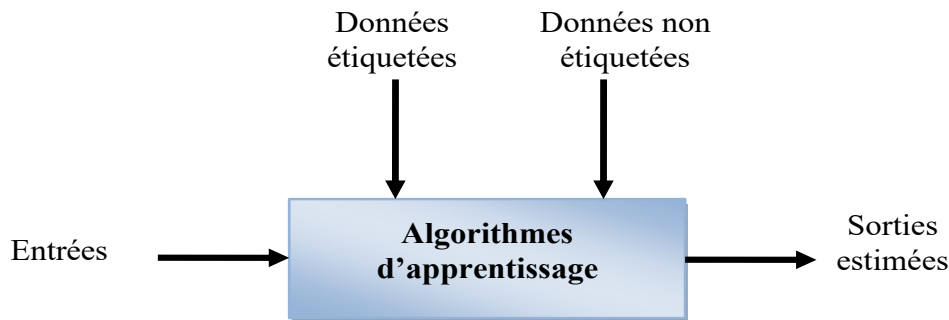


Figure I. 12. Apprentissage semi-supervisé.

d) Apprentissage par renforcement

L'apprentissage par renforcement, illustré dans la Figure I.13, est une méthode d'apprentissage dans laquelle un agent apprend à prendre des décisions en interagissant avec un environnement.



Figure I. 13. Apprentissage par renforcement [29].

L'agent reçoit des récompenses ou des pénalités en fonction de ses actions et apprend progressivement à maximiser ses récompenses cumulées au fil du temps. Cette approche est particulièrement adaptée aux problèmes de décision séquentielle et de contrôle, où les actions ont des conséquences à long terme.

I.5.2 Apprentissage profond

L'apprentissage profond ou Deep learning (DL) est une sous-catégorie du Machine Learning qui utilise des réseaux de neurones artificiels pour apprendre des modèles complexes à partir des données, comme l'illustre la Figure I.14.

Les techniques d'apprentissage profond sont basées sur l'utilisation de réseaux de neurones artificiels pour modéliser et résoudre des problèmes complexes. En utilisant une méthode d'apprentissage supervisé, le réseau est entraîné à partir d'un grand ensemble de données d'entraînement pour identifier des motifs et des relations cachées entre les entrées et les sorties. Ensuite, le modèle est utilisé pour prédire des sorties pour de nouvelles entrées en utilisant les motifs appris à partir des données d'entraînement [30].

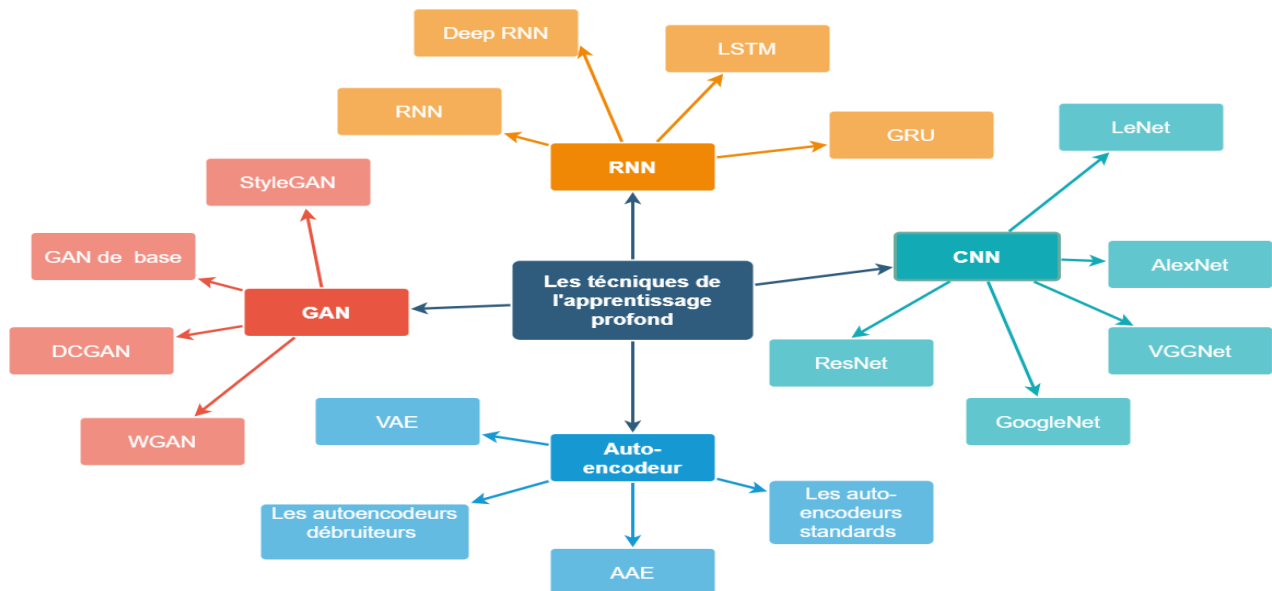


Figure I. 14. Techniques d'apprentissage en profondeur avec leurs différents types [30].

I.5.2.1 Réseaux de neurones artificiels

Les réseaux de neurones artificiels sont directement inspirés des réseaux neuronaux biologiques du cerveau humain, qui comportent de nombreux neurones biologiques hautement connectés. Ces neurones communiquent entre eux en transmettant des signaux par l'intermédiaire de synapses. Ces systèmes biologiques sont extrêmement complexes par rapport aux systèmes artificiels, en raison du grand nombre de neurones inter connectés qui ne peuvent pas être traités par une machine [31].

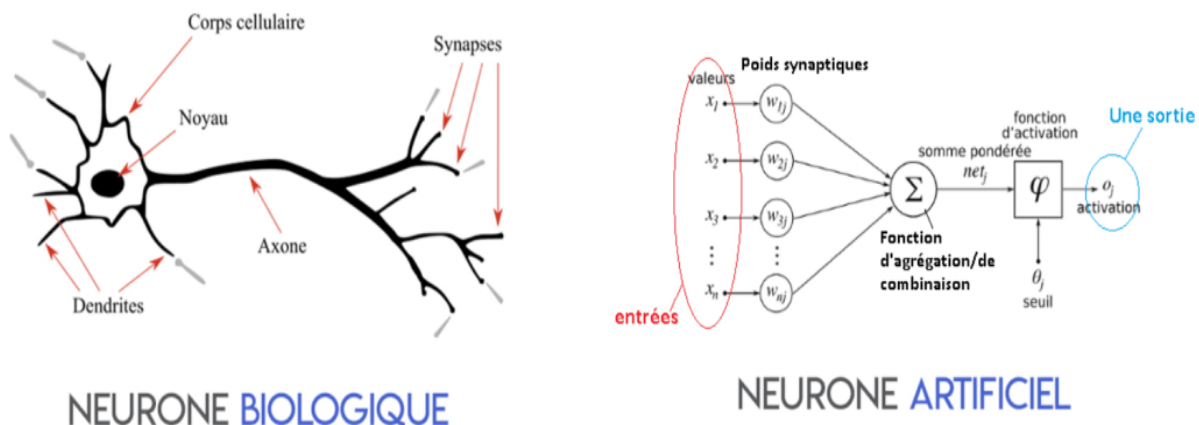


Figure I. 15. Comparaison entre un neurone biologique et un Neurone Artificiel [29].

Les réseaux de neurones artificiels constituent la base de l'apprentissage profond. Les techniques d'apprentissage profond sont également utilisées pour des applications telles que la détection de chute, recommandation de produits, la détection de fraude et la prédiction de l'activité sismique. Parmi ces techniques, on distingue notamment les réseaux de neurones convolutifs (CNN), les réseaux de neurones récurrents (RNN), les réseaux de neurones adverses génératifs (GAN), ainsi que les réseaux de neurones à mémoire à court et long terme (LSTM) [30]. Dans ce travail, nous nous intéressons particulièrement aux réseaux de neurones convolutif (CNN), qui seront détaillés dans la section suivante.

I.6 Réseaux de neurones convolutif

Un réseaux de neurones convolutif est une architecture de réseau de neurones qui utilise des couches convolutives pour extraire automatiquement des caractéristiques pertinentes à partir des données d'entrée, en particulier des images. Il est spécifiquement conçu pour exploiter la structure spatiale des données en utilisant des opérations de convolution pour détecter des motifs locaux [30].

Une architecture CNN, comme illustré à la Figure I.16, est formée par un empilement de couches de traitement indépendantes [32]:

- ✚ **Couche de convolution** : Une couche de convolution est un empilement de convolutions. En effet, l'image est parcourue par plusieurs noyaux de convolution qui donnent lieu à plusieurs cartes de caractéristiques de sorties. Chaque noyau de convolution possède des paramètres spécifiques à l'information qui est recherchée dans l'image. Le choix des paramètres du noyau de convolution dépend de la tâche à résoudre. Dans les approches d'apprentissage profond, ces paramètres ne sont pas définis manuellement, mais sont automatiquement appris par l'algorithme à partir des données d'entraînement.
- ✚ **Couche de sous échantillonnage (POOL)**: L'étape de pooling est une technique de sous-échantillonnage. Généralement, une couche de sous échantillonnage est insérée régulièrement entre les couches de correction et de convolution. En réduisant la taille des cartes de caractéristiques, donc le nombre de paramètres du réseau, cela accélère le temps de calcul et diminue le risque de sur apprentissage.
- ✚ **Couche de correction**: Il est possible d'améliorer l'efficacité du traitement en intercalant entre les couches de traitement une couche qui va opérer une fonction mathématique (fonction d'activation) sur les images de sortie.
- ✚ **Couche entièrement connectée (FC)**: Après plusieurs couches de convolution et de sous-échantillonnage, le raisonnement de haut niveau dans le réseau neuronal se fait

via des couches entièrement connectées. Les neurones dans une couche entièrement connectée ont des connexions vers toutes les sorties de la couche précédente. Leurs fonctions d'activations donc être calculées avec une multiplication matricielle suivie d'un décalage de polarisation.

✚ **Couche de perte (LOSS):** La couche de perte spécifie comment l'entraînement du réseau pénalise l'écart entre le signal prévu et réel. Elle est normalement la dernière couche dans le réseau. Diverses fonctions de perte adaptées à différentes tâches peuvent y être utilisées.

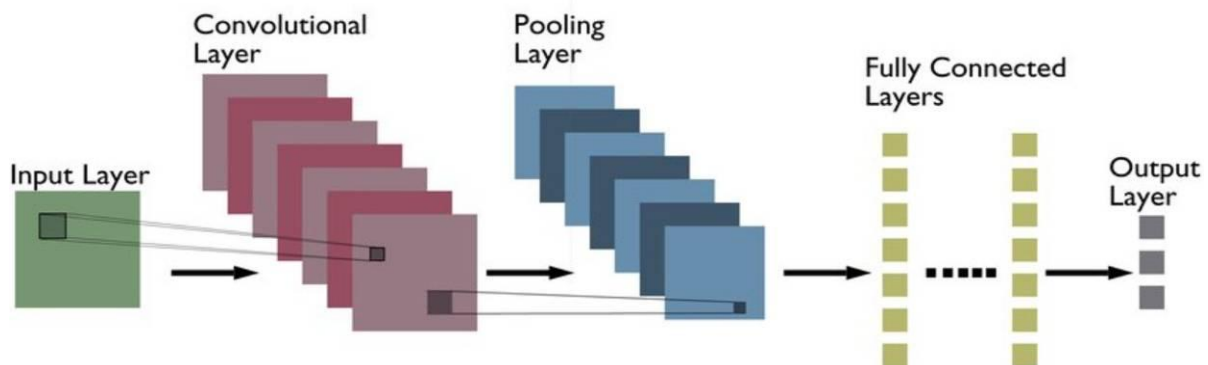


Figure I. 16. Architecture typique du CNN [30].

Les CNN ont trouvé de nombreuses applications dans divers domaines. Voici quelques-unes des principales applications des CNN [30].

- **Classification des images :** Les CNN sont largement utilisés pour la classification des images et peuvent détecter et classer automatiquement les objets présents dans les images.
- **Détection d'objet :** Les CNNs sont utilisés pour détecter et localiser des objets spécifiques dans les images. Cette capacité de détection d'objets est utilisée dans des applications telles que la vidéosurveillance, la conduite autonome et la réalité augmentée.
- **Segmentation sémantique :** CNN est utilisé pour effectuer une segmentation sémantique [30].

I.6.1 Fonctions d'activation

Les fonctions d'activation ont pour objectif d'introduire une couche de non-linéarité entre les couches d'un modèle, permettant ainsi d'apprendre des patterns plus complexes. Il a été démontré qu'en utilisant la fonction d'activation sigmoïde, un modèle à seulement deux couches peut approximer n'importe quelle fonction [33].

On peut diviser ces fonctions d'activation en 3 grosses catégories :

a) Les fonctions d'activations classiques

Ces fonctions d'activation sont parmi les plus anciennes et ont été largement utilisées dans les premiers réseaux de neurones. Elles sont toutes deux lisses et différentiables, ce qui permet de calculer le gradient, c'est-à-dire une mesure qui indiquant la direction et l'amplitude de la variation la plus forte d'une fonction, comme présenté dans le tableau suivant :

Tableau I. 2. Fonction d'activation classique.

Fonction d'activation	Formule mathématique
Sigmoid (ou logistique)	$\text{sigmoid}(x) = \frac{1}{1 + e^{-x}}$
Tanh (ou hyperbolique)	$\tanh(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}}$

b) Fonctions d'activations rectifiées

Quelques types de fonctions d'activation sont présentés dans le Tableau I.3. Ces fonctions d'activation sont largement utilisées dans les architectures modernes d'apprentissage profond (DL), en particulier dans les réseaux de neurones convolutionnels (CNN). Leur popularité s'explique par leur capacité à atténuer le problème du gradient qui disparaît, permettant ainsi la construction de réseaux de neurones beaucoup plus profonds.

Tableau I. 3. Fonctions d'activations rectifiées.

Fonction d'activation	Formule mathématique
ReLU (Rectified Linear Unit)	$\text{ReLU}(x) = \max(0, x)$
Leaky ReLU	$\text{Leaky ReLU}(x) = \begin{cases} x & \text{si } x > 0 \\ \alpha x & \text{sinon} \end{cases}$ où α est un petit coefficient positif.
Parametric ReLU (PReLU)	$\text{PReLU}(x) = \begin{cases} x & \text{si } x > 0 \\ \alpha x & \text{sinon} \end{cases}$ où α est un paramètre appris pendant l'entraînement.
Exponential Linear Unit (ELU)	$\text{ELU}(x) = \begin{cases} x & \text{si } x \geq 0 \\ \alpha(e^x - 1) & \text{sinon} \end{cases}$ où α est un hyperparamètre qui contrôle la pente pour les valeurs négatives

c) Fonctions d'activation pour les sorties

Ces fonctions sont couramment employées dans la couche de sortie des réseaux de neurones pour des tâches de classification. Softmax est spécifiquement utilisée pour les classifications multi-classes, tandis que la fonction Sigmoidale est préférée pour les tâches de classification binaire ou lorsque plusieurs classes peuvent être activées simultanément.

Tableau I. 4. Fonctions d'activation pour les sorties.

Fonction d'activation	Formule mathématique
Softmax	$\text{Softmax}(x_i) = \frac{e^{x_i}}{\sum_{j=1}^k e^{x_j}}$ Cela renormalise les valeurs (x_i) en probabilités, assurant que la somme des probabilités soit égale à 1 pour un ensemble de K classes.
Sigmoid	$\text{sigmoid}(x) = \frac{1}{1 + e^{-x}}$ Utilisée pour des sorties binaires ou dans des contextes multi-classes où une observation peut être associée à plusieurs classes.

I.7 Conclusion

L'étude présentée dans ce chapitre a permis d'établir un panorama complet des approches de télésurveillance et des méthodes de détection de chute. Bien que les solutions existantes offrent des performances intéressantes, elles présentent encore plusieurs limitations, notamment en termes de latence, de dépendance aux infrastructures cloud, de consommation énergétique et de protection des données personnelles.

L'intégration des techniques d'intelligence artificielle, en particulier du *deep learning*, a permis d'améliorer significativement la précision des systèmes de détection. Toutefois, le déploiement de ces solutions demeure limité par les contraintes liées aux ressources matérielles disponibles.

Dans cette perspective, l'adoption d'une approche basée sur l'*Edge AI* apparaît comme une solution pertinente pour répondre aux exigences de traitement en temps réel et de confidentialité. Cette analyse justifie ainsi les choix technologiques adoptés dans ce travail.

Dans le chapitre suivant, nous présenterons les plateformes matérielles et logicielles ainsi que les technologies de communication sélectionnées pour la conception et le développement de notre système de détection de chute.

I.8 Références citées dans le Chapitre I

- [1] V. Barras, L. Duncevic, and V. Christelle, "Records and archives, did you know," *Haute École de Gestion de Genève (HEG-GE), Filière Information documentation*, 2007.
- [2] N. Scholz, "L'Organisation mondiale de la santé au cœur de sa réforme," no. PE 571.307, p. 8, Oct. 2015.
- [3] "Documents fondamentaux," *Comprenant les amendements adoptés jusqu'au 31 décembre 2014*, Organisation mondiale de la santé, 2014.
- [4] M. Grognet, "Le télésoin en orthophonie: état des lieux des pratiques dans le monde et en France (revue de littérature)," *Mémoire de Capacité d'orthophoniste*, 2020.
- [5] L. Y. Djihad, "Bracelet connecté pour la télésurveillance médicale," Mémoire de Master, Spécialité: Réseaux et Télécommunication, Université 8 Mai 1945-Guelma, 2024.
- [6] K. Gallouj, "Quand la mémoire trébuche: Souvenir des chutes dans le vieillissement normal," *Thèse de Doctorat en psychologie*, Dec. 2025.
- [7] N. Haddag, "L'application des Tic dans le domaine de la santé : La télémédecine, cas du CHU Nedir Mohammed de Tizi-Ouzou," Mémoire de Master, Option : Economie de la santé, Université Mouloud Mammeri de Tizi-Ouzou, 2020.
- [8] B. Choufa, "Développement d'une plateforme pour le suivi de l'état de santé des patients à distance," Mémoire de Master, Spécialité: science et technologie de l'information et de la communication, Université 8 Mai 1945-Guelma, 2025.
- [9] R. Amri and R. Badi, "Système d'information pour la télésurveillance médicale," Mémoire de Master, Spécialité: Ingénierie des médias, Université 8 Mai 1945-Guelma, 2011.
- [10] K. Beyaz and A. Benbehaz, "Conception et réalisation d'un système de télésurveillance des patients en temps réel," Mémoire de Master, Option : Electronique des systèmes embarqués, université de Amar Thelidji-Laghouat, 2020.
- [11] M. Bouteaud, "Etats des lieux de la pratique des ergothérapeutes sur les syndromes d'immobilisation chez les personnes âgées en institution," Mémoire de Master, université de Limoges, 2021.
- [12] C. Y. S. Sébastien, "Télésurveillance de paramètres biométriques au service du maintien à domicile de la personne âgée polypathologique : identification des freins et des attentes

- des médecins généralistes par une étude qualitative ancillaire de l'essai clinique contrôlé randomisé eCOBAHLT,” *Thèse de docteur en Médecine*, Nov. 2024.
- [13] C. N-Gayot, “Apports de la télémedecine et de l’intelligence artificielle en gériologie: modélisation du besoin, stratégies de validation et de déploiement à l’échelle d’un territoire rural à faible densité médicale,” *Thèse de Doctorat*, Dec. 2022.
- [14] Y. A. Yahiaoui and S.R. Bourougua, “Réalisation d’un système de surveillance des personnes âgées et isolées,” *Mémoire de Master, Filière: Electronique, Spécialité: instrumentation, Université de Belhadj Bouchaib–Ain Temouchent-*, 2020.
- [15] A. Kabeshova, “Prédire la chute de la personne âgée: apports des modèles mathématiques non-linéaires,” *Thèse de Doctorat*, 2015.
- [16] J. Vaillant, “Kinésithérapie et amélioration du contrôle de l’équilibre du sujet âgé: effets de traitements cognitifs, manuels et instrumentaux,” *Thèse de Doctorat*, 2004.
- [17] L. Abderrazak, “Contribution à la détection de chute des personnes âgées par vidéosurveillance intelligente: L’analyse de la porteuse et l’activité humaine par Machine Learning,” *Thèse de Doctorat, Université de Mohamed V de Rabat*, 2021.
- [18] S. Msaad, “Détection de changement de comportement de vie chez la personne âgée par images de profondeur,” *Thèse de Doctorat*, 2022.
- [19] A. K. Bourke, “A threshold-based fall-detection algorithm using a bi-axial gyroscope sensor,” *G.M. Lyons*, vol. 30, no. 1, 2008.
- [20] M. Kangas, A. Konttila, P. Lindgren, I. Winblad, and T. Jämsä, “Comparison of low-complexity fall detection algorithms for body attached accelerometers,” *Gait & Posture*, vol. 28, no. 2, pp. 285–291, 2008.
- [21] A. Randrianarisaina, O. Pasquier, and P. Chargé, “Energy consumption modeling of smart nodes with a function approach,” *In Proceedings of the 2014 Conference on Design and Architectures for Signal and Image Processing*, pp. 1–6, 2014.
- [22] J. S. Madhubala, A. Umamakeswari, and B. J. A. Rani, “A survey on technical approaches in fall detection system,” *National Journal of Physiology, Pharmacy and Pharmacology*, vol. 5, no. 4, p. 275, 2015.
- [23] I. Halima, “Suivi de l’activité d’une personne à partir de capteurs multi-modalités préservant l’anonymat dans un cadre de détection et prévention des chutes chez les personnes âgées,” *Thèse de Doctorat, Université de Rennes I*, 2021.
- [24] A. S. Ykhlef and A. Lasla, “Mise au point d’une ‘Application de télésurveillance,’” *Mémoire de Master, Spécialité: Informatique, Université Abou Bekr Belkîd -Tlemcen*, 2014.

- [25] I. Benameur and A. Tazghat, "Installation et analyse d'un système de vidéosurveillance au niveau de la bibliothèque de l'insi," *Mémoire de Master, Filière: hygiène et sécurité industrielle, Spécialité: Sécurité prévention intervention, Université Mouhamed Ben Ahmed-d'Oran 2*, 2020.
- [26] J. N. Kaci and A. Khemisi, "Etude et mise en place d'un système de vidéosurveillance IP à l'ENIEM," *Mémoire de Master, Spécialité : Télécommunication et réseaux, Université Mouloud Mammeri de Tizi-Ouzou*, 2016.
- [27] F. Z. Guerrouj, "Méthodologie d'optimisation de la configuration matérielle-logicielle pour la détection d'objets dédiée aux véhicules autonomes," *Thèse de Doctorat, Spécialité : Robotique*, 2025.
- [28] D. Houfani, "Une approche basée sur les SMA et méta-heuristique pour la prédiction (PHM) médicale," *Thèse de doctorat en Informatique, Option: Intelligence Artificielle, Université de Mohammed Khider de Biskra*, 2022.
- [29] F. Malki, "Conception et évaluation d'un système d'IA embarqué basé sur les techniques d'apprentissage pour la carte Nvidia Jetson Nano," *Mémoire de Master, Spécialité: Electronique des systèmes embarqués, Université Mouloud Mammeri de Tizi-Ouzou*, 2024.
- [30] H. Boubidi, "Du visage vers une image de qualité Naturelle," *Mémoire de Master, Spécialité: Informatique, Option : Science et technologie de l'information et de la communication, Université 8 Mai 1945-Guelma*, 25 juin 2023.
- [31] K. Maizi, "Génértion d'image faciale synthétique à base de points caractéristiques," *Mémoire de Master, Filière: Informatique, Option : Science et technologie de l'information, Université 8 Mai 1945-Guelma*, 2023.
- [32] M. Ameer and A. Kelfi, "Classification des images utilisant les réseaux de neurones de convolution," *Mémoire de Master, Filière : Electronique Spécialité : Industrie électronique (MCIL), Université de Mohamed El-Bachir-Ibrahimi de Bordj Bou Arreridj*, 2020.
- [33] T. Oriol, "Détection des collemboles au microscope optique: utilisation du deep learning pour une analyse automatisée," *Thèse de Doctorat*, 2024.

Chapitre II

*Plateforme matérielle,
logicielle et technologies de
communication*

II.1 Introduction

La conception d'un système de télésurveillance intelligent repose sur une intégration harmonieuse entre les composantes matérielles et logicielles. Le choix de la plateforme embarquée constitue un élément déterminant, notamment dans le cadre des applications basées sur l'Edge AI, où le traitement des données doit être effectué localement et en temps réel.

Ce chapitre est consacré à la description détaillée de l'architecture matérielle et logicielle adoptée dans ce travail. Il présente les caractéristiques de la plateforme embarquée utilisée, les dispositifs de capture vidéo, ainsi que les technologies de communication permettant l'échange des données. Par ailleurs, les outils logiciels et les environnements de développement sont également décrits.

II.2 Choix du matériel

Comme indiqué dans le chapitre I, nous avons identifié plusieurs contraintes à prendre en compte lors du développement de notre système. Celui-ci doit notamment être économiquement réaliste et reposer sur des capteurs ambiants afin de ne pas perturber la personne surveillée.

Ainsi, les systèmes utilisant des capteurs portés par la personne, tels que les accéléromètres, n'ont pas été retenus. De même, les capteurs au sol ont été écartés, car ils ne constituent pas encore une solution facilement intégrable dans un environnement existant.

Les capteurs environnementaux et les caméras correspondent en revanche comme des solutions adaptées. Il est possible d'installer des caméras au domicile de la personne sans la gêner et à un coût relativement faible. Par ailleurs, une autre solution complémentaire et moderne consiste à développer une application Android destinée à la personne âgée. Cette application permet d'enregistrer des informations personnelles, ainsi que des contacts d'urgence à alerter en cas de chute. Elle peut également être connectée au réseau domestique afin de garantir la sécurité des données et le respect de la vie privée.

Nous avons donc opté pour l'utilisation de caméras permettant, comme nous le montrerons, de détecter l'activité de la personne, notamment en vue de la détection des chutes. Les caméras RGB, notamment les caméras de surveillance IP, ont été retenues, car elles fournissent des informations riches sur les objets en plus de l'image couleur. De plus, elles sont facilement disponibles sur le marché et présentent un coût relativement faible.

Comme l'illustre la Figure II.1, notre système est basé sur la vision. Il nécessite l'utilisation d'une caméra RGB ou d'une caméra de type IP, ainsi qu'une plateforme embarquée pour la

détection des chutes à l'aide de techniques d'intelligence artificielle. À cet effet, nous avons choisi une carte performante : Nvidia Jetson Nano, adaptée au traitement embarqué et à l'exécution d'algorithmes de Deep Learning.

Par ailleurs, une application Android est développée afin de faciliter l'utilisation du système par la personne âgée. Cette application permet également d'exploiter les fonctionnalités du smartphone pour l'envoi d'appels ou de SMS d'urgence. Ainsi, le réseau GSM du téléphone est utilisé pour assurer la transmission des alertes en cas d'urgence.

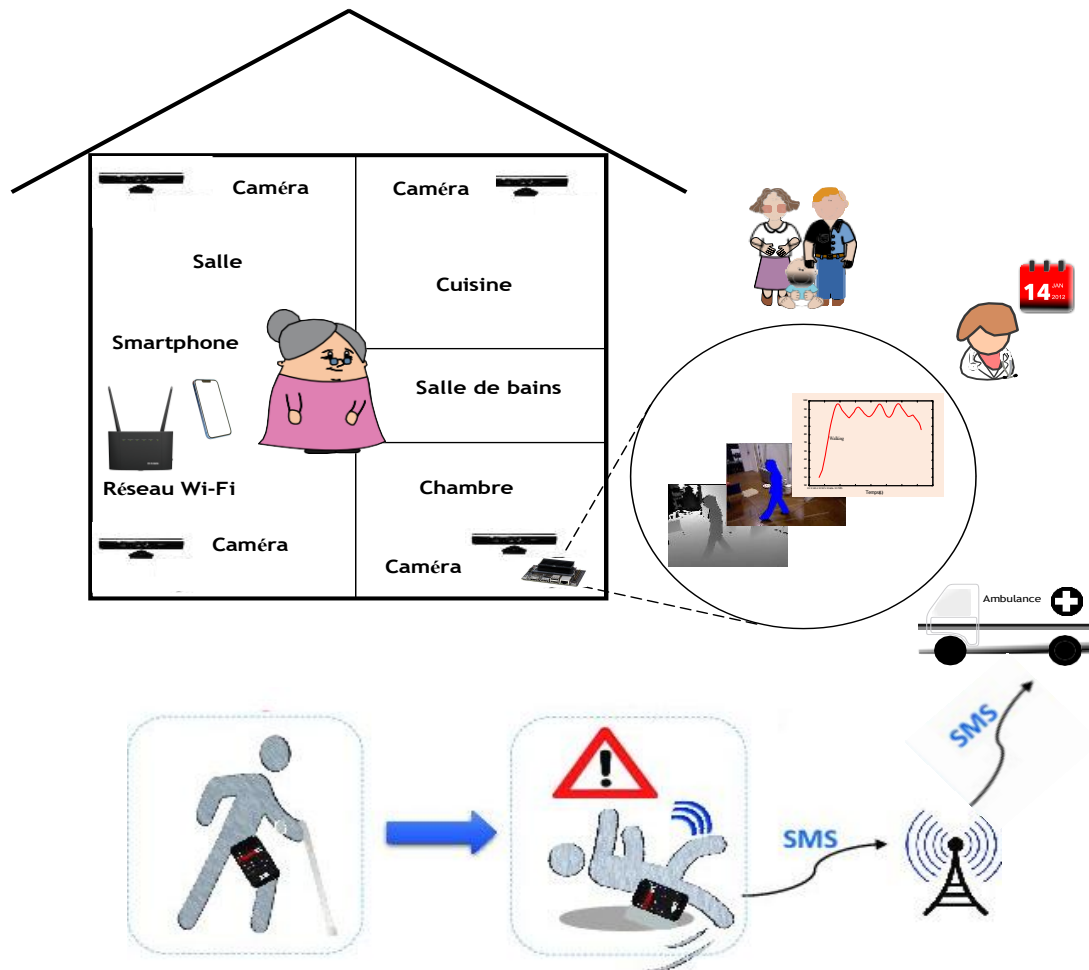


Figure II. 1. Vision pour la conception d'un système permettant le maintien à domicile des personnes âgées.

Dans les sections suivantes, nous présentons la plateforme matérielle du système, composée de la carte Nvidia Jetson Nano, de la caméra RGB ou IP, ainsi que du réseau domestique WiFi. Cette partie permet de comprendre la liaison entre les différentes composantes matérielles, telles que la carte de traitement Nvidia, la caméra et le smartphone.

Par ailleurs, lorsque le réseau GSM est utilisé, la section suivante fournit une description générale des technologies de communication utilisées, en particulier le réseau GSM.

Enfin, la dernière partie est consacrée à la plateforme logicielle, incluant la programmation de la carte Nvidia en Python ainsi que la construction de modèles d'intelligence artificielle à l'aide de Google Colab.

II.3 Plateforme matérielle

II.3.1 Carte Nvidia Jetson Nano

Le dispositif Nvidia Jetson Nano, représenté sur la Figure II.2, est un mini-ordinateur puissant, qui a été lancé par NVIDIA (Nvidia Corporation, États-Unis) en mars 2019.

La carte Nvidia Jetson Nano représente une solution innovante spécialement conçue pour répondre aux exigences croissantes de l'intelligence artificielle en périphérie de réseau, ou Edge. Son architecture compacte et performante lui confère un intérêt majeur pour les applications nécessitant une capacité de traitement locale, immédiate et sécurisée.

On peut fixer une caméra sur la carte Nvidia Jetson Nano, on peut fabriquer un appareil intelligent ou réaliser un système de sécurité intelligent et fiable, comme on peut construire des programmes tels que la vision par ordinateur ou la vision d'intelligence en appliquant des bibliothèques d'apprentissage automatique et d'apprentissage profond directement en local [1].



Figure II. 2. Carte Nvidia Jetson Nano.

II.3.1.1 Présentation de la carte Nvidia Jetson Nano

Le Nvidia Jetson Nano s'agit d'un petit ordinateur puissant qui permet d'exécuter plusieurs réseaux neuronaux en parallèle pour des applications afin de mettre en œuvre l'analyse d'image tel que la classification, la détection d'objets, la segmentation ou le traitement de la voix [2].

En outre, il dispose d'un connecteur d'entrée/sortie général à 40 broches (GPIO) entièrement configurable, permettant de connecter et de contrôler facilement divers périphériques externes [3].

La carte est équipée d'un processeur cadencé à 1,43 GHz et d'un GPU de 128 cœurs basé sur l'architecture Maxwell. La Figure II.3-(a) présente le premier modèle de Nvidia Jetson Nano, appelé Jetson Nano A02. Par la suite, Nvidia a introduit une version améliorée, le Nvidia Jetson Nano B01, illustrée à la Figure II.3-(b), qui est celle utilisée dans ce travail de détection de la chute.

D'un point de vue technique, les deux modèles possèdent le même CPU et le même GPU ; mais certains périphériques et interfaces ont été améliorés dans la version B01 [4].

Au cours des dernières années, Nvidia a également introduit d'autres cartes plus avancées, telles que la série Jetson Xavier, offrant des performances supérieures pour les applications d'intelligence artificielle, mais à un coût relativement élevé.

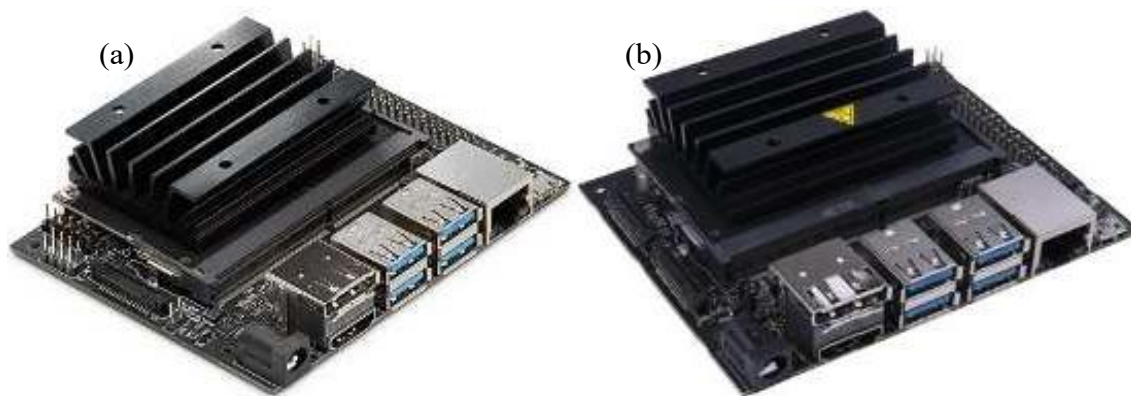


Figure II. 3. (a) Nvidia Jetson Nano A02, (b) Nvidia Jetson NanoB01.

II.3.1.2 Caractéristiques principales de la carte Nvidia Jeston Nano

La carte Nvidia Jetson Nano est une plateforme embarquée offrant un bon compromis entre performance, consommation énergétique et coût, ce qui la rend particulièrement adaptée aux systèmes embarqués, notamment pour des applications telles que la détection de chute.

Les principales caractéristiques matérielles de cette carte sont les suivantes :

-  **Stockage** : emplacement pour carte microSD utilisé comme mémoire principale.
-  **Connectivité GPIO** : équipée de plusieurs ports de connectivité, y compris quatre ports USB 3.0, un port Gigabit Ethernet, un port HDMI et un connecteur pour caméra CSI (Camera Serial Interface). Ces options de connectivité facilitent l'intégration avec divers périphériques et accessoires.

