

Projet de Fin d'Etudes

Pour l'obtention de la Licence Sciences et Techniques
Informatique, Réseaux et Multimédia

DÉTECTION ET RECONNAISSANCE AUTOMATIQUE DES MONUMENTS TOURISTIQUES

Présenté par :
Hamid Oufakir
Reda Lamtoueh

Soutenu le 30/06/2022 devant le jury composé de :

Sara Sekkate	Encadrante	ENSAM
Siham Akil	Co-encadrante	FSTM
Abdellah Adib	Rapporteur	FSTM
Mohammed Khalil	Examinateur	FSTM

Année Universitaire : 2021-2022

Dédicace

Nous dédions ce modeste travail

À nos très chers parents qui ont toujours été là pour nous et qui nous ont donné un magnifique modèle de courage, de labeur et persévérandce.

À nos frères et nos sœurs.

À tous les membres de notre famille.

En ce moment, nous ne pouvons pas oublier l'ensemble des amis que nous avons connu pendant nos études, en témoignage de l'amitié qui nous unit et des souvenirs de tous les moments que nous avons passés ensemble, je vous dédie ce travail et je vous souhaite une vie pleine de santé et de bonheur.

Remerciements

En premier lieu, on remercie le bon Dieu, tout puissant, de nous avoir donné la force pour survivre et dépasser toutes les difficultés. La réalisation de ce mémoire fût possible grâce à la contribution de plusieurs personnes à qui nous témoignons tout notre gratitude.

Nous avons le plaisir d'exprimer toute la gratitude à M.EL BOUZIRI ADIL, chef du département et M.MOUMKINE NOURDDINE , chef de la filière, ainsi que tout le corps enseignant de la filière Informatique Réseaux Multimédia (IRM) pour la qualité de l'enseignement qu'ils nous ont assuré et pour les efforts qu' ils ont déployés pour le bon déroulement des projets.

Nous tenons à exprimer notre profonde gratitude ainsi que toute notre reconnaissance à notre encadrantes Mme.SEKKATE SARA et Mme.AKIL SIHAM , pour leurs conseils inestimables et pour leur entière disponibilité tout au long de la période du projet de fin d'études, un très spécial remerciement pour votre patience, votre modestie et vos qualités professionnelles et personnelles.

Mes sincères remerciements vont également à l'ensemble des membres de jury : Mme.SEKKATE SARA, M.ADIB ABDELLAH et M.KHALIL MOHAMMED de bien avoir accepté d'examiner et évaluer ce travail.

Résumé

Ce document est élaboré à l'issue de notre travail dans le cadre de notre projet de fin d'études en licence science et technique en « Informatique, Réseau et Multimédia » à la Faculté des Sciences et Techniques de Mohammedia.

Le but de ce projet est le développement d'une application mobile qui permet aux touristes d'avoir les informations sur les monuments touristiques au Maroc en temps-réel, en prenant des photos ou en récupérant des images.

Ce rapport incarne le travail fourni dans le projet, et nous introduit aux outils et technologies employés.

Abstract

This document was accomplished as a final year project for a program degree in "Computer Science, Network and Multimedia" at the Faculty of Science and Technology of Mohammedia.

The main goal of this project is the development of a mobile application that allows tourists to get informations about Moroccan monuments using real time detection, by taking photos or retrieving images from the internet.

This report embodies the work done in the project, and introduces the tools and technologies employed.

Keywords : Deep Learning ; Object Detection ; Monument Recognition ; YoloV5 ; Android ;

Table des matières

Dédicace	1
Remerciements	2
Résumé	2
Abstract	3
Liste des figures	7
Liste des tables	8
Introduction générale	9
1 Contexte Général du projet	10
1.1 Organisme d'accueil	10
1.1.1 Présentation de la FSTM	10
1.1.2 Formation	10
1.1.3 Missions et valeurs	11
1.2 Présentation du sujet	11
2 Analyse et spécification des besoins	12
2.1 Cahier des charges	12
2.1.1 Besoins fonctionnels	12
2.1.2 Besoins non fonctionnels	12
2.2 Conduite du projet	13
3 La reconnaissance d'objets dans les images	15
3.1 Classification, localisation et détection d'objets	15
3.2 Algorithmes de détection d'objets avant l'apprentissage profond	16
3.3 Algorithmes de détection d'objets après l'apprentissage profond	18
3.3.1 Famille de modèles R-CNN	18
3.3.2 Famille de modèles YOLO	19
4 Étude conceptuelle	23
4.1 Diagramme des cas d'utilisation	23
4.2 Diagramme de séquence	25
5 Réalisation	26
5.1 Outils Techniques	26
5.2 Préparation de la base de données	27
5.2.1 Collecte des données	27

5.2.2	Labélisation	31
5.2.3	Augmentation des données	32
5.3	Choix et construction du modèle	33
5.3.1	Avantages et inconvénients de Yolov5	34
5.3.2	Construction du modèle	34
5.4	Réalisation de l'application mobile	39
5.4.1	L'intégration du Modèle dans l'application	40
5.4.2	Interfaces graphiques de L'application	40
5.5	Conclusion et perspectives	45
	Références	46

Liste des figures

2.1	Diagramme de GANTT.	13
2.2	Planification des tâches effectuées lors du PFE.	14
3.1	Détection d'objet.[1]	16
3.2	Caractéristiques de type HAAR[6].	16
3.3	Détecteur HOG[6].	17
3.4	Modèle à base de pièces déformables (DPM)[6].	17
3.5	Les différentes versions de YOLO.[2]	20
3.6	Boite englobante.[3]	21
3.7	Intersection sur Union. [4]	21
4.1	Diagramme des cas d'utilisation.	23
4.2	Diagramme de séquence.	25
5.1	Exemple d'annotation d'image avec Roboflow.	31
5.2	Exemple d'un label.	31
5.3	Répartition des données.	32
5.4	Augmentation des données	33
5.5	Architecture de YOLOV5.[13]	33
5.6	Code utilise pour GPU, installer les dépendances et le référence de yolov5	34
5.7	Le ficheie custom_data.yaml	35
5.8	Code de l'entraînement	35
5.9	Code de test	35
5.10	Résultat de test	36
5.11	Diagramme représentant des valeurs de précision.	36
5.12	Diagramme représentant des valeurs de rappel	37
5.13	Diagramme représentant la précision et le rappel.	37
5.14	Matrice de confusion	38
5.15	Interface d'accueil de l'application.	41
5.16	Interface d'à propos de nous.	41
5.17	Bouton pour choisir la langue.	42
5.18	Liste des langues.	42
5.19	Bouton pour prendre les photos.	42
5.20	Bouton pour choisir les images.	43
5.21	Bouton pour lancer la détection.	43
5.22	Interface des résultats.	44
5.23	La fenêtre de google.	44

Liste des tableaux

4.1 Description Textuelle.	24
5.1 Le nombre d'images fournies pour chacun des sites.	31
5.2 Performances obtenues du modèle.	39

Introduction générale

Il est admis partout dans le monde, que le tourisme a été parmi les secteurs les plus touchés par la crise sanitaire de 2020. L'interruption soudaine des circulations nationales et internationales s'est traduite par un arrêt de l'activité. Or, le manque à gagner est très important, surtout pour des pays comme le Maroc où l'activité a un poids considérable dans l'économie et la société.

Sachant que nous vivons dans une société où les nouvelles technologies et les systèmes informatisés ont rendu facile la production et la diffusion d'informations, il devient primordial de développer des outils informatiques qui permettent d'encourager les touristes à choisir de visiter notre pays comme destination pour relancer le secteur du tourisme.

“Une image vaut 1000 mots“ disait le philosophe Confucius. Aujourd’hui comme auparavant, l'image est un langage universel. Pour cela, nous allons utiliser les images pour identifier les sites touristiques au Maroc et aussi pour donner envie de choisir notre pays comme destination à visiter.

Dans le cadre de notre projet de fin d'études, l'objectif est de réaliser une application mobile qui permet aux touristes d'avoir toutes les informations nécessaires sur les sites touristiques et les monuments historiques du Maroc en prenant juste une photo par leurs téléphones mobiles, la détection se fait automatiquement en se basant sur des algorithmes puissants d'Intelligence Artificielle.

Dans ce rapport, nous présenterons le travail effectué sous 5 chapitres. Le premier chapitre est consacré au contexte général du projet, ainsi que l'organisme d'accueil. Le second chapitre est consacré à établir le cahier des charges et la planification du projet. Le troisième chapitre est dédié à l'étude théorique. Le quatrième chapitre est dédié à la conception de l'application et sa modélisation avec un diagramme des cas d'utilisation, un diagramme de classes et des diagrammes de séquences. Enfin, le dernier chapitre est consacré à la réalisation du projet. Enfin, nous dressons une conclusion et des perspectives.

Chapitre 1

Contexte Général du projet

Dans ce chapitre, nous lèverons le voile sur le contexte général du projet. Il comprend, une présentation de l'organisme d'accueil, une description de la problématique pour laquelle le projet a été établi, ainsi que l'objectif et la solution adoptée qui a permis la réalisation de ce projet.

1.1 Organisme d'accueil

1.1.1 Présentation de la FSTM

La Faculté des Sciences et Techniques de Mohammedia (FSTM) est un établissement public de l'enseignement supérieur à accès régulé qui est implanté au nord de la ville de Mohammedia. Elle constitue une composante importante de l'université Hassan II de Casablanca et fait partie d'un réseau de huit FST à travers le Maroc dont la vocation est la formation universitaire dans les sciences et techniques. La FSTM ; implantée dans une zone à forte activité industrielle ; a toujours veillé à s'intégrer dans son environnement en disposant des cursus aboutissant à des profils qui répondent par excellence aux besoins socio-économiques. Elle œuvre depuis son inauguration, en 1994, à offrir une formation technique et en ingénierie de qualité, et s'est distinguée par la pertinence de sa recherche. La FSTM compte son effectif 186 enseignants chercheurs, 57 personnels administratifs et techniques et plus de 3000 étudiants dont 750 en dernière année de licence 260 en master et 440 en filières d'ingénieurs. Elle délivre plus de 550 diplômes par an. C'est un établissement à accès régulé où l'entrée se fait après une sélection basée sur les notes au baccalauréat des matières définies en fonction du parcours choisi (Mathématiques, Informatique, Physique (MIP) ou Biologie, Chimie, Géologie (BCG)). La FSTM compte 8 départements et 13 laboratoires [15].

1.1.2 Formation

La FST de Mohammedia offre des cursus de formation dans les domaines des « Sciences et Techniques » et des « Sciences de l'Ingénieur » préparant aux grades universitaires de Licence (Bac + 3), Master (Bac + 5), Doctorat (Bac + 8) en Sciences et Techniques, Ingénieur d'Etat et de Technicien (DUT).

- DEUST : Diplôme d'Etudes Universitaire en Sciences et Techniques.
- LST : Diplôme de Licence en Sciences et Techniques.
- MST : Diplôme de Master en Sciences et Techniques.
- Dip. Ing : Diplôme d'Ingénieur d'Etat.
- DUT : Diplômes Universitaire de Technologie.
- Doctorat en Sciences et Techniques[15].

1.1.3 Missions et valeurs

La FSTM regroupe des structures qui s'investissent pleinement dans les domaines de la formation et de la recherche scientifique pour contribuer au développement du Maroc. Ces structures sont engagées pour :

- Assurer des formations scientifiques et techniques de qualité en adéquation avec le marché de l'emploi.
- Développer la recherche scientifique fondamentale et appliquée pour contribuer à la production du savoir, répondre au besoin des entreprises et encourager l'innovation.
- Dispenser des formations continues pour répondre aux besoins du secteur socio-économique.
- Participer au progrès scientifique, technique, professionnel et économique du pays [15].

