La segmentation sémantique : Introduction, état de l'art à Janvier 2022, et quelques résultats sur cityscape

Table des matières

A. Introduction

Définition :

La segmentation d'image consiste à classifier chaque pixel en fonction de classes prédéfinies :
Exemple, ce pixel appartient à la classe lit, l'autre à la classe mur…

La segmentation sémantique se différencie de la détection d'objet car ne prédit pas de boxes entourant les objets. Nous verrons une zone de voitures par exemple avec la segmentation mais nous ne ferons pas la distinction entre différentes voitures si ces dernières se chevauchent dans la zone.

Application :

Voici une liste non exhaustive des applications de la segmentation sémantique :

- L'imagerie médicale : réaliser des diagnostics comme segmenter des tumeurs
- L'analyse d'image satellite : segmentation de différents types de terrain
- Véhicules autonomes

B. Construction

Architecture de base :

Les CNN convolutionnal neural networks ont fait leur preuve en matière de Computer Vision. Les architectures typiques des CNN contiennent des couches convolutionnelles, des couches non linéaires d'activation, des batch normalization ainsi que des couches de pooling pour réduire le nombre de paramètres.

Les premières couches convolutionnelles, dites couches basses, apprennent des concepts bas niveau comme des bords alors que les couches élevées apprennent des concepts comme les objets.

Dans les couches basses, les neurones contiennent de l'information pour une petite zone alors que pour les couches élevées, les neurones contiennent de l'information pour une zone bien plus large.

Plus on ajoute de couche dans le CNN, plus les images diminuent en définition, en résolution et plus le nombre de channels (filtres) augmente.

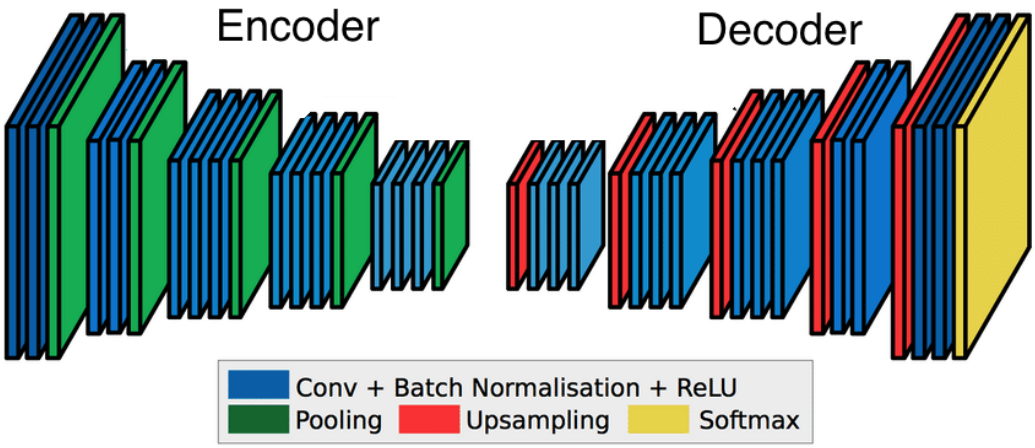
Dans le cas de la segmentation sémantique, il est nécessaire de garder l'information spatiale, nous ne pouvons donc pas utiliser de couche fully connected en fin de réseau.

Un autre gros problème réside dans le fait que nous obtenons un tenseur spatial de faible résolution contenant de l'information de haut niveau alors que nous devons produire une segmentation de haute résolution en évitant trop de perte d'information.

Comment obtenir en sortie une image de haute résolution ?

