Université du Québec

Institut national de la recherche scientifique

CENTRE ÉNERGIE MATÉRIAUX TÉLÉCOMMUNICATIONS

REGION-BASED SEGMENTATION OF ARCTIC ICE, LAND, SLUSH, AND WATER FROM SHORELINE CAMERAS

SEGMENTATION HIÉRARCHIQUE DE LA GLACE, DE LA NEIGE FONDANTE ET DE L'EAU À PARTIR D'IMAGES CÔTIÈRES

Par

Aryan Kargwal

Mémoire présenté pour l'obtention du grade de

Master en Télécommunications

Jury d'évaluation

Président du jury et examinateur interne	Tarek Djerafi, Institut National de la Recherche Scientifique
Examinateur externe	Abdessamad Ben Hamza, Université Concordia
Directeur de recherche	Shervin Vakili, Institut National de la Recherche Scientifique
Co-directeur de recherche	Saeid Homayouni, Institut National de la Recherche Scientifique

REMERCIEMENTS

First, I am grateful to **God** for granting me the health, strength, and patience to carry out this research. None of this would have been possible without His guidance.

I would like to express my deepest gratitude to my supervisors, **Dr. Shervin Vakili** and **Dr. Saeid Homayouni**, for their guidance, encouragement, and consistent support throughout this project. Their expertise and mentorship not only shaped the direction of this thesis but also significantly influenced my growth as a researcher.

A special thanks goes to **Jimmy Poulin**, whose work on the **CAIMAN project** and long-term image collection made this research possible. The availability of high-quality Arctic shoreline imagery was essential to developing and evaluating the segmentation pipeline, and this thesis would not have been feasible without his efforts.

I am also grateful to **Kaushik Roy** for his ongoing support, collaborative spirit, and technical insights throughout the development process.

I sincerely thank **INRS–Centre Énergie Matériaux Télécommunications (EMT)** and **INRS–Centre Eau Terre Environnement (ETE)** for providing the academic environment and research infrastructure that supported this work. I also acknowledge the **Fondation INRS** for its institutional support and encouragement.

Special thanks to **INRS Foundation and Glencore** for financially sponsoring this research, and to **Mitacs** for awarding me the **Globalink Graduate Fellowship**, which helped make my master's studies in Canada possible.

Finally, to my parents, my brother, friends, colleagues, and the broader INRS community who offered advice, support, and motivation along the way — thank you.

RÉSUMÉ

Ce mémoire présente un pipeline de segmentation d'images modulaire développé pour classifier des images du littoral arctique en cinq types de surface : terre, ciel, glace solide, gadoue et eau libre. Ce travail s'inscrit dans le cadre du projet CAIMAN — un réseau de caméras terrestres déployées à travers le Nunavik pour surveiller les conditions côtières dynamiques. Ces images sont visuellement complexes et souvent ambiguës, avec des reflets, des régions sans texture et des surfaces de transition qui posent problème aux méthodes de segmentation conventionnelles.

Les premières expérimentations avec des modèles établis, tels que U-Net, PSPNet et DeepLabv3+, ont révélé leurs limites face aux frontières incertaines et à l'apparition rare de certaines classes. Les fines couches de gadoue, la glace masquée par les reflets et les transitions à faible contraste entre l'eau et la glace mouillée entraînaient fréquemment des prédictions erratiques ou excessivement confiantes. En réponse, ce mémoire propose un cadre de segmentation basé sur les régions et les graphes, qui s'éloigne de l'étiquetage au niveau des pixels pour adopter une approche de raisonnement structurel en contexte d'ambiguïté.

Le pipeline final intègre la génération de superpixels à l'aide de SAM-ViT ou SLIC, la construction d'un graphe d'adjacence régionale (RAG), la fusion de régions basée sur des descripteurs, et la classification régionale avec DeepLabv3+. Notamment, la capacité de sur-segmentation de SAM permet de générer des propositions régionales en zero-shot, même dans des cas extrêmes ou inconnus, sans nécessiter d'ajustement de prompt ou de supervision spécifique à une classe.

Au fil de trois itérations d'amélioration du jeu de données — passant de l'annotation manuelle à l'étiquetage assisté par FastSAM — le pipeline a évolué vers un système robuste, capable de généraliser à travers différentes conditions d'éclairage, structures de scène et incertitudes spatiales. Évalué à l'aide de métriques standards (IoU, F1, précision et rappel) et d'une analyse qualitative des échecs, le pipeline a démontré des améliorations significatives en précision par classe et en interprétabilité, en particulier dans les scènes fortement réfléchissantes ou fragmentées.

Au-delà de ses contributions techniques, ce travail propose un paradigme de conception pour la segmentation en environnements incertains : une approche qui considère l'ambiguïté comme un signal à modéliser, structurer et raisonner. Il met en avant la valeur des pipelines modulaires et explicables dans des domaines où les modèles conventionnels peinent. Il offre une base évolutive pour des travaux futurs en surveillance environnementale, vision géospatiale, capteurs autonomes et robotique.

ABSTRACT

This thesis presents a modular image segmentation pipeline developed to classify Arctic shoreline imagery into five surface types: land, sky, solid ice, slush, and open water. The work builds on the CAIMAN project — a network of land-based cameras deployed across Nunavik to monitor dynamic coastal conditions. These images are visually complex and often ambiguous, with glare, textureless regions, and transitional surfaces that challenge conventional segmentation methods.

Early experiments with established models, such as U-Net, PSPNet, and DeepLabv3+, revealed their limitations in handling uncertain boundaries and rare class appearances. Thin slush layers, glare-obscured ice, and low-contrast transitions between water and wet ice frequently led to erratic or overconfident predictions. In response, this thesis proposes a region-first, graph-based segmentation framework that shifts away from pixel-level labeling and adopts a structural reasoning approach in the face of ambiguity.

The final pipeline integrates superpixel generation using either SAM-ViT or SLIC, region adjacency graph (RAG) construction, feature-based region merging, and region-wise classification with DeepLabv3+. Notably, the use of SAM's oversegmentation capability enables zero-shot region proposal generation, even in previously unseen or edge-case scenarios, without requiring prompt tuning or class-specific supervision.

Over three iterations of dataset refinement — from manual annotation to FastSAM-assisted labeling — the pipeline matured into a robust system capable of generalizing across variable lighting conditions, scene structures, and spatial uncertainties. Evaluated through standard metrics (IoU, F1, precision, and recall) and qualitative failure analysis, the pipeline demonstrated significant improvements in class-wise accuracy and interpretability, particularly in scenes with high glare or fragmentation.

Beyond its technical contributions, this work offers a design paradigm for segmentation in uncertain environments: one that treats ambiguity as a signal to be modeled, structured, and reasoned through. It underscores the value of modular, explainable pipelines in domains where conventional models struggle. It provides a scalable foundation for further work in environmental monitoring, geospatial vision, autonomous sensors, and robotics.

SOMMAIRE RÉCAPITULATIF

1. Introduction

1.1 Historique de ce mémoire

Ce mémoire est issu d'un stage Mitacs Globalink réalisé à l'été 2022 sous la supervision du professeur Saeid Humayouni. Ce stage portait sur le projet CAIMAN — Caméras aux Infrastructures Marines au Nunavik — un réseau de sept caméras RGB fixes géré par le Gouvernement régional Kativik, l'INRS, et Transports Québec. L'objectif du stage — créer des masques annotés pour la glace, la gadoue et l'eau libre — s'est révélé complexe : les reflets, la neige en mouvement et la glace sans texture ont produit des annotations sur lesquelles aucun annotateur ne s'accordait.

À la fin du stage, les vidéos partiellement annotées sont devenues la base de mon projet de maîtrise à l'INRS, co-supervisé par le professeur Humayouni et le professeur Shervin Vakili. Durant la maîtrise, le jeu de données a connu trois itérations successives :

- **Version 1** (~500 masques manuels) a montré à quel point les frontières de la gadoue peuvent disparaître.
- **Version 2** (~23 000 polygones Roboflow) a augmenté le volume mais introduit un biais de position.
- **Version 3** (~8 000 masques assistés par FastSAM) a équilibré les saisons et amélioré la précision des bords, mais contenait encore du bruit là où la gadoue rencontre l'eau.

Ces itérations ont permis de comprendre pourquoi les modèles conventionnels échouent et quel type de pipeline serait nécessaire pour réussir.

1.2 Motivation and Context

Des cartes précises, quasi en temps réel, de la glace solide, de la gadoue et de l'eau libre sont cruciales pour les planificateurs de routes marines, les chasseurs autochtones, les chercheurs en climat et les équipes de gestion des risques au Nunavik. Les caméras CAIMAN fournissent des images toutes les quinze minutes (**Figure 1.1**) ; cependant, la classification de chaque image doit être faite localement, sans traitement en nuage ni capteurs auxiliaires.

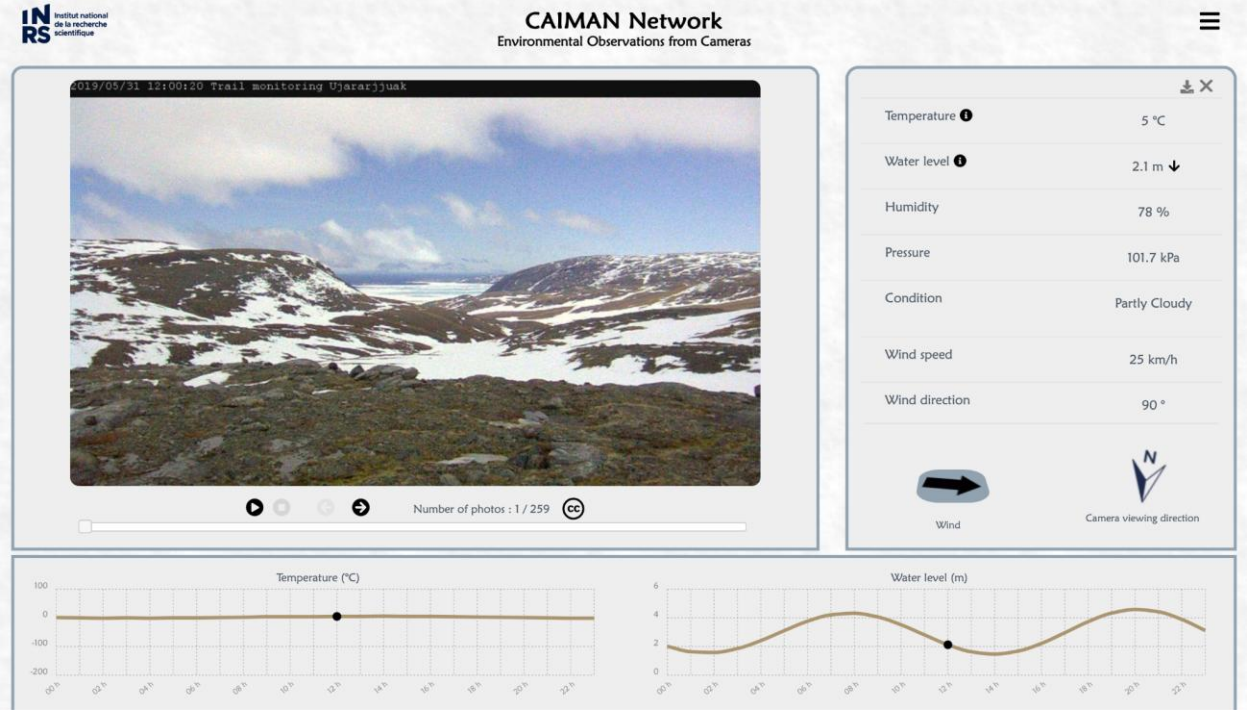

Figure 1.1 *Tableau de bord CAIMAN affichant le flux des caméras et les métadonnées environnementales.*

Les premiers essais avec **Mask R-CNN [1**] et **U-Net [2]** ont été insuffisants : les reflets masquaient les bords, les couches fines de gadoue disparaissaient des prédictions, et les sorties variaient d'une image à l'autre. Une étude antérieure sur l'estimation de profondeur d'**AdaBins**, qui extrait la structure à partir d'une seule image RGB via un regroupement adaptatif [3], a suggéré une autre voie. Plutôt que de forcer une solution pixel par pixel, la segmentation devait commencer par regrouper des régions cohérentes à analyser en contexte.

1.3 Défis de la segmentation glace – gadoue – eau

Les images CAIMAN ne contiennent que rarement des bords nets. L'eau, le ciel, la terre et la glace se confondent souvent, tandis que les reflets peuvent aplatir les textures au point que les experts ne s'accordent plus sur les limites entre gadoue et eau (**Figure 1.2** ; voir [27]). La gadoue se comporte comme une phase transitoire plutôt qu'un objet stable, et les données d'entraînement arctiques restent rares. Ces facteurs rendent l'apprentissage et l'évaluation des modèles fragiles : un masque net peut simplement refléter un excès de confiance.

Figure 1.2 *Exemples d'images CAIMAN montrant la gadoue, les reflets et des zones à faible contraste.*

La rareté des données d'entraînement accentue cette ambiguïté. La plupart des jeux de données en segmentation sémantique sont construits autour d'images du quotidien — animaux, humains, routes, bâtiments — où les frontières de classe sont intuitives et peuvent être annotées avec des outils de contour. Cela ne s'applique pas aux environnements arctiques. Même les experts ne s'accordent pas sur la frontière entre gadoue et eau. On travaille donc avec des seuils incertains, au risque d'introduire plus de bruit que de vérité terrain.

1.4 Objectifs du mémoire

Ce mémoire porte sur la **segmentation sémantique** d'images arctiques capturées par les caméras terrestres du réseau CAIMAN. L'objectif est d'assigner à chaque pixel l'une des trois classes de surface — **glace solide, gadoue ou eau libr**e. Ces catégories ne se comportent pas comme des objets rigides : leurs formes changent d'une image à l'autre, les bords se brouillent à cause des reflets, et la gadoue n'existe que comme un mélange transitoire d'eau et de glace.

Ce travail propose donc un pipeline de segmentation conçu explicitement pour les surfaces matérielles ambiguës. En combinant des propositions régionales, une fusion basée sur les graphes et une classification par région, le système vise à produire des cartes pixelisées cohérentes et fiables même lorsque les indices visuels sont faibles ou trompeurs. Ces cartes sont destinées à la surveillance environnementale, aux études d'impact côtier et aux efforts d'adaptation au changement climatique au Nunavik.

2 Contexte et hypothèses initiales

2.1 Segmentation dans des scènes imprévisibles et ambiguës

La plupart des réseaux de segmentation supposent que les indices saillants — tels que la couleur, la texture, le contraste et les contours d'objets — restent stables dans un jeu de données. Cette hypothèse échoue dans les images CAIMAN, où les reflets aplatissent les textures et où la gadoue dérive d'une image à l'autre. Pour comprendre comment les travaux précédents traitent cette instabilité, la littérature a été divisée en deux catégories : (i) **les études centrées sur la glace** dans des scènes dominées par la neige ou la glace, et (ii) **les CNN généralistes** qui servent d'architectures de base en vision par ordinateur.

2.1.1 Travaux de segmentation centrés sur la glace

Les premières références proviennent d'articles traitant de la glace fluviale, de la glace lacustre ou de la glace de mer en Antarctique. La plupart s'appuyaient sur des variantes **de Mask R-CNN ou de DeepLab**, entraînées sur de petits jeux de données soigneusement sélectionnés, capturés à partir de drones ou de caméras montées sur navire.

Leur principale limitation est le décalage de domaine, caractérisé par des géométries de vue différentes, des résolutions variées, et nettement moins d'artefacts de reflet que ceux observés avec les capteurs côtiers de CAIMAN. Le **tableau 2.1** résume les quatre études ayant eu le plus d'influence sur ce projet.

Article	Modèle	Type d'entrée	Classes	Limitation principale
IceMaskNet [7]	Mask R-CNN	Photos aériennes	6 types de glace fluviale	Résolution + plateforme mobile
Lake-Ice Monitoring [8]	DeepLab v3+ / Deep-U-Lab	Webcam	Glace lacustre / eau / fond	Qualité d'image incohérente
Sea-Ice SegNet [10]	PSPNet-101 / SegNet	Vidéo de brise-glace	Glace / océan / navire / ciel	Jeu de données minuscule, domaine restreint
Ice-DeepLab [12]	DeepLab + attention	Scènes antarctiques	Glace / océan / ciel	Transfert incertain vers CAIMAN

Tableau 2.1 : *Littérature sur la segmentation centrée sur la glace. Articles clés classés par type de modèle, source de données et principale limitation par rapport aux images CAIMAN.*

Ces études montrent que des masques sémantiques peuvent être générés même lorsque les classes sont visuellement ambiguës. Cependant, chaque système repose sur des hypothèses propres au domaine — éclairage stable, perspective aérienne ou liste de classes limitée — qui ne s'appliquent pas aux images CAIMAN.

2.1.2 Modèles de segmentation sémantique généralistes

En parallèle, un ensemble d'architectures prêtes à l'emploi a été évalué sur des images brutes de CAIMAN. La sélection comprenait U-Net [2], PSPNet [5], DeepLab v3+ [4] et YOLOv8-seg [14] (Tableau 2.2). Aucun entraînement personnalisé n'a été réalisé à ce stade ; l'objectif était de mesurer la capacité des réseaux "out-of-the-box" à gérer les reflets, le faible contraste, et la transition entre gadoue et eau.

Modèle	Conception de base	Force typique
U-Net [2]	Encodeur-décodeur avec connexions de saut	Imagerie biomédicale, petits jeux de données
PSPNet [5]	Pooling pyramidale pour contexte multi-échelle	Capture de la disposition globale des scènes
DeepLab v3+ [4]	Convolutions dilatées + tête de raffinement	Bon modèle de base en segmentation générale
YOLOv8-seg [14]	Détection et segmentation unifiées	Inférence rapide en temps réel

***Tableau 2.2** : Modèles généralistes de segmentation sémantique. Principales caractéristiques de conception et forces typiques des quatre architectures testées sur des images CAIMAN.*

Des tests rapides ont révélé un schéma connu : les classes dominantes (terre, eau libre) étaient segmentées correctement, tandis que les fines couches de gadoue n'étaient pas détectées et que les reflets entraînaient de graves erreurs de frontière. Ces observations ont conduit à deux choix de conception adoptés plus tard dans le mémoire : premièrement, les étapes de

prétraitement doivent normaliser les reflets ; deuxièmement, tout système final doit raisonner sur des régions, et non sur des pixels isolés.

2.2 Hypothèses initiales et orientation préliminaire

Au début du projet, le plan initial consistait à créer un jeu de données bien équilibré et à ajuster un réseau standard comme DeepLab v3+. L'idée était simple : avec suffisamment de données bien sélectionnées, un seul modèle devrait pouvoir apprendre la tâche. Les efforts initiaux se sont donc concentrés sur la sélection d'images représentatives, la rédaction de consignes d'annotation, et la création d'un sous-ensemble d'entraînement petit mais précis.

Les premières expériences ont validé l'architecture mais révélé des nuances essentielles. Les bords aplatis par les reflets, la gadoue disparaissant, et les sauts de label entre images persistaient même après un réglage minutieux. Ces échecs ont suggéré que les CNN pixel à pixel étaient insuffisants ; la segmentation devait donc commencer à un niveau intermédiaire — en regroupant les pixels en régions cohérentes avant toute attribution de classe.

Ce changement de paradigme a posé les bases du pipeline centré sur les régions décrit dans les chapitres suivants.

3. Création et annotation du jeu de données

Le pipeline de segmentation CAIMAN repose sur un jeu de données robuste et représentatif d'images de rivage. Des images brutes (2048 × 1536 px, format 4:3) sont capturées toutes les 15 minutes sur sept sites du Nunavik. Étant donné que les conditions de surface (glace, gadoue, eau) évoluent sur plusieurs jours plutôt qu'en quelques minutes, nous échantillonnons les images à un rythme quotidien pour capturer des transitions significatives.

Version	Nombre d'images	Prétraitement et augmentations	Problèmes identifiés
v1 (basée sur PixelAnnotationTool)	500 images	Auto-orientation, égalisation d'histogramme, filtrage des masques vides	Dimensions trop grandes pour la plupart des modèles (coût mémoire élevé) ; biais fort vers la terre et l'eau, très peu de gadoue
v2 (assistée par Roboflow)	23 330 images	Auto-orientation, redimensionnement 640×640, découpage 2×2, égalisation d'histogramme, filtrage des annotations nulles	Meilleures performances, mais surapprentissage spatial : les modèles associaient les classes à des régions fixes
v3 (basée sur FastSAM)	23 330 images	Auto-orientation, redimensionnement 512×512, découpage 2×2, égalisation d'histogramme, filtrage des annotations nulles	Structure finale du jeu de données. Le rapport d'aspect était mieux aligné sur les attentes du modèle, ce qui a réduit le biais spatial entre tuiles

***Tableau 3.1** : Résumé des versions du jeu de données : pipelines de prétraitement et effets observés sur le comportement des modèles.*

Chaque image sélectionnée est exportée en deux versions — plein cadre et recadrée — afin de s'adapter à différentes configurations et compositions de caméras. Pour construire notre jeu de données annoté, nous avons développé trois flux de travail successifs d'annotation, chacun équilibrant la vitesse, l'échelle et la précision. Les sections 3.1 à 3.3 décrivent ces versions, et la section 3.4 résume leur impact comparatif sur l'apprentissage des modèles.

3.1 Version 1 — Étiquetage manuel avec PixelAnnotationTool

Le jeu de données initial a été produit pendant un stage Mitacs Globalink de 12 semaines sous la supervision du Dr Saeid Humayouni. Nous avons sélectionné des images représentatives d'Ivujivik, Quaqtaq et Umiujaq, puis utilisé PixelAnnotationTool [19] pour dessiner manuellement

des masques correspondant à cinq classes : glace, gadoue, eau libre, terre et ciel. Les annotateurs ont suivi des consignes écrites afin de garantir une cohérence dans le placement des frontières, en révisant chaque masque image par image. Au total, 500 images ont été étiquetées. Ces masques ont servi d'entrées pour les premières expériences d'ensemble avec U-Net, PSPNet et Mask R-CNN.

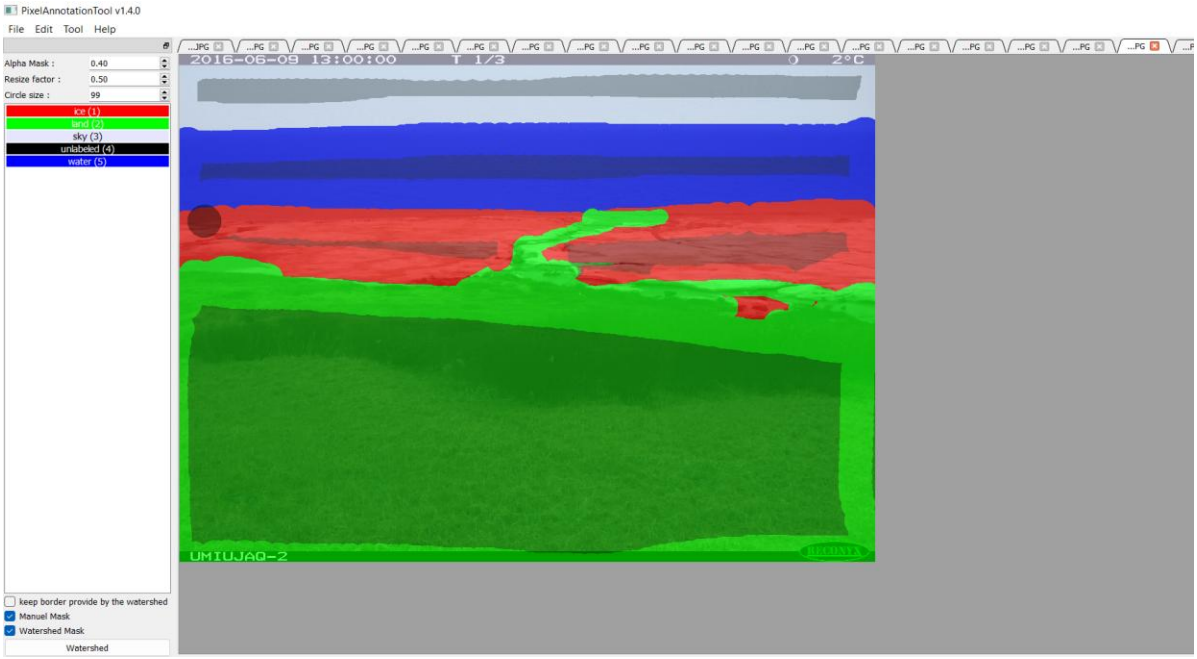

***Figure 3.1** : Exemple d'annotation de la version 1 avec PixelAnnotationTool, comportant cinq classes de surface étiquetées.*

Bien que chronophage, ce processus manuel a révélé la principale difficulté : la gadoue manque souvent de contours discernables, ce qui entraîne une forte variabilité entre annotateurs et expose les limites des modèles pixel à pixel. La version 1 a fourni une base de référence diagnostique pour le comportement des modèles, mais était trop petite et approximative pour permettre un apprentissage systématique.

3.2 Version 2 — Annotation assistée par Roboflow et augmentations

Début 2024, nous avons adopté l'outil basé sur les contours de Roboflow pour accélérer l'annotation. Nous nous sommes concentrés sur une fenêtre de 15 jours correspondant au début du gel — en sélectionnant environ 4 000 images capturant la formation rapide de glace et les transitions de gadoue.

Roboflow permettait de tracer des polygones avec des suggestions automatiques de contours, ce qui nous a permis d'annoter par lot des journées entières sous des conditions d'éclairage et de météo variables. Les annotateurs ont révisé et corrigé ces suggestions pour garantir la cohérence entre les images.

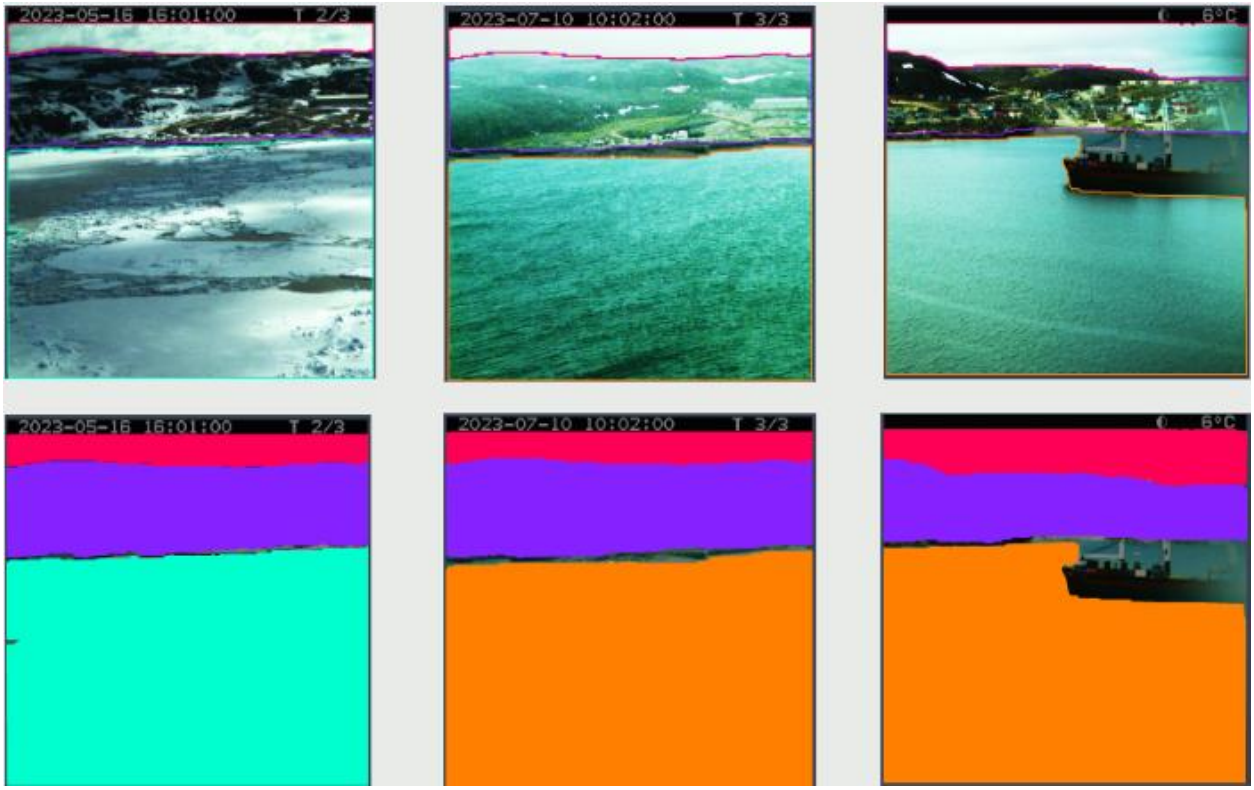

***Figure 3.2** : Annotations polygonales générées par Roboflow pendant la version 2 du jeu de données.*

Ces masques ont alimenté les premiers entraînements à grande échelle avec DeepLab v3+ [4]. Cependant, la focalisation sur une courte période de gel a introduit un biais temporel — les phases de gel/dégel stables (glace pure ou eau) étaient sous-représentées — et les contours automatiques coupaient parfois à travers les zones de gadoue à faible contraste. La version 2 a permis d'augmenter la taille du jeu de données et d'initier l'entraînement du modèle, mais souffrait de biais de position et d'erreurs de contour dans les régions ambiguës.

3.3 Version 3 — Pipeline d'annotation basé sur FastSAM

Pour combiner échelle et précision, nous avons développé un flux de travail personnalisé utilisant FastSAM [30]. FastSAM génère des propositions régionales en zero-shot via un décodeur de masque basé sur un transformeur. Pour chaque image sélectionnée, nous avons :

- Exécuté FastSAM en mode automatique pour produire 100 à 150 masques candidats
- Filtré et fusionné les propositions pour éliminer le bruit
- Réétiqueté manuellement les régions d'intérêt — en particulier les interfaces gadoue/eau
- Exporté les masques finaux au format COCO JSON ou PNG

Ce pipeline a traité environ 23 000 images hors ligne, puis extrait un "jeu d'or" de 8 000 images via des filtres de qualité. La sur-segmentation de FastSAM a permis de capturer des textures

subtiles, mais a nécessité des étapes supplémentaires pour corriger les régions fragmentées et fusionner les artefacts mineurs.