La FSTM véhicule des valeurs basées sur l'intégrité intellectuelle et morale et s'engage à :

- Développer le professionnalisme, le sens de la responsabilité, la créativité et le respect d'autrui.
- Respecter les principes d'équité, d'égalité des chances et de la pluralité culturelle.
- Émuler l'excellence, la créativité et l'innovation[15].

1.2 Présentation du sujet

De nos jours, l'image est devenue le premier outil d'identification et de représentation des sites touristiques, il arrive dans certains cas de voir une photo d'un lieu que l'on aimera visiter mais nous ne pouvons pas déterminer où il se trouve. D'un autre point de vue, il se trouve que beaucoup de touristes aiment avoir plus d'informations sur certains sites, considérant qu'ils n'ont aucune idée sur ces sites ou comment faire une recherche textuelle de ces sites sur le web pour obtenir plus de détails.

La capacité de reconnaître des sites touristiques à partir d'images peut être extrêmement utile lors du choix d'une destination de voyage et lors de la tentative d'identification de sites touristiques dans un lieu étranger.

Face à cette problématique et pour renforcer la compétitivité du marché du tourisme, présenter une belle image sur le tourisme au Maroc et contribuer au développement du secteur touristique, ce projet propose le développement d'une application mobile d'identification automatique des sites et des monuments touristiques.

Conclusion

Au cours de ce chapitre, nous avons présenté le contexte général du projet, ainsi que la problématique et enfin l'objectif et la solution proposée. Le chapitre suivant est consacré à l'étude préliminaire.

Chapitre 2

Analyse et spécification des besoins

Dans ce chapitre, nous commençons par établir le cahier des charges qui comporte les besoins fonctionnels et non fonctionnels ainsi que la planification du projet.

2.1 Cahier des charges

Le Maroc compte parmi les pays touristiques par excellence qui est caractérisé par : son climat, sa culture spécifique, ses villes impériales, ses monuments et sa position géographique. Mais ce qui compte vraiment pour nous dans ce projet est de faciliter l'accès aux informations exactes et précises sur les sites et les monuments touristiques.

Notre mission consiste à développer une application mobile qui permet de détecter les sites et les monuments touristique au Maroc et donner les informations nécessaires sur chaque site ou monument détecté.

2.1.1 Besoins fonctionnels

Les besoins fonctionnels expriment un ensemble d'actions que doit effectuer le système en répondant à une demande (sorties qui sont produites pour un ensemble donné d'entrées). Pour cela, nous devons définir les services souhaités. Dans ce qui suit, nous décrivons les différents besoins fonctionnels de notre système :

- **Charger les données** : consiste à charger l'image à partir de la caméra du téléphone mobile ou à partir d'une image importée.
- **Régler l'image** : consiste à régler l'image soit en la redimensionnant ou en lui appliquant une rotation.
- **Lancer l'analyse** : consiste à lancer la détection et la reconnaissance des monuments dans les images.
- **Consulter les résultats de l'analyse** : consiste à consulter les résultats fournis par l'application ou les résultats existant sur internet.

2.1.2 Besoins non fonctionnels

Les besoins non fonctionnels sont des besoins qui ont un aspect visible pour l'utilisateur et ils caractérisent le système. Ce sont des besoins en matière de performance qui exige la conformité aux standards, la complétude et la cohérence, mais ne concernent pas le comportement du système. Dans notre cas, ces besoins s'illustrent comme suit :

- **Disponibilité** : Notre application doit être disponible et opérationnelle à tout moment.
- **Performance** : Il s'agit d'optimiser le temps d'analyse des images prisent par l'utilisateur et déterminer les sites et les monuments par l'utilisation des bonnes pratiques du développement.

- **Utilisation** : L'application doit être intuitive et facile à utiliser . Les résultats affichés ne doivent pas être complexes ou compliqués à comprendre aux utilisateurs.
- **Fiabilité** : Les informations sont très importantes, il est primordial et essentiel de les avoir justes et dignes de confiance.
- **Maintenabilité** : L'application doit pouvoir être facile à mettre à jour et à maintenir.

2.2 Conduite du projet

La planification du projet est une étape importante, qui doit être effectuée avant de commencer la réalisation du projet. La planification consiste à ordonner les tâches du projet en utilisant le diagramme de GANTT. La FIGURE 2.1 représente le diagramme de GANTT qui offre la possibilité de pouvoir visualiser la durée des différentes tâches et leurs enchaînements mais aussi l'avancement de notre projet de manière claire et précise.



FIGURE 2.1 – Diagramme de GANTT.

Afin d'éclairer la chronologie de la réalisation du projet, nous exposerons dans la FIGURE 2.2 l'ensemble des tâches effectuées durant ce stage de fin d'études.

Nom de la tâche	Durée	Début	Fin
« Détection et reconnaissance des sites et monuments touristiques	41 jours	Mar 26/04/22	Jeu 23/06/22
Rédaction du rapport	41 jours	Mar 26/04/22	Jeu 23/06/22
« Définition et analyse de projet	8 jours	Mar 26/04/22	Lun 09/05/22
Étude de la problématique	2 jours	Mar 26/04/22	Mer 27/04/22
Définir le cahier de charge	2 jours	Jeu 28/04/22	Ven 29/04/22
Étude sur les différents algorithmes de détection d'objet	4 jours	Mer 04/05/22	Lun 09/05/22
« Analyse et conception	2 jours	Mar 10/05/22	Mer 11/05/22
Diagramme des cas d'utilisation	1 jour	Mar 10/05/22	Mar 10/05/22
Description textuelle	1 jour	Mar 10/05/22	Mar 10/05/22
Diagramme de séquence	1 jour	Mer 11/05/22	Mer 11/05/22
« Réalisation	31 jours	Jeu 12/05/22	Jeu 23/06/22
Collection de la base de données	7 jours	Jeu 12/05/22	Ven 20/05/22
Préparation de la base de données	2 jours	Lun 23/05/22	Mar 24/05/22
Choix du modèle de deep Learning	1 jour	Mer 25/05/22	Mer 25/05/22
Entraîner la base de données par le modèle choisi	1 jour	Jeu 26/05/22	Jeu 26/05/22
Tester les performances du modèle	1 jour	Jeu 26/05/22	Jeu 26/05/22
Étude sur l'environnement pour développer des applications mobiles Android	6 jours	Ven 27/05/22	Ven 03/06/22
Développer une application Android	12 jours	Lun 06/06/22	Mar 21/06/22
Intégration du modèle dans l'application	2 jours	Mer 22/06/22	Jeu 23/06/22
Maintenance et Test de fonctionnement de l'application	2 jours	Mer 22/06/22	Jeu 23/06/22

FIGURE 2.2 – Planification des tâches effectuées lors du PFE.

Description des phases

- Phase 1 : Rédaction du rapport

Cette phase du projet a duré tout au long du projet, elle comporte la rédaction du rapport qui résume toutes les étapes du projet.

- Phase 2 : Définition et analyse du projet

Elle consiste à étudier la problématique du projet et définir le cahier des charges.

- Phase 3 : Analyse et conception

Elle comporte l'analyse du cahier des charges, l'identification des acteurs ainsi que l'élaboration des différents diagrammes.

- Phase 4 : Réalisation

Elle comporte toutes les étapes qui entrent dans le développement de l'application.

Conclusion

Au cours de ce chapitre, nous avons présenté le cahier des charges et la planification suivie pour la réalisation de notre projet de fin d'études. Le chapitre suivant est consacré à l'étude théorique.

Chapitre 3

La reconnaissance d'objets dans les images

La reconnaissance d'objets est considérée comme un problème général de la vision par ordinateur. Cette dernière est le domaine de l'informatique qui se concentre sur la reproduction de parties de la complexité du système de vision humaine et permet aux ordinateurs d'identifier et de traiter des objets dans des images et des vidéos de la même manière que les humains [10]. Jusqu'à récemment, la vision par ordinateur ne fonctionnait que de manière limitée. Grâce aux progrès de l'intelligence artificielle et aux innovations en matière d'apprentissage profond et de réseaux de neurones, le domaine a pu faire de grands bonds ces dernières années et a pu surpasser les humains dans certaines tâches liées à la détection et à l'étiquetage d'objets. En plus d'une énorme quantité de données visuelles, la puissance de calcul nécessaire à l'analyse des données est désormais accessible. Au fur et à mesure que le domaine de la vision par ordinateur s'est développé avec de nouveaux matériels et algorithmes, les taux de précision pour l'identification d'objets ont également augmenté. En moins d'une décennie, les systèmes actuels ont atteint une précision de 99% contre 50% [9], ce qui les rend plus précis que les humains pour réagir rapidement aux entrées visuelles.

3.1 Classification, localisation et détection d'objets

La reconnaissance d'objets est un terme général pour décrire un ensemble de tâches de vision par ordinateur connexes qui impliquent l'identification d'objets dans des photographies numériques.

La classification des images, d'un autre côté, consiste à prédire la classe d'un objet dans une image. La localisation d'objets fait référence à l'identification de l'emplacement d'un ou plusieurs objets dans une image et au dessin d'une boîte abondante autour de leur étendue. La détection d'objets combine ces deux tâches et localise et classe un ou plusieurs objets dans une image.

Lorsque le terme de « reconnaissance d'objet » est généralement employé, il entend souvent « détection d'objet ».

Ainsi, nous pouvons distinguer ces trois tâches de vision par ordinateur :

- **Classification d'image** : consiste à prédire le type ou la classe d'un objet dans une image.
 - Entrée : Une image avec un seul objet, comme une photographie.
 - Sortie : Une étiquette de classe (par exemple un ou plusieurs entiers qui sont mappés à des étiquettes de classe).
- **Localisation d'objets** : Localise la présence d'objets dans une image et indique leur emplacement avec une boîte englobante.
 - Entrée : Une image avec un ou plusieurs objets, comme une photographie.

- Sortie : Une ou plusieurs boîtes englobantes (par exemple définies par un point, une largeur et une hauteur).
- **Détection d'objets** : Localise la présence d'objets avec une boîte englobante et les types ou classes des objets localisés dans une image.
 - Entrée : Une image avec un ou plusieurs objets, comme une photographie.
 - Sortie : Une ou plusieurs boîtes englobantes (par exemple définies par un point, une largeur et une hauteur) et une étiquette de classe pour chaque boîte englobante

Une autre extension de cette répartition des tâches de vision par ordinateur est la segmentation d'objets, également appelée «segmentation d'instance d'objet» ou «segmentation sémantique», où les instances d'objets reconnus sont indiquées en mettant en évidence les pixels spécifiques de l'objet au lieu d'un cadre de délimitation grossier [7].

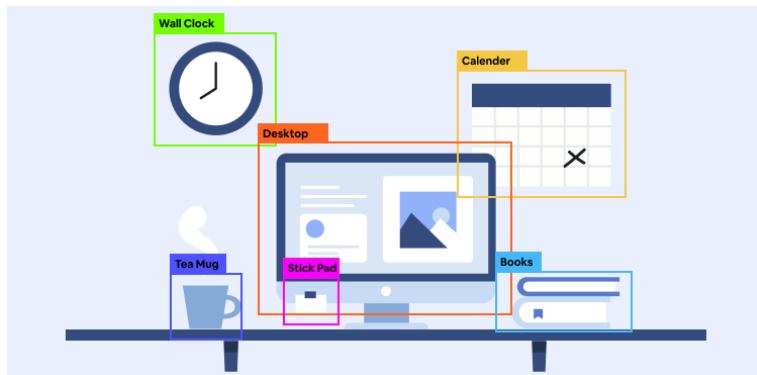


FIGURE 3.1 – Détection d'objet.[1]

3.2 Algorithmes de détection d'objets avant l'apprentissage profond

Dans cette section, nous passerons en revue l'historique de la détection d'objets de la "période de détection d'objets traditionnelle (avant 2014)".

