Utilisation des images satellites pour améliorer le repérage des logements à Mayotte

Maëlys Bernard
Insee
maelys.bernard@insee.fr

Raya Berova
Insee
raya.berova@insee.fr

Thomas Faria
Insee
thomas.faria@insee.fr

2025-07-25

Abstract Malgré leur disponibilité de plus en plus accrue, les images satellites très haute résolution sont jusqu'ici très peu utilisées au sein de la statistique publique. La manipulation de ces données non conventionnelles n'appartient pas encore aux pratiques usuelles des statisticiens publics, bien qu'elles offrent des opportunités significatives pour améliorer ou créer de nouveaux indicateurs statistiques. Cet article vise justement à démontrer qu'il est possible d'intégrer ces données satellitaires dans les processus statistiques afin d'assurer leur qualité. Mayotte constitue un cas emblématique où les difficultés d'accès au terrain et la forte dynamique de développement urbain complexifient l'organisation des enquêtes cartographiques qui permettent de repérer les logements en amont du recensement. Afin d'orienter les travaux de contrôle de la qualité de ces enquêtes, nous avons exploré l'utilisation des images satellitaires très haute résolution Pléiades (0,5 m de résolution spatiale), en combinaison avec des méthodes d'apprentissage profond, spécifiquement via des modèles de segmentation sémantique. La méthodologie développée consiste en l'entraînement d'un modèle SegFormer, fondé sur une architecture de type Transformer, largement reconnue dans la littérature puisqu'elle est à la base des modèles GPT. Le modèle a été entraîné grâce aux annotations issues du projet CoSIA de l'IGN, qui fournit une référence précise de la couverture du sol. À partir des prédictions du modèle, il est possible de produire des estimations précises des surfaces bâties sur un territoire donné, permettant ainsi d'analyser efficacement leur évolution dans le temps. Les résultats obtenus démontrent une performance élevée dans la détection automatisée des évolutions urbaines dans les DROM. Ces résultats sont disponibles sous la forme d'un tableau de bord interactif permettant aux équipes opérationnelles de cibler efficacement les zones prioritaires pour la réalisation d'enquêtes cartographiques de contrôle sur le terrain.

Après avoir décrit la méthode de prédiction de surface du bâti sur les images satellites, cet article présentera un cas pratique d'utilisation de ces prédictions à travers le contrôle

qualité des enquêtes cartographiques de la population à Mayotte. Depuis 2021, le recensement à Mayotte fonctionne sur un cycle quinquennal comme en métropole et dans les autres DOM. En amont de la collecte auprès des habitants, une enquête cartographique permet de repérer les logements. Le résultat de l'enquête cartographique détermine la base d'immeubles au sein de laquelle est sélectionné l'échantillon1 de l'enquête annuelle de recensement. Ainsi, si l'enquête cartographique ne couvre pas bien l'intégralité du bâti du territoire à enquêter, les populations et nombre de logements déduits de l'enquête annuelle de recensement seront sous-estimés du fait de ce défaut de couverture. Les données satellites permettent de contrôler la qualité des enquêtes cartographiques en comparant l'évolution de la couverture de bâti avec celle du nombre de logements observé suite aux enquêtes cartographiques au sein de chaque îlot2 à enquêter une année donnée. Les îlots sont répartis en cinq groupes de rotation correspondant à l'année à laquelle a lieu l'enquête annuelle de recensement. Afin d'assurer la qualité des enquêtes cartographiques, un deuxième passage sur le terrain a été planifié dans les grandes communes mahoraises pour certains îlots identifiés comme les plus prioritaires. Cet ordre de priorité a été défini par une méthode mobilisant les images satellites pour repérer les plus susceptibles d'être concernés par des manques potentiels dans les enquêtes cartographiques. L'enquête cartographique corrective ciblant ces îlots a permis d'assurer une meilleure couverture de ces enquêtes et d'éviter un risque de sous-estimation des logements.

Table des matières

1 Introduction

Reprendre l'abstract soumis pour faire l'intro Reprendre les slides du séminaire à la DIRAG pour le contexte « Momentum de Varsovie » Repartir de l'origine du projet (demande DROM Mayotte/ Guyane) Parler de l'IGN et de leur travaux avec leur dataset FLAIR et leur modèle CoSIA

2 Méthodologie

Nous allons dans cette partie revenir sur les concepts d'apprentissage profond (*deep learning*) afin d'expliquer notre démarche et l'application de ces méthodes à nos problématiques précises. Nous définirons ensuite les données que nous avons utilisée avant de détailler les méthodes d'évaluation que nous avons appliqués à nos algorithmes.

