

REPUBLIQUE DU CAMEROUN
REPUBLIC OF CAMEROUN

Paix-travail-patrie

Peace -Work-Fatherland

UNIVERSITE DE DSCHANG
UNIVERSITY OF DSCHANG

Scholae Thesaurus Dschangensis Ibi Codium
BP96, Dschang (Cameroun)- tel./Fax (237)233451381
Website: <http://w.vw.univ-dschang.org>
E-mail : iut.fotsovicu@univ-dschang.org



INSTITUT UNIVERSITAIRE DE TECHNOLOGIE
FOTSO VICTOR DE BANDJOUN
FOTSO VICTOR UNIVERSITY
INSTITUTE OF TECHNOLOGY

Département de Génie
Informatique

Département of Computer
Engineering

BP 134, Bandjoun - Tél./Fax (237) 697 92 29
38/666 58 54 22

Website : <http://w.vw.univ-dschang.org>
E-mail : iut.fotsovicu@univ-dschang.org

PROJET DE FIN D'ETUDES

SYSTEME DE RECONNAISSANCE ET D'EXTRACTION DES FORMES IDENTIFIABLE A PARTIR DE DONNEES GEOGRAPHIQUES

Réalisé par :

FOKO ALEXIS GHISLAIN : CM-UDS-20IUT0296

NGUETSA MEINKONDEM DARIUS CHATELAIN : CM-UDS-20IUT1682

TSAGUE KENFACK JOYCE LAURA: CM-UDS-22IUT1158

En vue de l'obtention de

La Licence de Technologie

Parcours : **Informatique et Réseaux**

Mention : **Génie Informatique**

Option : **Concepteur Développeur Réseaux Internet (CDRI)**

Qualité et Sécurité Internet Réseaux (QSIR)

Sous l'encadrement académique de :

Dr FOTSING ERIC

Chargé de cours

M. SEVERIN KAKEU

Enseignant



Année Académique 2022-2023

DEDICACE

Nous dédions ce travail comme un témoignage d'affection, de respect, d'admiration : A nos parents, familles et nos différents encadreurs qui nous ont aidés moralement et matériellement. A tous nos amis de confiance qui se sont préoccupés d'améliorer nos connaissances grâce à leurs efforts et leurs conseils.

AVANT-PROPOS

L'**Institut Universitaire de Technologie FOTSO VICTOR de BANDJOUN** (en abrégé **IUT FV**) est un établissement de formation professionnel de l'université de Dschang. Il est né suite à la réforme universitaire de 1993, suivant l'arrêté présidentiel N°008/CAB/PR du 19 Janvier 1993.

L'IUT FV a pour vocation principale d'assurer la formation professionnelle des citoyens camerounais et étrangers, dans les domaines industriels et commerciaux. À ce titre, il fournit aux entreprises ou administrations des prestations de recherches appliquées, des services ou des formations professionnelles correspondant à leurs besoins. L'IUT FV de BANDJOUN a pour devoir de former les jeunes titulaires d'un baccalauréat ou équivalent, de les préparer aux examens nationaux du **BTS, DUT, LICENCE PROFESSIONNELLE et LICENCE TECHNOLOGIQUE**.

L'IUT FV DE BANDJOUN comporte plusieurs cycles de formation :

➤ **Diplôme Universitaire et Technologique (DUT), obtenu en deux ans dans les filières suivantes :**

- ✓ Génie Électrique (**GE**) ;
- ✓ Génie Informatique (**GI**) ;
- ✓ Génie de Télécommunication et Réseau (**GTR**) ;
- ✓ Génie Civil (**GC**) ;
- ✓ Maintenance Industrielle et Productique (**MIP**) ;
- ✓ Génie Thermique Energétique et Environnement (**GTEE**) ;
- ✓ Mécatronique Automobile (**MKA**) ;
- ✓ Gestions des Entreprises et des Administrations (**GEA**).

➤ **Brevet des techniciens supérieur (BTS) obtenu en deux ans dans les filières suivantes :**

- ✓ Maintenance des Systèmes Electroniques (**MSE**) ;
- ✓ Electrotechnique (**ET**) ;
- ✓ Froid et Climatisation (**FC**) ;

- ✓ Génie civil (**GC**) ;
- ✓ Gestion des Entreprises et administration (**GEA**) ;
- ✓ Banque Finance (**BF**) ;
- ✓ Assistant Manager (**ASM**) ;
- ✓ Marketing Commerce Vente (**MCV**) ;
- ✓ Collectivité Territoriale (**CT**).

➤ **Licences de technologiques (LT), Mention :**

- ✓ Gestion et Maintenance des Installations Électriques (**GMIE**) ;
- ✓ Informatique et Réseau (**IR**) parcours Concepteur et Développement Réseau Internet (**CDRI**) ;
- ✓ Génie Électrique (**LGE**) ;
- ✓ Génie Civil (**GC**) ;
- ✓ Maintenance Industrielle et Productique (**MIP**) ;
- ✓ Ingénierie des Réseaux et Télécommunications (**IRT**) ;
- ✓ Génie Géomatique (**GG**) ;
- ✓ Génie Thermique et Energétique (**GTE**).

➤ **Licence professionnelles (LP), Mention :**

- ✓ Marketing Manager Opérationnel (**MMO**) ;
- ✓ Banque Gestion des Relations Clientèle (**BGRC**) ;
- ✓ Gestion Administrative, Management des Organisations (**GAMO**) ;
- ✓ Gestion Comptable et Financière (**GCF**).

L'IUT-FV offre plusieurs types de formation :

- ❖ La formation Initiale ;
- ❖ La formation continue ;
- ❖ La formation à distance ;

REMERCIEMENTS

Ce travail de recherche est avant tout une œuvre collective. Avant de mettre à la disposition des lecteurs la substance de ce rapport, l'occasion nous est offerte ici d'adresser nos sincères remerciements à tous ceux qui ont aidés de près ou de loin à la réalisation de ce projet. On ne saurait citer les noms de façon exhaustive. C'est ainsi que nous pensons :

- ✓ Au Directeur de l'IUT-FV **Pr. TCHINDA RENE** pour son engagement à faire de ses étudiants les meilleurs de la nation de demain ;
- ✓ Au **Pr. TAYOU DJAMEGNI Clémentin**, chef de Département du Génie Informatique de l'IUT-FV de Bandjoun pour l'attention qu'il porte à l'égard de notre formation et sa disponibilité ;
- ✓ A nos encadreurs **Dr. FOTSING Éric** et **M. KAKEU Severin** pour leurs disponibilités, explications, remarques et critiques ; à qui nous disons merci pour tout ce qu'ils nous ont inculqués non seulement en termes de connaissances mais aussi en termes de valeurs ;
- ✓ Aux Enseignants du Département du Génie Informatique ainsi que le personnel de l'IUT-FV de Bandjoun pour le dévouement qu'ils accordent à notre formation ;
- ✓ A nos très chers parents pour leurs perpétuelles efforts et sacrifices dans le sens de notre réussite ;

LISTE DES ABREVIATIONS

RESUME

Dans le cadre de l'obtention d'une licence de technologie, chaque étudiant ou groupe d'étudiant doivent développer un projet de fin d'étude c'est dans cette optique que nous avons décidé de réaliser ce projet. Un système de reconnaissance et d'extraction de formes est un système capable de reconnaître une forme, de l'extraire et de l'identifier. Cette forme peut être contenu sur une image satellite. Ce système peut être utilisé dans des domaines tels que la gestion de l'environnement, la planification urbaine, la cartographie des ressources naturelles, la sécurité nationale en automatisant la tâche d'analyse des données géographiques. Un tel système peut aider à économiser du temps et des ressources humaines, tout en permettant une analyse plus précise et approfondie des données. Pour implémenter ce système nous avons utilisé des algorithmes de Machine Learning et de Deep Learning pour l'entraînement de notre modèle basé sur les technologies telles que Google Colab, WanDB, TensorFlow et Keras ainsi que le langage de programmation Python. Pour faciliter l'utilisation de notre modèle, nous l'avons intégré à une application web. Les principales difficultés que nous avons rencontrées sont le manque de ressources matériels pour l'apprentissage de notre modèle ainsi que sont déploiement sur une application web.