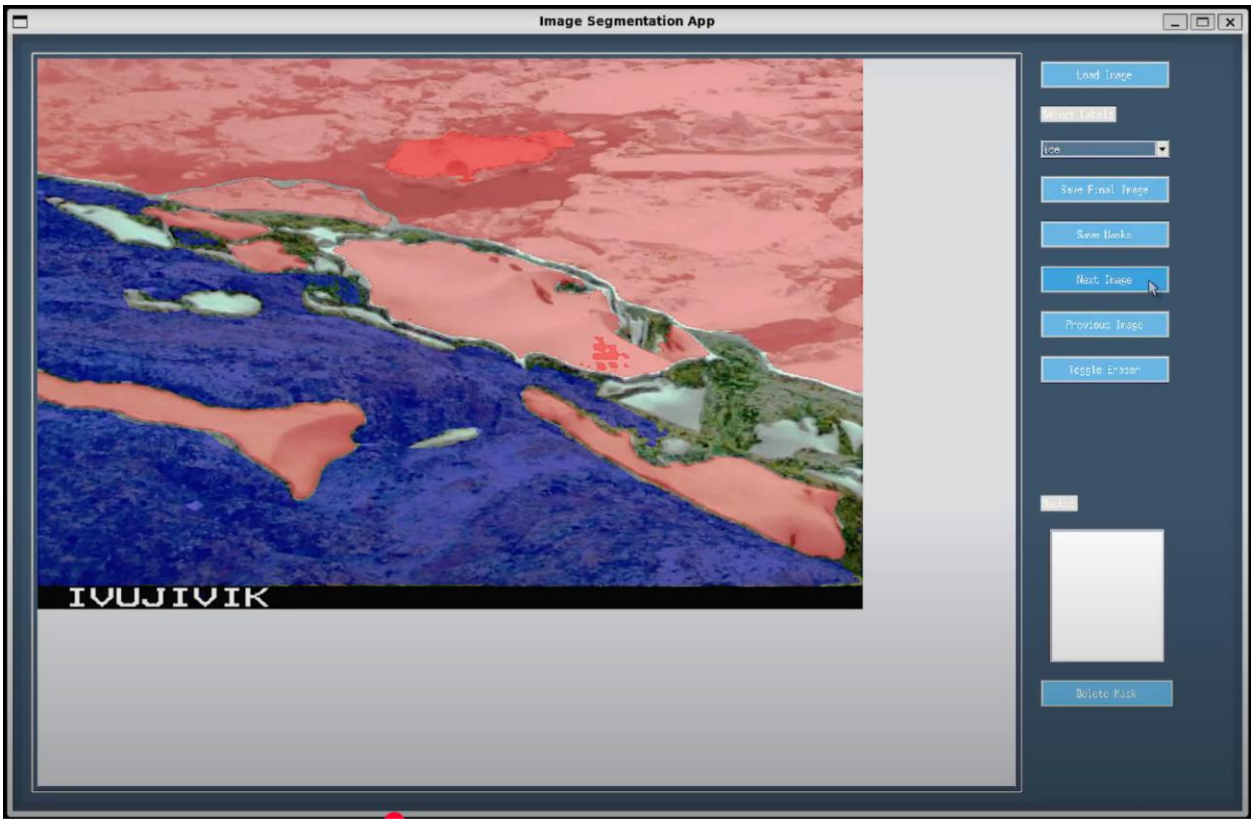

***Figure 3.3** : Propositions régionales générées par FastSAM et masques corrigés du flux de travail de la version 3.*

La version 3 combinait des propositions régionales automatisées avec une curation manuelle, produisant des masques de haute qualité à grande échelle tout en maîtrisant la sur-segmentation.

4. Premiers essais de modélisation

Les premières expériences ont porté sur des réseaux de segmentation bien connus. Nous avons commencé par des CNN pixel à pixel, testé une approche par ensemble, puis exploré l'inférence par tuiles et les hybrides centrés sur les régions. Chaque étape a mis en évidence une limitation — ambiguïté aux bords de gadoue, frontières aplaties par les reflets, ou biais spatial — et suggéré des ajustements successifs.

La section 4.1 énumère les modèles principaux évalués ; les sections suivantes expliquent comment ils ont été combinés et pourquoi un pipeline basé sur les régions a finalement remplacé la plupart.

4.1 Modèles de segmentation utilisés dans ce travail

L'étude a évalué cinq architectures principales, chacune choisie pour une capacité spécifique : préservation des détails, contexte global, isolation d'objets, sémantique multi-échelle ou génération de propositions régionales. Les variantes de DeepLab sont résumées dans le tableau 4.1, et celles de SAM dans le tableau 4.2.

4.1.1 U-Net

U-Net [2], conçu à l'origine pour l'imagerie médicale, a servi de point de départ grâce à sa simplicité et à ses connexions de saut qui préservent certains détails fins. Dans nos premiers tests, il capturait relativement bien les bandes étroites de gadoue ou de glace mince, et montrait de la stabilité sur des régions de taille moyenne.

Cependant, son principe d'encodeur–décodeur — alternant compressions et reconstructions — a révélé deux limites majeures : perte de fidélité aux frontières et coûts de calcul élevés sur des images arctiques 2K. Ces contraintes ont réduit sa pertinence face à des structures larges ou ambiguës, où l'approche par régions modulaires s'est montrée plus adaptée. Pour les mêmes raisons, des variantes plus complexes comme U-KAN [33] ou U²-Net [34] n'ont pas été poursuivies.

4.1.2 PSPNet

Le réseau PSPNet (Pyramid Scene Parsing Network) traite les images à plusieurs échelles spatiales simultanément, offrant une bonne cohérence globale. Il excellait dans la reconnaissance des larges zones de terre, glace et eau, réduisant la fragmentation observée avec U-Net.

Cette cohérence avait un coût : les transitions étroites ou abruptes étaient souvent floues, et les motifs fins de gadoue devenaient indistincts. En combinaison avec U-Net, PSPNet stabilisait les prédictions, mais seul, il ne résistait pas aux pertes de bord dues aux reflets.

4.1.3 Mask R-CNN

Mask R-CNN offre un pipeline en deux étapes — détection d'objets suivie d'un raffinement du masque — bien adapté aux instances discrètes et bien délimitées. Il s'est révélé efficace pour isoler les rochers et les blocs de glace détachés, mais a eu du mal avec les classes de surface amorphes.

La gadoue et les fines couches de fonte ne formaient que rarement des contours nets suffisants pour déclencher une détection, poussant le réseau à les ignorer ou à les fusionner avec les régions d'eau voisines. Ce modèle a mis en lumière le décalage entre les paradigmes de segmentation par instance et les problèmes de surface matérielle.

4.1.4 DeepLab v3+

DeepLab v3+ combine des convolutions à trous avec une structure encodeur-décodeur, permettant de capturer à la fois la texture locale et la sémantique globale. Un backbone ResNet-50 offrait une base rapide compatible CPU, tandis que le backbone ResNet-101 (voir tableau 4.1) était préféré pour les configurations GPU.

Variante	Backbone	Rôle dans le pipeline	Remarques
DeepLabv3+ (ResNet-50)	ResNet-50 [31]	Base rapide pour tester les entrées pré-segmentation	Léger ; utilisé lors de la conception compatible CPU
DeepLabv3+ (ResNet-101)	ResNet-101 [31]	Modèle de classification final dans le pipeline GPU	Plus de capacité ; meilleure détection des variations fines
DeepLabv3 (plein cadre)	ResNet-50	Expériences initiales en segmentation pixel à pixel	Remplacé plus tard par des approches régionales à cause du biais spatial

***Tableau 4.1** : Variantes de DeepLabv3+ utilisées dans ce travail.*

Ajusté sur les annotations de la version 2, DeepLab produisait des masques plus nets et un rappel par classe supérieur à l'ensemble. Néanmoins, le réseau présentait un biais de position hérité des données d'entraînement et avait tendance à sur-prédire les classes dominantes, occultant les gadoues subtiles ou les bords aplanis par les reflets.

4.1.5 Segment Anything Model (SAM)

SAM est un moteur basé sur un transformeur de vision qui génère des propositions régionales de haute fidélité sans supervision de classe. En mode automatique, il divise chaque image en 100 à 150 régions cohérentes dont les bords suivent les variations de couleur et de texture, même en présence de forts reflets.

Variante	Architecture de base	Rôle dans le pipeline	Remarques
SAM (ViT-H) [6]	Vision Transformer (Large)	Utilisé pour le pipeline final sur GPU	Masques de haute qualité avec excellente fidélité de contour
FastSAM [30]	Variante légère	Utilisé pour les expériences initiales et l'annotation de données	Plus rapide mais plus bruyant ; utile pour les prototypes

***Tableau 4.2** : Variantes de Segment Anything Model explorées lors des expérimentations.*

Le backbone ViT-H offrait les propositions les plus précises pour le pipeline GPU, tandis que FastSAM [30], plus léger, a servi à la prototypation rapide et à l'annotation hors ligne. En dissociant la découverte de régions de l'étiquetage sémantique, SAM a permis à DeepLab de fonctionner sur des unités discrètes et contextuelles, améliorant significativement l'intégrité des bords dans les transitions gadoue–eau.

4.2 Tentative d'apprentissage par ensemble

Le premier modèle composite combinait U-Net, PSPNet et Mask R-CNN pour équilibrer finesse des bords, contexte de scène et netteté des objets. Chaque réseau générait un masque complet ; une étiquette pixel était retenue si au moins deux modèles étaient d'accord, avec un seuil d'IoU pour résoudre les conflits, comme dans les études antérieures [8–9–12–14–21].

- U-Net – capture les filaments fins de gadoue et les fissures étroites.
- PSPNet – stabilise les grandes régions de terre, de glace et d'eau via le pooling pyramidal.
- Mask R-CNN – délimite les objets discrets tels que les rochers et les blocs de glace détachés.

L'ensemble a été entraîné et testé sur ≈ 500 images annotées avec PixelAnnotationTool (section 3.1). Comme les rivages et horizons occupaient des positions fixes, chaque réseau a appris des raccourcis spatiaux plutôt que des indices texturaux. Parmi les faiblesses majeures observées (voir figure 4.1) :

- Les masques ressemblaient à des moyennes floues, non à des cartes cohérentes.
- Deux modèles s'accordaient rarement sur les frontières de la gadoue.
- Les artefacts s'accumulaient dans les zones très réfléchissantes.
- L'IoU variait fortement d'un essai à l'autre, signalant une instabilité.

Le principal problème venait du vote à poids égal : dans les régions les plus difficiles, les trois réseaux divergeaient le plus, donc la majorité amplifiait plutôt qu'elle ne corrigeait leurs erreurs.

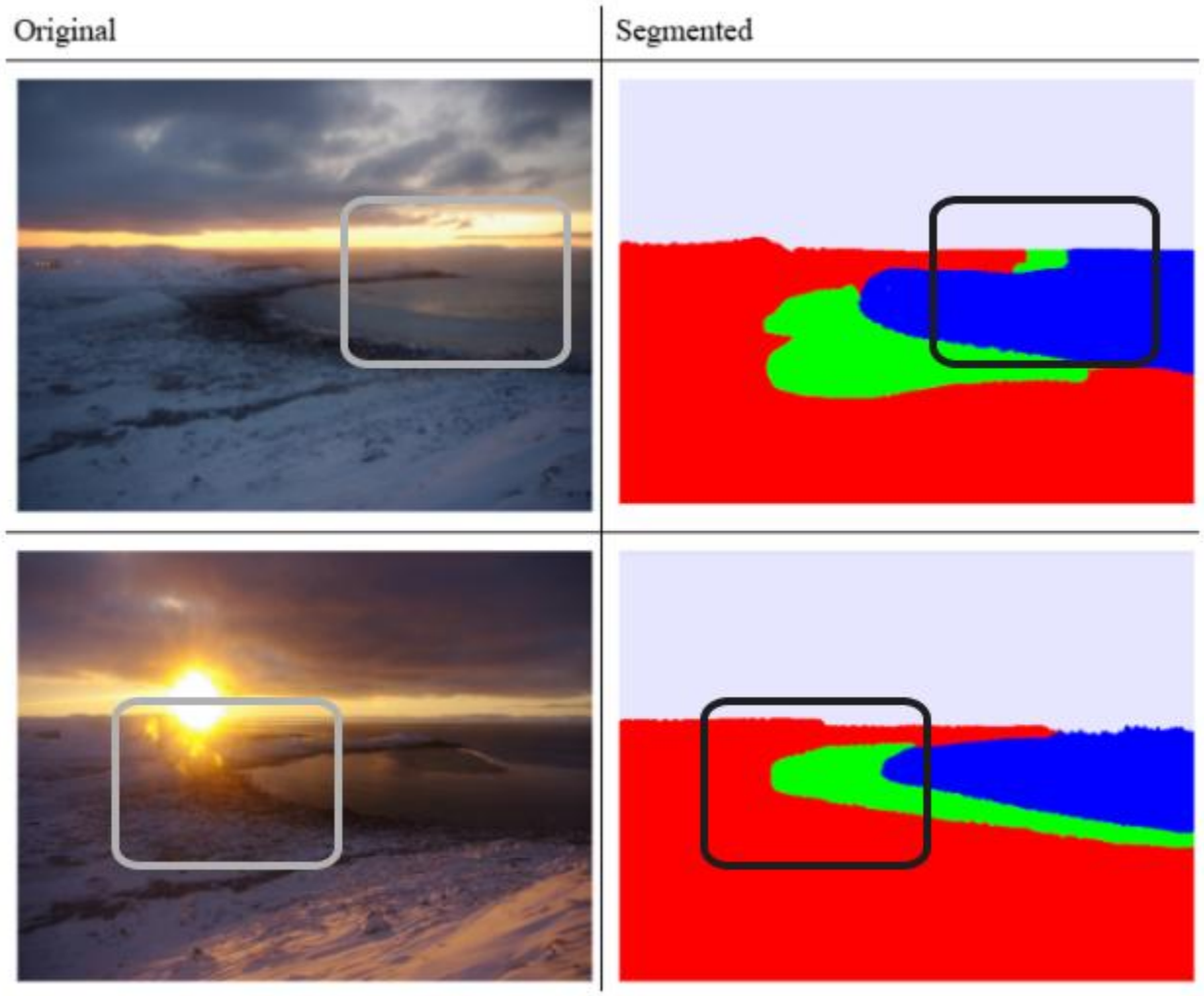

***Figure 4.1** : Sorties du modèle par ensemble montrant une faible cohérence inter-classes et des artefacts d'approximation.*

Comme outil diagnostique, l'ensemble a révélé l'ambiguïté des frontières et les conflits entre modèles, mais son instabilité et son coût élevé l'ont rendu inutilisable en pratique. Ces résultats ont réorienté l'étude vers des expériences avec un seul backbone et, finalement, vers la stratégie centrée sur les régions présentée au chapitre 5.

4.3 Modèles de segmentation sémantique généralistes

Après l'échec de l'ensemble, nous nous sommes tournés vers un modèle unique plus fiable : **DeepLabv3+** [4]. Cette version a été entraînée sur environ 4 000 images révisées — nettement plus propres que les ~500 masques bruités issus de **PixelAnnotationTool** (section 3.1). Même

sans réglage intensif, elle a rapidement surpassé le trio précédent [23], produisant des bords plus nets, un meilleur rappel par classe et une cohérence plus régulière entre les images.

Cela dit, l'expérience a montré plusieurs limites :

- Des fragments de gadoue ou de glace étaient manquants s'ils ne correspondaient pas aux schémas appris.
- Les grandes régions comme la terre et l'eau étaient remplies de manière excessive, avec une forte confiance.
- Le modèle s'appuyait trop sur la position — « terre » apparaissait en bas, même si ce n'était pas le cas.
- Les transitions de texture étaient ignorées, surtout en cas de reflets intenses.

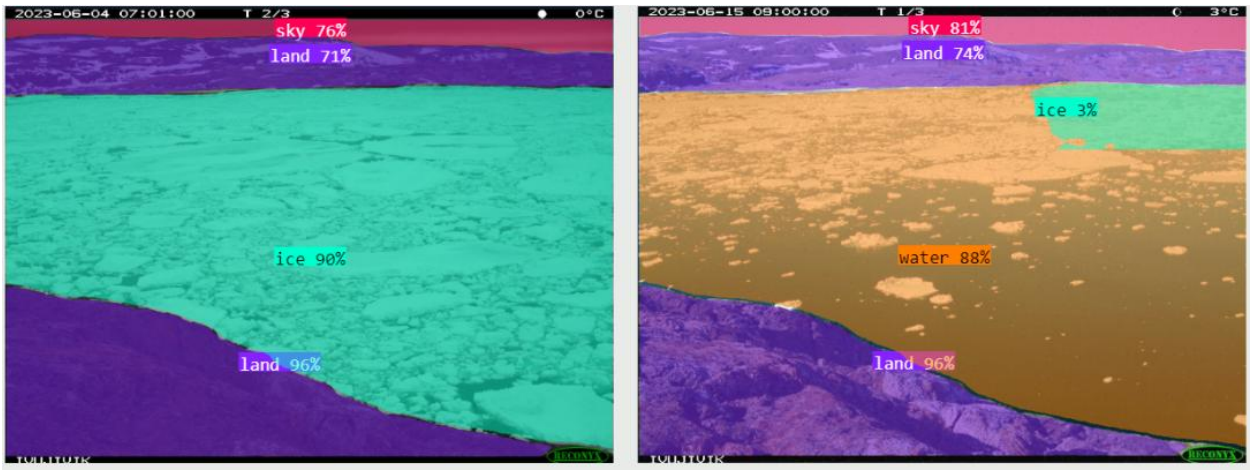

***Figure 4.2** : Exemples de prédictions de DeepLab v3+ montrant un biais de classe dominant et l'échec à séparer les textures ambiguës comme la gadoue ou les fragments de glace.*

Les forces de DeepLab étaient claires, mais ses angles morts l'étaient aussi. Le modèle apprenait à deviner en fonction de la position habituelle des objets, et non de leur apparence réelle. Pour briser ce raccourci, nous avons divisé l'image en fenêtres plus petites, imposant un raisonnement local, et avons progressivement abandonné le traitement plein cadre.

4.4 Découpage d'image et raisonnement sur les caractéristiques locales

Suite aux limites observées avec les prédictions plein cadre de DeepLabv3+, l'étape suivante a consisté à passer à une inférence localisée. Cette approche visait à réduire le surapprentissage spatial et à améliorer la précision des frontières de classe, en particulier dans les régions riches en gadoue ou présentant des textures ambiguës.

Les images d'entrée ont été découpées en tuiles de 512×512 pixels. Chaque tuile a été traitée indépendamment lors de l'inférence et du post-traitement. Cette méthode a éliminé les hypothèses spatiales globales et forcé le modèle à s'appuyer sur les informations texturales et

structurelles à l'intérieur de chaque fenêtre. Les prédictions des tuiles se chevauchant ont ensuite été réassemblées pour reconstruire des masques de segmentation complets.

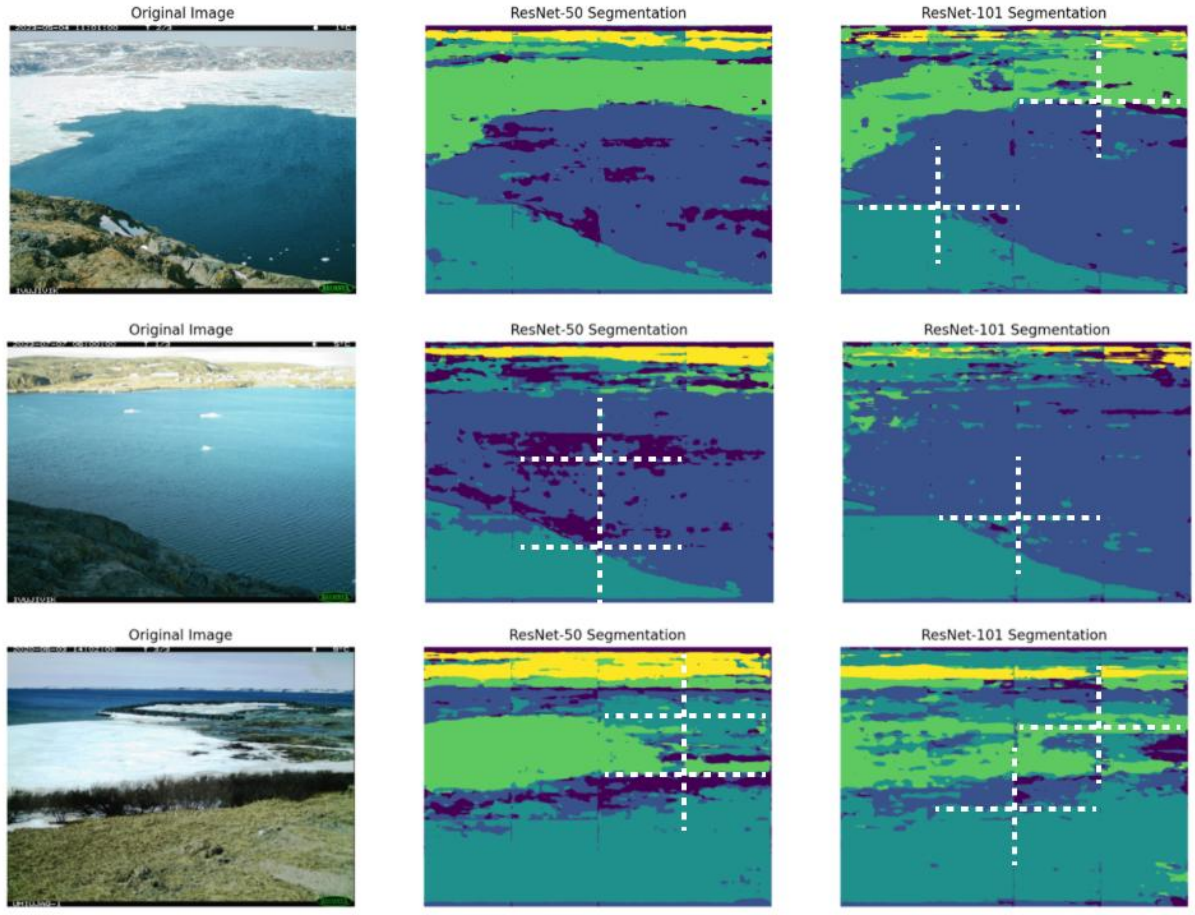

***Figure 4.3** : Prédictions DeepLabv3+ sur fenêtres d'entrée découpées, montrant une amélioration des positions mais des incohérences visibles aux bords des tuiles.*

Cette approche a apporté plusieurs améliorations :

- Les prédictions étaient plus sensibles aux textures locales — fines bandes de gadoue, fonte partielle, transitions eau–reflet.
- Le modèle ne supposait plus de positions fixes des classes (ex. « terre en bas »), car le contexte était restreint à chaque tuile.
- Les petites tuiles ont corrigé le déséquilibre de classe — des éléments auparavant sous-représentés, comme les plaques de gadoue isolées ou les bords obliques de terre, devenaient plus visibles.

Cependant, le découpage a aussi introduit de nouveaux défis. Lorsque des frontières de classe traversaient plusieurs tuiles, la continuité était rompue, ce qui entraînait un léger décalage lors de

la reconstruction. Les méthodes de réassemblage ont été raffinées pour minimiser ces discontinuités, mais certains artefacts de bordure subsistaient.

Malgré ces limites, la stratégie de découpage a marqué une transition nécessaire vers une segmentation spatialement adaptative. Elle a révélé le besoin d'une inférence sensible à la structure — capable de considérer les amas de pixels locaux et la relation entre les régions voisines.

4.5 Exploration panoptique et intégration de SAM

Face à l'instabilité des prédictions plein cadre, nous avons ensuite exploré des méthodes de segmentation qui raisonnent sur des régions discrètes plutôt que sur des champs de pixels homogènes. La segmentation panoptique offrait un compromis entre la segmentation sémantique et par instance — capturant à la fois « ce qu'est » une zone et « combien de parties » elle comprend [26–32]. Ce changement a permis au pipeline d'identifier plus explicitement la fragmentation des surfaces, comme la gadoue éparse ou les débris de glace.

Parallèlement, les modèles fondation ont commencé à gagner du terrain. Le Segment Anything Model (SAM) de Meta est apparu comme un bon candidat grâce à ses capacités de sur-segmentation et sa généralisation inter-domaines. Au lieu de prédire des étiquettes de classe, SAM génère un ensemble dense de masques régionaux à partir de prompts simples (points, boîtes, clics). Comme illustré à la figure 4.4, ce mécanisme produit des contours d'objets fins, même dans des textures visuellement ambiguës.

L'approche était simple :

- Exécuter SAM pour extraire les masques de type instance pour chaque image.
- Transmettre ces masques à DeepLab pour classification sémantique.
- Assembler les résultats dans une carte finale de prédiction avec estimation de confiance.

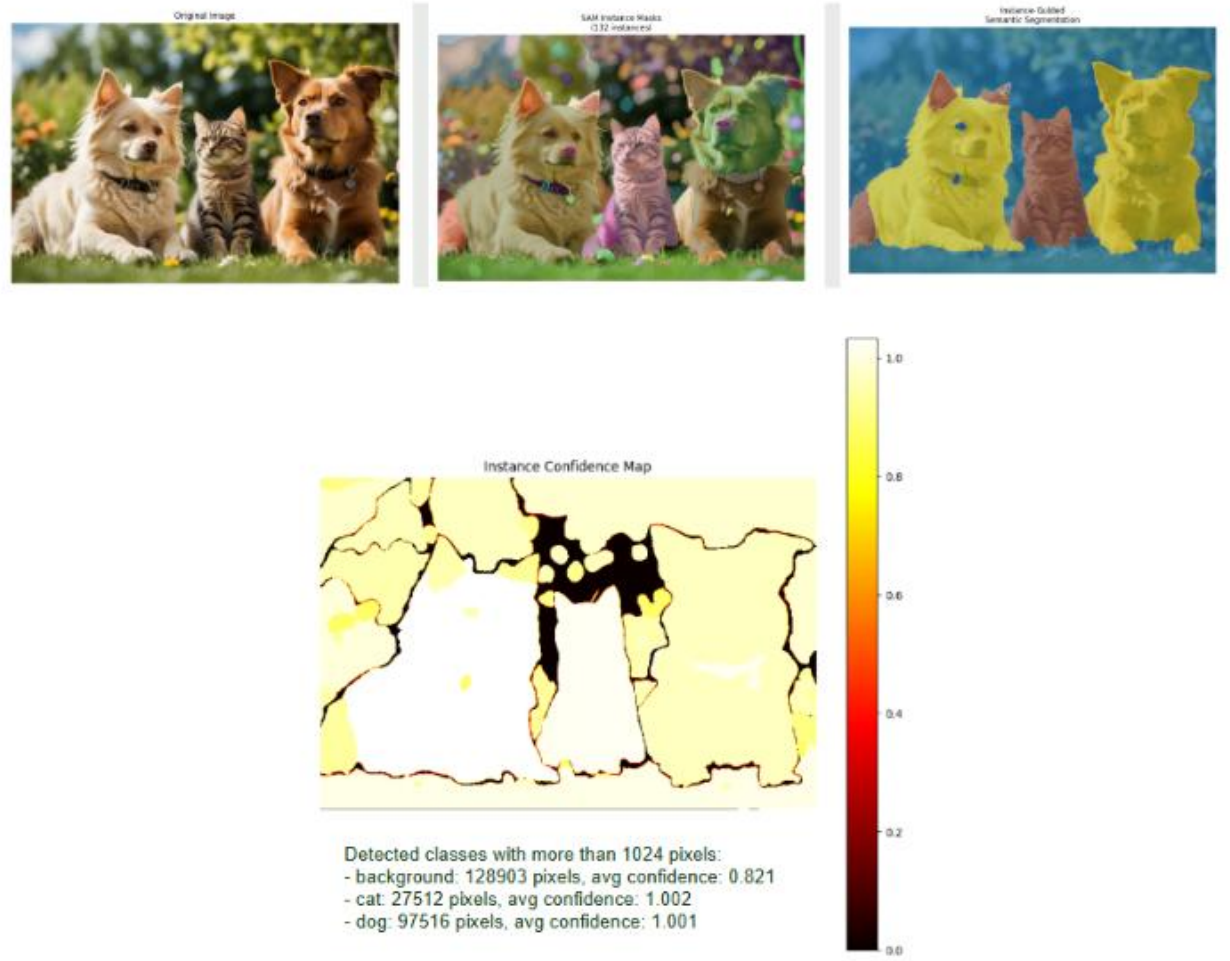

***Figure 4.4** : Prédictions hybrides SAM + DeepLab montrant des contours plus nets et des classes plus confiantes, avec des régions mieux alignées aux transitions physiques.*

Ce design hybride a produit des gains immédiats :

- Contours d'objets plus propres et confiance par classe améliorée
- Meilleure capture des textures ambiguës (comme la gadoue) que les modèles DeepLab seuls

Cette phase a reformulé le problème : la segmentation ne consistait plus simplement à classer des pixels, mais à identifier des motifs signifiants dans les données. Cela impliquait de décomposer l'image en unités cohérentes et de raisonner sur ces unités.

L'étape suivante consistait donc à formaliser cette idée — considérer les régions comme éléments primaires et introduire des pipelines structurés qui reflètent cette hiérarchie.

4.6 Enseignements diagnostiques issus des phases expérimentales

Chaque modèle testé a laissé quelque chose d'important de côté. Certains étaient rapides mais aveugles aux détails. D'autres étaient lents et trop confiants. La plupart excellaient dans certains cas mais échouaient dans d'autres, et aucun ne pouvait être utilisé seul de façon fiable.

Méthode / Phase	**Leçon retenue**	**Besoin identifié**
Ensemble U-Net, PSPNet, Mask R-CNN	Combiner des modèles faibles ne résout pas les problèmes fondamentaux de données ; les sorties manquent de cohérence.	Un modèle plus robuste et résilient face au manque de données
DeepLabv3 / v3+ (image complète)	La segmentation pixel à pixel est rapide mais échoue à généraliser ou à détecter les petits éléments.	Généralisation au-delà du biais spatial
Fenêtrage glissant avec DeepLab	La prédiction locale réduit le biais, mais les bords de tuile causent des incohérences, et l'approche manque de structure.	Focalisation locale sans perdre le contexte global
SAM + DeepLab hybride	Partir de régions cohérentes améliore considérablement la confiance aux bords et la rapidité.	Regrouper les régions avant classification

***Tableau 4.3** : Observations clés et enseignements tirés des différentes approches expérimentales.*

Pendant toute cette phase, j'ai exécuté les expériences localement — sur une RTX 4070 Super avec 12 Go de VRAM. Cela s'est avéré idéal pour pousser ce GPU à ses limites. Google Colab était trop restrictif, surtout pour les tests itératifs. Avoir tout en local permettait de modifier les paramètres à la volée et d'ajuster chaque composant du système sans subir de coupures de session ni de limitations de téléchargement.