2.1 Les modèles de d'apprentissage profond pour l'analyse d'images

Dans le domaine de l'apprentissage statistique, les réseaux de neurones profonds en sont un sous-domaine. Si les premiers algorithmes de réseaux de neurones ont été développé dès les années, les réseaux de neurones profonds ont eux fait leur révolution dans les années 2010 grâce notamment aux développement de nouvelles cartes graphiques (GPU) et d'algorithmes optimisées pour celles ci. En effet, leur développement a longtemps été freiné par la quantité de données nécessaires pour obtenir de bons résultats impliquant donc des temps de calculs très longs et des coûts d'annotations prohibitifs. La conception des réseaux de neurones profond est inspirée du fonctionnement du cerveau humain. En effet, l'idée est de représenter les neurones par des fonctions mathématiques très simples qui s'activent si le signal reçu en entrée est suffisamment fort et transmettent alors leur « activation » dans ce cas. Un grand nombre de neurones correctement organisés permet alors de réaliser des opérations plus complexes, résultantes des différentes activations.

Si l'apprentissage profond est devenu la solution de facto pour l'analyse d'images par ordinateur, et notamment les tâches de segmentation sémantique c'est en parti grâce aux travaux fondateurs de Long et al. (2015) qui ont développé les réseaux entièrement convolutifs, une extension de l'architecture des réseaux de neurones convolutif proposé par LeCun et al. (1989).

2.1.a Réseaux de neurones convolutifs

Les réseaux de neurones convolutifs (CNN) tirent leur inspiration du cortex visuel des animaux et se composent de deux parties (voir Figure 1) :

1. Une première partie composée par des couches de convolutions successives qui permet d'extraire des prédicteurs de l'image en entrée du réseau;
2. Une seconde partie permettant de classifier les pixels de l'image en entrée à partir des prédicteurs obtenus dans la première partie.

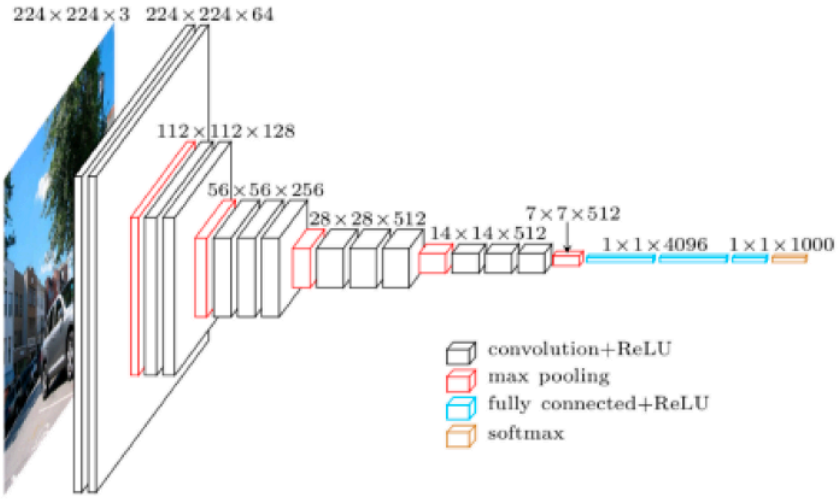

Figure 1. – Réseau de neurones convolutif

Note de lecture : La partie convolutive est représentée par les différents cubes à gauche et la partie classifiante est représentée par les rectangles fins et bleus à droite

Dans la partie convolutive, l'image en entrée du réseau se voit appliquer des filtres appelés convolutions, représentés par des matrices $A = (a_{ij})$ de petite taille appelés noyaux de convolution. L'image en sortie de l'opération de convolution est obtenue à partir de chaque pixel de l'image en entrée en calculant la somme des pixels avoisinant, pondérée par les coefficients a_{ij} du noyau de convolution. Cette opération de convolution est illustrée dans la Figure 2, extraite de l'ouvrage Kim (2017).

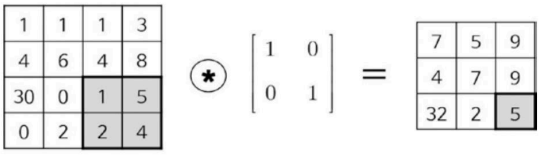

Figure 2. – Opération de convolution

Note de lecture : L'image résultante est une image plus petite dont les pixels sont égaux à la somme des pixels de l'image en entrée pondérée par les coefficients du noyau de convolution.