Mots clés :

- Machine Learning
- Deep Learning
- Segmentation d'images

ABSTRACT

As part of obtaining a technology license, each student or group of students must develop an end-of-study project. It is with this in mind that we have decided to carry out this project. A pattern recognition and extraction system is a system capable of recognizing a pattern, extracting it and identifying it. This shape can be contained on a satellite image. This system can be used in areas such as environmental management, urban planning, natural resource mapping, national security by automating the task of geographic data analysis. Such a system can help save time and human resources, while enabling more accurate and in-depth data analysis. To implement this system, we used Machine Learning and Deep Learning algorithms to train our model based on technologies such as Google Colab, WanDB, TensorFlow and Keras as well as the Python programming language. To facilitate the use of our model, we have integrated it into a web application. The main difficulties we encountered are the lack of material resources for learning our model as well as its deployment on a web application.

Key words:

- Machine Learning
- Deep Learning
- Image segmentation

SOMMAIRE

DEDICACE.....	i
AVANT-PROPOS	ii
REMERCIEMENTS	iv
LISTE DES ABREVIATIONS	v
RESUME.....	vi
ABSTRACT.....	vii
.....	viii
SOMMAIRE	viii
LISTES DES FIGURES ET TABLEAUX	x
INTRODUCTION.....	1
CHAPITRE I: PRESENTATION DU PROJET	2
I.1- PRESENTATION DE L'ARRETE DU PROJET	2
I.2- PROBLEMATIQUE	2
I.3- OBJECTIFS DU PROJET.....	3
I.4- LES ACTEURS CIBLES DU PROJET	3
I.5- RESULTATS ATTENDUS	4
I.6- ETAT DE L'ART	4
CHAPITRE II : ANALYSE FONCTIONNELLE ET TECNIQUE CONCEPTION.....	7
II.1- ANALYSE FONCTIONNELLE DU SYSTEME	7
II.2- CONCEPTION TECHNIQUE ET FONCTIONNELLE DU SYSTEME.....	7
CHAPITRE III: IMPLEMENTATION ET RESULTATS	17
III.1- TECHNOLOGIES UTILISEES	17
III.2- ENVIRONNEMENT DE DEVELOPPEMENT	20
IV- EVOLUTION DE L'APPRENTISSAGE DU MODELE	22
III.4 DEPLOIEMENT DE L'APPLICATION.....	25
III.5 RESULTATS OBTENUS.....	25
III.6 EVALUATION DES COUTS	27
CHAPITRE IV: DIFFILCUTES ET SUGGESTIONS.....	28
CONCLUSION	29
REFERENCES BIBLIOGRAPHIQUES	30
.....	31

TABLE DES MATIERES	31
--------------------------	----

LISTES DES FIGURES ET TABLEAUX

Figure 1: Exemple de dataset Scalaire	7
Figure 2: Image du dataset	8
Figure 3: Mask d'une image du dataset	9
Figure 4: Image en échelle de gris	10
Figure 5: Image en couleur.....	11
Figure 6: Images et mask au patch size	12
Figure 7: Classes identifiables.....	13
Figure 8: Encodage OneHot	14
Figure 9: Image après encodage OneHot	15
Figure 10: Evaluation de l'apprentissage du modèle	22
Figure 11: Evolution des pertes sur les données d'entrainement et de validation	24
Figure 12: Evolution du coefficient de Jaccard sur les données d'entrainement et de validation	24
Figure 13: Image de test : iut.....	25
Figure 14: Différents patchs obtenus.....	26
Figure 15: Résultat de la prédiction des différents patchs.....	26

INTRODUCTION

L'analyse de données géographiques est devenue essentielle dans de nombreux domaines tels que la gestion de l'environnement, le transport, l'urbanisme, la cartographie, l'agriculture, etc. Cependant, l'analyse manuelle de ces données peut être fastidieuse et coûteuse en temps et en ressources humaines. Par conséquent, les systèmes de reconnaissance et d'extraction des formes identifiables à partir de données géographiques ont été développés pour aider à automatiser cette tâche. Dans ce contexte, notre projet vise à développer un système de reconnaissance et d'extraction des formes identifiables à partir de données géographiques. Le système sera capable d'analyser des données géographiques brutes telles que des images satellite ou des cartes topographiques et d'extraire des informations importantes telles que les routes, les bâtiments, les cours d'eau, les forêts, etc. Le système sera basé sur des techniques de traitement d'image et de vision par ordinateur, telles que la segmentation d'image.

Plus qu'un Art, L'intelligence artificiel est une discipline jeune d'une soixantaine d'années, qui réunit des sciences, théories et technique (notamment logique, mathématique, statistiques, probabilités, neurobiologie computationnelle et informatique) et dont le but est de parvenir à faire imiter par une machine les capacités cognitives d'un être humain. Notre système de reconnaissance et d'extraction des formes identifiable est une forme d'intelligence artificiel.

Dans la suite de notre analyse nous expliciterons les différentes techniques, technologies et étapes qui entrent dans sa mise en place. Tout d'abord nous allons présenter une description de notre projet, puis une analyse fonctionnelle et technique de conception, ensuite l'implémentation de notre solution les résultats obtenus, enfin les difficultés rencontrées et suggestions.

CHAPITRE I: PRESENTATION DU PROJET

I.1- PRESENTATION DE L'ARRETE DU PROJET

L'analyse de données géographiques est devenue essentielle dans de nombreux domaines tels que la gestion de l'environnement, le transport, l'urbanisme, la cartographie, l'agriculture, etc. Cependant, l'analyse manuelle de ces données peut être fastidieuse et coûteuse en temps et en ressources humaines. Par conséquent, les systèmes de reconnaissance et d'extraction des formes identifiables à partir de données géographiques ont été développés pour aider à automatiser cette tâche.

Le projet vise à développer un système de reconnaissance et d'extraction des formes identifiables à partir de données géographiques. Ce système sera capable d'analyser des données géographiques brutes telles que des images satellite ou des cartes topographiques et d'extraire des informations importantes telles que les routes, les bâtiments, les cours d'eau, les forêts, etc.

Le système sera basé sur des techniques de traitement d'image et de vision par ordinateur, telles que la segmentation d'image, la reconnaissance de forme, l'apprentissage automatique et la classification. Des algorithmes seront développés pour identifier les formes géographiques spécifiques et extraire les informations pertinentes. Le projet comprendra également une évaluation de la performance du système en utilisant des données de test.

I.2- PROBLEMATIQUE

L'analyse manuelle des données géographiques est une activité très fastidieuse et coûteuse car les données géographiques sont souvent très complexes et peuvent inclure une grande quantité d'informations, la collecte de ces données peut être coûteuse et prendre beaucoup de temps. La nécessité d'analyses précises et détaillées de ces données, la nécessité de prise de décisions rapides et la gestion de zones géographiques vastes ou difficile d'accès sont les principaux problèmes liés au traitement de données géographiques.

I.3- OBJECTIFS DU PROJET

Ce projet a pour but d'aboutir à son terme à un système automatisé de traitement des données géographiques. Ces objectifs spécifiques sont :

- ✓ **Automatiser le traitement de données géographiques** pour réduire le temps et les coûts associés.
- ✓ **Améliorer la précision et la fiabilité des analyses géographiques**
- ✓ **Faciliter la prise de décisions** dans de nombreux domaines, tels que la gestion de l'environnement, la planification urbaine et la sécurité nationale.
- ✓ **Réduire les coûts de collecte de données géographiques** en utilisant des données existantes, telles que des images satellite ou des cartes topographiques.
- ✓ **Surmonter les défis liés à la gestion de zones géographiques vastes ou difficiles d'accès** en utilisant des données existantes pour identifier et extraire des informations pertinentes.
- ✓ **Améliorer la cartographie** : Le système peut aider à identifier et donc aider dans la cartographie de manière plus précise les caractéristiques géographiques telles que les routes, les bâtiments, les cours d'eau, les zones boisées, les zones humides, etc.
- ✓ **Aider dans la planification urbaine** : Les administrations locales peuvent utiliser ce système pour planifier l'utilisation des terres en fonction des caractéristiques géographiques de la région.
- ✓ **Aider dans la surveillance environnementale** : Les autorités environnementales peuvent utiliser ce système pour surveiller l'évolution des forêts, des zones humides et des autres écosystèmes.

I.4- LES ACTEURS CIBLES DU PROJET

Les principaux acteurs cibles de notre projet sont :

- ✓ **Les décideurs politiques** : qui peuvent utiliser ce système pour prendre des décisions en matière de planification urbaine, de gestion de l'environnement, de surveillance de la sécurité nationale.
- ✓ **Les professionnels de la cartographie et de la géomatique** : qui peuvent utiliser un tel système pour automatiser le traitement de données géographiques et améliorer la précision et la fiabilité de leurs analyses.

