

DÉPARTEMENT DE GÉOMATIQUE APPLIQUÉE
FACULTÉ DES LETTRES ET SCIENCES HUMAINES
UNIVERSITÉ DE SHERBROOKE

ESSAI

SEGMENTATION SÉMANTIQUE EN TEMPS RÉEL À PARTIR D'UN NANO ORDINATEUR : ÉTUDE DES PERFORMANCES ET DES LIMITES

*Essai présenté pour l'obtention du grade de Maître en sciences (M.Sc.),
cheminement géodéveloppement durable*

VINCENT LE FALHER

LONGUEUIL
SEPTEMBRE 2020

Remerciements

Je tiens à remercier ...

□ TODO

Table des matières

| | |
|--|----------|
| Liste des figures | 3 |
| Liste des tableaux | 4 |
| 1 Suivi des changements au rapport | 1 |
| 2 Introduction | 3 |
| 2.1 Mise en contexte | 3 |
| 2.2 Problématique | 4 |
| 2.3 Objectifs | 5 |
| 3 Cadre théorique (état des connaissances, revue de la littérature) | 5 |
| 3.1 Cadre théorique au sujet du nano ordinateur | 5 |
| 3.2 Cadre théorique au sujet de l'apprentissage profond et de la segmentation sémantique | 6 |
| 4 Matériel et méthodes | 6 |
| 4.1 Site d'étude | 6 |
| 4.2 Modèles et jeux de données | 9 |
| 4.3 Matériel et logiciels | 13 |
| 4.4 Méthodologie | 17 |
| 4.5 Documentation | 20 |
| 4.6 Recherche | 20 |
| 4.6.1 Revue de littérature | 20 |
| 4.6.2 Étude du site d'implantation | 21 |
| 4.6.3 Sélection des données et des modèles de réseaux de neurones | 21 |
| 4.6.4 Choix de l'équipement pour le nano ordinateur | 22 |
| 4.6.5 Exploration des solutions logicielles | 22 |
| 4.7 Environnement de travail | 22 |
| 4.7.1 Préparation du nano ordinateur | 22 |
| 4.7.2 Collecte des données | 31 |
| 4.7.3 Mise en place des solutions logicielles | 31 |
| 4.8 Évaluation | 34 |
| 4.8.1 Stratégie de test | 34 |
| 4.8.2 Stratégie de collecte des indicateurs de performance | 35 |
| 4.8.3 Segmentation avec des images | 36 |
| 4.8.4 Segmentation avec des vidéos | 36 |
| 4.9 Adaptation | 36 |
| 4.9.1 Choix du modèle de l'architecture FCN | 37 |
| 4.9.2 Adaptation du modèle | 37 |
| 4.9.3 Ré-entraînement du modèle | 37 |

| | |
|---|-----------|
| 5 Résultats | 37 |
| 5.1 Performances matérielles | 37 |
| 5.1.1 Stockage de données | 37 |
| 5.1.2 Performances système | 37 |
| 5.2 Performances de l'inférence | 44 |
| 5.2.1 Images | 44 |
| 5.2.2 Vidéos | 46 |
| 6 Interprétation et discussion des résultats | 47 |
| 6.1 Performances matérielles | 47 |
| 6.1.1 Stockage de données | 47 |
| 6.1.2 Performances système | 47 |
| 6.2 Performances de la segmentation | 49 |
| 6.2.1 Images | 49 |
| 6.2.2 Vidéos | 49 |
| 7 Conclusion et recommandations | 49 |
| 8 Annexes | 53 |

Liste des figures

| | |
|---|----|
| 1 | 7 |
| 2 Vue aérienne du pont Jacques-Cartier | 7 |
| 3 Carte du site d'implantation | 8 |
| 4 Schéma de la configuration de la piste multifonctionnelle | 9 |
| 5 Carte mère Jetson Nano de NVIDIA | 13 |
| 6 Diagramme de l'architecture du NVIDIA JetPack | 14 |
| 7 Organigramme de la méthodologie à haut niveau | 17 |
| 8 Organigramme de la méthodologie pour évaluer les performances | 17 |
| 9 Organigramme des détails de la méthodologie pour évaluer les performances | 18 |
| 10 Organigramme de la méthodologie pour évaluer les performances après une phase d'adaptation théorique | 18 |
| 11 Organigramme de la méthodologie pour évaluer les performances après une phase d'adaptation réaliste | 19 |
| 12 Organigramme des détails de la méthodologie pour évaluer les performances après une phase d'adaptation | 19 |
| 13 Préparation du nano ordinateur | 22 |
| 14 Montage du nano ordinateur | 23 |
| 15 Carte mère du nano ordinateur | 23 |
| 16 Adaptateur 5 volts 4 amps | 24 |
| 17 Boitier pour le nano ordinateur | 25 |
| 18 Vue arrière du boitier pour le nano ordinateur | 25 |
| 19 Disque SSD NVMe M.2 interne 250GB | 27 |

| | | |
|----|---|----|
| 20 | Caméra | 28 |
| 21 | Ventilateur | 28 |
| 22 | Hub USB 3.0 externe autoalimenté | 29 |
| 23 | Éléments pour l'évaluation des performances | 34 |
| 24 | Méthodologie du traitement et adaptation | 36 |
| 25 | Les périodes du test | 39 |
| 26 | Fréquence | 40 |
| 27 | Mémoire | 40 |
| 28 | I/O total en % de la segmentation | 41 |
| 29 | I/O en KBytes de la segmentation | 41 |
| 30 | I/O total du disque en KBytes | 42 |
| 31 | Températures d'opération | 42 |
| 32 | Consommation d'opération | 43 |
| 33 | vérité terrain ("ground truth")" de l'image b1-09517 | 44 |
| 34 | Segmentation sémantique de l'image b1-09517 générée par le modèle. Le IoU et le Dice score pour le chemin sont de +80%. | 45 |
| 35 | vérité terrain ("ground truth")" de l'image b378-61 | 45 |
| 36 | Segmentation sémantique de l'image b378-61 générée par le modèle. Le IoU pour le chemin est +69% | 46 |

Liste des tableaux

| | | |
|---|--|----|
| 1 | Suivi des changements | 1 |
| 2 | Tableau des données | 11 |
| 3 | Solutions logicielles de l'essai | 15 |
| 4 | Cartes microSD | 26 |
| 5 | Comparaison des performances du "data read" entre un SDD M.2 NVMe et une microSD | 37 |
| 6 | Résolutions et images-par-seconde (FPS) testés | 47 |

1 Suivi des changements au rapport

Cette section n'est disponible que pendant la rédaction du rapport. Les changements au rapport sont documentés dans ce tableau. Cela permet de faire un suivi et permet d'aider les personnes qui révisent et commentent le rapport.

Les changements les plus récents apparaissent en premier.

TABLE 1 – Suivi des changements

| Version | Date | Changement |
|---------|--------------------------|---|
| v1.2.1 | 27 28 29 juillet 2020 | <ul style="list-style-type: none">— Fixe les longues URL dans tout le document.— Bonifie la section Résultat avec un tableau des résolutions qui ont été testées, et spécifie avec quelle carte microSD les tests de performances sont exécutés.— Bonifie la section des Interprétations des résultats\Température avec le capteur AO et la durée de vie en opération constante.— Ajoute une image pour chaque microSD dans un tableau, mais le label n'est pas bien centré verticalement. (todo en cours).— Bonifie la section Solutions logicielles, et ajoute un tableau des logiciels.— fixe les Caption for LOF par le bon libellé pour la TOC. |

| Version | Date | Changement |
|----------------|-----------------|--|
| v1.2.0 | 26 juillet 2020 | <ul style="list-style-type: none"> — Modification dans l'ensemble du document de l'expression "ground truth" par vérité terrain ("ground truth"). — Ajout dans les notes de bas de page des URLs (images du pont ; github). — Site d'étude : ajout des URL en note de base de page ; enlève la deuxième image du pont ; correction du nom de la station de métro la plus proche. — Matériel et logiciels/Le nano ordinateur : Ajout d'une référence à l'image. — Méthodologie : référence à la documentation dans git-thug ; ajout d'une sous-section Documentation ; ajout des liens github dans les notes de bas de page. — Stratégie de collecte des indicateurs de performance : mise à jour majeure. — Préparation du nano ordinateur/Disque SSD NVMe M.2 interne 250GB : mise à jour ; ajout de notes de bas de page. — Préparation du nano ordinateur/Caméra : précision de laquelle (Raspberry Pi v2) — Résultats/Performances système : Précision du FCN + résolution sélectionnée lors du test. — Résultats/Performances de l'inférence/Vidéo : ajout des liens vidéos en notes de bas de page ... mais trop long. |
| v1.1.0 | 25 juillet 2020 | Section Résultats et Interprétation des résultats. |
| v1.0.2 | 19 juillet 2020 | Ajout dans le tableau des performances data read SSD vs microSD les 2 autres microSD utilisées pendant l'essai. |
| v1.0.1 | 18 juillet 2020 | <ul style="list-style-type: none"> — Ajout d'une section changelog pour documenter le suivi des changements et faciliter la révision. — Apport de clarification dans la section Matérielle & logiciels (hdparm, présentation SDK, etc). — Début de la rédaction de la section résultat ; inclusion d'un tableau des performances data read SSD vs microSD. |
| v1.0 | 17 juillet 2020 | Première version envoyée à Michaël G. |

2 Introduction

2.1 Mise en contexte

La compagnie LES PONTS JACQUES CARTIER ET CHAMPLAIN INCORPORÉE (PJCCI) désire évaluer la mise en service de la piste multifonctionnelle (vélos, piétons, etc.) du pont Jacques-Cartier, à Montréal, durant l'hiver. Pour ce faire, la piste doit rester sécuritaire et dégagée, malgré les évènements météorologiques.

L'université de Sherbrooke, qui participe à cette initiative, propose de mettre en place sur le pont une plateforme de détection innovatrice qui consiste à installer plusieurs paires d'objets connectés ultralégers et performants (des nano ordinateurs) à différents endroits du pont. Chacun de ces nano ordinateurs possède trois différents types de capteurs : vision, son, et météorologiques (température, humidité, etc.). Chaque nano ordinateur d'une paire perçoit le même environnement, mais d'une perspective différente que son homologue : la caméra pointe vers la même surface, mais d'un autre point de vue ; les sons et les données météorologiques sont captés dans le même voisinage. Les données collectées par les capteurs sont traitées en temps réel par des algorithmes de détections performants qui sont adaptés à ce type de problématiques : les réseaux de neurones, du domaine de l'intelligence artificielle. La déduction de l'état de la surface de la piste (sèche, mouillée, glacée, etc.) se fait en fusionnant les différentes perceptions (multicibles) de chaque capteur (multicapteurs).

L'objet principal de cet essai consiste à étudier la capacité du nano ordinateur du fabricant NVIDIA, le Jetson nano [17], à exécuter, en temps réel, un modèle de réseau de neurones entraîné à faire de la segmentation sémantique (classification) d'images de haute résolution qui sont perçues avec la caméra. Les résultats de cette étude permettront de déterminer le modèle de réseau de neurones le plus adapté pour répondre aux besoins du volet vision du projet pour PJCCI.

La détection d'objets et de surface en temps réel est de plus en plus précise et efficace depuis que les performances des systèmes informatisés permettent l'exécution d'algorithmes exigeants, en majeure partie depuis l'utilisation des processeurs graphiques "GPU" [5] [7] [2] [11] [[jia_real-time_2020](#)] [13]).

Les systèmes informatiques performants sont de plus en plus miniatures, on parle de nano ordinateurs et des objets connectés ("Internet of Things" ou "IoT") [4] [22]. Ils permettent la détection en temps réel à des endroits, dans des situations et dans des conditions qui n'étaient pas envisageables il y a encore 10 ans ([\[jia_real-time_2020\]](#) [3] [1] [4]).

Les réseaux de neurones ont aussi très rapidement progressé depuis 2012 [2], permettant d'offrir des alternatives aux solutions de détection et de classifications, entre autres [19]. Les réseaux de neurones convolutifs entiers ("FCN" en anglais, pour "Fully Convolutional Network") sont les derniers à avoir émergé ("state-of-art") [[jia_real-time_2020](#)] et à profiter au domaine de la vision et de la détection d'objets ([16] [[jia_real-time_2020](#)]).

La segmentation sémantique est une forme de classification d'image, pixel par pixel, qui tire profit des dernières évolutions de la classification supervisée grâce aux réseaux de neurones convolutifs entiers, et se permet d'être déduite en temps réel avec des nano ordinateurs ([14] [4]). Les images doivent être de très haute résolution, ce qui nécessite d'avoir à disposition un système informatique capable de fournir une puissance de calcul appropriée, particulièrement pour la manipulation de la mémoire et des nombres flottants pendant l'inférence [15]. Leur application par des nano ordinateurs est un défi en raison de la faible consommation d'énergie (Watts) et de la

puissance de calcul limité de ces derniers [6].

Pour PJCCI, les avantages d'une telle plateforme seraient multiples, et on peut en énumérer plusieurs, sans se limiter à : contrôler l'épandage de sel ; surveiller les conditions de la piste multifonctionnelle ; suivre les effets du gel et du dégel ; optimiser les couts des opérations d'entretien (déplacements, quantité) ; offrir aux usagers des conditions d'accès sécurisées et optimales même en hiver ; effets environnementaux atténus ; détecter ce qu'un spécialiste humain ne pourrait pas ou aurait des difficultés à détecter ; prise de décision et gestion proactive ; planification.

D'un autre côté, les défis ne sont pas à sous-évaluer : la détection doit être précise, fiable et consistante, tout cela afin d'assurer aux usagers un service de qualité dans un contexte sécuritaire.

2.2 Problématique

Dans le cadre du projet pour PJCCI, une plateforme technologique sera mise à la disposition des gestionnaires du pont afin de les aider à prendre les décisions les plus responsables et raisonnables possibles. Mais la mise en opération d'une solution innovante et fiable, qui concilie des algorithmes d'apprentissage profond, du temps réel, des nano ordinateurs, et des conditions climatiques variables, est complexe. Dans une certaine mesure l'essai va contribuer à la recherche de solutions afin de répondre au défi pour le domaine du transport actif et durable d'être soutenu par des solutions technologiques fiables (opérationnelles), l'objectif étant de pouvoir offrir des services de qualité et sécuritaires sur l'ensemble des quatre saisons.

La seconde problématique que l'essai va contribuer à résoudre concerne les limites d'un nano ordinateur. L'inférence nécessite une architecture et une puissance machine différente de celle nécessaire pour l'entraînement. Les modèles de réseaux de neurones sont adaptés et optimisés pour l'inférence. L'essai va permettre de préciser les capacités du nano ordinateur pour l'inférence de diverses architectures de réseaux de neurones convolutifs entiers (FCN en anglais) et la segmentation sémantique en temps réel avec des vidéos de différentes propriétés (résolutions et nombre d'images par seconde). Il existe des tests encourageants ([18] [16] [5]), qui seront utilisés comme modèle, même si ceux-ci sont limités à des types d'application qui ne sont pas les mêmes que pour l'essai.

Il est difficile de trouver des jeux de données pour entraîner les réseaux de neurones convolutifs entiers adaptés à la problématique. La technique de "Data augmentation" permet de démarrer d'un modèle qui a déjà appris avec un jeu d'images important (milliers d'images), et de lui faire apprendre davantage, en lui fournissant un plus petit jeu d'images (centaines d'images) de la nouvelle zone d'étude. Par exemple un modèle peut avoir appris à classifier des images de la Californie, États-Unis. Pour lui permettre de classifier des images de la Ville de Sherbrooke, il est souhaitable de lui fournir un nouveau jeu de données spécifique à cette ville afin qu'il s'adapte (ses paramètres) à cette région. Dans le contexte de cet essai, les données acquises sur le terrain seront fournies aux différents modèles qui seront évalués, et qui seront ré entraînés avec ce nouveau jeu d'images adapté à la zone d'étude.

La paramétrisation (des "hyper paramètres") des réseaux de neurones est très "subtile" et "intuitive" et requiert de l'expérience. C'est un processus d'essais-erreurs qui est très coûteux en temps, et risqué puisqu'il n'y a aucune garantie de succès. La technique de "Transfer Learning" permet d'hériter d'un modèle qui est déjà entraîné et configuré, et de l'adapter pour répondre à ses besoins. Cette technique permet un gain en temps et en énergie (et en argent) important puisque le temps de conception (architecture et configuration) et le temps d'entraînement, de validation et de tests

sont diminués de façon non négligeable. La problématique pour l'essai est de trouver le modèle qui est le plus adapté pour répondre au besoin, et il en existe des milliers [12]. La recherche dans la littérature permet heureusement de limiter les choix et donner des pistes ([jia_real-time_2020] [16] [18]). La problématique de la conception existe toujours, car le modèle a besoin d'être étudié, adapté et ré entrainé, jusqu'à l'obtention de résultats probants. Mais la paramétrisation des hyper paramètres n'est plus nécessaire (supposément), ce qui est très avantageux.

2.3 Objectifs

Le premier sous-objectif est de déterminer quelles sont les limites de la plateforme, d'un point de vue matériel (GPU, CPU, mémoire, transfert mémoire, consommation, etc.), mais aussi applicatif (entrainement, inférence). Cette phase du projet va permettre d'exécuter différents modèles déjà existants, sans modification, en tenant compte des éléments documentés dans la littérature [16] [jia_real-time_2020] [18]. Selon le déroulement de cette étape, un ou plusieurs modèles seront sélectionnés.