Une convolution vise ainsi à résumer l'information présente dans une région de l'image. Il est d'ailleurs intéressant de remarquer que lorsque le voisinage d'un pixel ressemble à la structure du noyau de convolution, la valeur de sortie est élevée. Ainsi, si le filtre est, par exemple, un détecteur de ligne verticale, il générera une grande valeur sur les pixels appartenant à des lignes verticales. Un filtre de convolution permet donc d'obtenir une sorte de carte de caractéristiques de l'image. Une couche convolutive correspond finalement à l'application d'un nombre n_f de filtres différents en parallèle. Ainsi, en sortie d'une couche, il y a autant d'images (carte de caractéristiques) que de filtres appliqués. Ces n_f cartes deviennent les canaux d'entrée de la couche suivante.

Les CNN se composent de couches successives, chacune ayant un rôle spécifique. Les premières couches détectent des motifs simples dans l'image (bords, textures, lignes horizontales ou verticales…) tandis que les couches intermédiaires et profondes, plus spécialisées, combinent ces motifs simples pour repérer des formes de plus en plus complexes (motifs, objets, visages…). La Figure 3 schématise un réseau de neurones convolutif.

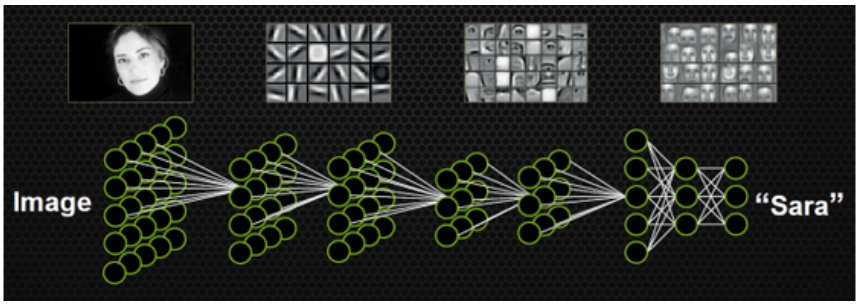

Figure 3. – Représentation d'une convolution

Note de lecture : Les premières couches convolutives reconnaissent les formes simples tandis que les couches les plus profondes à droite reconnaissent des formes plus concrètes.

Après une couche de convolution, la taille des cartes de caractéristiques reste identique à celle de l'image, ce qui peut rapidement entraîner un volume de données important et rendre le modèle coûteux à entraîner. Pour pallier cela, une méthode couramment utilisée est le *max pooling*. Cette opération, représenté par la Figure 4 consiste à diviser chaque carte de caractéristiques en petites régions, puis à ne conserver, pour chacune, que la valeur maximale. Le *max pooling* permet ainsi de réduire la dimension spatiale des données tout en conservant l'information la plus pertinente. Cette réduction favorise également une certaine invariance locale aux petites translations ou déformations de l'image, et contribue à limiter le nombre de paramètres et la complexité de calcul du réseau, tout en mettant en valeur les motifs les plus discriminants. D'autres opérations telles que le padding détaillé dans la Figure 5 permettent malgré tout de contrôler la dimension de l'image en sortie.

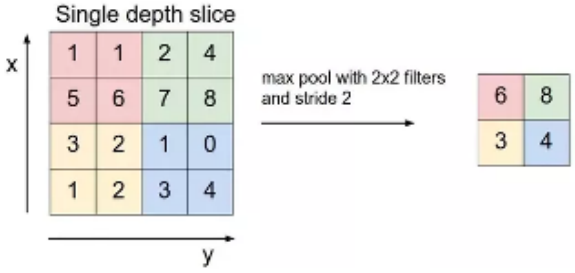

Figure 4. – Représentation du max pooling

Note de lecture : L'image résultante de l'opération de max-pooling est égale au maximum de chaque pixel dans chacune des zones dessinées sur l'image de gauche.

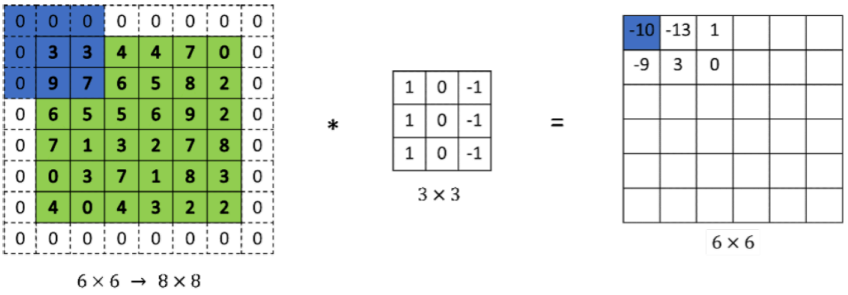

Figure 5. – Représentation du padding

Note de lecture : l'ajout d'une bande de 0 autour de l'image avant d'appliquer la convolution permet de limiter la réduction de dimension de l'image résultante.