- ✓ **Les chercheurs et les scientifiques** : qui peuvent utiliser les résultats de ce système pour mener des études et des recherches dans des domaines tels que la biologie, l'écologie et la géologie.
- ✓ **Les organisations gouvernementales et non gouvernementales** : qui peuvent utiliser les résultats d'un tel système pour mener des activités de surveillance et de gestion de l'environnement.
- ✓ **Les entreprises** : qui peuvent utiliser les résultats de ce système pour prendre des décisions en matière de développement immobilier et de transport, des décisions relatives à la localisation de leurs activités.

I.5- RESULTATS ATTENDUS

Le principal résultat attendu à l'issue de ce projet est un système de traitement automatisé de données géographiques. Ce système sera donc capable de :

- ✓ Reconnaître une forme sur image satellite,
- ✓ Extraire cette forme,
- ✓ L'identifier,
- ✓ Et traiter les résultats obtenus.

I.6- ETAT DE L'ART

Le but essentiel de notre projet est de réaliser un système capable de reconnaître, d'extraire et d'identifier des formes identifiables sur des données géographiques. Notre application devra permettre d'extraire un ensemble d'informations qui pourront être exploitées dans de nombreux domaines tels que le transport, la cartographie et la gestion de l'environnement pour par exemple faire des études statistiques sur de vastes zones non accessibles (quantité de végétation, cours d'eaux, etc).

De tels systèmes sont en constante évolution en raison des avancées technologiques récentes dans le domaine de l'intelligence artificielle et de la vision par ordinateur.

I.6.1- Techniques et approches utilisées

Voici quelques-unes des techniques et des approches actuelles utilisées dans ce domaine

- ❖ **Réseaux de neurones convolutifs (CNN) :** Les CNN sont actuellement l'une des techniques les plus populaires pour la reconnaissance et l'extraction de formes géographiques à partir d'images. Les CNN sont des réseaux de neurones profonds qui apprennent à détecter automatiquement les caractéristiques pertinentes des images, telles que les formes, les couleurs, les textures, etc. Les CNN ont été utilisés avec succès pour la détection de bâtiments, la classification de l'occupation des sols, la détection de cours d'eau, la segmentation d'images, etc.
- ❖ **Traitement d'images satellitaires :** Les images satellitaires sont souvent utilisées pour la reconnaissance et l'extraction de formes géographiques en raison de leur couverture géographique globale et de leur haute résolution spatiale. Les techniques de traitement d'images peuvent être utilisées pour extraire des informations à partir de ces images, telles que l'occupation des sols, la végétation, les cours d'eau, les routes, les bâtiments, etc. Les techniques de traitement d'images courantes comprennent la classification d'images, la segmentation d'images, l'analyse de texture, l'analyse de forme, etc.
- ❖ **Analyse de données LiDAR :** Les données LiDAR sont des données de télédétection qui utilisent des lasers pour mesurer la distance entre l'émetteur et la surface terrestre. Les analyses de données LiDAR peuvent être utilisées pour générer des modèles 3D de la surface terrestre, ainsi que pour extraire des informations sur les caractéristiques du terrain, telles que la topographie, la densité de la végétation, les cours d'eau, etc. Les techniques courantes d'analyse de données LiDAR comprennent la segmentation de nuages de points, la classification d'objets, la détection de changements, etc.
- ❖ **Systèmes d'informations géographiques (SIG) :** Les SIG sont des outils logiciels utilisés pour stocker, gérer, analyser et visualiser des données géographiques. Les SIG peuvent être utilisés pour intégrer des données géographiques provenant de différentes sources, telles que des images satellite, des données LiDAR, des cartes topographiques, etc., afin de faciliter la reconnaissance et l'extraction de formes géographiques. Les SIG peuvent également être utilisés pour effectuer des analyses spatiales, telles que l'interpolation de données, la modélisation de terrain, etc.
- ❖ **Techniques de segmentation d'images :** Les techniques de segmentation d'images sont utilisées pour diviser une image en régions homogènes en termes de texture, de couleur, de forme, etc. Ces régions peuvent ensuite être utilisées pour extraire des informations sur les caractéristiques de l'image, telles que les bâtiments, les routes, les cours d'eau, etc. Les techniques courantes de segmentation d'images comprennent la segmentation

basée sur les seuils, la segmentation par croissance de régions, la segmentation par régions actives, etc.

- ❖ **Deep Learning** : Le Deep Learning est une sous-catégorie de l'apprentissage automatique qui utilise des réseaux de neurones profonds pour apprendre automatiquement à partir de données. Le Deep Learning a été utilisé avec succès pour la reconnaissance et l'extraction de formes géographiques à partir de données d'images et de nuages de points LiDAR. Les techniques courantes de Deep Learning comprennent les réseaux de neurones convolutifs (CNN), les réseaux de neurones récurrents (RNN), les réseaux de neurones générateurs adversaires (GAN), etc.

Ces techniques sont en constante évolution et continuent de s'améliorer grâce aux avancées technologiques récentes.

CHAPITRE II : ANALYSE FONCTIONNELLE ET TECNIQUE CONCEPTION

II.1- ANALYSE FONCTIONNELLE DU SYSTEME

II.2- CONCEPTION TECHNIQUE ET FONCTIONNELLE DU SYSTEME

II.2.1- Acquisition d'un dataset

Afin de réaliser un modèle de segmentation sémantique d'image il est nécessaire d'avoir un dataset assez particulier.

- Dataset en Machine Learning Classique

Dans d'autre problème classique de **supervised Learning** (apprentissage supervisé) le **dataset** est sous forme de matrice contenant des scalaires. Ces scalaires sont de deux catégories on a en premier lieu les **features** puis la **target**.

Les features désignent ici un ensemble de valeurs qui seront pris en paramètres dont dépendent les valeurs de notre **target**

Les targets désignent les valeurs en résultantes des features.

y	x_1	x_2	x_3
Prix	Surface m2	N chambres	Qualité
€ 313,000.00	124	3	1.5
€ 2,384,000.00	339	5	2.5
€ 342,000.00	179	3	2
€ 420,000.00	186	3	2.25
€ 550,000.00	180	4	2.5
€ 490,000.00	82	2	1
€ 335,000.00	125	2	2

Figure 1: Exemple de dataset Scalaire

La figure ci-dessus nous montre l'exemple d'un dataset. Les valeurs de **X** sont nos **features** et les valeurs de **Y** sont nos **targets**. Le but de ce type de modèle est d'étudier la relation entre les **features** et les **targets** afin de fournir une fonction sous la forme $f(x_1, x_2, \dots, x_n) = y$ qui permettra de prédire les valeurs de **Y** en sorties

- Dataset en Deep Learning

Le Deep Learning vient comment une sorte d'extension au Machine Learning. Il permet de faire les mêmes opérations que celle faites en Machine Learning mais sur une quantité significativement plus importante de données. Dans le cadre de notre projet notre dataset sera constituer de ce qu'on appelle **les tile images** et **les tile masks** comme illustre la figure ci-dessous



Figure 2: Image du dataset



Figure 3: Mask d'une image du dataset

- Une **tile image** : celle-ci désigne un morceau d'une image satellitaire de type **matrice** qui sera traiter afin de servir à entrainer notre modèle. Une **tile image** est en réalité un morceau d'une image satellitaire beaucoup plus grande cette technique consistant à découper les images satellitaires est nécessaire afin de faciliter le stockage, le traitement et la diffusion de ce type d'image.

- Une **mask image** : chaque tile image du dataset possède sa **mask image**. Cette image est dite « masquer » ou « filtrer » le but de filtre ici est d'éliminer les zones de l'images qui ne sont pas pertinentes afin de ne conserver que nos zones d'intérêts. Chaque zone d'intérêts est représentée par une couleur et en ce qui concerne ce projet nous en avons 06 au total

L'élaboration d'un telle dataset est assez complexe car il est nécessaire de photographier la zone en question avec l'aide d'un satellite ou d'un drone et ensuite il faut pouvoir extraire les zones d'intérêt ce qui est une tâche assez fastidieuse.

Notre dataset a été élaborer à partir de données recueillies dans la ville de **Dubaï au Emirates Arabes Unies** et qui est mis à la disposition de tous sur le site web **Kaggle.com**.

II.2.2- Traitement des images

II.2.2.a- Représentation d'une image en mémoire

Pour l'élaboration de modèle de segmentation sémantique d'image il est important de premièrement comprendre comment une image est stockée en mémoire. Il faut d'abord que l'on distingue que type d'image : Les images en échelle de gris et les images en couleur.

