L'intégration des nouvelles technologies et de l'intelligence artificielle dans la conservation du patrimoine bâti pour détecter la présence de pathologies.

Auteur : Boutet, Simon
Promoteur(s) : Hallot, Pierre
Faculté : Faculté d'Architecture
Diplôme : Master en architecture, à finalité spécialisée en art de bâtir et urbanisme
Année académique : 2024-2025
URI/URL : http://hdl.handle.net/2268.2/22348

UNIVERSITÉ DE LIÈGE – FACULTÉ D'ARCHITECTURE

L'intégration des nouvelles technologies et de l'intelligence artificielle dans la conservation du patrimoine bâti pour détecter la présence de pathologies

Travail de fin d'études présenté par Simon BOUTET en vue de l'obtention du grade de Master en Architecture

Sous la direction de : Pierre HALLOT

Année académique 2024-2025

REMERCIEMENTS

Je souhaite d'abord remercier Monsieur Pierre Hallot, professeur de la Faculté d'Architecture de l'Université de Liège et promoteur de ce travail, lequel a su me conseiller et m'orienter tout le long de ma recherche.

J'exprime également ma plus grande gratitude envers les lecteurs et membres du jury, Monsieur Philippe Sosnowska et Madame Aurélie de Boissieu, professeurs de la Faculté d'Architecture de l'Université de Liège, pour l'intérêt et le temps consacrés à la lecture de ce mémoire.

De plus, je désire exprimer ma reconnaissance envers ma famille et les proches qui m'ont soutenu, conseillé et guidé tout au long de la rédaction de ce travail. Je remercie également tous les relecteurs qui ont donné de leur temps pour me permettre d'améliorer ce mémoire et le corriger. Enfin, je remercie personnellement Justin Henrotte pour sa persévérance et les longues heures passées à m'assister lorsque mes connaissances numériques ne suffisaient pas.

De manière générale, je tiens à remercier toutes les personnes qui ont eu de près ou de loin un impact dans le développement de mon intérêt pour le patrimoine et le numérique durant ces cinq années d'études.

TABLE DES MATIÈRES

I. INTRODUCTION

A. Problématique

Selon Kaja et Goyal (2023), le domaine de la construction est aujourd'hui responsable de 10 % des émissions mondiales de CO_2. Cette réalité contraste avec les objectifs ambitieux fixés lors de la COP26 à travers le Pacte de Glasgow pour le climat, qui vise à réduire de 45 % les émissions mondiales de dioxyde de carbone d'ici 2030 (UNFCCC, 2021). Face à ce défi, il est nécessaire de repenser nos modes de construction et d'occupation du territoire. En Wallonie, le SPW (2023) renforce cette vision avec son schéma de développement territorial, qui vise à limiter l'artificialisation des terres et met l'accent sur la réutilisation et la transformation du bâti existant.

Parallèlement, la reconnaissance des biens patrimoniaux connaît une croissance constante, notamment en Wallonie, où la liste des biens classés et inscrits dans l'inventaire régional s'allonge chaque année. À l'échelle mondiale, cette tendance se reflète également, car le nombre de sites inscrits au patrimoine mondial de l'UNESCO est passé de 830 en 2006 à 1326 en 2023. De nos jours, la notion de patrimoine aujourd'hui ne cesse donc de s'élargir, englobant de plus en plus de bâtiments à conserver et à entretenir. De plus, il est indispensable pour l'environnement de construire moins et de conserver l'existant. Ainsi, le patrimoine bâti dans son ensemble joue un rôle crucial à deux égards : préserver un héritage culturel élargi et répondre aux enjeux environnementaux de notre époque.

Alors que la Charte de Venise (1964) insiste sur l'importance d'un entretien continu du patrimoine afin de prévenir sa dégradation, ceci constitue un défi considérable en raison des effets conjugués du vieillissement naturel, des changements environnementaux, des interventions humaines et des catastrophes naturelles. Les pathologies visuelles comme les fissures, l'efflorescence et le décollement des matériaux compromettent progressivement l'intégrité et la pérennité du bâti existant. Cependant, le diagnostic d'un bâtiment et la détection de ses pathologies reste un processus extrêmement long, majoritairement manuel et sujet à l'interprétation des experts. Les résultats des inspections varient selon l'expertise et les méthodes employées, ce qui limite leur cohérence et leur fiabilité. Cette étape, préalable à toute analyse structurelle plus approfondie, constitue la base de toute intervention sur un bâtiment existant.

Avec l'émergence des nouvelles technologies comme l'intelligence artificielle (IA), de nombreux domaines d'activité bénéficient d'une automatisation et d'une accélération de nombreux processus. En architecture, l'IA générative permet déjà de créer des centaines de simulations de lotissements ou d'aménagements de plans en tenant compte de divers paramètres comme l'ensoleillement ou la densité bâtie. Cependant, dans le domaine du patrimoine, l'intégration technologique progresse à un rythme différent. Des outils comme la lasergrammétrie et la photogrammétrie assistées par IA offrent néanmoins des relevés 3D extrêmement précis et de plus en plus automatisés, assistants déjà les experts lors de l'inspection d'un bâtiment existant.

En ce qui concerne la détection des pathologies, la diversité des types de bâtiments, matériaux, pathologies et environnements ont longtemps rendu toute forme d'automatisation difficile. Toutefois, les avancées récentes dans les technologies de Computer Vision (CV) et de Deep Learning (DL), qui font partie de l'intelligence artificielle, promettent désormais d'accélérer et fiabiliser la détection de ces pathologies, mais aussi de permettre une meilleure gestion des inspections dans le temps grâce à la numérisation des résultats (Mishra & Lourenço, 2024). Depuis 2017, les applications dans la détection de pathologies sont étudiées et de nombreuses recherches sur des cas d'études variés ont été réalisées. Les modèles de Deep Learning, en combinant rapidité et efficacité, semblent particulièrement pertinents dans ce contexte contrôlé, permettant des analyses massives et homogènes tout en réduisant la subjectivité humaine. Reste à savoir si ces résultats sont pertinents dans un déploiement en situation réelle et s'ils peuvent remplacer, ou simplement compléter les méthodes traditionnelles.

Ce travail de recherche a pour but d'analyser et comparer les différentes approches de détection de pathologies par IA en examinant leurs avantages, leurs limites et leur pertinence dans des contextes variés. Ceci dans le but d'orienter les potentiels utilisateurs de ces technologies sur des modèles et méthodes spécifiques à leurs besoins. L'étude s'intéresse également à l'intégration de l'IA avec des technologies établies comme la photogrammétrie et la lasergrammétrie, qui offrent également des moyens non destructifs pour capturer des données précises pour le patrimoine bâti.

Selon l'hypothèse que la technologie a atteint un niveau de précision et d'exactitude suffisant, la question de recherche suivante se pose : **l'automatisation par IA de la détection de pathologies visuelles du patrimoine bâti est-elle pertinente aujourd'hui ?**

B. Méthodologie générale

Ce travail est structuré en trois parties afin d'évaluer l'intérêt et la pertinence de l'automatisation de la détection des pathologies du patrimoine bâti par intelligence artificielle, comparée aux méthodes traditionnelles. L'état de l'art va d'abord permettre de définir et d'énoncer tous les points nécessaires à la compréhension du travail, avec une définition de ce que sont le patrimoine et les pathologies, pour ensuite développer les techniques actuelles et par IA des inspections visuelles du patrimoine bâti. La seconde partie sera une comparaison sous forme de tableaux de nombreuses études sur la détection automatique de pathologies, tandis que la dernière consistera à déployer en situation réelle un modèle d'intelligence artificielle. Pour être plus correct, j'utiliserai désormais le terme « modèle de Deep Learning », que je définirai plus tard dans ce travail.

Pour commencer l'état de l'art, il faudra donc définir les critères de reconnaissance, ou valeurs, liés au patrimoine bâti et leur évolution à travers les siècles. Ceci commencera par Viollet-le-Duc et ses premiers travaux sur les monuments du Moyen Age et se terminera par les grands textes internationaux tels que le document de Nara. Cette section s'appuiera sur le travail de fin d'étude de Margot Heynen (2020) qui a étudié l'évolution des notions de valeur dans le domaine de la restauration. Les origines des pathologies susceptibles d'affecter le patrimoine et leurs différentes typologies seront par la suite développées.

Ensuite, les techniques et outils actuellement utilisés pour les inspections visuelles du patrimoine seront exposés. Cette exploration est essentielle pour comprendre les méthodes existantes et les comparer aux approches basées sur l'intelligence artificielle. Les méthodes des différents modèles de Deep Learning seront ensuite développées, puisque certaines étapes des inspections actuelles comme l'acquisition, le traitement et l'interprétation des données, sont similaires à celles demandées. L'objectif est d'identifier les points communs ou les complémentarités entre ces approches et de déterminer si certaines technologies actuelles peuvent être intégrées ou utilisées parallèlement à des outils de détection automatique. Les avantages et les limitations des approches traditionnelles et de l'IA seront brièvement abordés pour établir une base de comparaison.

La troisième partie de l'état de l'art s'intéressera aux différentes tâches qui peuvent être demandées de ces modèles de Deep Learning : la classification d'images, la détection d'objets, la segmentation sémantique et la segmentation d'instances. Ces tâches ont des degrés de précision, de complexité et

d’exigences computationnelles propres qui les rendent plus ou moins pertinentes selon la situation. Le travail de Guo et al. (2024) ainsi que celui de Mishra et Lourenço (2024) seront analysés afin de déterminer rapidement les modèles les plus adaptés à chaque tâche. Le premier a recensé la quasi-totalité des études abordant la détection de pathologies par Deep Learning sur les structures civiles de mars 2017 (première publication) à juin 2023. Le second permet de compléter ces données jusque mars 2024 et aborde exclusivement le patrimoine bâti. Un tableau synthétique sera dressé pour résumer les performances des modèles en fonction des tâches citées précédemment. Les architectures des principaux modèles seront ensuite décrites pour mieux comprendre leurs spécificités et leur potentiel d’application.

Finalement, une analyse non exhaustive des études scientifiques reprenant les différents modèles présentés au point précédent permettra d’établir des grilles d’évaluations de ces derniers selon des critères comme la précision, les limitations, le temps d’entrainement, le délai de traitement des images, etc. Ces observations seront mises en parallèle avec l’ensemble des données reprises dans ce travail afin de comparer les performances de chaque modèle. Pour ce faire, un guide dichotomique reprenant l’entièreté des études analysées sera établi. Il permettra d’orienter le lecteur sur les tâches, les méthodes d’acquisition et les objectifs pris en compte lors d’une situation précise

Après cette analyse, un déploiement en situation réelle d’un modèle aura pour but de déterminer la pertinence de l'automatisation par IA pour détecter des pathologies visuelles du patrimoine bâti, en se concentrant sur les fissures dans la maçonnerie de briques. Ces choix seront développés en détail plus tard dans le travail. Néanmoins, l’étude est volontairement limitée à une seule pathologie et une seule matérialité, afin de concentrer les observations sur le fonctionnement du modèle et éviter de répéter les mêmes observations pour des cas similaires. Celui-ci sera choisi selon une publication scientifique afin de comparer l’étude à des résultats déjà validés scientifiquement. Les conclusions permettront de répondre à la question de recherche en vérifiant si l’intégration de l’IA dans ce domaine est pertinente et sous quelles conditions.

C. Limitations de la recherche

L'analyse des études scientifiques de ce travail n'a pas pour but de déterminer le modèle idéal parmi ce qui existe aujourd'hui mais veut contribuer à orienter le potentiel utilisateur de ces technologies en prenant en compte les spécificités de ces dernières. Au vu de la rapide évolution des technologies présentées dans ce travail, il est cependant important de considérer que les informations présentées et les modèles analysés seront peut-être, dans un futur proche, très différents ou même obsolètes par rapport à ce qui est énoncé dans ce travail.

L'intelligence artificielle et la conservation du patrimoine étant deux sujets très vastes, il a été choisi de limiter la recherche et les études analysées selon les critères suivants :

- **La technique d'intelligence artificielle utilisée est du Deep Learning**

Le DL semble être le plus adapté pour des tâches visuelles parmi les catégories d'IA.

- **Les pathologies étudiées sont visibles à la surface du bâtiment**

Afin de limiter la recherche à une seule approche, les technologies radar telles que celles approchées par Hamrouche (2011) ne sont pas considérées. Ces dernières sont principalement utilisées lors d'interventions ultérieures à l'inspection visuelle des pathologies de surface.

- **Les données utilisées par les modèles sont des images en couleur (RVB)**

L'approche de la thermographie infrarouge à la place de photographie pour une détection automatisée des défauts a été abordée par Garrido et al (2021) ainsi que dans d'autres études mais cette approche s'intéresse à des dégâts non-observables à l'œil nu, principalement sous-surface. Dans le même ordre d'idée, les images thermiques, en niveaux de gris, etc… ne sont pas pris en compte.

- **L'équipement utilisé pour l'acquisition des images est libre**

Ceci a pour but de comparer la pertinence des différents équipements en fonction de la situation.

- **La localisation des bâtiments étudiés est libre**

Elle permet d'aborder un maximum d'études et observer les pathologies sur une échelle internationale.

II. ÉTAT DE L'ART

1. Patrimoine et pathologies : définitions

1.1. Évolution de la notion de patrimoine

Selon le Code wallon du Patrimoine (CoPat), on peut aujourd'hui définir le patrimoine comme « L'ensemble des biens immeubles qui reflètent et expriment des valeurs, des croyances, des connaissances, des savoir-faire et des traditions en constante évolution, dont la protection se justifie en raison de leur intérêt archéologique, historique, architectural, scientifique, artistique, social, mémoriel, esthétique, technique, paysager ou urbanistique et compte tenu de critères de rareté, d'authenticité, d'intégrité ou de représentativité. Cela inclut tous les aspects de l'environnement résultant de l'interaction dans le temps entre les personnes et les lieux. ». (Delnoy et Van Damme, 2021). Dans l'histoire de l'architecture, ces notions de patrimoine et de restauration sont pourtant relativement récentes et ont beaucoup évolué au fil du temps, principalement à travers l'établissement de valeurs propres à ce qui est considéré comme un bâtiment patrimonial.

C'est pendant la première moitié du XIXe siècle qu'un mouvement de restauration du patrimoine médiéval se développa en France. Eugène Viollet-le-Duc, architecte français, va restaurer de nombreux édifices religieux comme la cathédrale Notre-Dame de Paris ou la basilique de Saint-Denis, mais également les parties historiques des villes de Narbonne et de Carcassonne (Auzas, 1979). Selon lui, il est nécessaire de faire revenir le bâtiment à un état originel qui peut même ne pas avoir existé. Il détermine un état de référence propre au monument et efface ensuite les traces de sa durée de vie qui résultent pourtant d'une stratification historique (Gaussuin, 2022). Viollet-le-Duc accorde aux bâtiments restaurés une valeur esthétique, voire artistique. Il définira une première fois ce concept : « Restaurer un édifice, ce n'est pas l'entretenir, le réparer ou le refaire, c'est le rétablir dans un état complet qui peut n'avoir jamais existé à un moment donné. » (Viollet-le-Duc, 1866).

Pendant près de deux siècles, cette première définition sera revue, nuancée et critiquée. D'abord par John Ruskin, écrivain, poète, peintre et critique d'art britannique, qui considère par exemple le Moyen Age comme l'idéal d'une époque perdue. Selon lui, la valeur historique du patrimoine est mise en avant si on le conserve tel qu'il est, afin d'éviter un « faux témoignage » en intervenant sur le bâtiment. Bien que totalement opposées, ces deux approches ont pour objectif commun de transmettre des valeurs à travers le patrimoine, mais elles doivent être nuancées.

Un compromis entre les deux théories se dégage au XXe siècle. D'abord avec Camillo Boito, architecte et écrivain italien. Il développe ainsi la restauration philologique qui, à l'inverse de la restauration stylistique de Viollet-le-Duc, considère le bâtiment comme un témoignage historique et cherche à ne pas restaurer de façon mensongère (Heynen, 2020, p.11-13). Aloïs Riegl, un historien de l'art autrichien, va quant à lui constater deux catégories de valeurs, d'une part celle de remémoration (passé de l'objet) et d'autre part celle de contemporanéité (présent de l'objet). Il décline chacune d'elle en trois genres.

La catégorie de **remémoration** se décline en :

- Valeur d'ancienneté : témoin du passé.
- Valeur historique : représente une étape dans le développement de l'activité humaine.
- Remémoration intentionnelle : témoignage immortel voulu par le fondateur, mais peut devenir valeur historique si plus pertinente dans le présent.

La catégorie de **contemporanéité** comprend :

- Valeur d'art relative : le jugement artistique est en constante évolution.
- Valeur d'usage : utilité de l'édifice, entretien raisonnable.
- Valeur de nouveauté : l'impression que l'objet est neuf.

Néanmoins, la qualité de monument n'est pas définie comme telle, mais attribuée socialement un jour, à une époque et dans des circonstances déterminées, peu importe les valeurs (Dolff-Bonekämper, 2020). Louis Cloquet, architecte belge, ajoutera que la valeur d'usage est propre aux bâtiments vivants, autrement dit ceux qui sont habités et utilisés, tandis que les bâtiments morts ont une valeur historique.

Lors du premier congrès international des architectes et techniciens des monuments historiques en 1931, la charte d'Athènes mettra en place 7 résolutions importantes, dont la nécessité de procéder à un entretien continu du patrimoine. Par ailleurs, il sera déclaré que pour tout monument hors ruine, il faudrait conseiller « […] avant toute consolidation ou restauration partielle, l'analyse scrupuleuse des maladies de ces monuments. [Les experts] ont reconnu en fait que chaque cas constituait un cas d'espèce » (ICOMOS, 2012).

C'est après la seconde guerre mondiale que Cesare Brandi, historien de l'art et écrivain italien, va considérer la double historicité des œuvres. Il pense que la création de l'œuvre appartient à un passé révolu et qu'elle est, comme sa restauration, un fait historique dans son histoire. L'état initial du bâtiment

ne peut pas être retrouvé car il appartient au passé mais le caractère artistique doit être conservé. C'est dans cette réflexion qu'il décidera de réintégrer les éléments perdus ou détruits par la guerre, pour rendre à l'œuvre sa capacité de transmettre un message artistique. Ces éléments perdus, ou lacunes, signifient une « Perte locale de la matière du bien culturel qui entraine l'interruption de l'image et laisse une marge d'interprétation lors de la réception de l'œuvre » (Bergeon-Langle et Brunel, 2014). Selon leur nature, ces lacunes sont parfois bénéfiques pour représenter l'histoire de l'édifice (impacts de balle, perforations mineures…) ou tellement importantes qu'elles forcent une réinterprétation d'une partie du bâtiment (Detry, 2016). Afin de définir précisément des notions de valeurs du patrimoine qui sont propres à chaque culture, la Charte de Venise (1964) établira des principes communs à l'échelle internationale qui pourront s'adapter selon les cas et les cultures, mais en gardant des règles communes (Heynen, 2020, p.13-15).

C'est finalement avec le Document de Nara (1994), remis à jour en 2014, que la notion de patrimoine culturel sera élargie et que son évolution continue sera reconnue. Les valeurs qui définissent le patrimoine sont donc aujourd'hui fondées sur des réévaluations périodiques et considèrent un développement durable, selon le contexte socio-économique actuel (Heynen, 2020, p.20-23). C'est dans cette approche que l'Agence wallonne du Patrimoine (AWaP, 2024a) recense actuellement plus de 54000 biens dans l'inventaire du patrimoine immobilier culturel de Wallonie. La plupart de ces biens sont des églises et châteaux, mais surtout des fermes, maisons, granges ou autres éléments bâtis parfois issus du siècle dernier donc relativement récents, qui ont néanmoins été sélectionnés pour leur intérêt patrimonial. Cet inventaire est en constante évolution.

Aujourd'hui en Wallonie, l'AWaP (2024a) se base sur 13 critères et intérêts (utilisés seuls ou combinés) qui servent de balises pour une sélection objective des biens inventoriés :

- « **Authenticité :** la fonction et l'usage, la forme et les matériaux ainsi que l'environnement du bien correspondent à l'état d'origine.
- **Intégrité :** le bien présente une homogénéité et une cohérence. Les fonctions premières sont encore bien identifiables malgré le changement d'usage.
- **Rareté :** au niveau local, le bien est un témoignage unique, rare ou exceptionnel, même fragmentaire par sa typologie, son style, sa datation ou son intérêt social ou historique.
- **Typologie :** le bien possède des caractères architecturaux liés à une fonction spécifique.

- **Archéologique :** le bien conserve, hors-sol, le témoignage significatif d'une occupation ou d'un usage ancien.
- **Architectural :** le bien répond aux caractéristiques d'un style architectural et possède une qualité de composition, de proportions, d'exécution, de vocabulaire formel et/ou décoratif.
- **Artistique :** le bien est conçu comme une "œuvre d'art" ou se singularise par un élément décoratif particulier.
- **Historique :** le bien rappelle un évènement ou une période significative de l'Histoire.
- **Mémoriel :** le bien commémore un évènement ou une tradition liée à la mémoire collective.
- **Paysager :** le bien s'intègre particulièrement bien au terrain et/ou à l'environnement paysager.
- **Social :** le bien relève d'une organisation sociale, d'un mode de vie ou de pensée.
- **Technique :** le bien est représentatif d'innovations techniques ou témoigne de l'ingéniosité de l'homme.
- **Urbanistique :** le bien contribue à la structuration de la trame bâtie. »

Même si les grands principes énoncés au XIXe et XXe siècles restent applicables, la notion de patrimoine englobe au XXIe siècle un grand nombre de monuments, logements, édifices religieux, statues… qui ont chacun une typologie, une architecture, une intégrité ou d'autres valeurs qui leur sont propres. Il est évident que notre perception de ce qui doit impérativement être conservé va encore s'élargir avec les modifications de modes de consommation que le changement climatique engendre.

Bien que les méthodes d'inspections, qui seront présentées dans ce travail, concernent majoritairement un patrimoine protégé, il serait réducteur de ne pas s'intéresser à l'existant dans son ensemble. En effet, l'IA n'a pas besoin de faire la différence entre la façade d'un bâtiment classé et celle d'une maison vernaculaire par exemple. Chaque édifice construit par l'homme, qu'il ait été conçu il y a plusieurs siècles ou il y a quelques années, est susceptible d'être affecté par les mêmes pathologies. C'est pourquoi l'utilisation du terme « patrimoine bâti » fera ici référence à l'ensemble des bâtiments existants et ne se limitera pas seulement à la définition du CoPat.

1.2. Origines des dégradations du bâtiment

Avant d'analyser en détail les pathologies les plus susceptibles d'être observées lors d'un diagnostic du patrimoine bâti, il convient d'en examiner les origines. Puisque ce travail aborde le patrimoine bâti dans son ensemble sans se limiter au patrimoine exceptionnel reconnu par les autorités compétentes, il est essentiel de prendre en compte les désordres à la fois liés au vieillissement des édifices anciens et ceux résultant de défauts dans les constructions plus récentes.

Selon Philipparie (2019), quatre grandes catégories de phénomènes peuvent expliquer les origines des désordres du bâtiment : le climat, le comportement des matériaux, le non-respect des règles de l'art et les conditions d'exécution des ouvrages. Ces deux dernières, qui reprennent l'ensemble des bonnes pratiques et des techniques existantes dans le domaine du bâtiment, peuvent être fusionnées dans ce que Watt & Swallow (1995) considèrent comme les facteurs humains. Ils ajoutent également les phénomènes atmosphériques au climat, comme Carrie & Morrel (1975) avant lui. Finalement, le comportement des matériaux considère leur réponse aux différentes sollicitations et conditions auxquels ils sont soumis, notamment l'usure dans le temps ou la déformation en présence d'humidité, de changement de température, etc. (Philipparie, 2019).

On notera que d'autres classifications de ces origines existent, notamment celle d'Addleson (1993), qui identifie l'humidité, les mouvements et les modifications biologico-chimiques comme les trois principales causes de pathologies. Guelmine (2019) distingue quant à lui quatre types de pathologies distinctes. Ces dernières sont les défauts (de conception ou de matériau), les surcharges mécaniques, les gradients de température et d'humidité, et les attaques chimiques. Bien que ces catégorisations diffèrent de celle adoptée dans ce travail, les origines mentionnées restent similaires. Dans les faits, ces sources de désordres sont nombreuses et souvent interconnectées, ce qui rend crucial l'identification du principal « coupable ».

1.2.1. Phénomènes climatiques

Les préoccupations croissantes liées au changement climatique ont renforcé la prise de conscience des réactions et des réponses de l'environnement face aux activités humaines. Les évolutions de la fréquence et l'intensité des événements météorologiques comme les inondations, affectent déjà le patrimoine bâti. L'impact du climat sur les bâtiments varie toutefois selon leur situation géographique, leur exposition, leur conception, leur état, leur usage et selon les matériaux utilisés (Watt & Swallow, 1995).

Par exemple, une augmentation de l'intensité des précipitations peut compromettre les systèmes d'évacuation des eaux pluviales, alors que les fluctuations et la hausse des températures ambiantes affectent à la fois le bâti et le sol sur lequel il est construit (Watt & Swallow, 1995).

De manière générale, l'ouvrage doit résister aux phénomènes climatiques courants comme la pluie et les inondations qui peuvent en découler, mais aussi aux périodes de sécheresse ou de gel qui peuvent impacter les sols sur lesquels reposent les édifices. Le manque d'eau peut causer une rétraction de certains sols argileux tandis qu'on observe une augmentation de volume lorsque le sol gèle. La neige peut quant à elle ajouter une masse considérable à l'ouvrage et entrainer des dommages si la structure ne suffit pas. Enfin, l'ouvrage doit aussi se comporter correctement face à des événements plus rares comme un incendie ou un séisme (Philipparie, 2019).

Outre les phénomènes climatiques courants, la pollution atmosphérique et les poussières impactent également le patrimoine bâti. Souvent accusée par la majorité des usagers d'être la grande responsable des décolorations et des salissures sur les façades, la pollution atmosphérique de type industriel se constitue de particules solides comme des cendres, des particules de charbon ou d'autres imbrulés, et de vapeurs ou gaz comme l'oxyde de carbone. Ces émissions sont principalement issues des procédés industriels, de la combustion d'énergies fossiles et des véhicules automobiles (Carrie & Morrel, 1975).

En réalité, ce sont les poussières présentes dans l'atmosphère qui sont responsables des salissures. Ces poussières peuvent être classées en deux groupes : les aérosols temporaires et les aérosols permanents. Les premiers se déposent essentiellement par gravité sur les parois non abritées, principalement horizontales ou en saillie. Les autres, portés par l'air, peuvent atteindre n'importe quelle surface et y rester accrochés en s'y collant, parfois par réaction chimique. La pluie et le vent participent énormément à la dispersion de ces aérosols et jouent également un rôle dans la forme que prennent les salissures. Selon leur intensité et leur durée, les précipitations peuvent cependant nettoyer en partie les façades (Carrie & Morrel, 1975).

1.2.2. Facteurs humains

Les facteurs humains peuvent prendre plusieurs formes. Ils vont des circonstances liées à la conception et à la construction du bâtiment, à l'absence ou non d'entretien en passant par l'utilisation qui est faite de l'édifice (Philipparie, 2019). Bien que les bâtiments anciens soient fréquemment perçus comme mieux construits que leurs homologues modernes, cette idée est souvent erronée. Les constructions

réalisées au fil du temps et selon des méthodes de l'époque pouvaient souffrir de contraintes liées à un espace limité, à des matériaux insuffisants, à une main-d'œuvre réduite et à des ressources inadéquates. Cependant, beaucoup de bâtiments de qualité inférieure ont déjà été détruits ou démolis. Par conséquent, les structures encore debout aujourd'hui sont généralement mieux construites (Watt & Swallow, 1995).

En outre, un choix inapproprié et une spécification incorrecte des matériaux de construction, que ce soit lors de la construction initiale ou au cours de modifications et réparations ultérieures, peuvent causer des dommages durables à un bâtiment. De plus, l'industrie de la construction a souvent été critiquée pour ses normes insuffisantes en matière de qualité de la main-d'œuvre et des opérations sur site, notamment en raison de pratiques médiocres, d'une supervision inadéquate, d'une protection inefficace et de l'absence de conditions de stockage appropriées pour les matériaux (Watt & Swallow, 1995).

1.2.3. Comportement des matériaux

Chaque matériau réagit différemment aux contraintes telles que le contact avec l'humidité, le gel, la pollution, les rayons solaires et les variations de température auxquelles il peut être exposé. Ces contacts peuvent causer des changements physiques ou chimiques. Les changements physiques sont des dommages structurels causés par une action mécanique, à une échelle macro ou microscopique. Cela inclut la cristallisation des sels solubles et les effets du gel. Les changements chimiques désignent quant à eux une réaction ayant lieu dans des solides, des liquides ou des gaz. Dans le domaine des bâtiments et des matériaux de construction, des exemples de changements chimiques incluent la corrosion, la carbonatation, les attaques sulfatiques, la pollution, la contamination et les effets des radiations électromagnétiques. Ces réactions chimiques peuvent également entraîner des changements physiques (Watt & Swallow, 1995).

Les matériaux hygroscopiques (qui ont tendance à retenir l'humidité de l'air) absorbent l'humidité provenant de sources liquides ou de vapeur. Ils peuvent également fournir l'humidité nécessaire pour initier ou maintenir des réactions dans d'autres matériaux. Ces matériaux subissent généralement des mouvements et des gonflements selon l'humidité absorbée. Les matériaux contenant des sels peuvent subir des changements de phase (cristallisation, dissolution), qui risquent de provoquer des dommages mécaniques dans les matériaux contaminés, tels que la brique, la pierre et le mortier. La présence de pollution en surface peut accentuer les dommages en initiant des réactions avec certains matériaux grâce aux mouvements d'humidité. Dans certains cas, les matériaux poreux comme le calcaire peuvent même se dissoudre sous l'action de l'eau (Watt & Swallow, 1995).

1.3. Types de pathologies

Comme énoncé dans la méthodologie générale, ce chapitre va lister et définir de manière non exhaustive les pathologies visuelles recherchées lors d'une inspection d'un bâtiment existant. Bien que certaines familles de pathologies existent sous des formes différentes à l'intérieur des édifices ou dans les parois, ce travail n'aborde que les défauts visibles à l'œil nu.

Dans le domaine de la construction, on définit une pathologie comme une « maladie » structurelle qui, si elle s'aggrave, peut mener à des sinistres ou à la ruine des ouvrages.

1.3.1. Fissures

L'apparition de fissures sur une façade peut prendre plusieurs formes et est souvent signe d'un autre problème qui peut impacter la structure du bâtiment sur le long terme. Les petites fissures sont souvent superficielles mais nécessitent une intervention pour éviter l'infiltration d'eau à l'intérieur du mur. Dans le cas de plus grosses fissures installées au fil du temps, appelées "lézardes", la stabilité même du mur peut être compromise.

1.3.1.1. Fissures horizontales

Les fissures horizontales (fig. 1a) sont le plus souvent observées au pied d'une charpente, le long d'une dalle de sol ou le long d'un élément structurel qui aurait changé de volume. Les structures en béton, blocs ou dalles, sont sujettes à du retrait (raccourcissement) durant le séchage. Elles sont également susceptibles de changer de volume si elles subissent des variations de température trop importantes. Une maçonnerie de parement reliée de manière rigide à cet élément structurel (dalle de sol ou de toiture, poutre de ceinture) pourrait occasionner des fissures horizontales à l'endroit le plus sensible, qui n'est pas forcément aligné avec le bord intérieur de l'élément structurel. Dans les angles, les fissures peuvent se ramifier en d'autres fissures verticales et créer un léger désaffleurement de la façade.

Dans le même ordre d'idée, on peut observer des gonflements et des fissurations de la maçonnerie au niveau des linteaux en acier encastrés. Si l'acier se met à rouiller à cause d'une infiltration d'eau ou par capillarité, son volume va augmenter considérablement et disloquer la maçonnerie environnante (De Bie et al., 1996).

1.3.1.2. Fissures verticales

À l'angle de deux façades attenantes, il est possible d'observer de longues fissures verticales (fig. 1b) après des dilatations elles-mêmes causées par des variations de température et d'humidité dans la maçonnerie. Les fissures les plus graves sont généralement situées entre des façades perpendiculaires orientées au sud et à l'est. Puisqu'elles ne sont pas exposées aux mêmes conditions climatiques, leurs déformations sont différentes et occasionnent des fissures importantes.

À une plus petite échelle, les sulfates présents dans des briques exposées à la pluie peuvent réagir avec l'humidité et former un sel expansif, appelé sel de Candlot, qui fait gonfler la brique et la fissure. Pour des briques qui sont protégées, l'évaporation d'eau va précipiter la création de gypse, qui fera également gonfler la brique. Si l'eau de pluie pénètre dans les fissures existantes, le phénomène peut s'aggraver (De Bie et al., 1996). Selon Philipparie (2019), les briques utilisées aujourd'hui ne souffrent plus de ce genre de défauts du passé, comme la présence de points de chaux qui peut conduire au gonflement du matériau et à sa fissuration.

Tout comme pour les fissures horizontales, la dilatation et le retrait de la maçonnerie peut créer des contraintes causées par des variations de température. Ces contraintes peuvent entrainer des fissures qui apparaissent près des baies de la façade ou dans un angle, là où la maçonnerie est interrompue (De Bie et al., 1996).

1.3.1.3. Fissures en escalier

Les fissures en escaliers en angle de façade (fig. 1c) sont souvent la conséquence d'un tassement des fondations du bâtiment et indiquent donc parfois une instabilité structurelle. Ce phénomène est majoritairement observable sur les sols argileux ou limoneux et lorsque les fondations sont trop superficielles (Philipparie, 2019). La proximité d'arbres adultes peut amplifier ce phénomène en abaissant encore plus le niveau d'eau dans le sol et en faisant subir au sol argileux des retraits qui ne sont pas uniformes. Ce type de fissure peut aller jusqu'à plusieurs dizaines de millimètres de largeur et s'accompagner de désaffleurements (De Bie et al., 1996).

1.3.1.4. Fissures en moustache

Elles apparaissent aux angles des fenêtres ou des portes. Le plus souvent, la première se forme dans le coin supérieur, suivie d'une autre dans le coin inférieur de l'encadrement. Elles traduisent souvent une structure insuffisante pour supporter les menuiseries en place.

1.3.2. Écaillage

L'écaillage est observable sur le béton, la brique, la pierre naturelle ou même sur la peinture (fig. 1d). Il se réfère à la rupture en surface d'une partie du matériau, qui se détache du reste de la masse. Ceci peut être causé par l'érosion ou l'altération naturelle des matériaux mais également en cas de frottements continus ou d'impacts subits par l'élément. Dans des cas extrêmes, l'écaillage peut exposer la structure de l'édifice.

Les anciennes briques peuvent présenter des occlusions de chaux qui créent une augmentation de volume lorsque la brique est exposée aux conditions climatiques. Ces gonflements localisés peuvent être la cause d'expulsion d'écailles de brique. On peut aussi assister à un écaillage des briques à cause des sulfates, provenant de la brique si sa cuisson n'a pas été suffisante, ou absorbés depuis l'extérieur lors de pluies acides ou lors de remontées capillaires (De Bie et al., 1996).

Sur les joints, on peut constater qu'une pellicule superficielle se détache après une période de gel. Ceci est dû à l'exécution du lissage du joint, qui crée une pellicule riche en ciment superposée à une couche appauvrie en ciment. Un lissage tardif laisse le temps à l'eau de s'accumuler dans la partie plus sableuse, ce qui provoque l'expulsion de la 1ère pellicule en cas de gel (De Bie et al., 1996).

1.3.3. Efflorescence

L'efflorescence est un phénomène par lequel des sels solubles se déposent à la surface des matériaux poreux comme la brique, le béton et certaines pierres lorsque de l'eau infiltrée s'évapore, formant une fine couche blanche ou poudrée issue des sels transportés par capillarité (fig. 1e). Ces sels peuvent être signe d'humidité dans les murs et sont capables d'affaiblir les capacités structurelles des matériaux. On distingue plusieurs types d'efflorescence.

L'efflorescence primaire apparait sur les briques, le béton et les joints de mortiers peu après la construction. Elle résulte de l'eau utilisée dans le matériau qui dissout les sels solubles et les transporte à la surface en séchant. Elle ne présente aucun danger et est juste une gêne esthétique. Les sels restent très solubles et disparaissent d'eux-mêmes sous l'action de la pluie (De Bie et al., 1996).

L'efflorescence secondaire apparaît plus tard, et est due à des infiltrations d'eau ou à l'humidité extérieure, qui dissolvent les sels déjà présents ou nouvellement introduits (De Bie et al., 1996).

1.3.4. Dégâts de plantes grimpantes

Souvent un souci esthétique, la présence de végétation non-contrôlée (fig. 1f) peut affecter structurellement l'édifice si les racines parviennent à rentrer entre les joints de maçonnerie. Les plantes grimpantes les plus courantes sont la vigne et le lierre. La vigne n'abîme généralement pas les façades car elle s'y accroche à l'aide de petites ventouses et est reconnaissable par sa perte de feuilles en hiver. Le lierre s'accroche à l'aide de ses racines, qui peuvent pénétrer dans la façade par des fentes et autres discontinuités, abimant ainsi les joints de mortier. Contrairement aux joints de ciments, les joints de mortier à la chaux peuvent perdre leur cohésion et se décrocher (De Bie et al., 1996).

1.3.5. Salissures et décoloration

Causé par la pollution, l'exposition au soleil, l'oxydation, etc. les salissures sont surtout un souci esthétique (fig. 1g). On distingue cependant les salissures sèches, qui ne font que se déposer, des salissures grasses qui se collent au support pour faire corps avec lui. Dans certains cas, la présence de ces particules engendre des réactions chimiques, pouvant alors impacter la stabilité du support (Carrie & Morrel, 1975).

Un autre phénomène, une décoloration peut apparaitre lorsqu'on utilise des briques en argile contenant de la pyrite cuites dans un environnement pauvre en oxygène. Ces briques contiennent des éléments ferreux qui sont dégorgés par la pluie et créent des taches de rouille qui sont peu ou pas solubles dans l'eau, à l'inverse des sels efflorescents.

1.3.6. Autres

L'eau et l'humidité sont souvent la cause de certaines formes de fissures, d'écaillage et surtout d'efflorescence, mais une infiltration d'eau dans la façade peut aussi causer des taches (fig. 1h) et favoriser la prolifération de champignons, de mousses, de lichens ou d'autres matières organiques comme les déjections de pigeons. Ces taches peuvent prendre plusieurs formes selon le type de paroi et peuvent se manifester en extérieur ou même causer des dommages aux finitions intérieurs si les matériaux sont suffisamment poreux

Avec le temps et à cause d'autres désordres, certains éléments d'une façade peuvent se détacher, ce qui affaiblit la stabilité des éléments adjacents s'ils ne sont plus supportés correctement, comme des joints de maçonnerie, des briques ou d'autres éléments de parement. Finalement, le vandalisme et la présence de graffitis ne posent pas de risques pour le bâtiment mais peuvent aussi déprécier sa valeur esthétique.

1.3.7. Exemples de pathologies

Fig. 1 – Exemples de pathologies du bâtiment

2. Inspection et documentation de l'existant

Avant d'envisager toute intervention de restauration ou de rénovation sur un bâtiment existant, il est nécessaire de réaliser des relevés et une inspection de celui-ci. Les inspections peuvent dans certains cas être restreintes à une pathologie spécifique ou à un élément de construction particulier, plutôt qu'à l'ensemble de l'édifice. Puisque ce type de service est souvent long et coûteux, les inspections complètes sont parfois ignorées au profit de rapports ciblés sur des éléments ou des défauts particuliers. Ces rapports sont complétés de documents graphiques réalisés lors du relevé pour documenter précisément les résultats de l'inspection.

L'objectif des relevés est donc de fournir une représentation graphique de l'état actuel de l'édifice, datée et fidèle à sa forme. Selon l'âge du bâtiment et les transformations subies au fil du temps, il se peut que la documentation n'existe pas ou soit obsolète. Ces relevés servent ensuite de base aux experts qui, en inspectant visuellement l'état du bâtiment, déterminent si des travaux sont nécessaires ou si d'autres techniques d'inspection non destructives sont utiles pour déterminer l'état structurel du bâtiment (Saint-Aubin, 1992).

L'approche traditionnelle du relevé des bâtiments architecturaux consiste à utiliser des instruments de mesure directs (relevé direct) pour permettre de restituer les dimensions de l'objet mesuré. Le relevé est effectué à l'aide d'instruments de mesure placés le long des surfaces des objets à reproduire ou à proximité, afin de mesurer les longueurs, définir les alignements et noter sur un support graphique les dimensions des éléments significatifs (Croce, 2022). Les avancées dans les domaines de la conception et de l'informatique permettent de passer des mesures directes de la réalité à un relevé indirect, basé sur l'utilisation de capteurs actifs ou passifs. Parmi les techniques les plus couramment utilisées, on retrouve la lasergrammétrie et la photogrammétrie. (Pierrot-Deseilligny et al., 2011).

Si les techniques de relevés directs sont aujourd'hui moins utilisées, elles n'en sont pas pour autant obsolètes. Elles conservent pour avantage leur coût, leur facilité de mise en place et leur accessibilité. Elles peuvent encore être utilisées pour des relevés de petite envergure ou lorsque des équipements plus modernes ne sont pas disponibles.

2.1. Méthodes et technologies actuelles

2.1.1. Acquisition des données

2.1.1.1. Relevés directs

On distingue trois étapes dans ce type de relevé : l'esquisse, le relevé et le dessin.

L'esquisse consiste à se familiariser avec le lieu en le parcourant. Cette étape est souvent accompagnée de croquis dessinés à main levée sur lesquels on indique les formes principales du bâtiment. Cela sert ensuite à préparer les étapes suivantes en indiquant les éléments architecturaux à mettre en évidence, le parcours à suivre pour balayer l'entièreté du site ou encore les outils et méthodes de relevé à employer.

Avant l'apparition des instruments optiques comme le théodolite, on utilisait la technique de la triangulation, qui permet de déterminer la position d'un point en mesurant les angles entre ce point et d'autres points de référence dont la position est déjà connue. En général, les outils utilisés étaient un double-décamètre, un mètre, un fil à plomb, du matériel de dessin pour annoter les mesures ainsi qu'un télémètre laser, plus récemment (fig. 2). Le théodolite (fig. 3) est un appareil optique capable de mesurer les angles horizontaux et verticaux entre deux points cibles. Ces mesures d'angles servent ensuite à mesurer par triangulation la distance entre les points. Bien que l'utilisation de ces techniques soit limitée aux formes géométriques non arrondies, elle reste une méthode encore courante dans les méthodes de relevé aujourd'hui (Magri-Djenane et al., 2011).

Arrivera ensuite le tachéomètre qui, dans sa forme la plus simple, permettra de mesurer la distance entre deux points à l'aide d'un fil stadimétrique. Aujourd'hui, un tachéomètre numérique - ou station totale - (fig. 4) est capable de mesurer automatiquement les distances entre les points ciblés et permet de travailler seul. Si ces différentes techniques sont relativement simples à mettre en œuvre, la précision des documents techniques dépend de l'opérateur car celui-ci simplifie nécessairement le bâtiment et les imprécisions peuvent s'accumuler. Par conséquent, bien que le relevé manuel puisse être avantageux en termes de simplicité et de coût, le recours aux nouvelles technologies 3D offre souvent une rentabilité en termes de temps et de qualité de travail grâce à l'automatisation de ces procédés. (Héno & Chandelier, 2014, p. 20-21) (Saint-Aubin, 1992).

Fig. 2 – Outils de relevé manuel
(Saint-Aubin, 1992)

Fig. 3 – Théodolite
(Saint-Aubin, 1992)

Fig. 4 – Station totale Leica Icon icb70

Peu importe l'utilisation d'outils manuels, optiques ou numériques, les données obtenues après le relevé sont retranscrites à la main ou sur ordinateur sous forme de plans, coupes et élévations qui seront utilisés lors d'une future intervention.

2.1.1.2. Relevés numériques indirects

Depuis les années 1990, la collecte de données topographiques a évolué, passant de l'enregistrement manuel au traitement des données directement sur le terrain. Avant, les données étaient inscrites dans un carnet de terrain standard pour être ensuite ajustées, traitées et tracées au bureau. Les collecteurs de données modernes peuvent effectuer toutes ces fonctions sur le terrain. Ils sont soit intégrés à une station totale, soit à des instruments séparés. Un collecteur de données séparé (indépendant) présente l'avantage de pouvoir être combiné avec une variété d'instruments de relevé, tels que la station totale, le niveau numérique et le récepteur GPS. Les fichiers des collecteurs de données de terrain sont téléchargés sur une plateforme PC au bureau où les données de terrain peuvent être éditées et modifiées pour être directement intégrées dans un logiciel CADD ou SIG pour des utilisations ultérieures de conception et d'analyse (Saint-Aubin, 1992).

Aujourd'hui, ces technologies sont de plus en plus utilisées pour documenter le patrimoine, offrant des performances supérieures aux instruments de relevé directs en termes de précision, de manipulation, d'exploitation et de diffusion des données. Il est cependant nécessaire de souligner que cette transition numérique ne remplace en aucun cas les études sur site. Il est essentiel de ne pas négliger les observations interprétatives recueillies sur le terrain. Au contraire, ces outils permettent une acquisition tridimensionnelle des données géométriques et chromatiques qui, une fois analysées et traitées informatiquement, peuvent servir de support à l'analyse in situ des bâtiments. Elles peuvent également nourrir certaines réflexions en dehors du site lorsque celui-ci n'est plus accessible (Hallot et al., 2022).

Dans le cas de la documentation du patrimoine existant, on observe donc deux approches distinctes mais complémentaires, la lasergrammétrie et la photogrammétrie. Ces techniques offrent un potentiel prometteur lorsqu'elles sont utilisées en complément des modèles de détection automatique et seront abordées en détail. La capacité à localiser automatiquement une pathologie sur un modèle 3D pourrait considérablement simplifier le travail des experts. Cette technologie ouvre également la voie à des approches novatrices, comme l'utilisation de la réalité virtuelle, où l'investigateur pourrait réaliser son analyse à distance, depuis son bureau, avec les pathologies déjà identifiées et marquées sur le modèle (Yang et al., 2023a).

Bien que d'autres technologies d'acquisition 3D existent, elles ne seront pas abordées dans ce travail puisqu'elles n'ont pas ou peu d'applications dans la détection de pathologies à grande échelle. Citons par exemple le scanner à lumière structurée et le scanner avec contact, qui ont cependant l'avantage de pouvoir scanner les surfaces réfléchissantes, ce dont les scanners laser sont incapables. Avec une portée limitée à quelques mètres, le scanner laser à triangulation optique est également peu adapté à une inspection de grande échelle.

Lasergrammétrie

Parmi les technologies de numérisation 3D, le scanner laser 3D figure parmi les outils les plus couramment utilisés. Son principal avantage réside dans sa capacité à capturer une densité de points extrêmement élevée, atteignant plusieurs milliers de mesures par centimètre carré (Héno & Chandelier, 2014). Cette précision garantit un niveau de détail exceptionnel, ce qui en fait un choix privilégié pour la conservation du patrimoine, où chaque aspect matériel doit être minutieusement documenté et modélisé afin de représenter fidèlement l'état du bâtiment. De plus, le scanner laser 3D est une technologie non destructive : son fonctionnement sans contact réduit les risques de dommages lors des manipulations. Les scanners équipés de capteurs LiDAR émettent un faisceau lumineux qui balaie l'espace horizontalement et verticalement jusqu'à rencontrer des obstacles (sol, murs, mobilier, machines, etc.). La distance à chaque obstacle est mesurée, permettant de positionner des points dans l'espace pour créer une représentation numérique.

Toutefois, cette technologie présente certaines limites. Les surfaces brillantes, noires ou transparentes réfléchissent mal la lumière, ce qui peut entraîner des erreurs ou une capture imprécise des données. De plus, la lasergrammétrie ne détecte pas la couleur des objets scannés. Pour pallier cette limitation,

certains modèles plus récents intègrent désormais une caméra qui attribue une couleur à chaque point scanné. Alternativement, la photogrammétrie peut être utilisée parallèlement au relevé pour fournir une texture aux données lors de leur traitement.

Il existe plusieurs types de scanners laser 3D, chacun ayant ses spécificités en termes de portée, de précision et de vitesse d'acquisition, ce qui les rend performants dans certains domaines mais moins adaptés dans d'autres.

- **Scanner à temps de vol** : Ce type de scanner mesure le temps nécessaire au rayon laser pour effectuer un aller-retour entre l'appareil et l'objet. Grâce à sa précision acceptable et à sa très grande portée, il est particulièrement adapté aux relevés de grande envergure, tels que les paysages naturels ou les grands édifices.
- **Scanner à décalage de phase** : Contrairement au précédent, ce scanner émet un signal en continu et calcule la distance en comparant les phases de l'onde émise avec celles de l'onde réfléchie. Sa vitesse d'acquisition élevée en fait un outil idéal pour des relevés détaillés, comme ceux des bâtiments, des intérieurs ou des structures complexes.
- **Scanner hybride** : Ce type combine les avantages des technologies à temps de vol et à décalage de phase, offrant ainsi un compromis entre grande portée et rapidité d'acquisition, ce qui le rend polyvalent pour diverses applications.

Tous ces scanners fixes ont cependant pour défaut :qu'un objet ne peut généralement pas être numérisé à partir d'une seule station. Ce problème peut être dû au champ de vision limité du scanner, la grande taille de l'objet, à la complexité de son architecture ou à la présence d'obstacles empêchant une visée directe. Dans ce cas, il est nécessaire d'utiliser plusieurs points de vue pour couvrir toutes les faces de l'objet. Il faut donc comme pour un relevé direct planifier le chantier en amont et établir une phase de reconnaissance autour du site afin d'obtenir le maximum de données en utilisant le minimum de points de prise de mesure en une position donnée. On utilisera aussi des drones équipés des mêmes technologies de lasers afin de réaliser le scan pour la partie supérieure du bâtiment (Landes et al., 2011).

Bien que très précises, les technologies laser nécessitent donc beaucoup de préparation en amont du relevé pour prédéfinir la position de chaque station afin de couvrir l'ensemble de l'édifice. Ces dernières années, les méthodes d'acquisition en temps réel se sont fortement développées et permettent d'avoir des résultats plus rapides. Les technologies SLAM et Time of Flight en sont de parfaits exemples.

Le **SLAM (Simultaneous Localization and Mapping)** est une technologie initialement développée dans le domaine de la robotique. C'est un processus par lequel un robot mobile peut construire une carte de son environnement tout en utilisant cette même carte pour se localiser (Durrant-Whyte et al., 2006). Il présente de nombreux avantages, notamment une rapidité d'acquisition des données 10 à 15 fois plus importante qu'un relevé effectué avec un scanner laser 3D classique monté sur trépied. Il requiert également moins de temps de préparation. Le SLAM peut être utilisé aussi bien en intérieur qu'en extérieur et est capable de capturer différentes textures et couleurs. De plus, un élément mal scanné nécessitera simplement que l'opérateur de l'appareil repasse à proximité de la cible pour corriger l'erreur. Ceci permet donc à des scanners de se situer en temps réel dans leur environnement et offre la possibilité d'utiliser des scanners mobiles, qui peuvent être beaucoup plus accessibles en termes de prix (Barba et al., 2021) (Tanduo et al., 2023). Les scanners mobiles comme le GeoSlam Zeb1 (fig. 5) sont généralement moins précis mais à des prix très abordables, tandis que les scanners NavVis (fig. 6)visent à égaler les scanners fixes en termes de précision, ce qui les rend tout aussi chers.

Fig. 5 – Scanner GeoSlam Zeb1

Fig. 6 – Scanner NavVis VLX

Fig. 7 – Caméra TOF Lucid Helios2+

Une **caméra Time-of-Flight (ToF)** est un capteur de vision 3D qui mesure la distance entre la caméra et les objets en calculant le temps que met une lumière infrarouge à revenir après avoir été réfléchie par les objets (fig. 7). Elle fonctionne en émettant un faisceau de lumière infrarouge modulée. Chaque pixel du capteur enregistre le décalage de phase entre la lumière émise et la lumière réfléchie, comme pour le scanner laser homonyme (He & Chen, 2019). Cette technologie a l'avantage d'acquérir les données en temps réel en plus d'être compacte et relativement fiable. Elle reste cependant moins précise que les autres méthodes et est surtout adaptée à des scènes dynamiques.

Photogrammétrie

Une photogrammétrie peut être utilisée dans toutes les situations où un objet à mesurer pourrait être photographié. Son objectif principal est la reconstruction en 3D de l'objet sous forme numérique ou graphique (Luhmann et al., 2019). La prise des données peut se faire par caméra, ou par drone lorsque l'endroit est peu accessible (par exemple une toiture). Des perches peuvent servir à atteindre des endroits intermédiaires.

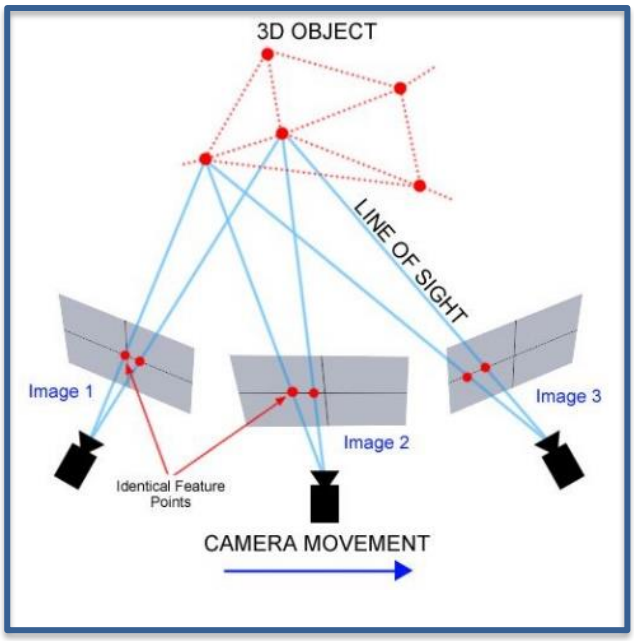


Fig. 8 – Principe de la reconstruction 3D d'un objet par photogrammétrie (Scherer et al., 2019)

Les principaux avantages de la photogrammétrie sont sa précision à courte portée et sa rapidité d'acquisition, car elle repose sur la prise de photographies. Cette technique est applicable à tous types d'objets et le coût du matériel est plus attractif que celui des scanners 3D. De plus, elle permet de texturer les modèles 3D sur base des photos prises, contrairement aux technologies lasers qui ne créent que des nuages de points, coloriés à l'aide d'une caméra ou grisés pour les plus classiques.

La précision du modèle dépend beaucoup de la qualité de la photo et de sa distance par rapport à l'objet cible. Il est donc important d'avoir un environnement propice à la prise de photographies. Dans le cas d'une détection de pathologies, on peut s'approcher des dégâts observés pour les détailler correctement si nécessaire, et ainsi ne pas avoir le même degré de détail partout sur le modèle, ce qui peut augmenter le temps de prise de vue et de traitement.

Afin de s'assurer de la qualité du modèle lors du traitement des photos prises, il est conseillé d'utiliser des points de contrôle. Ces points peuvent prendre la forme d'une échelle qui servira de repère pour mettre le modèle à l'échelle ou bien d'une cible dont la position est connue à l'aide d'un théodolite ou d'une station totale. Ces points de contrôle doivent donc être présents sur un maximum de photos et placés stratégiquement. Il est recommandé de superposer les images entre 66 % et 80 % pour garantir un appareillage précis lors du traitement des données (Hallot, et al., 2022).

Il est conseillé de modifier manuellement les réglages de l'appareil photo utilisé lors de l'acquisition des images pour pouvoir prendre en compte les ISO, l'ouverture, la vitesse d'obturation et la balance des blancs. Les trois premiers paramètres forment ce qu'on appelle l'exposition, qui exprime la quantité de lumière enregistrée par le capteur de l'appareil (Hallot, et al., 2022).

La valeur ISO détermine la sensibilité du capteur à la lumière. Elle peut varier de 100 à 51200 selon les appareils. Un ISO élevé est généralement utilisé en cas de faible luminosité, mais génère du grain, diminuant la qualité de l'image. En photogrammétrie, il est donc conseillé de laisser ce paramètre à 100.

Fig. 9 – Influence des ISO sur la qualité de l'image

L'ouverture du diaphragme exerce une influence sur la luminosité de la photo mais également sur sa profondeur de champ. Plus l'ouverture est importante, plus la lumière pénètre jusqu'au capteur et plus la profondeur de champ diminue. En photogrammétrie, il est conseillé d'avoir une grande profondeur de champ.

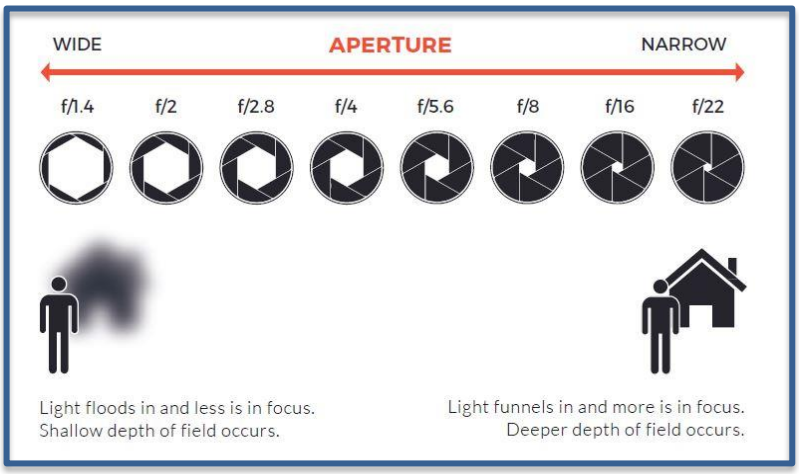


Fig. 10 – Influence de l'ouverture sur la profondeur de champ

La vitesse d'obturation désigne le temps que l'appareil prend pour capturer l'image, et donc la quantité de lumière qui va pénétrer jusqu'au capteur. Si l'appareil et l'objet sont fixes, on peut augmenter le temps de pose. S'il y a du mouvement, l'image risque d'être floue (Hallot, et al., 2022).

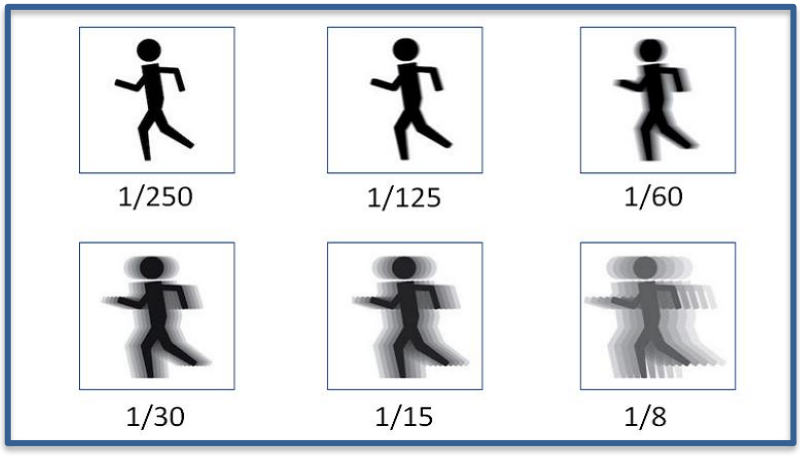


Fig. 11 – Influence de la vitesse d'obturation sur la netteté d'un objet (Georjon, 2008)

L'exposition est directement liée à ces trois paramètres, ce qui signifie qu'il est possible d'obtenir la même exposition si on augmente les ISO et diminue le temps de pose par exemple. Sur l'appareil photo, une mesure d'exposition à 0 indique une quantité de lumière présumée correcte. Une surexposition donne une photo trop claire tandis qu'une photo sous-exposée est trop sombre (Hallot, et al., 2022).

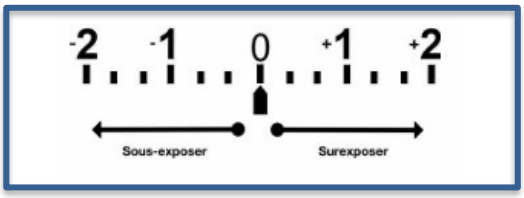


Fig. 12 – Mesure de l'exposition

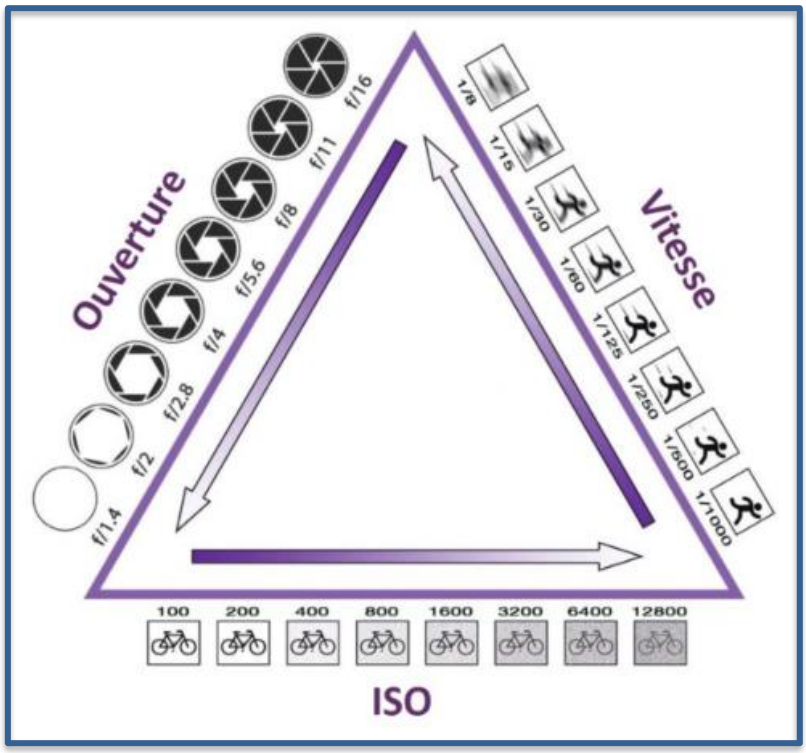


Fig. 13 – Triangle de l'exposition

La balance des blancs permet de régler le capteur de l'appareil selon l'éclairage ambiant afin de restituer les couleurs de manière fidèle. Un réglage non adapté de la balance des blancs risque de ne pas représenter les couleurs réelles de l'objet photographié sur la photo.