Une fonction d'activation non linéaire, comme la ReLU (Rectified Linear Unit), est appliquée systématiquement après chaque couche de convolution (cf. Figure 6). Inspirée du fonctionnement des neurones biologiques, cette opération introduit une non-linéarité dans le réseau. Sans cette étape, l'empilement de couches convolutives ne ferait qu'appliquer une combinaison linéaire de filtres, ce qui limiterait la capacité du modèle à apprendre des relations complexes. L'ajout de fonctions d'activation non linéaires permet ainsi au réseau d'extraire et de modéliser des motifs variés et des structures non linéaires présentes dans les données, ce qui est essentiel pour traiter efficacement des images. L'utilisation de fonctions d'activation non linéaires ne se limite pas aux réseaux convolutifs, c'est justement une composante fondamentale de l'ensemble des architectures d'apprentissage profond.

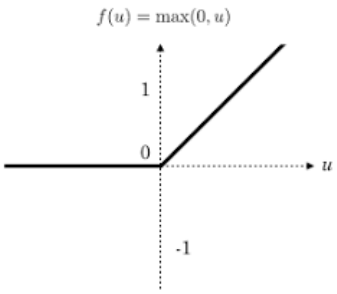

Figure 6. – Fonction d'activation ReLu

Note de lecture : la fonction ReLU « s'active » quand l'argument est positif.

Contrairement aux approches classiques du machine learning, la particularité des réseaux convolutifs réside dans l'automatisation de l'extraction des caractéristiques (*feature extraction*) de l'image. Dans un cadre traditionnel, cette étape est généralement réalisée de manière manuelle ou repose sur l'expertise métier : on choisit alors à l'avance quels prédicteurs utiliser (par exemple, des statistiques sur certaines bandes spectrales ou des filtres définis à la main). À l'inverse, dans un réseau de neurones convolutif, ce sont les poids des noyaux de convolution eux-mêmes qui sont appris automatiquement au cours de l'entraînement. Ces coefficients font partie du vecteur de paramètres du modèle, noté θ. Ainsi, une fois le réseau entraîné, l'ensemble des filtres optimaux θ^* représente la meilleure façon de transformer et d'extraire les informations pertinentes

de l'image. Les cartes de caractéristiques produites par l'enchaînement de ces filtres alimentent ensuite la partie classifiante du réseau. Cette approche permet au modèle de découvrir de manière autonome les motifs les plus discriminants pour la tâche visée, sans intervention manuelle dans la conception des prédicteurs.

En sortie d'un réseau de neurones convolutif classique, la partie dite classifiante permet d'associer une catégorie à l'image analysée. Par exemple, on peut vouloir déterminer si une image représente un chien ou un chat. Pour cela, la sortie de la partie convolutive (c'est-à-dire les cartes de caractéristiques extraites par les filtres) est d'abord transformée en un seul vecteur grâce à une opération de mise à plat (*flattening*). Ce vecteur est ensuite traité par un réseau de neurones dense, appelé aussi *fully connected*, qui produit en sortie un vecteur de scores $x = (x_0, ..., x_9)$ lorsqu'on cherche à classer l'image parmi 10 classes possibles.

Pour interpréter ces scores comme des probabilités, on applique la fonction *softmax* :

$$\text{Softmax } (x_i) = \frac{\exp x_i}{\sum_{j=0}^{9} \exp x_j}.$$

Cette opération convertit les scores en une distribution de probabilité, dont la somme vaut 1, et permet de sélectionner la classe associée à la probabilité la plus élevée comme prédiction finale. La fonction *softmax* est particulièrement utile car elle est infiniment dérivable, ce qui garantit la dérivabilité de tout le modèle f_θ par rapport à ses paramètres θ. Cela rend possible l'utilisation de la descente de gradient pour ajuster les poids du réseau au cours de l'apprentissage.

