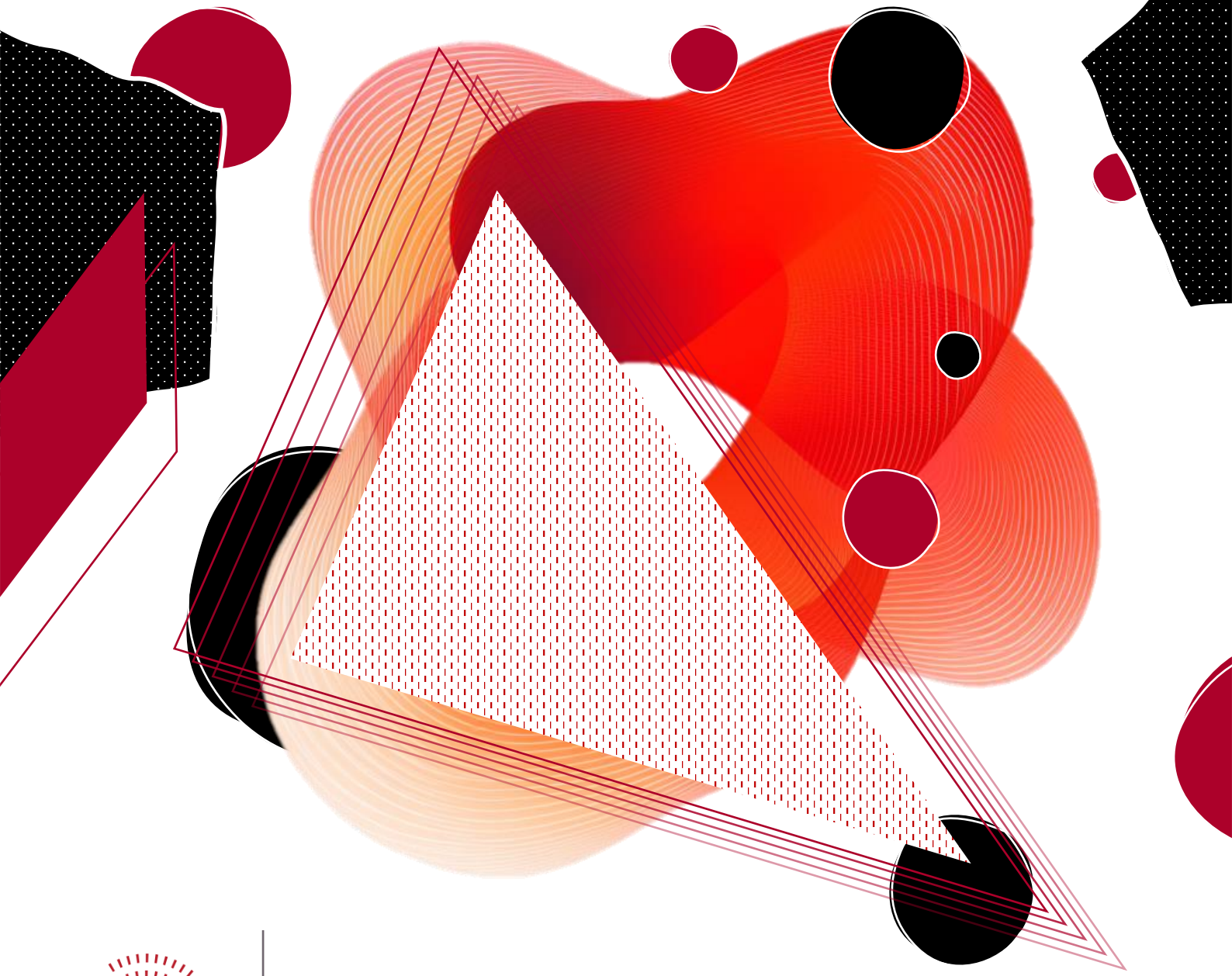
CUMUNEL Lucas , LEROUX Tara, LEROY Léo, SIAHAAN–GENSOLLEN Rémy
Encadré par : KIRSCHER Tristan , COUBEZ Xavier

Réduction de l'incertitude en segmentation médicale par méthodes d'ensemble

Rapport de projet de statistique et science des données appliquées



ENSAE

INSTITUT POLYTECHNIQUE DE PARIS

Rapport de projet de statistique et science des données appliquées

Cumunel Lucas, Leroux Tara, LEROY Léo, Siahaan–Gensollen Rémy
Supervisé par : Coubez Xavier PhD, Kirscher Tristan

Réduction de l'incertitude en segmentation médicale par méthodes d'ensemble

RÉSUMÉ

La segmentation automatique des organes, bien que très utile en imagerie médicale, reste sujette à une forte incertitude, notamment lorsqu'elle repose sur des annotations manuelles potentiellement subjectives. Ce projet présente une évaluation systématique de méthodes d'ensemble de réseaux U-Net pour réduire cette incertitude. Nous entraînons et inférons plusieurs modèles sur des scans tomodensitométriques de patients annotés par différents experts, que nous combinons à l'aide d'une méthode d'ensemble. Ensuite, nous proposons et appliquons à ces prédictions un cadre d'évaluation systématique de leur précision, de leur incertitude aléatoire et de leur incertitude épistémique. Nos résultats indiquent que les méthodes d'ensemble utilisées diminuent significativement les incertitudes des prédictions sans détériorer leur précision.

SOMMAIRE

I Contexte et projet

I.1 Introduction

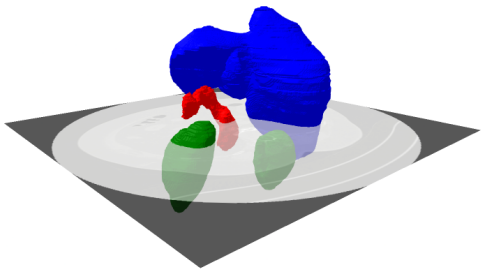


Figure 1 *Segmentation 3D du pancréas, des reins et du foie d'un patient, ainsi qu'une coupe du scanner abdominal utilisée pour les délimiter.*

Depuis plusieurs années, l'intelligence artificielle révolutionne la pratique médicale, en soutenant les médecins dans leurs diagnostics et leurs prises de décisions. L'imagerie médicale, en particulier, joue un rôle central dans l'évaluation de l'état de santé des patients et l'orientation de leur prise en charge [Li et al., 2023]. La segmentation automatique — c'est-à-dire la délimitation précise des organes et des structures par des algorithmes — facilite le diagnostic, la planification du traitement et le suivi clinique. On retrouve parmi ces algorithmes les réseaux de neurones convolutifs (*Convolutional Neural Network*, ou *CNN*), puissant outil d'apprentissage profond (*deep learning*) ayant surpassé les experts humains dans de nombreuses tâches de compréhension d'images [Sarvamangala & Kulkarni, 2022]. Une des architectures de CNN les plus utilisées pour la segmentation médicale est le réseau U-Net [Ronneberger et al., 2015].

Cependant, beaucoup des structures et anomalies analysées (organes, vaisseaux sanguins, tumeurs, etc.) sont particulièrement complexes et variables, conduisant à une certaine *incertitude* dans leur délimitation. Cette incertitude est accentuée par la *variabilité inter-experts* : différents spécialistes médicaux peuvent avoir des opinions divergentes sur l'emplacement précis des limites des entités segmentées. Elle s'accroît d'autant plus lorsque plusieurs structures sont prédites simultanément (*problème multi-classes*). Les réseaux de neurones doivent composer avec ces divergences, conduisant parfois à des incohérences dans les résultats de segmentation, ce qui peut directement impacter les décisions médicales prises.

Quantifier ces incertitudes permet de générer des cartes d'incertitude sur les images médicales, afin d'isoler les zones où les médecins doivent redoubler d'attention, fournir aux cliniciens des prédictions mieux calibrées et intégrer des mesures de confiance dans l'analyse des images médicales et la prise de décision qui en découle [Kahl et al., 2024]. Cela améliore non seulement la sécurité des diagnostics assistés par IA, mais rend également les algorithmes plus transparents et fiables pour les applications médicales. Quantifier les incertitudes permet également d'évaluer l'impact des choix de méthodologie pour l'apprentissage machine : architectures des réseaux, tailles des jeux de données, durée d'entraînement, etc… Les méthodes d'apprentissage d'ensemble, consistant à combiner plusieurs modèles individuels ou leurs prédictions, sont un choix courant pour améliorer la performance des modèles d'intelligence artificielle [Ganaie et al., 2022].

Ce projet de statistiques appliquées consiste à évaluer l'impact sur l'incertitude de méthodes d'ensemble de modèles de segmentation médicale automatique. La prochaine sous-section donnera une présentation de la quantification de l'incertitude et de son état de l'art. Les sections suivantes détailleront l'expérience, la méthodologie d'évaluation, et discuteront des résultats obtenus.

I.2 Quantification de l'incertitude

Pour les médecins, connaître la fiabilité d'une segmentation est essentiel. Cependant, les modèles d'apprentissage machine n'indiquent pas toujours clairement leur niveau de confiance dans les prédictions qu'ils produisent : c'est le problème de *l'incertitude dans les prédictions algorithmiques*. Par ailleurs, les experts médicaux peuvent annoter une même image différemment en raison de l'ambiguïté de certaines structures anatomiques. Ces désaccords réduisent la qualité des annotations utilisées pour entraîner les modèles et compliquent l'évaluation de leurs performances. Nous présentons ci-dessous, dans La figure 2, trois coupes du scan tomodensitométrique (qu'on désignera aussi dans ce rapport par scan abdominal, ou de *CT scan*, sans distinction) du premier patient du jeu de données fourni pour le défi CURVAS (plus de détails seront donnés plus loin), ainsi que les trois annotations du pancréas, du rein et du foie. La figure 3 met en évidence les zones de désaccord :

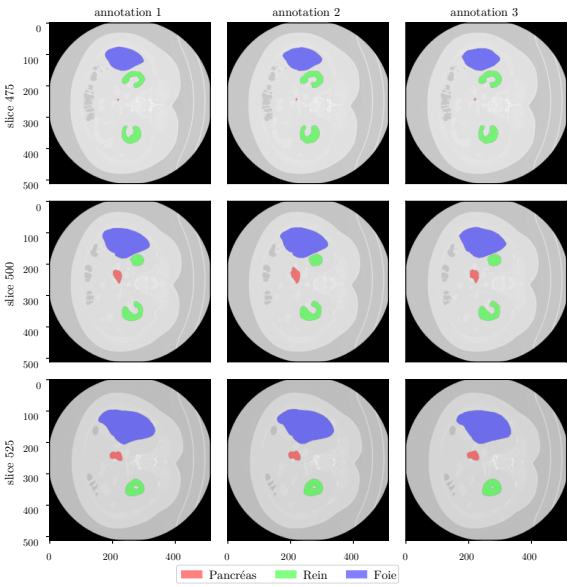


Figure 2 *Contours réalisés par trois médecins pour différents organes sur trois coupes de CT scan d'un même patient.*

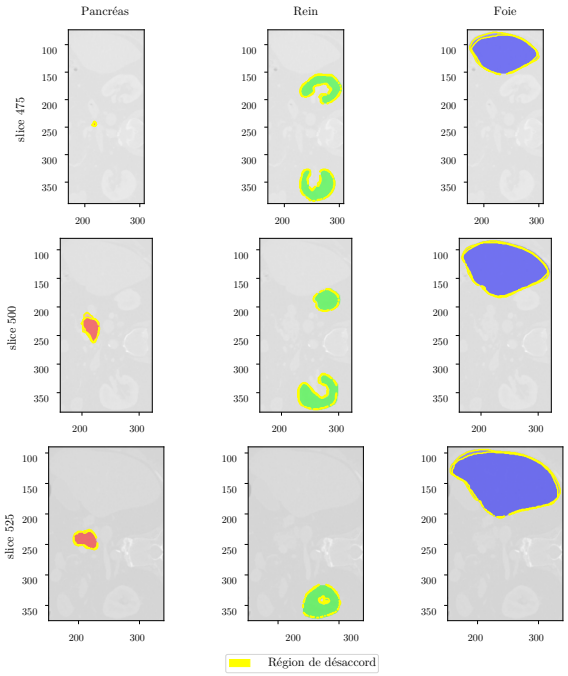


Figure 3 *Zones de dissensus mises en évidence en jaune*

Théoriquement, on distingue deux types d'incertitude, qui, une fois combinées donnent l'incertitude prédictive (*Predictive Uncertainty*, ou *PU*) :

— L'incertitude aléatoire (*Aleatoric Uncertainty*, ou *AU*) qui provient des données elles-mêmes. Elle est liée aux ambiguïtés intrinsèques à l'image. On peut citer

comme cause d'incertitude aléatoire les artefacts, les erreurs de numérisation, etc… Parmi ces causes, on peut notamment citer les désaccords entre annotateurs, comme illustré précédemment.

— L'incertitude épistémique (*Epistemic Uncertainty*, ou *EU*), qui provient du modèle d'apprentissage lui-même. On peut citer comme cause d'incertitude épistémique un manque de connaissances (pas assez de données diversifiées observées durant l'entraînement), une architecture ne permettant pas de bien les « apprendre », etc…

L'approche la plus notable pour capturer ces incertitudes a été introduite par [Kendall & Gal, 2017], qui l'abordent dans le cadre d'un classificateur bayésien. Ce classificateur reçoit une entrée x et produit des probabilités pour les classes Y :

$$\mathbf{P}(Y \mid x) = \mathbb{E}_{\omega \sim \Omega}\left[\mathbf{P}(Y \mid x, \omega)\right]$$

où les paramètres du modèle Ω suivent $\mathbf{P}(\omega \mid D)$ pour les données d'entraînement D.