- ✚ **Alimentation** : via port micro-USB (5 V) ou prise DC (jack).
- ✚ **Réseau** : port Ethernet Gigabit pour une connexion stable.
- ✚ **Ports USB** : plusieurs ports USB 3.0 pour connecter des périphériques externes.
- ✚ **Affichage** : sortie HDMI et DisplayPort pour écran externe.
- ✚ **Caméra** : connecteur MIPI CSI compatible avec les caméras haute performance.
- ✚ **Processeur** : ARM Cortex-A57 quadricœur cadencé à 1,43 GHz.
- ✚ **GPU** : architecture Maxwell avec 128 cœurs CUDA.
- ✚ **Mémoire** : 4 Go LPDDR4.
- ✚ **Support logiciel** : Nvidia fournit un ensemble complet d'outils logiciels pour le développement sur la Jetson Nano, y compris le kit de développement logiciel (SDK) JetPack, qui comprend des bibliothèques d'apprentissage profond telles que TensorFlow, PyTorch et Caffe, ainsi que des bibliothèques de vision par ordinateur comme OpenCV [5].

Les principales caractéristiques techniques de cette carte sont les illustrées dans le Tableau II.1.

Tableau II. 1. Spécifications techniques de la carte Nvidia Jetson Nano.

Caractéristique	Spécification
CPU	ARM Cortex-A57 quad-core @ 1,43 GHz
GPU	128 cœurs Maxwell
Mémoire	4 Go LPDDR4 (64 bits, 25,6 GB/s)
Stockage	Carte microSD
Encodage vidéo	4K @ 30 ; 4×1080p @ 30 ; 9×720p @ 30 (H. 264/H. 265)
Décodage Vidéo	4K @ 60 ; 2×4K @ 30 ; 8×1080p @ 30 ; 18×720p @ 30 (H.264/H.265)
Caméra	Connecteur MIPI CSI-2
Connectivité	Gigabit Ethernet et 802.11ac sans fil
Afficher	HDMI et port d'affichage
USB	1×USB 3.0 Type A, 2×USB 2.0 Type A, 1×USB 2.0 Micro-B
I/O	GPIO. I2C. I2S. SPI. UART
Dimensions	69 mm×45 mm

La Figure II.4 présente les différents composants de la carte Nvidia Jetson Nano.

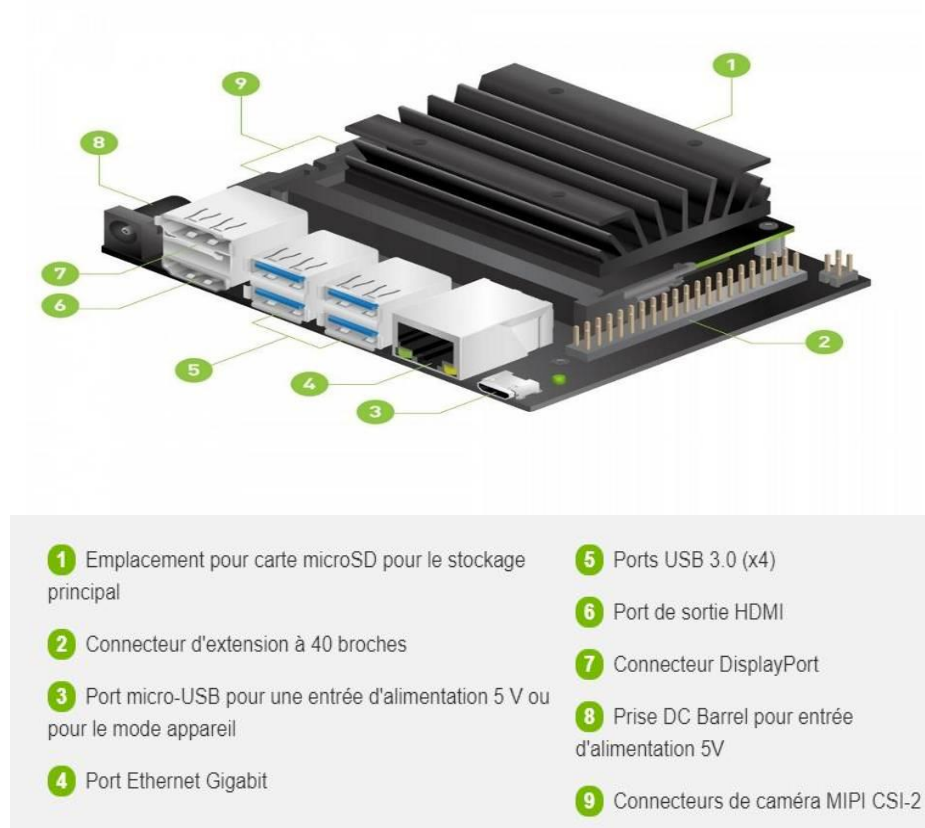


Figure II. 4. Schéma descriptif des différents composants de Jetson Nano de Nvidia [5].

II.3.1.3 Applications de Nvidia Jetson Nano

De plus l'utilisation de la carte NVIDIA Jetson dans le domaine de l'intelligence artificielle, elle peut être utilisée pour des activités informatiques quotidiennes.

Quelques applications de Nvidia Jetson Nano :

- Naviguer sur Internet pour rechercher des informations.
- Rédaction de documents.
- La création de feuilles de calcul et l'impression de documents.
- Installer des applications bureautiques à partir de libre office.
- Joindre des caméras externes via l'interface CSI ou une caméra USB.
- Utiliser le Jetson Nano pour créer d'excellents programmes de vision par ordinateur.

Dans le but de réaliser des applications de l'IA, le Nvidia Jetson prend en charge plusieurs bibliothèques telle que Pandas, Numpy, Tensorflow, Keras et openCV qui fournit diverses bibliothèques de traitement d'images et de vidéos pour la vision par ordinateur, et peut être utilisée sur des programmes tels que C/C++ et Python. Pour obtenir des calculs optimisés, il faut s'assurer que ces bibliothèques et les programmes prennent en charge les cœurs de GPU de la carte NVIDIA Jetson Nano.

II.3.2 Caméra

Pour la capture du flux vidéo, nous avons utilisé deux types de caméras : “Havit-HV-N5080” et “Caméra Imou IPC-G26EP”.

L'utilisation de ces caméras permet de transformer la carte Nvidia Jetson Nano en une plateforme de traitement capable de fonctionner comme une webcam, une caméra IP ou un système de vidéosurveillance, comme illustré dans les Figures II.5 et II.6.

✚ Caméra “Havit-HV-N5080”

- Résolution Full HD 1080 p.
- Trépied de support convenant à l’affichage LCD ou CRT d'un ordinateur de bureau ou d'un ordinateur portable.
- Prise en charge des ordinateurs courants supérieurs à XP, 2000, vista, Win7, Win8, Win10 et autres systèmes d'exploitation.
- La longueur de câble 1.5m.
- Sensibilité : 830 V/lux-sec@550nm.
- Gamme dynamique : 68db@8X Gain.
- Rapport signal/bruit : 45db.
- Éclairage minimum : 0.01lux.
- Résolution supportée : 640*480 1280**720 1920*1080 2592*1944.
- Capteur d'image : COMS.
- Vitesse : 30 fps/s(VGA).
- Type de connexion : USB2.0 haute vitesse (Isochrone).
- Format de sortie : MJPEG/YUV2(YUYV).



Figure II. 5. Caméra Havit (Hv-N 5080).

Caméra “IPC-G26EP-IMOU”

La caméra Imou IPC-G26EP est une caméra IP de type bullet, conçue pour la vidéosurveillance extérieure avec connectivité Wi-Fi. Elle offre des fonctionnalités avancées de détection intelligente et de dissuasion active, adaptées aux systèmes de sécurité et de surveillance en temps réel.



Figure II. 6. Caméra Imou (IPC-G26EP-IMOU).

Les caractéristiques techniques :

- Marque : Imou by Dahua.
- Modèle : IPC-G26EP.
- Capteur d'image : CMOS 1/2.7" 2 mégapixels.
- Résolution : Full HD 1080p (1920 × 1080) à 25 images/seconde.
- Compression vidéo : H.265/H.264.
- Angle de vision : 114,7° (horizontal), 62° (vertical), 135° (diagonal).
- Vision nocturne : infrarouge jusqu'à 30 mètres.
- Zoom numérique : 16×.
- Connectivité : Wi-Fi IEEE 802.11 b/g/n et Ethernet 100 Mbps.
- Audio : bidirectionnel (microphone et haut-parleur intégrés).
- Détection : capteur PIR (détection de mouvement).
- Dissuasion active : sirène intégrée 110 dB + lumière de dissuasion.
- Stockage : carte microSD jusqu'à 128 Go.
- Compatibilité : ONVIF, CGI, application mobile Imou Life (Android / iOS).
- Alimentation : 12V DC / 1A.

- Consommation : ≤ 6.5 W.
- Dimensions : $79 \times 79 \times 166.2$ mm.
- Poids : 411 g.

II.4 Technologies de communication sans fil

Dans le cadre de ce travail de détection de chute des personnes âgées, les technologies de communication sans fil jouent un rôle essentiel pour assurer une transmission fiable et rapide des données entre les différents composants du système. Elles permettent notamment l'échange d'informations telles que les flux vidéo en temps réel, les résultats de traitement ainsi que les alertes en cas de chute.

L'architecture global de notre système est composée de plusieurs équipements connectés au même réseau Wi-Fi, notamment : une caméra de surveillance IP assurant la capture des vidéo en temps réel; une unité de traitement basée sur la carte Nvidia Jetson Nano, utilisée pour l'analyse des données et la détection des chutes à l'aide de méthodes d'intelligence artificielle; ainsi qu'un smartphone destiné à la réception des notifications et à l'envoi des alertes vers les contacts d'urgence.

La classification des réseaux sans fil peut être établie selon leur périmètre géographique, appelé zone de couverture. On distingue ainsi quatre catégories : les réseaux personnels sans-fil, réseaux locaux sans fil, réseaux métropolitains sans fil et réseaux étendus sans fil [6], [7].

Ces technologies peuvent être classées en quatre parties :

- Les réseaux personnels sans fil : WPAN (**W**ireless **P**ersonal **A**rea **N**etwork) ;
- Les réseaux locaux sans fil : WLAN (**W**ireless **L**ocal **A**rea **N**etwork) ;
- Les réseaux métropolitains sans fil : WMAN (**W**ireless **M**etropolitan **A**rea **N**etwork) ;
- Les larges réseaux sans fil : WWAN (**W**ireless **W**ide **A**rea **N**etwork).



Figure II. 7. Les différentes technologies sans fil.

Par ailleurs, les réseaux sans fil peuvent également être classés en fonction de leurs usages spécifiques et des applications qu'ils supportent. À titre d'exemple [6], on peut citer :

- Les réseaux d'entreprises ;
- Les réseaux domestiques ;
- Les réseaux de capteurs (Sensor Networks) ;
- Les réseaux tactiques (Tactical Networks) ;
- Les réseaux pervasifs (Pervasive Networks) ;
- L'informatique portative (Wearable Computing) ;
- Les réseaux des véhicules automatisés (Automded Vehical Networks).

Ainsi, notre travail s'inscrit dans la catégorie des réseaux locaux sans fil (WLAN), avec une application de type domestique, dédiée à la surveillance et à la sécurité des personnes âgées.

Dans les sections suivantes, nous nous appuyons sur un réseau local sans fil (WLAN) basé sur la technologie Wi-Fi, ainsi que sur les protocoles de communication TCP/IP, qui constituent le socle de notre architecture. Par ailleurs, le protocole WebSocket (WS) sera utilisé afin de permettre un échange de données en temps réel entre un client, dans notre cas le smartphone, et un serveur représenté par la carte Nvidia Jetson Nano. Cette approche permet d'assurer une communication sans fil efficace, fiable et en temps réel entre les différents équipements du système, déployés dans un environnement domestique.

II.4.1 Réseau local sans fil (WLAN) basé sur la technologie Wi-Fi

Le WiFi est une technologie standard d'accès sans fil à des réseaux locaux WLAN. Le principe consiste à établir des liaisons radio rapides entre des terminaux et des bornes reliées aux réseaux à haut débit. Grâce à ces bornes WiFi, l'utilisateur se connecte à Internet ou au système d'informations de son entreprise et accède à de nombreuses applications reposant sur le transfert de données. Cette technologie a donc une réelle complémentarité avec les réseaux ADSL (Asymetric Digital Subscriber Line) ainsi que les réseaux d'entreprise (comme illustré dans la Figure II.8).

Ce standard a été développé pour favoriser l'interopérabilité du matériel entre les différents fabricants ainsi que pour permettre de futures évolutions compatibles avec la norme. Ainsi, les consommateurs peuvent mélanger des équipements de différents fabricants afin de satisfaire leurs besoins de connexion, ce qui a permis à cette norme un développement conséquent qui lui vaut d'être la plus utilisée actuellement et d'être la plus fournie par tous les fabricants d'équipements de réseaux et de téléphonie [8].

Notre système utilise un routeur domestique comme infrastructure réseau locale afin d'assurer la connectivité Wi-Fi entre le dispositif de détection de chute et les services distants. Le routeur permet la transmission des données et des alertes en temps réel vers les applications mobiles ou les plateformes cloud.

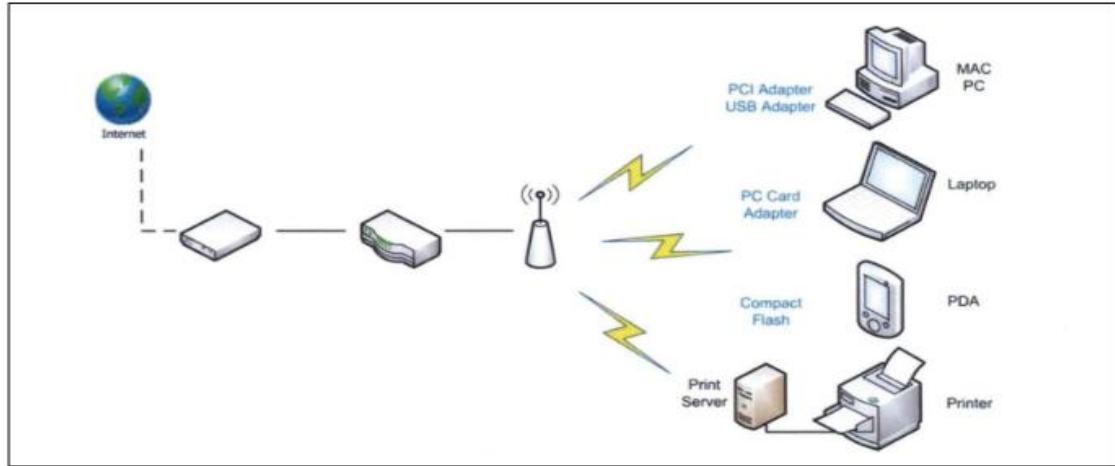


Figure II. 8. Connexion sans fil.

II.4.2 Protocoles de communication TCP/IP

Le modèle TCP/IP est un ensemble de couches de communication permettant l'interconnexion des réseaux informatiques et la transmission des données sur internet. Il constitue l'architecture de base des communications réseau modernes. Ce modèle est souvent comparé au modèle OSI, mais il est plus simple puisqu'il est composé de quatre (04) couches: la couche Application, la couche Transport, la couche Internet et la couche Accès Réseau.

Historiquement, le modèle TCP/IP a été développé dans les années 1970 par le département de la défense des États-Unis (DOD : Department Of Defense).

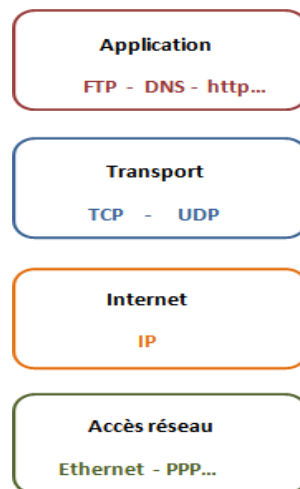


Figure II. 9. Le modèle TCP/IP.

Les différentes couches du modèle TCP/IP sont décrites comme suit :

a) Couche Application

Cette couche assure la communication entre les applications réseau et l'utilisateur. Elle fournit les services nécessaires aux applications telles que :

- La transmission des données ;
- L'encodage et le formatage des informations ;
- La gestion des sessions et de l'authentification ;
- La gestion des noms de domaine.

Les principaux protocoles utilisés dans cette couche sont : HTTP/HTTPS, FTP, SMTP, DNS, MQTT dans les applications IoT.

Dans notre projet de détection de chute, cette couche permet l'envoi des alertes et des données vers l'application mobile ou le serveur de supervision.

b) Couche Transport

La couche Transport assure une transmission fiable des données entre les équipements du réseau. Ses principales fonctions sont :

- Segmentation des données pour le transfert sur le réseau.
- Gestion de la fiabilité de la transmission des données.
- Gestion du contrôle de flux pour éviter la congestion du réseau.
- Gestion du multiplexage et du démultiplexage des données.
- Exemple de protocoles utilisés par cette couche : TCP – UDP.

Dans notre système, le protocole TCP est utilisé afin de garantir la transmission fiable des alertes de chute.

c) Couche Internet

Cette couche est responsable :

- Gestion du routage des paquets sur le réseau.
- Fragmentation et réassemblage des paquets si nécessaire.
- Attribution et gestion des adresses IP.
- Gestion du protocole de communication IP.
- Elle permet de déterminer les meilleurs chemins à travers les réseaux en fonction des adresses IP.
- Utilise le protocole: IP (Internet Protocol).

Dans notre projet, cette couche n'a pas été exploitée directement. Cependant, comme perspective future, elle pourra être utilisée pour assurer l'acheminement des données entre le dispositif IoT et les équipements connectés au réseau Wi-Fi.

d) Couche Accès Réseau

Cette couche gère l'accès au support physique de transmission et la communication avec le réseau matériel. Elle assure :

- Couche Accès Réseau.
- Encapsulation des données.
- Gestion de l'accès au support de transmission physique.
- Détection et correction d'erreurs.
- Gestion de la topologie réseau.

Dans notre système, cette couche correspond principalement au réseau Wi-Fi utilisé par le routeur domestique.

Lorsqu'une information est envoyée sur le réseau, les données traversent successivement les différentes couches du modèle TCP/IP au niveau de l'émetteur. À chaque couche, des informations de contrôle sont ajoutées sous forme d'en-têtes : ce mécanisme est appelé encapsulation.

À la réception, le processus inverse est effectué afin de récupérer les données originales; cette opération est appelée décapsulation [9].

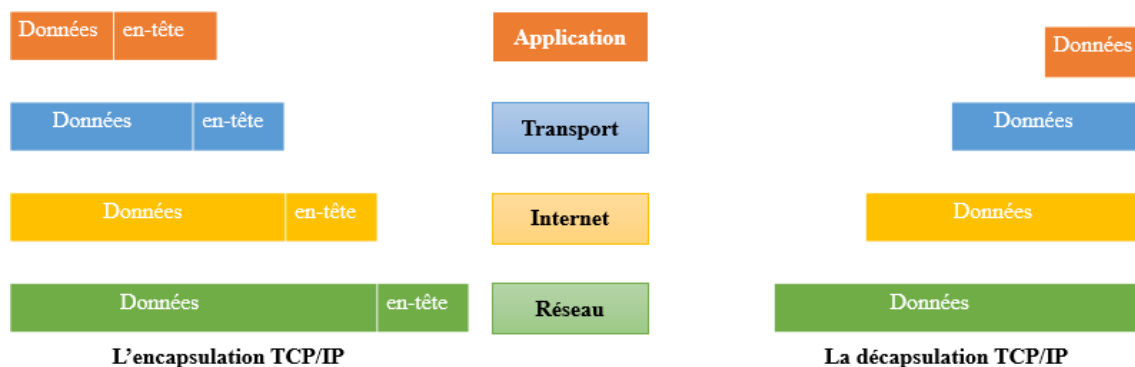


Figure II. 10. Encapsulation et la décapsulation du modèle TCP/IP.

II.4.3 WebSocket

WebSocket est un protocole de communication permettant d'établir une connexion bidirectionnelle permanente entre un client et un serveur à travers les ports standards du Web. Contrairement au protocole HTTP classique basé sur le modèle requête/réponse, WebSocket permet un échange de données en temps réel sans nécessiter l'envoi répétitif de requêtes par le client [10].

Grâce à ce protocole, le serveur peut envoyer directement des informations au client selon le principe du push, ce qui le rend particulièrement adapté aux applications temps réel telles que: les systèmes IoT, les applications de surveillance, les systèmes d'alerte et de télésurveillance.

Les navigateurs modernes compatibles avec HTML5 intègrent nativement le support du protocole WebSocket.

Le client initie une requête HTTP auprès du serveur afin de demander l'ouverture d'une connexion WebSocket. Après acceptation de cette requête par le serveur via une opération d'upgrade du protocole, un canal de communication bidirectionnel permanent est établi entre les deux entités.

Une fois la connexion établie, le client et le serveur peuvent échanger des messages à tout moment sans réinitialiser la connexion.

Dans notre projet de détection de chute, WebSocket peut être utilisé pour transmettre instantanément les alertes entre la carte Nvidia Jetson Nano et l'application mobile de supervision.

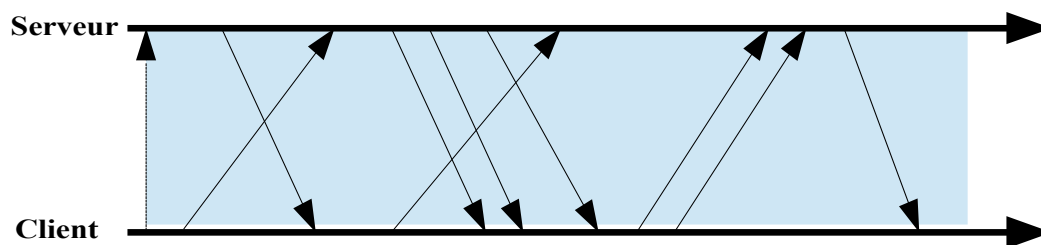


Figure II. 11. Communication en temps réel entre le serveur (Jetson Nano) et le client (smartphone) via WebSocket.

Le protocole WebSocket présente plusieurs avantages : faible latence, transmission en temps réel, communication bidirectionnelle continue, réduction du trafic réseau par rapport aux requêtes HTTP répétitives.

Ces caractéristiques rendent WebSocket particulièrement adapté aux applications de surveillance intelligente et aux systèmes de détection de chute nécessitant une réaction rapide en cas d'urgence.

II.5 Technologie GSM

Une autre technologie utilisée dans ce travail est la technologie GSM. Plusieurs méthodes peuvent être employées pour assurer une communication GSM, par exemple l'utilisation d'un shield Arduino GSM tel que SIM900. Dans notre cas, nous exploitons les fonctionnalités du téléphone mobile en utilisant le réseau GSM intégré. Cette solution présente l'avantage de réduire le nombre de composants électronique nécessaires, tout en offrant une solution fiable et facile à mettre en œuvre.

Le réseau étendu sans fil (WWAN) est également connu sous le nom de réseau cellulaire mobile. Il s'agit des réseaux sans fil les plus répandus puisque tous les téléphones mobiles sont connectés à un réseau étendu sans fil.

Les principales technologies associées sont les suivantes [7], [11]:

- **GSM** (*Global System for Mobile Communication*) : Ce standard a été initialement conçu pour les applications de transport de la parole, de transmission de données à bas débits (9600 bits/s) autour des fréquences 935MHz- 960MHz ou 890-915MHz. Il constitue la base du développement des communications mobiles modernes.
- **GPRS** (*General Packet Radio Service*) : Cette technologie est la première évolution de la norme GSM pour pouvoir atteindre des débits compatibles avec des applications à débits moyens de type multimédia.
- **UMTS** (*Universal Mobile Telecommunication System*) : Cette norme UMTS a pour but de développer une meilleure couverture radio et d'augmenter le nombre d'abonnés par unité de surface et promouvoir l'acheminement des services troisième Génération (3G). L'UMTS permet un débit théorique jusqu'à 2 Mbit/s.

Dans les sous-sections suivantes, nous donnons un aperçu général du réseau GSM afin de mieux comprendre son principe de fonctionnement et ses principales caractéristiques.

II.5.1 Réseau GSM

Le GSM est un système de radiotéléphonie cellulaire numérique, qui offre à ses abonnés des services qui permettent la communication de station mobile de bout en bout à travers le réseau [7].

Le réseau GSM constitue au début du 20e siècle, le standard de téléphonie mobile le plus utilisé. La norme GSM autorise un débit maximal de 9.6kbps, ce qui permet de transmettre la voix ainsi que des données numériques de faible volume, par exemple des messages textes ou des messages multimédias [6].

II.5.1.1 Architecture du réseau GSM

L'architecture de base du système GSM prévoit quatre sous-systèmes principaux, dont chacun dispose d'un certain nombre d'unités fonctionnelles et est connecté à l'autre à travers des interfaces standard qui seront décrites ultérieurement. Les principaux sous-systèmes du réseau GSM sont représentés dans la Figure II.12 [12]. On distingue ainsi :

- La station mobile (MS) ;
- Le sous-système radio (BSS) ;
- Le sous-système réseau (NSS) ;
- Le sous-système d'exploitation et de maintenance (OSS).

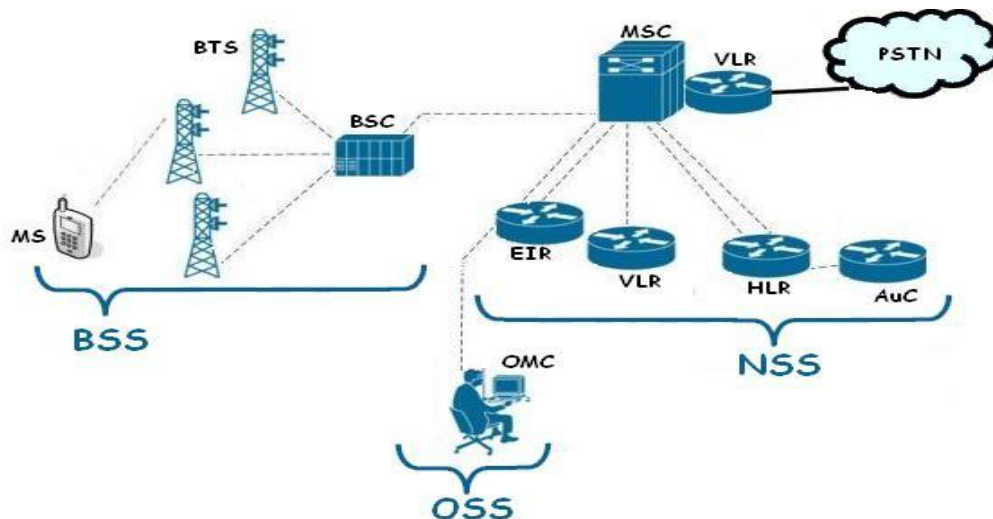


Figure II. 12. Architecture du réseau GSM [6].

- **BSS (Base Sub-system)**: sous système radio, sa fonction principale est la gestion de l'attribution des ressources radio, indépendamment des abonnés, de leur identité ou de leur communication.
- **NSS (Network Sub-System)** : le sous système d'acheminement, il assure principalement les fonctions de commutation et de routage. C'est donc lui qui permet l'accès au réseau public RTCP (**R**éseau **T**éléphonique **C**ommuté **P**ublic) ou RNIS (**R**éseau **N**umérique à **I**ntégration de **S**ervices). En plus des fonctions indispensables de commutation, on y retrouve les fonctions de gestion de la mobilité, de la sécurité et de la confidentialité qui sont implantées dans la norme GSM.
- **OSS (Opération Sub-System)** : le sous système d'exploitation et maintenance, Il assure la gestion et la supervision du réseau. C'est la fonction dont l'implémentation est laissée avec le plus de liberté dans la norme GSM. La supervision du réseau intervient à de nombreux niveaux [7], notamment : la détection de pannes, la mise en service de sites, la modification de paramétrage, et la réalisation de statistiques.

II.5.1.2 Services offerts par un réseau GSM

Les principaux services fournis par le GSM à ses abonnés se répartissent en trois catégories, qui sont :

a) Les services supports : Ils permettent les transferts de données de bout en bout à travers le réseau. Un service support particulier s'identifie par ses attributs. La norme GSM définit trois catégories, qui sont :

- De transfert d'information (mode de transfert circuit ou paquet, débit de transfert,

type d'information,.....).

- D'accès (canal et débit d'accès, commerciaux,....). Généraux (qualité de services, commerciaux,...).

b) Les télé-services : Ils constituent des applications opérationnelles offertes par le réseau à ses abonnés, ces derniers utilisent les possibilités offertes par les services supports. Les principaux services offerts par GSM sont [13]:

- La téléphonie (entre deux poste mobile ou un poste mobile et un poste fixe).
- Les messages courts (SMS).
- Le fax.

II.5.1.3 Principe de fonctionnement du GSM

Les antennes-relais font surtout référence à l'univers de la téléphonie mobile. Alors que le téléphone classique permet de convertir la voix en champs de radiofréquences, ces antennes-là réceptionnent ce signal électromagnétique, pour le convertir en signal électrique.

On parle de téléphone cellulaire, car chaque antenne couvre une partie de territoire, représentant une cellule. Sans ces émetteurs-récepteurs, constitués par les antennes-relais, la téléphonie mobile n'existerait pas. Il existe trois catégories d'antennes :

- Les antennes pico-cellulaires qui ont une couverture de quelques dizaines de mètres.
- Les antennes micro-cellulaires qui ont une couverture de quelques centaines de mètres.
- Les stations macro-cellulaires, qui sont les plus répandues. Elles sont généralement placées sur des pylônes ou des toits d'immeubles.

En milieu rural, ces antennes auront un champ de couverture de 10 à 30 *km*, (on limite le nombre de fréquences utilisatrices), tandis qu'en milieu urbain, la zone de couverture est limitée à 500 *m* et la puissance est partagée sur de nombreuses bandes de fréquences utilisatrices [14].

GSM utilise une combinaison d'accès multiple par répartition dans le temps (TDMA) et d'accès multiple par répartition en fréquence (FDMA).

✚ **Frequency Division Multiple Access (FDMA) :** Il consiste à diviser une bande de fréquences en plusieurs bandes de telle sorte que chaque bande de fréquences subdivisée soit attribuée à un seul abonné. Dans le GSM, l'accès multiple par répartition en fréquence divise la largeur de bande de 25 *Mhz* en 124 fréquences porteuses, espacées chacune de 200 *KHz*. Chaque station de base se voit attribuer une ou plusieurs fréquences porteuses.

✚ **Time Division Multiple Access (TDMA) :** Il s'agit d'attribuer le même canal de fréquences à différents abonnés en divisant la bande de fréquences en plusieurs tranches

de temps. Chaque utilisateur obtient son propre intervalle de temps, ce qui permet à plusieurs stations de partager le même espace de transmission.

Dans le cas du GSM, chaque fréquence porteuse subdivisée est divisée en différents intervalles de temps à l'aide de la technique TDMA. Chaque trame TDMA dure 4,164 millisecondes (*ms*) et contient 8 intervalles de temps. Chaque tranche de temps ou un canal physique dans cette trame dure 577 microsecondes et les données sont transmises dans la tranche de temps sous forme de rafales [15].

II.6 Plateforme logicielle

II.6.1 Python

Python est un langage de programmation puissant et facile à apprendre. Il été développé depuis 1989 par GuidoVan Rossumet .Il présente des structures de données de haut niveau et permet une approche simple et efficace de la programmation orientée objet [16].

Le langage de programmation Python (le dernier Python 3) est utilisé dans le développement Web, les applications d'apprentissage automatique, ainsi que dans toutes les technologies de pointe de l'industrie logicielle. Le langage de programmation Python convient très bien aux débutants, ainsi qu'aux programmeurs expérimentés dans d'autres langages de programmation comme C++ et Java [17].



Figure II. 13. Logo Python [16].

II.6.2 Pytorch

PyTorch est une bibliothèque open source d'apprentissage automatique en Python, développée par Facebook. Elle permet d'effectuer les calculs tensoriels nécessaires pour l'apprentissage profond, optimisés pour être réalisés par le processeur central (CPU) ou le processeur graphique (GPU), si disponible. Le CPU (Central Processing Unit), ou processeur central, est le composant principal de l'ordinateur responsable de l'exécution des instructions des programmes. Il réalise les calculs de base et gère les tâches générales de traitement des données. En revanche, le GPU (Graphics Processing Unit), ou processeur graphique, est spécialement conçu pour gérer des calculs complexes liés à l'affichage graphique [18]



Figure II. 14. Logo Pytorch [18].

II.6.3 OpenCV

OpenCV (Open Source Computer Vision Library) est une bibliothèque logicielle open source de vision par ordinateur et d'apprentissage automatique [17]. Cette bibliothèque est écrite en C et C++ et peut être utilisée sous Linux, Windows et Mac OS. Des interfaces ont été développées pour Python, Ruby, Matlab et autre langage. Open CV Est orienté vers des applications en temps réel. Un des buts d'OpenCV est d'aider les gens à Construire rapidement des applications sophistiquées de vision à l'aide d'infrastructure simple de vision par ordinateur [18].

La bibliothèque compte plus de 2500 algorithmes optimisés, ce qui inclut un ensemble complet d'algorithmes de vision par ordinateur et d'apprentissage automatique classiques et de pointe. Ces algorithmes peuvent être utilisés pour détecter et reconnaître des visages, identifier des objets, classer des actions humaines dans des vidéos, suivre les mouvements de la caméra, suivre des objets en mouvement, extraire des modèles 3D d'objets,... etc [17].

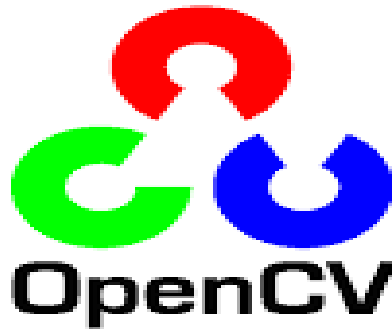


Figure II. 15. Logo OpenCV [17].

II.6.4 Google Colab

Google Colab est un service cloud de recherche, fourni par Google, de développement et de formation en machine learning. Il offre un environnement Jupyter Notebook qui ne nécessite aucune configuration et qui s'exécute entièrement dans le cloud.

Google Colab permet d'écrire et d'exécuter du code Python directement depuis un navigateur web, ainsi que de sauvegarder et partager facilement les notebooks. Il offre également un accès gratuit à des ressources matérielles puissantes, notamment des processeurs graphiques

(GPU) et des unités de traitement tensoriel (TPU), ce qui accélère considérablement les calculs nécessaires à l'entraînement des modèles d'apprentissage automatique [18].

Grâce à ces fonctionnalités, Google Colab est largement utilisé dans les projets de machine learning et de data science, en particulier pour le développement et l'entraînement de modèles complexes.



Figure II. 16. Logo Google colab [18].

Que nous offre Colab ? Nous pouvons tirer profit les avantage suivants en utilisant Google Colab [19]:

- Écrire et exécuter du code en Python.
- Documentation du code pour soutenir les équations mathématiques.
- Facilite la Création, téléchargement et le partager des carnets et l'importation des ensembles de données (ex : a partir de Google).
- Intégrer Python, TensorFlow, Keras, et OpenCV.
- Service Cloud gratuit avec GPU gratuit.



Figure II. 17. Environnement de Google Colab [19].