Un autre sous-objectif est d'optimiser ou d'adapter la plateforme, d'un point de vue matériel, mais aussi applicatif, afin d'avoir les meilleures performances et résultats possibles pendant l'entraînement et l'inférence.

Comme les résultats devront être disponibles en tout temps, une connexion à distance sécurisée devra être mise en place. Cette connexion permettra aussi de pouvoir prendre le contrôle du nano ordinateur à distance et de l'administrer.

L'approche, les tests, et les résultats seront documentés. Il y aura beaucoup d'activités relatives à la conception et aux tests, le cheminement complet ne sera pas fourni. Une synthèse sera préférée et les informations les plus pertinentes seront incluses. Les détails de l'installation de l'environnement de développement et des applications, bibliothèques et autres dépendances nécessaires seront inclus, ainsi que ceux de la configuration. Dans le cas où l'objectif principal n'est pas atteint, ou partiellement, la/les raison/s de l'échec seront spécifiées et des pistes de solutions potentielles proposées.

3 Cadre théorique (état des connaissances, revue de la littérature)

Il y a deux sections, la première qui concerne le nano ordinateur et ensuite la seconde, l'apprentissage profond et la segmentation sémantique.

3.1 Cadre théorique au sujet du nano ordinateur

TODO

Voici le plan qui est utilisé pour rédiger le cadre théorique au sujet du nano ordinateur.

- historique et évolution ; une brève présentation de l'historique des nano ordinateurs, de leur apparition à leur place aujourd'hui.
- usages ; quelques exemples d'usages des nano ordinateurs, dans un contexte professionnel.
- architecture ; brève présentation, fonctionnement et comparaison des architectures matérielles des nano ordinateurs, leurs couts, leurs avantages et limitations.

3.2 Cadre théorique au sujet de l'apprentissage profond et de la segmentation sémantique

TODO

Voici le plan qui est utilisé pour rédiger le cadre théorique au sujet des réseaux de neurones et de la segmentation sémantique.

- historique ; une brève présentation de l'historique et du contexte de l'intelligence artificielle, l'apprentissage machine et les réseaux de neurones.

Les concepts de l'Intelligence artificielle (AI) existe depuis les années 1950 et ont continué à se développer jusqu'à leur popularité des 10 dernières années ;

- popularité depuis 10 ans ; argumentation autour des raisons de la renaissance de l'apprentissage machine.

Trois raisons principales ont permis à ce domaine de sortir du champ de la recherche pour celui de l'industrie et la mise en production : amélioration de la capacité et la puissance des machines ; jeux de données plus larges ; algorithmes plus avancés ;

- domaine ; où se situent les réseaux de neurones et la segmentation sémantique dans la hiérarchie de l'IA.

L'apprentissage profond est un sous-domaine de celui de l'apprentissage machine qui est un sous-domaine de celui de l'intelligence artificielle.

La segmentation sémantique d'images ou de vidéos avec des algorithmes d'apprentissage profond fait partie du domaine de la télédétection par la vision.

- applications ; quelques exemples d'applications des réseaux de neurones selon leur type, dont les réseaux de neurones à convolution entiers.

Les applications sont plus sophistiquées, la segmentation sémantique en fait partie.

- principes ; présentation des principes théoriques de la segmentation sémantique.

La segmentation sémantique d'images est une technique élaborée de classification supervisée d'images.

À noter qu'il n'est pas prévu expliquer le fonctionnement des réseaux de neurones, tels que les différentes fonctions (activation, perte), les hyper-paramètres, les différentes architectures et les types de couches.

4 Matériel et méthodes

4.1 Site d'étude

Tel que trouvé dans le rapport détaillé sur le projet pilote d'entretien hivernal de la piste multifonctionnelle du pont Jacques-Cartier [21], la piste multifonctionnelle du pont Jacques-Cartier (figure 2) relie Montréal intramuros, proche de la station de métro De Lorimier / René-Lévesque, et la rive sud, à Longueuil, proche de la station de métro Longueuil et de son terminal de bus. Elle est longue d'une distance de 2.7km et est située d'un seul côté du pont, côté sud (figure 3). Elle est surtout utilisée par les cyclistes, et moindrement par les passants [21]. Sa configuration est bien particulière [20] (figire 4) : elle ne longe pas la route adjacente sur toute sa longueur ; elle est interrompue par une voie de sortie de l'île Notre-Dame ; des chicanes sont disposées à certains endroits ; sa largeur varie entre 2.5m et 1.8m ; elle possède une pente assez prononcée à certains moments ; il y a des courbes assez serrées. Elle est fermée l'hiver par mesure de sécurité. Elle est

ouverte au début du printemps jusqu'au début de l'hiver, lorsque les conditions ne nécessitent pas d'entretien.



FIGURE 1

FIGURE 2 – Vue aérienne du pont Jacques-Cartier (flickr PJCCI)¹.

1. <https://www.flickr.com/photos/pjcci/6830109134>

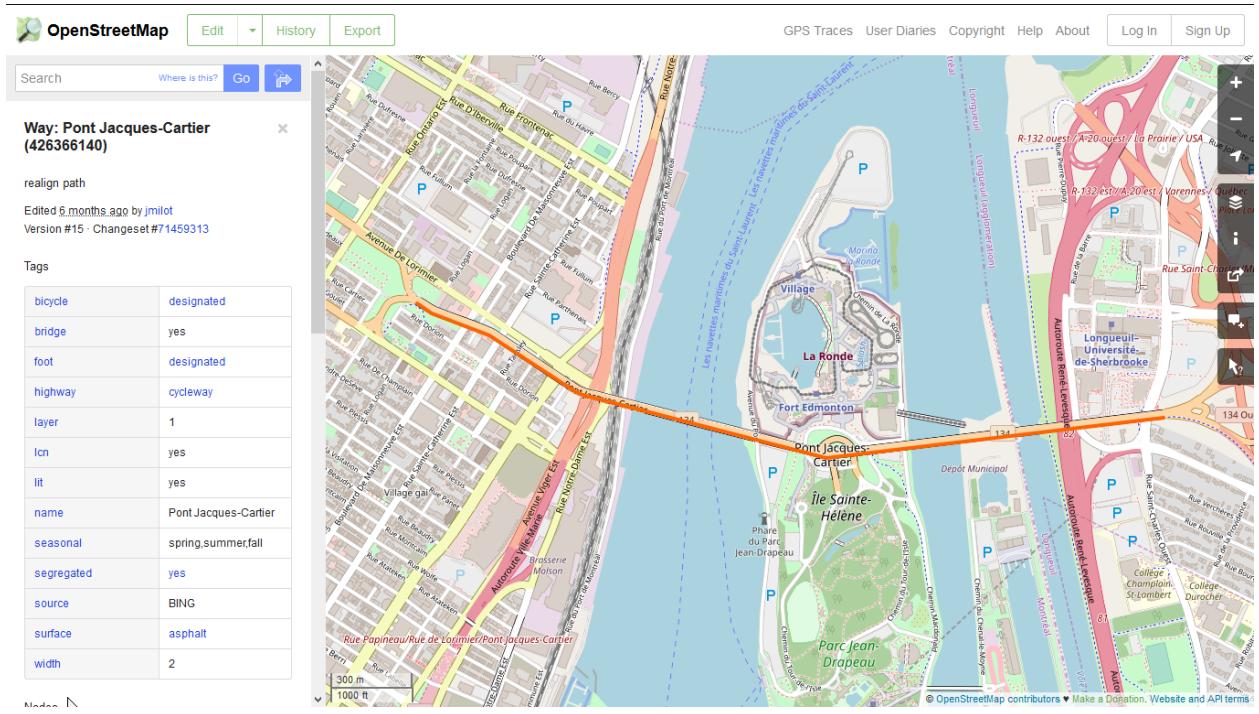


FIGURE 3 – Carte du site d’implantation : le pont Jacques-Cartier et la piste multifonctionnelle en orange sur le pont (OpenStreetMap).

Piste multifonctionnelle du pont Jacques-Cartier

La piste multifonctionnelle du pont Jacques-Cartier est un lien singulier aux défis uniques



Pont urbain à la géométrie atypique et exposé à des conditions météorologiques particulières.

Particularités de la piste et enjeux de sécurité

La combinaison de l'ensemble de ces éléments soulève des enjeux de sécurité accrus en hiver.

Géométrie atypique

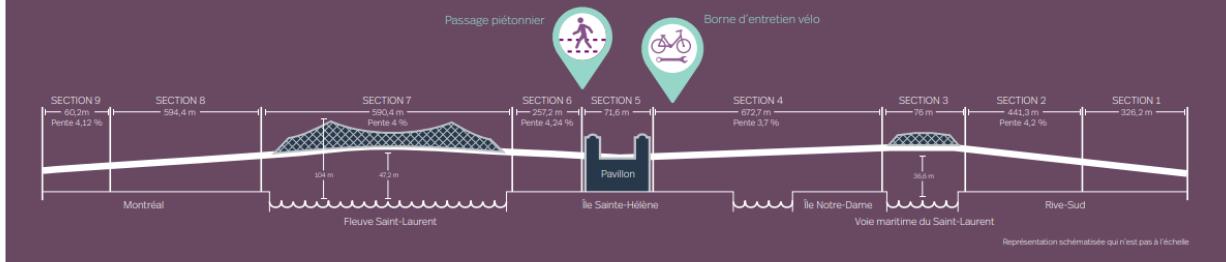
- + Longueur (2,7 km)
- + Dégagement latéral étroit (de 2,5 m à 1,8 m en hiver)
Normes du MTMDET : 3,5 m
- + Pentes longues et abruptes (4,2 %)
- + Virages serrés
- + Enclavement (garde-corps de chaque côté)

Piste surélevée

- + Dalle de béton (15 cm)
 - + Pas de matériaux en profondeur qui l'isolent et atténuent son refroidissement
 - + Plus affecté par les variations météorologiques qu'une piste sur remblai
- Danger de chute de glace**
- + Structures métalliques en hauteur (sections 3 et 7)
 - + Neige projetée des voies de circulation

Complexité des conditions météorologiques du fleuve Saint-Laurent

- + L'environnement climatique au-dessus du fleuve Saint-Laurent est unique, rigoureux et changeant
- + Milieu très humide, venteux et changeant propice à la formation imprévisible de glace noire qui affecte directement la qualité de la surface de roulement



Représentation schématisée qui n'est pas à l'échelle

AUTOMNE 2016

FIGURE 4 – Schéma de la configuration de la piste multifonctionnelle (PJCCI)².

4.2 Modèles et jeux de données

Données

Les ressources mises à disposition par le constructeur du Jetson nano, NVIDIA, font référence à des jeux de données qui sont disponibles publiquement.

En complément des ressources de NVIDIA, deux références scientifiques seront principalement étudiées, car leurs recherches ont été faites avec le Jetson nano ([16] et [5]). Beaucoup de références ont été publiées ces deux dernières années sur le sujet de la segmentation sémantique, ils existent donc de multiples alternatives inspirantes.

Internet est une mine de données : il existe des forums et des blogues dans lesquels des utilisateurs publient leurs expérimentations de la segmentation sémantique en temps réel avec le Jetson nano ([8]), ou plus génériquement la segmentation sémantique ; des sites comme "modelzoo.co" ou "kaggle.com" sont des entrepôts de données et de modèles FCN prêts à être utilisés ; une autre

2. https://jacquescartierchamplain.ca/wp-content/uploads/2018/10/IMG_Fiche_piste-multi_pont_JC_FR_vfinale_web_2018-10-10.pdf

option a été d'effectuer une recherche d'images ou de vidéos de la piste multifonctionnelle du pont Jacques-Cartier via les sites de recherche tels que Google.

L'Association des piétons et cyclistes du pont Jacques-Cartier existe depuis de nombreuses années pour promouvoir le transport actif et conserver la piste multifonctionnelle du pont Jacques Cartier ouverte durant l'hiver. Ils fournissent, via leurs sites Internet, des collections de vidéos et d'images qui seront utilisées après leur avoir demandé leur autorisation verbale et écrite. [9] [10]

Approche prévue pour le traitement des données

Il y a deux phases à cet essai : l'inférence avec des modèles déjà prêts et l'inférence avec des modèles ré entrainés. Les données utilisées par l'inférence sont des vidéos (d'une certaine résolution et d'un certain nombre d'images par seconde), et celles pour l'entraînement sont des images. Dans les deux cas, les images pour l'entraînement ou l'inférence doivent être d'une taille bien précise, celles avec lesquelles le modèle a été, ou sera, entraînées. La résolution et la qualité de l'image vidéo seront nivélées vers le bas afin de déterminer la limite inférieure acceptable pour la détection la plus efficace et fiable possible. La résolution et le nombre d'images par seconde de la vidéo sont contrôlés par le logiciel ("driver" en anglais) de la caméra, et sont configurables.

Tout cela signifie que les vidéos ou nouvelles images devront être traités pour répondre à une certaine taille et résolution requise par le modèle, tout en conservant une qualité élevée (nombre de pixels, niveaux de couleurs). De nouvelles images pour l'entraînement seront extraites des vidéos, et annotées.

Certains cadres d'application logicielle ("framework") d'apprentissage profond (par exemple "Keras") offrent l'option d'augmenter automatiquement le jeu de données avec des techniques d'augmentation de données (par exemple la rotation, le redimensionnement, l'effet miroir), ce qui est très utile et non négligeable.

Voici le tableau de synthèse des données qui ont été découvertes et qui seront potentiellement utilisées dans le cadre de l'essai, incluant la référence à l'architecture du modèle d'apprentissage profond.

TABLE 2 – Tableau des données

| | Spécification | Description |
|---|--|---|
| 1 | réseau : SegNet jeu de données : CamVid vidéo : 10 minutes résolution/s : HD | SegNet est un réseau qui a été créé pour la segmentation sémantique de vidéos. Il a été entraîné avec le jeu de données de CamVid, qui procurent des vidéos de la route avec la même perspective que le conducteur du véhicule. Un modèle entraîné est disponible pour le Jetson nano. https://github.com/PengKiki/camvid |
| 2 | réseau : MFANet jeu de données : Cityscapes nombre d'images : 5000 résolutions : 1280x1024 | MFANet est un réseau qui a été créé en 2019 pour la segmentation sémantique sur des appareils tel que le Jetson nano. Il a été entraîné avec le jeu de données de Cityscapes, qui procurent des images de scènes urbaines. Différentes stratégies d'augmentation de données sont utilisées. Des tests ont été faits avec le Jetson nano. leejy@ustb.edu.cn |
| 3 | réseau : RESNet18 jeu de données : Cityscapes nombre d'images : 25 000 résolutions : 360x720, 512x256, 1024x512, 2048x1024 | Cityscapes est un jeu de données qui fournit des images de rues spécifiquement destinées pour la segmentation sémantique. Il peut être utilisé par de nombreux réseaux. RESNet18 a été entraîné avec ce jeu et est disponible en diverses résolutions pour le Jetson Nano. https://github.com/tynguyen/MAVNet/tree/master/data/perch_drone |
| 4 | réseau : RESNet18 jeu de données : DeepScenes nombre d'images : 15 000 résolutions : 576x320, 864x480 | DeepScene propose un modèle et un jeu de données. Le modèle est entraîné avec différents jeux de données, comme Cityscapes, SUN-RGBD, Synthia. Le jeu de données fournit des images de forêt, qui est destinée pour la segmentation sémantique. RESNet18 a été entraîné avec ce jeu et est disponible en deux résolutions pour le Jetson Nano. http://deepscape.cs.uni-freiburg.de |
| 5 | réseau : DeepScene jeu de données : Synthia nombre d'images : 220 000 résolutions : 1280x760 | Le jeu de données Synthia fournit des images (et vidéos) de scènes de rue comme celui de Cityscapes, et qui est destiné pour la segmentation sémantique. DeepScene a été entraîné avec ce jeu. Il n'a pas été testé avec le Jetson Nano. http://3dvision.princeton.edu/datasets.html |

| | Spécification | Description |
|---|--|--|
| 6 | jeu de données : Association des piétons et cyclistes pont Jacques-Cartier nombre d'images : 313 résolutions : variées | L'Association des piétons et cyclistes du pont Jacques-Cartier a une collection d'images et de vidéos de la piste multifonctionnelle du pont Jacques-Cartier. Ce n'est pas un jeu de données qui est prêt à être utilisé pour l'apprentissage tel quel, il doit être préparé. Mais c'est une source de données qui est très importante pour l'essai. Il est envisagé de contacter l'association au besoin afin de leur demander leur collaboration pour la collecte d'autres d'images ou vidéos. https://www.flickr.com/photos/pontjacquescartier http://pontjacquescartier365.com/videos-pont-jacques-cartier |
| 7 | jeu de données : images et vidéo sur Internet nombre d'images : entre 30-50 résolutions : variées | Internet est une source de données non négligeable en termes de données. Quelques images et vidéos de la piste multifonctionnelle du pont Jacques-Cartier, autre que celles fournies par L'Association des piétons et cyclistes du pont Jacques-Cartier, sont disponibles. Ce n'est pas un jeu de données qui est prêt à être utilisé pour l'apprentissage tel quel, il doit être préparé. Mais c'est une source de données qui est très importante pour l'essai. https://google.ca |
| 8 | jeux de données : KITI Road/Lane Detection | Ce jeu de données contient 289 images d'entraînement et 290 images de tests d'image de routes urbaines. Il existe une grande multitude de modèles qui sont entraînés avec ce jeu de données. http://www.cvlibs.net/datasets/kitti/eval_road.php |

