Note méthodologique : preuve de concept

Dataset retenu

Le dataset utilisé dans ce projet est Cityscapes, une référence en segmentation sémantique urbaine. Il contient 5 000 images RGB haute résolution (2048×1024) capturées dans 50 villes allemandes, annotées pixel par pixel. Chaque pixel est associé à une classe sémantique parmi 34 catégories représentant un élément de la scène (route, trottoir, voiture, piéton, etc.).

La répartition des 5000 images est la suivante :

- **Entraînement** : 2 975 images
- **Validation** : 500 images
- **Test** : 1 525 images

Ce dataset est particulièrement adapté à des cas d'usage comme la conduite autonome, la robotique mobile ou la vidéosurveillance intelligente, car il couvre une grande diversité de scènes urbaines et d'objets (voitures, piétons, routes, feux, etc.).

Les concepts des algorithmes récents

Le modèle U-Net, utilisé dans le projet précédent, repose sur une architecture convolutive symétrique avec un encodeur et un décodeur reliés par des connexions de type "skip". Cette structure est efficace pour capturer à la fois des détails locaux et des informations de haut niveau, mais elle reste limitée dans sa capacité à modéliser le contexte global d'une image car elle est contrainte par la taille des filtres et la profondeur du réseau, ce qui complique la compréhension des relations à longue portée dans des scènes complexes.

Les modèles récents comme Mask2Former et SegFormer s'appuient sur une architecture issue des *Vision Transformers* (ViT), initialement développés pour le traitement du langage naturel. Adaptés à la vision, ils offrent une capacité d'attention globale (chaque élément de l'image peut interagir avec tous les autres), une modularité dans la combinaison encodeur/décodeur et une gestion améliorée des relations spatiales entre objets.

➢ Points communs : une approche ViT Transformer moderne

Le ViT classique [1], a été conçu pour la classification et n'est constitué que d'un encodeur. Cependant pour les tâches de segmentation, il faut un masque de sortie et donc reconstruire une représentation spatiale, un décodeur est donc ajouté.

[1] *« An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale », Alexey Dosovitskiy et al, 2021, https://arxiv.org/abs/2010.11929*

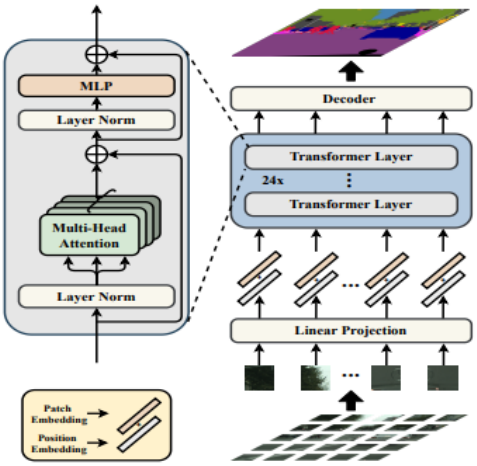


Figure 1 : « Rethinking Semantic Segmentation from a Sequence-to-Sequence Perspective with Transformers», Sixiao Zheng et al, 2021, https://arxiv.org/abs/2012.15840

- L'encodeur

Il extrait des représentations multi-échelles d'une image, c'est-à-dire des informations à différents niveaux de granularité : bas niveau (textures, bords, couleurs locales) et haut niveau (formes, objets, relations spatiales) grâce à :

❖ des patch embeddings pour réduire la complexité,

Ils découpent l'image en petits blocs appelés "patchs" (par exemple 16×16 pixels), que l'on transforme en vecteurs, contrairement aux CNNs qui traitent directement les pixels. Ces vecteurs sont appelés patch embeddings. C'est l'équivalent, pour une image, de ce que sont les tokens pour une phrase dans un modèle de langage.