Dans le cas particulier des images en échelle de gris, chaque **pixel** est stocké sur un **octet** c'est-à-dire une valeur entre 0 et 255. La bibliothèque **opencv** de python nous permet de lire des images sous ce format avec la commande « `cv2.imread('/path/image.jpg', 0)` » comme l'illustre la figure suivante



- Dimension : (644, 797)
- Taille : 513268 pixels

Figure 4: Image en échelle de gris

Ce type d'image est représentée dans la mémoire de l'ordinateur sous forme d'une **matrice à deux dimensions**.

Nous avons aussi les images de couleur. La couleur ici est formée d'une association de valeurs de trois couleurs fondamentales : le **rouge**, le **vert** et le **bleu** (RGB). Chaque pixel ici au lieu d'avoir une simple valeur est constitué d'un vecteur de 03 valeurs donc chaque valeur est représentée sur 1 octet. C'est typiquement ce type d'images que sera utilisé pour l'entraînement de notre modèle



- Dimension : (644, 797, 3)
- Taille : 1539804 pixels

Figure 5: Image en couleur

II.2.2.b- Preprocessing des images

Le preprocessing des images est une étape primordiale pour la conception d'un modèle de segmentation sémantique d'images. Le but essentiel de cette étape est de pouvoir préparer les images à être utilisées pour entraîner notre modèle. Les images qui présentent dans le dataset n'ont pas toujours les mêmes dimensions or pour entraîner un modèle de ce type il est nécessaire de toutes les données en entrée soit de la même dimension appelé **patch size**.

➤ Fixation du patch size

Le **patch size** est la valeur de dimension que nous utiliserons pour notre modèle cette taille est généralement de 256×256 pixels mais elle dépend grandement de la résolution de l'image.

Pour des images de hautes résolution le patch size peut aller de 16×16 à 64×64 pixels

Pour des images de moyennes résolutions le patch size peut aller de 128×128 à 512×512 pixels

Il est important de noter que plus la résolution de l'image satellitaire est grande plus la taille des pixels sera réduite et par conséquent une petite zone de l'image contiendra une grande quantité d'information c'est cela qui explique le fait que la taille du patch size est inversement proportionnelle à la résolution de l'image.

De façon général la segmentation sémantique permet d'attribuer une classe à chaque pixel de l'image c'est pour qu'il est nécessaire que le patch size soit aussi réduit afin de faciliter la classification de pixels. De plus il est nécessaire de trouver un certain équilibre en choisissant la valeur du patch size. Une patch size trop petit cela va grandement augmenter la charge de calcul et réduire la vitesse de traitement et choisir un patch size trop faible causera un manque de précision du modèle

Il est aussi important de noter à la fois les **tile images** et les **mask images**

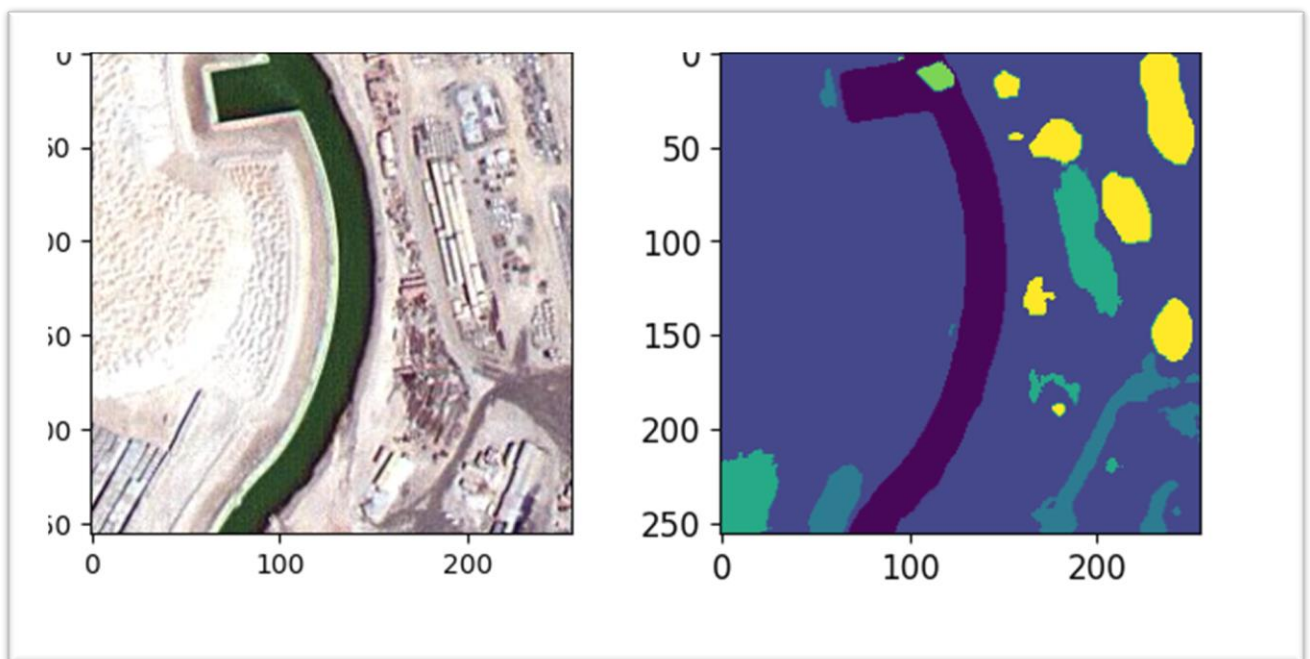


Figure 6: Images et mask au patch size

La figure ci-dessus nous montre le résultat que nous avons obtenu après avoir découpé à la fois une **tile image** et sa **mask image**. Ce tracé a été obtenu avec la bibliothèque **matplotlib** de python et nous montre parfaitement que chaque image est de la taille du patch size c'est-à-dire de 256×256 pixels.

➤ Classification des pixels

Cette opération les uniquement effectuer sur le **tile mask** de notre dataset. Lors du preprocessing des données il est crucial de pouvoir identifier le **land marks** (zones d'intérêt de notre modèle ou classe d'objet) et associer **chaque pixel de chaque tile image à une classe d'objet** dans le cadre de ce projet nous identifierons 06 classe d'objet avec chacun ayant sa propre couleur en hexadécimale

```

'classes': [
  {"title": "Water", "shape": "polygon", "color": "#50E3C2", "geometry_config": {}},
  {"title": "Land (unpaved area)", "shape": "polygon", "color": "#F5A623", "geometry_config": {}},
  {"title": "Road", "shape": "polygon", "color": "#DE597F", "geometry_config": {}},
  {"title": "Building", "shape": "polygon", "color": "#D0021B", "geometry_config": {}},
  {"title": "Vegetation", "shape": "polygon", "color": "#417505", "geometry_config": {}},
  {"title": "Unlabeled", "shape": "polygon", "color": "#9B9B9B", "geometry_config": {}}, "tags": []

```

Figure 7: Classes identifiables

Un fois les classe identifier nous allons créer une nouvelle matrice qui contient **la valeur de l'indice de la classe de chaque pixel dans le fichier ci-dessus** pour bien comprendre cette étape on va prendre l'exemple d'une image M

Notre image M est une image en couleur ce qui signifie qu'elle peut être assimilée à un **tenseur d'ordre 3** (vecteur à 3 dimensions) notre image M est alors constituée de N pixel. Chaque pixel P est caractérisé par $P(l, c, v)$

l = indice sur ligne

c = indice colonne

v = vecteur rgb (r, g, b)

Si notre pixel fait par exemple partie de la classe « Water » une autre matrice M' sera générée et **aura l'indice de la classe de ce pixel P à cette position**

Ainsi à la fin de cette étape si on avait N images au départ, on aura également N matrices chacune caractérisant chaque image avec l'indice à la place de chaque pixel.