4.3 Matériel et logiciels

Le nano ordinateur

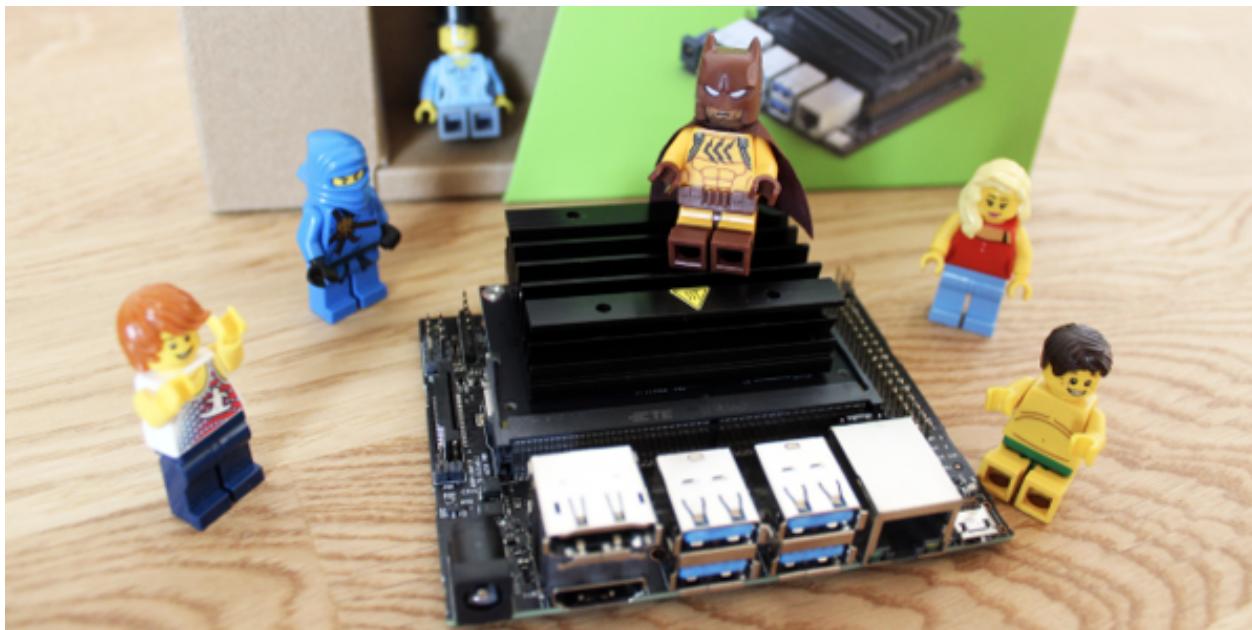


FIGURE 5 – Carte mère Jetson Nano de NVIDIA, représenté avec des Lego pour démontrer sa petite taille

L'objet d'étude de cet essai est le nano ordinateur "Jetson nano" du fabricant "NVIDIA" (figure 5). Ce modèle a été choisi car il a été conçu par la compagnie NVIDIA spécifiquement pour répondre au besoin d'inférence en temps réelle sur le terrain, afin d'éviter le transfert de données et le traitement à distance et différé.

L'architecture du nano ordinateur est ARM 64 bits (aarch64), ce qui le limite pour certaines portabilités de librairies, surtout dans le domaine assez restreind de la recherche, ou l'architecture la plus populaire et portable est x86-64.

Il est composé d'un quad-core ARM Cortex-A57, qui est conçu pour ce genre de nano ordinateur, comme le Raspberry Pi.

Les performances GPU sont faibles, 0.5 TFLOPs (16FP; 16bits/2 bytes floating points). Par comparaison la PlayStation 4 Pro (2016) supporte +4 TFLOPs.

La mémoire est limitée à 6GB.

Les autres caractéristiques à considérer sont le port pour une carte microSD, un port Ethernet 10/100/1000Mbs, un port HDMI, un hub USB 4 ports 3.0, un connecteur pour une caméra, et un port PCIe.

Logiciels

De même que pour les périphériques, les solutions logiciels principales qui seront utilisés dans le cadre de l'essai sont résumés dans le tableau suivant, où il est indiqué leur nom, le type de

licence, leur version, leurs rôles et responsabilités, comme pour le système d'exploitation, l'environnement de développement pour l'apprentissage profond, l'inférence, les logiciels de traitements vidéos et d'images.

Pour tester les performances de la microSD et du disque SSD interne M.2 NVMe, l'utilitaire "hdparm" a été utilisé. Il est nécessaire de l'installer ('sudo apt-get install hdparm') car il n'est pas inclus de base avec le système L4T.

Le SDK qui sera utilisé avec le nano ordinateur sera celui fourni par NVIDIA et qui se nomme "JetPack"^{3 4}. La version 4.4⁵ sera celle avec laquelle les tests de performance ont été exécuté. Il contient le système d'exploitation Linux For Tegra (L4T)⁶ (version L4T 32.4.3), qui est une version de la distribution Linux Ubuntu 18.04 mise à la saveur de NVIDIA. Jetpack contient aussi d'autres bibliothèques qui sont nécessaires pour l'inférence, tel que Cuda, CuDNN et TensorRT.

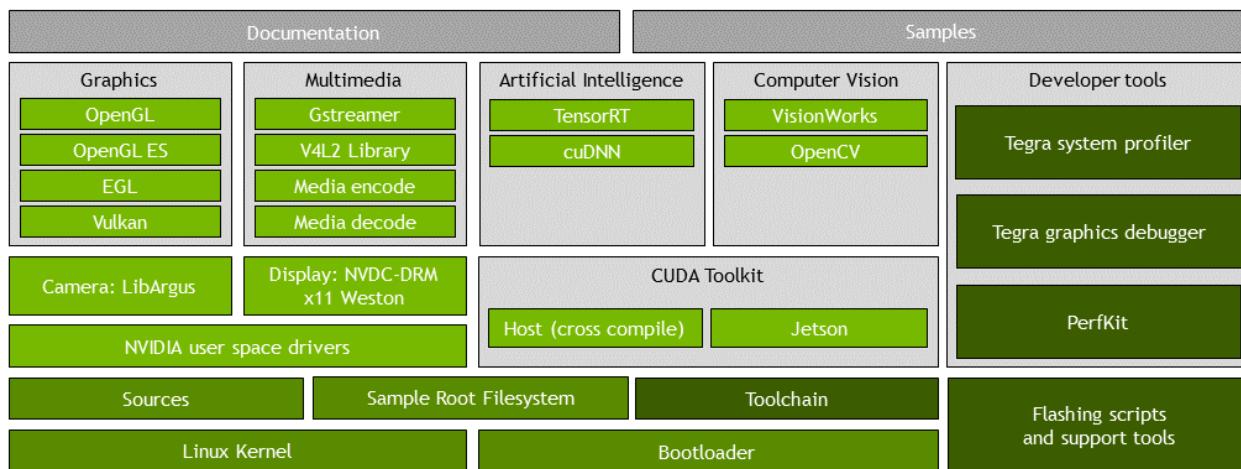


FIGURE 6 – Diagramme de l'architecture du NVIDIA JetPack⁷

Python et le C++ sont les langages utilisés par le framework de DeepLearning de NVIDIA. Python est utilisé comme language accessible et appelle les extensions écrit en C++ et qui optimisent les accès aux ressources systèmes tel que les CPUs et GPUS, les traitement des images et vidéos, les boucles et les traitements mémoires intensifs.

La bibliothèque d'apprentissage profond qui sera utilisée est PyTorch, bonifié avec une version adaptée par NVIDIA de torchvision, qui fournit des modèles d'architecture et des utilitaires pour la vision par ordinateur (computer vision). Des versions bien spécifiques sont nécessaires et il est important de s'y conformer au risque de tomber dans une investigation bien coûteuse en temps et énergie⁸.

Le nano ordinateur inclut un GPU qui est mis à contribution lors de l'inférence. Le compilateur de NVIDIA pour GPU 'cuda' est nécessaire pour régénérer le .onnx lors de la phase d'adaptation. La

3. <https://developer.nvidia.com/embedded/jetpack>

4. <https://docs.nvidia.com/jetson/jetpack/introduction/index.html>

5. <https://developer.nvidia.com/embedded/jetpack-archive>

6. <https://developer.nvidia.com/embedded/linux-tegra>

7. <https://docs.nvidia.com/jetson/l4t/index.html#page/Tegra%2520Linux%2520Driver%2520Package%2520Development%2520Guide%2520Overview.html%23>

8. <https://forums.developer.nvidia.com/t/trying-to-regenerate-onnx-for-jetson-nano/125494?u=vincelf>

version doit concorder avec la bonne version de PyTorch. La version adaptée (fork) de torchvision doit être recompilée avec la bonne version de pytorch et cuda.

Enfin pour régénérer le .onnx lors de la phase d'adaptation, les librairies TensorRT et ONNX ont été utilisées, en compagnie de l'utilitaire ‘trtexec’ qui permet de valider et tester le fichier .onnx généré.

Lors de la phase d'évaluation des performances systèmes, les utilitaires tegrastats, free, iotop ont été utilisés.

TABLE 3 – Solutions logicielles de l'essai

| Language | Version | Licence | Rôles et responsabilités |
|-----------------|---------------------------|----------------|--|
| JetPack | 4.4 | NVIDIA | Kit de développement de logiciels incluant le système d'exploitation L4T, et les librairies et utilitaires nécessaires pour l'inférence avec le nano ordinateur. |
| L4T | 32.4.3 | NVIDIA | Le système d'exploitation "Linux For Tegra" conçu par NVIDIA pour leurs solutions d'inférence légères, comme pour le nano ordinateur. |
| Python | 2.7 | GPL | Langage plus accessible que le C++. |
| C++ | GCC 7.3.1 ⁹ | GPL | Certaines extensions du cadre applicatif de NVIDIA pour l'inférence sont écrites en C++, pour des raisons d'optimisation. |
| pytorch | 1.1.0 | BSD 3-Clause | Cadres d'application logicielle ("framework") pour l'apprentissage machine et profond. |
| torchvision | | BSD 3-Clause | Branche de torchvision adaptée par NVIDIA ¹⁰ ; Doit être recompilée avec la version de pytorch 1.1.0 et cuda 10.0. |
| cuda | 10.0 | NVIDIA | Compilateur de code C++ pour GPU. |
| TensorRT | 6.0.1.5 | NVIDIA | SDK pour générer des modèles au format ONNX, optimisés et interopérables, pour l'inférence. |
| ONNX | | MIT | Permet de générer un format interopérable pour l'inférence de modèles d'architecture construit avec différent framework de machine learning (Caffe, PyTorch, TensorFlow, etc). |
| trtexec | | NVIDIA | Utilitaire qui a permis de tester le .onnx qui a été regénéré. |
| gstreamer | 1.1 | LGPL | Utilitaire qui a permis d'alimenter le modèle de la segmentation avec la vidéo. |
| v4l2loopback | | GPL | Utilitaire qui a permis de créer un matériel vidéo virtuelle permettant de remplacer la caméra, permettant ainsi au modèle d'être alimenté par une vidéo et non la caméra. |
| hdparm | | GPL | Utilitaire permettant de tester la capacité de lecture d'une unité de stockage, tel que'un SSD NVMe et différentes cartes microSD. |

9. <https://developer.nvidia.com/embedded/linux-tegra>

10. <https://github.com/dusty-nv/vision.git> et ensuite branche v0.3.0

| Language | Version | Licence | Rôles et responsabilités |
|-----------------|----------------|----------------|---|
| tegrastats | | NVIDIA | La commande offre différents indicateurs système tel que l'utilisation des processeurs, la température, la consommation, et qui sont utiles pour observer le comportement du système lors des tests de performance de la segmentation. |
| free | | GPL | La commande offre le status de la mémoire totale, utilisée, libre, swap, cachée, etc. Elle est utile pour observer le comportement de la mémoire du nano ordinateur lors des tests de performance de la segmentation. |
| iotop | | GPL | La commande offre le status des opérations "I/O" de lecture & écriture sur le disque, totale ou pour le processus de segmentation. Elle est utile pour observer le comportement des opérations sur le disque du nano ordinateur lors des tests de performance de la segmentation. |

4.4 Méthodologie

Voici à très haut niveau les grandes étapes de cet essai :

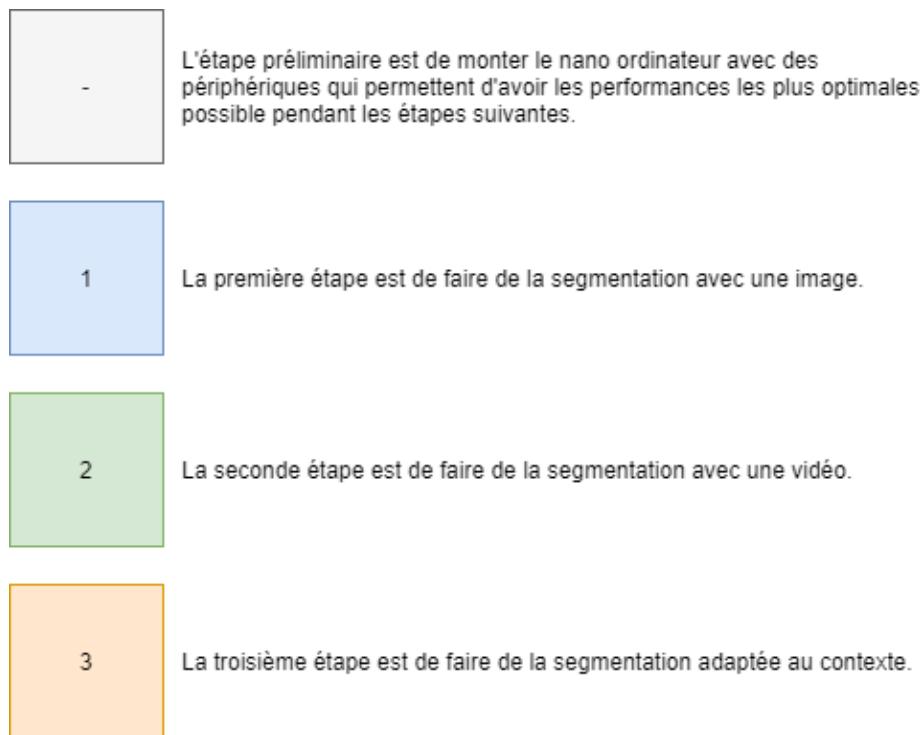


FIGURE 7 – Organigramme de la méthodologie à haut niveau

Pour y parvenir, la méthodologie suivante a été suivie et permet d'évaluer les performances de base de la segmentation sémantique avec le nano ordinateur.



FIGURE 8 – Organigramme de la méthodologie pour évaluer les performances

Si chacun des blocs est explosé, chacun d'eux s'organise autour des activités suivantes :

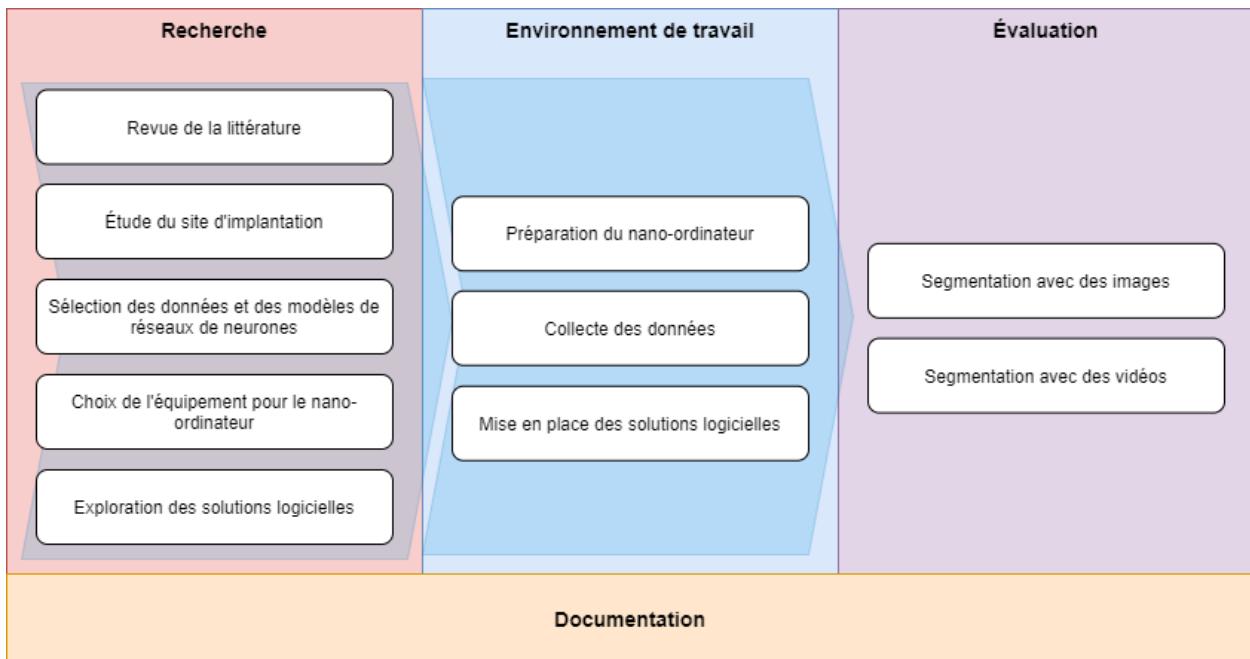


FIGURE 9 – Organigramme des détails de la méthodologie pour évaluer les performances

Si l'évaluation est probante, la méthodologie se verra bonifiée par des étapes d'adaptation et de traitement.