❖ Et un encodage positionnel pour conserver l'information spatiale

Un ViT est permutation-invariant donc il ne peut pas savoir où se situe chaque patch embedding dans l'image d'origine. Ainsi, à chaque patch embeddings est attaché un vecteur de position qui encode ses coordonnées spatiales dans l'image.

❖ des blocs Transformer empilés avec attention multi-tête.

Ce sont des unités de traitement qui permettent à un modèle de comprendre les relations entre les éléments d'une séquence, ici, les patchs d'image. Il est composé de deux grandes parties : le mécanisme d'attention multi-tête et un réseau feed-forward (MLP).

L'attention permet à chaque patch de pondérer l'importance des autres patchs dans l'image. Pour cela, chaque patch est transformé en trois vecteurs via trois matrices apprises différentes :

- **Query (Q)** : ce que ce patch cherche chez les autres
- **Key (K)** : ce que chaque autre patch a à offrir
- **Value (V)** : l'information qui sera agrégé si on décide d'écouter ce patch

Puis, on calcule une similarité entre Q et K pour chaque paire de patchs pour obtenir une matrice d'attention. Cette matrice est utilisée pour pondérer les valeurs V et chaque patch reçoit une représentation enrichie par les autres.

Cette attention est dite multi-tête car on en calcule plusieurs en parallèle et chaque tête peut se concentrer sur des relations locales (bordures, textures) ou des relations globales (objets, contexte) ou des interactions spécifiques (piéton sur trottoir, voiture sur route). Enfin, les résultats de toutes les têtes sont concaténés puis passés dans un MLP pour produire la sortie du bloc.

Contrairement au mécanisme d'attention, le MLP agit séparément sur chaque token. Le vecteur du patch est projeté vers un espace plus grand pour créer une représentation plus riche, puis une fonction GELU est appliquée pour ajouter de la non-linéarité et n'activer plus intensément que les informations le plus importantes, enfin, on recompresse l'information dans l'espace de départ afin d'avoir des représentations plus complexes, affinées.

En plus de ces deux opérations, des couches de normalisation et l'addition de connexions résiduelles permettent de stabiliser l'apprentissage et de préserver les informations initiales. A la sortie, on obtient une séquence de vecteurs ou patch embeddings enrichie par le mécanisme d'attention et affinée par le MLP.

- Le décodeur

Il réinterprète les patch embeddings enrichis pour produire une carte de segmentation dense en sortie grâce à une projection linéaire via une convolution 1×1 qui permet de réduire la dimension de canaux, indépendamment sur chaque pixel, en multipliant le patch embeddings par une matrice de poids. Sans mélanger l'information spatiale puisqu'elle ne regarde pas ses voisins, elle rend l'information plus compacte.

Ensuite, une restructuration spatiale (ou reshape) permet de redonner une structure spatiale 2D à la séquence pour passer en mode «image», riche en informations. Un upsampling agrandit l'image jusqu'à la taille de l'image d'entrée et enfin, une convolution 1×1 ramène le nombre de canaux au nombre de classes et un softmax est appliqué pour transformer cela en probabilité de classe par pixel.

Ces architectures permettent une meilleure compréhension contextuelle de la scène, une segmentation plus fine des objets, et une meilleure robustesse aux variations de l'image. Contrairement aux CNNs, les Transformers peuvent modéliser des relations à grande échelle, ce qui est crucial pour comprendre des scènes urbaines complexes.

➢ SegFormer : simplicité et robustesse

SegFormer, proposé par Nvidia en 2021, est conçu pour être léger, rapide et facilement entraînable from scratch.