2.1.b Segmentation sémantique

On vient de le voir, les réseaux de neurones convolutifs sont particulièrement adaptés pour réaliser une prédiction par image. C'est pour cela qu'ils ont été historiquement utilisé pour accomplir de la classification d'image (par exemple, classer des écritures manuscrites de chiffre). Dans le cadre de notre projet, l'objectif est plutôt d'être capable d'identifier automatiquement des zones de bâtis sur des images satellites. L'idée n'est donc pas de faire une classification à l'échelle de l'image entière mais à l'échelle du pixel. De sorte à ce qu'il soit possible de délimiter des zones appartenant à une même classe pré-définie au sein même d'une image. C'est justement le principe de la segmentation sémantique qui retourne donc une sortie dense, c'est à dire des labels de même taille que l'image en entrée. Cette tâche a réellement fait de gros progrès suite au papier fondateur de Long, Shelhamer, & Darrell (2015) sur les réseaux de neurones entièrement convolutifs.

Pour rappel, avec les réseaux de neurones convolutifs classiques on empile des couches de convolution et afin d'accroître la profondeur de l'image et grâce au mécanisme de *max pooling* on réduit sa taille. Au final, on aplatit tout afin de recréer un réseau de neurones dense *classique* permettant de réaliser une classification. Cependant cette étape d'aplatissement fait perdre complètement la structure spatiale de l'image (les liens entre pixels voisins), qui est absolument nécessaire pour faire de la segmentation sémantique, à savoir « à quel classe appartient chaque pixel ? ». L'idée derrière les FCN est donc de supprimer cet aplatissement ainsi que les couches entièrement connectées pour les remplacer par une convolution 1*1. De cette manière, chaque pixel devient

un classifieur local qui prédit sa classe en fonction de ce que les couches profondes ont extrait, on a alors autant de petites images que de classes dans lesquelles on souhaite segmenter.

Le problème est qu'après cette convolution 1*1, et suite à toutes les couches de* max pooling*, l'image en sortie est de plus petite taille que l'image en entrée, or pour prédire un label par pixel, l'image en sortie doit être de même taille que l'image en entrée. Long, Shelhamer, & Darrell (2015) propose alors une solution : la déconvolution (ou* transposed convolution*). Bien qu'il s'agissent d'un abus de langage, on peut considérer la déconvolution comme le processus inverse de la convolution dans le sens ou elle augmente la résolution. La **?@fig-deconvolution** représente schématiquement ce mécanisme.

Ainsi pour résumer, les modèles de segmentation sont généralement représentés par une structure en forme de U avec :

- Une partie descendante, l'encodeur, qui va permettre de transformer l'image en entrée en un vecteur numérique de taille réduite par rapport au nombre de pixels initial de précédente grâce à des couches successives de convolution.
- Une partie ascendante, le décodeur, qui va partir du vecteur obtenu précédemment (aussi appelé embedding) et remonter par opérations de déconvolutions à une sortie de la même dimension que l'image. Il est important de noter que ces deux parties du U sont paramétrées. En effet, les poids des différents noyaux de convolution et de déconvolution sont appris lors de l'entraînement.

Parmi les extensions aux FCN les plus remarquables on peut citer le modèle U-net Ronneberger et al. (2015). Dans les architectures FCN classique, les couches de convolution associée au pooling réduisent progressivement la taille des cartes de caractéristiques ce qui supprime les détails spatiaux fin et génère des segmentations floues. L'idée novatrice des auteurs est de construire une architecture en U symétrique de sorte à réutiliser les cartes caractéristiques produites lors de la partie descendante dans la partie ascendante afin qu'elle bénéficie des détails locaux comme le montre la **?@fig-unet**. D'autres architectures comme le modèle DeepLabv3 proposé par Chen & al. (2017) ont également permis de minimiser la perte d'information lors de l'opération de pooling. Leur idée est d'utiliser des convolutions dilatés (*atrous convolution* ou *dilated convolution*), c'est-à-dire des convolutions de plus grande taille mais avec des zéros entre les poids du filtre. Cela permet de maintenir une quantité d'information suffisante sans augmenter le nombre de paramètre à estimer.

Malgré ces réelles avancées, les architectures basées sur les convolutions présentent plusieurs limites. En effet, par construction ils sont une portée très locale puisqu'une convolution n'observe qu'un voisinage local. Pour élargir le contexte il est nécessaire d'empiler des couches au coût d'un accroissement du temps de calcul. Ainsi, s'ils sont très bon localement, ils ont tendance à ne pas être invariant spatialement (i.e. une voiture en haut à gauche de l'image ou en bas à droite reste une voiture.) et deux parties d'un même objet éloignées dans l'image sont rarement reliés ensemble.