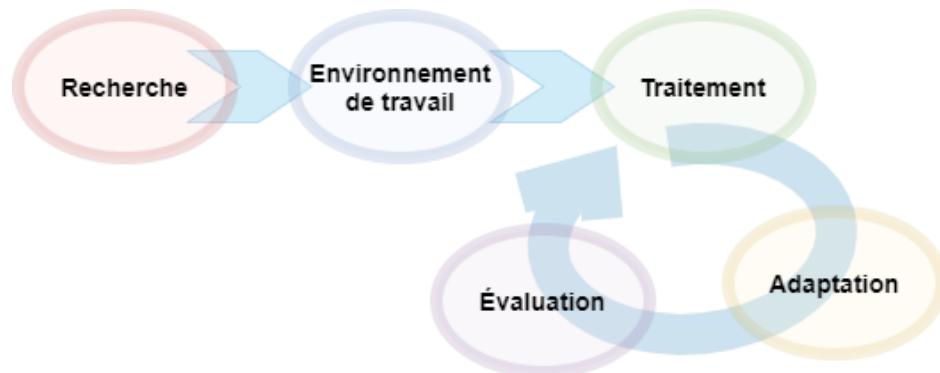


FIGURE 10 – Organigramme de la méthodologie pour évaluer les performances après une phase d'adaptation théorique

Mais sincèrement dans la pratique la méthodologie ressemblera plus à celle-ci :

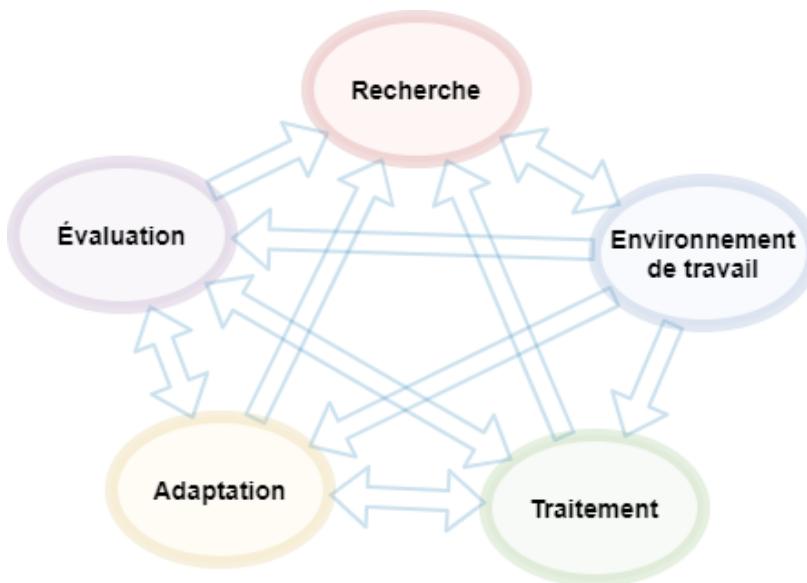


FIGURE 11 – Organigramme de la méthodologie pour évaluer les performances après une phase d'adaptation réaliste

Si chacun des blocs est exploité, chacun d'eux s'organise autour des activités suivantes :

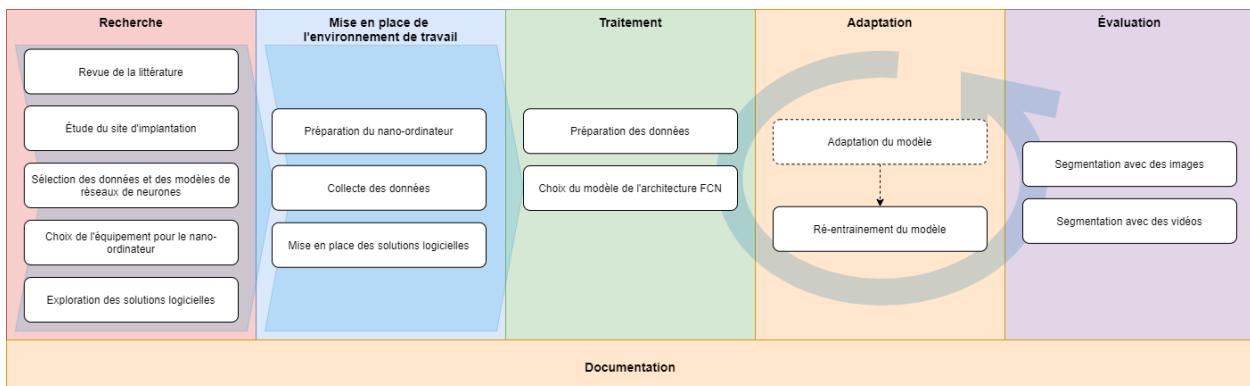


FIGURE 12 – Organigramme des détails de la méthodologie pour évaluer les performances après une phase d'adaptation

Les phases de la méthodologie présentée dans l'organigramme de la figure 12 peuvent être résumées de la façon suivante :

- Recherche des références, des modèles et des données, ainsi que l'équipement pour le nano ordinateur et des logiciels nécessaires.
- Installation sur le nano ordinateur le système d'exploitation, l'environnement de développement et de tests pour l'inférence.
- Itération entre les étapes suivantes :
 - Inférence avec le nano ordinateur en utilisant les modèles et les sources de données sélectionnées.
 - Adaptation des modèles à différentes résolutions d'images et à la zone d'étude.

— Traitement des données afin de les adapter au requis des modèles.

Finalement, il est à noter que le cheminement de l'essai a été entièrement documenté pendant tout le déroulement de l'essai.

4.5 Documentation

La méthodologie a été entièrement documenté pendant tout le déroulement de l'essai. Elle se retrouve pour référence dans un blog public sur le site de "github.io"¹¹. Cette méthodologie de documentation permet entre autre, de très facilement documenter, de ne pas perdre des notes très importantes, de suivre le cheminement, de pouvoir retrouver des notes, même si elles ont été effacées ou modifiées, puisque toute modification est sauvegardée dans un repository Git.

Par ailleur, tous les documents de rédaction LaTeX, les images, les scripts et code source qui ont été utiles et utilisés durant l'essai ont été géré dans un repository Git public avec "github.com"¹².

Ces sources d'information viennent bonifier grandement ce rapport et il est même recommandé de s'y référer pour atteindre un certains niveau détails et de compréhension.

4.6 Recherche

4.6.1 Revue de littérature

La revue de la littérature a débuté en octobre-novembre 2019, c'est-à-dire quelques mois après la disponibilité du nano ordinateur (juin 2019). La recherche s'est concentrée sur des références traitantes des concepts du sujet de l'essai : la segmentation sémantique, le temps réel, et les nano ordinateurs. Le premier objectif a été de trouver si des études avaient déjà expérimenté le nano ordinateur, en particulier pour la segmentation de vidéos en temps réel. Pendant cette recherche, j'en ai profité pour effectuer une révision de l'évolution des réseaux de neurones convolutionnels entiers (FCN Fully Convolutional Network) et des différentes architectures, et chercher d'autres solutions de détection de la route en temps réel grâce au FCN.

Il a été assez compliqué de trouver des références intégrant les nano ordinateurs. Comme l'objectif de l'essai est de valider les performances d'un nano ordinateur bien spécifique, les mots-clés "NVIDIA Jetson nano" font partie de la stratégie de recherche.

Les réseaux de neurones convolutifs entiers (FCN) sont implicitement inclus dans les résultats puisque c'est le "state-of-art" actuellement pour répondre au besoin de la segmentation sémantique d'images.

Plus de 75 références ont été collectées. Une quarantaine ont été sélectionnées. Cette sélection peut se décomposer en trois catégories : les références se rapprochant le plus du sujet de l'essai ; l'histoire et les antécédents des réseaux de neurones ; du matériel éducatif pour étudier et manipuler les réseaux de neurones.

Je me suis intéressé aux références des années les plus récentes, autour de 2020, 2019 et 2018, car les avancées dans le domaine des réseaux de neurones sont très rapides. Par curiosité je suis allé aussi parfois voir dans les années bien plus éloignées, comme 1998, où j'ai trouvé un article proposant une solution pour prédire la température de la surface de la route avec des réseaux de neurones.

11. <https://vince71f.github.io/>

12. <https://github.com/vince71f/gae724>

Je n'ai pas pu trouver de références spécifiquement pour la déduction de l'état de la surface (mouillé, gelée, etc.) d'une piste multifonctionnelle (vélo, piéton).

Il est intéressant de noter que la banque de données SCOPUS retourne plus de 11,000 documents avec l'expression "segmentation AND "real-time"". Il y en a plus de 700 uniquement pour l'année 2019.

4.6.2 Étude du site d'implantation

Le nano ordinateur est destiné à être déployé sur le chemin de la piste multifonctionnelle du pont Jacques-Cartier. L'étude du site a permis de chercher à comprendre, parmi ses caractéristiques, les difficultés de son usage l'hiver. Il a été tenté de comprendre les défis et les raisons, techniques, politiques, sécuritaires, de pouvoir la conserver ouverte toute l'année. Une carte du site permet de montrer un exemple de configuration où et comment seront installés les nano ordinateurs, et des images de ces zones d'intérêt permet de "visualiser" ce qui sera interprété par le modèle.

Un mot est réservé pour citer "L'Association des piétons et cyclistes du pont Jacques-Cartier" qui est un acteur actif pour le développement du transport actif dans cette région du Québec, et dont les membres sont des usagers habituels de la piste multifonctionnelle, même l'hiver.

4.6.3 Sélection des données et des modèles de réseaux de neurones

Les ressources mises à disposition par le constructeur du Jetson nano, NVIDIA, ont été étudiées pour apprendre et tester le nano ordinateur. Parmi les plus intéressantes, on peut citer le "Jetson Nano Developer Kit", le "NVIDIA Deep Learning Institute", la communauté Jetson, les tutoriels, les "benchmarks". Des jeux de données sont fournis gratuitement.

En complément des ressources de NVIDIA, deux références scientifiques ont été principalement utilisées comme points de départ et comme modèles pour l'essai, car leurs études ont été faites avec le Jetson nano ([16] et [5]). Beaucoup de références ont été publiées ces deux dernières années sur le sujet de la segmentation sémantique, ils existent donc de multiples alternatives inspirantes.

Internet est une mine d'information. Il existe des forums et des blogues dans lesquels des utilisateurs publient leurs expérimentations de la segmentation sémantique en temps réel avec le Jetson nano ([8]), ou plus génériquement la segmentation sémantique. Des sites comme "modelzoo.co" et "kaggle.com" sont des entrepôts de modèles déjà entraînés.

Une autre option est d'effectuer une recherche d'images ou de vidéos de la piste multifonctionnelle du pont Jacques-Cartier via les sites de recherche tels que Google.

L'Association des piétons et cyclistes du pont Jacques-Cartier existe depuis de nombreuses années pour promouvoir le transport actif et conserver la piste multifonctionnelle du pont Jacques Cartier ouverte durant l'hiver. Ils fournissent, via leurs sites Internet, des collections de vidéos et d'images qui pourraient être utilisées. Il serait aussi possible d'entrer en contact avec l'association et leur demander de prendre de nouvelles vidéos. [9] [10]

Les architectures des modèles FCN sélectionnés pour l'essai sont résumés dans un tableau récapitulatif, incluant leur type, leur application et leurs jeux de données respectifs, précisant les différentes variantes entre résolutions et nombre d'images par seconde (FPS).

4.6.4 Choix de l'équipement pour le nano ordinateur

L'objet d'étude de cet essai est un nano ordinateur. Un nano ordinateur est un ordinateur miniaturisé en taille, mais aussi limité en capacité. Il existe différents fabricants et modèles, de caractéristiques techniques variées, pour répondre à différents besoins. Le dernier né est le modèle "Jetson nano" du fabricant "NVIDIA", disponible depuis juin 2019 au prix très abordable de 99\$US. La compagnie NVIDIA a conçu ce matériel spécialement pour différentes applications d'inférence de modèles d'apprentissage profond sur une plateforme mobile (drone) ou proche des données ("edge" en anglais). Ce modèle a été choisi afin de répondre à l'intérêt que suscitent ses capacités et ses limites. Une image du Jetson nano et un tableau de ses caractéristiques techniques seront disponibles.

L'architecture matérielle sera étudiée et présentée avec l'aide d'images, de diagrammes et de textes explicatifs. Les éléments clés seront identifiés.

Afin d'optimiser les performances du nano ordinateur, une recherche des périphériques les plus adaptés pour répondre aux besoins de performance (et de budget) de l'essai est essentielle, telle que l'alimentation, le stockage, la caméra. Des images des périphériques seront incluses, et les caractéristiques principales seront présentées dans des tableaux.

Le matériel est commandé par le collaborateur de cet essai "Vision météo".

4.6.5 Exploration des solutions logicielles

De même que pour les périphériques, les solutions logicielles nécessaires sont résumés dans un tableau, où il sera indiqué leur nom, leur version, leur licence, les rôles et responsabilités dans l'essai, comme le système d'exploitation, l'environnement de développement, les bibliothèques pour l'inférence, les logiciels de traitements vidéos et d'images.

4.7 Environnement de travail

4.7.1 Préparation du nano ordinateur

L'organigramme de la figure 13 présente les activités qui composent la préparation du nano ordinateur.

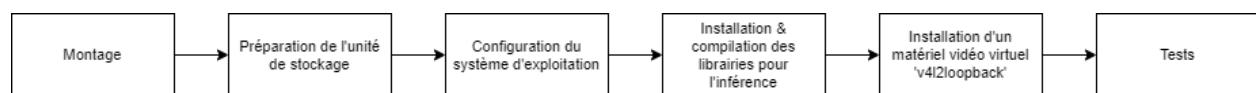


FIGURE 13 – Préparation du nano ordinateur

Montage

Le nano ordinateur est une carte mère livrée sans aucun périphérique ni même boîtier. Vu que les performances logicielles dépendent des performances matérielles, surtout pour une unité telle qu'un nano ordinateur où les capacités matérielles sont très limitées, la première partie de l'essai a été allouée à la sélection des accessoires et périphériques qui vont permettre d'augmenter les performances, protéger et utiliser confortablement le nano ordinateur.

L'organigramme de la figure 14 présente les activités qui composent le montage du nano ordinateur.

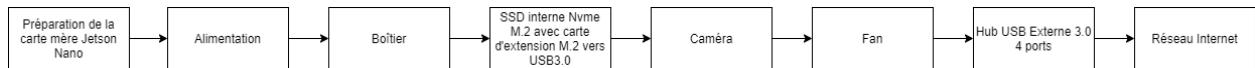


FIGURE 14 – Montage du nano ordinateur

Préparation de la carte mère Jetson Nano



FIGURE 15 – Carte mère du nano ordinateur

Le nano ordinateur qui est livré dans sa boîte est uniquement une carte mère, sans unité de stockage, ni boîtier, clavier, souris, écran, capacité wifi, ou caméra. Il est uniquement livré avec un câble micro USB qui lui permet d'être démarré avec une alimentation minimale de 5 Volts/2Amp et ne consommer que 5 Watts. Aucun système d'exploitation n'est livré non plus. Vu que de l'objectif de l'essai est de tester les capacités du nano ordinateur et que la consommation sera de plus de 5Watt dues aux branchements de multiples périphériques, certaines "broches" sur la carte mère doivent être activées : la broche J48 permet de brancher un adaptateur d'alimentation de 5Volts 4Amp au lieu de l'alimentation micro USB ; et la broche J38 permet d'activer le PoE (Power-Over-Ethernet) afin d'hériter de l'alimentation du câble Ethernet. Aucune autre préparation sur la carte n'est nécessaire.

Alimentation



FIGURE 16 – Adaptateur 5 volts 4 amps

L’alimentation du nano ordinateur est l’élément matériel le plus important du système. De base le nano ordinateur est livré avec un câble micro USB, lui permettant d’être alimenté en 5Volt 2Amp. Mais le besoin en énergie augmente avec les périphériques qui s’accumulent, tel qu’une caméra. Il est prudent de choisir un adaptateur 5Volt 4Amp d’un fournisseur recommandé par NVIDIA, car un changement de puissance sensible en entrée impacte le fonctionnement opérationnel du nano ordinateur. Deux adaptateurs ont été utilisés, l’un recommandé, et l’autre non, afin de tester leur performance.

Dans le cadre de l’essai, l’alimentation du nano ordinateur est utilisée pour alimenter la carte mère, qui comporte entre autres les CPUs, le GPU, le Hub USB 3.0 interne, le contrôleur Ethernet et le port HDMI. Mais aussi la caméra et le ventilateur et optionnellement une carte d’extension M.2 NVMe. Afin d’assister l’adaptateur, un hub USB 3.0 externe a été utilisé pour brancher la souris, le clavier, et à un moment donné le dongle Wifi.

Boitier



FIGURE 17 – Boîtier pour le nano ordinateur

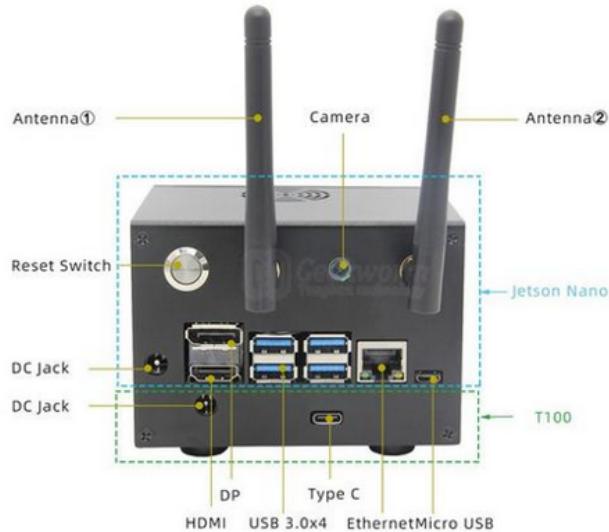


FIGURE 18 – Vue arrière du boîtier pour le nano ordinateur

Afin de protéger le nano ordinateur durant l'essai et l'utiliser dans les conditions les plus proches de son futur mode d'opération, il a été installé dans un boîtier en métal. Le boîtier a été choisi en tenant compte qu'une carte d'extension pour un SSD interne sera installée, ainsi qu'une caméra et un ventilateur. Durant l'essai le nano ordinateur sera manipulé très fréquemment en raison d'un manque d'espace réservé dans la maison. Le boîtier permet donc d'éviter de manipuler le matériel et les connecteurs, les protège, évitant de risquer de les briser, et donc ajouter des délais à l'essai.