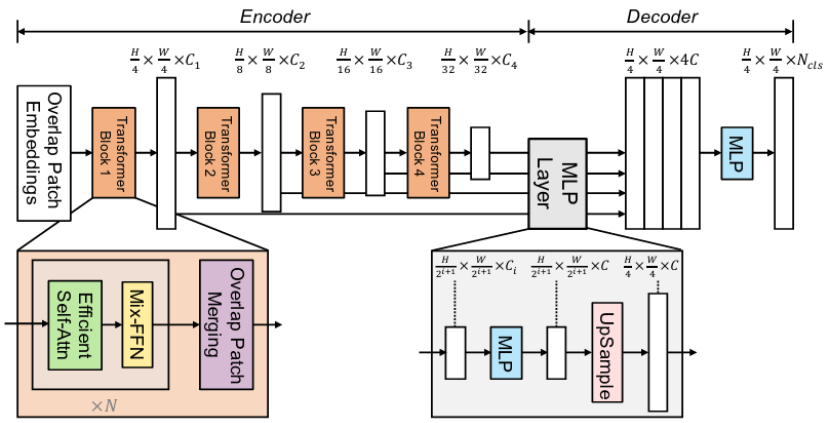


Figure 2 : "SegFormer: Simple and Efficient Design for Semantic Segmentation with Transformers" Enze Xie et al., NeurIPS 2021, arXiv:2105.15203

Il repose sur deux modules principaux :

- MiT (Mix Transformer) : un encodeur hiérarchique léger et sans positional encoding avec attention locales restreintes,

L'encodeur MiT est une architecture hybride conçue pour des tâches de segmentation dense. Il cherche à combiner la capture des relations spatiales des Transformers avec la hiérarchie et l'efficacité computationnelle des CNNs.

La première étape est l'Overlap Patch Embeddings qui utilise une convolution 3x3 pour capturer des informations spatiales locales dès le départ comme dans un CNN. Cela permet de créer des patchs chevauchants à la différence des Transformers classiques (ViT) qui utilisent des patchs non chevauchants. Ainsi des informations de voisinage sont conservées, la segmentation des contours est améliorée, sans recourir au positional encoding, ce qui le rend plus flexibles aux résolutions variables.

Après cette étape, l'image est transformée en une séquence de tokens spatiaux avec informations locales. Ces tokens passent ensuite par une série de blocs Transformers appelés MiT (Mix Transformers) blocs, chacun composé de :

❖ Efficient self-attention

Cette attention locale se concentre sur des fenêtres locales non chevauchantes au lieu d'avoir une attention globale comme dans les Transformers classique. La self attention est calculée uniquement à l'intérieur de chaque puis on concatène les sorties de chaque fenêtre pour reformer une carte complète. Chaque token intéragit seulement avec ses voisins proches afin de capturer les relations proches sans exploser la mémoire GPU. La capture des relations plus longues distances est compensée par le multi-échelle.

❖ Mix-FFN (Feed-Forward Network)

C'est un FFN amélioré qui intègre une convolution 3x3 entre les couches linéaires pour ajouter de l'information spatiale : Linear -> ReLU -> Conv 3x3 -> Linear. Contrairement au FFN standard qui agit indépendamment sur chaque token, le Mix-FFN permet de mieux distinguer les bords et formes locales de l'image.

❖ Overlap Patch Merging

A la fin de chaque bloc Transformers, l'Overlap Patch Merging est réalisé. C'est une convolution avec chevauchement qui permet de faire un downsampling progressif de la carte des caractéristiques tout en conservant des informations locales : le nombre de canaux est doublé et la taille spatiale est divisée par deux.

Après chaque bloc Transformers, la carte de caractéristiques est conservée. Ainsi, 4 cartes de caractéristiques de résolutions différentes (H/4, H/8, H/16, H/32) sont reçues par le décodeur. Cette hiérarchie multi-échelle contiennent des informations complémentaires : les premières couches capturent des détails fins tandis que les dernières appréhendent le contexte global.

- MLP-based Lightweight Decoder : un décodeur simple, uniquement composé de couches MLP et de fusions des cartes multi-échelles,

Le décodeur est simple et léger, toute la complexité revient à l'encodeur. Il repose sur des MLPs uniquement, sans opérations convolutionnelles lourdes mais juste des projections linéaires.