Une solution consiste donc à ajouter des couches convolutionnelles couplées à des couches de up sampling. Ainsi on augmente la résolution du tenseur et nous diminuons le nombre de channels. Au final nous avons une **structure d'encodeur décodeur** :

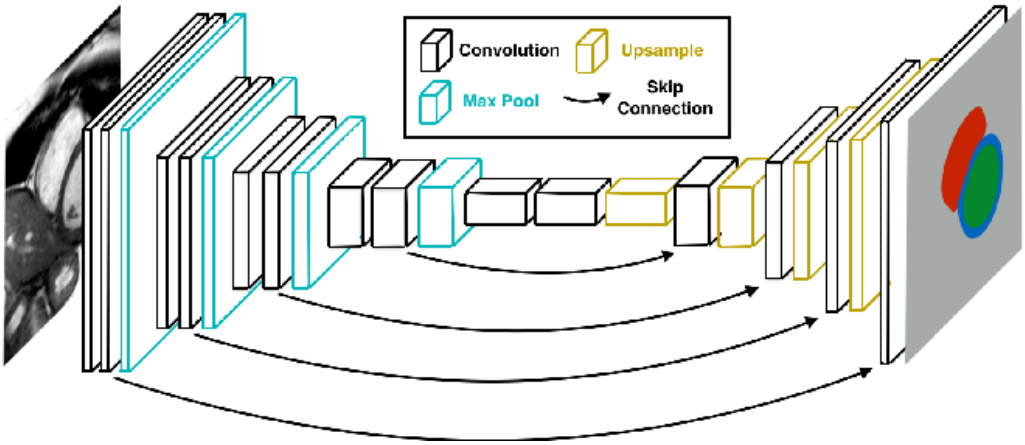
- La partie encodeur correspond au downsampling et a pour rôle de construire les features des objets (la features map)
- la partie décodeur au upsampling et a pour rôle de produire les segmentation map



Un des problèmes de cette architecture est la perte d'information pendant le up-sampling. Les limites des zones segmentés peuvent être au final inexactes.

Comment améliorer la précision des bords de classes ?

Une des solutions consiste à laisser accéder le décodeur à des caractéristiques de bas niveau de l'encodeur. On appelle cela les skip connections. On réalise cela par concaténation :

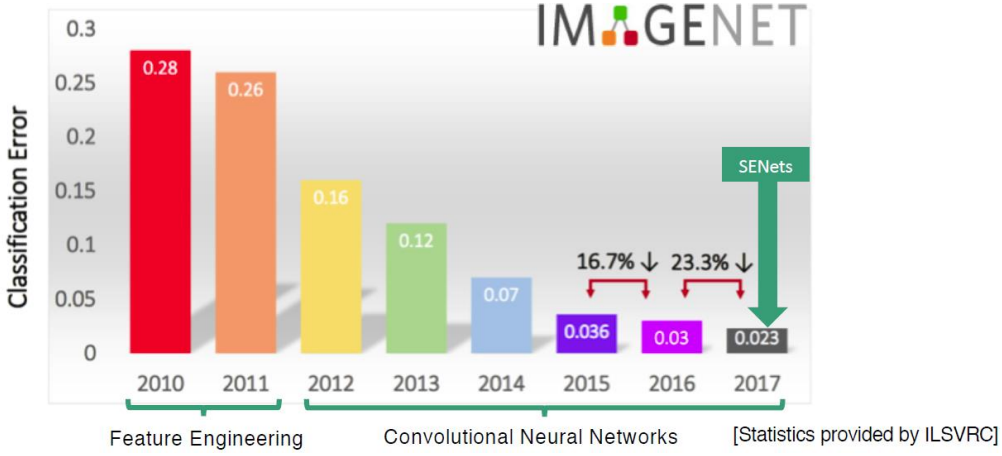


Réseau de base

Les encodeurs pré entrainés pour la classification d'images, comme ceux ayant fait leur preuve lors du concours ILSVRC imagenet peuvent être utilisés. VGG et RESNET sont des choix populaires.