Le format des images peut être en JPEG ou RAW. Le premier compresse les informations et simplifie les détails en échange d'une taille de fichier réduite. Le second conserve tous les détails et est donc très volumineux, en plus d'exiger des programmes spécifiques pour être ouvert, contrairement au JPEG.

En résumé, **la stratégie d'acquisition** d'une photogrammétrie doit respecter les points suivants :

- Il est essentiel de planifier la numérisation avant toute prise de vue en déterminant les positions de l'appareil photo et en s'assurant d'un recouvrement suffisant entre les images pour permettre l'identification des points communs nécessaires à la reconstruction 3D.
- Plusieurs rotations autour du sujet sont souvent nécessaires pour obtenir une couverture complète, qu'il s'agisse d'un bâtiment ou d'un objet.
- Il est important de maintenir les paramètres de l'appareil constants tout au long de la session pour une uniformité des données.
- L'appareil photo doit être déplacé pour varier les angles de vue, plutôt que de simplement modifier son orientation.
- Il est également recommandé de privilégier un éclairage diffus, naturel ou artificiel, pour éviter les ombres marquées et les zones surexposées (Hallot, et al., 2022).

2.1.2. Traitement des données

2.1.2.1. Traitement des images en nuage de points

Après le relevé sur terrain, les données brutes comme les nuages de points et les images doivent être traitées pour être utilisables ultérieurement. Si on utilise de la photogrammétrie, la première étape est d'appareiller les images entre elles. Cette reconstruction 3D de l'objet se fait grâce à la superposition des différentes photos réalisées, dans lesquelles vont être désigné des points de liaisons qui correspondent aux pixels homologues entre chaque image. Il est donc important de superposer suffisamment les images capturées sur terrain, sans quoi ce processus ne se fera pas correctement. Ce procédé est très long selon la quantité de photos prises lorsqu'il est fait manuellement. Cependant, il est depuis 1999 automatisé par l'algorithme SIFT, qui détecte les points correspondants entre deux images.

En se basant sur les positions des points correspondants sur différentes images, il est possible de déterminer des informations sur la position 3D de ces points ainsi que sur les paramètres de l'appareil utilisé pour capturer les photographies (paramètres optiques, localisation et orientation de l'appareil dans l'espace).

Ce processus d'optimisation est connu sous le nom de SFM (*Structure From Motion*). Il génère un nuage de points de faible densité répartis sur les surfaces photographiées, ainsi qu'une estimation des différentes positions, orientations et paramètres optiques de l'appareil photo (Mason, s.d.) (Dubois et al, 2017).

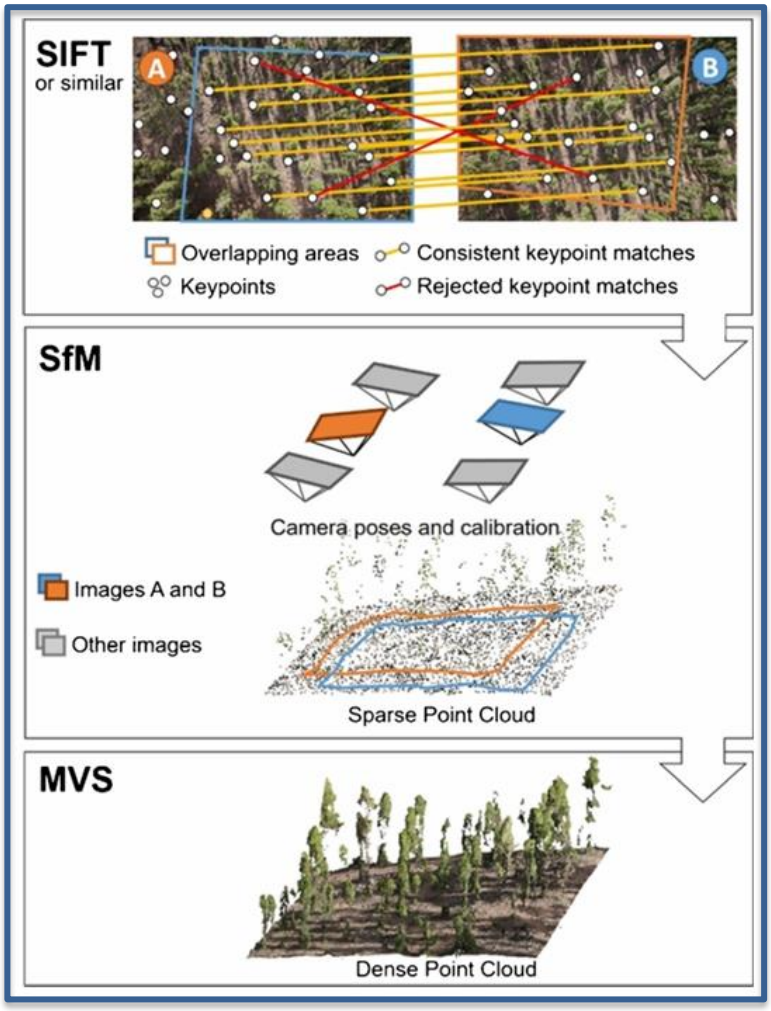


Fig. 14 – Les trois étapes clés d'un flux de travail SfM-MVS (Iglhaut et al., 2019)

Une fois les images orientées, un deuxième processus d'optimisation permet d'affiner ce nuage de points et de produire une reconstruction beaucoup plus dense, en se basant sur la calibration des caméras déterminée lors de la phase SFM. Cette seconde étape, qui repose sur l'identification de correspondances pixel par pixel, est appelée reconstruction dense (*Dense image matching*) et se fait à l'aide de l'algorithme MVS (*Multiview Stereo*). En résumé, les techniques SFM et MVS sont complémentaires et reposent sur des hypothèses différentes ; combinées, elles permettent de reconstruire en trois dimensions et avec une grande précision l'objet étudié à partir de simples photographies (Dubois et al, 2017).

Cependant, la phase de calcul pour créer la surface 3D peut être influencée par le nombre et la qualité des photographies, ce qui peut augmenter le temps de traitement et nécessiter l'utilisation d'ordinateurs très puissants. Un autre inconvénient est que l'objet n'est pas à l'échelle, ce qui nécessite un calibrage de l'image à l'aide des points de contrôle installés lors du relevé. Dans certains cas, il peut être préférable de réduire la densité du maillage si l'on souhaite reproduire une fracture d'un objet ou s'il n'est pas nécessaire d'obtenir une forte précision sur l'ensemble de l'édifice (Hallot, et al., 2022).

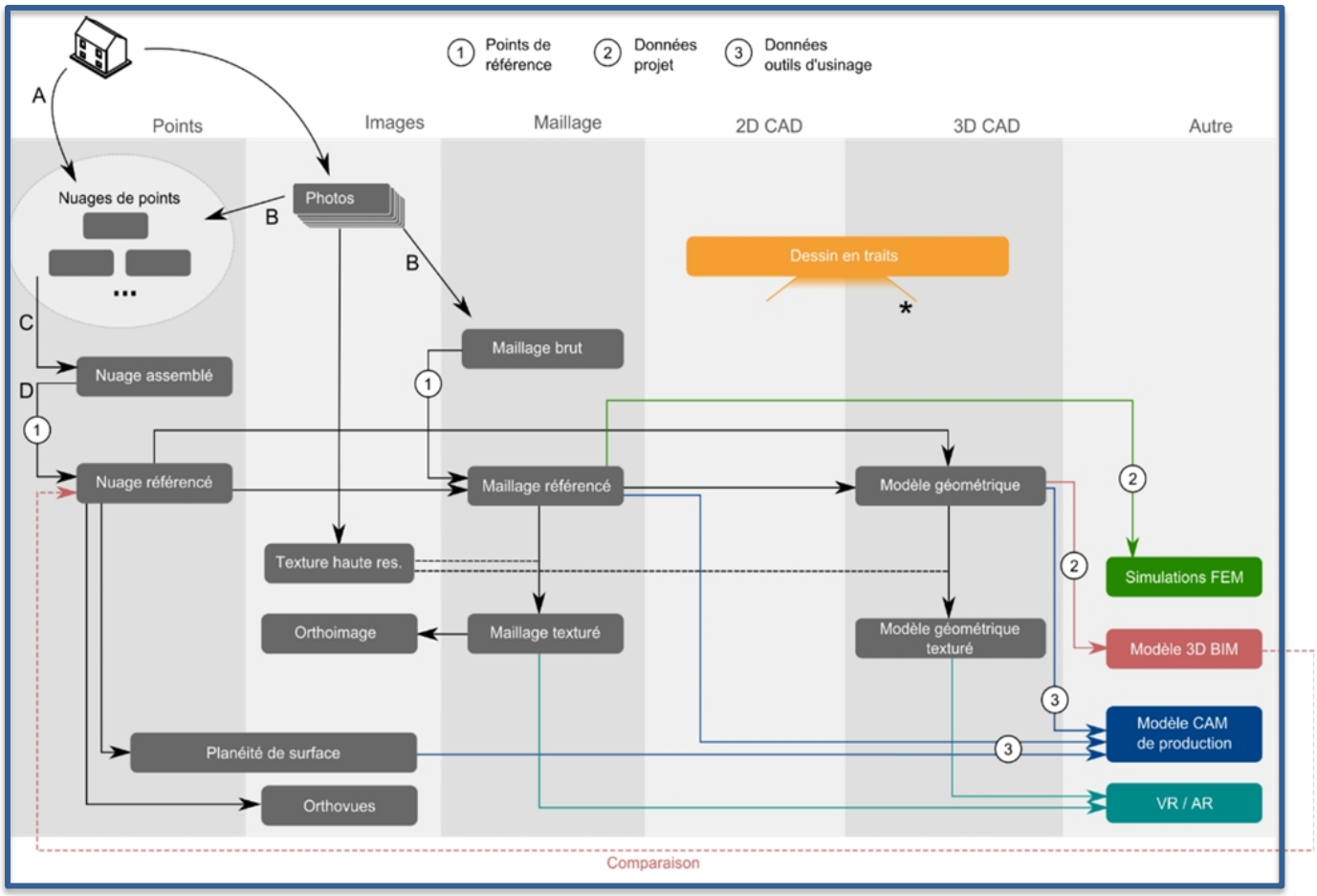


Fig. 15 – Le cheminement de l'information 3D (Dubois et al., 2017)

2.1.2.2. Assemblage des nuages de points

L'utilisation de plusieurs stations laser et de photogrammétrie nécessite de regrouper leurs nuages de points respectifs dans un même système de coordonnées. Cette opération est appelée "consolidation" ou "assemblage de nuages". Si le système de coordonnées choisi est un système de projection cartographique, l'opération est alors appelée "géoréférencement". Dans le cas d'un assemblage non référencé, la consolidation peut être réalisée sur base de zones de recouvrement entre les différents nuages. Cela nécessite de s'assurer d'un recouvrement suffisant entre les différentes stations et la fiabilité de l'assemblage dépend de la robustesse des algorithmes utilisés. On utilisera plus fréquemment les points de contrôle placés lors du relevé et dont la position est connue grâce aux stations totales pour géo-référencer le modèle. Si le procédé se fait presque automatiquement, il reste important pour l'opérateur de s'assurer de la qualité de l'assemblage des nuages afin de fusionner l'ensemble en un fichier commun (Dubois et al, 2017).

Le bâtiment numérisé se présente cependant toujours sous forme d'un nuage de points. On en obtient simplement une vision dans son ensemble. Ce nuage de points se caractérise par une grande quantité de points désorganisés, une densité inégale de points et la présence de bruit.

2.1.2.3. Segmentation

Si l'étape de consolidation du nuage de points permet de réaliser un premier nettoyage des points non-pertinents, les points restants ne possèdent aucune information sémantique intrinsèque, ce qui signifie qu'ils ne sont pas regroupés par attributs (comme l'orientation, l'élévation, ou l'identité). L'opérateur doit regrouper les points appartenant à un même élément architectural (par exemple, mur, colonne, châssis) et leur attribuer cette appartenance via le processus de segmentation. Cette opération est principalement manuelle et longue à réaliser, mais elle est souvent inévitable car elle permet, une fois les éléments classés selon leurs différents attributs, d'exporter certaines sections du modèle qui sont pertinentes au travail associé (Dubois et al, 2017).

Il est néanmoins possible d'automatiser ce traitement en partie, avec la segmentation analytique ou géométrique, basée sur les propriétés géométriques et les informations directement calculées à partir des données récupérées. Cela permet de réaliser une première segmentation sur base de la proximité des points, ou sur l'orientation qu'ils présentent en groupes, qui pourra être affinée par un expert (Billen et al., 2018)

2.1.2.4. Maillage

Également connue sous le nom de facettisation ou de polygonisation, cette étape consiste à lier les points pour en faire une surface. Certains opérateurs n'ont besoin que d'un nuage de points segmenté et nettoyé mais il est parfois souhaitable de le transformer en un maillage polygonal texturé dans le but de visualiser au mieux le modèle 3D. Que ce soit pour faciliter l'accessibilité à un utilisateur non initié et n'ayant pas accès aux machines nécessaires pour manipuler le modèle ou pour y appliquer des textures permettant une meilleure compréhension de l'objet. Ces textures peuvent être appliquées manuellement à l'aide des éléments segmentés précédemment ou automatiquement alignées avec les textures réelles issues d'une photogrammétrie (Dubois et al, 2017).

On distingue quatre étapes :

- Tout d'abord, les nuages de points sont prétraités manuellement ou automatiquement pour ne conserver que les données nécessaires.

- Ensuite, les sommets des points sont reliés par des polygones, principalement des triangles. L'opérateur peut ajuster les dimensions des triangles selon les besoins. Un maillage plus dense produit un modèle 3D plus détaillé, mais augmente également le volume de données.

- On utilise ensuite un shader, algorithme qui lisse les surfaces pour donner une impression de courbe, modifiant ainsi l'affichage sans altérer les données du fichier. C'est à ce moment que les textures peuvent être appliquées. Pour une application réaliste, l'image doit être correctement aplatie avant d'être projetée sur le modèle.

- Finalement, le maillage est optimisé et la densité des points est ajustée en fonction de l'utilisation prévue du modèle. C'est une étape cruciale qui inclut aussi le nettoyage des parasites et la fermeture des trous, ce qui peut être réalisé manuellement ou automatiquement (Heynen, 2020, p.53-54).

2.1.3. Synthèse des méthodes de relevé actuelles

La photogrammétrie et la lasergrammétrie sont deux techniques de mesure qui correspondent à l'échelle du patrimoine bâti, nécessitant une expertise pour garantir des prises de vues adéquates et des données exploitables. Toutefois, chacune présente des limites lorsqu'elle est utilisée seule, rendant difficile une modélisation globale complète. Une solution consiste à combiner ces deux procédés afin de pallier les lacunes dimensionnelles et maximiser la complémentarité de leurs caractéristiques. D'un côté, La photogrammétrie fournit des images riches en informations visuelles, tandis que la lasergrammétrie génère des données dimensionnelles précises mais parfois ambiguës avant la segmentation. Leur combinaison permet une meilleure interprétation des données (Alby, 2006).

De plus, la lasergrammétrie bénéficie d'une automatisation avancée, tandis que la photogrammétrie exige une intervention humaine plus importante. En les combinant, on peut réduire le travail manuel et accroître l'automatisation. En intégrant les photographies dans le nuage de points laser, il devient possible de combiner la précision des lasers et la richesse visuelle des photos. Les types de données produits sont également complémentaires, notamment grâce à l'automatisation des nuages de points en photogrammétrie, avec les algorithmes comme SIFT ou SfM (Alby, 2006).

En bref, la convergence technologique entre photogrammétrie et lasergrammétrie offre une opportunité de combiner leurs forces pour réduire les limites individuelles et favoriser une approche plus automatisée et précise de la documentation architecturale. Une fois le modèle finalisé, il peut être utilisé dans un objectif de conservation, en gardant une trace numérique si le bâtiment venait à être détruit ou pour suivre sa dégradation au fil du temps. Pourtant, au vu des coûts et des délais liés à un relevé complet d'un bâtiment, ce dernier indique souvent une intervention future sur l'édifice et sert entre autres à assister les experts lors de l'inspection et la détection des pathologies.

Mais les modèles 3D peuvent également servir pour d'autres fonctions comme :

- Études à distance et comparaisons avec d'autres structures.
- Reconstructions et restaurations virtuelles.
- Partage et accès à des bâtiments non visitables en raison de leur fragilité ou de leur statut privé.
- Accès à distance via des musées virtuels ou des visites virtuelles.
- Usage éducatif et pédagogique.
- Impression 3D pour répliques ou maquettes.
- Applications en réalité virtuelle et dans les jeux vidéo.

La lasergrammétrie et la photogrammétrie sont donc des techniques complémentaires de relevé qui permettent de réaliser des modèles 3D et des documents techniques du patrimoine bâti très précis. Néanmoins, elles demandent une formation ciblée pour l'acquisition et surtout le traitement des données. Ce dernier peut générer des fichiers très lourds selon la densité des nuages de points ou si le nombre de photos prises est trop élevé dans le cas d'une photogrammétrie. Il est donc important lors du relevé de prendre en compte la précision nécessaire afin d'ajuster l'acquisition des données à la finalité du 3D. Cependant, il est toujours important qu'un opérateur humain soit présent pour obtenir ces résultats (Wu et al., 2023).

Dans le cas de la détection de pathologies, celle-ci reste entièrement manuelle et les outils de numérisation cités précédemment assistent uniquement lors de la collecte d'informations afin d'éviter à l'expert d'être sur site tout le long de son inspection. Pourtant, le développement ces dernières années de l'intelligence artificielle et plus particulièrement du Deep Learning ouvre la porte à une possible automatisation de la détection de pathologies (Wu et al., 2023).

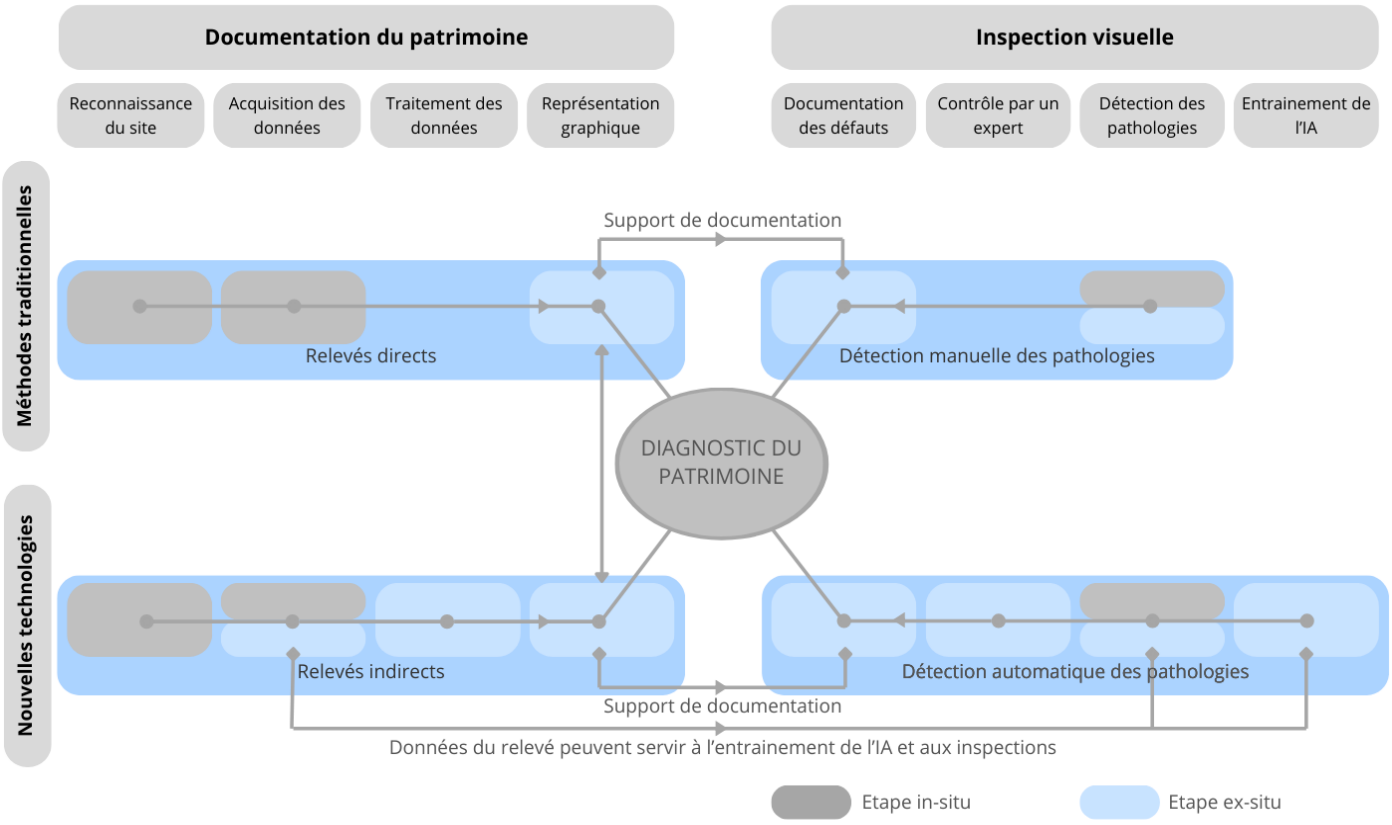


Fig. 16 – Diagramme des étapes de relevé et d'inspection visuelle du patrimoine

2.2. Intégration de l'intelligence artificielle

Malgré les nombreuses avancées en termes de relevé, l'inspection visuelle manuelle reste donc le moyen le plus répandu pour examiner le patrimoine bâti et détecter les pathologies de surface. L'utilisation des modèles 3D issus de la photogrammétrie ou la lasergrammétrie facilitent cependant le processus, en permettant une analyse ex-situ par l'expert. Si l'apparition de dégâts en surface comme de l'efflorescence ou de l'écaillage ne sont souvent en premier lieu qu'esthétiques, certaines pathologies peuvent progresser en profondeur et affecter la stabilité structurelle du bâtiment, ce qui engendre des travaux et coûts considérables. Les experts responsables de ces études doivent démontrer une expertise importante. De plus, l'identification des dégâts est une méthode longue et coûteuse. Les études sont donc souvent réalisées sur un seul type de défaut ou lorsque les dégradations sont importantes, en amont d'une intervention de grande échelle (Watt & Swallow, 1995). L'intelligence artificielle offre une opportunité d'assister les experts de différentes manières afin d'accélérer le processus.

Il existe de nombreux modèles de Deep Learning capables de détecter automatiquement les pathologies d'un bâtiment et malgré leurs différentes architectures, leur fonctionnement respecte toujours les étapes suivantes (fig. 17) :

- **Acquisition des images/données** *(data collection)*
- **Nettoyage et augmentation des datasets** *(data cleaning)*
- **Labellisation des datasets** *(data labelling)*
- **Entrainement du modèle choisi** *(DL tool)*
- **Évaluation des performances** *(testing dataset)*
- **Détection des pathologies** *(damage detection)*
- **Classification des pathologies détectées** *(damage classification)*

Ici, un dataset est un ensemble de données contenant les images d'entrainement et les labels de ces mêmes images. Bien que la plupart des étapes s'expliquent d'elles-mêmes, il est nécessaire d'aborder les différentes approches réalisées afin de comprendre les limites et les applications possibles du DL. Sur la fig. 17a, on peut observer que les premières étapes de l'utilisation d'un modèle de Deep Learning sont pratiquement identiques à une inspection réalisée aujourd'hui par un expert. Les données sont collectées et nettoyées, puis servent de support à l'identification des pathologies. Avec l'intelligence artificielle, ce processus sert de préparation à l'entrainement du modèle et à la détection d'éventuelles autres pathologies (fig. 17b).

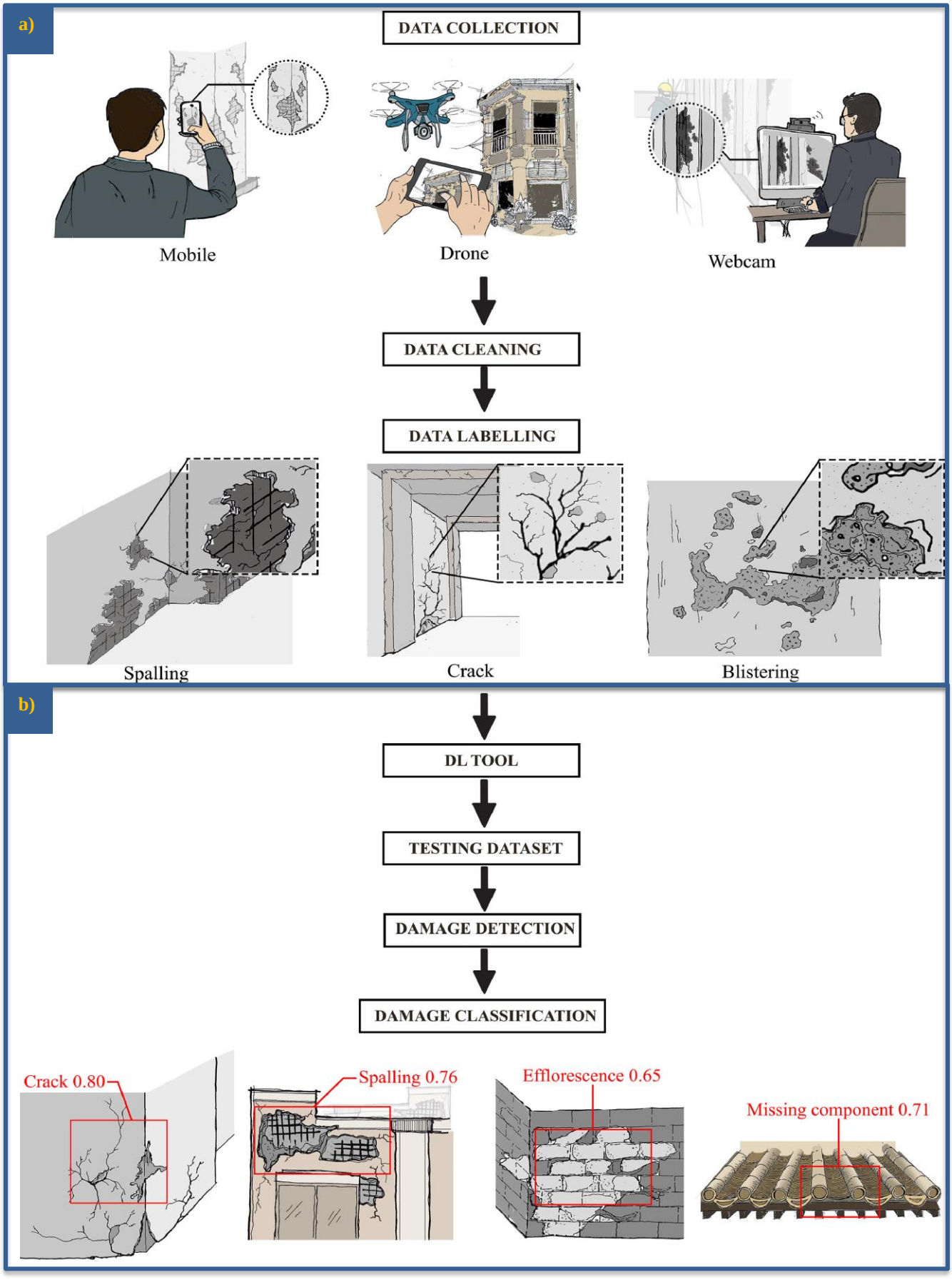


Fig. 17 – Organigramme du processus d'inspection du patrimoine assisté par IA (Mishra & Lourenço, 2024)

2.2.1. Acquisition des données

2.2.1.1. In situ

Dans la majorité des cas, la prise de photographies pour l'utilisation d'un modèle de Deep Learning est similaire à celles réalisées pour une photogrammétrie. Le matériel utilisé peut être une caméra ou un drone, s'il est disponible et que des endroits du bâtiment sont peu accessibles autrement (fig. 18). Certains modèles de DL sont même capables de réaliser la détection de pathologies en temps réel, permettant à l'opérateur d'avoir des résultats sur site et d'être assisté en temps réel. A la différence d'une photogrammétrie, les images capturées servent d'abord à l'entrainement du modèle. Elles ont pour but de représenter un maximum de situations observables. Pour optimiser cet entrainement il est recommandé de varier autant que possible les conditions météorologiques et de luminosité. Cela permet au modèle de s'adapter à différentes situations et d'améliorer sa robustesse. De même, l'angle de prise de vue et la distance par rapport aux défauts doivent être diversifiés, pour capturer un large éventail de perspectives et de configurations. Cette approche garantit une meilleure généralisation du modèle et une détection plus fiable des défauts dans des contextes variés.

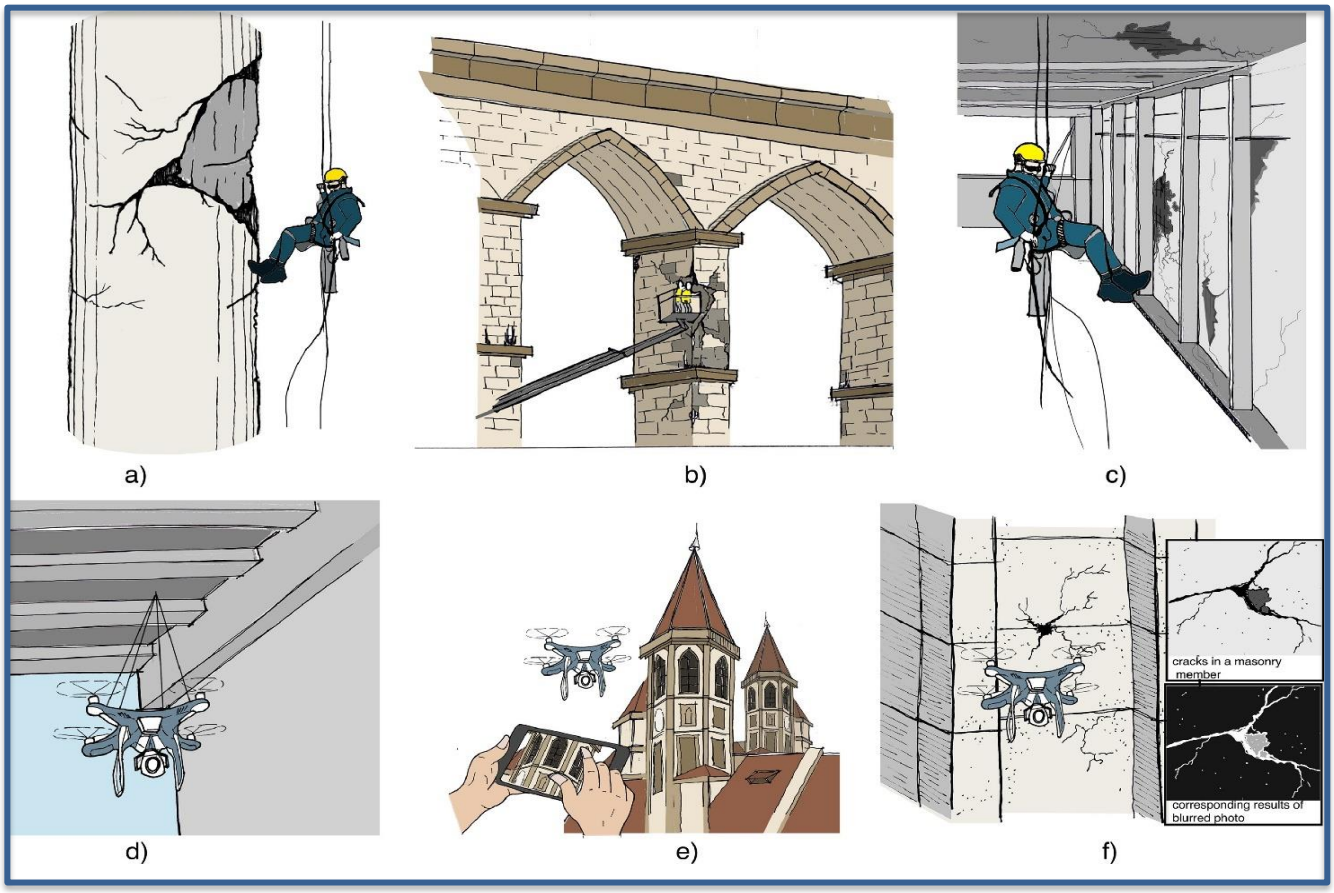


Fig. 18 – Schéma des difficultés liées aux inspections visuelles traditionnelles :
a-c) endroits peu accessibles et risqués, d-f) inspection par drone en temps réel (Mishra & Lourenço, 2024)

2.2.1.2. Ex situ

L'intérêt principal de l'automatisation de la détection des pathologies consiste à limiter au maximum l'implication humaine dans le processus. Par exemple, permettre l'acquisition des images nécessaires à la détection de ces pathologies sans un opérateur présent sur terrain, ou utiliser des algorithmes capables d'alerter l'expert en cas de nécessité, permettent ainsi d'éviter un contrôle humain régulier. Des caméras ou capteurs connectés présents sur site pourraient récolter et envoyer les données, soit sous forme d'images, soit sous forme d'alerte après analyse.

L'Internet of Things (IoT) désigne un réseau de dispositifs physiques, véhicules, appareils électroménagers et autres objets physiques qui sont dotés de capteurs, de logiciels et de connectivité réseau leur permettant de collecter et de partager des données. L'IoT permet ainsi de créer des systèmes intelligents qui peuvent surveiller et contrôler divers aspects de l'environnement et du patrimoine en temps réel. Des caméras connectées peuvent collecter des données en continu, permettant de suivre les changements dans les conditions structurelles ou de détecter les détériorations potentielles des bâtiments (Mishra et al, 2022b).

Le Mobile Crowd-sensing (MCS) exploite la disponibilité généralisée des appareils mobiles, tels que les smartphones et les objets connectés, pour collecter et analyser des données provenant d'un grand groupe d'individus. Il combine les concepts de **crowdsourcing** et de détection mobile, permettant de recueillir des ensembles étendus de données grâce à la participation de nombreux utilisateurs.

Des études réalisées par Meklati et al. (2023) ainsi que Wang et al. (2019) ont abordé cette approche en créant une application mobile capable de détecter les pathologies visuelles, ou en analysant automatiquement les dégâts de la Grande Muraille de Chine. Cette méthode n'est pas pertinente pour tout type de patrimoine car la détection des pathologies dépend directement de la fréquentation du site et du nombre d'appareils connectés. La qualité des résultats reste également limitée par les capacités des appareils mobiles.

2.2.2. Traitement des données

2.2.2.1. Nettoyage et augmentation des datasets

Le nettoyage consiste à retirer les images de mauvaise qualité tandis que l'augmentation du dataset a pour but d'élargir le nombre d'images servant à entrainer le modèle en modifiant et retouchant les images déjà récoltées. La quantité d'images utilisables est souvent trop faible pour prendre en compte toutes les différences d'environnement, orientation, taille, luminosité, texture ou apparence des pathologies. Parce que l'acquisition de données supplémentaires est assez couteuse (accès au matériel, aux bâtiments patrimoniaux, à la main d'œuvre), l'augmentation du dataset est la méthode la plus utilisée dans les études mentionnées dans ce travail. La retouche des images se fait avec des rotations, retournements haut-bas, différences de contraste, saturation et luminosité, floutage, conversions de couleur, etc. Si tous les types de pathologies ne sont pas propices à cette méthode, par exemple ceux ayant des directions très spécifiques, cette technique améliore globalement la fiabilité des modèles.

2.2.2.2. Labellisation des datasets

Une fois les données nettoyées et augmentées, il est nécessaire de les labelliser. Il faut donc indiquer sur chaque image où sont les pathologies et les nommer. Cette opération doit nécessairement se faire par un expert afin d'être précise et éviter les erreurs. Il est préférable de faire appel à différents experts ayant des formations différentes afin d'avoir une base de données correcte et objective. Cette opération est par ailleurs longue et minutieuse, car la labellisation se fait image par image et les opérateurs ayant les compétences requises ne sont pas toujours assez nombreux pour la quantité de travail (Guo et al., 2024). Kamel et al. (2023) ont étudié la possibilité d'utiliser le crowdsourcing, ou science participative, afin d'assister dans la labellisation d'images.

2.2.2.3. Entrainement du modèle

L'entrainement d'un modèle se compose habituellement de trois étapes : (1) *Forward pass*, (2) *Calculating loss* et (3) *Weight updating of network*. Elles ont respectivement pour fonction de faire passer les images dans le modèle, calculer la différence entre les prédictions réalisées et les valeurs réelles et équilibrer les différentes classes de pathologies. La perte, ou *Loss*, est la pénalité indiquant à quel point la prédiction est mauvaise pour un seul passage du modèle.

Lors du *forward pass*, une image labellisée est passée à travers le modèle et les résultats sont calculés. Ces derniers sont comparés avec les valeurs correctes pour calculer la perte. L'objectif principal de l'entrainement est de minimiser la perte. Ces deux étapes d'entraînement doivent être répétées jusqu'à ce que la perte soit minimale et presque constante. On appelle *epoch* (époque) le passage complet du dataset dans le modèle pour son entrainement. La quantité d'époques affecte directement le temps d'entrainement du modèle et la précision de la détection. En effet, avec un nombre réduit d'époques, le temps d'entrainement est court mais la précision est très faible et la perte ne peut pas être minimisée.

Une fois l'entrainement terminé, certains opérateurs appliquent le *weight updating of network*, il n'est pas toujours effectué car certains modèles n'en sont pas capables. Il a pour but de traiter le problème de déséquilibre des classes, qui se produit lorsque certaines classes d'objets (comme les fissures) sont beaucoup moins représentées que d'autres (comme l'arrière-plan) au niveau des pixels. Afin de régler ce problème, on peut rééchantillonner le dataset en équilibrant la quantité de pathologies représentées ou modifier la fonction de perte sur les classes minoritaires afin que le modèle se concentre davantage sur l'apprentissage de celles-ci.

Pour illustrer ces étapes, Mansuri et Patel (2022b) ont utilisé un dataset de 400 images labellisées pour entrainer un modèle Faster R-CNN à détecter l'écaillage et les fissures dans un cimetière en Inde. Afin d'accélérer le processus d'entrainement, les images furent réduites de 3648 × 2736 pixels à 800 × 600 pixels. Le nombre optimal d'étapes d'entrainement pour obtenir une perte constante a été trouvé par essais et erreurs avec 40 000 époques. La perte minimale a été enregistrée à 36 000 époques, ce qui a pris 5 heures de temps d'entrainement. Le faible nombre de pathologies et indirectement d'images, ainsi que la réduction de la taille de ces images ont fortement accéléré le processus d'entrainement.

2.2.2.4. *Evaluation des performances du modèle*

Mean Average Precision

Le mAP (*mean Average Precision*) est l'un des systèmes métriques les plus communément utilisés pour donner un score de performance aux modèles de détection d'objets et de reconnaissance d'images. Ce score correspond à la moyenne du score d'*Average Precision* pour chaque classe de pathologies détectée par le modèle. Le mAP donne donc un seul score permettant d'établir rapidement la fiabilité d'un modèle dans son ensemble.

L'AP se calcule quant à lui sur base de 4 autres métriques :

L'Intersection over Union (IoU) est une métrique, ou type de mesure, utilisée pour évaluer la précision des modèles de détection d'objets. Elle compare l'aire d'intersection entre le cadrage de détection prédit par le modèle et le cadrage réel avec l'aire de leur union. Une valeur proche de 1 indique une bonne correspondance entre la prédiction et la réalité (Rosebrock, 2016).

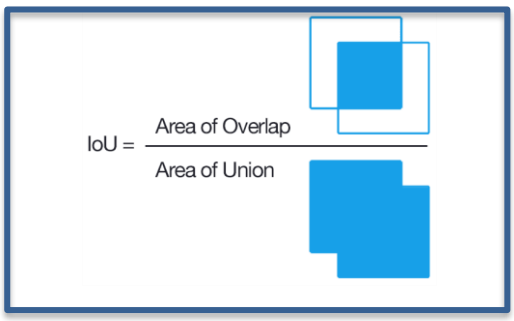


Fig. 19 – Formule de calcul de l'IoU (Rosebrock, 2016)

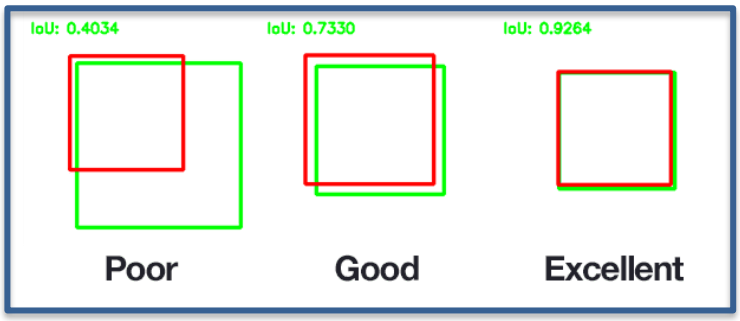


Fig. 20 – Exemple de calcul de l'IoU pour différentes prédictions (Rosebrock, 2016)

La matrice de confusion est un tableau qui permet de visualiser la performance d'un algorithme de classification. Elle montre les proportions des vrais positifs, faux positifs, vrais négatifs et faux négatifs entre les prédictions du modèle et les données réelles. On utilise la valeur du IoU afin de différencier le positif du négatif.

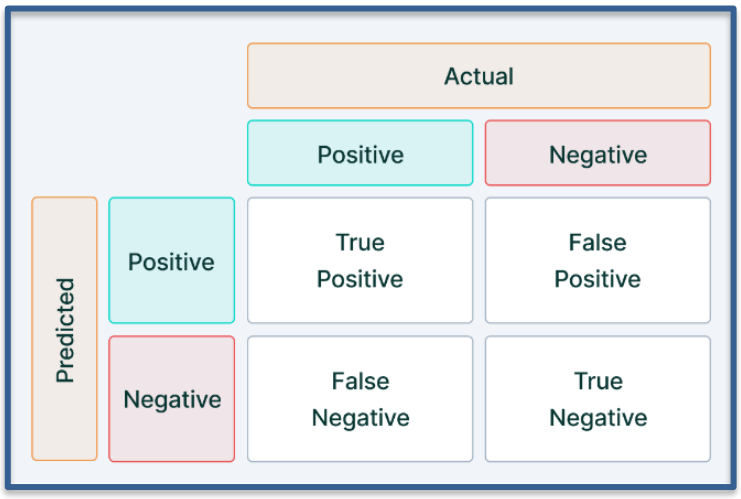


Fig. 21 – Matrice de confusion (Shah, 2022)

La précision mesure la proportion de vraies prédictions positives parmi toutes les prédictions positives faites par le modèle. Elle est calculée comme le ratio des vrais positifs sur le total des vrais positifs et des faux positifs.

Le **recall,** ou rappel, mesure la capacité d'un modèle à identifier tous les échantillons pertinents (vrais positifs) parmi toutes les instances positives présentes (Shah, 2022) (Gad, 2020).

Le mAP peut également être plus précis, selon le niveau de précision qu'il demande du modèle. Un mAP50, souvent utilisé par défaut, signifie qu'il a été calculé à un seuil d'intersection sur union (IoU) de 0,50. Il s'agit d'une mesure de la précision du modèle qui ne prend en compte que les détections "faciles". Le mAP50-95 est donc une moyenne de la précision plus sévère, calculée pour différents seuils d'IoU, allant de 0,50 à 0,95. Il donne une vue d'ensemble des performances du modèle à différents niveaux de difficulté de détection et est généralement nettement plus faible que le mAP50.

Accuracy (Acc)

L'exactitude est une autre manière de mesurer la précision d'un modèle. Elle est plus simple à mettre en place que le mAP car elle n'a besoin que des données de la matrice de confusion. La valeur est obtenue grâce au ratio des prédictions correctes (vrai positif et vrai négatif) par rapport à toutes les prédictions du modèle (Shah, 2022).

Frames per Second (FPS)

Dans le cas de modèles de détection en temps réel, il est aussi important de connaître le **temps d'inférence**, ou la vitesse de traitement des résultats. Les FPS représentent le nombre d'images par seconde qui peuvent être traitées par l'algorithme (Gündüz et Işık, 2023). Si les prédictions ne sont pas suffisamment rapides, on considère alors le temps d'exécution pour une seule image, généralement en millisecondes (ms).

Score F1

La note F1 est la moyenne de la précision et du rappel. Elle fournit une évaluation équilibrée des performances d'un modèle en tenant compte à la fois des faux positifs et des faux négatifs.

3. Le Deep Learning pour la détection de pathologies

L'intelligence artificielle, qui fait partie de la science des données, englobe diverses disciplines visant à doter les machines de capacités similaires à l'intelligence humaine. Parmi ces disciplines, le Machine Learning développe des algorithmes permettant aux systèmes informatiques d'apprendre et de s'améliorer à partir de données, sans être explicitement programmés pour chaque tâche. On distingue plusieurs manières de s'améliorer pour ces algorithmes, notamment l'apprentissage supervisé, non supervisé ou par renforcement. Dans le cas de la détection de pathologies, cet apprentissage est supervisé car les opérateurs décident précisément des données utilisées pour l'entrainement (Janiesch et al., 2021).

Le **Deep Learning** est une sous-catégorie du Machine Learning qui utilise des réseaux de neurones artificiels profonds pour modéliser des structures complexes à partir de grandes quantités de données. C'est une approche inspirée du fonctionnement du cerveau humain qui est très efficace pour des tâches comme le traitement d'images, la reconnaissance vocale et la traduction (Janiesch et al., 2021).

Le traitement d'images en Deep Learning est ce qu'on appelle le **Computer Vision**, qui permet aux machines de comprendre et d'interpréter visuellement le monde à partir d'images ou de vidéos, comme le fait l'œil humain.

Son fonctionnement débute par l'acquisition d'images d'entrainement à l'aide de caméras ou d'appareils photos. Ensuite, des algorithmes analysent ces données en extrayant des caractéristiques clés, comme les contours ou les formes. Ces informations sont ensuite traitées par des modèles qui les interprètent pour générer des prédictions (Mansuri et Patel, 2022a) (Janiesch et al., 2021).

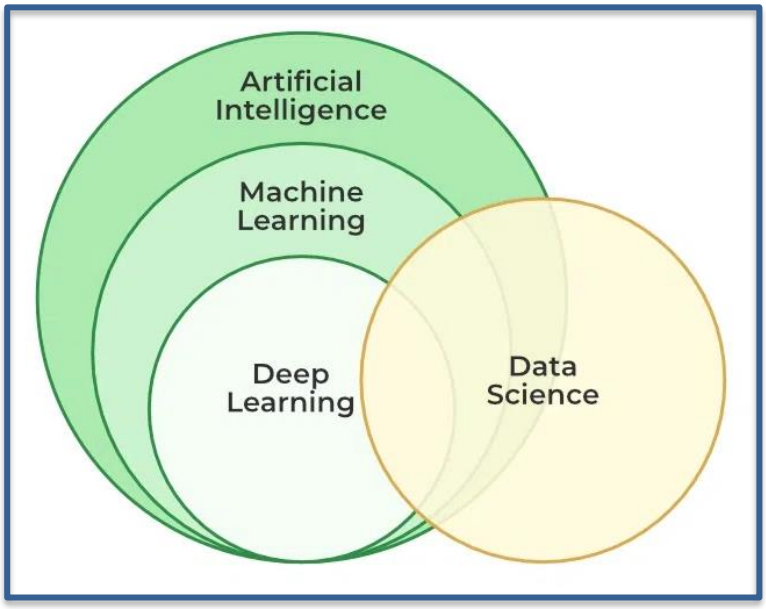


Fig. 22 – Hiérarchie des différents domaines de l'IA

3.1. Tâches de Deep Learning

Après avoir abordé les types d'acquisition et de traitement des images pour les modèles de Deep Learning, il est nécessaire d'expliquer les différentes tâches qui peuvent être demandées à ces modèles. Les tâches de base de Computer Vision sont la classification, la détection et la segmentation, qui fournissent respectivement des informations visuelles au niveau de l'image même, de l'objet et du pixel. La tâche de segmentation peut être subdivisée en deux sous-tâches : la segmentation sémantique et la segmentation d'instances (Guo et al., 2024). La plupart des modèles sont spécialisés dans une ou deux de ces catégories à cause de leur architecture, parfois trop simple pour la segmentation par exemple. Le fonctionnement des modèles les plus fréquemment utilisés sera abordé ultérieurement.

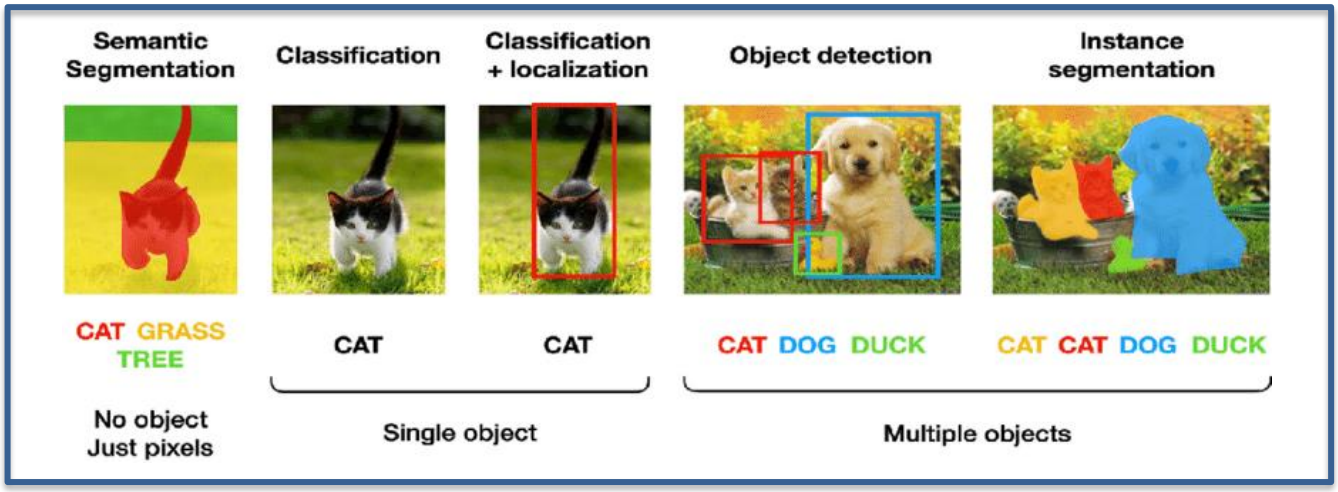


Fig. 23 – Comparaison des tâches de DL (Jaikumar et al., 2022)

3.1.1. Classification d'images

C'est une tâche fondamentale du Computer Vision qui consiste à classer des images en se basant sur les données d'entrainement. Elle a été la première application dans la détection de pathologies grâce à sa simplicité. Les modèles les plus couramment utilisés pour la classification des défauts de surface incluent AlexNet, Inception Networks, VGG, et ResNet. Ils sont tous considérés comme des réseaux de neurones convolutifs (CNN), dont le fonctionnement sera présenté au point 3.2.1. La complexité des défauts peut diminuer les performances des modèles CNN plus basiques (Guo et al., 2024).

Récemment, de nouvelles études ont fourni des avancées notables en classant différentes pathologies mais aussi leur degré de sévérité. Des modèles CNN plus complexes sont également capables de localiser l'objet sur l'image à l'aide de l'algorithme CAM (Class activation mapping). Cependant, cette méthode traite toujours l'image comme une entité unique et n'est donc pas capable d'y détecter plusieurs objets. La séparation de chaque pathologie en une image rend difficile leur localisation par rapport à l'ensemble du bâtiment et n'automatise pas réellement le processus. Ces études seront abordées plus en profondeur au chapitre suivant.

3.1.2. Détection d'objets

La détection d'objets permet de localiser chaque défaut dans une image en identifiant un cadrage autour de celui-ci, ce qui permet de cartographier précisément la localisation et l'étendue de la pathologie. Contrairement à la classification, plusieurs objets peuvent être détectés sur une même image. Cette méthode permet d'étudier la fréquence d'apparition et les caractéristiques de certains défauts mais n'est pas capable de connaitre précisément leur étendue par elle-même.

Les principaux modèles de détection d'objets incluent les réseaux neuronaux convolutifs basés sur les régions comme Faster R-CNN et les réseaux de détection en un temps, comme SSD (Single Shot MultiBox Detector) ou YOLO (You Only Look Once) (Guo et al., 2024).

3.1.3. Segmentation sémantique

Cette tâche divise une image en segments significatifs au niveau des pixels, où chaque pixel est classé en fonction de la classe de pathologie à laquelle il appartient. La segmentation sémantique permet d'obtenir des informations détaillées sur la forme et la taille des défauts, en classant séparément chaque pixel de l'image. Les modèles les plus utilisés sont des réseaux complètement convolutifs (FCN) comme DeepLab, U-Net DenseNet and SegNet.

La technique est prometteuse car très précise et peut facilement être combinée avec des nuages de points pour cartographier les pathologies sur l'ensemble d'un bâtiment. On connaitrait donc la taille et la forme exacte des défauts mais aussi leur localisation précise sur un modèle 3D numérique de l'édifice.

3.1.4. Segmentation d'instances

La segmentation d'instance est une méthode plus récente en détection de pathologies, et est une extension de la segmentation sémantique qui classe non seulement chaque pixel mais qui distingue aussi chaque instance de pathologie au sein d'une même classe. Chaque défaut est non seulement identifié et catégorisé, mais également quantifié par rapport aux autres défauts de la même catégorie, permettant une analyse plus fine et détaillée des défauts. Mask R-CNN et SOLO sont des modèles communs pour cette tâche (Guo et al., 2024). Certaines études ont obtenu des résultats similaires en combinant des algorithmes de détection d'objets et de segmentation sémantique.

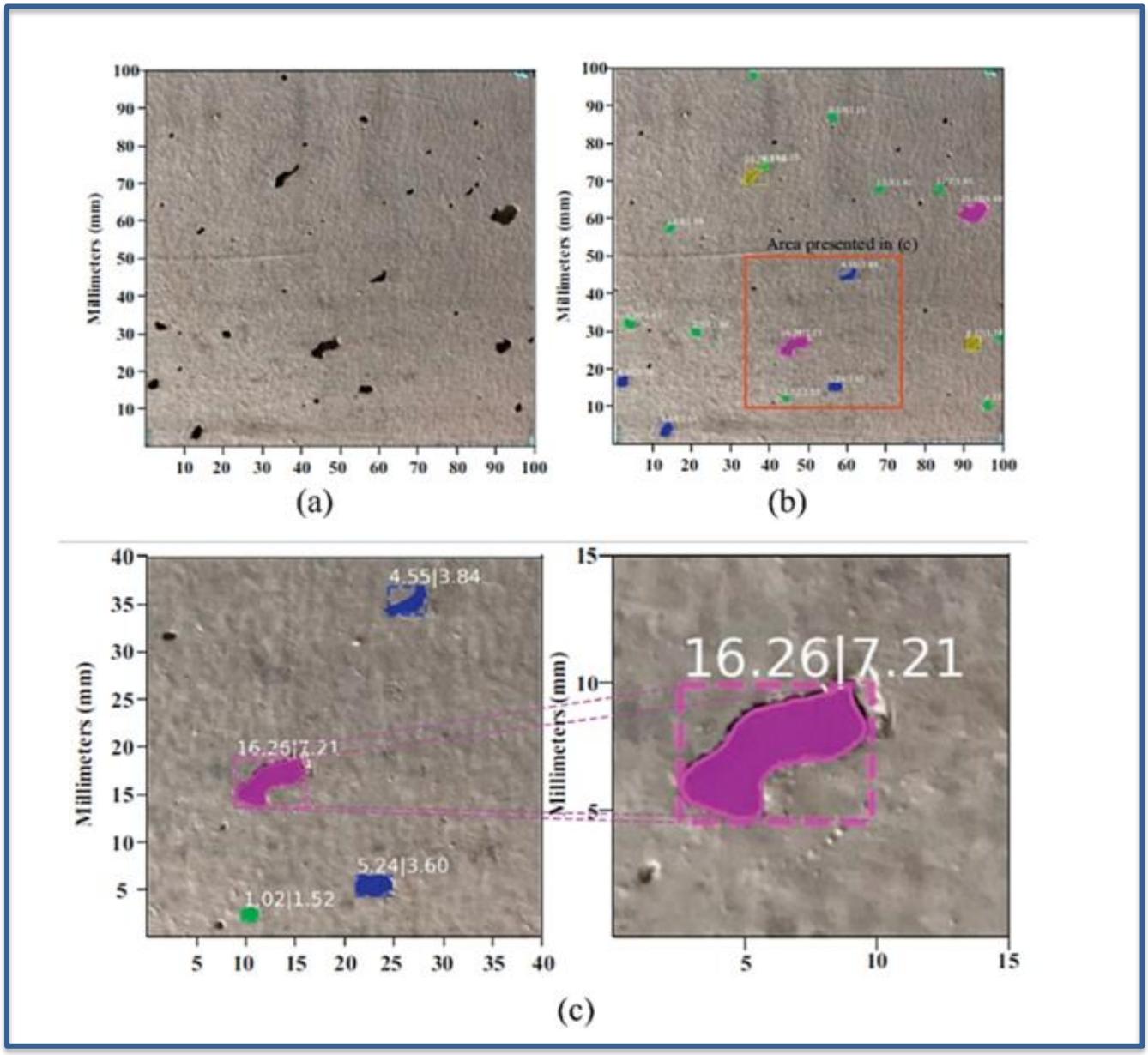


Fig. 24 – Exemple de segmentation d'instances sur du béton :
a) image d'entrée, b) image de sortie, c) résultats de la quantification des défauts (Guo et al., 2024)

3.2. Familles et architectures de modèles

Guo et al. (2024) ont analysé près de 237 études publiées entre 2017 et 2023 afin de recenser tous les modèles de Deep Learning réalisant l'une des 4 catégories de tâches citées précédemment. Leur recherche a permis de déduire les principaux modèles utilisés dans la détection de pathologies pour chaque tâche demandée. L'étude de Mishra et Lourenço (2024) a été utilisée pour compléter les données jusque mars 2024. Cela a ensuite permis de classer de manière non exhaustive les techniques les plus utilisées selon les tâches réalisées. Ce chapitre a pour but de comparer le fonctionnement de toutes ces techniques afin de comprendre leurs forces et limitations. Le tableau ci-dessous représente les noms des modèles mentionnés dans les deux travaux étudiés, ainsi que leur architecture.

Tâche réalisée	*Nom du modèle*	*Architecture du modèle*
Classification d'images	Alexnet	CNN
	Inception network	CNN
	VGG network	CNN
	Resnet	CNN
Détection d'objets	R-CNN	R-CNN
	Faster R-CNN	Faster R-CNN
	SSD	SSD
	YOLO	YOLO
	Vision transformer	ViT
Segmentation sémantique	FCN	FCN
	DeepLab	CNN
	U-Net	FCN
	DenseNet	CNN
	SegNet	CNN
	LinkNet	CNN
	MobileNet	CNN
Segmentation d'instances	Mask R-CNN	Mask R-CNN
	SOLO	SOLO
	YOLO & U-Net	YOLO & FCN
	SSD & SegNet	SSD & CNN

Fig. 25 – Tableau des tâches de DL et des architectures associées [adapté de Guo et al. (2024) et Mishra et Lourenço (2024)]

Un CNN est l'architecture la plus basique dans le traitement d'images par DL. Cette architecture sert de fondation à la plupart des modèles utilisés aujourd'hui. Le Vision transformer étant très peu représenté dans les nombreuses études réalisées, il ne sera pas couvert en détail dans la suite de ce chapitre.

3.2.1. Convolutional Neural Network (CNN)

Un CNN, ou réseau neuronal convolutif, est une catégorie de réseau neuronal profond construit spécifiquement pour traiter et reconnaitre des images. Son architecture s'apparente à l'organisation du cortex visuel chez les êtres vivants et sert de base à tous les modèles de détection automatique par image (Voulodimos et al., 2018). Un CNN contient en amont une partie convolutive, ou feature extractor, responsable d'extraire les caractéristiques de l'image et de la compresser tout en gardant les informations importantes et en aval une partie de classification chargée de traiter le code issu de la première partie pour classer l'image. Les images RVB fournies en entrée sont traitées comme une matrice de pixels en 3 dimensions représentant les 3 couleurs fondamentales (rouge, vert et bleu). Le modèle VGG-16, de la famille des réseaux VGG, est utilisé ci-dessous pour illustrer le fonctionnement d'un CNN. Il contient les mêmes couches et opérations que les autres modèles actuellement sur le marché mais à une échelle relativement simple. Ce type de modèle réalise principalement de la classification d'images.