Chaque carte de caractéristiques multi-échelle a un nombre de canaux différent, et, pour les uniformiser elles sont projetées via une convolution 1×1. Ainsi, tous les niveaux sont projetés dans un espace commun de dimension C. Puis elles sont agrandies à la même taille spatiale via un upsampling (souvent H/4).

Les 4 cartes de caractéristiques ayant maintenant la même résolution spatiale et le même nombre de canaux sont ensuite sommées ou concaténées, selon l'implémentation. Une dernière projection linéaire via une convolution 1x1 prédit la segmentation par pixel sur les classes cibles et un upsampling final à la taille de l'image d'origine est réalisé.

L'architecture de SegFormer le rend rapide, léger et particulièrement adapté à des environnements contraints (peu de ressources, pas de pré-entraînement) grâce à ce décodeur minimaliste tout en offrant des performances compétitives (mIoU > 82% sur Cityscapes) avec son encodeur hiérarchique qui capture efficacement à la fois les détails fins et le contexte global.

➢ Mask2Former : segmentation unifiée

Mask2Former, introduit par Meta AI en 2022, est une architecture plus complexe et plus puissante, conçue pour unifier la segmentation sémantique, panoptique et par instance.

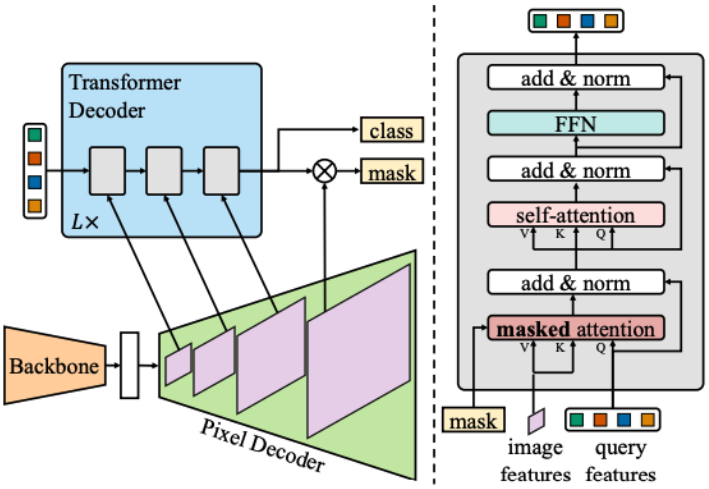


Figure 3: "Masked-attention Mask Transformer for Universal Image Segmentation"
Bowen Cheng et al, 2022, arXiv:2112.01527

Il repose sur 3 module principaux :

- un encodeur pré-entrainé,

Il peut être de type CNN, comme ResNet, ou Transformer hiérarchique, comme MiT ou Swin. Son rôle est d'extraire des cartes de caractéristiques multi-échelles riches.

- un Pixel Decoder,

Le Pixel Decoder, inspiré du FPN (Feature Pyramid Network), fusionne les cartes de caractéristiques des différentes échelles pour créer une pyramide de représentations multi-échelles pour compenser les pertes de précision spatiale liées au downsampling.

- un décodeur basé sur des requêtes et une attention masquée,

C'est l'innovation clé de Mask2Former. Le Transformer Decoder est un transformer standard enrichi d'une masked cross-attention. Il sert à apprendre les requêtes (queries), initiées aléatoirement, qui représentent des objets ou segments et leur associer une classe et un masque binaire indiquant la position de la requête à chacune.

Au fil de l'entraînement chaque requête va interroger les pixels des cartes de caractéristiques de Pixel Decoder mais seulement dans la région définie par son masque courant associé, grâce à la masked cross-attention, afin qu'elle se focalise à des zones pertinentes au lieu de l'image entière. Itérativement, le masque devient de plus en plus précis et restreint de plus en plus l'attention.