Détecteurs Viola Jones :

Développé en 2001 par Paul Viola et Michael Jones, ce cadre de reconnaissance d'objets permet la détection de visages humains en temps réel. Il utilise des fenêtres coulissantes pour parcourir tous les emplacements et échelles possibles dans une image pour voir si une fenêtre contient un visage humain. Les fenêtres coulissantes recherchent essentiellement des caractéristiques "de type haar" (du nom d'Alfred Haar qui a développé le concept des ondelettes de haar)[6].

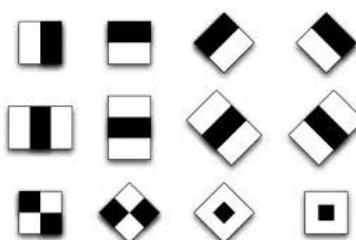


FIGURE 3.2 – Caractéristiques de type HAAR[6].

Ainsi, l'ondelette haar est utilisée comme représentation caractéristique d'une image. Pour accélérer la détection, elle utilise une image intégrale, ce qui rend la complexité de calcul de

chaque fenêtre glissante indépendante de sa taille de fenêtre. auteurs est d'utiliser l'algorithme Adaboost pour la sélection de fonctionnalités qui sélectionne un petit ensemble de fonctionnalités qui sont principalement utiles pour la détection de visage à partir d'un vaste ensemble de pools de fonctionnalités aléatoires. L'algorithme a également utilisé Détection Cascades qui est un paradigme de détection à plusieurs étapes pour réduire son calcul surcharge en dépensant moins de calculs sur les fenêtres d'arrière-plan mais plus sur les cibles de visage[6].

Détecteur HOG :

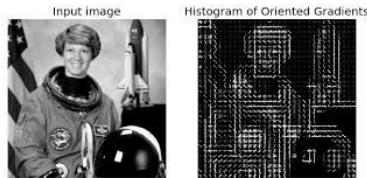


FIGURE 3.3 – Détecteur HOG[6].

Proposé à l'origine en 2005 par N. Dalal et B. Triggs, Hog est une amélioration des contextes de transformation et de forme des caractéristiques invariantes à l'échelle de son époque. HOG fonctionne avec quelque chose appelé blocs (similaire à une fenêtre coulissante), une grille de pixels dense dans laquelle les gradients sont constitués à partir de l'amplitude et de la direction du changement des intensités des pixels à l'intérieur du bloc. Les HOG sont largement connus pour leur utilisation dans la détection des piétons. Pour détecter des objets de différentes tailles, le détecteur HOG redimensionne l'image d'entrée plusieurs fois tout en gardant la taille d'une fenêtre de détection inchangée[6].

Modèle à base de pièces déformables (DPM) :

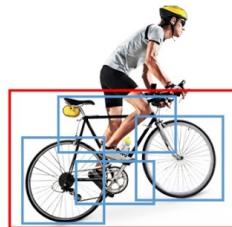


FIGURE 3.4 – Modèle à base de pièces déformables (DPM)[6].

DPM a été proposé à l'origine par P. Felzenszwalb en 2008 comme une extension du détecteur HOG. Plus tard, une variété d'améliorations ont été apportées par R. Girshick. Le problème de la détection d'une « voiture » peut être décomposé en une stratégie « diviser pour mieux régner » en détectant sa fenêtre, sa carrosserie et ses roues. DPM utilise cette stratégie. Le processus de formation implique l'apprentissage d'une manière appropriée de décomposer un objet, et l'inférence implique l'assemblage de détections de différentes parties d'objet.

Le détecteur DPM se compose d'un filtre racine et d'un certain nombre de filtres partiels. Une méthode d'apprentissage faiblement supervisé est développée dans DPM où toutes les configurations (taille, emplacement, etc.) des filtres partiels peuvent être automatiquement apprises en tant que variables latentes. Pour améliorer la précision de la détection, R. Girshick a utilisé un cas particulier d'apprentissage multi-instance à cette fin, et d'autres techniques importantes telles que "l'extraction négative dure", la "régression par boîte englobante" et "l'amorçage contextuel". Plus tard, il a même utilisé une technique qui implémente une architecture en cascade, qui a atteint une accélération 10 fois supérieure sans sacrifier la précision[6].

3.3 Algorithmes de détection d'objets après l'apprentissage profond

après la naissance de l'apprentissage en profondeur, deux familles d'algorithmes étaient dominants dans la détection d'objets, la famille R-CNN (algorithmes à deux niveaux) et la famille YOLO (algorithmes à un niveau).

3.3.1 Famille de modèles R-CNN

Cette famille de méthodes fait référence au R-CNN (Region-based Convolutional Neural Network), qui peut signifier « Régions avec des caractéristiques CNN » ou « Réseau de neurones convolutionnels basés sur les régions », développé par Ross Girshick et al [7].

Cela inclut les techniques R-CNN, Fast R-CNN et Faster-RCNN conçues et démontrées pour la localisation et la reconnaissance d'objets.

R-CNN

R-CNN est l'une des premières applications importantes et réussies des réseaux de neurones convolutifs au problème de la localisation, de la détection et de la segmentation d'objets. R-CNN est composé de trois modules ; elles sont :

- **Module 1 : Proposition de Région.** Génère et extrait des propositions de régions indépendantes des catégories, par exemple des boîtes englobantes candidates.
- **Module 2 : Extracteur de fonctionnalités,** Extrait une caractéristique de chaque région candidate, par exemple en utilisant un réseau de neurones à convolution profonde.
- **Module 3 : Classificateur,** Classe les caractéristiques comme faisant partie de la classe connue.

Une technique de vision par ordinateur est utilisée pour proposer des régions candidates ou des boîtes englobantes d'objets potentiels dans l'image appelée « recherche sélective », bien que la flexibilité de la conception permette d'utiliser d'autres algorithmes de proposition de région.

L'extracteur de caractéristiques utilisé par le modèle était le CNN profond AlexNet qui a remporté le concours de classification d'images ILSVRC-2012. La sortie du CNN était un vecteur de 4 096 éléments qui décrit le contenu de l'image qui est transmis à un SVM linéaire pour la classification, en particulier un SVM est formé pour chaque classe connue.

Il s'agit d'une application relativement simple et directe des CNN au problème de la localisation et de la reconnaissance d'objets. Un inconvénient de l'approche est qu'elle est lente, nécessitant une passe d'extraction de caractéristiques basée sur CNN sur chacune des régions candidates générées par l'algorithme de proposition de région. C'est un problème car le document décrit le modèle fonctionnant sur environ 2 000 régions proposées par image au moment du test [7].

Fast R-CNN

Compte tenu du grand succès de R-CNN, Ross Girshick, alors chez Microsoft Research, a proposé une extension pour résoudre les problèmes de vitesse de R-CNN dans un article de 2015 intitulé « Fast R-CNN » [7].

L'article s'ouvre sur un examen des limites de R-CNN, qui peuvent être résumées comme suit :

- **La formation est un pipeline en plusieurs étapes :** Implique la préparation et le fonctionnement de trois modèles distincts.
- **La formation est coûteuse en espace et en temps :** Former un CNN profond sur autant de propositions de régions par image est très lent.

- **La détection d'objet est lente :** Faire des prédictions en utilisant un CNN profond sur autant de propositions de régions est très lent.

Fast R-CNN est proposé comme un modèle unique au lieu d'un pipeline pour apprendre et produire directement des régions et des classifications.

L'architecture du modèle prend la photographie comme entrée d'un ensemble de propositions de régions qui sont transmises à travers un réseau de neurones à convolution profonde. Un CNN pré-entraîné, tel qu'un VGG-16, est utilisé pour l'extraction de caractéristiques. La fin du CNN profond est une couche personnalisée appelée couche de regroupement de régions d'intérêt, ou regroupement RoI, qui extrait des fonctionnalités spécifiques à une région candidate d'entrée donnée.

La sortie du CNN est ensuite interprétée par une couche entièrement connectée, puis le modèle se divise en deux sorties, une pour la prédiction de classe via une couche softmax, et une autre avec une sortie linéaire pour la boîte englobante. Ce processus est ensuite répété plusieurs fois pour chaque région d'intérêt dans une image donnée.

Le modèle est beaucoup plus rapide à former et à faire des prédictions, mais nécessite toujours qu'un ensemble de régions candidates soit proposé avec chaque image d'entrée[7].

Faster R-CNN

L'architecture du modèle a été encore améliorée pour la vitesse de formation et de détection par Shaoqing Ren, et al. L'architecture a servi de base aux résultats de première place obtenus dans les tâches de reconnaissance et de détection d'objets ILSVRC-2015 et MS COCO-2015. L'architecture a été conçue à la fois pour proposer et affiner les propositions de région dans le cadre du processus de formation, appelé Réseau de proposition de région, ou RPN. Ces régions sont ensuite utilisées de concert avec un modèle Fast R-CNN dans une conception de modèle unique. Ces améliorations réduisent à la fois le nombre de propositions de régions et accélèrent le fonctionnement en temps de test du modèle en temps quasi réel avec des performances de pointe. Bien qu'il s'agisse d'un modèle unifié unique, l'architecture est composée de deux modules :

- **Module 1 : Réseau de proposition de région,** Réseau de neurones convolutifs pour proposer des régions et le type d'objet à considérer dans la région.
- **Module 2 : R-CNN rapide,** Réseau neuronal convolutif pour extraire les caractéristiques des régions proposées et produire la boîte englobante et les étiquettes de classe.

Les deux modules fonctionnent sur la même sortie d'un CNN profond. Le réseau de proposition de région agit comme un mécanisme d'attention pour le réseau Fast R-CNN, informant le deuxième réseau de l'endroit où regarder ou prêter attention. Le RPN fonctionne en prenant la sortie d'un CNN profond pré-entraîné, tel que VGG-16, et en passant un petit réseau sur la carte des caractéristiques et en produisant plusieurs propositions de régions et une prédiction de classe pour chacune. Les propositions de régions sont des boîtes englobantes, basées sur des boîtes dites d'ancrage ou des formes prédefinies conçues pour accélérer et améliorer la proposition de régions. La prédiction de classe est binaire, indiquant la présence d'un objet, ou non, ce que l'on appelle « l'objectivité » de la région proposée [7].

Une procédure d'apprentissage alterné est utilisée où les deux sous-réseaux sont entraînés en même temps, bien qu'entrelacés. Cela permet aux paramètres du CNN profond du détecteur de caractéristiques d'être adaptés ou affinés pour les deux tâches en même temps.

3.3.2 Famille de modèles YOLO

YOLO est l'abréviation du terme "You Only Look Once". Il s'agit d'un algorithme qui détecte et reconnaît divers objets dans une image (en temps réel). La détection d'objets dans YOLO se fait comme un problème de régression et fournit les probabilités de classe des images

déetectées. L'algorithme YOLO utilise des réseaux de neurones convolutionnels (CNN) pour détecter des objets en temps réel. Comme son nom l'indique, l'algorithme ne nécessite qu'une seule propagation vers l'avant à travers un réseau de neurones pour détecter des objets. Cela signifie que la prédiction dans l'image entière est effectuée en une seule exécution d'algorithme. Le CNN est utilisé pour prédire simultanément diverses probabilités de classe et boîtes englobantes. L'algorithme YOLO se compose de différentes variantes. Certains des plus courants incluent les minuscules YOLO, YOLOv3, YOLOv4 et YOLOv5 [7].



FIGURE 3.5 – Les différentes versions de YOLO.[2]

YOLO est l'un des meilleurs algorithmes de détection d'objets pour les raisons suivantes :

- **Vitesse** : cet algorithme améliore la vitesse de détection car il peut prédire les objets en temps réel.
- **Haute précision** YOLO est une technique prédictive qui fournit des résultats précis avec un minimum d'erreurs de fond.
- **Capacités d'apprentissage** : l'algorithme possède d'excellentes capacités d'apprentissage qui lui permettent d'apprendre les représentations d'objets et de les appliquer à la détection d'objets.