II.6.5 Android Studio

Android Studio est un environnement de développement pour développer des applications Android. Il est basé sur IntelliJ IDEA. Il permet principalement d'éditer les fichiers Java et les fichiers de configuration d'une application Android.

Il permet notamment d'éditer le code source (Java et/ou Kotlin), de gérer les fichiers de configuration et de concevoir les interfaces utilisateur grâce à un éditeur graphique intuitif. Android Studio propose également des fonctionnalités avancées telles que la gestion des ressources multilingues, l'émulation de différents appareils Android et la visualisation des interfaces sur des écrans de résolutions variées [20].

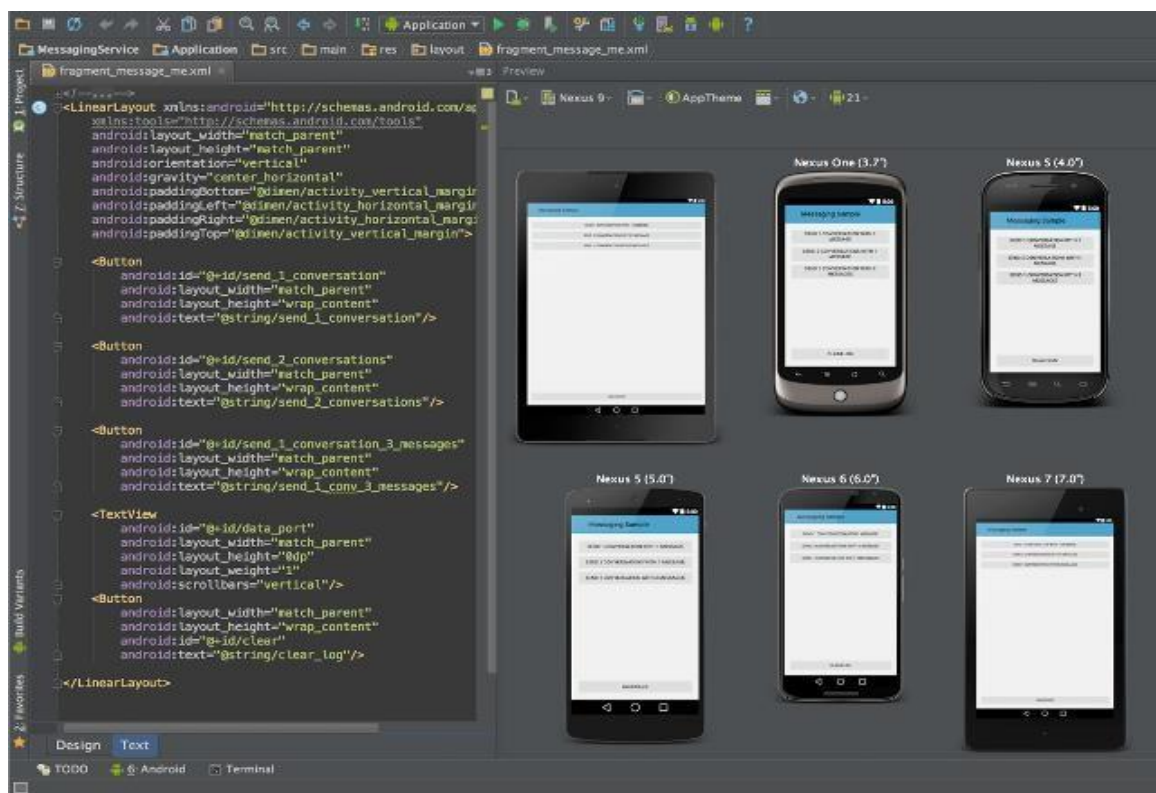


Figure II. 18. Android studio [20].

Grâce à ses outils intégrés de débogage, de compilation et d'analyse des performances, Android Studio facilite le développement d'applications mobiles performantes et adaptées à différents environnements.

Dans le cadre de notre travail, nous avons développé une application Android destinée à un smartphone afin de faciliter son utilisation par la personne âgée. Cette application offre plusieurs fonctionnalités, notamment l'envoi de SMS, ainsi que la surveillance en temps réel via le téléphone. Plusieurs langages de programmation peuvent être utilisés pour le développement Android, tels que Java et Kotlin. Dans notre cas, nous avons privilégié le langage Kotlin en raison de sa simplicité, de sa concision et de sa facilité d'utilisation.

II.6.6 Kotlin

Kotlin est un langage de programmation orienté objet et fonctionnel, avec un typage statique qui permet de compiler pour la machine virtuelle Java, JavaScript, et vers plusieurs plateformes. Son développement provient principalement d'une équipe de programmeurs chez JetBrains basée à Saint-Petersbourg en Russie (son nom vient de l'île de Kotlin, près de St. Pétersbourg).

Google annonce pendant la conférence Google I/O 2017 que Kotlin devient le second langage de programmation officiellement pris en charge par Android après Java. Le 8 mai 2019, toujours lors de la conférence Google I/O, Kotlin devient officiellement le langage de programmation voulu et recommandé par le géant américain Google pour le développement des applications Android.

Pivotal Software annonce le 4 janvier 2017 le support officiel de Kotlin sur la cinquième version du Framework Spring.



Figure II. 19. Logo de Kotlin

Kotlin est totalement compatible avec l'ancien langage JAVA qui est l'un le plus utilisé dans le développement Android et backend, cela signifie que les projets existant déjà en JAVA peuvent facilement intégrer du code Kotlin sans réécriture complète.

Et aussi on peut utiliser les bibliothèques JAVA en Kotlin ce qui donne la facilité d'utilisation de Kotlin.

Kotlin est plus concise que JAVA, le code généralement nécessite moins de ligne pour accomplir les mêmes tâches qu'en JAVA ou d'autre langage.

La concision de Kotlin permet un code plus lisible, maintenable, et moins sujet aux erreurs humaines.

L'une des principales causes de bugs dans Java est l'exception Null Pointer Exception (NPE), qui se produit lorsque l'on tente d'accéder à une référence d'objet null. Kotlin introduit une gestion stricte de la nullabilité, en forçant les développeurs à gérer explicitement les références nulles.

Ce système réduit considérablement le risque de bugs liés à la gestion des nulls, offrant une plus grande sécurité et stabilité du code.

Kotlin est considéré comme facile à apprendre, surtout pour les développeurs Java, car sa syntaxe est simple et son modèle de programmation est intuitif. Il élimine beaucoup de complexités que les développeurs rencontrent souvent en Java [21].

II.7 Conclusion

Ce chapitre a permis de définir l'ensemble des composantes matérielles et logicielles nécessaires à la mise en œuvre du système de télésurveillance. La plateforme embarquée retenue se distingue par sa capacité à exécuter des algorithmes complexes d'intelligence artificielle tout en respectant les contraintes de performance et de consommation.

Les technologies de communication sélectionnées assurent une transmission efficace et sécurisée des données, tandis que l'environnement logiciel offre une grande flexibilité pour le développement et l'intégration des modèles.

Ces choix constituent une base robuste pour l'implémentation des méthodes d'intelligence artificielle, qui seront abordées dans le chapitre suivant.

II.8 Références citées dans le Chapitre II

- [1] Y. Zhang, J. Yu, Y. Chen, W. Yang, and W. Zhang, "Real-time strawberry detection using deep neural networks on embedded system (rtsd-net): An Edge AI application," *Computers and Electronics in Agriculture*, vol. 192, 2022.
- [2] R. G. Manso, C. J. G. Orellana, H. M. G. Velasco, and M. M. Macías, "Towards selective and automatic harvesting of broccoli for agri-food industry," *Computers and Electronics in Agriculture*, vol. 188, p. 8, 2021.
- [3] F. E. Khoury and A. Zgheib, *Building a dedicated GSM GPS module tracking system for fleet management: hardware and software*. CRC Press, 2018.
- [4] B. Beldjilali and A. Belkhiter, "Intégration d'un service de localisation GSM/GPS," *Mémoire de Master, Université Abou Bakr Belkaïd -Tlemcen*, 2013.
- [5] F. Malki, "Conception et évaluation d'un système d'IA embarqué basé sur les techniques d'apprentissage pour la carte NVIDIA Jetson Nano," *Mémoire de Master, Filière : Génie Electrique, Spécialité : Electronique des Systèmes Embarqués, Université Mouloud Mammeri de Tizi-Ouzou*, 2023.
- [6] K. Belhadi and A. Abidi, "Etude et réalisation d'un réseau WiFi hotspot dans le service public," *Mémoire de Master, Option : Contrôle, Université Mouloud Mammeri de Tizi-Ouzou*, 2011.

- [7] A. Beladelli and M. Oukaz, “Dimensionnement D’un Réseau Sans Fil Wifi,” Mémoire de Master, Option : systèmes des télécommunications, Université Abou Bekr Belkaid - Tlemcen, 2011.
- [8] A. R. Rebai, “Contributions à l’amélioration de la qualité de service dans les réseaux sans fil Wifi,” Thèse de Doctorat, Spécialité Automatique et Informatique des Systèmes Industriels et Humains, Université de Valenciennes et du Hainaut-Cambrésis, 2009.
- [9] S. Guadir and C. Guerrah, “Etude de la téléphonie sur IP,” Mémoire de Master, Filière : Télécommunications, Spécialité : Réseaux et Télécommunications, Université Mouloud Mammeri de Tizi-Ouzou, 2024.
- [10] J. Bousquié, “Websocket,” In : *JRES (Journées réseaux de l’enseignement et de la recherche)*, 2013.
- [11] W. Aboura and I. Benhabib, “Etude et caractérisation de la couche physique du standard IEEE802.16/WIMAX,” Mémoire de Master, Option : Réseaux Mobiles et Service (RMS), Université Abou Bakr Belkaid -Tlemcen, 2012.
- [12] K. Meziane and F. Azem, “Etude et réalisation du Roaming international dans le GSM (Mobilis),” Mémoire de Master, Option : Communication, Université Mouloud Mammeri de Tizi-Ouzou, 2008.
- [13] A. Slimani, “Dimensionnement et Planification d’un Réseau GSM,” Mémoire de Master, Option : Systèmes et Réseaux, Université Abou Bakr Belkaid -Tlemcen, 2010.
- [14] F. Krief, *Les systèmes embarqués communicants : mobilité, sécurité, autonomie*, Hermes Science Publications, Série réseaux et télécoms, Lavoisier librairie, 2008.
- [15] T. B. Eddine and B. Ismail, “Étude comparative pour l’optimisation d’un modèle de propagation approprié pour la communication mobile,” Mémoire de Master, Université de M’sila, 2019.
- [16] H. Boubidi, “Du Visage Vers Une Image De Qualité Naturelle,” Mémoire de Master, Spécialité : Informatique, Option: Science et technologie de l’information et de la communication, Université du 8 mai 1945-Guelma, 2023.
- [17] F. Mesbah, “Détection d’objets par Deep Neural Network à l’aide du modèle YOLO en temps réel,” Mémoire de Master, Filière : Informatique, Option : Sciences et Technologie de l’Information et de la Communication, Université 8 Mai 1945-Guelma, 2020.
- [18] T. Rahmani and F. Sadi, “Système embarqué pour la détection d’objets basé sur YOLOv5,” Mémoire de Master, Filière : Electronique, Spécialité : Electronique des Systèmes Embarqués, Université Mouloud Mammeri de Tizi-Ouzou, 2023.

- [19] W. Benkamouch, “Détection d’objets et deep Learning dans un trafic routier,” Mémoire de Master, Spécialité: Informatique, Option: sciences et technologies de l’information et de La communication, Université 8 Mai 1945-Guelma, juin2023.
- [20] K. Djaouti and D. Amenouche, “Conception et réalisation d’une application Anroid de géolocalisation des stations-service de NAFTAL,” Mémoire de Master, Filière : Informatique, Spécialité : Ingénierie des Systèmes d’Information, Université Mouloud Mammeri de Tizi-Ouzou, 2025.
- [21] N. Mansouri and S. Amnache, “Implémentation d’un logiciel GMAO dans une entreprise de travaux publics,” Mémoire de Master, Filière : Electromécanique, Spécialité : Maintenance Industrielle, Université Mouloud Mammeri de Tizi-Ouzou, 2023.

Chapitre III

*Méthodes d'intelligence
artificielle utilisées*

III.1 Introduction

L'intelligence artificielle occupe aujourd'hui une place essentielle dans les systèmes modernes de surveillance intelligente et dans le domaine de la santé connectée. Grâce aux avancées du Deep Learning et particulièrement des réseaux neuronaux convolutifs (CNN), il est désormais possible d'analyser automatiquement des images et des flux vidéo afin de détecter avec précision des événements critiques, tels que les chutes des personnes âgées ou des patients nécessitant une surveillance continue.

Dans le contexte de la télésurveillance médicale, ces technologies permettent de développer des systèmes intelligents capables d'assurer une surveillance en temps réel, tout en réduisant les délais d'intervention et les risques liés aux accidents domestiques. Les méthodes de détection d'objets basées sur l'intelligence artificielle offrent également une meilleure précision et une plus grande robustesse face aux variations d'éclairage, de position ou d'environnement.

Dans ce chapitre, nous présentons les différentes méthodes d'intelligence artificielle utilisées dans notre système de télésurveillance basé sur la carte Nvidia Jetson Nano. Nous commençons par introduire les modèles de détection d'objets, en particulier les approches modernes fondées sur les réseaux neuronaux convolutifs (CNN). Ensuite, nous abordons les algorithmes de détection d'objets les plus connus ainsi que l'évolution de la famille YOLO (You Only Look Once).

Une attention particulière est portée au modèle YOLOv5, choisi pour notre système en raison de sa rapidité et de sa précision dans les applications temps réel. Nous détaillons son architecture, ses différentes composantes, les variantes disponibles, ainsi que les étapes d'entraînement et les métriques utilisées pour évaluer ses performances.

Enfin, nous présentons la mise en œuvre du système intelligent de télésurveillance des personnes âgées, en décrivant la base de données utilisée, le processus d'entraînement du modèle YOLOv5, puis les résultats expérimentaux obtenus et leur discussion. Ce chapitre permet ainsi de mettre en évidence l'apport des techniques d'intelligence artificielle dans la conception d'un système de détection de chute fiable, rapide et adapté aux contraintes des systèmes embarqués.

III.2 Modèles de la détection d'objet

Comme indiqué dans les chapitres I et II, l'objectif principal de ce travail est de construire un système robuste de détection des chutes basé sur la vidéo, en s'appuyant sur des modèles de détection d'objets.

Dans le domaine de la vision par ordinateur, plusieurs techniques sont utilisées, parmi lesquelles se trouve la détection d'objets. Ce domaine de recherche est particulièrement actif en raison de sa combinaison de la classification et de la localisation, et par extension, de l'identification, en utilisant différents algorithmes liés aux modèles de détection d'objets.

Le principe de la détection d'objets, ou des termes alternatifs tels que la reconnaissance d'objets, la classification d'images, la localisation d'objets et la segmentation d'images, repose sur la capacité à identifier et à localiser des objets spécifiques dans des images ou des vidéos. Pour obtenir une méthode efficace de détection d'objets, il est essentiel de disposer à la fois d'un algorithme robuste de détection des régions et d'un algorithme précis de classification d'images.

Il existe différents algorithmes pour la détection d'objets, certains sont des techniques classiques tandis que d'autres sont des techniques avancées développées récemment [1].

III.2.1 Modèles modernes de détection basés sur les réseaux neuronaux convolutifs (CNN)

En apprentissage profond, un réseau de neurones convolutifs (CNN) constituent une classe de modèles principalement utilisés pour analyser les images visuelles. Contrairement aux réseaux de neurones classiques, les CNN utilisent une technique spéciale appelée convolution, qui est une opération mathématique sur deux fonctions produisant une troisième fonction. Cela permet d'exprimer comment la forme de l'une est modifiée par l'autre [1].

Les méthodes modernes de détection d'objets basées sur les CNN se divisent principalement en deux catégories :

- **Détection en deux étapes** : cette approche regroupe des modèles bien connus tels que R-CNN (Region-based Convolutional Neural Network), Fast R-CNN et Faster R-CNN. Le processus se déroule en deux phases distinctes: la proposition initiale de régions et la classification suivie du raffinement de la localisation des objets. Cette approche permet un ajustement plus précis des positions des objets détectés, ce qui peut améliorer la précision, mais elle peut aussi être plus lente en raison de ces deux étapes séparées.
- **Détection en une seule étape** : cette catégorie inclut des modèles comme MultiBox, Attention-Net, G-CNN, YOLO (You Only Look Once), SSD (Single Shot Detector), etc., combine la localisation et la classification des objets en une seule étape. Cela rend cette méthode généralement plus rapide, car il n'y a pas de phase distincte de proposition de régions.

III.2.1.1 Algorithmes de détection d'objet par CNN

Dans un cas de détection d'objet, il pourrait y avoir de nombreuses boîtes englobantes représentant différents objets d'intérêt dans l'image, et nous ne pourrions pas savoir combien il y en a à l'avance.

La principale raison pour laquelle nous ne pouvons pas résoudre ce problème en construisant un réseau convolutif standard suivi d'une couche entièrement connectée est que la longueur de la couche de sortie est variable, c'est parce que le nombre d'occurrences des objets d'intérêt est non fixé. Une approche pour résoudre ce problème serait de prendre différentes régions d'intérêt de l'image et d'utiliser un CNN pour classer la présence de l'objet dans cette région. Le problème avec cette approche est que les objets d'intérêt peuvent avoir différents emplacements spatiaux dans l'image et différents rapports d'aspect. Par conséquent, nous devons sélectionner un grand nombre de régions et cela pourrait faire beaucoup trop en termes de calcul. Par conséquent, des algorithmes comme R-CNN, YOLO, etc. ont été développés pour trouver ces occurrences rapidement [1].

Les figures suivantes illustrent les principales méthodes de détection en deux étapes : R-CNN [2], Fast R-CNN [3] et Faster R-CNN [4].

Malgré leurs performances, ces approches présentent certaines limitations :

- ✚ **R-CNN** : Ce modèle souffre d'un temps de calcul très élevé, en raison du traitement indépendant de chaque région proposée. Il nécessite également une importante capacité mémoire et repose sur un pipeline complexe comportant plusieurs étapes distinctes (extraction de caractéristiques, classification et régression).
- ✚ **Fast R-CNN** : Bien qu'il améliore significativement la vitesse et la précision par rapport à R-CNN, Fast R-CNN dépend encore de la méthode de recherche sélective (Selective Search) pour générer les régions d'intérêt. Par conséquent, il reste peu adapté aux applications en temps réel.
- ✚ **Faster R-CNN** : Cette méthode réduit considérablement le temps de détection grâce à l'introduction du réseau de propositions de régions (RPN). Toutefois, elle présente une complexité d'entraînement plus élevée, nécessitant souvent une stratégie d'apprentissage alternée ou multitâche. De plus, son temps de traitement, bien qu'amélioré, peut encore être insuffisant pour certaines applications en temps réel strictes.

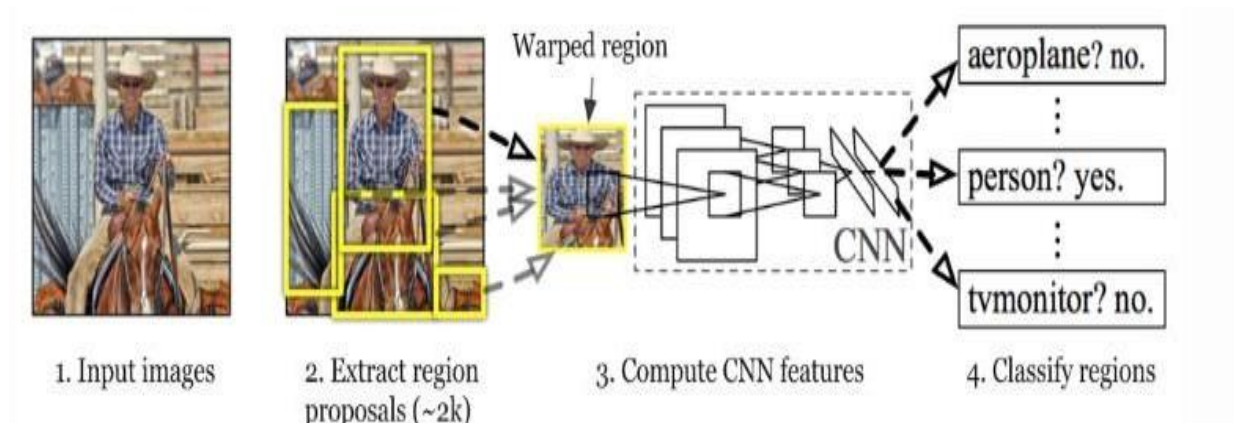


Figure III. 1. Réseaux de neurones convolutifs basés sur les régions (R-CNN) [2].

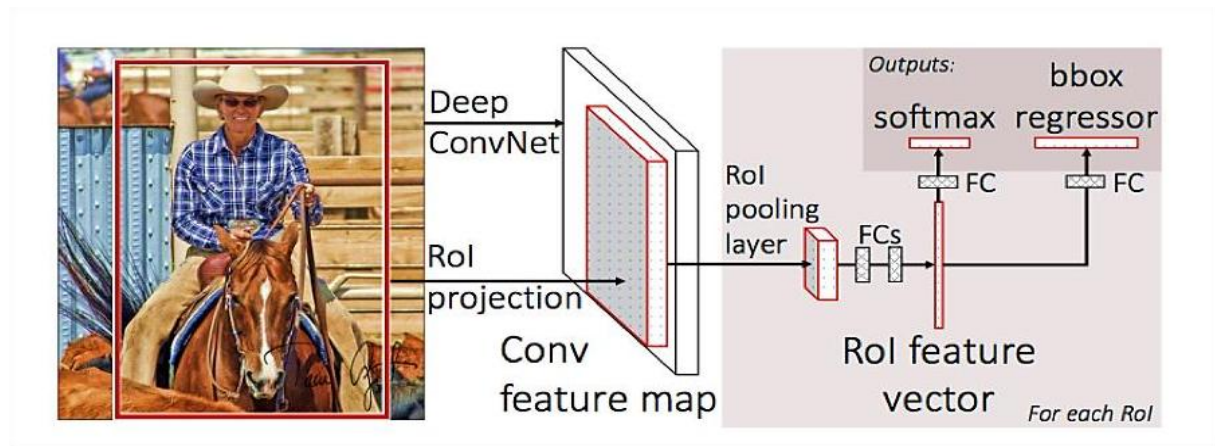


Figure III. 2. Fast R-CNN [3].

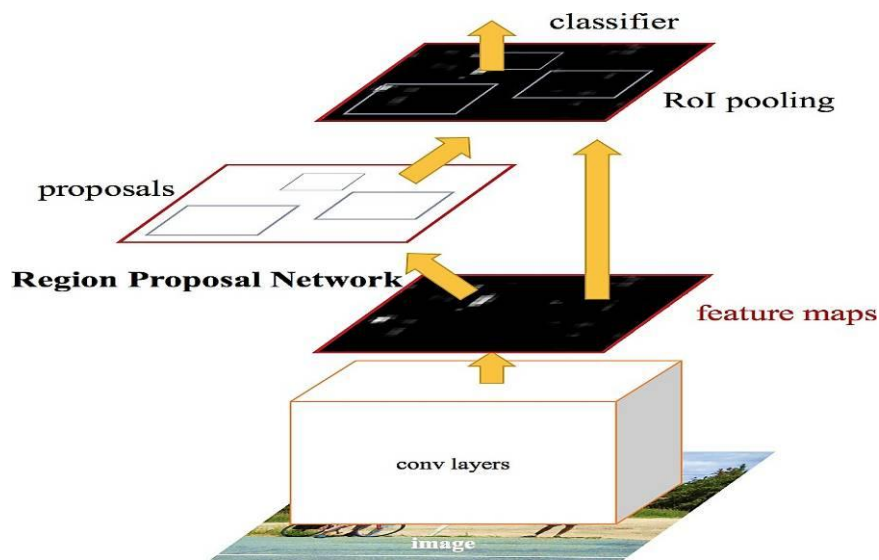


Figure III. 3. Faster R-CNN [4].

Bien que ces algorithmes constituent des solutions performantes pour la détection d'objets avec une grande précision, leurs exigences élevées en termes de calcul et de temps de traitement limitent leur utilisation dans des applications pratiques à grande échelle ou en temps réel, telles que la détection des chutes envisagée dans ce travail.

Face à ces contraintes, des modèles dits one-stage, notamment ceux de la famille des YOLO (You Only Look Once), se sont imposés comme des alternatives efficaces. Ces approches se distinguent leur rapidité d'exécution, leur simplicité d'implémentation et leur capacité à offrir de bonnes performances de détection.

III.2.2 YOLO (You Only Look Once)

Les méthodes classiques de détection d'objets reposent généralement sur l'utilisation de propositions de régions afin de localiser les objets dans une image. Le réseau ne regarde pas l'image complète mais plutôt les parties de l'image qui ont de fortes probabilités de contenir l'objet [5].

YOLO (You Only Look Once) est un algorithme de détection d'objets basé sur un réseau neuronal profond, introduit en 2015 par ses auteurs [6]. Contrairement aux approches de type R-CNN, qui détectent des objets en parcourant l'image en différentes étapes, YOLO quant à lui ne regarde l'image qu'une seule fois (d'où son nom) ce qui accélère de manière conséquente tout le processus de détection et de localisation d'un objet sur cette même image. Les R-CNN étant composés de plusieurs éléments affectés à une tâche dédiée sont ainsi plus difficiles à optimiser car chacun de ses composants doit alors être entraîné séparément ce qui peut rendre la tâche fastidieuse. Dans le cas de YOLO, toutes les opérations s'effectuent au sein d'un seul et même réseau de neurones [7].

YOLO s'est rendu célèbre avec une rapidité de détection sans égal avec un traitement de l'image allant jusqu'à 45 images/seconde dans sa première version (155 pour sa version réduite/light mais moins précise en termes de précision) [7]. Comme illustré dans la Figure III.4, le principe de l'algorithme YOLO repose sur une approche unifiée de la détection d'objets, où la localisation et la classification sont réalisées simultanément en une seule passe du réseau.

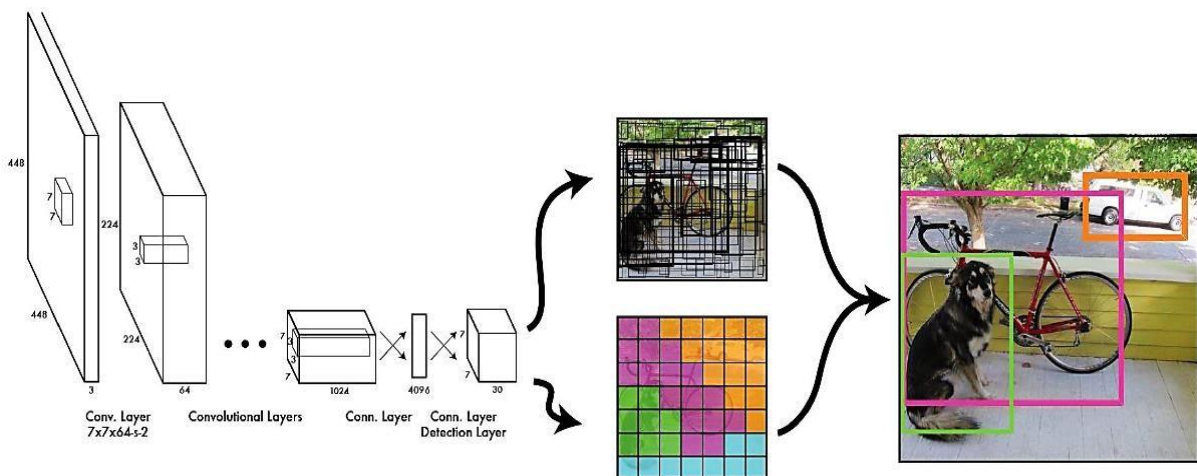


Figure III. 4. Principe de l'algorithme de détection d'objets «YOLO» [6].

III.2.2.1 Evolution de YOLO

Depuis l'apparition du modèle YOLO en 2016, celui-ci n'a cessé d'évoluer, avec des versions successives de plus en plus performantes et robustes (Tableau III.1).

Tableau III. 1. Évolution des différentes versions du modèle YOLO [8].

Différentes versions	Année de publication	Améliorations
YOLO	2015	Modèle initial
YOLOv2	2016	Ce modèle provient du modèle initial mais a été amélioré en ajoutant la normalisation des lots, les boîtes d'ancrage et les grappes de dimensions.
YOLOv3	2018	Ce modèle a amélioré les performances du modèle en employant un réseau dorsal plus efficace, des ancres multiples et surtout une mise en commun des pyramides spatiales.
YOLOv4	2020	Cette version a introduit des innovations telles que l'augmentation des données Mosaic, une nouvelle tête de détection sans ancrage et une nouvelle fonction de perte.
YOLOv5	2020	Ce modèle a amélioré les performances en ajoutant de nouvelles fonctionnalités telles que l'optimisation des hyperparamètres, le suivi intégré des expériences et l'exportation automatique vers les formats d'exportation.
YOLOv6	2022	Ce modèle est utilisé dans plusieurs robots de livraison autonomes de l'entreprise.
YOLOv7	2022	Cette version a ajouté des tâches additionnelles à savoir l'estimation de la pose sur le jeu de données COCO keypoints.

YOLOv8	2023	Ce modèle a ajouté des fonctionnalités en plus et des améliorations pour accroître les performances, la flexibilité et surtout l'efficacité, en utilisant l'ensemble de tâches d'IA.
YOLOv9	2023	Il insère de nouvelles méthodes telles que le réseau d'agrégation de couches efficace généralisé (GELAN) et l'information programmable de gradient (PGI).
YOLOv10	2024	Ce modèle présente des avancées sur la détection d'objet en temps réel en introduisant une tête de bout en bout qui élimine les exigences en termes de suppression non maximale (NMS).
YOLOv11	2024	Cette version offre des performances de pointe (SOTA) dans de plusieurs tâches, en plus de la segmentation, la détection, l'estimation de la pose, le suivi et la classification.
YOLOv12	2025	Ce modèle introduit une architecture centrée sur l'attention qui s'écarte des approches traditionnelles basées sur le CNN dans les modèles YOLO précédent.

Depuis son introduction, plusieurs versions de YOLO ont été développées, chacune apportant des améliorations significatives en termes de précision, de vitesse et de robustesse. Dans ce contexte, le choix s'est porté sur YOLOv5 en raison des limitations matérielles de la carte Nvidia Jetson Nano, ce modèle offrant un excellent compromis entre performance et légèreté. Les versions plus récentes de YOLO, bien que généralement plus précises, sont plus coûteuses en ressources et donc moins adaptées aux systèmes embarqués à faible capacité de calcul.

Dans la suite de ce travail, nous nous concentrerons sur l'algorithme YOLOv5.

III.2.2.2 Architecture de YOLOv5

Dans YOLOv5, l'image est d'abord traitée par une couche d'entrée, puis envoyée au "Backbone" pour extraire les caractéristiques de l'image. Le Backbone produit des cartes de caractéristiques de différentes tailles, qui sont ensuite fusionnées via un réseau de fusion de caractéristiques (le "Neck"). Cela un ensemble de couches qui relie et combinent ces caractéristiques pour les transmettre à la prédiction. Chaque carte est utilisée pour détecter les petits, moyens et grands objets dans l'image [9].

La Figure III.5 qui illustre l'architecture générale du modèle YOLOv5. Les différentes parties de cette architecture seront détaillées dans les sous-sections suivantes.

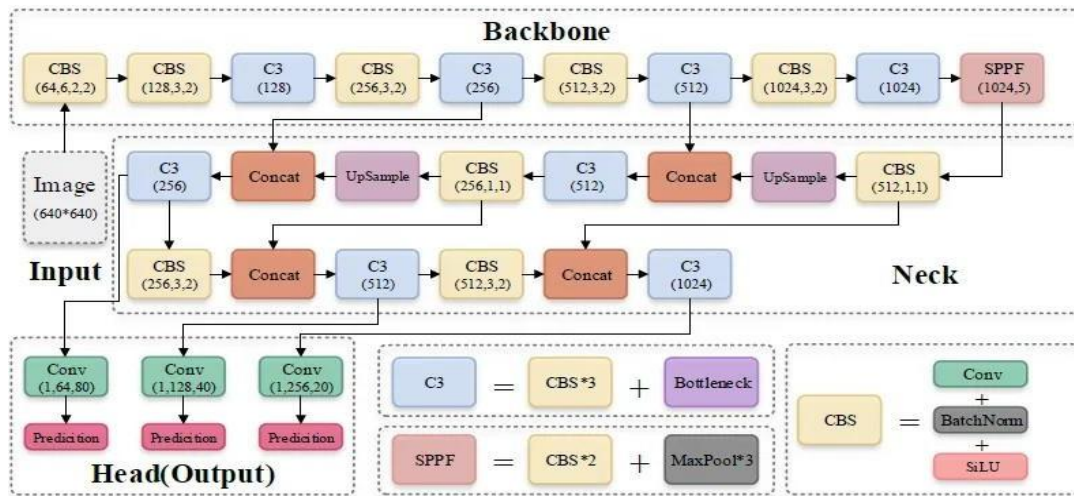


Figure III. 5. Architecture du modèle YOLOv5 [10].

III.2.2.3 Différentes parties de YOLOv5

YOLOv5 est composé de trois parties : la colonne vertébrale (backbone), le cou (neck), et la tête (head).

a) Backbone

Le Backbone examine l'image et identifie les détails de base à différentes échelles, comme un filtre photo qui capture les contours et les formes. Ensuite, le backbone basé sur CSP Darknet53, intègre plusieurs modules CBS et C3 comme cité dans la Figure III.5, ainsi qu'un module SPPF pour améliorer l'extraction et l'expression des caractéristiques. Le module CBS assiste le module C3 dans cette tâche, tandis que le SPPF optimise l'agrégation des fonctionnalités spatiales, améliorant ainsi la vitesse de traitement [9].

b) Neck

Le cou prend les caractéristiques détectées par le backbone et les mélange pour obtenir une vue d'ensemble de l'image, prête à être analysée pour la détection des objets. Dans la Figure III.6, le Neck utilise les méthodes FPN et PAN pour détecter des cibles à différentes échelles.

FPN sur-échantillonne les cartes de caractéristiques de sortie (C3, C4 et C5) pour générer de nouvelles cartes de caractéristiques (P3, P4 et P5), permettant ainsi une détection multi-échelle [9].