Unité de stockage

Le nano ordinateur est conçu pour fonctionner avec un système d'exploitation hébergé sur une carte microSD. Il existe différentes cartes microSD, et certaines sont beaucoup plus performantes que les autres. Malheureusement les cartes microSD ne sont pas destinées à exécuter un système d'exploitation à temps plein, et leur espérance de vie reste très limitée. Étant donné que l'objectif du nano ordinateur est d'être en service continue à l'extérieur, l'utilisation un disque SSD interne comme alternative semble logique.

Carte microSD

Il existe différentes cartes microSD, de multiples constructeurs, et pour différents usages, mais généralement destiné pour stocker des images et vidéos directement par les appareils multimédias. Leur conception est faite pour la manipulation de gros blocs de données, et non des petits fichiers. Trois cartes microSD seront évaluées :

TABLE 4 – Cartes microSD



microSD Samsung EVO 64Gb Plus XC I Grade 3 Class 10



microSD Samsung EVO 64Gb Select XC I Grade 3 Class 10



microSD Scan Disk Ultra 32Gb HC I Class 10

fix
le ta-
bleau

Disque SSD

Un disque SSD et une carte microSD sont différents type de matériel pour différents usages. Le disque SSD est plus adapté pour manipuler les petits fichiers et héberger un système d'exploitation. Il est aussi plus résilient à long terme. C'est donc une option qui ne doit pas être négligée dans le contexte de tests de performance, encore plus avec un nano ordinateur dont les capacités matérielles sont limitées, et qui est un appareil destiné à être continuellement en service et à l'extérieur. L'unité de stockage doit être non seulement performante, mais aussi endurante. Néanmoins, il y a un contreparti important dans la situation d'un nano ordinateur : la consommation d'énergie. Un SSD interne va demander plus d'énergie qu'une carte microSD, et si le nano ordinateur n'est pas capable de gérer correctement les besoins en énergie de ses extensions matériels, le SSD interne risque d'échouer en pleine opération et le nano ordinateur devenir non fonctionnel soudainement.

Un disque SSD interne pour un nano ordinateur est soit une carte d'extension M.2 NVMe ou SATA (selon la carte d'extension), connecté au port PCIe ou USB. Les SSD internes Samsung 970 EVO 250GB NVMe M.2 et Samsung 860 EVO M.2 500GB SATA seront évalués.



FIGURE 19 – Disque SSD NVMe M.2 interne 250GB

Il y a deux choix qui ont été retenus pendant l'essai pour brancher un SSD interne au nano ordinateur : soit via une carte d'extension M.2 MVMe, et connecté via le Hub USB, soit via une carte d'extension M.2 NVMe connectée au port PCIe interne du nano ordinateur, normalement destinée à une carte d'extension Wifi.

Concernant le disque SSD M.2 NVMe connecté à la carte d'extension M.2 via le Hub USB 3.0 interne, le système L4T de NVIDIA ne supporte pas les SSD M.2 NVMe connecté au port USB¹³. Il n'est pas reconnu / détecté, il est donc impossible de le formater, de le partitionner, de l'utiliser. Comme il serait risqué pour l'essai de se lancer dans la recompilation du kernel du L4T, une alternative trouvée sur le développeur forum de NVIDIA est de passer par un adaptateur M.2 MVMe connecté au port PCIe interne.

Malheureusement cette alternative a rapidement été abandonnée. Il a été possible de démarrer et installer le système d'opération sur le SSD M.2, et faire quelques tests, mais pour une raison inconnue, le système n'était pas stable et devenait non opérationnel assez rapidement, le système perdant la connexion au SSD. La durée la plus longue de stabilité observée a été de moins 30 minutes. Une hypothèse est une baisse d'énergie qui survient à un moment et qui impacte l'alimentation du SSD, chaque volt et milliampère étant important pour la stabilité du nano ordinateur. De plus, le raccordement du câble de la carte d'extension M.2 NVMe PCIe avec le SSD M.2 NVMe est très compliqué et risqué pour le câble lui-même. Une autre limitation importante est que cette solution ne permet pas d'utiliser le boîtier, car le SSD M.2 ne rentre pas et ne peut même pas être fixé.

Différentes options pour optimiser l'alimentation ont été explorées : utiliser un HUB USB externe et auto alimenté ; brancher un câble Ethernet au lieu d'utiliser un Dongle Wifi ; allumer le ventilateur dès le démarrage du nano ordinateur ; et l'option de fournir 6Amp directement supportée par la carte mère via les pins ; explorer les solutions sur les forums de discussion^{14 15}.

Caméra

13. À noter que la carte d'extension T100 est discontinuée et remplacée par la T130

14. <https://www.kingston.com/en/community/articledetail/articleid/48543>

15. <https://geekworm.com/products/nvidia-jetson-nano-nvme-m-2-ssd-shield-t100-v1-1>

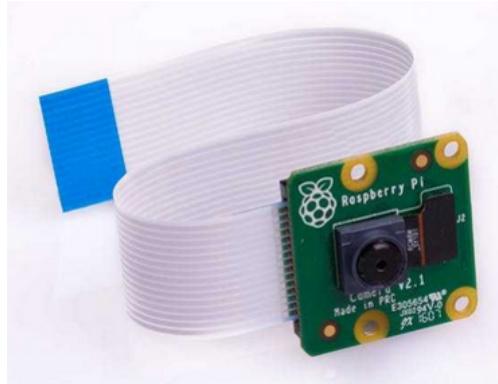


FIGURE 20 – Caméra

L'objectif du nano ordinateur est d'être utilisé pour détecter continuellement les délimitations de la piste cyclable. Il est évident qu'une caméra doit donc faire partie du système et faire partie de l'évaluation des performances. Néanmoins, durant le déroulement de l'essai, la caméra sera très peu utilisée. En effet il n'est pas évident d'être dans un mode de développement directement sur le terrain. Un matériel vidéo virtuel sera utilisé pour simuler la caméra et alimenter l'inférence avec des vidéos préenregistrées, permettant ainsi d'évaluer les performances de l'inférence avec des vidéos, même si d'un point de vue performance matérielle l'utilisation ne sera pas équivalente. Les performances matérielles de l'inférence en temps réel seront évaluées avec la caméra, même si la vue de la caméra n'est pas la piste cyclable, ce qui n'est pas important pour ce test, peu importe ce qui est détecté.

la caméra qui a été sélectionnée est la version 2 de celle du fournisseur Raspberry Pi, le concurrent direct du nano ordinateur NVIDIA Jetson Nano. Cette caméra a été éprouvée avec le temps, est performante, elle semble être la plus adaptée pour ce genre de projet.

Ventilateur



FIGURE 21 – Ventilateur

Un système informatique a besoin d'un ventilateur pour évacuer la chaleur produite par ses processeurs et les autres éléments électroniques, et éviter une faute opérationnelle et des bris de matériel. L'objectif du nano ordinateur étant d'être opérationnel continuellement, et ses éléments

étant contenus dans un boîtier, il est encore plus indispensable d’installer un ventilateur. Le ventilateur choisi a pu être installé dans le boîtier, même si le boîtier ne possède de support pour le fixer. Le ventilateur est capable de démarrer automatiquement au besoin, mais il est volontairement démarré manuellement dès que le nano ordinateur est démarré. Cela évite que la chaleur ne s’accumule, qu’elle soit tout de suite ventilée à l’extérieure, évitant un risque de surchauffe, la capacité du ventilateur étant tout de même limité (petit modèle).

Hub USB externe 3.0 4 ports



FIGURE 22 – Hub USB 3.0 externe autoalimenté

Le nano ordinateur comprend un hub USB 3.0 4 ports internes, les 4 ports étant connectées via le même contrôleur. Ce hub consomme de l’énergie pour alimenter les périphériques qui y sont connectés, comme un SSD interne ou un dongle Wifi, et gérer l’ échange de données. Afin de minimiser les besoins en alimentation et optimiser le plus possible le transfert de données, la souris, le clavier et le dongle USB ont été branchés a un hub USB 3.0 externe autoalimenté. Malheureusement cette option complexifie le déploiement sur le terrain du nano ordinateur. L’alternative pour s’en passer est d’utiliser un câble Ethernet, PoE préférablement, à la place d’un dongle Wifi qui est très gourmant en termes de besoin en alimentation, et chauffe rapidement.

Réseau Internet

Le nano ordinateur comprend un contrôleur Ethernet pour brancher un câble réseau et se brancher sur Internet. Selon la configuration de la carte mère, le nano ordinateur peut hériter de l’alimentation via Ethernet (PoE), via la broche J38. Il comprend aussi un port PCIe interne qui permet de brancher une carte d’extension Wifi. L’autre alternative étant de passer par un dongle USB Wifi, ou un périphérique Wifi externe connecté au port USB.

Dans le cadre de cet essai, le périphérique Wifi externe USB a été utilisé en premier puisque déjà disponible. Malheureusement les performances étaient assez décevantes, le réseau Wifi à la maison n’étant pas non plus très performant dans la pièce où le nano ordinateur était installé (table de la cuisine). Un débit d’environ 5Mbits était disponible. Par curiosité un dongle USB Wifi a été

acquis, mais autant décevant. La meilleure alternative pour améliorer le déroulement de l'essai a été de tirer un câble Ethernet et d'installer un router secondaire, et de brancher le nano ordinateur à ce nouveau router. L'accès Internet a été plus stable et de bien meilleure qualité, la connexion étant d'environ 11Mbs.

Le PoE n'a pas été évalué.

Préparation de l'unité de stockage

Le nano ordinateur est conçu pour fonctionner avec une microSD, et NVIDIA fournit uniquement de la documentation à cet effet. L'option d'utiliser un disque SSD interne est disponible sur Internet, mais n'est pas supporté officiellement par NVIDIA. Il existe néanmoins des articles à ce sujet dans le forum des développeurs¹⁶.

Carte microSD

NVIDIA fournit de la documentation très claire et simple afin de préparer la carte microSD (formatage) et installer l'image du JetPack.¹⁷.

Disque SSD

La procédure d'installation pour installer JetPack sur le SSD interne est disponible sur le site "jetsonhacks.com"¹⁸. À noter qu'une carte microSD est tout de même nécessaire pour "boots-trapper" le système d'exploitation. Il n'est pas nécessaire d'avoir une carte microSD performante puisqu'elle n'est utilisée que pour démarrer le système qui se trouve sur le SSD interne.

Configuration du système d'exploitation

La première fois que le système démarre, le système Ubuntu Linux For Tegra (L4T) doit être configuré avec toutes les options personnalisées (langue, clavier, timezone, etc.).

Installation & compilation des bibliothèques pour l'inférence

Les bibliothèques pour la segmentation sémantique d'images et de vidéos via l'inférence de modèles déjà préparée sont mises à disposition par NVIDIA via un projet dans GitHub. La documentation pour l'installation et l'inférence est disponible directement dans la page GitHub.

Installation d'un matériel vidéo virtuel 'v4l2loopback'

L'inférence fournie par NVIDIA est conçue pour utiliser la caméra du nano ordinateur. Ce qui n'est pas forcément "pratique" pour évaluer la segmentation sémantique d'une vidéo d'une piste

16. <https://forums.developer.nvidia.com/t/how-to-connect-ssd-to-jetson-nano/74053>

17. <https://developer.nvidia.com/embedded/learn/get-started-jetson-nano-devkit#write>

18. <https://www.jetsonhacks.com/2019/09/17/jetson-nano-run-from-usb-drive/>

cyclable. Heureusement un matériel vidéo virtuel permet de simuler la caméra et d'alimenter l'inférence avec une vidéo enregistrée, au lieu de la caméra. La contrepartie concerne l'évaluation des performances : en effet la caméra demande plus de puissance au nano ordinateur que le simulateur logiciel.

Tests

Afin de s'assurer que le nano ordinateur est prêt pour être évalué, des tests matériels et logiciels sont effectués une fois le système monté et stabilisé. Les résultats des tests servent de référence pour évaluer l'état de santé du nano ordinateur.

4.7.2 Collecte des données

TODO

4.7.3 Mise en place des solutions logicielles

Jetson Nano

Le nano ordinateur est destiné à l'inférence. NVIDIA fournit tout un système d'installation, qui est nommé JetPack, et qui contient un système d'exploitation basée sur Ubuntu, Linux For Tegra L4T), le cadre applicatif ('framework') et les librairies nécessaires pour l'inférence, tel que Python, pytorch, les modèles pré-entraînés au format ONNX, le compilateur CUDA, et le SDK TensorRT.

Compute Canada

Le nano ordinateur est destiné à l'inférence, et non l'entraînement de modèles. Il n'est pas non plus destiné à être un environnement de développement. Un autre environnement de travail est donc nécessaire pour développer, et doit posséder les capacités matérielles (GPUs, mémoires, espace de stockage) et logicielles (librairies) pour entraîner un modèle. Heureusement mon directeur de projet m'a introduit à Compute Canada, ou Calcul Québec. Compute Canada fournit un espace de travail puissant aux chercheurs et aux universitaires. Il n'est pas évident de posséder à la maison un environnement permettant de faire de l'apprentissage profond. Ce que je ne pouvais faire avec le nano ordinateur, j'ai pu le faire dans l'environnement de Compute Canada, tel que compiler un fork de torchvision, réentraîner des modèles, générer des onnx. Avoir accès à cet environnement de travail a été un élément déterminant dans le cadre de cet essai.

Compte Compute Canada

Compute Canada mets à disposition des ressources matérielles puissantes et l'accès à des librairies de haute technologie telle que pour l'apprentissage profond, permettant d'avoir un environnement de travail professionnel et performant rapidement. Les ressources matérielles à disposition sont des grappes de serveurs, de CPUs et GPUs de différents types, ainsi que de l'espace de

stockage. Les librairies sont disponibles via un repository privé, et lorsque certaines étaient manquantes (onnx et onnxruntime), j'ai fait une demande par courriel. L'administrateur a pu rendre disponible l'une des deux (onnx), la seconde (onnxruntime) étant beaucoup plus complexe à installé, pour l'avoir tenté sur le nano ordinateur.

L'autre avantage de l'environnement de Compute Canada est la mise à disposition de Jupyter Notebook, afin de tester rapidement du code Python. Par contre il n'est pas conseillé d'exécuter du code nécessitant des délais, tels que l'entraînement d'un modèle.

L'un des irritants est de ne pas pouvoir exécuter un container docker tel quel. Il faut le convertir au format Singularity. Dans le cadre du projet cela m'aurait facilité la tâche, car NVIDIA fournit des docker prêt à l'utilisation pour le réentraînement. Je n'ai malheureusement pas pris le temps et la chance de convertir un container docker au format Singularity. Je ne sais pas si c'est une activité assez simple ou complexe, mais du peu que j'ai lu cela semble assez "rapide".

Jupyter Notebook

Le besoin de tester du code Python est toujours nécessaire. La console Python n'étant vraiment pas conviviale, un environnement Jupyter Notebook est un compromis incontournable. Heureusement Compute Canada fournit un accès à des notebooks depuis Internet, permettant en plus d'hériter de leur environnement de travail. Il est à noter que les notebooks n'ont pas été utilisés pour entraîner un modèle ou générer des onnx, mais de tester du code Python simple, comme visualiser des images, transformer des tensors, et évaluer la segmentation prédictive générée avec le véritable terrain ("ground truth")".

NVIDIA

Compte NVIDIA

NVIDIA mets à disposition tout un écosystème éducatif permettant aux développeurs et aux chercheurs d'obtenir de l'aide au sujet de leur produit et librairies. Dans le cadre de l'essai, un compte NVIDIA a été créé, permettant d'accéder au forum de développeurs, et les containers docker par exemple. Il est aussi possible d'accéder à du matériel éducatif grâce à l'institut DeepLearning de NVIDIA, dont l'accès a été commandité par mon directeur de projet. Le forum de développeurs a été un outil très utile dans le cadre de ce projet, car le dépôt d'une question m'a permis de me débloquer. Je n'étais pas capable de régénérer l'ONNX à partir du code source et de la documentation fournie par NVIDIA pour un modèle FCN. Le développeur principal de l'application a répondu et m'a guidé dans la résolution du problème. Les autres ressources ont eu un impact limité dans le cadre de ce projet, puisque par exemple le container docker et DIGITS n'ont pas pu être utilisé. Le code source des modèles est disponible sans nécessiter de compte, de même que les SDKs Jetpack.

NVIDIA DIGITS

NVIDIA fournit aux développeurs un environnement visuel permettant de réentraîner les modèles FCN qu'ils fournissent avec leurs propres dataset. Cet environnement se nomme DIGITS.

Malheureusement il est nécessaire d'avoir son propre matériel, le système d'exploitation Ubuntu 18.04 LTS, et très recommandé d'avoir au moins un GPU et un ordinateur performant. Ce qui n'est malheureusement pas mon cas. DIGITS ne s'installe pas sur le nano ordinateur, ni sous Windows, ni même un Ubuntu sous windows (WSL). Cette option a donc été abandonnée rapidement.