Ce cadre bayésien [Smerkous et al., 2024] suppose que l'incertitude épistémique est représentée par l'entropie prédictive (*Predictive Entropy*, ou *PE*), qui est la somme de l'information mutuelle (*Mutual Information*, ou *MI*) et de l'entropie attendue (*Expected Entropy*, ou *EE*), représentant respectivement l'incertitude épistémique et l'incertitude aléatoire. En notant $\mathbf{H}$ l'entropie de Shannon, on a :

$$\underbrace{\mathbf{H}(Y \mid x)}_{PU=PE} = \underbrace{\mathrm{MI}(Y, \Omega | x)}_{EU=MI} + \underbrace{\mathbb{E}_{\omega \sim \Omega}[H(Y|\omega, x)]}_{AU=EE \text{ (pour x i.i.d.)}}$$

La figure 4 illustre les deux types d'incertitudes pour une régression unidimensionnelle :

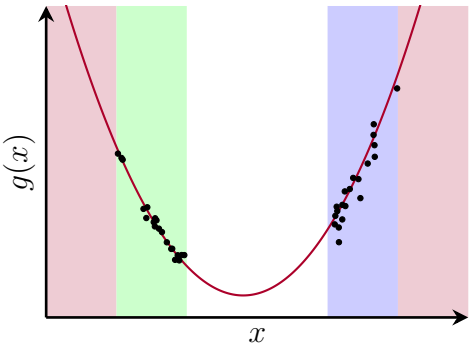


Figure 4 *Illustration des incertitudes épistémiques et aléatoires pour une régression 1D.*

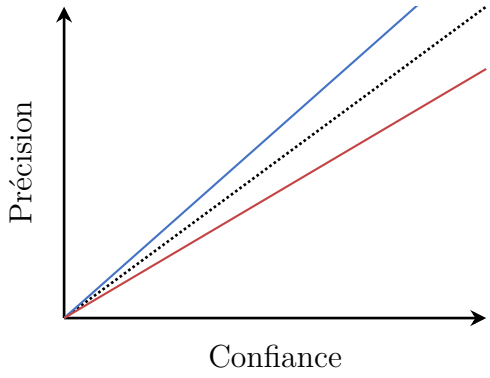
— dans les régions rouge, on a une forte incertitude épistémique car le modèle n'a jamais été exposé aux données dans ces plages et a pourtant fourni une approximation ;

— dans la région blanche, on a seulement quelques points de données, donc une incertitude épistémique modérée ;

— dans la région verte, on a peu de variance donc une incertitude aléatoire faible ;

— dans la région bleue, enfin, on a plus de variance donc une incertitude aléatoire plus importante.

Un autre concept très important de *calibration*. Les réseaux de neurones produisent des distributions de probabilités sur les étiquettes de classe possibles, ce qui constitue une mesure naturelle de l'incertitude. Idéalement, un modèle bien calibré devrait avoir une confiance élevée pour les prédictions correctes et une faible confiance pour les prédictions incorrectes. Cependant, les architectures modernes échouent souvent à atteindre cette calibration idéale. Pour évaluer la calibration, on utilise des diagrammes de fiabilité (ou graphiques de calibration), qui comparent la confiance prédite à la précision réelle, mettant ainsi en évidence les écarts — appelés écarts de calibration. La figure 5 illustre trois cas :



— calibration parfaite (ligne en pointillés noirs), où la précision correspond à la confiance ;
— sous-confiance (ligne bleue), où le modèle est trop prudent (la précision dépasse la confiance) ;
— sur-confiance (ligne rouge), où le modèle est trop confiant (la confiance dépasse la précision).

Figure 5 *Illustration des calibrations*

Mathématiquement, un modèle parfaitement calibré satisfait :

$$\forall p \in [0,1]\,, \qquad \mathbf{P}\left(\hat{Y} = Y \;\middle|\; \hat{P} = p\right) = p$$

Autrement, cela signifie que si le modèle attribue une probabilité de 80% à une prédiction, il devrait avoir raison 80% du temps.

II Expérience

II.1 Données et modèles

Tenu de mai à octobre 2024, le challenge CURVAS (*Calibration and Uncertainty for Multi-Rater Volume Assessment in Multiorgan Segmentation*) mettait les équipes au défi de produire un modèle de segmentation précis, capable de déterminer la meilleure calibration et quantification de la variabilité inter-expert. Nous utilisons pour ce projet le jeu de données mis à disposition à l'occasion de ce challenge, contenant au total 90 CT scans de patients, chacun annoté par 3 experts différents pour délimiter le pancréas, les reins et le foie de chaque patient. Les figures 1, 2 et 3 sont réalisées à partir des données du premier patient de la cohorte. Ces scans tomodensitométriques ont été recueillis à l'University Hospital Erlangen entre août et octobre 2023. 20 CT scans ont été fournis pour l'entraînement (groupe A), 5 pour la validation (groupe A), et 65 pour le test (20 en groupe A, 22 en groupe B et 23 en groupe C).

Pour les entraînements, nous avons utilisé le framework nnU-Net (no-new-UNet) [Isensee et al., 2018], une bibliothèque d'outils permettant d'entraîner des réseaux U-Net pour la segmentation, conçue spécifiquement pour la segmentation automatisée d'images biomédicales. Contrairement aux réseaux U-Net classiques [Ronneberger et al., 2015], régulièrement utilisés pour la segmentation sémantique, nnU-Net configure automatiquement de nombreux paramètres en fonction des caractéristiques de l'ensemble de données. Ces configurations sont indispensables car, dans les hôpitaux, les images médicales sont produites avec différents instruments, ne respectent pas les mêmes conventions et ont des formats différents (2D, 3D), des saturations et des dimensions variables, … Toutefois, nnU-Net présente l'inconvénient d'être très coûteux en calcul et nécessite des GPU performants.

Nous avons d'abord entraîné 9 modèles différents sur le jeu de données d'entraînement (20 patients) : pour chaque annotateur, nous avons entraîné trois modèles avec des initialisations différentes des poids, afin d'explorer des points distincts de l'espace des pertes (*loss landscape*). Par souci de reproductibilité, les générateurs aléatoires des poids ont été déterministes en fixant les graines (aussi appelées nombres aléatoires, ou « seeds ») `112233`, `445566` et `778899`. Ensuite, nous avons inféré chacun de ces modèles sur le jeu de données de test (65 patients). Systématiquement, nous avons généré les probabilités (sorties softmax du modèle) pour chacun des modèles et des patients, que nous avons ensuite utilisées pour produire 4 ensembles : un pour chaque triplet de modèles pour un même annotateur, et un général sur l'ensemble des 9 modèles. Enfin, nous avons exécuté, pour chacun des patients et les 13 modèles différents, des calculs évaluant la précision des prédictions ainsi que les incertitudes aléatoires et épistémiques. Ces calculs et leurs résultats sont présentés dans les sections suivantes.

II.2 Outils et ressources

Tout d'abord, nous avons dû modifier la bibliothèque nnU-Net pour y intégrer les fonctionnalités dont nous avions besoin et qui n'étaient pas initialement présentes. La première a été l'arrêt anticipé/précoce (*early stopping*) des entraînements, car ceux-ci pouvaient prendre plusieurs jours à finir même sans progrès notable. Nous avons donc limité la durée d'entraînement à 300 epochs (cycles d'entraînement) au plus, avec un arrêt anticipé lorsqu'il n'y avait pas d'amélioration notable pendant 20 epochs. Nous avons ensuite ajouté la possibilité de fixer des « seeds » d'initialisation aléatoire mentionnée plus haut, également absente de la bibliothèque de base. Quand le réseau de neurones commence son entraînement, les poids des neurones sont dans un premier temps fixés aléatoirement selon un générateur de nombres aléatoires déterminé par la « seed » choisie, pour être ensuite ajustés pendant l'entraînement. Il est à noter que d'autres sources d'aléatoire peuvent intervenir — notamment au sein du module `PyTorch`, sur lequel repose nnU-Net, en particulier du point de vue des algorithmes de descente de gradient et d'optimisation (ou *optimizers*) — néanmoins, cette première déterminisation permettait de mieux contrôler l'exploration de l'espace de pertes (*loss landscape*).

Les entraînements des modèles U-Net sur les scans tomodensitométriques en trois dimensions sont conséquents, même avec la fonctionnalité d'arrêt anticipé. Ils nécessitent par ailleurs des cartes graphiques de calcul (GPU). Nous avons donc utilisé des instances disponibles sur les services Onyxia de l'Insee et du Groupe Genes auxquels nous avons eu accès. Même avec cela, les entraînements prenaient près d'une journée pour chacun des modèles. Par ailleurs, plusieurs difficultés sont survenues pour l'inférence, la méthode d'ensemble et l'évaluation. En effet, les volumes de données transférés à chaque fois étaient particulièrement importants, et chaque instance étant limitée à 100 Go, nous avons dû traiter individuellement chaque tâche et chaque patient (parfois même chaque modèle) sur des instances différentes.

Du fait de cette décomposition importante de chaque tâche, nous avons dû faire très attention à la reproductibilité. Pour le stockage, l'Insee nous a généreusement mis à disposition un espace de stockage compatible avec le standard S3 d'Amazon (plus précisément de la solution open source MinIO) pour le transfert de fichiers. Celui-ci contient

actuellement plusieurs milliers de Go correspondant aux artefacts produits par nos entraînements. Nous avons alors développé une interface en ligne de commande (*Command-Line Interface*, ou *CLI*) avec le module `Typer`, permettant d'interagir avec le stockage distant via une clé de service créée pour le projet et partagée entre tous nos services.

Cette application avait sinon pour utilisation principale de permettre le lancement systématique de chaque tâche avec une granularité de contrôle maximale. Les figures 6 et 7 ci-dessous présentent respectivement les commandes de la CLI (voir également en annexe) et un exemple d'utilisation (inférence sur le patient 80 du troisième modèle de l'annotateur 2, montrant bien la granularité). Le développement de cet outil a été une des étapes les plus importantes du projet. Le code source contient près de 6 500 lignes de Python et de documentation, écrites spécifiquement pour le projet (voir annexe). Ce code est également disponible sur le dépôt GitHub du projet.

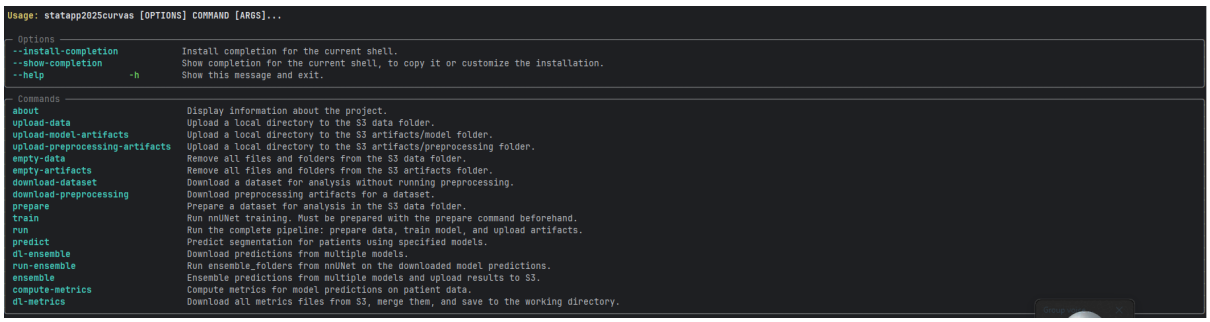


Figure 6 *Liste des commandes de la CLI*

Figure 7 *Exemple d'utilisation de la CLI pour l'inférence*

Comme le code de calcul des métriques était trop lent, nous l'avons optimisé en utilisant le module `numba`. Celui-ci nous a permis d'optimiser les fonctions d'évaluation en les compilant à la volée en code machine, via le compilateur `llvm`. Ainsi, l'interprétation n'était plus effectuée par l'interpréteur Python, mais à un niveau natif (comme du code C ou FORTRAN compilé), ce qui accélère fortement les boucles et les opérations numériques. Il est particulièrement efficace pour les traitements sur les tableaux, à condition d'utiliser des variables simples et statiquement typables. En l'occurrence, certains traitements voxel-à-voxel dans les images 3D médicales étaient, après optimisation, près de 100 fois plus rapides que l'interprétation Python, et le temps total d'évaluation a été divisé par plus de 4.

II.3 Protocole expérimental détaillé

Téléversement des jeux de donnés

Dans un premier temps, nous avons récupéré les données du challenge CURVAS [Riera-Marín et al., 2024], c'est-à-dire obtenu et mis en ligne sur le S3 les scans de coupes transversales de l'abdomen de 90 patients ainsi que les annotations de 3 experts

différents pour chacune d'elles. Ces annotations représentent les contours de trois organes : le pancréas, les reins et le foie. Tous ces fichiers sont au format `.nii.gz`, et contiennent des tenseurs de forme (organe, X, Y, Z), pour 866 tranches de 512 par 512 voxels de CT scans, avec une valeur de voxel en niveaux de gris pour l'image et un entier entre 0 et 4 pour chaque annotation (0 représentant le fond, 1 le pancréas, 2 les reins et 3 le foie).

Entraînement

Ensuite, il a fallu entraîner le réseau de neurones nnU-Net sur 20 de ces patients. Pour ce faire, nous avons modifié le code de nnU-Net, qui fonctionne sur une base similaire à notre CLI, avec plusieurs commandes, pour appeler directement les fonctions nécessaires. Une fois cela fait (notamment pour respecter la structure des dossiers et de l'enregistrement des fichiers d'entraînement, qui est fixée dans nnU-Net), nous avons tout d'abord téléchargé les données depuis le S3 puis lancé une étape de pré-traitement (*preprocessing*) propre à nnU-Net, où des algorithmes analysent automatiquement chaque image pour normaliser les intensités lumineuses, les tailles, etc… Cette étape étant la même pour chaque annotateur, notre CLI téléversait à la fin du preprocessing les fichiers générés sur le S3 pour ne pas avoir à relancer cette étape lors des entraînements ultérieurs.