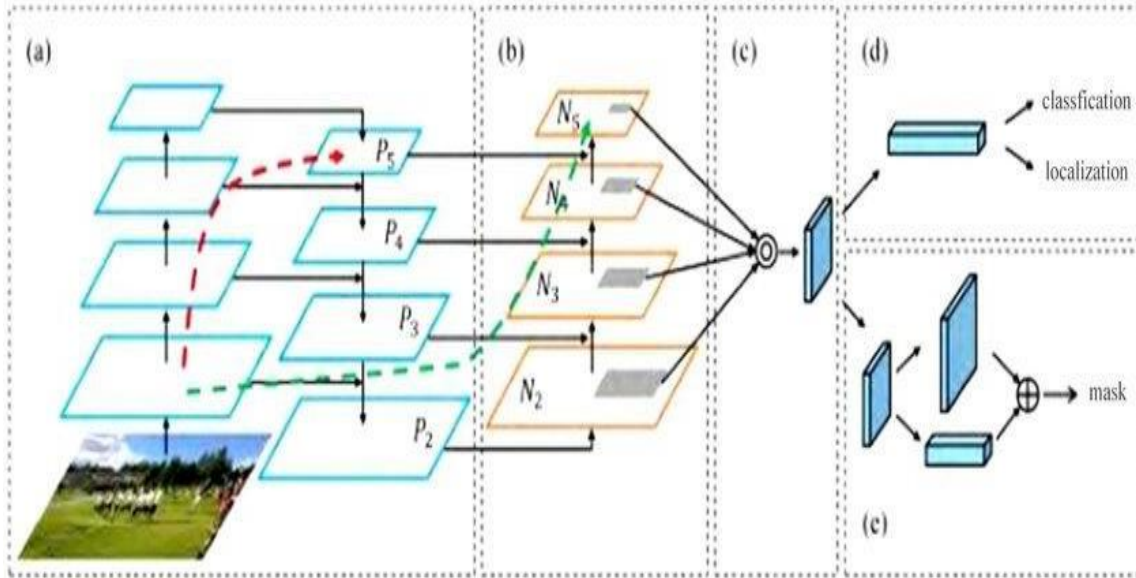


Figure III. 6. PANet. (a) FPN. (b) voie ascendante. (c) mise en commun adaptative des fonctionnalités. (d) branche de détection. (e) couche de fusion.

c) La tête

Prend cette vue d'ensemble créée par le cou et l'utilise pour identifier ce qu'il y a dans l'image qu'il s'agisse d'un chat, d'une voiture ou autre chose. Ensuite, elle prend des décisions sur les boîtes qui entourent ces objets et les classes auxquelles ils appartiennent [9].

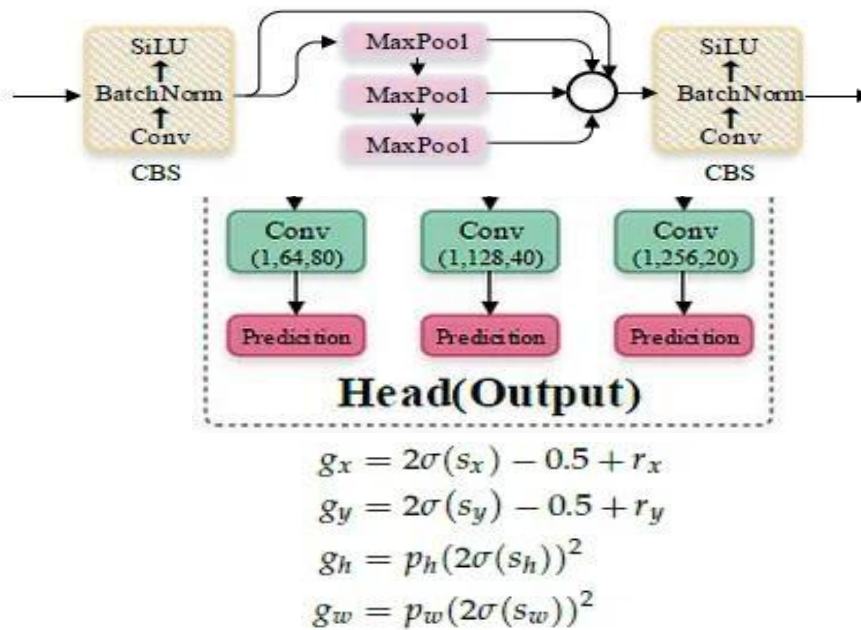


Figure III. 7. Structure du SPPF / Régression de la boîte englobante.

La Figure III.7 représente un schéma de la structure du SPPF et de la régression de la boîte englobante. Le SPPF utilise des niveaux de pooling spatial pour extraire des caractéristiques à différentes échelles, tandis que la régression de la boîte englobante affine les prédictions de localisation des objets [9].

- Les coordonnées du coin supérieur gauche de la carte des caractéristiques sont fixées à $(0, 0)$.
- rx et ry représentent les coordonnées non ajustées du point central prédit.
- gx, gy, gw et gh contiennent les informations de la boîte de prédiction ajustée.
- $pwet\ ph$ correspondent aux informations de l'ancre précédente.
- sx, sy représentent les décalages calculés par le modèle.
- Le processus vise à ajuster la coordonnée centrale et la taille de l'ancre préalablement définie.
- **Activation et optimisation** : YOLOv5 utilise l'activation sigmoïde et ReLU fuyante pour normaliser les sorties et éviter les problèmes de gradient, ainsi que SGD et ADAM comme options d'optimisation et permettant une mise à jour flexible des poids du modèle.
- **Fonction de perte** : Il utilise l'entropie croisée binaire avec une perte, adaptée à la classification binaire des objets détectés.

YOLOv5 améliore également les prédictions à plusieurs échelles en affinant davantage la fusion des caractéristiques des différentes résolutions. Le réseau produit des prédictions à trois échelles différentes : petite, moyenne et grande, permettant de détecter efficacement des objets de diverses tailles avec une meilleure précision. Cette approche multi-échelle est renforcée par des améliorations dans les mécanismes d'up sampling et de down sampling, ce qui permet une meilleure intégration des informations contextuelles à différentes échelles. Un autre aspect clé de YOLOv5 est l'amélioration de l'utilisation des ancres (anchors). YOLOv5 introduit un mécanisme d'auto-ajustement des ancres (auto-anchor) qui permet au modèle de déterminer automatiquement les meilleures tailles d'ancres en fonction des données d'entraînement. Cette optimisation des ancres permet d'améliorer la précision de détection en ajustant les boîtes englobantes aux caractéristiques spécifiques des objets présents dans les données. En résumé, YOLOv5 introduit des améliorations significatives par rapport à ses prédécesseurs en optimisant l'architecture du réseau, en améliorant les mécanismes de prédiction multi-échelle, et en ajustant automatiquement les ancres. Ces

améliorations permettent à YOLOv5 d'atteindre un équilibre optimal entre rapidité et précision [11].

III.2.2.4 Modèles de YOLOv5

YOLOv5, cinquième itération de la famille YOLO, a été présentée par Glenn Jocher [12]. Cette version se distingue comme une référence majeure et une source d'inspiration pour les modèles de détection d'objets en un seul passage. Comparativement aux modèles à deux étages, YOLOv5 se distingue par sa rapidité d'exécution tout en maintenant une précision élevée.

Cette itération représente une extension de YOLOv3, largement utilisée par la communauté pour la conversion des poids de Darknet vers PyTorch. Bien que YOLOv4 ait été publiée avant YOLOv5, cette dernière a été renommée afin d'éviter toute confusion entre les différentes versions du modèle [13].

Comme illustré dans la Figure III.8, YOLOv5 est décliné en cinq variantes principales: nano (*n*), small (*s*), medium (*m*), large (*l*) et extra-large (*x*). Chaque version présente un compromis différent entre vitesse d'inférence et précision, les modèles les plus légers étant plus rapides mais moins précis, tandis que les modèles plus volumineux offrent une meilleure précision au prix de temps d'entraînement et d'inférence plus importants [14].

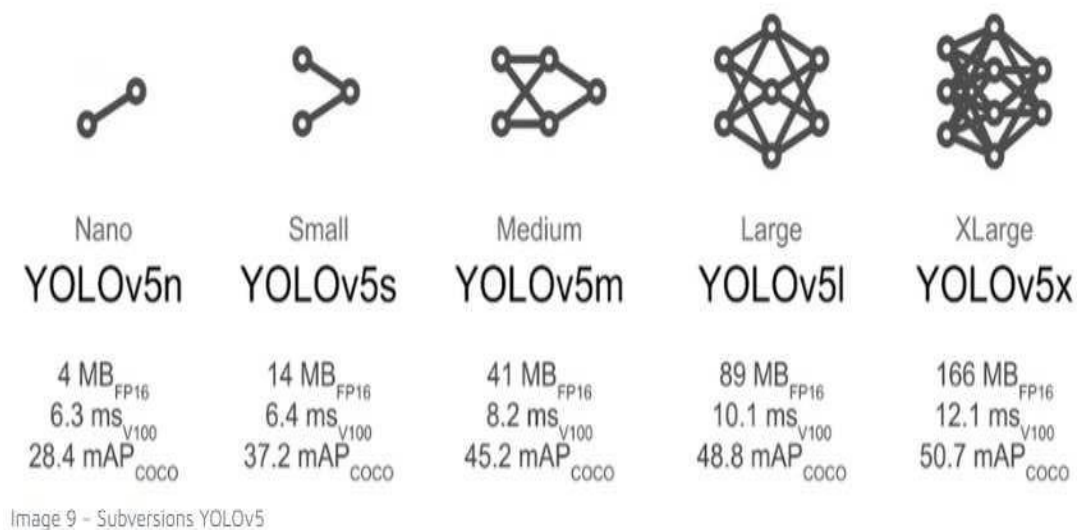


Figure III. 8. Différentes modèles proposées par YOLOv5 [15].

Dans la Figure III.9, chaque version de YOLO est comparée en termes de précision et d'amélioration des performances des procédures d'entraînement de PyTorch, avec le temps d'inférence sur l'axe X. YOLOv5 peut traiter des images rapidement et avec une précision presque identique à celle du modèle EfficientDet [9].

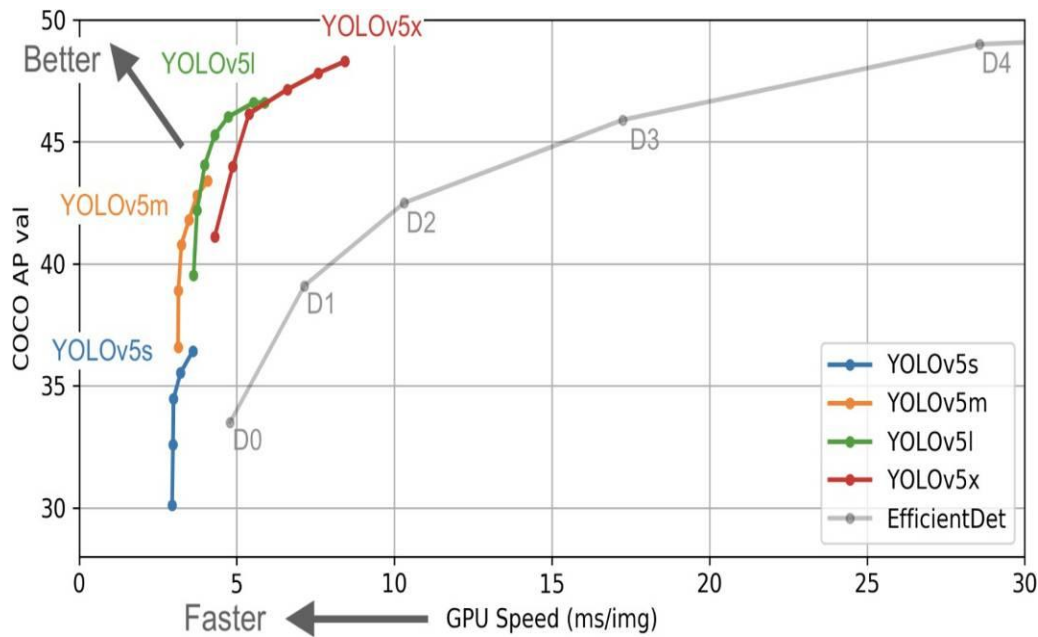


Figure III. 9. Version initiale de YOLOv5 montre la promesse d'une détection d'objets de pointe [15].

III.2.2.5 Entraînement

Les hyperparamètres sont des paramètres de haut niveau qui définissent la structure et le comportement d'un algorithme d'apprentissage. Leurs réglages est un processus itératif qui vise à optimiser les mesures de performance du modèle d'apprentissage automatique, comme l'exactitude, la précision et le rappel. Dans le contexte de Ultralytics YOLO, ces hyperparamètres couvrent un large éventail d'éléments, allant du taux d'apprentissage aux choix architecturaux, comme le nombre de couches ou les fonctions d'activation utilisés. Ils sont définis avant la phase d'entraînement et restent généralement constants au cours de celle-ci [8].

Parmi les hyperparamètres les plus couramment ajustés dans Ultralytics YOLO, on peut citer :

- **Taux d'apprentissage (learning rate):** il détermine l'amplitude des mises à jour des poids du modèle à chaque itération lors de la minimisation de la fonction de perte.
- **Taille du lot (batch size):** elle correspond au nombre d'images traitées simultanément lors d'une passe avant (propagation des données d'entrée à travers les couches du modèle pour obtenir une sortie).
- **Spécificités de l'architecture:** il incluent le nombre de couches, le nombre de canaux ainsi que les types de fonctions d'activation utilisées.
- **Nombre d'époques (Epoch):** une époque correspond à un passage complet l'ensemble des données d'apprentissage, comprenant la propagation avant (prédiction) et la rétropropagation (ajustement des poids) [8], [16].

Plusieurs méthodes permettent d'optimiser le choix des hyperparamètres, telles que la recherche en grille (grid search), la recherche aléatoire (random search) ou encore l'optimisation bayésienne [17]. Dans ce travail, nous avons adopté la méthode de recherche en grille. Cette approche consiste à définir un ensemble de valeurs possibles pour chaque hyperparamètre, puis à entraîner le modèle sur l'ensemble d'apprentissage et à évaluer ses performances sur l'ensemble de validation. Ce processus est répété pour toutes les combinaisons possibles, afin de sélectionner la configuration offrant les meilleures performances sur les données de validation.

III.2.2.6 Métriques d'évaluation

L'évaluation des performances des modèles de détection d'objets nécessite des mesures normalisées qui quantifient la précision et la fiabilité avec lesquelles le système peut identifier et localiser des objets dans une image [18].

Il existe différentes façons de mesurer l'efficacité d'un modèle, notamment sa capacité à confondre les exemples des différentes classes, l'exactitude de ses prédictions, sa précision à classer les différents exemples, sa sensibilité, ainsi que son score. Ces différentes méthodes sont présentées ci-dessous. Dans la pratique, l'utilisation de ces différentes métriques nécessite la séparation du jeu de données en données d'entraînement et en données de test, et de données de validation [19].

- **Matrice de confusion (Confusion Matrix)**

Elle fournit une visualisation des prédictions faites par le modèle par rapport aux exemples réels des données de test. Elle fournit également une représentation tabulaire des résultats [19].

Tableau III. 2. Matrice de confusion.

	Prédiction Positive	Prédiction Négative
Réel Positif	<i>TP</i>	<i>FN</i>
Réel Négatif	<i>FP</i>	<i>TN</i>

Avec :

- ✓ *TP* (True Positive) : le modèle prédit une classe positive et cette prédiction est correcte.
- ✓ *TN* (True Négative) : le modèle prédit une classe négative et cette prédiction est correcte.
- ✓ *FP* (False Positive) : le modèle prédit une classe positive alors que la réalité correspond à une classe négative.

- ✓ **FN** (False Négative) : le modèle prédit une classe négative alors que la réalité correspond à une classe positive.

Dans notre étude, la classe positive est définie comme correspondant aux situations de non-chute. Par conséquent, les vrais positifs (TP) désignent les non-chutes correctement détectées, tandis que les vrais négatifs (TN) correspondent aux chutes correctement identifiées comme telles.

Les faux positifs (FP) représentent les instances incorrectement classées comme appartenant à la classe positive (non-chute), alors qu'elles relèvent en réalité de la classe négative (chute). À l'inverse, les faux négatifs (FN) correspondent aux instances de la classe positive (non-chute) qui ont été incorrectement classées comme appartenant à la classe négative (chute).

- **Exactitude (Accuracy)**

Mesure la proportion des prédictions correctes faites par le modèle sur les données d'entraînement en utilisant la formule :

$$A = \frac{TP + TN}{TP + TN + FP + FN} \quad (\text{III.1})$$

- **Précision (Precision)**

Permet de mesurer le taux de prédiction correcte sur l'ensemble du jeu de données avec la formule :

$$P = \frac{TP}{TP + FP} \quad (\text{III.2})$$

- **Rappel (Recall)**

Également connu sous le nom de sensibilité, mesure la proportion de prédictions positives réelles entre tous les cas positifs réels à travers la formule :

$$R = \frac{TP}{TP + FN} \quad (\text{III.3})$$

- **Average Precision (AP)**

À partir de ces deux métriques, à savoir la précision et le rappel, il est possible d'obtenir, pour chaque classe, un ensemble de valeurs associées (P, R). Sur la base de ces valeurs, on calcule une métrique essentielle appelée Average Precision (AP).

Il est important de noter que l'AP ne correspond pas à une simple moyenne des précisions obtenues pour les différentes classes. Au contraire, l'AP est calculée en intégrant la courbe de Précision-Rappel sur les différentes valeurs de rappel. Elle représente ainsi l'aire sous cette courbe pour une classe donnée. D'un point de vu mathématique, l'AP est définie par l'équation [20] :

$$AP = \int_0^1 P(R) dr \quad (\text{III.4})$$

- **Score F1 (F1-Score)**

Le score F1 est une métrique d'évaluation largement utilisée en apprentissage automatique, en particulier pour les problèmes de classification et de détection d'objets. Il correspond à la moyenne harmonique de la précision et du rappel, ce qui lui permet de prendre en compte simultanément ces deux mesures [19].

$$F1-score = 2 \times \frac{P \times R}{P + R} \quad (\text{III.5})$$

Ainsi, le score F1 pénalise les valeurs extrêmes, notamment lorsque l'une des deux métriques (précision ou rappel) est faible. Cela le rend particulièrement pertinent pour évaluer les performances d'un modèle dans les situations où un compromis entre précision et rappel est nécessaire [20].

- **Spécificité (Specificity)**

Permet de mesurer la proportion de vrais négatifs correctement identifiés parmi tous les cas négatifs réels. Elle détermine la capacité du modèle à identifier correctement les cas négatifs.

Sa formule est donnée par :

$$S = \frac{TN}{TN + FP} \quad (\text{III.6})$$

- **L'intersection par rapport à l'union (IoU)**

L'Intersection over Union (IoU) est un indicateur qui quantifie le degré de chevauchement entre la boîte prédite et la boîte réelle [17].

Il est défini comme suit :

$$IOU = \frac{\text{Aire}(B_{\text{prédite}} \cap B_{\text{réelle}})}{\text{Aire}(B_{\text{prédite}} \cup B_{\text{réelle}})} \quad (\text{III.7})$$

Ainsi, l'IoU permet d'évaluer la qualité de localisation d'une prédiction. Lorsque $IoU=1$, la boîte prédite coïncide parfaitement avec la boîte réelle. A l'inverse, lorsque $IoU=0$, il n'existe aucun chevauchement entre les deux boîtes.

Par conséquent, plus la valeur de l'IoU est élevée, plus la précision spatiale de la prédiction est importante.

- **mAP (Mean Average Precision)**

Fournit une évaluation complète en prenant en compte la précision moyenne pour toutes les classes d'objets de l'ensemble de données. Elle représente la performance moyenne du

modèle pour toutes les classes. La mAP est calculée en faisant la moyenne des valeurs de précision obtenues pour chaque classe d'objets. La formule simplifiée de la mAP est la suivante:

$$mAP = \frac{1}{N} \sum_{i=1}^N AP_i \quad (\text{III.8})$$

où N est le nombre total de classes d'objets et AP_i est la précision moyenne pour la classe i . Une valeur mAP élevée indique que le modèle est performant en moyenne pour toutes les classes d'objets. Cette mesure est souvent utilisée pour comparer les performances de différents modèles de détection d'objets [17].

- **$mAP@0.5$**

La $mAP@0.5$ correspond à la moyenne des AP calculées avec un seuil d'Intersection over Union (IoU) fixé à 0.5.

Autrement dit, une prédiction est considérée comme correcte si le chevauchement entre la boîte prédite et la boîte réelle est supérieur ou égal à 50%. Cette métrique est relativement tolérante et met l'accent sur la capacité du modèle à détecter les objets. Ce critère d'évaluation est défini par :

$$mAP@0.5 = \frac{1}{N} \sum_{i=1}^N AP_i^{IoU \geq 0.5} \quad (\text{III.9})$$

où :

N est le nombre total de classes, et AP_i est l'Average Precision de la classe i .

- **$mAP@0.5:0.95$**

La $mAP@0.5:0.95$ (souvent appelée mAP COCO) est une métrique plus stricte et plus complète. Elle est obtenue en calculant la moyenne des valeurs d'Average Precision (AP) pour plusieurs seuils d'Intersection over Union (IoU), variant de 0.5 à 0.95 avec un pas de 0.05 :

$$mAP@0.5:0.95 = \frac{1}{N} \sum_{i=1}^N \left(\frac{1}{T} \sum_{t=1}^T AP_i^{IoU_t} \right) \quad (\text{III.10})$$

où :

$IoU_t \in [0.5, 0.95]$ avec un pas de 0.05, T représente le nombre total de seuils (généralement égal à 10). $AP_i^{IoU_t}$ désigne l'Average Precision calculée pour un seuil IoU donné.

Cela signifie que le modèle est évalué non seulement sur sa capacité à détecter les objets, mais aussi sur la précision de leur localisation. Par conséquent, cette métrique est plus représentative des performances réelles du modèle.

III.3 Système intelligent pour la télésurveillance des personnes âgées

La détection des chutes a suscité un intérêt considérable ces dernières années, notamment dans les domaines des soins aux personnes âgées, de la surveillance et du suivi de la santé. Avec le vieillissement de la population, il devient essentiel de disposer de systèmes de détection des chutes robustes, capables de fournir une réponse rapide en cas de situation critique et de réduire le nombre de décès en diminuant le délai d'intervention.

Ces systèmes peuvent améliorer significativement la qualité de vie des patients ou les personnes âgées et réduire le risque de blessures graves, même lors de chutes mineures. Ils reposent principalement sur la vision par ordinateur et des réseaux neuronaux, permettant la détection des chutes en temps réel à partir de données vidéo.

Dans les sections suivantes, nous présentons la méthodologie adoptée pour la réalisation de notre système de détection. Celle-ci est structurée en plusieurs phases, chacune traitant un aspect essentiel du développement et garantissant une approche globale du problème :

- **Base de données :** cette phase peut être divisée en deux étapes. La première étape est concentrée à l'étude bibliographique, à l'acquisition et à l'enrichissement des connaissances, ainsi qu'à la définition de la méthodologie et de l'approche du système. Elle comprenait des recherches approfondies sur les systèmes de détection des chutes et sur les différents modèles et approches utilisés dans ce domaine.
La deuxième concerne la préparation des données. Elle débute par la conversion des vidéos en images, suivie de leur annotation pour l'entraînement du modèle YOLO. Les annotations jouent un rôle essentiel dans les performances du modèle. Cette phase comprend également le choix de la version de YOLO la plus adaptée à notre système. Après avoir expérimenté plusieurs variantes et effectué des tests sur la carte Nvidia, nous avons retenu YOLOv5 comme étant la plus appropriée et la plus stable dans notre cas.
- **Entraînement sur YOLOv5:** cette phase correspond à l'apprentissage du modèle sur l'ensemble de données annoté, en utilisant la version YOLOv5.
- **Résultats et discussions :** elle est dédiée aux tests et à l'évaluation des performances du modèle.
- **Phase finale :** elle consiste à déployer le modèle le plus performant sur la carte Nvidia Jetson Nano afin d'assurer la détection des chutes en temps réel. Cette phase est détaillée dans le chapitre IV.

III.3.1 Base de données

Il existe de nombreux ensembles de données basés sur la vision, dédiés aux systèmes de détection des chutes. A l'issue d'une étude approfondies, nous avons retenu l'ensemble de données LE2I, qui contient des vidéos d'activités de chute et de non-chute, annotées image par image (0 pour non-chute et 1 pour chute) [21].

Cet ensemble est largement utilisé car il propose des scénarios réalistes capturés sous plusieurs angles, chaque image étant annotée. Il est très varié, car il inclut des cas difficiles tels que des conditions d'éclairage variables.

Les approches basées sur la vision peuvent s'appuyer sur n'importe quelle caméra, telle qu'une webcam, une caméra de profondeur comme Kinect, des caméras de surveillance IP ou encore des caméras RVB classique. De nombreuses actions, telles que les changements de direction, le transport d'un objet et l'allumage/extinction de la lumière, ont été simulées.

Description de la base de données LE2I

L'ensemble de données LE2I pour la détection des chutes, utilisées pour l'entraînement et l'évaluation du modèle YOLOv5, contient des séquences vidéo annotées image par image, spécialement conçues pour la détection des chutes dans des situations réelles. Chaque image est étiquetée avec une classe 0 (non-chute) ou 1 (chute), et est accompagnée de cadres de sélection indiquant l'absence ou la présence de chutes. L'ensemble de données comprend des vidéos capturées dans différentes situations, garantissant ainsi un entraînement robuste du modèle pour divers angles. De plus, certaines vidéos de l'ensemble de données LE2I ont également été traitées et utilisées pour évaluer les performances de notre modèle sur des données non vues.

LE2I est une base de données est capturée par une seule caméra avec une résolution de 320×240 pixels. LE2I contient 191 vidéos enregistrées avec une réalisation et un paramétrage de vidéosurveillance réelle. Les vidéos de cette base de données ont été enregistrées à partir de différents lieux, permettant de définir plusieurs protocoles d'évaluation [22].

Caractéristiques de la base de données LE2I

Notre objectif de ce mémoire est à relever les défis susmentionnés en développant un système de détection de chutes en temps réel basés sur la vision. Pour ce faire, nous exploitons le modèle de détection d'objets de YOLOv5, qui nous affinons à l'aide de l'ensemble de données LE2I dédié aux chutes, afin d'améliorer la précision de la détection et de la classification en temps réel.

L'ensemble de données LE2I nous offre une base solide et robuste pour la reconnaissance des activités liées aux chutes. Son adaptation à nos besoins permet d'obtenir de meilleures

performances du modèle dans des contextes variés. L'utilisation conjointe de YOLOv5 et LE2I représente ainsi une avancée significative vers la mise en place d'un système capable de prédire avec précision les chutes dans des environnements réels et dynamiques.

La capacité à analyser des flux vidéo en temps réel et à distinguer les chutes des autres activités est essentielle pour améliorer la sécurité des populations vulnérables, notamment dans les établissements de soins pour personnes âgées, les domiciles et les structures de santé. Dans le chapitre suivant, ce modèle YOLOv5 sera déployé sur un système embarqué de type Nvidia Jetson Nano, afin d'assurer une détection en temps réel directement sur le dispositif.

Annotation des données

Nous avons finalement opté pour l'utilisation du modèle YOLOv5 sur un sous-ensemble de l'ensemble de données LE2I, celui-ci étant particulièrement volumineux. Afin d'entraîner notre modèle avec YOLOv5, nous avons adopté la structure et les annotations suivantes :

- a) **Structure de l'ensemble de données** : l'ensemble de données a été divisé en trois sous-ensembles distincts : entraînement, validation et test. Chacun de sous-ensembles contient des répertoires dédiés aux images ainsi qu'aux annotations correspondantes.
- b) **Annotations** : des cadres de sélection pour les événements de chute et de non-chute sont déjà fournies dans cet ensemble de données. Elles sont stockées dans des fichiers au format *.txt*, compatibles avec YOLOv5, et contiennent les informations suivantes :
 - **ID de classe** : un entier représentant la classe de l'objet.
 - **Centre X** : la coordonnée x normalisée du centre du cadre de sélection (valeur comprise entre 0 et 1).
 - **Centre Y** : la coordonnée Y normalisée du centre du cadre de sélection (valeur comprise entre 0 et 1).
 - **Largeur** : la largeur normalisée du cadre de sélection (valeur comprise entre 0 et 1)
 - **Hauteur** : la hauteur normalisée du cadre de sélection (valeur comprise entre 0 et 1).

Cette organisation facilite l'entraînement du modèle et assure la compatibilité avec les exigences du format YOLOv5.

III.3.2 Entraînement sur YOLOv5

L'algorithme YOLOv5 est fourni par Ultralytics et repose sur la bibliothèque PyTorch. YOLOv5 est codé en Python et hébergé par les développeurs sur la plateforme open source Github.

Nous avons utilisé la plateforme Google Colaboratory pour l'entraînement du modèle, car elle offre un accès gratuit à des processeurs graphiques (GPU) performants sans nécessiter

d'installation locale complexe. Cette plateforme permet également une intégration fluide avec Google Drive, où sont stockés l'ensemble de données ainsi que la bibliothèque YOLOv5, ce qui facilite l'organisation des fichiers et optimise le processus d'entraînement du modèle.

Pour entraîner le modèle YOLOv5 sur notre base de données dédiée à la détection des chutes, plusieurs étapes ont été suivies, comme détaillé dans la suite :

a) Montage de notre disque personnel :

Afin d'utiliser l'ensemble de données que nous avons téléchargé sur le Drive, nous allons monter notre Drive à l'aide du code ci-dessous.

```
[ ]  
from google.colab import drive  
drive.mount('/content/drive')
```

Une fenêtre contextuelle s'affichera avec la question «Autoriser ce notebook à accéder à vos fichiers Google Drive ?». Sélectionnez le bouton bleu «Se connecter à Google Drive».



Figure III. 10. Connexion du notebook Google Colab à Google Drive.

Une fois le Drive monté, le message «Monté» s'affichera. La racine de notre Google Drive sera montée dans /content/drive/My Drive/.

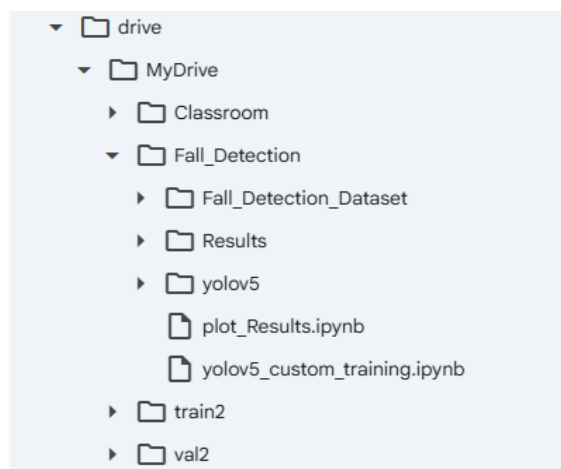


Figure III. 11. Structure des dossiers du jeu de données.

Le répertoire Fall_Detection_Dataset parent contient trois fichiers et trois sous-répertoires :

- **data.yaml**: il contient les configurations relatives aux données, telles que le chemin d'accès aux répertoires de données d'entraînement et de validation, le nombre total de classes dans l'ensemble de données et le nom de chaque classe.
- **train**: images d'entraînement accompagnées de leurs étiquettes.
- **valid**: images de validation avec annotations.
- **test**: images de test et étiquettes. Il est facile d'évaluer les performances de modèle si des données de test avec étiquettes sont disponibles.

La Figure III.12 représente une analyse statistique et graphique des annotations utilisées pour l'entraînement du modèle YOLOv5. Elle permet de visualiser la répartition des classes ainsi que les caractéristiques des boîtes englobantes (bounding boxes) présentes dans la base de données. Comme l'illustre cette figure, l'ensemble d'entraînement contient 2477 images annotées.

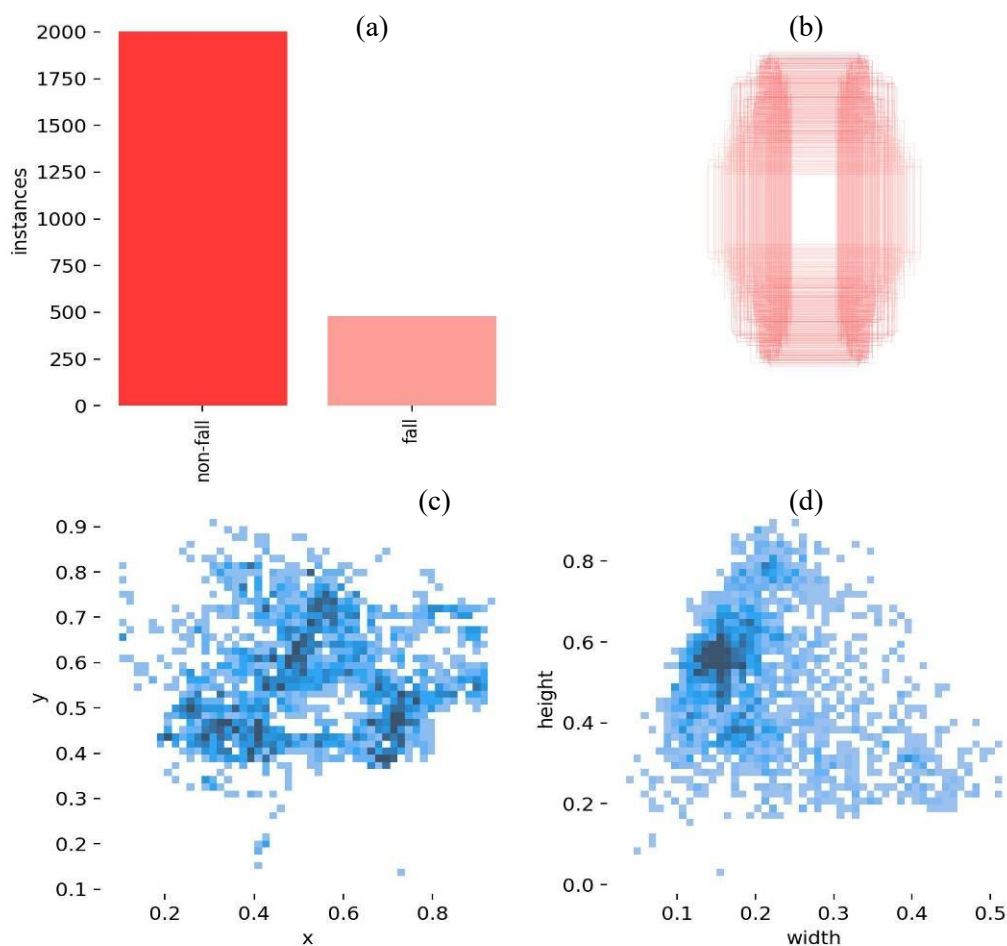


Figure III. 12. (a) Répartition des classes, (b) superposition des boîtes englobantes, (c) distribution des centres des objets (x, y) et (d) distribution des dimensions des boîtes (width, height).

Pour les ensembles de validation et de test, nous disposons respectivement de 710 images pour la validation et de 354 images pour le test. Au total, la base de données contient 3541 images extraites à partir des vidéos.

b) Installation de l'environnement YOLOv5

La première étape consiste à cloner le dépôt YOLOv5. Les lignes de commande suivantes permettent de cloner le dépôt, d'accéder au répertoire de travail YOLOv5, puis d'installer toutes les dépendances nécessaires à l'exécution et à l'entraînement du modèle.

```
[ ]
#clone YOLOv5 and
%cd /drive/MyDrive/Fall_Detection/yolov5
%pip install -qr requirements.txt # install dependencies

import torch
import os
from IPython.display import Image, clear_output # to display images

print(f"Setup complete. Using torch {torch.__version__} ({torch.cuda.get_device_properties(0).name if torch.cuda.is_available() else 'CPU'})")

/drive/.shortcut-targets-by-id/1Lu0mDMGRpTVvrY1u8uBMeS4nyDicZ5ss/Fall_Detection/yolov5
1.2/1.2 MB 65.7 MB/s eta 0:00:00
131.6/131.6 kB 17.8 MB/s eta 0:00:00
Setup complete. Using torch 2.10.0+cu128 (Tesla T4)
```

Figure III. 13. Clonage et installation du dépôt YOLOv5.

c) Entraînement de YOLOv5

À l'aide de la ligne de commande illustrée à la Figure III.14, le modèle sera entraîné par le fichier train.py, avec ses arguments configurables.

```
[ ]
exp = "yolov5x"

fall_detection_exp = f"Fall_Detection_{exp}"
yolo_name = f"{exp}.pt"
#yolov5s.pt
#yolov5n.pt
#yolov5m.pt
#yolov5l.pt
#yolov5x.pt

Double-click (or enter) to edit

Double-click (or enter) to edit

[ ]
!python train.py --img 416 --batch 16 --epochs 100 \
--data /drive/MyDrive/Fall_Detection/Fall_Detection_Dataset/data.yaml \
--weights {yolo_name} --cache \
--project /drive/MyDrive/Fall_Detection/Results/{fall_detection_exp} \
--name "train"

> Show hidden output
```

Figure III. 14. Mise en œuvre du processus d'entraînement.