➤ Encodage OneHot Division du dataset

A cette étape d'avancement de réalisation notre dataset se divise en trois parties grandes parties

La partie A former de nos **tile image**

La partie B former de nos **tile mask**

Enfin la partie C constituer de nos matrices contenant la classe de chaque pixel pour chaque image de la partie B du dataset

- **Encodage OneHot**

L'encodage OneHot est une méthode utilisée lors de la segmentation sémantique d'image afin de pouvoir plus facilement stocker la partie C du dataset en mémoire. En effet pour que la valeur de l'indice du pixel n'est pas d'incidence significatif lors de l'entraînement du modèle (par exemple la classe « water » à pour valeur « 0 » et la classe « vegetation » à pour valeur « 4 ») il est nécessaire de trouver une méthode qui permettra non seulement d'éviter le problème énoncer précédemment mais en plus rendra la tâche de traitement de l'ordinateur plus simple à réaliser c'est donc là l'importance de cette méthode d'encodage son principe de fonctionnement est le suivant :

Premièrement nous allons identifier le nombre total de classe dans notre dataset (06 dans notre cas particulier) puis nous allons créer un vecteur contenant 06 valeurs soit à « 0 » ou à « 1 » **la seule valeur à « 1 » de ce vecteur sera celle qui correspond à l'indice de la classe qui lui est associée dans notre dataset.**

Par exemple si nous avons un pixel P qui appartient à la classe « vegetation » par exemple (indice 4) ce pixel sera représenté par un vecteur V sous la forme V [0, 0, 0, 0, 1, 0]. Cet encodage est rendu possible grâce à la fonction **to_categorical** de la bibliothèque **tensorflow.keras.utils** illustrer comme suit

```
In [41]: labels_categorical_dataset = to_categorical(labels, num_classes=total_classes)
```

Figure 8: Encodage OneHot

Cette fonction prend 02 paramètre : le premier est le nombre total de classe présent de notre dataset (num_class) et la second est le dataset précédent contenant les valeurs de l'indice de chaque classe (labels). A la fin cette étape chaque image sera sous la forme suivante :

```
In [42]: labels_categorical_dataset[0]
Out[42]: array([[0., 1., 0., 0., 0., 0.],
                [0., 1., 0., 0., 0., 0.],
                [0., 1., 0., 0., 0., 0.],
                ...,
                [0., 1., 0., 0., 0., 0.],
                [0., 1., 0., 0., 0., 0.],
                [0., 1., 0., 0., 0., 0.]],
               [[0., 1., 0., 0., 0., 0.],
                [0., 1., 0., 0., 0., 0.],
                [0., 1., 0., 0., 0., 0.],
                ...,
                [0., 1., 0., 0., 0., 0.],
                [0., 1., 0., 0., 0., 0.],
                [0., 1., 0., 0., 0., 0.]],
               [[0., 1., 0., 0., 0., 0.],
                [0., 1., 0., 0., 0., 0.],
                [0., 1., 0., 0., 0., 0.],
                ...,
                [0., 1., 0., 0., 0., 0.],
                [0., 1., 0., 0., 0., 0.],
                [0., 1., 0., 0., 0., 0.]],
               ...]
```

Figure 9: Image après encodage OneHot

- **Division du dataset**

Dans le Machine Learning en général il est toujours nécessaire de séparer le dataset de départ en 04 parties comme suit :

X_train : c'est la partie des **tile image du dataset** qui va être utiliser pour entrainer notre modèle

Y_train : c'est la partie des **tile mask** qui sera utiliser pour entrainer notre modèle

X_test : il s'agit de la partie des **tile image** sur lesquelles nous allons tester notre modèle

Y_test : qui représente ici nos données de validation c'est à partir de ces **tile mask** que nous allons voir à quel point notre modèle épouse la réalité et est précis

Il existe aussi une fonction qui nous permet de réaliser cette séparation du dataset du non de **train_test_split** de la bibliothèque **sklearn.model_selection**. Cette opération est réalisée comme suit :

```
x_train, x_test, y_train, y_test
    = train_test_split(master_training_dataset, labels_categorical_dataset, test_size
    = 0.15, random_state = 100)
```

La fonction **train_test_split** prend ici 04 paramètres

master_training_dataset : représente notre dataset contenant les **tile image**

labels_categorical_dataset : représente notre dataset contenant les **tile mask**

test size : représente la proportion entre les images contenues dans la *X_test* et dans la *X_train*

labels_categorical_dataset : indique que les images doivent être prise de façon aléatoire dans le dataset

Une fois notre dataset diviser nous sommes désormais prêts à entamer l'entraînement du modèle.

II.2.1.d- Entraînement du modèle

CHAPITRE III: IMPLEMENTATION ET RESULTATS

Ce chapitre a pour objectif majeur de présenter le produit final. C'est la phase d'entraînement de notre modèle et son déploiement sur une application web en utilisant des technologies spécifiques. Il est composé de trois parties, la première partie présente les différentes technologies utilisées, la seconde l'environnement de développement et la dernière partie concerne les principaux résultats obtenus.

III.1- TECHNOLOGIES UTILISEES

Dans le cadre de la réalisation de ce projet nous avons utilisés les technologies telles que TensorFlow et Keras.

III.1.1- TensorFlow

TensorFlow est une bibliothèque open source de Google pour l'apprentissage automatique et le traitement des données. Elle est utilisée pour la création et l'entraînement de modèles de Machine Learning, y compris les réseaux de neurones convolutifs (CNN), les réseaux de neurones récurrents (RNN) et les réseaux de neurones profonds (DNN). Elle nous a permis de définir l'architecture, d'entraîner, d'évaluer, de sauvegarder et de charger notre modèle de réseaux de neurones convolutif de segmentation d'images.

Ces principales caractéristiques sont les suivantes :

- **Structure de données** : TensorFlow utilise des tenseurs (ou tableaux multidimensionnels) pour représenter les données. Les tenseurs peuvent avoir n'importe quel nombre de dimensions et peuvent contenir des valeurs de différents types de données, tels que des nombres entiers, des nombres à virgule flottante et des chaînes de caractères.
- **Graphes de calcul** : TensorFlow utilise un modèle de programmation basé sur des graphes de calcul. Les graphes de calcul représentent les opérations mathématiques qui sont effectuées sur les tenseurs pour produire des résultats. Les graphes de calcul sont construits en créant des nœuds pour les tenseurs et les opérations, puis en reliant les nœuds pour former le graphe.

- **Fonctionnement distribué** : TensorFlow prend en charge le calcul distribué, ce qui permet de distribuer le traitement sur plusieurs unités de traitement. Cela permet d'accélérer le temps de traitement des tâches d'apprentissage automatique et de gérer des ensembles de données volumineux.
- **API** : TensorFlow fournit une API pour plusieurs langages de programmation, notamment Python, C++, Java, Go et Swift. Les API permettent aux développeurs de créer des modèles de machine learning, de les entraîner et de les déployer.
- **Bibliothèques supplémentaires** : TensorFlow dispose d'un écosystème de bibliothèques supplémentaires pour des tâches spécifiques telles que la vision par ordinateur (TensorFlow Object Detection API), le traitement du langage naturel (TensorFlow Text) et l'optimisation (TensorFlow Probability).
- **Compatibilité avec d'autres outils** : TensorFlow est compatible avec d'autres outils de machine learning tels que Keras, une bibliothèque de haut niveau pour la création de modèles de machine learning, et TensorFlow Lite, une version de TensorFlow conçue pour les appareils mobiles et l'IoT.

III.1.2- Keras

Keras est une bibliothèque open source de haut niveau pour l'apprentissage automatique qui fournit une interface simple pour la construction de modèles de machine learning. Elle est compatible avec plusieurs moteurs de calcul de base, notamment TensorFlow et Theano, et est souvent utilisée pour les tâches de vision par ordinateur, de traitement du langage naturel et de prédiction de séries temporelles. Grâce à son API couplé à TensorFlow nous l'avons utilisé pour le prétraitement de nos données d'apprentissage c'est-à-dire la normalisation des données, la mise en forme des images et la transformation de données non structurées en données structurées ; la construction de notre modèle en utilisant ses fonctions telles que : **MaxPooling2D** pour ajouter des couches de pooling, et **Dense** pour ajouter des couches entièrement connectées. Elle nous a également aidé à configurer notre modèle en définissant le nombres de couches, la taille des filtres de convolution, le taux d'apprentissage et les fonctions d'activations ; puis d'entraîner notre modèle grâce a ses fonctions telles que **compile** pour compiler le modèle avec une fonction de perte et une fonction d'optimisation, **fit** pour entraîner le modèle sur les données d'entraînement ; enfin d'évaluer notre modèle en utilisant des métriques telles que l'exactitude, la précision et le rappel.