Ensuite, les requêtes passent par une couche de self-attention afin qu'elles partagent leurs informations et capturent leur dépendances/relations. Ainsi, chaque requête apprend progressivement à se spécialiser dans un type de segment ou une région.

Le FFN (Feed Forward Network) est un traitement classique avec 2 couches linéaires (ou Dense) et une activation GELU appliqué à chaque query indépendamment. Il permet de garder les informations importantes activés en ajoutant de la non-linéarité.

Ces étapes sont répétées dans les 6 blocs dans le Transformer Decoder pour affiner progressivement la segmentation pour toutes les requêtes. A la sortie, chaque requête produit un type de classe et un vecteur de masque qui sera multiplié (produit scalaire) avec la carte de caractéristique de plus haute résolution du Pixel Decoder pour former le masque final.

Ce modèle montre une robustesse démontrée face à des perturbations réalistes (pluie, nuit, flou…) et est particulièrement adapté à des cas d'usage exigeants comme la conduite autonome ou la cartographie urbaine.

La modélisation

L'entraînement des différents modèles a été conduit selon une méthodologie commune afin d'assurer une comparaison équitable. Chaque architecture est initialisée conformément à ses spécificités (pré-entraînement éventuel ou apprentissage from scratch), puis optimisée dans un cadre identique de réglages et d'évaluation.

- Méthodologie d'entraînement

Les modèles sont optimisés avec l'algorithme AdamW, particulièrement adapté aux architectures de type Transformer. Il combine la stabilité de l'optimiseur Adam avec une décroissance des poids (weight decay) qui agit comme un terme de régularisation et limite le surapprentissage. Une légère data augmentation (rotations, flips, variations de luminosité) identique à celle utilisée pour U-Net est appliquée afin de garantir une comparaison cohérente.

Plusieurs hyperparamètres critiques ont été explorés :

- **Taux d'apprentissage** (learning rate) : valeurs comprises entre 5×10^{-4} et 1×10^{-5}, avec un scheduler décroissant pour faciliter la convergence en fin d'entraînement.

- **Taille de batch** (batch size) : fixée entre 2 et 8 selon les contraintes de mémoire GPU.

- **Décroissance des poids** (weight decay) : valeurs de 0.01 et 0.05, afin de comparer différents niveaux de régularisation.

L'optimisation a suivi une démarche progressive :

1. **Phase exploratoire** : séries d'entraînements courts (5 epochs) pour tester différentes combinaisons des trois hyperparamètres.

2. **Phase affinée** : réplication de ces mêmes entraînements en ajoutant la data augmentation pour observer son impact.

3. **Sélection** : choix du modèle ayant obtenu la meilleure performance selon la métrique d'évaluation retenue.

4. **Entraînement final** : réentraînement complet du meilleur modèle sur 50 epochs avec les hyperparamètres optimisés.

- Inférence de Mask2Former

Mask2Former, en raison de sa complexité (plus de 80M de paramètres et un besoin mémoire conséquent), n'a pas été entraîné from scratch. Dans notre étude, il a été utilisé en inférence uniquement, en exploitant un modèle pré-entraîné sur Cityscapes. Cette approche reflète un usage réaliste de ce type de modèle moderne, souvent exploité en transfert direct plutôt qu'en ré-entraînement complet. L'évaluation est réalisée avec les mêmes données et les mêmes métriques que pour les autres modèles, garantissant la comparabilité des résultats.

- Métrique d'évaluation

La performance est mesurée avec le **IoU moyen pondéré (weighted mean Intersection over Union)**, métrique de référence en segmentation d'images. Contrairement à l'IoU classique qui calcule, pour chaque classe, le rapport entre l'intersection et l'union des pixels prédits et des pixels réels, puis effectue une moyenne sur toutes les classes, cette version pondère la contribution de chaque classe en fonction de sa fréquence. Elle permet ainsi de mieux refléter la qualité globale de la segmentation dans des jeux de données déséquilibrés, en évitant qu'une classe très minoritaire n'influence exagérément la métrique.