Nous définissons quelques paramètres du modèle :

- **Img:** définir la taille de l'image d'entrée.
- **batch:** détermine la taille du lot.
- **Epochs:** définit le nombre d'époques d'entraînement.
- **Data:** définir le chemin d'accès à notre fichier YAML.

- **Cfg**: spécifier la configuration de notre modèle.
- **Weights**: spécifier un chemin d'accès personnalisé vers les poids.
- **Name**: noms des résultats.
- **Nosave**: n'enregistrer que le point de contrôle final.
- **Cache**: mettre les images en cache pour accélérer l'entraînement.

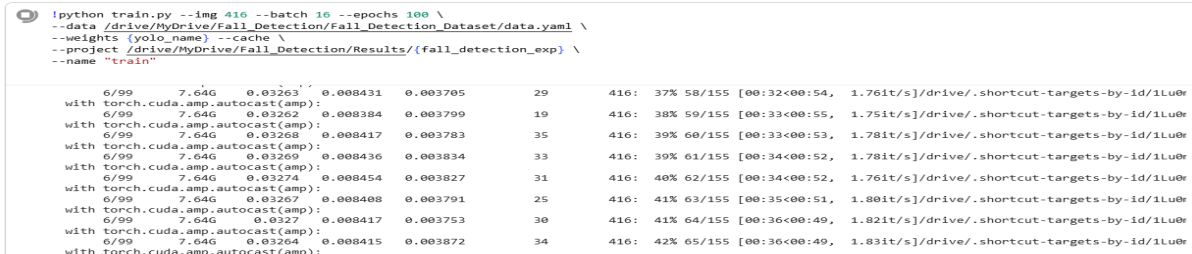


Figure III. 15. Progression de l'entraînement sur 100 époques.

Un module complémentaire appelé `Fall_Detection` a été créé pour visualiser clairement l'ensemble du processus d'entraînement.

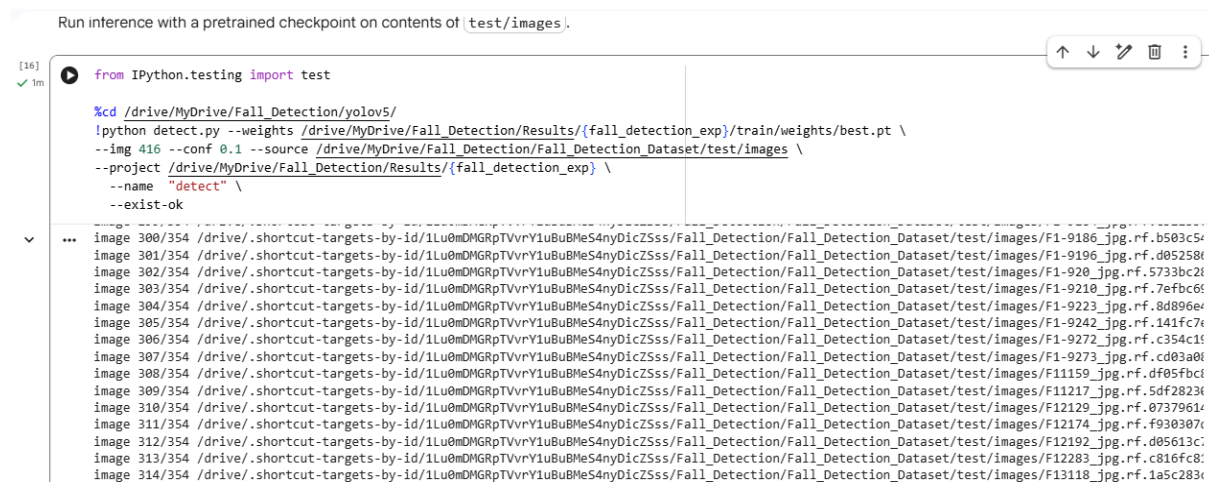


Figure III. 16. Processus d'entraînement enregistré dans le dossier «Fall_Detection».

De plus, le modèle enregistre deux résultats de pondération sous forme de fichiers `.pt` : «last.pt» correspond aux poids de la dernière époque et «best.pt» correspond aux poids de la dernière époque ayant donné la meilleure précision.

d) Inférence avec les poids entraînés

La session d'inférence du notebook, où nous pouvons tester le modèle que nous avons entraîné, la méthode de détection d'objets avec les poids entraînés est similaire à l'entraînement du modèle.

À l'aide de la commande illustrée à la Figure III.17, le fichier `detect.py` sera compilé, ce qui reconstruira l'architecture utilisée lors de l'entraînement. Les poids entraînés seront utilisés pour prédire les objets et les cadres de sélection.

```
#export your model's weights for future use
from google.colab import files
files.download(f'/drive/MyDrive/Fall_Detection/Results/{fall_detection_exp}/train/weights/best.pt')

from google.colab import files
files.download(f'/drive/MyDrive/Fall_Detection/Results/{fall_detection_exp}/train/weights_{fall_detection_exp}.zip')

from google.colab import files
files.download(f'/drive/MyDrive/Fall_Detection/Results/{fall_detection_exp}/detect/test_{fall_detection_exp}.zip')

# Validate YOLOv5s on Fall Detection
%cd /drive/MyDrive/Fall_Detection/yolov5/
!python val.py --weights /drive/MyDrive/Fall_Detection/Results/{fall_detection_exp}/train/weights/best.pt \
--data /drive/MyDrive/Fall_Detection/Fall_Detection_Dataset/data.yaml \
--project /drive/MyDrive/Fall_Detection/Results/{fall_detection_exp}/ \
--name "val"
```

Figure III. 17. Mise en œuvre du processus de détection.

Une fois la détection terminée, les cadres de sélection prédits qui recouvrent les objets (chutes, non chutes) seront dessinés sur l'image. Ils seront enregistrés dans le même dossier que celui contenant les résultats de la phase d'entraînement. Le chemin d'accès sera du type (Fall_Detection/detect/exp).

Les résultats que nous avons obtenus sont présentés dans la sous-section suivante :

III.3.3 Résultats et discussions

Les modèles YOLO se déclinent en plusieurs variantes : Nano (n), Small (s), Medium (m), Large (l) et Extra Large (x). Afin d'évaluer leurs performances pour la détection des chutes, nous avons réalisé un entraînement de 100 époques sur chacun des cinq modèles.

Les principaux paramètres d'entraînement utilisés sont les suivants :

- Taux d'apprentissage: 0,001,
- Optimiseur: Adam,
- Taille des images: 640×640 pixels
- Taille du lot (Batch Size): 16.

Cette configuration nous a permis de comparer les performances des différentes variantes de YOLOv5 en termes de précision, de rapidité et de capacité de détection.

La Figure III.18 illustre les résultats de l'entraînement et de la validation des différents modèles personnalisés sur l'ensemble de données utilisé. À droite sont présentés les résultats obtenus sur l'ensemble d'entraînement, tandis qu'à gauche figurent ceux obtenus sur l'ensemble de validation.

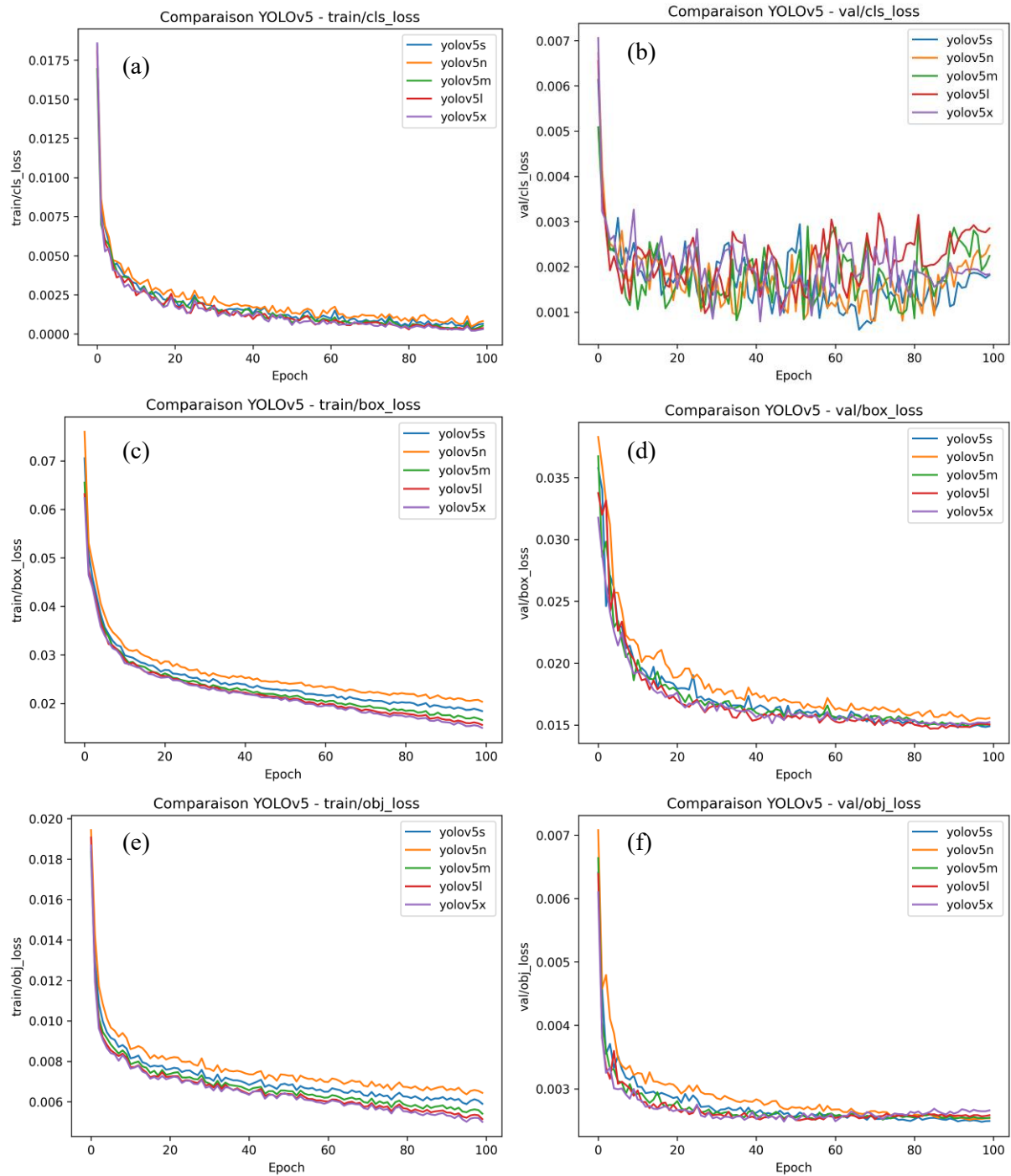


Figure III. 18. Performances des modèles YOLOv5, basées sur les différentes métriques d'évaluation : la perte de classification (a–b), la perte de localisation des boîtes (c–d), ainsi que la perte d'objectivité (e–f).

Afin d'évaluer les performances des modèles, plusieurs métriques ont été analysées au cours des différentes époques d'entraînement, notamment la perte de classification (*cls_loss*), la perte de localisation des boîtes (*box_loss*) ainsi que la perte d'objectivité (*obj_loss*), aussi bien sur l'ensemble d'entraînement que sur l'ensemble de validation. Ces métriques permettent

d'analyser la capacité des modèles à détecter correctement les objets, à les localiser avec précision et à améliorer progressivement leurs performances au cours de l'apprentissage.

Les courbes obtenues montrent une diminution progressive des différentes pertes au fil des époques, ce qui indique que les modèles apprennent correctement et améliorent progressivement leurs performances. De plus, les métriques deviennent relativement stables vers la fin de l'entraînement, avec seulement de légères fluctuations, ce qui traduit une bonne convergence des modèles.

La perte de classification (`cls_loss`) évalue la capacité du modèle à attribuer correctement la classe correspondante à l'objet détecté. Une diminution de cette perte traduit une amélioration de la précision de classification des objets par le modèle.

La métrique `box_loss` mesure la capacité du modèle à localiser correctement les objets détectés, notamment la précision du positionnement du centre de l'objet ainsi que le degré de recouvrement entre la boîte prédite et l'objet réel.

Enfin, la métrique `objectness` (`obj_loss`) représente la probabilité qu'un objet soit présent dans une région proposée de l'image. Une valeur élevée d'`objectness` signifie que la région analysée contient probablement un objet pertinent.

Au cours de la dernière phase de notre projet, le modèle a été validé à l'aide de l'ensemble de validation en utilisant plusieurs métriques d'évaluation (Figure III.19), notamment la précision (P), le rappel (R), ainsi que les indicateurs mAP50 (mean Average Precision à 50% d'Intersection over Union, IoU) et mAP50-95 (mean Average Precision calculée entre 50% et 95% d'IoU, par pas de 5%).

Comme le montre la Figure III.19, les modèles ont obtenu de très bonnes performances globales. Ces résultats confirment la capacité du système à détecter les chutes avec une grande fiabilité ainsi qu'une bonne généralisation sur les données de validation. Pour plus de détails, les résultats obtenus pour chaque variante de YOLO sont présentés dans le Tableau III.3, sur les trois ensembles: entraînement (2477 images), validation (710 images) et test (354 images). De manière générale, on observe que tous les modèles atteignent des performances élevées sur les trois ensembles, ce qui confirme la capacité du système à apprendre efficacement les caractéristiques liées à la détection des chutes. Les valeurs de précision et de rappel restent globalement stables et proches de 1, ce qui indique un bon équilibre entre la capacité du modèle à détecter correctement les chutes (vrais positifs) et à limiter les fausses détections.

Concernant le mAP50, toutes les versions de YOLOv5 affichent des scores très élevés (supérieurs à 0,98), ce qui montre que les modèles sont très performants pour la détection des

objets avec un seuil d'IoU de 50 %. Cela confirme que les modèles sont capables de localiser correctement les personnes ou les chutes dans la majorité des cas.

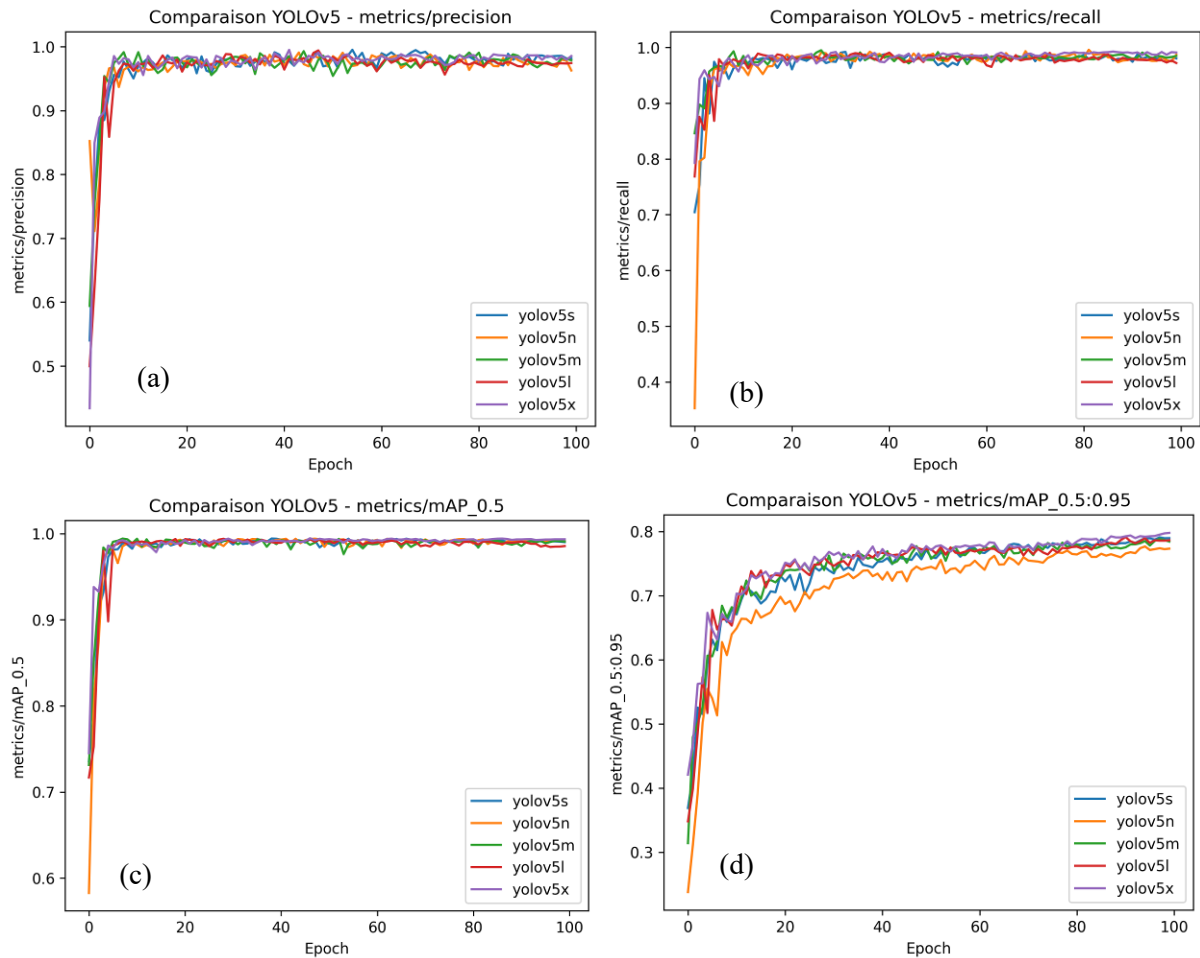


Figure III. 19. Performances des modèles YOLOv5 sur l'ensemble de validation, basées sur les différentes métriques d'évaluation : (a) précision, (b) rappel, (c) mAP50, (d) mAP50-95.

En revanche, le mAP50-95, qui est une métrique plus stricte car elle évalue la performance sur plusieurs seuils d'IoU, présente des valeurs plus modérées (entre 0,68 et 0,79 pour la plupart des modèles). Cela montre que, même si les modèles sont très efficaces pour des détections globales, la précision de localisation diminue lorsque le critère devient plus exigeant.

Sur le plan de la comparaison entre modèles, on remarque que :

- YOLOv5n présente les performances les plus faibles, ce qui est attendu car il s'agit de la version la plus légère.
- YOLOv5s et YOLOv5m offrent un bon compromis entre performance et complexité.
- YOLOv5l et YOLOv5x donnent les meilleures performances globales, avec des valeurs légèrement supérieures en précision et en mAP dans certains cas.

Tableau III. 3. Performances des modèles YOLOv5 sur les ensembles d'entraînement, de validation et de test, basées sur les différentes métriques d'évaluation : précision (P), rappel (R), mAP50 et mAP50-95.

Métriques	Entraînement	Validation	Test
YOLOv5n			
P	0,975	0,957	0,952
R	0,976	0,964	0,941
mAP50	0,992	0,99	0,974
mAP50-95	0,776	0,721	0,68
YOLOv5s			
P	0,986	0,978	0,97
R	0,982	0,983	0,96
mAP50	0,991	0,991	0,984
mAP50-95	0,791	0,743	0,715
YOLOv5m			
P	0,982	0,977	0,972
R	0,981	0,982	0,977
mAP50	0,992	0,993	0,99
mAP50-95	0,788	0,756	0,743
YOLOv5l			
P	0,981	0,984	0,982
R	0,979	0,983	0,977
mAP50	0,991	0,992	0,989
mAP50-95	0,791	0,762	0,931
YOLOv5x			
P	0,985	0,984	0,977
R	0,991	0,985	0,971
mAP50	0,993	0,992	0,985
mAP50-95	0,798	0,764	0,733

Cependant, il est important de noter que les différences entre les modèles restent globalement faibles, ce qui signifie que toutes les variantes sont capables d'assurer une détection fiable. Le choix final du modèle dépend donc principalement du compromis entre précision et contraintes de calcul, notamment pour un déploiement sur une plateforme embarquée comme la NVIDIA Jetson Nano.

Pour la visualisation, les figures suivantes (III.20 à III.24) illustrent l'application des différents modèles sur des images de test. On observe que, même dans des conditions difficiles, telles que la présence de reflets ou d'ombres, le modèle est capable d'identifier les chutes avec une bonne précision.

Il convient de noter que les images présentées ne représentent qu'un échantillon des situations rencontrées lors de la phase de test. Par conséquent, les performances du modèle peuvent varier selon les scénarios et les conditions d'éclairage ou d'environnement.

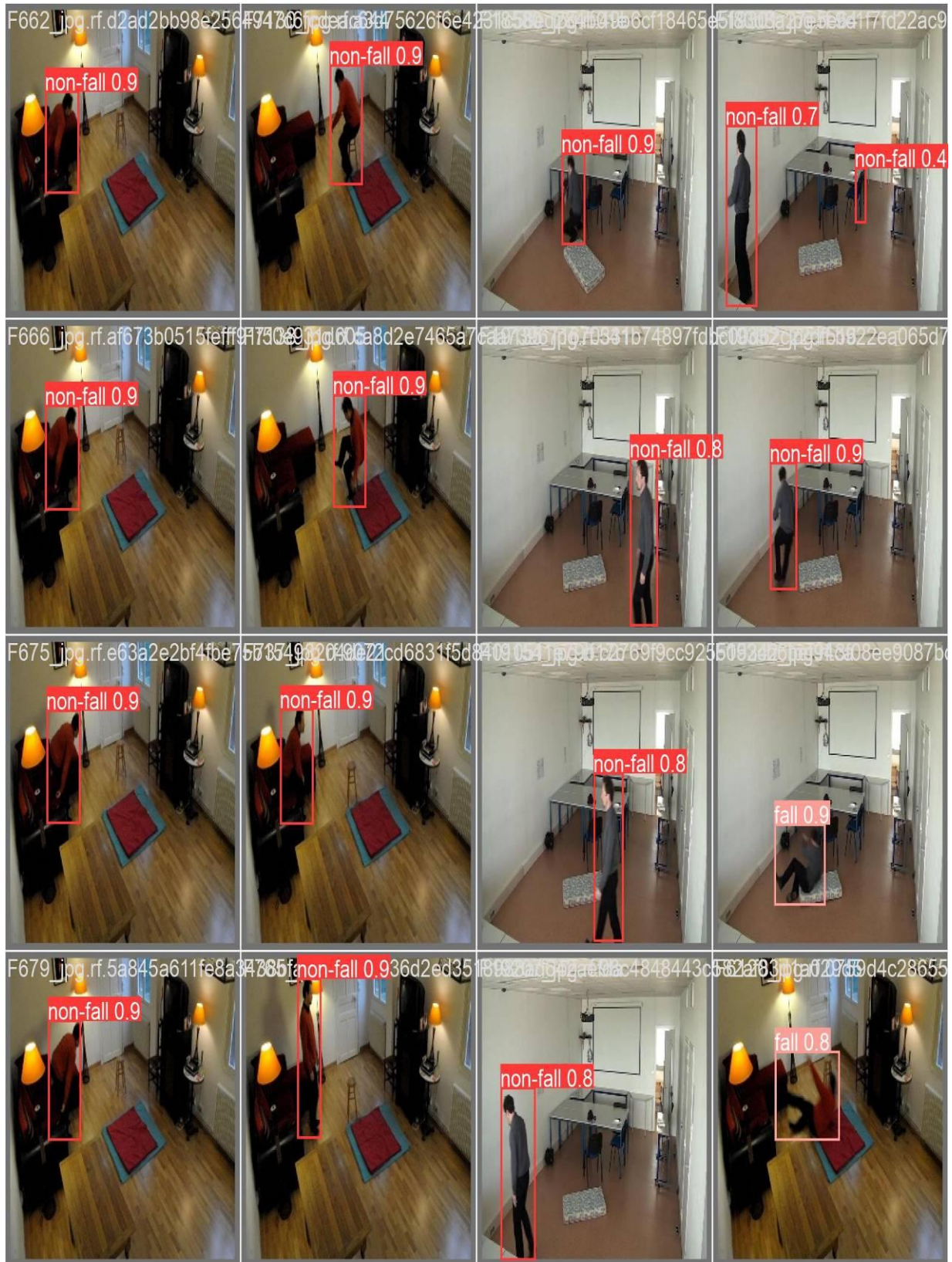


Figure III. 20. Application du modèle YOLOv5n sur des images de test.

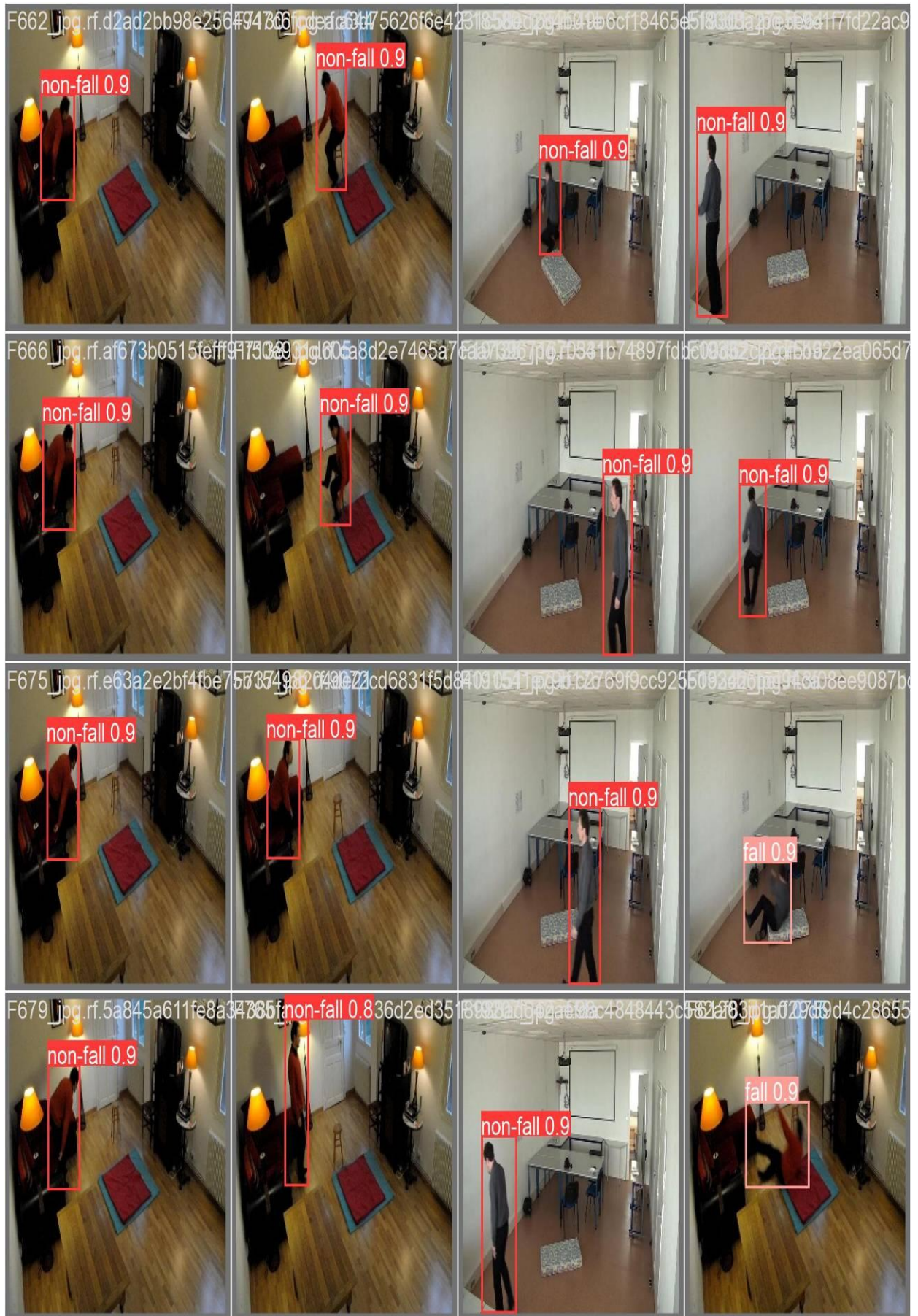


Figure III. 21. Application du modèle YOLOv5s sur des images de test.

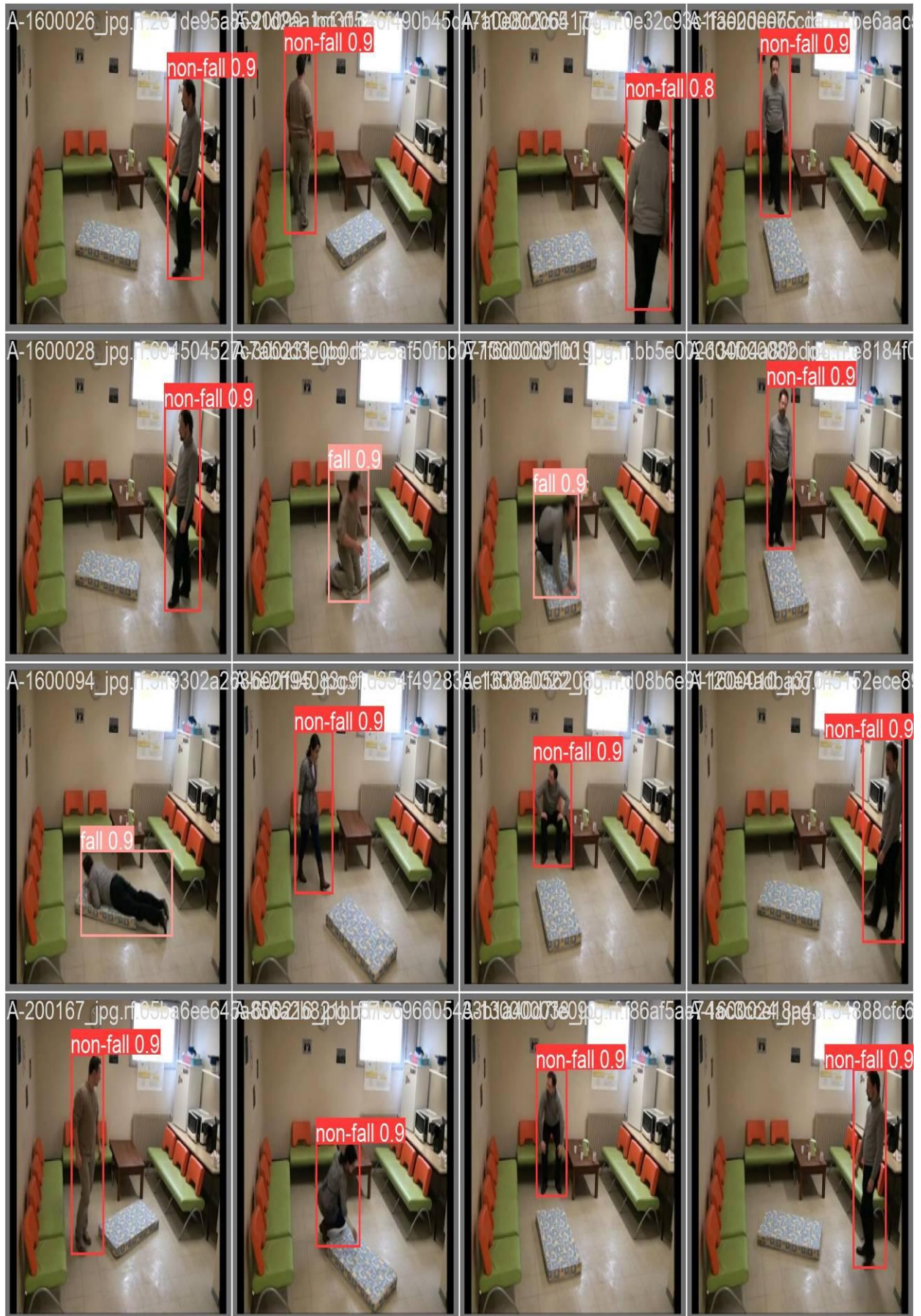


Figure III. 22. Application du modèle YOLOv5m sur des images de test.

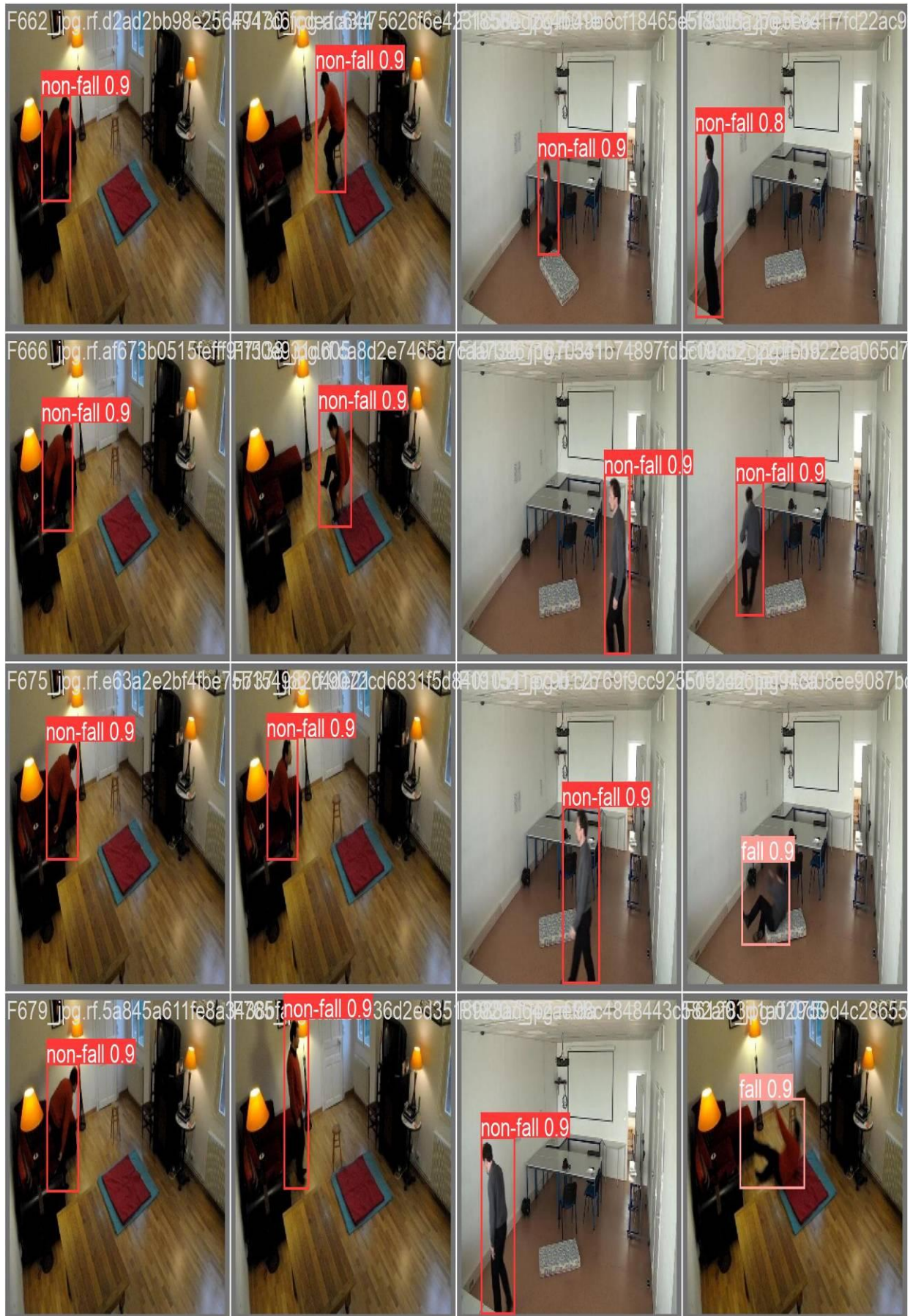


Figure III. 23. Application du modèle YOLOv5l sur des images de test.