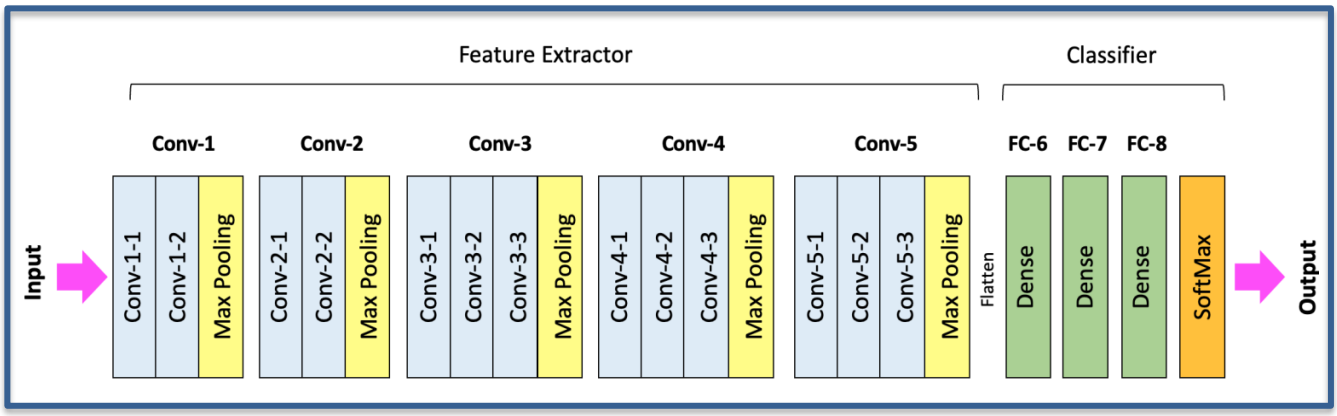


Fig. 26 – Structure d'un réseau de neurones convolutifs (VGG-16) (Voulodimos et al., 2018)

La partie convolutive est donc un extracteur de caractéristiques composé de blocs convolutifs. Ces blocs peuvent être considérés comme les « yeux » du CNN car ils sont capables de détecter les caractéristiques de l'image. La partie de classification traite quant à elle les données pour ensuite les envoyer dans la fonction d'activation SoftMax qui leur donne une valeur entre 0 et 1, permettant de classer l'image selon l'entrainement reçu par le modèle.

3.2.1.1. Blocs convolutifs

Couche convolutive

Cette couche consiste à appliquer des filtres sur l'image d'entrée pour en extraire les caractéristiques. Cette image peut être l'originale ou celle extraite d'un bloc convolutif d'une couche précédente du modèle. Elle se compose de trois canaux, un pour chaque couleur fondamentale. Par exemple, un input original a une forme 224x224x3, où 224x224 est sa taille en pixels et le nombre 3 représente les 3 canaux. Le filtre balaie l'image et réalise une opération convolutive à chaque endroit où il s'arrête, ce qui produit un seul nombre qui est ensuite passé à travers une fonction d'activation avant d'être inscrit sur une carte d'activation, ou *activation map* (224x224x1). Cette dernière contient un résumé des informations extraites lors de l'opération (*Convolutional Neural Network: A Complete Guide*, 2023).

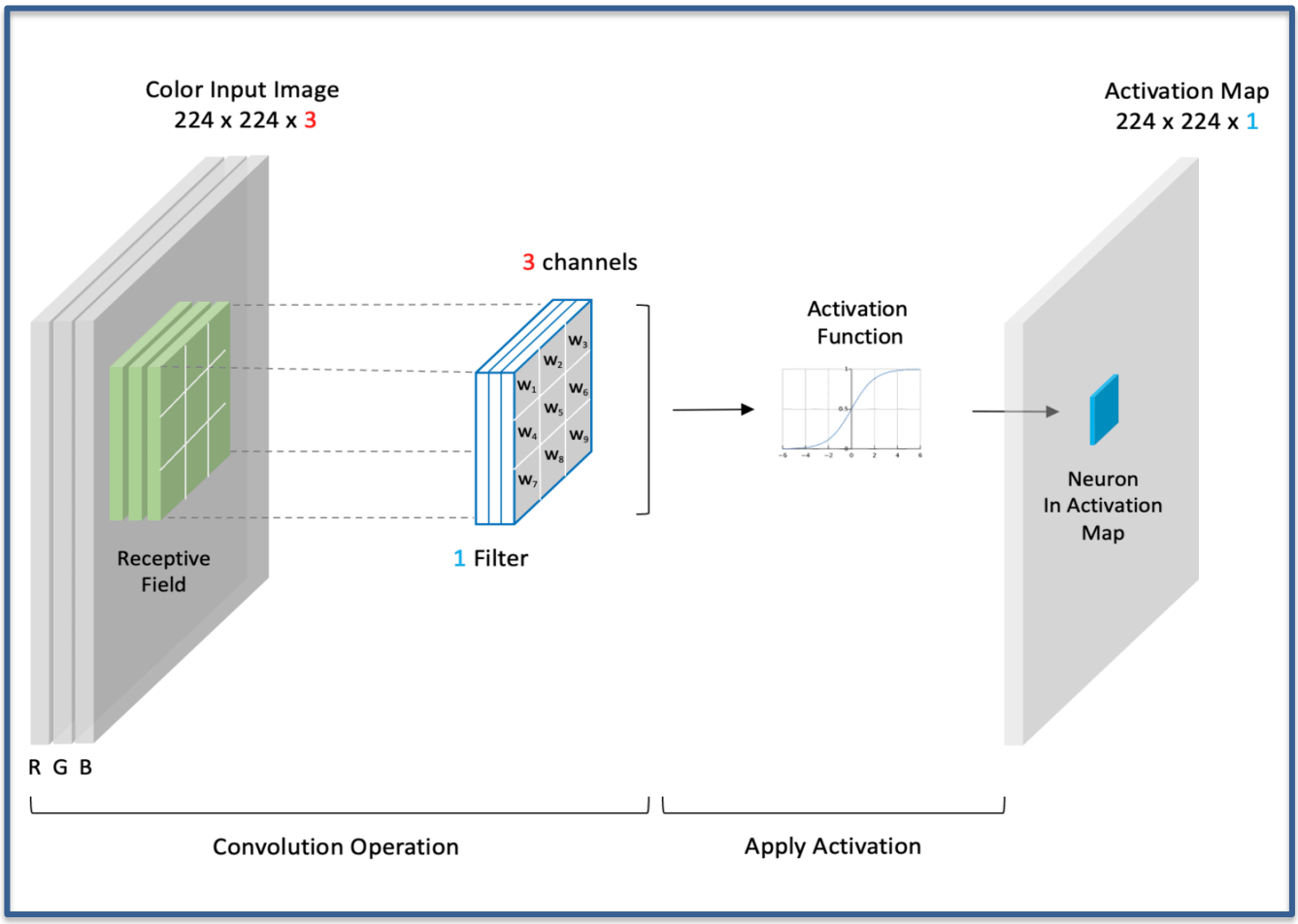


Fig. 27 – Structure d'une couche convolutive (VGG-16)
(Convolutional Neural Network: A Complete Guide, 2023)

Avant de décrire les couches convolutives plus en détail, il est d'abord nécessaire d'expliquer comment l'opération de convolution est réalisée. Le filtre est une matrice dont les valeurs représentent des éléments détectables dans l'image comme des bords verticaux ou horizontaux, des textures ou couleurs spécifiques, etc. Dans le cas d'un CNN, ces valeurs du filtre sont d'abord aléatoires puis sont apprises lors de l'entrainement du modèle. Dans la fig. 28, on peut observer l'exemple des premières itérations de CNNs capables d'identifier les chiffres de 0 à 9, où les valeurs du filtre identifient les bords de l'objet et leur orientation. Les caractéristiques se complexifient au fur et à mesure que les couches se suivent, ce qui permet de discerner les bords d'un objet, puis une partie de sa forme ou d'une texture, et finalement l'objet entier.

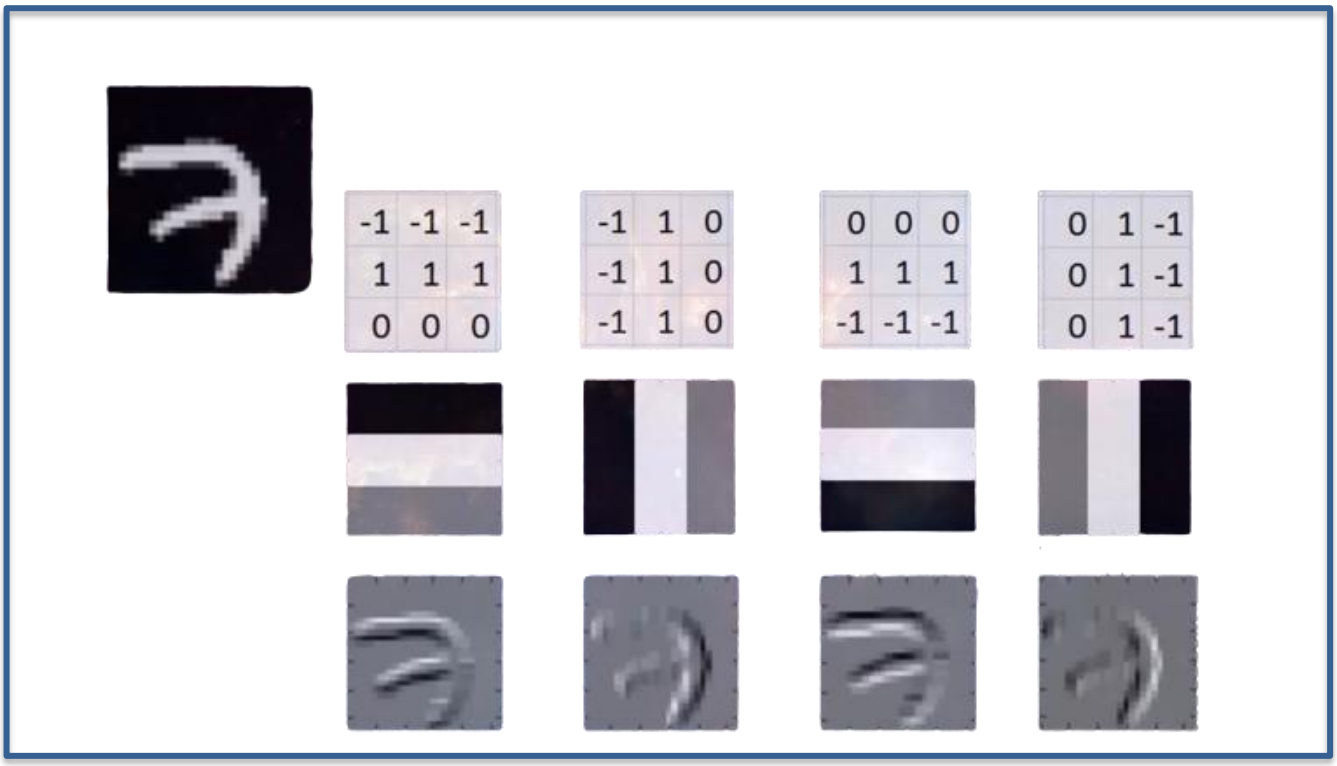


Fig. 28 – Extraction des caractéristiques d'une image par un filtre 3x3 (Deeplizard, 2017)

Dans l'exemple présenté dans la fig. 29, un input de 6x6 est traité par un filtre de 3x3. La valeur qui en résulte est un nombre unique représentant l'output de l'opération de convolution pour une position donnée du filtre. Le processus est répété à chaque position du filtre jusqu'à ce que l'image soit entièrement balayée. La position du filtre est ici montrée en bleu foncé. Cette région est également appelée le champ récepteur. Remarquez que le centre du filtre est indiqué en vert. L'opération de convolution effectuée à chaque position du filtre est simplement le produit de la matrice d'input avec celle du filtre.

Le *stride*, ou foulée, est la valeur qui détermine le nombre de pixels que le filtre franchit après une opération. Dans cet exemple, le *stride* est égal à 1. Augmenter le *stride* revient à accélérer le processus au détriment de la collecte d'informations tandis que le réduire permet d'extraire plus d'informations et d'avoir des cartes d'activation plus larges. Cependant, malgré le stride à son minimum, les coins de l'image sont altérés car chaque opération diminue sa taille selon la taille du filtre.

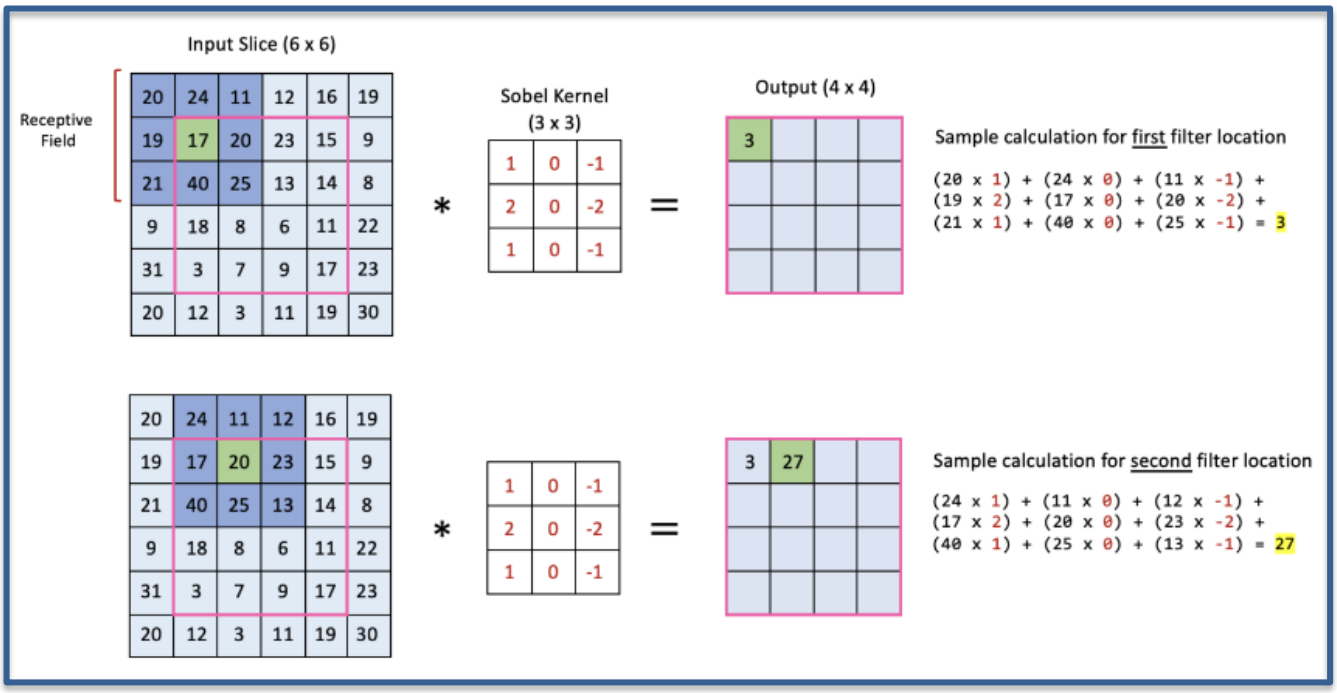


Fig. 29 – Opération convolutive de matrice 6x6 par un filtre 3x3

(*Convolutional Neural Network: A Complete Guide*, 2023)

Ici, l'input mesure 6x6 mais l'output ne fait plus que 4x4 pixels. Le processus de *padding*, ou remplissage, permet cependant de conserver la taille d'origine de l'image d'entrée en rajoutant des pixels aux extrémités de cette dernière. De nombreuses techniques de *padding* existent mais la plus fréquente est le *zero padding* car elle est simple et demande peu de calculs à l'ordinateur. Elle consiste à ajouter des zéros autour des extrémités de la matrice d'entrée.

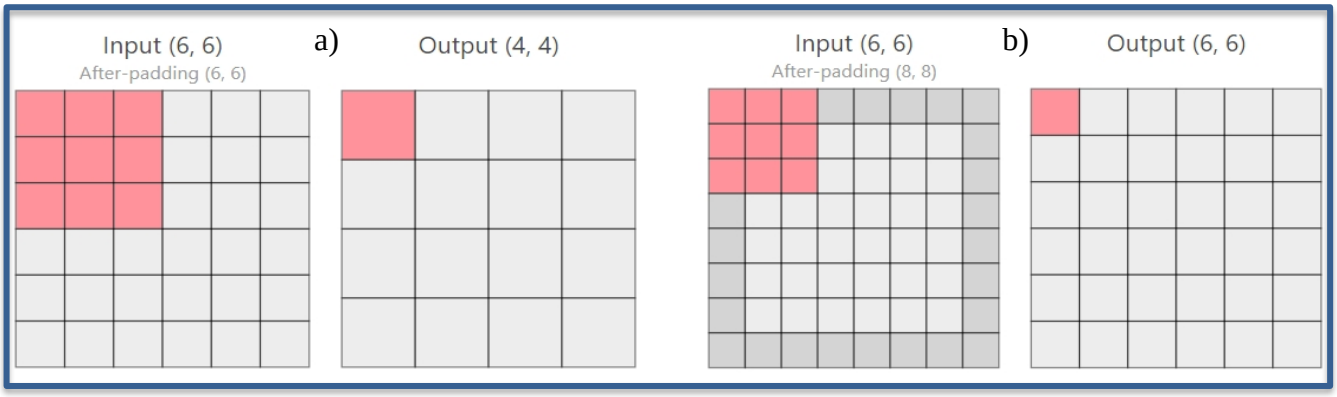


Fig. 30 – Opération convolutive de matrice 6x6 : a) sans *padding*, b) avec *padding* (Wang et al., 2020a)

Fonction d'activation (ReLU)

La fonction *Rectified Linear Unit* est ce qui permet aux modèles d'atteindre leur précision en les rendant non-linéaires. Sans fonction d'activation non-linéaire, les architectures CNN se réduiraient à une seule couche convolutive équivalente, ce qui serait beaucoup moins performant. Cette fonction est appliquée à chaque valeur des cartes d'activation et élimine toute donnée négative.

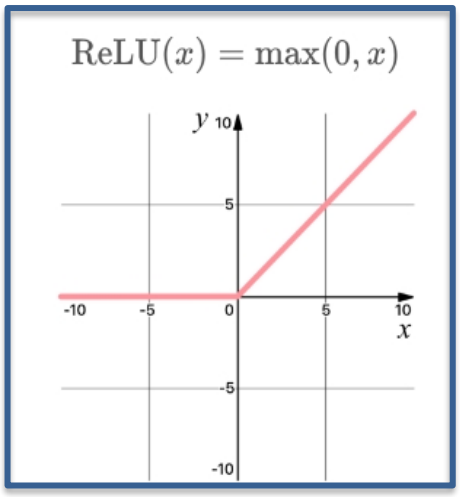


Fig. 31 – Représentation graphique de la fonction d'activation ReLU (Wang et al., 2020a)

Couche de Pooling

Il existe plusieurs techniques de *pooling* mais elles ont toutes pour objectif de réduire graduellement la taille du réseau afin de diminuer la quantité de paramètres et la puissance de calcul requise pour son fonctionnement. L'opération de *Max Pooling*, la plus fréquente, commence par la sélection d'une taille de filtre et d'une valeur de stride. Elle ne s'effectue que sur la carte d'activation et le filtre récupère la plus grande valeur identifiée dans celui-ci. La fig. 32 illustre un filtre de 2x2 et un stride de 2, ce qui diminue de 75% la quantité de données à traiter pour les prochains blocs convolutifs. Cela est particulièrement important car la quantité de filtres dans les blocs suivant est parfois plus élevée. Sans cette opération, la quantité de données à traiter grandirait de manière exponentielle. Le filtre dans cette opération ne peut pas être entrainé, contrairement à ceux qui gèrent les opérations convolutives.

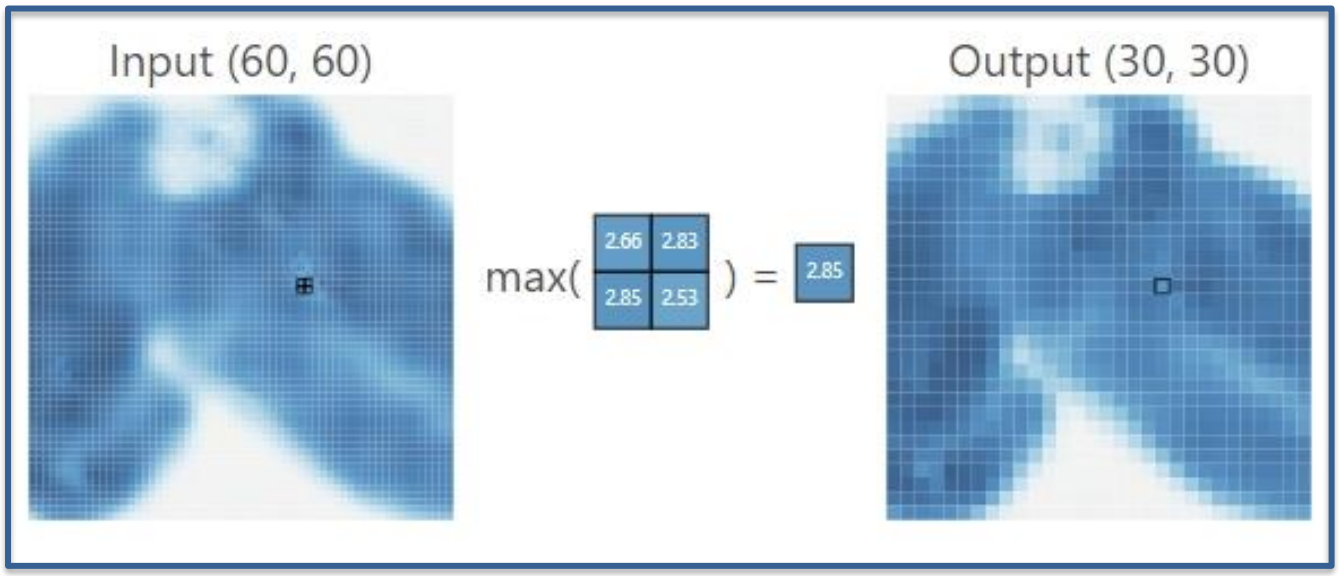


Fig. 32 – Opération de Max Pooling (Wang et al., 2020a)

3.2.1.2. Classification

Couche d'aplatissement

Les couches convolutives précédentes ont extrait les caractéristiques de l'image d'entrée, mais il est maintenant temps de classer ces caractéristiques. L'aplatissement sert à convertir les strates tridimensionnelles créées par les couches précédentes en un vecteur unidimensionnel. Ce vecteur correspond à l'input demandé par les couches suivantes et la fonction Softmax qui sont chargées de classifier l'image. Par exemple, une carte d'activation de 7x7x512 serait convertie en un vecteur de taille 25 088 (fig. 33a).

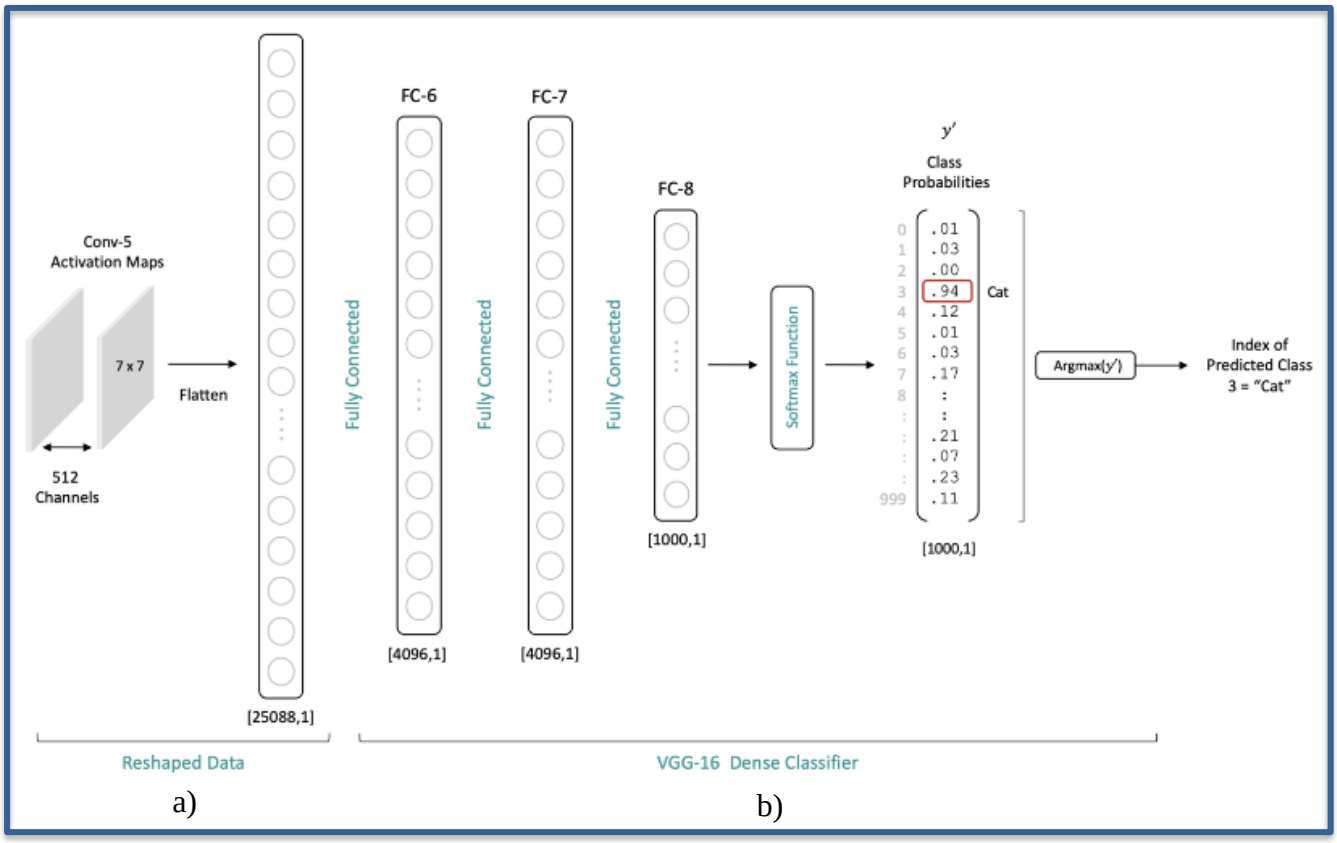


Fig. 33 – Structure de la partie de classification d'un CNN (*Convolutional Neural Network: A Complete Guide*, 2023)

Couches complétement connectées (Fully Connected)

Dans ces couches, chaque neurone est connecté à tous les neurones de la couche précédente. C'est pour cette raison que l'input doit être un vecteur, peu importe les dimensions originales de l'input, car toutes les données sont traitées en même temps et comparées entre elles. Ces comparaisons permettent de réaliser des prédictions d'appartenance de l'objet pour chaque classe (fig. 33b). Le nombre de neurones de la couche de sortie de ces prédictions est égal au nombre de classes détectables, avec un neurone associé à chaque classe. Cependant, les valeurs que ces neurones représentent ne sont pas encore utilisables.

Fonction d'activation Softmax

La fonction Softmax a un seul objectif, s'assurer que l'output du CNN soit compris entre 0 et 1 et puisse servir de score de probabilité. La valeur est calculée en prenant le rapport de la classe choisie par rapport à la somme de toutes les classes détectables par le modèle. Le résultat est la probabilité que l'image insérée dans le modèle appartienne à cette classe (Wang et al., 2020) (*Convolutional Neural Network : A Complete Guide*, 2023).

3.2.1.3. En résumé

Reprenons le diagramme de la fig. 26 présentant VGG-16 au point 3.2.1. Il comprend donc cinq blocs convolutifs contenant chacun deux ou trois couches convolutives et une couche de *Max Pooling*.

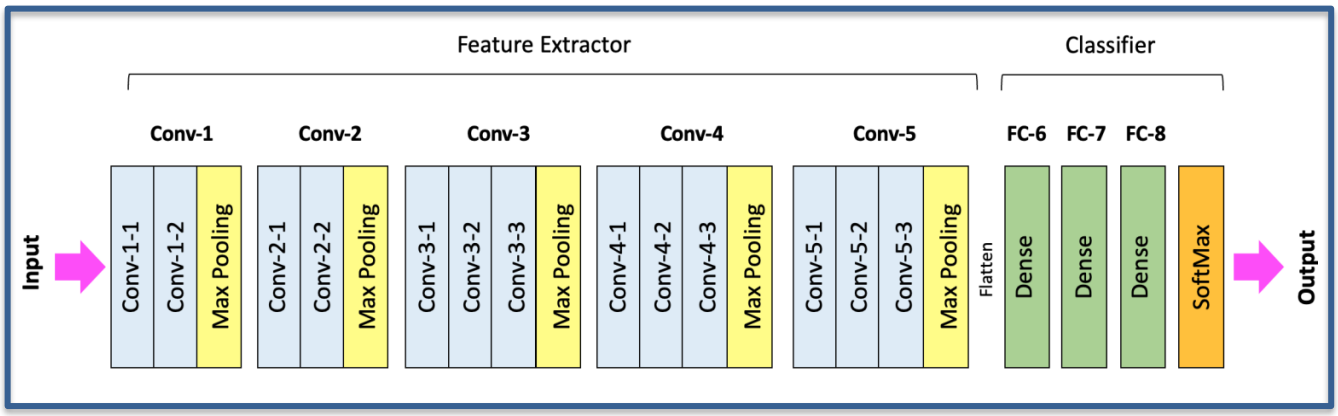


Fig. 26bis – Structure d'un réseau de neurones convolutifs (VGG-16) (Voulodimos et al., 2018)

Pour résumer, les CNNs conçus pour une tâche de classification d'images contiennent donc :

- Un extracteur de caractéristiques en amont.
- Des couches de pooling dans l'extracteur de caractéristiques.
- Un classificateur en aval.

L'extracteur de caractéristiques :

- Se compose de blocs convolutifs ayant une structure similaire composée d'une ou plusieurs couches convolutives suivies d'une couche de max *pooling*.
- Les couches convolutives extraient des caractéristiques de la couche précédente et stockent les résultats dans des cartes d'activation.
- Le nombre de filtres dans une couche convolutive est un choix de conception dans l'architecture d'un modèle et peut varier entre blocs convolutifs.
- La profondeur de l'output d'une couche convolutive (le nombre de cartes d'activation) est dictée par le nombre de filtres dans la couche.

Les couches de pooling :

- Sont souvent utilisées à la fin d'un bloc convolutif pour réduire la taille des cartes d'activation.
- Réduisent le nombre total de paramètres entraînables dans le réseau et, par conséquent, le temps d'entrainement ou de calcul requis.

Le classificateur :

- Transforme les caractéristiques extraites en probabilités de classes en utilisant une ou plusieurs couches entièrement connectées.
- Utilise une couche SoftMax pour normaliser l'output brut des couches entièrement connectées dans la plage [0,1].
- Les valeurs normalisées peuvent être interprétées comme la probabilité que l'image d'entrée corresponde à l'étiquette de classe pour chaque neurone de sortie.

Il est important de noter que l'exemple présenté dans ce chapitre ne comprenait que quelques couches et quelques filtres, alors que les modèles les plus complexes peuvent contenir jusqu'à 200 - 300 couches convolutives. On distingue un CNN classique d'un *Deep* CNN lorsque le nombre de couches devient important, cependant ce nombre ne semble pas être constant. De plus, les CNNs sont principalement conçus pour la classification d'images et ne sont pas réellement capables de détecter et localiser plusieurs objets sur une même image. Cela s'explique car la quantité d'objets à détecter sur l'image n'est pas constante, ce qui veut dire que l'output serait variable.

Il existe aussi les *Dense* CNNs, qui connectent chaque couche convolutive à toutes les couches précédentes, ce qui permet d'obtenir de meilleures performances avec moins de paramètres. Cette architecture les rend capable de réaliser de la segmentation sémantique.

Le fonctionnement de tous les modèles présentés dans ce travail se base sur ces couches convolutives, car ce sont elles qui permettent de traiter la complexité d'une image. L'architecture d'un CNN ne varie cependant pas uniquement en quantités de couches, et il existe de nombreuses autres approches au traitement d'images dans le but d'accélérer les temps de traitement ou favorisant la réalisation de tâches de détection plus complexes. On distingue les architectures directement issues des CNNs des modèles ayant une approche différente, comme les modèles de détection en un temps, qui sont plus rapides.

3.2.2. R-CNNs et dérivés

3.2.2.1. R-CNN (Region-based Convolutional Neural Network)

Un R-CNN est une évolution du CNN qui introduit les propositions par région afin d'être capable de détection d'objets. L'objectif est d'isoler chaque instance d'objet à détecter pour transférer ces images réduites dans un CNN et les classifier. Un algorithme de recherche sélective est donc chargé d'extraire 2000 régions d'intérêt, appelées propositions de régions. Ces propositions sont ensuite recadrées et redimensionnées pour correspondre à l'input d'un CNN, qui va ensuite classifier chaque proposition. Un SVM, ou *Support Vector Machine*, reprend ensuite les propositions de régions ainsi que les classifications du CNN pour préciser le cadrage autour de chaque objet de l'image. Si cette approche permet de localiser différents objets au sein d'une même image, l'extraction et le traitement individuel de 2000 régions pour chaque image passée dans le modèle rend l'entrainement de ce dernier extrêmement long.

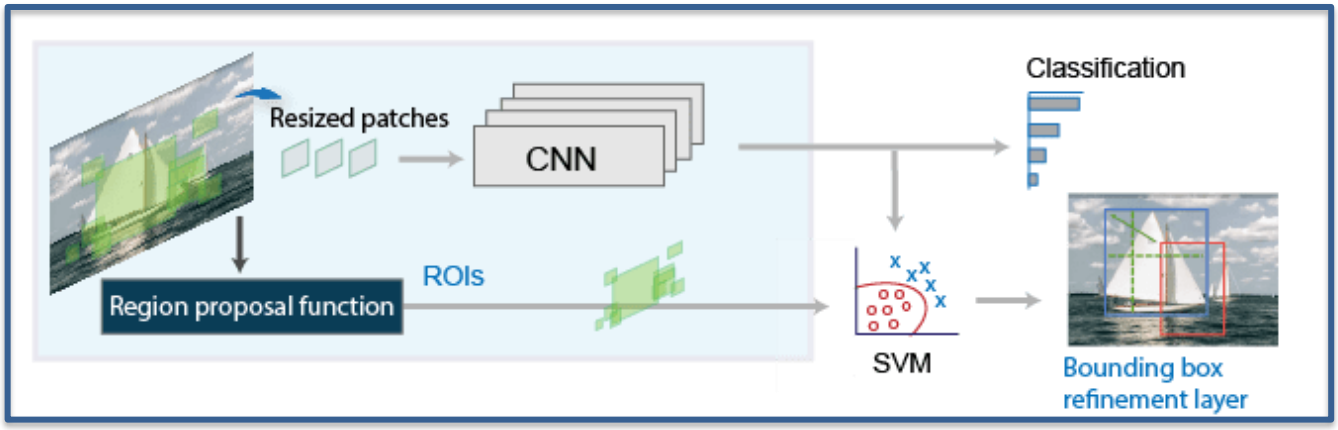


Fig. 34 – Architecture du modèle R-CNN (Mijwil et al., 2022)

3.2.2.2. Fast R-CNN

Fast R-CNN améliore l'efficacité de son prédécesseur en traitant l'image entière avec un CNN pour produire une carte d'activation au lieu de traiter chaque région individuellement. Les propositions de régions sont ensuite projetées sur cette carte d'activation dans une couche de *RoI* (*Region of Interest*) *Pooling*, qui identifie les propositions de régions dans la carte d'activation et les redimensionne pour que chaque région ait une taille identique. Le redimensionnement permet d'avoir des vecteurs de tailles égales lorsque les cartes d'activation sont aplaties pour être classifiées. Ce modèle est plus rapide car les régions qui se superposent sont traitées simultanément, et le CNN analyse l'image entière au lieu de 2000 régions (Mijwil et al., 2022).

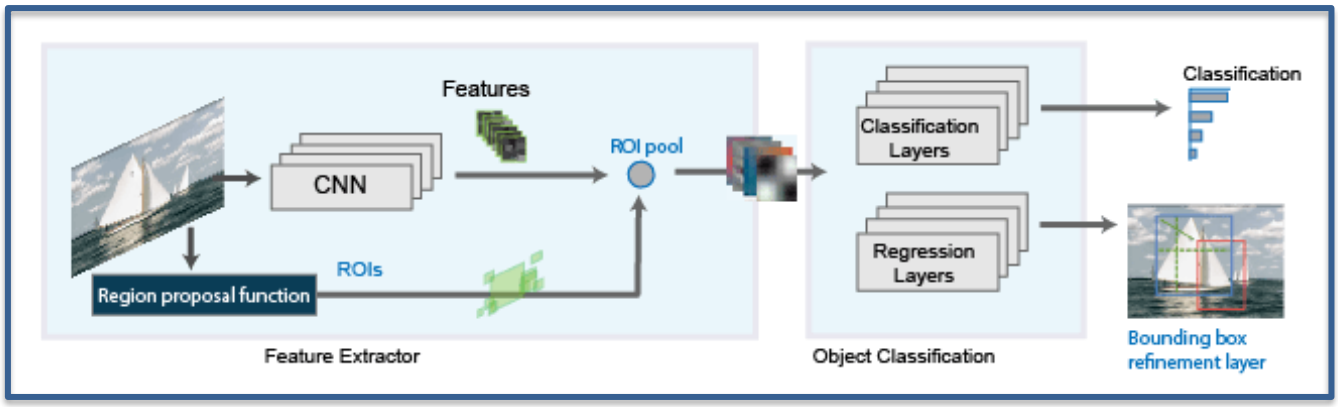


Fig. 35 – Architecture du modèle Fast R-CNN (Mijwil et al., 2022)

3.2.2.3. Faster R-CNN

Faster R-CNN intègre un réseau de propositions de régions (RPN) dans l'architecture du CNN. Le RPN génère des propositions de régions directement à partir de la carte d'activation de l'image entière. Ces propositions sont ensuite affinées et classifiées dans la couche de *RoI Pooling* de la même manière que dans le Fast R-CNN. Cette méthode est encore plus rapide et optimisée car elle élimine l'étape indépendante de génération de propositions de régions. Cette optimisation rend le Faster R-CNN capable de détection en temps réel.

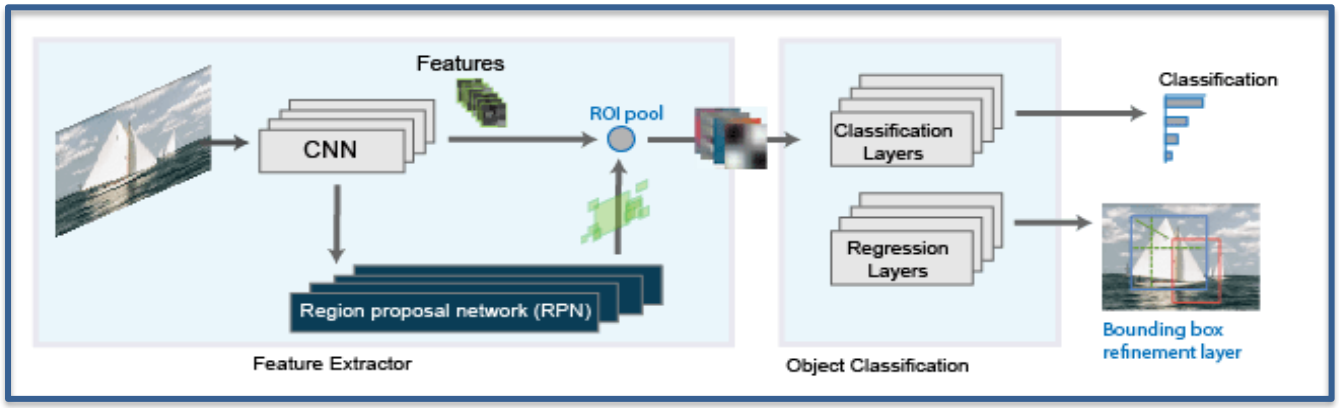


Fig. 36 – Architecture du modèle Faster R-CNN (Mijwil et al., 2022)

3.2.2.4. Mask R-CNN

C'est une extension directe de Faster R-CNN qui ajoute une branche au modèle pour prédire des masques de segmentation. Mask R-CNN remplace aussi la couche de *RoI Pooling* par une couche de *RoI Alignment*, qui est capable d'aligner précisément les cartes d'activation avec l'image d'entrée afin de réduire les erreurs et améliorer la précision globale du modèle. Le réseau FPN (*Feature Pyramid Network*) incorporé est responsable de la création des masques de segmentation pour chaque instance d'objet détecté sur l'image. Cette opération permet de localiser plus précisément les objets sur l'image, en marquant clairement l'espace qu'ils occupent au lieu d'un cadrage lors de la détection d'objets. Contrairement aux trois modèles précédents, Mask R-CNN est avant tout prévu pour réaliser de la segmentation d'instances. Les opérations de *RoI Alignment* et le réseau FPN permettent une meilleure précision du modèle mais augmentent également les exigences en matière de puissance de calcul. Le modèle est donc légèrement moins rapide que son prédécesseur mais reste proche du temps réel (Lee et al., 2022).

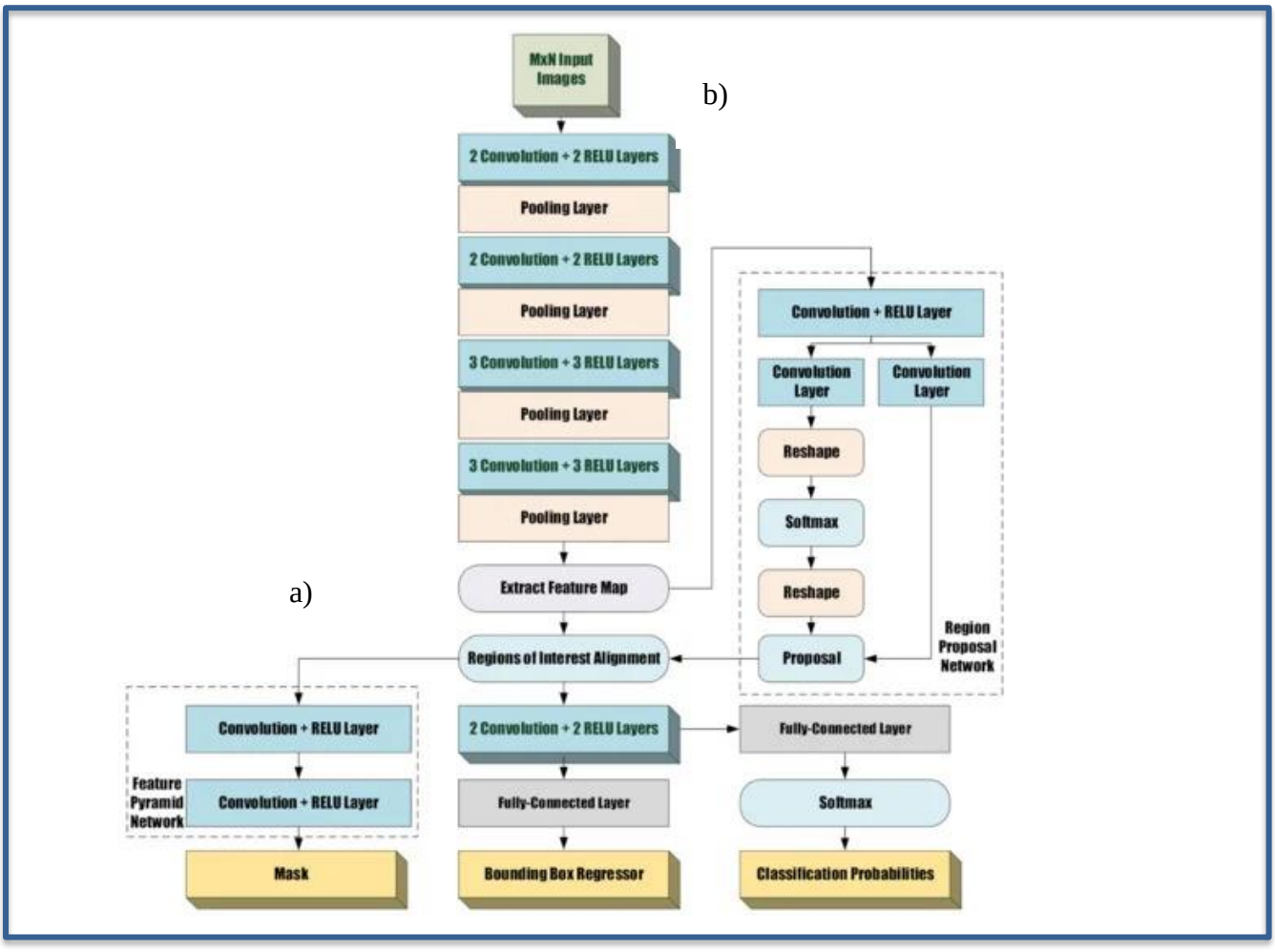


Fig. 37 – Architecture du modèle Mask R-CNN : a) extension réseau FPN, b) architecture de modèle Faster R-CNN (Lee et al., 2021)

3.2.3. Autres

3.2.3.1. SSD (Single-Show Multi-Box Detector)

SSD est un modèle de détection d'objets en un seul temps. Contrairement aux modèles qui nécessitent la génération de propositions de régions, SSD simplifie ce processus en éliminant complètement l'étape. L'extraction de caractéristiques est basée sur un CNN comme le VGG-16, qui va produire des cartes d'activation à différentes échelles. La classification est remplacée par des couches de convolution supplémentaires chargées de localiser les objets à détecter et une couche de *Non-Maximum Suppression* (NMS). Les cartes d'activation du CNN et les données des couches supplémentaires sont ensuite transmises dans la couche de NMS, qui permet de supprimer les prédictions redondantes et émettre les classifications finales.

Les couches convolutives supplémentaires balaient les cartes d'activation issues du CNN avec différentes tailles et formes de cadrages qui détectent les objets. De nombreux cadrages sont susceptibles de détecter un même objet et c'est la fonction IoU qui en détermine la précision. Le NMS va ensuite détecter les cadrages se chevauchant et supprimer ceux dont le score d'IoU est le plus faible, laissant la meilleure prédiction en output.

La simplicité du modèle et son fonctionnement en un temps le rendent rapide à entrainer et capable de détection en temps réel. Il est cependant moins performant pour la détection de petits objets (Liu et al., 2016) (Lee et al., 2022).

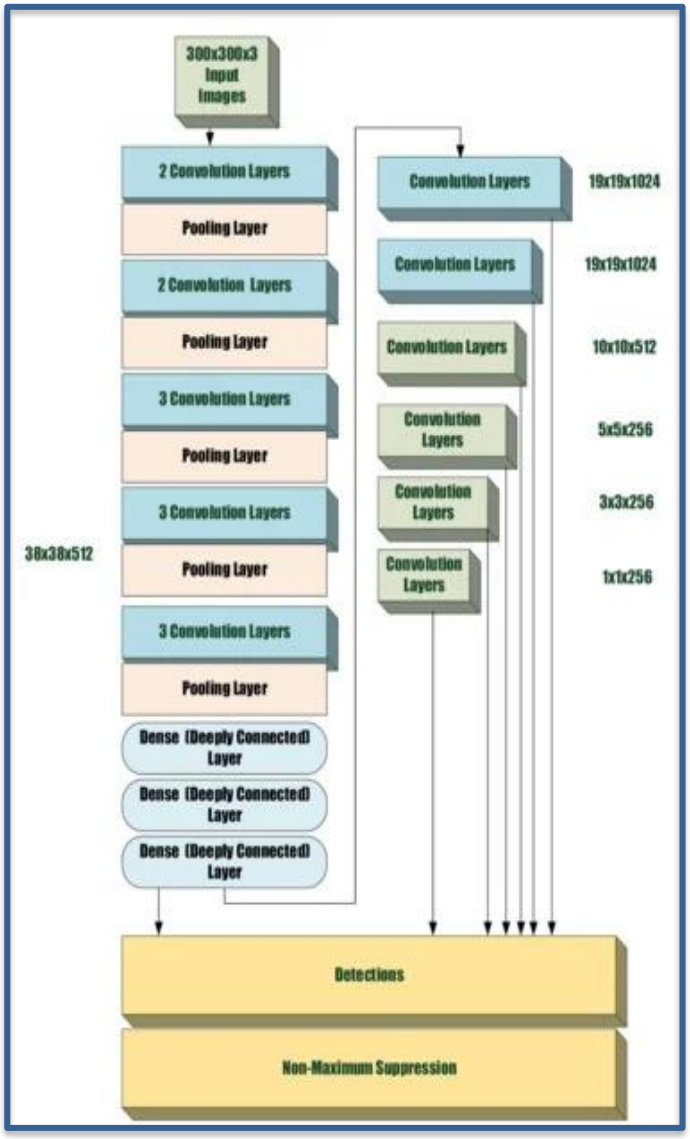


Fig. 38 – Architecture du modèle SSD (Lee et al., 2021)

3.2.3.2. YOLO (You Only Look Once)

Similaire à un SSD, YOLO est un modèle de détection d'objets en un seul temps. Cependant son architecture est différente, le modèle divise l'image en une grille de cellules de taille SxS et prédit simultanément les cadrages et les probabilités de classe pour chaque cellule de la grille. Si le centre d'un objet se situe dans une cellule, cette dernière est responsable pour la détection de l'objet. Une fois que la grille de prédictions est mise en place, chaque cellule prédit des cadrages de délimitation associés à un score de confiance indiquant la certitude que le cadrage contient un objet ou pas. Simultanément, les couches convolutives du modèle prédisent l'appartenance de chaque cellule à une classe d'objet à détecter. Dans la partie de classification du modèle, les données sont fusionnées pour associer la classe propre à la cellule et l'objet qu'elle a détecté. Les cadrages redondants ayant un score de confiance inférieur sont supprimés. YOLO est également capable de détection en temps réel grâce à son architecture simple mais, comme le SSD, a davantage de difficultés à détecter les petits objets (Redmon et al., 2016).

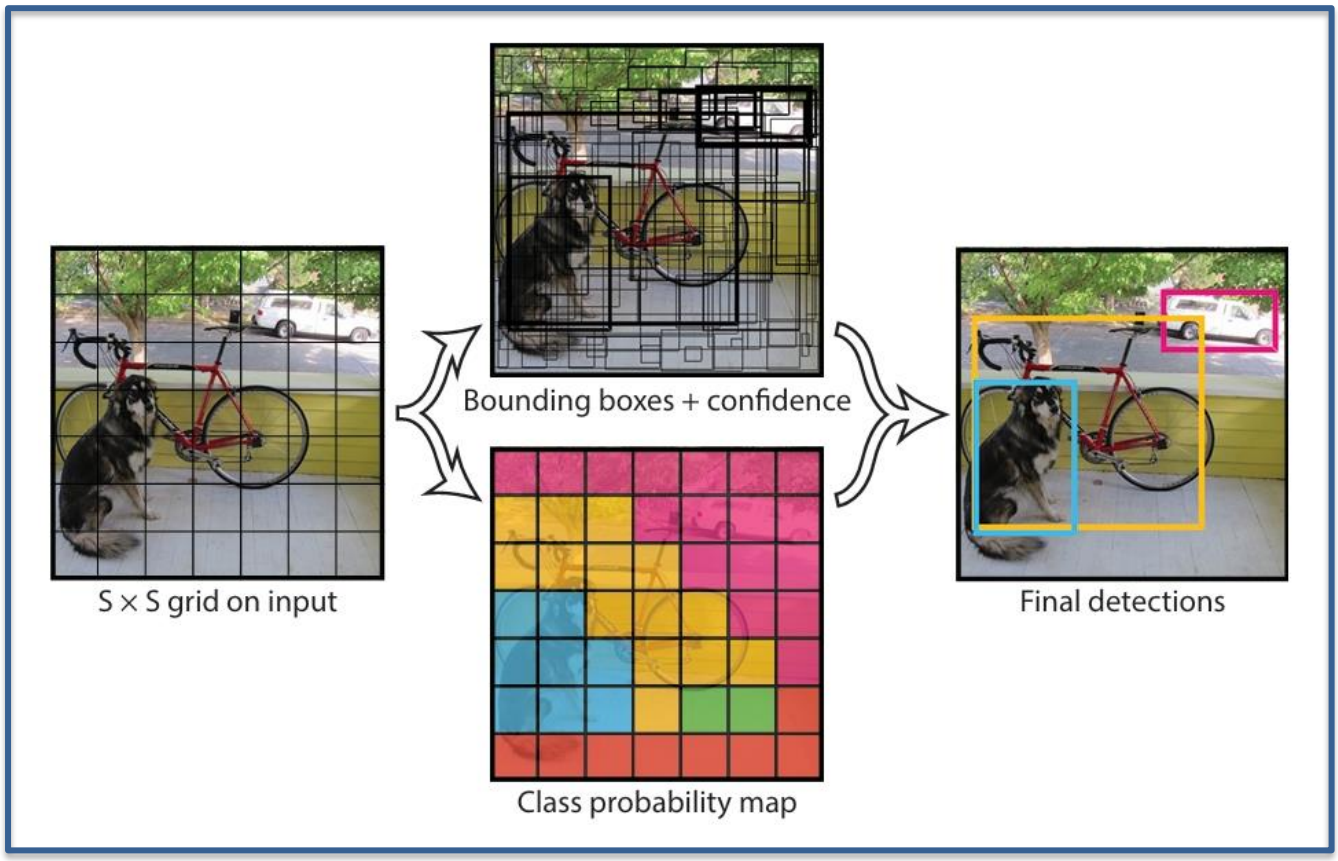


Fig. 39 – Fonctionnement de la division par cellules du modèle YOLO (Redmon et al., 2016)

3.2.3.3. FCN (Fully Convolutional Network)

Les FCNs sont conçus pour les tâches de segmentation sémantique. Ils remplacent les couches entièrement connectées habituellement trouvées dans la partie de classification d'un CNN par d'autre couches convolutives. Cette approche permet aux FCNs de conserver les informations spatiales et de générer des prédictions denses sur chaque pixel de l'image d'entrée. Dans l'extracteur de caractéristiques, les images passent par plusieurs couches de convolution et de *pooling*, tout comme dans les réseaux CNN de classification d'images. Les cartes d'activation réduites sont ensuite déconvolutionnées par *upsampling* et connectées à des couches de convolution situées plus haut dans le modèle, permettant des prédictions de plus en plus localisées. Cette opération est réalisée jusqu'à ce que l'output ait la même définition que l'image d'entrée et à l'attribution d'une classe d'objets pour chaque pixel (Long et al., 2015).

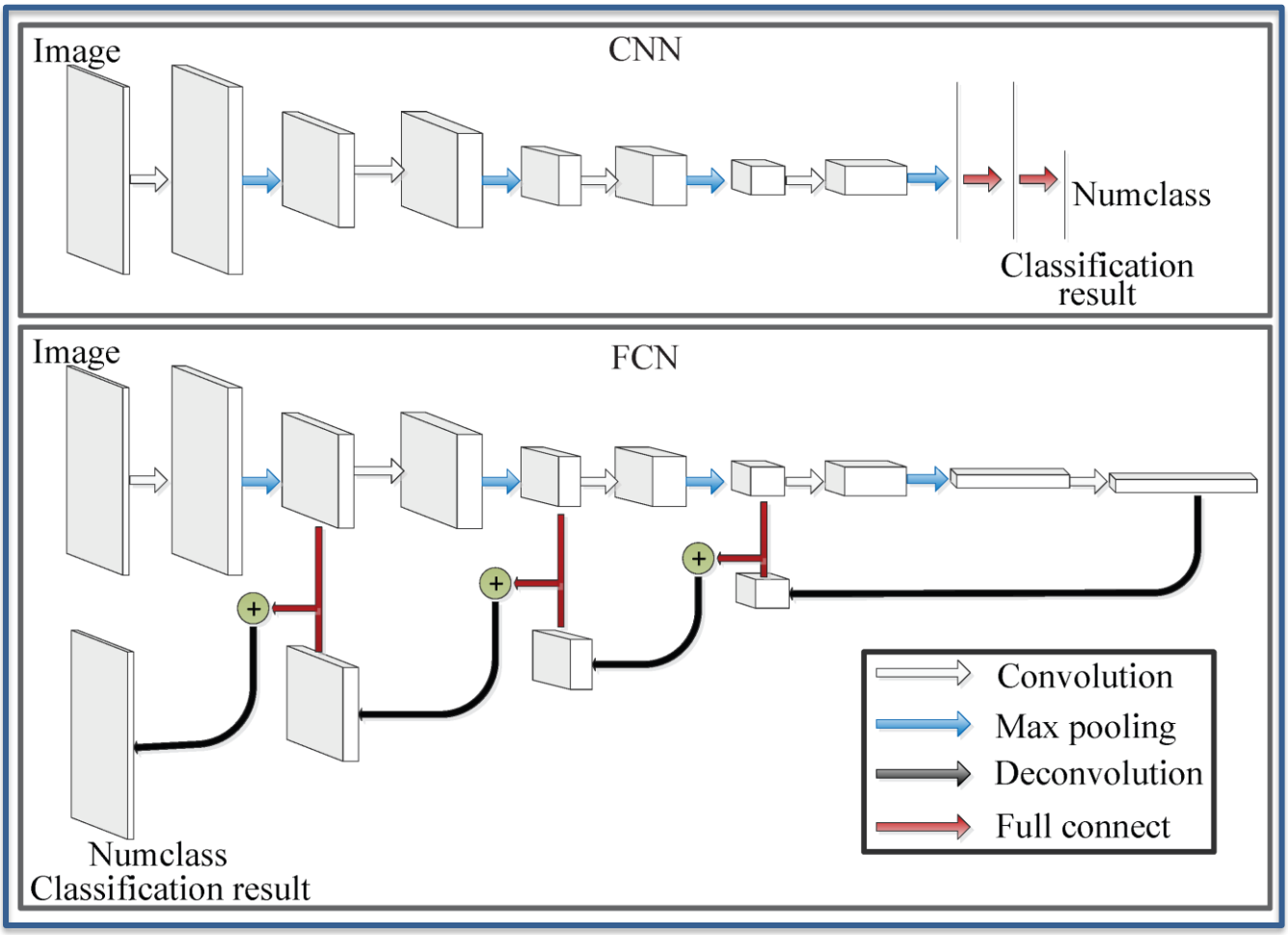


Fig. 40 – Architecture du modèle FCN

3.2.3.4. SOLO (Segmenting Objects by Locations)

SOLO est spécialisé dans la segmentation d'instances. Il se distingue d'autres modèles qui reposent sur une détection préalable des objets, suivie ensuite de leur segmentation pour localiser les objets (Mask R-CNN). L'objectif central de SOLO est de reformuler la segmentation d'instances en deux problèmes de prédictions simultanés. L'image d'entrée est divisée en une grille de cellules de taille SxS. Si le centre d'un objet se situe dans une cellule, cette dernière est responsable de prédire la catégorie de cet objet et en segmenter l'instance. L'opération de segmentation sémantique est possible car une architecture FCN sert d'abord de base au modèle. Un FPN est également employé afin de distinguer les instances d'objets de tailles différentes et de les classifier par catégories d'instances (Wang et al., 2020b).

Pour chaque cellule, une prédiction de classe pour chaque instance d'objet est réalisée. En parallèle de cette opération, chaque cellule dont la prédiction indique la présence d'un objet génère le masque d'instance lui correspondant. La prédiction de la catégorie et le masque sont naturellement associés par leur cellule de référence dans la grille. Sur cette base, nous pouvons former directement le résultat final de la segmentation d'instance pour chaque grille. Les données brutes sont ensuite rassemblées et fusionnées. Enfin, on applique un NMS pour supprimer les détections redondantes les moins précises et ainsi obtenir les résultats finaux de la segmentation d'instance (Wang et al., 2020b).

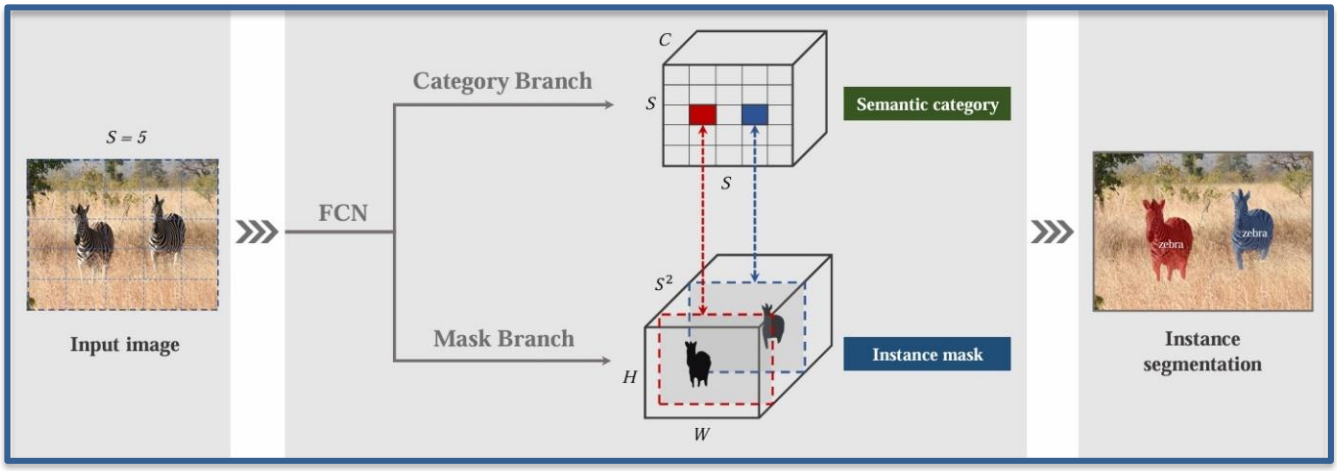


Fig. 41 – Fonctionnement des doubles prédictions du modèle SOLO (Wang et al., 2020b)

III. ANALYSE DES ÉTUDES

1. Tableaux comparatifs

Il est essentiel d'analyser les différentes études réalisées ces dernières années afin de déterminer si la détection de pathologies a atteint un niveau de précision suffisant, et identifier les aspects importants du déploiement en situation réelle d'un modèle. La comparaison de ces études a donc pour but d'établir les points nécessaires à la confirmation de l'hypothèse.