5. Le pipeline hybride : choix de conception

5.1 Objectifs et contraintes

À ce stade final du projet, il était clair qu'aucun modèle de segmentation unique ne pouvait gérer de manière fiable l'ambiguïté propre au jeu de données CAIMAN. Des conditions comme les reflets, la fine couche de glace semblable à la terre, et la gadoue se mêlant à l'eau libre provoquaient systématiquement des erreurs de classification, même chez les meilleurs modèles. Un système plus adaptatif et interprétable était nécessaire.

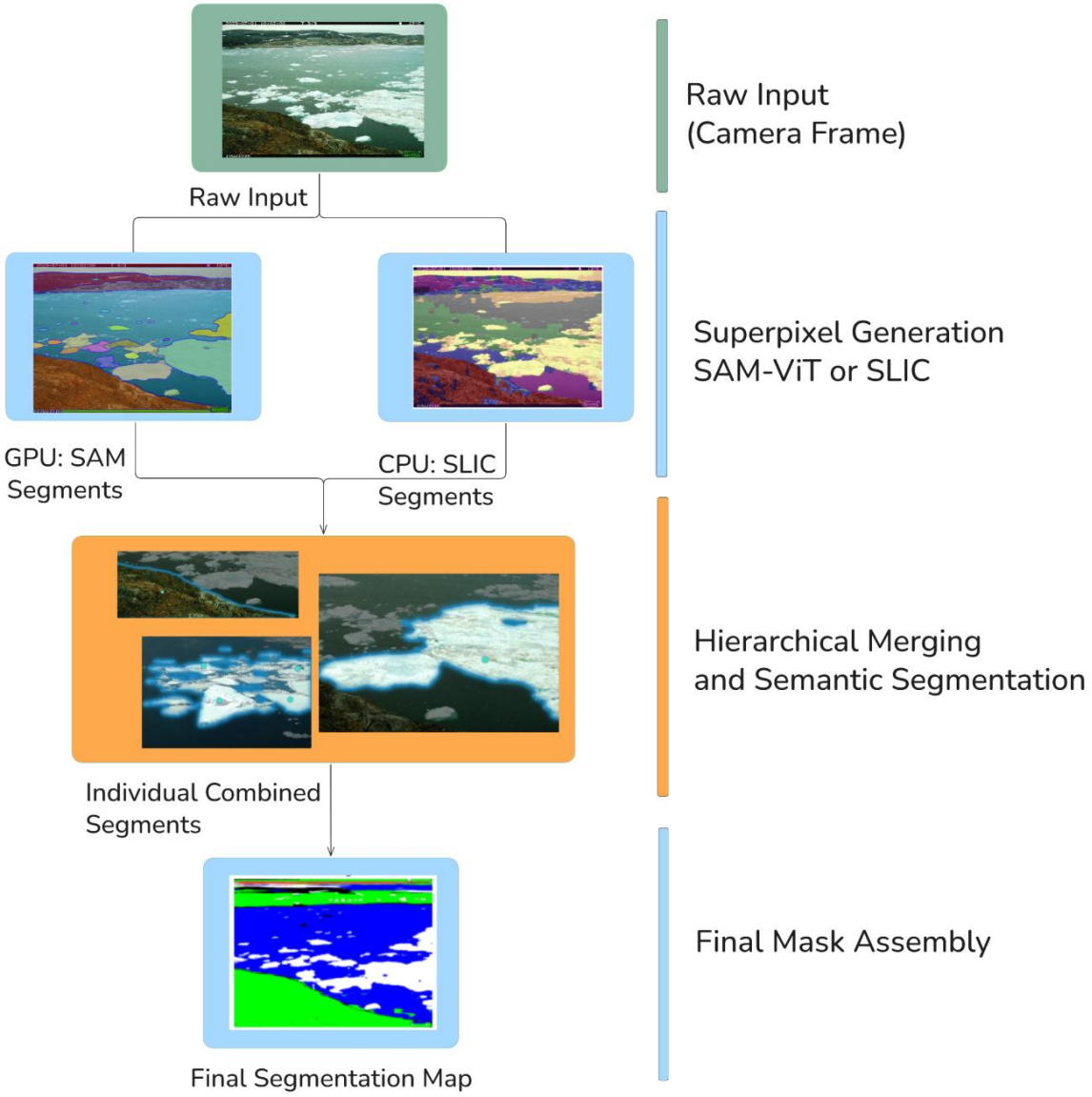

***Figure 5.1** : Progression visuelle d'une image dans le pipeline de segmentation. Le système bifurque à l'étape de génération de superpixels, en utilisant soit SAM-ViT (GPU), soit SLIC (CPU).*

Nous avons donc conçu deux variantes du pipeline final :

- Une version optimisée GPU pour les expérimentations haute résolution
- Une version compatible CPU pour une inférence rapide sur des systèmes peu puissants

Toutes deux suivent la même logique : diviser l'image en régions, raisonner localement, étiqueter globalement. Les contraintes suivantes ont guidé la conception :

- **Ambiguïté structurelle** — les transitions de classe étaient progressives ou mal définies, notamment dans les zones de gadoue
- **Annotations bruitées** — le jeu de données contenait des étiquettes imprécises, surtout dans les zones de transition
- **Rapidité d'exécution** — le système devait permettre des itérations rapides
- **Variabilité matérielle** — bien que développé sur un GPU RTX 4070 Super de 12 Go, il devait pouvoir fonctionner sans GPU

Ce chapitre décrit l'architecture du pipeline et explique comment chaque composant a été choisi ou conçu pour répondre à ces contraintes.

5.2 Configuration de l'implémentation

L'ensemble des expériences a été réalisé localement sur une station de travail dédiée. Les spécifications du système sont les suivantes :

- **Processeur** : Intel® Core™ i7-14700KF, fréquence de base 3,4 GHz
- **Mémoire** : 32 Go RAM
- **GPU** : NVIDIA® GeForce RTX™ 4070 Super Ti, 16 Go VRAM

Cette configuration offrait des ressources suffisantes pour exécuter aussi bien la variante GPU (SAM) que la variante CPU (SLIC) du pipeline, sans recourir au calcul distribué ni à des ressources infonuagiques.

L'environnement logiciel reposait sur PyTorch comme cadre central d'apprentissage profond, complété par les bibliothèques scientifiques standards. Les dépendances clés incluaient :

- **Torchvision** pour l'architecture DeepLabv3-ResNet101 et la pré-traitement
- **Segment Anything (SAM, ViT-H)** pour les propositions de sur-segmentation
- **scikit-image (SLIC)** pour la génération de superpixels sur CPU
- **OpenCV** et **Pillow** pour la gestion des images
- **NetworkX** pour la construction du graphe d'adjacence des régions

- **NumPy** et **Matplotlib** pour le calcul numérique et la visualisation

Cette conception préservait une flexibilité importante, permettant d'interchanger certains composants (par ex. SAM vs. SLIC pour les propositions) tout en maintenant une évaluation cohérente. Sauf indication contraire, la variante GPU optimisée avec SAM a été utilisée pour les résultats présentés au Chapitre 6. La version CPU basée sur SLIC a été réservée aux expériences comparatives et aux tests sur configurations plus limitées.

5.3 Architecture finale du pipeline

Le système de segmentation final est organisé en un pipeline modulaire en cinq étapes. Chaque étape transforme l'image en une représentation intermédiaire plus structurée, permettant un raffinement progressif :

- **Génération de superpixels** — produit des régions cohérentes dans l'image
- **Construction d'un graphe d'adjacence régionale (RAG)** — encode les connexions entre régions voisines
- **Fusion de régions** — simplifie et regroupe les zones sur-segmentées
- **Classification régionale** — attribue une étiquette à chaque région fusionnée
- **Assemblage des masques** — combine les prédictions pour générer une carte sémantique finale

Les deux versions partagent ce flux, mais diffèrent sur les deux premiers composants. La figure 5.1 illustre l'ensemble du processus. Malgré des différences d'implémentation, les variantes suivent la même séquence générale et peuvent être adaptées en modifiant des composants individuellement.

5.4 Génération de superpixels et prétraitement

Le pipeline commence par diviser l'image en superpixels — des regroupements de pixels visuellement similaires. Ceux-ci ne servent pas directement à la prédiction mais constituent les unités de base pour la construction du graphe et la classification. Chaque superpixel devient un nœud dans le graphe d'adjacence régionale.

Propriété	SAM-ViT (GPU)	SLIC (CPU)
Type d'algorithme	Segmentation par vision transformer	Regroupement couleur–position

Sortie	80–150 masques binaires	300–500 superpixels
Alignement des contours	Élevé (aligné texture–sémantique)	Faible (aucune sensibilité aux bords)
Sensibilité visuelle	Robuste aux reflets, transitions	Sensible uniquement à la couleur locale
Apprentissage requis	Non (poids préentraînés)	Non
Besoin matériel	GPU (12 Go ou plus)	CPU uniquement
Cas d'usage	Propositions dans zones ambiguës	Alternative légère
Post-traitement	Léger (filtrage de petits objets)	Lourd (regroupement, débruitage)

***Tableau 5.1** : Comparaison des méthodes SAM-ViT et SLIC pour la génération de superpixels, mettant en évidence les différences algorithmiques, la qualité des sorties et la pertinence pour le pipeline.*

5.4.1 SAM-ViT (pipeline GPU)

Sur GPU, nous utilisons SAM avec un backbone ViT pour extraire environ 80 à 150 masques binaires fins par image. Ces régions suivent naturellement les contours des objets et capturent des caractéristiques sémantiques telles que les transitions gadoue–eau.

- Basé sur un Vision Transformer préentraîné
- Produit des segments alignés sur les bords d'objets
- Nécessite 12 Go de VRAM
- Nécessite peu de filtrage avant la construction du graphe

5.4.2 SLIC (pipeline CPU)

La version CPU utilise SLIC [15], qui regroupe les pixels selon leur proximité spatiale et chromatique. Le résultat est d'environ 300 à 500 superpixels compacts par image, mais sans conscience sémantique.

- Rapide et indépendant du matériel
- Produit des régions propres mais non alignées sur les objets
- Demande un post-traitement important pour gérer le bruit et les zones incohérentes

5.5 Construction du graphe et fusion guidée par les caractéristiques

Une fois les superpixels générés, l'étape suivante consiste à comprendre leurs relations. Plutôt que de comparer des pixels bruts, nous raisonnons désormais sur des régions cohérentes via un **graphe d'adjacence régionale (RAG)**. Cette structure transforme l'image en un ensemble de nœuds (régions) et d'arêtes (adjacence).

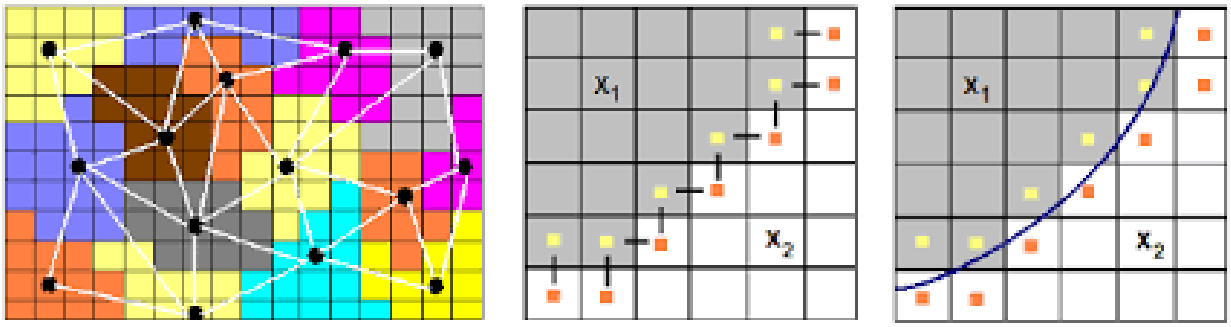

***Figure 5.4** : Visualisation d'un graphe d'adjacence régionale (RAG) à partir des superpixels. Chaque nœud représente une région, et les arêtes encodent l'adjacence spatiale utilisée pour les décisions de fusion.*

Chaque superpixel devient un nœud dans le graphe, contenant des descripteurs utiles comme :

- Couleur RGB moyenne
- Position du centroïde
- Empreinte du masque binaire
- Aire (nombre de pixels)

Des arêtes sont ajoutées entre les régions qui partagent une frontière. Cela est déterminé en vérifiant si leurs masques binaires se touchent après une légère dilatation morphologique. Si deux masques se chevauchent, les régions sont considérées comme voisines et sont reliées dans le graphe.

Le RAG obtenu est une structure compacte et interprétable. Au lieu de prédire directement des étiquettes à partir des données d'image, on opère sur ce graphe pour déterminer quelles régions doivent être fusionnées, en se basant sur leur similarité quantitative. Comme illustré à la figure

5.4, cette approche met en évidence la manière dont les relations spatiales et visuelles entre les zones peuvent être modélisées proprement.

Le RAG prépare ainsi l'étape suivante : appliquer une fusion simple, basée sur des règles, pour regrouper les régions visuellement similaires en segments plus larges et sémantiquement significatifs.

5.6 Ingénierie des caractéristiques pour la fusion

Une fois le RAG construit, l'objectif est d'identifier les régions adjacentes pouvant être regroupées sans risque. Plutôt que d'entraîner un modèle à apprendre ces relations, nous utilisons des caractéristiques transparentes et interprétables pour guider les décisions de fusion. Cela permet de conserver le contrôle et d'éviter le surapprentissage à partir de données d'entraînement bruitées ou ambiguës.

Pour chaque nœud du graphe, les attributs suivants sont stockés :

- **Masque binaire** – empreinte spatiale de la région
- **Centroïde** – position moyenne des pixels
- **Couleur RGB moyenne** – utilisée comme descripteur de surface
- **Taille** – nombre total de pixels

La fusion est effectuée uniquement entre nœuds adjacents (c'est-à-dire connectés par une arête dans le graphe). Chaque paire est évaluée à l'aide de deux comparaisons simples :

- **Distance de couleur** – distance euclidienne entre les valeurs RGB moyennes des régions
- **Distance des centroïdes** – distance euclidienne entre les centres spatiaux des régions

Cela garantit que seules les régions spatialement proches et visuellement cohérentes sont regroupées. Ce principe est essentiel pour préserver des frontières acceptables dans les zones avec des reflets ou des transitions gadoue–eau, où les classificateurs pixel-par-pixel échouent souvent.

En représentant toutes les régions — qu'elles proviennent de SAM, SLIC ou d'annotations manuelles — avec ces mêmes descripteurs numériques, le pipeline aligne des entrées structurellement différentes dans un espace vectoriel commun. Cette unification était cruciale pour faire fonctionner un système hybride. Bien que chaque source de segmentation repose sur des hypothèses différentes, le RAG permet de les traiter comme des unités comparables pour les étapes suivantes.

5.7 Classification avec DeepLab et post-traitement

À l'étape finale, les régions fusionnées sont transmises à un classificateur DeepLabv3+ affiné, basé sur une architecture ResNet-101. Ce modèle attribue une étiquette sémantique à chaque région, traitée de manière indépendante.

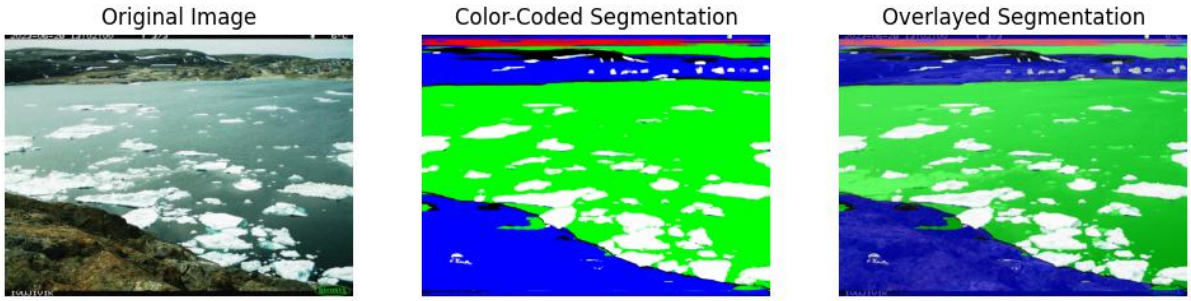

***Figure 5.5** : Résultats finaux du pipeline modulaire, avec superposition des prédictions par région.*

DeepLabv3+ a été choisi pour son équilibre entre précision spatiale et raffinement des frontières, en utilisant des convolutions dilatées pour gérer plusieurs échelles. Le modèle a été entraîné sur plus de 8 000 images CAIMAN annotées manuellement, issues de la troisième version du jeu de données (voir section 3.3).

Le classificateur segmente cinq classes :

- Terre
- Ciel
- Eau
- Glace
- Arrière-plan

Pour compenser le déséquilibre des classes et les étiquettes bruitées, la fonction de perte utilisée est une entropie croisée pondérée par classe, avec des poids ajustés. Cette configuration favorisait la généralisation, permettant à DeepLab d'apprendre des structures visuelles au-delà des annotations disponibles.

Avec le temps, le modèle a commencé à surpasser les annotations humaines :

- Les zones de gadoue et les bords ambigus étaient mieux segmentés que dans le jeu de données d'origine
- Les masques étaient plus lisses et évitaient les artefacts de scintillement
- Les régions auparavant incertaines recevaient des prédictions cohérentes et explicables

Ces résultats suggèrent que DeepLab apprenait des règles visuelles stables pour les surfaces arctiques, même en présence de lumière difficile, de brouillard ou de fonte avancée. Le résultat

est une carte de segmentation plus cohérente et interprétable, avec des sorties régionales respectant à la fois la continuité visuelle et la sémantique des classes.

6. Évaluation et Résultats

Ce chapitre évalue le pipeline final de segmentation à l'aide de mesures quantitatives (IoU, score F1, précision, rappel) ainsi que de visualisations qualitatives. L'accent est mis sur les performances dans des conditions réelles — éblouissement, frontières ambigües entre la neige fondante et l'eau, et fragmentation des zones de glace.

Tous les résultats présentés proviennent du pipeline hybride complet (SAM ou SLIC, graphe RAG, fusion, DeepLabv3+), sans lissage, ensemblage ou correction post-hoc supplémentaire.

6.1 Jeu de données et protocoles d'évaluation

Le jeu de test a été extrait d'une portion réservée du jeu de données CAIMAN, comprenant environ 500 images manuellement sélectionnées issues de différents sites du Nunavik. Ces images couvrent :

- des rivages gelés et des transitions de fonte
- des zones de surface neigeuse fondante et d'eau libre
- des terrains post-débâcle sous divers éclairages

Chaque image a été évaluée de manière indépendante, sans lissage temporel ni correction autre que celle appliquée par le pipeline.

Étant donné que les annotations de la neige fondante dans le jeu de données sont imparfaites, l'évaluation a mis l'accent sur l'interprétabilité et la cohérence sémantique plutôt que sur une correspondance stricte pixel à pixel. Les résultats qualitatifs sont donc considérés aussi importants que les mesures numériques.

Spécifications du jeu de test :

- **Taille :** environ 500 images
- **Contenu :** sites et phases variés
- **Traitement :** pipeline complet image-par-image (SAM - RAG - fusion - DeepLabv3+)
- **Évaluation :** priorité à la cohérence visuelle et aux performances statistiques

6.2 Métriques quantitatives (IoU, rappel, etc.)

Pour évaluer la performance dans des conditions de rivage complexes, nous avons utilisé cinq métriques :

Métrique	Ce qu'elle mesure	Pourquoi elle est importante pour ce projet
Précision globale	Pourcentage de pixels correctement classifiés	Mesure la justesse générale sur les images nettes (glace, terre, eau bien segmentées).
IoU moyen	Chevauchement moyen entre les régions prédites et réelles	Évalue l'alignement structurel, essentiel pour les frontières de la neige fondante.
Score F1	Moyenne harmonique de la précision et du rappel	Mesure l'équilibre entre détection valide et sur-prédiction dans les zones ambigües.
Précision	Vrais positifs / Total des positifs prédits	Réduit les faux positifs — essentiel pour éviter des erreurs confiantes dans les zones incertaines.
Rappel	Vrais positifs / Total des positifs réels	Évite de manquer les régions valides — utile pour détecter les transitions progressives de surface.

Tableau 6.1 : *Définitions des métriques et leur pertinence pour la qualité de la segmentation.*

Le pipeline est resté modulaire durant tous les tests — seul le backbone du classifieur **DeepLabv3+** a été modifié (ResNet-50 et ResNet-101). Les propositions régionales (via SAM ou SLIC) et la logique de fusion ont été conservées identiques. Les résultats des versions de décembre 2024 et mars 2025 sont présentés ci-dessous :

Pipeline	Précision	IoU moyen	Score F1	Précision	Rappel
Déc 2024 (ResNet-50)	0,73	0,15	0,37	0,40	0,95
Mar 2025 (ResNet-50)	0,92	0,83	0,92	0,92	0,92
Déc 2024 (ResNet-101)	0,75	0,15	0,35	0,37	0,95
Mar 2025 (ResNet-101)	0,89	0,78	0,88	0,90	0,89

Tableau 6.2 : *Comparaison quantitative des versions du pipeline utilisant DeepLabv3+ avec les backbones ResNet-50 et ResNet-101.*

Le pipeline final basé sur ResNet-50 a surpassé celui basé sur ResNet-101, obtenant les meilleurs scores en précision, score F1 et IoU, malgré une architecture plus simple. Cela suggère que le raisonnement basé sur les régions a compensé la complexité du modèle, évitant le surapprentissage.

Principaux enseignements :

- **Le score F1** est passé de ~0,36 à >0,90, indiquant une prédiction confiante sans perte de couverture.
- **La précision a augmenté**, réduisant les faux positifs, notamment dans les zones de neige fondante.
- **L'IoU** moyen a dépassé 0,78, validant l'alignement spatial au niveau des bords.
- **Le modèle** final a surpassé les annotations humaines dans les scènes complexes (éblouissement, surfaces mixtes).

Cela dit, les moyennes globales masquent des faiblesses spécifiques — surtout pour la neige fondante. Le **rappel élevé** (~0,95) dans les premières versions reflétait une sur-segmentation. Seule la version de mars 2025, avec un entraînement orienté précision et des seuils de fusion ajustés, a commencé à gérer la neige fondante de manière plus prudente et contextuelle. Le modèle n'étiquette désormais la neige fondante que lorsque les indices visuels sont fiables — et s'abstient sinon.

6.3 Comparaisons visuelles qualitatives

Pour évaluer le fonctionnement réel du système au-delà des chiffres, j'ai mené des tests visuels sur une large gamme de scènes — allant de cadres nets et à fort contraste à des conditions fortement encombrées et saturées d'éblouissement.

6.3.1 Conditions idéales

Dans des scènes bien éclairées avec des surfaces clairement définies, le système fonctionne de manière impressionnante. Les frontières entre terre, glace et eau sont respectées, même dans des arrangements denses de floes, et le classifieur sépare les régions avec confiance. Les superpositions finales montrent un alignement spatial précis avec les éléments réels de l'image.

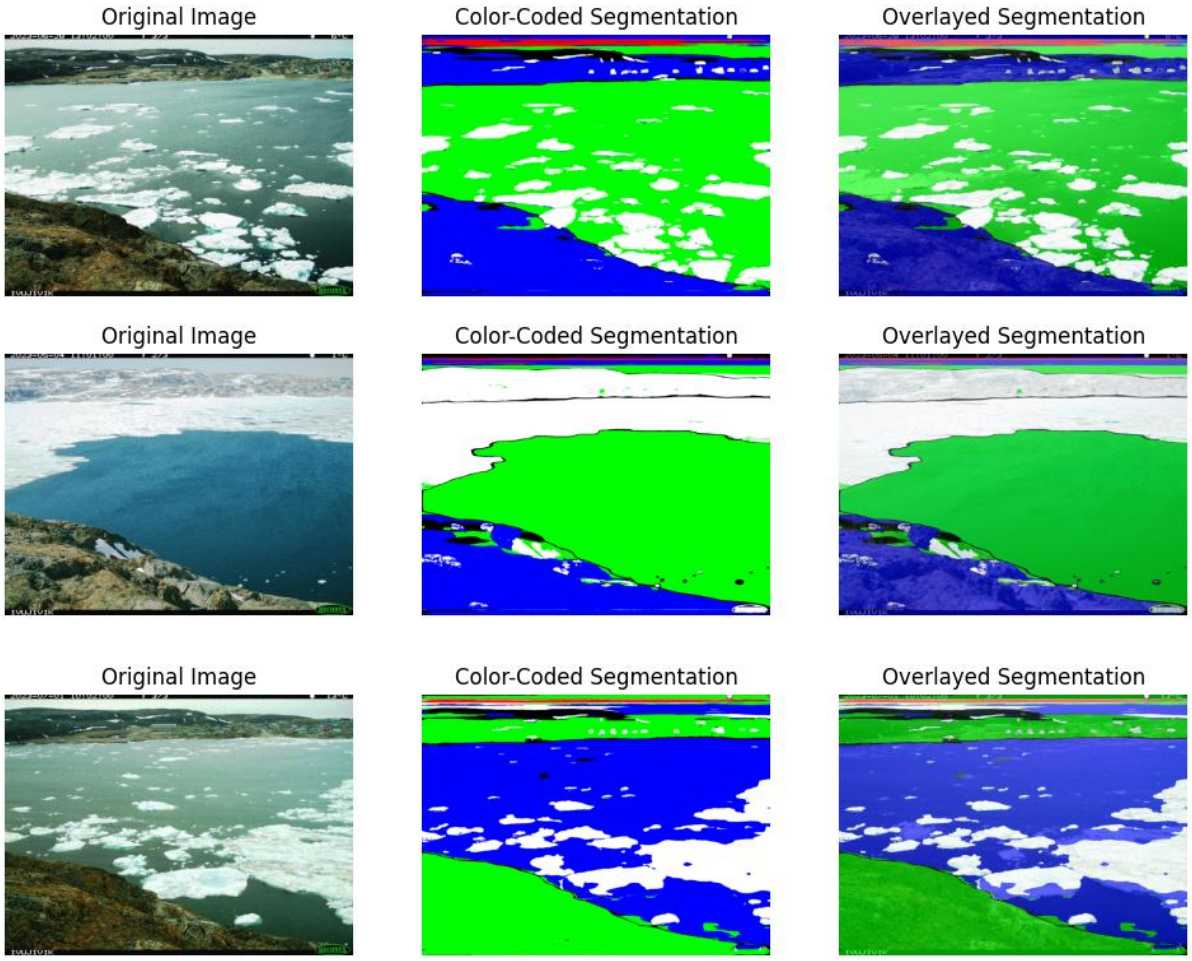

Figure 6.1 : *Résultats de segmentation DeepLabv3+ dans des conditions idéales. Les floes individuels ainsi que les limites entre la terre et l'eau sont clairement identifiés et étiquetés avec une grande cohérence.*

La distinction entre la glace terrestre et la glace sur l'eau devient beaucoup plus nette dans ces conditions, avec des floes individuels correctement identifiés et affectés à des étiquettes cohérentes.

6.3.2 Surexposition et éblouissement

Dans les images présentant un fort éblouissement ou une fragmentation excessive, le pipeline de fusion rencontre souvent des difficultés à bien fonctionner. Les régions qui devraient être distinctes se retrouvent regroupées, ce qui rend difficile le suivi de la neige fondante.

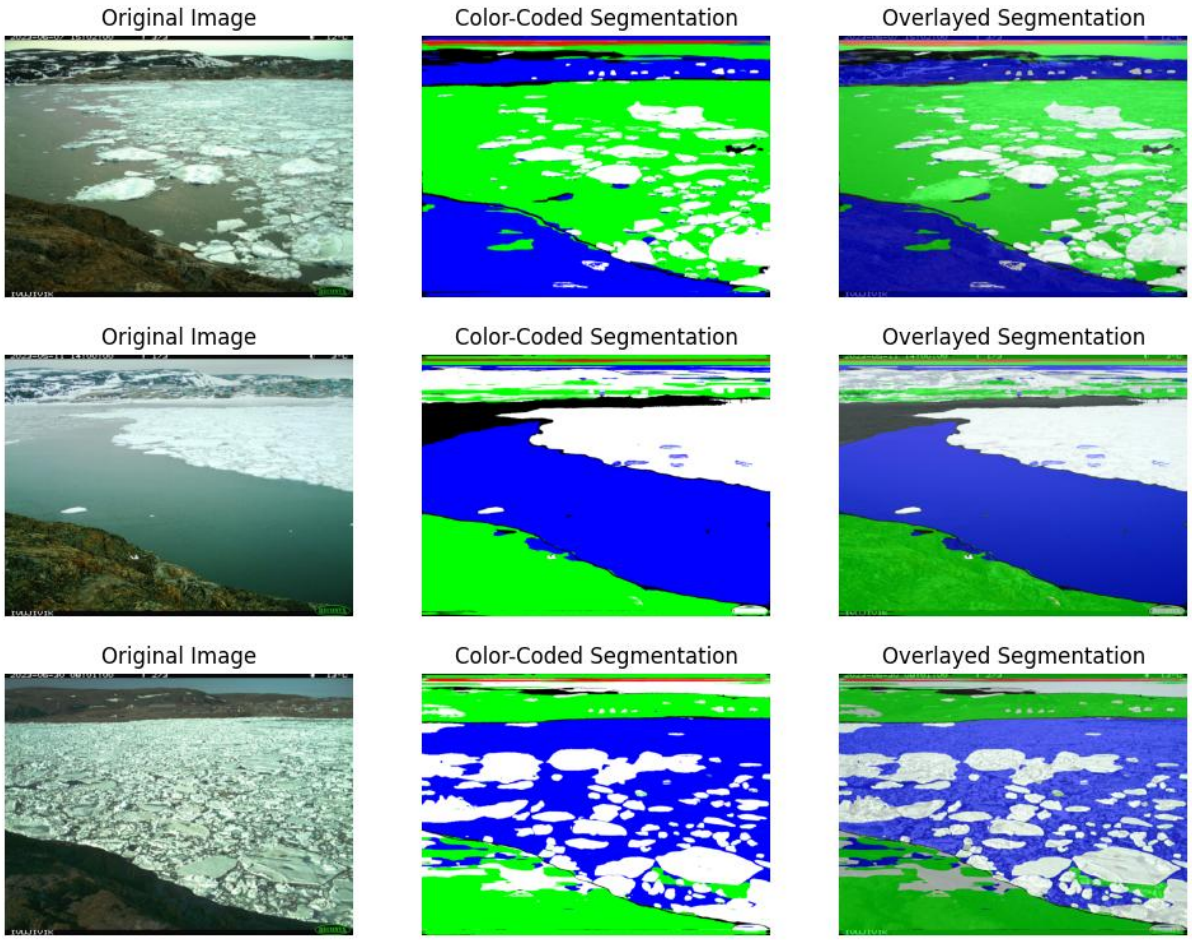

Figure 6.2 : *Résultats de segmentation DeepLabv3+ dans des scènes surexposées contenant de l'éblouissement et de l'encombrement visuel.*

Ces problèmes ne sont pas causés par une mauvaise classification des pixels par DeepLab — ils proviennent de la structure en amont. La logique de fusion hiérarchique n'arrive pas toujours à suivre les variations rapides de texture et de luminosité.

