République Algérienne Démocratique et Populaire

Ministère de l'Enseignement Supérieur et de la Recherche Scientifique

Université de Blida 1

Faculté des Sciences

Département de l'informatique

****

Mémoire de fin d'étude pour l'obtention du diplôme de

Master

Spécialités : **Ingénierie logiciel et Système Informatique et Réseau**

Thème :

Reconnaissance d'images immobilières : Application à un site de vente immobilière

Réalisé par :

Sanogo Mohamed

Ibrahima Soly Coulibaly

Encadré par :

Mme Zahra Fatma Zohra

Soutenu : 13/06/2024

Devant le jury Composé de

Mme CHERFA Imane (Présidente)

Mme. MANCER Yasmine (Examinatrice)

Promotion : 2023-2024

Dédicaces

Je dédie ce travail en tout premiers lieu à mes très chers parents, qui ont le droit de recevoir mes chaleureux remerciements pour le courage et le sacrifice qu'ils ont consentis pendant la durée de mes études, J'espère qu'ils trouveront dans ce travail toute l'expression de ma reconnaissance.

À mes frères et sœurs. À mes grands-parents. À toute ma famille. En fin, je dédie ce travail à tous mes enseignants du primaire à l'université, à mes amis et à toute personne qui a un jour fait partie de ma vie.

SANOGO Mohamed.

Je tiens à exprimer ma gratitude la plus sincère à mon père, Ibrahima Baba Coulibaly. Ton soutien indéfectible, tant moral que financier, tout au long de mon parcours scolaire, a été le pilier de ma réussite. Tes encouragements constants ont été la force motrice qui m'a permis de réaliser ce projet de fin d'études de master. Merci du fond du cœur.

À ma mère, Fatoumata Coulibaly, je dédie ce travail avec toute ma reconnaissance. Ton amour inconditionnel et ta tendresse m'ont insufflé le courage et la confiance nécessaires pour mener à bien ce projet. Ton soutien affectueux a été un réconfort inestimable tout au long de ce parcours.

Je souhaite également exprimer ma profonde gratitude à mon grand frère, Bakary Coulibaly. Ton aide précieuse, tes conseils avisés et tes encouragements constants m'ont guidé à chaque étape de mes études universitaires. Ta présence à mes côtés a été une source de motivation et d'inspiration. Merci infiniment pour tout ce que tu as fait pour moi.

Enfin, je remercie chaleureusement mes sœurs pour leur bienveillance et leur soutien continu. Votre présence et vos encouragements ont été d'une grande aide et m'ont permis de surmonter les défis rencontrés en cours de route.

Coulibaly Soly Ibrahima.

Remerciements

Avant toutes choses, nous remercions Allah Tout Puissant de nous avoir donné la volonté et le courage d'accomplir notre cursus universitaire et de mener ce travail jusqu'à la fin et cela dépit des difficultés.

On tient aussi à exprimer nos vifs remerciements à notre Encadreur PHD ZAHRA FATMA ZOHRA qui nous a aidés à réaliser ce Travail. Nos profondes gratitudes vont également à l'attention des Membres de jury qui nous ferons honneur d'examiner notre travail, Ainsi tous les enseignants qui ont contribués à notre formation. Afin de ne pas oublier personne, nous remercions tous ceux qui Ont contribués de près ou de loin à l'élaboration de ce projet.

Table des matières

Liste des figures

Liste des tableaux

Résumé

La reconnaissance et la détection d'images immobilières sont des technologies en plein essor, offrant des perspectives prometteuses pour le secteur de l'immobilier en ligne. Ces technologies automatisent l'analyse des images de biens immobiliers pour extraire des informations cruciales telles que le type de propriété (maison, appartement, hôtel) et ses caractéristiques.

Ce travail se concentre sur l'exploration des méthodes les plus efficaces pour la reconnaissance et la détection d'images immobilières, et leur application dans un contexte pratique sur un site web de vente immobilière. À cette fin, une étude approfondie des articles scientifiques a été menée pour identifier les modèles les plus performants pour l'analyse d'images. Deux APIs ont été développées pour notre application web : la première utilise le modèle VGG pour la reconnaissance, tandis que la seconde utilise le modèle YOLO pour la détection. Ces choix sont basés sur leur performance comparée à d'autres modèles étudiés, tels que ResNet pour la reconnaissance et U-Net pour la détection.

Les objectifs principaux de cette recherche sont de fournir une analyse approfondie des technologies de reconnaissance et de détection d'images immobilières, et de démontrer leur application concrète dans un environnement de vente immobilière en ligne.

Les mots clés : reconnaissance d’immeuble, détection d’immeuble, apprentissage automatique, apprentissage profond, application web de vente immobilière, image aérienne, image satellitaire.

Abstract

Real estate image recognition and detection are rapidly growing technologies, offering promising prospects for the online real estate sector. These technologies automate the analysis of property images to extract crucial information such as the type of property (house, apartment, hotel) and its characteristics.

This work focuses on exploring the most effective methods for real estate image recognition and detection, and their practical application on a real estate sales website. To this end, a thorough review of scientific articles was conducted to identify the most efficient models for image analysis. Two APIs were developed for our web application: the first uses the VGG model for recognition, while the second uses the YOLO model for detection. These choices are based on their performance compared to other studied models, such as ResNet for recognition and U-Net for detection.

The primary objectives of this research are to provide an in-depth analysis of real estate image recognition and detection technologies, and to demonstrate their concrete application in an online real estate sales environment.

Keywords: building recognition, building detection, machine learning, deep learning, real estate sales web application, aerial image, satellite image.

الملخص

التعرف على الصور العقارية واكتشافها هي تقنيات سريعة النمو، تقدم آفاقًا واعدة لقطاع العقارات عبر الإنترنت. تعمل هذه التقنيات على أتمتة تحليل صور العقارات لاستخراج معلومات مهمة مثل نوع العقار (منزل، شقة، فندق) وخصائصه.

يركز هذا العمل على استكشاف أكثر الطرق فعالية للتعرف على الصور العقارية واكتشافها، وتطبيقها في سياق عملي على موقع ويب لبيع العقارات. ولتحقيق ذلك، تم إجراء دراسة متعمقة للمقالات العلمية لتحديد النماذج الأكثر كفاءة لتحليل الصور. تم تطوير واجهتي برمجة تطبيقات (API) الأولى تستخدم نموذج VGG للتعرف، لتطبيق الويب الخاص بنا: للاكتشاف. تستند هذه الاختيارات إلى أدائها مقارنةً بالنماذج الأخرى التي تمت دراستها، مثل ResNet للتعرف و U-Net للاكتشاف. بينما تستخدم الثانية نموذج YOLO.

الأهداف الرئيسية لهذه الدراسة هي توفير تحليل معمق لتقنيات التعرف على الصور العقارية واكتشافها، وإثبات تطبيقها الفعلي في بيئة بيع العقارات عبر الإنترنت.

الكلمات المفتاحية: التعرف على المباني، اكتشاف المباني، التعلم الآلي، التعلم العميق، تطبيق ويب لبيع العقارات، الصور الجوية، الصور الفضائية.

Introduction générale

La reconnaissance et la détection d'images immobilières est un domaine en plein essor de l'intelligence artificielle qui trouve des applications variées, notamment dans le secteur de l'immobilier. Cette technologie permet d'analyser et d'interpréter automatiquement les images émises pour en extraire des informations pertinentes, telles que le type de bien (maison, appartement, hôtel, etc.), le nombre de pièces, etc.

La reconnaissance et la détection d'images immobilières restent une tâche difficile. Certaines méthodes ont relativement plus de capacité de détection et de reconnaissance d'images immobilières, tandis-que d'autres n'ont que des capacités très réduites. Les différentes catégories des modèles et des méthodes d'apprentissage profond devraient être étudiées. Les sélections de ces modèles dépendront de la meilleure précision obtenue par rapport à l'autre. Par conséquent, les choix de ces méthodes soulèvent les questions de recherche suivante :

- Quelles sont les technologies et les méthodes les plus adaptées pour la détection et la reconnaissance d'images immobilières à partir des différentes sources d'images (aériennes et satellitaires) ?
- Comment concevoir une api de détection et de reconnaissance d'images immobilières et l'intégrer dans une application web de vente immobilière ?

Dans ce contexte, ce travail vise à étudier diverses méthodes pour résoudre le problème de reconnaissance et de détection d'images immobilières afin de trouver celles capables d'analyser efficacement ces images.

L'intérêt de développer une application web de vente immobilière intégrant des APIs de reconnaissance et de détection réside dans le fait que l'administrateur n'a plus besoin de vérifier manuellement les informations fournies par les fournisseurs, ni de se déplacer pour confirmer la présence du bâtiment à l'adresse indiquée.

Le présent mémoire est constitué d'une introduction générale, conclusion générale et trois chapitres, qui sont présentés comme suit :

Le premier chapitre intitulé « Etudes antérieures » définit les concepts de l'intelligence artificielle et les études antérieures connexes.

Le deuxième chapitre qui est intitulé « Conception d'une plateforme de vente immobilière intégrant une api de détection de bâtiment » présente l'aspect dynamique de notre application

on utilise les diagrammes d'interactions et l'aspect statique on utilise le diagramme de classe de conception.

Le troisième chapitre qui est intitulé « Réalisation et l'implémentation » présente les langages de programmation et les outils qui nous ont servis pour l'implémentation de notre projet. Nous faisons aussi un aperçu sur les interfaces.

Enfin, nous terminerons ce mémoire par une conclusion générale.

Chapitre I : Apprentissage automatique pour la détection et la reconnaissance d'images immobilières

1 Introduction

Dans ce chapitre, nous définissons les concepts de l'apprentissage automatique, y compris l'apprentissage profond et les différents types de réseaux de neurones, en mettant l'accent sur les réseaux de neurones convolutifs. Nous explorerons également l'utilisation d'images aériennes et satellitaires comme sources de données pour ce projet de recherche. En outre, un examen des travaux liés à la reconnaissance et à la détection d'images immobilières sera présenté à la fin de ce chapitre.

2 Apprentissage automatique

L'apprentissage automatique (ML) est un sous-ensemble de l'intelligence artificielle (IA) qui vise à permettre aux machines d'apprendre à partir de données sans être explicitement programmées. Le ML vise à apprendre aux ordinateurs à améliorer leurs performances sur une tâche lorsqu'ils sont exposés à davantage de données, en s'adaptant et en tirant les leçons de leurs expériences.

2.1 Catégories de l’apprentissage automatique

2.1.1 L'apprentissage supervisé

L'apprentissage supervisé est une technique fondamentale de l'apprentissage automatique dans laquelle les algorithmes sont entraînés à l'aide d'ensembles de données étiquetés pour prédire les résultats avec précision. Cette approche consiste à fournir au modèle des données d'entrée associées aux sorties correctes correspondantes, permettant à l'algorithme d'apprendre la relation entre l'entrée et la sortie[1]. Grâce à des processus tels que la validation croisée, les poids du modèle sont ajustés jusqu'à ce qu'il corresponde exactement aux données[2].

L’apprentissage supervisé peut être divisé en deux sous-catégories : la classification et la régression :

2.1.1.1 La régression

La régression peut être définie comme une méthode ou un algorithme dans l'apprentissage automatique qui modélise une valeur cible basée sur des prédicteurs indépendants. Il s'agit essentiellement d'un outil statistique utilisé pour découvrir la relation entre une variable dépendante et une variable indépendante [3].

2.1.1.2 La classification

La classification est un processus de catégorisation d'un ensemble de données en classe. Elle peut être effectuée sur des données structurées ou non structurées. Le processus commence par la prévision de la classe des points de données. Les classes sont souvent appelées cible, étiquette ou catégories (en anglais : Label).

2.1.2 L'apprentissage non supervisé

Un apprentissage est dit non supervisé si le système ne dispose d'aucun étiquetage préalable des données, et que le nombre de classes et leur nature n'ont pas été prédéterminés. Aucun expert n'est requis. L'algorithme doit découvrir par lui-même la structure plus ou moins cachée des données. Le partitionnement des données (en anglais : data clustering) est un exemple d'algorithme de l'apprentissage non supervisé [4]. Il existe deux principaux domaines de modèles dans l'apprentissage non-supervisé pour retrouver les regroupements :

- Les méthodes par partitionnement : les algorithmes des K-means.
- Les méthodes de regroupement hiérarchique : classification ascendante hiérarchique (CAH).

2.1.3 L'apprentissage par renforcement

C'est une méthode d'apprentissage ou la machine se comporte comme un agent qui apprend de son environnement d'une manière interactive jusqu'à ce qu'il découvre les comportements qui produisent des récompenses. Il peut être divisé en deux sous-catégories principales :

- Apprentissage par renforcement basé sur les valeurs
- Apprentissage par renforcement basé sur les politiques.

3 L’apprentissage profond

L'apprentissage en profondeur est un espace croissant d’analyse de l’apprentissage automatique. Il comprend plusieurs couches cachées de réseaux neuronaux artificiels. La méthodologie de l’apprentissage profond applique une transformation non linéaire et des abstractions de modèles de haut niveau dans des bases de données plus importantes [4].

3.1 Réseaux de Neurones Convolutionnels (CNN) :

Un réseau neuronal convolutif (CNN) est un type de modèle d'apprentissage profond principalement utilisé dans les tâches de vision par ordinateur, capable d'atteindre des performances similaires à celles des humains en tirant les leçons de l'expérience[5]. Ces réseaux sont particulièrement adaptés à la reconnaissance d'images et au traitement des informations au niveau des pixels, ce qui les rend idéaux pour les applications de détection d'objets critiques et de vision par ordinateur[6]. Ils présentent trois principaux types de couches :

- Couche de convolution
- Couche de pooling
- Couche entièrement connectée (FC).

Les réseaux neuronaux convolutifs comportent plusieurs architectures telles que : AlexNet, VGGNet, ResNet, etc.

3.2 Réseaux de Neurones Récurrents (RNN) :

Un réseau de neurones récurrent (RNN) est un type de réseau de neurones artificiel qui utilise des données séquentielles ou des données de séries temporelles. Ces algorithmes d'apprentissage en profondeur sont couramment utilisés pour des problèmes ordinaux ou temporels, tels que la traduction linguistique, le traitement du langage naturel, la reconnaissance vocale et le sous-titrage d'images ; ils sont incorporés dans des applications populaires telles que Siri, la recherche vocale et Google Translate. Les trois principales architectures des RCNN sont : les réseaux de neurones récurrents bidirectionnels (BRNN), mémoire longue à court terme (LSTM) et unité récurrente fermée (GRU) [8].

4 Les architectures des CNN les plus connues pour la détection et la reconnaissance d'immeubles

Dans cette section, nous présentons les architectures des CNN les plus connues pour la détection et la reconnaissance d'images immobilières

4.1 Architecture U-Net

U-Net est une architecture de réseau neuronal convolutif utilisée pour les tâches de segmentation d'images. Il se caractérise par une structure codeur-décodeur avec des connexions sautées, ce qui permet au modèle de combiner des caractéristiques de bas niveau et de haut niveau pour produire des résultats de segmentation détaillés au niveau du pixel. [9]

4.2 Réseaux VGG

En 2014, le Visual Geometry Group (VGG) [7], de l'Université d'Oxford, a proposé un réseau de plus de 10 couches avec des principes de conception concis pour construire des modèles de réseaux neuronaux plus profonds. La structure du réseau VGG est illustrée à la figure 1, les principaux composants étant un 3×3, l'opération de convolution et un 2×2 max-pooling. La couche convolutive de petite taille a un plus petit nombre de paramètres et de calculs que la couche convolutive avec des noyaux convolutifs de grande taille (par exemple, 5×5 ou 7×7 opérations de convolution dans AlexNet[8]) pour obtenir un champ perceptif similaire. En outre, une caractéristique remarquable consiste à augmenter le nombre de cartes d'entités après l'utilisation de la couche de regroupement, ce qui réduit le phénomène de perte d'informations utiles dans les cartes d'entités après le sous-échantillonnage.