Une fois les différents pré-traitements achevés, nous avons lancé les entraînements des 9 modèles. Par défaut, nnU-Net entraîne 5 « folds » en validation croisée sur les 80 % des données fournies. Manquant de ressources de calcul, nous avons exécuté un seul « fold » sur l'ensemble des données. Chacune des initialisations était déterminisée à l'aide d'une seed, en utilisant le code que nous avions implémenté. Nous avons fixé une limite à 300 epochs, avec un arrêt anticipé lorsque le DICE mesuré pendant l'entraînement n'évoluait plus pendant 20 epochs. Les modèles entraînés étaient ensuite automatiquement transférés sur le stockage S3.

NnU-Net génère automatiquement des fichiers checkpoint, qui contiennent les poids du réseau de neurones à une étape donnée de l'entraînement. Nous utilisons ces fichiers pour l'inférence : à partir de ces poids et d'un nouveau CT scan, nous pouvons générer une nouvelle annotation. Ces fichiers permettent aussi de reprendre l'entraînement en cas d'erreur ou de perte de connexion (arrêt du service). nnU-Net sauvegarde également automatiquement les poids offrant la meilleure performance. En effet, en raison de la descente de gradient stochastique, la performance peut diminuer légèrement à l'epoch suivante (période d'entraînement).

Inférence

Une fois les modèles entraînés, nous avons pu récupérer les prédictions pour les patients restants (64). L'inférence avec nnU-Net nécessite tout d'abord de télécharger les modèles entraînés (depuis notre S3) ainsi que les scans abdominaux bruts. Comme nous devions effectuer cette inférence sur 9 modèles, cela représente 576 lancements, avec un temps de calcul très conséquent. L'inférence était particulièrement longue et volumineuse en raison de la nécessité de produire les tenseurs des probabilités de sortie (c'est-à-dire, pour chaque voxel, le softmax des logits — scores non normalisés — associés aux trois classes d'organes, et non simplement la classe prédite, le tout généré dans un fichier au format `.nii.gz`). Ces tenseurs avaient une dimension importante : (4, 866, 512, 512). Les fichiers `.npz` faisaient donc tous plus de 3 Go. Une fois les prédictions effectuées, elles

étaient également transférées automatiquement sur le S3.

Méthodes d'ensemble

En utilisant les softmax générés à la phase d'inférence, nous avons pu prendre la moyenne des probabilités pour obtenir des prédictions correspondant aux différents ensembles. Cette phase téléchargeait donc, pour chaque patient, les 9 prédictions de la phase précédente (prédiction et softmax), calculait de nouveaux softmax moyens pour les ensembles décrits plus haut (un par annotateur, et un global), puis générait les prédictions associées à ces modèles. Ainsi, en prenant la moyenne des prédictions, les ensembles sur un même annotateur capturent une partie de l'incertitude épistémique, qui découle des limitations inhérentes du modèle à apprendre à partir des données disponibles. En moyennant sur plusieurs annotateurs, on peut également capturer une partie de l'incertitude aléatoire. Les approches par méthodes d'ensemble sont particulièrement efficaces pour améliorer la robustesse et réduire la variance dans les tâches de segmentation [Ganaie et al., 2022]. Une fois les softmax et prédictions calculés, ils sont automatiquement téléversés sur le S3.

Préparation et récupération des métriques d'incertitude

Le but de ce projet étant de tester l'efficacité de nnU-Net et d'évaluer l'incertitude de sa segmentation à l'aide d'un large panel de métriques, nous avons produit un fichier de calcul de ces métriques en les récupérant depuis les dépôts de CURVAS et de VALUES. Le détail de ces métriques sera évoqué dans la partie suivante. Comme pour les étapes précédentes, pour chaque patient, nous avons téléchargé les prédictions (dont les softmax, nécessaires au calcul des métriques) ainsi que les annotations des experts, puis appliqué notre jeu de métriques aux 13 modèles, avant de transférer les données dans le S3 sous forme de `.csv`. Par ailleurs, nous avons dû mettre en place un système de parallélisation des calculs pour optimiser le temps d'exécution des métriques. Grâce au module `numba`, nous avons ainsi accéléré la durée d'exécution de plus d'un facteur 4.

II.3.1 Schéma résumé

La figure 8 ci-dessous présente un résumé des différentes étapes. On traitera sinon plus qualitativement de ce dernier dans la section Discussion.

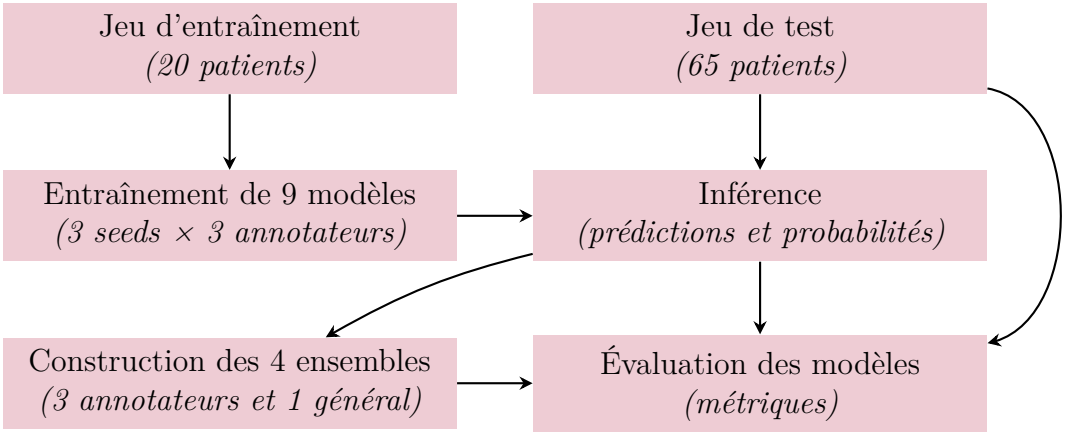


Figure 8 *Diagramme résumant le protocole expérimental*

III Évaluation de l'incertitude

Cette section décrit de manière exhaustive le cadre d'analyse de la précision et de l'incertitude que nous avons utilisé pour évaluer les modèles et ensembles. Pour cela, nous avons recours à un panel de métriques issues du challenge CURVAS (Consensus-based DICE, Confidence, ECE, CRPS) [Riera-Marín et al., 2024] et du framework VaLUES (ACE, AUROC, AURC, EAURC, NCC) [Kahl et al., 2024], ainsi que des mesures classiques de performance telles que l'entropie ou la distance de Hausdorff. Ces métriques permettent de capturer les deux types d'incertitudes, aléatoire et épistémique, ainsi que la performance globale des modèles.

III.1 Métriques de performance

Pour mesurer la performance des modèles, nous mesurons les écarts entre les annotations et les prédictions. Ces écarts sont obtenus selon les métriques suivantes : le consensus-based DICE, le CRPS, et la distance de Hausdorff.

Consensus-based DICE

Le *consensus-based DICE* est un DICE score ordinaire (mesure de la similitude entre deux espaces) mais prenant la zone de consensus (i .e. tous les experts sont d'accord) et la prédiction. Cela permet de tenir compte des variabilités inter-experts (donc de l'incertitude aléatoire). Nous le calculons pour chaque organe et, comme un DICE classique, plus il s'approche de 1, plus la prédiction est correcte. Cette mesure nous permet d'avoir une idée générale de la performance du modèle en tenant compte de la variabilité inter-experts.

$$\text{DICE} = \frac{2|P \cap G|}{|P| + |G|}$$

— P : Segmentation prédite.
— G : Zone de consensus entre annotateurs.
— $|P \cap G|$: Intersection entre les segmentations (i.e. voxels qui se recoupent).
— $|X|$: Nombre total de voxel dans la segmentation X.

CRPS (Continuous Ranked Probability Score)

Le *Continuous Ranked Probability Score* estime la proximité entre le masque des annotations et le masque des probabilités prédites (une probabilité associée à chaque classe pour chaque voxel). Plus précisément il s'agit de la distance quadratique entre la fonction de répartition de la prédiction probabiliste et la fonction de répartition des annotations (qui est une fonction de Heaviside). Le CRPS, contrairement à la plupart des métriques, n'oscille pas entre 0 et 1 mais est un volume (somme des distances, mesurée en mm^3), que l'on souhaite le plus faible possible (faible écart entre annotations et prédiction). Comme le CRPS s'axe sur les probabilités prédites, il offre une perspective

de perfomance complémentaire du DICE, d'où sa présence dans le challenge CURVAS et notre choix de le conserver.

$$\mathrm{CRPS}(F, x) = \int_{-\infty}^{\infty} \left(F(y) - \mathbb{1}_{\{y \geq x\}}\right)^2 dy$$

— $F(y)$: Fonction de répartition des probabilités prédites au point y.

— $\mathbb{1}_{\{y \geq x\}}$: Fonction de répartition (ou de Heaviside) des annotations au seuil x.

Distance de Hausdorff

La *distance de Haussdorff* mesure l'écart le plus grand entre une annotation et une prédiction (soit la plus grande erreur de segmentation possible). Contrairement au DICE et au CRPS, nous la calculons pour chaque annotateur, afin de tenir compte de la variabilité inter-experts. Bien qu'absente des évaluations de CURVAS et de ValUES, cette métrique nous paraissait importante car elle s'intéresse aux erreurs extrêmes de prédiction et est donc plus spécifique.

$$d_H(A, B) = \max\left\{\sup_{a \in A} \inf_{b \in B} \|a - b\|, \sup_{b \in B} \inf_{a \in A} \|b - a\|\right\}$$

— A : Segmentation prédite composée des valeurs des voxels a.

— B : Segmentation par un expert composée des valeurs des voxels b.

III.2 Métriques d'incertitude aléatoire

Comme expliquée plus haut, l'incertitude aléatoire elle est liée aux données d'entrée et notamment à l'ambiguïté des annotations humaines. Les autres sources possibles incluent donc les désaccords inter-annotateurs, le bruit ou les artefacts dans les images, ou des cas atypiques, pathologiques... Pour mesurer l'incertitude aléatoire ici, nous mesurons l'incertitude dans les annotations grâce à la Confiance (ou Uncertainty Assessment) et la NCC.

Confiance (ou Uncertainty Assessment)

L'*Uncertainty Assessment* est justement une mesure de la confiance globale des annotations. C'est la moyenne de la confiance (consensus pour la classe) pour le « Background » C_B (éléments non pris en compte et le « Foreground » C_F (éléments d'intérêt, comme le pancréas ici). Ainsi, plus elle est haute, plus la confiance dans les annotations des experts est élevée. Cette métrique ne dépend pas d'un modèle (et de ses prédictions).

$$C_{\text{seg}} = \frac{(1 - C_B) + C_F}{2}$$

NCC (Normalised Cross Correlation)

La *Normalised Cross Correlation* correspond, comme son nom l'indique, à une variante du coefficient de corrélation. Elle mesure donc, comme une corrélation, la similarité

entre deux annotations de 0 à 1 en valeur absolue (1 indiquant une similarité parfaite), en utilisant leur carte d'incertitude (voir plus loin). Cette métrique permet de capturer la variabilité inter-expert par paires, permettant une plus grand précision que l'Uncertainty Assessment.

$$\mathrm{NCC}(A, B) = \frac{\sum(A - \bar{A})(B - \bar{B})}{\sqrt{\sum(A - \bar{A})^2 \sum(B - \bar{B})^2}}$$

— A : Carte d'incertitude d'une annotation.

— B : Carte d'incertitude d'une autre annotation.

— $\bar{X}$: Moyenne empirique des valeurs de X.

III.3 Métriques d'incertitude épistémique — Calibration

Les métriques d'incertitude épistémique peuvent se décomposer en deux catégories. La première regroupe les métriques traitant des erreurs de *calibration*, c'est-à-dire de décalage entre la confiance d'un modèle dans une prédiction et sa précision réelle. Elles estiment à quel point le modèle est fiable. La deuxième catégorie regroupe quant à elle les métriques traitant de la reconnaissance d'erreur, c'est-à-dire la quantification de la capacité du modèle à reconnaître les erreurs qu'il commet. Décrivons dans un premier temps les métriques de la première catégorie. Nous utilisons l'ECE — proposée par CURVAS — et l'ACE — son pendant proposé par VALUES — décrites plus bas. Si cette dernière a une méthode de calcul très similaire, elle permet une approche plus fine, notamment par le calcul des intervalles qui dépend de la distribution des probabilités prédites.

ECE (Expected Calibration Error)

L'*Expected Calibration Error* se calcule en séparant les probabilités prédites en plusieurs intervalles B_m, de 0 à 1 (appelés *bins*). Puis, au sein de chaque *bin*, on détermine la confiance et la précision moyenne. L'ECE est ainsi la somme pondérée (par la taille de l'intervalle) de la différence entre précision et confiance moyenne pour chaque (d'où le décalage évoqué au-dessus). L'ECE est une mesure phare de la calibration des modèles (elle est d'ailleurs l'unique mesure de calibration du challenge CURVAS et est même directement intégrée à la librairie `torchmetrics`, ce qui facilite son utilisation).

$$\mathrm{ECE} = \sum_{m=1}^{B} \frac{|B_m|}{n} \left|\mathrm{acc}(B_m) - \mathrm{conf}(B_m)\right|$$

— B : Nombre de bins.

— B_m : Le m-ème bin.

— $|B_m|$: Nombre de prédictions dans le bin B_m.