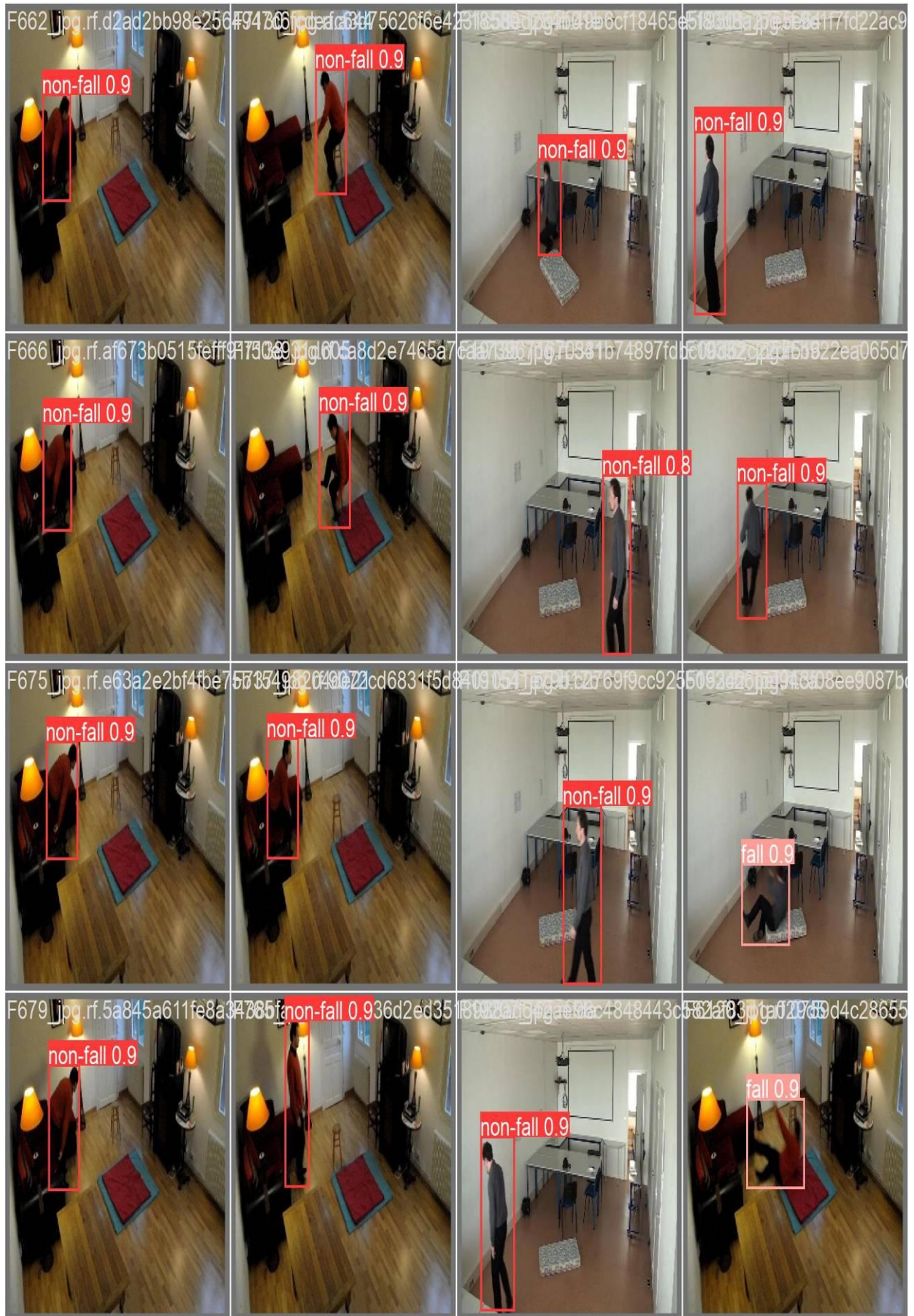


Figure III. 24. Application du modèle YOLOv5x sur des images de test.

Les modèles présentés, à savoir YOLOv5n, YOLOv5s, YOLOv5m, YOLOv5l et YOLOv5x, ont tous montré de bonnes performances en matière de précision pour la détection des chutes. Cependant, compte tenu des contraintes liées au traitement en temps réel et à la nécessité d'un temps de réponse rapide dans un système de surveillance, nous avons opté pour le modèle YOLOv5s comme base de notre projet. Ce choix s'explique par le fait que YOLOv5s constitue un excellent compromis entre rapidité d'exécution et précision de détection, ce qui le rend particulièrement adapté à un déploiement sur une plateforme embarquée.

Afin de confirmer les performances du modèle YOLOv5s sur les images de test, les Figures III.25 à III.28 présentent respectivement la matrice de confusion du modèle, la courbe de précision, la courbe de rappel ainsi que la courbe F1 obtenues après 100 époques d'entraînement.

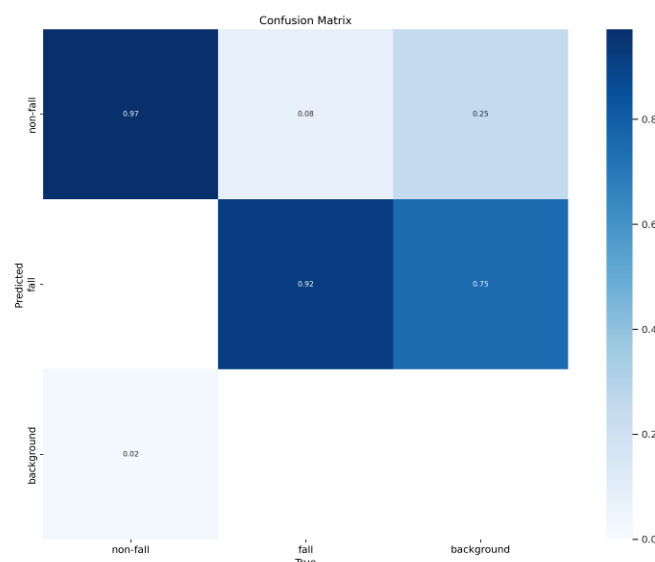


Figure III. 25. Matrice de confusion du modèle YOLOv5s après 100 époques d'entraînement.

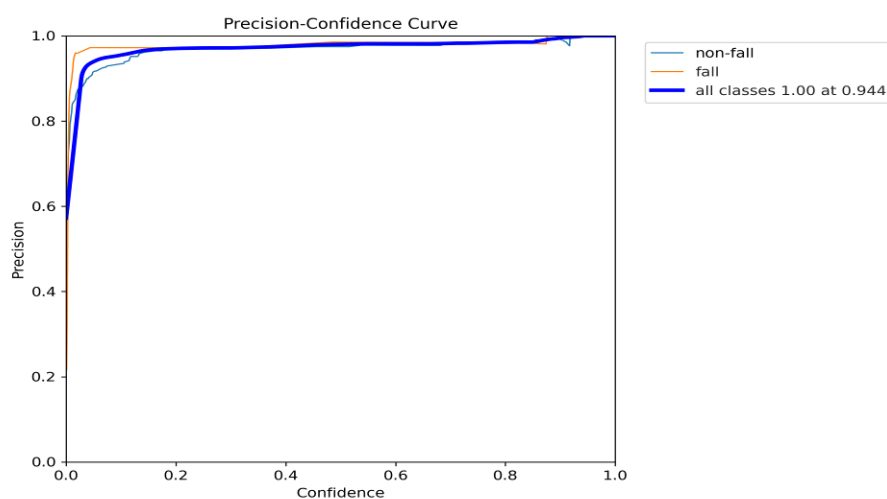


Figure III. 26. Courbe de précision du modèle YOLOv5s après 100 époques d'entraînement.

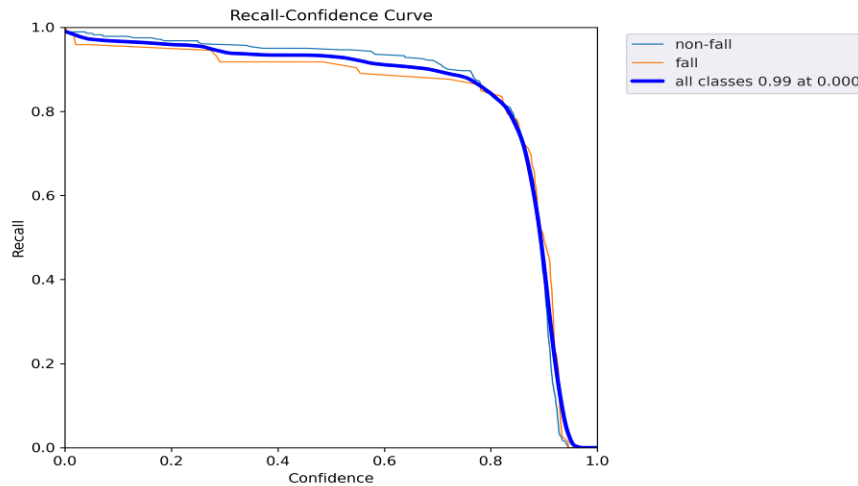


Figure III. 27. Courbe de rappel du modèle YOLOv5s après 100 époques d'entraînement.

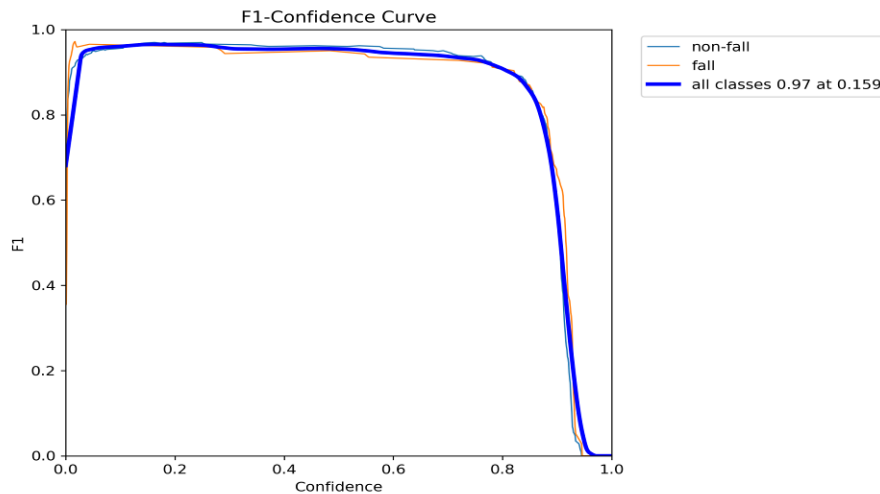


Figure III. 28. Courbe F1 du modèle YOLOv5s après 100 époques d'entraînement.

Comme le montre la Figure III.25, les performances du modèle pour les deux classes sont excellentes, ce qui confirme son efficacité pour la détection des chutes. La matrice de confusion met en évidence un très faible taux de mauvaises classifications (0.08 et 0.03), ainsi qu'un taux élevé de classifications correctes (0.97 et 0.92), traduisant une bonne capacité du modèle à distinguer les situations de chute des activités normales.

La Figure III.26 montre que le modèle présente des niveaux de précision élevés, proches de 1, ce qui indique sa capacité à prédire correctement vrais positifs avec un faible taux de fausses alertes, même lorsque le seuil de confiance augmente.

La Figure III.27 illustre l'évolution du rappel en fonction du seuil de confiance. On observe que les performances diminuent légèrement pour des niveaux de confiance élevés (supérieurs à 0,8). Cela signifie qu'à des seuils très stricts, certaines chutes peuvent être classées comme des activités normales, entraînant ainsi une augmentation des faux négatifs.

Enfin, comme le montre la Figure III.28, le score F1, qui représente la moyenne harmonique entre la précision et le rappel, suit une évolution similaire à celle du rappel. Le modèle maintient des performances très élevées jusqu'à un seuil de confiance d'environ 85 %, avant de connaître une légère diminution. Cela confirme que le modèle offre un très bon équilibre entre précision et rappel dans des conditions normales d'utilisation.

III.4 Conclusion

Dans ce chapitre, nous avons présenté l'ensemble des méthodes d'intelligence artificielle mises en œuvre pour la détection des chutes. Une étude des modèles modernes de détection d'objets basés sur les réseaux neuronaux convolutifs a été présentée, avec une attention particulière portée au modèle YOLOv5 et à ses différentes variantes.

Les résultats obtenus démontrent l'efficacité du modèle proposé, notamment en termes de précision, de robustesse et de capacité de généralisation. L'évaluation des performances, basée sur plusieurs métriques telles que la précision, le rappel, le mAP50 et le mAP50-95, a permis de valider les choix méthodologiques adoptés ainsi que la pertinence de l'utilisation des techniques de deep learning dans un système de télésurveillance embarqué.

Les expérimentations effectuées sur les différentes variantes de YOLOv5 ont montré que tous les modèles offrent de bonnes performances pour la détection des chutes. Cependant, compte tenu des contraintes liées au traitement en temps réel et à la rapidité de réponse exigée par un système de surveillance intelligent, le modèle YOLOv5s a été retenu pour la suite de notre travail.

En effet, YOLOv5s représente le meilleur compromis entre précision de détection, vitesse d'exécution et consommation de ressources, ce qui le rend particulièrement adapté à une implémentation sur une plateforme embarquée telle que la Nvidia Jetson Nano. Dans le chapitre suivant, nous présenterons l'intégration et l'implémentation pratique de ce modèle au sein du système complet de télésurveillance proposé.

III.5 Références citées dans le Chapitre III

- [1] T. Rahmani and F. Sadi, "Système embarqué pour la détection d'objets basé sur YOLOv5," *Mémoire de Master, Filière : Electronique, Spécialité : Electronique des Systèmes Embarqués, Université Mouloud Mammeri de Tizi-Ouzou*, 2023.
- [2] R. Girshick, J. Donahue, T. Darrell, J. Malik, "Rich feature hierarchies for accurate object detection and semantic segmentation," *In Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 580–587, 2014.

- [3] R. Girshick, “Fast r-cnn,” *In Proceedings of the IEEE international conference on computer vision*, pp. 1440–1448, 2015.
- [4] S Ren, K He, R Girshick, J Sun, “Faster R-CNN: Towards real-time object detection with region proposal networks,” *Advances in neural information processing systems*, pp. 91–99.
- [5] H. Loqmane and I. Khammar, “Détection automatique d’objets à partir d’un drone,” *Mémoire de Master, Université de Haute Alsace*, 2021.
- [6] J. Redmon, S.Divvala, R. Girshick, A. Farhadi, “You Only Look Once: Unified, Real-Time Object Detection,” *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 779–788, 2016.
- [7] Y. Boulifa and M. A. Ghenai, “Etude et réalisation d’un système de Tri,” *Mémoire de Master, Spécialité : Automatique Et Informatique Industrielle, Université Ibn Khaldoun-Tiaret*, 2023.
- [8] U. F. F. Kenmoe, “Utilisation de la vision artificielle en temps réel pour la détection de la dégradation de l’enveloppe structurelle des toits,” *Mémoire de Maître en sciences en géomatique appliquée et télédétection*, May 2025.
- [9] T. Oriol, “Détection des collemboles au microscope optique: utilisation du deep learning pour une analyse automatisée,” *Thèse de Doctorat*, 12 2024.
- [10] R. Xu, H. Lin, K. Lu, L. Cao, Y. Liu, “A forest fire detection system based on ensemble learning,” *Forests*, vol. 12, no. 2, p. 217, 2021.
- [11] F. Poirier, “Détection d’anomalies en temps réel dans un flux vidéo,” *Thèse de Doctorat*, Sep. 2023.
- [12] G. Jocher, “YOLOv5 by Ultralytics,” *doi: 10.5281/zenodo.3908559*, May 2020.
- [13] A. A. Matiale, “Détection de visages par YOLOv8 et YOLOv5,” *Mémoire de Master, Université du Qubec Trois-Rivières*, 2025.
- [14] W.Bey and A. Hammaz, “Collaboration entre drones pour une meilleure gestion du territoire,” *Mémoire de Master, Spécialité : Génie Informatique, Université Ibn Khaldoun-Tiaret*, 2020.
- [15] “YOLOv5 models,” <https://github.com/ultralytics/yolov5>.
- [16] E. Solan, “Evaluation de l’apport d’une approche de localisation spatiale pour la détection par Deep Learning des nodules thoraciques chez le chien et le chat,” *Thèse de Doctorat*, Sep. 2024.

-
- [17] F. Z. Guerrouj, “Méthodologie d’optimisation de la configuration matérielle-logicielle pour la détection d’objets dédiée aux véhicules autonomes,” *Thèse de Doctorat, Spécialité : Robotique*, Oct. 2025.
- [18] M. Modybalde, “État de l’art sur les architectures de Deep Learning : Perceptron, CNN et RNN,” *Mémoire de Master, Spécialité : Génie Logiciel*, Université Assane Seck de Ziguinchor, 2024.
- [19] A. Chakroun, “Accélération de l’interférence en périphérie :implantation matérielle sur circuits FPGA du modèle de détection d’objets YOLOv5 à l’aide de l’outil VITIS AI,” *Mémoire de maître en sciences appliquées*, Université du Québec à Rimouski, 2025.
- [20] M. Desmarescaux, “Intelligence artificielle pour l’automatisation du processus de sélection et de classification des colis dans un entrepôt: conception d’un système de conditionnement intelligent,” *Thèse de doctorat, Spécialité : Signal, Image, Vision, Son*, Université de Bretagne Occidentale, 2025.
- [21] S. B. Ahmed, A. Shakeel, and R. Maira, “Video Based Fall Detection System using YOLO,” p. 12, 01 2025.
- [22] C. Youla and H. Ghozlane, “Système intelligent basé sur la vision pour la détection des chutes des personnes âgées,” *Mémoire de Master, Filière : Télécommunication Spécialité : Système et Réseaux de télécommunication*, Université Badji Mokhtar-Annaba, 2021.

Chapitre IV

Conception et réalisation du système proposé

IV.1. Introduction

Après avoir défini les fondements théoriques et les choix méthodologiques dans les chapitres précédents, ce quatrième chapitre est consacré à la conception et à la réalisation du système proposé. L'objectif est de traduire les spécifications établies en une solution concrète, intégrant à la fois les aspects matériels et logiciels.

La phase de conception permet de structurer l'architecture globale du système, en mettant en évidence les différentes étapes de la chaîne de traitement des données, depuis l'acquisition jusqu'à la génération d'alertes, ainsi que les interactions entre les différents équipements matériels tels que la plateforme Nvidia Jetson, le smartphone Android et le point d'accès Wi-Fi. La phase de réalisation consiste à implémenter les différents modules sur la plateforme matérielle choisie, à savoir la carte Nvidia Jetson, tout en assurant l'optimisation du modèle, l'exploitation de l'accélération GPU et la gestion du temps réel.

Ce chapitre présente également la mise en place d'un serveur de diffusion et de communication, ainsi que le développement d'une application Android destinée à fournir une interface conviviale et une gestion efficace des alertes.

Pour une bonne compréhension, ce chapitre est organisé comme suit:

- **Architecture globale du système :** présentation du système proposé avec ses différents composants matériels (Jetson Nano, caméras, point d'accès Wi-Fi) et leurs interactions.
- **Système de détection sur Jetson :** configuration de la carte Nvidia Jetson Nano, installation des bibliothèques et mise en œuvre du modèle de détection.
- **Développement de l'Application Android :** conception de l'interface utilisateur, gestion des contacts d'urgence et notifications.
- **Validation expérimentale :** mise en place du prototype et tests en conditions réelles.
- **Résultats :** analyse des performances du système (détection des chutes, activités normales, fiabilité des alertes).

IV.2. Architecture globale du système

Comme l'illustre la Figure IV.1, contrairement aux systèmes traditionnels qui reposent sur des capteurs portés par la personne ou sur des capteurs installés au sol, le système proposé est basé sur la vision par ordinateur et l'intelligence artificielle. Il nécessite l'utilisation d'une caméra RGB ou d'une caméra IP pour l'acquisition des images, ainsi qu'une plateforme embarquée, la carte Nvidia Jetson Nano, chargée du traitement des données. La détection des

chutes est réalisée à l'aide de l'algorithme YOLOv5, capable d'identifier automatiquement les situations de chute en temps réel.

Par ailleurs, une application Android a été développée en utilisant le langage de programmation Kotlin afin de faciliter l'utilisation du système par les personnes âgées. Cette application permet non seulement la réception et la consultation des alertes, mais également l'exploitation des fonctionnalités du smartphone pour l'émission d'appels téléphoniques ou l'envoi de SMS d'urgence. Ainsi, le réseau GSM du téléphone est utilisé pour assurer la transmission rapide des alertes vers les services de santé, tels que les hôpitaux et les services d'urgence (ambulance, protection civile, etc.), ainsi que vers les aidants, les proches et les médecins, garantissant une intervention rapide en cas de situation critique.

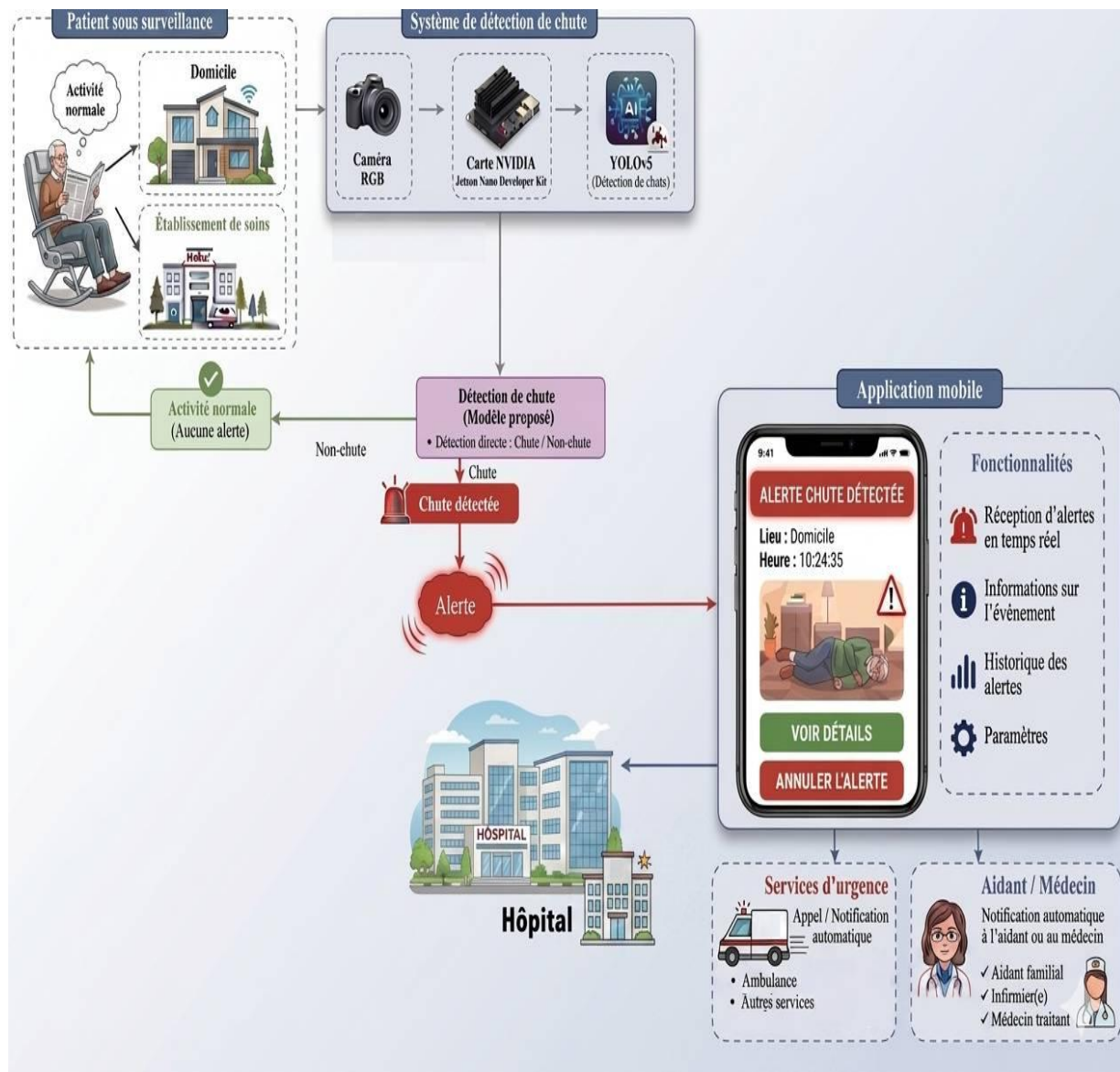


Figure IV. 1. Système proposé de surveillance en temps réel.

Pour plus de détails, comme l'illustre la Figure IV.2, le système proposé est composé de plusieurs modules interconnectés assurant l'acquisition, le traitement, la transmission et l'affichage des données. Il comprend principalement une unité de traitement, des capteurs visuels (caméras), un point d'accès réseau ainsi qu'une application Android dédiée à la surveillance et à la gestion des alertes.

L'unité de traitement, basée sur la plateforme Nvidia Jetson, assure l'exécution des algorithmes de détection en temps réel à partir des flux vidéo capturés par les caméras. Les données traitées sont ensuite transmises via le point d'accès Wi-Fi vers l'application Android, permettant à l'utilisateur de recevoir des notifications et de superviser l'état du système à distance.

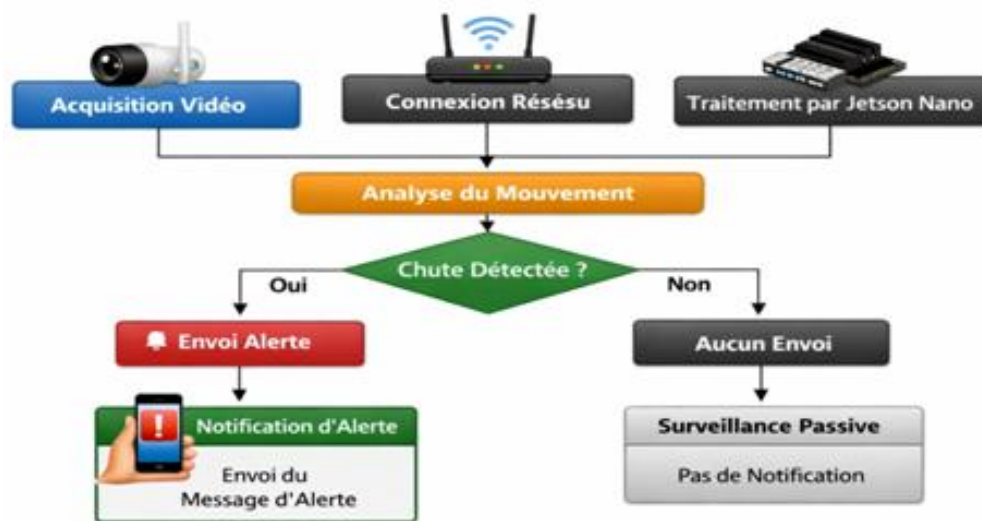


Figure IV. 2. Architecture globale du système de surveillance et de détection de chute.

Cette architecture garantit une communication efficace entre les différents composants matériels et logiciels, tout en assurant une détection rapide et fiable des chutes en temps réel.

Les unités de traitement peuvent inclure de puissants microcontrôleurs, des ordinateurs mono-cartes ou des processeurs IA dédiés. Par exemple, le Nvidia Jetson Nano, qui fournit la puissance de calcul et les ressources nécessaires pour exécuter des algorithmes complexes et des modèles d'IA. En intégrant les méthodes d'intelligence artificielle, ces unités de traitement sont capables d'implémenter des algorithmes avancés tels que la détection d'objets et le traitement d'images et de données en temps réel.

Unité de traitement : Le Nvidia Jetson Nano, grâce à ses capacités avancées de traitement, est particulièrement adapté à l'exécution des tâches nécessitant une forte puissance de calcul. Cette plateforme permet l'implémentation de modèles de deep learning et de machine learning, ce qui est crucial pour la détection d'objets et l'analyse de données en temps réel.

En exploitant cette technologie, notre système est capable non seulement de collecter et de traiter efficacement les données issues des caméras, mais également de prendre des décisions rapides et précises afin d'assurer une surveillance fiable et d'améliorer la sécurité ainsi que la qualité de vie des personnes âgées, tout en respectant leur vie privée.

Caméras : Comme mentionné dans le chapitre II, afin d'évaluer les performances des différentes méthodes d'intelligence artificielle proposées dans le chapitre précédent, nous avons utilisé deux types de capteurs visuels: la caméra "Havit-HV-N5080" et la caméra IP "Imou Bullet 2E". L'utilisation de ces caméras permet de transformer la carte Nvidia Jetson en Webcam, en caméra IP ou encore en système de vidéosurveillance intelligent.

Dans notre projet, les caméras IP jouent un rôle essentiel en assurant la capture et la transmission des données visuelles via le réseau IP, tout en permettant une intégration fluide avec les différents composants du système de sécurité.

De plus, ces caméras sont indispensables pour la surveillance en temps réel, car elles fournissent un retour visuel continu sur la zone surveillée et permettent ainsi une détection rapide des événements anormaux, tels que les chutes des personnes âgées.

Point d'accès : Un point d'accès est un dispositif qui sert de centre de communication sans fil, permettant aux appareils sans fil de se connecter à un réseau câblé. Il crée un réseau local (LAN) en diffusant un signal Wi-Fi que les appareils peuvent détecter et auquel ils peuvent se connecter. Les points d'accès jouent un rôle crucial dans la mise en place de la connectivité sans fil dans les maisons, les bureaux et les espaces publics.

Dans notre projet, le point d'accès a pour rôle d'établir et de gérer les connexions sans fil avec les appareils de notre système de sécurité. Il agit comme une station de base qui diffuse un signal Wi-Fi et permet à des appareils tels que la caméra IP, l'ordinateur portable, le smartphone et le Nvidia Jetson Nano de se connecter sans fil.

De plus, lorsqu'un appareil de notre système de sécurité souhaite communiquer, il se connecte au point d'accès via le réseau Wi-Fi de celui-ci, qui attribue une adresse IP à l'appareil connecté, servant ainsi d'identifiant unique dans le réseau. Ce système d'adressage permet aux appareils d'envoyer et de recevoir des paquets de données vers et depuis les autres appareils du réseau.

En outre, les points d'accès jouent un rôle dans la gestion de la sécurité du réseau, où nous pouvons mettre en œuvre des mesures telles que le chiffrement et l'authentification afin de garantir que seuls les appareils autorisés puissent se connecter. Cela contribue à protéger notre système de sécurité contre les accès non autorisés et les violations de données.

Par ailleurs, le point d'accès peut prendre en charge des fonctionnalités comme la qualité de service (QoS) et la priorisation du trafic. Cela nous permet d'allouer efficacement la bande passante et de garantir que les données critiques, telles que les alertes de sécurité ou les flux vidéo en temps réel, soient prioritaires par rapport au trafic moins sensible au temps. Ainsi, la performance et la réactivité de notre système de sécurité sont préservées.

Application Android : En cas de détection positive, un message d'alerte est automatiquement envoyé vers le smartphone Android de l'utilisateur ou du personnel médical à travers l'application dédiée. Si aucune chute n'est détectée, le système reste en veille sans générer d'alerte. Cette architecture permet une communication fluide entre les différents modules et garantit une réaction rapide en situation critique, tout en minimisant les fausses alarmes.

IV.2.1. Chaîne de traitement de données

La chaîne de traitement du système de détection de chute s'articule autour d'un flux d'informations bien défini, tel qu'illustré à la Figure IV.3. Les données visuelles sont d'abord captées par la caméra IP, qui constitue le module d'acquisition. Ces flux sont ensuite transmis via le réseau Wi-Fi vers la carte Jetson Nano, où s'effectue le traitement principal. L'algorithme embarqué analyse les séquences vidéo afin d'identifier les indicateurs caractéristiques d'une chute. Lorsque la chute est confirmée, le système déclenche automatiquement la fonction notification d'alerte, envoyée vers le smartphone Android via l'application dédiée. En l'absence de chute, le dispositif poursuit son rôle de surveillance passive sans générer de messages. Cette organisation en chaîne assure une interaction fluide entre acquisition, traitement et communication, garantissant une solution efficace et adaptée aux besoins de protection des personnes âgées.

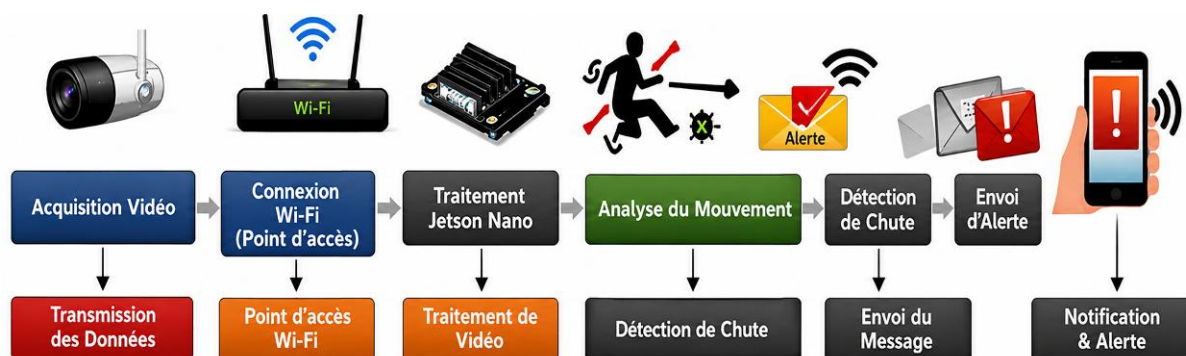


Figure IV. 3. Chaîne de traitement des données de système de détection la chute.

La chaîne de traitement des données du système se déroule selon une succession logique d'étapes allant de la capture à la notification.

1. **Acquisition des données vidéo** : la caméra IP Imou Bullet 2E enregistre en continu les mouvements de la personne surveillée.
2. **Transmission des données** : les flux vidéo sont envoyés via le point d'accès Wi-Fi vers la carte Jetson Nano.
3. **Analyse du mouvement** : les algorithmes de vision par ordinateur calculent la chute.
4. **Décision de chute** : si les paramètres dépassent les seuils définis, une chute est confirmée.
5. **Envoi d'alerte** : le module de communication transmet automatiquement un message via le wifi vers le smartphone Android.
6. **Affichage et notification** : l'application mobile affiche une alerte sonore et visuelle pour informer le personnel médical ou la famille.

IV.3. Système de détection sur Jetson

Pour réaliser notre système, la première étape consiste à intégrer le système de détection de chute sur la plateforme Nvidia Jetson Nano, en exploitant ses capacités de calcul adaptées aux applications d'intelligence artificielle embarquée. Les flux vidéo capturés par la caméra IP sont transmis via le réseau Wi-Fi vers la carte Jetson Nano, où ils sont traités en temps réel. Les algorithmes de vision par ordinateur et de traitement d'image sont optimisés afin de tirer parti du GPU intégré, ce qui permet une analyse rapide et efficace des mouvements humains. Cette intégration garantit non seulement la détection fiable des chutes, mais également une faible consommation énergétique, ce qui rend le système adapté à une utilisation continue dans un environnement domestique ou hospitalier. De plus, la Jetson Nano offre une flexibilité pour le développement futur, permettant l'ajout de nouvelles fonctionnalités comme la reconnaissance de comportements ou l'adaptation dynamique des seuils de détection.