Répondre à cette hypothèse revient en premier temps à répondre aux questions suivantes :

- Quelles sont les **limitations au sens large**, du traitement par intelligence artificielle d'images issues du patrimoine bâti ? Est-il possible de réduire leur impact ?
- La **précision** des différents modèles est-elle satisfaisante pour envisager une intégration dans les processus d'inspection actuels ?
- Quels sont les **avantages et limitations de chaque modèle** ? Quelles sont les solutions envisagées pour réduire ces limitations ?
- Quels sont les **formations et outils** nécessaires à l'utilisation de ces technologies ?

Cette analyse des travaux réalisés se présente sous forme de tableaux comparatifs. Le tableau 1 recense l'ensemble des études observées, avec pour chacune, leur cas d'étude, la tâche de Computer Vision réalisée, le modèle utilisé, les objectifs de l'étude et les données qui en découlent. Afin de comparer les études de traitement d'images en temps réel indépendamment de la tâche réalisée, la dernière colonne de ce tableau indique si la détection se fait en temps réel. Pour plus de lisibilité, les catégories de tâches sont exprimées comme tel dans le tableau 1 :

- Classification d'images : CI
- Détection d'objets : DO
- Segmentation sémantique : SS
- Segmentation d'instances : SI

Les tableaux 2, 3 et 4 reprennent uniquement les études traitant respectivement de classification d'images, de détection d'objets et de segmentation (sémantique ou d'instance). Le tableau 5 compare quant à lui les études de modèles en temps réel.

Chacun des tableaux 2 à 5 indique le modèle utilisé, la taille du dataset et des images utilisées, le temps d'entrainement du modèle, les différentes typologies détectées, le temps de calcul, les limitations énoncées et une métrique d'analyse de performance comme le mAP. Si ce dernier n'a pas été exprimé dans l'article, c'est la valeur d'exactitude qui est représentée. Le temps de calcul des modèles de détection en temps réel est formulé en *Frames per Second*, ou FPS, et représente la quantité d'images traitées en une seconde.

Une croix (X) indique que l'information n'a pas été exprimée dans la recherche, tandis qu'un astérisque (*) montre que la valeur n'était pas exprimée mais a pu être calculée à partir des informations présentes dans l'article.

Il est important de noter que certaines études comparent plusieurs modèles et leurs générations ainsi que des paramètres comme la taille et la quantité d'images dans le dataset, la longueur d'entrainement et le temps de calcul. Dans les tableaux, seuls les modèles ayant le mAP le plus élevé sont indiqués. Les autres paramètres représentés sont donc les plus favorables lorsque plusieurs ont été testés par les auteurs. De plus, certaines études ne concernent pas directement la détection de pathologies ou le patrimoine, mais contribuent néanmoins à approfondir la recherche dans au moins l'un de ces domaines.

1.1. Tableau 1 : Recensement des études analysées

Référence	*Cas d'étude*	*Tâche réalisée*	*Modèle*	*Objectif*	*Données de sortie*	*Temps réel ?*
[1] Perez et al. (2019)	X	CI & SI	CNN (VGG-16 network)	Classement différents degrés d'humidité	Dégâts classifiés et localisés au niveau pixel	Non
[2] Cha et al. (2018)	X	DO	Faster R-CNN	Détection pathologies béton et acier	Dégâts classifiés et localisés au niveau d'instances	Non
[3] Wang et al. (2019)	Grande Muraille de Chine	DO	Faster R-CNN	Détection pathologies et acquisition images par mobile crowdsourcing (MCS)	Dégâts classifiés et localisés au niveau d'instances	Non
[4] Pathak et al. (2021)	Temple Ayutthaya Wat Phra Si Sanphet, Thaïlande	DO	Faster R-CNN	Détection pathologies sur base d'images extraites d'un nuage de point texturé	Images extraites du nuage de points par SS. Dégâts ensuite classifiés et localisés au niveau d'instances	Non
[5] Matrone et al. (2020)	3 scènes issues d'ArCH dataset	SS	DGCNN	Détection éléments directement depuis nuage de points	Modèle 3D segmenté sémantiquement	Non
[6] Yang et al. (2023b)	Patrimoine mondial à Macau	DO	YOLOv4	Détection pathologies sur différents murs de briques	Dégâts classifiés et localisés au niveau d'instances	Non
[7] Dini et al. (2023)	Lasem, Indonésie	CI	CNN (VGG-16 network)	Classement degrés de dégradation des façades	Images classifiées selon leur dégradation	Non
[8] Samhouri et al. (2022)	Al-Salt, Jordanie	CI & DO	CNN (VGG-16 network)	Classement différentes pathologies	Dégâts classifiés	Non
[9] Wei et al. (2023)	Images de façades de bâtiments	DO	YOLOv7	Détection pathologies sur façades	Dégâts classifiés et localisés au niveau d'instances	Oui
[10] Idjaton et al. (2023)	Château de Chaumont-sur-Loire, France	DO	YOLOv5	Détection pathologies sur base d'images de photogrammétrie	Dégâts classifiés et localisés au niveau d'instances	Oui

Référence	*Cas d'étude*	*Tâche réalisée*	*Modèle*	*Objectif*	*Données de sortie*	*Temps réel ?*
[11] Hallee et al. (2021)	Murs de briques construits en laboratoire	CI	CNN	Classement binaire état mur de maçonnerie	Images classifiées selon bon état/fissuré	Non
[12] Marín-García et al. (2023)	Murs de briques en Espagne	DO	YOLOv5	Détection pathologies et constat de l'intervention nécessaire	Dégâts classifiés et localisés au niveau d'instances	Oui
[13] Pratibha et al. (2024)	Bhubaneswar et Cuttack, Inde	DO	YOLOv5	Détection pathologies de fissures sur différents matériaux	Dégâts classifiés et localisés au niveau d'instances	Oui
[14] Ma et al. (2022)	Université de Shen-yang Jianzhu, Chine	DO	YOLOv5	Détection pathologies de fissures sur bois	Dégâts classifiés et localisés au niveau d'instances	Oui
[15] Kwon & Yu (2019)	Patrimoine national en pierre, Corée du Sud	DO	Faster R-CNN	Détection pathologies sur pierre	Dégâts classifiés et localisés au niveau d'instances	Non
[16] Hatır et al. (2021)	Site archéologique de Hattusa, Turquie	SI	Mask R-CNN	Détection pathologies sur pierre et identification de leur forme	Dégâts classifiés et localisés au niveau pixel	Non
[17] Zou et al. (2019)	Cité interdite, Pékin	DO	Faster R-CNN	Détection et quantification d'éléments manquants	Dégâts classifiés, quantifiés et localisés au niveau d'instances	Non
[18] Abed et al. (2020)	Dataset patrimoine bâti	CI	CNN (ResNet 18)	Détection d'éléments du patrimoine	Eléments classifiés	Non
[19] Kumar et al. (2020)	92 sites patrimoniaux autour du globe	CI	CNN	Détection dégâts après catastrophes naturelles, sur base d'images de réseaux sociaux	Images classifiées selon patrimoine intact, endommagé ou non-patrimoine intact, endommagé	Non

Référence	*Cas d'étude*	*Tâche réalisée*	*Modèle*	*Objectif*	*Données de sortie*	*Temps réel ?*
[20] Perez & Tah (2021)	X	DO	SSD (MobileNet)	Détection pathologies en temps réel sur appareil mobile	Dégâts classifiés et localisés au niveau d'instances	Oui
[21] Jiang & Zhang (2020)	Bâtiment de 3 étages	DO	SSD (MobileNet)	Détection pathologies en temps réel avec drone grimpant sur la façade	Dégâts localisés au niveau d'instances, taille des dégâts connue	Oui
[22] Li et al. (2019)	X	SS	FCN	Détection pathologies sur béton	Dégâts classifiés et localisés au niveau pixel	Non
[23] Dung et al. (2019)	Dataset public	SS	FCN	Détection pathologies sur béton	Dégâts classifiés et localisés au niveau pixel	Non
[24] Kalfarisi et al. (2020)	Dataset public	SI	Mask R-CNN	Détection pathologies sur béton et mise en parallèle avec modèle 3D texturé (photogrammétrie)	Dégâts classifiés et localisés au niveau pixel, modèle 3D segmenté sémantiquement	Non
[25] Kim et al. (2019)	Mur de béton à l'université de Séoul	SS	Mask R-CNN	Détection pathologies sur béton et quantification de leur taille	Dégâts classifiés, quantifiés et localisés au niveau pixel	Non
[26] Lamas et al (2021)	Façades de monuments issues du dataset MonuMAI	DO & CI	Faster R-CNN	Détection éléments architecturaux et classification styles	Styles classifiés au niveau d'image, éléments architecturaux classifiés et localisés au niveau d'instance	Non
[27] Liu et al. (2022)	Château de Bothwell, Ecosse	SS	DeepLabV3+	Détection végétation et mise en parallèle avec modèle 3D texturé (photogrammétrie)	Dégâts classifiés et localisés au niveau pixel, modèle 3D segmenté sémantiquement	Non
[28] Mishra et al. (2022a)	Tombes Dadi-Poti, New Delhi	DO	YOLOv5	Détection différentes pathologies	Dégâts classifiés et localisés au niveau d'instances	Oui

1.2. Tableau 2 : Classification d'images

Référence	*Modèle*	*Dataset*	*Temps d'entrainement*	*Typologies détectées*	*Temps de calcul*	*mAP / Exactitude*	*Limitations*
[6] Yang et al. (2023b)	YOLOv4	1000 images	X	Fissures, élément manquant, érosion, décoloration	X	85.7%	Angle d'incidence fort, textures
[7] Dini et al. (2023)	CNN (VGG-16 network)	260 images (224x224)	X	4 classifications : bon état, dégâts légers, moyens, sévères	X	60%	Dataset trop faible, difficulté, environnement limite la prise d'images pertinentes
[11] Hallee et al. (2021)	CNN	2542 images (512x512)	X	Fissures	X	92.5%	Conditions contrôlées rendent le modèle moins performant pour des conditions réelles
[18] Abed et al. (2020)	CNN (ResNet 18)	500 images (128x128)	X	10 éléments : autel, abside, clocher, colonne, dôme (intérieur, extérieur), arc-boutant, gargouille, vitrail, voûte	X	95.6%	X
[19] Kumar et al. (2020)	CNN	10281 images	X	Patrimoine ou non-patrimoine, ensuite bon état ou endommagé	X	94%	Moins de 10% des images postées pendant une catastrophe naturelle sont pertinentes, trop de diversité crée ambiguïté pour le modèle

1.3. Tableau 3 : Détection d'objets

Référence	*Modèle*	*Dataset*	*Temps d'entrainement*	*Typologies détectées*	*Temps de calcul*	*mAP / Exactitude*	*Limitations*
[2] Cha et al. (2018)	Faster R-CNN	2366 images (500x375)	4h	Fissures béton, corrosion acier, délamination acier	30ms/ image	87.8%	Forte luminosité
[3] Wang et al. (2019)	Faster R-CNN	610 images	3h	Ecaillage, fissures, efflorescence	60ms/ image	78.2%	Taille du dataset, faible précision avec angle d'incidence fort
[4] Pathak et al. (2021)	Faster R-CNN	7500 images (512x512)	X	Ecaillage, fissures	X	58.19%	mAP faible car testé sur monument différent du dataset
[8] Samhouri et al. (2022)	CNN (VGG-16 network)	1024 images (224x224)	Quelques heures	Erosion, décoloration, élément manquant, vandalisme	430ms/ image	96.0%	Puissance de calcul nécessaire, défauts analysés peu complexes
[15] Kwon & Yu (2019)	Faster R-CNN	400 images	X	Fissures, éléments manquants, végétation, écaillage	X	94.6%	Superposition des détections
[17] Zou et al. (2019)	Faster R-CNN	435 images (3264x2448)	X	Différents éléments manquants, finitions de tuiles en toiture	X	85%	Précision diminue si distance augmente, double quantification d'éléments répétés dans plusieurs images.
[26] Lamas et al (2021)	Faster R-CNN	1514 images	X	4 styles : baroque, gothique, renaissance, hispanique et 15 classes d'éléments	X	90%	X

1.4. Tableau 4 : Segmentation sémantique et d'instances

Référence	*Modèle*	*Dataset*	*Temps d'entrainement*	*Typologies détectées*	*Temps de calcul*	*mAP / Exactitude*	*Limitations*
[1] Perez et al. (2019)	CNN (VGG-16 network)	2622 images (224x224)	X	Ecaillage, infiltration d'eau et moisissure	X	87.5%	Une seule catégorie, luminosité et orientation peu explorés
[5] Matrone et al. (2020)	DGCNN	3 scènes de nuages de points (ArCH dataset)	46h	Mur, toiture, colonne, escaliers…	30 minutes	80-91%	Temps de calcul et d'entrainement, puissance de calcul nécessaire
[16] Hatır et al. (2021)	Mask R-CNN	2460 images (800x600)	3h 52 min	Végétation, fissures, dégâts d'impacts, éléments manquants, micro-karsts, écaillage	X	99%	Ombres du relief
[22] Li et al. (2019)	FCN	2750 images (504x376)	4h 12 min	Fissures, écaillage, efflorescence, trous	50ms/ image	91.6%	Profondeur dégâts pas prise en compte car 2D, erreurs peuvent être limitées par un plus grand dataset
[23] Dung et al. (2019)	FCN	40000 images (227x227)	X	Fissures	72ms/ image*	89.3%	Evaluation difficile de la densité des dégâts
[24] Kalfarisi et al. (2020)	Mask R-CNN	1250 images	X	Fissures	X	78%	Précision inversement proportionnelle à la vitesse de calcul
[25] Kim et al. (2019)	Mask R-CNN	376 images	X	Fissures	X	76.15%	Quantification de la taille des petites fissures limitée par la résolution de l'image
[27] Liu et al. (2022)	DeepLabV3+	113 images (900x900)	1h	7 classes : végétation, ciel, et éléments architecturaux	600ms/ image	93.7%	Petits objets difficiles à détecter, forte quantité de faux positifs

1.5. Tableau 5 : Détection en temps réel, toutes tâches confondues

Référence	*Modèle*	*Dataset*	*Temps d'entrainement*	*Typologies détectées*	*FPS*	*mAP / Exactitude*	*Limitations*
[9] Wei et al. (2023)	YOLOv7	1907 images (640x640)	X	Délamination, écaillage, éléments manquants	76	81.6%	X
[10] Idjaton et al. (2023)	YOLOv5	1012 images (256×256)	X	Ecaillage	45	81%	Petits défauts difficiles à détecter, surtout avec des plus grands défauts sur la même image
[12] Marín-García et al. (2023)	YOLOv5	765 images (608x608)	X	Efflorescence légère et sévère	50*	89%	Taille de dataset, une seule classe de pathologies, différentes conditions non envisagées (météo, angle photo)
[13] Pratibha et al. (2024)	YOLOv5	4000 images (224x224)	X	Fissures	160*	92%	Conditions de prises d'images contrôlées, ombre de l'environnement gène la détection
[14] Ma et al. (2022)	YOLOv5	474 images (640x640)	13h	Fissures	166	92.9%	X
[20] Perez & Tah. (2021)	SSD (MobileNet)	875 images (300x300)	5 jours	Fissures, moisissure, détérioration peinture, et infiltration d'eau	X	80%	Résolution des images de fissures trop faible et en trop petite quantité dans le dataset
[21] Jiang & Zhang (2020)	SSD (MobileNet)	1330 images (640x480)	X	Fissures	6	94.5%	Proximité avec la façade, besoin de surface plane, faible résolution de la caméra
[28] Mishra et al. (2022a)	YOLOv5	10291 images (416x416)	1h 53 min	Fissures, écaillage, décoloration, briques exposées	X	93.7%	X

2. Approches intéressantes

2.1. Acquisition des images

L'acquisition des images nécessaires à l'entrainement et aux tests de chacun des modèles étudiés est la première étape à toute détection automatique. On distingue deux approches souvent appliquées : la prise de photographies par les experts sur le site ou l'acquisition d'images issues d'autres datasets / capturées par des tiers. Dans les deux cas, les outils sont principalement des appareils photos, caméras ou smartphones, mais l'utilisation d'un drone peut être conseillée en fonction de l'accessibilité des pathologies.

La première approche, la prise de photographies directes, permet aux opérateurs de capturer les images pertinentes au modèle et de contrôler l'environnement dans lequel il doit opérer. Par exemple, pour les études [11], [12] et [13], ciblées sur les murs de briques, des conditions idéales ont été choisies pour la préparation du dataset. Dans l'étude [12], une distance et un angle similaire ont été maintenu pour chaque image prise. Pour l'étude [13], les auteurs ont exclu les images représentant d'autres pathologies que celles étudiées, ayant une couleur hétérogène (peinture ou décoloration trop importante), présentant des faces de briques déformées, ou n'ayant pas des couches de mortier uniformes. Les auteurs de l'article [11] ont simplement construit des segments de murs de brique en laboratoire et photographié les fissures créées par leurs manipulations. Limiter les conditions de prises d'images a souvent pour conséquence d'obtenir des modèles performants mais peu applicables à des conditions réelles, qui sont souvent beaucoup plus diverses. Dans une réflexion inverse, certains chercheurs comme ceux de l'article [17] varient les conditions de prises d'images afin d'entrainer le modèle aux possibles variations de l'environnement. Dans l'étude [6], on prend en compte les différentes typologies, tailles de joints et textures des briques pour entrainer le modèle à détecter les pathologies sur différents types de murs.

A l'inverse, la deuxième approche revient à acquérir les images indirectement, soit sur base d'un dataset existant, soit par le biais de science citoyenne. La complexité des différentes pathologies à détecter rend la création d'un dataset public de pathologies compliqué, cependant certains existent pour des pathologies précises, notamment les fissures de béton car elles sont observées dans les structures civiles comme les ponts.

Dans le cas des sciences citoyennes, il est impossible de limiter les conditions dans lesquelles sont prises les images, vu qu'elles sont prises par différents opérateurs, ignorant généralement les paramètres à

prendre en compte pour faciliter la détection des pathologies par le modèle. La luminosité, le format et la résolution de l'image, l'angle de prise de vue, la distance et même les conditions météos peuvent donc fortement varier entre chaque image. Le format et la résolution impactent en général peu les modèles, car leur code compresse souvent les images à un format précis. Cependant, une résolution trop basse ou un éloignement trop important de l'objet à détecter diminuent les performances du modèle.

Si les sciences citoyennes se distinguent en plusieurs sous catégories, seul le crowdsourcing a réellement été appliqué dans les études analysées. Il permet aux citoyens de participer comme collecteurs de données (ici, d'images). N'ayant pas trouvé de terme plus précis, je distinguerais cependant le crowdsourcing participatif des études [3], [26] et [27] au crowdsourcing indirect de [19].

Le crowdsourcing participatif implique directement le citoyen dans la prise d'images. L'étude [3] met en place une application et un site permettant au citoyen de se localiser sur la Grande Muraille de Chine et d'indiquer l'état de la section visitée sous forme de questionnaires et d'envoi d'images analysées par Faster R-CNN. Dans le cas d'une structure aussi étendue et fréquemment visitée, cela évite aux experts de parcourir des centaines de kilomètres et réagir aux emplacements qui subissent le plus de dégradation.

De manière similaire, les auteurs de [26] ont développé une application mobile capable d'identifier et localiser sur l'image des éléments architecturaux ainsi que le style du monument auquel ces éléments appartiennent. L'entrainement initial a été réalisé sans crowdsourcing, mais l'envoi d'images par les citoyens utilisant l'application participe activement à élargir le dataset du modèle, et donc sa précision.

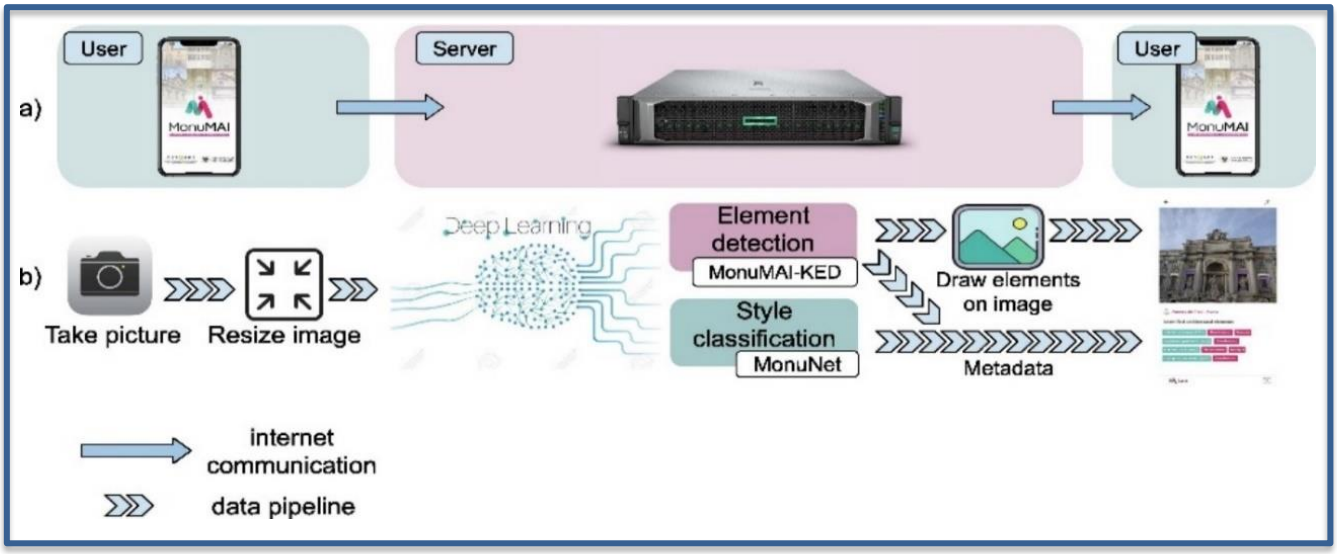


Fig. 42 – Fonctionnement de l'application MonuMAI : a) communication internet, b) étapes de traitement de l'image (Lamas et al., 2021)

Dans le travail [27], on utilise pour l'entrainement du modèle les images issues du projet participatif Monument Monitor en Ecosse, créé en 2018 par Brigham (2022). Ce projet englobe des dizaines de sites patrimoniaux écossais et indique sous forme d'enseignes visibles sur les sites ce qui est surveillé et quels types de photos sont demandées. Chaque visiteur peut ensuite fournir les images prises aux autorités par mail ou via les réseaux sociaux. Ceci a pour avantage de cibler les éléments à photographier et d'informer les participants des paramètres importants (par exemple, un angle idéal ou une distance maximale).

L'approche indirecte revient à utiliser les images postées en ligne sur les réseaux sociaux, alors qu'elles n'ont pas été prises dans ce but. Evidemment, de nombreuses photos ne sont pas pertinentes car ne prennent pas en compte les paramètres énoncés précédemment. [19] utilise les images des réseaux sociaux publiées lors de catastrophes naturelles comme un tremblement de terre pour détecter si un bâtiment est patrimonial, et s'il a subi des dégâts importants. Cela permet d'établir rapidement si un monument a besoin d'une intervention, pour assurer sa stabilité structurelle et ainsi permettre de le conserver.

Dans un autre registre, [21] cherche à faciliter la capture d'images à l'aide d'un drone grimpant sur la façade du bâtiment et prenant les images très proches des pathologies, même à des endroits inaccessibles par un opérateur humain. Cela réduit des limitations causées par l'éloignement de l'objet ou l'angle d'incidence, car ces derniers restent constants. Une automatisation complète du parcours du drone permettrait d'accélérer le processus. Il existe également des robots grimpeurs qui ont un fonctionnement similaire. Cependant, l'efficacité de ces appareils dépend fortement de l'homogénéité de la façade et ils deviennent moins adaptés si cette dernière n'est pas lisse.

2.2. Complémentarité avec les technologies actuelles

Un intérêt majeur de la photogrammétrie pour la détection de pathologies est la prise de nombreuses photos selon des paramètres souvent similaires à ceux employés pour l'entrainement des modèles. La distance et la résolution sont des facteurs importants pour un modèle 3D précis mais également pour obtenir des prédictions rigoureuses.

Dans les études [4] et [10], les auteurs ont utilisé des modèles 3D préalablement réalisés par photogrammétrie et scans lasers du monument étudié. Puisque le modèle englobait un site étendu dans l'étude [4], une première segmentation sémantique a permis d'isoler les images contenant des pixels de « bâtiment » ou « mur » d'autres images du site (fig. 43a). Cette étape a permis de rapidement nettoyer le modèle 3D et de conserver les images pertinentes pour le modèle de détection (fig. 43b). Les éléments du modèle 3D furent ensuite extraits et redimensionnés à 512x512 pixels pour être traités par le modèle (fig. 43c). Dans les deux cas, les pathologies détectées étaient ensuite localisées au niveau d'instance sur l'ensemble du modèle en replaçant les images sources par les images annotées.

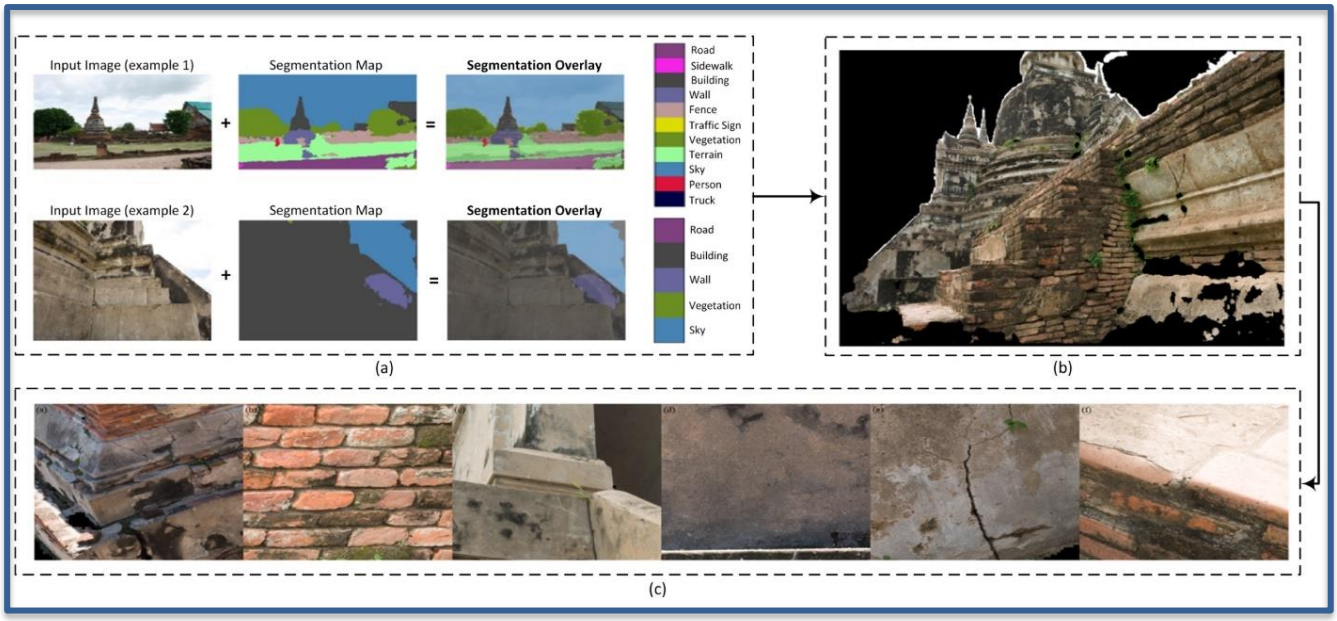


Fig. 43 – Diagramme représentant le pré-traitement des images (Pathak et al., 2021)

Les chercheurs de l'étude [27] ont réalisé la détection automatique et la photogrammétrie en parallèle. La superposition des images nécessaire pour une photogrammétrie a également été mise en valeur par le modèle, qui compare les premières prédictions ayant des pathologies communes pour affiner ses prédictions au niveau pixel. Il en résulte une localisation précise en trois dimensions de chaque élément du bâtiment et des pathologies observées.

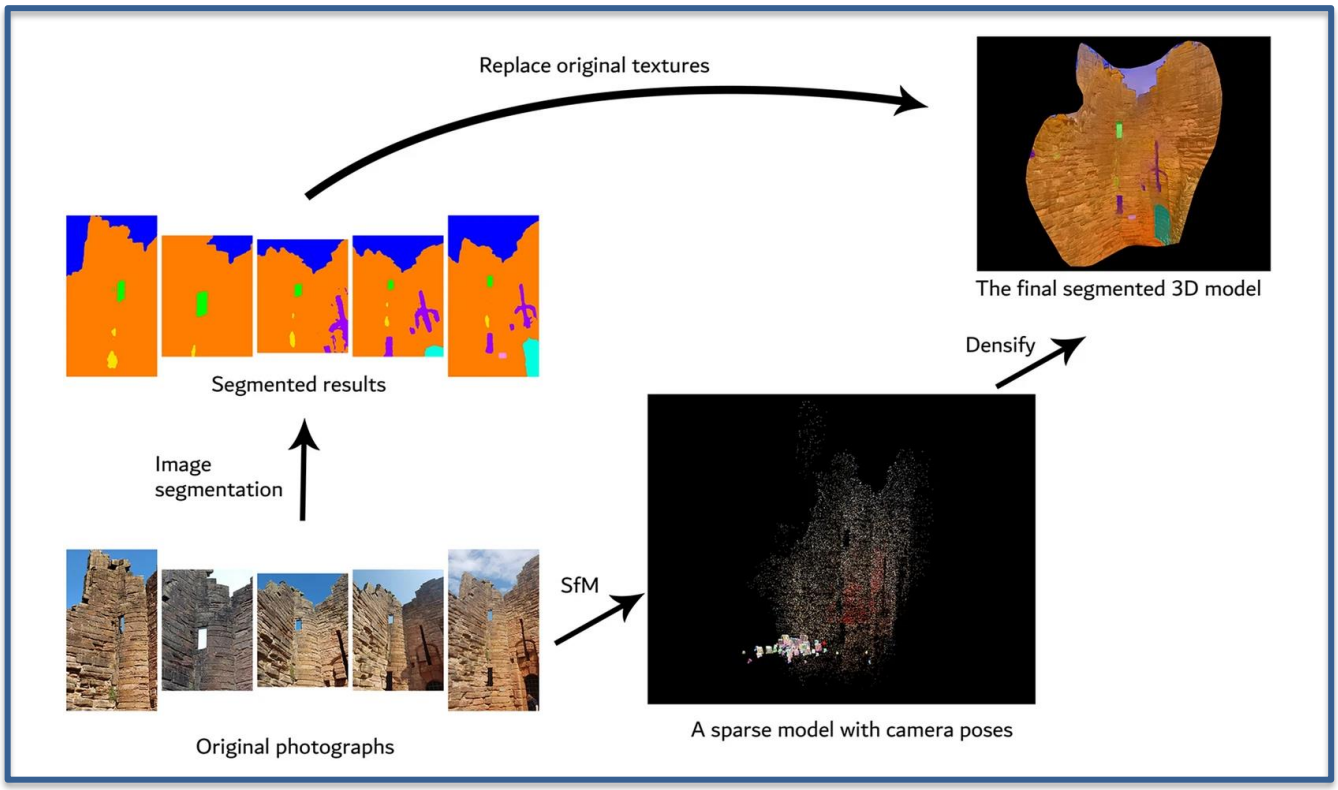


Fig. 44 – Processus de combinaison de la segmentation sémantique et de la photogrammétrie (Liu et al., 2022)

L'étude [5] ne touche pas réellement à la détection de pathologies, mais consiste en la segmentation sémantique automatisée d'éléments architecturaux directement depuis un nuage de points. Les techniques d'extraction des caractéristiques des différents éléments sont similaires aux modèles déjà présentés, si ce n'est qu'elles opèrent en trois dimensions. Si les éléments en question sont ici facilement identifiables, comme un mur ou une colonne, on peut envisager dans un futur proche une détection des pathologies directement depuis le nuage de point. Cela permettrait de représenter une caractéristique clé de pathologies comme les fissures ou l'écaillage : la profondeur. Cependant, le traitement d'un modèle 3D entier nécessite énormément de puissance de calcul et pourrait ne pas toujours être nécessaire.

2.3. Autres applications

Outre la simple détection de pathologies, certains chercheurs aspirent à automatiser d'autres aspects de la conservation du patrimoine. Par exemple, les études [7] et [19] classifient les images en fonction de la présence ou de l'absence de dégâts observables. [7] précise même le degré de dégradation observé, d'intact à sévèrement endommagé. Analogiquement, [25] dépasse la segmentation sémantique et quantifie aussi la largeur des fissures détectées et donc leur sévérité.

D'autres experts mettent l'accent sur les éléments architecturaux du bâtiment. [18] classe ces derniers, tandis que [26] est capable de les localiser sur l'image et de déterminer quel style architectural est représenté. [8], [15] et [16] mettent la priorité sur la localisation les éléments de façade manquants et [17] va plus loin en cherchant à quantifier ces éléments pour ensuite faciliter les interventions de remplacement de ces derniers.

Les études présentées dans le tableau 5 s'orientent quant à elles vers une détection en temps réel, sous forme de vidéo d'un smartphone, caméra de surveillance autour d'un site protégé ou même en réalité augmentée avec un drone survolant le monument.

3. Limitations générales

Malgré les performances individuelles des modèles, la communauté scientifique exprime de nombreuses limitations au Computer Vision, et pas uniquement dans le domaine de la conservation du patrimoine. Ces limitations concernent les données nécessaires à l'entrainement des modèles, les appareils chargés de capturer ces données ou, dans le cas du patrimoine, les caractéristiques des pathologies à détecter et leur contexte. Les études présentées plus tôt énoncent certaines limitations, tandis que le travail de Guo et al. (2024) va permettre de comprendre ces limitations ainsi que les mesures mises en place pour les surmonter. Ce dernier reprend un ensemble de données et d'observations de la communauté scientifique de 2017 à 2023 et de nombreuses études, ne concernant pas le patrimoine, ne seront pas abordées dans ce mémoire. C'est pourquoi je recommande la lecture de leur travail pour toute information plus précise sur les observations qui vont suivre.

3.1. Contexte

Le premier problème relatif au contexte d'un bâtiment à analyser est le « bruit de fond » généré par des objets non-pertinents à proximité de l'objet (arbres, voitures…) ou des éléments de surfaces qui ressemblent aux pathologies détectées (écriture, tags…). Ces éléments n'ont pas de lien avec les défauts observés mais gênent le processus de détection du modèle. Plusieurs chercheurs ont cependant pris en compte ces problèmes à l'aide de modèles plus avancés et capable de différencier ce bruit de fond des réels défauts ou en entrainant directement les modèles pour détecter les classes de bruits de fond comme l'écriture.

Ensuite, les conditions météo comme l'ensoleillement jouent un rôle majeur dans la visibilité des défauts et la qualité des images prises. Comme expliqué dans le chapitre précédent, plusieurs études excluent les images de mauvaise qualité afin d'obtenir de meilleurs résultats aux dépens des performances en conditions réelles. L'augmentation du dataset est une technique très populaire pour faire face à ce défi. Des copies d'images de bonne qualité sont floutées ou voient leur luminosité altérée pour élargir le dataset et représenter au mieux les diversités du terrain. Cependant, le résultat n'est pas toujours fidèle aux données réelles et certaines conditions ne peuvent pas être prises en compte par cette technique. Dans le cas d'images prises de nuit, il a été proposé de les convertir en « images de jour » pour faciliter la détection. Il est cependant nécessaire d'explorer plus en profondeur cette méthode afin de vérifier son efficacité.

Une pathologie incomplète, à cause de la géométrie du bâtiment ou de sa taille, empêche le modèle d'extraire l'ensemble des caractéristiques qui lui permettent d'établir une prédiction correcte. La fusion de plusieurs images en vue panoramique permet une vue d'ensemble d'un plus grand défaut, mais est également source de distorsion et donc de mauvaise performance. L'utilisation des technologies actuelles pour la création d'un modèle 3D reprenant l'ensemble de la géométrie du bâtiment résout effectivement ces problèmes, mais demande beaucoup de ressources. De plus, cela n'est pas nécessairement possible ou pertinent, selon la situation.

Enfin, les styles architecturaux, méthodes de construction ou les matériaux diffèrent souvent selon les régions, ce qui influence directement la capacité de détection des pathologies étudiées si les images d'entrainement n'ont pas pris en compte cette diversité. Un modèle très performant en Europe est susceptible d'être peu efficace ailleurs, ce qui déforce l'un des principaux arguments de cette

technologie, le fait que ses résultats restent constants. La meilleure solution reste l'élargissement du dataset afin d'englober un maximum de données, impliquant alors un temps d'entrainement plus long. Le modèle étant alors plus complexe, sa vitesse d'exécution en est impactée (Guo et al., 2024).

3.2. Équipement

En Computer Vision, il est évident que le choix de l'appareil responsable de la prise des images est un moment clé, et qu'un appareil photo de qualité assure souvent des images pertinentes. D'autres équipements sont pourtant nécessaires dans une majorité des cas, principalement pour atteindre des endroits difficiles d'accès. Dans le cas du patrimoine, on utilise surtout des drones aériens ou capables de grimper sur les surfaces verticales. L'angle duquel est pris la photo influence fortement la capacité d'évaluer les formes et dimensions des éléments capturés sur l'image. Si le problème de distorsion créé par un angle d'incidence trop élevé existe aussi pour un appareil manuel, on considère que le mouvement du drone peut également être responsable de distorsion. Il est possible de limiter ce problème en reconstruisant la scène à l'aide de photogrammétrie ou en calculant l'angle d'incidence entre l'objet pris en photo et l'axe optique de l'appareil.

Dans les études [1], [2], [11], [13], [16] et [17], on considère aussi l'excès de lumière ou d'ombre et une faible résolution de l'image comme limitations de l'équipement. Le premier amène souvent à la perte de détails et est remédié par augmentation du dataset lors de l'entrainement. La résolution est d'autant plus complexe que les défauts observés varient d'une fissure de quelques centimètres de long à de grandes surfaces de végétation ou d'efflorescence. Comme énoncé au point précédent, il est important de ne pas couvrir une surface trop restreinte du monument analysé sous peine de nuire à la détection des plus grandes pathologies. Malheureusement, le contexte du site étudié ne permet pas toujours d'obtenir la proximité ou l'éloignement idéals pour la prise d'images. Concernant les drones, un algorithme a récemment été introduit pour les contrôler de manière autonome et les garder à une distance suffisante du bâtiment tout en étant capable de détecter des fissures de petite taille (Guo et al., 2024).

Stancel & Hulic (2019) ont déterminé l'importance de la résolution des images en comparant avec YOLO la quantité d'éléments que le modèle était capable de détecter pour une même image en variant la résolution de celle-ci. Le modèle ne détecte donc que 8 objets pour une résolution minimale de 378x284, contre 14 avec des images de 4032x3024. Les étapes intermédiaires confirment que la quantité des éléments détectés est directement proportionnelle à la résolution de l'image utilisée.

3.3. Rareté des données labellisées

Les limitations relatives aux données se séparent en deux catégories, la rareté de données brutes et la difficulté de labélisation de ces données brutes.

Une raison majeure du manque de données est la quantité de pathologies et leurs différentes formes, tailles, textures et apparences. Obtenir un dataset prenant toutes ces variations est compliqué et onéreux. Il faut avoir accès aux sites ayant des pathologies recherchées observables, qui sont souvent protégés, mais aussi le matériel et la main d'œuvre nécessaires. Comme exposé précédemment, il est aussi pertinent de capturer les données sous des conditions variées, ce qui pousse les opérateurs à réaliser plusieurs fois ce processus. Hsu et al (2023) ont envisagé l'utilisation du « *Internet of Things* », qui reprend l'ensemble des appareils physiques capable de partager des données sur internet, pour élargir les datasets et obtenir un maximum d'images utilisables.

L'augmentation du dataset peut évidemment aborder les limitations de luminosité ou d'environnement, mais est aussi susceptible d'élargir la quantité de types de défauts en effectuant des opérations de rotation, inversion, recadrage qui simulent différents défauts observés et non les conditions de prise d'images [15]. Les images risquent cependant d'être trop similaires aux originales ou risquent de nuire à la détection de certains défauts ayants des caractéristiques très spécifiques. La couleur de la végétation peut être affectée par des modifications de saturation ou luminosité, alors que des pathologies ayant des orientations spécifiques peuvent être mal représentées par des rotations d'images.

Le Transfer Learning, comme exploré dans l'étude [20], permet de pré-entrainer un modèle sur base de larges datasets publics au lieu de l'entrainer directement avec le dataset de pathologies. Ce dernier est ensuite utilisé pour affiner le modèle selon les classes de détections souhaitées. Cela contribue à réduire le temps d'entrainement et à réduire les problèmes de performances liées à un dataset trop petit.

La rareté de données est encore plus problématique dans le cas de pathologies rares, pour lesquelles il est difficile d'acquérir des exemples en quantités suffisantes. Ce phénomène crée un déséquilibre entre les classes de prédiction du modèle, puisqu'elles fonctionnent justement mieux avec un nombre égal d'exemples pour chaque classe. De plus, ces pathologies rares sont souvent celles qui sont les plus sévères. Plusieurs solutions ont été proposées par les experts, mais elles abordent principalement la classification d'image qui, par sa simplicité, est la catégorie de Computer Vision ayant le moins d'applications pour une détection automatique des pathologies.

En réalité, le nombre de datasets en open-source (disponible à tous) grandit au fil des années, ce qui a le potentiel de résoudre ce problème de rareté de données. Le processus de labellisation reste néanmoins très important, mais la qualité prend désormais le dessus sur la quantité. La première limitation est le facteur humain, car les experts ont des perspectives et formations différentes, ce qui peut rendre la labellisation inconsistante. C'est d'ailleurs un problème énoncé dans les méthodes actuelles de détection de pathologies. De plus, la complexité des pathologies et les différents protocoles rendent le procédé très exigeant et sujet aux erreurs. En effet, la labellisation manuelle est chronophage, onéreuse et très demandeuse en ressources. Deux approches, l'amélioration de la qualité des labels et l'utilisation d'images brutes, tentent donc de résoudre ce défi (Guo et al., 2024).

La question de la qualité des labels a été très peu traitée par les chercheurs, assumant que les datasets fournis ne présentaient pas d'erreurs qui risquaient d'impacter les performances du modèle. Outre les erreurs, il a été observé que les fissures étaient souvent labellisées plus larges qu'elles ne l'étaient vraiment, punissant donc parfois le modèle détectant précisément la fissure pendant son entrainement. Kamel et al. (2023) ont étudié la possibilité d'utiliser du crowdsourcing afin d'assister dans la labellisation d'images en rendant l'annotation des images attractive. Cette méthode a pour but d'accélérer le processus mais de nombreuses études semblent favoriser la mise en place de directives ayant pour but de guider les experts dans la manière de labelliser les pathologies. Cela permettrait d'améliorer la qualité des données mais surtout d'avoir des résultats consistants. D'autres envisagent l'utilisation d'équipes d'annotateurs, avec un expert chargé de la formation et du suivi des autres opérateurs. La sélection d'images plus pertinentes avant d'être labellisées pourrait aussi permettre de maximiser la rentabilité du travail humain et des données qui en sont tirées. De manière générale, les études tendent vers une meilleure qualité des labels, avec des directives et des images sélectionnées.

Encore peu explorée, l'utilisation d'images brutes envisage un entrainement semi-supervisé des modèles, à l'inverse de l'entrainement entièrement supervisé mentionné précédemment. Des algorithmes permettraient d'extraire des caractéristiques d'images labellisées et d'images brutes pour limiter la nécessité de labelliser l'entièreté d'un dataset (Guo et al., 2024).

3.4. Apparence des pathologies

L'ensemble des pathologies susceptibles d'être détectées par un modèle ont des tailles variables, des limites parfois difficiles à définir ou sont complexes à définir correctement.

Le déséquilibre des classes au niveau des pixels se réfère à une distribution inégale de la quantité de pixels pour chaque classe dans une image. Ce déséquilibre apparait lorsque des pathologies comme les fissures sont rares ou peu représentées par rapport à d'autres classes. C'est un problème largement reconnu et très peu abordé par les modèles standards, qui sont incapables de traiter la situation. Certaines études explorent l'optimisation de certains paramètres pour ces classes lors de l'entrainement du modèle.

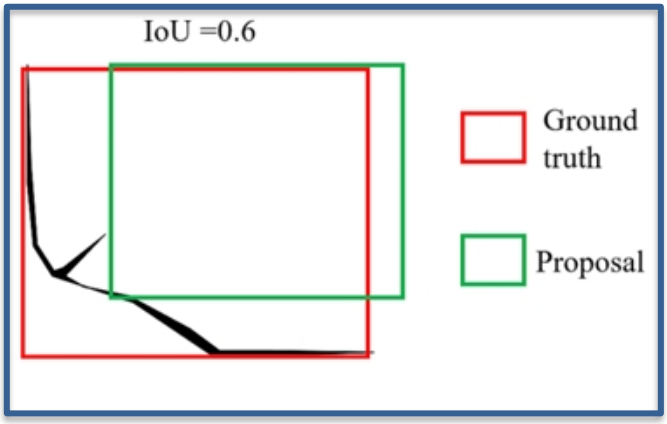


Fig. 45 – Problème causé par l'apparence des défauts et la formule IoU (Guo et al., 2024)

Lors de détection d'objets, on remarque parfois que le cadrage effectué par le modèle a une IoU avec le cadrage réel suffisamment élevée pour être jugée correcte, alors que le défaut détecté n'est pas dans le cadrage. La fig. 45 illustre très bien ce phénomène qui apparait lorsque les pathologies ont des formes singulières, principalement dans le cas de fissures. Mask IoU, une nouvelle métrique favorisant les prédictions présentant la plus haute proportion de pixels de pathologies prend en compte ce problème.

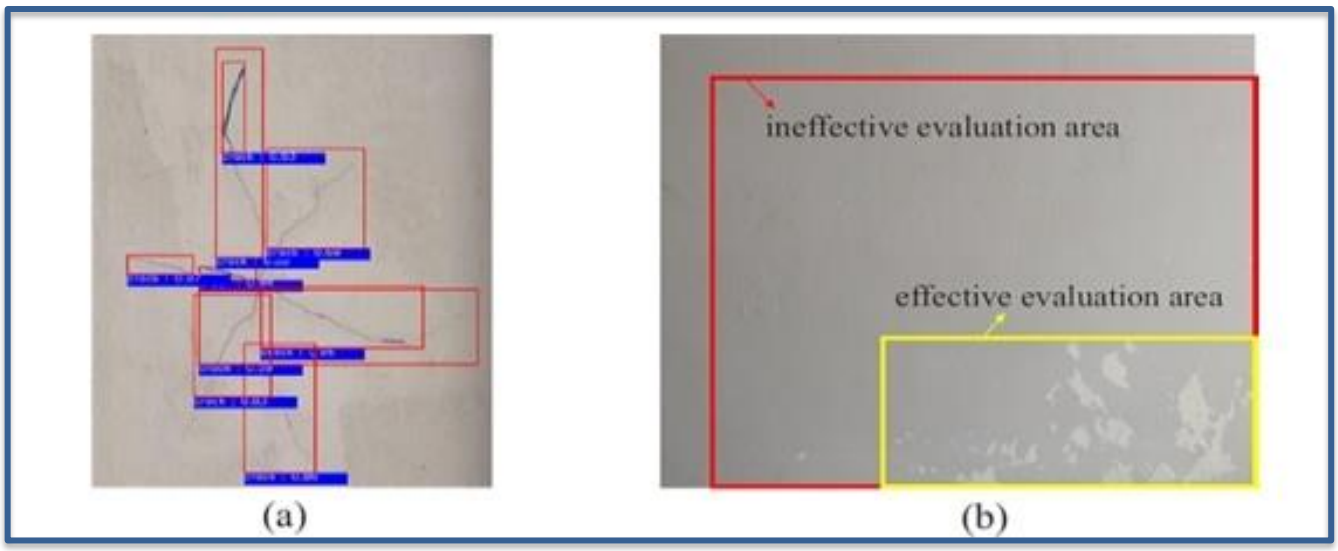


Fig. 46 – Problèmes causés par les typologies des défauts :
a) superposition des prédictions, b) défaut concentré dans une petite partie de la prédiction (Guo et al., 2024)

Certains défauts sont difficiles à identifier comme une seule instance ou comme plusieurs petites instances. Considérer plusieurs instances revient à superposer des cadrages de prédictions les uns sur les autres (fig. 46a), alors que considérer une seule instance revient souvent à avoir une majorité de pixels dans le cadrage qui n'appartiennent pas au défaut (fig. 46b). Dans une même thématique, les défauts comme l'efflorescence ou la végétation n'ont pas de contours clairs et sont donc difficiles à clairement identifier. Lors de la labellisation d'un dataset, il est important de rester consistant entre les différentes instances afin de ne pas mal orienter l'entrainement du modèle entre deux types de détection. C'est pourquoi la mise en place de directives pourrait indirectement aborder ce problème (Guo et al., 2024).

Ces limitations sont cependant propres à la tâche de détection d'objets, c'est pourquoi de nombreuses études de segmentation sémantique ou d'instances s'intéressent à détecter précisément ces fissures. Les études [22], [23], [24] et [25] abordent toutes ce problème avec des résultats encourageants. [23] se distingue des autres en qualifiant la gravité d'une fissure à l'aide de la densité de pixels classifiés comme tel par rapport aux autres pixels présents sur l'image. Les autres approches cherchent à mesurer précisément la taille de ces fissures. [22] mesure la surface de la pathologie tandis que [24] mesure le point le plus large de cette dernière et est capable de quantifier les défauts à l'aide de la segmentation d'instances. Contrairement à l'étude citée précédemment, [25] mesure la largeur de la fissure sur son entièreté et le représente graphiquement par un code couleur. Cela permet ainsi à l'expert de rapidement contrôler les fissures les plus graves.

4. Comparaison des modèles

Une première comparaison des différentes études et des modèles employés peut se faire au travers des tâches qui leur sont demandées. La classification d'image étant la plus basique, les CNNs sont souvent les plus utilisés, mais toutes les itérations de R-CNN en sont capables, de même que les autres modèles souvent utilisés en détection d'objets comme YOLOv4 [6]. L'approche par pixel des segmentations sémantiques et d'instances demande des architectures plus spécifiques, et les modèles sont souvent spécialisés dans ces tâches.

Ce chapitre a donc pour but d'établir des comparaisons d'ordres de grandeur entre les différents modèles, ce qui servira ensuite à orienter leur utilisation pour des situations spécifiques. Il est évident que ces comparaisons n'ont de sens que pour des tâches similaires car les demandes computationnelles des tâches de segmentation sont par exemple beaucoup plus élevées que celles de classification d'images.

Certaines études mentionnées dans les tableaux comparatifs ainsi que d'autres travaux cherchant à comparer les modèles serviront à l'établissement de cet ordre de grandeur. Le tableau suivant représente les tâches principalement exécutées par les différents modèles dans les études énoncées au point 1.

Une grande croix (**X**) indique la tâche principalement effectuée par le modèle tandis qu'une petite croix (x) représente celles qui peuvent également être réalisées.

Arch. du modèle	*Classification d'images*	*Détection d'objets*	*Segmentation sémantique*	*Segmentation d'instances*	*Détection en temps réel*
CNN	**X**		x		
R-CNN	x	**X**			
Fast R-CNN	x	**X**			
Faster R-CNN	x	**X**			x
Mask R-CNN	x	x	x	**X**	
SSD	x	**X**		x	**X**
YOLO	x	**X**		x	**X**
FCN			**X**		
DeepLab (FCN)			**X**		
SOLO			x	**X**	

Fig. 47 – Tableau des tâches de DL réalisées par les différentes architectures de modèles

Faster R-CNN et l'introduction du RPN le rendent théoriquement capable de détection en temps réel. Son architecture en deux temps l'empêche pourtant d'être aussi rapide que les modèles en un temps, c'est pourquoi ces derniers sont majoritairement utilisés pour ces applications. L'étude [3] a néanmoins utilisé Faster R-CNN en temps réel. SSD et YOLO sont indirectement capables de segmentation d'instances s'ils sont employés en parallèle avec des modèles capables de segmentation sémantique mais ces approches ne sont pas couvertes dans ce travail. SOLO étant une architecture assez récente, il n'existe pas à ma connaissance d'articles l'ayant spécifiquement appliquée dans la détection de pathologies du patrimoine.

Outre les tâches des modèles, des critères comme le temps d'entrainement, la vitesse d'inférence (ou de traitement), l'exactitude des prédictions ou la puissance de calcul nécessaire sont importants à comparer. Ces critères de comparaison indiquent directement la complexité et l'adaptabilité d'un modèle ainsi que les applications potentielles de ce dernier. Cependant, ils sont difficiles à établir entre chaque modèle

car il n'existe pas réellement de comparaison objective de l'ensemble des modèles. De plus, certains chercheurs n'expriment pas toujours les critères comme le temps d'inférence du modèle. Cela est compréhensible car les quelques études évaluant les performances de plusieurs modèles sont rarement comparables aux résultats d'autres études, puisque la taille des images, leur nombre ou simplement la puissance de l'ordinateur exécutant le modèle peuvent faire varier les résultats. Il est donc très compliqué de comparer les chiffres de ces études car même le mAP peut être indirectement influencé par le choix sélectif des images d'entrainement du modèle.

4.1. Evaluation des performances

4.1.1. R-CNN, Fast R-CNN et Faster R-CNN

Afin de réellement comprendre les différences entre R-CNN, Fast R-CNN et Faster R-CNN, il est nécessaire de comparer leur vitesse d'entrainement et de traitement d'image. En théorie, Faster R-CNN dépasse largement les deux modèles précédents, mais à quel point ? Une comparaison par Girshick et al. (2014) des performances de chaque modèle sur les datasets publics Pascal VOC 2007 et 2012 permet de répondre à cette question. Ces datasets public ont servi pour l'évaluation de nombreux modèles sous forme de concours mais ont également servi pour des modèles créés postérieurement. Ils ne comprennent néanmoins pas d'images de pathologies, plus complexes mais sont suffisants pour déterminer si Faster R-CNN est réellement plus performant et si les anciens modèles sont simplement obsolètes.

En termes d'exactitude, R-CNN atteint un mAP de 58,5% et 53,3% pour Pascal VOC 2007 et 2012, respectivement (Girshick et al, 2014). Les autres modèles ont des résultats légèrement supérieurs mais similaires avec 66,9% et 65,7% pour Fast R-CNN (Girshick, 2015) et 69,9% et 67% pour Faster R-CNN (Ren et al, 2016). Malgré une légère différence, on remarque que les performances s'améliorent nettement pour chaque évolution du modèle. Cependant, la principale limitation de R-CNN n'est pas sa précision mais bien sa rapidité, tant pour l'entrainement du modèle que pour le traitement d'une image. La création de 2000 propositions de régions pour chaque image est un processus extrêmement long qui diminue drastiquement la vitesse d'exécution du modèle.

Girshick (2015) a donc comparé plusieurs tailles (quantité de paramètres) de modèles R-CNN et Fast R-CNN, et comptabilise pour le modèle le plus large 9,5 heures d'entrainement pour Fast R-CNN contre 85 heures chez R-CNN. Cela correspond à une accélération du temps d'entrainement d'un facteur de 8,8. Cette accélération va jusqu'à être d'un facteur de 18,3 pour les petits modèles, avec 1,2 heure

d'entrainement contre 22. Pour le temps d'inférence, le modèle R-CNN le plus large obtient un temps de 47 secondes par image tandis que Fast R-CNN se démarque en étant 213 fois plus rapide avec 0,22 seconde par image. Les modèles ayant plus de paramètres sont plus complexes et obtiennent des résultats plus précis en sacrifiant légèrement leur rapidité d'exécution. Le modèle Fast R-CNN permet également de fortement diminuer la quantité de données dans le disque dur car il ne conserve pas les caractéristiques extraites en cache.

Ren et al. (2016) ont quant à eux cherché à comparer le temps d'entrainement entre Fast R-CNN et Faster R-CNN. En termes de vitesse de traitement, Fast R-CNN prend jusqu'à deux secondes par image avec sa méthode de sélection sélective des RoI (Regions of Interest) tandis que l'introduction du RPN de Faster R-CNN permet d'obtenir un temps d'inférence de 0,2 seconde. Notons que ce résultat ne coïncide pas exactement avec l'étude de Girshick (2015), qui déclare que Fast R-CNN est également capable d'un temps d'inférence de 0,2 seconde par image pour le même dataset. Les résultats de l'étude de Ren et al (2016) seront cependant pris en compte car ils sont plus récents et répertoriés près de 48429 fois dans d'autres articles, selon Google scholar.

L'intégration du RPN dans Faster R-CNN améliore donc ses performances en termes de mAP, de temps d'inférence et de temps d'entrainement. Avec 2000 époques d'entrainement pour Fast R-CNN contre 300 pour Faster R-CNN, ce dernier obtient des résultats plus performants. Le tableau suivant permet de visualiser les observations de chaque modèle pour les datasets Pascal VOC 2007 et 2012. Il est important de rappeler que ces résultats ne sont pas gages de performances dans la détection de pathologies et ne sont pas réellement comparables par rapport aux études plus récentes présentées dans les tableaux comparatifs du point 1.

	R-CNN	*Fast R-CNN*	*Faster R-CNN*
Méthode de RoI	Sélection sélective	Sélection sélective	RPN
Temps d'inférence	47 secondes	2 secondes	**0,2 secondes**
Temps d'entrainement	Très élevé	Elevé	**Bas**
mAP Pascal VOC 2007	58.5%	66.9%	**69.9%**
mAP Pascal VOC 2012	53.3%	65.7%	**67%**

Fig. 48 – Tableau des performances de R-CNN, Fast R-CNN et Faster R-CNN
[adapté de Girshick et al. (2014), Girshick (2015) et Ren et al. (2016)]

4.1.2. Détection d'objets

Bharati & Pramanik (2020) ont comparé les performances des architectures Fast R-CNN, Faster R-CNN, YOLO, SSD, FCN et Mask R-CNN entrainées par des datasets publics comme Pascal VOC et MS COCO (*Common Objects in Context*). MS COCO est un dataset public rendu disponible pour la première fois en 2014, il contient à l'heure actuelle plus de 328.000 images réparties sur 80 classes d'objets du quotidien, et sert encore aujourd'hui de standard pour évaluer les performances des modèles de Deep Learning. La comparaison est difficile car il est impossible de déterminer le modèle le plus performant, étant donné qu'une meilleure précision du modèle impacte toujours négativement sa vitesse d'exécution. De plus, ces deux critères dépendent ensuite de la résolution des images, du temps d'entrainement, du dataset et de tous les différents paramètres qui influencent le fonctionnement du modèle.

Selon les résultats présentés dans l'article, les auteurs affirment que les architectures SSD et FCN sont rapides mais restent incapables d'égaler la précision et l'exactitude de Faster R-CNN. Faster R-CNN, FCN et YOLO sont néanmoins dépendant de la qualité du CNN utilisé comme base du modèle tandis que les résultats de SSD n'en sont pas affectés. SSD est capable de meilleures prédictions que les deux autres modèles pour des grands objets mais est moins efficace pour la détection de petits objets. Mask R-CNN reste cependant le plus précis tandis que YOLO est le modèle en un temps le plus rapide avec des temps d'inférence entre 21 et 155 FPS.

L'étude [22] présente des résultats très encourageant pour le modèle YOLOv5 en détection de pathologies mais explore également les performances de Faster R-CNN en comparaison. Celui-ci atteint des performances comparables à YOLOv5 mais requiert près de 4 fois plus de temps d'entrainement pour un même dataset. Le tableau ci-dessous compare les performances de YOLOv5 et Faster R-CNN de l'article [22].

	YOLOv5	*YOLOv5*	*Faster R-CNN*
Taille du dataset	4451	10291	10291
Epoques d'entrainement	100	**50**	8000
mAP	85.7%	**93.7%**	85.04%
Temps d'entrainement	4 h 42 minutes	**1 h 53 minutes**	8 h 38 minutes

Fig. 49 – Tableau des performances de YOLOv5 et Faster R-CNN [adapté de Li et al. (2019)]

Faster R-CNN semble dans ce cas d'étude avoir plus de difficultés pour la détection de fissures et d'écaillage alors qu'il est légèrement plus performant que YOLOv5 pour identifier des briques et la décoloration des surfaces. Ces observations ne sont pas explorées plus en détail dans l'article mais peuvent être expliquées par le déséquilibre des classes dans le dataset. On compte pour l'ensemble du dataset 5896 exemples de décoloration et 7792 exemples de briques alors que seulement 975 fissures sont représentées, et 124 pour l'écaillage. Cette recherche démontre donc que YOLO est capable d'un entrainement beaucoup plus rapide qui nécessite moins d'images pour atteindre des performances similaires à Faster R-CNN. Celui-ci semble néanmoins obtenir de meilleurs résultats lorsque le dataset est plus large et que chaque classe est correctement représentée.