— n : Nombre total de prédictions.

— $\mathrm{acc}(B_m)$: Précision (Proportion de prédictions correctes dans le bin B_m).

— $\mathrm{conf}(B_m)$: Confiance (moyenne des probabilités prédites dans le bin B_m).

— $\frac{|B_m|}{n}$: Poids associés aux bins selon la proportion de prédictions qu'ils contiennent.

ACE (Average Calibration Error)

L'*Average Calibration Error* est calculé de façon identique mais sans la pondération. Les bins sont déterminés de sorte à être de taille égale. ACE comme ECE se situent entre 0 et 1 avec une calibration qui peut être considérée comme correcte en-dessous des 0.1 (ou même 0.05). L'ACE permet un regard plus fin que l'ECE sur la calibration des modèles, notamment en isolant les potentiels effets de la distribution des probabilités prédites.

$$\mathrm{ACE} = \frac{1}{B}\sum_{b=1}^{B} |\mathrm{acc}(b) - \mathrm{conf}(b)|$$

— B : Nombre de bins b (de tailles égales).
— $\mathrm{acc}(B_m)$: Précision (Proportion de prédictions correctes dans le bin B_m).
— $\mathrm{conf}(B_m)$: Confiance (moyenne des probabilités prédites dans le bin B_m).

III.4 Reconnaissance d'erreur (Failure Detection)

La reconnaissance d'erreur peut se mesurer à l'aide de métriques comme l'AUROC, l'AURC et l'EAURC.

AUROC (Area Under the Receiver Operating Characteristic curve)

L'*Area Under the Receiver Operating Characteristic curve* est une mesure de l'aire sous la courbe ROC (Taux de vrais positifs en fonction du taux de faux positifs). Plus précisément, cette courbe représente le rapport en vrais et faux positifs pour différents seuils (valeur à partir de laquelle on considère qu'une probabilité renvoie une valeur positive). Ainsi, une courbe ROC idéale serait dans le coin supérieur gauche d'un axe orthonormé (100 % de vrais positifs et 0 % de faux positifs) et donc son aire (AUROC) serait égale à 1. Cela signifie donc que le modèle distingue parfaitement les valeurs positives des négatives (absence d'organe). On voit ici l'intérêt de l'ajout du framework ValUES avec des métriques qui capturent d'autres causes de l'incertitude épistémique (en l'occurrence, liée à l'identification des erreurs).

$$\mathrm{AUROC} = \int_0^1 \mathrm{TPR}(t)\, d\mathrm{FPR}(t)$$

— $\mathrm{TPR}(t)$: Taux de vrais positifs au seuil t.
— $\mathrm{FPR}(t)$: Taux de faux positifs au seuil t.

AURC (Area Under the Risk Curve)

L'*Area Under the Risk Curve* est aussi une aire mais d'une courbe de risque. Elle est tracée en prenant pour chaque probabilité prédite, le risque associé (ici, la différence entre annotation et prédiction). Chaque point correspond ainsi à un seuil de confiance (comme pour la ROC curve, on calcule le risque à différents seuils). Plus l'aire est faible, plus le risque est faible à chaque seuil et plus le modèle prédit avec une faible erreur. L'AURC complète l'AUROC en insistant sur l'arbitrage risque/qualité ; un autre apport bénéfique de ValUES.

$$\text{Risk}(t) = \frac{1}{N}\sum_{i=1}^{N} |y_i - \hat{y}_i(t)|\,, \qquad \text{AURC} = \int_0^1 \text{Risk}(t)\,dt$$

— y_i : La valeur issue de l'annotation.

— $\hat{y}_i(t)$: La valeur prédite au seuil t.

— N : Nombre de voxels de l'image.

EAURC (Expected Area Under the Risk Curve)

L'*Expected Area Under the Risk Curve* est une version normalisée de l'AURC. Il s'agit de la différence entre l'AURC et l'AURC optimale (c'est-à-dire avec une répartition optimale des risques à chaque seuil). L'interprétation est sensiblement la même à cela près que l'EAURC est moins influencée par les erreurs récurrentes ou extrêmes.

$$\text{EAURC} = \frac{1}{A^*}\int_0^1 \text{Risk}(t)\,dt - \int_0^1 \text{Risk}^*(t)\,dt$$

— A^* : Aire sous la courbe de risque optimale.

— $\text{Risk}^*(t)$: Le risque optimal au seuil t.

III.5 Entropie et cartes d'incertitudes

Comme on l'a mentionné en introduction, il est possible de générer des cartes d'incertitudes indiquant les points (ici les voxels) sur lesquels le modèle est le moins « confiant » dans sa prédiction. Ici, on peut associer à chaque voxel v un vecteur mesure de probabilité $p_v = (p_1, p_2, p_3, p_4)^\top$, tel que p_1 est la probabilité que ce voxel soit dans le fond, p_1 que ce soit le pancréas, p_2 les reins et p_3 le foie. Dans le cas où $p_1 = p_2 = p_3 = p_4 = 0,25$, la confiance est au plus bas (on ne peut distinguer une classe des autres). Dans le cas où $p_i = 1$ et $p_{j\neq i} = 0$, elle est au plus haut. On peut alors utiliser l'entropie pour caractériser cette confiance. L'*entropie de Shannon* est une mesure assez générale de l'incertitude d'une segmentation. Elle l'établit en utilisant les probabilités prédites pour chaque classe des valeurs des voxels :

$$\text{H}(p) = -\sum_i p_i \log p_i$$

— p : Vecteur de probabilités contenant les probabilités p_i associées à chaque classe.

L'entropie peut être calculée à l'échelle de l'image, ou pour chaque voxel, ce qui permet notamment de générer des cartes d'incertitudes comme en figure 9 ci-dessous.

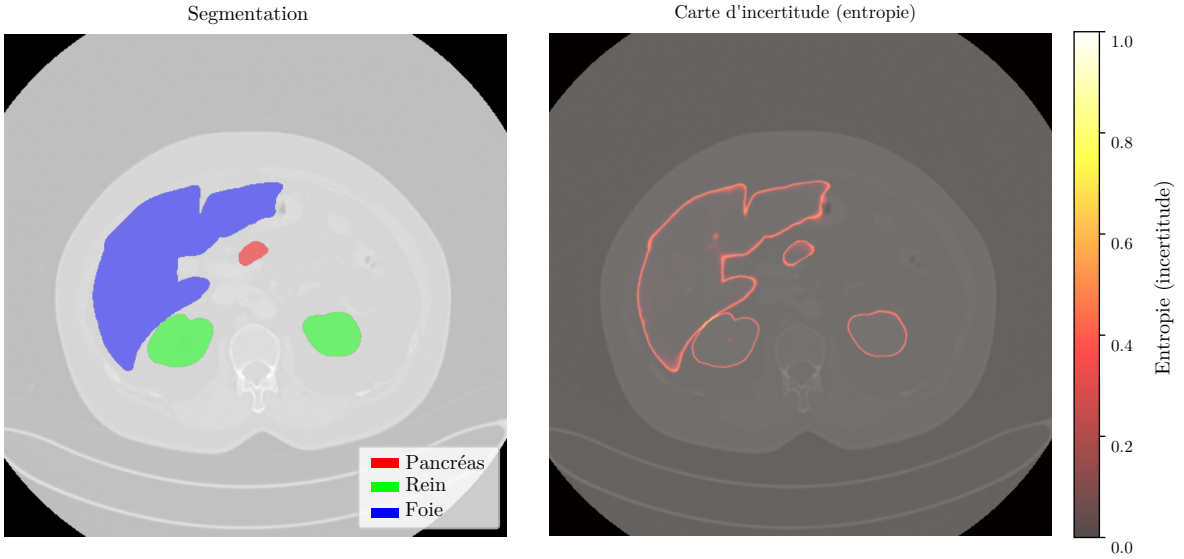


Figure 9 *Carte d'incertitude (à droite) d'une coupe de la segmentation des organes sur le patient 3, selon l'entropie de* SHANNON *des probabilités de sortie du modèle d'ensemble général (sur les 9 modèles initiaux)*

IV Résultats

Dans le long chemin qui mène des données du challenge CURVAS à l'évaluation de la qualité des prédictions de nos différents modèles, certaines parties du projet n'ont pas pu aboutir. La première est l'inférence, qui n'a pas pu être menée à bien pour quatre patients du *test set* renvoyant l'erreur "seg fault" (erreur de segmentation), que nous n'avons pas réussi à corriger. Nous pensons qu'il est probable que cette erreur provienne de l'optimisation du temps de calcul avec `numba`. Nous avons au final traité 61 patients du *test set* sur 65.

Une erreur plus importante s'est glissée dans le code de nos métriques, que nous avons repérée — hélas — une fois toutes les métriques calculées (donc impossible à corriger dans les temps). La métrique en question est la NCC, pour laquelle notre méthode de calcul des cartes d'incertitude est incorrecte. En effet, plutôt que d'être composées de probabilités, elles sont constituées des valeurs des classes dans les annotations experts. Il aurait sans doute fallu passer par une fonction type HEAVYSIDE pour les convertir en probabilités. Fort heureusement, les métriques étant indépendantes les unes des autres, cela n'a pas affecté le reste de nos résultats.

Par ailleurs, plusieurs points affichent des valeurs extrêmes, ce qui nous a quelque peu surpris (DICE nuls, valeurs d'ECE plus importantes pour les ensembles que pour les modèles individuels). Il est fortement possible qu'il y ait eu des erreurs de calcul ou d'approximation dans certains cas, pour certains modèles et certains patients. Néanmoins, nous avons conservé ces valeurs, car les résultats n'en sont pas significativement affectés.

IV.1 Réduction de l'incertitude

Nos résultats indiquent clairement que les modèles d'ensemble permettent de réduire l'incertitude. La figure 10 ci-dessous montre les graphiques « violon » indiquant la distribution de l'ECE moyen (moyenne des ECE sur les 3 annotateurs), accompagnés d'une boîte à moustaches classique. Elle montre chaque point de la distribution, correspondant à chaque patient et modèle (3 fois plus pour les modèles individuels que pour les ensembles par annotateur, et 9 fois plus que pour l'ensemble général).

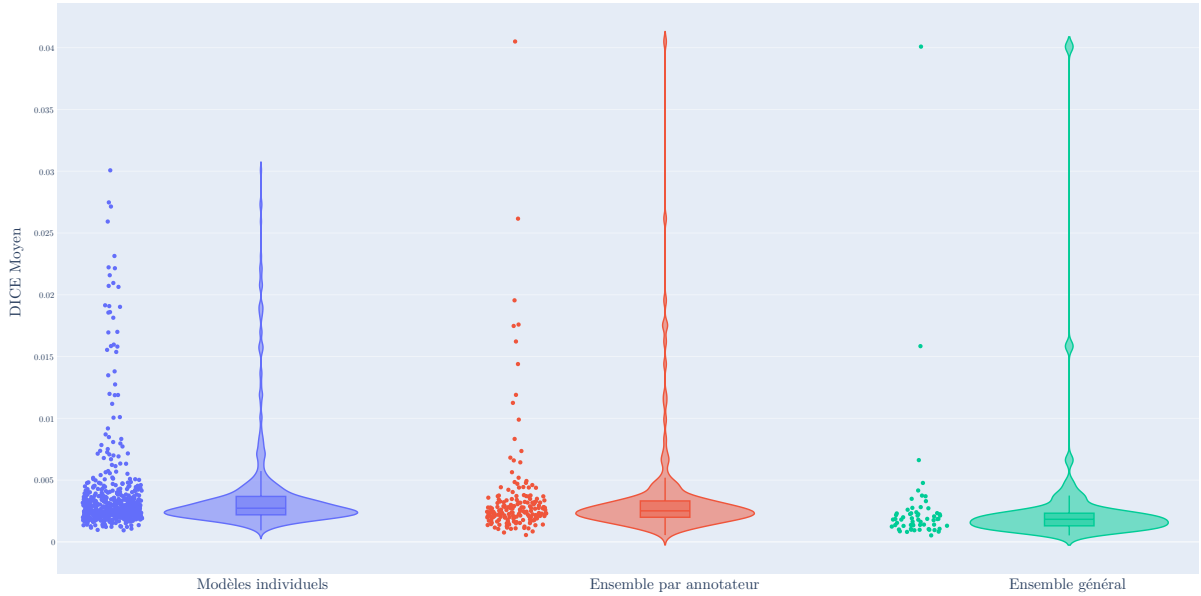


Figure 10 *ECE moyen pour les différents modèles*

On observe, au-delà de la réduction du nombre de points, une forte diminution du nombre de valeurs extrêmes. Par ailleurs, moyennes et médianes diminuent, ce qui indique très clairement une réduction de l'incertitude. On peut tester statistiquement la significativité de ce résultat : on utilise pour cela un test U de Mann–Whitney [Mann & Whitney, 1947] unilatéral (alternative « less ») entre chaque paire de groupes :

Soient n_i et n_j les tailles respectives des deux groupes, et R_i la somme des rangs des observations du groupe i après avoir rangé conjointement tous les ECE_{mean}. On définit

$$U_i = R_i - \frac{n_i(n_i+1)}{2} \quad , \quad U = \min(U_i, U_j)$$

Sous l'hypothèse nulle (H_0 : distributions identiques), U suit asymptotiquement

$$\mu_U = \frac{n_i\, n_j}{2} \quad , \quad \sigma_U^2 = \frac{n_i\, n_j\, (n_i + n_j + 1)}{12},$$

et on calcule le score normalisé $z = \dfrac{U - \mu_U}{\sigma_U}$ avec $Z \sim \mathcal{N}(0,1)$, puis la p-valeur unilatérale $p = P(Z \leq z)$. Enfin, pour les $k = 3$ comparaisons paires, on applique la correction de Bonferroni

$$p_{\text{corr}} = \min(p \times k,\, 1)\,,$$

et on considère $p_{\text{corr}} < 0.05$ comme significatif.