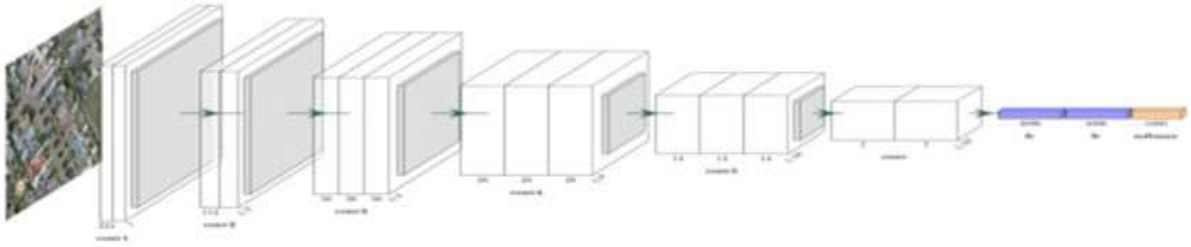


Figure 4-1 : Architecture réseau VGG[7].

Le réseau VGG est l'un des modèles CNN les plus influents en raison de son renforcement de l'idée importante dans DL que les CNN avec des architectures plus profondes peuvent faciliter la représentation hiérarchique des données visuelles. Il pourrait s'agir d'un guide pour la conception structurelle des modèles CNN profonds ultérieurs. Entre-temps, VGG à 16 couches (VGG-16) est devenu l'un des extracteurs de caractéristiques courants pour les tâches en aval.

4.3 ResNet

Présenté en 2015, ResNet est un résultat de recherche historique qui a poussé les réseaux neuronaux vers des couches plus profondes. ResNet à 152 couches s'est classé parmi les cinq premiers à l'ILSVRC 2015 avec un taux d'erreur de 3,6 % et a atteint un nouveau record en matière de classification, de détection et de localisation dans une architecture réseau unique. Grâce à des expériences et à l'analyse de plusieurs modèles CNN profonds, il a été constaté que les réseaux profonds subissent une dégradation du réseau lors de l'approfondissement de la couche et ne peuvent pas nécessairement être plus performants que les réseaux peu profonds. En réponse, une structure résiduelle profonde a été proposée, comme le montre la figure 4 , permettant au réseau de passer à l'apprentissage des résidus. Le réseau résiduel apprend de nouvelles informations différentes de celles qui étaient disponibles auparavant, ce qui soulage la pression exercée sur le réseau profond pour apprendre les représentations des entités et mettre à jour les paramètres. Cela permet au modèle DL d'aller une fois de plus dans une direction plus profonde et meilleure [9].

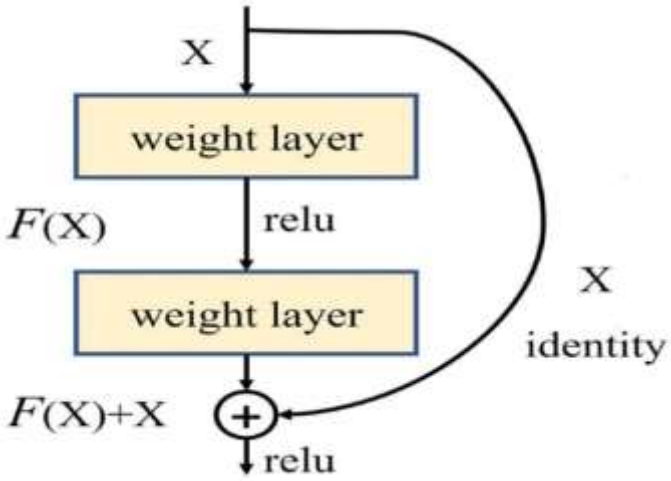


Figure 4-2 : Bloc résiduel de l'architecture ResNet[9].

4.4 YOLO

You Only Look Once (YOLO) propose d'utiliser un réseau neuronal de bout en bout qui prédit les boîtes englobantes et les probabilités de classe en une seule fois. Cette approche diffère de celle adoptée par les algorithmes de détection d'objets précédents, qui réutilisaient des classificateurs pour effectuer la détection. En suivant une approche fondamentalement différente de la détection d'objets, YOLO a obtenu des résultats de pointe, battant largement les autres algorithmes de détection d'objets en temps réel. Les méthodes qui utilisent les réseaux de proposition de régions effectuent plusieurs itérations pour la même image, alors que YOLO

se contente d'une seule itération. Plusieurs nouvelles versions du même modèle ont été proposées depuis la publication initiale de YOLO en 2015, chacune s'appuyant sur son prédécesseur et l'améliorant [10].

5 Étude comparative

Dans cette section, nous faisons une étude comparative d'un ensemble de travaux qui traite le problème de détection des immeubles dans les images aériennes et satellitaires en utilisant les méthodes basées sur l'apprentissage automatique et profond.

5.1 Détection des immeubles à partir des images aériennes

Nous présentons dans la section suivante, un ensemble de travaux qui traitent le problème de détection d'images immobilières à partir d'images aériennes en se basant sur le deep learning et les méthodes classiques de l'apprentissage automatique.

5.1.1 Les approches basées sur les méthodes classiques de l'apprentissage automatique

L'extraction de bâtiments peut être considérée comme une tache de classification binaire, et de nombreuses approches ont utilisé des méthodes basées sur l'apprentissage automatique pour distinguer les bâtiments des non-bâtiments. Pour la détection des bâtiments, les méthodes classiques se composent de plusieurs étapes telles que l'extraction des caractéristiques spectrales, texturales, morphologiques et contextuelles. De telles caractéristiques peuvent décrire les caractéristiques du bâtiment sous différents angles.

Zoraster et al.[11] ont utilisé les méthodes tradionnelles telles que la détection de contours avec l'algorithme de Canny et les caractéristiques avec de Haar pour la détection des bâtiments. Le processus commence par la détection des contours pour générer des candidats de bâtiments, suivie par l'extraction des caractéristiques de Haar pour améliorer la précision de la classification. Ils mettent en avant les défis liés à la définition des paramètres de seuil appropriés et expliquent comment la combinaison de différents seuils peut optimiser les résultats de la détection.

Une autre méthode a été présentée par Wegner et al. [27]. Ils ont utilisé les techniques de machine learning traditionnelles pour détecter les bâtiments dans les images telles que les machines à vecteurs de support (SVM) et les forets aléatoires (Random Forest). L'étude démontre que ces techniques peuvent efficacement distinguer les bâtiments des autres éléments en analysant les caractéristiques spécifiques des images, telles que les textures et les formes.

Les résultats montrent une précision significative dans la détection des bâtiments, soulignant la pertinence des méthodes utilisées.

5.1.2 Les approches basées sur l'apprentissage profond

L'utilisation de deep learning pour la détection des immeubles dans les images aériennes est une technique avancée qui repose sur des modèles de réseaux de neurones profonds. Ces modèles sont capables de traiter et d'analyser de grandes quantités de données visuelles pour identifier et classer les objets avec une précision élevée.

W. Boonpook et al [12] ont proposé une approche pour la détection de bâtiments à partir d'images UAV (la photogrammétrie basée sur les véhicules aériens sans pilote) en utilisant l'architecture SegNet pour la segmentation sémantique. Ils ont entraîné le réseau sur un ensemble de données d'images UAV étiquetées manuellement capturées sur la zone riveraine de la ville de Chongqing, en Chine. L'ensemble de données comprenait divers types de bâtiments, d'architectures et de caractéristiques de scène. En termes de résultat, l'approche proposée a obtenu d'excellentes performances en matière d'extraction précise des informations sur les bâtiments à partir d'images de drones, avec une précision globale moyenne de plus de 90 % pour la détection de bâtiments depuis des emplacements non entraînés. La procédure a ensuite été évaluée sur deux ensembles de données standard, les ensembles de données aériennes ISPRS et Inria, et a atteint des précisions globales de plus de 93 % et 95 % respectivement, confirmant la généralité et l'avantage de la méthode proposée.

Une autre méthode a été présentée par N. Pumpong et al. [13] pour la détection des bâtiments dans les aéroports utilise l'algorithme You Only Look Once (YOLO), en particulier YOLOv3, qui est basé sur des réseaux de neurones convolutifs (CNN). Ils ont utilisé la Jet Saliency Map pour ajuster chaque image d'entrée dans le but d'améliorer la détection des bâtiments. Le modèle est entraîné et testé sur un ensemble de données de 4 933 images provenant de 322 aéroports différents en Asie. Les résultats montrent que le modèle amélioré peut détecter efficacement les terminaux de passagers, les tours de contrôle, les bâtiments de fret et les hangars avec une précision de 85 % ou plus, surpassant ainsi le modèle original.

5.1.3 Comparaison entre les différentes approches

Tableau 1 : Comparaison entre les différentes approches pour la détection à partir d'image aérienne

Travaux	Zoraster et al.[11]	Wegner et al. [34].	W. Boonpook et al [12]	N. Pumpong et al. [13]
Méthodes	Canny et Haar	SVM et RF	SegNet	YOLO
Dataset	Inria Aerial Image Labeling Dataset	200 bâtiments	Ensemble de données d'images UAV étiquetées manuellement	Ensemble de données d'images de télédétection
Précision	Une précision moyenne	Une précision significative dans la détection des bâtiments	Plus de 90%	85% ou plus
Limites	Elles nécessitent une extraction manuelle des caractéristiques et ne capturent pas efficacement les structures complexes et variées des bâtiments présents dans les images aériennes.	Elle Peut être sensible au choix du noyau et aux paramètres de régularisation, peut être lent sur de grands ensembles de données. Elle peut être lent pour la prédiction, nécessite plus de mémoire pour stocker de nombreux arbres, peut être difficile à interpréter.	Nécessite un ensemble de données d'apprentissage étiqueté manuellement. Performances affectées par la qualité des images UAV. Non applicable à la détection en temps réel.	Précision affectée par des formes similaires ou des couleurs de fond. Temps de détection plus long que le modèle original.
Avantages	Elles offrent une méthode rapide et relativement simple pour détecter les contours et les motifs structuraux des bâtiments.	Bonne performance pour les données de haute dimension, peut gérer les données non linéaires avec des noyaux, peu de paramètres à ajuster.	Précision élevée dans la détection des bâtiments. Généralisabilité à différents types de bâtiments et de paysages urbains.	Précision de détection de bâtiments élevée. Adaptabilité à des applications pratiques en temps réel.

Dans cette section, on constate que la détection des immeubles dans les images aériennes avec la méthode SegNet utilisée par W. Boonpook et al. a démontré une bonne performance sur l'ensemble de données d'images UAV avec une précision de 90%. Dans l'ensemble, les approches basées sur l'apprentissage profond ont donné de meilleurs résultats que les méthodes classiques de l'apprentissage automatique.

5.2 Détection des immeubles à partir des images satellitaires

Nous présentons dans la section suivante, un ensemble de travaux qui traitent le problème de détection d'images immobilières à partir d'images satellitaires en se basant sur le deep Learning et les méthodes classiques de l'apprentissage automatique.

5.2.1 Les approches basées sur les méthodes classiques de l'apprentissage automatique

Unsalan et al. [14] ont utilisé la théorie des graphes pour détecter des bâtiments en utilisant des images satellitaires IKONOS. Ils ont utilisé une variante de l'algorithme de clustering *k-means* (KMC) pour extraire les maisons et les réseaux de rues possibles en combinant à la fois des caractéristiques spatiales et spectrales. Cette combinaison d'informations améliore les résultats finaux de l'agrégation. À partir de l'agrégation, ils obtiennent une image binaire contenant d'éventuels fragments de réseau routier et des maisons. Ils décomposent ensuite cette image binaire à l'aide d'un algorithme de bulle basé sur la morphologie mathématique binaire. Ils représentent ensuite cette décomposition dans un graphe pour lequel les bulles servent de sommets, tandis que leurs relations de voisinage sont codées en tant qu'arêtes. Le réseau routier est extrait du graphe à l'aide d'un algorithme des chemins les plus courts. Les sommets restants (bulles) sont désignés comme des maisons possibles.

Une autre méthode a été présentée par Soumya k. Das et al. [15] qui ont utilisé les réseaux neuronaux artificiels (ANN) pour la détection des bâtiments sur les images satellitaires WorldView-2. L'ANN a obtenu les meilleurs résultats en termes de détection avec une précision de 92.34%. Ils ont également utilisé les machines à vecteurs de support (SVM) et les forets aléatoires (Random Forest) sur les mêmes images satellitaires et ces techniques ont donné des bons résultats avec une précision respective 93.09% et 91.43%.

5.2.2 Les approches basées sur l'apprentissage profond

Z. Lari et H. Ebadi [16] ont utilisé une méthode de détection automatique des bâtiments basée sur les réseaux neuronaux artificiels (RNA) qui utilise des données structurelles et spectrales des images satellites à haute résolution dans les pays du Moyen-Orient plus

précisément dans la zone urbaine de Kashan en Iran. Le système proposé se compose de trois parties principales qui ont chacune des taches spécifiques à savoir : le traitement initial de l'image et segmentation, l'extraction de caractéristiques et le réseau neuronal déterminant. L'approche proposée peut détecter correctement environ 80% des bâtiments de l'image.

Référence [17] traite le même problème mais cette fois-ci avec le réseau neuronal à convolution (CNN) sur la base du U-Net. Ils ont remplacé l'encodeur du U-Net par ResNet et VGGNet, qui sont tous deux pré-entrainés. Cette approche a donné des meilleurs résultats par rapport au modèle classique du U-Net. Avec une précision de 84,9%, l'U-Net avec VGGNet s'est avéré être le meilleur.

5.2.3 Comparaison entre les différentes approches

Tableau 2 : Comparaison entre les différentes approches pour la détection à partir d'image satellitaire

Travaux	Z. Lari et H. Ebadi [16]	Référence [17]	Unsalan et al. [14]	Soumya k. Das et al. [15]
Méthodes	RNA	U-Net et VGGNet	Théorie des graphes et clustering k-means	RNA, SVM et RF
Dataset	Images dans la zone de Kashan	Les données ouvertes de [35]	Les images satellitaires IKONOS	Les images satellitaires WorldView-2
Précision	80%	84,9%	94,4% à 98,7%	L'algorithme d'ensemble a excellé
Limites	La création d'algorithmes d'extraction ne résout pas complètement le problème. La précision du système en matière de reconnaissance des bâtiments nécessite une évaluation avec des résultats réels.	Difficultés liées à la catégorisation en pixels des photos satellites en raison de leur variabilité. Déséquilibre dans l'ensemble de données d'entrainement, pixels non liés à la construction étant dominants.	Difficultés de détection dues aux petites empreintes et à l'occlusion des arbres. Les formes complexes des maisons et les courbes sinueuses des rues posent des problèmes de détection.	La vitesse de SVM est lente en raison de la taille des vecteurs de support. La précision de l'ANN diminue avec l'augmentation du nombre de nœuds caché. La RF ne peut pas améliorer la précision avec plus de fonctionnalités.
Avantages	Amélioration de l'efficacité d'extraction des bâtiments à l'aide de réseaux de neurones artificiels. Système automatisé de détection de bâtiments avec une grande précision à partir d'images satellites.	U-Net avec VGG surpasse les autres modèles sur les images	Identifie avec succès les zones d'activité humaine, extrait les maisons, les réseaux de rues.	Détection rapide et efficace

Dans cette section, nous constatons que la détection des immeubles dans les images satellitaires avec les méthodes de théorie des graphes et de clustering k-means utilisées par Unsalan et al. Ont démontré la meilleure précision sur les images satellitaires IKONOS avec une précision allant de 94,4% à 98,7%.