L'algorithme YOLO fonctionne en utilisant les trois techniques suivantes :

1. Blocs résiduels

Tout d'abord, l'image est divisée en différentes grilles. Chaque grille a une dimension de $S \times S$. L'image suivante montre comment une image d'entrée est divisée en grilles. Dans l'image ci-dessus, il existe de nombreuses cellules de grille de dimension égale. Chaque cellule de la grille détectera les objets qui y apparaissent. Par exemple, si un centre d'objet apparaît dans une certaine cellule de la grille, cette cellule sera responsable de sa détection.

2. Régression de boîte englobante

Une boîte englobante est un contour qui met en évidence un objet dans une image. Chaque cadre de délimitation de l'image se compose des attributs suivants :

- Largeur
- Hauteur
- Classe (par exemple, personne, voiture, feu de circulation, etc.) - Ceci est représenté par la lettre c .
- Centre de la boîte englobante (bx, by).

L'image suivante montre un exemple de zone de délimitation. La boîte englobante a été représentée par un contour jaune.

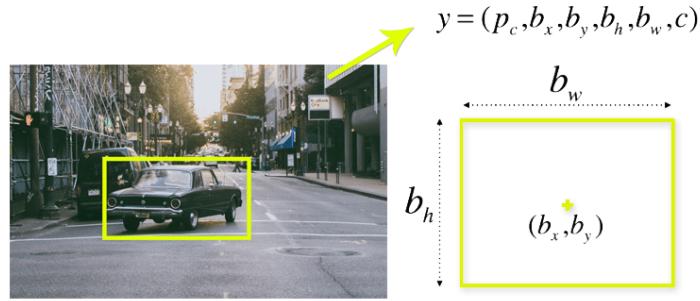


FIGURE 3.6 – Boite englobante.[3]

YOLO utilise une seule régression de boîte englobante pour prédire la hauteur, la largeur, le centre et la classe des objets. Dans l'image ci-dessus, représente la probabilité qu'un objet apparaisse dans la boîte englobante.

3. Intersection sur Union (IOU)

L'intersection sur l'union (IOU) est un phénomène de détection d'objet qui décrit comment les boîtes se chevauchent. YOLO utilise IOU pour fournir une boîte de sortie qui entoure parfaitement les objets.

Chaque cellule de la grille est chargée de prédire les boîtes englobantes et leurs scores de confiance. L'IOU est égal à 1 si la boîte englobante prédite est la même que la boîte réelle. Ce mécanisme élimine les boîtes englobantes qui ne sont pas égales à la boîte réelle[7].

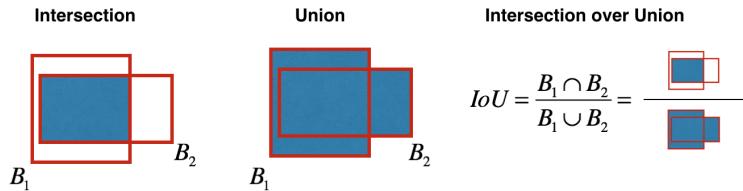


FIGURE 3.7 – Intersection sur Union. [4]

Single Shot Detector (SSD)

SSD a deux composants : un modèle de backbone et une tête SSD. Le modèle de dorsale est généralement un réseau de classification d'images pré-entraîné en tant qu'extracteur de caractéristiques. Il s'agit généralement d'un réseau comme ResNet formé sur ImageNet à partir duquel la couche de classification finale entièrement connectée a été supprimée[20]. Nous nous retrouvons donc avec un réseau de neurones profonds capable d'extraire le sens sémantique de l'image d'entrée tout en préservant la structure spatiale de l'image mais à une résolution inférieure. Pour ResNet34, le backbone donne 256 cartes de caractéristiques 7x7 pour une image d'entrée. Nous expliquerons plus tard quelles sont les fonctionnalités et la carte des fonctionnalités. La tête SSD juste une ou plusieurs couches convolutives ajoutées à ce squelette et les sorties sont interprétées comme les boîtes englobantes et les classes d'objets dans l'emplacement spatial des activations finales des couches[11].

Au lieu d'utiliser une fenêtre glissante, SSD divise l'image à l'aide d'une grille et chaque cellule de la grille est responsable de la détection des objets dans cette région de l'image. La détection d'objets signifie simplement prédire la classe et l'emplacement d'un objet dans cette région. Si aucun objet n'est présent, il est considéré comme la classe d'arrière-plan et l'emplacement est ignoré[11].

Chaque cellule de grille dans SSD peut être affectée à plusieurs cases d'ancrage/prior. Ces boîtes d'ancrage sont prédéfinies et chacune est responsable d'une taille et d'une forme dans une cellule de la grille[11].

SSD utilise une phase de correspondance lors de la formation, pour faire correspondre la boîte d'ancrage appropriée avec les boîtes englobantes de chaque objet de vérité terrain dans une image. Essentiellement, la boîte d'ancrage avec le plus haut degré de chevauchement avec un objet est responsable de la prédiction de la classe de cet objet et de son emplacement. Cette propriété est utilisée pour former le réseau et pour prédire les objets détectés et leurs emplacements une fois que le réseau a été formé. En pratique, chaque case d'ancrage est spécifiée par un format d'image et un niveau de zoom[14].

Tous les objets ne sont pas de forme carrée et peuvent différer en termes de longueur et de largeur, à des degrés divers. L'architecture SSD permet des rapports d'aspect prédéfinis des boîtes d'ancrage pour en tenir compte. Le paramètre ratios peut être utilisé pour spécifier les différents rapports d'aspect des boîtes d'ancrage associées à chaque cellule de la grille à chaque niveau de zoom/échelle.

Conclusion

Nous avons présenté dans ce chapitre la vision par ordinateur ainsi que Les familles d'algorithmes utilisées pour la détection d'objets. Le chapitre suivant est consacré à l'analyse et la conception de notre système, en se basant sur les différents diagrammes UML.

Chapitre 4

Étude conceptuelle

Cette section sera présentée comme suit : nous commençons par l'identification du diagramme des cas d'utilisation, puis nous présentons la description textuelle et son diagramme de séquence, enfin le diagramme de classe.

4.1 Diagramme des cas d'utilisation

Chaque usage que les acteurs font du système est représenté par un cas d'utilisation. Chaque cas d'utilisation représente une fonctionnalité qui leur est offerte afin de produire le résultat attendu. Ainsi, le diagramme des cas d'utilisation décrit l'interaction entre le système et l'acteur en déterminant les besoins de l'utilisateur et tout ce que doit faire le système pour l'acteur.

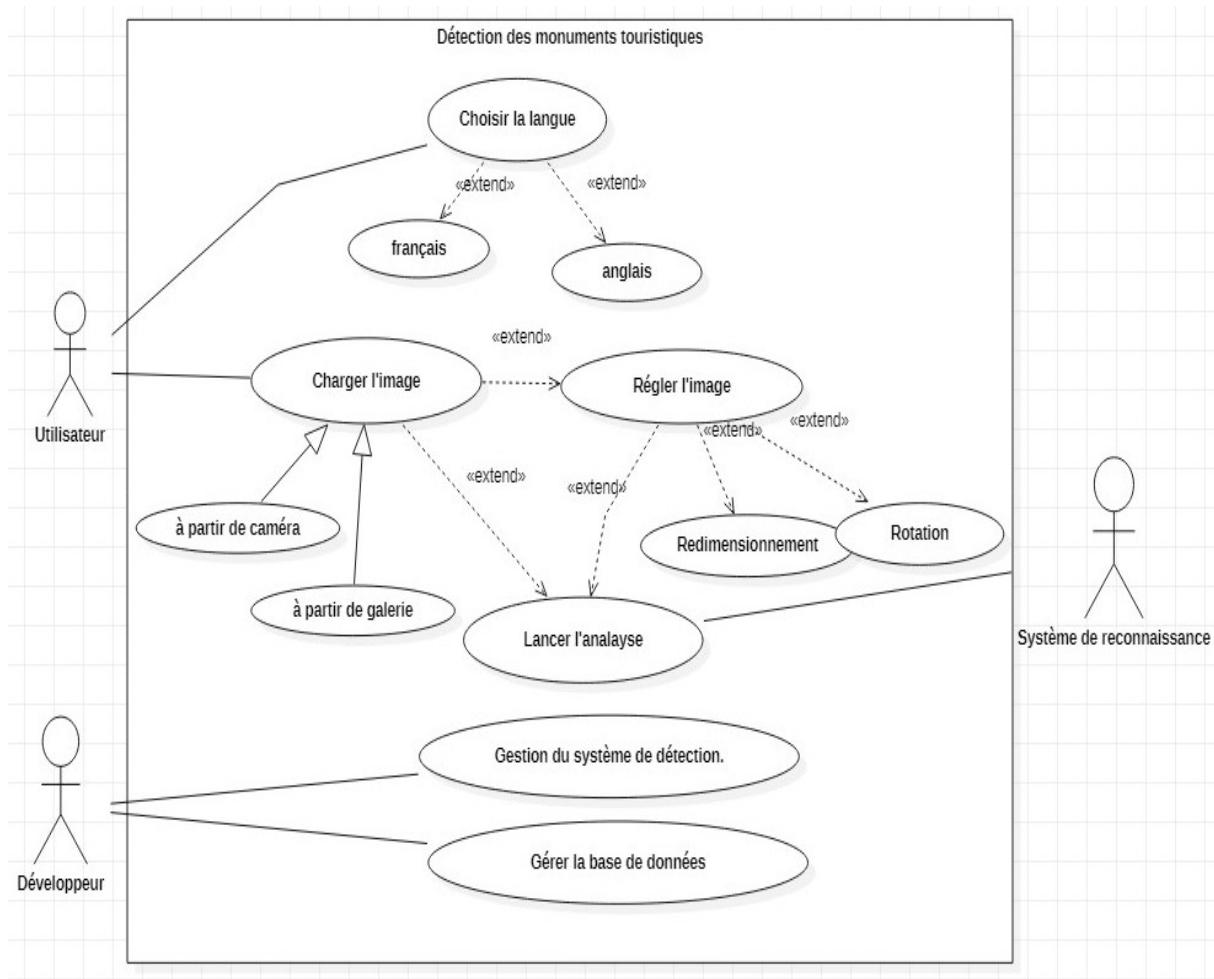


FIGURE 4.1 – Diagramme des cas d'utilisation.

Description textuelle

La description textuelle permet de clarifier le déroulement des actions du cas d'utilisation ainsi que de décrire la chronologie des actions qui devront être réalisées. De plus, cette action aide à identifier les parties redondantes pour en déduire des cas d'utilisation plus précis qui seront utilisés par inclusion et d'indiquer les contraintes dont les développeurs vont devoir tenir compte lors de la réalisation du système.

Ce tableau décrit les étapes de détection que le touriste doit suivre :

NOM	Détection des monuments touristiques.
RÉSUMÉ	Permet l'utilisateur de détecter les monuments touristiques automatiquement.
ACTEURS	Utilisateurs
PRÉ-CONDITIONS	Le système de téléphone est opérationnel.
SCÉNARIO NOMINAL	<ol style="list-style-type: none"> 1. L'utilisateur charge l'image. 2. L'utilisateur règle l'image choisie 3. Le système affiche l'image choisie. 4. L'utilisateur lance l'analyse. 5. Le système lance la procédure de détection. 6. Le détecteur commence la détection. 7. Le détecteur retourne les résultats de détection. 8. Le système affiche les résultats.
SCÉNARIOS ALTERNATIFS	<ul style="list-style-type: none"> • L'utilisateur demande plus d'informations sur le monument détecté. • Le système effectue une recherche sur internet. • Le système affiche les résultats trouvés.
SCÉNARIOS D'ERREURS	Aucun
POST-CONDITIONS	Aucun

TABLE 4.1 – Description Textuelle.

4.2 Diagramme de séquence

Le diagramme de la FIGURE 4.2 présente les différentes interactions qui doivent être effectuées pour la reconnaissance d'un site touristique.