La révolution amorcée par les Transformers (voir Vaswani et al. (2017)) dans le traitement automatique du langage naturel (NLP) a également marqué un tournant dans le domaine de la vision

par ordinateur. En 2020, les Vision Transformers (ViT), introduits par Dosovitskiy et al. (2021) chercheurs de Google, ont bouleversé les approches classiques de la vision artificielle. Leur particularité réside dans l'adoption du même mécanisme fondamental que celui utilisé en NLP : le *self-attention*. Ce mécanisme permet au modèle de se concentrer sur différentes parties d'une image (comme il le ferait avec des mots dans une phrase), afin de mieux en comprendre la structure et le contenu.

Contrairement aux réseaux convolutifs (CNN) qui balayent une image via des filtres, les Vision Transformers commencent par diviser l'image en petits blocs appelés *patches*. Chaque patch est ensuite aplatit et encodé comme un vecteur (*embedding*), exactement comme un mot l'est dans un modèle de langage. Il faut donc s'imaginer qu'une image devient une séquence de vecteurs de la même manière qu'une phrase est une séquence de mots. Cependant, contrairement au texte, les *patches* d'images n'ont pas d'ordre intrinsèque donc il est nécessaire de rajouter une information sur la position de chaque patch dans l'image. C'est ce qu'on appelle l'encodage positionnel (*positional encoding*). Finalement, une fois que tous les *patches* de l'image sont encodés on leur applique le mécaniste de *self-attention* qui va permettre de lier tous les *patches* entre eux en fonction de leur importance et ainsi capturer les dépendances aussi bien locale que globale. On obtient alors de nouveaux vecteurs qui sont des représentations des *patches* individuels, enrichies par le contexte global. Il est important de noter qu'on empile généralement plusieurs couches Transformer (par exemple, « $L \times$ » signifie L couches dans la **?@fig-vit**). Finalement, les vecteurs obtenus pour chaque patch peuvent être utilisé pour réaliser une tâche de classification classique avec une dernière couche de réseaux de neurones entièrement connectés.

Le succès des ViT, originalement conçus plutôt pour des tâches de classification d'images a permis de nouvelles avancées pour la segmentation d'image. Les ViT classiques présentent plusieurs limites pour réaliser de la segmentation :

1. Les informations fines de localisation sont perdues et l'encodage positionnel n'est pas assez précis pour de la segmentation
2. Contrairement aux CNN qui construisent des représentations à différentes échelles (petits détails et vue globale), le ViT standard traite tous les *patches* à la même résolution.
3. Le mécanisme d'attention a un coût quadratique avec la taille des patches. Plus on veut de détails (donc des patches plus petits), plus ça devient lourd à calculer.

Ce sont ces limites que le modèle Segformer Xie et al. (2021), développé par Nvidia, tentent de palier. Le modèle SegFormer, est défini par deux blocs clés un encodeur (hiérarchique) basé sur des couches Transformer et un décodeur très simple pour produire la carte de segmentation. L'idée derrière le SegFormer est de faire un mix entre les bénéfices des CNN (hiérarchie et efficacité locale) et ceux des ViT (efficacité globale). Ainsi, au lieu d'un seul niveau comme dans les ViT, le SegFormer propose un encodeur en plusieurs étages, comme un CNN. L'image passe par plusieurs couches Transformer, chacune réduisant la résolution progressivement. De plus, les patchs sont construits en se chevauchant (*Overlapped Patch Merging*) de sorte à garder une meilleure continuité spatiale. Les auteurs montrent que cette structure hiérarchique à multi-résolution ainsi que ces fenêtres locales chevauchantes permettent de se passer de l'encodage positionnel utilisé dans les ViT. Finalement, l'une des particularité du Segformer est également son décodeur très

simple qui contraste avec les décodeurs complexes de l'U-Net ou DeepLab. En effet, il prend les sorties des différents niveaux - 4 dans le papier - de l'encodeur qui représentent des résolutions différentes. Il les projette dans un espace commun, les interpole à la même taille, puis les concatène pour produire la carte de segmentation via quelques couches simples. Tout cela permet au Segformer d'être un modèle avec relativement peu de paramètres et donc très rapide à entraîner ou inférer.

D'autres modèles de segmentation peuvent être considéré à l'état de l'art comme le SETR Zheng et al. (2021), Swin Transformer Liu et al. (2021) ou encore l'impressionnant SAM 2 (Segment Anything Model) Ravi et al. (2024) développé par Meta. Cependant, dans le cadre de notre projet, après avoir testé des modèles DeepLabv3, nous avons choisi d'utiliser exclusivement le modèle Segformer-B5[1] avec lequel nous avons obtenus les meilleurs résultats avec des temps de calculs tout à fait raisonnable. Pour cela, nous avons utilisé le modèle pré-entrainé mis à disposition par Nvidia et nous l'avons spécialisé sur nos images satellites des DROM.