Ces principales caractéristiques sont les suivantes :

- **Interface utilisateur simple** : Keras fournit une interface utilisateur simple et cohérente pour la construction de modèles de machine learning. Elle permet aux utilisateurs de créer des modèles de machine learning en quelques lignes de code et fournit des fonctions pré-construites pour les tâches courantes telles que la classification d'images, la segmentation sémantique et la reconnaissance vocale.
- **Modularité** : Keras est conçu pour être modulaire, ce qui permet aux utilisateurs de construire des modèles de machine learning en combinant différents blocs de construction. Les blocs de construction peuvent être des couches de réseau de neurones, des fonctions d'activation, des fonctions de perte et des optimiseurs.
- **Compatibilité avec plusieurs moteurs de calcul** : Keras est compatible avec plusieurs moteurs de calcul de base, notamment TensorFlow, Theano et CNTK. Les utilisateurs peuvent choisir le moteur de calcul qui convient le mieux à leur projet et Keras s'occupera de la communication avec le moteur de calcul choisi.
- **Traitement de données intégré** : Keras fournit des outils intégrés pour le prétraitement des données, tels que la normalisation des données, l'augmentation des données et la division des données en ensembles d'entraînement, de validation et de test.
- **Fonctionnalités avancées** : Keras dispose de nombreuses fonctionnalités avancées pour la construction de modèles de machine learning, notamment le transfert de connaissances, la régularisation, la normalisation de lot et le dropout.
- **Compatibilité avec d'autres bibliothèques** : Keras est compatible avec d'autres bibliothèques de machine learning telles que TensorFlow, ce qui permet aux utilisateurs de bénéficier des fonctionnalités avancées de ces bibliothèques tout en utilisant l'interface utilisateur simple et cohérente de Keras.

III.1.3 WanDB

WandB, abréviation de "Weights and Biases", est une plateforme de gestion expérimentale pour la science des données et l'apprentissage automatique. La plateforme permet aux utilisateurs de suivre, visualiser et partager les résultats de leurs expériences d'apprentissage automatique de manière collaborative. Elle nous a permis de visualiser les résultats d'apprentissage de notre modèle notamment l'évolution du coefficient de jaccard, de la précision et la perte sur les données d'apprentissage et de validation.

WandB offre une variété de fonctionnalités notamment :

- **Suivi des expériences** : WandB permet aux utilisateurs de suivre les métriques et les hyperparamètres de leurs expériences d'apprentissage automatique en temps réel. Les

utilisateurs peuvent enregistrer des métriques telles que l'exactitude, la perte, le temps d'entraînement, etc., ainsi que les hyperparamètres tels que le taux d'apprentissage, la taille du lot, la fonction d'activation, etc.

- **Visualisation des expériences** : Les utilisateurs peuvent visualiser leurs expériences d'apprentissage automatique à l'aide de graphiques interactifs, de tableaux de bord et de diagrammes. WandB offre également des outils de visualisation pour les données d'entrée et de sortie, ainsi que pour les modèles.
- **Collaboration** : WandB permet aux utilisateurs de partager leurs expériences avec des collègues et de collaborer sur des projets. Les utilisateurs peuvent ajouter des commentaires, des notes et des marques à leurs expériences, et également les partager avec d'autres utilisateurs.
- **Intégration avec les outils ML courants** : WandB s'intègre facilement avec les outils d'apprentissage automatique courants tels que TensorFlow, PyTorch, Keras, Scikit-learn, etc. Les utilisateurs peuvent ainsi facilement suivre leurs expériences et visualiser les résultats dans une interface unifiée.
- **Automatisation des workflows** : WandB offre des fonctionnalités d'automatisation des workflows pour faciliter la gestion des expériences d'apprentissage automatique. Les utilisateurs peuvent automatiser des tâches telles que l'enregistrement des résultats, la visualisation des données, le déploiement de modèles, etc.

III.2- ENVIRONNEMENT DE DEVELOPPEMENT

Le développement de notre système a été subdivisé en deux grandes parties à savoir l'entraînement du modèle et son déploiement sur une application web. Tout d'abord nous allons vous présenter l'environnement développement utilise pour l'entraîne de notre modèle, ensuite celui utilisé pour son déploiement sur une application web.

III.2.1 Entraînement du modèle

Nous avons utilisé la version premium de la plateforme **Google Colab** pour l'entraînement de notre modèle qui est une plateforme de développement de machine learning basée sur le cloud qui permet aux utilisateurs de créer, de partager et de collaborer sur des projets de machine learning. Ces principales fonctionnalités sont :

Environnement de développement : Google Colab fournit un environnement de développement basé sur Jupyter Notebook qui permet aux utilisateurs d'écrire et d'exécuter du

code Python dans leur navigateur web. Il est équipé de toutes les bibliothèques couramment utilisées en machine learning telles que TensorFlow, Keras, PyTorch, etc.

- **Accès au GPU et TPU** : Google Colab permet aux utilisateurs d'accéder gratuitement à des ressources de calcul haute performance telles que les GPU (unités de traitement graphique) et les TPU (unités de traitement tensoriel). Cela permet aux utilisateurs de former leurs modèles de machine learning plus rapidement et plus efficacement.
- **Stockage de données** : Les utilisateurs peuvent stocker leurs données sur Google Drive, Google Cloud Storage ou localement sur leur ordinateur. Colab permet également de charger facilement des ensembles de données à partir de sources publiques telles que Kaggle et GitHub.
- **Collaboration** : Google Colab permet aux utilisateurs de collaborer sur des projets de machine learning en temps réel. Les utilisateurs peuvent partager des notebooks avec d'autres personnes et travailler ensemble sur le même notebook. Les utilisateurs peuvent également commenter et discuter du code à l'intérieur du notebook.
- **Sauvegarde des notebooks** : Les notebooks créés dans Google Colab sont automatiquement sauvegardés sur Google Drive. Les utilisateurs peuvent également télécharger leur notebook au format **.ipynb** pour le sauvegarder localement ou le partager avec d'autres personnes.

III.2.1.a- Environnement matériel

Pour des raisons de performances nous avons utilisé la version premium de Google Colab qui donne l'accès aux ressources matérielles suivantes :

- RAM : 25 Go
- CPU : Xeon 2,30 GHz à 2,80 GHz avec 8 cœurs virtuels
- Disque : 166.8 Go
- GPU: NVIDIA A100 Tensor Core, NVIDIA V100 Tensor Core.

III.2.1.b- Environnement logiciel

Google Colab Pro fournit un environnement logiciel complet et puissant pour l'apprentissage automatique et l'analyse de données, avec de nombreuses bibliothèques et outils populaires préinstallés telles que :

- **Python** : Google Colab Pro prend en charge Python 2 et Python 3

- **Bibliothèques de données et d'apprentissage automatique** : Google Colab Pro est préconfiguré avec de nombreuses bibliothèques de données et d'apprentissage automatique, notamment NumPy, Pandas, Matplotlib, TensorFlow, Keras, PyTorch, Scikit-learn, et bien d'autres.
- **Outils de développement** : Google Colab Pro propose des outils de développement pratiques tels que **Jupyter Notebook** qui a été notre principal éditeur, qui permettent de créer des notebooks interactifs, ainsi que des éditeurs de texte et des terminaux pour travailler avec du code.

III.2.2 Déploiement du modèle sur une application web

IV- EVOLUTION DE L'APPRENTISSAGE DU MODELE

La figure suivante illustre les différentes courbes d'évolution du coefficient de Jaccard, de la précision et la perte sur les données d'apprentissage et de validation issu de la plateforme WanDB :