On obtient, comme indiqué en table 1 ci-dessous, que cette diminue de l'ECE est signifiicative.

	U-stat	p (brut)	p_{corr}	Significatif ?
Ens. général < Ens. par annotateur	2222.0	5.36×10^{-7}	1.61×10^{-6}	✓
Ens. général < Modèles individuels	5215.0	3.30×10^{-12}	9.90×10^{-12}	✓
Ens. par annotateur < Modèles individuels	31561.0	2.21×10^{-3}	6.63×10^{-3}	✓

Table 1 *Résultats du test de Mann–Whitney pour l'ECE moyen*

L'AURC et l'EAURC sont très similaires dans leurs distributions (voir figure 13 ci-dessous), ce qui signifie que nos modèles se rapprochent fortement d'un modèle avec un arbitrage risque/qualité parfait. De même, les différences entre modèles sont assez faibles, à cela près que les modèles d'ensemble présentent une plus faible dispersion, et donc une plus grande constance dans leurs résultats. Pour autant, cette concentration de la distribution fait aussi que, pour certains patients, certains modèles initialisés aléatoirement sont légèrement plus performants que les modèles d'ensemble. On observe par ailleurs toujours les différences entre les organes, avec le foie en tête : peu de risques d'erreurs pour une bonne qualité de prédiction. Malgré ces différences, les scores restent globalement très bons et indiquent une bonne gestion de l'incertitude par les modèles.

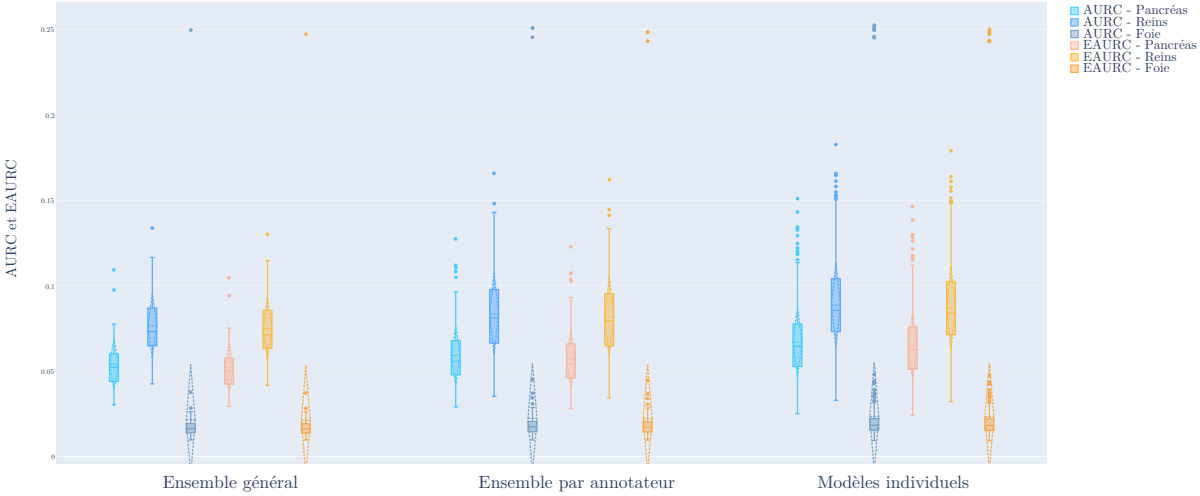


Figure 11 *Mesures AURC et EAURC sur les modèles*

IV.2 Performance de prédiction

Concernant la performance de prédiction, on observe encore une fois la supériorité des modèles d'ensemble sur les autres modèles. Le DICE moyen sur les trois organes de l'ensemble global est supérieur à celui des ensembles par annotateurs, lui-même supérieur à celui des modèles individuels. Cependant, ce résultat n'est pas statistiquement significatif (on manque probablement d'observations), après un test de Mann–Whitney unilatéral (alternative « more »). Voir la figure 12 et la table 2 ci-dessous.On retiendra que les stratégies d'ensemble ne détériorent pas la performance de prédiction et ont une légère tendance à l'améliorer.

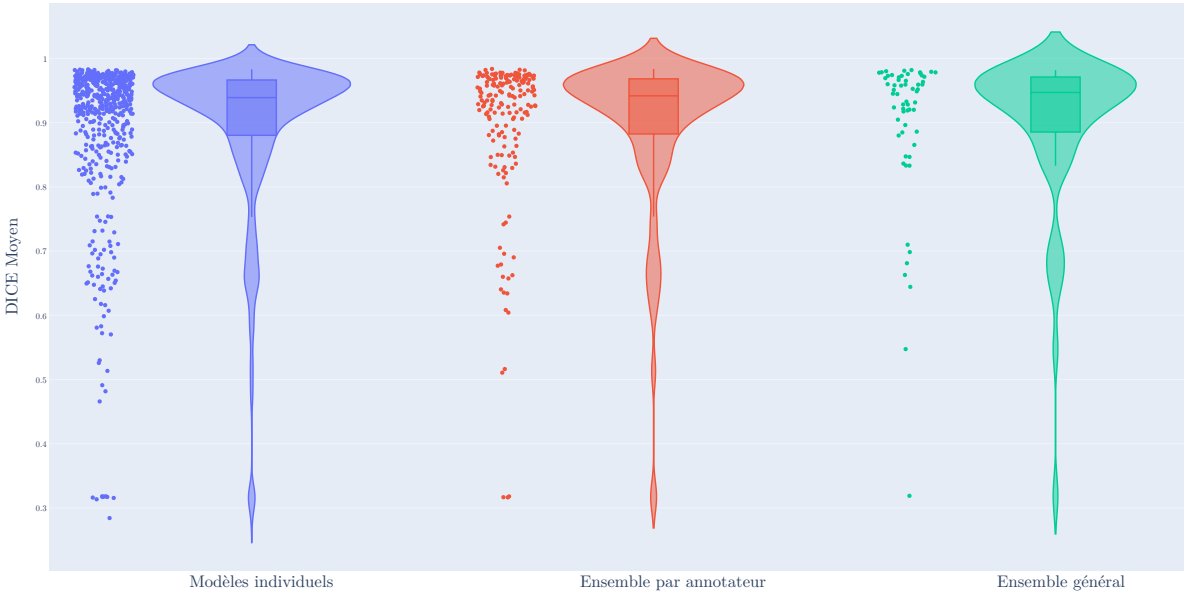


Figure 12 *DICE moyen pour les différents modèles*

	U-stat	p (brut)	p_{corr}	Significatif ?
Ens. général > Ens. par annotateur	5701.0	2.59×10^{-1}	7.78×10^{-1}	✗
Ens. général > Modèles individuels	17790.0	1.54×10^{-1}	4.62×10^{-1}	✗
Ens. par annotateur > Modèles individuels	50979.5	2.61×10^{-1}	7.83×10^{-1}	✗

Table 2 *Résultats du test de Mann–Whitney pour le DICE moyen*

La distribution des distances d'Hausdorff se concentre autour de 0 (moyenne entre 25 et 50). Si on observe une concentration similaire pour les autres modèles, la dispersion est bien plus grande avec davantage de valeurs extrêmes (plus de 400) donc davantage de grands écarts de prédiction.

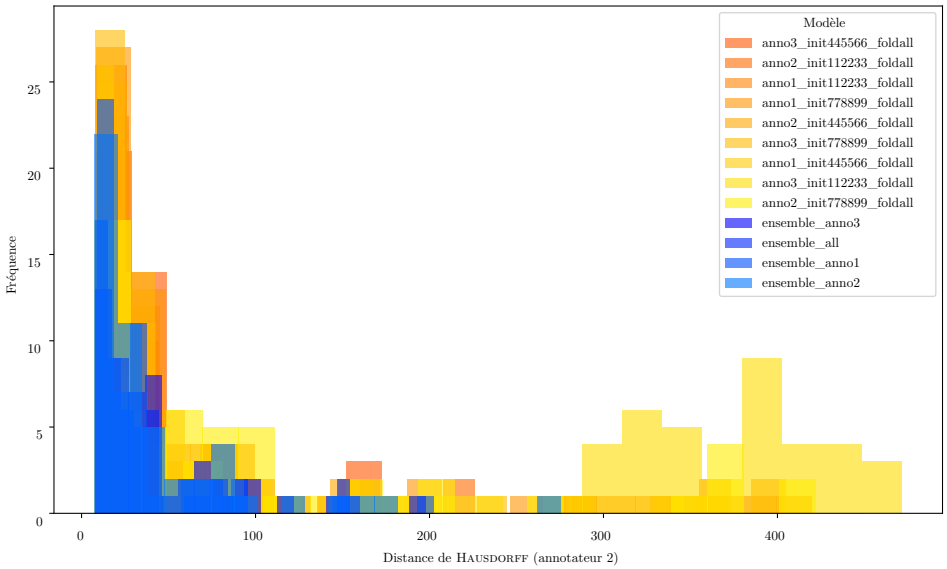


Figure 13 *Distance de Haussdorf entre les prédictions des modèles et l'annotateur 2*

Nous nous sommes également intéressés à un potentiel lien entre qualité de la prédiction (DICE) et incertitude (entropie prédictive). La figure 14 ci-dessous présente l'entropie prédictive en fonction du score DICE moyen.

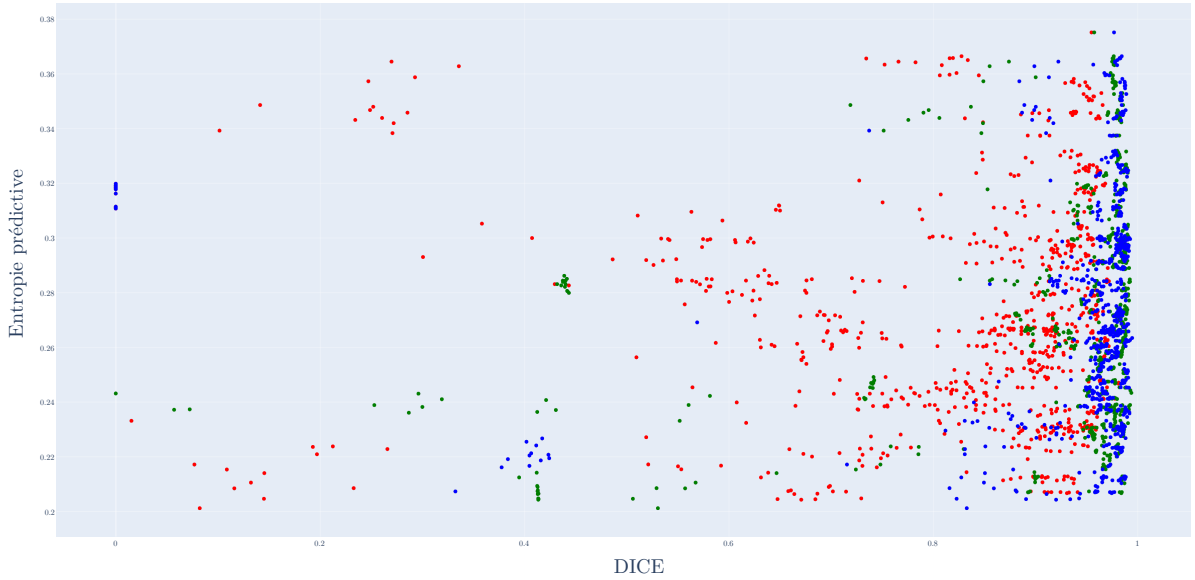


Figure 14 *Qualité de prédiction et incertitude*

Comme on peut le voir, il n'y a pas de lien clair entre ces deux variables. Le coefficient de corrélation de Pearson vaut 0.41, et l'information mutuelle 0.74, cependant le R^2 est négatif et vaut -4.2. Ce résultat n'est pas forcément surprenant : prédire mieux ne signifie pas nécessairement être plus certain.

V Discussion

Les méthodes d'ensemble ont permis de garantir une meilleure moyenne et médiane des métriques, ainsi qu'une forte réduction de la variance. Ce résultat semble montrer qu'il s'agit d'une méthode très générale, dépassant à la fois le cadre des modèles d'apprentissage classiques (forêts aléatoires, descentes de gradients boostées, etc.…) et s'appliquant à l'apprentissage profond. Elle présente par ailleurs des intérêts allant au-delà de l'amélioration de la performance et de la précision — ici, notamment, sur la réduction de l'incertitude.

V.1 Comparaison avec le benchmark du challenge CURVAS

En mai 2025, les résultats du challenge CURVAS ont été publiés [Riera-Marin et al., 2025]. Sept équipes professionnelles se sont affrontées avec différentes méthodes pour réduire l'incertitude, par exemple en utilisant des techniques variées d'agrégation des données ou en adoptant des architectures de réseaux de neurones différentes. Voici les principaux résultats concernant les meilleures performances atteintes sur certaines métriques, en comparaison avec nos performances.