6.4 Analyse des échecs

Bien que le pipeline final fonctionne correctement dans la majorité des conditions d'éclairage, il présente encore des difficultés dans les zones de texture ambigüe — en particulier la glace morcelée (brash ice), la neige fondante (slush), et le désordre textural entre la glace et l'eau. Ces échecs ne sont pas dus au classifieur DeepLab lui-même, mais à la difficulté à regrouper ces régions de manière structurée avant la classification.

Pour isoler ce problème, nous avons réalisé des analyses d'histogrammes au niveau des régions en utilisant deux caractéristiques calculées : la luminance moyenne et l'écart-type des couleurs, comme illustré à la **Figure 6.3**.

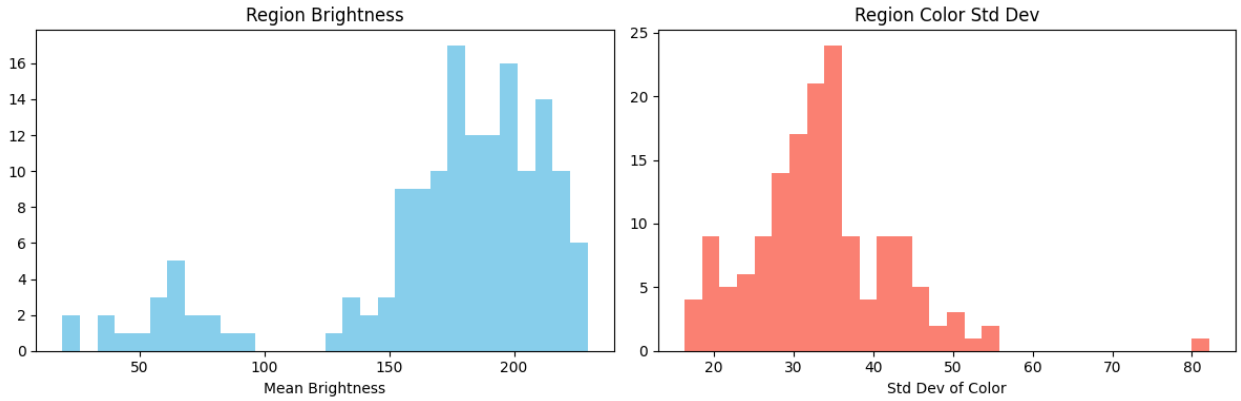

Figure 6.3 : *Histogrammes de la luminance moyenne par région (à gauche) et de l'écart-type de couleur (à droite). Utilisés pour détecter l'éblouissement, la complexité ou l'ambiguïté texturale dans les régions fusionnées.*

Ces histogrammes montrent les distributions régionales pour :

- **Luminosité.** On observe un regroupement stable des régions dans l'intervalle 150–220. Les valeurs aberrantes en dessous de 80 (zones d'ombre) et au-delà de 230 (reflets éblouissants) sont corrélées à des fusions instables et à une fragmentation des régions.
- **Variance.** La majorité des régions se situent entre 30 et 40. Les valeurs supérieures à 70 correspondent systématiquement à la glace pilée et à la neige fondue, où la logique de fusion se désagrège.

Analyse conjointe. Ces résultats montrent que les échecs de segmentation apparaissent de manière récurrente aux extrêmes de luminosité et de variance. Le pipeline reste stable dans les plages centrales, mais devient fragile lorsque les seuils s'éloignent vers les valeurs extrêmes. En pratique, cette analyse définit une **« zone de stabilité »** pour la logique de fusion : une luminosité modérée (150–220) et une variance intermédiaire (30–40). En dehors de ces plages, l'instabilité est prévisible.

En particulier, les zones de type **neige fondue** combinent une luminosité moyenne avec une variance exceptionnellement élevée, produisant une signature récurrente d'ambiguïté. Cette

observation met en évidence la sensibilité intrinsèque du processus de fusion à ces extrêmes statistiques, qui représentent les points de rupture où la segmentation perd sa fiabilité.

Les régions correspondant aux signatures connues de la neige fondante (luminance modérée, forte variance)

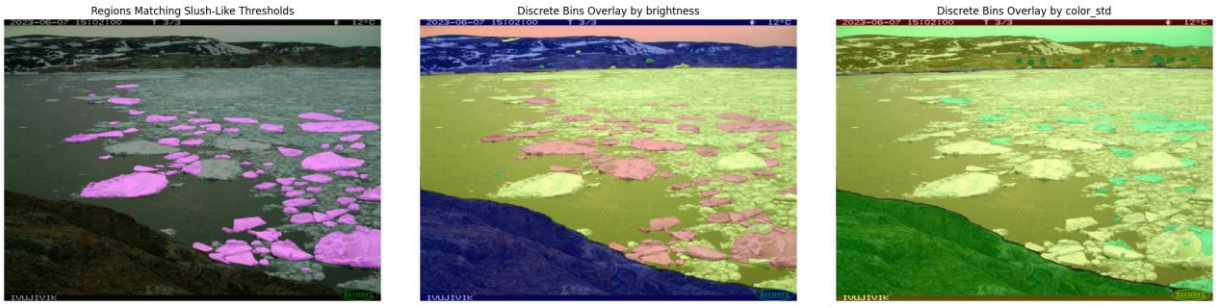

Figure 6.4 : *Cartes de surimpression des échecs : régions incorrectement divisées ou omises durant la fusion.*

Pour tester si ces signatures pouvaient isoler des cas d'échec systématiques, j'ai appliqué des seuils manuels afin de marquer certaines régions :

- **Zones fortement éblouissantes (haute luminosité).** Les grandes surfaces très brillantes se fragmentent en plusieurs petites régions, ce qui rend la fusion instable et produit des frontières incohérentes.

- **Zones encombrées (forte variance).** Les régions visuellement complexes, comme les amas de glace pilée, présentent un regroupement instable. Elles sont soit sur-segmentées en fragments incohérents, soit entièrement éliminées, ce qui obscurcit la structure pertinente.

- **Zones de transition de type neige fondue (luminosité intermédiaire + forte variance).** Ces profils échouent systématiquement à former des régions cohérentes, générant des masques fragmentés que DeepLab classifie à tort comme des mélanges d'eau et de glace.

La concentration d'échecs dans ces signatures spécifiques indique que les erreurs ne sont pas distribuées au hasard, mais bien liées à des extrêmes de caractéristiques prédictibles. Cela suggère que l'intégration de règles de fusion sensibles aux caractéristiques ou de seuils adaptatifs pourrait cibler directement ces zones fragiles, renforçant ainsi la stabilité et la cohérence du pipeline dans de futures itérations.

6.5 Tentatives de résolution des modes d'échec

Après avoir identifié les problèmes de segmentation incohérente de la neige fondante et d'instabilité de la fusion dans les scènes surexposées, j'ai mené plusieurs expériences

complémentaires pour mieux comprendre ces limitations et tenter de les résoudre. Un exemple de ces tentatives est illustré à la **Figure 6.5**.

Certaines de ces tentatives incluent :

- **Seuils régionaux :** filtrage des régions selon la luminance et la texture pour isoler les zones potentiellement correspondantes à de la neige fondante.
- **Binning des caractéristiques :** utilisation de plages de couleur et de variance pour visualiser les zones mal étiquetées et détecter les erreurs de seuils.
- **Fusion tardive ajustée :** application de règles de fusion plus permissives en post-traitement pour reconnecter les régions fragmentées de neige fondante — efficace structurellement, mais incohérente d'un point de vue sémantique.
- **Injection manuelle de régions :** ajout de masques manuellement annotés pour la neige fondante durant l'entraînement ; mais sans labels nets, le classifieur n'a pas pu en tirer des règles utiles.

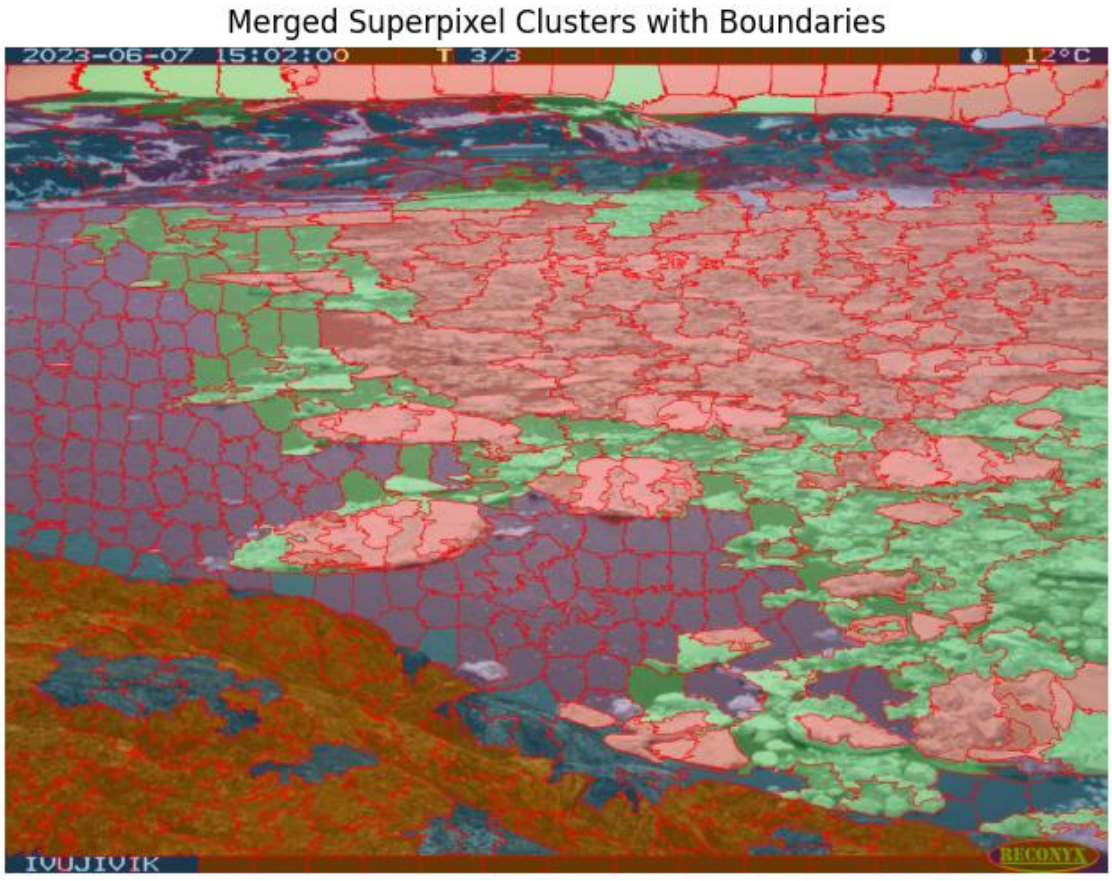

Figure 6.5 : *Corrections post-hoc appliquées aux régions de neige fondante. Comprend l'injection manuelle de masques pour restaurer la cohérence sémantique.*

Bien que ces tentatives aient révélé la structure sous-jacente des échecs, elles n'ont pas permis d'améliorations directement exploitables. Les prédictions de DeepLab se détériorent dès que les régions sont trop modifiées en amont, ce qui suggère que les améliorations futures nécessiteront soit de meilleurs labels, soit une formation intégrée avec la phase de fusion.

Malgré cela, ces observations fournissent une base solide pour une fusion plus sensible aux règles et pour des stratégies spécifiques à la neige fondante.

7. Réflexions et perspectives générales

7.1 Enseignements techniques et contributions

Ce travail présente un pipeline de segmentation hiérarchique combinant des transformeurs visuels avec un raisonnement structuré au niveau des régions — une approche rarement utilisée pour des images ambigües et à faible contraste, comme celles des scènes arctiques. Au lieu de classifier chaque pixel individuellement, le système regroupe les régions visuellement similaires et raisonne sur ces segments, ce qui améliore les performances sur des surfaces comme la gadoue ou l'eau réfléchissante, où les frontières sont floues.

Bien qu'il ait été développé pour les glaces polaires, le caractère modulaire du pipeline le rend facilement adaptable à d'autres domaines, tels que la cartographie de la végétation ou la détection urbaine. Chaque composant fonctionne indépendamment, permettant une réutilisation flexible. Bien que le jeu de données contienne peu d'exemples de gadoue, le système généralise tout de même efficacement dans ces scènes, ce qui démontre sa robustesse.

7.2 Résumé des résultats

Ce mémoire présente un pipeline de segmentation modulaire adapté aux environnements ambigus et à faible contraste — spécifiquement, la terre, la glace, la gadoue et l'eau capturées par des caméras fixes en milieu arctique. Le système combine une sur-segmentation (via SAM ou SLIC), une fusion de régions par graphe, et une classification par région via DeepLabv3+. Ensemble, ces étapes permettent au pipeline de raisonner localement et sémantiquement sur des scènes où les modèles classiques échouent.

Le système final a montré des gains importants en précision et en capacité de généralisation, en particulier dans les images denses en frontières ou avec des reflets intenses. Grâce à une prétraitement structuré et une classification finement ajustée, le pipeline a surpassé même ses propres annotations bruitées. Bien que la gadoue reste un point faible, les résultats démontrent que les régions ambiguës peuvent être modélisées sans recourir à des règles manuelles ou à des capteurs denses.

7.3 Limites et problèmes ouverts

Malgré les progrès réalisés, certains défis fondamentaux demeurent irrésolus ; ils sont résumés dans le **tableau 7.1**. Ces limites reflètent des problèmes plus profonds, liés soit à la structure du système, soit à la qualité des données, et restreignent la généralité et la précision du pipeline.

Domaine	Problème	Pourquoi c'est important	Ce qui pourrait aider
Détection de gadoue	La gadoue est souvent ignorée ou fusionnée avec la glace/l'eau en raison d'un manque de texture et de données d'apprentissage de faible qualité	C'est la classe la plus ambiguë, mais essentielle pour la surveillance du littoral	Collecter de meilleures annotations, idéalement avec des experts ou du crowdsourcing
Critères de fusion	Les seuils fixes échouent dans les scènes avec fort éblouissement ou durant la fonte	Provoque des fusions incorrectes et des frontières mal définies	Remplacer les règles fixes par des métriques apprises, adaptatives
Compréhension de texture	Le classificateur DeepLab n'est pas assez sensible aux variations fines de texture	Réduit la précision près des transitions, comme la glace fondue ou fine	Ajouter des descripteurs centrés sur la texture ou des encodeurs multi-échelles
Stabilité temporelle	Les prédictions varient d'une image à l'autre ; aucune cohérence dans le temps	Rend l'analyse vidéo peu fiable	Ajouter un lissage temporel ou intégrer le contexte temporel au modèle

Tableau 7.1 : *Résumé des limitations connues et des pistes d'amélioration sur les plans structurel et sémantique.*

7.4 Perspectives pour les travaux futurs

Ce projet n'avait jamais pour but de s'arrêter à la segmentation. Sa véritable force réside dans la structure qu'il génère — une vision modulaire du monde qui ouvre la voie à une compréhension d'image plus large, au-delà des environnements arctiques.

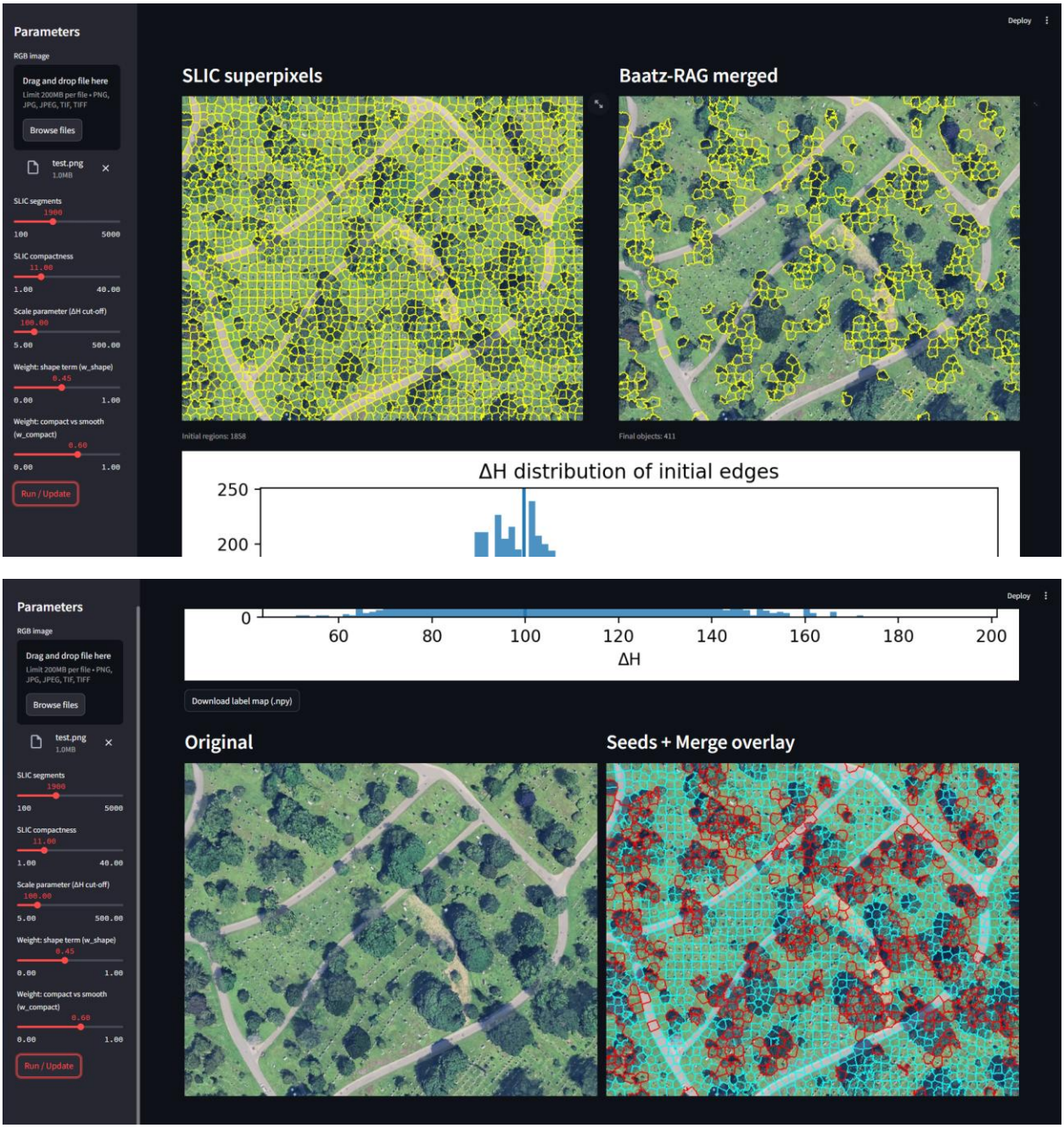

Figure 7.1 : *Exemple d'un nouveau projet de détection d'objets de type OBIA utilisant une logique basée sur les régions dans des jeux de données aériens.*

Une de ces directions est déjà en cours (l'illustration de la **Figure 7.1** en donne un aperçu). Je travaille actuellement sur un framework de détection d'objets de type OBIA [18] adapté aux grands jeux de données géospatiaux. Le système repose sur une logique régionale : sur-segmentation par SLIC, fusion des régions basée sur l'adjacence, des indices de texture inspirés de Baatz, puis classification par un modèle compact spécifique à la tâche. Cette approche est guidée par le même principe qui a inspiré ce mémoire : partir de la structure, puis laisser émerger la sémantique.

Une orientation centrale de ce projet concerne la **mise à l'échelle dynamique des scènes pour les superpixels**. Contrairement à la segmentation à résolution fixe, qui applique une granularité uniforme à l'ensemble de l'image, l'approche proposée adapte le niveau de partitionnement à la **complexité visuelle locale**. Les grandes zones homogènes, telles que les étendues d'eau ou de ciel uniforme, sont représentées par des **partitions plus grossières** afin de réduire la redondance. En revanche, les **zones de transition ou ambiguës** — incluant la neige fondue, la glace pilée et les frontières entre l'eau et la glace — sont segmentées avec une **granularité plus fine**, ce qui permet de préserver les variations subtiles.

Cette stratégie adaptative répond directement à l'une des principales limites identifiées dans ce travail : la **sensibilité du pipeline à la qualité des superpixels**. En modulant la résolution en fonction du contexte, le système réduit la charge de calcul dans les régions simples tout en améliorant la précision dans les zones où la classification est la plus sujette à l'erreur. Une telle mise à l'échelle dynamique ouvre la voie à une **meilleure efficacité et robustesse**, en diminuant la dépendance à une résolution de segmentation fixe.

En regardant plus loin, mon projet de doctorat proposé explore comment ces idées structurelles peuvent être appliquées à la robotique d'intérieur. Dans des environnements complexes et mal définis — comme des maisons encombrées ou des bâtiments partiellement effondrés — les catégories sémantiques traditionnelles perdent leur sens. Mon objectif est de développer des systèmes qui ne se contentent pas de voir « ce qui » se trouve devant eux, mais qui comprennent « où » et « comment » se déplacer. Cela implique la création de représentations spatiales permettant aux robots de planifier des trajectoires sûres à travers des environnements incertains tout en restant interprétables et adaptables.

Dans mes travaux en cours comme futurs, une idée centrale persiste : l'ambiguïté est elle-même une forme d'information qui, lorsqu'elle est bien exploitée, peut rendre les systèmes plus puissants.

TABLE DES MATIÈRES

LISTE DES FIGURES

LISTE DES TABLEAUX

LISTE DES ABREVIATIONS

CAIMAN	Caméras aux Infrastructures Marines au Nunavik shoreline-monitoring network
CNN	Convolutional Neural Network
CPU	Central Processing Unit
GPU	Graphics Processing Unit
IoU	Intersection-over-Union (overlap metric)
mIoU	Mean Intersection-over-Union
OBIA	Object-Based Image Analysis
PSPNet	Pyramid Scene Parsing Network
RAG	Region Adjacency Graph
RGB	Red–Green–Blue colour space
SAM	Segment Anything Model
SLIC	Simple Linear Iterative Clustering (superpixels)
U-Net	“U”-shaped encoder–decoder network for segmentation
ViT	Vision Transformer

1. INTRODUCTION

1.1 History of This Research

This thesis addresses the ice–slush–water classification problem. In the summer of 2022, I completed a Mitacs Globalink internship under the supervision of Prof. Saeid Humayouni, working on the CAIMAN project. CAIMAN — Caméras aux Infrastructures Marines au Nunavik — is a shoreline-monitoring network of fixed RGB cameras installed at seven Nunavik sites by the Kativik Regional Government, INRS, and Transports Québec.

The camera footage is primarily used to monitor and analyze the evolution of ice conditions, enhancing understanding and supporting adaptation strategies, particularly for Indigenous populations. The main objective of my internship was straightforward on paper: transform months of land-based Arctic camera footage into a dataset suitable for reliable segmentation.

Over the following twelve weeks, I came to understand that glare, snow storms, drifting snow, and featureless ice render “straightforward” effectively meaningless. The same footage was later brought into a master’s program, where we continued working, focusing on the segmentation and classification of ice in the landscape imagery captured by the CAIMAN network.

During the first three semesters, beginning in September 2023, we focused on reviewing the relevant literature, followed by the creation of an annotated image dataset and related refinements. Subsequently, we began exploring effective machine learning models and appropriate training strategies for ice segmentation. This phase involved iterative improvements to the dataset and a systematic investigation of the persistent limitations found in existing models.

1.2 Motivation and Context

The goal of this research is to develop effective machine learning–based image segmentation techniques that support the monitoring of seasonal ice conditions and generate data to inform studies on coastal impacts and climate change adaptation. Each camera streams a new frame at regular intervals (**Figure 1.1**); each frame must be converted, on the fly and without the use of extra sensors, into three surface classes: solid ice, slush, and open water. The images are cheap to collect and easy to send, but they tell us little if we cannot reliably separate firm ice from slush or open water.

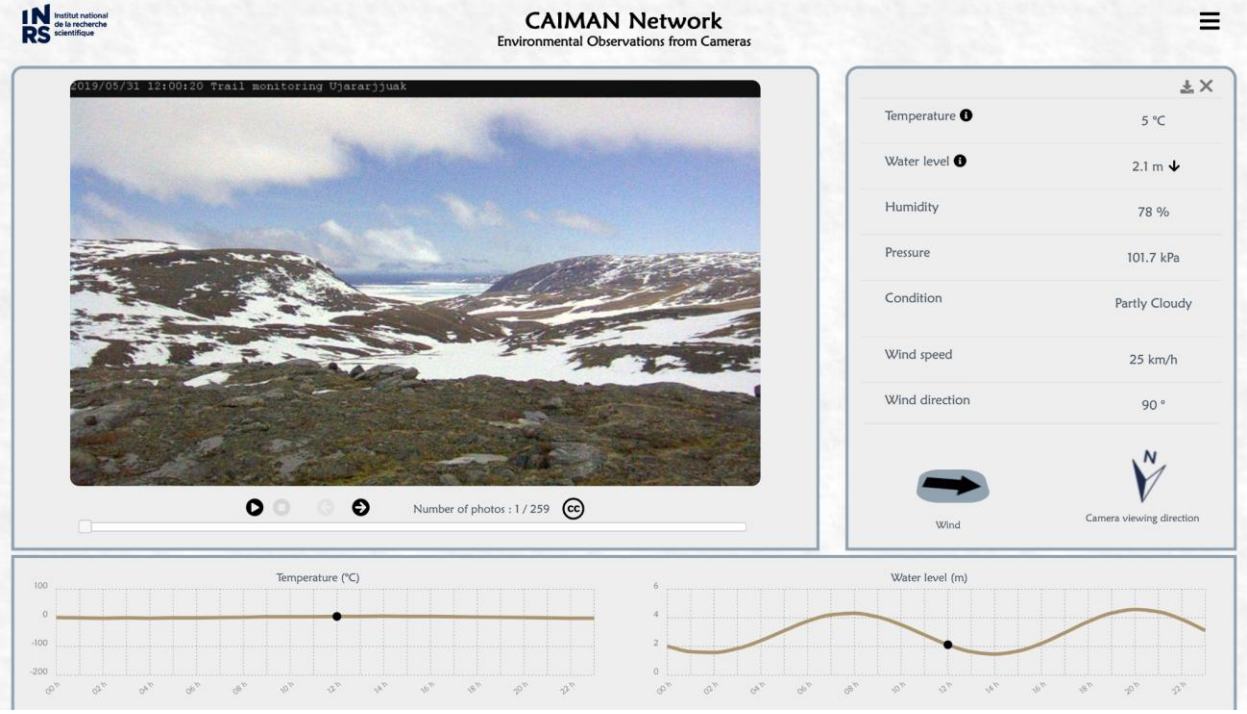

Figure 1.1 *CAIMAN dashboard showing camera feed and environmental metadata.*

Sail-route planners, local hunters, field researchers, and hazard teams all require that separation in near real-time, on ordinary hardware. The image segmentation results developed in this thesis aim to support safe marine and ground transportation in Arctic environments by providing timely and reliable surface classification without relying on specialized sensors or cloud-based computation.

When this research on the CAIMAN project began in early 2023, we initially applied commonly used image segmentation models such as Mask R-CNN [1] and U-Net [2]. However, these models failed to achieve the required accuracy due to several challenges, including glare-obscured edges, lack of clear separation between surface types, the disappearance of thin slush layers, and inconsistent label predictions across consecutive frames.

Working with a single fixed camera instilled a useful habit: treating each frame as nothing more than a three-channel matrix and looking for ways to rearrange those numbers until a structure emerged. This idea earlier in a separate project on reimplementing AdaBins [3], where adaptive bins and attention layers extracted depth estimates from a single camera view. The algorithm itself is unrelated to ice segmentation, but it showed that precise geometry can emerge from a single sensor if the grouping rule is right.

That insight shaped the rest of this thesis. Instead of searching for a one-shot model, I built a pipeline that groups pixels by simple cues and tests those groups against local context to assign a label. The pages that follow explain why the standard tools fail on this task, how the new pipeline is assembled, and how well it works on such ambiguous data.

1.3 Challenges of Ice-Slush-Water Segmentation

In most CAIMAN images — especially those taken over Hudson Bay — the water, sky, land, and ice often blend into each other. On bright days, glare flattens the frame so severely [27] that even a human viewer would struggle to distinguish between slush and shallow water, or wet ice and cloud reflection, if shown the images out of context, as illustrated in **Figure 1.2**. These distinctions often depend on subtle textures that shift frame to frame and are not tied to reliable color or shape cues.

***Figure 1.2**: Sample CAIMAN images showing slush, glare, and low-contrast regions.*

Unlike typical image segmentation tasks, there are no well-defined boundaries. Ice does not come in discrete objects. Slush, in particular, exists as a vague transition zone — visually and physically — between water and ice, with no clear point where one ends and the other begins.

The scarcity of training data exacerbates this ambiguity. The most common datasets used in semantic segmentation are built around everyday images — such as animals, people, streets, and buildings — where class boundaries are intuitive and can be annotated using edge tools. That does not translate to Arctic environments. Even expert annotators disagree on where to draw the line between slush and water. We are left working with uncertain thresholds and the risk of introducing more noise than ground truth.

As a result, both model training and evaluation become fragile. A model may produce a clean mask, but it is not apparent whether that mask is correct or simply confident.

1.4 Objectives of the Thesis

Image understanding tasks are typically divided into three broad categories: **image classification**, where a single label is assigned to the whole image; **object detection**, where bounding boxes are drawn around specific instances; and **segmentation**, where each pixel is assigned a class label (**Figure 1.3** visualizes the difference).

Segmentation itself is further split into **semantic segmentation,** where all pixels of a given class share the same label, and **instance segmentation**, which also distinguishes between individual objects of the same class.

***Figure 1.3**: Examples of classification, detection, semantic segmentation, and instance segmentation.*

This project focuses on semantic segmentation of images captured by the CAIMAN camera network, intending to assign each pixel in land-based Arctic imagery one of three labels: ice, slush, or open water.