Rappel des Résultats ILSVRC :

- 2014: GoogleLeNet-inception V1 : top-5 error=6.67% (proche d'un niveau de performance humaine)
- 2014 : VGGNet : top-5 error : de 6.8 à 7.3% (souvent choisi par la communauté pour l'extraction de caractéristiques)
- 2015 : ResNet-residual neural network : top-5 error : 3.57% (introduit les skip connections)
- 2016: ResNeXt: top-5 error :3% (introduit la cardinality dimension)
- 2017: SENet – Sequeeze and excitation: 2.25%



Etat de l'art de la classification d'image ImageNet

Aprés la fin du concours ILRSVC en 2017, sont apparus des modèles un peu plus efficients considérés comme plus ou moins l'état de l'art dans la classification d'images. Ce sont les EfficientNet :

https://medium.com/mlearning-ai/understanding-efficientnet-the-most-powerful-cnn-architecture-eaeb40386fad

https://arxiv.org/abs/1905.11946

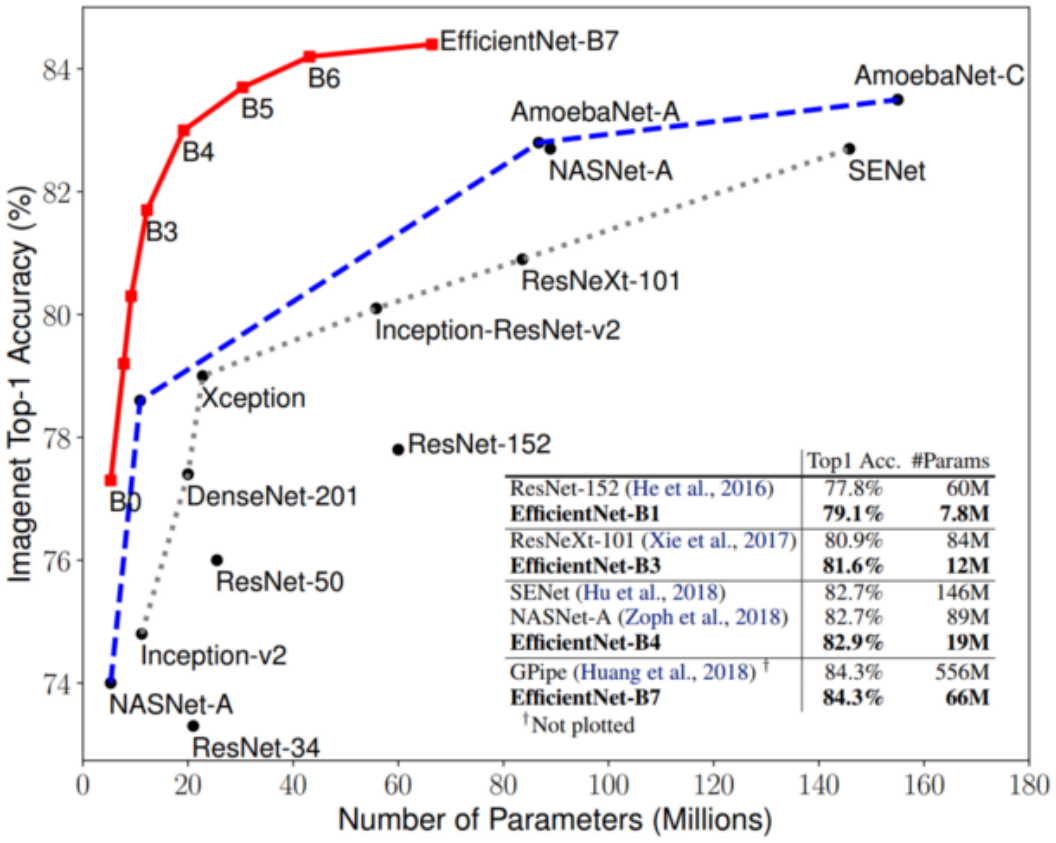
Apport des EfficientNet :

Jusqu'à l'arrivée des EfficientNet, la précision des modèles était améliorée en augmentant aléatoirement la taille de l'architecture en fonction de la disponibilité des ressources. Cette augmentation de taille pouvait se faire de 3 façons différentes :

- En profondeur
- En épaisseur.
- En augmentant la définition de l'image en entrée.