Docker NVIDIA

NVIDIA fournit aux développeurs des containers docker, avec tout ce qui est nécessaire pour réentraîner un modèle et régénérer un ONNX, par exemple. Malheureusement la capacité du nano-ordinateur ne permet pas de travailler efficacement avec un container docker, le nano-ordinateur devient sans réponse, nécessitant un redémarrage forcé ("hard-reboot"). Cette option a donc été aussi abandonnée rapidement.

NVIDIA DeepStream

Durant le déroulement de l'essai, NVIDIA a mis à disposition un environnement d'apprentissage profond, nommé "DeepStream", facilitant la conception et la génération de modèles, jusqu'à l'inférence. Cet outil n'a pas été évalué, mais pourrait être un outil alternatif pour réentraîner un modèle.

4.8 Évaluation

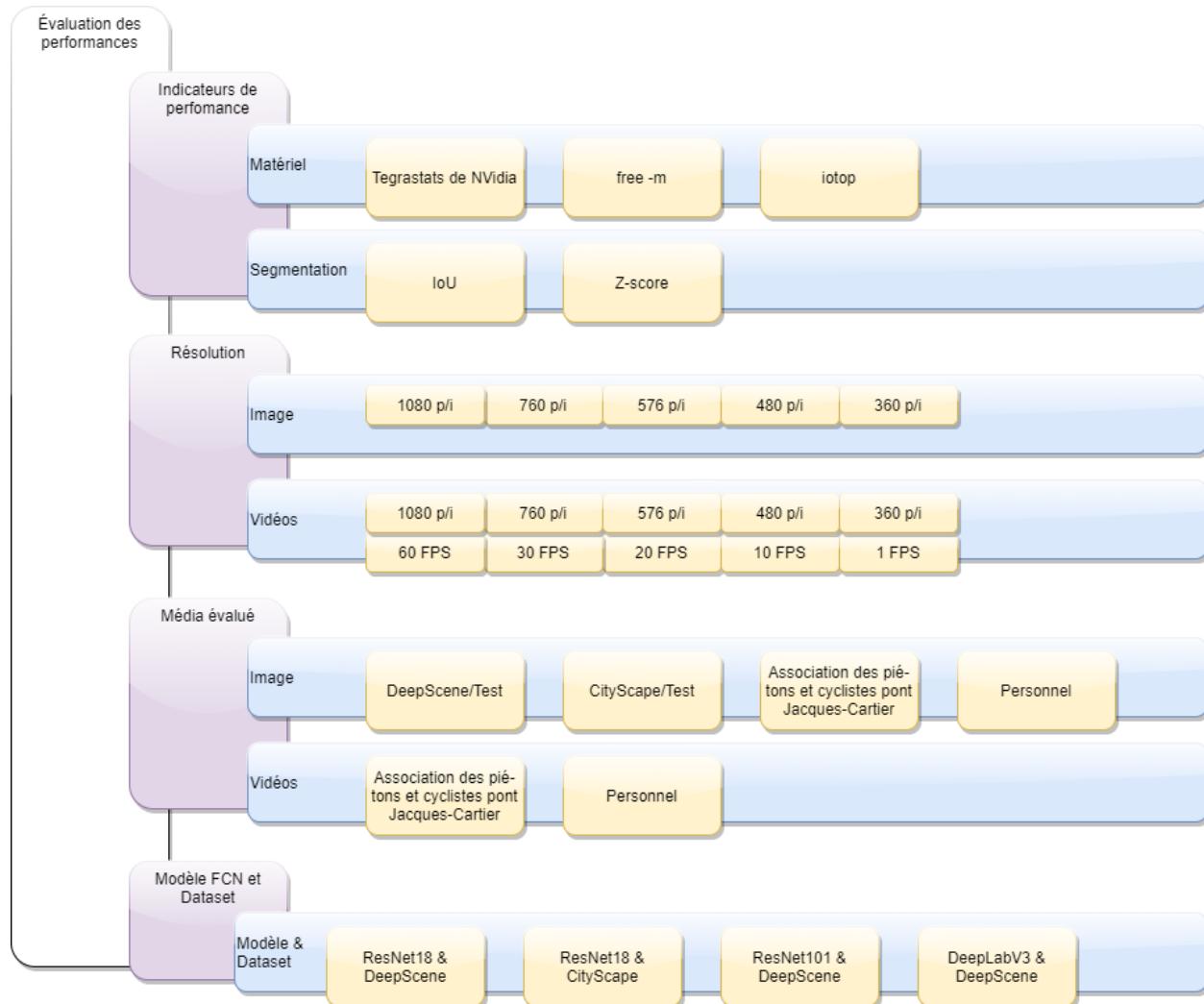


FIGURE 23 – Éléments pour l'évaluation des performances

bonifie

4.8.1 Stratégie de test

L'objectif principal de l'essai est de déterminer la capacité et les limites du nano ordinateur d'inférer en temps réel des modèles de réseau de neurones à convolution entier pour la segmentation sémantique de vidéos. La stratégie qui sera appliquée sera de tester avec divers modèles et divers niveaux de qualité vidéos, en espérant trouver le compromis qui répond le mieux à cet objectif.

1. Afin de s'assurer du bon fonctionnement du nano ordinateur et d'avoir des résultats de référence propre à notre environnement, l'inférence sera testée avec des modèles existants et pré entraînés pour la segmentation sémantique, avec les images et les vidéos provenant des références, et dont les caractéristiques et les résultats sont disponibles.

2. En espérant que les tests de l'étape #1 précédente donnent les résultats documentés dans les articles de références, ils seront repris avec les mêmes modèles, mais avec les images et les vidéos du site d'étude possédant la meilleure qualité acquise (1080p/i, 30FPS). Les données sources (images et vidéos) devront subir certains prétraitements à ce effet, afin de répondre aux requis des modèles.
3. Selon les résultats de l'étape #2, les tests se concentreront sur l'inférence avec des vidéos, en réduisant progressivement la résolution (760p/i, 576p/i, 480p/i, 360p/i) et le nombre d'images par seconde (20FPS, 10FSP, 1FPS).
4. Les étapes intermédiaires de l'étape #3 précédente seront de 1) valider les résultats de l'inférence avec des images avant de tester avec les vidéos, et 2) évaluer si les modèles de réseaux de neurones à convolution entiers doivent et/ou peuvent être adaptés facilement, en tenant compte de l'échéancier de l'essai, et ce afin de répondre à l'objectif principal.

4.8.2 Stratégie de collecte des indicateurs de performance

La méthodologie de la collecte des indicateurs est la suivante¹⁹ :

- La collecte est démarrée après un démarrage frais, manuellement, via un script shell, qui exécute chaque outil, et attend l'interruption du test.
- Chaque outil qui est utilisé pour collecter les mesures, possède son propre fichier.
- La date et l'heure de chaque indicateur collecté sont précisées.
- Afin de faciliter la documentation et l'analyse du test, des points d'intérêt sont ajoutés dans un fichier séparé pour marquer un moment particulier du test, avec la date, l'heure et un libellé. Ce point d'intérêt est fait grâce à une commande "shell" qui vient ajouter une trace dans ce fichier.
- Chaque indicateur est collecté toutes les secondes.
- Une fois le test complété, la collecte est arrêté manuellement.
- Chaque fichier est ensuite transformé en fichier CSV, via des commandes shell.
- À partir des fichiers CSV un script Python génère les graphiques automatiquement.

Chaque indicateur est une colonne du fichier CSV. Il existe le même nombre d'indicateurs à tout moment. La date et l'heure sont un champ.

Avant tout début de tests, la collecte est démarrée sans activité autre que la collecte des indicateurs. Cela permet de prendre une base de référence sans aucune charge.

Ensuite les tests débutent.

Les indicateurs collectés permettent de créer des graphiques qui montrent la progression de chacun.

Les performances matérielles du Jetson Nano sont évaluées grâce à différentes commandes : "tegrastats" fournis par NVIDIA, "free" et "iostop".

Les performances de la segmentation sont évaluées grâce au IoU et au z-score pour la classe du chemin / route. Une fonction Python est utilisée. Les fonctions IoU et le z-score utilisent l'image prédite (généré par le modèle FCN) et l'image vérité terrain ("ground truth"). Les images originales sont donc présélectionnées selon leur intérêt et l'image vérité terrain ("ground truth") créée. L'image prédite et vérité terrain ("ground truth") doivent utiliser la même palette de couleurs et doivent être de la même résolution. Pour les images qui ne possèdent pas d'image vérité terrain

19. <https://vince7lf.github.io/2020/05/26/metrics.html>

("ground truth"), celle-ci est créée à la main avec l'éditeur "Gimp". Comme la résolution de la segmentation de l'image prédite par le modèle de NVIDIA est très faible ("carrée"), l'image vérité terrain ("ground truth") ne sera pas précise au pixel prêt. Le besoin est d'évaluer et non d'entraîner, l'importance de la précision de la classification est moindre dans ce cas.

4.8.3 Segmentation avec des images

TODO

4.8.4 Segmentation avec des vidéos

TODO

4.9 Adaptation

TODO

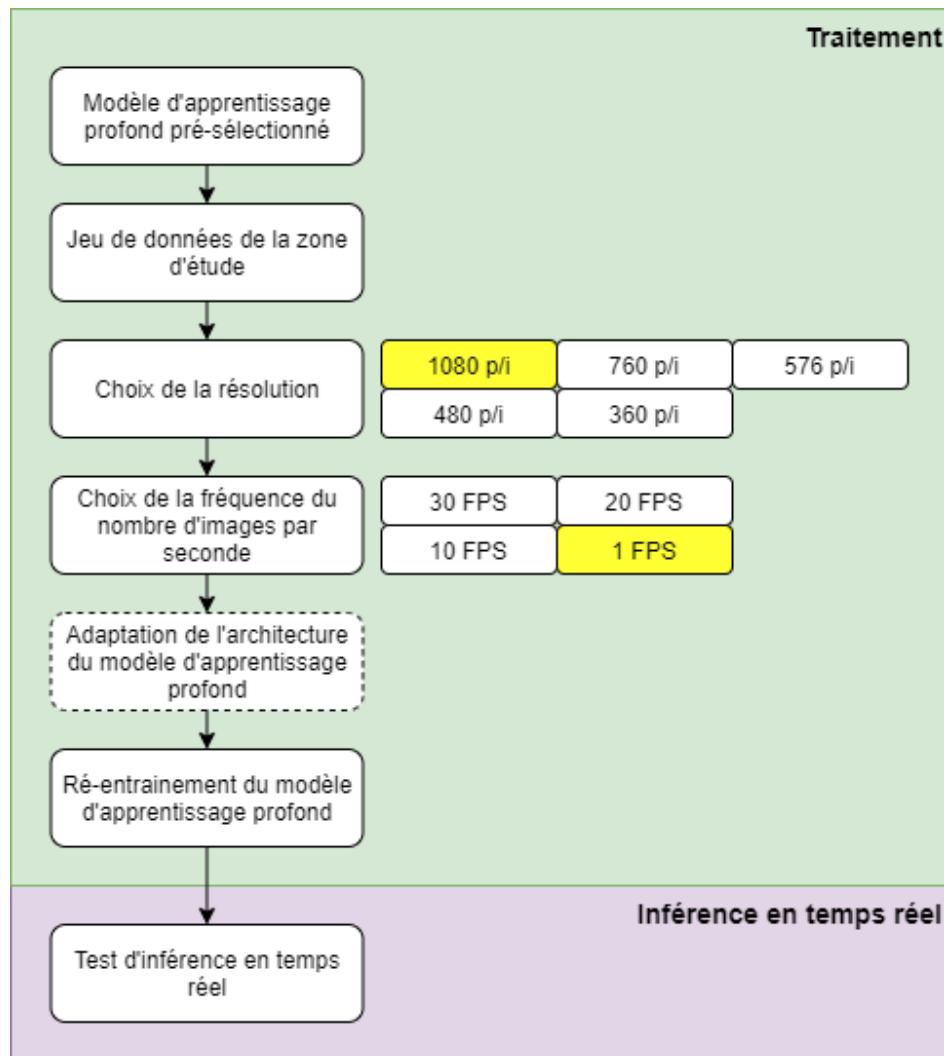


FIGURE 24 – Méthodologie du traitement et adaptation

4.9.1 Choix du modèle de l'architecture FCN

TODO

4.9.2 Adaptation du modèle

TODO

4.9.3 Ré-entraînement du modèle

TODO

5 Résultats

5.1 Performances matérielles

5.1.1 Stockage de données

Pour tester les performances de la microSD et du disque SDD interne M.2 NVMe, l'utilitaire "hdparm" peut être facilement utilisé.

TABLE 5 – Comparaison des performances du "data read" entre un SDD M.2 NVMe et une microSD

| Disk reads | MB | sec | MB/sec |
|--|------|------|--------|
| Samsung 970 EVO Plus 250GB M.2 NVMe Internal Solid State Drive | 1004 | 3 | 334.15 |
| microSD Scan Disk Ultra 32Gb HC I Class 10 | 122 | 3.03 | 40.22 |
| microSD Samsung EVO 64Gb Plus XC I Grade 3 Class 10 (rouge) | 256 | 3.02 | 84.71 |
| microSD Samsung EVO 64Gb Select XC I Grade 3 Class 10 (verte) | 92 | 3.01 | 30.54 |

5.1.2 Performances système

Les diagrammes suivants présentent l'état du nano-ordinateur avant la segmentation, pendant et après. Les indicateurs qui sont observés sont ceux de la mémoire, la fréquence, le I/O, la consommation, la température. Afin de montrer l'impact potentiel de l'application Chromium, elle est démarrée entre deux segmentations, et pendant la segmentation.

La carte microSD "Scan Disk Ultra 32Gb class 10 HC I" a été utilisé pour les tests de performances système. La carte microSD "Samsung EVO 64Gb Plus class 10 HC I" n'était malheureusement plus fonctionnelle au moment des tests, celle-ci ayant été réservé pour tenter d'adapter le modèle aux images terrain locales.

Le test infère en temps réelle la vidéo qui est capturée avec la caméra du nano ordinateur. Le réseau FCN qui est utilisé est celui fournit par NVIDIA "fcn-resnet18-deepscene-576x320". Ce modèle détecte automatiquement la résolution la plus appropriée avec cette caméra, c'est-à-dire 30 image par seconde (FPS) et une résolution de 1920x1080. Le test dure 1400 secondes, un peu de plus de 23 minutes. Il peut se diviser en onze périodes d'observation, qui sont brièvement décrites ci-dessous :

1. La première période est celle entre la 1^{re} seconde et la 200^e seconde, et qui permet d'observer l'état du système au démarrage du nano-ordinateur sans opération mis à part celle de la collecte des statistiques.
2. La seconde période est entre la 200^e seconde et la 400^e, et qui correspond à la première segmentation avec la caméra. Elle permet d'observer le système lors du premier démarrage de la segmentation.
3. La troisième période est celle entre la 400^e seconde et le premier démarrage de Chromium. Elle permet d'observer la réaction du système après l'arrêt de la segmentation.
4. La quatrième période est celle entre le premier démarrage de Chromium et son arrêt. Elle permet d'observer le comportement du système lors de l'utilisation de Chromium, qui est suspecté de ralentir le système, lorsqu'actif (observations faites durant l'essai).
5. La cinquième période est celle entre l'arrêt de Chromium et le démarrage de la seconde segmentation avec la caméra. Cette période permet d'observer la réaction du système après l'arrêt de Chromium.
6. La sixième période est celle entre le démarrage de la seconde segmentation avec la caméra et son arrêt. Cette période permet d'observer la réaction du système pendant la seconde segmentation.
7. La septième période est celle entre l'arrêt de la seconde segmentation et le démarrage de la troisième segmentation avec la caméra. Elle permet d'observer la réaction du système après l'arrêt de la segmentation la seconde fois.
8. La huitième période est celle entre le démarrage de la troisième segmentation et le démarrage de Chromium la seconde fois. Cette période permet d'observer la réaction du système pendant le démarrage de la segmentation la troisième fois.
9. La neuvième période est celle entre le deuxième démarrage de Chromium et son arrêt. Elle permet d'observer le comportement du système lors de l'utilisation de Chromium pendant l'inférence.
10. La dixième période est celle entre l'arrêt Chromium la seconde fois et l'arrêt de la troisième segmentation. Cette période permet d'observer la réaction du système après l'arrêt de Chromium pendant l'inférence.
11. La onzième période est celle entre l'arrêt de la troisième segmentation et l'arrêt du test et de la collecte des statistiques. Elle permet d'observer la réaction du système après l'arrêt de la segmentation la troisième fois.