Unlike traditional object classes, these categories represent material surfaces that lack consistent shape or boundaries. This thesis proposes a segmentation pipeline capable of handling the inherent ambiguity of such scenes — especially in cases where visual cues are weak or misleading — and to generate pixel-level labels that are consistent, reliable, and suitable for use in downstream environmental monitoring applications.

1.5 Structure of the Document

This thesis is structured to reflect both the chronological evolution and modular design of the segmentation pipeline developed for Arctic surface understanding.

- **Chapter 1 (Introduction)** defines the environmental and operational challenges of Arctic field imagery, introduces the core task of ice–slush–water separation, and outlines the motivation behind object-based segmentation over pixel-level approaches.

- **Chapter 2 (Background and Related Work)** reviews segmentation architectures, structural instance segmentation, graph merging techniques, and superpixel clustering. It also highlights gaps in current methodologies for ambiguous surface segmentation in constrained environments.

- **Chapter 3 (Data and Annotation Process)** explains the evolution of the dataset across three major versions. V1 was created manually using the PixelAnnotationTool; V2 resolved inconsistencies and class ambiguities across ~4,000 curated samples; and V3 incorporated high-confidence pipeline outputs along with additional edge cases for improved generalization.

- **Chapter 4 (Initial Modeling Attempts)** presents early experiments with U-Net, PSPNet, and Mask R-CNN, as well as a simple ensemble approach. These trials reveal the limitations of pixel-level models in handling ambiguous transitions, motivating the shift toward region-first methods.

- **Chapter 5 (Proposed Pipeline)** introduces the final modular architecture, which integrates region generation (SAM/SLIC), greedy region merging over a graph-based representation, and region-wise classification using DeepLabv3+. It discusses design choices for GPU and CPU variants, and explains how each component contributes to stable surface segmentation across varying visual conditions.

- **Chapter 6 (Evaluation and Results)** presents both quantitative metrics and qualitative visual analysis across varied conditions. It covers success cases, failure modes, boundary-level histograms, and post-hoc corrections to assess the pipeline's robustness.

- **Chapter 7 (Conclusion and Future Work)** summarizes the system's contributions, discusses unresolved issues like slush misclassification, and suggests extensions to other geospatial applications or robotics domains.

This structure mirrors the technical progression of the project — from exploration and failure to modular design and practical deployment — and emphasizes the decisions made under real-world constraints.

2. BACKGROUND AND INITIAL HYPOTHESES

2.1 Segmentation in Unpredictable and Ambiguous Scenes

Most segmentation models rely on the assumption that structural information can be inferred from cues such as color, contrast, or object boundaries, and that these visual features remain consistent across the dataset. However, that assumption breaks down in many real-world scenes. In the CAIMAN dataset, which utilizes fixed RGB cameras in dynamic environments, the appearance of a class can shift from frame to frame without geometric consistency.

This challenge is not unique to ice. Any visual setting characterized by vague or dynamic material boundaries suffers from the same challenge: how can pixel-wise labels be assigned in the absence of stable, class-defining features?

To explore the existing approaches to this type of problem, we categorized the relevant literature into three general groups:

- Prior research on segmentation in ice- and snow-covered environments
- General-purpose CNN-based segmentation models
- Graph-based or OBIA-inspired methods

Each group offered something helpful, even if none fully solved the problem on their own.

2.1.1 Ice-Focused Segmentation Work

Some of the closest prior work came from ice segmentation itself, although most of it used drone or ship-mounted cameras and aerial images. These studies typically adapt established segmentation architectures—such as DeepLab or Mask R-CNN—trained on small, often manually annotated datasets whose characteristics differ markedly from those of the CAIMAN dataset.

Table 2.1 summarizes the most relevant papers we reviewed in this category.

Paper	Model	Input Type	Classes	Limitations
IceMaskNet [7]	Mask R-CNN	Aerial photos	6 river ice types	Different resolution, non-stationary view
Lake-Ice Monitoring [8]	DeepLabv3+ / Deep-U-Lab	Webcam images	Lake ice, water, and background	Good backbone, but inconsistent input quality
Sea-Ice SegNet [10]	PSPNet101, SegNet	Icebreaker footage	Ice, ocean, vessel, sky	Tiny dataset, narrow domain
Ice-Deeplab [12]	DeepLab + attention	Antarctic scenes	Ice, ocean, sky	Accuracy may not generalize to the CAIMAN setup

Table 2.1*: Summary of related segmentation work in ice-covered environments.*

A review of these models reveals that segmentation is possible even with ambiguous classes, but also that most systems rely heavily on curated data or prior assumptions about the environment.

2.1.2 General Semantic Segmentation Models

Alongside the ice-focused literature, direct experimentation was conducted using a set of well-established semantic segmentation models. These included DeepLabv3+ [4], U-Net, PSPNet [5], and, more recently, the segmentation variant of YOLO (YOLOv8-seg) [14]. These architectures are widely adopted across segmentation tasks; their core designs are listed in Table 2.2**,** and frequently serve as baselines in both academic and applied research.

Model	Core Design	Strengths / Common Use
U-Net [2]	Skip connections with a symmetric encoder-decoder	Widely used in biomedical and low-data settings
PSPNet [5]	Pyramid pooling module for multi-scale context	Effective for capturing global spatial information
DeepLabv3+ [4]	Atrous convolutions with refinement modules	Strong on standard semantic segmentation tasks
YOLOv8-seg [14]	Unified detection + segmentation head	Fast, real-time inference, easy integration

Table 2.2*: Semantic segmentation models used in this study and their core design features.*

These models were evaluated on a subset of the CAIMAN dataset using various preprocessing configurations, including resized crops, downsampled frames, and contrast-adjusted inputs. No custom training was performed at this stage; the objective was to assess their out-of-the-box generalization performance and determine whether any outputs could serve as a basis for post-processing.

These inspections and experiments yielded critical insights that guided the design of input representations and model architectures in subsequent stages of the pipeline.

2.1.3 Graph-Based Approaches

Graph-based methods were only explored later in the project. Initial research focused on OBIA [17] and remote sensing literature, which predominantly involved large, multi-channel satellite images. These approaches appeared overly complex and not directly applicable to the CAIMAN setup, which involved a single RGB frame from a fixed camera.

However, as deep learning models consistently failed to produce reliable outputs, simplified adaptations of these region-based ideas were tested, including partitioning the image into coherent regions, assigning meaning to each area, analyzing spatial connectivity, and merging based on color or texture similarity. Although complete OBIA systems were not adopted, the underlying logic of object-based structuring made a meaningful contribution to the final pipeline.

This direction, although not initially planned, proved valuable: the structural representations offered by region-based methods became crucial for understanding the spatial relationships between scene components. These representations provided a scaffold for semantic segmentation by enabling the system to reason over groups of pixels as coherent entities rather than isolated classifications.

2.2 Initial Hypotheses and Directions

At the start, the direction seemed obvious. Given the structure of models like DeepLabv3+ or U-Net, the most straightforward path was to create a clean, well-annotated dataset and use it to fine-tune a segmentation model on the CAIMAN frames.

The architecture was present, and numerous papers demonstrated strong results using similar setups on environmental datasets. All indications suggested that the challenge lay primarily in data curation and effective training.

The early focus was on selecting representative frames from the CAIMAN archive, developing consistent annotation guidelines, and building a small but clean training set. Once that was in place, the idea was to compare outputs from a few standard semantic segmentation models, and either use one as-is or adapt it into a more customized pipeline.

This approach was logically sound — at least in principle. It was based on the assumption that, given a sufficiently curated dataset, a standard model would be capable of learning the task. While this assumption held to a certain extent, the experiments revealed important limitations and nuances, which are discussed in the following chapters.

3. DATASET CREATION AND ANNOTATION

The creation of a high-quality annotated image dataset plays a crucial role in the success of machine learning–based image segmentation methods. The datasets developed in this work are derived from raw images collected by the CAIMAN monitoring network. Each camera captures **a single frame approximately every 15 minutes**, continuously monitoring shoreline conditions throughout the freeze–thaw cycle.

The original images have a resolution of **2048×1536 pixels (4:3 aspect ratio)** and are exported in both full-frame and cropped formats depending on the site layout. While each frame offers sufficient spatial detail for segmentation, significant changes in surface conditions tend to occur over multi-day timescales rather than within individual days. Consequently, temporal sampling for annotation and analysis is conducted on a daily scale, focusing on frames that reflect significant environmental transitions.

In this research, we employed three distinct techniques and strategies to generate an annotated image dataset from the raw imagery. The following sections detail each approach and discuss the resulting dataset quality, along with their respective advantages and limitations.

3.1 Version 1 — PixelAnnotationTool-Based Manual Labels

The first version of the dataset was created during a Mitacs Globalink Research Internship under the supervision of Dr. Saeid Humayouni. At that stage, raw CAIMAN imagery from locations such as Ivujivik, Quaqtaq, and Umiujaq in Nunavik, Quebec, was available, and there was a need for a rapid method to generate segmentation masks for five surface classes: ice, sky, land, slush, and open water **(example in Figure 3.1)**.

We used **PixelAnnotationTool** [19], a simple open-source desktop utility, to manually draw class masks frame by frame. During the internship, we labeled approximately **500 images**. These early annotations were used to test ensemble segmentation models, including U-Net, PSPNet, and Mask R-CNN.

At that stage, the focus was purely on model experimentation. The dataset itself was relatively small, but it made the challenge visible, especially in the case of slush, which rarely forms a clearly defined boundary.

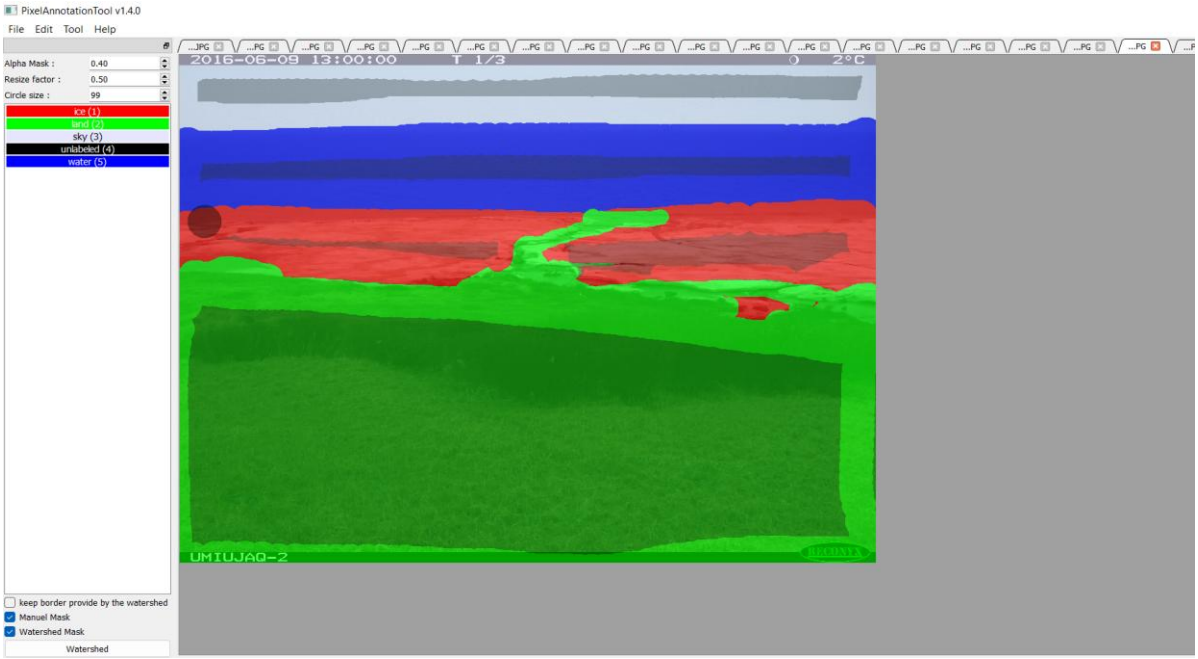

***Figure 3.1**: Example annotation from Version 1 using PixelAnnotationTool, with five labeled surface classes.*

Summary Insight: This version served as a foundational prototype for early model experimentation, helping to surface the segmentation challenges specific to slush boundaries. However, its limited scale and coarse annotations made it unsuitable for more systematic training or evaluation.

3.2 Version 2 — Roboflow-Assisted Annotation and Augmentations

By early 2024, a second version of the dataset was developed using Roboflow's contour-based annotation tool. This version focused on a 15-day frost onset window and included approximately 4000 unlabelled frames that best captured seasonal ice formation patterns.

Roboflow provided a faster interface for polygon-based masks (see **Figure 3.2**) and automatic contour suggestions, making large-batch annotation easier and allowing me to standardize labels across different lighting and weather conditions.

Although the 15-day frost onset window was chosen to capture transitional dynamics in ice formation, this selection introduced a temporal bias into the dataset. Frames from this period are visually distinct and often rich in mixed-class boundaries; however, they do not fully reflect the seasonal progression. As a result, surface classes such as entirely ice or open water during stable freeze/melt periods are underrepresented.

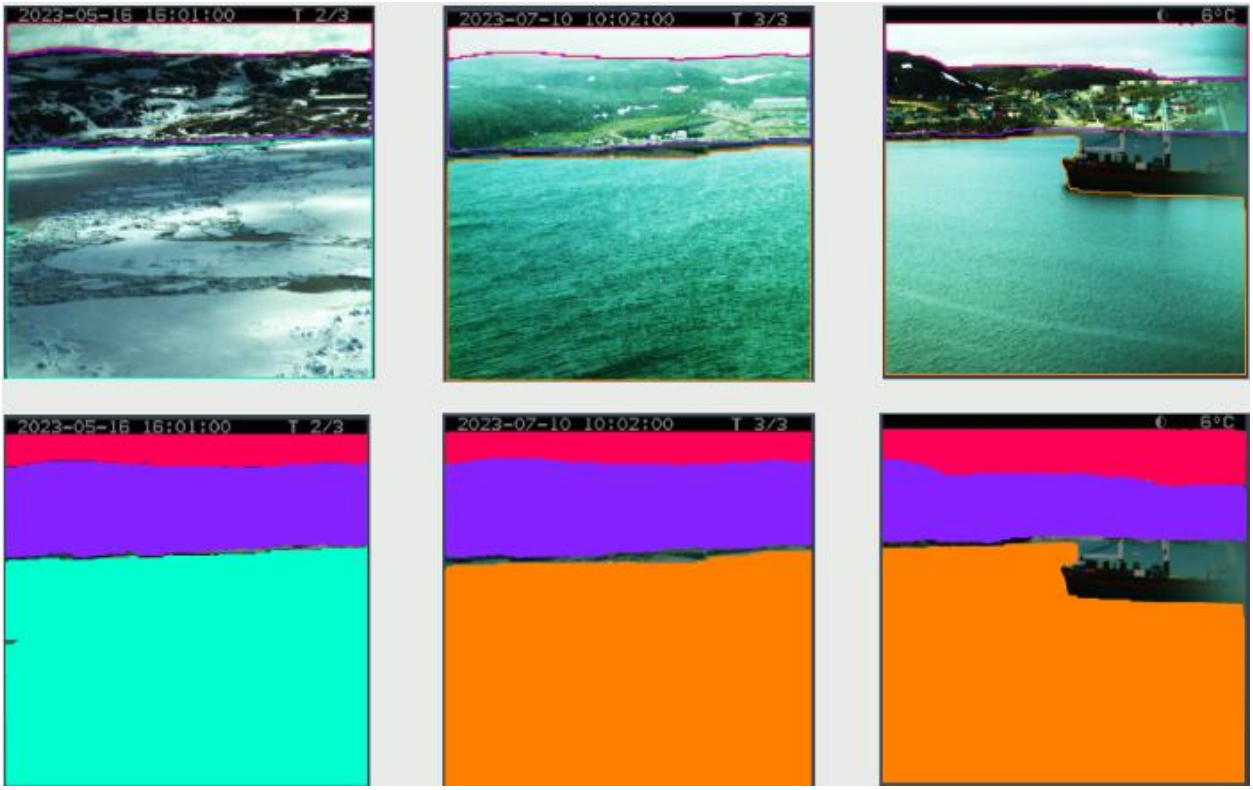

***Figure 3.2**: Roboflow-based polygon annotations created during Version 2 of the dataset.*

Summary Insight: This version prioritized scale and consistency, enabling the first full training experiments with DeepLabv3+. While faster to annotate, it occasionally introduced spatial ambiguity due to automated contouring in low-contrast areas.

3.3 Version 3 — FastSAM-Based Annotation Pipeline

As the project scaled, we developed a lightweight, open-source annotation tool built on FastSAM [30]. FastSAM (Fast Segment Anything Model) is an optimized and accelerated variant of Meta's Segment Anything Model (SAM), designed for high-speed object segmentation with low computational overhead. SAM performs image annotation by generating high-quality segmentation masks based on user prompts (such as points, bounding boxes, or masks) or in a fully automatic mode. It employs a transformer-based architecture comprising an image encoder, a prompt encoder, and a mask decoder to generate precise pixel-level masks for any object in an image. SAM supports both interactive annotation and zero-shot segmentation of all visible objects, making it a powerful tool for labeling large-scale datasets. **Figure 3.3** shows a typical mask set. This tool allowed me to:

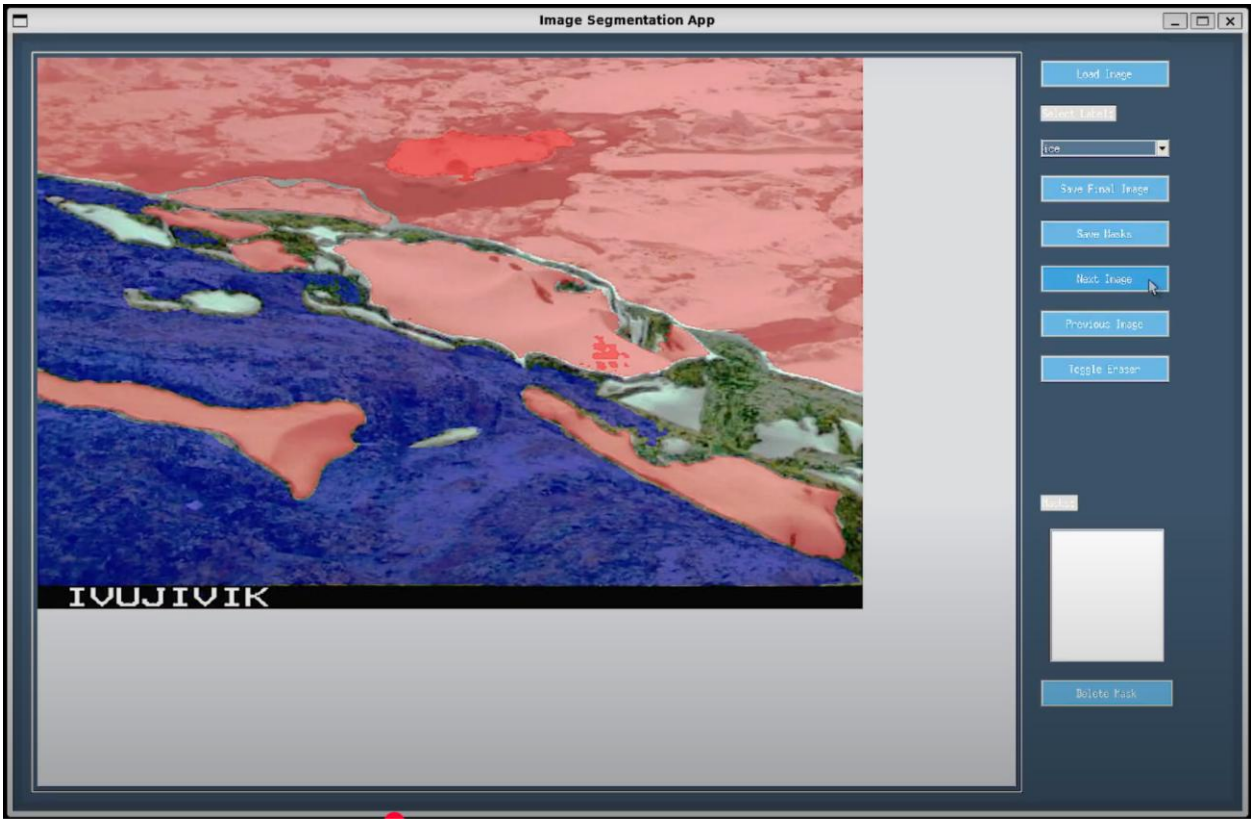

***Figure 3.3**: FastSAM-generated region proposals and edited masks from Version 3 workflow.*

- Automatically generate region proposals from CAIMAN images.
- Select and relabel relevant segments.
- Export masks in **COCO JSON** or **PNG** format.

The FastSAM-based workflow replaced Roboflow for most later annotations. It gave me complete control over mask editing and allowed me to annotate offline, without compromising precision, especially in frames where separating slush and water was difficult.

Summary Insight: This version enabled efficient annotation over ample time spans using promptable segmentation. It introduced the possibility of structured post-processing, though additional filtering was required to correct for over-segmentation and misalignment in texture-rich regions.

3.4 Dataset Summary

The dataset evolved across three major versions, each refining the structure, preprocessing logic, and usability for model training. The table below summarizes the changes across each stage.

Version	Image Count	Preprocessing & Augmentations	Issues Identified
v1 PixelAnnotationTool-Based	500 images	Auto-Orient- Histogram Equalization- Filter out null masks	Image dimensions are too large for most models (very high memory cost); the dataset was heavily skewed toward **Land** and **Water**, with very few **Slush** regions
v2 Roboflow-Assisted	23,330 images	Auto-Orient- Resize: 640×640- Tiling: 2×2 grid-Histogram Equalization- Filter null annotations	Improved performance, but class location became overfit — models started associating classes with fixed spatial regions.
v3 FastSAM-Based	23,330 images	Auto-Orient- Resize: 512×512- Tiling: 2×2- Histogram Equalization- Filter null annotations	Final dataset structure. The aspect ratio was better aligned with model expectations, which helped reduce positional bias across tiles.

Table 3.1*: Summary of dataset versions: preprocessing pipelines and observed effects on model behavior.*

In total, **~8,000 high-quality images** from these versions were selected and cleaned for training and evaluation across different pipeline stages. Earlier versions were retained for model benchmarking and ablation tests.

4. INITIAL MODELING ATTEMPTS

Existing machine learning-based segmentation methods and prior research served as both reference points and catalysts throughout the early stages of development. As challenges arose—such as structural ambiguity and weak class separation—each limitation guided iterative refinements to the segmentation model's design.

4.1 Segmentation Models Used in This Work

This section outlines the segmentation models explored and implemented throughout this project. Each model was chosen for specific capabilities, such as robustness to noisy data, edge sensitivity, or computational efficiency. Some were combined into ensembles; others were integrated into hierarchical pipelines. These models form the foundational building blocks of the CAIMAN segmentation framework.

4.1.1 U-Net

U-shaped encoder–decoder models such as U-Net [2] were a natural starting point, since they are widely used for segmentation. U-Net itself, originally designed for medical images, works by compressing the input to capture context and then expanding it to make pixel-level predictions. One of its strengths lies in skip connections, which preserve fine details and helped in our early experiments with smaller features such as thin ice strips or slush patches. In an initial ensemble with PSPNet and Mask R-CNN, U-Net was the most stable on mid-sized, high-contrast regions.

However, its **core mechanism — repeated downsampling and upsampling — introduced two limitations that proved challenging for this project**:

- **Loss of spatial fidelity at boundaries.** Every compression step discarded fine structure, and although skip connections recovered some detail, subtle features like slush transitions or thin ice strips were often blurred or fragmented.
- **Runtime and scalability concerns.** Reconstructing full-resolution predictions after multiple downsampling passes was computationally heavy. On 2K Arctic images, this quickly became impractical compared to modular region-based processing.

Because of these fundamental constraints, U-Net and similar designs produced acceptable results only in mid-sized regions, but consistently struggled with large, ambiguous, or spatially dispersed structures. This made them less aligned with the goals of the pipeline, which emphasizes adaptive partitioning and interpretable region-level reasoning.

For the same reason, more complex U-shaped variants such as **U-KAN [33]** and **U²-Net [34]** were not pursued further. While these introduce refinements, they inherit the same encoder–decoder trade-offs and would have compounded runtime issues on high-resolution Arctic imagery. With larger curated datasets and targeted training, these models could regain relevance, but within this scope they were ultimately set aside.

.

4.1.2 PSPNet

PSPNet (Pyramid Scene Parsing Network) [5] was added to the ensemble for its ability to incorporate global context into the segmentation process. Unlike U-Net, which focuses heavily on preserving local details, PSPNet processes the image at multiple spatial scales simultaneously. This makes it particularly effective in scenes where understanding the broader layout helps refine local predictions — for example, recognizing a slushy patch as part of a water boundary based on its surroundings.

In practice, we observed that PSPNet generated more coherent segmentations in regions with gradual texture transitions. It performed well on large ice sheets and open water, exhibiting a reduced tendency to unnecessarily fragment regions. This was helpful in the Arctic scenes, where visual cues like glare or shadows could make boundaries ambiguous at a local level.

However, the model often blurred smaller or more sharply defined features. It tended to "smooth over" fine-grained edges, making it less effective in capturing narrow or isolated elements, such as fragmented slush or hairline cracks. Within the ensemble, it played a complementary role to U-Net, where U-Net offered crisp detail, and PSPNet brought structure and global reasoning.

Although it was not used beyond the ensemble stage, PSPNet provided a useful counterbalance to more localized models, helping to highlight the tradeoff between global context and local precision.

4.1.3 Mask R-CNN

Mask R-CNN [1] was included in the ensemble for its instance-level segmentation capabilities. Unlike U-Net or PSPNet, which assign a class label to every pixel across the frame, Mask R-CNN detects individual object instances and produces masks for each. This property made it an appealing choice in the early stages of the project, particularly for scenes where isolated blobs of ice or slush needed to be identified independently from their surroundings.

In theory, the model’s two-stage architecture — object detection followed by mask refinement — offered a structured way to extract discrete entities. However, in practice, it struggled with the abstract and amorphous nature of surface classes [20] in the CAIMAN dataset.

Since classes like “slush” or “thin ice” do not always form well-bounded instances, the model frequently failed to trigger detections or merged regions incorrectly. It was particularly poor at capturing soft transitions and edge zones, which were central to this task.

Despite these shortcomings, Mask R-CNN brought value during ensemble experimentation. It served as a contrast to the fully convolutional models, showing how standard object detection pipelines underperform when the segmentation problem is not object-centric. This reinforced the need to move away from instance segmentation altogether and toward region-based methods that reason at the mid-levels.

4.1.4 DeepLab v3+

DeepLabv3+ [4] became the backbone of the later stages of this project due to its superior ability to capture both local and global semantic information. Its encoder-decoder structure, combined

with atrous (dilated) convolution, enabled it to extract multi-scale features without compromising resolution — a critical requirement for detecting thin or fragmented features, such as slush trails or transparent ice layers.

Unlike the earlier models in the ensemble, DeepLabv3+ generalizes better to new scenes, particularly when trained on curated, cleaned datasets. It proved especially reliable when paired with structured region proposals — such as those from SAM or SLIC — where the classification target is a coherent region instead of raw pixels. Its flexibility and compatibility with various backbones made it ideal for iterative refinement and testing under different constraints.

To better understand its evolution in the pipeline, the table below summarizes the configurations used during development:

Variant	Backbone	Purpose in Pipeline	Notes
DeepLabv3+ (ResNet-50)	ResNet-50 [31]	Fast baseline for testing pre-segmentation inputs	Lightweight; used during CPU-compatible pipeline design
DeepLabv3+ (ResNet-101)	ResNet-101 [31]	Final classification model in the GPU pipeline	Higher capacity; better at detecting fine texture differences
DeepLabv3 (Full-frame)	ResNet-50	Early experiments on raw pixel-based segmentation	Replaced later by region-wise approaches due to spatial bias issues

Table 4.1: *DeepLabv3+ Variants Used in This Work*

4.1.5 Segment Anything Model (SAM)

SAM [6] is a general-purpose segmentation engine developed by Meta AI. Instead of classifying pixels directly, SAM predicts object masks by interpreting spatial prompts — like points, boxes, or coarse labels — using a large Vision Transformer (ViT). Internally, it builds a high-dimensional embedding of the image, then decodes it into mask proposals that can respond flexibly to prompts or be generated automatically.

In this project, SAM was used in **automatic mask generation mode**, with no prompts provided. Instead, SAM over-segments the image into a dense set of instance masks, each representing a visually coherent region. These masks are not semantic classes, but rather spatial groupings — regions that share a common shape, texture, or boundary.

This was especially useful for Arctic images where boundaries are ambiguous and pixelwise models struggle with noise. SAM provided region proposals that respected edges and texture transitions, even in the presence of glare, slush, or mixed surfaces. These proposals became the inputs to later classification stages (e.g., DeepLab), enabling a modular approach: first discover meaningful regions, then label them.

Variant	Base Architecture	Role in Pipeline	Notes
SAM (ViT-H) [6]	Vision Transformer (Large)	Used for final GPU-based pipeline	High-quality masks with excellent boundary fidelity
FastSAM [30]	Lightweight variant	Used for early experiments and data annotator	Faster but noisier; good for prototyping

***Table 4.2**: Segment Anything Model variants explored during experimentation.*

By separating region discovery from classification, SAM enabled a form of **region-based reasoning**: instead of segmenting the whole image in one pass, the pipeline operated on discrete, semantically meaningful chunks — something traditional models struggled to produce on their own.

4.2 Ensemble Learning Attempt

The first modeling attempt utilized a combination of three well-known segmentation models: U-Net, PSPNet, and Mask R-CNN. These models were initially developed for medical and object-based segmentation tasks. They were chosen because they each specialize in a different strength: smooth shapes, broad scene context, or sharp object boundaries.

U-Net produced relatively clean segmentations for broad surfaces, such as land or open water. However, it struggled with smaller features and often merged slush with surrounding classes.

PSPNet effectively captured the overall layout, particularly in scenes with strong lighting contrasts. However, it tended to blur the details, and often misclassified slush regions that had subtle transitions.

Mask R-CNN performed best at outlining distinct structures. It sometimes drew crisp edges around ice floes or rocks, but failed when surfaces blended gradually, as is often the case with slush.

To combine their outputs, a simple majority-vote strategy based on intersection-over-union (IoU) thresholds was applied, aiming to produce cleaner and more stable masks than any single model could achieve independently [8][12][9][14][21].