IV.3.1. Configuration de la carte Nvidia

La configuration de la Jetson Nano constitue une étape essentielle pour assurer le bon fonctionnement du système de détection de chute (Figure IV.4). La carte est généralement configurée sous Ubuntu Linux, avec l'installation du kit de développement Nvidia JetPack qui intègre les bibliothèques nécessaires telles que CUDA pour l'accélération GPU, cuDNN pour l'optimisation des réseaux neuronaux, et TensorRT pour l'inférence rapide des modèles. Le système est ensuite enrichi par des frameworks comme OpenCV pour le traitement d'image et PyTorch pour l'implémentation des algorithmes d'apprentissage profond. La configuration inclut également la mise en place de la connexion réseau Wi-Fi pour assurer la communication avec la caméra IP et le smartphone Android. Cette organisation logicielle et matérielle permet d'obtenir une plateforme embarquée performante, capable d'exécuter en

temps réel les algorithmes de détection de chute tout en maintenant une consommation énergétique réduite.



Figure IV. 4. Configuration de la carte Nvidia Jetson Nano.

L'utilisation d'une clé USB avec la Jetson Nano permet de simplifier le transfert et la sauvegarde des fichiers nécessaires au fonctionnement du système. Après insertion de la clé dans l'un des ports USB, le système Ubuntu la détecte automatiquement et la monte pour un accès direct. L'utilisateur peut alors copier les scripts de détection de chute, les modèles d'apprentissage ou les fichiers de configuration vers le répertoire de travail. Cette méthode offre une solution rapide et pratique pour déployer les programmes sans connexion réseau, tout en facilitant la mise à jour ou la sauvegarde des données du projet.



Figure IV. 5. Utilisation d'une clé USB avec la Jetson.

Le schéma présenté à la Figure IV.5 met en évidence, de manière claire, les différentes étapes essentielles du processus :

1. **Insertion de la clé USB** dans un port de la Jetson Nano.
2. **Montage automatique** par Ubuntu.
3. **Accès aux fichiers** via explorateur ou terminal.
4. **Transfert des scripts** vers le dossier de travail.
5. **Exécution du programme** directement depuis la Jetson Nano.

L'utilisation d'une clé USB dans le cadre du système Jetson Nano représente une solution pratique et efficace pour le transfert et la mise à jour des fichiers. Elle simplifie considérablement le déploiement des programmes en permettant de copier directement les scripts de détection de chute ou les fichiers de configuration sans recourir à une connexion

réseau. Cette méthode garantit une rapidité d'exécution et une sécurité accrue des données, tout en offrant une flexibilité appréciable pour les interventions sur le terrain. Ainsi, l'intégration de la clé USB constitue un atout majeur pour assurer la continuité et la fiabilité du projet, notamment dans les environnements où l'accès à Internet est limité ou inexistant.

IV.3.2. Principaux étapes de la configuration de la carte Nvidia

Pour commencer, nous devons écrire un système d'exploitation sur la carte microSD, le Jetson nano utilise une carte microSD comme périphérique de démarrage et pour le stockage principal.

Il est essentiel d'avoir une carte rapide et volumineuse pour notre projet, nous avons utilisé une carte 128 Go. Dans cette étape, nous avons besoin d'un ordinateur avec connexion Internet et d'un lecteur de carte SD :

Étape 1: Téléchargez l'image JetPack 4.5 sur le site Web de Nvidia, puis suivez ces étapes pour écrire correctement l'image sur la carte microSD à l'aide de notre ordinateur Windows.

- ✓ Formater la carte microSD à l'aide du formateur de carte mémoire SD de l'association SD.
- ✓ Téléchargez, installez et lancez Etcher pour écrire l'image sur la carte SD. une fois la carte microSD prête, nous procédons à la configuration et à la première phase de démarrage. Il existe deux façons d'interagir avec le developer kit :
 - 1) avec écran, clavier et souris connectés.
 - 2) en "headless mode" depuis un autre ordinateur.

Étape 2: Configuration initiale avec écran connecté :

- 1) Insérez la carte microSD (avec l'image JetPack 4.5 déjà écrite).
- 2) Placez developer kit sur une surface non conductrice avec l'écran allumé et le clavier et la souris USB connectés.
- 3) Connectez ensuite l'alimentation USB-C (5V-4A) et le Jetson Nano s'allumera et démarrera automatiquement.

Étape 3 (Premier démarrage) : Lors du premier démarrage, le système vous guidera à travers une configuration initiale, notamment :

- ✓ Sélectionner la langue du système, la disposition du clavier et le fuseau horaire.
- ✓ Créer un nom d'utilisateur, un mot de passe et un nom d'ordinateur.
- ✓ Sélectionnez la taille de la partition APP. il est recommandé d'utiliser la taille maximale suggérée.

Maintenant nous avons un système Linux basé sur Ubuntu prêt à être flashé sur la carte SD, nous procédons.

En suivant ces étapes, nous aurons un environnement de développement complet et fonctionnel sur la carte Jetson Nano, prêt à exécuter et développer des programmes en utilisant le Terminal.

IV.3.3. Terminal

Puisque l'image du système Nvidia Jetson Nano est basée sur Ubuntu, il est possible d'utiliser le logiciel Terminal pour effectuer des tâches d'administration, telles que la création et la modification de fichiers et dossiers, ainsi que la compilation et l'exécution de programmes. Le Terminal de la Nvidia Jetson peut être ouvert en cliquant sur l'option Rechercher, située en haut à gauche de l'interface. Ensuite, il suffit de saisir «Terminal» pour afficher l'application correspondante. Après avoir cliqué sur l'icône Terminal, l'interface de commande s'ouvre, comme illustré dans la Figure IV.6.

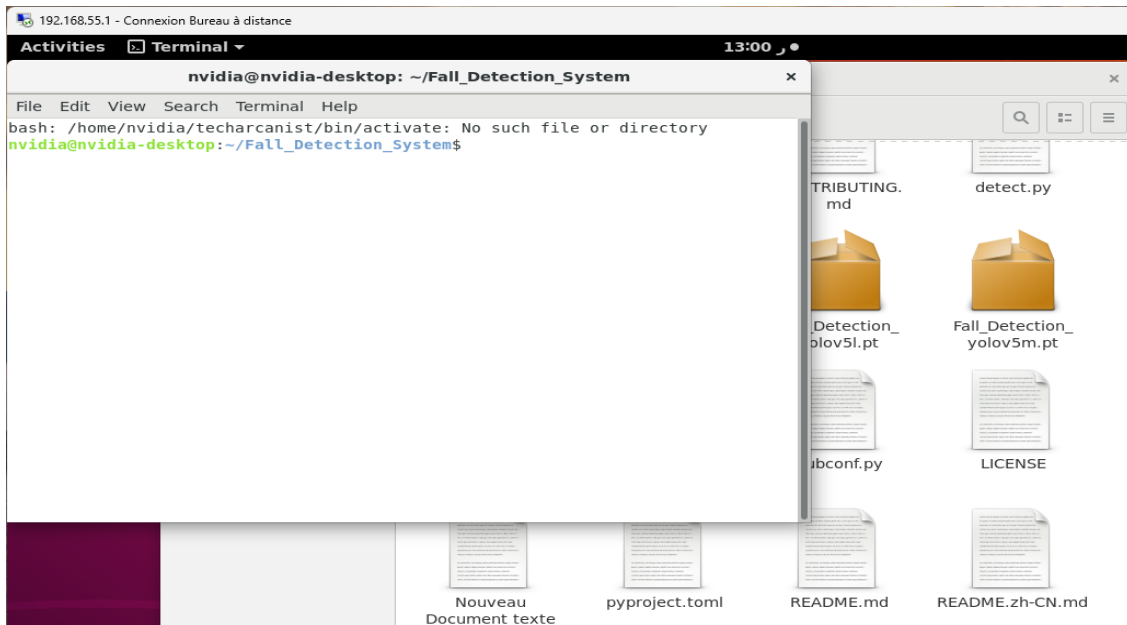


Figure IV. 6. Application du terminal.

IV.3.4. Serveur de diffusion et communication

Le serveur de communication présenté à la Figure IV.7, hébergé sur la carte Jetson Nano, joue un rôle fondamental dans l'architecture du système. Il est responsable de la gestion simultanée de deux types de flux : le flux vidéo et les alertes de détection.

Pour la partie vidéo, un serveur MJPEG est configuré afin de diffuser en continu les images capturées par la caméra IP via le réseau Wi-Fi, en utilisant le protocole HTTP sur un port dédié (généralement 8080).

Ce choix garantit une compatibilité avec la majorité des navigateurs et applications clientes. En parallèle, un serveur WebSocket est implémenté pour assurer une communication bidirectionnelle en temps réel avec le smartphone Android.

Les messages transmis sont structurés au format JSON, ce qui facilite leur interprétation et leur intégration dans l'application mobile.

Cette architecture repose sur une séparation claire des flux : le flux vidéo est traité indépendamment des alertes, ce qui améliore la robustesse et la fiabilité du système.

Ainsi, l'utilisateur peut visualiser en direct le flux vidéo tout en recevant instantanément des notifications en cas de chute détectée, garantissant une surveillance efficace et réactive.

Le système repose sur quatre éléments matériels principaux :

- 1) **Carte Jetson Nano (serveur)**: cœur du traitement et de la gestion des flux. Elle héberge le serveur MJPEG pour la vidéo et le serveur WebSocket pour les alertes.
- 2) **Caméra IP Imou Bullet 2E**: source du flux vidéo, connectée au Jetson Nano via le réseau Wi-Fi fourni par le modem/routeur.
- 3) **Modem Wi-Fi (Routeur)**: point d'accès qui relie la caméra IP, le Jetson Nano et le smartphone.
- 4) **Smartphone Android (client)**: reçoit le flux vidéo via MJPEG et les alertes via WebSocket, affichées dans l'application mobile.

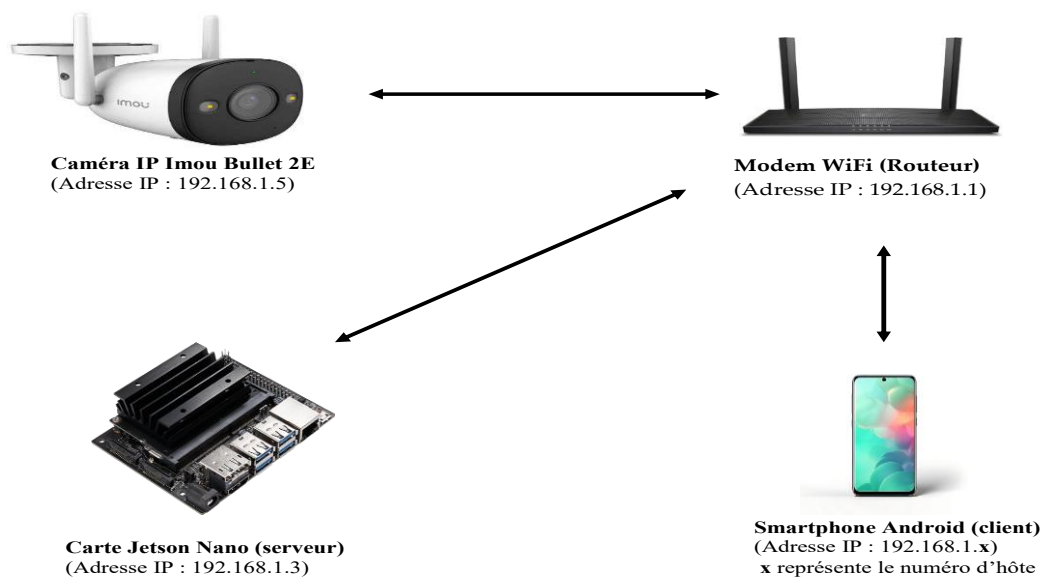


Figure IV. 7. Schéma du serveur de diffusion et communication.

IV.4. Développement de l'Application Android

L'application AI Safety Emergency Monitor est conçue pour assurer la protection des personnes âgées ou vulnérables grâce à un système intelligent de détection de chute. Elle repose sur l'analyse en temps réel des mouvements capturés par une caméra connectée et traités par une unité embarquée.

Lorsqu'une chute est détectée, l'application envoie immédiatement une alerte vers le smartphone du proche ou du personnel médical, garantissant une intervention rapide en cas

d'urgence. Ce dispositif offre une surveillance continue, réduit les risques liés à l'isolement et contribue à renforcer la sécurité et la tranquillité d'esprit des utilisateurs et de leurs familles.



Figure IV. 8. Logo de l'application «Al Safety Monitor».

Dans les sous-sections suivantes, nous expliquons en détail l'utilisation de notre application, développée à l'aide de l'environnement Android Studio et du langage de programmation Kotlin.

Nous présentons également les différentes fonctionnalités de l'application, son interface utilisateur ainsi que les étapes de communication avec le système de détection afin d'assurer la réception et la gestion des alertes en temps réel.

IV.4.1. Connexion à l'application

Cela se fait en appuyant sur l'icône de l'application «Al Safety Monitor» présente dans le Smartphone.

Avant de procéder à l'inscription, l'utilisateur sélectionne en haut à droite, comme illustré à la Figure IV.9, la langue d'enregistrement souhaitée, soit l'arabe soit l'anglais. Il accède ensuite à l'application en cliquant sur le bouton (*Get Started*).



Figure IV. 9. Interface de l'application «Al Safety Monitor» (a) Connexion en langue arabe et (b) Connexion en langue anglais.

IV.4.2. Remplir les champs

Après avoir appuyé sur le bouton (*Get Started*), nous saisissons le nom d'utilisateur et le mot de passe de l'utilisateur. Ensuite, nous appuyons sur (*Login*). Il est possible d'appuyer sur (*Remember Password*) afin de mémoriser le mot de passe et faciliter la connexion lors d'une prochaine utilisation, sans avoir à remplir de nouveau les données précédentes. La Figure IV.10 illustre l'interface correspondante.

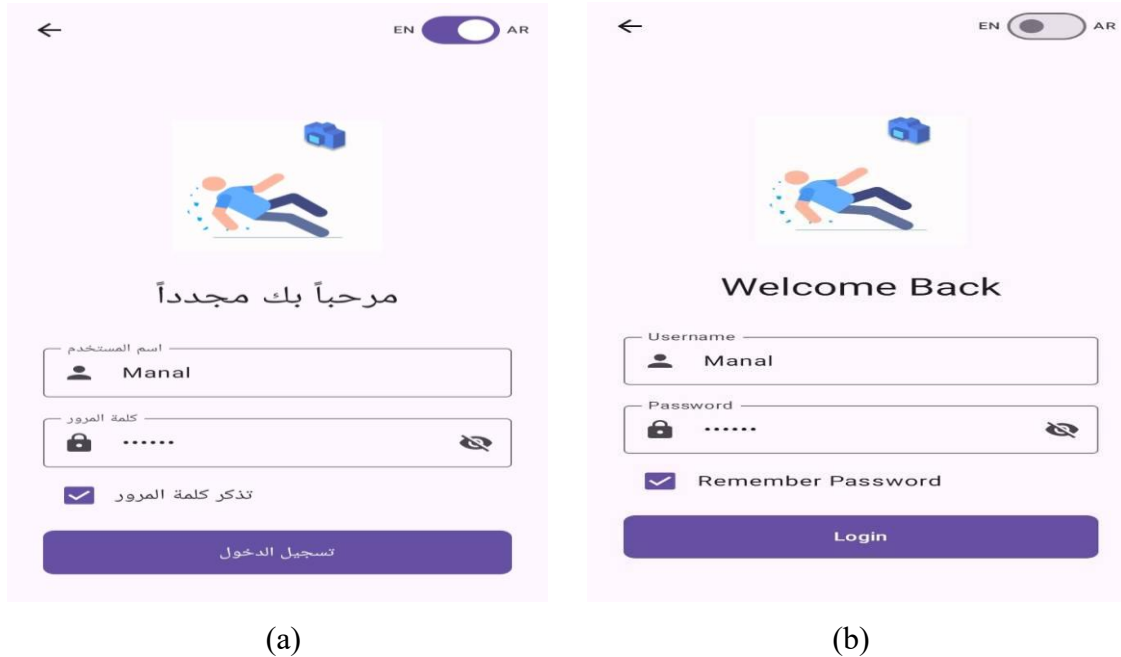


Figure IV. 10. Interface et saisie des données : (a) en arabe et (b) en anglais.

IV.4.3. Accéder à l'application et ouvrir l'interface du tableau de bord

Après une authentification réussie, l'utilisateur est redirigé vers le tableau de bord, qui constitue l'interface principale de l'application. Cette interface centralise les différentes fonctionnalités du système et permet un accès rapide aux modules de surveillance et de gestion. Comme illustré dans la figure suivante, le tableau de bord est composé de cinq sections principales :

- ✚ **Lien des appareils (Hardware Link):** permet de gérer la connexion et la communication entre l'application mobile et les équipements matériels du système.
- ✚ **Détection de chute (Fall Detection):** assure la surveillance en temps réel et l'affichage des alertes liées aux chutes détectées.
- ✚ **Contacts d'urgence (Emergency Contacts):** permet d'ajouter, de modifier ou de supprimer les contacts à prévenir en cas d'incident.
- ✚ **Horaires d'urgence (Emergency Times):** permet de définir les plages horaires d'activation des alertes.

🚦 **Informations personnelles (Personal Information):** regroupe les informations relatives à l'utilisateur et aux paramètres de son profil.

Cette organisation facilite la navigation au sein de l'application et offre un accès intuitif à l'ensemble des fonctionnalités nécessaires au suivi et à la gestion des alertes de chute.



Figure IV. 11. Interface du tableau de bord de l'application mobile: version en arabe (à gauche) et version en anglais (à droite).

Lien des appareils (Hardware Link) : En appuyant sur (Hardware Link), trois champs apparaissent comme suit :

- a) **Jetson IP / Host :** La carte Jetson Nano agit comme hôte IP, c'est le cœur du système. Elle reçoit les flux vidéo de la caméra, les analyse en temps réel et gère la communication avec l'application Android.
- b) **Stream Port (MJPEG):** Ce port est dédié au flux vidéo compressé en Motion JPEG. Il permet de transmettre les images de la caméra sous forme de vidéo continue, assurant une visualisation fluide et compatible avec de nombreux appareils.
- c) **WebSocket Port (Alerts):** Ce port est réservé aux alertes. Grâce au protocole WebSocket, il envoie instantanément des notifications en cas de détection de chute, garantissant une communication rapide et fiable entre le Jetson et le smartphone.

Figure IV. 12. Les champs de «Hardware Link» : version en arabe (à gauche) et version en anglais (à droite).

Détection de chute (Fall Detection) : La figure suivante présente la fonctionnalité de détection de chute, qui constitue le cœur du système. Lors de l'activation du bouton « Fall Detection », une fenêtre s'ouvre et affiche le message « Stream is Offline » lorsque la caméra n'est pas connectée.

En revanche, en sélectionnant l'option (*Connect to Stream*), l'utilisateur accède au flux vidéo en direct et peut visualiser la personne surveillée, qu'il y ait chute ou non.

Dans le cas où une chute est effectivement détectée, le système déclenche automatiquement l'envoi d'un message d'alerte vers les contacts d'urgence enregistrés.

Ce mécanisme assure une surveillance continue et une réaction immédiate, renforçant la sécurité et la fiabilité du dispositif.

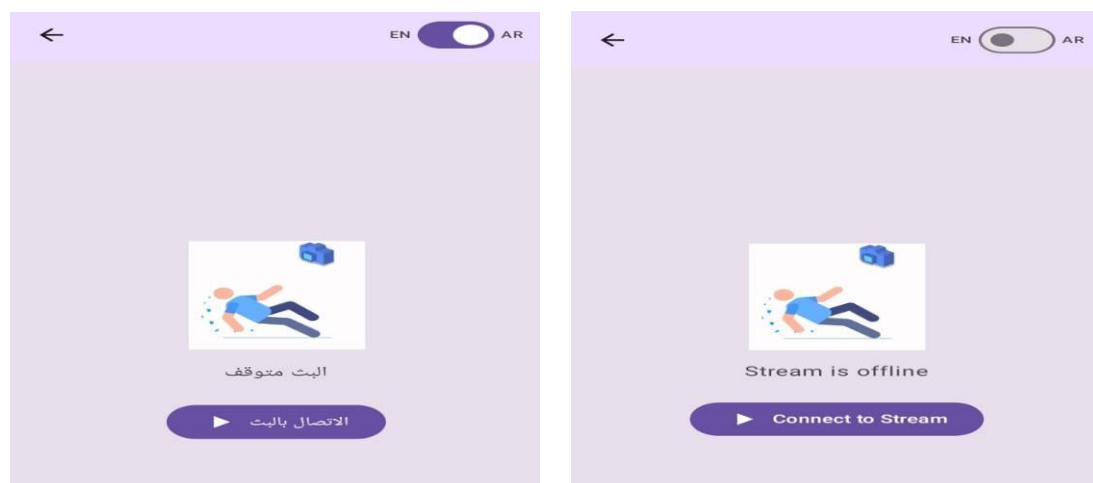


Figure IV. 13. Détection de chute: version en arabe (à gauche) et version en anglais (à droite).

Contacts d'urgence (Emergency Contacts): La rubrique Contacts d'urgence joue un rôle essentiel dans le système de détection de chute. Elle permet d'enregistrer les coordonnées des personnes à prévenir immédiatement en cas d'incident, telles que les proches, les aidants ou les référents médicaux.

Lorsqu'une chute est détectée, le système envoie automatiquement une notification vers ces contacts afin de garantir une intervention rapide.

Cette fonctionnalité renforce la sécurité de l'utilisateur en assurant que les alertes ne restent pas sans réponse et que l'assistance nécessaire est mobilisée dans les plus brefs délais. La figure suivante présente le message d'urgence généré par le système, disponible en deux langues, à savoir l'arabe et l'anglais, afin de garantir une meilleure accessibilité pour les utilisateurs.

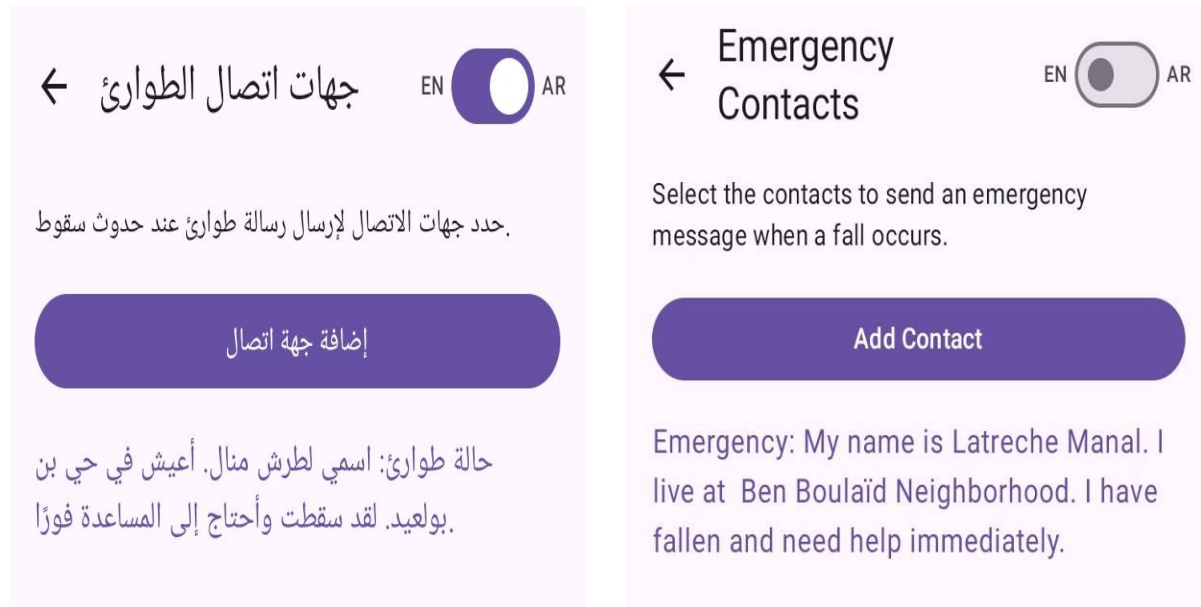


Figure IV. 14. Écran de gestion des contacts d'urgence de l'application : interface en arabe (à gauche); interface en anglais (à droite).

Le système permet à l'utilisateur d'ajouter des contacts d'urgence afin qu'ils soient automatiquement alertés en cas de situation critique, notamment lors de la détection d'une chute. Chaque contact est enregistré avec son nom et son numéro de téléphone. La Figure IV.15 présente l'interface dédiée à la gestion des contacts d'urgence.

Dans notre système, nous exploitons les fonctionnalités offertes par le téléphone mobile, en particulier le réseau GSM, pour assurer la transmission des alertes. Ainsi, lorsqu'un événement anormal est détecté, l'application peut envoyer automatiquement des notifications ou des messages aux contacts d'urgence enregistrés, permettant une intervention rapide et renforçant la sécurité de la personne surveillée.

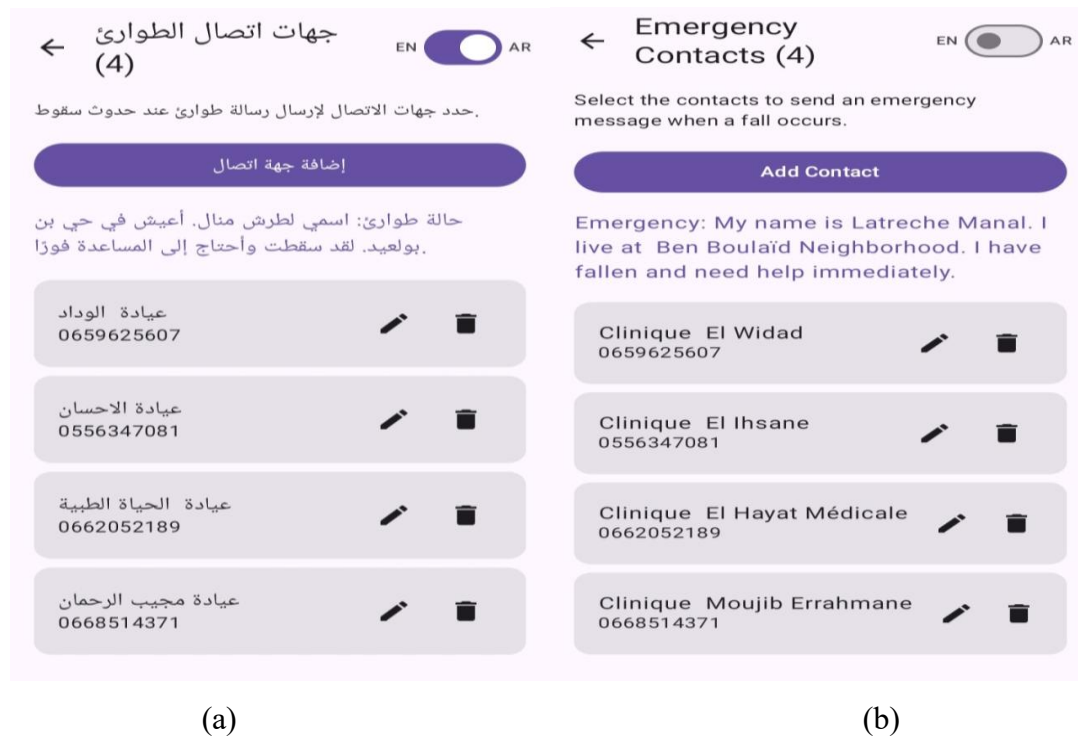


Figure IV. 15. Intégration des contacts d'urgence dans l'application : a) en arabe et b) en anglais.

Horaires d'urgence (Emergency Times): La section Horaires d'urgence (Emergency Times), présentée à la Figure IV.16, permet à l'utilisateur de paramétrer les délais associés au mécanisme d'alerte du système. Elle comprend deux réglages principaux : le délai de la première alerte (First Alert), qui détermine l'intervalle de temps précédant l'envoi de la notification initiale après la détection d'une chute, et le délai de la seconde alerte (Second Alert), qui définit le temps supplémentaire au terme duquel une nouvelle notification est envoyée si aucune prise en charge ou action corrective n'a été signalée.

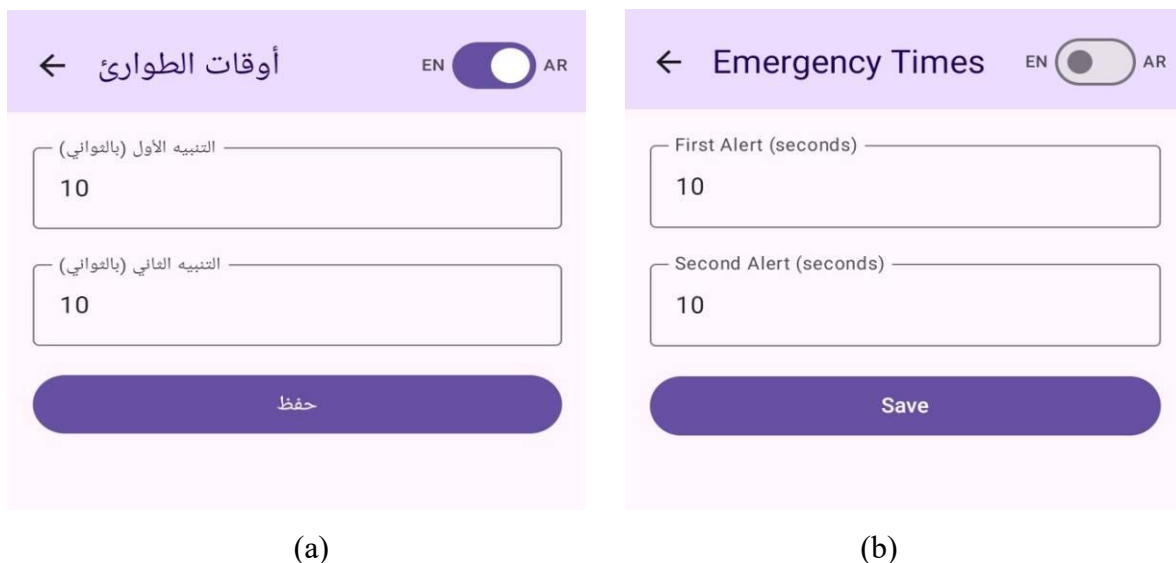


Figure IV. 16. Les champs de Emergency Times: a) en arabe et b) en anglais.

Ce mécanisme en deux étapes garantit une meilleure fiabilité du système : la première alerte assure une réaction rapide, tandis que la seconde renforce la sécurité en cas de non réponse, réduisant ainsi les risques liés aux situations critiques.

Informations personnelles (Personal Information) : La rubrique Informations personnelles (Personal Information) est destinée à identifier la personne responsable du suivi en cas de chute.

Elle permet d'enregistrer les coordonnées essentielles du proche ou du référent médical, telles que le nom, la date de naissance, le genre de personne (Male/Female), l'adresse Email, le numéro de téléphone, et aussi l'adresse d'habitation (Figure IV.17).

The screenshot shows a mobile application interface for 'Profile Settings'. At the top, there is a back arrow, the title 'Profile Settings', and a language toggle switch set to 'EN' (English). Below this is the section header 'Personal Information'. The form contains several input fields: 'Name' with the placeholder 'Please enter your full name.', 'Date of Birth' with a calendar icon and placeholder 'Please enter a valid date (MM/DD/YYYY) and ensure you are over 18.', 'Sex' with radio buttons for 'Male' and 'Female' (the 'Female' option is selected), 'Email' with the placeholder 'Please enter a valid email address.', 'Phone number' with the placeholder 'Please enter a 10-digit phone number.', and 'Address' with the placeholder 'Please enter your address.'. At the bottom of the form are two buttons: 'Save' (with a floppy disk icon) and 'Clear' (with an 'X' icon).

Figure IV. 17. Les informations personnelles de personne à détecter.

Ces données sont utilisées par le système pour transmettre automatiquement les alertes générées lors d'une détection de chute.

Ainsi, la personne désignée reçoit directement les notifications et peut intervenir rapidement, garantissant une prise en charge efficace et une sécurité renforcée pour l'utilisateur surveillé.

(a)
(b)

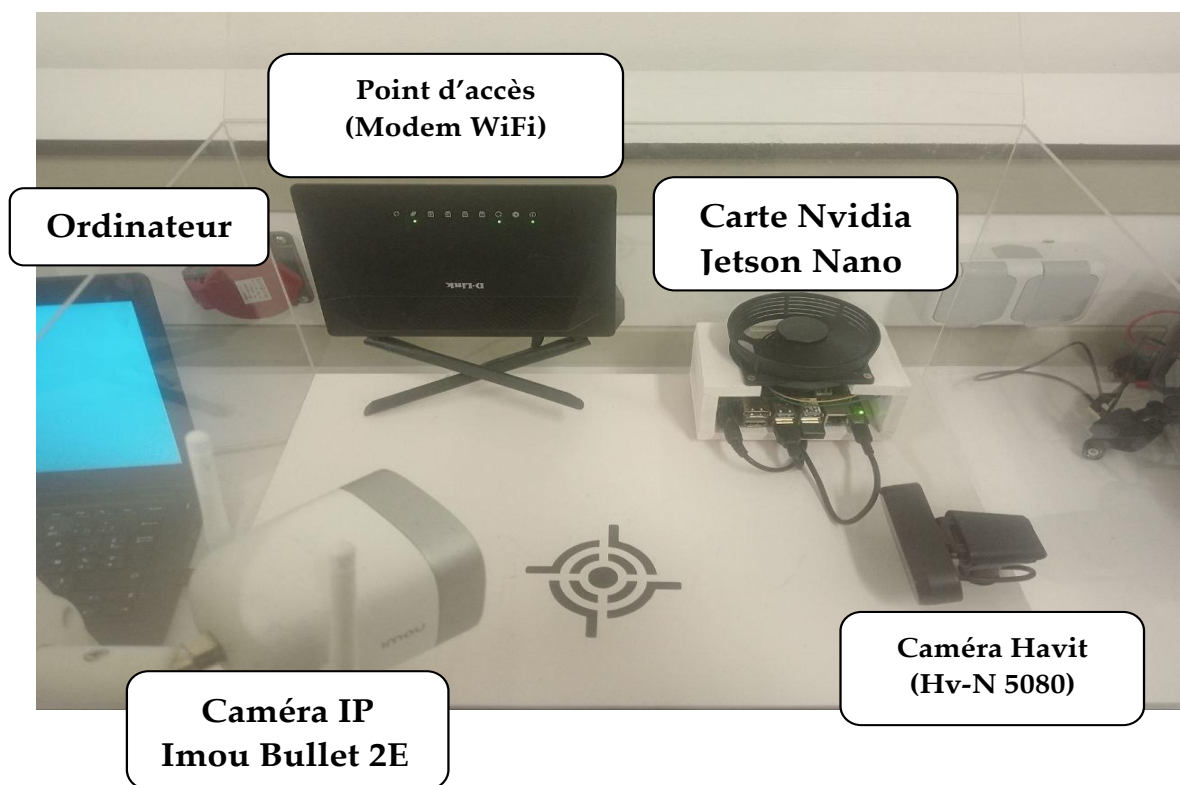
Figure IV. 18. Exemple de remplir les champs des informations personnelles de personne chuté : a) en arabe et b) en anglais.