Cette augmentation de taille se faisait un peu aléatoirement au prix de nombreuses heures de réglage sans pour autant toujours obtenir les résultats escomptés. **EfficientNet a donc apporté un framework à l'augmentation d'échelle de façon** à lier l'augmentation des 3 dimensions (resolution, profondeur, largeur) du réseau.

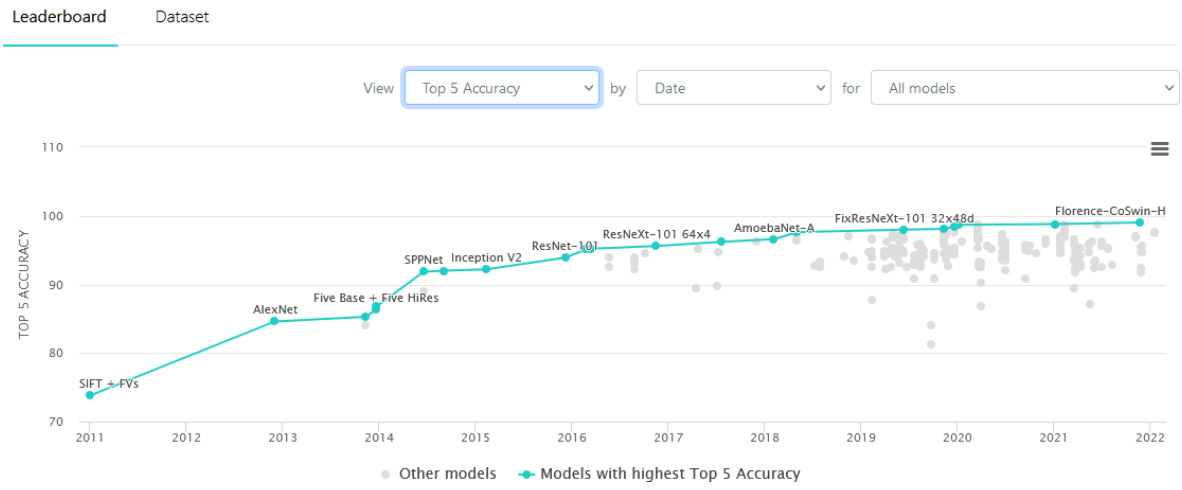
L'intuition est la suivante : si on augmente la résolution de l'image, on a alors besoin de plus de couches et de plus de channels (filtres) pour capturer tous les motifs. Le modèle est aussi pénalisé lorqu'il est devient trop lourd. Au final **EfficientNet** obtient une meilleure précision que les modèles précédents avec moins de paramètres.



	Top1 Acc.	#Params
ResNet-152 (He et al., 2016)	77.8%	60M
EfficientNet-B1	**79.1%**	**7.8M**
ResNeXt-101 (Xie et al., 2017)	80.9%	84M
EfficientNet-B3	**81.6%**	**12M**
SENet (Hu et al., 2018)	82.7%	146M
NASNet-A (Zoph et al., 2018)	82.7%	89M
EfficientNet-B4	**82.9%**	**19M**
GPipe (Huang et al., 2018) †	84.3%	556M
EfficientNet-B7	**84.3%**	**66M**

†Not plotted

On peut également suivre l'état de l'art de la classification d'images via le suivi de performance malgré la fin du concours ILRSVC en suivant le lien suivant : https://paperswithcode.com/sota/image-classification-on-imagenet



Les Transformers, déjà réputés dans le domaine du NLP commencent à prendre la tête du classement.