These ensemble tests were conducted on the earliest version of the dataset, generated using the PixelAnnotationTool during the initial internship phase (Section 3.1). At the time, ~500 images were manually annotated across five surface classes.

While this dataset provided a necessary starting point, it lacked spatial consistency and contained frequent misclassifications of slush, which negatively impacted ensemble reliability. Nonetheless, it served as a valuable diagnostic tool for exposing model disagreement and boundary ambiguity.

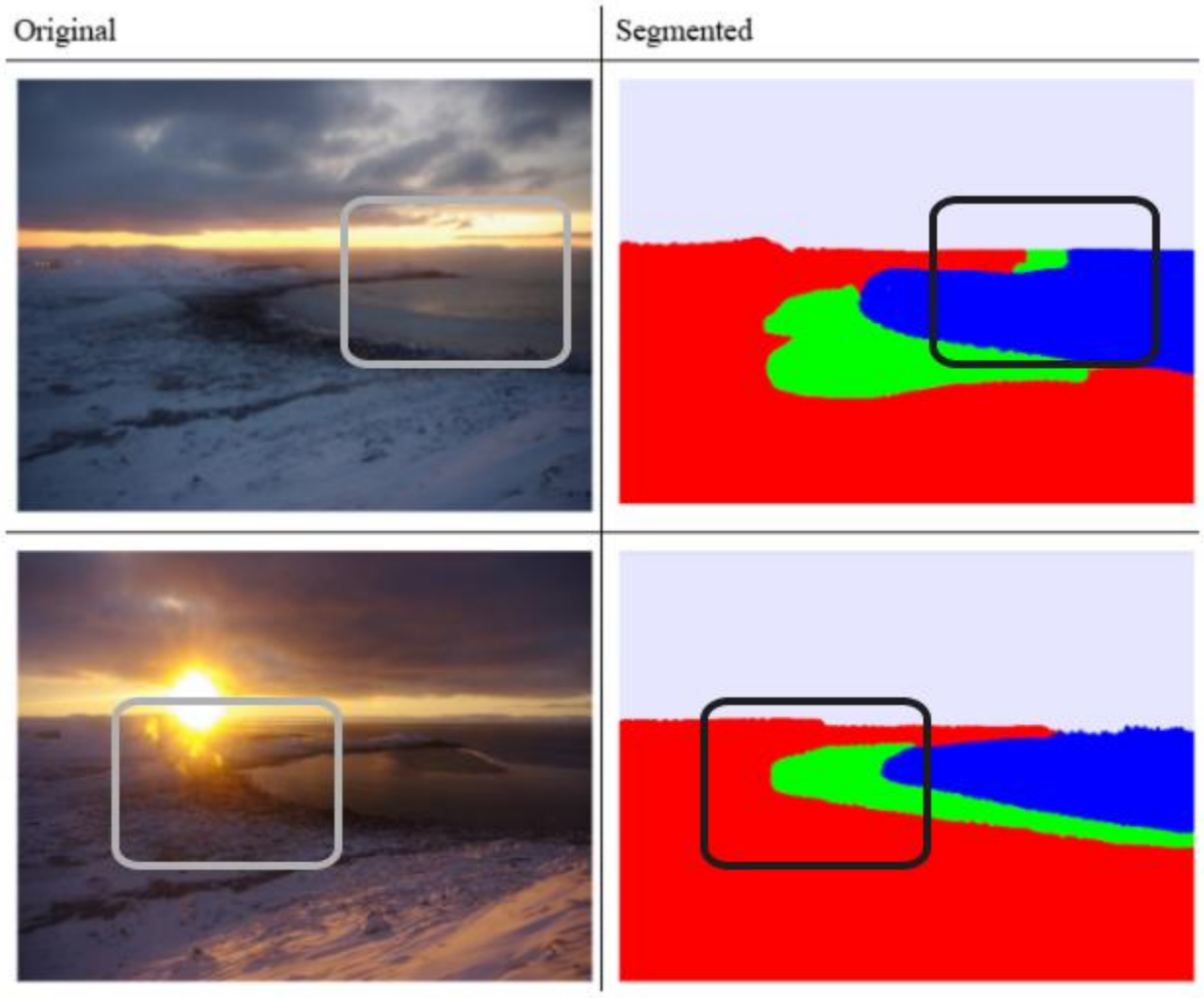

***Figure 4.1**: Ensemble model failures. Slush (green) bleeds into water (blue), and glare zones (horizon, sun reflection) cause fragmented boundaries. Boxes mark typical misclassified regions.*

The segmentation maps had apparent flaws (see **Figure 4.1** for a typical failure):

- Labels bled across boundaries
- Slush was rarely appropriately identified.
- The output felt like the average of three different guesses
- The average also failed to stay consistent across evaluations

Inter-run consistency also suffered, and the ensemble introduced unnecessary computational overhead without offering clear gains in interpretability or robustness. While useful as a diagnostic tool, the ensemble was not suitable for deployment.

4.3 General Semantic Segmentation Models

After the ensemble pipeline, the next stage focused on training a single, well-established segmentation model: **DeepLabv3+**. This architecture is commonly used in real-world scene understanding tasks, especially when precise boundaries and stable predictions are needed.

This version featured improved annotations, encompassing ~4,000 images, and addressed many of the inconsistencies present in the original training set — namely, **Version 1 of the dataset created using PixelAnnotationTool (see Section** 3.1). The enhanced data quality had a noticeable impact—DeepLab, even without fine-tuning, outperformed the earlier ensemble. It produced tighter masks, achieved higher class-wise recall, and delivered more consistent predictions across similar frames.

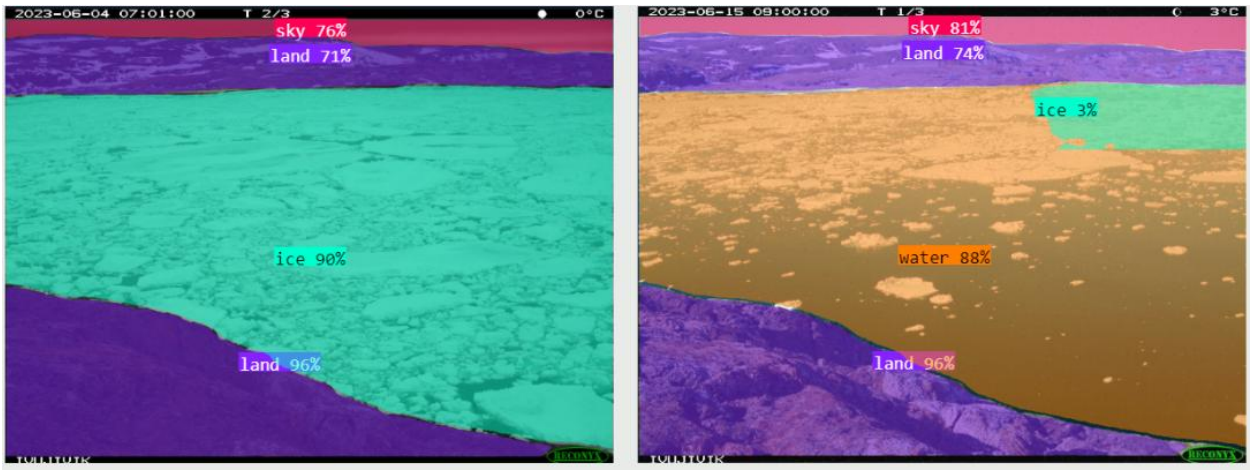

Figure 4.2: *Example predictions from DeepLab v3+ showing dominant class bias and failure to separate ambiguous textures like slush or ice fragments.*

However, as illustrated in **Figure 4.2**, the model continued to exhibit notable shortcomings, including:

- **Minor object confusion**: Slush and ice chunks that appeared as isolated blobs were either missed entirely or wrongly merged with water.
- **Overconfidence in dominant classes**: As shown in the visual predictions (**Figure 4.2**), the model strongly favored large, uniform areas, such as land or water, assigning them almost the entire frame and ignoring edge transitions.
- **Spatial bias**: Since Roboflow annotations were somewhat positional (e.g., land always at the bottom), the model learned spatial shortcuts rather than actual texture differences, as evident in **Figure 4.3**.

Despite its flaws, DeepLabv3+ performed more consistently than the earlier ensemble [23]. With a cleaner dataset, it produced tighter masks and clearer class boundaries. This showed that a single, well-trained model can offer better generalization than combining multiple weaker ones. At the same time, it emphasized the need for models that understand spatial structure.

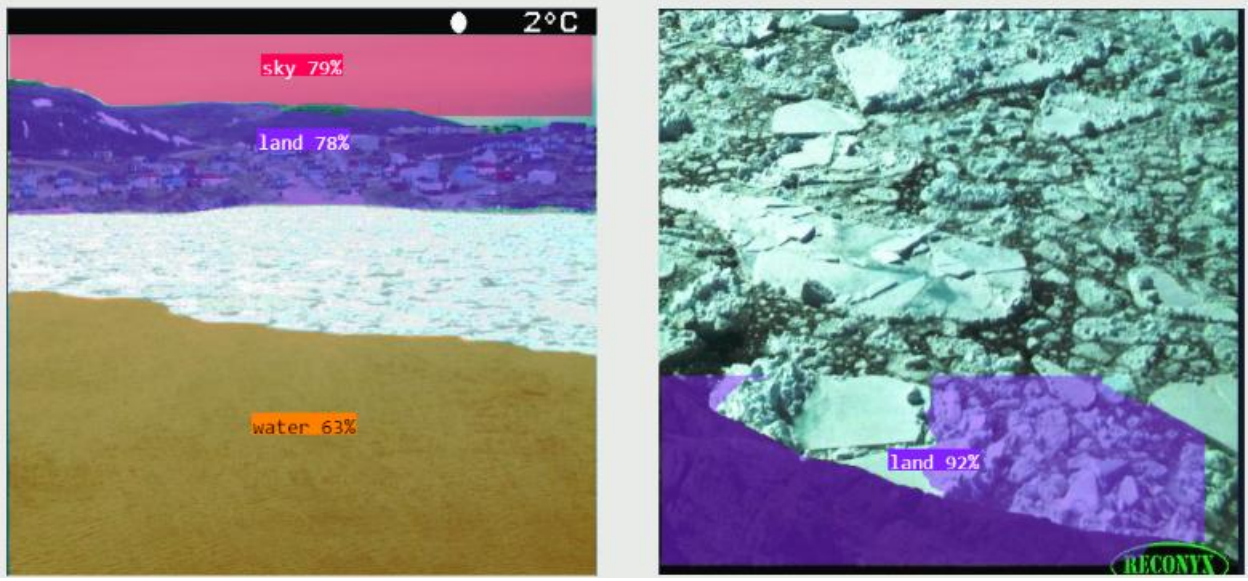

Figure 4.3: *Further illustration of spatial overfitting in DeepLab v3+, where predictions depend heavily on frame layout rather than local features.*

This failure set the stage for the next phase — breaking the image into spatial windows and reasoning locally, instead of forcing global forecasts from a single pass.

4.4 Tiled Inference with Rolling Windows

To reduce the spatial bias introduced by full-frame predictions, a tiled inference approach was implemented using **DeepLabv3+** [22]. Instead of passing the entire image through the model at once, the frame was divided into smaller overlapping windows. Each tile was processed independently, and the outputs were stitched together to produce a full-resolution segmentation map.

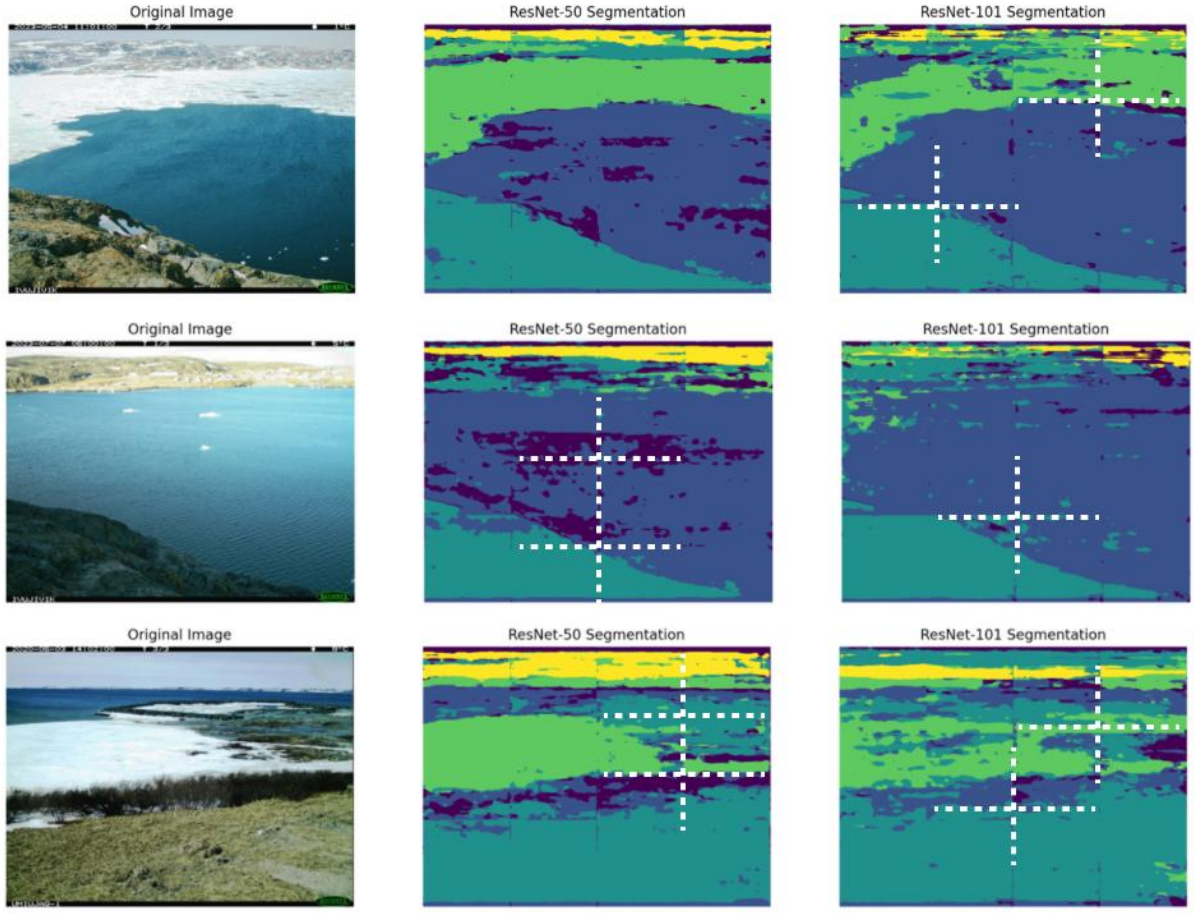

Figure 4.4: *DeepLabv3+ predictions on tiled input windows, showing positional improvements but noticeable inconsistencies across tile boundaries.*

This method helped in two ways:

- It **reduced positional overfitting** — each tile was treated independently, forcing the model to rely more on texture and pattern than on frame layout.
- It preserved **local context**, making small features, such as slush patches, more visible and improving prediction granularity in areas with complex boundaries.

However, the obtained improvements were modest. The final stitched masks frequently exhibited visible seams and inconsistent edges along tile boundaries, as illustrated in **Figure 4.4**. In some instances, segmentation failed entirely in the overlap regions. While the tiled approach countered some of the spatial shortcutting seen in earlier predictions, it did not allow the model to reason about continuity, shape, or connectedness. The result was often closer to a heatmap than a coherent surface map.

This limitation highlighted the need to move beyond pixel-level processing and toward region-based reasoning that could understand the relationships between different regions.

4.5 Panoptic Exploration & SAM Integration

Following the rolling window approach, the next stage explored models capable of reasoning over distinct regions and surfaces — rather than treating the image as a continuous field [32]. This led to the use of **panoptic segmentation** [26], a hybrid task that combines **semantic segmentation** (assigning a class label to every pixel) with **instance segmentation** (distinguishing between individual objects of the same class).

In theory, this enables models to understand both *what* is present in a scene and *how many* of each type exist—a critical capability for parsing fragmented and transitional ice surfaces.

At this stage, an emerging interest in vision transformers and foundation models led to the integration of the SAM by Meta. SAM is designed to oversegment scenes by default, generating fine-grained region masks that align closely with object boundaries, even in complex or ambiguous textures, using simple prompts like a point, as shown in **Figure 4.5**.

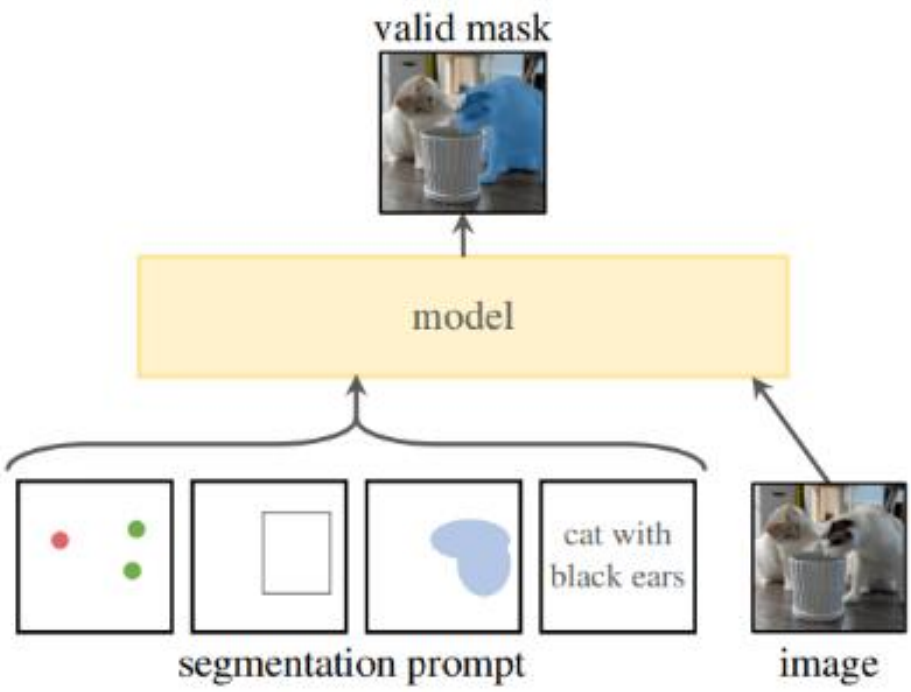

Figure 4.5: *Working principle of the SAM, which accepts various prompts and returns over-segmented regions.*

This phase marked a shift toward more structured segmentation. The SAM model, designed to over-segment images by default, was used to generate high-resolution instance masks. These masks were then passed to DeepLab for semantic classification, forming a hybrid pipeline.

This combination yielded several key improvements:

- Produced cleaner object boundaries and improved class-wise confidence
- Captured ambiguous textures (such as slush) more effectively than standalone DeepLab models

Although the results were not uniformly consistent, **Figure 4.6** demonstrates how this pairing began to uncover the spatial logic of the scene. The segmentation output started to reflect relationships between surface classes — how they cluster, interact, transition, and fragment across the frame.

This marked a conceptual shift in the project's direction. Rather than simply identifying class presence, the pipeline began to suggest how surfaces break apart and where transition zones might lie. These insights laid the groundwork for transitioning toward region-based reasoning and structured segmentation.

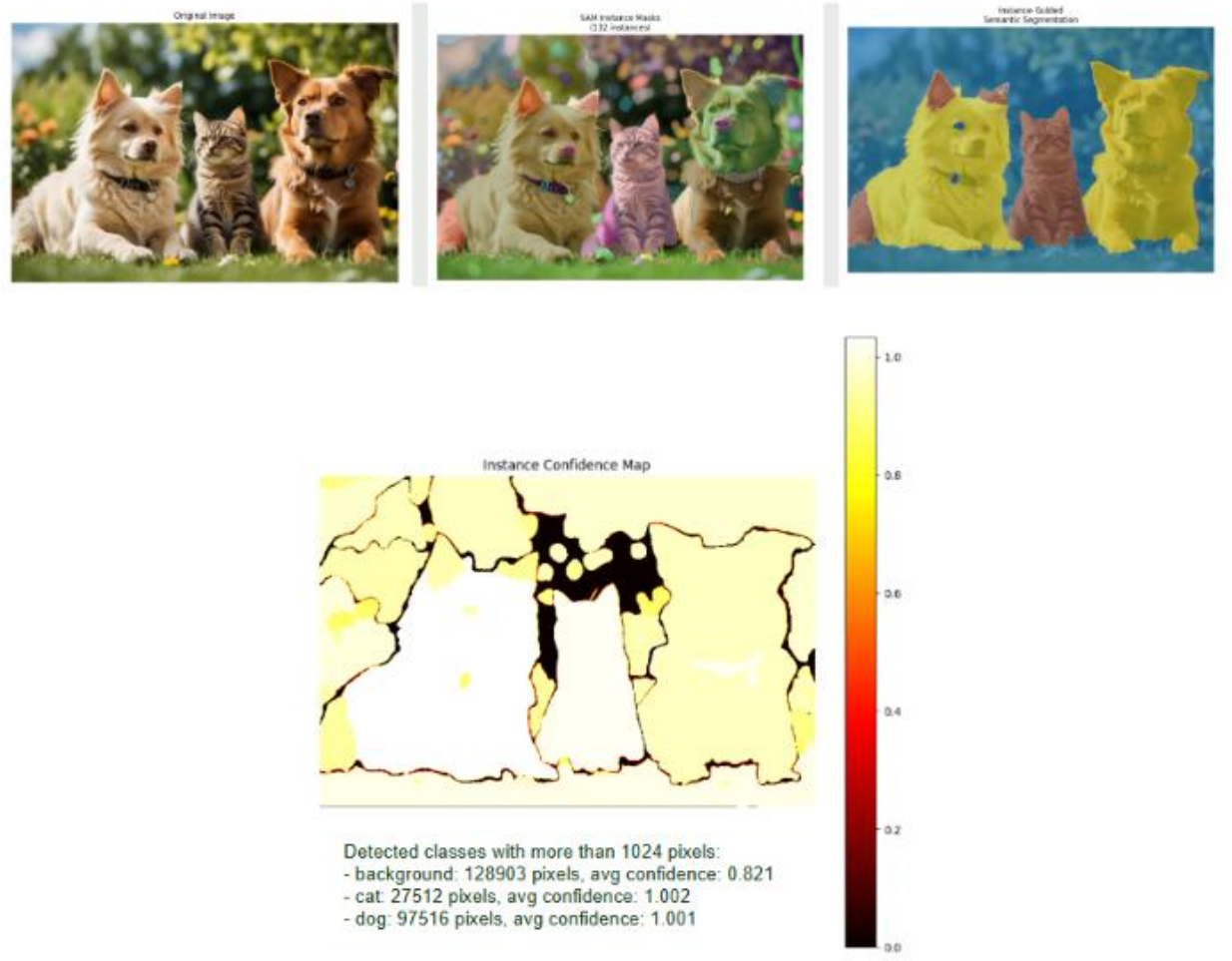

Figure 4.6: *SAM + DeepLab hybrid predictions showing cleaner boundaries and class confidence, with regions better aligned to physical transitions.*

4.6 Diagnostic Learnings from Experimental Stages

Most models excelled in certain areas but fell short in others, and none performed well enough to be relied upon independently.

Method / Phase	Lesson Learned	What I Needed
Ensemble of U-Net, PSPNet, Mask R-CNN	Combining weak models does not address fundamental data problems; the outputs lack cohesion.	A stronger, more resilient model under insufficient data
DeepLabv3 / DeepLabv3+ (Full Image)	Pixel-level segmentation is fast, but it fails to generalize or detect small features.	Generalization beyond positional bias
Rolling Window with DeepLab	Local prediction helps mitigate bias, but tile edges cause inconsistency, and it lacks a structured approach.	Local focus without losing global context
SAM + DeepLab Hybrid	Starting from meaningful regions drastically improves edge confidence and speed.	Grouped regions before classification

***Table 4.3**: Key observations and takeaways from successive modeling approaches.*

Initially, as shown in Table 4.3, we believed the problem was solely with the dataset. Eventually, we realized it was not about one fix. It was about how each part of the process fit together — or did not.

Throughout this process, everything ran on a local machine — an RTX 4070 Super with 12GB of VRAM. It turned out to be a great chance to push that GPU to its limits. Google Colab felt too restrictive, especially for iterative experiments. Having everything local made it easier to try things quickly by changing parameters and tuning each part of the system without hitting session limits or upload caps.

Instead of throwing more compute or complexity at the problem, we started working with what the models could do. That meant leaning into structural enhancements — pre-segmentation, merging, masking, and careful post-processing.

5. THE HYBRID PIPELINE: DESIGN CHOICES

5.1 Goals and Constraints

By the time we reached the final phase of the project, it was clear that no single model could adequately handle the ambiguity of the CAIMAN dataset. There were too many edge cases — glare, slush blending into water, thin ice misclassified as land — for any single black-box solution to hold up.

We focused on building a modular pipeline that could adapt to different constraints. In doing so, we created two variants of the final system:

- A GPU-optimized pipeline, built for experimentation and detailed prediction
- A CPU-friendly version, designed to run on machines without a dedicated GPU

Both versions shared the same **core philosophy: break the image into regions, reason locally, then label globally**.

The key constraints that shaped the pipeline were:

- **Structural ambiguity**: Labels often lacked sharp edges; class boundaries were soft or undefined.
- **Annotation noise**: The ground truth data was imperfect, especially in slush regions.
- **Speed and modularity**: The system needed to be fast, testable, and extensible.
- **Hardware limitations**: While development was conducted on a 12GB RTX 4070 Super, we also wanted a version that could run entirely on the CPU.

This section outlines the complete pipeline architecture and explains how each part was selected or developed to meet these constraints.

5.2 Implementation Setup

All experiments were conducted locally on a dedicated workstation. The system specifications are as follows:

- **Processor:** Intel® Core™ i7-14700KF, 3.4 GHz base clock
- **Memory:** 32 GB RAM
- **GPU:** NVIDIA® GeForce RTX™ 4070 Super Ti, 16 GB VRAM

This configuration provided sufficient resources to handle both GPU-based (SAM) and CPU-based (SLIC) variants of the pipeline without requiring distributed computing or cloud resources.

The software environment relied on **PyTorch** as the core deep learning framework, complemented by standard scientific libraries. Key dependencies include:

- **Torchvision** for the DeepLabv3-ResNet101 backbone and preprocessing
- **Segment Anything (SAM, ViT-H)** for over-segmentation proposals
- **scikit-image (SLIC)** for CPU-based superpixel generation
- **OpenCV** and **Pillow** for image handling
- **NetworkX** for region adjacency graph construction
- **NumPy** and **Matplotlib** for numerical computation and visualization

This design preserved flexibility, allowing individual components to be swapped (e.g., SAM vs. SLIC for proposals) while maintaining consistent evaluation. Unless otherwise stated, the GPU-optimized SAM variant was used for the results presented in Chapter 6. The CPU-based SLIC pipeline was reserved for comparative experiments and testing with lower resources.

5.3 Final Pipeline Architecture

The segmentation system is implemented as a modular pipeline that operates on compact image regions rather than raw pixels. Instead of relying on a single model, the task is divided into five distinct stages: region generation, adjacency graph construction, region merging, region-wise classification, and final mask assembly.

Two parallel implementations of the pipeline were developed:

- **GPU-based architecture**: This version uses SAM-ViT for high-resolution region proposals. The segments are merged using a graph-based approach, and final classification is performed using DeepLabv3+. It is optimized for both accuracy and flexibility, making it suitable for experimentation and large-scale inference.
- **CPU-compatible architecture**: This version replaces SAM with SLIC (Simple Linear Iterative Clustering) [15] superpixels for faster, lightweight region generation. It uses a simplified merging strategy and the same DeepLabv3+ classifier. This version sacrifices fine-grained precision for speed and accessibility on resource-constrained systems.

Despite these differences, both pipelines follow the same overall logic. Each component is implemented independently, allowing for easy replacement or tuning without disrupting the rest of the pipeline.

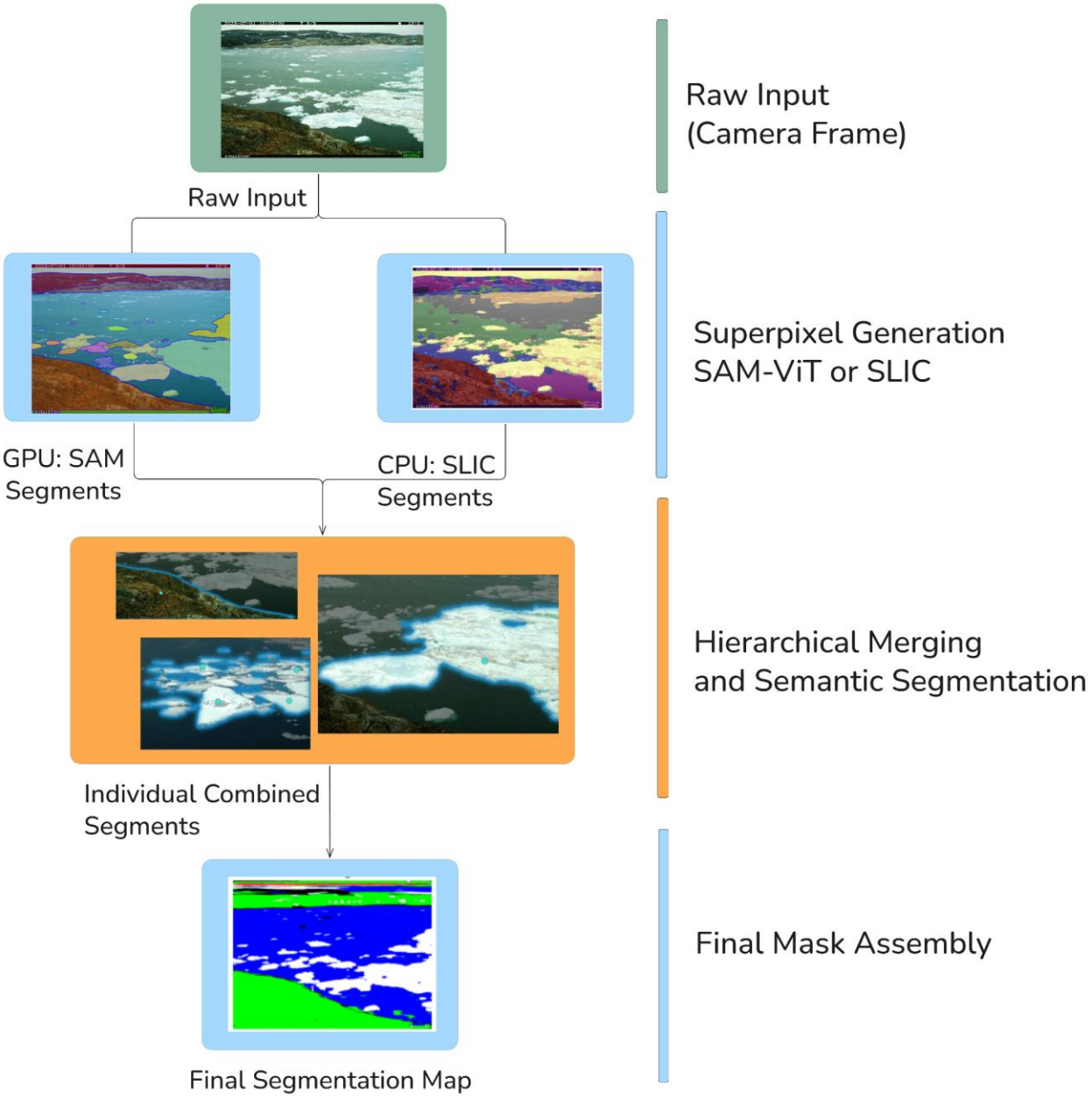

Figure 5.1: *Visual progression of an image through the segmentation pipeline. The system branches at the superpixel generation stage, using either SAM-ViT (GPU) or SLIC (CPU), but converges through shared graph-based merging and region-wise classification to produce the final semantic mask.*

5.4 Superpixel Generation and Preprocessing

Before classification or graph reasoning can occur, the image must be decomposed into coherent regions. These regions, known as superpixels, are spatially contiguous groups of pixels that share

low-level visual properties, such as color, texture, or intensity. Superpixels reduce image complexity while preserving local structure, providing a mid-level representation that enables the pipeline to reason about regions rather than individual pixels.