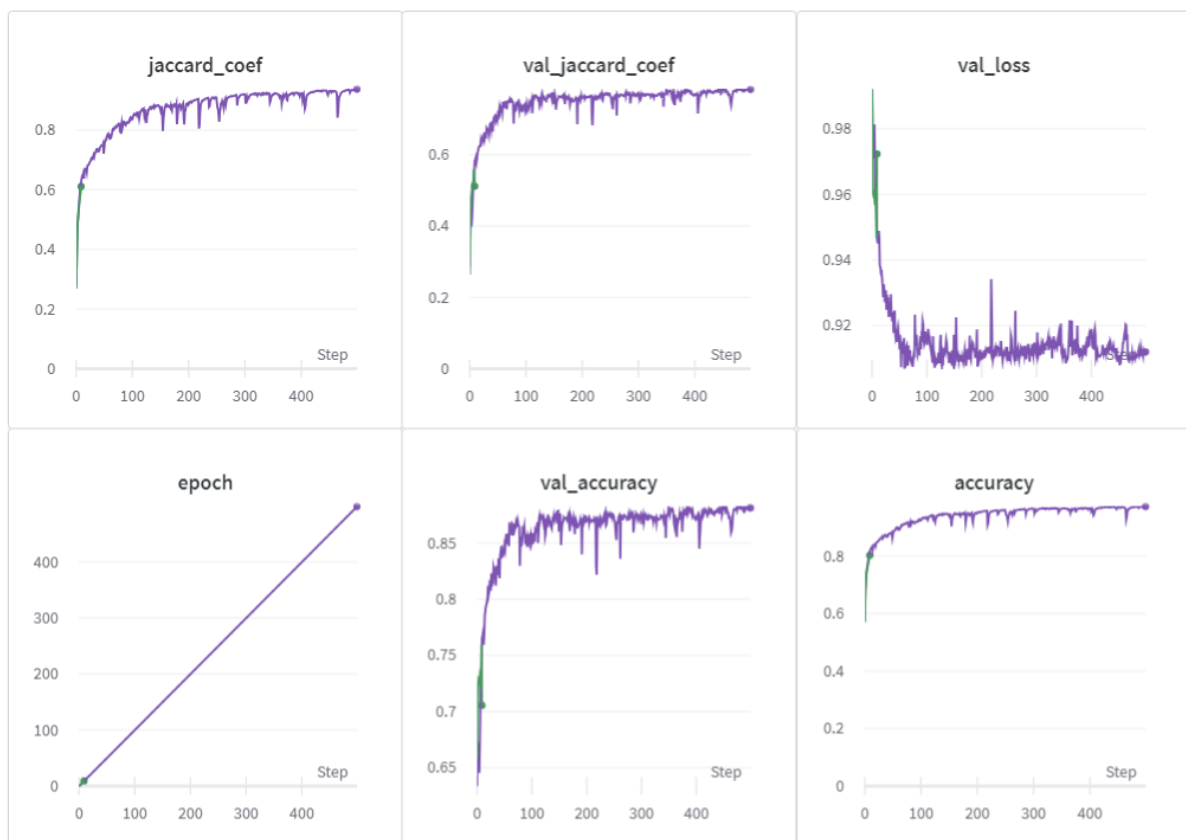


Figure 10: Evaluation de l'apprentissage du modèle

La première courbe « **jaccard_coef** » représente l'évolution du coefficient de Jaccard ou coefficient d'IoU sur les données d'entraînement. Le coefficient de Jaccard est utilisé pour évaluer la qualité de la segmentation en comparant la zone de chevauchement entre la segmentation prédite et la segmentation réelle.

La deuxième courbe « **val_jaccard_coef** » représente l'évolution du coefficient de Jaccard sur les données de validation.

La troisième courbe « **val_loss** » représente la perte sur les données de validation. C'est une mesure utilisée pour évaluer la différence entre les prédictions du modèle et les valeurs réelles de la sortie souhaitée.

La quatrième courbe « **epoch** » représente l'évolution du nombre d'époques.

La sixième courbe « **val_accuracy** » représente l'évolution de la précision sur les données de validation. C'est une mesure qui permet d'évaluer la performance du modèle en termes de précision des prédictions, c'est-à-dire le pourcentage de prédictions correctes sur l'ensemble des prédictions effectuées sur les données de validation.

La dernière courbe « **accuracy** » représente l'évolution de la précision sur les données d'entraînement. C'est une mesure qui permet d'évaluer la performance du modèle en termes de précision des prédictions, c'est-à-dire le pourcentage de prédictions correctes sur l'ensemble des prédictions effectuées sur les données d'entraînement.

La figure suivante illustre l'évolution des pertes sur les données d'entraînement et les données de validation :

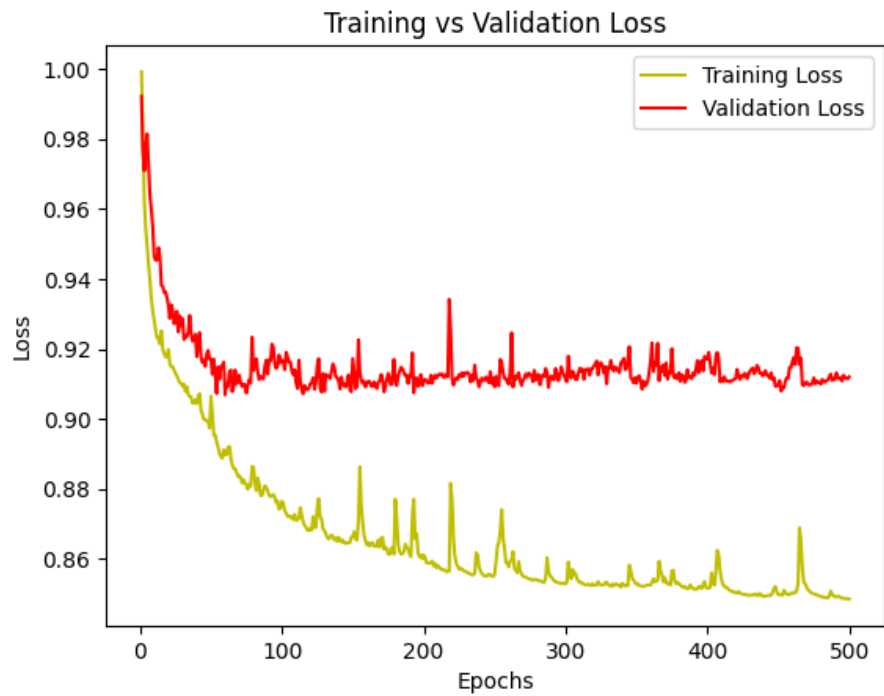


Figure 11: Evolution des pertes sur les données d'entrainement et de validation

La figure suivante illustre l'évolution du coefficient de Jaccard sur les données d'entrainement et les données de validation :

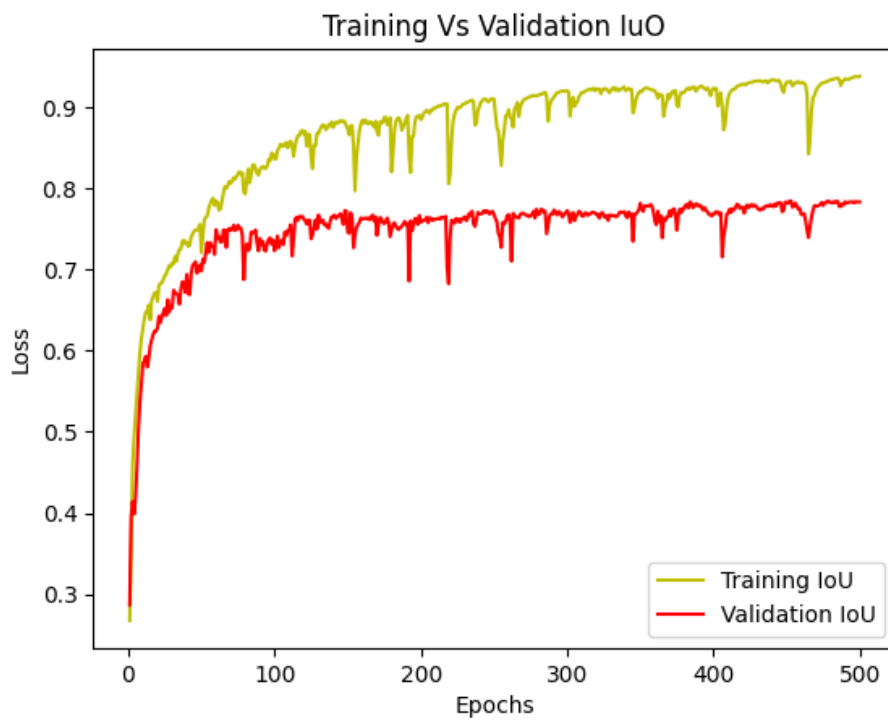


Figure 12: Evolution du coefficient de Jaccard sur les données d'entrainement et de validation

III.4 DEPLOIEMENT DE L'APPLICATION

III.5 RESULTATS OBTENUS

III.5.1- Test du model

Après l'apprentissage de notre model nous avons effectues des tests sur l'image suivante :



Figure 13: Image de test : iut

Tout d'abord l'image est rognée pour avoir une taille qui est multiple de 256x256 et divisé en patch de petites images de taille 256x256. La figure suivante illustre le différent patch obtenu a l'issu de cette étape :