Une synthèse des résultats

L'évaluation comparative des différentes architectures testées (U-Net avec encodeur VGG16, SegFormer-B1, et Mask2Former) met en évidence des différences nettes en termes de précision, de robustesse et de coût computationnel.

	U-net	SegFormer	Mask2Former
IoU global	69.9%	65.4%	-
IoU pondéré	82.1%	78.6%	91.0%
IoU macro	71.8%	65.8%	70.8%
Dice global	88.9%	87.8%	95.0%
Dice pondéré	89.6%	87.2%	95.0%
Dice macro	82.0%	76.7%	77.3%

Sur ce tableau, on voit que **U-Net** reste compétitif avec des scores solides, notamment en IoU pondéré (82.1%) et Dice pondéré (89.6%). Il s'avère particulièrement efficace sur les classes majoritaires, mais son IoU global (69.9%) reflète une difficulté à gérer les objets de petite taille ou peu représentés.

SegFormer, malgré une architecture plus moderne, affiche ici des résultats légèrement inférieurs à U-Net sur la majorité des métriques (IoU pondéré 78.6% contre 82.1% pour U-Net, Dice pondéré 87.2% contre 89.6%). Ce décalage s'explique principalement par l'entraînement from scratch sans recours à un pré-entraînement, ce qui limite sa capacité de généralisation, en particulier sur un jeu de données restreint. Dans la littérature, SegFormer tire pleinement parti de son encodeur hiérarchique lorsqu'il est initialisé avec des poids pré-entraînés.

Il est intéressant de noter que lorsque l'on compare des conditions strictement équivalentes, c'est-à-dire U-Net sans encodeur pré-entraîné vs SegFormer sans encodeur pré-entraîné, tous deux entraînés sur seulement 5 epochs, SegFormer démontre une meilleure capacité d'apprentissage initiale :

- IoU pondéré : 69.7% (U-Net) vs 73.7% (SegFormer)
- IoU macro : 54.9% (U-Net) vs 58.4% (SegFormer)

Ces résultats partiels suggèrent que, même sans pré-entraînement, l'architecture hiérarchique et multi-échelle de SegFormer lui permet de mieux exploiter les premiers signaux d'apprentissage que U-Net. Cela confirme que son déficit global dans nos expériences provient surtout du manque de pré-entraînement et non d'une faiblesse intrinsèque de l'architecture.

Mask2Former, utilisé uniquement en inférence à partir d'un modèle pré-entraîné, obtient les meilleures performances sur presque toutes les métriques pondérées et globales (IoU pondéré 91.0%, Dice global et pondéré 95.0%). Cela confirme la supériorité des modèles récents basés sur attention masquée et requêtes, capables de capturer à la fois contexte global et détails fins. En revanche, ses résultats en IoU macro (70.8%) et Dice macro (77.3%) montrent que la performance est moins homogène sur les classes rares.

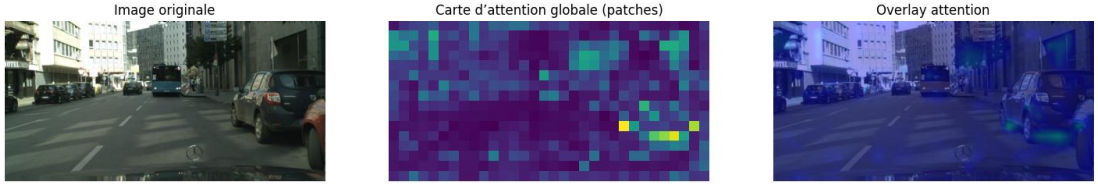
Analyse de la feature importance globale et locale

L'analyse des cartes d'attention vise à mieux comprendre le fonctionnement interne des modèles basés sur des Transformers appliqués à la segmentation d'image. Contrairement aux U-Net classiques, ces architectures disposent de mécanismes d'attention multi-têtes qui mettent en évidence les relations entre patches de l'image.