Pathologies	*Quantité d'exemples*	*Pourcentage*	*mAP YOLOv5*	*mAP Faster R-CNN*
Ecaillage	134	0.9%	**94.2%**	87.9%
Décoloration	5896	39.81%	85.3%	**89.9%**
Briques	7792	52.63%	96.4%	**96.9%**
Fissures	975	6.66%	**98.9%**	65.76%
Total	*14797*	*100%*	***93.7%***	*85.04%*

Fig. 50 – Tableau du mAP de YOLOv5 et Faster R-CNN pour différentes pathologies [adapté de Li et al. (2019)]

4.1.3. Détection en temps réel

Les modèles les plus populaires de détection d'objets en temps réels sont SSD et YOLO. Les différentes études présentées dans ce travail ne comparent pas les deux architectures mais il existe de nombreux articles expliquant leurs spécificités et ayant des conclusions similaires. YOLO est connu pour sa vitesse d'inférence mais perd en précision en comparaison d'autres modèles. SSD est quant à lui plus équilibré et a une vitesse plus importante que Faster R-CNN tout en ayant une meilleure précision que YOLO.

« Faster R-CNNs are complex but slow, YOLO models are fast but less accurate, and SSD strikes a balance between speed and accuracy (*Keylabs*. 2022) ».

Model	Complexity	Speed	Accuracy	Efficiency
Faster R-CNN	High	Slow	High	Less efficient
YOLO	Medium	Fast	Medium	Efficient
SSD	Low	Fast	High	Efficient

Fig. 51 – Comparaison des modèles de détection d'objets en temps réel (*Keylabs*, 2022)

Comme indiqué dans la fig. 51, les modèles Faster R-CNN sont excellents en précision, mais sont plus lents et moins efficaces en raison de leur architecture complexe. Les modèles YOLO offrent des temps d'inférence plus rapides, mais peuvent sacrifier un peu de précision. SSD offre un équilibre entre vitesse et précision, ce qui en fait un choix polyvalent pour de nombreuses applications *(Keylabs.* 2022*).* Ici, l'efficacité d'un modèle représente la puissance de calcul nécessaire pour le faire fonctionner. En général, on considère que YOLO est plus rapide tandis que SSD est moins complexe et plus précis (Nanos & Aibin, 2022). YOLO est plus pertinent lorsqu'on peut ignorer de légères imprécisions alors qu'utiliser SSD est préférable pour des détections en temps réel plus précises (Algoscale, 2021).

Le premier point de comparaison des deux modèles est donc leur vitesse d'inférence. YOLO est le plus capable en atteignant avec son itération la plus récente (v8) des vitesses de traitement entre 40 et 155 FPS selon sa configuration. Comme présenté aux points précédents, Faster R-CNN met en moyenne 0,2 seconde à traiter une image, ce qui correspond à 5 FPS. SSD se positionne entre les deux avec une moyenne comprise entre 22 et 46 FPS *(Keylabs.* 2022*).*

Model	Frames Per Second (FPS)
YOLOv8	40-155
SSD	22-46
Faster R-CNN	5-7

Fig. 52 – Vitesse d'exécution des modèles de détection d'objets en temps réel (*Keylabs*, 2022)

En termes de précision, SSD est donc supposé plus performant que YOLO. Les deux modèles rencontrent néanmoins des difficultés à détecter de petits objets, principalement lorsque de grands objets sont présents sur la même image (Liu et al., 2016). YOLO accentue ce problème en ne créant que deux propositions par cellule, ce qui le rend incapable de localiser précisément et de détecter plusieurs petits objets concentrés (Redmon et al, 2016). Le tableau suivant reprend les différents points abordés pour les modèles en un temps afin de les comparer en ordre de grandeur avec Faster R-CNN.

	YOLO	*SSD*	*Faster R-CNN*
Complexité	++	+	+++
Vitesse	+++	++	+
Précision	+	++	+++
Temps d'entrainement	+	+	++
Efficacité	+++	++	+

Fig. 53 – Tableau de comparaison entre YOLO, SSD et Faster R-CNN

4.1.4. Modèles de segmentation

Il n'existe pas à ma connaissance d'articles comparant directement les différents modèles de segmentation sémantique et d'instances, mais les tâches de segmentation peuvent globalement être évaluées par rapport aux autres tâches. En effet, la segmentation permet d'obtenir une précision que même les meilleurs modèles de détection d'objets sont incapables d'atteindre grâce à l'identification des contours de ces objets au lieu d'une détection par cadrage. Ce type d'opération est moins rapide et nécessite beaucoup plus de puissance de calcul pour obtenir des résultats.

Mask R-CNN, SOLO, ou FCN sont donc souvent très complexes mais offrent plus de potentiel d'automatisation de la détection de pathologies que les modèles de détection d'objets. En effet, l'approche par pixel s'avère être complémentaire aux technologies de relevé actuelles, comme la photogrammétrie et la lasergrammétrie. Les modèles de segmentation échangent donc leur vitesse d'exécution et leur simplicité de calcul pour une précision et un potentiel d'automatisation beaucoup plus importants.

On distingue toutefois quelques spécificités entre ces modèles. Mask R-CNN s'avère être beaucoup plus polyvalent que les autres architectures car il est construit sur celle de Faster R-CNN et est donc capable de réaliser d'autres tâches. De leur côté, FCN et SOLO sont spécifiquement construits, le premier pour la segmentation sémantique et le second pour la segmentation d'instances (Long et al., 2015) (He et al., 2018).

De plus, FCN est légèrement plus simple et plus rapide que Mask R-CNN car il est prévu uniquement pour la segmentation sémantique. Sa simplicité le rend pourtant moins précis pour distinguer les contours ou détecter de petits objets à cause de la perte de la résolution des images après plusieurs couches convolutives (Long et al., 2015). DeepLab, basé sur l'architecture de FCN, améliore la précision de ce dernier en introduisant le module ASPP (*Atrous Spatial Pyramid Pooling)*, qui traite l'image avec plusieurs filtres de différentes échelles et permet donc de détecter des objets de tailles variées. DeepLab est en échange moins rapide et moins efficace que son prédécesseur, puisque le module ASPP le rend plus long à entrainer (Chen et al., 2017).

4.2. Avantages et limitations

Avec ces différentes analyses, les forces et faiblesses des différents modèles étudiés deviennent plus claires. Certains se spécialisent dans un domaine particulier, comme YOLO et sa vitesse d'inférence, tandis que des modèles comme SSD ou Faster R-CNN semblent plus polyvalents et équilibrés. Le tableau suivant reprend donc les avantages et limitations liés à l'utilisation de chacun des modèles. Les limitations générales, présentées au point 3, concernent l'ensemble ou la majorité des modèles et ne sont pas reprises dans ce tableau puisqu'elles sont liées de manière inhérente à la technologie de Computer Vision et non à un modèle spécifique.

Dans le cas de modèles comme R-CNN et Fast R-CNN, rendus obsolètes par les performances nettement supérieures de Faster R-CNN, les avantages et limitations font référence à leurs prédécesseurs.

Modèle	Avantages	Limitations
CNN	- Très efficace pour la classification d'images - Simplicité	- Nécessite de très larges datasets - Temps d'entrainement long - Simplicité
R-CNN	- Capable de localisation des objets - Capable de détection de plusieurs objets	- Extraction individuelle de 2000 RoI - Temps de traitement très long - Temps d'entrainement long
Fast R-CNN	- Précision et vitesse supérieures à R-CNN - Introduction du RoI Pooling	- Temps d'entrainement long - Précision inférieure aux modèles récents
Faster R-CNN	- Haute précision et vitesse en détection d'objets - Introduction du RPN - Entrainement plus rapide que Fast R-CNN	- Vitesse trop basse pour détection efficace en temps réel - Plus complexe et lent que YOLO et SSD
SSD	- Capable de détection en temps réel - Bon équilibre entre précision et vitesse - Moins complexe que Faster R-CNN - Plus précis que YOLO	- Mauvaise détection de petits objets
YOLO	- Idéal pour détection en temps réel - Extrêmement rapide - Entrainement rapide	- Mauvaise détection de petits objets groupés
Mask R-CNN	- Capable de segmentation - Extension de Faster R-CNN - Haute précision des contours d'objets - Polyvalent	- Très complexe et lent - Puissance de calcul importante
FCN	- Capable de segmentation sémantique - Moins complexe et plus rapide que Mask R-CNN	- Précision des contours d'objets plus faible que Mask R-CNN - Mauvaise détection de petits objets - Complexe et lent
DeepLab (FCN)	- Capable de segmentation sémantique - Résout la mauvaise détection de petits objets de FCN	- Très complexe et lent

Fig. 54 – Tableau des avantages et des limitations de l'ensemble des architectures de DL

5. Synthèse de l'analyse

Selon l'analyse des différentes études et des observations réalisées, il semble que l'hypothèse, selon laquelle la détection automatique de pathologies par DL a atteint un niveau de précision suffisant pour envisager son intégration dans la conservation du patrimoine, se vérifie. On remarque par exemple que certaines méthodes sont déjà mises en pratique de manière simplifiée, notamment sous la forme d'applications mobiles accessibles aux experts et au public. Certaines classifient les styles de monuments et leurs éléments tandis que d'autres cherchent à faire participer le citoyen dans la conservation des monuments. De plus, la diversité des modèles et des applications possibles semble assurer la pertinence de la technologie pour l'ensemble de la conservation du patrimoine.

Malgré les nombreuses limitations issues du Computer Vision et de la complexité du patrimoine bâti, de nombreuses solutions sont explorées et les performances des modèles semblent globalement prometteuses. Dans le cas de données utilisées dans la détection de pathologies, l'ensemble des opérations, techniques et algorithmes cherchant à améliorer leur qualité, servent directement à optimiser les performances des modèles et leurs capacités à être appliqués dans des situations réelles.

La performance d'un modèle est souvent mesurée lors d'un test utilisant des images différentes de celles utilisées pour l'entrainement par des métriques comme le mAP ou l'IoU. Néanmoins, sa capacité à être appliqué sur le terrain se mesure aussi en temps d'application, en temps d'entrainement, en coût, en adaptabilité à d'autres datasets et selon la pertinence des résultats pour l'opérateur. Par exemple, les études mettant en place des scores de dégradation ou liant les résultats à des modèles 3D rendent l'intégration de cette technologie intéressante pour les opérateurs responsables de la restauration du patrimoine.

En dépit du fait que de nombreuses études réduisent ou éliminent la présence de données trop complexes dans leurs datasets pour simplifier leur cas d'étude et les résultats obtenus, d'autres ont volontairement inclus des images moins adaptées lors de l'entrainement des modèles proposés pour les préparer le plus adéquatement à des situations réelles et essayer de réduire ces différentes limitations.

De plus, les différentes architectures de modèles et méthodes de Deep Learning explorées présentent leurs propres limitations mais également leurs spécificités. Cela permet donc de spécialiser certains modèles pour la réalisation de tâches et applications spécifiques. Il reste cependant à déterminer quand

ces tâches et applications permettent une utilisation pertinente de la technologie, et si la détection automatique des pathologies d'un bâtiment pose parfois plus de contraintes que de réels avantages par rapport à une inspection visuelle manuelle. Il est donc nécessaire de déployer en situation réelle des modèles pour explorer leur pertinence et celle de la technologie en général.

Tâche	*Modèle*	*Précision*	*Vitesse*	*Efficacité*	*Accessibilité*	*Potentiel d'automatisation*	*Polyvalence*
CI	CNN	/	++	++	+++	+	++
DO	R-CNN	++	+	+	++	++	++
	Fast R-CNN	++	+	++	++	++	++
	Faster R-CNN	+++	++	++	++	++	++
	SSD	++	+++	+++	+++	++	++
	YOLO	++	+++	+++	+++	++	++
SS & SI	Mask R-CNN	+++	+	+	+	+++	+++
	FCN	+++	++	++	++	+++	+
	DeepLab (FCN)	+++	+	++	+	+++	+

Fig. 55 – Tableau de comparaison de l'ensemble des architectures de DL

Le tableau ci-dessus reprend les performances des différents modèles établis précédemment. La précision d'un modèle exprime sa capacité à indiquer l'emplacement de la pathologie détectée. La vitesse détermine quant à elle la quantité d'images qu'il peut traiter en un temps déterminé. Le premier critère est largement dominé par les modèles de segmentation qui sont capables de précisément marquer les contours des objets, alors que YOLO et les modèles en un temps se démarquent par leur rapidité d'exécution.

Les CNNs capables de classification d'image ne sont pas catégorisés pour leur précision car cette tâche ne cherche pas à localiser les objets. Une efficacité élevée se traduit par des temps d'entrainement réduits et une faible complexité du modèle. L'accessibilité représente la facilité avec laquelle sont calculés les résultats. Plus simplement, une faible accessibilité indique une puissance de calcul nécessaire très importante et donc implique l'acquisition de matériel performant et couteux. Le potentiel d'automatisation concerne la facilité avec laquelle les experts utiliseront les données. Finalement, la polyvalence explique la quantité de tâches qu'un modèle est capable de réaliser.

Certains modèles se démarquent en se spécialisant dans une ou plusieurs catégories. Le large éventail de spécialités semble offrir une possibilité de détection automatique adaptée à n'importe quelle situation, que celle-ci nécessite :

- Des résultats en temps réel.
- Une excellente précision.
- L'utilisation de machines ou ordinateurs peu performants.
- Un minimum d'opérations supplémentaires de la part des opérateurs.

Ainsi, avec des résultats prometteurs, des limitations qui semblent diminuer au fur et à mesure que la recherche progresse et des modèles spécialisés, une première réponse à la question de recherche semble justifiée : l'automatisation de la détection de pathologies visuelles du patrimoine bâti semble pertinente aujourd'hui. Le point suivant va représenter la pertinence des différentes approches selon la littérature scientifique afin de mettre en avant toutes les applications possibles de la détection automatique des pathologies

6. Applications du Deep Learning

Afin de résumer les tâches de détection automatique et les approches les plus adaptées selon les situations observables dans le domaine du patrimoine, ce chapitre va s'organiser sous forme de deux schémas.

Le premier va représenter des situations de conservation du patrimoine bâti. Pour chaque situation, l'objectif de la détection automatique sera défini, accompagné de propositions de tâches de Deep Learning ainsi que de méthodes d'acquisition de données adaptées pour atteindre cet objectif. Pour illustrer ces situations, elles seront ensuite associées à une étude présentée dans les tableaux comparatifs du point 1.

Le second a quant à lui pour but d'orienter le lecteur dans le choix de modèles capables de réaliser les tâches suggérées dans le premier. Cette orientation se base sur les spécificités des tâches de DL ainsi que sur les critères énoncés au point précédent.

6.1. Tableau des applications

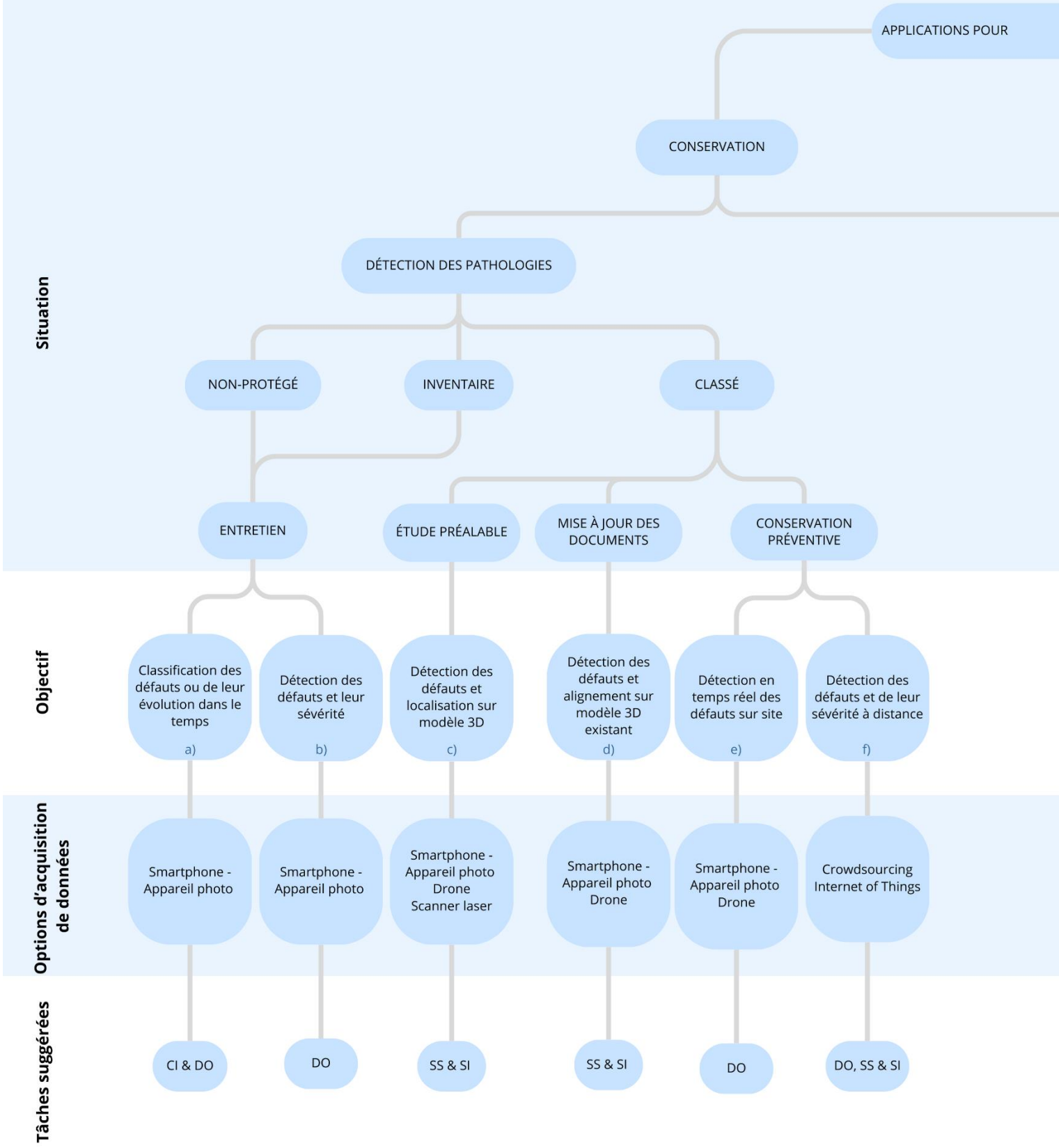
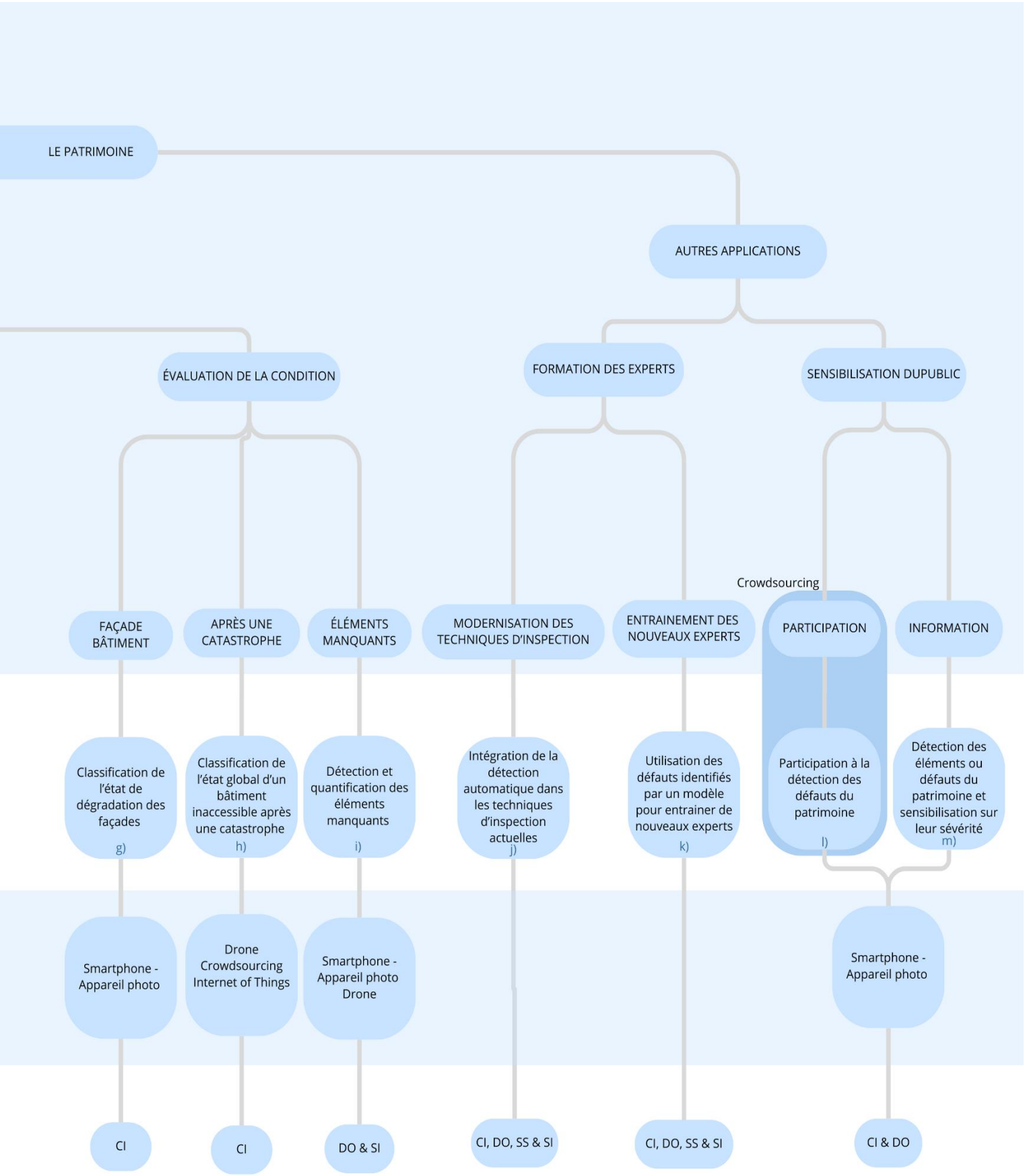


Fig. 56 – Tableau des applications en situation réelle


LE PATRIMOINE
AUTRES APPLICATIONS
ÉVALUATION DE LA CONDITION
FORMATION DES EXPERTS
SENSIBILISATION DUPUBLIC
FAÇADE BÂTIMENT
APRÈS UNE CATASTROPHE
ÉLÉMENTS MANQUANTS
MODERNISATION DES TECHNIQUES D'INSPECTION
ENTRAINEMENT DES NOUVEAUX EXPERTS
Crowdsourcing
PARTICIPATION
INFORMATION
Classification de l'état de dégradation des façades
g)
Classification de l'état global d'un bâtiment inaccessible après une catastrophe
h)
Détection et quantification des éléments manquants
i)
Intégration de la détection automatique dans les techniques d'inspection actuelles
j)
Utilisation des défauts identifiés par un modèle pour entrainer de nouveaux experts
k)
Participation à la détection des défauts du patrimoine
l)
Détection des éléments ou défauts du patrimoine et sensibilisation sur leur sévérité
m)
Smartphone - Appareil photo
Drone Crowdsourcing Internet of Things
Smartphone - Appareil photo Drone
Smartphone - Appareil photo
CI
CI
DO & SI
CI, DO, SS & SI
CI, DO, SS & SI
CI & DO

Dans ce tableau, un ou plusieurs objectifs sont liés à chaque situation. Ces objectifs sont issus des études considérées durant le travail d'analyse. Ils ont été « reliés » à une situation dans laquelle la méthode semble la plus pertinente. Il est toutefois évident que ces dernières peuvent être employées dans d'autres situations mais avec une efficacité parfois réduite en comparaison d'autres méthodes.

Par exemple, les objectifs de classification et de détection des défauts dans les points a) et b) concernent tout propriétaire de bien non classé cherchant à entretenir ce dernier. Ces propriétaires sont peu susceptibles d'avoir accès à des méthodes d'acquisition de données comme le crowdsourcing ou la formation et les moyens nécessaires pour réaliser un scan laser. Les options d'acquisition de données les plus adaptées pour ces propriétaires doivent donc répondre à un critère d'accessibilité mais surtout de pertinence par rapport à la situation. Dans ce cas-ci par exemple, on n'utiliserait jamais de drone pour photographier une simple maison, comme le point d) n'aurait pas d'intérêt à employer un scanner laser si un modèle 3D existe déjà. L'option d'acquisition de données pour a) et b) est donc tout appareil capable de capturer des images, comme un smartphone ou un appareil photo.

On distingue deux approches de détection automatique pertinentes pour entretenir un bien non classé. La classification des défauts ou de leur évolution dans le temps (a) permet au propriétaire de se renseigner sur la nature des pathologies à l'aide d'un modèle simple fonctionnant sur un appareil mobile ou un ordinateur quelconque. L'utilisation à intervalles régulier du modèle permettrait ensuite d'évaluer si les pathologies s'aggravent ou non. L'étude [1] traite notamment de différents degrés de dégâts liés à l'eau et on pourrait mettre en place un système de comparaison des images pour observer une possible aggravation des défauts. Une autre approche aurait pour but de déterminer la gravité des pathologies et donc encourager le contact d'experts pour déterminer les interventions à réaliser sur les pathologies les plus graves. [12] détermine justement lors de la détection d'écaillage et d'efflorescence des murs de briques quelles pathologies sont trop avancées et nécessitent une intervention. Ces différentes approches permettent à un propriétaire de réaliser une inspection visuelle des pathologies visuelles de son bâtiment. Cela permet d'avoir un premier verdict sur la situation du bâtiment avant une dégradation trop importante de celui-ci sans nécessairement recourir aux connaissances d'un expert. Les tâches de CI et DO suffisent à la réalisation de cet objectif.

Les bâtiments classés reçoivent une attention différente. Ils sont plus complexes, moins accessibles et sont soumis à des obligations juridiques assez fortes. Lors du classement du bien, il est obligatoire de réaliser une fiche patrimoniale de celui-ci et donc une étude préalable de son état, comme présenté dans les points c) et d). Aujourd'hui, les techniques comme la lasergrammétrie et la photogrammétrie sont largement employées lors de cette opération afin de numériser en trois dimensions le monument. Il est donc justifié d'employer des modèles de segmentation afin de bénéficier de leur complémentarité avec les technologies actuelles. Cette complémentarité est clairement mise à profit dans l'étude [5], avec la réalisation du modèle 3D en parallèle avec la détection de pathologies. Un autre exemple serait l'utilisation modèle 3D déjà réalisé pour réaliser la détection des pathologies, comme dans le cas des études [4] et [10]. La segmentation est une technique très couteuse en temps et en matériel mais les experts chargés de la conservation du patrimoine sont généralement équipés d'un matériel similaire pour l'utilisation des technologies actuelles. De plus, c'est dans la conservation du patrimoine classé que la précision de la segmentation est la plus pertinente.

Outre les études préalables, la mise en place de systèmes de conservation préventive du patrimoine permet de détecter les pathologies de l'édifice avant qu'elles n'affectent son intégrité structurelle. Le point e) représente une détection semi-automatique des pathologies avec la présence d'un expert sur site analysant les défauts détectés en temps réel par un drone. Les études [9], [13] et [14] se concentrent justement sur la détection en temps réel de pathologies comme des fissures, de l'écaillage ou l'absence d'éléments, toutes capables d'endommager le bâtiment. Au point f), une méthode presque totalement automatisée capture les données à distance, grâce à *l'Internet of Things* ou au crowdsourcing, ce qui permet de limiter l'intervention des experts [3].

Les applications précédentes concernaient la détection de pathologies mais on peut considérer l'évaluation de la condition d'un édifice comme une autre sous-catégorie de la conservation du patrimoine. Elle se divise selon le tableau des situations en 3 objectifs : la classification de l'état de dégradation des façades (g) ou de l'ensemble d'un bâtiment (h), et la quantification des éléments manquants (i).

L'article [7] explore la classification des images de façade selon deux catégories : bon état ou endommagé. Les auteurs de l'étude [8] vont plus loin en classant les différentes pathologies observables sur les images de façades considérées comme endommagées. Pour la condition globale du bâtiment, les chercheurs de [19] ont extrait par crowdsourcing des images publiées sur les réseaux sociaux lors de catastrophes naturelles, principalement des tremblements de terre. Les images de monuments étaient ensuite isolées par classification d'images pour établir la condition du bâtiment dans son ensemble. Cette approche a pour but de rapidement déterminer quels monuments requièrent une intervention urgente pour assurer leur conservation après une catastrophe.

Puisque le but est d'accéder rapidement à des images de bâtiments rendus peu accessibles, l'utilisation de *l'Internet of Things* (caméras connectées) ou de drones pourrait également être envisagée. Les éléments manquants énoncés dans le point i) peuvent aussi menacer la condition d'un bâtiment s'ils sont laissés sans contrôle. Leur quantification permet de rapidement prévoir la fabrication et le remplacement de pièces spécifiques. Par exemple, [17] a détecté et quantifié de nombreuses pièces de toiture et d'ornements disparus ou partiellement manquants dans la cité interdite à Pékin. Ceci permet d'éviter un recensement manuel et sujet aux imprécisions.

Si les technologies de Deep Learning sont étudiées pour assister directement la conservation du patrimoine, elles peuvent également avoir d'autres applications indirectes. Par exemple, si les modèles sont entrainés grâce à des images labellisées par des experts, il est envisageable de mettre à profit des modèles performants pour la formation de nouveaux experts. Une autre application indirecte est la sensibilisation du public au patrimoine et à ses pathologies. Cette sensibilisation peut se faire de manière participative, comme l'étude [27] qui utilise le crowdsourcing participatif afin de réaliser ses acquisitions d'images. En retour, les citoyens participants peuvent apprendre sur le patrimoine et les moyens mis en œuvre pour le conserver. Les travaux [18] et [26] présentent une autre forme de sensibilisation se concentrant sur l'accès à l'information concernant les styles architecturaux et les éléments qui les caractérisent. Les citoyens peuvent donc, à l'aide d'une application mobile comme MonuMAI, prendre en photo un monument et apprendre aisément sur ce qui le compose.

6.2. Guide des modèles

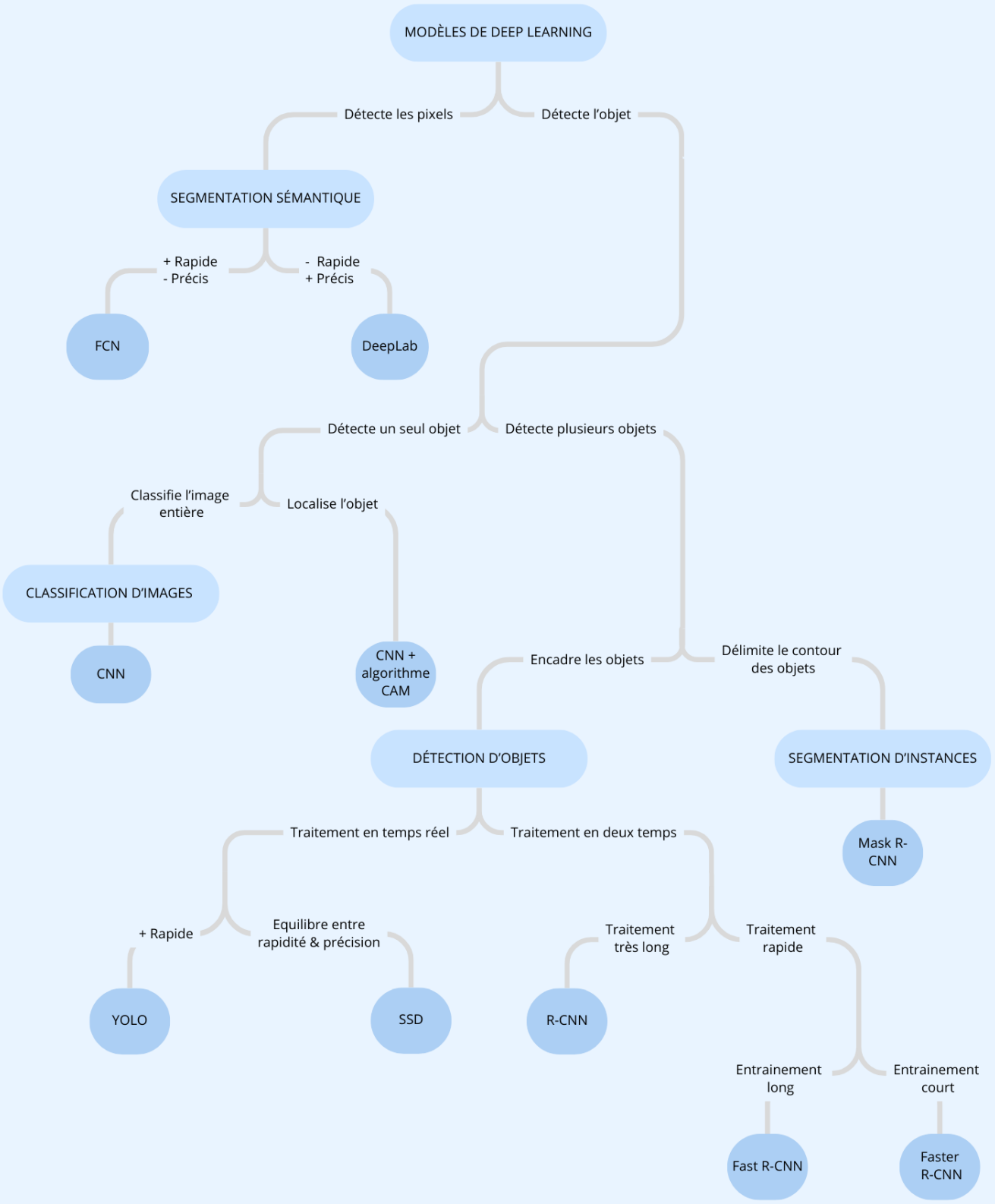


Fig. 57 – Guide pour le choix des modèles

IV. DÉPLOIEMENT EN SITUATION RÉELLE

1. Méthodologie du déploiement

Si les nombreuses études analysées au chapitre précédent semblent confirmer que la détection automatique de pathologies a aujourd'hui atteint un niveau de précision et d'exactitude suffisant pour considérer son intégration dans certains travaux d'inspection, il semble nécessaire de vérifier ces affirmations dans des conditions réelles. De plus, l'obtention de bons résultats ne justifie pas nécessairement que la technologie est pertinente, car il est important de savoir si le processus dans son ensemble reste abordable pour son utilisateur, techniquement et financièrement. Vérifier ces deux hypothèses permettra de répondre à la question de recherche : **l'automatisation par IA de la détection de pathologies visuelles du patrimoine bâti est-elle pertinente aujourd'hui ?**

Pour être le plus fidèle possible par rapport à l'état actuel du Computer Vision dans le domaine du patrimoine, il a été décidé d'utiliser un modèle et des données déjà vérifiées par la communauté scientifique. Ceci a permis de comparer les résultats avec ceux des auteurs tout au long de ce travail, et ainsi remarquer des potentielles erreurs lors de ma préparation du modèle, qui auraient faussé les observations dans le reste du travail. Dans le même ordre d'idée, il a semblé nécessaire de travailler sur un article récemment publié comme celui de Karimi et al., publié en 2024, afin d'éviter que celui-ci et le modèle associé ne soient trop rapidement considérés obsolètes.

L'étude se limite volontairement à une seule catégorie de pathologie, un seul type de matériau et un seul modèle de manière à concentrer les observations sur le fonctionnement du modèle, et obtenir des résultats suffisamment précis durant la période de recherche. Selon l'analyse de la littérature scientifique et des limitations présentées au chapitre précédent, les fissures ont été jugées les pathologies les plus pertinentes car elles sont souvent mentionnées pour leur complexité et leurs caractéristiques très variées.

Selon une réflexion analogue, le matériau choisi a été la maçonnerie de briques. La maçonnerie représente une importante partie du patrimoine bâti susceptible d'être ciblé par cette technologie, et présente des couleurs, des appareillages et des formes variant selon le contexte, ce qui les rend difficiles à représenter parfaitement dans un dataset. Cela permet d'évaluer la robustesse du modèle en situation réelle, autrement dit sa capacité à reconnaitre une pathologie dans un contexte différent de celui dans lequel il a été entrainé.

Enfin, le modèle YOLOv5 et l'article *« Automated Surface Crack Detection in Historical Constructions with Various Materials Using Deep Learning-Based YOLO Network »* de Karimi et al. (2024) ont été choisis parce qu'ils répondaient aux critères précédents et parce que le code nécessaire à l'utilisation du modèle a été fourni par les auteurs. Bien que de nombreux matériaux soient étudiés dans ce travail, les résultats et le dataset ont été présentés de manière à facilement séparer les données relatives aux fissures de maçonnerie. Dans un premier temps, la détection d'objets avec YOLO est préférable car les données et les articles sont plus abondants que pour des tâches de segmentation. De plus, la puissance de calcul nécessaire pour faire tourner le modèle devait aussi être réduite pour assurer l'obtention de résultats.

Pour vérifier au mieux les deux hypothèses citées précédemment, ce chapitre est divisé en trois parties :

- Une présentation détaillée de l'article et des méthodes employées par les auteurs.
- Un guide décrivant chaque étape de l'utilisation du modèle, de sa préparation à l'analyse des résultats.
- Une discussion sur les contraintes techniques et les pratiques rencontrées.

Les conclusions permettront finalement de vérifier si l'intégration de l'IA dans l'inspection des pathologies du patrimoine bâti est pertinente et sous quelles conditions. Ce sera aussi l'occasion d'établir les perspectives de prochaines études sur ce sujet.

La première partie vise à décrire tel que présenté dans l'article d'origine le processus des auteurs, la taille du dataset utilisé, l'origine des images, les résultats des prédictions et les différents paramètres permettant d'obtenir ces résultats.

La seconde reprend mon déploiement de YOLOv5, de l'acquisition du code de ce dernier à l'analyse des résultats. Cette partie a comme objectif d'illustrer chaque étape de l'utilisation du modèle, ainsi que les différentes variables qui peuvent influencer les prédictions, que ce soit dans les paramètres lors de l'entrainement, lors de la détection ou lors de la prise d'images. Elle est sous divisée en plusieurs catégories : la préparation du modèle, l'acquisition des images et l'analyse des résultats.

Pour terminer, la dernière partie comparera les divergences entre les différentes prédictions du modèle pour ensuite les mettre en parallèle avec l'état de l'art et les nombreuses observations qui y sont reportées. Cette comparaison mettra aussi en avant les avantages liés à l'utilisation d'un modèle de Deep Learning, avant d'introduire les contraintes rencontrées lors de l'utilisation du modèle.

2. Détection d'objets avec YOLOv5

2.1. Présentation de l'article d'origine

Karimi et al. (2024) déclarent que la majorité des études sur la détection de pathologies se concentrent sur des matériaux modernes comme le béton, l'asphalte et le métal. Cependant, ils remarquent un intérêt grandissant pour la détection de fissures sur des maçonneries du patrimoine bâti. Ceci s'avère être plus difficile que pour les matériaux modernes à cause de la complexité des matériaux comme la brique et la pierre, d'autant plus qu'il y a un manque de datasets concernant des fissures sur des matériaux de maçonnerie.

Des images de fissures sur quatre types de matériaux de maçonnerie (torchis, briques, pierres et carreaux) ont été collectées et leurs résultats comparés avec un dataset de béton, représentant des matériaux modernes. Les images de béton et de torchis ont été combinées dans un seul ensemble, tandis que les pierres et les briques ont été isolées dans des ensembles distincts. Au total, 6 datasets ont été utilisés pour entrainer le modèle YOLOv5. Bien que des versions supérieures existent, ce dernier a été choisi pour cette étude car il présentait selon les auteurs de meilleurs résultats que les modèles YOLO plus récents dans le cadre de leur travail.

Chaque dataset a été entrainé sur 100 époques avec un *batch size* de 16. Ces paramètres seront développés en détail au point 2.2. Le modèle utilise les poids *yolov5s*, qui signifient qu'il a été pré-entrainé sur le dataset MS COCO. Toutes les opérations ont été réalisées en utilisant la bibliothèque Pytorch sur Google Colab avec la mémoire GPU disponible sur la plateforme (NVIDIA Tesla T4, 16 Go). Google colab permet la création de notebooks, qui sont des environnements interactifs laissant l'utilisateur visualiser, modifier et exécuter du code. Ils permettent aussi d'observer les résultats ou d'annoter les étapes à suivre pour d'autres utilisateurs.

2.1.1. Datasets utilisés

En raison de la diversité des matériaux étudiés, les images proviennent de lieux variés. Les photos de briques et de pierre ont été prises sur des ponts historiques d'Ispahan, en Iran, tandis que les images de béton proviennent des ponts modernes de la ville. Le torchis a été observé sur des bâtiments historiques autour de la ville d'Ispahan et les carreaux proviennent d'ornements dans des villes du nord du Portugal.

Le dataset comprenait initialement 4912 images, toutes capturées avec un Samsung Galaxy A32 de 64 mégapixels dans des conditions météos diverses. Un tri éliminant les photos non-pertinentes et de mauvaise qualité fut ensuite réalisé. Afin d'augmenter la robustesse du modèle pour la distinction entre fissures et joints de mortiers, une rotation des images de 45° a été appliquée, ainsi que des ajustements de luminosité et de contraste pour améliorer la visibilité des défauts. Cette opération a permis de doubler la taille du dataset. Les images d'une résolution initiale de 3456 × 3456 pixels ont été réduites à 416 × 416 pixels afin de s'adapter au modèle. Les annotations (fig. 58) ont quant à elles été réalisées manuellement sur Roboflow (Karimi & al., 2024).

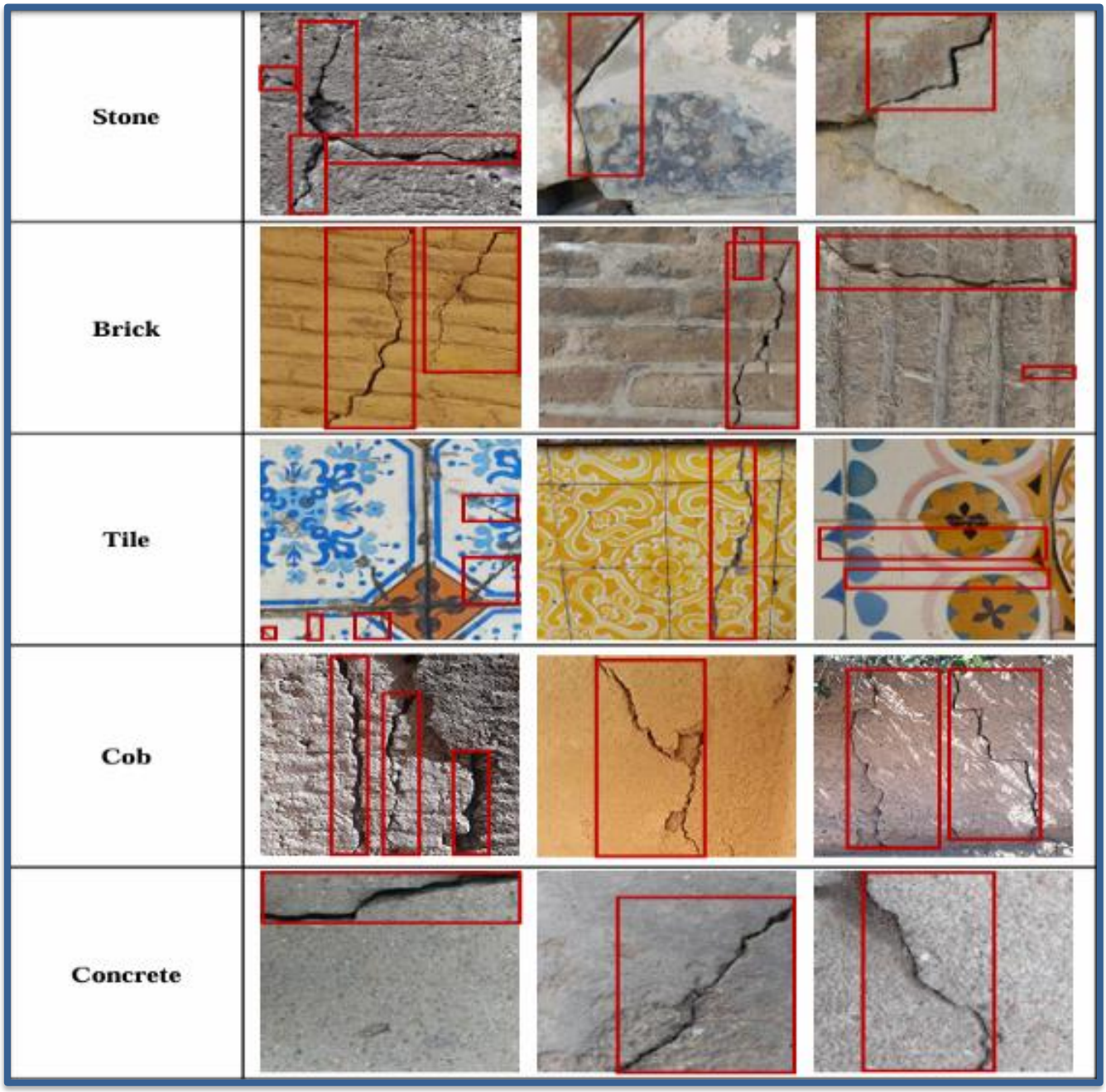


Fig. 58 – Images de fissures annotées (Karimi et al., 2024)

La fig. 59 représente la quantité d'images présente dans chaque dataset après toutes ces opérations. En réalité, l'ensemble des images est subdivisé en trois sous-ensembles : entrainement, validation et test, avec une répartition de 70:20:10. Ici, les ensembles de validation et d'entrainement sont confondus. Pour les briques, on compte donc 861 photos d'entrainement et 234 de validation (Karimi & al., 2024).

	Concrete	Concrete and cob	Cob	Stone	Stone and brick	Brick	Tile
Train	1005	1865	860	788	2693	1095	167
Test	111	206	95	94	212	118	41
Total	1116	2071	955	882	2905	1213	208

Fig. 59 – Composition des différents datasets (Karimi et al., 2024)

2.1.2. Résultats présentés dans l'article

La fig. 60 illustre que les valeurs de précision, de rappel et de mAP50 pour le béton ont surpassé celles des autres matériaux, indiquant que le modèle a mieux réussi à y détecter les fissures. Ceci pourrait être attribué à la quantité légèrement plus importante d'images d'entrainement, mais aussi à l'arrière-plan qui est bien plus uniforme qu'avec des matériaux comme la brique ou les carreaux. Pourtant, le torchis et le béton obtiennent respectivement un mAP50 de 92.2 et 94.4% contre 81.6% pour les briques, dont la quantité d'images d'entrainement est plus importante. Dans ce cas-ci, la nature de l'arrière-plan (complexe ou uniforme) joue donc un rôle déterminant, pouvant parfois l'emporter sur l'influence de la quantité d'images d'entrainement. Par ailleurs, la présence de joints pour les images de briques et de pierre contribue probablement aux mauvaises prédictions du modèle, expliquant le *recall* de 77.0 et 80.3% contre 90% pour le béton & torchis. Dans le cas des carreaux, les motifs accentuent le phénomène de complexité de l'arrière-plan, ce qui amène le modèle à les considérer comme des fissures, expliquant le *recall* de 67.4%.

Materials	Precision (%)	Recall (%)	mAP50 (%)	mAP50:95 (%)
Concrete	92.6	89.2	94.4	76.5
Concrete and cob	93.2	90	93.9	69.2
Cob	92.0	87.9	92.7	65.8
Stone	86.3	80.3	87.2	63.5
Stone and brick	80.6	77.5	83.4	59.4
Brick	82.6	77.0	81.6	55.2
Tile	71.2	67.4	70.3	47.1

Fig. 60 – Performances de YOLOv5 pour les différents datasets (Karimi et al., 2024)

On observe aussi dans les datasets hétérogènes que le mAP50 forme une moyenne par rapport aux datasets du matériau dont ils sont formés. Ceci est par exemple illustré avec le mAP50 de 83.4% du dataset pierre & brique, situé entre les mAP50 de la pierre, 87.2%, et de la brique, 81.6%.

Concernant les prédictions sur les images de briques, le modèle a obtenu des niveaux de confiance allant de 61% à 78% (fig. 61j-l) pour des images de validation (images jamais vues durant l'entrainement), ce qui démontre sa capacité à identifier les fissures. Toutefois, dans un cas spécifique, le modèle a incorrectement identifié une fissure dans le joint entre les briques et a échoué à détecter une petite fissure se ramifiant à partir du même joint (Karimi & al., 2024).

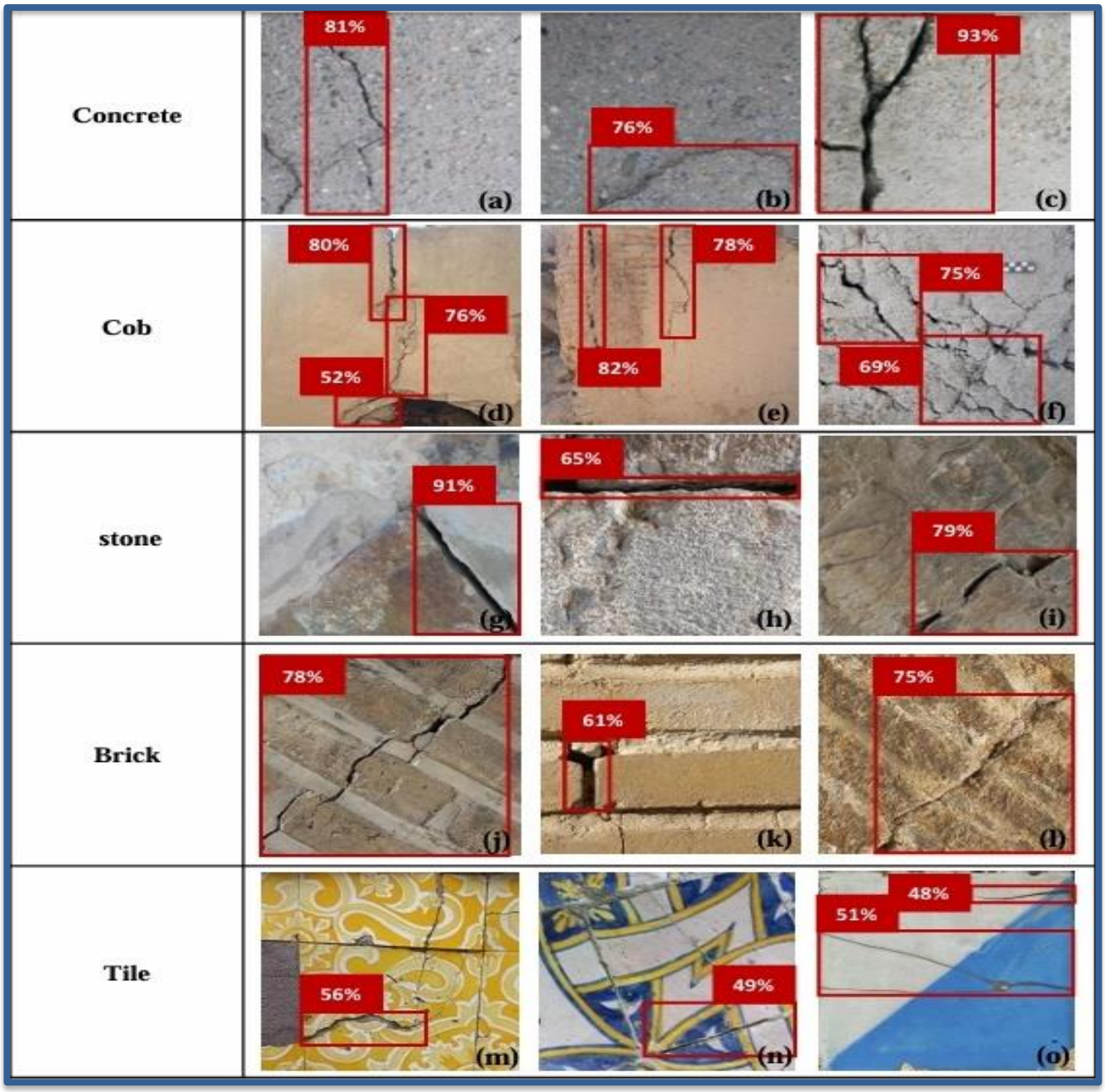


Fig. 61 – Prédictions et niveaux de confiance de YOLOv5 pour les différents datasets (Karimi et al., 2024)

Les auteurs ne mentionnent pas la tendance du modèle à détecter moins efficacement les fissures horizontales ou dont l'orientation s'en rapproche, alors que ceci semble se vérifier dans l'analyse des résultats en situation réelle. Malgré les techniques d'augmentation du dataset, une majorité des instances d'entrainement sont verticales, ce qui pourrait expliquer ce phénomène.

Bien que ce soit peu marqué, on distingue que les niveaux de confiance les plus bas pour le béton et le torchis concernent des prédictions de fissures horizontales (fig. 61b et fig. 61d). On observe aussi des faux positifs (fig. 61k et fig. 61h), le modèle ayant confondu un joint avec une fissure. Les mauvais résultats et niveaux de confiance des carreaux (fig. 61m-o) sont liés à la complexité de l'arrière-plan et à la faible quantité d'images d'entrainement.

2.2. Préparation de YOLOv5

Ce point va mettre en œuvre le modèle YOLOv5 et l'entrainer tel qu'il a été présenté dans l'article d'origine. Une fois entrainé, le cas d'étude et les différentes expériences d'utilisation du modèle en situation réelle seront développés.

2.2.1. Prérequis de l'environnement d'exécution

Avant d'utiliser un modèle, il faut d'abord obtenir son code. Il suffit souvent de télécharger un fichier via des sites d'hébergement comme GitHub, ou d'obtenir ce fichier directement par un tiers.

Après la récupération du code, il est nécessaire de choisir l'environnement d'exécution adapté. Un environnement d'exécution est un logiciel responsable de l'exécution de commandes et programmes informatiques. Ici, on distingue deux plateformes permettant l'exécution de ces environnements : locale et cloud. Un environnement local revient à utiliser la puissance de calcul (CPU, GPU et mémoire RAM) de l'ordinateur où le code est exécuté, ce qui demande une machine relativement puissante selon la complexité du modèle. Il requiert également l'installation de logiciels permettant l'utilisation du modèle sur l'ordinateur. Google colab, qui est un environnement cloud, permet quant à lui d'utiliser la puissance de calculs des serveurs GPU mis à disposition par le service, ce qui accélère le processus. Dans le cas d'un notebook sur Google colab, l'acquisition du code peut également se faire par le partage de ce dernier sur votre google drive. Puisque tout se fait en ligne, il demande peu de prérequis et est donc plus abordable pour un utilisateur non formé dans la programmation. La puissance de calcul disponible pour la version gratuite est néanmoins limitée, et il peut être nécessaire d'acheter du temps de calcul si les opérations sont trop importantes.

Pour ces différentes raisons et dans le but d'obtenir des résultats comparables au travail initial, Google colab servira d'environnement d'exécution. Ayant cependant travaillé localement et sur le cloud, une comparaison des deux environnements au point 4.1 permettra de comprendre les avantages et contraintes liés à l'utilisation de chacun.

Afin de préparer le modèle pour son entrainement, il faut :

1. **Se rendre sur le site de Google colab « https://colab.research.google.com/ »**
2. **Cliquer en haut à gauche sur Fichier/Ouvrir le notebook**
3. **Sélectionner et ouvrir le fichier .ipynb contenant le modèle recherché**

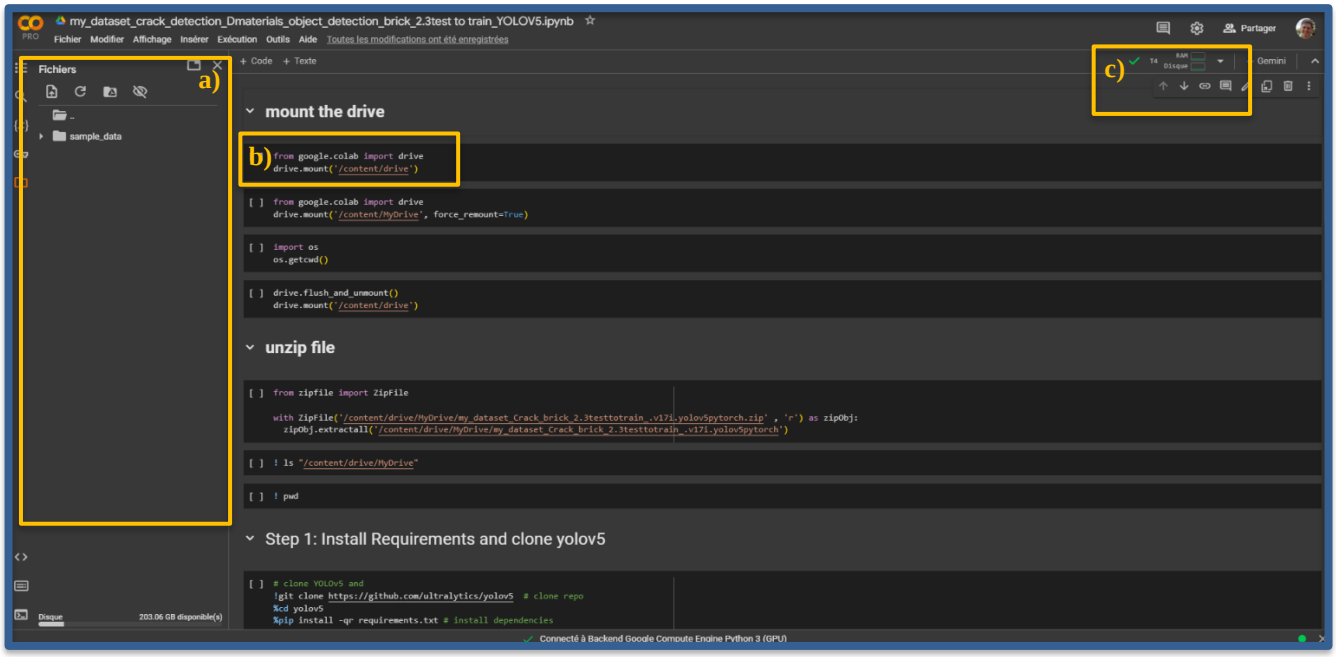


Fig. 62 – Interface de Google colab : a) barre de navigation, b) cellule de commande, c) environnement d'exécution

Une interface interactive apparait. On peut observer des cellules en gris foncé, contenant chacune des lignes de commandes à réaliser. Ces cellules sont organisées dans l'ordre dans lequel elles doivent être exécutées, mais elles sont également annotées pour permettre à l'utilisateur d' identifier facilement les actions qu'il réalise. Le GPU auquel le notebook est connecté, ainsi que sa consommation, sont indiqués en haut à droite. La barre de navigation à gauche représente les dossiers et fichiers accessibles. Cette barre permet de manipuler le modèle, et se réinitialise dès que le GPU est modifié ou après un temps défini sans l'utiliser. Cela signifie que toutes les données non exportées vers Google drive ou non téléchargées sur l'ordinateur seront perdues.

4. Lier le notebook à son Google drive

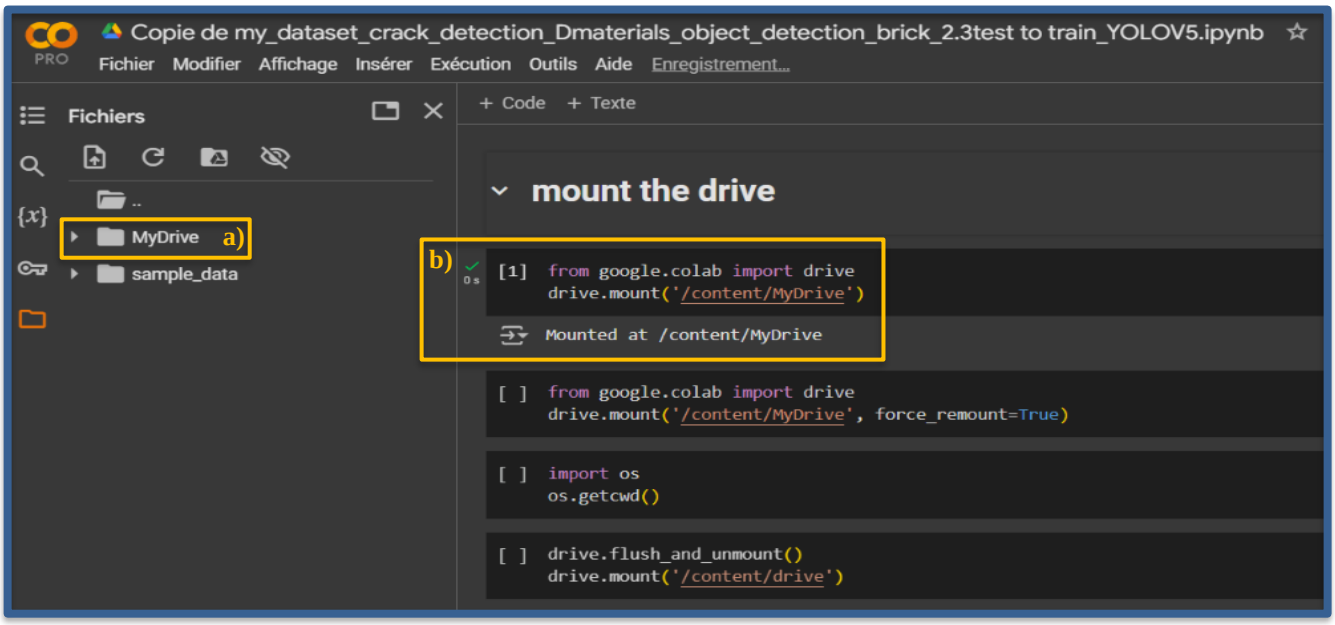


Fig. 63 – Préparation de YOLOv5 sur Google colab, connexion à Google drive

La première cellule dans « mount the drive » effectue justement cette action (fig.63b). Il faut ensuite de compléter ses identifiants et se connecter à Google drive à partir de la fenêtre qui apparait. Un ✓ à gauche de la cellule indique que toutes les commandes de cette dernière ont été réalisées. On y trouve aussi son temps d'exécution.