6 Conclusion

Dans ce chapitre, on a défini les concepts de l'intelligence artificielle et fait une étude comparative d'un ensemble de travaux scientifiques traitant le problème de détection de bâtiments dans les images aériennes et satellitaire. Nous avons constaté que les approches basées sur l'apprentissage profond donnent des bons résultats que les méthodes classiques de l'apprentissage automatique.

Chapitre II : Conception d'une plateforme de vente immobilière intégrant des apis de détection de bâtiments

1 Introduction

L'immobilier est un secteur en constante évolution, où la technologie joue un rôle de plus en plus crucial dans l'amélioration des processus et de l'expérience client. Dans ce contexte, nous allons mettre en place une conception d'une plateforme de vente immobilière intégrant une API de détection et reconnaissance des images immobilières dont ces diagrammes seront basés sur le diagramme de cas d'utilisation, diagramme de classe et diagramme de séquence. Aussi nous allons aborder en détail toutes les méthodes et les algorithmes que nous avons utilisés pour la reconnaissance et la détection des images immobilières à partir des images simples et des images aériennes dans notre programme.

2 Objectif de l'application

Au Mali, avant l'émergence des agences immobilières en ligne, toutes les transactions se faisaient par l'intermédiaire d'investisseurs visitant des agences immobilières physiques. Ce processus entraînait une perte de temps et d'argent, en raison du manque de moyens de communication et d'annonce, à l'exception des médias traditionnels comme les journaux ou les petites affiches. Avec l'utilisation croissante d'internet, le Mali commence à adopter l'idée du commerce en ligne, ce qui a permis de surmonter les difficultés mentionnées ci-dessus. Cependant, malgré cette ouverture, les sites web de commerce électronique restent limités en nombre et en qualité, offrant une navigation et une recherche d'informations dans un cadre restreint.

Pour répondre à ce besoin, nous allons concevoir une application web de vente immobilière avec une intelligence artificielle intégrée. Cette application permettra aux investisseurs maliens d'acheter, de vendre et de louer des biens immobiliers (maisons, appartements, etc.) en ligne, tout en offrant une expérience de navigation simple et rapide.

La méthode utilisée dans ce projet est l'eXtreme Programming (XP), une méthode de gestion de projet qui applique de manière intensive les principes du développement agile, en se

concentrant sur les besoins du client, en mettant en place un développement itératif et une intégration continue.

2.1 Identification des acteurs

Client : Utilisateur final de la plateforme, chercheur de biens immobiliers.

Fournisseurs : Professionnel de l'immobilier gérant les annonces et les biens.

Administrateur (Agent immobilier) : Gestionnaire du système, responsable de la gestion des comptes et des annonces.

2.2 Identification des cas d'utilisations

Un cas d'utilisation représente un ensemble de séquences d'interactions entre le système et ses acteurs. Les cas d'utilisation permettent de décrire ce que le système devra faire, sans spécifier comment il le fera. Chaque acteur identifié précédemment effectue un certain nombre de tâches, résumées dans le tableau suivant :

Acteurs	**Tâches**
Client	**T1** : Naviguer Dans Le Site **T3** : Rechercher Des Logement **T4** : Demander Un Logement
Fournisseur	**T1** : Se Connecter **T2** : Gérer Les Annonce **T3** : Ajouter Des Biens Immobilier **T4** : Se Déconnecter
Administrateur (Agent immobilier)	**T1** : Se Connecter **T2** : Gérer Les Logements **T3** : Gérer Les Comptes Utilisateurs **T4** : Exploiter les statistiques **T5** : Se Déconnecter

Tableau 3 : Tache de chaque acteur

3 Diagramme de cas d'utilisation générale

Le diagramme de cas d'utilisation générale représentera les interactions entre les acteurs externes et le système, en mettant en évidence les différentes fonctionnalités offertes par la plateforme immobilière.

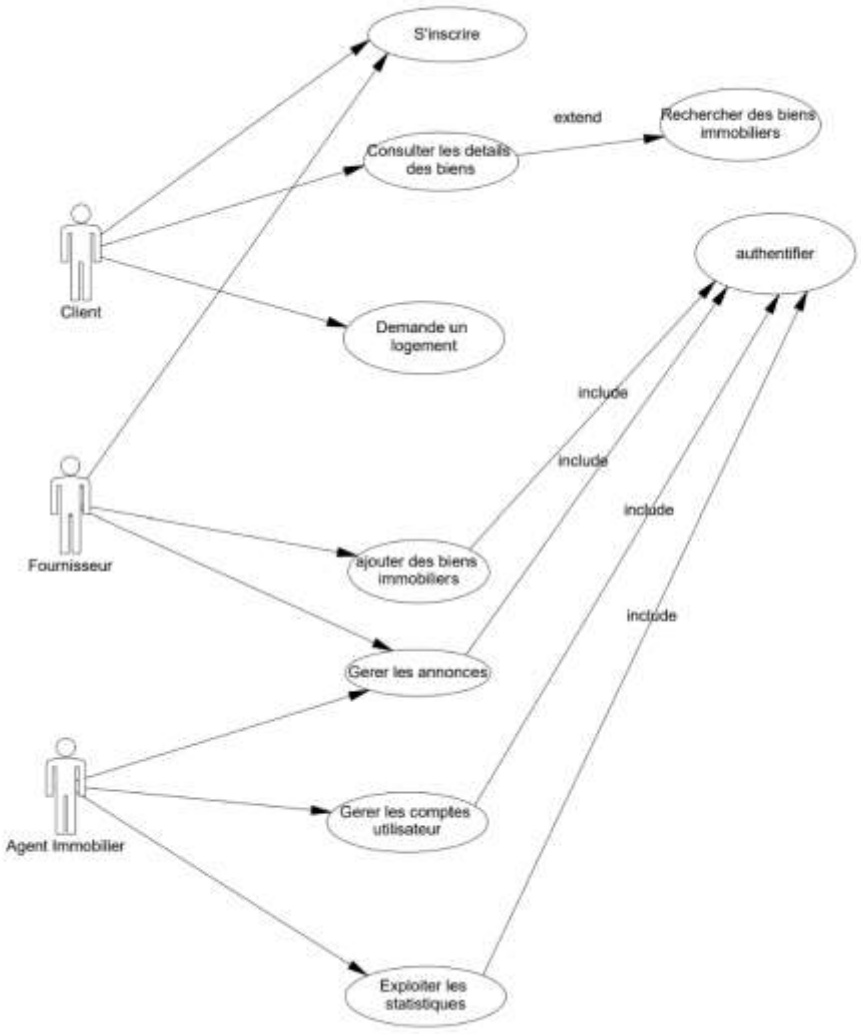


Figure 3-1 : diagramme de cas d'utilisation

4 Diagramme de classe de la plateforme de vente immobilier

Le diagramme de classe de la plateforme de vente immobilière est conçu pour représenter les différentes entités et leurs relations dans le système. Il comprendra des classes. Et des attributs pour gérer les opérations courantes de la plateforme.

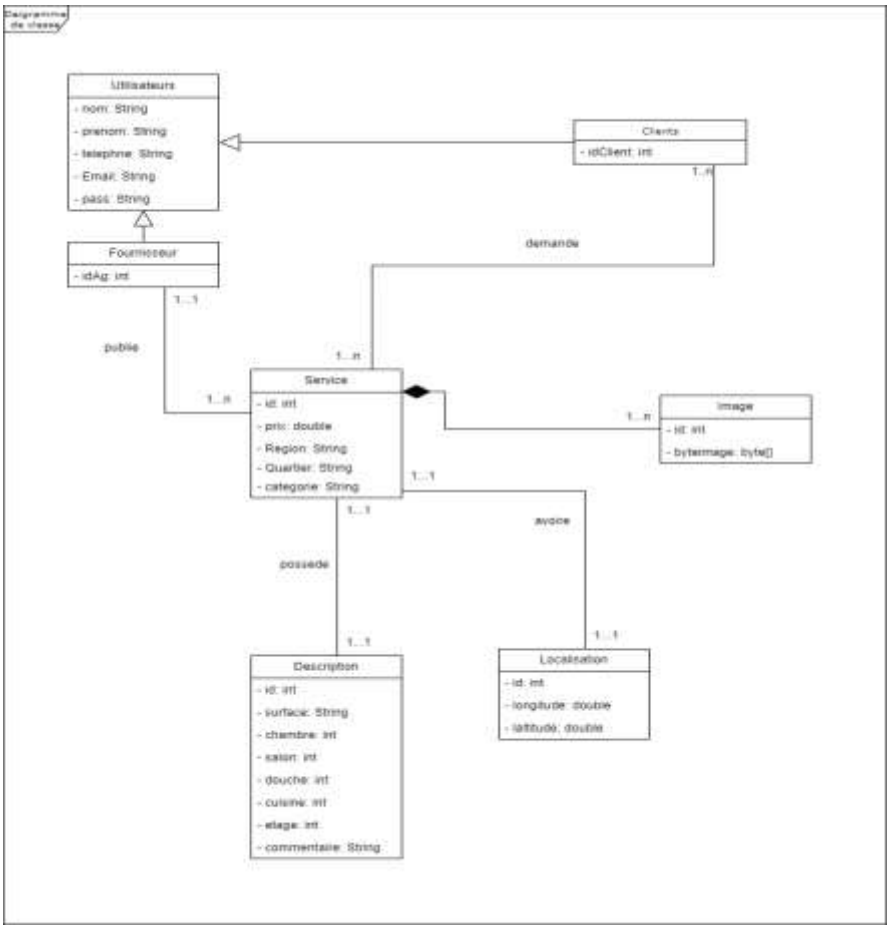


Figure 4-1 : Diagramme de classe

5 Diagrammes de séquence de quelques cas d'utilisation

Le diagramme dynamique mettra en lumière les interactions entre les différents objets du système lors de l'exécution des cas d'utilisation. Il inclura des séquences d'actions entre les acteurs et les objets du système, montrant comment les opérations sont effectuées et les résultats obtenus.

Vu le nombre important de cas d'utilisations que nous avons, nous limitons à représenter les cas d'utilisations suivants :

Inscription des fournisseurs

Gestion des logements

Consultation des logements

Pour chacun des cas d'utilisations que nous venons de citer, nous allons lui représenter son diagramme de séquence

5.1 Inscription des Fournisseurs

Cette section de l'application web sert de plateforme en ligne pour les services immobiliers. Elle offre aux fournisseurs diverses options, notamment la possibilité de créer un compte.

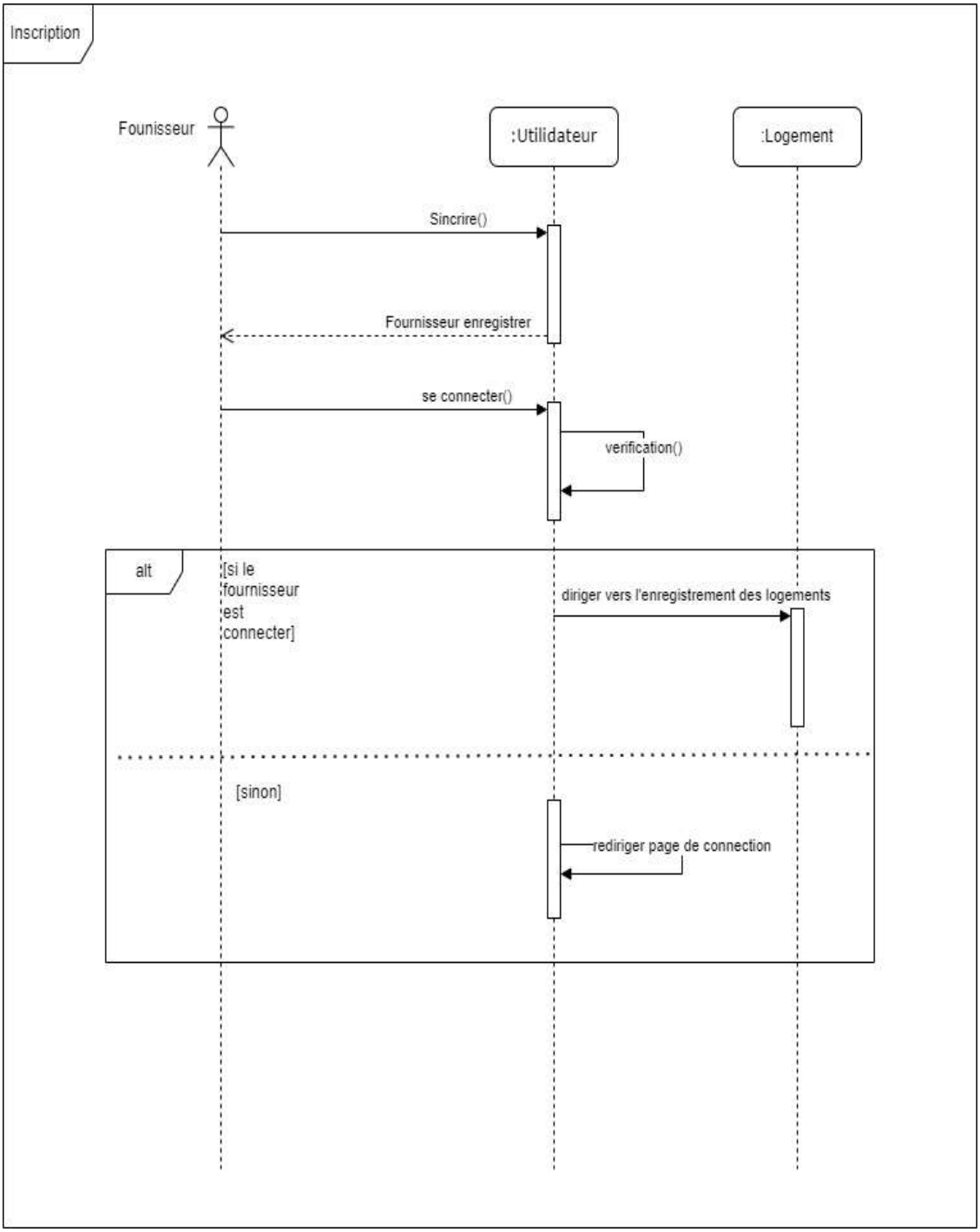


Figure 5-1 : Inscription des visiteurs

5.2 Gestion des logements

Le diagramme ci-dessous montre les étapes impliquées dans la gestion des logements, de l'ajout d'un niveau logement à la suppression d'un logement.