	Métriques (valeurs moyennes)			
	DSC (%)	Confiance (%)	ECE ($\times 10^{-3}$)	CRPS (cm^3)
MedIG	**94.57 (1)**	**97.87 (1)**	**1.82 (2)**	**8.108 (1)**
PrAEcision	**93.29 (2)**	**97.18 (2)**	**2.22 (3)**	**10.438 (3)**
BreizhSeg	92.60 (4)	**97.17 (3)**	**1.61 (1)**	12.326 (4)
DLAI	**92.72 (3)**	96.23 (4)	3.90 (4)	12.625 (5)
BCNAIM	90.52 (5)	95.88 (5)	6.21 (6)	**9.727 (2)**
CAI4CAI	84.98 (7)	92.10 (7)	4.48 (5)	12.828 (6)
PredictED	85.79 (6)	92.39 (6)	6.64 (7)	25.895 (7)
Modèles individuels	89,36	96,30	3,90	19,77
Ensembles par annotateur	89,62	96,26	3,57	19,33
Ensemble général	89,93	96,26	2,83	16,64

Table 3 *Comparaison entre les performances des équipes participant au challenge CURVAS 2025 et celles de nos modèles.*

Si nous avions participé à cette compétition, nous aurions été classés quatrièmes pour la confiance et pour l'ECE. Ces résultats sont encourageants pour la technique de l'ensembling général car, par rapport aux équipes professionnelles, nous disposions de moins de temps, de ressources et de données d'entraînement. De plus, nous n'avons pas entraîné nos modèles sur le jeu de validation, qui est composé de 5 patients, alors que le jeu d'entraînement en contient 20. Faute de temps, nous n'avons pas non plus utilisé l'option de *cross-validation* gérée automatiquement par `nnU-Net`. Avec plus de ressources, nous aurions souhaité entraîner des réseaux U-Net à encodeurs résiduels (*Residual Encoder U-Net*, ou *ResEnc*), suivant des travaux réalisés en 2024 sur le sujet [Isensee et al., 2024].

V.2 Envergure du projet

Le protocole expérimental du projet est très ambitieux, d'une part en raison de la complexité des outils manipulés, et d'autre part du coût logistique très important. En effet, nous avons dû manipuler des réseaux de neurones avancés (U-Net), et donc également nous familiariser avec la littérature existante. L'écart théorique avec les cours de l'ENSAE était conséquent, d'autant plus que nous n'avions pas encore eu de cours d'apprentissage machine — lesquels n'abordent pas forcément dans le détail les notions d'apprentissage profond nécessaires — à ce moment-là de l'année.

De plus, les différents dépôts nous ont demandé un temps d'acclimatation important pour bien appréhender leurs fonctionnements et leurs commandes. L'exploitation du dépôt VALUES était particulièrement complexe, en raison de la syntaxe dépréciée de certaines fonctions. Comme le montre la table 4 récapitulative ci-dessous, les coûts en temps, en stockage et en bande passante sont très conséquents.

Sur l'ensemble du projet, nous cumulons 1 080 heures de temps de calcul et de téléchargement (soit 45 jours), et ce sans tenir compte des nombreux essais et échecs rencontrés à toutes les étapes. En effet, un entraînement de modèle dure une journée (9 modèles), l'inférence dure une heure par patient et par modèle (64 patients), et la méthode d'ensemble, deux heures par patient. Sans compter le calcul des métriques, d'une durée approximative d'une heure par patient (selon la vitesse de téléchargement des fichiers).

Les fichiers stockés sont très volumineux, notamment les probabilités de prédiction (3 Go) et très nombreux. Notre stockage S3 atteignait 2,9 To de modèles entraînés, obtenus par méthodes d'ensemble, et de résultats prédits à la fin du projet. En somme, avec les ressources disponibles, le projet fut très difficile et, sans l'accès aux instances du Genes et de l'Insee (plateformes Onyxia), il aurait été impossible.

	Qte	Par unité			Total		
		Durée (h)	Transfert ↓ (Go)	↑ (Go)	Durée (h)	↓ (Go)	↑ (Go)
Entraînement	9	24	6	1	216	54	9
Inférence	576	1	1.5	4	576	864	2304
Ensemble	64	2	18	16	128	1152	1024
Évaluation	64	1	3.5	0	64	224	0
		Total			**984**	**2294**	**3337**

Table 4 *Coûts en temps, en transferts descendant et montant (donc stockage)*

Nous pouvons, à ce propos, dresser une estimation rapide du bilan financier pour ce projet. Pour l'utilisation des GPU, en prenant comme prix horaire de référence 0,50 € (location de modèles T4), il aurait fallu débourser 492,00 € pour atteindre les 984 heures de calcul. Pour le stockage, en admettant que le projet est réalisé en une semaine, et en prenant un coût de 0,024 € HT par Go (prix de référence au 15 mai 2025 chez Amazon Web Services, service S3, sur le datacenter de Paris) par mois, nous ajoutons 20,80 € au total. Enfin, en comptant les coûts de transfert des données (0,09 € HT par Go, soit un total de 247,70 € sur 2,294 To), on obtient un total de 760,50 € pour ce projet.

Il s'agit là d'une approximation : des ressources plus puissantes auraient permis de faire plusieurs étapes en une seule fois, ce qui aurait économisé une quantité importante de ressources, et donc de fonds.

Ainsi, comme souvent dans le domaine des statistiques appliquées, l'idéal théorique d'une expérimentation exhaustive se voit limité par des contraintes techniques : limites de mémoire, quotas de calcul et efficacité du stockage. Malgré ces contraintes, nous avons réussi à proposer des résultats probants, et surtout à appliquer de nombreuses métriques d'incertitude, enjeu principal de ce projet.

VI Conclusion

En conclusion, nos résultats montrent bien l'efficacité des méthodes d'ensemble pour la prédiction de segmentations médicales. En effet, grâce aux métriques d'incertitude CURVAS et ValUES, nous observons clairement que non seulement les quatre modèles d'ensemble sont plus performants en général, mais aussi qu'ils sont mieux calibrés et plus confiants dans leurs prédictions.

Nous tenons à remercier nos encadrants StatApp Tristan Kirscher et Xavier Coubez, qui ont su nous aiguiller dans ce projet ambitieux et nous permettre d'obtenir tous ces résultats. De même, nous remercions l'Insee et le groupe Genes pour l'accès au sspcloud, sans lequel tout ceci aurait été matériellement impossible.

A Liste des figures, tables, liens

Liste des figures

Liste des tables

Liens des jeux de données et du code

— Dépôt GitHub du projet : https://github.com/Kirscher/statapp_2025_curvas/tree/main
— Site web CURVAS : https://curvas.grand-challenge.org/curvas/
— Dépôt GitHub du défi CURVAS : https://github.com/SYCAI-Technologies/curvas-challenge
— Dépôt GitHub ValUES : https://github.com/IML-DKFZ/values
— Dépôt GitHub nnU-Net : https://github.com/MIC-DKFZ/nnU-Net

Moyennes des métriques

Modèle	DICE	CONF	ECE	ACE	AUROC	AURC	EAURC	CRPS	EntGT	EntPred	Hausdorff
annotateur 1, seed 1	$8{,}97 \cdot 10^{-1}$	$9{,}61 \cdot 10^{-1}$	$3{,}85 \cdot 10^{-3}$	$8{,}71 \cdot 10^{-2}$	$7{,}66 \cdot 10^{-1}$	$6{,}56 \cdot 10^{-2}$	$6{,}43 \cdot 10^{-2}$	$2{,}05 \cdot 10^{4}$	$2{,}76 \cdot 10^{-1}$	$2{,}68 \cdot 10^{-1}$	$7{,}55 \cdot 10^{1}$
annotateur 1, seed 2	$8{,}94 \cdot 10^{-1}$	$9{,}60 \cdot 10^{-1}$	$3{,}73 \cdot 10^{-3}$	$8{,}88 \cdot 10^{-2}$	$7{,}30 \cdot 10^{-1}$	$5{,}69 \cdot 10^{-2}$	$5{,}56 \cdot 10^{-2}$	$1{,}98 \cdot 10^{4}$	$2{,}76 \cdot 10^{-1}$	$2{,}67 \cdot 10^{-1}$	$4{,}02 \cdot 10^{1}$
annotateur 1, seed 3	$8{,}94 \cdot 10^{-1}$	$9{,}61 \cdot 10^{-1}$	$3{,}70 \cdot 10^{-3}$	$8{,}73 \cdot 10^{-2}$	$8{,}20 \cdot 10^{-1}$	$7{,}64 \cdot 10^{-2}$	$7{,}51 \cdot 10^{-2}$	$2{,}03 \cdot 10^{4}$	$2{,}76 \cdot 10^{-1}$	$2{,}67 \cdot 10^{-1}$	$6{,}15 \cdot 10^{1}$
annotateur 2, seed 1	$9{,}02 \cdot 10^{-1}$	$9{,}68 \cdot 10^{-1}$	$3{,}92 \cdot 10^{-3}$	$9{,}25 \cdot 10^{-2}$	$7{,}44 \cdot 10^{-1}$	$6{,}09 \cdot 10^{-2}$	$5{,}95 \cdot 10^{-2}$	$1{,}73 \cdot 10^{4}$	$2{,}76 \cdot 10^{-1}$	$2{,}74 \cdot 10^{-1}$	$6{,}23 \cdot 10^{1}$
annotateur 2, seed 1	$9{,}06 \cdot 10^{-1}$	$9{,}71 \cdot 10^{-1}$	$3{,}58 \cdot 10^{-3}$	$8{,}70 \cdot 10^{-2}$	$7{,}47 \cdot 10^{-1}$	$5{,}95 \cdot 10^{-2}$	$5{,}82 \cdot 10^{-2}$	$1{,}62 \cdot 10^{4}$	$2{,}76 \cdot 10^{-1}$	$2{,}74 \cdot 10^{-1}$	$5{,}29 \cdot 10^{1}$
annotateur 2, seed 2	$8{,}84 \cdot 10^{-1}$	$9{,}59 \cdot 10^{-1}$	$4{,}66 \cdot 10^{-3}$	$9{,}24 \cdot 10^{-2}$	$7{,}31 \cdot 10^{-1}$	$5{,}80 \cdot 10^{-2}$	$5{,}66 \cdot 10^{-2}$	$2{,}31 \cdot 10^{4}$	$2{,}76 \cdot 10^{-1}$	$2{,}77 \cdot 10^{-1}$	$1{,}09 \cdot 10^{2}$
annotateur 3, seed 1	$8{,}65 \cdot 10^{-1}$	$9{,}52 \cdot 10^{-1}$	$5{,}09 \cdot 10^{-3}$	$7{,}86 \cdot 10^{-2}$	$6{,}58 \cdot 10^{-1}$	$4{,}71 \cdot 10^{-2}$	$4{,}57 \cdot 10^{-2}$	$2{,}40 \cdot 10^{4}$	$2{,}76 \cdot 10^{-1}$	$2{,}76 \cdot 10^{-1}$	$2{,}91 \cdot 10^{2}$
annotateur 3, seed 2	$9{,}01 \cdot 10^{-1}$	$9{,}68 \cdot 10^{-1}$	$3{,}19 \cdot 10^{-3}$	$7{,}64 \cdot 10^{-2}$	$7{,}74 \cdot 10^{-1}$	$6{,}39 \cdot 10^{-2}$	$6{,}26 \cdot 10^{-2}$	$1{,}82 \cdot 10^{4}$	$2{,}76 \cdot 10^{-1}$	$2{,}74 \cdot 10^{-1}$	$7{,}92 \cdot 10^{1}$
annotateur 3, seed 3	$8{,}98 \cdot 10^{-1}$	$9{,}67 \cdot 10^{-1}$	$3{,}35 \cdot 10^{-3}$	$7{,}52 \cdot 10^{-2}$	$7{,}90 \cdot 10^{-1}$	$6{,}98 \cdot 10^{-2}$	$6{,}85 \cdot 10^{-2}$	$1{,}86 \cdot 10^{4}$	$2{,}76 \cdot 10^{-1}$	$2{,}74 \cdot 10^{-1}$	$5{,}24 \cdot 10^{1}$
ens. annotateur 1	$8{,}94 \cdot 10^{-1}$	$9{,}60 \cdot 10^{-1}$	$3{,}59 \cdot 10^{-3}$	$7{,}57 \cdot 10^{-2}$	$7{,}69 \cdot 10^{-1}$	$6{,}58 \cdot 10^{-2}$	$6{,}45 \cdot 10^{-2}$	$1{,}97 \cdot 10^{4}$	$2{,}76 \cdot 10^{-1}$	$2{,}67 \cdot 10^{-1}$	$4{,}16 \cdot 10^{1}$
ens. annotateur 2	$9{,}00 \cdot 10^{-1}$	$9{,}66 \cdot 10^{-1}$	$3{,}87 \cdot 10^{-3}$	$7{,}24 \cdot 10^{-2}$	$7{,}24 \cdot 10^{-1}$	$5{,}65 \cdot 10^{-2}$	$5{,}52 \cdot 10^{-2}$	$1{,}75 \cdot 10^{4}$	$2{,}76 \cdot 10^{-1}$	$2{,}74 \cdot 10^{-1}$	$5{,}17 \cdot 10^{1}$
ens. annotateur 3	$8{,}94 \cdot 10^{-1}$	$9{,}62 \cdot 10^{-1}$	$3{,}24 \cdot 10^{-3}$	$5{,}78 \cdot 10^{-2}$	$6{,}98 \cdot 10^{-1}$	$5{,}05 \cdot 10^{-2}$	$4{,}92 \cdot 10^{-2}$	$2{,}09 \cdot 10^{4}$	$2{,}76 \cdot 10^{-1}$	$2{,}73 \cdot 10^{-1}$	$5{,}09 \cdot 10^{1}$
ens. général	$8{,}99 \cdot 10^{-1}$	$9{,}63 \cdot 10^{-1}$	$2{,}83 \cdot 10^{-3}$	$5{,}16 \cdot 10^{-2}$	$7{,}17 \cdot 10^{-1}$	$5{,}34 \cdot 10^{-2}$	$5{,}20 \cdot 10^{-2}$	$1{,}66 \cdot 10^{4}$	$2{,}76 \cdot 10^{-1}$	$2{,}71 \cdot 10^{-1}$	$4{,}19 \cdot 10^{1}$