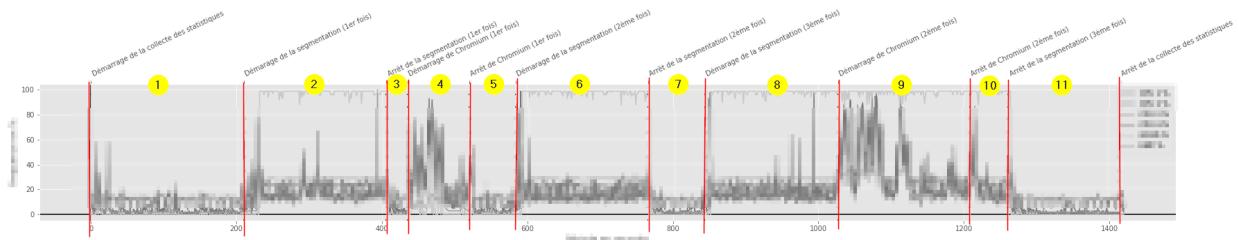


FIGURE 25 – Les périodes du test

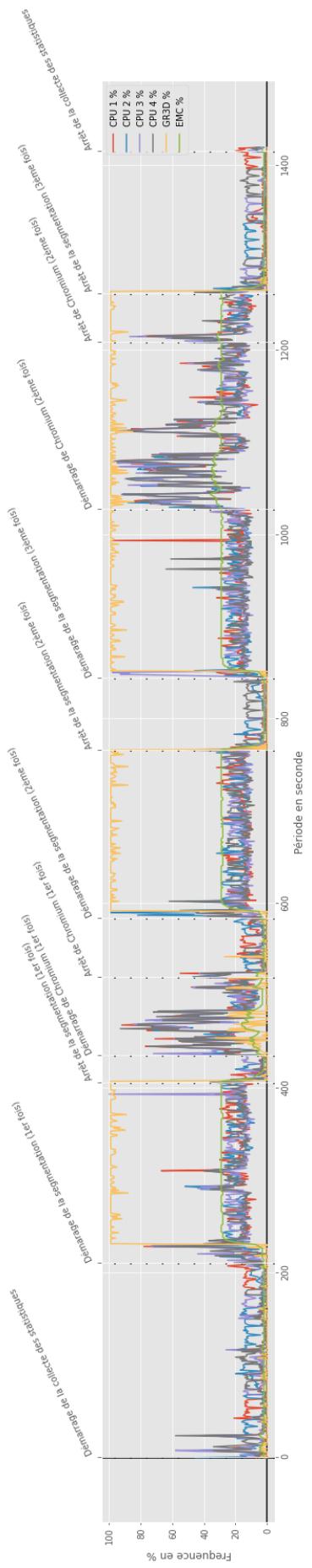


FIGURE 26 – Fréquence

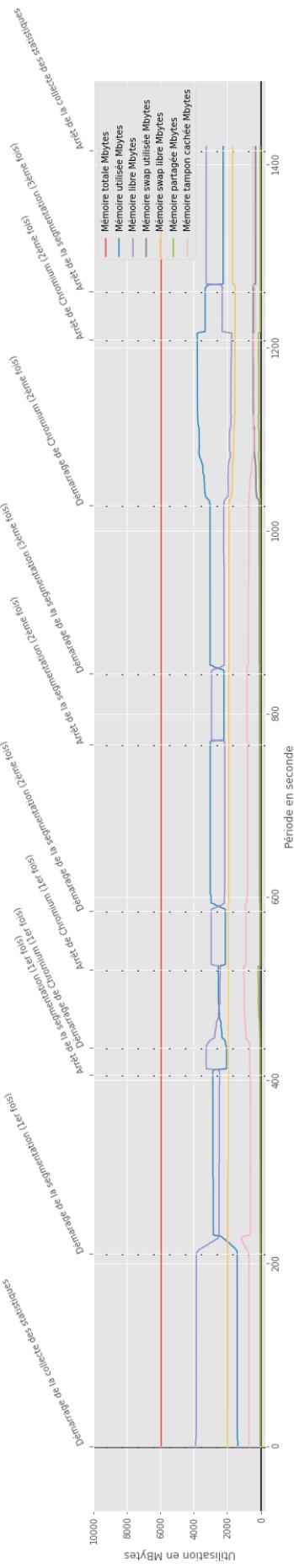


FIGURE 27 – Mémoire

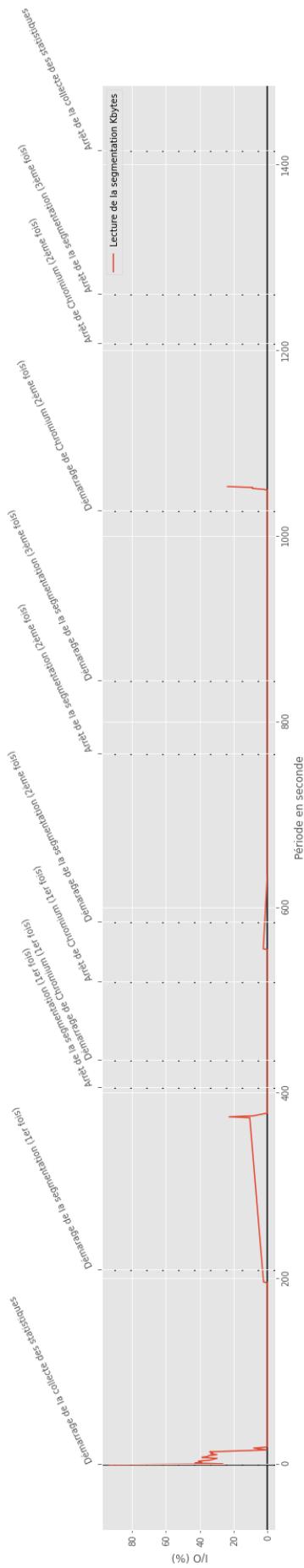


FIGURE 28 – I/O total en % de la segmentation



FIGURE 29 – I/O total en KBytes de la segmentation

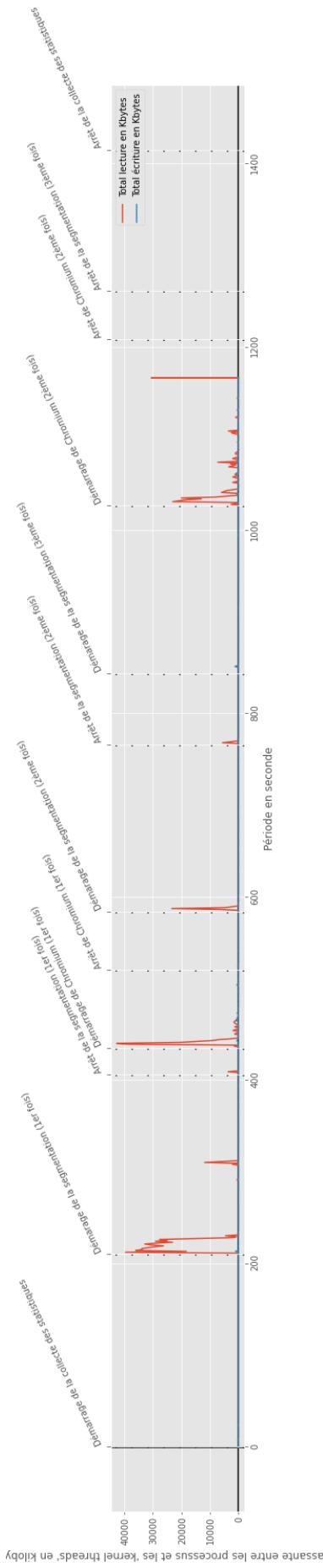


FIGURE 30 – I/O total du disque en KBytes

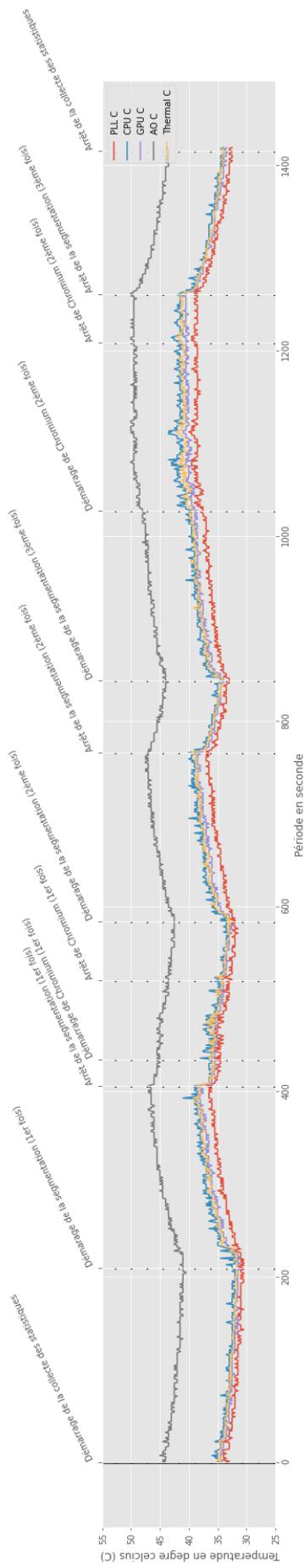


FIGURE 31 – Températures d’opération²⁰

20. PLL : Phase locking loop thermal sensor; AO : Always on thermal sensor. <https://forums.developer.nvidia.com/t/operating-temperature-range-on-jetson-nano/73555/10>

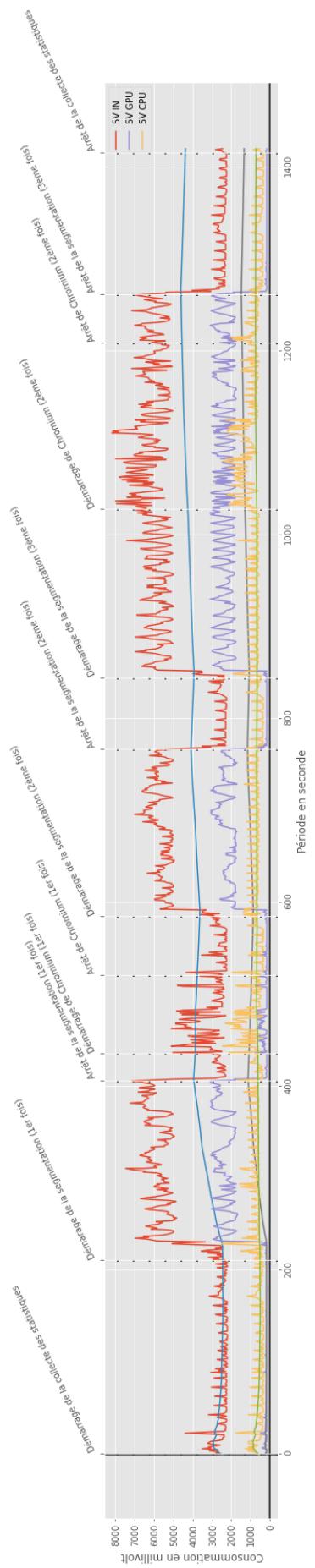


FIGURE 32 – Consommation d’opération

5.2 Performances de l'inférence

5.2.1 Images

Les tests ont été fait avec le modèle "fcn-resnet18-deepscene-576x320" fourni par NVIDIA.

Lors de l'entraînement et l'inférence, le script montre un IoU moyen du modèle de 75%. Mais l'objet d'intérêt de l'essai n'est pas la qualité de la segmentation de l'image complète, mais seulement de la piste cyclable. Certains efforts ont dû être dépensés²¹ afin de pouvoir observer le IoU et le Z-score (Dice) de la segmentation sémantique de la piste cyclable uniquement.

Le résultat de la segmentation sémantique peut-être visualisé avec ces deux photos, prises du jeu de donnée de test de la forêt de Freiburg et utiliser comme jeu de données de test pour le modèle. L'image utilisée possède une version vérité terrain ("ground truth"). L'image générée est l'image prédictive et peut être comparée avec l'image vérité terrain ("ground truth"), tant que la palette de couleur est identique à la version vérité terrain ("ground truth").

Il s'avère que le IoU et Dice score sont assez élevés pour les deux photos.

Ajout de la photo originale, et légende code couleur-classe

TODO

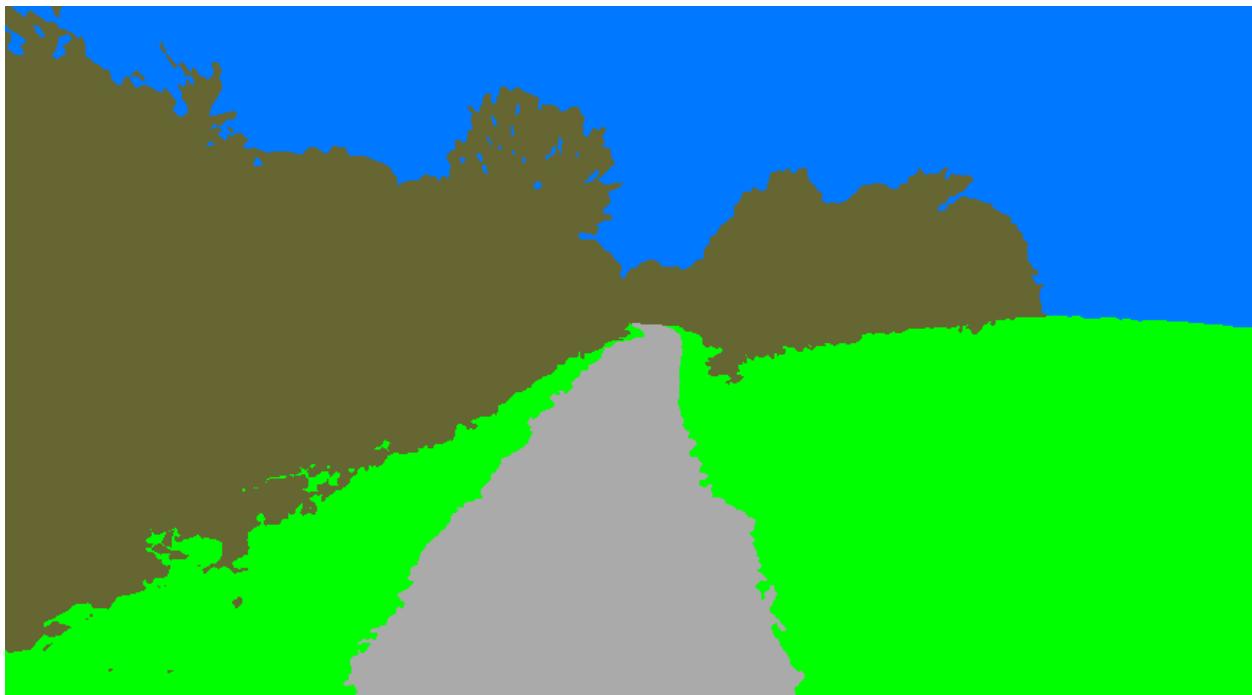


FIGURE 33 – vérité terrain ("ground truth") de l'image b1-09517

21. https://github.com/vince71f/vince71f.github.io/blob/master/_notebooks/2020-06-21-image_pred_color.ipynb



FIGURE 34 – Segmentation sémantique de l'image b1-09517 générée par le modèle. Le IoU et le Dice score pour le chemin sont de +80%.

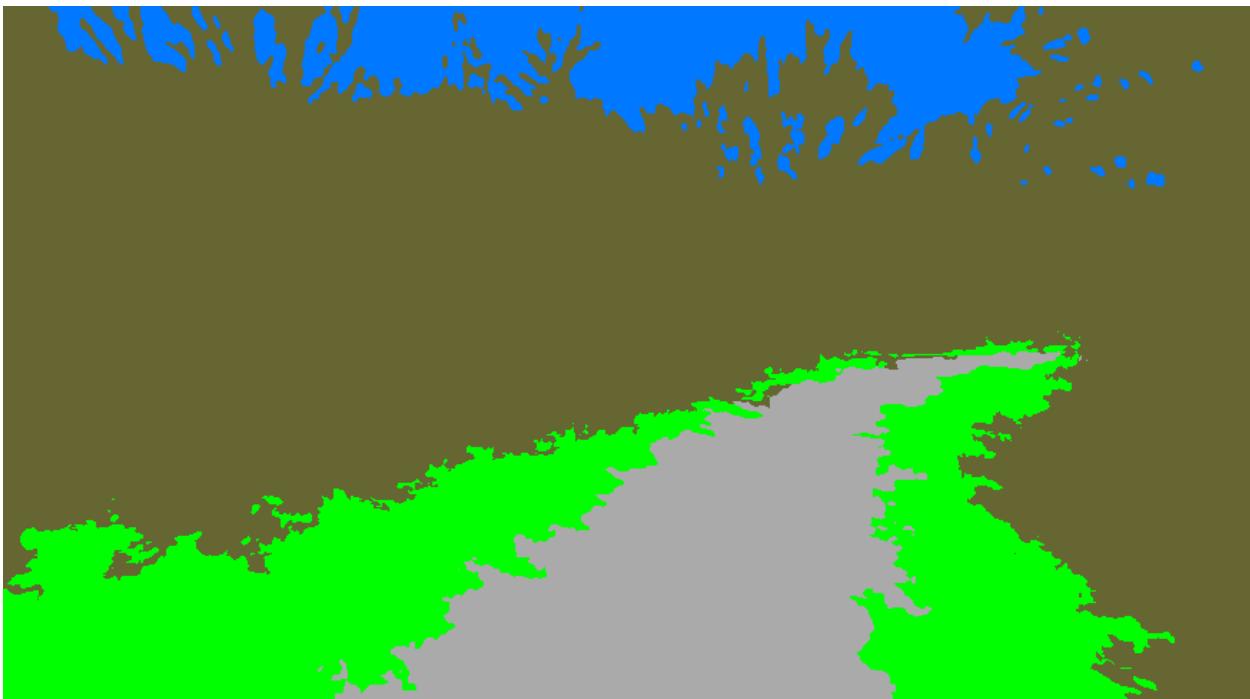


FIGURE 35 – vérité terrain ("ground truth") de l'image b378-61

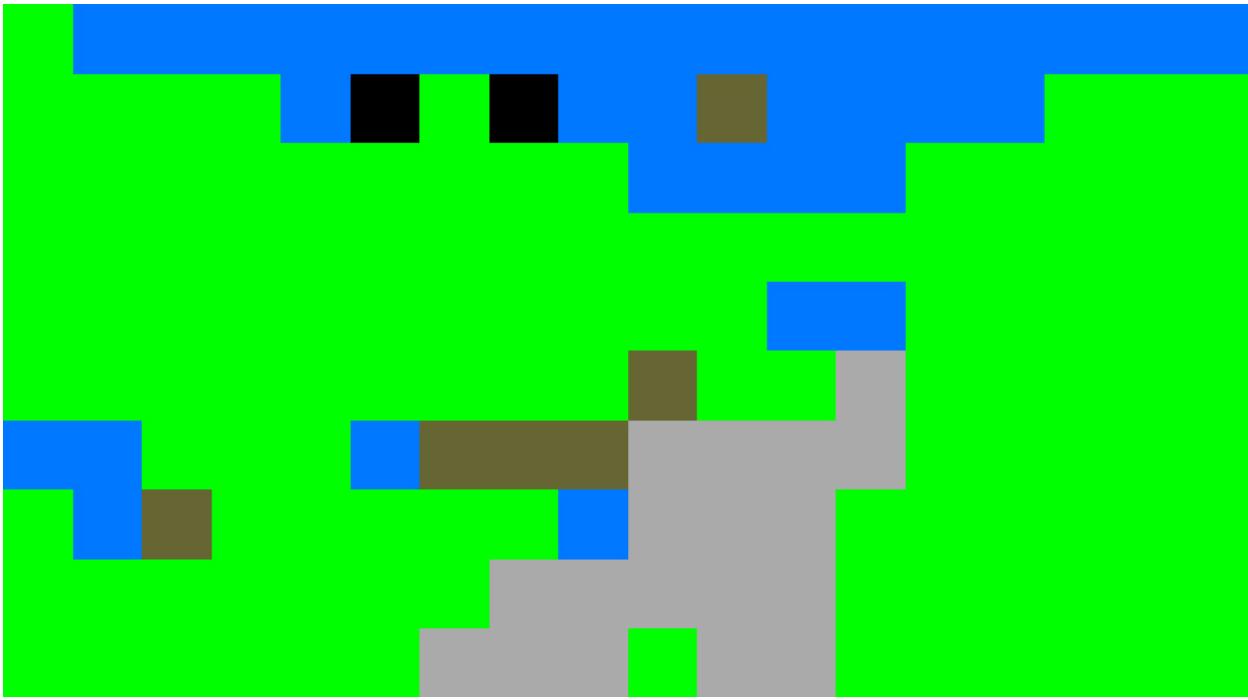


FIGURE 36 – Segmentation sémantique de l'image b378-61 générée par le modèle. Le IoU pour le chemin est +69%.