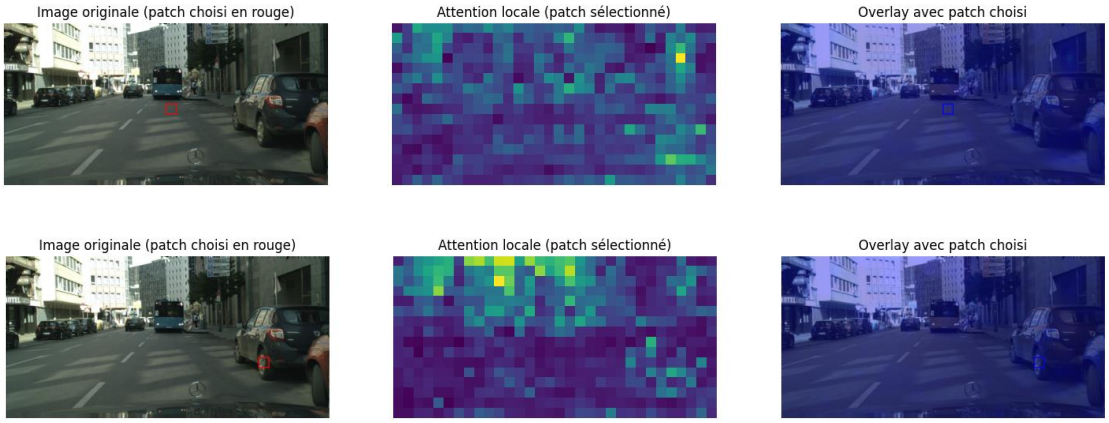
Deux niveaux d'interprétation ont été étudiés :

- Importance globale : zones qui captent en moyenne le plus l'attention du modèle (moyenne sur têtes et couches).
- Importance locale : distribution de l'attention à partir d'un patch choisi (ex. une roue de véhicule), pour caractériser les interactions entre éléments de la scène.

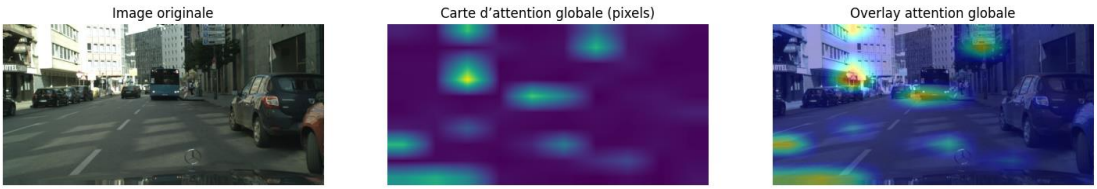
SegFormer repose sur un encodeur hiérarchique qui produit des représentations multi-échelles. Pour l'importance globale, en moyennant les attentions de toutes les couches, nous observons qu'il exploite les objets structurés, comme les voitures, comme points de repère principaux. En revanche, les zones de contexte, comme la route, sont secondaires mais nécessaires pour la cohérence générale (activation faible). Cette focalisation traduit une bonne exploitation des structures globales, mais une attention limitée aux petits objets. En effet, les patches correspondant à des piétons ou à des éléments fins sont souvent dilués dans l'attention globale.