Les résultats de chaque cellule sont toujours indiqués en dessous de cette dernière. On peut remarquer que l'onglet « Mydrive » est apparu dans la barre de navigation (fig. 63a). Cela permet au code d'accéder à l'ensemble de vos documents sur Google drive directement depuis le notebook. Les autres cellules servent à lier le Google drive automatiquement ou à le déconnecter, mais elles ne sont pas pertinentes ici.

5. Extraire et préparer le dataset

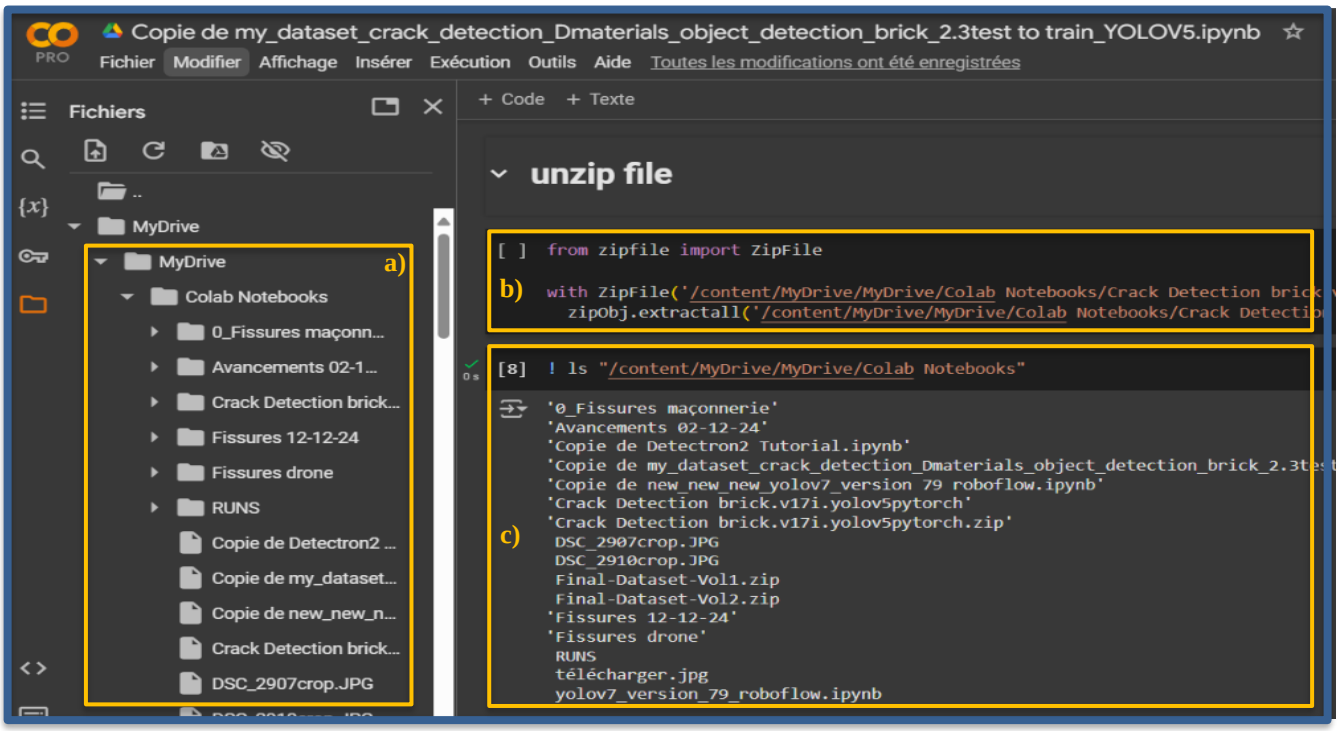


Fig. 64 – Préparation de YOLOv5 sur Google colab, extraction du dataset

Les datasets étant des dossiers très lourds, ils sont souvent téléchargés sous forme de fichiers compressés .zip. Une fois importés dans le Google drive, la première commande ci-dessus (fig. 64b) permet d'extraire les fichiers directement depuis le drive. Il faut néanmoins spécifier le chemin d'accès du dossier à décompresser. On peut le trouver dans le menu déroulant de la barre de navigation, visible dans la fig. 64a, ou grâce à la commande « ! ls ». Cette commande liste tous les fichiers et dossiers présents dans le dossier indiqué (fig. 64c).

Il est également possible d'importer directement le dataset décompressé dans le drive mais cette opération prend plus de temps et remplit rapidement l'espace de stockage si plusieurs datasets y sont conservés. Pour ce travail, le dataset a été fourni par l'auteure de l'article d'origine, mais il en existe de nombreux sur des bases de données comme Mendeley et Roboflow.

6. Installer YOLOv5, les dépendances et les poids prés entrainés

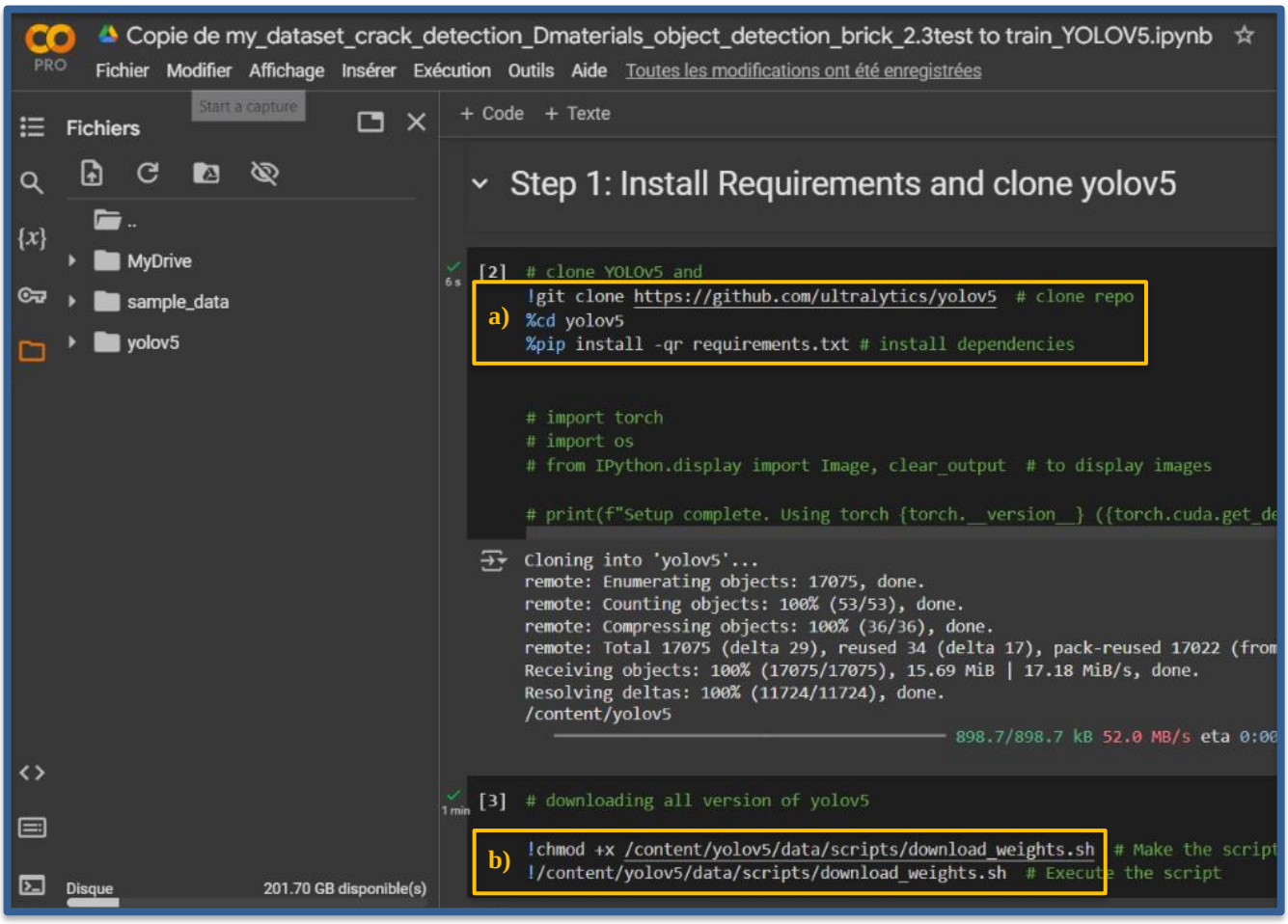


Fig. 65 – Préparation de YOLOv5 sur Google colab, installation du modèle

Dans la cellule de la fig. 65a, la première commande permet de télécharger et cloner le modèle dans l'environnement d'exécution. On peut d'ailleurs voir le dossier apparaitre dans la barre de navigation. La seconde permet de changer de répertoire, et plus précisément d'entrer dans le dossier yolov5 qui vient d'être créé. Ceci signifie que les prochaines commandes y seront exécutées et auront accès aux différents fichiers de ce même dossier. La dernière commande installe tous les packages qui permettent au modèle de fonctionner selon les versions indiquées dans le fichier *requirements.txt*. Sans ce fichier, l'utilisateur devrait manuellement télécharger les packages sans savoir si les versions sont compatibles entre elles, ce qui empêcherait alors le modèle de fonctionner. On compte des dizaines de packages interdépendants.

La cellule suivante permet quant à elle de télécharger les poids pré-entrainés de YOLOv5 (fig. 65b). Ces poids servent à accélérer l'entrainement pour des nouveaux datasets en évitant que le modèle ne commence son apprentissage de zéro. Le modèle est désormais préparé, on peut alors l'entrainer pour réaliser les prédictions sur n'importe quelle image. Il faut remarquer que ces étapes doivent être exécutées à chaque fois que l'environnement d'exécution aura été déconnecté.

2.2.2. Entrainement du modèle

2.2.2.1. Commandes et paramètres

Une fois le modèle mis en place, il suffit pour l'entrainer de :

1. **Accéder au fichier .yaml**

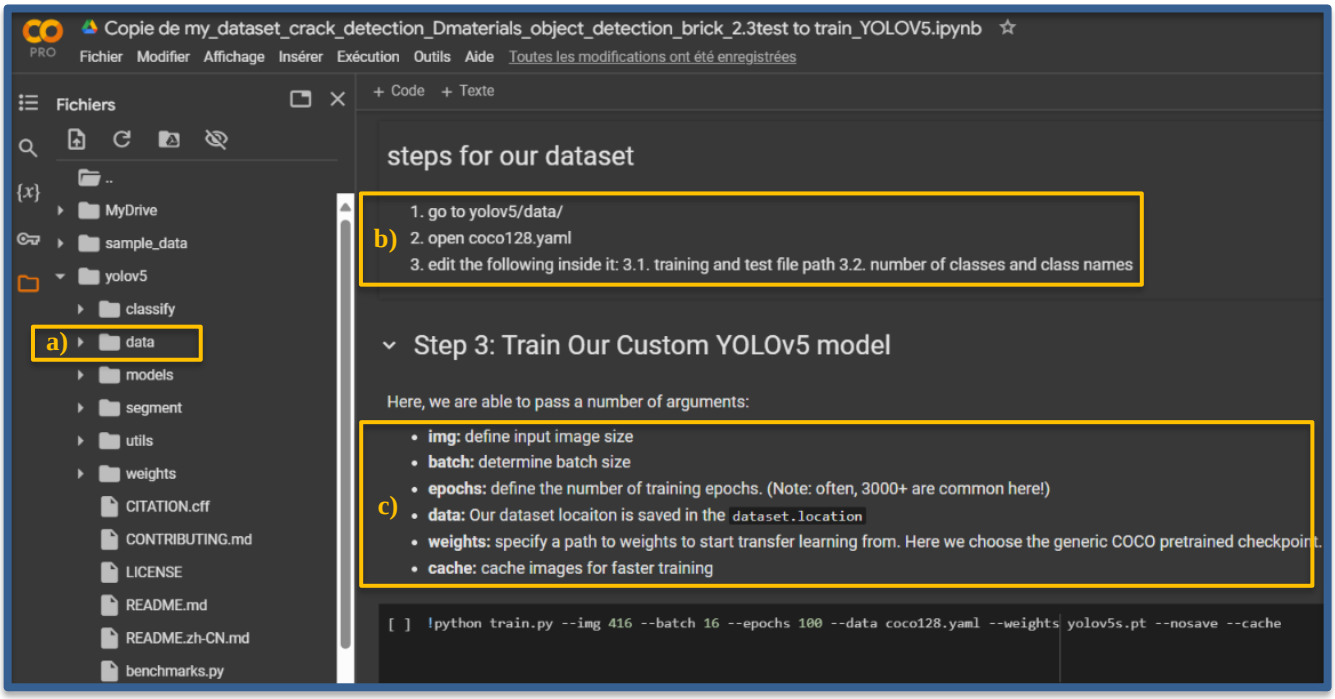


Fig. 66 – Entrainement de YOLOv5 sur Google colab, accès au fichier .yaml

L'entrainement est exécuté en deux étapes, la configuration d'un fichier .yaml et l'exécution de la commande d'entrainement. Pour YOLOv5, les fichiers .yaml sont conservés dans le dossier « yolov5/data », visible dans la barre de navigation sur la fig. 66a. On observe ici l'avantage des notebooks et des annotations entre les cellules de code, qui montrent justement comment configurer le fichier .yaml (fig. 66b) et expliquent les différents paramètres de la commande d'entrainement (fig. 66c).

La variable *img* indique à quelle résolution les images seront réduites pour passer dans le modèle. Dans le cas d'un rectangle, c'est toujours le côté le plus long qui est réduit selon cette résolution, et les proportions de l'image sont conservées. Les paramètres *batch* et *epoch* seront développés aux points suivants. *Data* fait référence au fichier .yaml modifié précédemment pour indiquer le chemin d'accès du dataset, et *weights* indique quels poids pré-entrainés vont être utilisés.

2. Mentionner les chemins d'accès et le nombres de classes dans le fichier .yaml

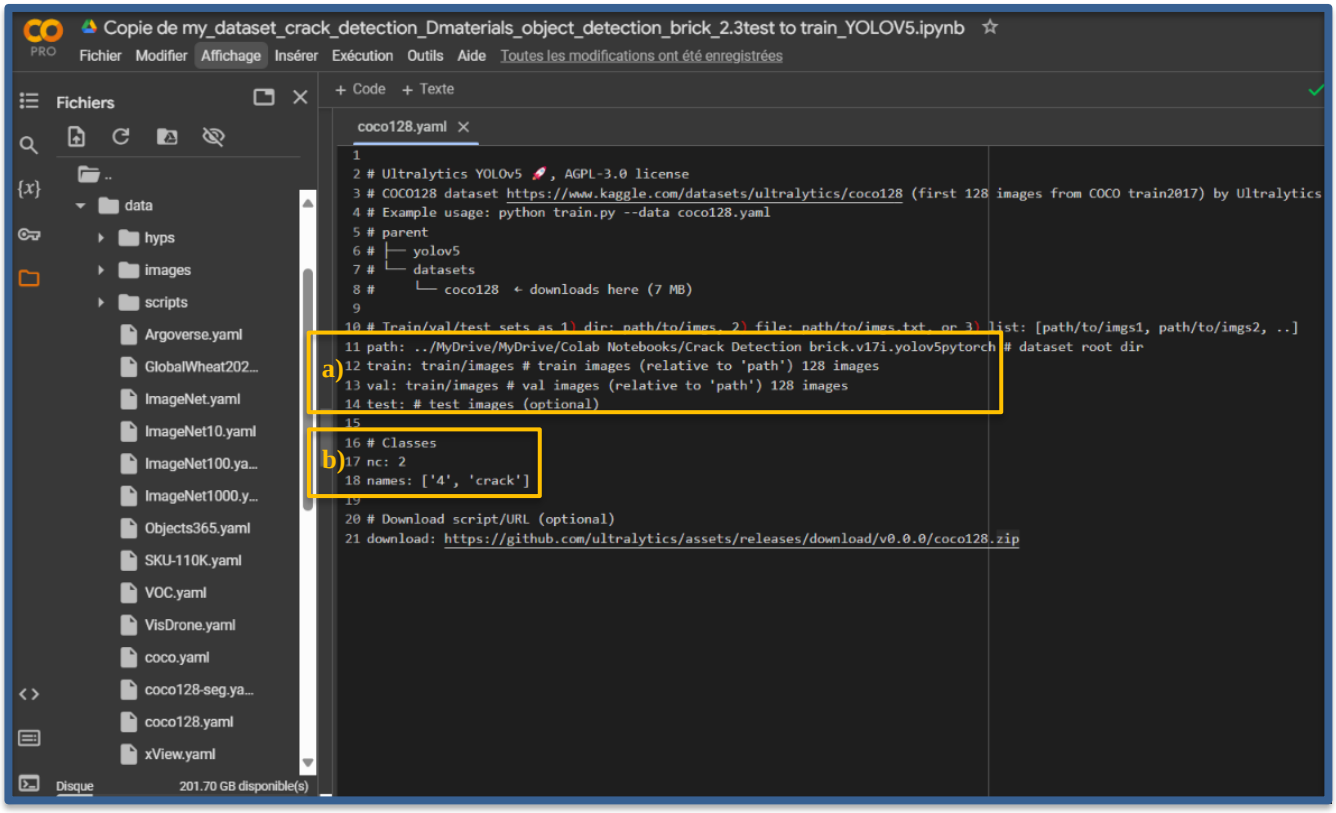


Fig. 67 – Entrainement de YOLOv5 sur Google colab, configuration du fichier .yaml

Configurer le fichier .yaml revient à y mentionner le chemin d'accès du dataset (fig. 67a), ainsi que les noms et le nombre de classes d'objets à détecter dans celui-ci (fig. 67b). Dans ce notebook, il est proposé par Karimi et al. (2024) de ne pas créer de fichier .yaml mais de simplement modifier coco128.yaml et d'y remplacer les informations pour correspondre au nouveau dataset. Pour entrainer le modèle sur n'importe quel autre dataset, il faut remodifier ce fichier.

3. Configurer puis exécuter la commande d'entrainement

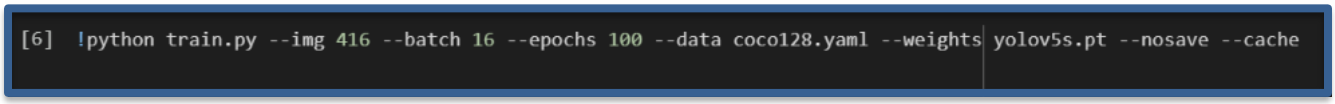


Fig. 68 – Commande d'entrainement de YOLOv5

La commande d'entrainement de la fig. 68 reprend exactement les paramètres utilisés par Karimi et al. (2024) lors de l'entrainement de YOLOv5.

2.2.2.2. *Variation du nombre d'époques*

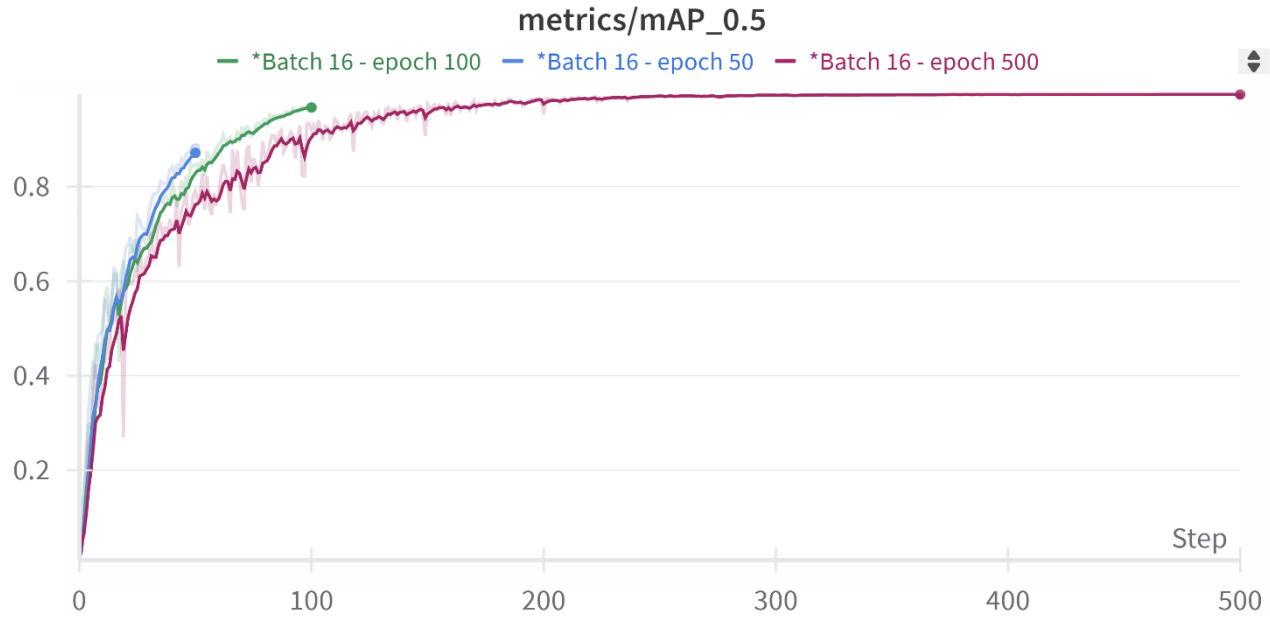


Fig. 69 – Evolution du mAP de YOLOv5 pour différentes durées d'entrainement

Une époque correspond à un cycle pendant lequel le modèle a parcouru une seule fois l'intégralité des images du dataset. Cette opération est répétée lors de l'entrainement jusqu'à obtenir un mAP stable. Entrainer le modèle sur trop peu d'époques ne lui laisse pas le temps d'améliorer ses prédictions, ce qui le rend moins performant. Dans ce cas ci-dessus (fig. 69), limiter YOLOv5 à 50 époques le restreint à un mAP de 87%, alors qu'on observe sur le graphe qu'il n'était pas encore en train de se stabiliser.

A l'inverse, trop entrainer un modèle risque de créer de l'*overfitting*, ce qui fait que le modèle apprend les données d'entrainement « par cœur » et a de moins bonnes performances lorsqu'on présente de nouvelles images. Après 500 époques, le mAP est de 99.5%, ce qui semble indiquer des performances quasiment parfaites par rapport aux annotations du dataset. Pourtant, on observera lors de l'étape de validation, qui permet de vérifier la qualité du modèle en présentant des images non-vues par celui-ci, que ce n'est pas nécessairement le cas. Ces images sont issues du même dataset et sont donc fort semblables aux données d'entrainement, mais permettent d'observer le comportement des prédictions.

Contrairement au *batch size* qui sera développé au point suivant, le temps d'entrainement est directement proportionnel au nombre d'époques. On compte donc 10min 55s pour 50 époques, 21min 12s pour 100 époques, et 1h 39m 3s pour 500 époques.

2.2.2.3. *Variation du batch size*

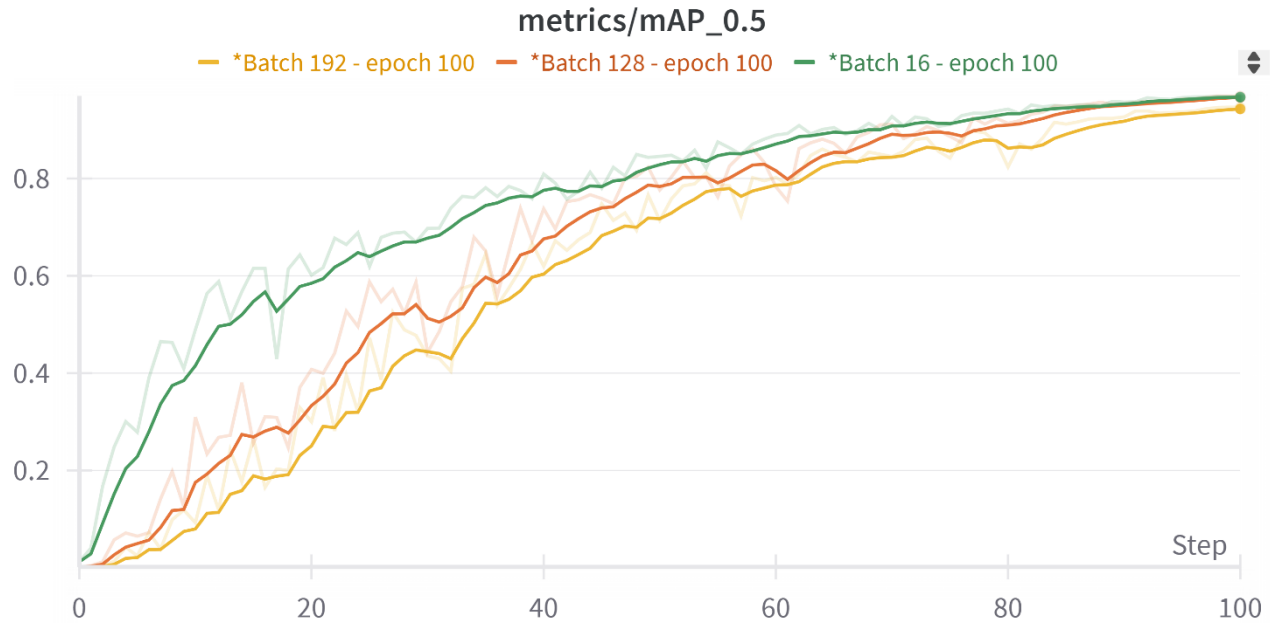


Fig. 70 – Evolution du mAP de YOLOv5 pour des entrainements avec différentes tailles de lot

Le *batch size*, ou taille du lot, fait référence au nombre d'images traitées simultanément par le modèle avant que ses paramètres, comme les poids, ne soient mis à jour. Pour un dataset de 861 images, un *batch size* de 16 signifie donc que chaque époque nécessite 54 itérations (861 divisé par 16) pour couvrir toutes les images. Ce paramètre peut aller de 1, en mettant à jour le modèle après chaque image, à l'entièreté du dataset. Ceci est souvent déconseillé car le processus demande énormément de RAM, et risque de demander plus d'époques pour que le modèle converge, malgré un temps d'exécution pour chaque époque plus faible. Ces extrêmes sont donc rarement utilisés en Deep Learning, et l'apprentissage par mini-lot est souvent préféré avec des valeurs comme 16 ou 32. Dans la fig. 70, on remarque qu'un *batch size* élevé ralentit la convergence du modèle, ce qui veut dire qu'il a besoin de plus d'époques pour se stabiliser, alors que le *batch size* 16 obtient un meilleur mAP tout au long de l'entrainement. Les 100 époques suffisent néanmoins pour que les modèles se stabilisent et obtiennent des performances similaires, avec un mAP de 96.7, 96.6 et 94.2% pour les tailles de lot de 16, 128 et 192.

L'augmentation du *batch size* a également permis d'accélérer l'entrainement : Il a fallu 18m 54s et 19m 2s pour les tailles de lot 192 et 128, contre 21m 12s pour la taille de lot 16. Ceci correspond à une accélération de l'entrainement d'un peu plus de 10% par rapport à celui d'origine.

Cependant, la mémoire nécessaire a plus que doublé pour le CPU, qui est passé d'un peu moins de 4GB à 8.6GB et 7.9GB pour les tailles de lot 192 et 128 (fig. 71a). La fig. 71b illustre que le GPU a quant à lui vu sa mémoire utilisée passer de 2.3GB à 15.1GB et 14.9GB, ce qui représente une augmentation de près de 650%. De plus, les deux dernières valeurs sont proches de la limite de 16GB du GPU utilisé, et le modèle donnait une erreur avec un *batch size* supérieur à 192 car la mémoire demandée était trop importante. Au vu de la nature exponentielle de la mémoire GPU utilisée pour un *batch size* plus important et des faibles gains en vitesse d'exécution, il est conseillé de conserver les paramètres à 16 ou 32, voire de les baisser si le GPU utilisé n'est pas suffisamment puissant.

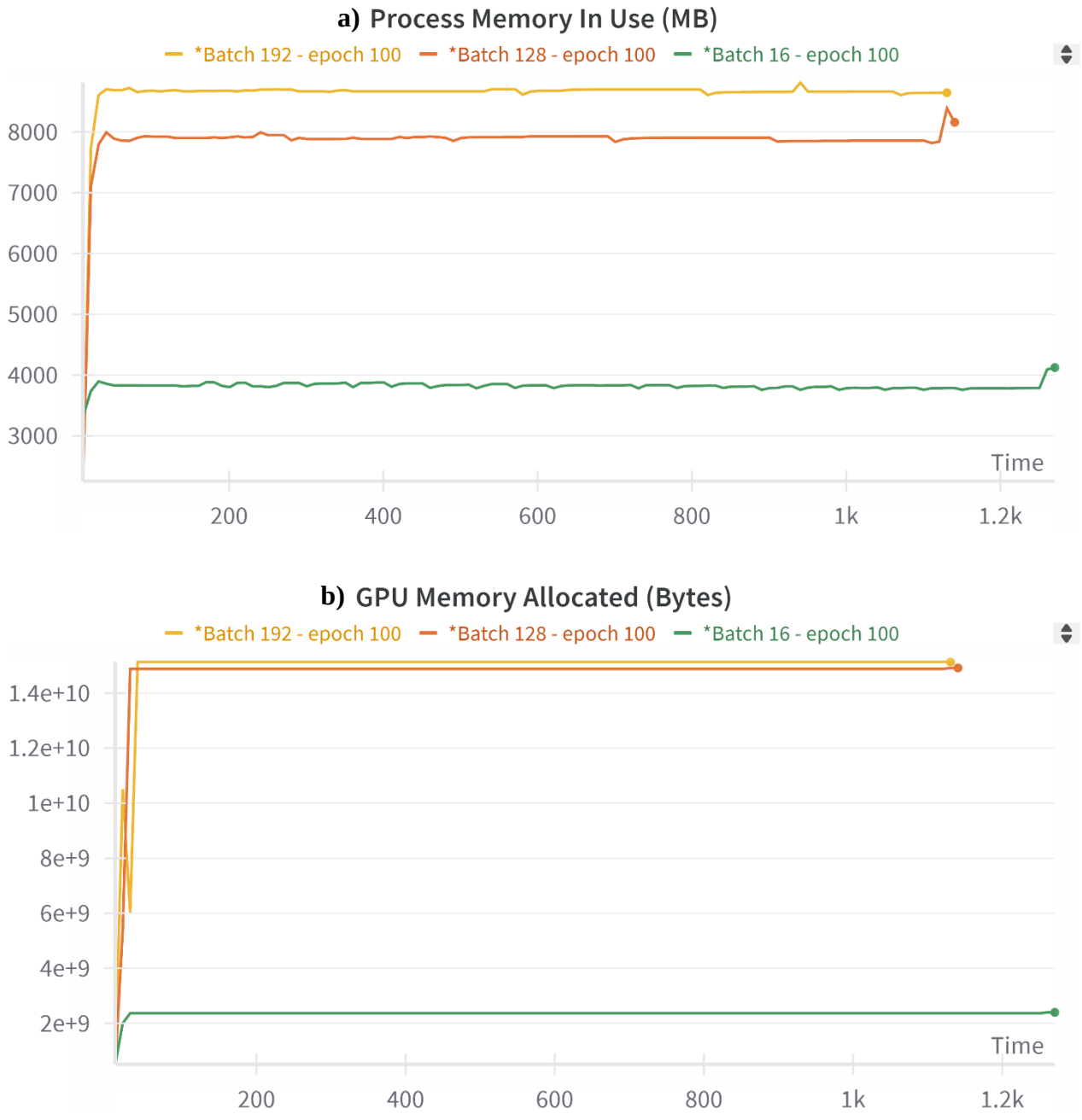


Fig. 71 – Mémoire de CPU (a) et GPU (b) nécessaire pour des entrainements de YOLOv5 avec différentes tailles de lot

2.2.2.4. Résultats de l'entrainement

Les 861 images reprenant 2073 instances de fissures ont donc été utilisées pour l'entrainement sur 100 époques en 0.346 heure (20 minutes) avec un GPU NVIDIA Tesla T4 d'une mémoire de 16 GB et les mêmes paramètres que l'article d'origine. Le modèle enregistre de nombreuses données liées à l'entrainement, comme les valeurs de mAP, précision et *recall*, pour chaque époque.

En réalité, le modèle conserve automatiquement deux poids (fig. 72a) : celui de la meilleure époque (*best.pt*) et celui de la dernière (*last.pt*). Si on la compare à l'entrainement de l'article original, cette instance de l'entrainement obtient une précision, un *recall* et un mAP50 tous supérieurs à 90% (fig. 72b). On obtient avec cet entrainement un mAP50 de 96.8% et un mAP50-95 de 68.3%, contre 81.6 et 51.2% pour Karimi et al. (2024). Après de nombreux essais, le modèle semble toujours atteindre ces valeurs de mAP et les raisons de cette différence n'ont pas pu être expliquées, si ce n'est que YOLOv5 reçoit chaque semaine des modifications à son code qui auraient pu améliorer ses performances depuis la publication de l'article.

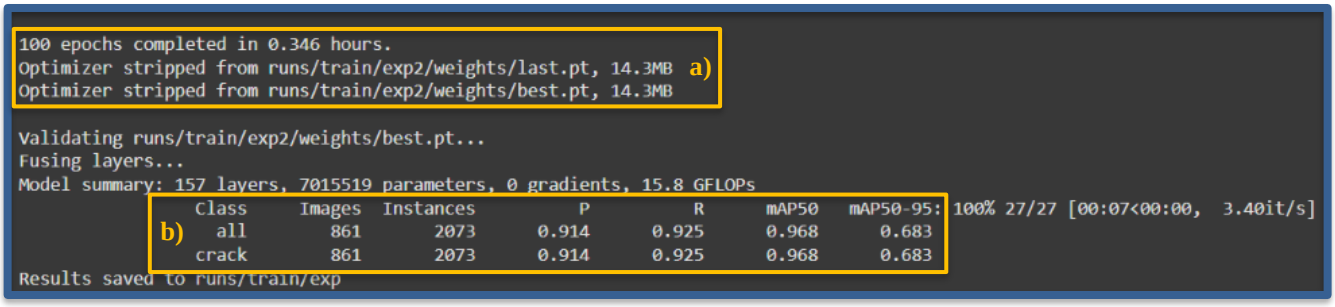


Fig. 72 – Extrait des logs de YOLOv5 après entrainement du modèle sur 100 époques

Si l'entrainement est satisfaisant, il est important d'exporter les poids générés par celui-ci, étant donné que l'environnement disparait une fois qu'il est déconnecté. L'exécution de la commande ci-dessous (fig. 73) permet de transférer une copie de tous les fichiers d'un dossier ciblé de l'environnement du modèle vers un autre dossier sur Google drive. Les fichiers peuvent aussi être téléchargés manuellement sur l'ordinateur depuis la barre de navigation. On peut changer le dossier « *train* » par « *detect* » dans le chemin d'accès de la commande afin d'exporter les prédictions de la même manière.

```
[ ]  !cp -r ./runs/train/exp/ "/content/MyDrive/MyDrive/Colab Notebooks/RUNS"
```

Fig. 73 – Commande d'export vers Google drive de YOLOv5

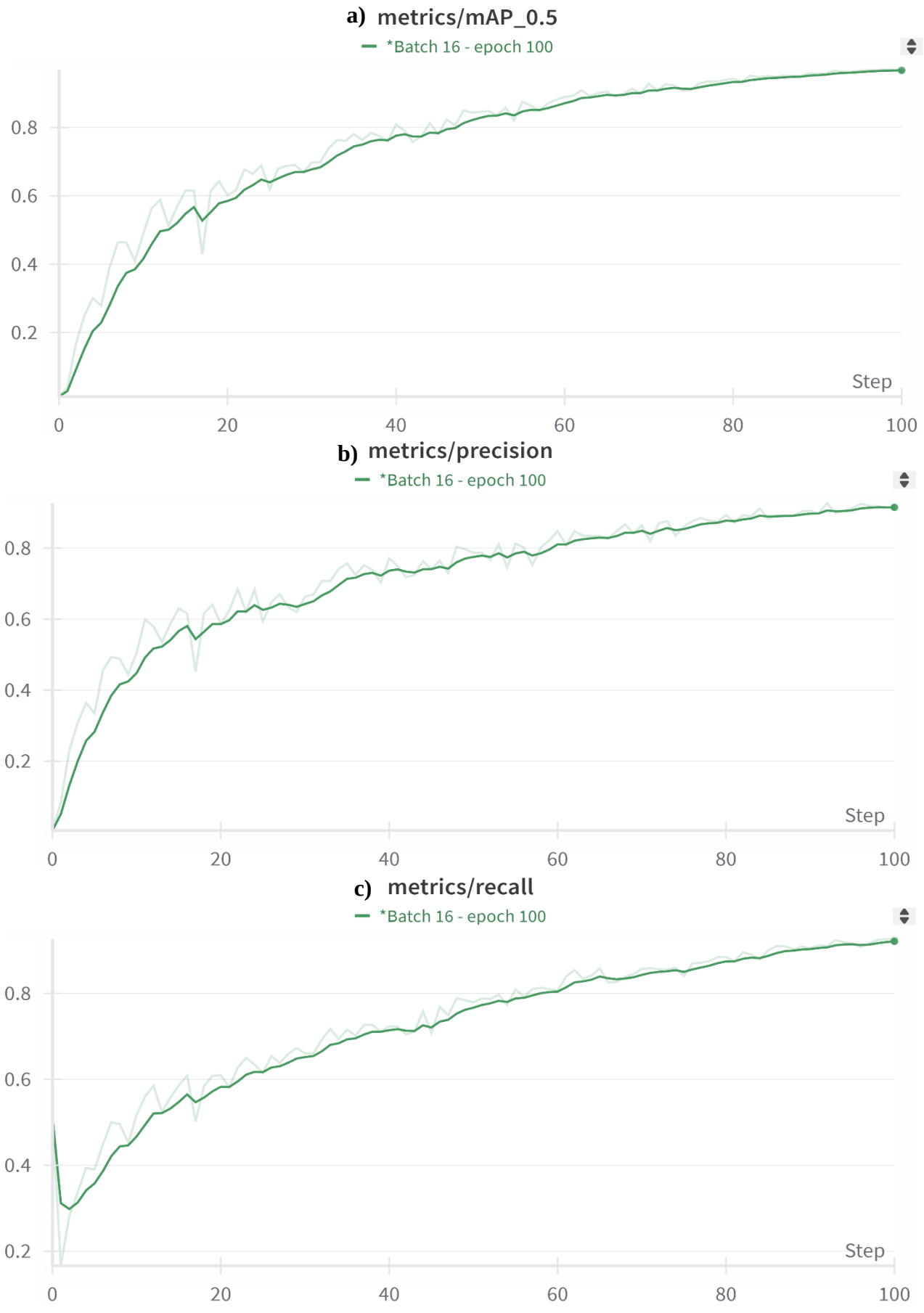


Fig. 74 – Evolution du mAP (a), de la précision (b) et du *recall* (c) de YOLOv5 durant son entrainement

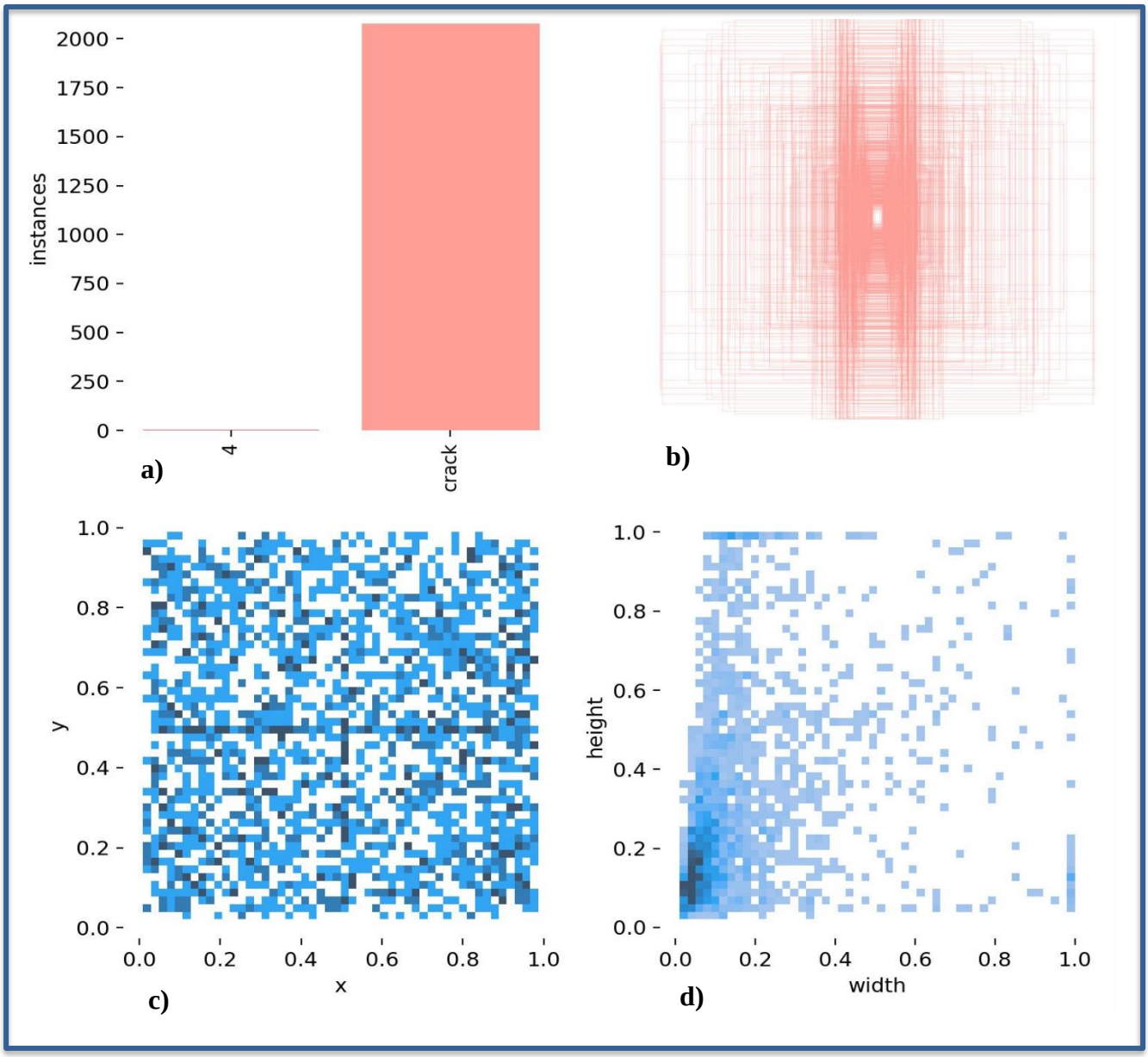


Fig. 75 – Représentation des labels de fissures du dataset

Les graphes ci-dessus sont automatiquement générés par YOLOv5 après chaque entrainement et représentent diverses données liées au dataset utilisé. La fig. 75a représente par exemple le nombre d'instances pour chaque classe du dataset. Ici, le modèle n'est entrainé qu'à détecter des fissures. Ensuite, les fig.75b et 75d représentent l'ensemble des labels, ou annotations, du dataset. On remarque rapidement que malgré les techniques d'augmentation, une majorité écrasante de ces labels sont verticaux, avec une largeur très faible et une longueur variable, malgré une plus forte concentration de petits labels. Ceci peut en partie justifier les niveaux de confiance plus bas lorsque le modèle détecte une typologie de fissure qu'il a moins observé lors de son entrainement. La fig. 75c montre quant à elle la répartition de ces labels sur l'image. La représentation est ici plus homogène.

2.2.3. Validation de l'entrainement

Après l'entrainement, YOLOv5 génère automatiquement des prédictions sur l'ensemble de validation présent dans le dataset, ce qui permet de les comparer avec les labels de ce dernier. Les prédictions sont majoritairement correctes, avec des niveaux de confiance plutôt élevés, et on peut même observer des détections de fissures qui n'étaient pas considérées comme telles dans le dataset (fig. 76d). Comme pour Karimi et al. (2024), les fissures horizontales sont plus difficilement identifiées, voire non détectées.

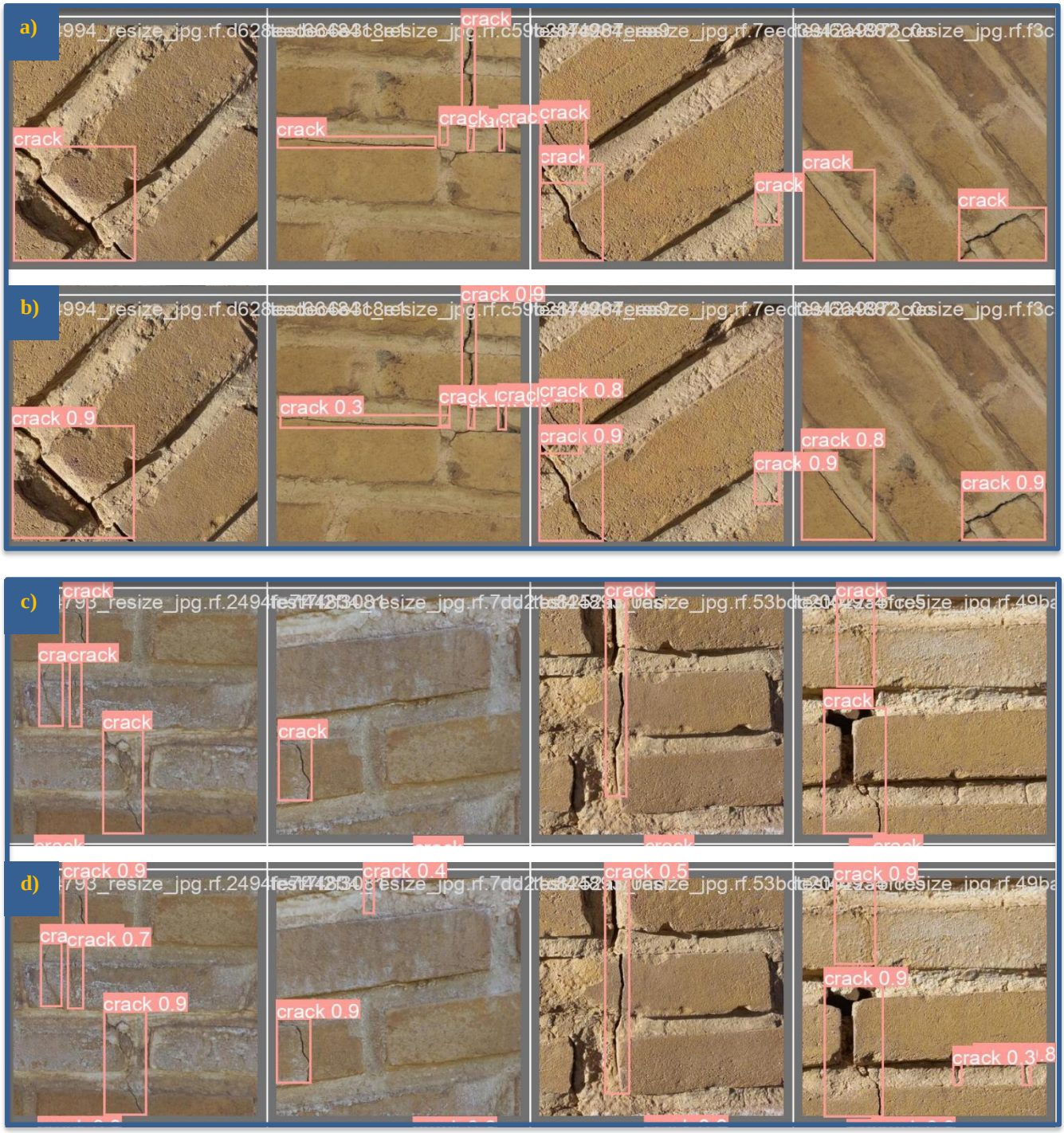


Fig. 76 – Comparaison entre les images de validation du dataset (a et c) et les prédictions de YOLOv5 entrainé sur 100 époques (b et d)

Pour vérifier plus en profondeur les performances du modèle des entrainements de 100 et 500 époques, il a été décidé de poursuivre cette validation manuellement sur des nouvelles images récupérées sur Google images, représentant des fissures de maçonnerie diverses.

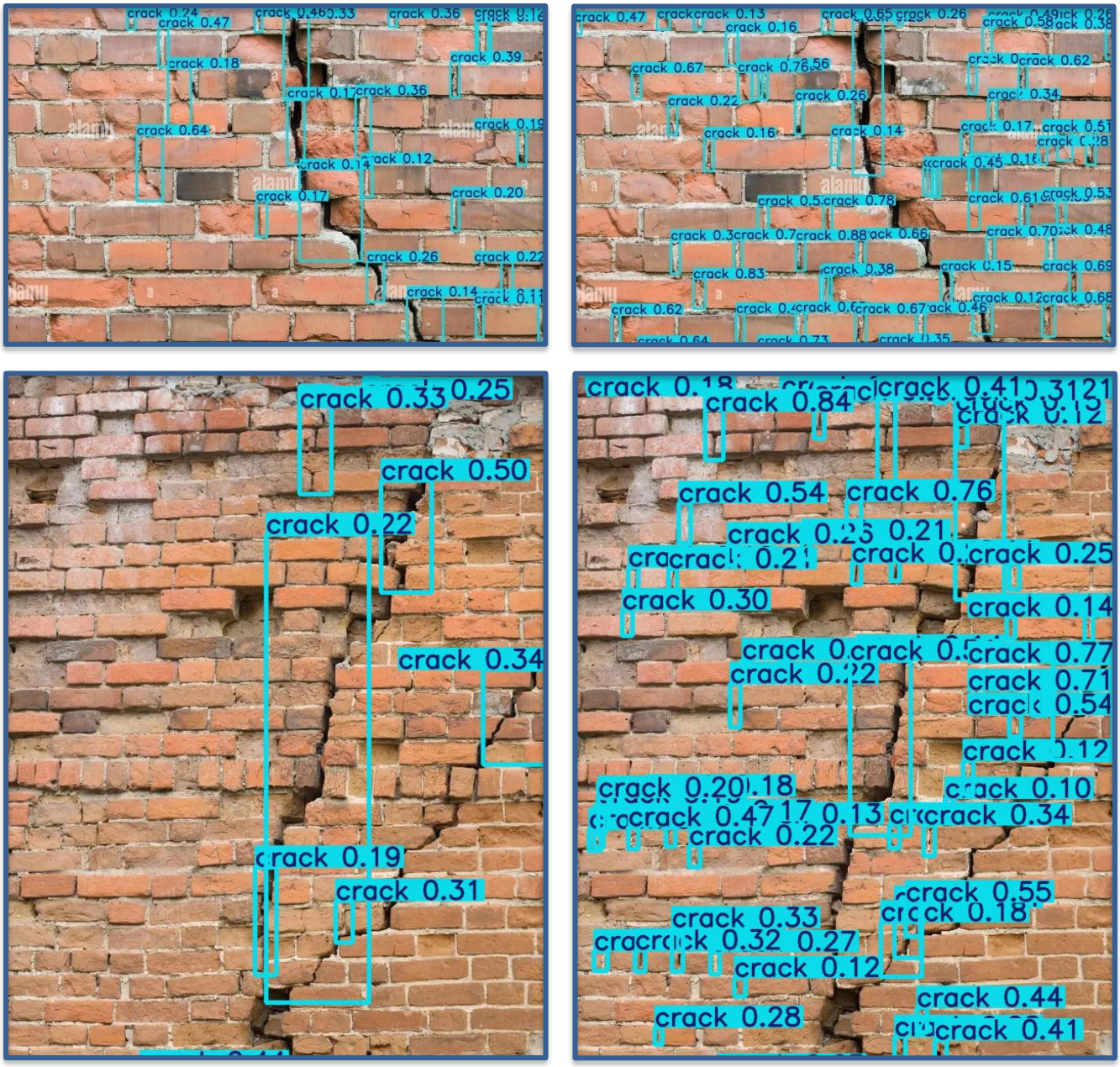


Fig. 77 – Validation supplémentaire de l'entrainement pour 100 et 500 époques, à l'aide de photos trouvées sur Google images

*Train : batch 16 – epoch **100***	*Train : batch 16 – epoch **500***
Detect : img 416 – conf 0.1	*Detect : img 416 – conf 0.1*

Les deux modèles montrent des résultats variés pour ces images de validation. L'arrière-plan complexe laisse entrevoir les faiblesses du modèle surentrainé sur 500 époques par rapport à celui de 100 époques (fig. 77). Il produit beaucoup plus de faux positifs, principalement sur les joints de mortier, et néglige les vraies fissures. Le modèle entrainé sur 100 époques obtient globalement de meilleures performances.

2.3. Acquisition et traitement des images

2.3.1. Matériel et paramètres

L'ensemble des images présentées dans ce chapitre ont été capturées par un appareil photo Nikon Z 6 avec une résolution de 6048x4024 pixels. Pour les photos de nuit, un flash Nikon speedlight SB-700 a été employé. Un trépied Vanguard a permis de marquer puis réutiliser les mêmes positions entre les photos de jour et de nuit, afin d'obtenir les images les plus similaires possibles (fig. 78).

Fig. 78 – Appareil photo et matériel

Puisque l'expérience s'est déroulée sur une courte période durant le mois de décembre, il n'a pas été possible de prendre en compte toutes les conditions susceptibles d'être observées pendant l'année. La localisation des échantillons observés et la saison ont donc empêché de considérer des façades surexposées et sous-exposées au soleil. De ce fait, l'impact d'une ombre partielle ou totale sur les performances du modèle n'a pas été étudié.

Certains paramètres de l'appareil ont été adaptés manuellement afin d'assurer une qualité optimale selon le contexte de chaque image. Ils sont indiqués pour chaque résultat présenté. Le paramètre de balance des blancs a néanmoins été conservé sur « nuageux » pour l'entièreté de l'étude. Par défaut, les photos ont été capturées avec un temps d'exposition de 1/30 et une ouverture F/5. Pour les photos de jour, les ISO ont été maintenu à 100. Le choix de ces valeurs est basé sur les valeurs conseillées pour une photogrammétrie.

Paramètres de la commande de détection

Au même titre que pour l'entrainement, on dispose dans cette commande de plusieurs variables qui peuvent impacter profondément les prédictions du modèle.

```
[ ]  !python detect.py --weights runs/train/exp/weights/best.pt --img 416 --conf 0.1 --source '../MyDrive/MyDrive/Colab Notebooks/test.jpg'
```

Fig. 79 – Commande de détection de YOLOv5

a)
```
[ ]  !python detect.py --weights runs/train/exp/weights/best.pt
```

b)
```
[ ]  !python detect.py --weights '../MyDrive/MyDrive/Colab Notebooks/RUNS/exp100/weights/best.pt'
```

Fig. 80 – Première moitié de la commande de détection

La première moitié de la commande concerne le chemin d'accès aux poids. Elle utilise par défaut les meilleurs poids issus du premier entrainement (fig. 80a). Si l'entrainement a déjà été réalisé sur une autre session et exporté, il faut indiquer le chemin d'accès du fichier à travers le Google drive (fig. 80b). Il n'est pas possible d'accéder directement aux documents de l'ordinateur, c'est pourquoi télécharger manuellement les poids après l'entrainement est déconseillé, car cela demande d'importer ensuite les fichiers sur Google drive manuellement.

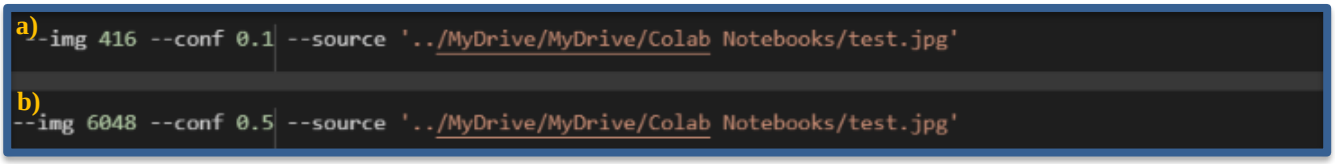


Fig. 81 – Seconde moitié de la commande de détection

Comme pour l'entrainement, *img* indique à quelle résolution les images vont être réduites pour passer dans le modèle. Elle est par défaut à 416 pixels comme pour les images du dataset (fig. 81a) , mais peut aller jusqu'à la résolution originale de l'image (fig. 81b). Il est également possible d'augmenter la résolution au-dessus de l'originale, mais cela ne fait que ralentir le modèle sans obtenir de meilleurs résultats. Le modèle effectue en moyenne une itération à 416 pixels en 29.5ms, contre 313.2 ms pour la même image à 6048 pixels. La variable *conf* concerne quant à elle les niveaux de confiance du modèle qui sont affichés sur l'image de sortie. Elle varie de 0 à 1, et une valeur proche de 1 sert à trier automatiquement les confiances de prédictions pour ne présenter que les plus hautes. Dans le cas d'une détection de pathologies sur une façade, il semble plus avisé de garder ce paramètre bas afin d'observer l'ensemble des éléments détectés, et de supprimer ensuite les prédictions non-pertinentes. Comme pour les paramètres de l'appareil photo, ceux-ci vont varier d'image à image, et ils seront indiqués pour chacune d'entre elles. La *source* fait référence au chemin d'accès pour l'image ou le dossier qui doit être détecté.

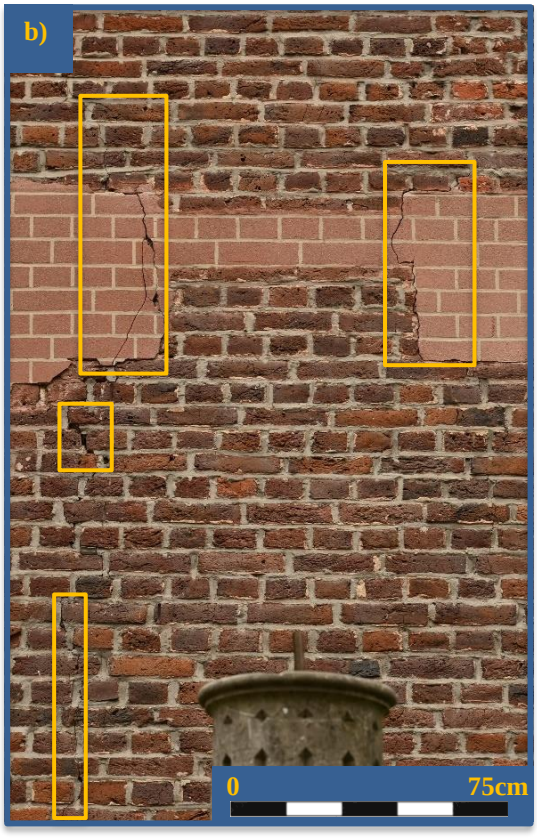
2.3.2. Contexte du cas d'étude

Toutes les images ont été prises dans l'enceinte partagée de la faculté d'architecture de l'université de Liège et de l'école supérieure des arts de Saint-Luc, à Liège. Trois localisations de fissures ont principalement été observées, mais d'autres exemples ont permis de compléter l'étude. Ces trois échantillons sont sélectionnés de manière à varier les tailles et typologies des fissures mais également les arrière-plans.

Les prédictions réalisées par YOLOv5 sont toujours présentées en bleu cyan, tandis que les annotations réalisées manuellement le sont en jaune. Néanmoins, certaines annotations du modèle seront parfois accentuées en jaune et indiquées comme tel dans le texte pour assurer une bonne visibilité des résultats.

2.3.2.1. Echantillon A – Façade sud de la cafétaria

Fig. 82 – Echantillon A, façade sud de la cafétaria



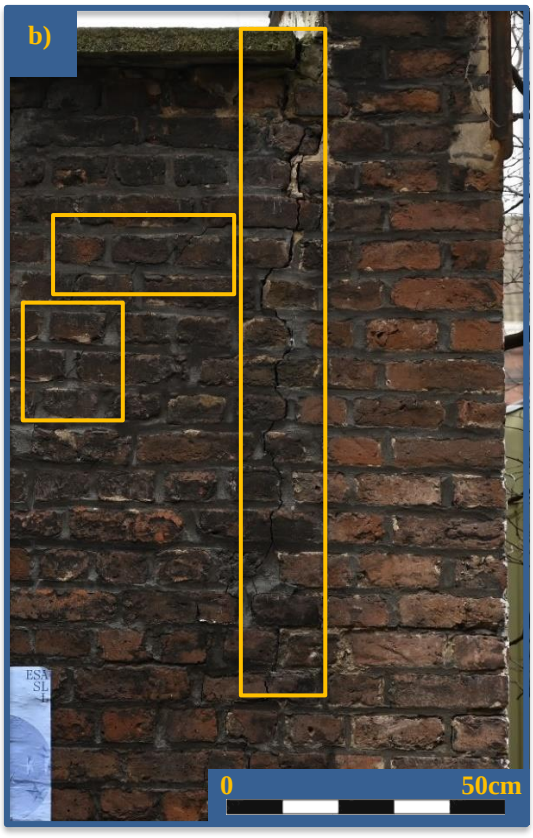
Ce premier échantillon va permettre d'observer la réponse du modèle à un changement de matérialité pour une même fissure. Il présente exclusivement des pathologies verticales, similaires aux données d'entrainement (fig. 82b). Les prédictions devraient donc être relativement proches de la réalité.