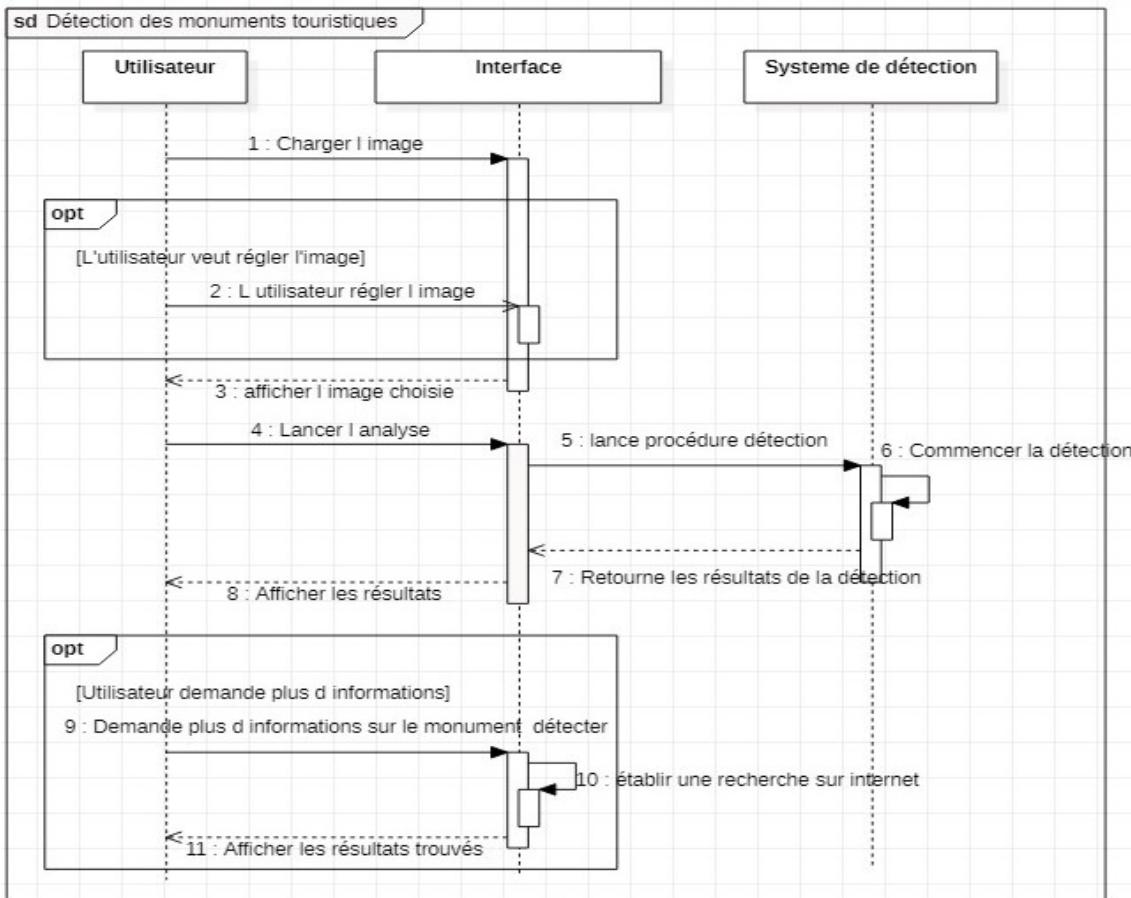


FIGURE 4.2 – Diagramme de séquence.

Conclusion

Au cours de ce chapitre, nous avons présenté les différentes étapes de conception qui décrivent les fonctionnalités de notre solution comme le diagramme de cas d'utilisation, le diagramme de séquence et le diagramme de classes. Le chapitre suivant est consacré à la réalisation de l'application.

Chapitre 5

Réalisation

Dans cette section, nous analyserons la procédure et le raisonnement derrière l'expérience menée, ainsi que la sélection de la technique spécifique. Nous visons à créer une application mobile de détection d'objets utilisant le modèle YOLOv5 qui nous permet de détecter 19 monuments marocains célèbres en temps réel.

5.1 Outils Techniques

Langages de programmation utilisés :



UML (en anglais Unified Modeling Language ou langage de modélisation unifié) est un langage de modélisation graphique à base de pictogrammes. Il est apparu dans le monde du génie logiciel, dans le cadre de la « conception orientée objet ». Couramment utilisé dans les projets logiciels, il peut être appliqué à toutes sortes de systèmes ne se limitant pas au domaine informatique. Les 14 diagrammes UML sont dépendants hiérarchiquement et se complètent, de façon à permettre la modélisation d'un projet tout au long de son cycle de vie[19].

Java est un langage de programmation orienté objet créé par James Gosling et Patrick Naughton, employés de Sun Microsystems, avec le soutien de Bill Joy (cofondateur de Sun Microsystems en 1982), présenté officiellement le 23 mai 1995 au SunWorld. Java reprend en grande partie la syntaxe du langage C++. Néanmoins, Java a été épuré des concepts les plus subtils du C++ et à la fois les plus déroutants, tels que les pointeurs et références, ou l'héritage multiple contourné par l'implémentation des interfaces[16].

Python Le langage Python est un langage de programmation open source multi-plateformes et orienté objet. Grâce à des bibliothèques spécialisées, Python s'utilise pour de nombreuses situations comme le développement logiciel, l'analyse de données, ou la gestion d'infrastructures. Il n'est donc pas, comme le langage HTML par exemple, uniquement dédié à la programmation web[12].

Environnement de développement :



Android Studio Est un environnement de développement pour des applications mobiles Android. Il est basé sur IntelliJ IDEA et utilise le moteur de production Gradle. Il peut être téléchargé sous les systèmes d'exploitation Windows, macOS, Chrome OS et Linux[5].



Google Colab ou Colaboratory est un service cloud, offert par Google (gratuit), basé sur Jupyter Notebook et destiné à la formation et à la recherche dans l'apprentissage automatique. Cette plateforme permet d'entraîner des modèles de Machine Learning directement dans le cloud. Sans donc avoir besoin d'installer quoi que ce soit sur notre ordinateur à l'exception d'un navigateur[17].



Chaqueopy est une option pour utiliser python dans les applications Android. Elle fournit tout ce qui est nécessaire pour inclure des composants Python dans une application Android, notamment, l'intégration complète avec le système de construction Gradle standard d'Android Studio, des API pour appeler du code Python depuis Java/Kotlin et vice-versa ainsi qu'une large gamme de packages Python tiers[8].

5.2 Préparation de la base de données

La préparation des données présente la partie la plus chronophage de l'expérience. Elle comprend aussi bien la collecte que le pré-traitement des données, qui sont des tâches essentielles pour former un bon modèle d'apprentissage automatique.

5.2.1 Collecte des données

Nous avons rassemblé les images de 19 sites touristiques marocains situés dans différentes villes à l'aide de deux méthodes :

- La première utilise l'extension Google Chrome **Download All Images**, qui permet de télécharger le nombre des images souhaitées. Elle offre aussi la possibilité de filtrer les images soient par la taille, les dimensions ou le type.
- La deuxième méthode est le **Web Scraping**, nous avons créé un code Python qui prend la liste des monuments en entrée et retourne les images des monuments qui se trouvent sur l'internet.

Malgré tout cela, les images collectées par les deux méthodes ne représentent pas exactement les monuments souhaités, elles peuvent aussi représenter d'autres sites. Pour ce faire, nous avons filtré les images, ce qui explique le nombre d'images de chaque monument qui ne dépasse pas 60, Ce qui fait un total de 1140 images collectées.

Les différentes classes sont représentées dans le tableau suivant :

Site	Image du site	Nombre d'images
Bab El-khamis		60
Bab Mansour		60
Bab berdaine		60
Bab chellah		60
Grande mosquée de Meknès		60
Heri es Souani		60

Koutoubia		60
Medersa Attarine		60
Menara		60
Mosquée Hassan II		60
Musee Mohammed VI		60
Musee Nejjarin		60

Oualili		60
Palais El Badi		60
Palais Royal de Fès		60
Porte Bab Boujloud		60
Tannerie Chouara		60
Tombeaux saadiens		60

Tour hassan		60
-------------	--	----

TABLE 5.1 – Le nombre d’images fournies pour chacun des sites.

5.2.2 Labélisation

Nous avons labélisé les images de notre base de données par **Roboflow** qui est un outil en ligne gratuit pour la préparation des données par la labélisation, l’augmentation et la répartition automatique des données.

Labélisation se fait par dessiner une boîte englobante autour de chaque objet que nous voulons et l’associe à sa classe. Roboflow nous offre également la possibilité d’enregistrer l’annotation avec différents formats, y compris le format YOLO.

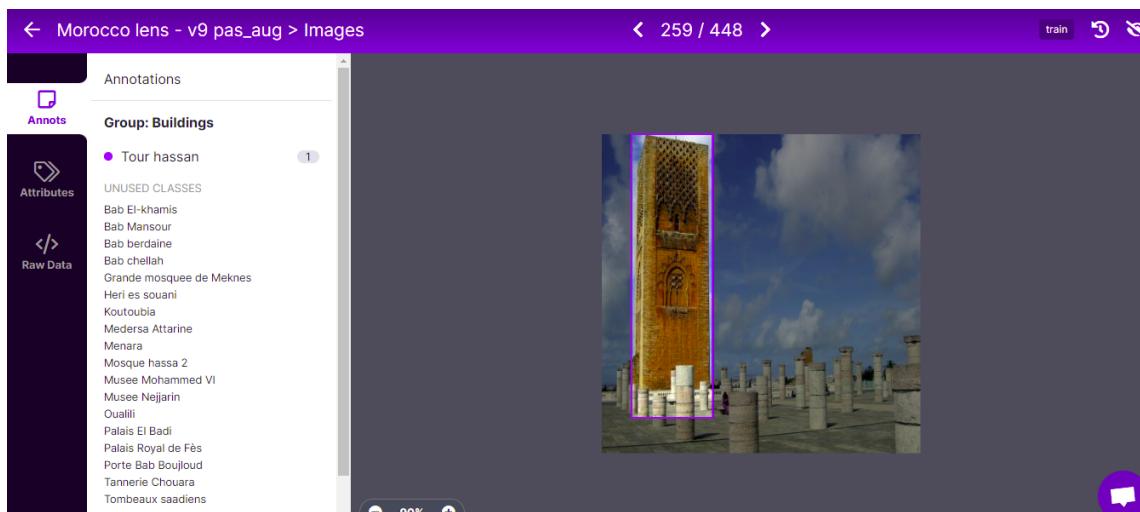


FIGURE 5.1 – Exemple d’annotation d’image avec Roboflow.

Les labels sont représentés sous forme d’un fichier texte, chaque label est représenté sous forme d’une ligne qui contient 5 colonnes. La première colonne est le numéro de la classe et les quatre colonnes restantes sont les coordonnées d’objet X_centre, Y_center, largeur de la boîte et hauteur de la boîte normalisées et se situant dans l’intervalle de 0 à 1. Elles sont représentées au format suivant :

```
2 0.3076923076923077 0.4788524343924817 0.6153846153846154 0.8182817918618864
2 0.6574519230769231 0.5084134615384616 0.6850961538461539 0.9783653846153846
```

FIGURE 5.2 – Exemple d’un label.

RoboFlow offre également l'option de diviser vos données que vous voulez, en donnant un pourcentage à l'entraînement, à la validation et au test. Nous avons divisé nos images et leurs labels en trois ensembles l'entraînement (80%), test (10%) et validation (10%), ce qui est le plus recommandé et vous donne une bonne précision dans la prédiction. La séparation est mise en œuvre de manière aléatoire pour garantir l'homogénéité, attribuer les données dans trois fichiers distincts, pour l'entraînement, le test et la validation.