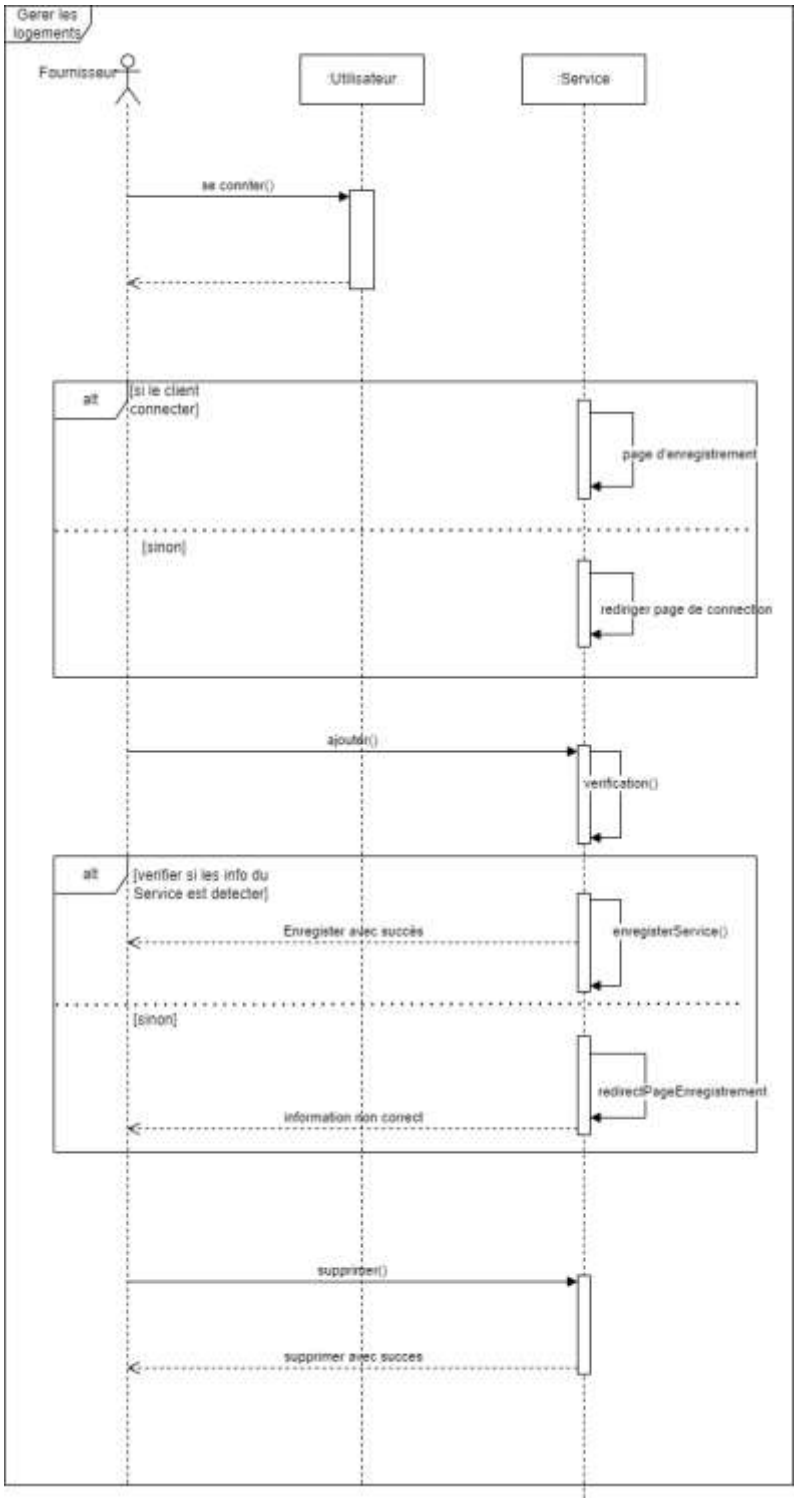


Figure 5-2 : Gestion de logement

5.3 Consultation des logements

Ce diagramme montre les étapes impliquées dans la recherche et la postulation pour un logement, depuis la consultation initiale avec un agent immobilier jusqu'à la soumission d'une demande de location.

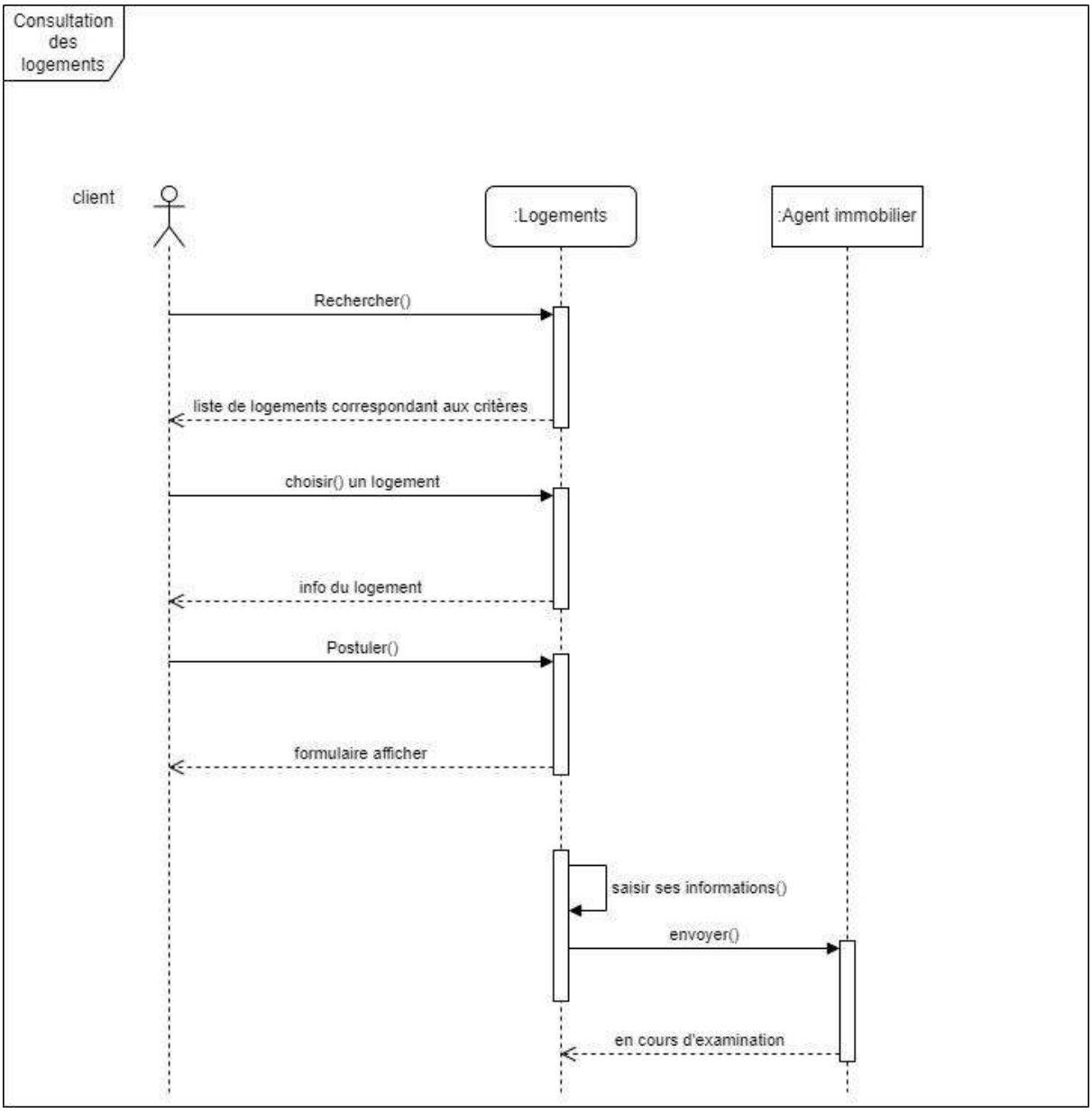


Figure 5-3 : Consultation des logements

6 Diagramme de collaboration entre la plateforme immobilier et API de reconnaissance d'image immobilier et détection des bâtiments

Ce diagramme représente un système de gestion et de reconnaissance d'une image immobilière. Ce système est conçu pour automatiser la reconnaissance des images immobilière et la détection des bâtiments à partir de l'adresse fournie par le fournisseur pour voir s'il existe des bâtiments ou pas.

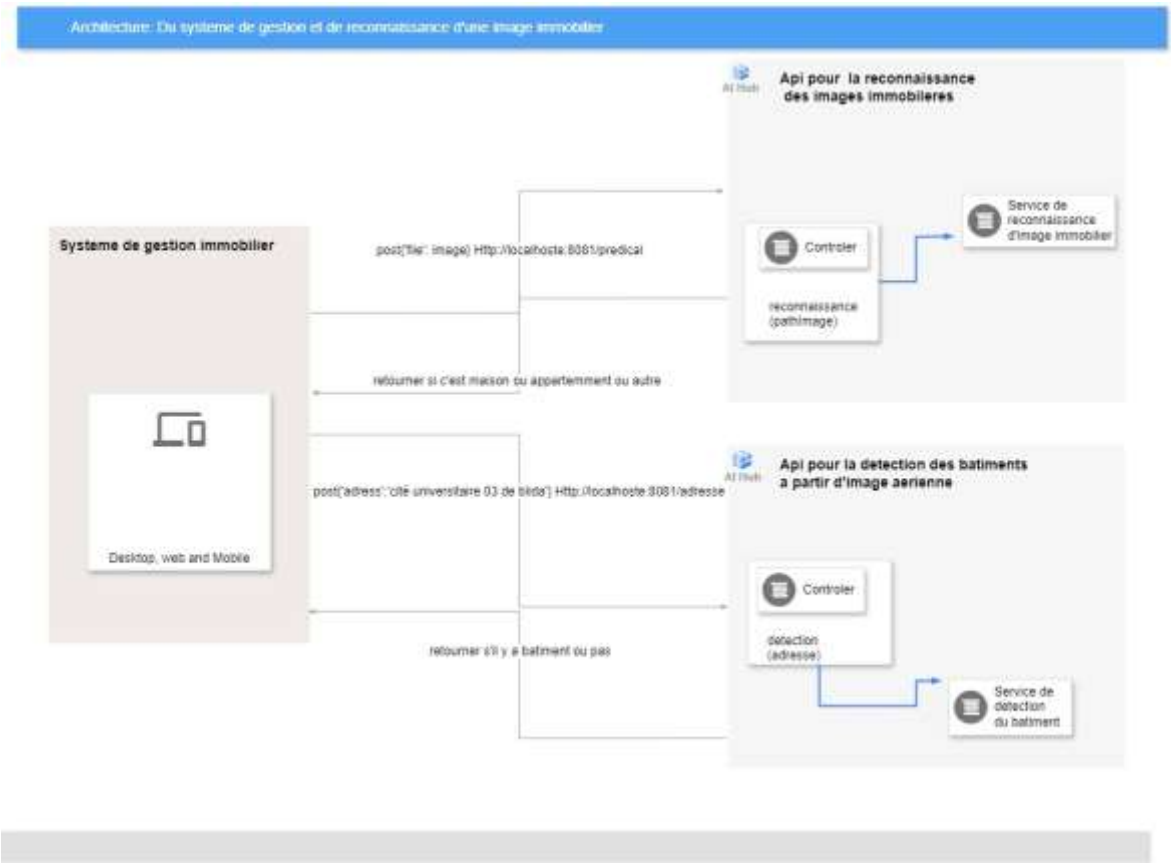


Figure 6-1 : Architecture du système de gestion et de reconnaissance d'image immobilier

7 API de reconnaissance d'immeubles à partir d'images simple

À travers la revue des articles discutés précédemment, on a remarqué l'importance du traitement d'image, qui est actuellement bien effectué par un être humain, mais qui n'est pas assez adéquat lorsqu'il est effectué par des méthodes d'intelligence artificielle. Dans ce contexte L'architecture VGG16 est un modèle de réseau neuronal convolutionnel (CNN) qui permet d'extraire des caractéristiques complexes à partir d'images et il est aussi utilisé pour la classification d'images, c'est-à-dire pour prédire la catégorie à laquelle appartient une image donnée parmi un ensemble de catégories prédéfinies. Comme le montre la figure (en supposant

qu'il soit déjà formé) qui prendra les images sur les bâtiments (maison ou appartement) et en sortie classera les images comme contenant une maison ou un appartement ou bien d'autres images. Le diagramme proposé, comme indiqué ci-dessous, peut être lié à un organigramme avec lequel chacune représente un composant.

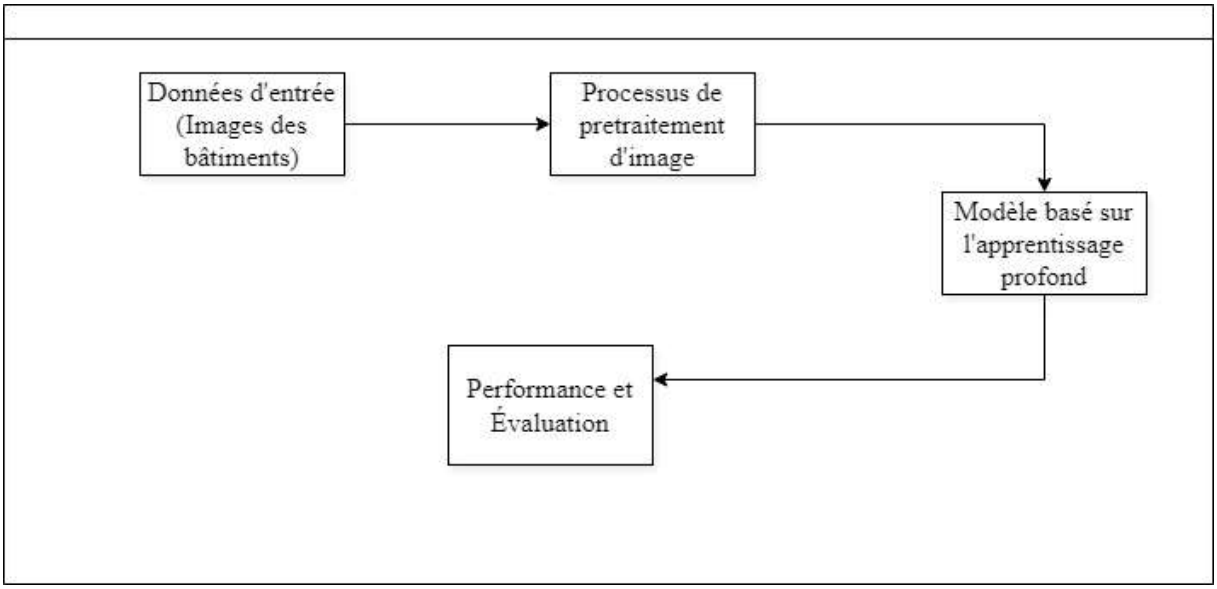


Figure 7-1 : Principales phases impliquées dans un système de reconnaissance d'image immobilier

7.1 Description du système de reconnaissance d'images immobilières

Cette phase repose sur le prétraitement et la classification, en utilisant par la suite les modèles prédéfinis suivants : VGG16 et ResNet, qui seront détaillés dans les sections suivantes :

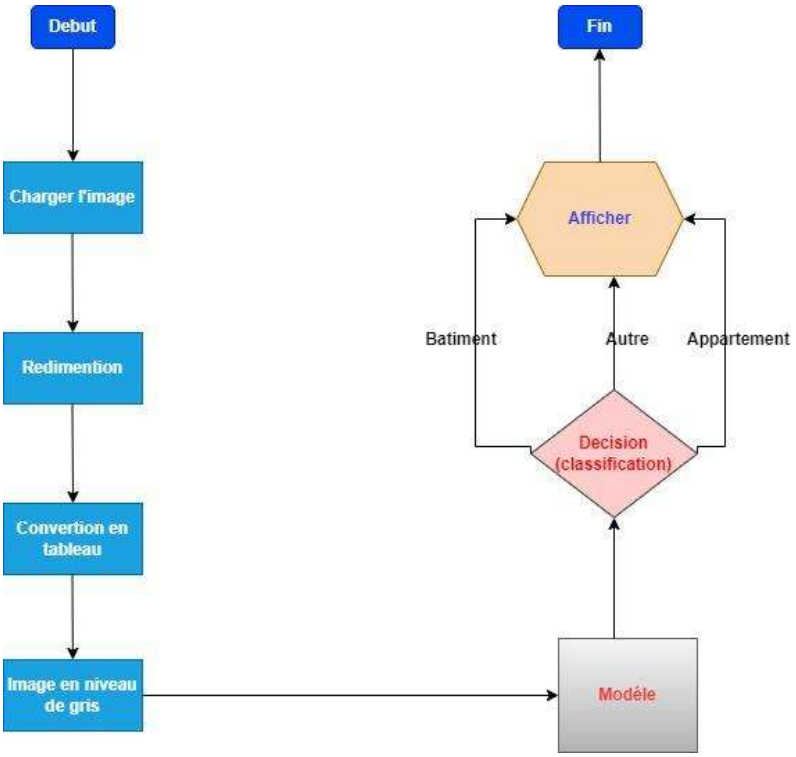


Figure 7-2 : schéma du plan de travail

7.1.1 Dataset

Nous avons utilisé un jeu de données qui provient de trois sources distinctes : Tout d'abord, nous avons le premier ensemble de données qui associe les prix des maisons aux images de Southern California (SoCal). Ce jeu de données se compose de 7 colonnes et de plus de 15 000 lignes. Les colonnes incluent: image_id, street, city, n_city, bed et bath **[18]**. Pour obtenir les images nécessaires, nous avons extrait celles-ci du jeu de données, totalisant environ 15 474 images.