De nombreuses petites fissures dans les briques sont également observables. Il est intéressant de voir si YOLOv5 est capable d'également les détecter, malgré sa difficulté à déceler de petits objets, selon la littérature.

Plus largement, il est intéressant de voir l'impact sur les prédictions d'un contexte non-pertinent comme la végétation, les colonnes et les baies vitrées ou leur reflet (fig. 82a).

2.3.2.2. Echantillon B – Mur extérieur

Fig. 83 – Echantillon B, mur extérieur

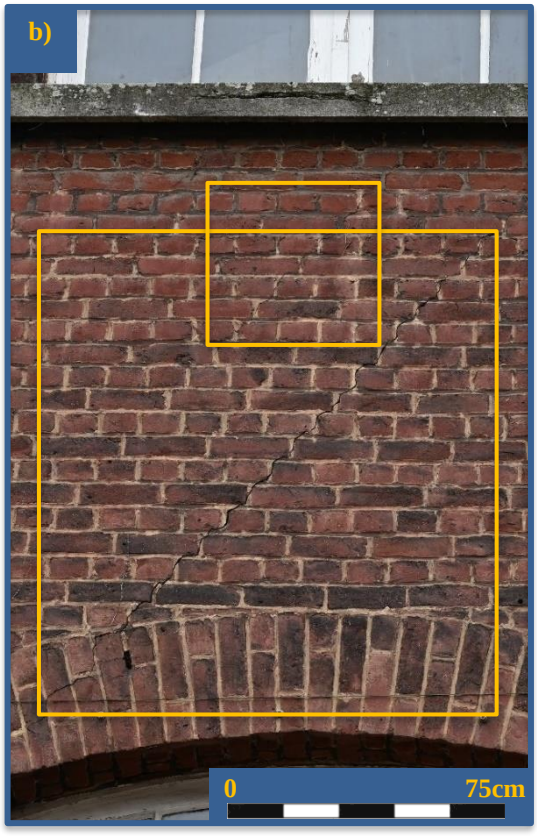


Avec un arrière-plan encore différent, YOLOv5 va ici devoir détecter des fissures aux orientations verticales, diagonales ou horizontales. Il faut également observer si la prédiction de la grande fissure se fait en une seule instance comme sur l'exemple de la fig. 83b, ou si le modèle n'en détecte que plusieurs fractions et superpose ses prédictions.

En théorie, le modèle devrait malgré tout rester performant sur la fissure verticale, car elle est semblable aux données d'entrainement malgré sa taille plus importante.

2.3.2.3. Echantillon C – Façade nord du bâtiment central

Fig. 84 – Echantillon C, façade nord du bâtiment central



Aucune fissure en escalier n'est observable sur plus de 2000 instances d'entrainement dans le dataset, c'est pourquoi il est intéressant de voir si YOLOv5 est capable d'identifier correctement les deux fissures de cet échantillon.

La grande fissure observable sur la fig. 84b est d'autant plus complexe qu'elle a une taille importante, ce qui risque, comme à l'échantillon B, de voir apparaitre plusieurs prédictions pour la même instance.

Peu importe les performances de cet exemple, on remarque que les fissures avec une typologie autre que verticale ou horizontale nécessitent une aire de prédiction beaucoup plus large que la pathologie elle-même, diminuant la lisibilité.

2.3.3. Premières observations

Avant d'analyser les capacités du modèle sous des conditions différentes du dataset, les images sont recadrées de manière que les fissures à analyser aient la même proportion que les fissures d'entrainement par rapport au reste de l'image. Ainsi, les conditions sont théoriquement idéales pour le modèle.

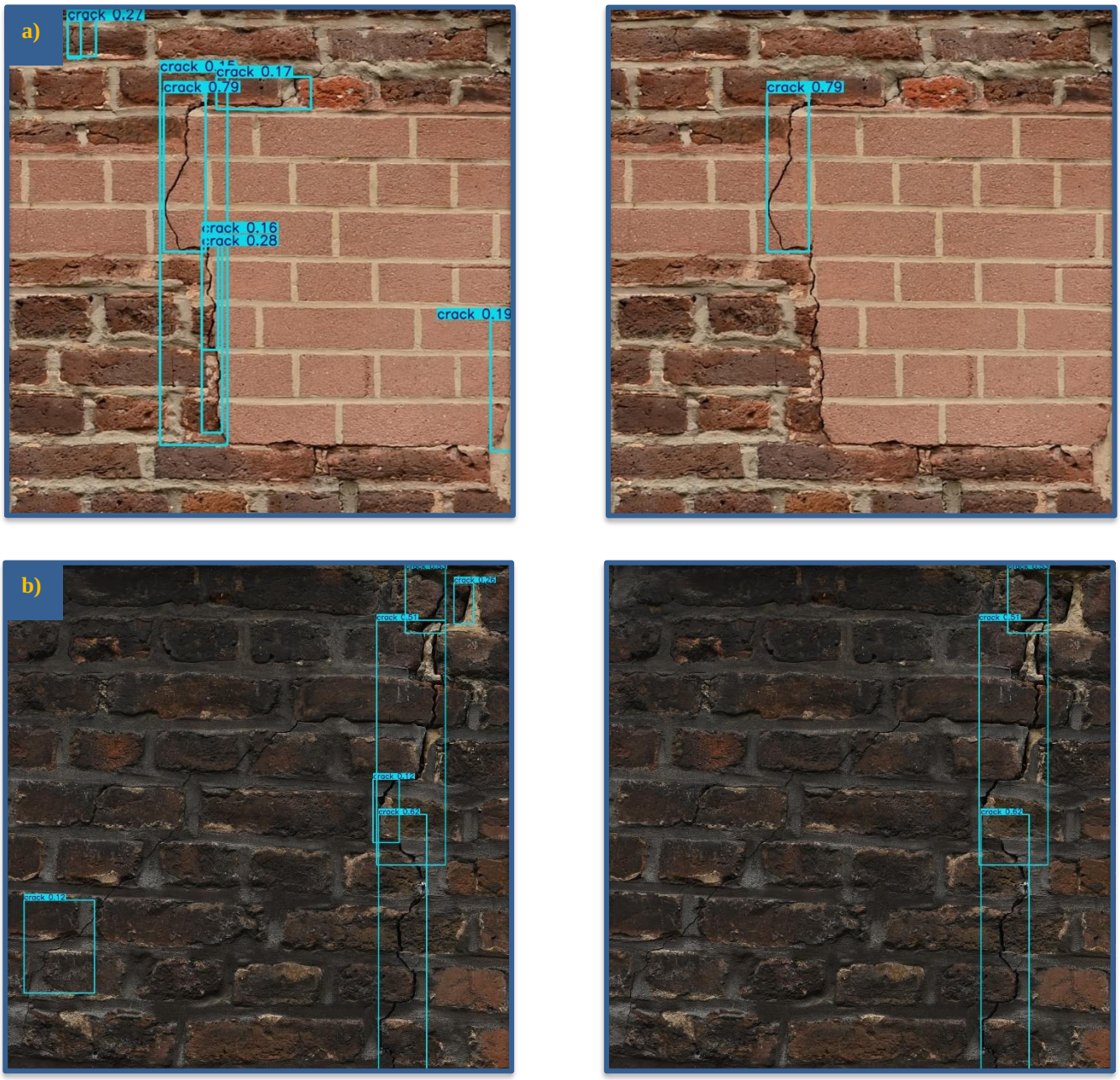


Fig. 85 – Détection de fissures verticales pour les échantillons A et B

*Detect : img 416 – conf **0.1***	*Detect : img 416 – conf **0.3***

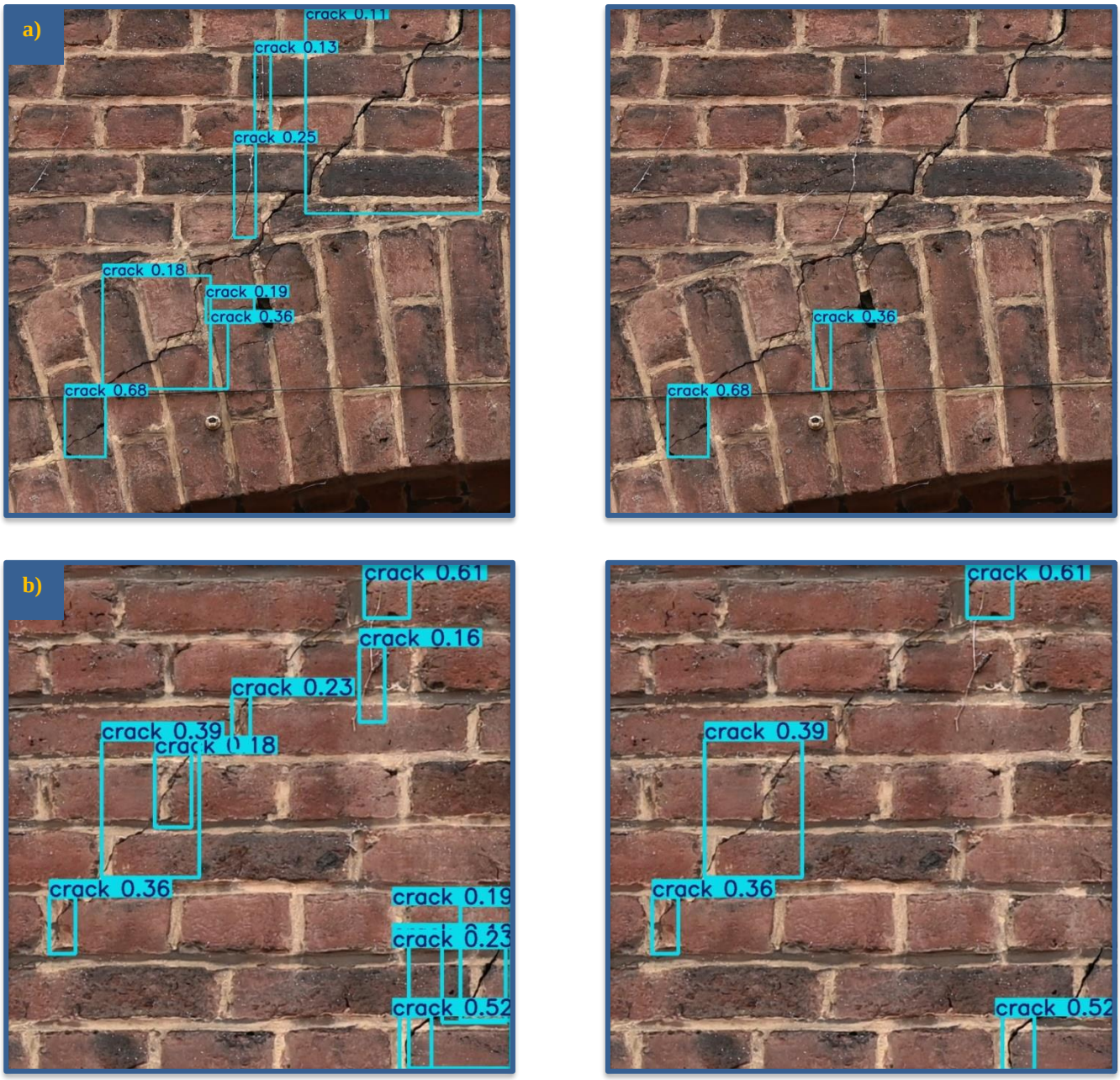


Fig. 86 – Détection de fissures en escalier pour l'échantillon C

Detect : img 416 – conf ***0.1***

Detect : img 416 – conf ***0.3***

Pour rapidement visualiser les performances de YOLOv5, les images de la colonne de droite représentent les mêmes détections qu'à gauche, mais sans représenter les prédictions dont le niveau de confiance est inférieur à 0.3. On observe que les fissures verticales sont les plus facilement détectées, malgré une localisation incomplète entre les deux matérialités de la fig. 85a. Le modèle ne détecte pas dans ce cas-ci la fissure horizontale dans la fig 85b. Pour la fig. 86, la majorité de la fissure est détectée, mais pas en une seule instance, et les niveaux de confiance sont très bas. Pour une raison inconnue, la fissure 86b semble obtenir de meilleurs niveaux de confiance que celle de 86a, alors que la 86b est moins visible.

Dans l'ensemble, les résultats sont très satisfaisants pour YOLOv5. Un entrainement de 20 minutes suffit à obtenir ces prédictions dans des contextes très différents de ceux présentés dans le dataset. Malgré une superposition parfois importante des prédictions, les typologies de fissures semblables aux données d'entrainement sont correctement identifiées, et le modèle est capable de détecter des typologies qu'il a moins rencontré. Augmenter la taille du dataset pour inclure plus de typologies devrait pouvoir améliorer les performances.

Cependant, ce processus n'a aucun intérêt pour un architecte ou un expert cherchant à diagnostiquer une façade ou un bâtiment entier. En effet, localiser les pathologies avant de les prendre en photo ou en les recadrant ex-situ pour ensuite utiliser le modèle est moins efficace qu'une annotation manuelle des mêmes images. En plus d'observer les réactions du modèle par rapport à un environnement non-pertinent ou des typologies variées, il faut avant tout vérifier que la détection d'une pathologie peut se faire à des distances plus importantes. Selon les résultats, il sera pertinent d'ensuite vérifier si l'angle sous lequel la photo est prise influence les prédictions ou si le modèle est capable de s'adapter.

2.3.4. Distance de l'objet

Fig. 87 – Détection à distance pour l'échantillon A

Detect : img 416 - conf 0.1

Distance : 7m *Distance focale : **70mm***

Fig. 88 – Détection à distance pour l'échantillon C

Detect : img 416 - conf 0.1

Distance : 7m *Distance focale : **24mm***

Lancer une prédiction sur des images plus éloignées des pathologies tout en conservant la résolution à 416 pixels diminue drastiquement la qualité des résultats. On remarque cependant que les éléments que le modèle détecte et reconnait comme des fissures ont des caractéristiques à cette échelle qui sont similaires aux fissures observées pendant l'entrainement. Les faux positifs du modèle (accentués en jaune sur les fig. 87 et 88) les plus récurrents sont les branches et les croisillons verticaux des fenêtres.

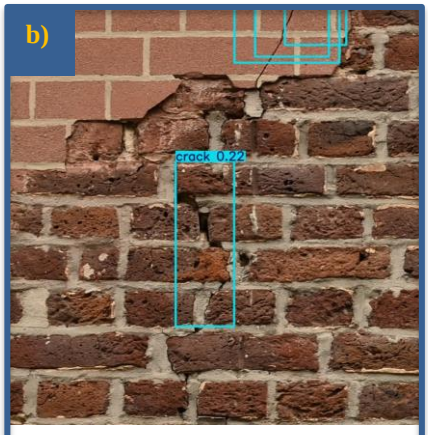
Ceci peut s'expliquer avec le *Ground Sampling Distance* (GSD), qui correspond à la distance entre les centres de deux pixels consécutifs sur une image. Beaucoup utilisé en photogrammétrie, cette mesure permet de connaitre la résolution spatiale des images obtenues, et donc l'échelle de ces dernières. Par exemple, une valeur de 10mm/pix indique que chaque pixel mesure 10x10mm en réalité. Le GSD varie selon la taille physique du pixel sur le capteur de l'appareil, sa longueur focale et la distance par rapport à l'objet (Hallot et al., 2022). Dans le cas du Computer Vision et de la détection de pathologies, le GSD doit donc être le plus similaire possible au GSD des données d'entrainement pour des résultats optimaux. Si le GSD est trop grand par rapport au dataset, le modèle risque comme dans l'exemple ci-dessus de ne pas être capable de détecter les pathologies recherchées. S'il est trop petit, il est possible que le modèle réalise de mauvaises prédictions en identifiant beaucoup de faux positifs.

Comme pour la photogrammétrie, il est donc important de conserver une distance focale et une distance avec l'objet constante pour toutes les images afin d'éviter une répartition trop large des valeurs de GSD. Puisque qu'il est impossible de calculer le GSD des images d'entrainement, les prochaines prédictions vont comparer différentes résolutions afin d'identifier les meilleurs résultats. Cette expérience va se faire avec l'échantillon A car il présente les meilleures conditions pour différentes échelles de fissures. La distance focale des images suivantes est de 70mm et la distance à l'objet est de 7m. Les paliers de résolutions ont été choisis arbitrairement et sont en grande partie des multiples de 416, résolution par défaut de YOLOv5.

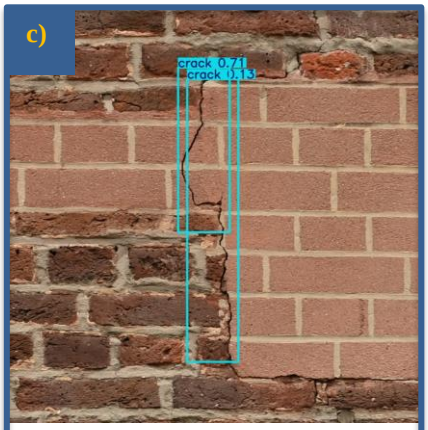
Dès le passage à une résolution de 624 pixels, les deux fissures principales sont détectées. Les niveaux de confiance du modèle s'améliorent à 832 pixels et le modèle détecte une 3^{e} fissure (fig. 89b). L'aire des prédictions semble englober une majorité des fissures détectées malgré une superposition de certaines d'entre elles (fig. 89c). Il y a peu de faux positifs, et ils sont facilement identifiables car les prédictions sont peu nombreuses. A partir de 1040 pixels, la 4^{e} fissure est détectée, et les 3 premières sont toujours facilement identifiées. Le palier de 1664 pixels marque les premières détections de microfissures au sein même d'une brique ou le long d'un joint de mortier (fig. 90c). La quantité de détections et de faux positifs reste acceptable, et ces derniers ne représentent toujours pas la majorité des prédictions du modèle. La qualité et les niveaux de confiance des prédictions chutent drastiquement à 4000 et 6048 pixels pour les fissures principales (fig. 91d), alors qu'on observe des détections très précises de nombreuses microfissures (fig. 91c). La quantité de détections et de faux positifs grandit, et il devient difficile de visualiser rapidement les prédictions correctes. Dans ce travail, 832 pixels sont idéals pour identifier rapidement les défauts principaux de la zone d'étude, alors que 1664 pixels permettent de voir la quasi-totalité des fissures, même minimes.

Fig. 89 – Détection sur l'échantillon A avec une image de 832 pixels

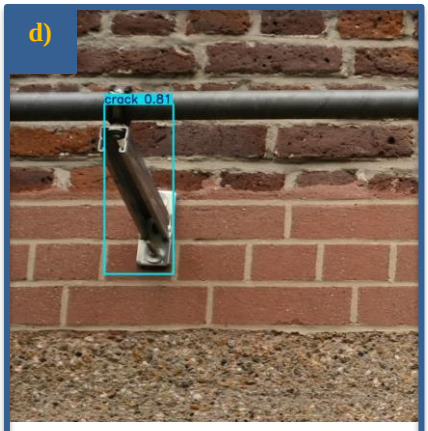
*Detect : img **832** - conf 0.1*	*Distance : 7m*	*Distance focale : 70mm*



Première détection de cette fissure, le niveau de confiance reste bas.



Prédictions satisfaisantes pour les fissures verticales, avec une confiance de 0.71.



Faux positif du modèle, la couleur et l'orientation en sont probablement la cause. Il est systématiquement présent pour les résolutions de 416 à 1664 pixels.

a)

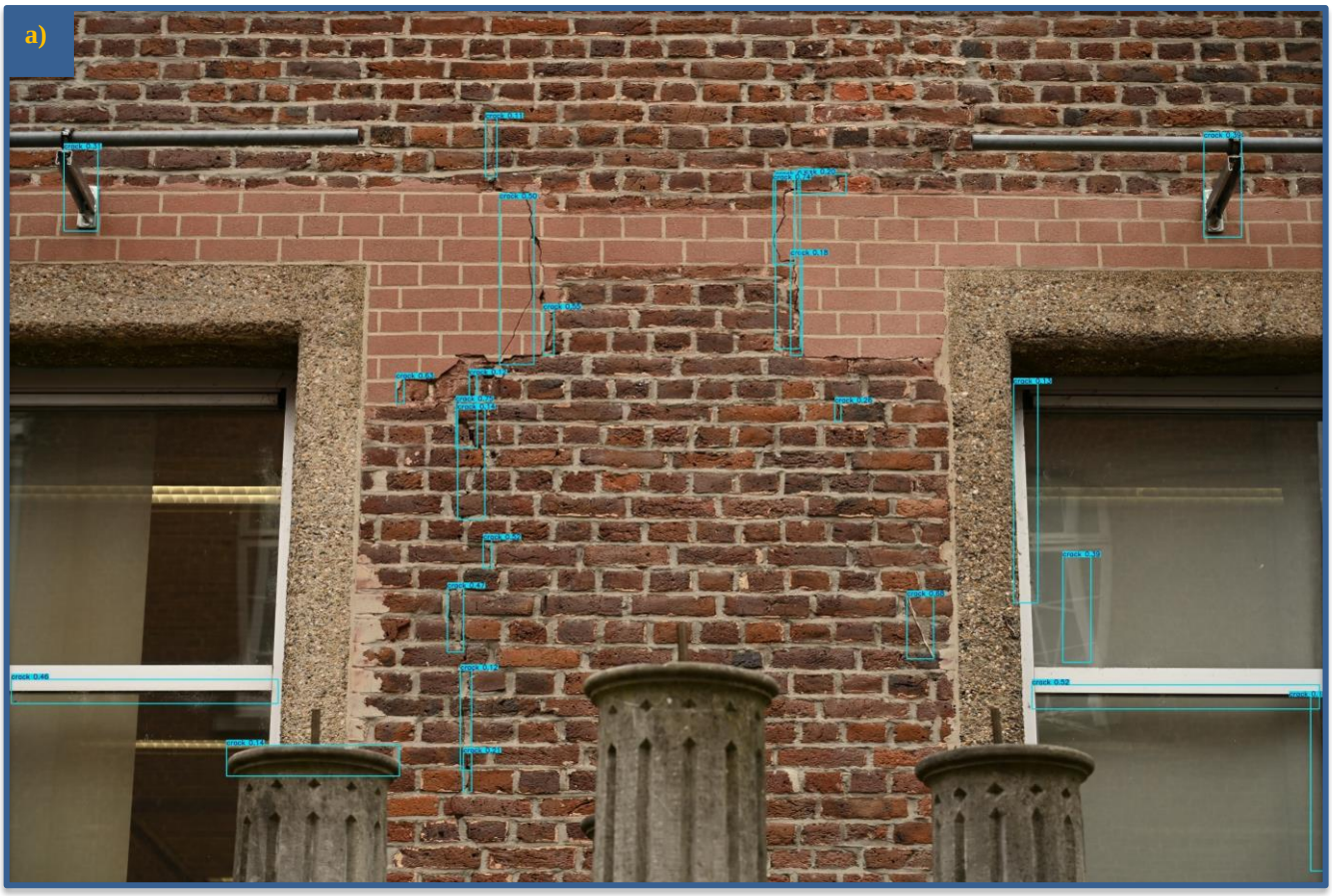
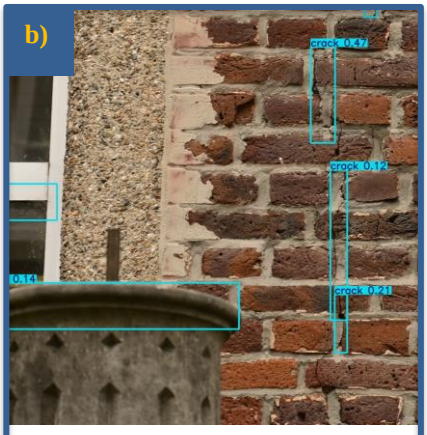


Fig. 90 – Détection sur l'échantillon A avec une image de 1664 pixels

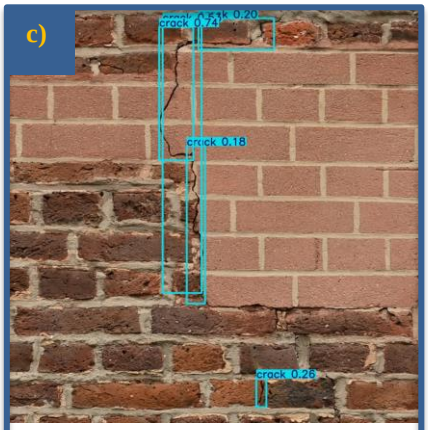
*Detect : img **1664** - conf 0.1*	*Distance : 7m*	*Distance focale : 70mm*

b)



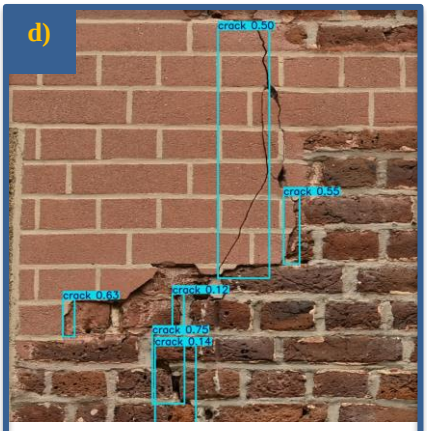
Première détection de la 4e fissure. Des faux positifs autres que les croisillons apparaissent.

c)



La majorité des deux grandes fissures verticales est identifiée avec un niveau de confiance toujours satisfaisant. Un exemple de microfissure est observable.

d)



La confiance du modèle s'améliore par rapport aux prédictions à 832 pixels.

a)

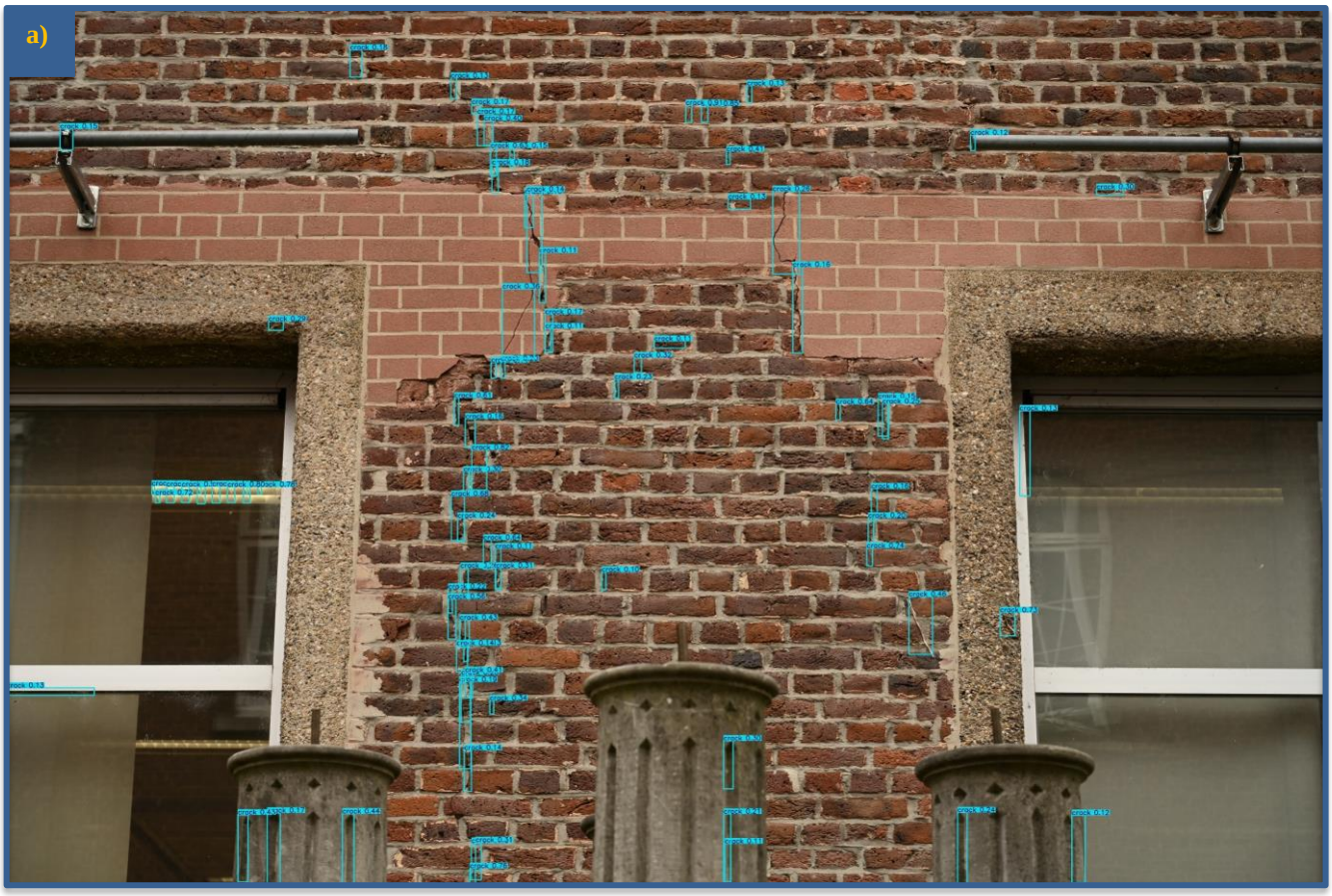
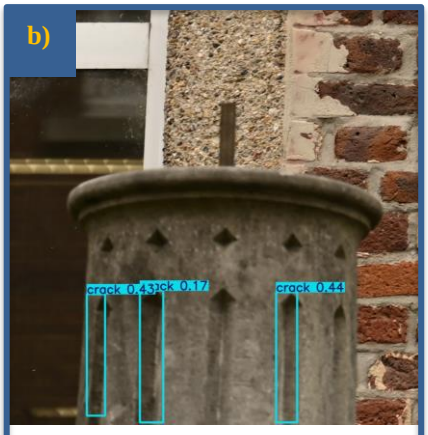


Fig. 91 – Détection sur l'échantillon A avec une image de 6048 pixels

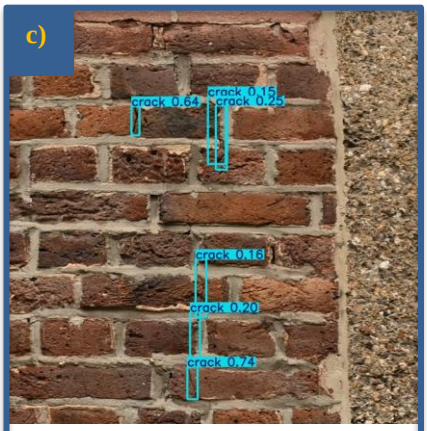
*Detect : img **6048** - conf 0.1*	*Distance : 7m*	*Distance focale : 70mm*

b)



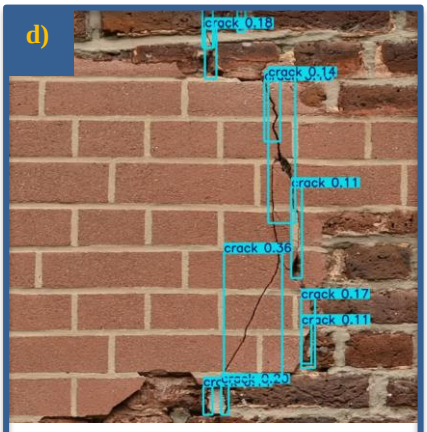
Les faux positifs se multiplient et se diversifient. Les niveaux de confiance du modèle sont plus élevés pour ceux-ci que pour les vraies fissures

c)



De nombreuses microfissures sont détectées.

d)



Les prédictions des fissures principales éclatent et les niveaux de confiance baissent.

2.3.5. Angle d'incidence

Afin d'étudier l'impact de l'angle d'incidence par rapport à l'objet lors de la prise de vue, deux variations d'axes ont été explorées. Pour l'incidence verticale, on cadre le centre de la zone photographiée en se rapprochant progressivement de la façade et en restant perpendiculaire à celle-ci. Modifier l'angle d'incidence horizontal permet à l'inverse de conserver une distance constante mais en variant l'angle de prise de vue par rapport à la perpendiculaire de la façade.

Pour assurer la visibilité des résultats sur les fissures principales dans ce document, les images sont recadrées manuellement après passage dans le modèle. Les paramètres de détection et conditions de prises d'image restent toutefois inchangés, mais les résultats présentés se concentrent avant tout sur les fissures principales, identifiées au point 2.3.2, et non sur l'ensemble de la façade. Pour cette expérience, les échantillons A et C ont donc été choisis. Le premier va permettre un suivi des observations sur l'entièreté de l'analyse et le second doit tester l'adaptabilité de YOLOv5 à des typologies de fissures qu'il ne connait pas.

Concernant l'incidence verticale (fig. 92), les résultats sont très satisfaisants. YOLOv5 semble reconnaitre avec succès les fissures verticales et obtient des prédictions plus justes pour la fissure en escalier proche de la façade que pour l'image de référence (fig. 92b). Pour les angles d'incidence horizontaux de 45° et 60° (fig. 93), les détections des fissures verticales sont également acceptables. En revanche, la fissure en escalier n'obtient pas le même succès. La prédiction avec un angle d'incidence de 0° (fig. 93b) est toujours peu concluante à cause des données d'entrainement, et il semblerait qu'une variation de l'angle de prise de vue n'améliore pas les prédictions du modèle (fig. 93 d et f).

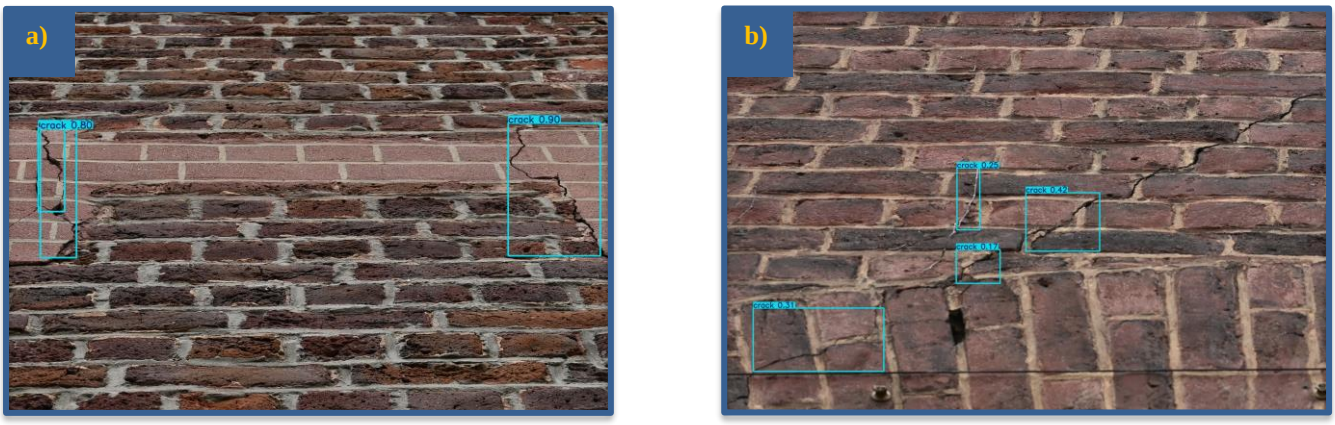


Fig. 92 – Détection sur les échantillons A et C avec un angle d'incidence vertical élevé

Detect : img 1664 - conf 0.1	*Distance : 1m*	*Distance focale : 24mm*

Angle d'incidence : 0° *Angle d'incidence : 0°*

Angle d'incidence : 45° *Angle d'incidence : 45°*

Angle d'incidence : 60° *Angle d'incidence : 60°*

Fig. 93 – Détection sur les échantillons A et C avec différents angles d'incidence horizontaux

Detect : img 1664 - conf 0.1	*Distance : 7m*	*Distance focale : 70mm*

La fig. 92 montre généralement de meilleures prédictions et niveaux de confiance que pour la fig 93, même par rapport aux images de référence (fig. 93 a et b). On peut donc supposer que le modèle s'adapte très bien à différents angles d'incidence verticaux. Cependant, ne pas se positionner perpendiculairement à la façade ou à l'objet risque d'affaiblir la confiance du modèle sur ses prédictions. Il reste toutefois préférable de conserver une perpendiculaire la plus fidèle possible par rapport aux deux axes pour éviter que la texture ou le relief de la façade ne dissimulent des détails.

2.3.6. Conditions lumineuses

ISO : 800 *ISO : 800*

ISO : 16000 *ISO : 16000*

ISO : 51200 *ISO : 51200*

Fig. 94 – Détection sur les échantillons A et B avec différentes valeurs d'ISO

Detect : img 1664 - conf 0.1	*Distance : 7m*	*Distance focale : 70mm*	***Flash : activé***

Pour les photos en luminosité basse avec flash, augmenter les ISO n'impacte pas négativement les prédictions de YOLOv5, mais leur qualité est cependant plus faible que les photos de jour. Il semblerait que la lumière générée par le flash de l'appareil photo illumine les fissures les plus larges sur l'échantillon C (fig. 94b, d et f), les rendant très peu visibles pour le modèle (et pour l'être humain), qui se base sur cette caractéristique pour identifier la pathologie. Les réglages ISO et de temps d'exposition devraient être prioritairement augmentés avant de considérer l'utilisation d'un flash.

2.3.7. Autres observations

2.3.7.1. Détection sur pathologies excentrées

Fig. 95 – Détection sur l'échantillon A avec un cadrage différent

Detect : img 1664 - conf 0.1	*Distance : 7m*	*Distance focale : 70mm*

Il a été remarqué tout au long de l'analyse de YOLOv5 que le modèle réagit correctement aux différentes variables qui lui sont imposées, mais les différentes expériences visent toujours précisément les fissures principales, qui sont donc centrées sur les photos.

Les données d'entrainement montrent une répartition homogène des positions de fissures sur les images, le modèle devrait donc, en théorie, ne pas être impacté par la vérification suivante. Cadrer une même fissure de manière à la positionner à différents endroits sur la photo permet de vérifier si le modèle obtient des résultats différents. Après plusieurs essais et positions différentes, les prédictions restent globalement inchangées (fig. 95), et on peut affirmer que les prédictions du modèle sont insensibles aux positions des pathologies sur l'image.

2.3.7.2. Rotation images

Bien qu'il soit peu probable que des photos soient prises à des angles aussi extrêmes en situation réelle, il a été décidé de simuler la prise d'images à 45° en modifiant les images grands angles des échantillons A, B et C. Cette rotation a été effectuée de manière à orienter les fissures initialement inclinées à 45° verticalement, et inversement.

L'opération a pour but de vérifier si les mauvais résultats concernant les fissures diagonales, principalement présentes dans l'échantillon C, sont uniquement la cause d'un manque de données d'entrainement pour ces typologies de fissures. Il est également intéressant de déterminer si la position des briques permet au modèle de reconnaitre les pathologies ou si leur orientation n'est pas directement liée aux performances du modèle.

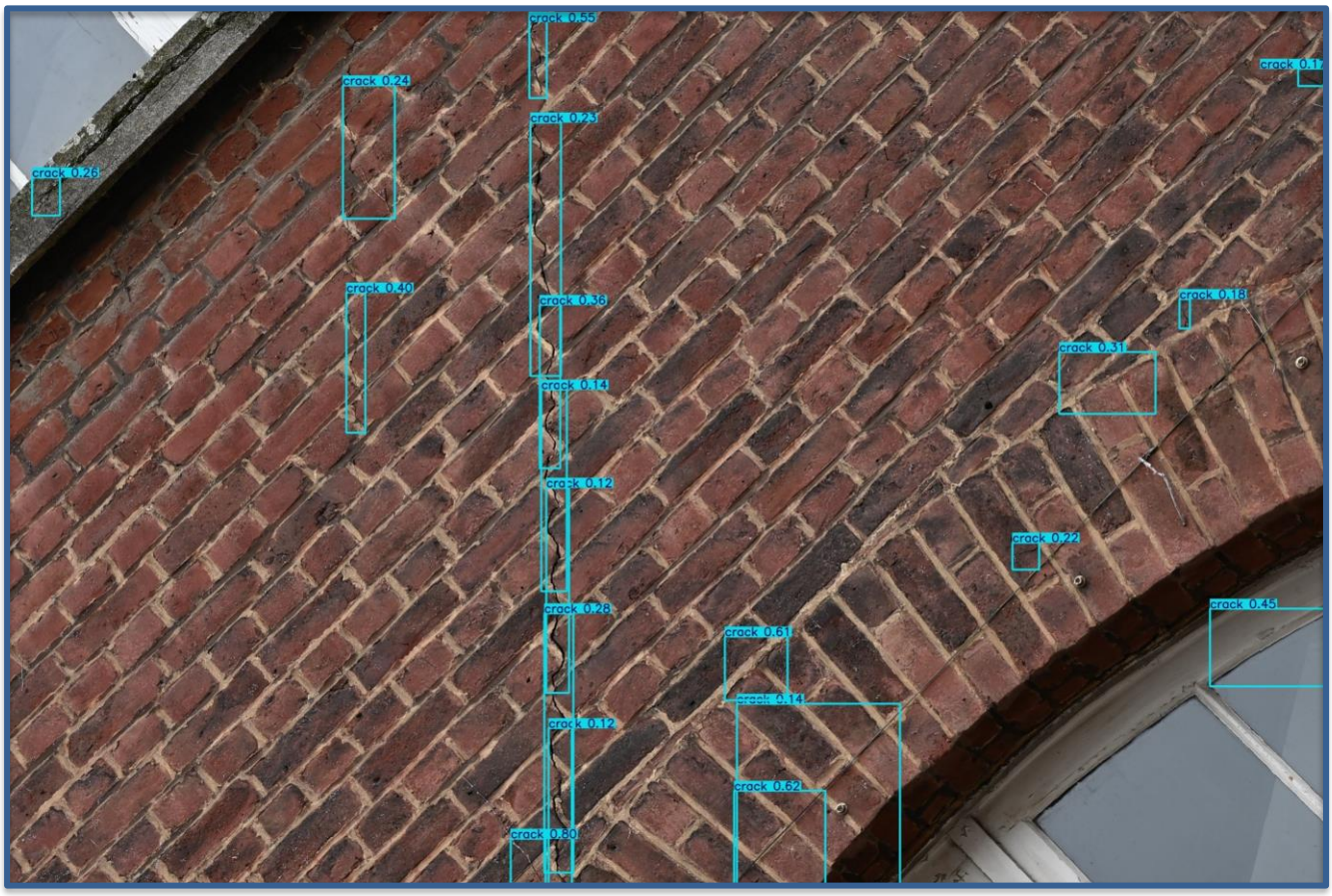


Fig. 96 – Détection sur l'échantillon C après une rotation de 45°

Detect : img 1664 - conf 0.1	*Distance : 7m*	*Distance focale : 70mm*

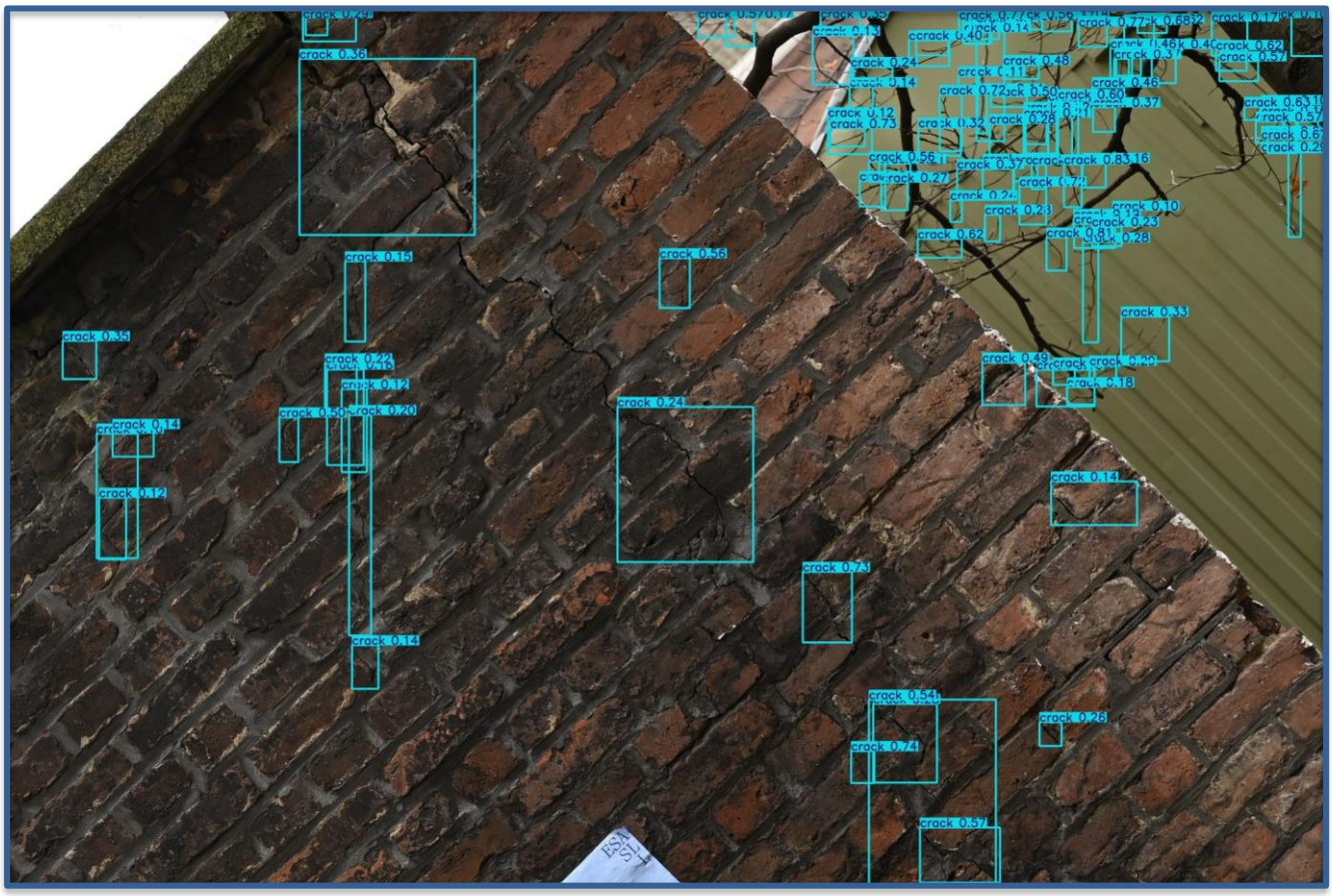


Fig. 97 – Détection sur l'échantillon B après une rotation de 45°

Detect : img 1664 - conf 0.1	*Distance : 7m*	*Distance focale : 70mm*

Les résultats des fig. 96 et 97 permettent d'affirmer que le modèle est parfaitement capable d'identifier l'ensemble des pathologies initialement présentées pour chaque échantillon. Néanmoins, toutes les typologies de fissures n'ont pas été représentées de façon équilibrée dans le dataset d'entrainement, engendrant de mauvaises prédictions pour celles qui n'ont pas été suffisamment observées par le modèle.

2.3.7.3. Contexte non-pertinent

Sur la façade nord du bâtiment de l'échantillon C, le mur est recouvert à mi auteur de tiges de lierre qui voilent en grande partie les fissures qu'on peut y observer. Le modèle est pourtant complètement capable de déceler la majorité des pathologies tout en ayant une quantité étonnamment faible de faux positifs. YOLOv5 discerne non seulement les tiges des fissures (fig. 98b), mais est également capable de correctement annoter les fissures recouvertes partiellement par ces tiges (fig. 98c).

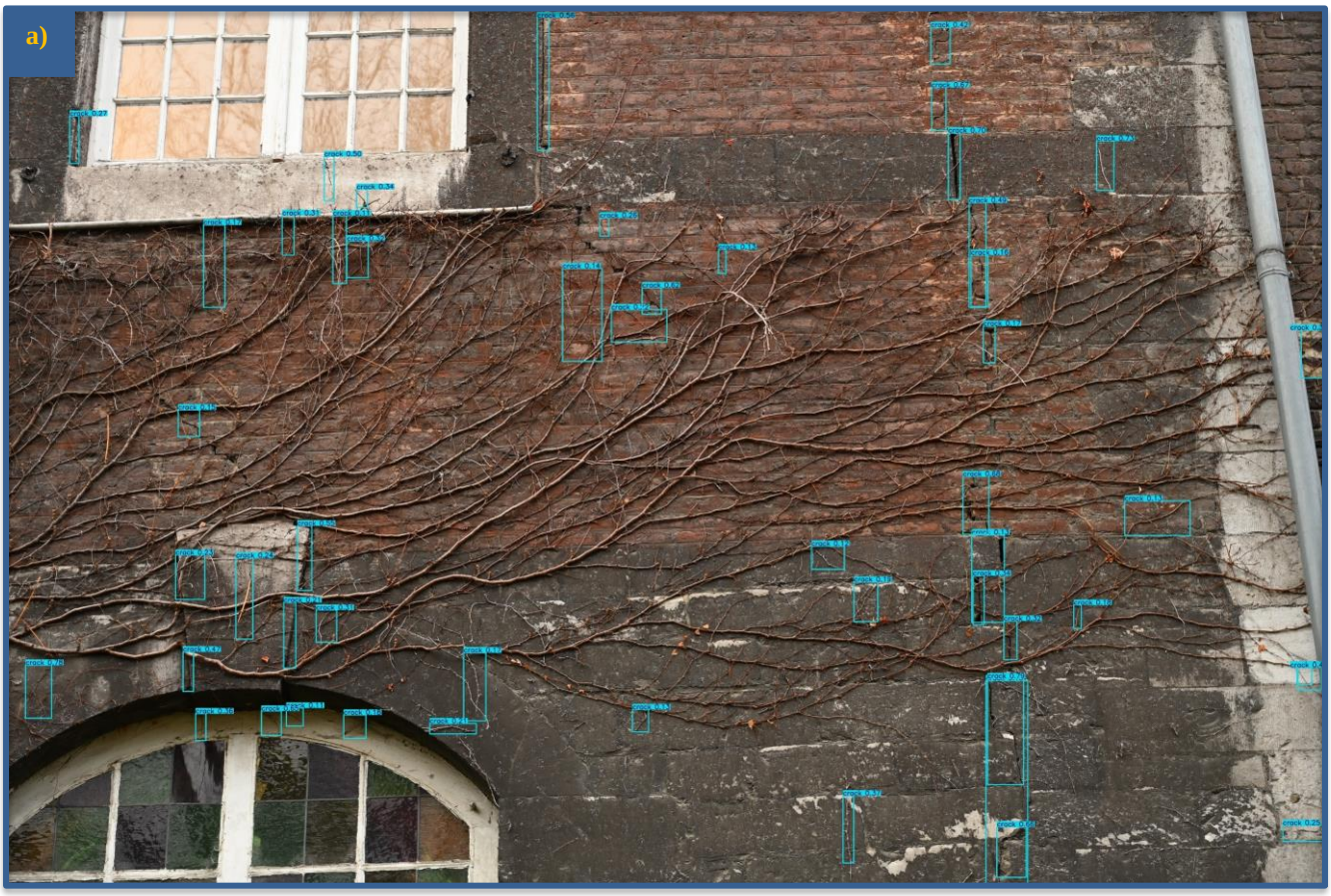
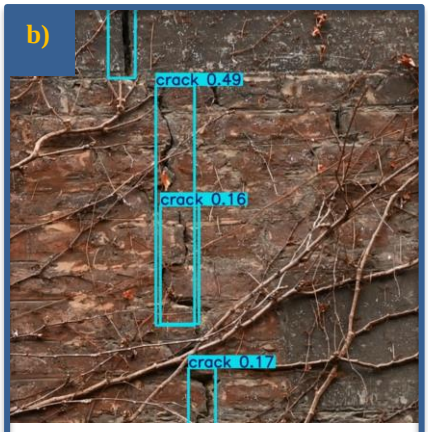
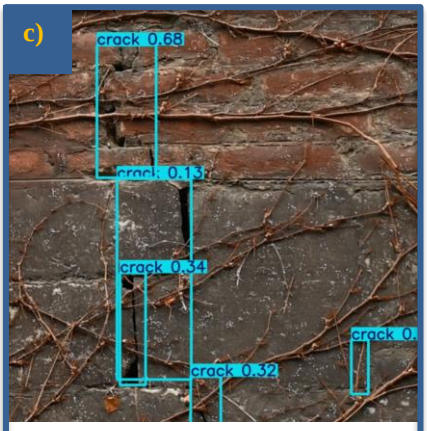


Fig. 98 – Détection sur un autre échantillon présentant de la végétation

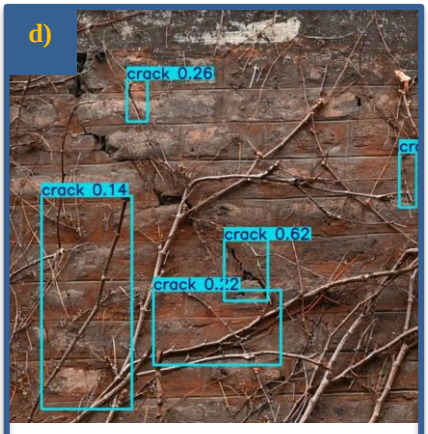
Detect : img 1664 - conf 0.1	*Distance : 7m*	*Distance focale : 70mm*



Détection des fissures tout autour du contexte non-pertinent.



Prédictions satisfaisantes pour les fissures verticales, avec une confiance allant jusqu'à 0.68 malgré le lierre qui traverse les prédictions.



Plusieurs faux positifs autour d'une fissure diagonale, mais le niveau de confiance le plus élevé est pour une partie de la fissure.

3. Discussion

Pour aller plus loin que les observations de Karimi et al. (2024), YOLOv5 a été employé dans une variété de paramètres et de situations réelles. Ces modifications de paramètres ont avant tout permis de comprendre le fonctionnement de chacune des variables lors de l'entrainement et de la détection d'objets, mais surtout leur impact sur la qualité des prédictions. Le déploiement en situation réelle du modèle a vu se confirmer les mêmes observations que dans l'article d'origine, mais a également permis d'explorer le potentiel réel du Computer Vision dans le domaine du patrimoine.

Si la mise en place de YOLOv5 sur un dataset précis semble à première vue complexe, l'utilisation d'environnements d'exécution sur le cloud, comme Google colab, augmente énormément l'accessibilité des modèles de Deep Learning. En effet, la puissance computationnelle et les logiciels normalement nécessaires sur un ordinateur local sont tous pris en charge par des serveurs qui permettent à tout utilisateur d'exécuter le modèle rapidement et sans se soucier des prérequis d'un environnement local.

De plus, l'entrainement du modèle pourrait être complétement négligé si le fichier contenant les poids était mis à disposition par les auteurs ou un autre opérateur, et YOLOv5 pourrait donc lancer une prédiction en quelques clics. Le seul paramètre demandant alors d'être ajusté pour obtenir des résultats satisfaisants est celui de la résolution des images passant dans le modèle, si le GSD de ces mêmes images à détecter varie fortement des données d'entrainement.

Ensuite, malgré un dataset augmenté de seulement 861 images de maçonneries au Portugal, YOLOv5 est capable de prédire efficacement les fissures sur des couleurs et types de briques très variés du dataset, ici en Belgique. Les images à angle d'incidence élevé, à luminosité faible, contenant du contexte non-pertinent ou étant moins bien cadrées sur les pathologies ont toutes obtenu des résultats très satisfaisants pour les échantillons A et B. Les résultats sont en revanche beaucoup plus décevants pour l'échantillon C, mais la rotation des images au point 2.3.7 a prouvé que ces mauvaises performances étaient majoritairement liées à l'entrainement, qui n'avait pas suffisamment mis en avant les fissures en escalier. L'échantillon C permettait principalement de tester l'adaptabilité du modèle sur des typologies de fissures qu'il avait très peu rencontrées lors de son entrainement, les fissures horizontales et en escalier.

Bien que les performances du modèle soient encourageantes dans de nombreuses situations, il est conseillé de reprendre des directives similaires à un processus de photogrammétrie lors de l'acquisition

des images pour obtenir les meilleurs résultats. Néanmoins, les différentes expériences ont permis d'affirmer que si le contexte autour du bâtiment n'offre pas de conditions idéales pour l'acquisition des images, des angles de vue différents suffisent généralement à l'obtention de prédictions correctes.

Concernant la luminosité et l'ombrage, il n'a pas été possible d'étudier ni l'impact d'une façade surexposée en plein été, ni l'impact d'une ombre portée sur les prédictions du modèle. Les environnements peu éclairés, notamment sous les toitures ou dans des caves, demandent quant à eux plus d'approfondissement, car il serait intéressant d'observer les prédictions selon différents angles d'éclairage. En effet, il semble important de confirmer si les prédictions étaient uniquement affectées par le suréclairage provoqué par le flash directement dans les fissures ou si la lumière artificielle a également une influence quelconque.

Enfin, la capacité du modèle à détecter des pathologies en fonction de la distance de l'appareil et de la résolution de l'image permet de réellement envisager le potentiel de la technologie. Il n'est aucunement intéressant pour un expert ou un architecte de détecter une pathologie qu'il a dû identifier manuellement et pour laquelle il a dû se rapprocher afin d'en prendre une photo. Il ne faut pas oublier que ces résultats sont obtenus par le modèle YOLO, qui est le modèle largement utilisé le plus rapide, mais surtout le moins précis en détection d'objets, et que la détection d'objets reste une tâche de Computer Vision relativement basique. De plus, les fissures sur la maçonnerie ont justement été choisies comme cas d'étude pour leur complexité, que ce soit pour les typologies et la taille des fissures ou les arrière-plans complexes et variés des façades de brique.

En conclusion, le Deep Learning obtient les résultats qu'il promet pour la détection de pathologies, mais son utilisation à l'heure actuelle n'est malheureusement pas encore très transparente. Les résultats présentés ici sont le fruit de nombreux essais et surtout d'erreurs, certaines visiblement identifiables tandis que d'autres plus discrètes, voire invisibles. Une erreur dans le code du modèle indique clairement un problème qui est facile à identifier et donc à régler. Cependant, les erreurs invisibles, comme l'*overfitting*, permettent toujours au modèle de fonctionner en obtenant des résultats médiocres. Avant de répondre à la question de la pertinence de l'automatisation par IA de la détection de pathologies visuelles du patrimoine bâti, il est important de parcourir toutes les contraintes liées au fonctionnement d'un modèle de Deep Learning, de l'environnement d'exécution aux difficultés rencontrées lors de l'entrainement, en passant par la manière de se procurer un tel modèle.

4. Remarques et contraintes

4.1. Comparaison environnements d'exécution

Comme brièvement abordé lors du point 2.2.1, il existe deux plateformes d'environnements d'exécution. L'environnement local utilise la puissance de calcul de l'ordinateur sur lequel on exécute le modèle, tandis que le cloud permet de se connecter à des serveurs faisant les calculs « en ligne ». Puisque l'approche par cloud a été choisie par les auteurs de l'article d'origine, le déploiement du modèle dans ce travail s'est également fait de cette manière. Cependant, les deux méthodes ont été expérimentées.

Dans ce chapitre, je compare les deux types de plateformes en m'appuyant sur mon expérience au cours de ce travail. Cette analyse se concentre sur les avantages et les inconvénients de chaque plateforme mais reflète uniquement mes observations en tant qu'utilisateur amateur pendant une période relativement courte, laissant la possibilité à d'autres perspectives ou alternatives d'exister. Google colab servira donc d'exemple pour le cloud, mais il existe d'autres services similaires comme Kaggle ou Jupyter par exemple. J'ai recensé plusieurs catégories de comparaison, comme les prérequis nécessaires pour préparer le modèle, le stockage des données, l'inter compatibilité des deux environnements et l'accessibilité du modèle. Cette accessibilité est non seulement computationnelle, mais également financière et pédagogique.

4.1.1. Prérequis

CLOUD	LOCAL
L'utilisation de Google colab ne demande que peu de prérequis, si ce n'est l'obtention du code ou la création d'un compte google pour avoir accès à google drive et aux fonctionnalités du service. Les logiciels nécessaires à l'utilisation du modèle sont compris dans le service.	L'exécution de commandes de code sur un environnement local passe avant tout par la console de commande, qui permet donc de naviguer dans les fichiers de l'ordinateur tout en exécutant des lignes de code. Cependant, ces lignes de code peuvent être écrites dans différents langages de programmation. Dans ce cas-ci, python est utilisé, et il est donc nécessaire de télécharger le logiciel pour exécuter le code qui y est lié, sans quoi l'ordinateur ne reconnait pas le langage. Ce logiciel reçoit des mises à jour constantes qui peuvent modifier le langage et donc rendre des commandes anciennes obsolètes, si les versions sont incompatibles.

Une fois la version compatible de Python installée dans l'environnement local, il est toujours nécessaire d'installer des packages spécifiques pour répondre aux besoins du modèle. L'outil pip, qui doit également être téléchargé en amont, est utilisé pour installer ces packages. Ils sont essentiels pour exécuter les modèles, mais permettent aussi de les entraîner en exploitant les capacités du GPU plutôt que celles du CPU, ce qui améliore considérablement les performances. Dans la plupart des cas, le GPU est plus adapté à l'entrainement du modèle, mais l'ordinateur ne l'utilise pas sans des outils comme CUDA et PyTorch, qui jouent un rôle crucial dans l'efficacité de l'entrainement des modèles. Chaque projet peut nécessiter une configuration différente de packages. Le fichier nommé *requirements.txt* liste les dépendances nécessaires au fonctionnement du modèle. Ce fichier constitue une référence importante pour comprendre les outils et bibliothèques utilisés par le code.