https://viso.ai/deep-learning/vision-transformer-vit/

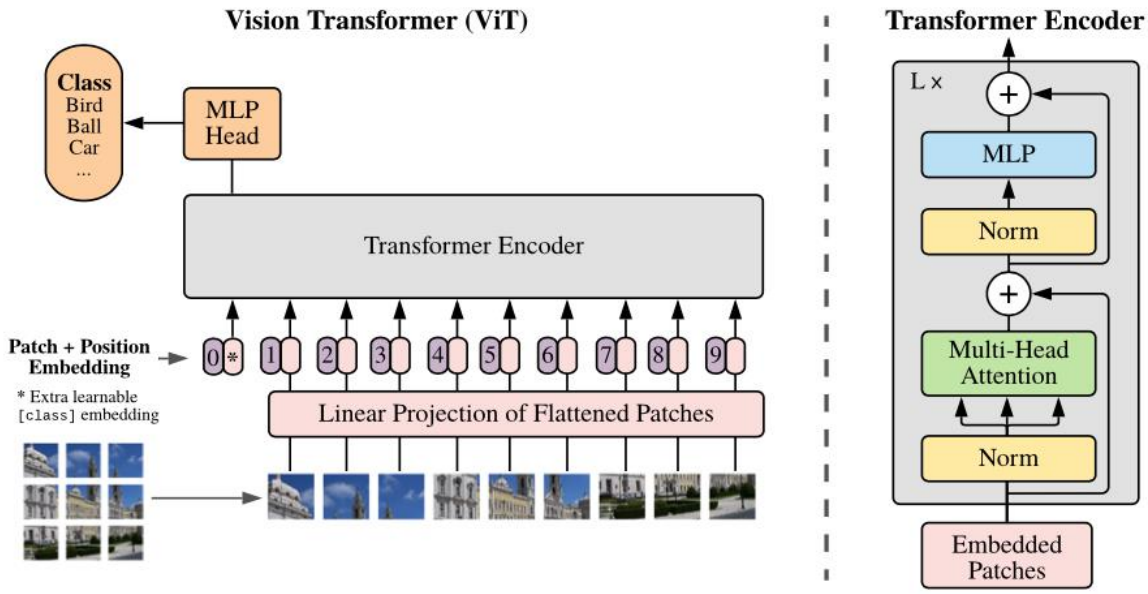
Pour rappel, un transformer est un modèle de deep learning utilisant le mécanisme d'attention qui pondère différemment chaque partie de l'input en fonction de son importance. Les visions

Transformer ont en général moins de biais inductif que le CNN comme l'invariance de l'output à la translation de l'input. Ce qui ajoute de la précision en cas de data augmentation.

Le ViT représente l'input comme une séquence de morceaux d'images de tailles fixes. Il inclut un plongement positionnel en input de l'encodeur

L'encodeur du VIT possède un MSP (multi head self attention layer) qui permet d'entrainer des dépendances locales et globales

Il possède également une couche MLP et des couches de normalisation :



Les ViT ont déjà prouvé leurs performances spécialement sur les gros datasets. Sur des petits datasets mieux faut privilégier des resnet ou efficientnet

Construction du modèle de segmentation classique :

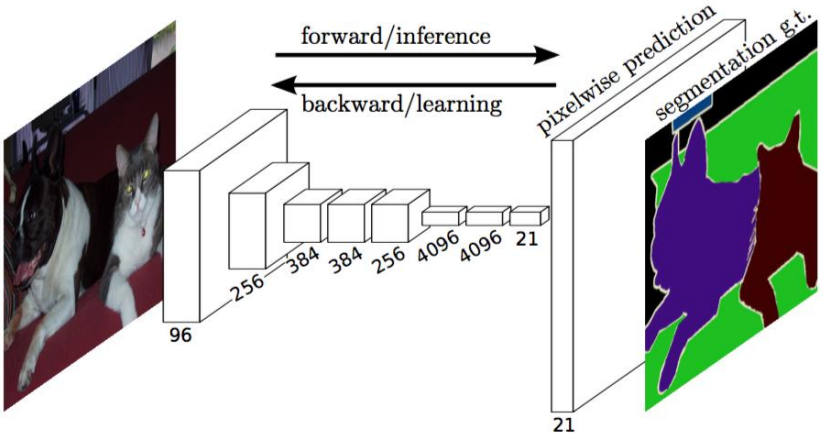
La première étape consiste à choisir un encodeur Pour de nombreuses applications (non médicales par exemple), choisir un modèle pré entrainé sur imagenet est un bon choix. En général, le pré entrainement imagenet est bénéfique pour des images indoor ou outdoor.