Importantly, superpixels are not used for prediction. Instead, they serve as the fundamental units for downstream operations, including graph construction, merging, and region-wise classification [25]. Each superpixel corresponds to a node in the region adjacency graph.

To generate these regions, we used two methods, depending on the computational environment. Each produces qualitatively different outputs and supports a distinct version of the pipeline.

5.4.1 SAM-ViT (GPU Pipeline)

The GPU-accelerated version of the pipeline uses the SAM with a ViT (Vision Transformer) backbone to produce fine-grained, high-confidence masks. ViT is a deep learning architecture that applies the Transformer model, originally designed for natural language processing, to image data. The resulting masks serve as pre-segmented object-like proposals, characterized by sharp boundaries and strong semantic separation.

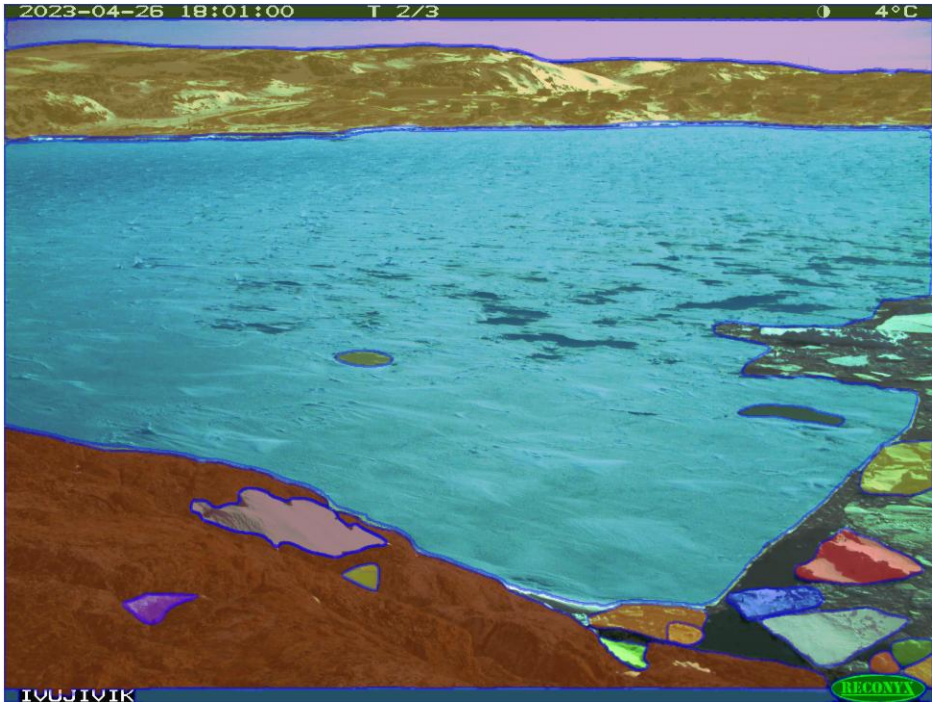

***Figure 5.2**: SAM-ViT-generated region masks showing high-resolution boundary alignment.*

Unlike traditional superpixel algorithms, SAM-ViT segments the image using a vision transformer [16] that encodes spatial and semantic information, yielding ~80–150 binary masks per frame, as

illustrated in **Figure 5.1**. These masks naturally align with object boundaries — including subtle transitions like slush-water zones — making them well-suited for high-ambiguity scenes.

This method requires a GPU with at least 12 GB of VRAM but introduces minimal post-processing overhead — only small, noisy regions are filtered before graph construction.

5.4.2 SLIC (CPU Pipeline)

The CPU-compatible version relies on SLIC [15], a classic algorithm that clusters pixels based on color and spatial proximity. It generates approximately 300 to 500 compact superpixels per image (**Figure 5.2**), but these segments lack semantic awareness—they do not respect object boundaries and may arbitrarily split or merge across real-world structures.

While SLIC is highly efficient and does not require any pretrained weights, its outputs are noisier. Many small or fragmented regions must be cleaned and merged in later stages. Nonetheless, it offers a lightweight fallback that supports deployment on non-GPU systems.

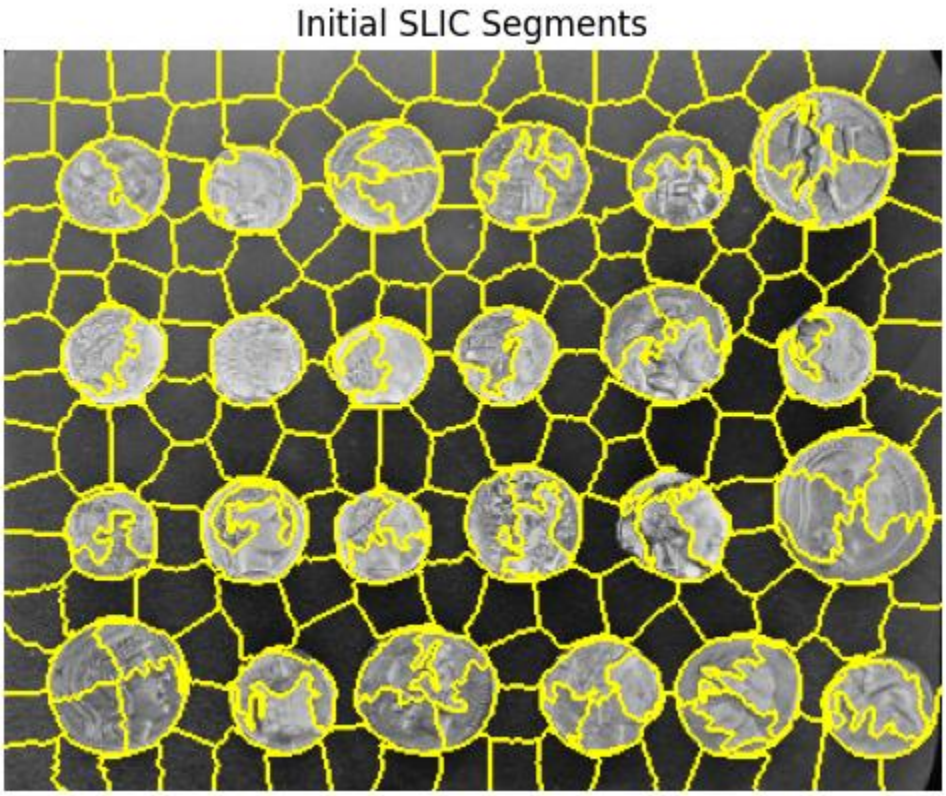

***Figure 5.3**: SLIC superpixel output illustrating compact but less precise region partitioning.*

5.4.3 Summary Table

Table 5.1 compares the two methods used for superpixel generation in terms of algorithmic behavior, quality, and suitability for the pipeline.

Property	**SAM-ViT (GPU Pipeline)**	**SLIC (CPU Pipeline)**
Algorithm Type	Vision Transformer-based segmentation	Spatial–color clustering
Output	Binary masks (80–150 per image)	Compact superpixels (~300–500 per image)
Edge Alignment	High (sensitive to semantic and texture boundaries)	Low (no boundary awareness)
Sensitivity to Visual Cues	Robust to glare, color variance, haze, and transitions	Sensitive only to local color and distance
Requires Training	No (uses pretrained weights)	No
Hardware Requirement	GPU (12GB VRAM recommended)	CPU only
Use Case	High-precision region proposal for complex scenes	Lightweight fallback for low-resource systems
Postprocessing Needs	Minimal (small masks filtered)	Significant (noisy regions filtered and merged later)

***~~Figure~~ Table 5.~~5~~1**: Final modular pipeline results, with overlay highlighting region-wise predictions.*

5.5 Region Adjacency Graph Construction

Once superpixels are generated, the next step is to understand how they relate to each other spatially. Since each region is now a self-contained chunk of the image, we can start comparing them based on measurable properties, such as average brightness, color distribution, or location. This is where the Region Adjacency Graph (RAG) comes in.

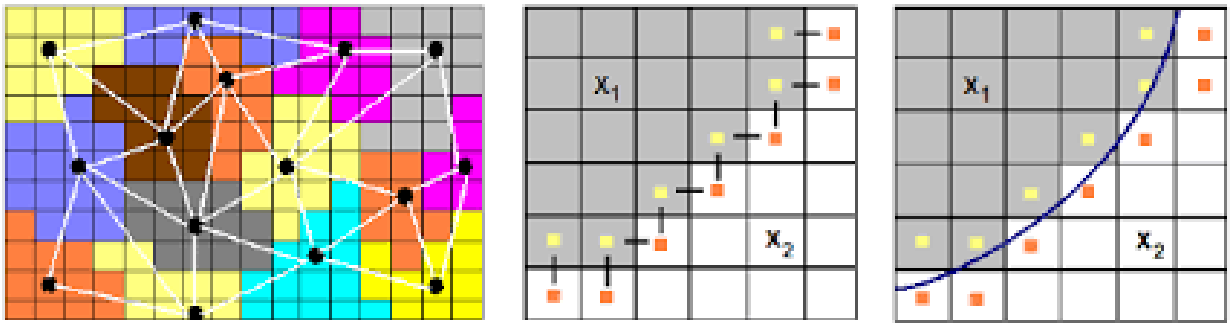

Figure 5.4: *Region adjacency graph (RAG) visualization constructed from superpixels. Each node represents a region, and edges encode spatial adjacency used for merging decisions.*

A Region Adjacency Graph is a structure where:

- **Each node** represents a region (in this case, a superpixel from SAM or SLIC).
- **Each edge** connects two regions that are directly adjacent in the image.
- **Node attributes** store information such as average color, centroid, and area.
- **Edges** signal that two regions can be merged.

This makes the problem easier: instead of trying to process the entire image at once, we now work with a graph where merging decisions can be made based on clear mathematical thresholds, such as color similarity or distance between centroids. **Figure 5.3** illustrates an example RAG constructed from superpixels.

In this implementation, each region is compared to every other nearby area. Adjacency is determined by checking if their binary masks touch, using morphological dilation to detect overlaps. If two masks share a boundary pixel, they are connected by an edge in the graph.

Once the graph is built, each node knows:

- What region does it represent
- Its location (centroid)
- Its average RGB color

- Its size in pixels

This sets the stage for the merging logic: a controlled, rule-based approach to group regions that are visually and spatially similar, rather than relying on raw model predictions to do so upfront.

This stage parallels earlier graph-based segmentation methods, such as Felzenszwalb and Huttenlocher's efficient graph-based approach [25], and aligns with object-based image analysis techniques used in remote sensing, where region adjacency graphs provide a structured basis for stable grouping.

5.6 Feature Engineering for Merging

The purpose of this stage is to determine which regions in the graph can be safely merged into larger units without breaking structural coherence. Rather than relying on a model or embedding space, merging is driven by simple, interpretable features computed for each region [24]. This preserves transparency, eliminates the need for additional training, and enables fine-tuning of merging decisions using explicit thresholds.

Each node in the Region Adjacency Graph stores the following attributes:

- **Binary mask**: the spatial footprint of the region
- **Centroid**: the mean pixel location of the region
- **Average RGB color**: used as a proxy for surface similarity
- **Size**: total number of pixels in the region

Merging is applied only to regions that are already connected in the graph—i.e., spatially adjacent. For each connected pair, two comparisons are made:

- **Color distance**: Euclidean distance between the regions' average RGB values
- **Centroid distance**: Euclidean distance between their spatial centers

If both values fall below fixed thresholds, the regions are merged using a disjoint-set (union–find) structure. The merged result inherits a new combined mask, an updated centroid, an updated color (weighted by area), and a total size.

This approach ensures that only visually and spatially consistent regions are fused. It avoids the risk of over-merging across subtle boundaries, which is especially important in scenes with faint texture transitions or glare artifacts.

Importantly, these features also act as a **common representation space** for integrating region proposals from different sources. SAM, SLIC, and even manually-defined segments all produce various types of regions, but once reduced to centroid and color descriptors, they become comparable. This enabled the pipeline to treat binary masks from SAM and grid-based clusters from SLIC as equivalent units during graph construction and merging.

Bringing these sources into the same vector space was critical. Each method carries different structural assumptions, and no component was designed to operate within the RAG framework alone. By defining region-level similarity solely in terms of shared numeric features, the pipeline successfully integrated previously incompatible modules into a coherent hierarchical system.

5.7 DeepLab Classification and Postprocessing

The final stage in the pipeline assigns semantic labels to the merged regions. For this, we rely on a custom-trained DeepLabv3+ model with a ResNet-101 [31] backbone, which has been fine-tuned over several rounds on the full CAIMAN dataset. Each area is passed independently to the classifier, which returns a class label at the pixel level, as depicted in **Figure 5.4**.

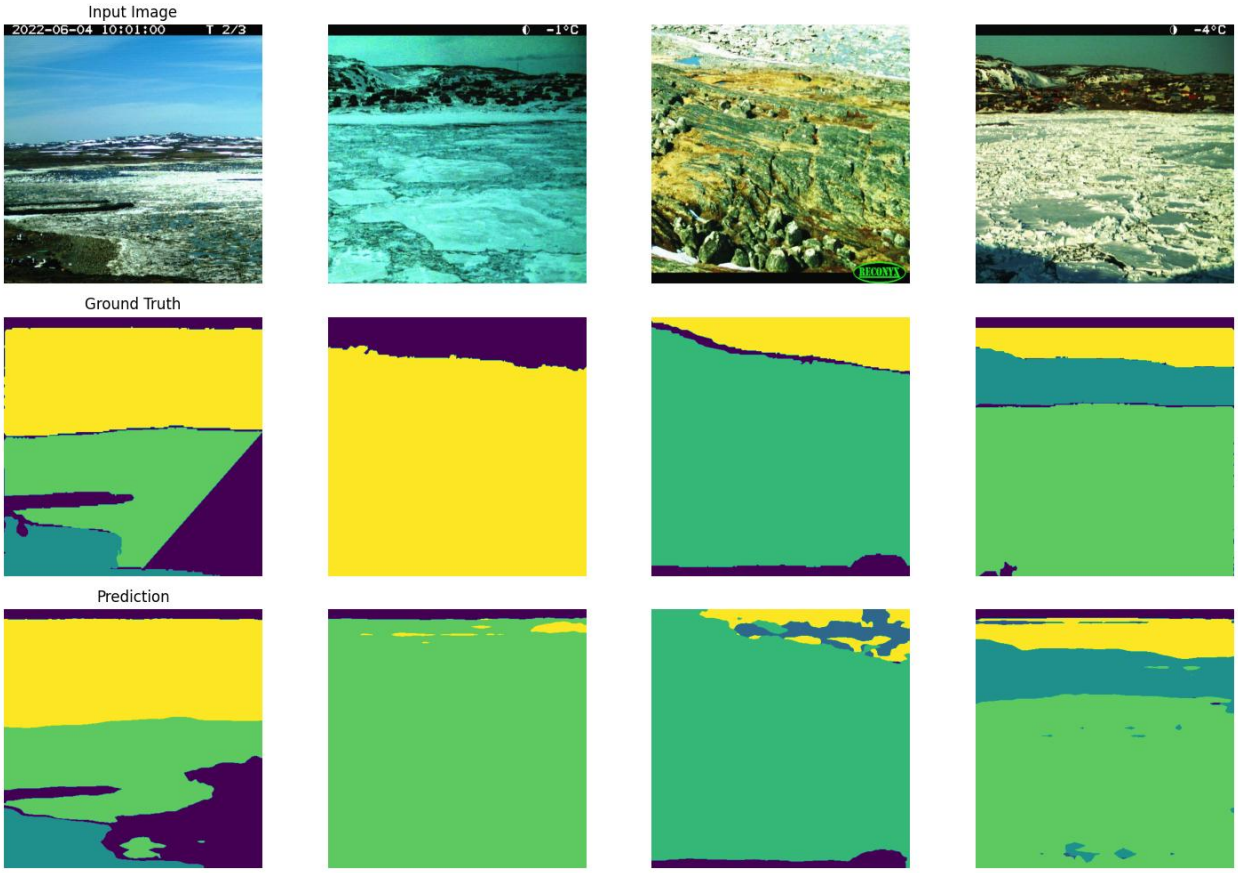

Figure 5.5: *DeepLabv3+ predictions on validation scenes. While the model captures coarse structure, it struggles with fine boundaries and misclassifies slush–water transitions (compare bottom row to ground truth)*

DeepLab was selected for its ability to capture both coarse structure and fine edges using atrous (dilated) convolutions. The model was trained on over 8,000 annotated images, with careful augmentations that included rotation, color jitter, flipping, and downsampling.

Crucially, the model was trained to segment five semantic classes — land, sky, water, ice, and background — using class balancing and a robust cross-entropy loss function with tuned weights.

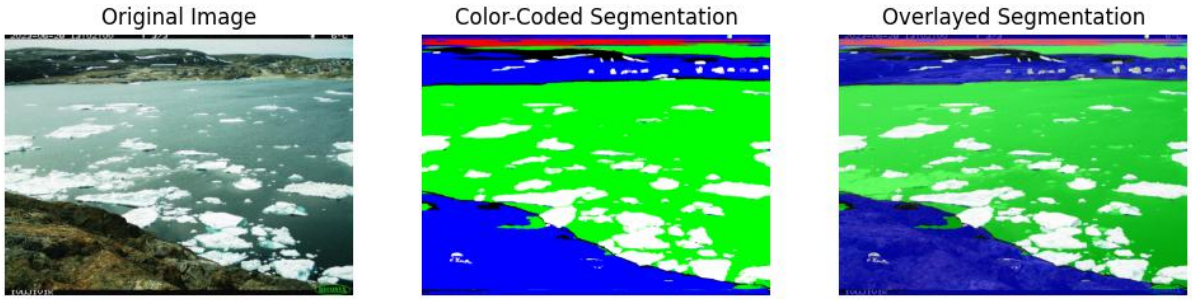

***Figure 5.6**: Final modular pipeline outputs. The modular approach improves region boundaries and preserves ice–water separation under clutter, correcting many of the failures seen in baseline CNN predictions*

This training setup was particularly focused on generalization, enabling us to assign a label without any bias confidently. Eventually, DeepLab began to outperform the human annotations on which it was trained. Here is what made that result stand out:

- **Slush and edge regions** were often mislabeled or ignored in the dataset. The model began predicting them more clearly than even expert annotators.
- **Smoothness and continuity** have improved: predictions now avoid the jagged, flickering artifacts that earlier masks exhibited.
- **Confidence in ambiguous zones** improved — areas that were once hard to define now showed consistent and explainable label behavior.
- **Training beyond label quality** paid off: DeepLab started to capture a more stable visual grammar of Arctic surfaces [28].

This meant that the model was developing a latent structure of how land, ice, and water behave visually, even in harsh conditions like glare or post-breakup clutter.

6. EVALUATION AND RESULTS

The results presented here focus on both **quantitative performance**, as measured by IoU (Intersection over Union) and recall metrics, as well as **qualitative visual comparisons**. Special attention is paid to how the system handles challenging conditions such as glare, low-contrast slush-water boundaries, and fragmented ice structures.

6.1 Dataset and Evaluation Protocols

Evaluation was conducted on a **held-out subset of the CAIMAN dataset**, comprising frames from multiple sites across Nunavik. These frames were **not used during training** and were manually curated to encompass a diverse range of seasonal and lighting conditions. The test set specifications are as follows:

- **Test set size**: ~500 images
- **Scenes covered**: frozen shorelines, melt phases, slushy transitions, open water, post-breakup terrain

Each frame was processed independently. No temporal smoothing, ensemble correction, or post-hoc filtering was applied beyond what is described in the final pipeline. All evaluations were conducted using the output of the SAM, RAG, Merging, and DeepLab in the established processing pipeline described in Chapter 5.

Because the ground truth annotations themselves were imperfect — particularly in slush regions — the evaluation prioritizes consistency and semantic correctness over strict pixel-wise match [29]. For this reason, qualitative results are given equal weight alongside standard metrics.

6.2 Quantitative Metrics (IoU, Recall, etc.)

To evaluate the performance of the segmentation pipeline, five core metrics were used: Accuracy, Mean Intersection-over-Union (IoU), F1 Score, Precision, and Recall. These metrics together provide a comprehensive view of both structural fidelity and prediction reliability under ambiguous or low-contrast conditions.

Before presenting the results, **Table 6.1** defines each metric and explains its relevance to this segmentation task.

The full segmentation pipeline remained modular throughout development: SAM or SLIC was used for region generation, a graph-based merging strategy assembled mid-level structures, and a DeepLabv3+ classifier assigned final class labels to each region.

Metric	What It Measures	Why It Matters for This Project
Accuracy	Percentage of all pixels correctly classified	Indicates general correctness on clean frames; ensures the model gets basic land, ice, and water labels right.
Mean IoU	Average overlap between predicted and true regions	Reflects structural alignment—especially important for edge fidelity in slush and boundary zones.
F1 Score	Harmonic mean of precision and recall	Demonstrates balance between catching valid regions and avoiding over-prediction in ambiguous surfaces.
Precision	Ratio of true positives to total predicted positives	Reduces false positives, ensuring the model does not confidently mislabel uncertain areas.
Recall	Ratio of true positives to all actual positives	Shows that the model does not miss valid regions, especially critical when surfaces transition gradually.

Table 6.1: *Metric definitions and their relevance to segmentation quality.*

When we refer to the **"ResNet-50 pipeline"** or **"ResNet-101 pipeline,"** we specifically identify the backbone **used in the DeepLabv3+ classifier** at the final stage of the pipeline. All other components — region generation and merging — remained constant. Only the classifier's architecture was varied to evaluate its impact on precision, structural fidelity, and generalization.

Table 6.2 compares performance across different pipeline snapshots — before and after architectural improvements, and across both backbone variants.

Both backbones showed dramatic improvements between December 2024 and March 2025, as region-level annotations, merging rules, and classifier tuning matured. However, the final **ResNet-50 pipeline slightly outperformed ResNet-101**, despite being shallower. It achieved the highest Accuracy (0.92), F1 Score (0.92), and Mean IoU (0.83), suggesting it found a better trade-off between generalization and overfitting.

This is notable given the complexity of the task — shallow models often struggle with subtle features, but the region-first approach enabled even ResNet-50 to maintain high performance without relying solely on depth.

Pipeline Version	Accuracy	Mean IoU	F1 Score	Precision	Recall
Dec 2024 (ResNet-50)	0.73	0.15	0.37	0.40	0.95
Mar 2025 (ResNet-50)	0.92	0.83	0.92	0.92	0.92
Dec 2024 (ResNet-101)	0.75	0.15	0.35	0.37	0.95
Mar 2025 (ResNet-101)	0.89	0.78	0.88	0.90	0.89

Table 6.2: *Quantitative comparison of pipeline versions using DeepLabv3+ with ResNet-50 and ResNet-101 backbones.*

Key Takeaways:

- **F1 Score** rose from ~0.36 to 0.90+, reflecting confident labeling without sacrificing coverage.
- **Precision** gains reduced false positives, especially in slush regions.
- **A mean IoU of greater than** 0.78 confirms boundary-level structural accuracy.
- In many cases, the final model **outperformed its training labels**, especially under glare or mixed-surface conditions.

Although the metrics in Table 6.2 present strong overall gains, they conceal class-specific struggles — particularly for slush. The mean IoU, averaged across classes, does not accurately reflect how poorly earlier models performed on slush boundaries.

Similarly, recall stayed high across all runs (~0.95), but this was due to aggressive over-labeling rather than accurate segmentation. It was only through gains in precision and F1 score — especially in the March 2025 ResNet-50 pipeline — that the model began to suppress false positives and treat slush with appropriate caution.

The result is a model that now marks slush when confident, and ignores it when surface cues are ambiguous or noisy.

6.3 Qualitative Visual Comparisons

To understand how well the system works beyond just numbers, visual tests were carried out across a wide range of scenes — from clear, high-contrast frames to highly cluttered, glare-heavy conditions.

Ideal Conditions

In well-lit scenes with clearly defined surfaces, the system performs impressively (Figure 6.1). Land–ice–water boundaries are respected even in dense floe arrangements, and the classifier confidently separates regions. The final overlays show tight spatial alignment with real-world features.

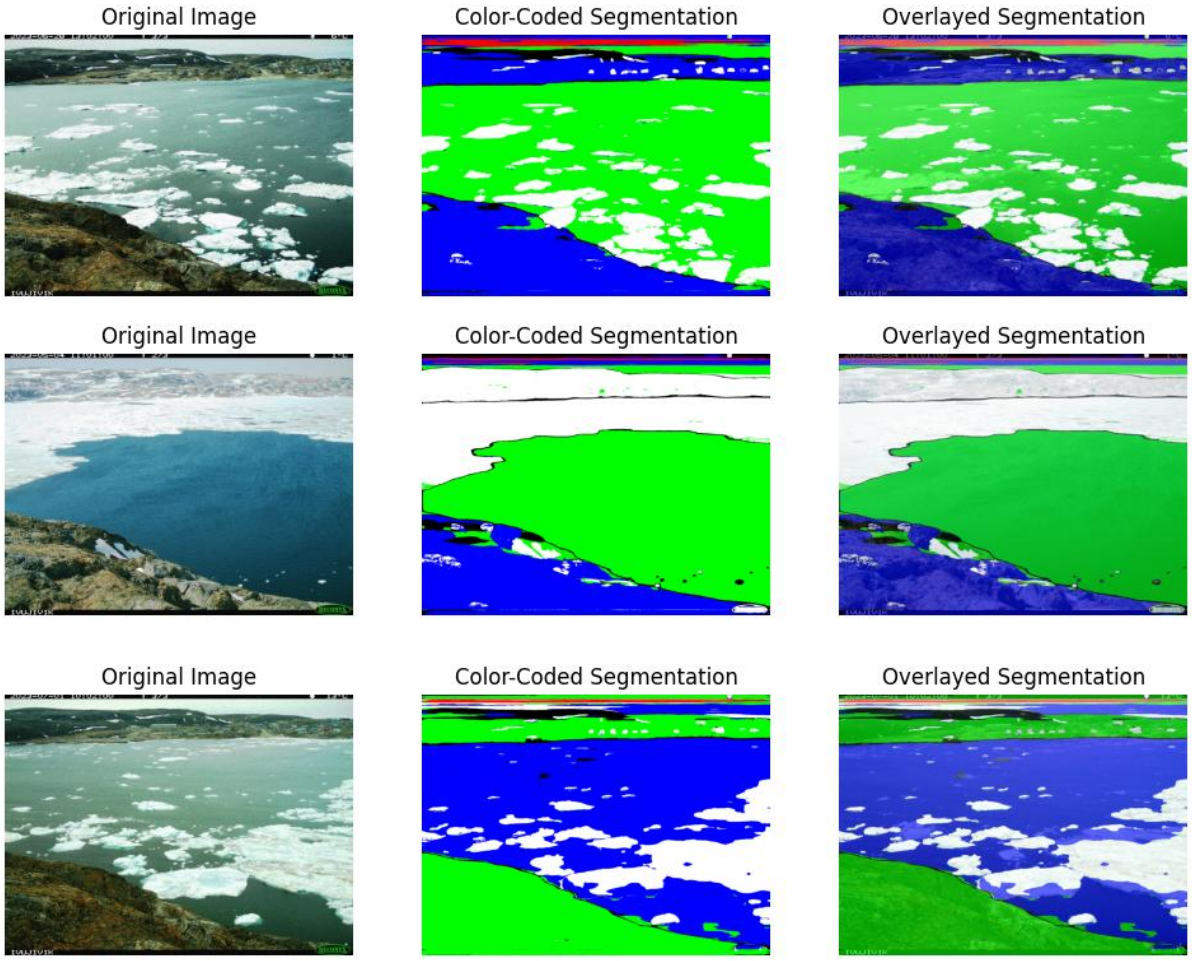

***Figure 6.1**: DeepLabv3+ segmentation results under ideal conditions. Individual floes and land-water boundaries are clearly identified and labeled with high consistency.*

The distinction between land ice and water ice became much clearer in these conditions, with individual floes being accurately identified and assigned consistent labels.

Overstimulation and Glare

In frames with intense glare or excessive fragmentation, the merging pipeline often struggles to perform effectively. Regions that should be distinct get clumped together, making it hard to track the slush.