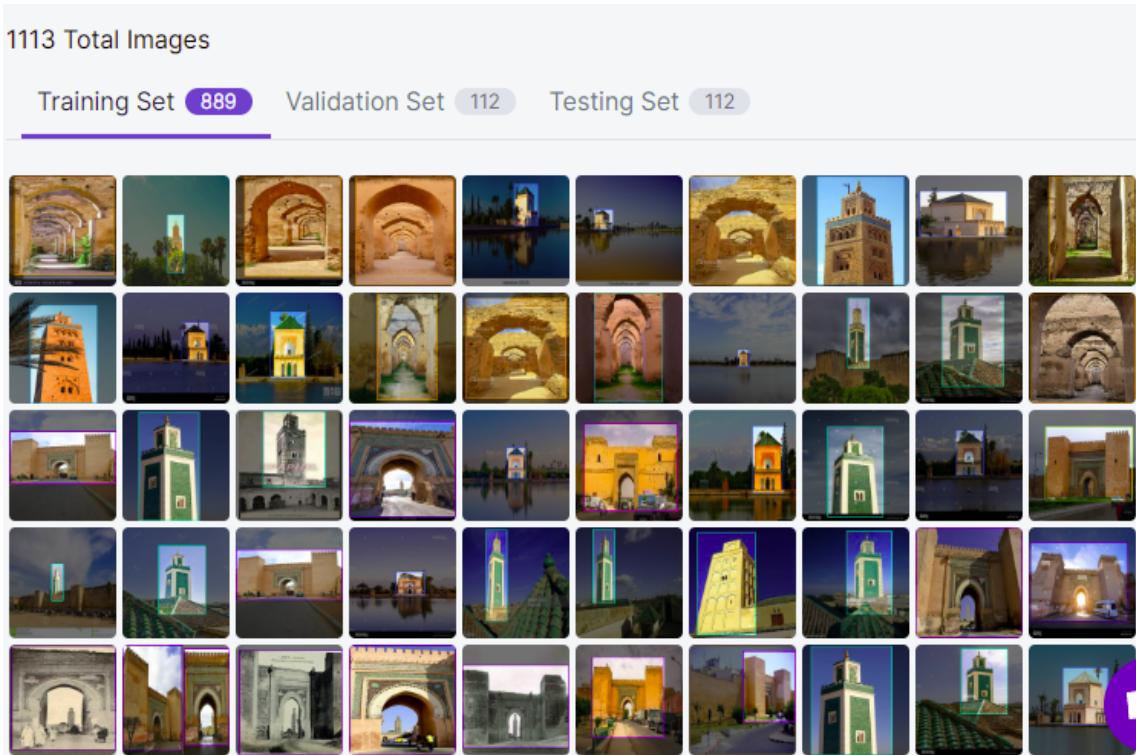


FIGURE 5.3 – Répartition des données.

5.2.3 Augmentation des données

L'augmentation des données est un ensemble de techniques visant à augmenter artificiellement la quantité de données en générant de nouveaux points de données à partir de données existantes. Il s'agit notamment d'apporter de petites modifications aux données ou d'utiliser des modèles d'apprentissage en profondeur pour générer de nouveaux points de données [18]. Voici les techniques qui sont couramment appliquées :

- rotation aléatoire.
- retournement vertical et horizontal.
- translation (l'image est déplacée le long de la direction X, Y).
- zoom.
- niveaux de gris.
- changement de contraste.
- ajouter du bruit.

pour augmenter nos données nous avons utilisé roboflow parce qu'il a offert la possibilité d'augmenter les données, Les techniques choisies sont représentées dans la [Figure 5.4](#) .



FIGURE 5.4 – Augmentation des données

Nous avons obtenu 3350 images au total après cette augmentation, qui sont divisées en trois ensembles entraînement 2680 (80%), validation 336 (10%) et test 336 (10%).

5.3 Choix et construction du modèle

Nous utilisons la version 5 de YOLO, qui a été lancée par Ultralytics en juin 2020 et est maintenant l'algorithme d'identification d'objet le plus avancé disponible. Il s'agit d'un nouveau réseau neuronal convolutif (CNN) qui détecte les objets en temps réel avec une grande précision. Cette approche utilise un seul réseau de neurones pour traiter l'ensemble de l'image, puis la sépare en parties et prédit les cadres de délimitation et les probabilités pour chaque composant. Ces boîtes englobantes sont pondérées par la probabilité attendue. La méthode "You Only Look Once" fait des prédictions après une seule propagation vers l'avant à travers le réseau de neurones. Il délivre ensuite les éléments détectés après suppression non maximale (ce qui garantit que l'algorithme de détection d'objet n'identifie chaque objet qu'une seule fois) [13].

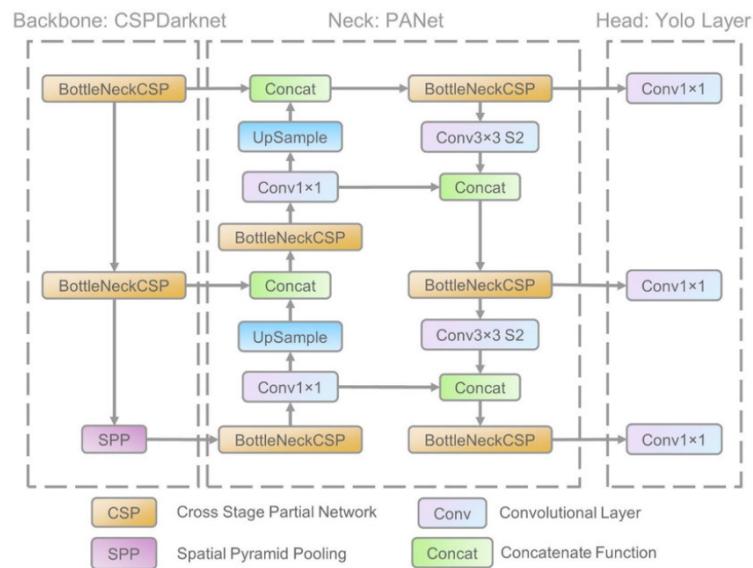


FIGURE 5.5 – Architecture de YOLOv5.[13]

- **Backbone** : Backbone du modèle est principalement utilisé pour extraire les caractéristiques clés d'une image d'entrée. Les CSP (Cross Stage Partial Networks) sont utilisés comme épine dorsale dans YOLO v5 pour extraire des caractéristiques riches en caractéristiques utiles à partir d'une image d'entrée.
- **Neck** : Neck est principalement utilisé pour créer des pyramides de caractéristiques. Les pyramides de caractéristiques aident les modèles à généraliser avec succès en ce qui concerne la mise à l'échelle des objets. Il aide à l'identification du même objet dans différentes tailles et échelles. Les pyramides de caractéristiques sont très utiles pour aider les modèles à fonctionner efficacement sur des données inédites. D'autres modèles, tels que FPN, BiFPN et PANet, utilisent différentes sortes d'approches pyramidales de caractéristiques. PANet est utilisé comme cou dans YOLO v5 pour obtenir des pyramides de fonctionnalités.
- **Head** : Head du modèle est principalement responsable de l'étape de détection finale. Il utilise des boîtes d'ancrage pour construire des vecteurs de sortie finaux avec des probabilités de classe, des scores d'objectivité et des boîtes englobantes.

La tête du modèle YOLO v5 est la même que dans les précédentes éditions YOLO V3 et V4 [13].

5.3.1 Avantages et inconvénients de Yolov5

- Il est environ 88% plus petit que YOLOv4 (27 Mo contre 244 Mo).
- Il est environ 180% plus rapide que YOLOv4 (140 FPS contre 50 FPS).
- Il est à peu près aussi précis que YOLOv4 sur la même tâche (0,895 mAP contre 0,892 mAP) [13].

Mais le problème majeur est qu'il n'y a pas de documentation yolov5 officielle, De plus, YOLO v5 est toujours en cours de développement.

5.3.2 Construction du modèle

Il est fortement recommandé, avant de commencer à entraîner le modèle, de modifier le type d'exécution pour GPU, car Google Colab offre la possibilité d'utiliser son GPU, qui sans aucun doute peut fournir des résultats beaucoup plus rapides.cloner le référentiel YOLO v5 et configurer les dépendances nécessaires pour exécuter YOLOv5.

```
[ ] !git clone https://github.com/ultralytics/yolov5 # clone
%cd yolov5
%pip install -qr requirements.txt # install

import torch
import utils
display = utils.notebook_init() # checks
```

FIGURE 5.6 – Code utilise pour GPU, installer les dépendances et le référence de yolov5

Fichiers de configuration

Fichiers de configuration décrit les paramètres du jeu de données. Étant donné que nous nous entraînons sur notre ensemble de données personnalisé sur les monuments touristiques, nous allons modifier ce fichier et fournir : les chemins vers les ensembles de données d'entraînement, de validation et de test (facultatif) ; le nombre de classes (nc) ; et les noms des classes. Nous

avons nommé notre fichier de configurations de données personnalisées 'custom_data.yaml' et l'avons placé sous le répertoire 'data'. Le contenu de ce fichier YAML est le suivant :

```
train: ../train/images
val: ../valid/images
test: ../test/images

nc: 19
names: ['Bab El-khamis', 'Bab Mansour', 'Bab berdaine', 'Bab chellah', 'Grande mosquee de Meknes',
'Heri es souani', 'Koutoubia', 'Medersa Attarine', 'Menara', 'Mosque hassa 2', 'Musee Mohammed VI',
'Musee Nejjarin', 'Oualili', 'Palais El Badi', 'Palais Royal de Fes', 'Porte Bab Boujloud', 'Tannerie Chouara',
'Tombeaux saadiens', 'Tour hassan']
```

FIGURE 5.7 – Le ficheie custom_data.yaml

L’entraînement

Pour entraîner le modèle YOLOv5s sur des données personnalisées consiste à précise l’ensemble de données, la taille du lot, l’époque, la taille de l’image et le poids du yolov5 préformé. Tous les résultats d’entraînement sont enregistrés dans runs/train/ avec des répertoires d’exécution incrémentiels.

```
!python train.py --img 640 --batch 16 --epochs 90 --data custom_data.yaml --weights yolov5s.pt --cache
```

FIGURE 5.8 – Code de l’entraînement

- **img** : taille de l’image en pixels
- **Batch** : taille des lots.
- **Epochs** : nombre d’époques.
- **Data** : chemin d’accès au fichier de configuration des données (custom_data.yaml).
- **Weights** : schemain vers les poids initiaux.
- **Cache** : images en cache pour une formation plus rapide.

Test du modèle

Après avoir entraîné notre modèle, nous allons tester ses performances pour voir l’exactitude de la détection. Les résultats sont enregistrés dans runs/detect en exécutant la commande suivante :

```
[ ] !python detect.py --weights runs/detect/exp/weights/best.py --img 640 --conf 0.25 --source data/images
display.Image(filename='runs/detect/exp/zidane.jpg', width=600)
```

FIGURE 5.9 – Code de test

- **Source** : le chemin de l’image par laquelle en applique le test.
- **Conf** : seuil de confiance.