2.2 Données

2.2.a Images satellites

Une image satellite est une matrice à trois dimensions. Chaque élément de cette image est un **pixel**, qui correspond à une surface au sol (par exemple 10m*10m). Le pixel contient plusieurs valeurs numériques. Ces valeurs expriment l'intensité du rayonnement solaire reflété dans chaque **bande spectrale** pour ce pixel. Une bande spectrale correspond à une portion du spectre électromagnétique, qui peut être par exemple le bleu, le rouge, le vert, le proche infrarouge. Donc, pour une image satellite de 200*200 pixels avec les trois bandes du visible (rouge, vert, bleu), chaque pixel aura trois valeurs différentes représentant l'intensité dans chacune des bandes du visible. Ainsi, un pixel qui représente 10m*10m au sol donnera une couleur : du violet sur l'exemple de la Figure 7.

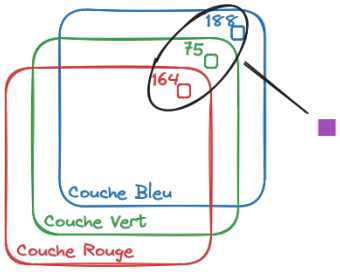

Figure 7. – Exemple d'un pixel d'une image satellite

Il existe de nombreux produits satellitaires disponibles. Nous nous sommes surtout concentrés sur deux d'entre eux : **Pléiades et Sentinel2**.

[1] https://github.com/NVlabs/SegFormer

Figure 8. – Mamoudzou, Mayotte (2024)

Les images satellites **Pléiades** constituent une ressource précieuse dans notre cas d'usage. Ce sont des images à très haute résolution, spécifiquement conçues pour l'observation fine des territoires. Elles offrent trois bandes spectrales dans le visible (**rouge, vert, bleu**) auxquelles s'ajoute une quatrième bande dans le proche infrarouge (NIR), bien que cette dernière ne soit pas disponible dans nos données actuelles. Leur résolution spatiale remarquable de **0,5m** par pixel permet une détection très précise des objets au sol, ce qui est essentiel pour nos traitements d'analyse fine.

Ces images peuvent être obtenues soit via les **archives gratuites** (sous conditions d'accord), soit par acquisition à la demande, un service payant encadré par une licence Airbus©, avec des délais d'environ 6 à 8 mois par département.

Dans le cadre du cyclone Chido qui a frappé Mayotte en décembre 2024, un plan d'urgence a permis l'accès gratuit à une mosaïque d'images Pléiades post-cyclone couvrant l'île. Une **mosaïque** est une image composite constituée d'assemblages de prises de vues réalisées à différentes dates. Elle permet d'obtenir une couverture complète, avec un minimum de zones nuageuses et une meilleure homogénéité visuelle. Ce travail de reconstitution, mené par **l'Institut National de l'information géographique et forestière** (IGN), est crucial pour garantir des données exploitables, notamment dans le cadre de l'entraînement de modèles d'analyse automatique.

Mais Pléiades n'est pas notre seule source d'imagerie satellite. Les satellites **Sentinel-2**, développés par l'Agence spatiale européenne (ESA) dans le cadre du programme Copernicus, offrent une alternative open source, particulièrement intéressante pour les analyses à large échelle ou à forte fréquence temporelle. Contrairement à Pléiades, Sentinel-2 capte **treize bandes spectrales**, réparties entre le visible, le proche infrarouge (NIR) et l'infrarouge à ondes courtes (SWIR), ce qui

ouvre la voie à une multitude d'indices et d'analyses thématiques (comme la détection de la végétation, de l'humidité, ou des matériaux).

En revanche, la résolution spatiale de Sentinel-2 est moindre : selon les bandes, elle varie entre **10 m**, 20 m et 60 m, ce qui rend l'observation de petits objets ou de détails fins plus difficile. Néanmoins, ces images ont l'avantage d'être acquises automatiquement **tous les cinq jours**, garantissant une fréquence de revisite élevée et une mise à disposition régulière des données, y compris en période de crise.

Ainsi, les images Pléiades semblent être les plus adaptés pour de la détection de bâtiment dans les DROM.