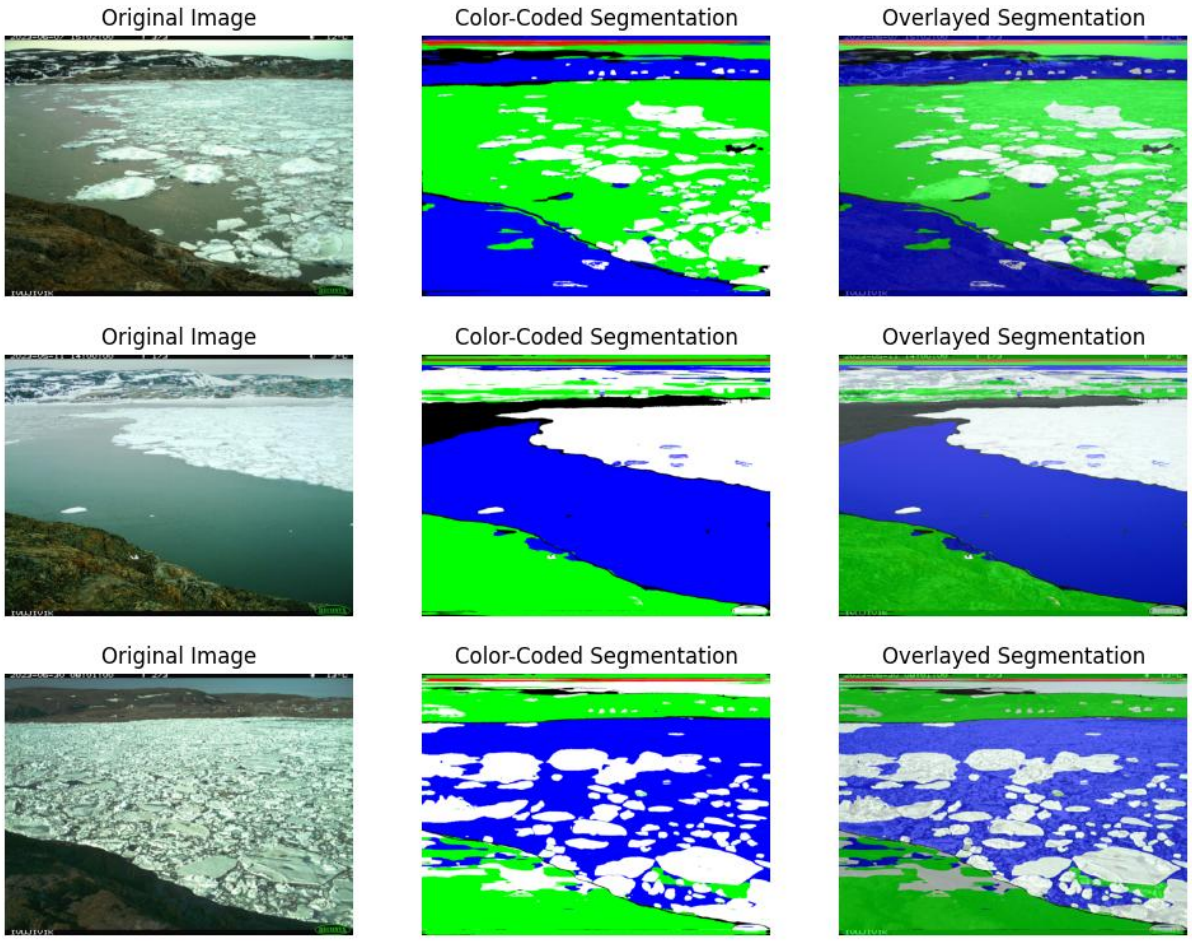

***Figure 6.2**: DeepLabv3+ segmentation under overstimulated scenes with glare and clutter.*

These issues are not caused by DeepLab misclassifying pixels — they result from the structure before it. The hierarchical merging logic cannot always keep up with rapid shifts in texture and brightness.

6.4 Failure Analysis

While the final pipeline performs well under most lighting conditions, it continues to struggle with ambiguous texture zones such as brash ice, slush, and the cluttered boundaries between ice and water.

This section doubles as a partial sensitivity analysis: by relating failures to brightness and variance thresholds, it highlights how the merging stage behaves under different parameter regimes. These features capture illumination and textural complexity, respectively, and expose where the merging logic becomes unstable.

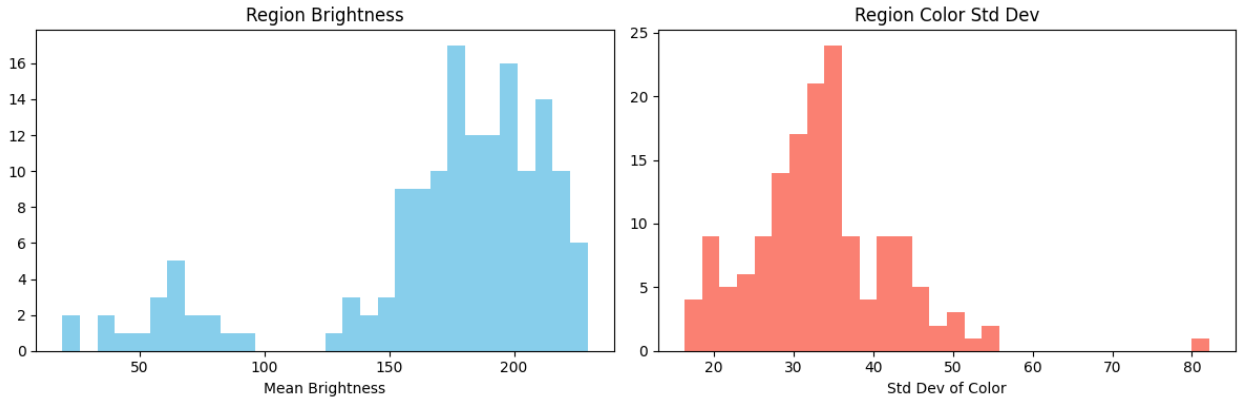

***Figure 6.3**: Histogram of region-wise brightness (left) and color standard deviation (right). Used to detect glare, complexity, or texture ambiguity across merged regions.*

These histograms show region-wise distributions for:

- **Brightness:** Stable clustering occurs between 150–220. Outliers below 80 (shadows) and above 230 (glare) correlate with unstable merges and fragmented regions.
- **Variance:** Most regions fall around 30–40. Outliers above 70 consistently map to brash ice and slush, where merging logic breaks down.

Together, these results show that segmentation failures consistently arise at the extremes of brightness and variance. The pipeline remains stable in the central ranges but becomes fragile when thresholds drift toward the tails. In effect, this analysis indicates a “safe zone” for merging logic: moderate brightness (150–220) and variance (30–40). Outside these ranges, instability is predictable.

Slush-like areas combine moderate brightness with unusually high variance, producing a recurring signature of ambiguity. This observation demonstrates that the merging process is inherently sensitive to these statistical extremes, which define the breakpoints where segmentation becomes unreliable.

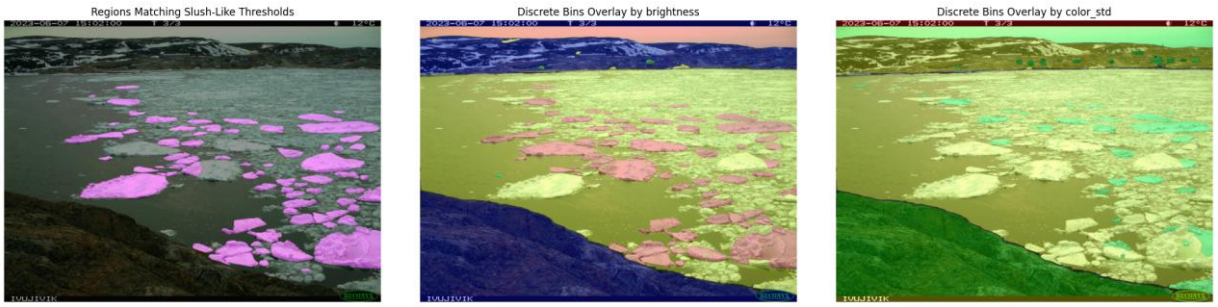

***Figure 6.4**: Failure overlay maps: regions incorrectly split or dropped during merging.*

To test whether these signatures could isolate systematic failure cases (Figure 6.4), manual thresholds were applied to flag regions with:

- **Glare-heavy zones (high brightness):** Large, intensely bright surfaces fragment into multiple smaller regions, making merges unstable and leaving inconsistent boundaries.
- **Cluttered zones (high variance):** Visually complex regions such as brash ice clusters exhibit unstable grouping. These areas are either over-segmented into incoherent fragments or dropped entirely, obscuring meaningful structure.
- **Slush-like transition zones (intermediate brightness + high variance):** These profiles consistently fail to form coherent regions, producing fragmented masks that DeepLab misclassifies as mixtures of water and ice.

The concentration of failures within these specific signatures indicates that errors are not randomly distributed but tied to predictable feature extremes. This suggests that incorporating feature-aware merging rules or adaptive thresholds could directly target these fragile zones, improving the stability and consistency of the pipeline in future iterations.

By mapping failures to brightness and variance extremes, the evaluation demonstrates precisely where the system remains fragile and where it is stable — a level of reliability directly relevant to CAIMAN's operational monitoring tasks

6.5 Runtime Analysis

To evaluate computational efficiency, the pipeline was benchmarked using a GPU. The raw logs report ~145 seconds per image on GPU, which naively suggests long runtimes. This is misleading because the logger tracked each image sequentially, while inference actually ran in parallel. In practice, a batch of 8 images required ~5 minutes total, corresponding to ~37 seconds per image—demonstrating that parallelized GPU inference is more efficient than raw logs indicate.

Metric		**Mean ± SD**
Processing Time	Proposals – SAM	138.7 ± 1.0 s (95.7% of total time)
	RAG build	1.45 ± 0.55 s (1.0%)
	Merge	0.11 ± 0.02 s (0.07%)
	Classification DeepLab	4.06 ± 0.48 s (2.8%)
	Post-processing	0.32 ± 0.08 s (0.22%)
	Total time per image	**145.0 ± 0.6 s**
Peak GPU VRAM		~614 MB
Peak CPU RAM		~4.4 GB
Regions (raw → merged)		48.0 → 37.4
Avg. merged region size		~121.7k px
Throughput (logged)		0.0069 img/s

Table 6.3: *Runtime and resource utilization breakdown of the GPU/SAM pipeline (N = 8).*

All tests used 2048×1536 images, single-image inference, with GPU/SAM+DeepLab. The results are summarized in Table 6.3, which makes clear that SAM proposals dominate wall-time (>95%), while downstream graph stages and DeepLab contribute marginally. The pipeline is compute-bound at the SAM proposal stage, where ViT-H accounts for more than 95% of execution time. Also, the modest VRAM footprint (~614 MB) indicates that the bottleneck lies in computing power rather than memory. Graph construction, merging, and post-processing are negligible, while DeepLab classification adds only about 3% overhead.

6.6 Attempts to Resolve Failure Modes

After identifying issues with inconsistent slush segmentation and merging instability in overstimulated scenes, several follow-up experiments to understand and potentially address these limitations, as shown in **Figure 6.6**.

Some of these attempts included:

- **Region Thresholding:** Filtered regions by brightness and texture to isolate slush-like patches.
- **Feature Binning:** Used color and variance bins to visualize mislabeled zones and detect threshold issues.
- **Late Merging Tweaks:** Applied relaxed merging rules post-hoc to reconnect fragmented slush — worked structurally, broke semantically.
- **Manual Region Injection:** Added hand-marked slush areas during training; without clean labels, the classifier could not learn from them.

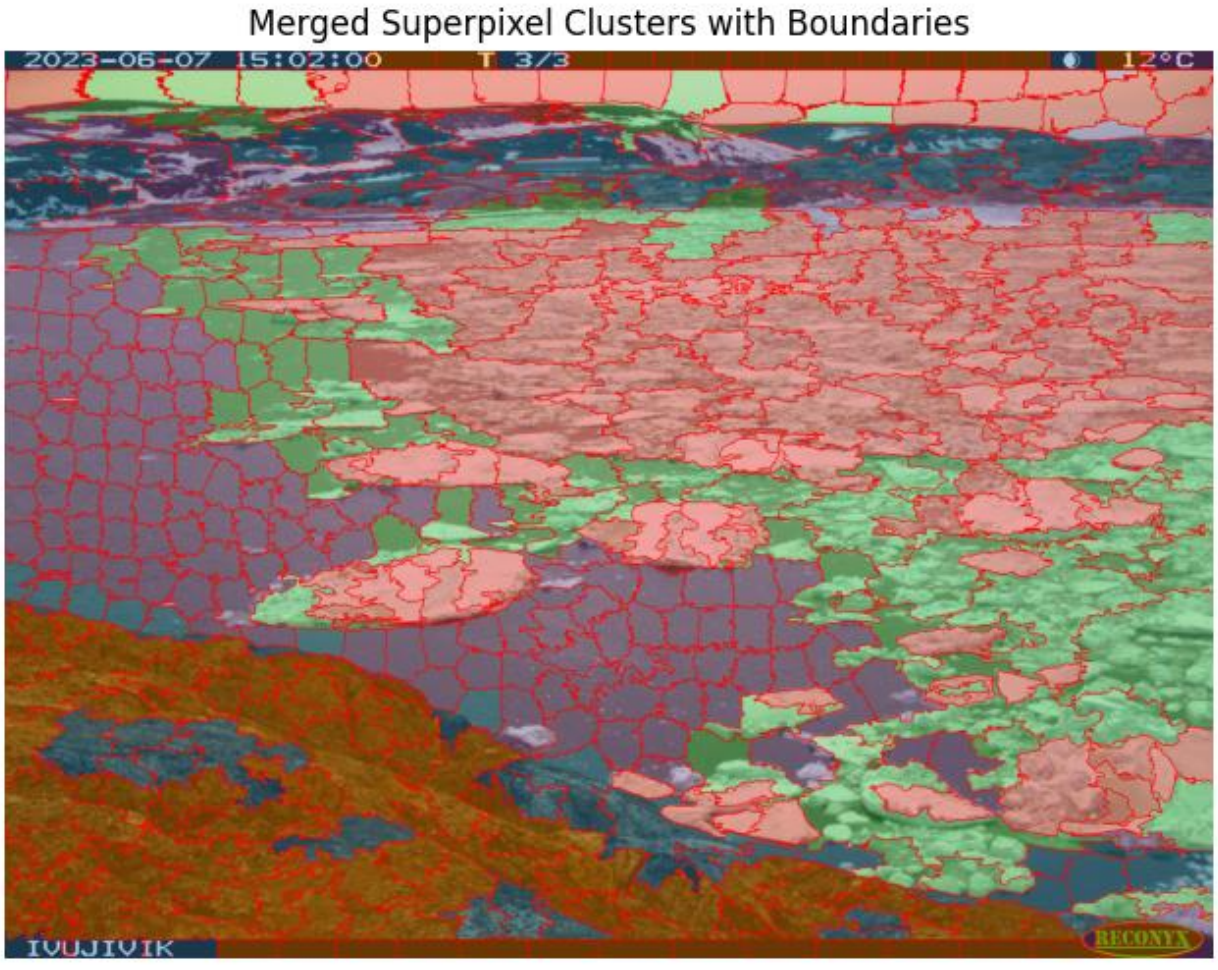

***Figure 6.6**: Post-hoc corrections applied to slush regions. Includes manual mask injection to restore semantic consistency.*

While these approaches helped expose the structure of the failure, they did not lead to directly deployable improvements. DeepLab's predictions often drifted once regions were altered too

heavily, confirming that the solution likely lies in a more integrated training strategy or stronger label data.

These insights nonetheless provide a foundation for future work, especially in developing rule-aware merging strategies or slush-specific pretraining.

7. REFLECTIONS AND BROADER INSIGHTS

7.1 Technical Insights and Contributions

This work introduces a hierarchical segmentation pipeline that integrates vision transformers with structured pixel-level reasoning—a combination rarely applied to ambiguous, low-contrast imagery. The layered system proves effective in parsing complex scenes, especially those where the boundaries between surfaces are visually entangled, as seen in Arctic imagery where ice and slush blur into water or land.

Ambiguous regions — where colors blend and boundaries blur — routinely trip up standard segmentation models. This pipeline avoids those traps by working structurally: instead of predicting over every pixel, it groups regions with similar visual traits and reasons over them as unified segments. That shift makes it possible to handle surfaces like slush or glare-heavy water, where edge cues are unreliable or absent.

Though developed for polar ice, the pipeline's modular design is easily transferable. Because each component — from region generation to final classification — operates independently, adapting the system to new environments requires only minimal adjustments. Whether it is vegetation mapping or detecting urban sprawl, the same structure holds. We will return to this idea in Section 8.3.

The dataset is the most consolidated and usable resource available for ice segmentation tasks today. Its one weakness is the underrepresentation of slush, which remains difficult to define and collect at scale. That said, the system's performance in slush-prone scenes suggests strong generalization potential.

7.2 Personal Growth

When I began this thesis, I did not expect it to carry such an emotional weight. Revisiting the results — from the rough ensemble masks of the first iteration to the structured outputs I now have — reminded me how far the work had come. I entered the program with enthusiasm for machine learning, but over time I developed the engineering mindset and practical fluency needed to build systems that hold up under real-world constraints.

Beyond technical growth, this project reshaped how I see visual understanding. I learned to think of vision not as isolated pixels but as structured patterns shaped by probability and context. That shift carried into everything I worked on afterward — LLMs, VLMs, demos, internships — reframing the central question: what part of this problem truly needs to be learned?

Some challenges benefit more from thoughtful design than from sheer model complexity. A clear rule, a lightweight heuristic, or the right mathematical lens can often outperform stacks of computation. That mindset anchored this thesis and guided my approach to AI more broadly.

By building a system that works under ambiguity, and does so consistently across multiple frames, this work moved beyond simple experimentation. It became a step toward building something that understands.

8. CONCLUSION AND FUTURE WORK

8.1 Summary of Findings

This thesis presents a modular segmentation pipeline tailored to ambiguous, low-contrast environments — specifically, land, ice, slush, and water in land-based Arctic camera footage. The system combines over-segmentation (via SAM or SLIC), graph-based merging, and region-wise classification with DeepLabv3+. Together, these steps allow the pipeline to reason locally and semantically across scenes where traditional models fail.

The final system showed significant gains in precision and generalization, particularly in boundary-dense or glare-heavy images. Through structured pre-processing and carefully tuned classification, the pipeline was able to outperform even its noisy labels. While slush remains a weak point, the results demonstrate that ambiguous regions can be modeled without relying on handcrafted rules or dense sensor inputs.

8.2 Limitations and Open Problems

Despite progress, a few core challenges remain unresolved; they are summarised in **Table 8.1**. These shortcomings represent deeper structural or data-related issues that limit the pipeline's generality and precision.

Area	Problem	Why It Matters	What Might Help
Slush Detection	Slush is often missed or merged into ice/water due to a lack of texture and poor training data.	It is the most ambiguous class and critical for shoreline monitoring	Collect better labels, possibly with expert guidance or crowd-verified annotation
Merge Criteria	Fixed thresholds for merging fail in glare-heavy or melt-phase scenes	Causes over- or under-merged regions with poor boundaries	Replace rule-based merge logic with learned, data-adaptive metrics
Texture Understanding	DeepLab's classifier is not sensitive enough to small texture shifts	Limits precision near transitions like wet ice or thin melt layers	Add texture-focused features or multi-scale encoders
Temporal Stability	Labels flicker between frames; no consistency over time	Makes video-based monitoring unreliable	Add temporal smoothing or model temporal context directly

Table 8.1: *Summary of known limitations and suggested improvements across structural and semantic aspects.*

8.3 Directions for Future Work

This project was never meant to end at segmentation. Its strength lies in the structure it creates — a modular view of the world that opens up possibilities for broader image understanding beyond Arctic environments.

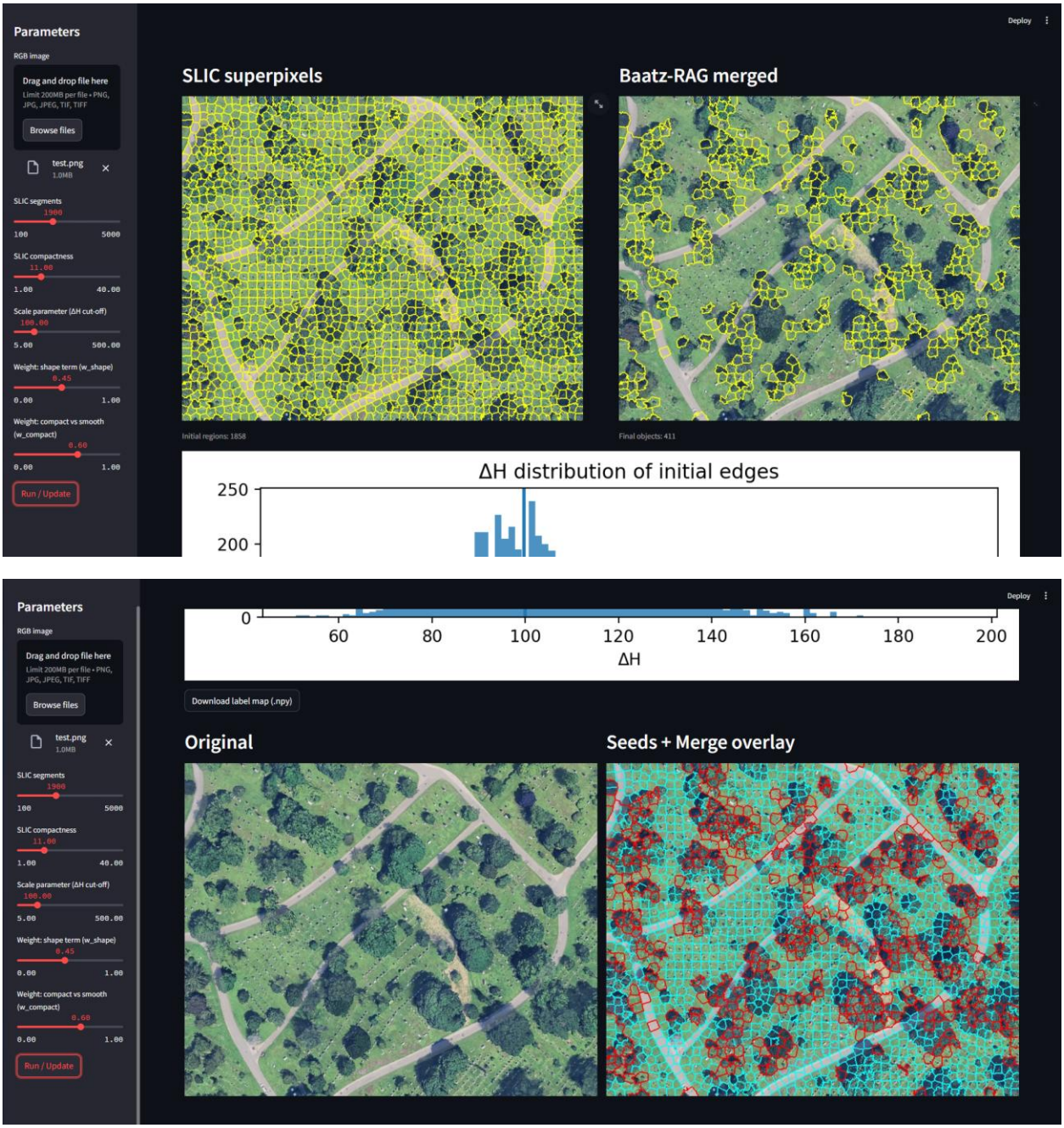

Figure 8.1: *Example from a new OBIA-style object detection project using region-based logic on aerial datasets.*

One such direction is already underway (**Figure 8.1** gives a preview). an OBIA-style [18] object detection framework tailored for large-scale geospatial datasets. The system relies on region-based logic: over-segmenting images with SLIC, merging areas through a combination of adjacency, Baatz-inspired texture cues, and classifying regions using a compact, task-specific model. The approach is rooted in the same principle that guided this thesis — begin with structure, and let semantics follow.

A key focus in this project is **dynamic scene scaling for superpixels**. Rather than using a fixed segmentation resolution, the system adapts granularity to local complexity: large homogeneous areas (e.g., open fields or water) are represented by coarser partitions, while transitional or ambiguous zones receive finer segmentation.

This adaptive approach addresses one of the major limitations identified in the thesis — the heavy reliance on superpixel quality — and has the potential to reduce computational overhead while improving accuracy in both detection and classification.

Beyond this immediate refinement, future work may also explore multimodal integration — for instance, combining RGB imagery with thermal or radar sensing — to further stabilize segmentation under Arctic extremes. In the longer term, adaptive graph-based reasoning could provide a framework for uncertainty-aware merging, allowing the system to generalize beyond fixed thresholds. Together, these directions point toward vision systems that are both efficient and resilient in real-world deployment

Looking further ahead, my proposed PhD work explores how these structural insights can be applied to the field of indoor robotics. In complex and poorly defined spaces — such as cluttered homes or partially collapsed buildings — traditional semantic categories often fall apart. I aim to build systems that do not just see "what" is in front of them, but reason about "where" and "how" they can move. This involves creating spatial representations that enable robots to plan safe paths through uncertain environments while remaining interpretable and adaptable.

In both ongoing and future work, the idea remains: ambiguity is itself a form of information that, if handled correctly, can make systems more capable.

BIBLIOGRAPHIE

[1] K. He, G. Gkioxari, P. Dollár, and R. Girshick, “Mask R-CNN,” Proc. ICCV, 2017.

[2] O. Ronneberger, P. Fischer and T. Brox, “U-Net: Convolutional Networks for Biomedical Image Segmentation,” Proc. MICCAI, 2015.

[3] S. Bhat, I. et al., “AdaBins: Depth Estimation Using Adaptive Bins,” Proc. CVPR, 2021.

[4] L.-C. Chen et al., “Encoder-Decoder with Atrous Separable Convolution for Semantic Image Segmentation,” Proc. ECCV, 2018. (DeepLab v3+)

[5] H. Zhao et al., “Pyramid Scene Parsing Network,” Proc. CVPR, 2017. (PSPNet)

[6] A. Kirillov et al., “Segment Anything,” arXiv:2304.02643, 2023.

[7] S. Ansari, C. D. Rennie, S. P. Clark and O. Seidou, “IceMaskNet: River ice detection and characterization using deep learning algorithms applied to aerial photography,” 2021.

[8] R. Prabha et al., “Lake Ice Monitoring with Webcams and Crowd-Sourced Images,” arXiv:2002.07875, 2020.

[9] A. Singh, H. Kalke, M. Loewen and N. Ray, “River Ice Segmentation with Deep Learning,” arXiv:1901.04412, 2019.

[10] B. Dowden, O. De Silva and W. Huang, “Sea Ice Image Semantic Segmentation Using Deep Neural Networks,” Proc. IEEE, 2021.

[11] X. Zhang et al., “ICENET: A Semantic Segmentation Deep Network for River Ice by Fusing Positional and Channel-Wise Attentive Features,” Remote Sensing, vol. 12, no. 2, 2019.

[12] C. Zhang, X. Chen and S. Ji, “Semantic image segmentation for sea-ice parameters recognition using deep convolutional neural networks,” 2022.

[13] N. Panchi, E. Kim and A. Bhattacharyya, “Deep Learning-Based Image Segmentation System for Automatic Detection and Localization of Sea-Ice Formations,” IEEE, 2021.

[14] G. Jocher, A. Chaurasia, J. Stoken et al., “YOLOv8: Next-Generation, State-of-the-Art Object Detection and Segmentation,” Ultralytics Tech. Report, 2023. [Online]. Available:https://docs.ultralytics.com

[15] R. Achanta, A. Shaji, K. Smith, A. Lucchi, P. Fua and S. Süsstrunk, “SLIC Superpixels Compared to State-of-the-Art Superpixel Methods,” IEEE Trans. Pattern Anal. Mach. Intell., vol. 34, no. 11, pp. 2274-2282, 2012.

[16] A. Dosovitskiy, L. Beyer, A. Kolesnikov et al., “An Image Is Worth 16×16 Words: Transformers for Image Recognition at Scale,” Proc. ICLR, 2021. (Vision Transformer, ViT)

[17] M. Baatz and A. Schäpe, “Multiresolution Segmentation—An Optimization Approach for High-Quality Multi-Scale Image Segmentation,” in Object-Based Image Analysis, Springer, 2000, pp. 110-122.

[18] R. O. Duda, P. E. Hart and D. G. Stork, Pattern Classification, 2nd ed., Wiley-Interscience, 2000, ch. 2 (for the formal definition and properties of Euclidean distance).

[19] B. Gerke, “PixelAnnotationTool: Manual Image Annotation Tool for Semantic Segmentation,” GitHub, 2017. [Online]. Available: https://github.com/abreheret/PixelAnnotationTool

[20] C. Arnab et al., “Weakly Supervised Instance Segmentation using Class Peak Response,” CVPR, 2018.

[21] S. W. Park et al., “Ensemble Learning for Semantic Segmentation with Uncertainty,” Sensors, 2021

[22] Y. Zhou et al., “Multi-Scale Patch-Based CNN for Semantic Segmentation of Aerial Imagery,” Remote Sensing, vol. 13, 2021.

[23] S. Mehta et al., “ESPNet: Efficient Spatial Pyramid of Dilated Convolutions for Semantic Segmentation,” ECCV, 2018.

[24] X. Ren and J. Malik, “Learning a Classification Model for Segmentation,” ICCV, 2003.

[25] P. F. Felzenszwalb and D. P. Huttenlocher, “Efficient Graph-Based Image Segmentation,” IJCV, 2004.

[26] A. Kirillov et al., “Panoptic Segmentation,” CVPR, 2019.

[27] S. C. Pang et al., “Glare-Resistant Image Segmentation for Aquatic Remote Sensing,” IEEE Journal of Selected Topics in Applied Earth Observations, 2019.

[28] J. P. Cohen et al., “Distributionally Robust Learning for Weakly Supervised Segmentation,” NeurIPS, 2020.

[29] A. Kendall et al., “Bayesian SegNet: Model Uncertainty in Deep Convolutional Encoder-Decoders for Scene Understanding,” BMVC, 2015.

[30] Y. Zhao et al., “Fast Segment Anything,” *arXiv preprint arXiv:2306.12156*, 2023.

[Online]. Available: https://arxiv.org/abs/2306.12156

[31] K. He, X. Zhang, S. Ren, and J. Sun, “Deep Residual Learning for Image Recognition,” *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016, pp. 770–778.

[32] A. Kumar, G. Carneiro, and I. Reid, “Learning Local Image Descriptors with Deep Siamese and Triplet Convolutional Networks by Minimizing Global Loss Functions,”

Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2016

[33] U-KAN: Li, C., Liu, X., Li, W., Wang, C., Liu, H., Liu, Y., & Chen, Z., & Yuan, Y. (2024). *U-KAN Makes Strong Backbone for Medical Image Segmentation and Generation*. arXiv.

[34] U^2-Net: Qin, X., Zhang, Z., Huang, C., Dehghan, M., Zaiane, O. R., & Jagersand, M. (2020). *U^2-Net: Going Deeper with Nested U-Structure for Salient Object Detection*. *Pattern Recognition*, 106, 10740