On peut en citer quelques-uns comme :

- RestNet (microsoft): 2016 : 96.4% de précision à l'ILRSVC de 2016
- VGG16 (oxford) : 2013 : 92.7% de précision. VGG 16/19 possède Moins de couches que ResNet50 ou 101 , et est donc plus facile à entrainer mais moins précis.
- MobileNet (Google): optimisé pour des petites tailles et des temps de prédiction faibles . Pas très précis mais adapté aux mobiles, ou aux supports à faible ressources
- CustomCNN non pré entrainé pour des applications très simples.
- EfficientNet

La 2ème étape consiste à choisir le décodeur responsable de la segmentation. Différentes approches classiques existent :

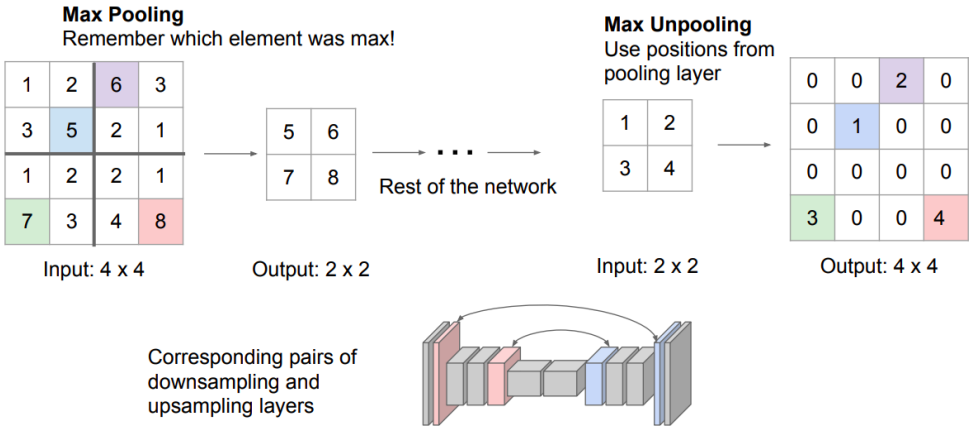
- Approche FCN (fully convolutionnal network):
  Des convolutions transposées sont utilisées pour upsampler
  On peut utiliser des skip connections pour améliorer la précision des limites.



- Approche SegNet :

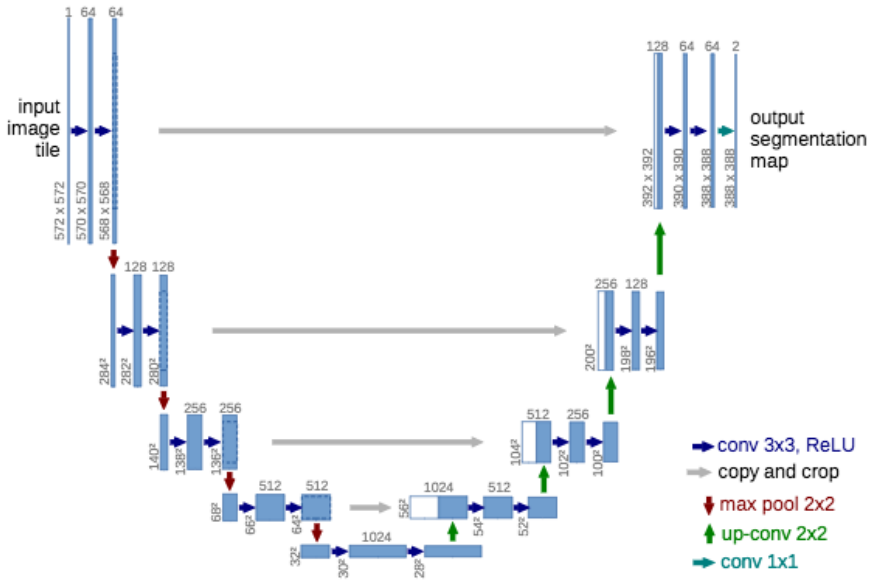
C'est une approche décodeur encodeur symétrique sans skip connection :