- definition
- produits
- acquisition/livraison (partenariat IGN a promouvoir)

2.2.b Annotations

- Rappeler que c'est le plus important pour avoir de bons modèles
- Nous n'avons rien annoté manuellement => full automatique
- Rappel du coût d'annotation pour en avoir de qualité (en ETP ?)
- RIL (Insee),
- BDTOPO (better)
- CoSIA

peut etre rappeler la difficulté d'avoir des couples qui sont iso temporel

2.3 Evaluation

- IOU, Loss, zone de test manuel (calculer la taille des zones de test)

2.4 Inférence et mise à disposition des résultats

Une fois le modèle entraîné et validé, l'étape d'inférence consiste à appliquer ce modèle sur de nouvelles images satellites pour prédire la présence de bâtiments. Afin de lisser les prédictions et réduire le bruit, nous avons utilisé une moyenne mobile sur les sorties du modèle. Cette technique permet d'améliorer la cohérence spatiale des résultats, en atténuant les fausses détections isolées et en renforçant les zones de prédiction continue.

Pour faciliter l'accès aux résultats, nous avons mis en place une API dédiée. Cette API permet d'interroger le modèle à la demande, en soumettant de nouvelles images satellites et en récupérant les cartes de segmentation correspondantes. Cette approche garantit une intégration aisée dans des chaînes de traitement existantes et permet une exploitation rapide des prédictions, que ce soit pour des analyses ponctuelles ou pour un suivi opérationnel à grande échelle.

3 Résultats

3.1 Evaluation des performances

Metriques et images

inférence à grande échelle et robustesse des résultats (inférence avec fenetre mouvante) parler des ressources computationnelles requises

3.2 Mise à disposition pour les statisticiens

3.2.a Une application interactive

Pour les enquêteurs

3.2.b Vers de nouveaux indicateurs statistiques ?

Fichiers parquet d'évolution de bâti

4 Cas d’usage

Bibliographie

Chen, L.-C., & al. (2017). Rethinking Atrous Convolution for Semantic Image Segmentation. *arXiv*. http://arxiv.org/abs/1706.05587

Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., Dehghani, M., Minderer, M., Heigold, G., Gelly, S., & others. (2021). An Image Is Worth 16x16 Words: Transformers for Image Recognition at Scale. *arXiv preprint arXiv:2010.11929*. http://arxiv.org/abs/2010.11929

Kim, P. (2017). Convolutional Neural Network. In P. Kim (éd.), *MATLAB Deep Learning: With Machine Learning, Neural Networks and Artificial Intelligence* (p. 121-147). Apress. https://doi.org/10.1007/978-1-4842-2845-6_6

LeCun, Y., Boser, B., Denker, J. S., Henderson, D., Howard, R. E., Hubbard, W., & Jackel, L. D. (1989). Backpropagation Applied to Handwritten Zip Code Recognition. *Neural Computation*, *1*(4), 541-551. https://doi.org/10.1162/neco.1989.1.4.541

Liu, Z., Lin, Y., Cao, Y., Hu, H., Wei, Y., Zhang, Z., Lin, S., & Guo, B. (2021). *Swin Transformer: Hierarchical Vision Transformer using Shifted Windows*. https://arxiv.org/abs/2103.14030

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

Ravi, N., Gabeur, V., Hu, Y.-T., Hu, R., Ryali, C., Ma, T., Khedr, H., Rädle, R., Rolland, C., Gustafson, L., Mintun, E., Pan, J., Alwala, K. V., Carion, N., Wu, C.-Y., Girshick, R., Dollár, P., & Feichtenhofer, C. (2024). *SAM 2: Segment Anything in Images and Videos*. https://arxiv.org/abs/2408.00714

Ronneberger, O., Fischer, P., & Brox, T. (2015). U-Net: Convolutional Networks for Biomedical Image Segmentation. *arXiv:1505.04597 [cs]*. http://arxiv.org/abs/1505.04597

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., & Polosukhin, I. (2017). Attention Is All You Need. *Advances in Neural Information Processing Systems*, *30*. https://proceedings.neurips.cc/paper/2017/hash/3f5ee243547dee91fbd053c1c4a845aa-Abstract.html

Xie, E., Wang, W., Yuille, A. L., Anandkumar, A., & Alvarez, J. M. (2021). SegFormer: Simple and Efficient Design for Semantic Segmentation with Transformers. *arXiv preprint arXiv:2105.15203*. http://arxiv.org/abs/2105.15203

Zheng, S., Lu, J., Zhao, H., Zhu, X., Luo, Z., Wang, Y., Fu, Y., Feng, J., Xiang, T., Torr, P. H. S., & Zhang, L. (2021). *Rethinking Semantic Segmentation from a Sequence-to-Sequence Perspective with Transformers*. https://arxiv.org/abs/2012.15840