Table 5 *Résumé des métriques moyennes par modèle.*

B Bibliographie

Ganaie, M., Hu, M., Malik, A., Tanveer, M., & Suganthan, P. (2022). Ensemble deep learning : A review. *Engineering Applications of Artificial Intelligence*, *115*, 105151. https://doi.org/10.1016/j.engappai.2022.105151

Isensee, F., Petersen, J., Klein, A., Zimmerer, D., Jaeger, P. F., Kohl, S., Wasserthal, J., Koehler, G., Norajitra, T., Wirkert, S., & Maier-Hein, K. H. (2018). nnU-Net : Self-adapting Framework for U-Net-Based Medical Image Segmentation. https://arxiv.org/abs/1809.10486

Isensee, F., Wald, T., Ulrich, C., Baumgartner, M., Roy, S., Maier-Hein, K., & Jaeger, P. F. (2024). nnU-Net Revisited : A Call for Rigorous Validation in 3D Medical Image Segmentation. https://arxiv.org/abs/2404.09556

Kahl, K.-C., Lüth, C. T., Zenk, M., Maier-Hein, K., & Jaeger, P. F. (2024). ValUES : A Framework for Systematic Validation of Uncertainty Estimation in Semantic Segmentation. https://arxiv.org/abs/2401.08501

Kendall, A., & Gal, Y. (2017). What Uncertainties Do We Need in Bayesian Deep Learning for Computer Vision ? https://arxiv.org/abs/1703.04977

Li, M., Jiang, Y., Zhang, Y., & Zhu, H. (2023). Medical image analysis using deep learning algorithms. *Frontiers in Public Health*, *11*. https://doi.org/10.3389/fpubh.2023.1273253

Mann, H. B., & Whitney, D. R. (1947). On a Test of Whether one of Two Random Variables is Stochastically Larger than the Other. *The Annals of Mathematical Statistics*, *18*(1), 50-60. https://doi.org/10.1214/aoms/1177730491

Riera-Marin, M., K, S. O., Rodriguez-Comas, J., May, M. S., Pan, Z., Zhou, X., Liang, X., Erick, F. X., Prenner, A., Hemon, C., Boussot, V., Dillenseger, J.-L., Nunes, J.-C., Qayyum, A., Mazher, M., Niederer, S. A., Kushibar, K., Martin-Isla, C., Radeva, P., … Galdran, A. (2025). Calibration and Uncertainty for multiRater Volume Assessment in multiorgan Segmentation (CURVAS) challenge results. https://arxiv.org/abs/2505.08685

Riera-Marín, M., Kleiss, J.-M., Aubanell, A., & Antolín, A. (2024). *CURVAS dataset* (Version v1.0.1). Zenodo. https://doi.org/10.5281/zenodo.12687192

Ronneberger, O., Fischer, P., & Brox, T. (2015). U-Net : Convolutional Networks for Biomedical Image Segmentation. https://arxiv.org/abs/1505.04597

Sarvamangala, D. R., & Kulkarni, R. V. (2022). Convolutional neural networks in medical image understanding : a survey [Epub 2021 Jan 3]. *Evolutionary Intelligence*, *15*(1), 1-22. https://doi.org/10.1007/s12065-020-00540-3

Smerkous, D., Bai, Q., & Li, F. (2024). Enhancing Diversity in Bayesian Deep Learning via Hyperspherical Energy Minimization of CKA. https://arxiv.org/abs/2411.00259

C Annexe

C.1 Précisions sur l'incertitude aléatoire et épistémique

Incertitude aléatoire : Plusieurs prédictions plausibles pour un échantillon dues à l'ambiguïté ou à d'autres facteurs sont couramment attribuées à l'AU et conduisent donc à l'hypothèse que la variable de variabilité Z capture essentiellement l'AU apprise du modèle de prédiction. Par conséquent, l'information mutuelle entre l'étiquette de classe Y et la variable de variabilité Z étant donné un échantillon x décrit combien d'informations sur l'AU pourraient être obtenues en obtenant l'étiquette de classe y.

$$\mathrm{MI}(Y, Z|x) = H(Y|x) - \mathbb{E}_{z\sim Z}[H(Y|x, z)]$$

Incertitude épistémique : Suivant la notion qu'il n'y a aucune raison pour qu'un modèle de prédiction de variable de variabilité soit jamais incertain de sa prédiction sur des données i.i.d. s'il dépend encore de la variable de variabilité $p(Y|x, z)$. Par conséquent, l'incertitude du classificateur $H(Y|x)$ qui ne peut pas être attribuée à la variable de variabilité Z devrait être nouvelle et précédemment inconnue (par le modèle de prédiction). Suivant ce raisonnement, nous émettons l'hypothèse que l'entropie attendue des modèles de variable de variabilité EU.

$$\mathbb{E}_{z\sim Z}[H(Y|x, z)] = H(Y|x) - \mathrm{MI}(Y, Z|x)$$

C.1.1 Framework ValUES

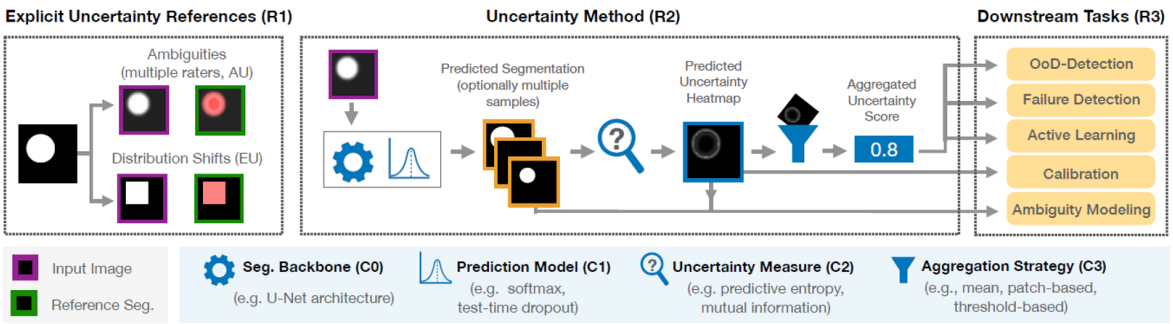


Figure 15 *Architecture ValUES*

Ce cadre est constitué de 3 parties : R1, R2 et R3.

— **R1 :** Il évalue les méthodes d'incertitude prétendant séparer l'Incertitude Aléatoire et l'Incertitude Épistémique au moyen de références explicites, de métriques et de jeux de données de test.

— **R2 :** Il évalue les méthodes d'incertitude concernant tous les composants d'une méthode d'incertitude. Afin d'évaluer les capacités d'une méthode d'incertitude, il est crucial de retracer les améliorations à ses composants individuels C0, C1, C2 et C3.

- **C0 :** Appelé le backbone de segmentation, il s'agit de l'architecture utilisée pour la segmentation sémantique. Dans notre cas, nous utilisons l'architecture nnU-Net.
- **C1 :** Le modèle de prédiction est le modèle obtenu après l'entraînement des neurones de l'architecture sélectionnée.
- **C2 :** La mesure d'incertitude implique le calcul d'un score d'incertitude par voxel basé sur les scores de classe prédits, qui peuvent être représentés sous forme de carte thermique d'incertitude. Des exemples de mesures d'incertitude incluent l'Entropie Attendue et l'Information Mutuelle.
- **C3 :** La stratégie d'agrégation correspond à la stratégie utilisée pour agréger la carte thermique d'incertitude en une seule valeur scalaire au niveau de granularité souhaité en fonction de la tâche en aval. Cela permet aux cliniciens de passer de l'incertitude au niveau du voxel à l'incertitude au niveau de la lésion, par exemple. Comme nous nous intéressons à la quantification de l'incertitude appliquée aux voxels, nous ne considérerons pas cette partie.

— **R3 :** Pour que les praticiens puissent décider si une méthode d'incertitude existante est adéquate pour leur tâche spécifique, il est crucial que les méthodes proposées soient généralement validées sur un large spectre de tâches en aval telles que : Bancs d'essai pour les 5 applications prédominantes de l'incertitude : Détection Hors Distribution (OoD-D), Apprentissage Actif (AL), Détection de Défaillance (FD), Calibration (CALIB), et Modélisation de l'Ambiguïté (AM). Plus d'informations sur ces métriques seront fournies ci-dessous.

C.1.2 Quantification de l'incertitude inter-expert et défi CURVAS

Le challenge *CURVAS* (Calibration et Incertitude pour l'Évaluation de Volume Multi-Évaluateurs dans la Segmentation Multi-Organes), qui s'est tenu de mai à octobre 2024, a mis au défi des équipes de produire un modèle de segmentation qui détermine avec précision la meilleure calibration et quantification de la variabilité inter-experts. Nous utilisons le jeu de données fourni pour ce défi, qui contient un total de 90 scans CT de patients – c'est-à-dire des images produites par un scanner à section transversale (CT) – ainsi que 3 jeux d'annotations faites par 3 différents experts pour le pancréas, les reins et le foie(Voir La figure 1 pour le premier patient de la cohorte. Ces scans CT ont été obtenus à l'Hôpital Universitaire d'Erlangen entre août et octobre 2023. 20 scans CT ont été fournis pour l'entraînement (Groupe A), 5 pour la validation (Groupe A), et 65 pour les tests (20 dans le Groupe A, 22 dans le Groupe B, et 23 dans le Groupe C).).

Dans le repository git CURVAS, plusieurs métriques peuvent être utilisées pour calculer l'incertitude générée par les annotations multi-évaluateurs.

— **Dice Similarity Coefficient (*DSC*) :** La première métrique que nous utiliserons est l'évaluation du Score de similarité de Dice (DSC), qui évaluera uniquement les zones de consensus de premier plan et d'arrière-plan pour trois classes : pancréas, rein et foie. Cela signifie que toute prédiction dans la zone de dissensus sera ignorée. Les Faux Positifs (FP) ne peuvent survenir que dans la zone de consensus d'arrière-plan, et les Faux Négatifs (FN) ne peuvent survenir que dans la zone de consensus de premier plan. Le DSC permet de quantifier le recouvrement spatial entre la

prédiction et la réelle annotation (appelée "ground truth" ou "vérité terrain"). Cette métrique doit être haute.

$$\begin{aligned} \text{DSC} &= \frac{2 \times |\text{Prédiction} \cap \text{Vérité Terrain}|}{|\text{Prédiction}| + |\text{Vérité Terrain}|} \\ &= \frac{2TP}{2TP + FP + FN} \end{aligned}$$

— **Confidence or Uncertainty assessement (C_{seg}) :** De plus, nous prévoyons d'étudier l'incertitude au sein des régions de consensus. Cette étude sera divisée en deux parties : la confiance moyenne sera calculée séparément pour les zones de consensus d'arrière-plan (C_B) et de premier plan (C_F) de chaque classe (pancréas, rein et foie). Ensuite, nous moyennons la métrique de confiance par classe en considérant les deux régions de consensus pour obtenir C_{seg}. Cette métrique doit être haute.

$$C_{\text{seg}} = \frac{(1 - C_B) + C_F}{2}$$

— **Expected Calibration Error (ECE) :** Cette métrique mesure la mauvaise calibration en comparant la précision et la confiance à travers différents bins de prédiction. Un bin, noté B_m, est le groupe ou intervalle de confiance auquel appartient le voxel m (pixel en 3D) relativement à sa probabilité softmax donnée par le réseau de neurones. Chaque bin B_m contient $|B_m|$ échantillons, avec $acc(B_m)$ représentant la précision et $conf(B_m)$ représentant la confiance. Avec n étant le nombre total d'échantillons, nous obtenons l'ECE comme suit. Cette quantité doit être faible.

$$\text{ECE} = \sum_{m=1}^{M} \frac{|B_m|}{n} \Big| acc(B_m) - conf(B_m) \Big|$$

— **Continous Ranked Probability Score (CRPS) :** Le Score de Probabilité Classée Continue est utilisé pour évaluer à quel point la distribution volumétrique prédite s'aligne avec la vérité terrain (précision). Pour conserver la variabilité multi-annotateurs, nous définissons une Fonction de Densité de Probabilité Gaussienne (PDF) basée sur la moyenne et l'écart-type des volumes dérivés des trois annotations d'experts. À partir de cela, nous calculons la Fonction de Distribution Cumulative (CDF) correspondante. La fonction de Heaviside, notée $\mathbb{1}_{\{x \geq y\}}$, est essentielle pour transformer la vérité terrain en une fonction indicatrice, permettant ainsi de comparer directement la CDF prédite à la réalité. Le volume prédit est obtenu en sommant toutes les valeurs probabilistes pour la classe correspondante à partir de la sortie probabiliste fournie par le participant. Cette approche intègre l'incertitude du modèle dans l'estimation du volume.

$$\text{CRPS}(F, y) = \int \Big(F(x) - \mathbb{1}_{\{x \geq y\}} \Big)^2 \, dx$$

Documentation de l'Interface en Ligne de Commande StatApp

Table des matières

1 Introduction

Ce document fournit une documentation complète pour l'interface en ligne de commande StatApp. La CLI StatApp est un outil de segmentation médicale qui gère l'incertitude et la variabilité entre les annotateurs. Elle est construite sur nnU-Net, un framework pour la segmentation d'images prêt à l'emploi. La CLI fournit des commandes pour :

- Préparer des ensembles de données
- Entraîner des modèles
- Exécuter des prédictions
- Combiner plusieurs modèles
- Calculer des métriques
- Gérer les artefacts dans le stockage S3

2 Aperçu des Commandes

2.1 about

Description

Affiche des informations sur le projet, y compris son objectif et ses contributeurs.