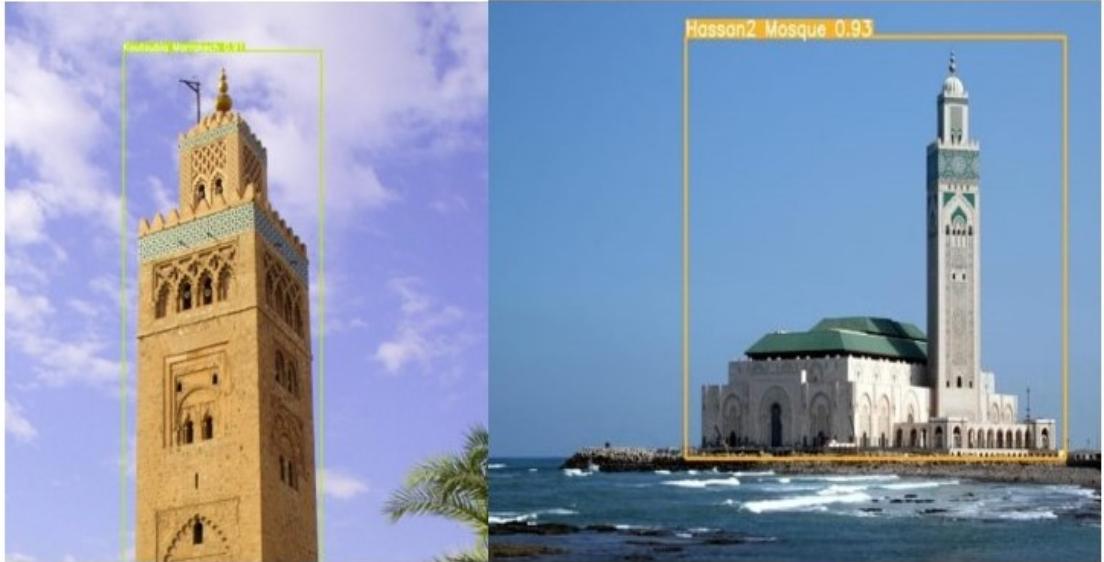


FIGURE 5.10 – Résultat de test

Évaluation du modèle

Dans cette partie nous allons évaluer les performances de notre modèle de détection grâce au calcul de ces paramètres suivants :

— Précision :

est le nombre de documents pertinents retrouvés rapporté au nombre de documents total proposé pour une requête donnée. On calcule la précision avec la formule suivante :

$$\text{Précision}_i = \frac{\text{Nombre de documents correctement attribués à la classe}_i}{\text{Nombre de documents attribués à la classe}_i}.$$

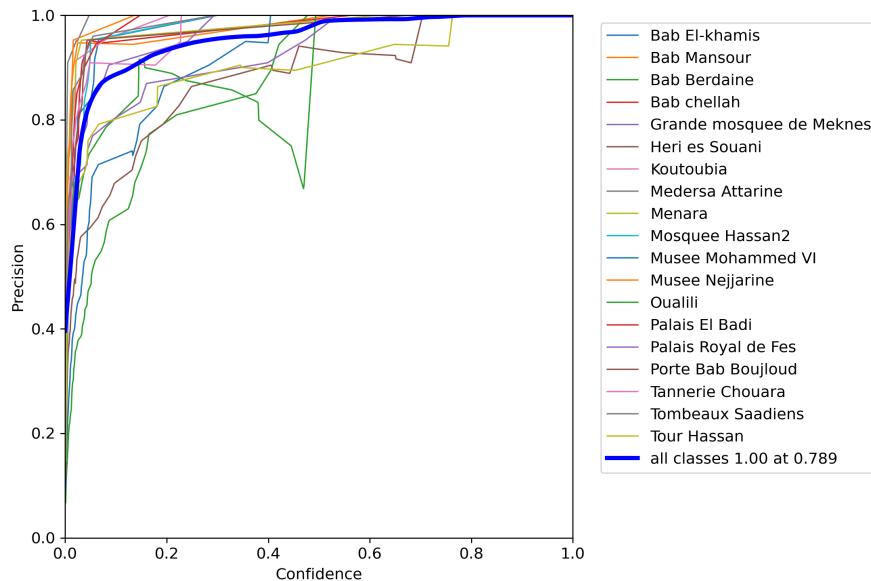


FIGURE 5.11 – Diagramme représentant des valeurs de précision.

— **Rappel** : est défini par le nombre de documents pertinents retrouvés au regard du nombre de documents pertinents que possède la base de données. Le rappel est calculé

comme suit :

$$Rappel_i = \frac{\text{Nombre de documents correctement attribués à la classe}_i}{\text{Nombre de documents appartenant à la classe}_i}.$$

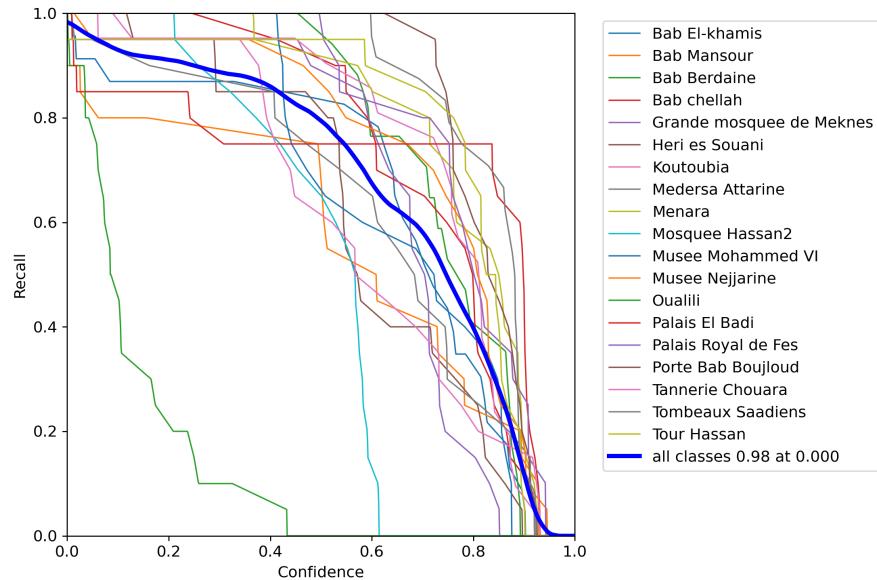


FIGURE 5.12 – Diagramme représentant des valeurs de rappel

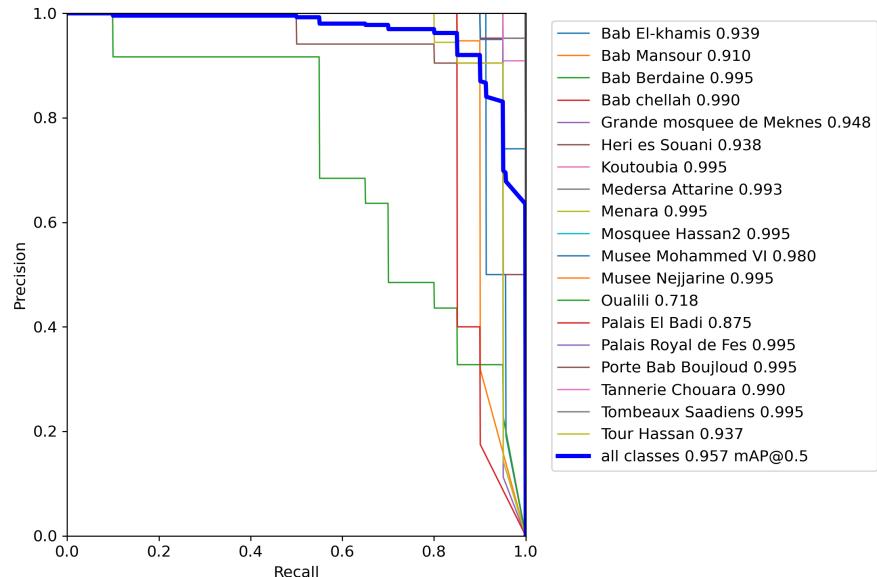


FIGURE 5.13 – Diagramme représentant la précision et le rappel.

- **Matrice de confusion** : ou tableau de contingence est un outil permettant de mesurer les performances d'un modèle de Machine Learning en vérifiant notamment à quelle fréquence ses prédictions sont exactes par rapport à la réalité dans des problèmes de classification.

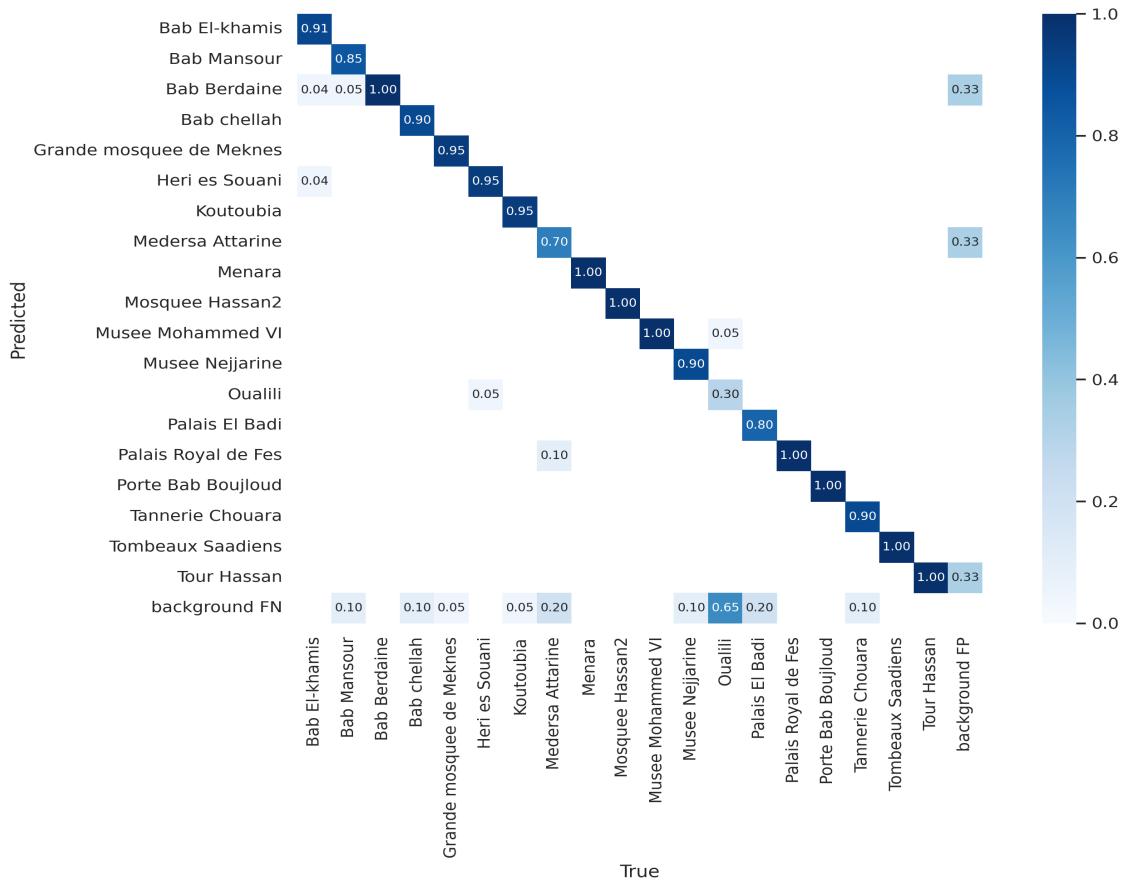


FIGURE 5.14 – Matrice de confusion

Performances du modèle

TABLE 5.2 présentation des résultats obtenus après l’entraînement du modèle par une base des données qui contient 3350 images et qui est divisé en trois ensembles entraînement (80%), validation (10%) et test (10%).

Classe	Précision	Rappel	mAP@.5
Bab El-khmis	0.99	0.91	0.93
Bab Mansour	0.96	0.85	0.91
Bab Berdaine	0.81	1	0.99
Grande mosquée de Meknès	0.95	0.95	0.94
Bab chellah	0.96	0.9	0.99

Heri es Souani	0.86	0.95	0.93
Koutoubia	1	0.98	0.99
Medersa Attarine	1	0.79	0.99
Menara	0.97	1	0.99
Mosquée Hassan II	0.99	1	0.99
Musée Mohammed VI	0.88	0.95	0.98
Musée Nejjarine	1	0.94	0.99
Oualili	0.87	0.34	0.71
Palais El Badi	1	0.83	0.87
Palais Royal de Fés	0.88	1	0.99
Bab Boujloud	0.96	1	0.99
Tannerie Chouara	1	0.94	0.99
Tambeaux Saadiens	0.99	1	0.99
Tour Hassan	0.88	0.95	0.93

TABLE 5.2 – Performances obtenues du modèle.

mAP : une moyenne des valeurs de précision moyenne, qui est une autre moyenne des précisions moyennes pour toutes les classes.

mAP@.5 : mAP avec IoU thresholds de 0.5

5.4 Réalisation de l'application mobile

Comme nous l'avons mentionné précédemment, notre objectif consiste à créer une application mobile pour les touristes qui viennent au Maroc, nous avons essayé de rendre l'application facile autant que possible, l'interface de l'application ne contient que le composant nécessaire pour rendre le processus de détection très facile pour l'utilisateur.