Ensuite, nous disposons du deuxième ensemble de données qui comprend un ensemble de photos d'appartements réparties en cinq catégories :

- Bâtiments (957 images)
- Entrées de bâtiments (1076 images)
- Plans d'appartement (1740 images)
- Intérieurs (1806 images)
- Vues des fenêtres (894 images) **[19]**.

Enfin, le troisième ensemble de données a été constitué par no soins. Il contient diverses images dans le but de reconnaître si une image représente une maison, un appartement ou tout autre

type d'image, comme illustré dans la Figure 7-3. Nous avons utilisé les modèles prédéfinis suivants : VGG16, ResNet.

7.1.2 Prétraitement

La première étape du système de reconnaissance d'images immobilières est le traitement de l'image, qui implique l'utilisation de diverses fonctions et méthodes pour manipuler l'image dans la forme souhaitée pour la solution donnée. Une fois les images lues au format RVB, avec des canaux séparés pour le Rouge, le Vert et le Bleu, les étapes suivantes sont mises en œuvre :

- **Chargement des images**

  Pour amorcer le processus, nous avons importé les images dans notre programme informatique. Cela a été réalisé en utilisant les noms de fichiers et les chemins d'accès pour localiser les images stockées sur notre ordinateur. Ensuite, en raison de la moindre puissance de notre ordinateur, nous avons réduit le nombre d'images dans chaque dossier (maison, appartement et autres) à 200 pour garantir une exécution rapide et efficace.

- **Redimensionnement**

  Pour optimiser l'entraînement de notre modèle, il est avantageux de s'assurer que toutes les images sont de taille uniforme. Ainsi, nous redimensionnons chaque image pour avoir des dimensions identiques, soit une largeur et une hauteur de 224 pixels. Cette standardisation permet une formation plus efficace.

- **Conversion en tableau numérique**

  Les ordinateurs comprennent mieux les nombres que les images. C'est pourquoi nous convertissons chaque image en un tableau de nombres, où chaque nombre représente un pixel de l'image. Ce processus permet à l'ordinateur de manipuler et de comprendre les images plus facilement.

- **Normalisation**

  Les valeurs des pixels dans une image peuvent varier de 0 à 255, selon la luminosité de chaque pixel. Pour faciliter le travail du modèle, nous normalisons ces valeurs pour qu'elles se situent toutes dans une plage de 0 à 1. Pour ce faire, nous divisons simplement chaque valeur de pixel par 255, ce qui garantit que toutes les valeurs se trouvent dans la même plage.

7.1.3 Classification

Pour catégoriser les images, nous avons utilisé une approche algorithmique supervisée. Ces images d'entrée ont déjà subi un prétraitement, comme expliqué précédemment. Ils sont présentés sous la forme d'un tableau matriciel, où chaque propriété est représentée par une matrice. Par exemple, [1 0 0] signifie des maisons, [0 2 0] représente des appartements et [0 0 3] correspond à d'autres images. Par la suite, nous avons introduit ces données entraînées dans le modèle VGG16 et Restnet pour déterminer si l'image d'entrée est dans la catégorie d'une maison, d'un appartement ou d'une autre image.

7.1.4 Phase de reconnaissance

7.1.4.1 Reconnaissance avec le modèle VGG16

7.1.4.1.1 Fonctionnement du modèle

Le processus de fonctionnement du modèle est organisé de la manière suivante : Après le chargement et le prétraitement des images, elles sont assemblées en un ensemble de données unique, et les étiquettes sont transformées en format one-hot. Les données sont ensuite réparties en ensembles d'apprentissage et de test grâce à la fonction train_test_split.

L'architecture VGG16 pré-entraînée est importée via VGG16 de tensorflow.keras.applications, avec des poids qui ont été initialement formés sur ImageNet. Au-dessus de cette architecture, des couches fully connected sont ajoutées pour adapter le modèle à la classification d'images. Une couche Flatten est utilisée pour aplatir la sortie de VGG16, suivie d'une couche Dense de 512 neurones avec une fonction d'activation ReLU, et une couche Dense de sortie avec une fonction d'activation softmax pour obtenir les probabilités de classe. Par la suite, le modèle est compilé avec l'optimiseur Adam et la fonction de perte categorical_crossentropy. Après la compilation, le modèle est formé sur les données d'apprentissage via model.fit. Enfin, le modèle est évalué sur les données de test via model.evaluate, affichant la perte et la précision.

7.1.4.2 Reconnaissance Avec le modèle ResNet

7.1.4.2.1 Fonctionnement du modèle

Le processus de reconnaissance des images avec le modèle ResNet commence par l'importation du modèle à partir de la bibliothèque TensorFlow.keras.applications. ResNet est

un réseau de neurones convolutifs (CNN) pré-entraîné couramment utilisé pour la classification d'images en raison de sa capacité à apprendre des représentations complexes.

Ensuite, les couches fully connected du modèle ResNet sont supprimées en utilisant l'argument include_top=False. Cela permet d'exclure les couches de classification finales du modèle, ce qui permet de les adapter à une nouvelle tâche de classification spécifique.

Les poids des couches de base du modèle ResNet sont gelés pour empêcher qu'ils ne soient mis à jour pendant l'entraînement. Cela est particulièrement utile lorsqu'on utilise des modèles pré-entraînés pour le transfert de connaissances.

Des couches fully connected personnalisées sont ajoutées au-dessus des couches convolutionnelles de ResNet pour adapter le modèle à la tâche spécifique de classification des images immobilières. Cela inclut l'ajout d'une couche Flatten pour aplatir la sortie des couches convolutionnelles en un vecteur, suivie d'une ou plusieurs couches Dense pour effectuer la classification.

En combinant les couches de base de ResNet avec les couches personnalisées, un nouveau modèle est construit à l'aide de l'API fonctionnelle de Keras. Cela crée un modèle séquentiel où les sorties d'une couche sont les entrées de la couche suivante.

Le modèle est compilé en spécifiant l'optimiseur (dans ce cas, Adam) et la fonction de perte (cross-entropy catégorique) à utiliser pendant l'entraînement. L'exactitude est également spécifiée comme métrique pour suivre les performances du modèle.

Le modèle est ensuite entraîné sur les données d'entraînement en utilisant la méthode fit(). Pendant l'entraînement, les poids des couches personnalisées sont ajustés pour minimiser la perte sur les données d'entraînement.

Une fois l'entraînement terminé, le modèle est évalué sur les données de test pour estimer ses performances en termes de perte et d'exactitude.

Ce processus permet au modèle ResNet d'apprendre à reconnaître et classer les images immobilières avec précision en utilisant des représentations apprises à partir de données pré-existantes.

8 API de détection d'immeuble à partir d'image aérienne et satellitaire

8.1 Description du système de détection d'immeuble à partir d'image aérienne et satellitaire

Cette phase repose sur le prétraitement et la détection des bâtiments, en utilisant par la suite les modèles prédéfinis et non prédéfinis suivants : YOLOv4 et U_Net, qui seront détaillés dans les sections suivantes :

8.1.1 Dataset du modèle YOLO

Pour la détection des bâtiments à partir d'images aériennes, nous avons utilisé une base de données contenant un ensemble d'images aériennes annotées. Ces annotations comprennent les emplacements et les contours des bâtiments dans chaque image, ce qui permet au modèle de détecter et de localiser les bâtiments avec précision.

Fractionnement de l'ensemble de données : ensemble d'entraînement qui contient 13 528 images et chaque image a son annotation, l'ensemble de validation qui contient 1934 et l'ensemble de test qui contient 967.[33]

8.1.2 Prétraitement

La phase initiale du prétraitement consiste à ajuster automatiquement l'orientation de l'image pour garantir son alignement optimal. Ensuite, chaque image est redimensionnée à une taille standard de 640 x 640 pixels. Pour enrichir la diversité des données, diverses transformations sont appliquées à chaque image lors de l'étape suivante. De cette manière, chaque image contribue à la création de deux exemples d'entraînement distincts, renforçant ainsi la robustesse du modèle.[33]

Pour accroître la variabilité des données d'entraînement, les images subissent des retournements horizontaux et verticaux, ainsi que des rotations de 90° dans le sens des aiguilles d'une montre et dans le sens inverse, introduisant ainsi une gamme de perspectives.

Afin de cadrer les objets d'intérêt de manière optimale, les images sont recadrées avec un niveau de zoom minimal de 0 % et un niveau de zoom maximal de 20 %. De plus, pour simuler différents angles de vue, les images sont inclinées dans une plage allant de -12° à +12°.

Pour introduire une légère déformation aux images, des forces de cisaillement horizontales et verticales de ±2° sont appliquées. Cela permet de diversifier les caractéristiques des données en appliquant une conversion en niveaux de gris à 10 % de chaque image.

La coloration des images est ajustée en modifiant la teinte dans une plage de -20° à +20°, offrant ainsi une plus grande variété visuelle. De même, la saturation des couleurs peut être ajustée dans une plage de -20 % à +20 % pour varier l'intensité des couleurs.

Pour simuler différentes conditions d'éclairage, la luminosité des images peut être ajustée dans une plage de -20 % à +20 %. De plus, pour simuler divers niveaux d'exposition, l'exposition de l'image peut être modifiée dans une plage de -15 % à +15 %.

Afin de créer un effet plus doux et de réduire le bruit, un flou maximal de 0,5 pixel est appliqué pour atténuer les détails indésirables.

Enfin, pour mettre en évidence des zones spécifiques de l'image, celles-ci sont divisées en cinq cases, chacune représentant 2 % de la taille totale de l'image.

8.1.3 Dataset du modèle U_Net

L'ensemble de données comprend des images bi-temporelles capturées dans quatre régions distinctes de l'Égypte : New Mansoura, El Galala City, New Cairo et New Thebes. Qui est constitué de trois dossier l'image B qui contient 6091 images, l'image A qui contient 6091 images et le label qui contient la segmentation des bâtiments de l'image B[20] :

8.1.4 Prétraitement

- **Gestion des fichiers**

  Nous avions deux dossiers à notre disposition : un pour les images et un autre pour les labels. Les labels sont des images qui identifient les objets comme dans notre cas on a le bâtiment qui est colorer en blanc et les autres en noire qui peut correspondre à des arbres ou voiture ou les zones spécifiques dans chaque image.

- **Chargement et préparation des données**

  Nous avons commencé par lister tous les fichiers d'images dans le dossier. Pour chaque image, nous avons trouvé l'étiquette correspondante. Nous avons ouvert les images et les étiquettes, puis nous les avons redimensionnées pour qu'elles soient toutes de la même taille. Pour s'assurer que les images soient uniformes et compatibles avec le modèle.

  Nous avons converti ces images en un format numérique que l'ordinateur pourra traiter. Cela signifiait que chaque image était convertie en un tableau de chiffres. Nous avons normalisé les valeurs des pixels des images et des étiquettes pour qu'elles soient

comprises entre 0 et 1. La normalisation était essentielle car elle permet au modèle d'apprendre plus rapidement et plus efficacement en évitant les problèmes liés à des valeurs de pixels trop élevées ou trop basses.

- **Séparation des données en ensembles d'entraînement et de test**

  Une fois que les images et les étiquettes étaient prêtes, nous les avons divisées en deux ensembles. L'ensemble d'entraînement qui représente la majorité des données (par exemple, 80%) et était utilisé pour entraîner le modèle. C'est avec cet ensemble que le modèle a appris à identifier les objets dans les images.

  L'ensemble de tests qui représente aussi le reste des données (20%). Cet ensemble a été utilisé pour évaluer les performances du modèle une fois qu'il a été formé. En d'autres termes, il a permis de vérifier si le modèle avait bien appris à identifier les objets.

8.1.5 Phase de détection

8.1.5.1 Détection basée sur le modèle YOLO

8.1.5.1.1 Fonctionnement du modèle

Le fonctionnement de YOLO peut être résumé de la manière suivante : L'image en entrée du modèle est divisée en grilles de cellules (SxS) dont chaque grille de cellule est responsable de la prédiction d'un seul objet présent dans l'image ainsi que les boites englobantes (boundry boxes) et les probabilités de classes. Le résumé du fonctionnement est présenté dans la figure.

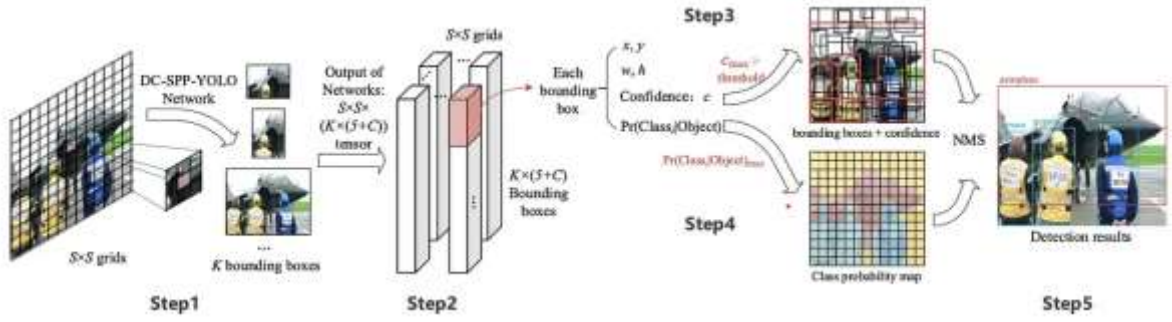


Figure 8-1 : La distribution des grilles de cellules dans l'image pour déterminer les objets

Dans notre cas, nous avons utilisé un modèle pré-entraîné appelé Darknet avec la version YOLOv4-tiny. Ce modèle, YOLOv4-tiny, est un réseau de neurones convolutionnels (CNN) spécialement conçu pour la détection d'objets en temps réel. Pour notre étude, nous avons adapté

ce modèle en l'entraînant sur un ensemble de données personnalisé afin de détecter des bâtiments dans des images aériennes.