Utilisation

```
statapp about
```


Paramètres

Aucun

2.2 upload

Description

La commande upload fournit des fonctionnalités pour téléverser des répertoires locaux vers le stockage S3. Elle comprend trois sous-commandes :

upload_data Téléverse un répertoire local vers le dossier de données S3 défini dans le fichier .env.

upload_model_artifacts Téléverse un répertoire local vers le dossier S3 artifacts/model défini dans le fichier .env.

upload_preprocessing_artifacts Téléverse un répertoire local vers le dossier S3 artifacts/preprocessing défini dans le fichier .env.

Utilisation

```
statapp upload upload-data <directory> [--verbose]
statapp upload upload-model-artifacts <directory> <modelfolder> [--verbose]
statapp upload upload-preprocessing-artifacts <directory> <preprocessingfolder> [--verbose]
```

Paramètres

Sous-commande	Paramètre	Type	Description
upload-data	directory	chaîne	Chemin du répertoire local à téléverser
upload-data	–verbose, -v	drapeau	Activer la sortie détaillée
upload-model-artifacts	directory	chaîne	Chemin du répertoire local à téléverser
upload-model-artifacts	modelfolder	chaîne	Nom du sous-dossier du modèle
upload-model-artifacts	–verbose, -v	drapeau	Activer la sortie détaillée
upload-preprocessing-artifacts	directory	chaîne	Chemin du répertoire local à téléverser
upload-preprocessing-artifacts	preprocessingfolder	chaîne	Nom du sous-dossier de prétraitement
upload-preprocessing-artifacts	–verbose, -v	drapeau	Activer la sortie détaillée

2.3 empty-artifacts

Description

Supprime tous les fichiers et dossiers du dossier d'artefacts S3 défini dans le fichier .env. Cette opération ne peut pas être annulée, donc à utiliser avec précaution.

Utilisation

```
statapp empty-artifacts [--verbose] [--confirm]
```

Paramètres

Paramètre	Type	Description
–verbose, -v	drapeau	Activer la sortie détaillée
–confirm, -c	drapeau	Confirmer la suppression sans demander

2.4 empty-data

Description

Supprime tous les fichiers et dossiers du dossier de données S3 défini dans le fichier .env. Cette opération ne peut pas être annulée, donc à utiliser avec précaution.

Utilisation

```
statapp empty-data [--verbose] [--confirm]
```

Paramètres

Paramètre	Type	Description
–verbose, -v	drapeau	Activer la sortie détaillée
–confirm, -c	drapeau	Confirmer la suppression sans demander

2.5 prepare

Description

La commande prepare fournit des fonctionnalités pour préparer des ensembles de données pour l'analyse. Elle comprend trois sous-commandes :

download-dataset Télécharge un ensemble de données pour analyse sans exécuter de prétraitement.

download-preprocessing Télécharge des artefacts de prétraitement pour un ensemble de données.

prepare Prépare un ensemble de données pour l'analyse en téléchargeant les données et en exécutant le prétraitement.

Utilisation

```
statapp prepare download-dataset <annotator> <patients> [--verbose]
statapp prepare download-preprocessing <annotator> <patients> [--verbose]
statapp prepare prepare <annotator> <patients> [--skip] [--num-processes-
    fingerprint <num>] [--num-processes <num>] [--verbose]
```


Paramètres

Sous-commande	Paramètre	Type	Description
download-dataset	annotator	chaîne	Annotateur (1/2/3)
download-dataset	patients	liste/-chaîne	Liste de numéros de patients (ex. 001 034) ou 'all', 'train', 'validation', 'test'
download-dataset	–verbose, -v	drapeau	Activer la journalisation détaillée
download-preprocessing	annotator	chaîne	Annotateur (1/2/3)
download-preprocessing	patients	liste/-chaîne	Liste de numéros de patients (ex. 001 034) ou 'all', 'train', 'validation', 'test'
download-preprocessing	–verbose, -v	drapeau	Activer la journalisation détaillée
prepare	annotator	chaîne	Annotateur (1/2/3)
prepare	patients	liste/-chaîne	Liste de numéros de patients (ex. 001 034) ou 'all', 'train', 'validation', 'test'
prepare	–skip	drapeau	Ignorer le téléchargement et exécuter uniquement le prétraitement
prepare	–num-processes-fingerprint, -npfp	entier	Nombre de processus à utiliser pour l'extraction d'empreintes (par défaut : 2)
prepare	–num-processes, -np	entier	Nombre de processus à utiliser pour le prétraitement (par défaut : 2)
prepare	–verbose, -v	drapeau	Activer la journalisation détaillée

2.6 train

Description

Exécute l'entraînement nnUNet. L'ensemble de données doit être préparé avec la commande prepare au préalable.

Utilisation

```
statapp train [<seed>] [--fold <fold>] [--patients <patients>] [--annotator <
    annotator>] [--verbose]
```

Paramètres

Paramètre	Type	Description
seed	entier	Définir la graine aléatoire pour la reproductibilité
–fold, -f	chaîne	Pli à utiliser pour l'entraînement. Peut être 'all' pour utiliser tous les plis, ou un numéro de pli spécifique (0-4)
–patients, -p	liste/-chaîne	Liste de numéros de patients (ex. 001 034) ou 'all', 'train', 'validation', 'test'
–annotator, -a	chaîne	Annotateur (1/2/3)
–verbose, -v	drapeau	Activer la journalisation détaillée

2.7 run

Description

Exécute le pipeline complet : préparer les données, entraîner le modèle et téléverser les artefacts. Cette commande combine les fonctionnalités des commandes prepare, train et upload_artifacts.

Utilisation

```
statapp run <annotator> <seed> <patients> [--fold <fold>] [--skip] [--num-processes
    -fingerprint <num>] [--num-processes <num>] [--verbose]
```

Paramètres

Paramètre	Type	Description
annotator	chaîne	Annotateur (1/2/3)
seed	entier	Définir la graine aléatoire pour la reproductibilité
patients	liste/-chaîne	Liste de numéros de patients (ex. 001 034) ou 'all', 'train', 'validation', 'test'
–fold, -f	chaîne	Pli à utiliser pour l'entraînement. Peut être 'all' pour utiliser tous les plis, ou un numéro de pli spécifique (0-4)
–skip	drapeau	Ignorer le téléchargement et exécuter uniquement le prétraitement
–num-processes-fingerprint, -npfp	entier	Nombre de processus à utiliser pour l'extraction d'empreintes (par défaut : 2)
–num-processes, -np	entier	Nombre de processus à utiliser pour le prétraitement (par défaut : 2)
–verbose, -v	drapeau	Activer la journalisation détaillée

2.8 predict

Description

Prédit la segmentation pour les patients en utilisant les modèles spécifiés. Télécharge les images des patients et les points de contrôle des modèles, exécute la prédiction pour chaque modèle et téléverse les résultats vers S3.

Utilisation

```
statapp predict <patients> [--models <models>] [--jobs <num>] [--verbose]
```

Paramètres

Paramètre	Type	Description
patients	liste/-chaîne	Liste de numéros de patients (ex. 001 034) ou 'all', 'train', 'validation', 'test'
–models, -m	liste/-chaîne	Liste de modèles à utiliser pour la prédiction (ex. anno1_init112233_foldall) ou 'all'
–jobs, -j	entier	Nombre de processus à exécuter (par défaut : 10)
–verbose, -v	drapeau	Activer la journalisation détaillée

2.9 ensemble

Description

La commande ensemble fournit des fonctionnalités pour combiner les prédictions de plusieurs modèles. Elle comprend trois sous-commandes :

dl-ensemble Télécharge les prédictions de plusieurs modèles.

run-ensemble Exécute ensemble_folders de nnUNet sur les prédictions de modèles téléchargées.

ensemble Combine les fonctionnalités des commandes dl-ensemble et run-ensemble.

Utilisation

```
statapp ensemble dl-ensemble <patients> [--models <models>] [--jobs <num>] [--
    verbose]
statapp ensemble run-ensemble <patients> [--verbose] [--jobs <num>]
statapp ensemble ensemble <patients> [--models <models>] [--jobs <num>] [--verbose]
```

Paramètres

Sous-commande	Paramètre	Type	Description
dl-ensemble	patients	liste/-chaîne	Liste de numéros de patients (ex. 001 034) ou 'all', 'train', 'validation', 'test'
dl-ensemble	–models, -m	liste/-chaîne	Liste de modèles à utiliser pour l'ensemble (ex. anno1_init112233_foldall) ou 'all'
dl-ensemble	–jobs, -j	entier	Nombre de processus à exécuter (par défaut : 10)
dl-ensemble	–verbose, -v	drapeau	Activer la journalisation détaillée
run-ensemble	patients	liste/-chaîne	Liste de numéros de patients (ex. 001 034)
run-ensemble	–jobs, -j	entier	Nombre de processus à exécuter (par défaut : 10)
run-ensemble	–verbose, -v	drapeau	Activer la journalisation détaillée
ensemble	patients	liste/-chaîne	Liste de numéros de patients (ex. 001 034) ou 'all', 'train', 'validation', 'test'
ensemble	–models, -m	liste/-chaîne	Liste de modèles à utiliser pour l'ensemble (ex. anno1_init112233_foldall) ou 'all'
ensemble	–jobs, -j	entier	Nombre de processus à exécuter (par défaut : 10)
ensemble	–verbose, -v	drapeau	Activer la journalisation détaillée

2.10 metrics

Description

La commande metrics fournit des fonctionnalités pour calculer et télécharger des métriques pour les prédictions de modèles. Elle comprend deux sous-commandes :

compute-metrics Calcule des métriques pour les prédictions de modèles sur les données des patients.

dl-metrics Télécharge tous les fichiers de métriques de S3, les fusionne et les enregistre dans le répertoire de travail.

Utilisation

```
statapp metrics compute-metrics <patients> [--models <models>] [--verbose]
statapp metrics dl-metrics [--output <output_name>] [--verbose]
```

Paramètres

Sous-commande	Paramètre	Type	Description
compute-metrics	patients	liste/-chaîne	Liste de numéros de patients (ex. 075 034) ou 'all', 'train', 'validation', 'test'
compute-metrics	–models, -m	liste/-chaîne	Liste de modèles à utiliser pour le calcul des métriques (ex. anno1_init112233_foldall) ou 'all'
compute-metrics	–verbose, -v	drapeau	Activer la journalisation détaillée
dl-metrics	–output, -o	chaîne	Nom du fichier CSV de sortie (par défaut : metrics.csv)
dl-metrics	–verbose, -v	drapeau	Activer la journalisation détaillée

3 Exemples

3.1 Préparation d'un Ensemble de Données

```
# Télécharger un ensemble de données pour l'annotateur 1 avec tous les patients
statapp prepare download-dataset 1 all --verbose

# Télécharger des artefacts de prétraitement pour l'annotateur 1 avec les patients d'entraînement
statapp prepare download-preprocessing 1 train --verbose

# Préparer un ensemble de données pour l'annotateur 1 avec les patients de validation
statapp prepare prepare 1 validation --num-processes 4 --verbose
```

3.2 Entraînement d'un Modèle

```
# Entraîner un modèle avec l'annotateur 1, la graine 42 et tous les plis
statapp train 42 --annotator 1 --fold all --patients train --verbose

# Exécuter le pipeline complet pour l'annotateur 1, la graine 42 et les patients de validation
statapp run 1 42 validation --fold all --num-processes 4 --verbose
```

3.3 Exécution de Prédictions

```
# Prédire la segmentation pour les patients de test en utilisant tous les modèles
statapp predict test --models all --jobs 8 --verbose

# Prédire la segmentation pour des patients spécifiques en utilisant un modèle spécifique
statapp predict 001 034 --models anno1_init42_foldall --jobs 4 --verbose
```


3.4 Combinaison de Modèles

```
# Télécharger les prédictions pour les patients de test à partir de tous les modèles
statapp ensemble dl-ensemble test --models all --jobs 8 --verbose

# Exécuter l'ensemble pour des patients spécifiques
statapp ensemble run-ensemble 001 034 --jobs 4 --verbose

# Exécuter le pipeline d'ensemble complet pour les patients de test
statapp ensemble ensemble test --models all --jobs 8 --verbose
```

3.5 Calcul de Métriques

```
# Calculer des métriques pour les patients de test en utilisant tous les modèles
statapp metrics compute-metrics test --models all --verbose

# Télécharger et fusionner tous les fichiers de métriques
statapp metrics dl-metrics --output metriques_combinees.csv --verbose
```

3.6 Gestion du Stockage S3

```
# Téléverser des données vers S3
statapp upload upload-data ./mes_donnees --verbose

# Téléverser des artefacts de modèle vers S3
statapp upload upload-model-artifacts ./mon_modele anno1_init42_foldall --verbose

# Vider le répertoire d'artefacts (avec confirmation)
statapp empty-artifacts --verbose

# Vider le répertoire de données (sans confirmation)
statapp empty-data --confirm --verbose
```

4 Variables d'Environnement

La CLI StatApp utilise plusieurs variables d'environnement pour la configuration. Celles-ci doivent être définies dans un fichier .env dans le répertoire racine du projet.

Variable	Description
S3_BUCKET	Le nom du bucket S3
S3_DATA_DIR	Le répertoire dans S3 pour stocker les données
S3_ARTIFACTS_DIR	Le répertoire dans S3 pour stocker les artefacts
S3_OUTPUT_DIR	Le répertoire dans S3 pour stocker la sortie
S3_METRICS_DIR	Le répertoire dans S3 pour stocker les métriques
S3_MODEL_ARTIFACTS_SUBDIR	Le sous-répertoire pour les artefacts de modèle (par défaut : models)
S3_PROPROCESSING_ARTIFACTS_SUBDIR	Le sous-répertoire pour les artefacts de prétraitement (par défaut : preprocessing)
SEED	Graine aléatoire pour la reproductibilité