Cependant, puisque l'environnement local conserve tous les fichiers téléchargés, utiliser un quelconque autre modèle ayant des dépendances différentes du premier, ou simplement nécessitant des versions différentes des packages, peut introduire des problèmes de compatibilité. L'ordinateur va parfois réutiliser les packages déjà téléchargés et créer des erreurs dans le code. Ces erreurs impliquent de manuellement désinstaller tous ces packages et recommencer jusqu'à ce qu'ils fonctionnent entre eux.

Pour éviter cela, il faut utiliser un gestionnaire de packages, tel que *venv* ou *conda*, pour créer un environnement virtuel isolé. De manière similaire à l'environnement d'exécution sur le cloud, on crée un espace de travail coupé du reste de l'ordinateur dans lequel on peut finalement installer les packages et exécuter le modèle. Il conserve les fichiers mais doit être réactivé après fermeture de la console.

Pour résumer, il est obligatoire de télécharger Python et pip (ou d'autres logiciels similaires) si on veut utiliser un environnement local pour la première fois. En plus de **se fournir le code**, **ouvrir la console de commande et préparer le dataset**, il faut :

1. **Installer Python (Download Python | Python.org) et pip**
2. **Installer CUDA et PyTorch (Start Locally | PyTorch)**
3. **Vérifier que PyTorch détecte correctement le GPU**
4. **Créer un environnement virtuel** (commande : *python -m venv venv*)
5. **Activer l'environnement virtuel** (commande : *venv\Scripts\activate*)
6. **Cloner le modèle YOLOv5**
7. **Installer les dépendances**
8. **Télécharger les poids prés entrainés**

6e étape de la préparation du modèle sur Google colab

4.1.2. Stockage des données

CLOUD	LOCAL
La totalité des données utilisées par le modèle doit être accessible sur Google drive. Il est donc obligatoire d'avoir un compte google et suffisamment d'espace de stockage sur celui-ci. Par défaut, 15 GB sont accessibles gratuitement, mais j'ai plusieurs fois dû supprimer mes données pour laisser place aux datasets, aux images à détecter et aux prédictions du modèle durant mon étude. Cependant, l'environnement disparait avec ses données une fois qu'il s'éteint. De ce fait, on peut uniquement extraire les images pertinentes lors de l'utilisation du modèle et ne pas remplir l'espace de stockage inutilement. Les packages et autres données nécessaires au fonctionnement du modèle disparaissent également. L'utilisation de modèles différents n'a donc aucun impact sur l'espace de stockage utilisé, si ce n'est le fichier .ipynb contenant le code.	Le modèle, les datasets, les images à prédire, les prédictions et les logiciels ou packages nécessaires à l'utilisation du même modèle sont tous conservés sur l'ordinateur de l'utilisateur. La quantité de stockage disponible sur celui-ci dépend de l'appareil, mais les données peuvent rapidement devenir volumineuses puisque tout est conservé localement. Il faut donc nettoyer manuellement les données non-pertinentes et les résultats sont uniquement conservés sur une machine physique. Dans mon cas, je compte actuellement 30GB d'espace de stockage alloué au modèle et aux images, avec 6GB uniquement réservés à python et aux autres packages nécessaires pour le fonctionnement du modèle. L'utilisation de différents modèles requiert différents packages et versions de ceux-ci, qui seront tous conservés sur l'ordinateur. Ces fichiers sont non seulement volumineux, mais peuvent entrainer des problèmes de compatibilités entre les modèles s'ils ne sont pas isolés.

En réalité, le stockage des données n'est pas un problème pour une utilisation occasionnelle du modèle, mais il peut rapidement devenir une contrainte s'il n'est pas considéré correctement. Néanmoins, un disque dur externe pour l'environnement local et une extension de l'abonnement Google drive permettent d'étendre largement la capacité de stockage si nécessaire.

4.1.3. Compatibilité entre les environnements

En théorie, les poids générés lors de l'entrainement d'un modèle sont identiques, qu'ils soient issus d'un entrainement local ou via le cloud. Pourtant, il m'a été impossible d'utiliser les poids issus de Google colab sur mon ordinateur car YOLOv5 cherchait des éléments propres au cloud qui n'existaient donc pas sur mon fichier. Je n'ai donc pas été capable d'utiliser les mêmes poids entre les deux environnements sans modifier le code. De plus, je peux confirmer que les versions de YOLO ne sont pas compatibles entre elles, car les poids issus de l'entrainement de YOLOv7 n'ont jamais fonctionné pour YOLOv5. Un passage à une version supérieure d'un modèle induit nécessairement un réentrainement de celui-ci pour tous les datasets.

4.1.4. Accessibilité

Les trois catégories d'accessibilité sont étroitement liées, puisque chacune impacte directement les autres. Je considère l'accessibilité computationnelle comme étant la capacité d'utiliser YOLOv5 avec le moins de puissance de calcul possible. Si l'utilisation de celui-ci nécessite des ordinateurs de plusieurs milliers d'euros, alors YOLOv5 est peu accessible. Enfin, l'accessibilité pédagogique reprend la facilité avec laquelle on peut prendre en main le modèle ou être formé à son utilisation, tandis que l'accessibilité financière prend en compte tous les coûts associés à l'utilisation des deux environnements d'exécution.

4.1.4.1. Puissance de calcul

CLOUD	LOCAL
Peu importe l'ordinateur sur lequel est exécuté le modèle, la version gratuite de Google colab offre jusqu'à 12.7GB de CPU et 15GB de GPU. Cependant, cette offre est limitée et il m'a été impossible d'entrainer le modèle sur plus de 50 époques avant que l'environnement ne s'éteigne de lui-même. La version pro de Google colab donne l'accès à plusieurs types d'exécution différents, allant jusqu'à plus de 50GB de CPU et 23GB de GPU. Cette mémoire allouée est exclusivement réservée à l'exécution du code et permet donc de concentrer toutes les ressources sur le modèle. L'entrainement de 100 époques avec un *batch size* de 16 a pris 21m 12s, ce qui correspond à **13s** pour une seule époque. Un entrainement d'une seule époque avec un *batch size* de 2 a pris **40s.**	Les ressources disponibles en environnement local se limitent au matériel de l'appareil. Dans mon cas, j'ai utilisé un ordinateur portable MSI GL63 8SE, avec une carte graphique (GPU) NVIDIA GeForce RTX 2060 de 6GB de mémoire, un processeur (CPU) i7-8750H et 16GB de RAM. Celui-ci a plus de 5 ans et n'est pas capable de réserver toutes ses ressources au modèle puisque les autres éléments de l'ordinateur en ont également besoin. L'ordinateur n'a pas su compléter un entrainement du dataset dans des délais acceptables. Pour une seule époque avec un *batch size* de 2, il a fallu **7m 16s**. Le *batch size* devait être limité sans quoi l'exécution du modèle gelait complétement l'ordinateur et nécessitait un redémarrage de celui-ci.

`1 epochs completed in 0.011 hours.`
`Optimizer stripped from runs/train/exp2/weights/last.pt, 14.3MB`
`Optimizer stripped from runs/train/exp2/weights/best.pt, 14.3MB`
	`1 epochs completed in 0.121 hours.`
	`Optimizer stripped from runs\train\exp15\weights\last.pt, 14.3MB`
	`Optimizer stripped from runs\train\exp15\weights\best.pt, 14.3MB`

Fig. 99 – Résultat d'un entrainement sur le cloud d'une époque	Fig. 100 – Résultat d'un entrainement local d'une époque

4.1.4.2. Prise en main

La plus grosse différence entre les deux méthodes d'exécution est sans aucun doute la complexité de prise en main du modèle. Comme expliqué pour les prérequis, l'environnement local demande beaucoup plus de préparation en amont et n'est pas nécessairement très intuitif pour un utilisateur sans formation puisqu'il requiert l'utilisation de code, et indirectement l'apprentissage de ce langage de code. Les logiciels à installer ont eux-mêmes de nombreuses configurations qui ne fonctionneront pas avec le modèle s'ils sont mal configurés.

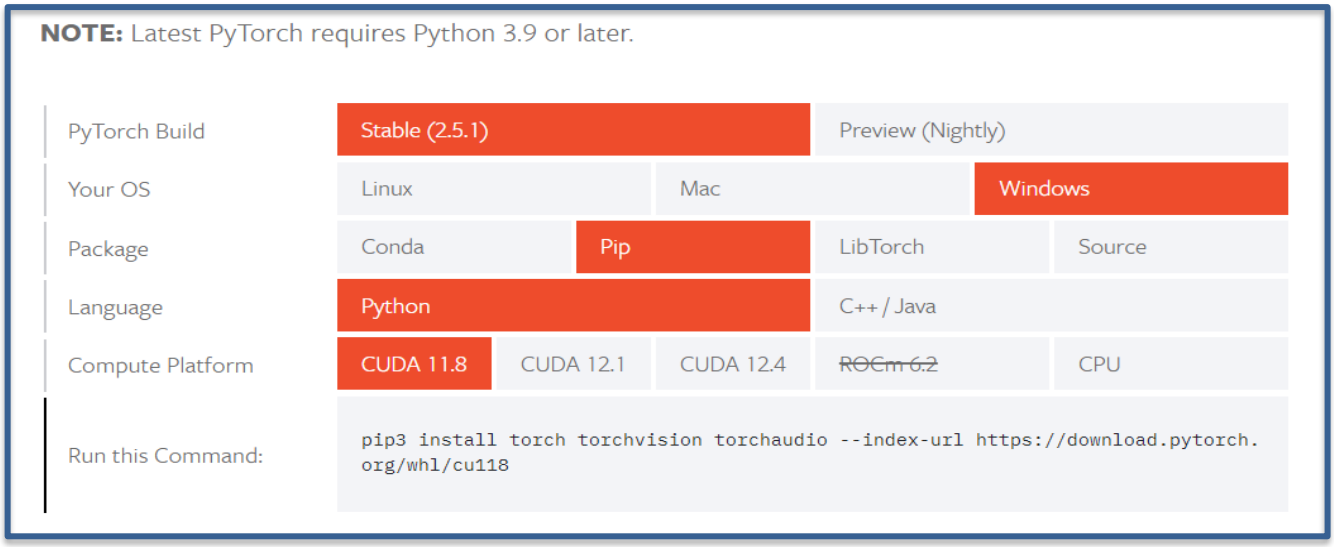


Fig. 101 – Options de téléchargement de Pytorch

Par exemple, Pytorch dépend pour fonctionner du système d'exploitation de l'ordinateur, du gestionnaire de packages utilisé et du langage de programmation (fig. 101). Evidemment, ces opérations restent simples si considérées seules mais le risque d'une erreur d'inattention augmente avec chaque étape ou paramètre supplémentaire, tandis que l'environnement d'exécution sur le cloud ne requiert rien de cela.

De plus, l'interface par défaut de la console de commande (local) est très peu lisible et difficile à naviguer. Il est donc conseillé de télécharger des logiciels supplémentaires, comme Microsoft Visual Studio Code (Visual Studio Code - Code Editing. Redefined), qui offrent une interface similaire à celle qu'on peut trouver sur Google colab.

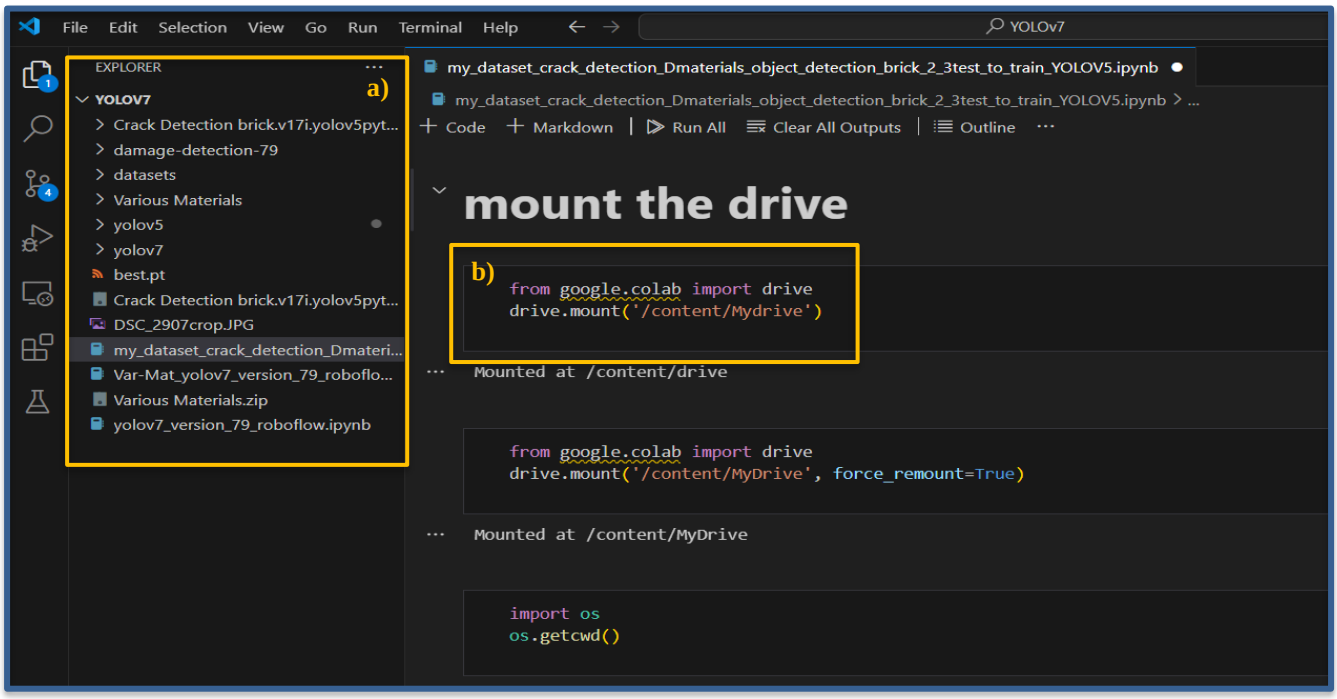


Fig. 102 – Interface de Visual Studio Code : a) barre de navigation, b) cellule de commande

4.1.4.3. Coût

CLOUD	LOCAL
La version gratuite de Google colab ne permet pas d'entrainer suffisamment YOLOv5, c'est pourquoi j'ai utilisé colab Pro pour cette étude. Pour 11,19€, j'ai eu accès à 100 unités de calcul. J'ai consommé au total 96 de ces unités à travers l'entièreté de mes entrainements, prédictions et temps passé avec l'environnement d'exécution allumé. Ce forfait ne m'a en réalité été indispensable que pour les entrainements dépassant 50 époques, car les autres actions auraient pu se faire avec la version gratuite.	Le coût de l'utilisation du modèle se limite exclusivement au prix de l'ordinateur et aux composants nécessaires pour l'exécuter. Le MSI GL63 8SE coûtait près de 1400€ lors de son achat en 2019. Bien qu'un ordinateur portable ne soit pas l'outil le plus approprié pour des opérations de ce genre, ses composants ne suffisent aujourd'hui pas pour entrainer efficacement YOLOv5 sur un dataset de 861 images.
L'espace de stockage n'a finalement pas été un problème, mais il est possible d'étendre la capacité de stockage de google drive de 100GB pour 1,99€ par mois.	L'espace de stockage de l'ordinateur a suffi pour l'entièreté de l'étude, et aucun frais supplémentaire n'a été nécessaire.
L'ordinateur utilisé peut être d'entrée de gamme et obtenir des résultats très satisfaisants puisque les calculs ne se font pas sur l'appareil.	L'ordinateur utilisé doit être spécifiquement construit pour avoir des CPU et GPU puissants ainsi que suffisamment de RAM et d'espace de stockage.

Financièrement, l'utilisation du cloud est selon moi l'alternative la plus adaptée à une utilisation occasionnelle de modèles de Deep Learning. Si on réalise beaucoup d'entrainement, c'est définitivement le plus avantageux car c'est l'opération qui nécessite le plus de puissance de calcul. Si les poids sont disponibles, les deux options se valent car les prédictions demandent moins de puissance et le choix dépend plutôt des capacités de l'ordinateur. Le cloud demande cependant de régulièrement couper l'environnement s'il n'est pas utilisé, tandis que l'environnement local est disponible en permanence.

De manière générale, un environnement d'exécution comme Google colab demande peu de préparation, limite le stockage excessif de données sur l'ordinateur de l'opérateur, offre une meilleure puissance de calcul que de nombreux appareils, n'impacte pas les autres actions sur celui-ci, et est généralement plus facile à utiliser qu'un environnement local. Cela nécessite uniquement une dépendance aux services de Google et aux coûts associés à leur utilisation. Ces prix restent cependant raisonnables à l'heure actuelle et permettent une flexibilité dans l'utilisation du service. Les différents paliers permettent d'atteindre des performances élevées selon les besoins puis de revenir à une utilisation presque gratuite du service lorsqu'elle suffit. Il est également possible que d'autres services similaires aient des offres plus adaptées.

4.2. Difficultés rencontrées

Ce chapitre retrace de manière synthétique mon parcours pour faire fonctionner un modèle de Deep Learning. Il a pour objectif d'illustrer les défis majeurs liés à ma démarche, qu'il s'agisse de l'acquisition du modèle, de la configuration des environnements d'exécution, de la gestion des dépendances ou de l'intégration des données.

Mon parcours a débuté avec des recherches visant à trouver un modèle de Deep Learning de détection d'objets disponible en open-source. Il devait déjà être entrainé sur un dataset de fissures, préférablement sur des façades de maçonnerie, et lié à une publication scientifique avec laquelle il serait possible de comparer mes résultats. Les différents environnements d'exécution n'étaient pas considérés, car je ne connaissais pas l'existence de plateformes comme Google colab. Cependant, j'ai rapidement réalisé que peu de modèles correspondaient à ces critères. Premièrement, aucun modèle ni publication ne partage les poids de l'entrainement de son modèle, ce qui signifie qu'il était nécessaire d'entrainer un modèle moi-même. Deuxièmement, le code associé aux articles scientifiques, lorsqu'il m'était fourni par les auteurs, n'était pas entretenu ou destiné à être réutilisé. En effet, de nombreux codes partagés dans le cadre de publications académiques sur la détection de pathologies sont avant tout des preuves de concept, plutôt que des outils prêts à l'emploi et mis à jour comme tels. Des fichiers essentiels comme *requirements.txt* manquaient, ce qui m'a poussé à abandonner ces modèles après plusieurs tentatives de trouver manuellement les dépendances de chaque package.

Sans initialement le savoir, l'installation des dépendances de ces différents modèles créerait des problèmes de compatibilité avec le code de YOLOv5 fourni par Karimi et al. (2024) en environnement local. Utiliser la version gratuite de Google colab ne fonctionnait pas non plus, étant donné que l'entrainement n'avait pas le temps de se finaliser avant que l'environnement ne s'éteigne. Après avoir contourné les problèmes de versions des packages avec la création d'environnements virtuels isolés (*venv*) pour les y installer, il était enfin temps d'entrainer le modèle sur les 861 images du dataset.

Les premières tentatives se soldèrent par une perte totale de réponse de l'ordinateur, nécessitant à chaque essai un redémarrage manuel de celui-ci. Après avoir fermé tous les autres programmes et diminué le *batch size* à 2, il a été possible d'entrainer YOLOv5 sur une seule époque en **1h 16m 22s**. Sachant que YOLOv5 est connu pour sa vitesse d'entrainement et que les études analysées plus tôt dans le travail ne montraient pas de délais aussi importants pour des tailles de datasets similaires, il était évident qu'une erreur s'était produite. Pour référence, l'entrainement aux mêmes paramètres de YOLOv5 sur Google colab a pris **40s**.

J'ai ensuite compris que ma version de CUDA, qui permet au modèle d'utiliser le GPU au lieu du CPU pour son entrainement, n'était pas correctement installée. Il a fallu régler le problème pour finalement obtenir un entrainement suivant des paramètres identiques de **7m 16s**.

A ce rythme, entrainer YOLOv5 sur 100 époques prendrait aux alentours de **12h,** tandis que 500 époques prendraient plus de **60h.** Pour surmonter les limitations de puissance de calcul de mon environnement local, j'ai donc finalement décidé d'acheter des unités de calcul sur Google colab pour pouvoir accélérer mon entrainement. Les GPU plus rapides de colab Pro ont permis de raccourcir drastiquement l'entrainement et le réaliser en moins de 30 minutes pour 100 époques. Les premières prédictions ont rapidement été réalisées sur les images de validation du dataset et des échantillons de l'étude. Les résultats étaient peu concluants, et le modèle rencontrait même des difficultés à reconnaitre toutes les fissures de validation malgré un mAP supérieur à 90%.

Après différentes tentatives d'entrainement, j'ai remarqué dans les *logs* (historique des évènements du code) que sur les 861 images du dataset, le modèle n'en considérait que 404 comme exemples de fissures, tandis que le reste était considéré comme arrière-plan. Une vérification manuelle de l'entièreté du dataset a confirmé qu'une ou plusieurs instances de fissures étaient visibles sur chaque image. Une mauvaise manipulation des fichiers lors de la décompression du dossier ou de son importation sur Google Drive avait empêché l'action de s'achever correctement. De ce fait, 457 fichiers indiquant les coordonnées des annotations sur les images d'entrainement étaient manquants.

a) `404 images, 457 backgrounds, 0 corrupt: 100% 861/861 [00:00<?, ?it/s]`

b) `861 images, 0 backgrounds, 0 corrupt: 100% 861/861 [00:00<?, ?it/s]`

Fig. 103 – Extrait des logs de YOLOv5 après l'entrainement du modèle pour : a) un dataset incomplet, b) un dataset complet

Après avoir rétabli le dataset complet, il a fallu comparer les performances du modèle avec les poids de 100 et 500 époques, pour comprendre l'impact de l'*overfitting* sur la qualité des prédictions de celui-ci, et enfin analyser les échantillons dans l'enceinte de la faculté avec les poids pertinents.

Finalement, les difficultés liées à l'obtention du code sont surtout causées par la volonté d'utiliser un code et des paramètres déjà vérifiés par la communauté scientifique, car il est parfaitement possible de trouver le code de modèles comme YOLOv5 et divers datasets de fissures de maçonnerie en open source, séparément. Les autres difficultés ont été causées, et difficilement identifiées, par mon manque de connaissances en matière de programmation et utilisation de ces modèles de Deep Learning.

V. CONCLUSION

1. Réponse à la question de recherche

L'objectif de ce travail était d'analyser l'intégration des nouvelles technologies et de l'intelligence artificielle dans la conservation du patrimoine bâti pour détecter la présence de pathologies.

En premier lieu, l'apparition et l'amélioration des technologies de relevés indirects comme la photogrammétrie et la lasergrammétrie ont grandement amélioré la précision et l'efficacité des inspections du patrimoine bâti en réduisant le temps nécessaire sur le terrain et en améliorant la qualité des données recueillies lors d'un relevé. Les deux technologies sont complémentaires et sont aujourd'hui souvent utilisées en parallèle. La photogrammétrie offre des informations détaillées sur la texture et la couleur de l'objet, tandis que la lasergrammétrie atteint une précision élevée indépendante des conditions lumineuses lors de l'acquisition des données. Cependant, malgré une amélioration constante de ces techniques de relevés, l'inspection visuelle d'un bâtiment existant, qui est souvent la première étape d'une étude plus complexe de celui-ci, reste un processus manuel. Celui-ci est très long et couteux, dépendant de l'expertise des opérateurs responsables, et peut être inconsistant sur le long terme selon les formations de ces mêmes opérateurs.

Pour accélérer ce processus, l'approche par Computer Vision vise aujourd'hui à automatiser les inspections visuelles du patrimoine et la détection de pathologies. Elle promet d'être plus performante qu'une inspection réalisée exclusivement par un expert, d'une part en étant une méthode purement objective et constante entre les inspections, d'autre part en permettant également une plus grande fiabilité sur le long terme, là où des opérateurs différents en sont incapables. Elle permet également de numériser les résultats de ces inspections pour un meilleur contrôle de l'évolution des bâtiments au cours du temps. Perez et Tah (2021) mentionnent que l'inspection automatisée des pathologies ne vise pas à remplacer l'expert, mais plutôt à l'assister en temps réel lorsqu'il est sur terrain, ou à accélérer le processus d'identification des pathologies ex-situ, à condition de la combiner avec les méthodes de relevés indirects.

Afin de confirmer que la détection automatique de pathologies a un niveau de précision et d'exactitude suffisant pour considérer son intégration dans certains travaux d'inspection visuelle, il a été décidé d'analyser 28 études en considérant la performance des modèles de Deep Learning utilisés, les tâches réalisées, les approches intéressantes, la complémentarité avec les méthodes de relevé indirects ainsi que les limitations des modèles et de la technologie en général.

Plusieurs travaux, comme ceux de Cha et al. (2018) et Jiang et Zhang (2020), montrent des niveaux de précision élevés pour la détection de différentes pathologies. Dung et Anh (2019) affirment que les modèles permettent de minimiser la nécessité d'une intervention humaine grâce à leur capacité de traiter rapidement de grandes quantités de données. Cependant, des études comme celles de Hatır et al. (2021) et Mishra et al. (2022a) révèlent également que l'ensemble des modèles ont des limitations face à la complexité des matériaux et de leur contexte. Les motifs irréguliers ou les conditions d'éclairage complexes peuvent par exemple réduire la précision des prédictions quand le modèle n'est pas entrainé pour ces conditions.

De manière générale, les études analysées expriment des performances très prometteuses pour l'intégration de la technologie en situation réelle. Néanmoins, sa véritable applicabilité sur le terrain se mesure également selon des critères comme le temps d'application, le coût, l'impact des limitations sur les prédictions, l'adaptabilité à d'autres datasets, et bien sûr, la pertinence des informations générées pour l'opérateur.

Pour répondre à ces critères, YOLOv5 a été mis en œuvre selon les paramètres et datasets présentés dans le travail de Karimi et al. (2024). Ce dernier traite les pathologies de fissures sur plusieurs matériaux représentés dans des datasets distincts. L'expérience en situation réelle n'a utilisé que les données d'entrainement de maçonnerie, et avait pour but de représenter les étapes d'utilisation d'un modèle de Deep Learning ainsi que de tester la robustesse de YOLOv5 face à des images nettement différentes de celles présentées lors de l'entrainement.

Les résultats obtenus démontrent que YOLOv5 est capable de détecter efficacement des fissures sur des surfaces de maçonnerie variées, même lorsque les images présentent des conditions d'acquisition imparfaites, comme une prise d'image inclinée ou avec un angle d'incidence élevé.

Cependant, il éprouve également des difficultés, notamment face à des typologies de fissures peu représentées dans les données d'entrainement (fissures horizontales et en escalier) ou lorsqu'une source de lumière artificielle diminue la visibilité des fissures sur l'image. Ceci reste négligeable car il suffit d'agrandir le dataset et éviter l'utilisation d'un flash. YOLOv5 réalise aussi de nombreux faux positifs sur des éléments non-pertinents comme les branches d'arbres et certains éléments de châssis verticaux. Ces erreurs peuvent cependant être écartées par l'opérateur lors de la visualisation des prédictions.

Les résultats dépendent fortement de la qualité et de la diversité des images utilisées pour l'entrainement des modèles, comme souligné par Dung et Anh (2019) et Marín-García et al. (2023). Un dataset correctement annoté, compris d'images prenant en compte diverses typologies de défauts et conditions environnementales, est essentiel pour garantir des prédictions fiables. De plus, entrainer le modèle avec des datasets comprenant différentes pathologies permettrait de limiter les faux positifs et améliorer la distinction de ces différentes pathologies entre elles.

Bien que les performances soient globalement très positives, la mise en œuvre d'un modèle de Deep Learning dans un environnement local reste malgré tout peu intuitive et requiert beaucoup de puissance de calcul, surtout lors de la phase d'entrainement. Les environnements d'exécution sur cloud, comme Google Colab, améliorent toutefois l'accessibilité de la technologie aux utilisateurs disposant de ressources limitées ou ayant peu de connaissances dans les langages de programmation, en éliminant les risques d'incompatibilités et d'autres erreurs principalement rencontrées sur un environnement local.

Par ailleurs, la mise à disposition de poids pré-entraînés par des opérateurs formés pourrait éliminer les risques d'erreurs « invisibles » liées à l'entrainement du modèle, comme l'overfitting ou les mauvaises manipulations du dataset.

En conclusion, je pense pouvoir affirmer que **l'automatisation par IA de la détection de pathologies visuelles du patrimoine bâti est pertinente aujourd'hui**, à condition de former les opérateurs sur les tâches, modèles, paramètres et contraintes de cette technologie. Les résultats obtenus par YOLOv5 montrent que le modèle est capable d'assister un expert dans des conditions d'acquisition de données très variées, permettant de traiter des centaines d'images en quelques secondes une fois l'entrainement réalisé. Les contraintes principalement rencontrées ne sont en réalité que liées à une mauvaise connaissance des outils, et un opérateur quelconque pourrait rapidement obtenir des résultats similaires à ceux présentés dans ce travail après une courte formation.

Il serait également intéressant de mettre en place des outils collaboratifs améliorant encore l'accessibilité des modèles en simplifiant leur interface, comme Lamas et al. (2021) ont pu faire de manière rudimentaire avec l'application MonuMAI. Cela permettrait de ne pas nécessiter l'apprentissage des langages de programmation et rendrait l'utilisation de la technologie plus fluide. Il serait également utile de permettre le partage des poids d'entrainement liés à un dataset pour ne pas nécessiter un réentrainement systématique des modèles utilisant les mêmes données. Mettre ces fichiers à la portée de tous rendrait la technologie plus accessible aux utilisateurs sans accès à la puissance de calcul nécessaire pour ces poids.

2. Limites de l’expérience et perspectives

Il est important de rappeler la nature restreinte de ce travail, car malgré les quatre tâches de Deep Learning, les nombreux modèles présentés dans l’état de l’art et les multiples pathologies et matériaux observables sur le patrimoine bâti, le déploiement en situation réelle de ce mémoire ne prend en compte que la détection d’objets à l’aide du modèle YOLOv5 pour des fissures sur les maçonneries de brique.

Il serait réducteur de ne considérer qu’un cas d’étude limité comme celui-ci pour l’ensemble des techniques de Deep Learning. En effet, l’architecture YOLO en détection d’objets ne se distingue pas pour sa précision mais plutôt pour sa rapidité et simplicité, alors que les résultats obtenus dans ce travail sont plus que satisfaisants. De plus, la détection d’objets reste une application relativement simple du Computer Vision, et les tâches de segmentation permettent notamment une meilleure représentation des pathologies en indiquant clairement le contour de celles-ci. Les modèles de segmentation sont également plus complexes, mais semblent être la clé d'une véritable automatisation de la détection des pathologies visuelles du patrimoine bâti, principalement grâce à leur traitement au niveau des pixels des images, ce qui facilite leur complémentarité avec les technologies de lasergrammétrie et de photogrammétrie.

Dans les études analysées au chapitre III, Liu et al. (2022) ont mis en évidence la création de modèles 3D par photogrammétrie tout en intégrant la segmentation sémantique pour localiser des pathologies sur les mêmes images. Pathak et al. (2021) ont eu une approche opposée, en extrayant des images de modèles 3D générés par une photogrammétrie pour y détecter des pathologies.

Pour terminer, il serait pertinent dans de futurs travaux, d’explorer les tâches de segmentation pour déterminer si leurs performances sont comparables à celles de YOLOv5. Il serait également utile d'étudier les contraintes liées à la complexité de ces modèles et à la puissance de calcul requise. Cela permettrait de déterminer si elles impactent fortement l’efficacité des modèles pour justifier leur utilisation systématique, par rapport à des tâches plus simples comme la détection d’objets. De plus, il serait important d'analyser l’intégration des méthodes de relevés indirects avec ces tâches de segmentation.

Concernant la détection d’objets, expérimenter avec le même modèle pourrait permettre d’observer plus en détail l’impact des variations de conditions lumineuses, que ce soit avec une surexposition du soleil en été, avec des ombres portées ou avec des lumières artificielles autres qu’un flash orienté perpendiculairement à l’objet. Il faudrait également analyser la robustesse de YOLOv5 avec un dataset plus large, prenant en compte plusieurs pathologies et matérialités de façades. Il faudrait non seulement y comparer les prédictions avec celles présentées dans ce travail, mais également les temps d’entrainement et les métriques, pour un dataset contenant des milliers, ou dizaines de milliers d’images.

VI. BIBLIOGRAPHIE

Abed, M. H., Al-Asfoor, M., & Hussain, Z. M. (2020). *Architectural heritage images classification using deep learning with CNN*. https://ro.ecu.edu.au/ecuworkspost2013/9176/

Addleson, L. (1993). *Les Défauts de la construction : prévention, diagnostic et remèdes des principales pathologies de la construction*. De Boek.

Alby, E. (2006). *Élaboration d'une méthodologie de relevé d'objets architecturaux. Contribution basée sur la combinaison de techniques d'acquisition* [Phdthesis, Université Henri Poincaré - Nancy I]. https://theses.hal.science/tel-00132784

Algoscale. (2021). *YOLO vs SSD : Which One is a Superior Algorithm | Algoscale*. Algoscale GPT
Auzas, P.-M. (1979). *Eugène Viollet-le-Duc, 1814-1879*. FeniXX, Caisse nationale des monuments historiques et des sites.

AWaP. (2024a). *Inventaire du patrimoine immobilier culturel*. Consulté le 26 mai 2024, à l'adresse https://lampspw.wallonie.be/dgo4/site_ipic/

AWaP. (2024b). Protection du Patrimoine. Consulté le 27 juillet 2024, à l'adresse https://agencewallonnedupatrimoine.be/protection-du-patrimoine/

AWaP. (2024c). Code wallon du Patrimoine. *20240601_CoPat_Textes_legislatifs_consolides.pdf (agencewallonnedupatrimoine.be)

Barba, S., Ferreyra, C., Cotella, V. A., di Filippo, A., & Amalfitano, S. (2021). A SLAM Integrated Approach for Digital Heritage Documentation. In M. Rauterberg (Éd.), *Culture and Computing. Interactive Cultural Heritage and Arts* (p. 27-39). Springer International Publishing. https://doi.org/10.1007/978-3-030-77411-0_3

Bergeon-Langle S. et Brunel G. (2014) La restauration des oeuvres d'art : Vade-mecum en quelques mots. éd. Hermann. p.214.

Bharati, P., Pramanik, A. (2020). Deep Learning Techniques—R-CNN to Mask R-CNN: A Survey. In: Das, A., Nayak, J., Naik, B., Pati, S., Pelusi, D. (eds) Computational Intelligence in Pattern Recognition. Advances in Intelligent Systems and Computing, vol 999. p657–668. Springer, Singapore. https://doi.org/10.1007/978-981-13-9042-5_56

Biens du patrimoine mondial de l'Unesco 2006-2017. (2023). Statista. Consulté le 30 juillet 2024, à l'adresse https://fr.statista.com/statistiques/577272/nombre-biens-patrimoine-mondiale-unesco/

Billen, R., Jonlet, B., Luczfalvy Jancsó, A., Neuville, R., Nys, G-A., Poux, F., Van Ruymbeke, M., Piavaux, M., Hallot, P., (2018). La transition numérique dans le domaine du patrimoine bâti : un retour d'expériences. *Bulletin de la commission royale des monuments, sites et fouilles - Tome 30, 119-148* 180906LR_CRMSFBulletin30_ssCP_TransNum.pdf (uliege.be)

Brigham, R. (2022). *Monument Monitor : Using citizen science to preserve heritage*. Monument Monitor: using citizen science to preserve heritage | Request PDF (researchgate.net)

Carrie, C., & Morrel, D. (1975). Salissures de façades. Editions Eyrolles. p. 9-62.

Cha, Y.-J., Choi, W., Suh, G., Mahmoudkhani, S., & Büyüköztürk, O. (2018). Autonomous Structural Visual Inspection Using Region-Based Deep Learning for Detecting Multiple Damage Types. *Computer-Aided Civil and Infrastructure Engineering*, *33*(9), 731-747. https://doi.org/10.1111/mice.12334

Chen, L.-C., Papandreou, G., Kokkinos, I., Murphy, K., & Yuille, A. L. (2017). *DeepLab : Semantic Image Segmentation with Deep Convolutional Nets, Atrous Convolution, and Fully Connected CRFs* (arXiv:1606.00915). arXiv. https://doi.org/10.48550/arXiv.1606.00915

CNN Explainer. (2020). https://poloclub.github.io/cnn-explainer/

Convolutional Neural Network: A Complete Guide. (2023). https://learnopencv.com/understanding-convolutional-neural-networks-cnn/

Croce, V. (2022). *Transfert et repérage d'annotations sémantiques pour le patrimoine architectural. Un système méthodologique combinant Intelligence Artificielle, H-BIM et plateformes d'annotations collaboratives reality-based.* [Phdthesis, HESAM Université ; Université de Florence. Clinique orthopédique]. https://pastel.hal.science/tel-03708142

De Bie, A., Dewilde, G., Draelants, R., Henderieckx, F., Provost, M., Venstermans, J., & Wagner, M. (1996). *Guide pratique : Défauts de construction*. Kluwer Editorial.

Deeplizard. (2017). *Convolutional Neural Networks (CNNs) explained* [Vidéo]. YouTube. https://www.youtube.com/watch?v=YRhxdVk_sIs

Delnoy, M. & Van Damme, N. (2021). Code wallon du Patrimoine (CoPat). In Code wallon du Patrimoine (CoPat). Anthemis. Section 2 Delnoy M. et - Le Code wallon du patrimoine - CUP 2021.pdf (uliege.be)

Detry, N. (2016) Le patrimoine martyr et la restauration post bellica : théories et pratiques de la restauration des monuments historiques en Europe pendant et après la Seconde Guerre mondiale. [Thèse de doctorat, Université de Lyon]. Le patrimoine martyr et la restauration post bellica : théories et pratiques de la restauration des monuments historiques en Europe pendant et après la Seconde Guerre mondiale - TEL - Thèses en ligne (hal.science)

Dini, S. F., Wibowo, E. P., Iqbal, M., Bahar, Y. N., & Alfiandy, A. (2023). Applying Deep Learning and Convolutional Neural Network System to Identity Historic Buildings : The « Little China » Building in Central Java, Indonesia. Applying-Deep-Learning-and-Convolutional-Neural-Network-System-to-Identity-Historic-Buildings-The-Little-China-Building-in-Central-Java-Indonesia.pdf (researchgate.net)

Dolff-Bonekämper, G. (2020). Valeurs de contemporanéité. Pour une rénovation de la théorie des monuments d'Aloïs Riegl. *Revue de l'art*, *208*(2), p.65-73. https://doi.org/10.3917/rda.208.0065

Du, J. (2018). Understanding of Object Detection Based on CNN Family and YOLO. *Journal of Physics: Conference Series*, *1004*(1), 012-029. https://doi.org/10.1088/1742-6596/1004/1/012029

Dubois, S., Vanhellemont, y., & De Bouw, M. (2017). Le relevé géométrique à haute définition. *Centre Scientifique et Technique de la Construction (CSTC)* 151095.pdf (uantwerpen.be)

Dung, C. V., & Anh, L. D. (2019). Autonomous concrete crack detection using deep fully convolutional neural network. *Automation in Construction*, *99*, 52-58. https://doi.org/10.1016/j.autcon.2018.11.028

Durrant-Whyte, Hugh ; Bailey, Tim (2006). Simultaneous Localization and Mapping: Part I. IEEE Robotics & Automation Magazine 13, no 2 (juin 2006) : p. 99-110. https://doi.org/10.1109/MRA.2006.1678144

Gad, A. F. (2020). Evaluating Object Detection Models Using Mean Average Precision (mAP) | Paperspace blog. https://blog.paperspace.com/mean-average-precision/

Garrido, I., Erazo-Aux J., Lagüela S., Sfarra, S., Ibarra-Castanedo, C., Pivarčiová, E., Gianfranco, Maldague, X., & Arias, P. (2021). Introduction of Deep Learning in Thermographic Monitoring of Cultural Heritage and Improvement by Automatic Thermogram Pre-Processing Algorithms. *Sensors*, *21*(3), p.750. https://doi.org/10.3390/s21030750

Gaussuin, B. (2022). *Restaurer—Projeter. Les manières d'Eugène Viollet-le-Duc* [Thèse de doctorat, Paris Est]. https://theses.fr/2022PESC1004

Getting Started with R-CNN, Fast R-CNN, and Faster R-CNN - MATLAB & Simulink—MathWorks Benelux. (s. d.). Consulté le 25 juillet 2024, à l'adresse https://nl.mathworks.com/help/vision/ug/getting-started-with-r-cnn-fast-r-cnn-and-faster-r-cnn.html

Girshick, R. (2015). *Fast R-CNN* (arXiv:1504.08083). arXiv. https://doi.org/10.48550/arXiv.1504.08083

Girshick, R., Donahue, J., Darrell, T., & Malik, J. (2014). *Rich feature hierarchies for accurate object detection and semantic segmentation* (arXiv:1311.2524). arXiv. https://doi.org/10.48550/arXiv.1311.2524

Guelmine, L. (2019). *Pathologies des Constructions*, p.54-58 https://www.researchgate.net/publication/336287624_Pathologies_des_Constructions

Gündüz, M. Ş., & Işık, G. (2023). A new YOLO-based method for real-time crowd detection from video and performance analysis of YOLO models. *Journal of Real-Time Image Processing*, *20*(1), 5. https://doi.org/10.1007/s11554-023-01276-w

Guo, J., Liu, P., Xiao, B., Deng, L., & Wang, Q. (2024). Surface defect detection of civil structures using images: Review from data perspective. *Automation in Construction*, *158*, 105186. https://doi.org/10.1016/j.autcon.2023.105186

Hallee, M. J., Napolitano, R. K., Reinhart, W. F., & Glisic, B. (2021). Crack Detection in Images of Masonry Using CNNs. *Sensors*, *21*(14), Article 14. https://doi.org/10.3390/s21144929

Hallot, P., Mathys, A., & Jouan, P. (2022). Cours de documentation et modélisation du patrimoine tangible, Documentation et modélisation du patrimoine. Université de Liège.

Hamrouche, R. (2011). Reconnaissance géométrique des structures en maçonnerie ou en béton par imagerie radar multi récepteurs : Approche numérique et expérimentale. [Thèse de doctorat, Toulouse 3], 205-207. (PDF) Reconnaissance géométrique des structures en maçonnerie ou en béton par imagerie radar multi récepteurs : approche numérique et expérimentale | Rani Hamrouche - Academia.edu

Hatır, E., Korkanç, M., Schachner, A., & İnce, İ. (2021). The deep learning method applied to the detection and mapping of stone deterioration in open-air sanctuaries of the Hittite period in Anatolia. *Journal of Cultural Heritage*, *51*, 37-49. https://doi.org/10.1016/j.culher.2021.07.004

He, K., Gkioxari, G., Dollár, P., & Girshick, R. (2018). *Mask R-CNN* (arXiv:1703.06870). arXiv. https://doi.org/10.48550/arXiv.1703.06870

He, Y., & Chen, S. (2019). Recent Advances in 3D Data Acquisition and Processing by Time-of-Flight Camera. *IEEE Access*, *7*, 12495-12510. IEEE Access. https://doi.org/10.1109/ACCESS.2019.2891693

Héno, R., & Chandelier, L. (2014). *Numérisation 3D de bâtiments- Cas des édifices remarquables*. ISTE Editions, Londres. Numérisation 3D de bâtiments : Cas des édifices remarquables - Raphaële Héno, Laure Chandelier - Google Livres

Heynen, M. (2020). *La production numérique d'éléments du patrimoine, conservateurs d'authenticité du monument restauré.* [Mémoire, ULiège] https://matheo.uliege.be/handle/2268.2/9124

Hsu, S.-H., Hung, H.-T., Lin, Y.-Q., & Chang, C.-M. (2023). Defect inspection of indoor components in buildings using deep learning object detection and augmented reality. *Earthquake Engineering and Engineering Vibration*, *22*(1), p. 41-54. https://doi.org/10.1007/s11803-023-2152-5

ICOMOS (2012). *La Charte d'Athènes pour la Restauration des Monuments historiques*, *Conclusions de la Conférence d'Athènes, 21-30 octobre 1931*. Ier Congrès international des architectes et des techniciens des monuments historiques. ICOMOS La Charte d'Athènes pour la Restauration des Monuments Historiques - 1931 - International Council on Monuments and Sites (icomos.org)

ICOMOS. (1994). Charte internationale sur la conservation et la restauration des monuments et des sites, Charte de Venise. IIe Congrès international des architectes et des techniciens des monuments historiques. Charte internationale sur la conservation et la restauration des monuments et des sites - International Council on Monuments and Sites (icomos.org)

Idjaton, K., Janvier, R., Balawi, M., Desquesnes, X., Brunetaud, X., & Treuillet, S. (2023). Detection of limestone spalling in 3D survey images using deep learning. *Automation in Construction*, *152*, 104919. https://doi.org/10.1016/j.autcon.2023.104919

Iglhaut, J., Cabo, C., Puliti, S., Piermattei, L., O'Connor, J., & Rosette, J. (2019). Structure from Motion Photogrammetry in Forestry: A Review. *Current Forestry Reports*, *5*(3), 155-168. https://doi.org/10.1007/s40725-019-00094-3

Janiesch, C., Zschech, P., & Heinrich, K. (2021). Machine learning and deep learning. *Electronic Markets*, *31*(3), 685-695. https://doi.org/10.1007/s12525-021-00475-2

Jiang, S., & Zhang, J. (2020). Real-time crack assessment using deep neural networks with wall-climbing unmanned aerial system. *Computer-Aided Civil and Infrastructure Engineering*, *35*(6), 549-564. https://doi.org/10.1111/mice.12519

Kaja, N., & Goyal, S. (2023). IMPACT OF CONSTRUCTION ACTIVITIES ON ENVIRONMENT. *International Journal of Engineering Technologies and Management Research*, *10*(1), p.17-24. https://doi.org/10.29121/ijetmr.v10.i1.2023.1277

Kalfarisi, R., Wu, Z. Y., & Soh, K. (2020). Crack Detection and Segmentation Using Deep Learning with 3D Reality Mesh Model for Quantitative Assessment and Integrated Visualization. *Journal of Computing in Civil Engineering*, *34*(3), 04020010. https://doi.org/10.1061/(ASCE)CP.1943-5487.0000890

Kamel, M. M., Gil-Solla, A., Guerrero-Vásquez, L. F., Blanco-Fernández, Y., Pazos-Arias, J. J., & López-Nores, M. (2023). A Crowdsourcing Recommendation Model for Image Annotations in Cultural Heritage Platforms. *Applied Sciences*, *13*(19), Article 19. https://doi.org/10.3390/app131910623

Karimi, N., Mishra, M., & Lourenço, P. B. (2024). Automated Surface Crack Detection in Historical Constructions with Various Materials Using Deep Learning-Based YOLO Network. International Journal of Architectural Heritage, 0(0), 1-17. https://doi.org/10.1080/15583058.2024.2376177

Keylabs. (2022). YOLOv8 vs SSD : Choosing the Right Object Detection Model. Keylabs : Latest News And Updates. *YOLOv8 vs SSD: Choosing the Right Object Detection Model (keylabs.ai)*

Kim, B., & Cho, S. (2019). Image-based concrete crack assessment using mask and region-based convolutional neural network. *Structural Control and Health Monitoring*, e2381. https://doi.org/10.1002/stc.2381

Kumar, P., Ofli, F., Imran, M., & Castillo, C. (2020). Detection of Disaster-Affected Cultural Heritage Sites from Social Media Images Using Deep Learning Techniques. *J. Comput. Cult. Herit.*, *13*(3), 23:1-23:31. https://doi.org/10.1145/3383314

Kwon, D., & Yu, J. (2019). AUTOMATIC DAMAGE DETECTION OF STONE CULTURAL PROPERTY BASED ON DEEP LEARNING ALGORITHM. *The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, *42*(2), p.639-643. https://doi.org/10.5194/isprs-archives-XLII-2-W15-639-2019

Lamas, A., Tabik, S., Cruz, P., Montes, R., Martínez-Sevilla, Á., Cruz, T., & Herrera, F. (2021). MonuMAI : Dataset, deep learning pipeline and citizen science based app for monumental heritage taxonomy and classification. *Neurocomputing*, *420*, 266-280. https://doi.org/10.1016/j.neucom.2020.09.041

Landes, T., Grussenmeyer, P., & BOULAASSAL, H. (2011). Les principes fondamentaux de la lasergrammétrie terrestre: acquisition, traitement des données et applications (partie 2/2). *XYZ*, (129), p. 25-38.

Lee, J.-D., Chien, J.-C., Hsu, Y.-T., & Wu, C.-T. (2021). Automatic Surgical Instrument Recognition—A Case of Comparison Study between the Faster R-CNN, Mask R-CNN, and Single-Shot Multi-Box Detectors. *Applied Sciences*, *11*(17), Article 17. https://doi.org/10.3390/app11178097

Li, S., Zhao, X., & Zhou, G. (2019). Automatic pixel-level multiple damage detection of concrete structure using fully convolutional network. *Computer-Aided Civil and Infrastructure Engineering*, *34*(7), p. 616-634. https://doi.org/10.1111/mice.12433

Li, Y., Chen, Y., Liu, G., & Jiao, L. (2018). A Novel Deep Fully Convolutional Network for PolSAR Image Classification. *Remote Sensing*, *10*, 1984. https://doi.org/10.3390/rs10121984

Liu, W., Anguelov, D., Erhan, D., Szegedy, C., Reed, S., Fu, C.-Y., & Berg, A. C. (2016). *SSD : Single Shot MultiBox Detector* (Vol. 9905, p. 21-37). https://doi.org/10.1007/978-3-319-46448-0_2

Liu, Z., Brigham, R., Long, E. R., Wilson, L., Frost, A., Orr, S. A., & Grau-Bové, J. (2022). Semantic segmentation and photogrammetry of crowdsourced images to monitor historic facades. *Heritage Science*, *10*(1), 27. https://doi.org/10.1186/s40494-022-00664-y

Long, J., Shelhamer, E., & Darrell, T. (2015). *Fully Convolutional Networks for Semantic Segmentation* (arXiv:1411.4038). arXiv. https://doi.org/10.48550/arXiv.1411.4038

Luhmann, T., Robson, S., Kyle, S., & Boehm, J. (2019). *Close-Range Photogrammetry and 3D Imaging*. Walter de Gruyter GmbH, p.2-4

Ma, J., Yan, W., Liu, G., Xing, S., Niu, S., & Wei, T. (2022). Complex Texture Contour Feature Extraction of Cracks in Timber Structures of Ancient Architecture Based on YOLO Algorithm. *Advances in Civil Engineering*, *2022*(1), 7879302. https://doi.org/10.1155/2022/7879302

Magri-Djenane, S., Madhoui, M. & Belarbi, S. (2011-2012). C*ours : Technique du relevé architectural. Université Mohamed Khider Biskra*. Département d'architecture, p.3-11.

Mansuri, L. E., & Patel, D. A. (2022a). Artificial Intelligence for Heritage Conservation : A Case Study of Automatic Visual Inspection System. *Current State of Art in Artificial Intelligence and Ubiquitous Cities, p.* 1-15 https://doi.org/10.1007/978-981-19-0737-1_1

Mansuri, L. E., & Patel, D. A. (2022b). Development of Automated Web-Based Condition Survey System for Heritage Monuments Using Deep Learning. In S. Belayutham, C. K. I. Che Ibrahim, A. Alisibramulisi, H. Mansor, & M. Billah (Éds.), *Proceedings of the 5th International Conference on Sustainable Civil Engineering Structures and Construction Materials* (p. 1179-1192). Springer Nature. https://doi.org/10.1007/978-981-16-7924-7_76

Mansuri, L. E. & Patel, D.A. (2022c). *Ar*tificial intelligence-based automatic visual inspection system for built heritage, *Smart and Sustainable Built Environment*, *11*(3), p. 622-646. https://doi.org/10.1108/SASBE-09-2020-0139

Marín-García, D., Bienvenido-Huertas, D., Carretero-Ayuso, M. J., & Torre, S. D. (2023). Deep learning model for automated detection of efflorescence and its possible treatment in images of brick facades. *Automation in Construction*, *145*, 104658. https://doi.org/10.1016/j.autcon.2022.104658

Mason, A. (s. d.). *The Haskins Society—3D Photogrammetry with PhotoScan*. Consulté le 17 juillet 2024, à l'adresse https://thehaskinssociety.wildapricot.org/photogrammetry

Matrone, F., Grilli, E., Martini, M., Paolanti, M., Pierdicca, R., & Remondino, F. (2020). Comparing Machine and Deep Learning Methods for Large 3D Heritage Semantic Segmentation. *ISPRS International Journal of Geo-Information*, *9*(9), Article 9. https://doi.org/10.3390/ijgi9090535

Meklati, S., Boussora, K., Abdi, M. E. H., & Berrani, S.-A. (2023). Surface Damage Identification for Heritage Site Protection : A Mobile Crowd-sensing Solution Based on Deep Learning. *Journal on Computing and Cultural Heritage*, *16*(2), https://doi.org/10.1145/3569093

Mijwil, M., Aggarwal, K., Doshi, R., Hiran, K., & Gök, M. (2022). The Distinction between R-CNN and Fast R-CNN in Image Analysis : A Performance Comparison. *Asian Journal of Applied Sciences*, *10*, 429-437. https://doi.org/10.24203/ajas.v10i5.7064

Mishra, M., & Lourenço, P. B. (2024). Artificial intelligence-assisted visual inspection for cultural heritage: State-of-the-art review. *Journal of Cultural Heritage*, *66*, p. 536-550. https://doi.org/10.1016/j.culher.2024.01.005

Mishra, M., Barman, T., & Ramana, G. V. (2022a). Artificial intelligence-based visual inspection system for structural health monitoring of cultural heritage. *Journal of Civil Structural Health Monitoring*, *14*(1), 103-120. https://doi.org/10.1007/s13349-022-00643-8

Mishra, M., Lourenço, P. B., & Ramana, G. V. (2022b). Structural health monitoring of civil engineering structures by using the internet of things : A review. *Journal of Building Engineering*, *48*, 103954. https://doi.org/10.1016/j.jobe.2021.103954

Nanos, G., & Aibin, M. (2022). *Object Detection : SSD Vs. YOLO | Baeldung on Computer Science*. https://www.baeldung.com/cs/object-detection-ssd-yolo

Pathak, R., Saini, A., Wadhwa, A., Sharma, H., & Sangwan, D. (2021). An object detection approach for detecting damages in heritage sites using 3-D point clouds and 2-D visual data. J*ournal Of Cultural Heritage*, 48, p.74-82. https://doi.org/10.1016/j.culher.2021.01.002

Perez, H., & Tah, J. H. M. (2021). Deep learning smartphone application for real-time detection of defects in buildings. *Structural Control and Health Monitoring*, *28*(7), e2751. https://doi.org/10.1002/stc.2751

Perez, H., Tah, J. H. M., & Mosavi, A. (2019). Deep Learning for Detecting Building Defects Using Convolutional Neural Networks. *Sensors*, *19*(16), Article 16. https://doi.org/10.3390/s19163556

Philipparie, P. (2019). *Pathologie générale du bâtiment : Diagnostic, remèdes & prévention*. Editions Eyrolles.

Pierrot-Deseilligny, M., De Luca, L., Remondino, F., 2011. Automated Image-Based Procedures for Accurate Artifacts 3D Modeling and Orthoimage Generation. Geoinformatics FCE CTU 6, 291–299. https://doi.org/10.14311/gi.6.36

Pratibha, K., Mishra, M., Ramana, G. V., & Lourenço, P. B. (2024). Deep Learning-Based YOLO Network Model for Detecting Surface Cracks During Structural Health Monitoring. In Y. Endo & T.
Redmon, J., Divvala, S., Girshick, R., & Farhadi, A. (2016). *You Only Look Once : Unified, Real-Time Object Detection*. 779-788. Redmon_You_Only_Look_CVPR_2016_paper.pdf (cv-foundation.org)

Ren, S., He, K., Girshick, R., & Sun, J. (2016). *Faster R-CNN : Towards Real-Time Object Detection with Region Proposal Networks.* 1506.01497 (arxiv.org)

Rosebrock, A. (2016). Intersection over Union (IoU) for object detection. *PyImageSearch*. https://pyimagesearch.com/2016/11/07/intersection-over-union-iou-for-object-detection/

Saint-Aubin, J.-P. (1992). *Le relevé et la représentation de l'architecture*. Inventaire général - ELP. https://hal.science/hal-02902017

Samhouri, M., Al-Arabiat, L., & Al-Atrash, F. (2022). Prediction and measurement of damage to architectural heritages facades using convolutional neural networks. *Neural Computing and Applications*, *34*(20), 18125-18141. https://doi.org/10.1007/s00521-022-07461-5

Shah, D. (2022). Mean Average Precision (mAP) Explained: Everything You Need to Know. V7. https://www.v7labs.com/blog/mean-average-precision

SPW (2023). *Schéma de développement du territoire*. sdt-projet-30-mars-2023.pdf (wallonie.be)

Stancel, M., & Hulic, M. (2019). An Introduction to Image Classification and Object Detection using YOLO Detector. http://ceur-ws.org/Vol-2403/paper4.pdf

Tanduo, B., Teppati Losè, L., & Chiabrando, F. (2023). DOCUMENTATION OF COMPLEX ENVIRONMENTS IN CULTURAL HERITAGE SITES. A SLAM-BASED SURVEY IN THE CASTELLO DEL VALENTINO BASEMENT. *The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, *XLVIII-1-W1-2023*, p. 489-496. https://doi.org/10.5194/isprs-archives-XLVIII-1-W1-2023-489-2023

UNFCCC. (2021). Pacte de Glasgow pour le climat. Report of the Conference of the Parties serving as the meeting of the Parties to the Paris Agreement on its third session, held in Glasgow from 31 October to 13 November 2021 cma2021_10_add1_adv.pdf (unfccc.int)

Vallière, G. (1998). *Le ravalement de façade : Mode d'emploi*. Editions Eyrolles.

Verma, Y. (2021). *R-CNN vs Fast R-CNN vs Faster R-CNN - A Comparative Guide*. AIM. Consulté le 29 mai 2024, à l'adresse https://analyticsindiamag.com/r-cnn-vs-fast-r-cnn-vs-faster-r-cnn-a-comparative-guide/

Viollet-le-Duc, E.-E. (1866). Dictionnaire raisonné de l'architecture française du XIe au XVIe siècle. Morel.

Voulodimos, A., Doulamis, N., Doulamis, A., & Protopapadakis, E. (2018). Deep Learning for Computer Vision : A Brief Review. *Computational Intelligence And Neuroscience*, 2018, 1-13. https://doi.org/10.1155/2018/7068349

Wang, J., Turko, R., Shaikh, O., Park, H., Das, N., Hohman, F., Kahng, M., & Chau, P. (2020a). CNN Explainer. Consulté le 24 juillet 2024, à l'adresse https://poloclub.github.io/cnn-explainer/

Wang, N., Zhao, X., Wang, L., & Zou, Z. (2019). Novel System for Rapid Investigation and Damage Detection in Cultural Heritage Conservation Based on Deep Learning. *Journal of Infrastructure Systems*, *25*(3), 04019020. https://doi.org/10.1061/(ASCE)IS.1943-555X.0000499

Wang, X., Kong, T., Shen, C., Jiang, Y., & Li, L. (2020b). *SOLO : Segmenting Objects by Locations* (arXiv:1912.04488). arXiv. https://doi.org/10.48550/arXiv.1912.04488

Watt, D., & Swallow, P. (1995). *Surveying Historic Buildings*. Donhead. p.1-140

Wei, G., Wan, F., Zhou, W., Xu, C., Ye, Z., Liu, W., Lei, G., & Xu, L. (2023). BFD-YOLO : A YOLOv7-Based Detection Method for Building Façade Defects. *Electronics*, *12*(17), Article 17. https://doi.org/10.3390/electronics12173612

Wu, J., Shi, Y., Wang, H., Wen, Y., & Du, Y. (2023). Surface Defect Detection of Nanjing City Wall Based on UAV Oblique Photogrammetry and TLS. *Remote Sensing*, *15*(8), Article 8. https://doi.org/10.3390/rs15082089

Yang, S., Hou, M., & Li, S. (2023a). Three-Dimensional Point Cloud Semantic Segmentation for Cultural Heritage : A Comprehensive Review. *Remote Sensing*, *15*(3), Article 3. https://doi.org/10.3390/rs15030548

Yang, X., Zheng, L., Chen, Y., Feng, J., & Zheng, J. (2023b). Recognition of Damage Types of Chinese Gray-Brick Ancient Buildings Based on Machine Learning—Taking the Macau World Heritage Buffer Zone as an Example. *Atmosphere*, *14*(2), Article 2. https://doi.org/10.3390/atmos14020346

Zhang, K., Mea, C., Fiorillo, F., & Fassi, F. (2024). CLASSIFICATION AND OBJECT DETECTION FOR ARCHITECTURAL PATHOLOGY: PRACTICAL TESTS WITH TRAINING SET. T*he International Archives Of The Photogrammetry,* Remote Sensing And Spatial Information Sciences, 48(2), p.477-484. https://doi.org/10.5194/isprs-archives-xlviii-2-w4-2024-477-2024

Zou, Z., Zhao, X., Zhao, P., Qi, F., & Wang, N. (2019). CNN-based statistics and location estimation of missing components in routine inspection of historic buildings. *Journal of Cultural Heritage*, *38*, p.221-230. https://doi.org/10.1016/j.culher.2019.02.002

VII. TABLE DES ILLUSTRATIONS