Pour La classe de détection des bâtiments, On a le vecteur suivant :

Pc	Bx	By	Bh	Bw	C

Dont les éléments sont :

Pc : probabilité qu'il existe un objet dans la cellule.

Bx, By : les coordonnées normalisées entre 0 et 1 représentant le centre de la cellule.

Bw, Bh : la largeur et la hauteur normalisées de la cellule sélectionnée.

C (Building) : représente la classe des bâtiments.

La figure illustre la détection avec YOLOv4.

Figure 8-2 : Détection par YOLOv4

Afin d'identifier plusieurs objets, le modèle YOLO utilise le processus NMS pour éliminer les boîtes en double avec un faible niveau de confiance. YOLO est un modèle polyvalent et complet spécialement créé pour la détection d'objets.

Profitant de sa précision reconnue, nous avons spécifiquement sélectionné un scénario unique pour en récolter les bénéfices.

Dans la mise en œuvre de ce modèle, nous utilisons une classe unique appelée classe « Building » pour identifier et détecter exclusivement les bâtiments.

8.1.5.2 Détection basée sur le modèle U_Net

8.1.5.2.1 Fonctionnement du modèle :

Tous les réseaux développés ont été créés à l'aide de la bibliothèque Keras et du framework Tensorflow. Keras est une bibliothèque open source écrite en Python. Cette bibliothèque contient de nombreuses implémentations des blocs structurels des réseaux neuronaux tels que les couches, les fonctions d'activation, les optimiseurs et des outils prêts à l'emploi pour le prétraitement des images et des données textuelles[21].

De plus, cette bibliothèque permet d'entraîner et de tester les réseaux sur GPU. Comme le montre la figure ci-dessous, le réseau U-Net se compose de deux parties : un encodeur (à gauche) et un décodeur (à droite). L'encodeur est un réseau neuronal qui possède une architecture CNN typique comprenant quatre blocs. Chacun de ces blocs se compose de deux couches convolutives avec un filtre 3 × 3, d'une fonction d'activation ReLU appliquée à chacune d'elles et d'une opération de maxpooling avec un filtre 2 × 2 et une étape 2. Le décodeur contient le même nombre de blocs. Chaque bloc du décodeur consiste en une opération de suréchantillonnage avec un filtre 2 × 2 combinés à une carte correspondante de caractéristiques provenant du codeur, deux couches convolutives avec un filtre 3 × 3 et une fonction d'activation ReLU appliquée à chacune d'entre elles. La dernière couche du réseau effectue une opération de convolution avec un filtre 1 × 1, qui associe chaque pixel à une classe spécifique. Le réseau comporte donc 19 couches convolutives, 18 fonctions d'activation ReLU, 4 opérations de maxpooling, 4 opérations de suréchantillonnage et 4 opérations de fusion.

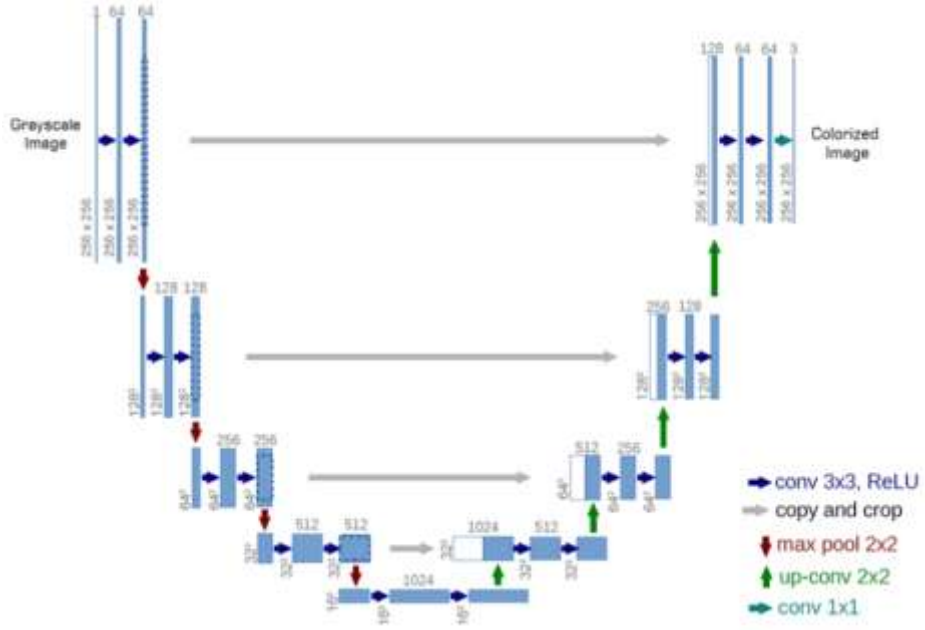


Figure 8-3 : architecture du modèle U-Net [9]

Dans notre cas, nous avons utilisé un modèle appelé U-Net. Ce modèle est un réseau de neurones convolutionnels (CNN) spécialement conçu pour la segmentation des objets. Pour notre étude, nous avons adapté ce modèle en l'entraînant sur un ensemble de données personnalisé afin de détecter des bâtiments dans des images aériennes pendant la détection il segmentation pour identifier un batiment qui colorer en blanc.

Voici une figure d'illustration de notre modèle :

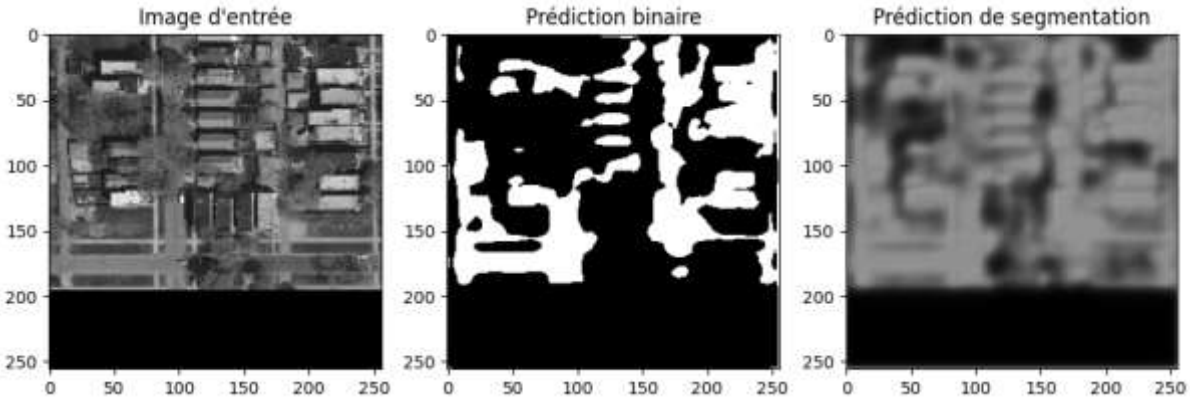


Figure 8-4 : Détection par U_Net

9 Résultats et discussion

9.1 Évaluation matrices

Dans un test de classification, nous classons les éléments selon deux classes : positive ou négative, le résultat peut être correct ou incorrect. Pour cela, nous distinguons quatre combinaisons de résultats possibles :

— Vrais positifs « TP » (True Positive) : un résultat est dit vrai positif lorsqu'un élément positif est correctement classé.

— Faux positifs « FP » (False Positive) : un résultat est dit faux positif lorsqu'un élément est classé positif, alors qu'il ne l'est pas.

— Vrais négatifs « TN » (True Negative) : un résultat est dit vrai négatif lorsqu'un élément négatif est correctement classé.

— Faux négatifs « FN » (False Negative) : un résultat est dit faux négatif lorsqu'un élément est classé négatif, alors qu'il est positif.

Nous présentons dans la suite les mesures les plus utilisées dans le domaine de RI :

(a) La précision : La proportion de prédictions correctes parmi les points que l'on a prédits positifs

$$précision = True\ positive\ /\ True\ positive + False\ positive$$

(b) Le rappel : (sensibilité) C'est le taux de vrais positifs, c'est-à-dire la proportion de positifs que l'on a correctement identifiés.

$$rappel = True\ positive\ /\ True\ positive + False\ négative$$

(c) F1-score : Pour évaluer un compromis entre rappel et précision, on peut calculer la "F-mesure", qui est leur moyenne harmonique.

$$\text{F1-score} = 2 * Précision + Recall\ /\ Précision * Recall$$

9.2 Précision de chaque modèle

- **Pour le modèle VGG16 :** au début de notre entraînement, nous avons observé une précision de 80,83 % sur l'ensemble de la validation lors de l'époque 1. Cependant, au fur et à mesure de la progression de l'entraînement, cette précision a régulièrement augmenté pour atteindre un remarquable 97,50 % à partir de l'époque 4. Ce niveau élevé

de précision a été maintenu tout au long du reste du processus d'entraînement. De manière similaire, la perte sur l'ensemble de validation a diminué progressivement au fil du temps, atteignant son niveau le plus bas de 0,0438 à l'époque 10.

Nos résultats démontrent que notre modèle a atteint une convergence, comme en témoigne la stabilisation de la précision et de la perte après un nombre spécifique d'époques. Cette convergence est corroborée par notre évaluation finale sur l'ensemble des tests, qui révèle une précision de 98,06 % et une perte de 0,0405.

- **Pour le modèle ResNet :** La précision de notre modèle sur l'ensemble de validation commence à 53.33% à l'Epoch 1, atteignant un sommet de 85.83% à l'Epoch 4, mais subissant une légère baisse par la suite. En ce qui concerne la perte sur cet ensemble, elle diminue initialement avant de connaître une légère augmentation vers la fin de l'apprentissage.

  En ce qui concerne la convergence du modèle, il semble que notre modèle n'ait pas atteint une convergence complète sur l'ensemble de validation. Cette conclusion découle de la baisse de précision et de l'augmentation de la perte après l'Epoch 4. Alors que la précision sur l'ensemble d'entraînement continue d'augmenter, celle sur l'ensemble de validation stagne ou diminue légèrement, ce qui pourrait indiquer un surapprentissage potentiel.

  Enfin, l'évaluation finale sur l'ensemble de test révèle une précision de 69.96% avec une perte de 0.8477, ce qui est cohérent avec les performances observées sur l'ensemble de validation.

- **Pour le modèle Yolo** : En termes de performances du modèle YOLO, la précision du modèle s'élève à 71 %, accompagnée d'un taux de rappel de 49 % et d'un score F1 de 58 %. La métrique de précision mesure le pourcentage d'instances correctement identifiées sur toutes les instances détectées par le modèle.

  Pour évaluer les performances de détection, un seuil de confiance de 0,25 a été utilisé. Parmi les 123 443 détections, le modèle a réussi à reconnaître 7 478 bâtiments avec un taux de précision louable de 58,69 %.

  La détermination de la précision et du rappel repose sur la comparaison entre les détections et les vérités terrain. L'évaluation de l'intersection sur l'union (IoU) sert de mesure.

La mesure connue sous le nom d'intersection sur union quantifie le degré de chevauchement entre les boîtes de détection et les boîtes de vérité terrain. En utilisant un seuil IoU de 50 %, le modèle a démontré une précision moyenne de 58,69 %.

- **Pour le modèle U-Net :**

  Dans notre analyse, l'Intersection over Union (IoU) moyen est d'environ 0,24, indiquant une superposition partielle entre les masques prédits et réels. La précision moyenne des prédictions est également de 0,24, ce qui signifie que seulement environ 24% des prédictions positives sont correctes. En revanche, le rappel moyen est élevé, environ 0,88, ce qui indique une capacité élevée à identifier les vrais positifs. Le F1-Score moyen est d'environ 0,36, reflétant une balance entre précision et rappel. Enfin, l'exactitude moyenne est d'environ 0,34, démontrant la précision globale du modèle. Ces valeurs fournissent une évaluation complète des performances de notre modèle de segmentation d'images.

9.3 Matrice de confusion de chaque modèle

Chaque colonne de la matrice représente le nombre d'occurrences d'une classe estimée, tandis que chaque ligne représente le nombre d'occurrences d'une classe réelle (ou de référence). Un des intérêts de la matrice de confusion est qu'elle montre rapidement si un système de classification parvient à classifier correctement.

- **Pour le modèle VGG16 :**

  Dans cette matrice ci-dessous, les classes réelles sont représentées sur l'axe vertical et comprennent "maison", "building" et "autre". Les classes prédites par notre modèle sont représentées sur l'axe horizontal et sont également "maison", "building" et "autre".

  Les cellules en bleu foncé sur la diagonale représentent les prédictions correctes de notre modèle. Par exemple, notre modèle a correctement prédit 43 maisons, 36 buildings, et 41 autres objets.

  Les cellules hors de cette diagonale représentent les erreurs de classification. Dans notre cas, toutes ces cellules contiennent le chiffre zéro, ce qui signifie que notre modèle n'a commis aucune erreur de classification pour cet ensemble de données.

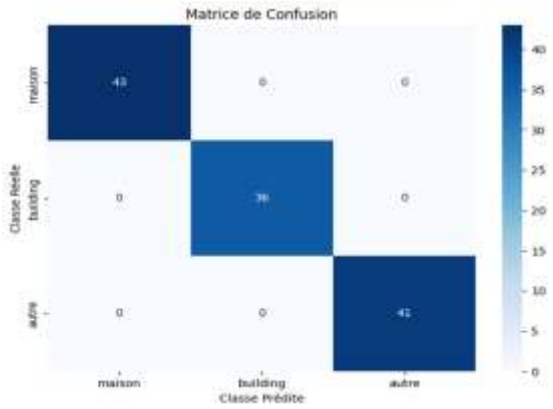


Figure 9-1 : Matrice de confusion du modèle VGG16

Pour le modèle ResNet

Comme expliqué ci-dessus, on a juste montré les détails de cette figure ci-dessous :

- Les étiquettes prédites par le modèle sont sur l'axe des x ("maison", "building", "autre").
- Les vraies étiquettes sont sur l'axe des y.
- Chaque cellule de la matrice montre le nombre d'observations pour chaque combinaison d'étiquette prédite et réelle.
- La couleur des cellules varie du blanc au bleu foncé, indiquant le nombre d'observations. Le bleu foncé représente un nombre plus élevé d'observations.

Par exemple, il y a 41 observations où à la fois l'étiquette prédite et réelle est "maison". Cette matrice est utilisée pour comprendre où le modèle fait des erreurs et comment améliorer sa performance.