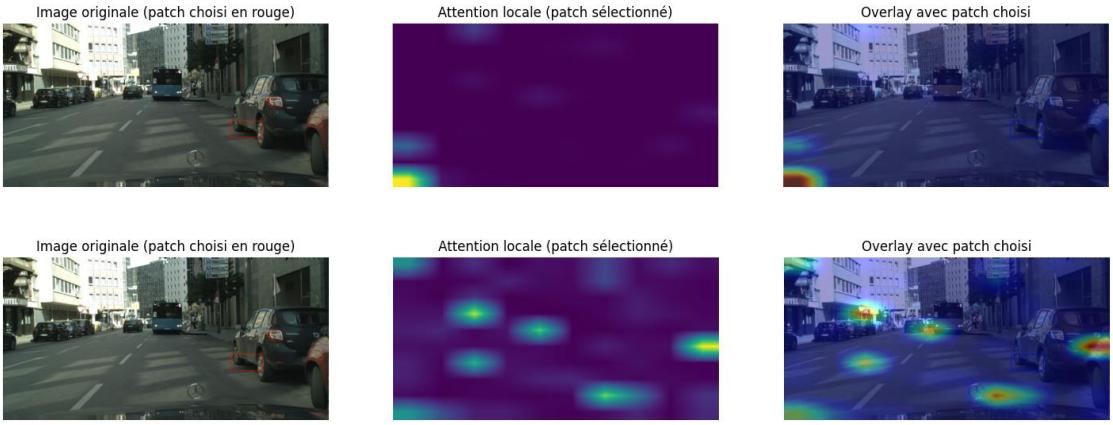
Pour l'importance locale, avec un patch sélectionné, l'attention de SegFormer se distribue principalement vers des zones proches du patch (structure locale), les frontières ou bords de l'objet du patch, et également vers des régions homogènes, comme la route, qui apportent un contexte simple mais peu discriminant.



Quant à l'importance globale de Mask2Former, en agrégeant les attentions de toutes les têtes, on observe une activation des objets centraux (bus, voitures), mais aussi sur des relations de contexte, comme les immeubles et panneaux qui structurent la scène. Les zones uniformes, comme la route, sont peu activées. Cela traduit une compréhension plus holistique de la scène par rapport à SegFormer : Mask2Former combine objets et contexte dans son attention globale.



Pour l'importance locale, avec un patch sélectionné, l'attention de Mask2Former révèle une grande variabilité selon les têtes (voir cartes d'attentions ci-dessous sur le même patch mais avec des têtes différentes) : certaines privilégient les relations et d'autres projettent des connexions longues portées renforçant la cohérence globale. Cette diversité traduit une complémentarité entre têtes : certaines spécialisées dans la détection de détails, d'autres dans la contextualisation.



Les limites et les améliorations possibles

L'approche a montré de bonnes performances, mais plusieurs limites ont été identifiées :

- la taille limitée du jeu de données d'entrainement ainsi que son déséquilibre entre les classes impactent négativement la généralisation du modèle, en particulier sur les classes rares.

- interprétabilité limitée : les cartes d'attention révèlent une grande complexité, et leur interprétation reste partielle.

- Mask2Former permet l'augmentation de la performance mais au prix d'un coût computationnel très élevé.

Plusieurs pistes d'amélioration peuvent être envisagées :

- Enrichissement des données : augmenter la diversité des classes minoritaires (ex. via sur-échantillonnage ou data augmentation ciblée) et la taille du jeu de données.

- Optimisation des architectures : test de variants plus légers de Mask2Former pour réduire le coût d'inférence ou de variants SegFormer plus complexe (B2 à B5). Pour SegFormer, il est également possible de prendre plus que la dernière couche de l'encodeur afin d'apporter au décodeur des informations plus riches.

- Interprétabilité renforcée : utilisation de méthodes complémentaires aux cartes d'attention, comme Grad-CAM ou attribution par perturbation.

En résumé, SegFormer et Mask2Former offrent deux visions complémentaires : l'un plus léger et rapide mais limité sur les petits objets, l'autre plus riche mais coûteux.
L'avenir passe par une meilleure synergie entre performance et interprétabilité, d'où les nouvelles générations de modèles multimodaux explicables comme EOMT (eXplainable Omni-Modal Transformers) qui pourraient jouer un rôle clé en rendant les décisions des systèmes de perception plus fiables et transparentes.