5.4.1 L'intégration du Modèle dans l'application

Pour intégrer un modèle yolov5 dans une application mobile Android, il existe plusieurs méthodes, parmi les méthodes efficaces, les méthodes basées sur le cloud, car elles rendent la détection très rapide et rendent les performances de l'application très bonnes, comme Firebase.

Firebase nécessite de convertir votre modèle en modèle TensorFlow Lite. En déployant des modèles avec Firebase, vous pouvez réduire la taille de téléchargement initial de votre application et mettre à jour les modèles ML de votre application sans publier une nouvelle version de votre application, le problème que nous avons rencontré avec cette méthode est payant.

Une deuxième méthode consiste à utiliser des applications Android pré-construites **Tensorflow lite** pour la détection d'objets, dont tout ce que nous avons à faire est d'intégrer notre modèle, mais la personnalisation est limitée et nous ne pouvons pas ajouter les fonctionnalités que nous voulons.

en raison des limitations des deux méthodes précédentes, nous avons dû choisir une autre méthode qui, même si elle ralentit un peu la détection et augmenté la taille de l'application, mais elle nous permet de personnaliser l'application comme nous le souhaitons et d'ajouter les fonctionnalités que nous voulons. Cette méthode permet l'utilisation de Python dans une application Android, **Chaquopy** fournit tout ce dont vous avez besoin pour inclure des composants Python dans une application Android, notamment :

- Intégration complète avec le système de construction Gradle standard d'Android Studio.
- Des API simples pour appeler du code Python depuis Java/Kotlin, et vice versa.
- Une large gamme de packages Python tiers, notamment SciPy, OpenCV, TensorFlow et bien d'autres.

5.4.2 Interfaces graphiques de L'application

La page d'accueil de l'application mobile se compose de deux boutons et un lien "à propos",



PRENDRE UNE PHOTO

CHOISIR UNE IMAGE



FIGURE 5.15 – Interface d'accueil de l'application.

Un clic sur "à propos" affiche une petite introduction à l'application



Cette application est un projet de fin d'études créé par Hamid Oufakir et Reda Lamfoueh sous l'encadrement du professeurs Sara Sekkate et Siham Akil. L'objectif principal de cette application est de détecter les monuments touristiques marocains célèbres et de fournir des informations sur ces monuments afin d'aider les touristes à reconnaître la culture marocaine. Cette application est capable de détecter 19 monuments marocains dans différentes villes.



FIGURE 5.16 – Interface d'à propos de nous.

Ce bouton permet l'utilisateur de choisir la langue soit le français ou l'anglais,



FIGURE 5.17 – Bouton pour choisir la langue.

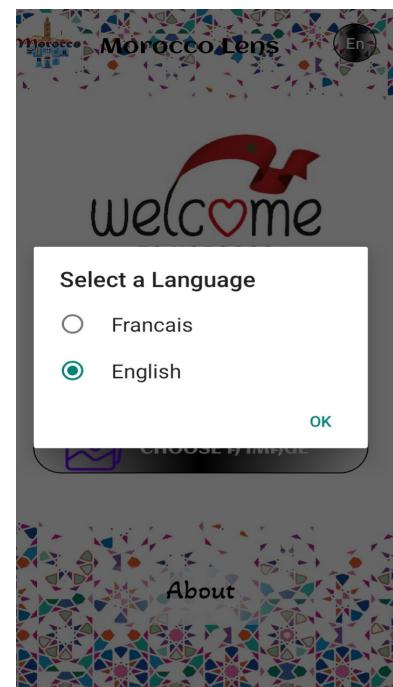


FIGURE 5.18 – Liste des langues.

Le premier bouton permet à l'utilisateur de prendre une photo du monument qu'il veut détecter et dont il veut avoir plus de détails,



FIGURE 5.19 – Bouton pour prendre les photos.

Le deuxième bouton permet à l'utilisateur d'obtenir une photo de la galerie ou de la mémoire interne de son téléphone,

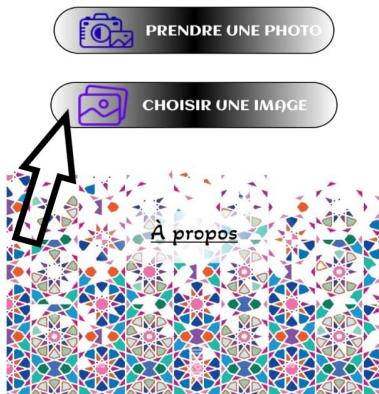


FIGURE 5.20 – Bouton pour choisir les images.

Puis l'image apparaîtra au centre de l'interface, et il y aura un bouton de détection juste en dessous de l'image,



FIGURE 5.21 – Bouton pour lancer la détection.

Après avoir cliqué sur le bouton de détection, le nom du monument apparaîtra en bas en gras, dans l'image, une boîte englobante avec le nom et la précision sera esquissée autour du monument, et sous l'image il y aura les informations sur le monument avec un lien "plus d'informations" à la fin,



Mosquee Hassan2

The Hassan II Mosque is a mosque in Casablanca, Morocco. It is the second largest functioning mosque in Africa and is the 7th largest in the world. Its minaret is the world's second tallest minaret at 210 metres. [Plus d'informations...](#)



FIGURE 5.22 – Interface des résultats.

Cliquer sur le lien "plus d'informations" redirigera l'utilisateur vers google avec le nom du monument dans la barre de recherche,

Mosquée Hassan II
Mosquée à Casablanca

La mosquée Hassan-II Maroc). Érigée en partie sur la mer, elle est un complexe religieux et culturel, aménagée sur neuf hectares et comporte une salle de prières, une salle d'ablutions, des bains, une école coranique (madrasa), une bibliothèque, un musée et une Académie des arts traditionnels. [Wikipédia](#)

H www.hertz.ma › ... › Patrimoine
La mosquée Hassan II de Casablanca - Hertz Maroc

Après la mosquée de la Mecque, la mosquée Hassan II de Casablanca est le deuxième plus haut monument religieux de la planète. Avec un minaret culminant à 210 ...

fr.m.wikipedia.org › wiki › Mosqué...
Mosquée Hassan-II - Wikipédia

Images Vidéos English mosqu...

G Mosque Hassan2

FIGURE 5.23 – La fenêtre de google.

5.5 Conclusion et perspectives

Ce projet de fin d'études nous a permis de découvrir les techniques de développement des applications d'intelligence artificielle et plus précisément le traitement des images et la détection des objets en se basant sur le Framework (YOLO) pour obtenir des résultats de bonnes performances. Ce projet nous a aussi permis de prendre connaissance du niveau actuel de la recherche scientifique dans ce domaine.

Au cours de la phase de réalisation de notre application, nous avons élaboré une étude préalable sur les applications existantes et qui sont en relation avec la détection des objets. Cette phase a constitué le point de départ pour l'étape d'analyse et de spécification des besoins.

Une fois nos objectifs fixés, nous avons enchaîné avec la conception afin de mener à bien notre projet. À son issue, nous avons procédé à la phase de réalisation.

À la fin de ce projet, nous avons pu développer une application qui pourrait aider les utilisateurs et surtout les touristes du Maroc à identifier les monuments historiques dans un temps réduit et d'une façon très simple. Ceci pourrait en effet contribuer au Marketing touristique du pays et donner une nouvelle impulsion au secteur du tourisme.

Nous sommes satisfaits du travail que nous avons fait jusqu'à présent dans notre projet et notre application mobile. Cependant, il reste certaines choses que nous voulons encore améliorer comme par exemple :

- Étant donné que la détection est un peu lente pour certains sites en raison de la méthode d'intégration du modèle utilisée, nous souhaitons explorer comment intégrer notre modèle en utilisant des méthodes basées sur le cloud, ce qui rendra la détection beaucoup plus rapide.
- Pour certains monuments comme Oualili qui ne contiennent pas d'objets homogènes, la précision reste faible, nous allons donc essayer de trouver les moyens d'augmenter la précision au maximum.
- Nous avons également rencontré un problème par android studio pour régler les images qui sont capturées par la caméra, nous tenterons de résoudre le problème.

Références

- [1] URL : <https://deeplobe.ai/object-detection-a-simplified-solution-in-2021/>.
- [2] URL : <https://www.v7labs.com/blog/yolo-object-detection>.
- [3] URL : <https://www.kdnuggets.com/2018/09/object-detection-image-classification-yolo.html>.
- [4] URL : <https://towardsdatascience.com/guide-to-car-detection-using-yolo-48caac8e4ded>.
- [5] *Android Studio*. URL : <https://developer.android.com/studio>.
- [6] Chinmoy BORAH. *Evolution of Object Detection*. Nov 1, 2020. URL : <https://medium.com/analytics-vidhya/evolution-of-object-detection-582259d2aa9b>.
- [7] Jason BROWNLEE. *A Gentle Introduction to Object Recognition With Deep Learning*. January 27, 2021. URL : <https://machinelearningmastery.com/object-recognition-with-deep-learning/>.
- [8] *chaquopy SDK Python pour Android*. URL : <https://chaquo.com/chaquopy>.
- [9] *Computer vision*. URL : <https://www.quantmetry.com/glossaire/computer-vision/>.
- [10] *Définition Computer Vision*. URL : <https://actualiteinformatique.fr/intelligence-artificielle/definition-computer-vision>.
- [11] Eddie FORSON. *Understanding SSD MultiBox — Real-Time Object Detection In Deep Learning*. Nov 18, 2017. URL : <https://high-tech-guide.com/article/what-is-ssd-model-in-machine-learning>.
- [12] la rédaction FUTURA. *Python : qu'est-ce que c'est ?* futura-sciences. URL : <https://www.futura-sciences.com/tech/definitions/informatique-python-19349/>.
- [13] Aryan GARG. *How to Use Yolo v5 Object Detection Algorithm for Custom Object Detection*. December 14, 2021. URL : <https://www.analyticsvidhya.com/blog/2021/12/how-to-use-yolo-v5-object-detection/>.
- [14] Jonathan HUI. *SSD object detection : Single Shot MultiBox Detector for real-time processing*. Mar 14, 2018. URL : <https://jonathan-hui.medium.com/ssd-object-detection-single-shot-multibox-detector-for-real-time-processing-9bd8deac0e06>.
- [15] *La Faculté des Sciences et Techniques de Mohammedia*. URL : <https://www.fstm.ac.ma>.
- [16] Wilfried MBOUNDA. *Les bases de la programmation Java*. Nov 16, 2018. URL : <https://medium.com/@wilfried.mbounda/les-bases-de-la-programmation-java-partie-1-cf4eeff57e99>.
- [17] Henri MICHEL. *Google Colab : Le guide Ultime*. ledatascientist, Nov 16, 2018. URL : <https://ledatascientist.com/google-colab-le-guide-ultime/>.
- [18] Jacob SOLAWETZ. *Data Augmentation in YOLOv4*. mai 13, 2020. URL : <https://blog.roboflow.com/yolov4-data-augmentation/>.

- [19] UML. URL : <https://www.lucidchart.com/pages/fr/langage-uml>.
- [20] WHAT IS SSD MODEL IN MACHINE LEARNING ? June 27, 2022. URL : <https://high-tech-guide.com/article/what-is-ssd-model-in-machine-learning>.