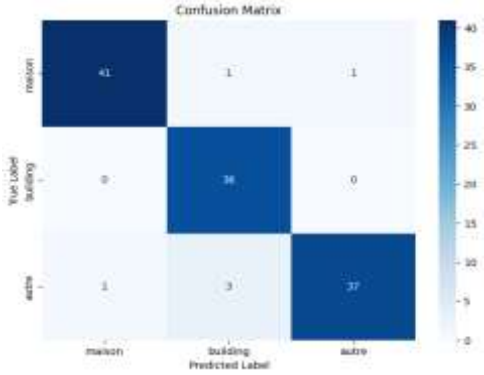


Figure 9-2 : Matrice de confusion du modèle ResNet

- **Pour le model U_Net**

  Les informations suivantes sont présentées dans la matrice de confusion ci-dessous : 394553 (True Negative, TN), qui représente le nombre de pixels identifiés avec précision comme « vides ». De plus, certains pixels 1,7e+06 (faux positif, FP) ont été classés par erreur comme « bâtiment » au lieu de « vide ».

  Le nombre de pixels étiquetés par erreur comme « vides » alors qu'il s'agit en réalité de « bâtiment » est de 10 849 (faux négatif, FN). D'un autre côté, il existe des pixels 5e+05 (True Positive, TP) qui sont identifiés avec précision comme « Bâtiment ».

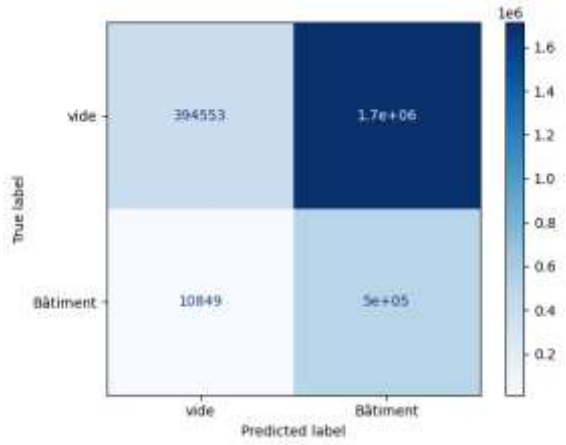


Figure 9-3 : Matrice de confusion du modèle U_Net

9.4 Comparaison des approches et décision

Pour le modèle VGG16, nous avons observé une progression régulière de la précision au fil des époques d'entraînement, atteignant finalement un niveau élevé de 98,06 % sur l'ensemble de tests. En revanche, le modèle ResNet, bien qu'il ait atteint une précision de 85,83 % à l'époque 4 sur l'ensemble de validation, semble avoir du mal à maintenir cette performance, ce qui se traduit par une précision finale sur l'ensemble de test de seulement 69,96 %.

Les matrices de confusion fournissent des informations sur les performances de classification de chaque modèle en détaillant le nombre de prédictions correctes et incorrectes pour chaque classe. Pour le modèle VGG16, toutes les prédictions sont correctes, comme en témoignent les zéros en dehors de la diagonale principale de la matrice, confirmant ainsi la haute précision observée dans les résultats. En revanche, pour le modèle ResNet, bien qu'il ait obtenu des résultats variés, la matrice de confusion révèle qu'il identifie parfois d'autres images comme des maisons, ce qui est préoccupant. Cette anomalie pourrait compromettre notre crédibilité si des images non pertinentes sont considérées comme des maisons.

C'est pourquoi nous avons choisi d'opter pour le modèle VGG16 pour la reconnaissance des images immobilières en raison de son haut niveau de précision et de la qualité de sa matrice de confusion, offrant ainsi une meilleure assurance de la fiabilité de nos résultats.

En ce qui concerne le modèle YOLO et U-Net, nous avons déterminé que YOLO surpasse U-Net en termes de détection d'objets. Les données révèlent que YOLO atteint une précision impressionnante de 71 %, un taux de rappel de 49 %, d'un score F1 de 58 % et un IoU moyen de 0,5908. D'un autre côté, alors que U-Net démontre un taux de rappel élevé de 0,88, sa précision moyenne de 0,24 et son IoU moyen de 0,24 indiquent qu'il détecte un plus grand nombre de vrais positifs, bien qu'avec une précision moindre.

Nous avons choisi de mettre en œuvre le système Yolo pour la détection de l'emplacement des bâtiments afin d'atteindre des niveaux de précision élevés.

10 Conclusion

La conception de cette plateforme de vente immobilière intégrant un API de détection de bâtiment représente une opportunité significative d'améliorer l'efficacité et l'expérience utilisateur dans le domaine de l'immobilier. En combinant des fonctionnalités avancées de gestion des biens immobiliers avec des capacités de reconnaissance automatique des caractéristiques des bâtiments, cette plateforme promet d'offrir des solutions innovantes pour les acteurs du marché immobilier. Ainsi que les détails des méthodes et algorithmes permettant de reconnaître et d'identifier des bâtiments dans des images, à la fois simples et aériennes. Deux modèles de reconnaissance, VGG16 et ResNet, ainsi que les modèles de détection YOLO et U-Net ont été étudiés en détail. VGG16 a montré une amélioration constante de la précision pendant l'entraînement, atteignant finalement un niveau élevé de 98,06 % lors des tests, la matrice de confusion confirmant une grande fiabilité. D'un autre côté, alors que ResNet a atteint une précision de 85,83 % lors de la validation, lors des tests, ses performances ont chuté à 69,96 % en raison de problèmes de classification détectés. YOLO, également à une meilleure performance avec une précision impressionnante de 71 %, un taux de rappel de 49 %, d'un score F1 de 58 % et un IoU moyen de 0, 5908.tandis que le modèle U-Net démontre un taux de rappel élevé de 0,88, sa précision moyenne de 0,24 et son IoU moyen de 0,24 ce qui démontre que Yolo est meilleur par rapport à U-net.

Chapitre III : Implémentation et Expérimentation

1 Introduction

Après avoir fait une étude détaillée et avoir défini une conception mieux appropriée aux besoins de l'application, nous allons nous pencher sur la mise en œuvre de notre système, en présentant d'abord l'environnement de travail, les différents outils de développement utilisés et notre base de données implémentée, puis expliquer le fonctionnement de l'application en présentant quelques interfaces illustratives.

2 Ressources matérielles et logicielles utilisées lors du développement

Afin de réaliser ce projet de recherche, nous avons utilisé les ressources suivantes :

2.1 Ressources Matérielles

Dans le développement de l'application web de vente immobilier et APIs de reconnaissance et détection des images immobilières, nous avons travaillé sur un ordinateur personnel ASUS avec une configuration de 12Go de RAM, un processeur Intel (I7) 6eme génération, et Windows 10 comme un système d'exploitation.

2.2 Outils et matérielles

Outils/matérielles	Fonctions
JAVA	Java est un langage de programmation orienté objet utilisé pour développer des applications web, mobiles et de bureau. Dans notre projet, nous l'avons utilisé pour développer des composants serveur et pour gérer la logique métier.
JAVA EE	Java Enterprise Edition (Java EE) est une plateforme Java conçue pour les applications d'entreprise. Elle fournit une API et un environnement d'exécution pour le développement et l'exécution d'applications réseau robustes et évolutives. Nous avons utilisé Java EE pour construire notre application de vente immobilière.
SQL	SQL (Structured Query Language) est un langage standard pour interagir avec des bases de données relationnelles. Nous l'avons utilisé pour gérer et interroger les données dans notre base de données MySQL.
JSP	JavaServer Pages (JSP) est une technologie qui aide les développeurs à créer dynamiquement des pages web basées sur HTML, XML, ou d'autres types de documents. Nous l'avons utilisé pour la génération de contenu dynamique sur nos pages web.
JSTL	JavaServer Pages Standard Tag Library (JSTL) est une bibliothèque de balises pour JSP qui encapsule la fonctionnalité la plus commune. Nous l'avons utilisé pour simplifier et réduire le code Java dans nos pages JSP.
CSS	Css est un langage informatique de stylage qui décrit la présentation des pages html, et il est utilisé dans la conception de notre pages web et le rendre plus beau, pour donner une bonne expérience aux utilisateurs.
HTML	Un langage de balisage, utilise pour Représenté, structurer et de mettre en forme le contenu de nos pages web.
JAVASCRIPT	JavaScript est un langage de programmation permettant de créer du contenu web dynamique et interactif. Nous l'avons utilisé pour améliorer l'interactivité et l'expérience utilisateur sur notre application web immobilier.
PHYTON	Python est un langage de programmation polyvalent utilisé pour divers types de développement, notamment le développement web, l'analyse de données et l'intelligence artificielle. Nous l'avons utilisé pour la partie backend de nos APIs de reconnaissance et détection d'image immobilière et pour l'implémentation de modèles de machine learning.

Tensorflow	TensorFlow est une bibliothèque open-source pour la machine learning développée par Google. Nous l'avons utilisée pour créer et entraîner nos modèles de reconnaissance et de détection de bâtiments.
Keras	Keras est une bibliothèque open-source qui fournit une interface Python pour les réseaux de neurones artificiels. Il s'agit d'une extension de haut niveau de TensorFlow, et nous l'avons utilisée pour simplifier le développement de nos modèles de machine learning.
Flask	Flask est un Mini Framework du langage pythons, nous l'avons utilisé pour développer le coté backend (non-visuel) de l'API de reconnaissance et détection des bâtiments puisqu'il est conçu pour faciliter et accélérer la mise en route des pages
Eclipse IDE	Eclipse IDE est un environnement de développement intégré utilisé principalement pour le développement Java. Nous l'avons utilisé pour écrire, déboguer et tester notre code Java
Visual studio code	Visual Studio Code est un éditeur de code source léger mais puissant. Nous l'avons utilisé pour écrire et éditer nos APIs en utilisant le langage python
MySQL	MySQL est un système de gestion de base de données relationnelle open-source. Nous l'avons utilisé pour stocker, gérer et interroger les données de notre application.
Apache Tomcat	Apache Tomcat est un serveur open-source qui implémente les spécifications Java Servlet, JSP, et Java Expression Language. Nous l'avons utilisé pour déployer et exécuter notre application web Java.

Tableau 4 : Outils et leur fonction

3 Conception de la base de données

Afin d'implémenter notre base de données, on aura donc besoin d'élaborer un modèle logique de données.

3.1 Le modèle logique des données

Le modèle logique nous permet d'extraire les attributs, leurs appartenances aux différentes classes, ainsi que les associations entre celles-ci. Pour notre plateforme, nous avons définis le modèle logique suivant :

- Utilisateur (id, nom, prénom, téléphone, email, password)

Champ	Type de champ	Contrainte
Id	Int	Clé primaire
Nom	String	Not null
Prénom	String	Not null
Téléphone	String	Not null
Email	String	
Password	String	Not null

Tableau 5 : Modèle logique de l'utilisateur.

- Service (id, région, localisation, prix, catégorie, image, #id_fournisseur)

Champ	Type de champ	Contrainte
Id	Int	Clé primaire
Région	String	Not null
Localisation	String	Not null
Prix	Double	Not null
Catégorie	String	Not null
Image	Long Blob	Not null
#Id_fournisseur	Int	Clé étrangère

Tableau 6 : Modèle logique du service

- Images (id, image, #id_service)

Champ	Type de champ	Contrainte
Id	Int	Clé primaire
Image	Long Blob	Not null
#Id_service	Int	Clé étrangère

Tableau 7 : Modèle logique des images

- Description (#idService, surface, chambre, salon, douche, cuisine, étage, commentaire)

Champ	Type de champ	Contrainte
IdService	Int	Clé étrangère
Surface	String	Not null
Chambre	Int	Not null
Salon	Int	Not null
Douche	Int	Not null
Cuisine	Int	Not null
Etage	Int	
Commentaire	String	Not null

Tableau 8 : Modèle logique de la description

- Localisation (id, longitude, latitude, #idService)

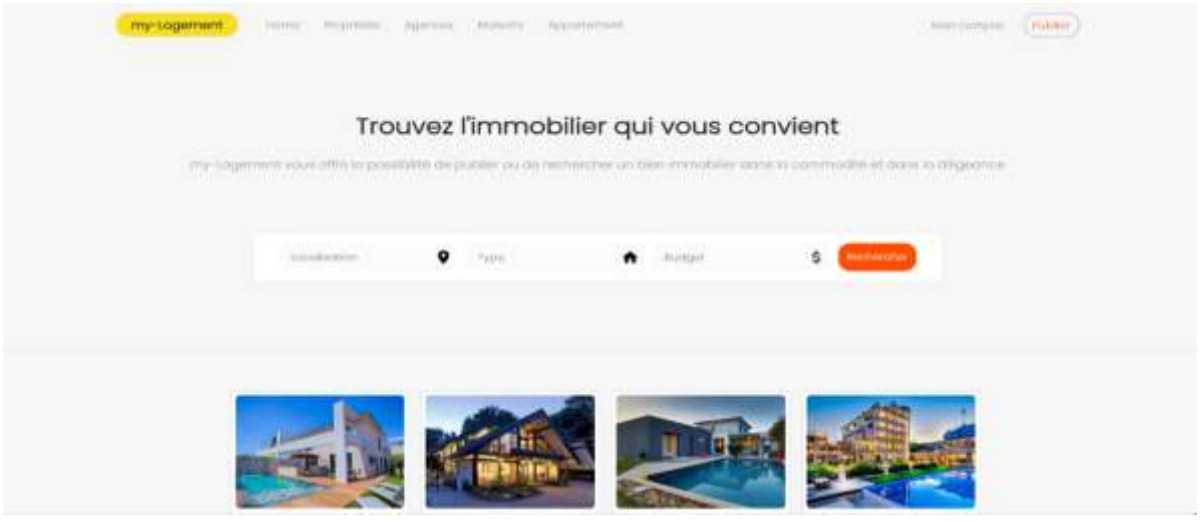
Id	Int	Clé primaire
Longitude	String	Not null
Latitude	String	Not null
#idService	Int	Clé étrangère

Tableau 9 : Modèle logique de la localisation

4 Représentation de page de l'application

4.1 Page d'accueil

Dans ce qui suit, nous allons présenter la page d'accueil qui fournit aux visiteurs une possibilité de voir quelques fonctions.



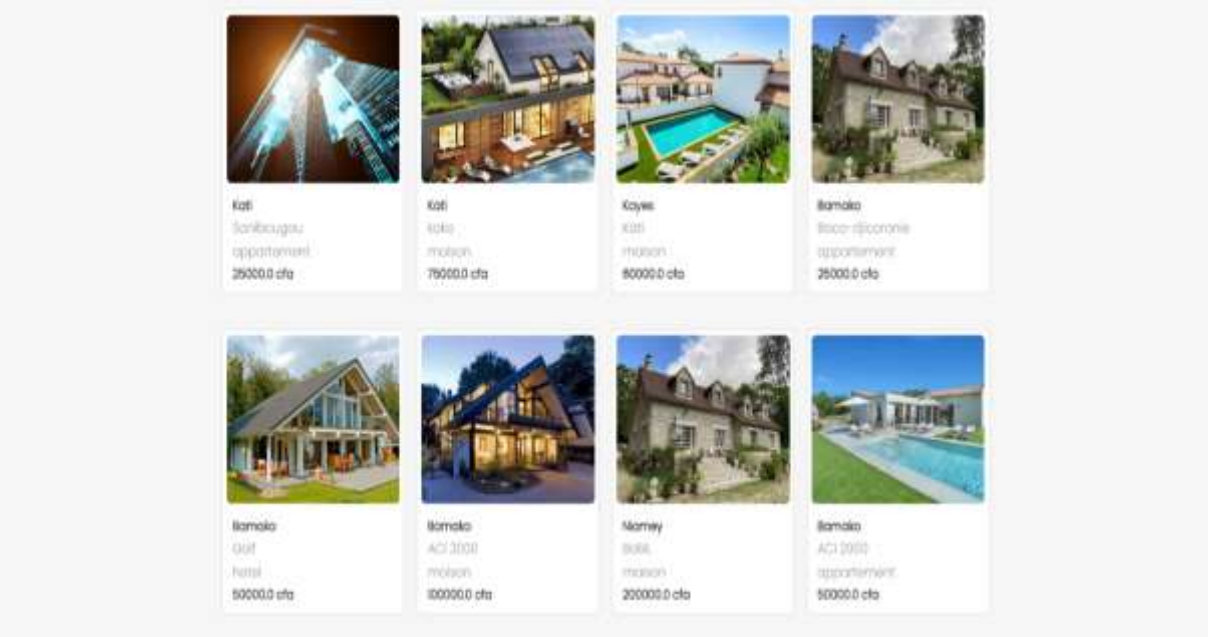


Figure 4-1 : page d'accueil.