IV.5. Validation expérimentale

Afin de valider expérimentalement la solution proposée, un prototype a été conçu et réalisé, comme illustré à la Figure IV.19. Ce prototype intègre l'ensemble des composants matériels et logiciels nécessaires à l'acquisition des données, au traitement des images et à la détection automatique des chutes. Il constitue une plateforme de test permettant d'évaluer les performances du système dans un environnement réel.

Dans le cadre de la mise en œuvre de notre système de détection de chutes, l'aspect matériel constitue une composante essentielle. Bien que plusieurs approches puissent être envisagées pour la reconnaissance des chutes, nous avons opté pour l'utilisation de la carte Jetson Nano, sur laquelle sont déployés les fichiers nécessaires à l'exécution du modèle YOLOv5.

L'entraînement et l'évaluation des différents modèles YOLOv5 ont été présentés en détail dans le chapitre III. Parmi les cinq variantes étudiées, nous avons retenu le modèle YOLOv5s en raison de son bon compromis entre précision de détection et rapidité d'exécution, répondant ainsi aux contraintes du traitement en temps réel.



(a)



(b)

Figure IV. 19. Prototype expérimental du système de détection de chutes proposé : (a) vue de dessus ; (b) vue de face.

Le fonctionnement du prototype repose sur une série d'étapes successives permettant son utilisation et son exploitation correcte.

Etape 1 : Alimentation des équipements

Avant toute configuration, il est essentiel de fournir l'alimentation électrique aux différents modules du système :

- Brancher la carte Jetson Nvidia Nano à son adaptateur secteur.
- Connecter le modem Wi-Fi (point d'accès) à une source d'alimentation stable.
- Alimenter la caméra IP via son câble d'alimentation.
- Vérifier que les voyants lumineux de chaque appareil s'allument correctement, indiquant une mise sous tension réussie.

Etape 2 : Connexion de la carte Jetson Nano

La carte Jetson Nano offre plusieurs modes de connexion, notamment via le réseau Wi-Fi, un câble Ethernet (RJ45) ou un câble USB Type-C. Dans notre expérimentation, la connexion entre Jetson Nano et l'ordinateur a été réalisée à l'aide d'un câble Type-C, facilitant ainsi l'accès à la carte et sa configuration. Cette étape permet de préparer l'environnement matériel et logiciel nécessaire au bon fonctionnement du système.

Etape 3 : Accès à distance

Ouvrir la connexion Bureau à distance et saisir de l'adresse IP : (192.168.55.1), puis validation de la Connexion

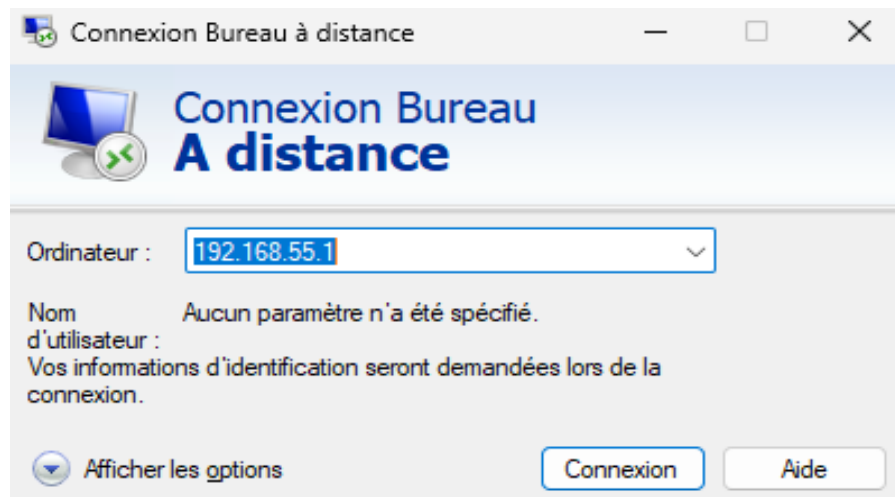
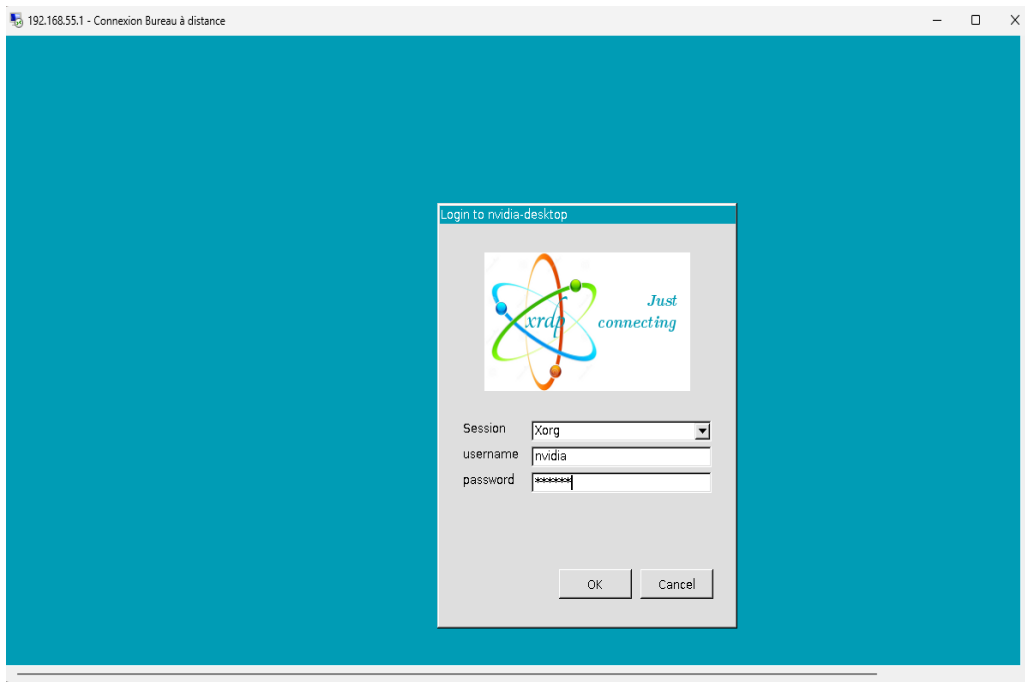


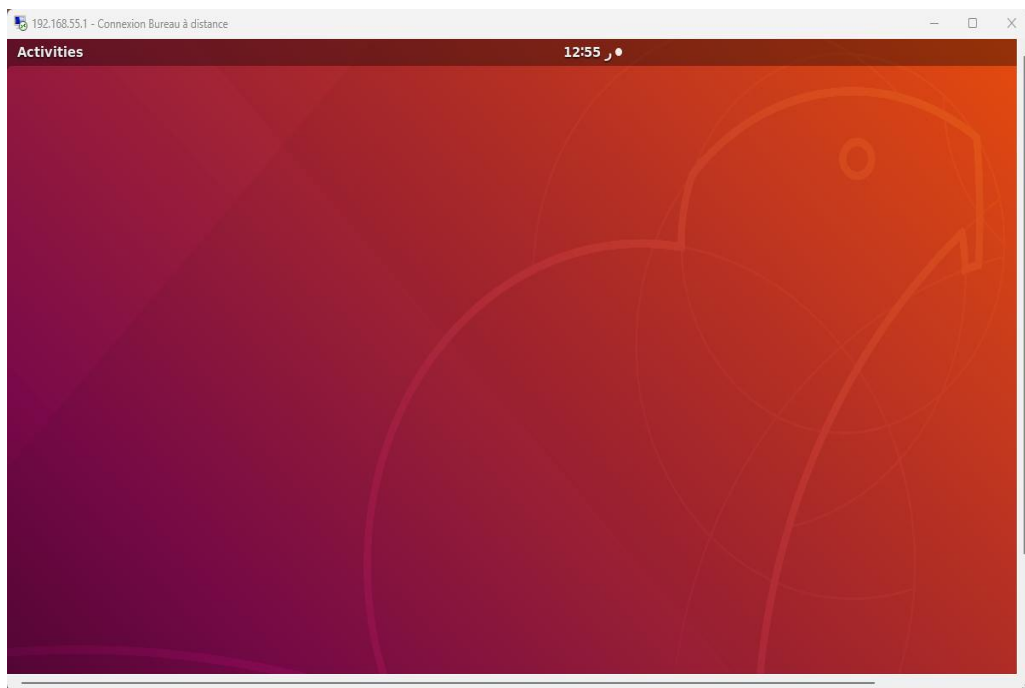
Figure IV. 20. Connexion Bureau à distance vers la carte Jetson Nano.

Etape 4 : Authentification

Saisie du nom d'utilisateur et du mot de passe associés à la carte Jetson Nano. Une fois validée, l'interface du système s'affiche, confirmant l'accès à la plateforme embarquée.



(a)



(b)

Figure IV. 21. (a) Interface d'authentification de la carte Jetson Nano, (b) fenêtre apparaît, indiquant l'accès à la carte Jetson Nvidia Nano.

Etape 5 : Accès au projet

Cliquer sur Activities, puis sur Files, et accéder au dossier Fall_Detection_System, qui contient le code Python du système de détection.

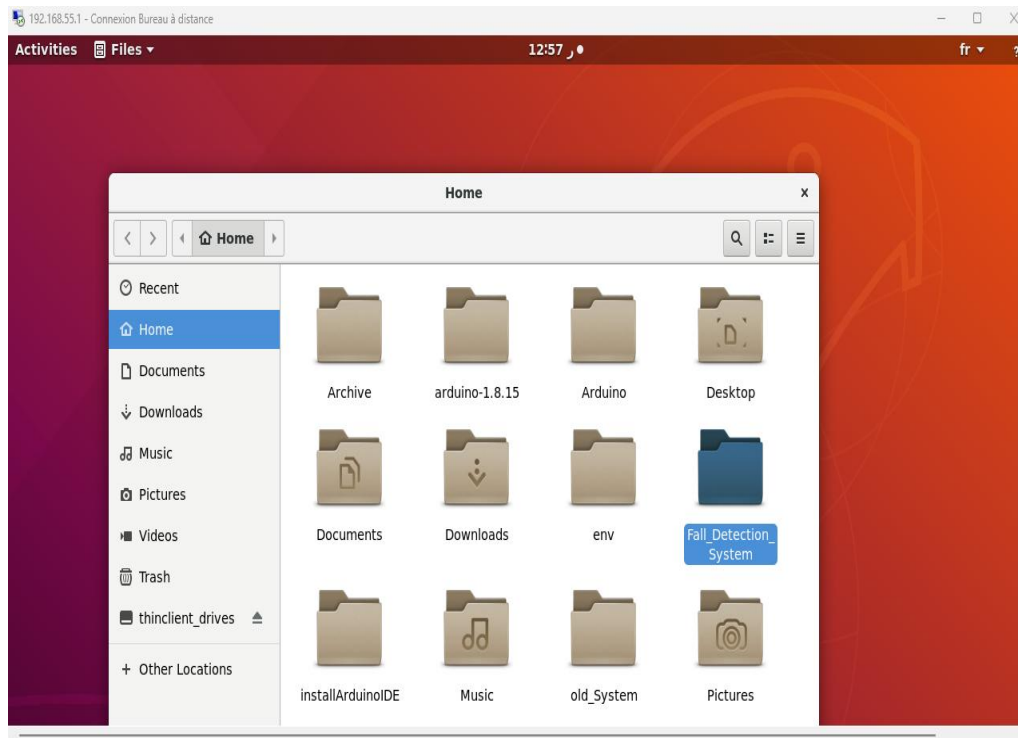


Figure IV. 22. Répertoire du projet Fall_Detection_System.

Etape 6 : Lancement du système

Dans le dossier du projet, exécuter le fichier *detect_server.py* pour configurer le type de caméra utilisé dans le processus de détection des chutes.

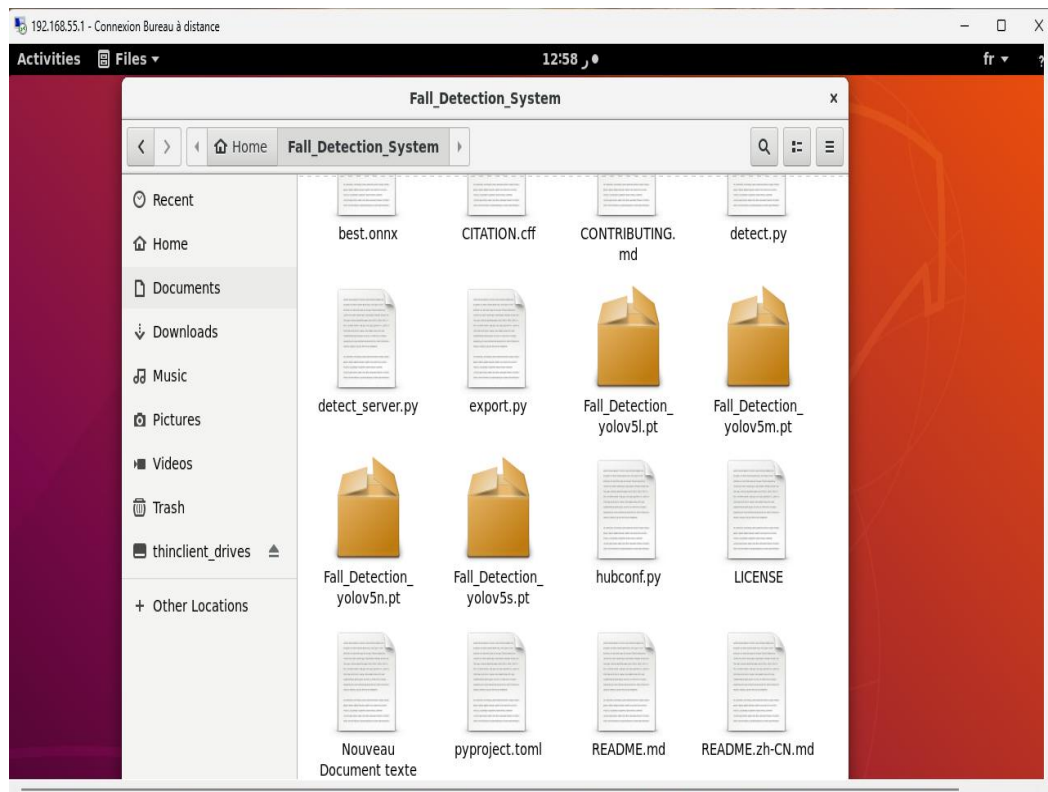
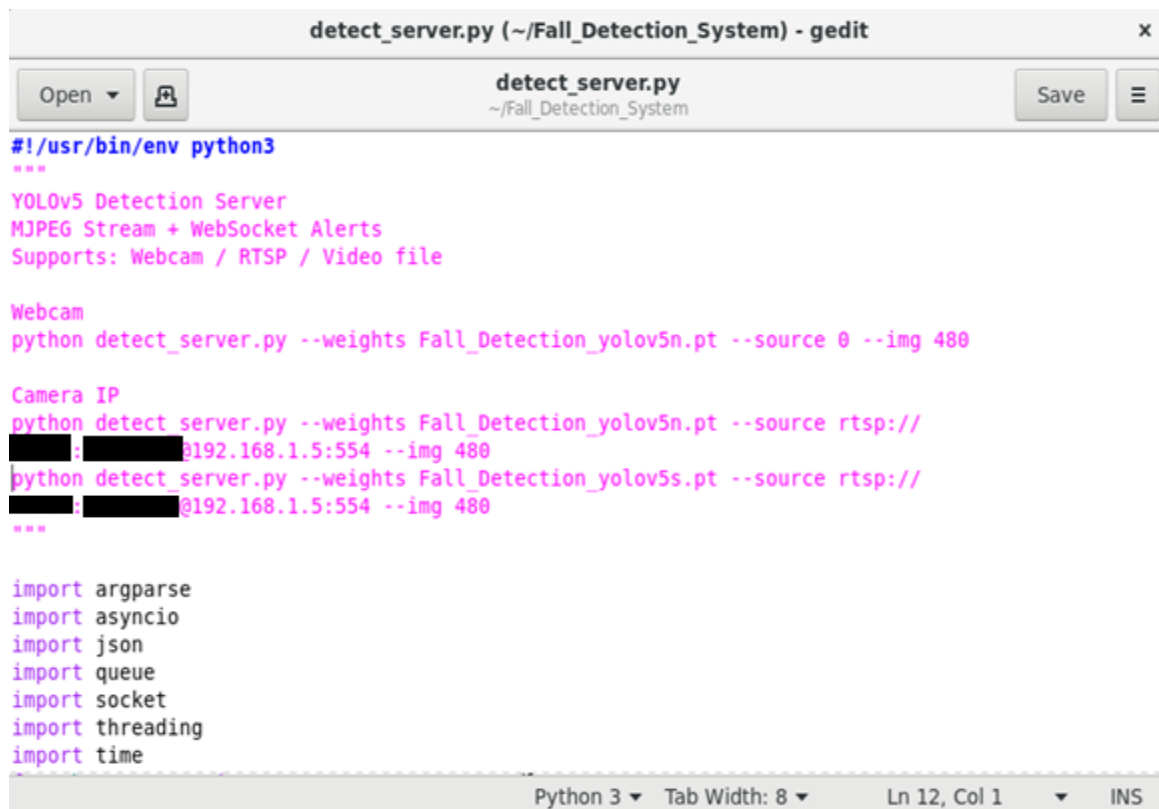


Figure IV. 23. Contenu de dossier Fall_Detection_System.

La fenêtre ci-dessous s'affiche, indiquant l'accès au système :



```

detect_server.py (~/.Fall_Detection_System) - gedit
Open Save
detect_server.py
~/.Fall_Detection_System

#!/usr/bin/env python3
"""
YOLOv5 Detection Server
MJPEG Stream + WebSocket Alerts
Supports: Webcam / RTSP / Video file

Webcam
python detect_server.py --weights Fall_Detection_yolov5n.pt --source 0 --img 480

Camera IP
python detect_server.py --weights Fall_Detection_yolov5n.pt --source rtsp://
[REDACTED]:[REDACTED]@192.168.1.5:554 --img 480
python detect_server.py --weights Fall_Detection_yolov5s.pt --source rtsp://
[REDACTED]:[REDACTED]@192.168.1.5:554 --img 480
"""

import argparse
import asyncio
import json
import queue
import socket
import threading
import time

```

Figure IV. 24. Code Python créé pour la détection de chute.

Dans le chapitre III, nous avons entraîné plusieurs modèles YOLOv5 à partir de la base de données LE2I, qui contient des séquences vidéo d'activités de chute et de non-chute, annotées image par image selon les classes Fall et non-Fall. Les expérimentations ont été réalisées sur la plateforme Google Colab, ce qui nous a permis d'évaluer différentes variantes du modèle YOLOv5 (YOLOv5n, YOLOv5s, YOLOv5m, YOLOv5l et YOLOv5x).

Après analyse des performances obtenues, et compte tenu des contraintes de temps de traitement et des ressources matérielles disponibles pour une exécution embarquée, le modèle YOLOv5s a été retenu comme le meilleur compromis entre précision et rapidité d'inférence.

Le modèle sélectionné a ensuite été transféré vers la carte Nvidia Jetson Nano à l'aide d'une clé USB afin de permettre son exploitation en temps réel. Lors de la phase de validation expérimentale, deux caméras de test ont été utilisées pour l'acquisition des vidéos. Dans ce contexte, deux configurations de fonctionnement sont possibles :

- **Caméra Webcam :**

```
python detect_server.py --weights Fall_Detection_yolov5s.pt --source-img 480
```

- **Camera IP :**

```
python      detect_server.py      --weights      Fall_Detection_yolov5s.pt      source
rtsp://****.****@192.168.1.5:554 img 480
```

Etape 7 : Choix de la caméra et exécution

Nous choisissons l'une des deux caméras, par exemple la caméra IP. Nous copions et collons le code correspondant à la caméra IP, puis nous appuyons sur la touche Entrée afin d'exécuter le code.



```
nvidia@nvidia-desktop: ~/Fall_Detection_System
File Edit View Search Terminal Help
bash: /home/nvidia/techarcanist/bin/activate: No such file or directory
nvidia@nvidia-desktop:~/Fall_Detection_System$ python detect_server.py --weights
Fall_Detection_yolov5n.pt --source rtsp://[redacted]:[redacted]@192.168.1.5:554 --img
480
detect_server: weights=Fall_Detection_yolov5n.pt, source=rtsp://[redacted]:[redacted]@1
92.168.1.5:554, imgsz=(480, 480), stream_port=8080, ws_port=8765
YOLOv5 v7.0-457-g84ef1e59 Python-3.8.0 torch-1.11.0a0+gitbc2c6ed CUDA:0 (NVID
IA Tegra X1, 3964MiB)

Using device: cuda:0
Fusing layers...
Model summary: 157 layers, 1761871 parameters, 0 gradients, 4.1 GFLOPs
1/1: rtsp://[redacted]:[redacted]@192.168.1.5:554... Success (inf frames 1280x720 at 2
5.00 FPS)

HTTP server started on port 8080
WebSocket server started on port 8765

=== SERVER STARTED ===
Stream: http://192.168.1.3:8080/stream
WebSocket: ws://192.168.1.3:8765
```

Figure IV. 25. Exécution du système avec caméra IP.

Le système génère ensuite les flux suivants :

- **Stream:** <http://192.168.1.3:8080/stream> (visualisation sur PC).
- **WebSocket:** ws://192.168.1.3:8765 (communication avec l'application Android AI Safety Monitor).

IV.6. Résultats

Le système proposé, Android AI Safety Monitor, est constitué d'une carte Nvidia Jetson Nano, de deux caméras de surveillance, d'un réseau Wi-Fi assurant la communication des données, ainsi que d'une application Android spécialement développée pour la réception et la gestion des alertes. L'ensemble du système a été testé dans un environnement intérieur reproduisant les conditions d'un domicile de personne âgée.

Le système exploite simultanément les flux vidéo provenant des deux caméras et utilise le modèle d'intelligence artificielle YOLOv5s pour détecter automatiquement les situations de chute en temps réel. Lorsqu'une chute est identifiée, une notification d'urgence est immédiatement transmise à l'application Android afin d'alerter les proches ou les aidants.

Les résultats expérimentaux obtenus démontrent l'efficacité de la solution proposée. En effet, le système est capable de distinguer avec précision les activités normales des situations de chute, tout en assurant une détection rapide et fiable. Ces performances confirment la pertinence de l'approche adoptée pour la surveillance et la sécurité des personnes âgées à domicile.

a) Détection d'une chute

Comme l'illustre la Figure IV.26, lors du scénario de test simulant une chute, le système a détecté la présence de la personne âgée allongée au sol avec un score de confiance de 0.73. Dès la détection de l'événement, une alerte visuelle de couleur rouge et une alerte sonore, accompagnée du message suivant:

“DANGER: FALL DETECTED!”

a été affichée sur l'interface de l'application Android afin d'informer immédiatement l'utilisateur de la situation critique.

Par la suite, un compte à rebours a été déclenché, permettant à l'utilisateur de confirmer son état en appuyant sur le bouton :

“I AM FINE”

En l'absence de confirmation avant l'expiration du délai imparti, le système déclenche automatiquement une procédure d'urgence consistant à envoyer un premier SMS d'alerte aux contacts préenregistrés. Cette action est confirmée par l'affichage du message :

“Emergency SMS sent automatically.”

Si aucune réaction ou intervention n'est constatée après un second délai prédéfini, un deuxième message est transmis afin de signaler une situation de très haute urgence nécessitant une prise en charge immédiate. Cette stratégie d'alerte progressive permet d'informer efficacement les proches ou les aidants de la persistance de l'état critique de la personne et de l'éventuelle nécessité d'une intervention rapide.

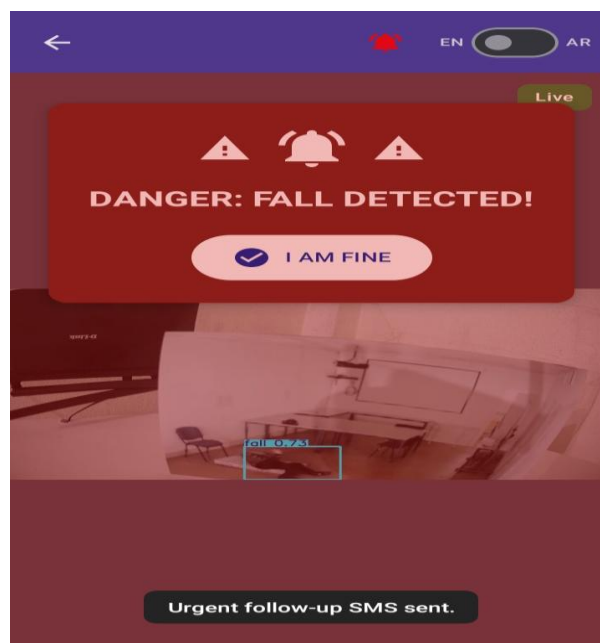


Figure IV. 26. Affichage de l'alerte de détection de chute d'une personne âgée à domicile sur l'application AI Safety Monitor.

Les résultats obtenus démontrent que le système est capable d'identifier de manière fiable les situations de chute et d'activer rapidement les mécanismes d'assistance appropriés. Cette réactivité contribue à réduire le temps d'intervention en cas d'accident domestique et renforce ainsi la sécurité des personnes âgées vivant seules à domicile.

b) Détection d'une activité normale

Comme l'illustre la Figure IV.27, dans le second scénario expérimental, la personne se déplace normalement à l'intérieur de la pièce. Le modèle YOLOv5s a correctement identifié cette situation comme une activité normale en la classant dans la catégorie :

“non-Fall”

avec un score de confiance de 0,83.

Aucune alerte ni notification d'urgence n'a été générée par le système, ce qui confirme sa capacité à distinguer efficacement les activités quotidiennes des situations de chute. Ce résultat met en évidence la robustesse du modèle et sa capacité à limiter les fausses alertes, un aspect essentiel pour garantir la fiabilité et l'acceptabilité du système dans un environnement domestique réel.

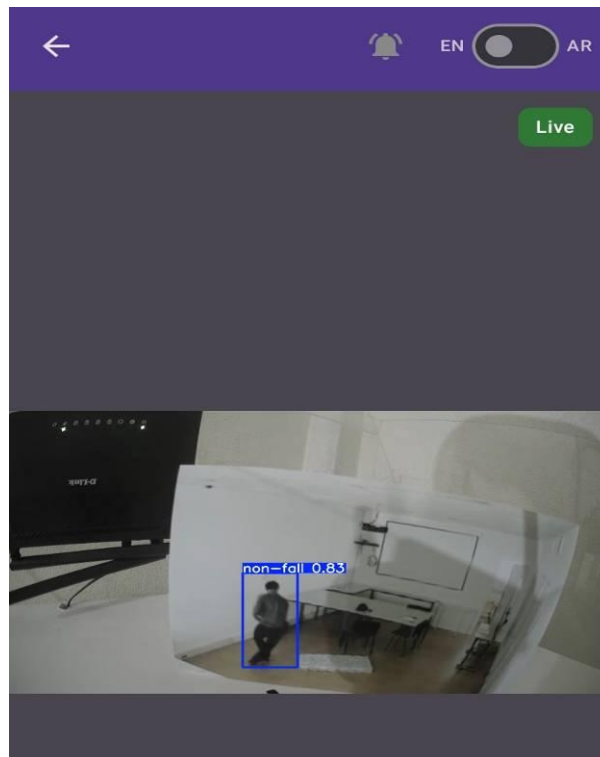


Figure IV. 27. Détection d'une activité normale (non-Fall) d'une personne âgée à domicile par l'application AI Safety Monitor.

Les expérimentations réalisées montrent que l'application développée présente plusieurs avantages importants pour la surveillance des personnes âgées à domicile.

Premièrement, l'intégration de l'intelligence artificielle permet une détection automatique des chutes en temps réel, sans intervention humaine. Cela améliore considérablement la sécurité des personnes vivant seules.

Deuxièmement, l'application offre une réaction rapide grâce à l'envoi automatique des SMS d'urgence. Cette fonctionnalité peut contribuer à réduire les conséquences graves liées aux chutes, notamment chez les personnes âgées.

De plus, la présence du bouton de confirmation **"I AM FINE"** permet de limiter les fausses alertes lorsque la personne n'est pas réellement en danger.

Cependant, certains défis ont également été observés durant les tests :

- ✓ La précision du modèle peut être influencée par la qualité de l'éclairage ;
- ✓ Certains mouvements complexes peuvent parfois être mal interprétés ;
- ✓ Les performances dépendent fortement de la position de la caméra ;
- ✓ Une connexion stable est nécessaire pour certaines fonctionnalités de communication.

Malgré ces limitations, les résultats obtenus restent satisfaisants et démontrent la faisabilité d'un système intelligent de télésurveillance basé sur Android et l'intelligence artificielle.

En conclusion, l'application AI Safety Monitor a montré une bonne capacité à détecter les chutes et à distinguer les activités normales. Le système représente une solution innovante pour le maintien à domicile des personnes âgées, en offrant sécurité, rapidité d'intervention et assistance automatisée.

IV.7. Conclusion

Ce chapitre a présenté de manière détaillée le processus de conception et de réalisation du système de détection de chute proposé. L'architecture globale, fondée sur une chaîne de traitement structurée allant de l'acquisition des données à la génération d'alertes, a permis de mettre en évidence la cohérence et la complémentarité des différentes étapes. L'intégration sur la plateforme Jetson Nano, optimisée par l'accélération GPU et la gestion du temps réel, a confirmé la pertinence du choix technologique pour répondre aux contraintes de performance et de fiabilité.

Par ailleurs, la mise en place d'un serveur de diffusion et le développement d'une application Android ont assuré une interaction efficace entre le système embarqué et l'utilisateur final, en offrant une interface ergonomique et une gestion réactive des notifications. Les validations expérimentales réalisées en conditions réelles ont démontré la robustesse et l'efficacité du dispositif, attestant de sa capacité à répondre aux besoins de surveillance et d'assistance des personnes vulnérables.

Ainsi, la conception et la réalisation du système proposé constituent une étape essentielle dans la concrétisation de la solution envisagée. Les résultats obtenus ouvrent la voie à des perspectives d'amélioration, notamment l'optimisation des algorithmes de détection, l'enrichissement des fonctionnalités de communication et l'adaptation du système à des environnements plus complexes.

Conclusion générale

Conclusion générale

Dans ce mémoire, nous avons abordé la problématique de la télésurveillance médicale appliquée aux personnes âgées, en mettant l'accent sur la détection des chutes, considérée comme un enjeu majeur de santé publique. Le travail réalisé a porté sur la conception et le développement d'un système embarqué intelligent reposant sur la carte Nvidia Jetson Nano et une application Android dédiée. L'objectif principal était de proposer une solution capable d'assurer une détection fiable des chutes en temps réel, tout en garantissant la sécurité, la confidentialité et l'intégrité des données personnelles.

Pour atteindre cet objectif, une démarche méthodologique progressive a été adoptée. Celle-ci a débuté par une étude approfondie de l'état de l'art dans le domaine de la télésurveillance médicale et des techniques de détection des chutes. Elle s'est poursuivie par la sélection des technologies matérielles et logicielles les plus adaptées. Le développement du système s'est appuyé sur le langage Python pour l'implémentation des algorithmes embarqués et l'entraînement des modèles YOLO, tandis que l'application mobile a été développée en Kotlin sous Android Studio. Afin d'accélérer les phases d'apprentissage profond nécessitant d'importantes ressources de calcul, les infrastructures de Google Cloud ont été utilisées, notamment les GPU et TPU dédiés à l'entraînement des réseaux de neurones.

Les résultats obtenus se sont révélés particulièrement encourageants. Sur le plan technique, l'implémentation du modèle YOLOv5 sur la plateforme Jetson Nano a démontré la faisabilité d'un traitement embarqué performant, capable de détecter les chutes avec une précision élevée et une faible latence. L'exploitation des capacités d'accélération GPU ainsi que les optimisations réalisées pour le traitement en temps réel ont contribué à améliorer l'efficacité et la robustesse du système. Sur le plan applicatif, le développement d'une application Android intuitive a permis de rendre le système accessible aux familles et au personnel médical, en assurant la réception des alertes et la visualisation des flux vidéo. Enfin, sur le plan scientifique, ce travail contribue à la recherche sur la télésurveillance intelligente, en proposant une solution hybride combinant l'intelligence artificielle embarquée (Edge AI) et les technologies mobiles, répondant ainsi aux exigences croissantes en matière de confidentialité, de réactivité et d'autonomie.

Malgré ces résultats encourageants, certaines limites doivent être soulignées. La base de données Le2i utilisée pour l'entraînement et l'évaluation du modèle reste relativement limitée en termes de diversité des scénarios, des environnements et des profils d'utilisateur, ce qui peut affecter la capacité de généralisation du système.

Par ailleurs, bien que la consommation énergétique de la Jetson Nano ait été optimisée, elle demeure un facteur contraignant pour les applications nécessitant une autonomie prolongée. De plus, la transmission des alertes repose sur une connexion Wi-Fi, dont la qualité peut varier selon les conditions de couverture réseau. Enfin, la validation expérimentale, bien qu'encourageante, a été réalisée sur un nombre restreint de cas, ce qui nécessite des tests plus larges pour confirmer la robustesse du système dans des conditions réelles variées.

Les perspectives de développement de ce travail sont nombreuses. Sur le plan technique, l'intégration de modèles d'intelligence artificielle plus avancés, tels que les architectures basées sur les Transformers, ou les modèles multimodaux, pourrait améliorer davantage les performances de détection et réduire les faux positifs. L'utilisation d'un cloud hybride combiné à l'Edge Computing permettrait de bénéficier de la puissance du cloud tout en réduisant la latence et en préservant la confidentialité des données. Sur le plan applicatif, l'évolution vers une plateforme multi-utilisateurs offrirait une solution adaptée aux structures médicales et aux maisons de retraite. L'intégration de la 5G améliorerait considérablement la rapidité et la fiabilité des communications. Le renforcement de la sécurité des données et l'optimisation énergétique restent des axes prioritaires pour garantir une adoption à grande échelle.

Une autre perspective intéressante consiste à intégrer la dimension temporelle dans le processus de détection. Dans le cadre de ce travail, la détection des chutes repose principalement sur l'analyse d'images individuelles, sans prise en compte explicite de l'évolution temporelle des mouvements. Ainsi, de futures améliorations pourraient s'appuyer sur une architecture hybride combinant un modèle YOLO pour l'extraction des caractéristiques spatiales et un réseau de neurones récurrent, tel qu'un LSTM (Long Short-Term Memory) ou un Bi-LSTM (Bidirectional Long Short-Term Memory), pour l'analyse temporelle des séquences vidéo. Cette approche permettrait de mieux modéliser l'évolution des mouvements humains, d'améliorer la distinction entre une chute réelle et des activités quotidiennes similaires, et de réduire le taux de fausses alertes.

Par ailleurs, l'application développée dans le cadre de ce travail pourrait être étendue à d'autres domaines de surveillance intelligente, notamment à l'étude et à l'analyse du comportement humain. En intégrant des algorithmes d'estimation de pose (Pose Estimation), il serait possible d'extraire et de suivre en temps réel les points clés du corps humain afin d'analyser les postures, les gestes et les mouvements des individus.

En conclusion, ce mémoire a permis de démontrer la faisabilité d'un système intelligent de télésurveillance basé sur la carte Nvidia Jetson Nano et une application Android, capable de détecter les chutes en temps réel avec des performances satisfaisantes. Les résultats obtenus confirment l'intérêt de l'approche proposée et mettent en évidence le potentiel des technologies d'intelligence artificielle embarquée dans l'amélioration de la sécurité et de la qualité de vie des personnes âgées. Les perspectives identifiées ouvrent la voie au développement d'une solution plus complète, plus robuste et pleinement exploitable dans un contexte réel de télésurveillance médicale.