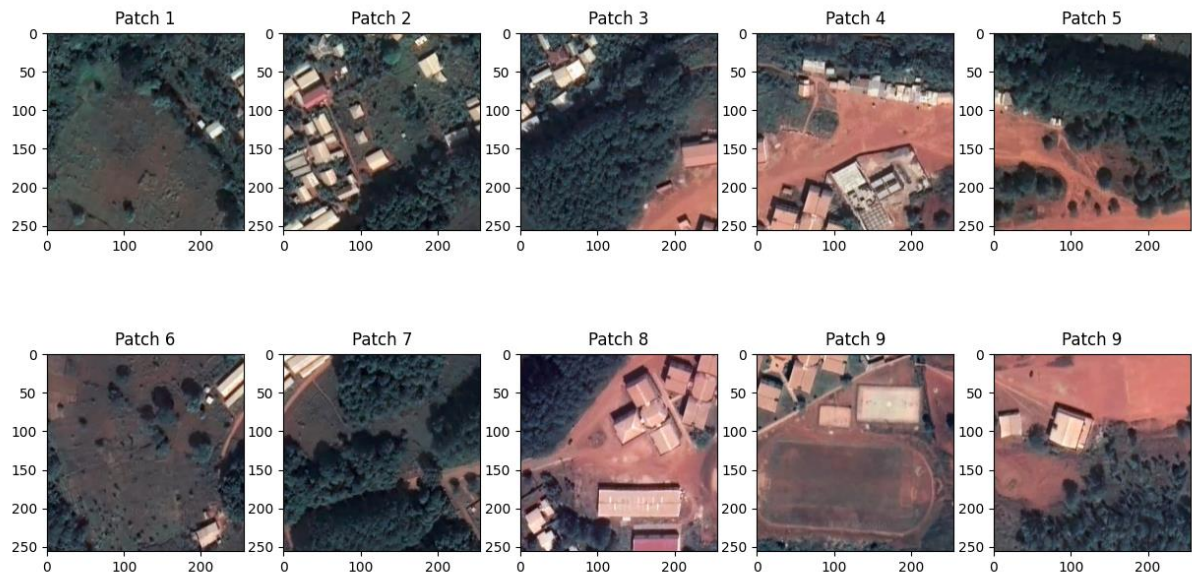


Figure 14: Différents patches obtenus

Après avoir divisé l'image en patch, ces différents patches sont prédits par le modèle et les résultats obtenus sont les suivants :

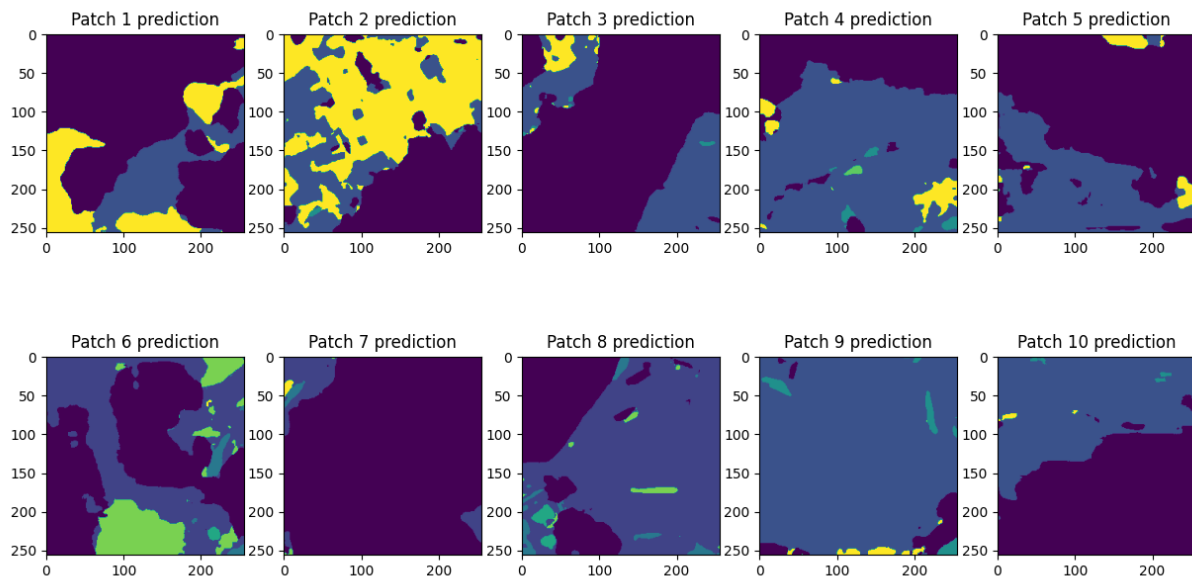



Figure 15: Résultat de la prédiction des différents patches

On peut donc constater que sur chaque patch prédit le modèle a pu reconnaître, extraire et identifier des formes présentes en donnant une couleur spécifique pour chaque forme identifiée. Ces différentes couleurs ont les significations suivantes :

: Végétation

: Terre

 : Bâtiments

 : non labélisé (forme inconnue)

III.5.2- Application web

III.6 EVALUATION DES COUTS

CHAPITRE IV: DIFFICULTES ET SUGGESTIONS

CONCLUSION

REFERENCES BIBLIOGRAPHIQUES

[1] (2023, Janvier 30). Récupéré sur tensorflow.org: <https://www.tensorflow.org/guide>

CHauhan, A. (2023, janvier 15). *Probramp*. Récupéré sur Youtube:
<https://www.youtube.com/@Prodramp/playlists/Deep Learning>

Nada, B. (2023, Janvier 20). *detection-images-yolo-tensorflow*. Récupéré sur blent.ai:
<https://blent.ai/detection-images-yolo-tensorflow/>

Saint-Cirges, G. (2019). *Apprendre le Machine Learning* . Paris.

Saint-Cirges, G. (2022). *Apprendre le Machine Learning en une semaine*. Récupéré sur Machine
Learnia: <https://machinelearnea.com>

Saint-Cirges, G. (2023, janvier 20). *Machine Learnea*. Récupéré sur Youtube:
[https://www.youtube.com/@MachineLearnia/playlists/Machine Learning Français
Formation complete](https://www.youtube.com/@MachineLearnia/playlists/Machine Learning Français Formation complete)

Saint-Cirges, G. (2023, janvier 20). *Machine Learnea*. Récupéré sur Youtube:
<https://www.youtube.com/@MachineLearnia/playlists/Formation Deep Learning>

TABLE DES MATIERES

DEDICACE.....	i
AVANT-PROPOS	ii
REMERCIEMENTS	iv
LISTE DES ABREVIATIONS	v
RESUME.....	vi
ABSTRACT	vii
.....	viii
SOMMAIRE	viii
LISTES DES FIGURES ET TABLEAUX	x
INTRODUCTION.....	1
CHAPITRE I: PRESENTATION DU PROJET	2
I.1- PRESENTATION DE L'ARRETE DU PROJET	2
I.2- PROBLEMATIQUE	2
I.3- OBJECTIFS DU PROJET.....	3
I.4- LES ACTEURS CIBLES DU PROJET	3
I.5- RESULTATS ATTENDUS	4
I.6- ETAT DE L'ART	4
I.6.1- Techniques et approches utilisées.....	4
CHAPITRE II : ANALYSE FONCTIONNELLE ET TECNQUE CONCEPTION	7
II.1- ANALYSE FONCTIONNELLE DU SYSTEME	7
II.2- CONCEPTION TECHNIQUE ET FONCTIONNELLE DU SYSTEME.....	7
II.2.1- Acquisition d'un dataset.....	7
II.2.2- Traitement des images.....	10
II.2.2.a- Représentation d'une image en mémoire	10
II.2.2.b- Preprocessing des images	11
II.2.1.d- Entrainement du modèle.....	16
CHAPITRE III: IMPLEMENTATION ET RESULTATS	17
III.1- TECHNOLOGIES UTILISEES	17
III.1.1- TensorFlow	17
III.1.2- Keras	18

III.1.3 WanDB	19
III.2- ENVIRONNEMENT DE DEVELOPPEMENT	20
III.2.1 Entraînement du modèle.....	20
III.2.1.a- Environnement matériel	21
III.2.1.b- Environnement logiciel	21
III.2.2 Déploiement du modèle sur une application web	22
IV- EVOLUTION DE L'APPRENTISSAGE DU MODELE	22
III.4 DEPLOIEMENT DE L'APPLICATION.....	25
III.5 RESULTATS OBTENUS	25
III.5.1- Test du model.....	25
III.5.2- Application web	27
III.6 EVALUATION DES COUTS	27
CHAPITRE IV: DIFFICULTES ET SUGGESTIONS	28
CONCLUSION	29
REFERENCES BIBLIOGRAPHIQUES	30
.....	31
TABLE DES MATIERES	31