5.2.2 Vidéos

La segmentation sémantique d'une vidéo de la piste cyclable pour différentes résolutions a été filmée avec un téléphone et est disponible dans le projet Vision Météo Teams de l'université de Sherbrooke. On peut y voir les différentes segmentations se succéder, ainsi que les performances du système et les statistiques "tegrastats".

J'ai tenté de capturer le résultat (vidéos/images) de l'inférence directement depuis le nano, mais ce n'est pas une bonne idée, car trop intrusive. Cela ralentit l'inférence. Il y a en fait deux images qui sont produites par le network : overlay et mask, qui sont directement "rendered" dans un XWindow.

J'ai filmé mon écran avec mon cellulaire. Cela produit une vidéo HD 1080p 30FPS.

- Lien²² vers une courte vidéo de quelques secondes démontrant l'inférence en temps réel de la segmentation sémantique d'une vidéo de la piste cyclable du pont Jacques-Cartier dans des conditions ensoleillée, mais avec un angle de vue qui change rapidement. Inférence en 30FPS 1920x1080 avec le "network" "fcn-resnet18-deepscene-576x320" ;
- Lien²³ vers la vidéo longue de +8 minutes démontrant l'inférence en temps réel de la segmentation sémantique d'une vidéo d'une piste cyclable dans des conditions ensoleillée, mais mouillée et avec de la neige. +8min / 800Mb, une seule vidéo qui montre successivement l'inférence de 60 / 30 / 15 / 1 FPS en 720x1280 / 480x640 / 320x480 / 240x320. Le

22. https://usherbrooke.sharepoint.com/sites/ProjetVisionMto/Documents/%20partages/General/projet_visionmeteo/videos/gae724_lefv2603/resultats/20200221_020044.mp4

23. https://usherbrooke.sharepoint.com/sites/ProjetVisionMto/Documents/%20partages/General/projet_visionmeteo/videos/gae724_lefv2603/resultats/20200412_232155.mp4

"network" est "fcn-resnet18-deepscene-576x320". Il est limité à 26FPS et 576x320.

Voici un tableau montrant les différentes résolutions et images-par-seconde (FPS) qui ont été testés avec le modèle :

TABLE 6 – Résolutions et images-par-seconde (FPS) testés

| Résolutions qui fonctionnent |
|--|
| 320x576, 480x640, 720x1280, 768x1024, 768x1152, 800x1152, 832x1024, 864x1024 |
| Résolutions qui ne fonctionnent pas |
| 832x1120, 832x1152, 768x1280, 800x1280, 864x1152, 900x1152, 900x1280, 960x1600, 1080x1920, 1024x1024 |
| Images-par-secondes (FPS) supportées |
| 60/1, 30/1, 15/1, 1/1 |

6 Interprétation et discussion des résultats

6.1 Performances matérielles

6.1.1 Stockage de données

Les tests montrent que le SSD interne est de 8 fois plus efficace que la meilleure microSD à notre disposition, pour la lecture de données.

6.1.2 Performances système

Performances globales

Concernant les performances globales du nano ordinateur, il est à noter que celui-ci est capable d'exécuter l'inférence en temps réel pour une durée prolongée (23 minutes dans ce cas), et rester réactif aux commandes. L'exemple qui le démontre est le démarrage du navigateur Chromium entre deux segmentations, et pendant la segmentation.

Fréquence

La commande "tegrastats" offre la fréquence des CPU (4 pour le nano ordinateur), le GR3D (GPU) et EMC. On peut noter que l'inférence prend 100% du GR3D pendant toute la durée. Les CPUs sont tous utilisés équitablement pendant l'inférence, en dépassant rarement les 30% d'utilisation. En fait la période qui montre une exploitation élevée des CPUs est lors de l'utilisation de Chromium, où l'ensemble des CPUs sont employés entre 0% et 90%.

Il faut donc rester vigilant quand à l'utilisation des CPUs pendant l'inférence sur le long terme, au risque de perdre le système en raison d'un ralentissement progressif dû à un manque de ressources processeurs CPUs.

Mémoire

La commande "free -m" offre l'utilisation mémoire du système en Mb. Le nano ordinateur au démarrage ne consomme qu'environ 1.5Gb de mémoire totale, et possède 4Gb de libres sur un total de 6 Gb. À la fin du test de 25 minutes, il ne reste qu'environ 3Gb de mémoire libre, un peu plus de 2Gb semble resté utilisé. De la mémoire swap a commencée à être consommée lors du démarrage de Chromium pendant la 3e segmentation, et ne semble jamais avoir été libérée. La mémoire tampon cachée est aussi sensiblement utilisée et revient un peu en dessous de son niveau original à la fin du test.

De même que pour l'utilisation des processeurs, il semble être préférable de rester vigilant lors de l'utilisation opérationnelle du nano ordinateur, la segmentation consommant de la mémoire qui semble ne plus être disponible pour les autres ressources du système, comme le démontre l'état de la mémoire totale libre à la suite de l'arrêt de la 1re segmentation.

I/O

La commande "iostat" offre les performances I/O du nano ordinateur pendant le test de 25 minutes. Le I/O de la segmentation est très raisonnable, de même que celle du système. Il n'y a quasiment pas d'opération visible en écriture, même la collecte des statistiques durant le test, aux secondes, n'apparaît pas. Les opérations en lecture sont plus visibles, mais très ponctuelles. La période la plus occupée en lecture semble être due durant le démarrage de la segmentation la première fois : le système semble lire le modèle en mémoire, et le conserver en mémoire, car les opérations en lecture suivantes sont peu ou non visibles pendant le démarrage des segmentations suivantes.

Cela expliquerait l'augmentation de l'utilisation de la mémoire à la suite de la segmentation.

Température

La commande "tegrastats" offre grâce à des capteurs intégrés à la carte mère la température de différents éléments matériels du nano ordinateur. La commande "sudo jetson_clock" est démarrée manuellement dès que le système est démarré, permettant de profiter de la fréquence maximale d'utilisation supportée par le nano ordinateur. Le succès de la commande est simple à vérifier : le ventilateur se met à ventiler aussitôt²⁴.

La température dans la pièce au moment du test est de 27C. Au démarrage, on note que la température mesurée de la plupart des capteurs thermiques, sauf pour le AO ("Always on") est entre 33C et 36C. Le démarrage de la 1re segmentation fait graduellement monter la température, entre 37C et 39C, jusqu'au point d'arrêt de la segmentation, après 200 secondes approximativement, et qui diminue graduellement approximativement pendant 200 secondes vers son point d'origine

24. https://docs.nvidia.com/jetson/14t/index.html#page/Tegra\%20Linux\%20Driver\%20Package\%20Development\%20Guide/power_management_nano.html

lorsqu'elle est arrêtée. Le démarrage de Chromium pendant cette période semble ralentir un peu le refroidissement. L'observation lors de la seconde segmentation est identique à la première. La troisième segmentation est plus longue, 400 secondes, et voit la température se stabiliser entre 41C et 43C. L'arrêt de la segmentation voit la température baisser et revenir assez rapidement à sa température d'origine.

Le capteur thermique AO ("Always on") est plus particulier, puisqu'il mesure une température de 10C supérieures aux autres capteurs. Selon le modérateur Trumany de NVIDIA²⁵, "AO_therm is used for a truly robust thermtrip and as an LP0 wake source, as other zones will cease to operate during LP0.". Mes compétences en la matière ne me permettent pas d'expliquer clairement ce renseignement, mais cela semble signifier que ce capteur est plus robuste que les autres et devient l'indicateur de référence pour gérer une surchauffe.

Il est donc à noter que l'opérationnalisation constante de la segmentation aurait un impacte non négligeable sur la durée de vie du Jetson nano. Selon la documentation de NVIDIA, une carte Jetson Xavier TX2i qui opère 24/7, selon certaines conditions, a une durée de vie théorique de 4,4 années²⁶

Au besoin, plus d'informations peuvent être trouvées dans le guide de conception thermique du Jetson Nano²⁷

Consommation

La commande "tegrastats" offre de visualiser la consommation du nano ordinateur, soit globale, pour les CPUs et pour les GPU. En mode opérationnel continu, cela peut avoir une importance sur le budget, car la consommation est clairement beaucoup plus élevée pendant la segmentation. Il peut être observé aussi qu'elle est beaucoup plus volatile avec Chromium démarée.

6.2 Performances de la segmentation

6.2.1 Images

TODO

6.2.2 Vidéos

TODO

7 Conclusion et recommandations

TODO

Voici le plan qui est utilisé pour rédiger la conclusion.

— Synthèse des réussites et des échecs de l'essai par rapport aux objectifs ;

25. <https://forums.developer.nvidia.com/t/operating-temperature-range-on-jetson-nano/73555/10>

26. https://docs.nvidia.com/jetson/14t/index.html#page/Tegra\%2520Linux\%2520Driver\%2520Package\%2520Development\%2520Guide\%2Fjetson_module_support.html

27. https://developer.download.nvidia.com/assets/embedded/secure/jetson/Nano/docs/Jetson_Nano_Thermal_Design_Guide_TDG-09383-001_v1.3.pdf?2P65awpy13RwXu6jWjsqFgresjNSqh0-N2uI3BPNH2Wcbp9LNh91GF3UtmC3JgEWd6MX2-BC5xoL80tY5Wp15cEl1IMR4IawEflJehkxKH3yDAgxV-HpXyOo5Ge8a32mdntMcfRzjRZZTP2-hsJ1IuT5FB7G36zHkCva7uPS9ntgWDff-w1W0LBJLH5DvpE1qU-3yZM5hjsz9g9cpFM

- Synthèse des capacités et des limites du nano ordinateur "Jetson nano" pour l'inférence en temps réel à des fins de segmentation sémantique de vidéos ;
- Synthèse des capacités et des limites des modèles de réseaux de neurones pour la segmentation sémantique en temps réel de la zone d'études ;
- Recommandations ;

Bibliographie

- [1] S. ABOUZAHIR, M. SADIK et E. SABIR. “IoT-Empowered Smart Agriculture : A Real-Time Light-Weight Embedded Segmentation System”. In : *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)* (2017), p. 319-332. DOI : 10.1007/978-3-319-68179-5_28.
- [2] A. BEAM. *Deep Learning 101 - Part 1 : History and Background*. 2017. URL : https://beamandrew.github.io/deeplearning/2017/02/23/deep_learning_101_part1.html.
- [3] M. BERNAS, B. P{\textbackslash}LACZEK et A. SAPEK. “Edge Real-Time Medical Data Segmentation for IoT Devices with Computational and Memory Constraints”. In : *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)* (2017), p. 119-128. DOI : 10.1007/978-3-319-67077-5_12.
- [4] B. BLANCO-FILGUEIRA et al. “Deep Learning-Based Multiple Object Visual Tracking on Embedded System for IoT and Mobile Edge Computing Applications”. In : *IEEE Internet of Things Journal* (juin 2019), p. 5423-5431. DOI : 10.1109/JIOT.2019.2902141.
- [5] C. P. CHONG, C. A. T. SALAMA et K. C. SMITH. “Real-Time Edge Detection and Image Segmentation”. In : *Analog Integrated Circuits and Signal Processing* (1992), p. 117-130. DOI : 10.1007/BF00142412.
- [6] M. COPEL. *What's the Difference Between Deep Learning Training and Inference ?* Août 2016. URL : <https://blogs.nvidia.com/blog/2016/08/22/difference-deep-learning-training-inference-ai/>.
- [7] T. DETTMERS. *Deep Learning in a Nutshell : History and Training*. Déc. 2015. URL : <https://devblogs.nvidia.com/deep-learning-nutshell-history-training/>.
- [8] F. DUSTIN. *Realtime Semantic Segmentation on Jetson Nano in Python and C++*. Oct. 2019. URL : <https://www.linkedin.com/pulse/realtime-semantic-segmentation-jetson-nano-python-c-dustin-franklin>.
- [9] Association des piétons et cyclistes du pont JACQUES-CARTIER. *PontJacques-Cartier365.com*. 2020. URL : <http://pontjacquescartier365.com>.
- [10] Association des piétons et cyclistes pont JACQUES-CARTIER. *Flickr Association des piétons et cyclistes pont Jacques-Cartier*. 2020. URL : <https://www.flickr.com/photos/pontjacquescartier>.
- [11] JIACONDA. *A Concise History of Neural Networks*. Avr. 2019. URL : <https://towardsdatascience.com/a-concise-history-of-neural-networks-2070655d3fec>.
- [12] J. Y. KOH. *Model Zoo - Deep Learning Code and Pretrained Models for Transfer Learning, Educational Purposes, and More*. 2018. URL : <https://modelzoo.co/>.
- [13] A. KURENKOV. *A 'Brief' History of Neural Nets and Deep Learning*. 2015. URL : <http://www.andreykurenkov.com/writing/ai/a-brief-history-of-neural-nets-and-deep-learning/>.

- [14] J. LONG, E. SHELHAMER et T. DARRELL. “Fully Convolutional Networks for Semantic Segmentation”. In : *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. Juin 2015, p. 3431-3440. DOI : 10.1109/CVPR.2015.7298965.
- [15] M. MODY et al. “Low Cost and Power CNN/Deep Learning Solution for Automated Driving”. In : *Proceedings - International Symposium on Quality Electronic Design, ISQED*. 2018, p. 432-436. DOI : 10.1109/ISQED.2018.8357325.
- [16] T. NGUYEN et al. “MAVNet : An Effective Semantic Segmentation Micro-Network for MAV-Based Tasks”. In : *arXiv :1904.01795 [cs]* (juin 2019). URL : <http://arxiv.org/abs/1904.01795>.
- [17] NVIDIA. *Jetson Nano*. Mar. 2019. URL : <https://developer.nvidia.com/embedded/jetson-nano>.
- [18] NVIDIA. *Jetson Nano : Deep Learning Inference Benchmarks*. Avr. 2019. URL : <https://developer.nvidia.com/embedded/jetson-nano-dl-inference-benchmarks>.
- [19] D. PATHAK et M. EL-SHARKAWY. “Architecturally Compressed CNN : An Embedded Realtime Classifier (NXP Bluebox2.0 with RTMaps)”. In : *2019 IEEE 9th Annual Computing and Communication Workshop and Conference (CCWC)*. Jan. 2019, p. 0331-0336. DOI : 10.1109/CCWC.2019.8666495.
- [20] PJCCI. *Fiche de la piste multifonctionnelle du pont Jacques-Cartier*. 10 oct. 2018. URL : https://jacquescartierchamplain.ca/wp-content/uploads/2018/10/IMG_Fiche_piste-multi_pont_JC_FR_vfinale_web_2018-10-10.pdf.
- [21] PJCCI. *Rapport post-mortem sur le projet pilote d'entretien hivernal de la piste multifonctionnelle du pont Jacques-Cartier*. 10 oct. 2018. URL : https://jacquescartierchamplain.ca/wp-content/uploads/2018/10/RPP_piste_PJC_2018-10-10-1.pdf.
- [22] N. SHARMA, M. SHAMKUWAR et I. SINGH. *The History, Present and Future with IoT*. Intelligent Systems Reference Library. Springer Science et Business Media Deutschland GmbH, 2019. ISBN : 18684394 (ISSN). DOI : 10.1007/978-3-030-04203-5_3.

8 Annexes

TODO