4.2 Page de compte :

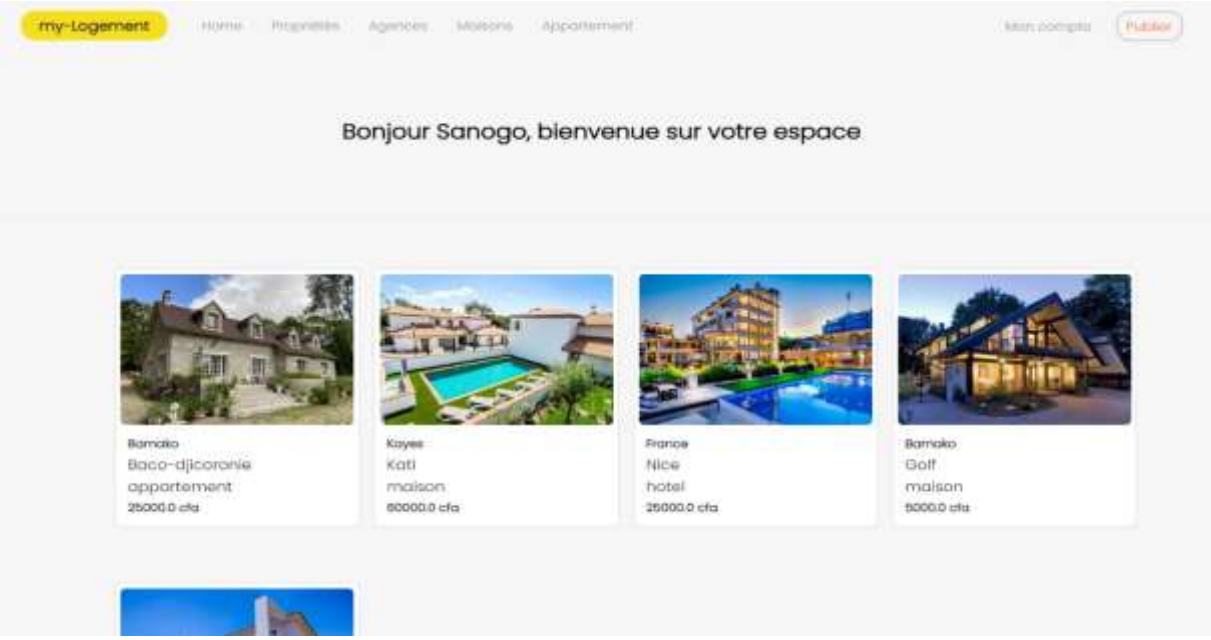


Figure 4-2 : page de compte

4.3 Page de connexion

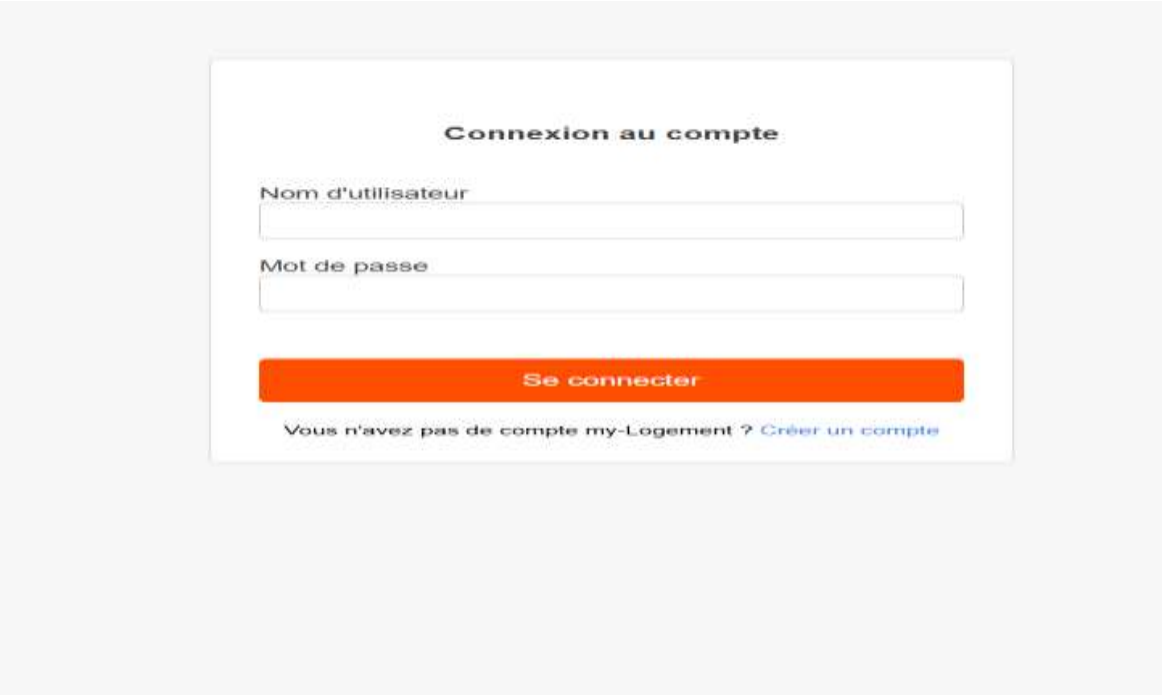


Figure 4-3 : Page de connexion.

4.4 Page de création de compte

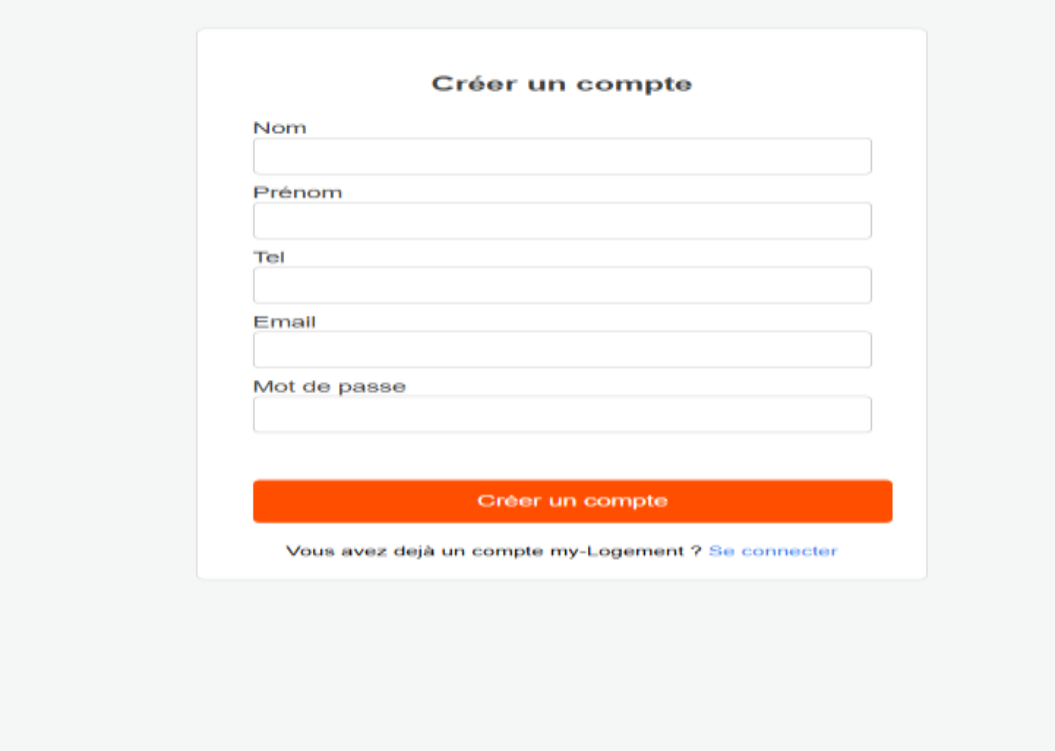


Figure 4-4 : page de création de compte.

4.5 Page de détails

Figure 4-5 : page de visite

5 Conclusion

Dans ce chapitre, qui est la phase finale de notre travail, nous avons présenté brièvement la structure de notre site, les langages de programmation et les outils de développement utilisés pour accomplir ce projet. Finalement, on peut dire que notre application répond à l'objectif souhaité dès le départ.

Conclusion générale

Dans ce travail, nous avons mené une étude comparative de divers articles traitant de la reconnaissance et de la détection d'images immobilières. Cette analyse a révélé que les méthodes basées sur l'apprentissage profond offrent une précision supérieure par rapport à celles basées sur l'apprentissage automatique. En conséquence, nous avons choisi d'utiliser des méthodes d'apprentissage profond telles que VGG, YOLO, ResNet et U-Net dans notre étude.

Pour la reconnaissance d'images, le modèle VGG a atteint une précision de 98,06 %, tandis que le modèle ResNet a obtenu une précision de 69,96 % sur une fusion de trois ensembles de données. Pour la détection, le modèle YOLO a obtenu une précision de 71 %, tandis que le modèle ResNet a atteint 24 % sur un ensemble de données couvrant quatre régions distinctes de l'Égypte.

Nous avons également développé deux API distinctes pour la reconnaissance et la détection d'images immobilières en utilisant les modèles VGG et YOLO. Ces APIs ont ensuite été intégrées dans notre application web pour automatiser les processus de vérification.

Cette application web fonctionne comme une agence immobilière virtuelle, jouant le rôle de médiateur entre le client, le vendeur et le locataire. Elle permet à ses membres de mettre des biens (maisons et appartements) en vente ou en location, et de publier des demandes d'achat avec des caractéristiques bien définies. Les visiteurs du site peuvent consulter les fiches des biens, effectuer des recherches et choisir les biens désirés.

Dans une perspective future, l'amélioration continue des modèles d'apprentissage profond et l'intégration de nouvelles technologies, telles que : un estimateur de prix, un système de recommandation et un système de personnalisation du contenu, pourraient encore enrichir l'expérience utilisateur et optimiser les processus de vente et de location immobilières. De plus, l'expansion de l'application à d'autres régions géographiques et l'inclusion de nouvelles fonctionnalités basées sur les retours des utilisateurs pourraient renforcer l'efficacité et l'adoption de cette solution innovante.

Bibliographies

[1] K. Tyagi, C. Rane, R. Sriram, and M. Manry, "Unsupervised learning," in *Artificial intelligence and machine learning for edge computing*: Elsevier, 2022, pp. 33-52.

[2] A. P. Silalahi, H. G. Simanullang, and M. I. Hutapea, "Supervised Learning Metode K-Nearest Neighbor Untuk Prediksi Diabetes Pada Wanita," *METHOMIKA: Jurnal Manajemen Informatika & Komputerisasi Akuntansi,* vol. 7, no. 1, pp. 144-149, 2023.

[3] D. A. Freedman, *Statistical models: theory and practice*. cambridge university press, 2009.

[4] S. M. I. Chaibeddra, "Détection visuelle d'objets statiques et dynamiques dans un environnement de type route et classification en exploitant l'apprentissage profond ", Université des Sciences et de la Technologie, Houari Boumediene, 2019 [Online]. Available:

[5] A. Celeghin *et al.*, "Convolutional neural networks for vision neuroscience: Significance, developments, and outstanding issues," *Frontiers in Computational Neuroscience,* vol. 17, p. 1153572, 2023.

[6] P. Shruti and R. Rekha, "A Review of Convolutional Neural Networks, its Variants and Applications," in *2023 International Conference on Intelligent Systems for Communication, IoT and Security (ICISCoIS)*, 2023: IEEE, pp. 31-36.

[7] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *arXiv preprint arXiv:1409.1556,* 2014.

[8] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," *Advances in neural information processing systems,* vol. 25, 2012.

[9] L. Luo, P. Li, and X. Yan, "Deep learning-based building extraction from remote sensing images: A comprehensive review," *Energies,* vol. 14, no. 23, p. 7982, 2021.

[10] R. Kundu. "YOLO: Algorithm for Object Detection Explained [+Examples]." https://www.v7labs.com/blog/yolo-object-detection (accessed 12 Sptembre, 2024).

[11] J. P. Cohen, W. Ding, C. Kuhlman, A. Chen, and L. Di, "Rapid building detection using machine learning," *Applied Intelligence,* vol. 45, pp. 443-457, 2016.

[12] W. Boonpook, Y. Tan, Y. Ye, P. Torteeka, K. Torsri, and S. Dong, "A deep learning approach on building detection from unmanned aerial vehicle-based images in riverbank monitoring," *Sensors,* vol. 18, no. 11, p. 3921, 2018.

[13] N. Pumpong, P. Boonserm, K. Kobayashi, and N. Cooharojananone, "Building detection in airports through remote sensing image using YOLOv3 with Jet Saliency map," in *2021 IEEE 8th International Conference on Industrial Engineering and Applications (ICIEA)*, 2021: IEEE, pp. 491-496.

[14] C. Ünsalan and K. L. Boyer, "A system to detect houses and residential street networks in multispectral satellite images," *Computer Vision and Image Understanding,* vol. 98, no. 3, pp. 423-461, 2005.

[15] S. K. Das, H. Bharath, and P. Prakash, "Automated building extraction using high-resolution satellite imagery though Ensemble modelling and Machine learning," *Remote Sensing of Land,* vol. 2, pp. 1-14, 2018.

[16] Z. Lari and H. Ebadi, "Automated building extraction from high-resolution satellite imagery using spectral and structural information based on artificial neural networks," in *ISPRS Hannover workshop*, 2007, no. 1346.

[17] W. Alsabhan, T. Alotaiby, and B. Dudin, "Detecting buildings and nonbuildings from satellite images using U-Net," *Computational Intelligence and Neuroscience,* vol. 2022, 2022.

[18] TED8080. *House Prices and Images - SoCal*. [Online]. Available: https://www.kaggle.com/datasets/ted8080/house-prices-and-images-socal/data?select=socal2.csv

[19] T. ALEXANDER. "Photos of apartments and local area." https://www.kaggle.com/datasets/tumanovalexander/home-bro-images (accessed 14 Mai, 2024).

[20] KHLAIFIABILEL. *Egypt Building Change Detection*. [Online]. Available: https://www.kaggle.com/datasets/khlaifiabilel/egypt-building-change-detection

[21] V. Khryashchev, L. Ivanovsky, V. Pavlov, A. Ostrovskaya, and A. Rubtsov, "Comparison of different convolutional neural network architectures for satellite image segmentation," in *2018 23rd conference of open innovations association (FRUCT)*, 2018: IEEE, pp. 172-179.