L'upsampling se fait par max unpooling en récupérant les indices du max pooling correspondant du décodeur. Il n'y a donc pas de paramètres à apprendre dans la partie upsampling. Pour des jeux de données simples avec de un petit nombre de gros objets , Des modèles simples comme FCN et Segnet seront suffisants



- Approche Unet :

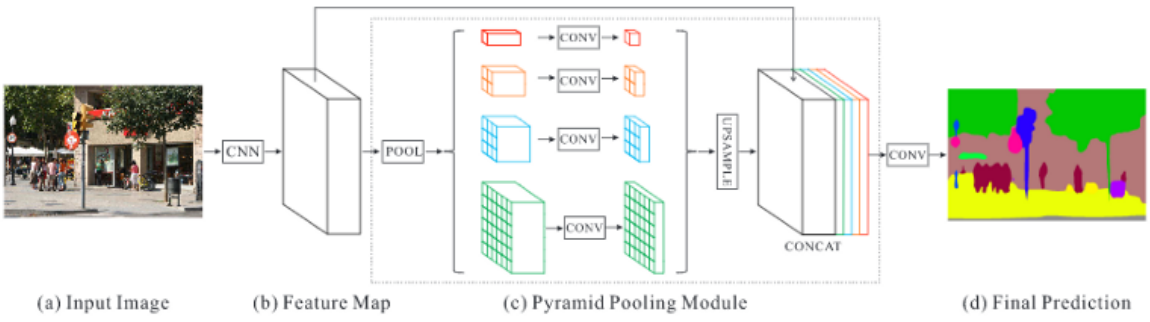
L'architecture Unet en comme son nom l'indique en U. C'est une approche encodeur – decodeur symétrique à la SegNet mais utilisant des skip connections pour la précision des limites. Dans le domaine médical Unet, grâce aux détails apportés par les skip connections , a été souvent préféré. Il est également utile pour les scènes intérieures/extérieures avec de petits objets.



- Approche PSPNet (Pyramid Scene Parsing Network) : https://towardsdatascience.com/review-pspnet-winner-in-ilsvrc-2016-semantic-segmentation-scene-parsing-e089e5df177d

PSPNet vient apporter de l'information globale contextuelle afin de mieux labeliser des objets. Par exemple quelque chose qui ressemble à une voiture au milieu d'un lac a plus de chance d'être un bateau.
La partie encodeur est assez classique avec un modèle de base de type imagenet pour recupérer la features map. La feature map est downsamplée à différentes échelles sur lesquelles on applique une couche de convolution avant de les upsampler à une même echelle puis de les concatener. On appliquer une dernière couche de convolution pour la segmentation finale.

Les petits objets sont bien capturés par les hautes résolutions (les faibles downsampling du PSP) alors que les gros objets sont bien capturés par les faibles résolutions (les forts downsampling du PSP)



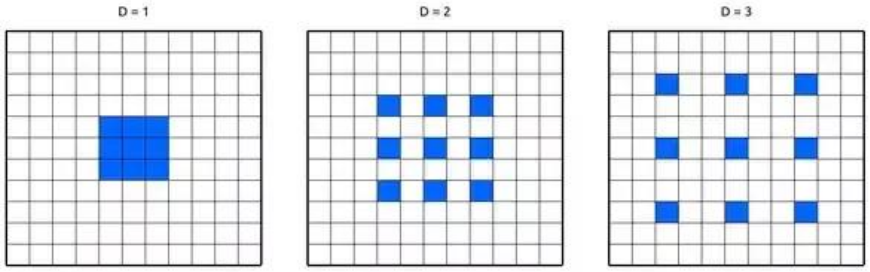
Pour des scènes interieurs et exterieurs, PSPNet est souvent utilisé car les objets sont de différentes tailles. L'input doit être assez large au départ.

- Approche DeepLab (Google) :

Deeplab a apporté 3 améliorations pour obtenir des sorties plus fines avec un cout de calcul plus faible :

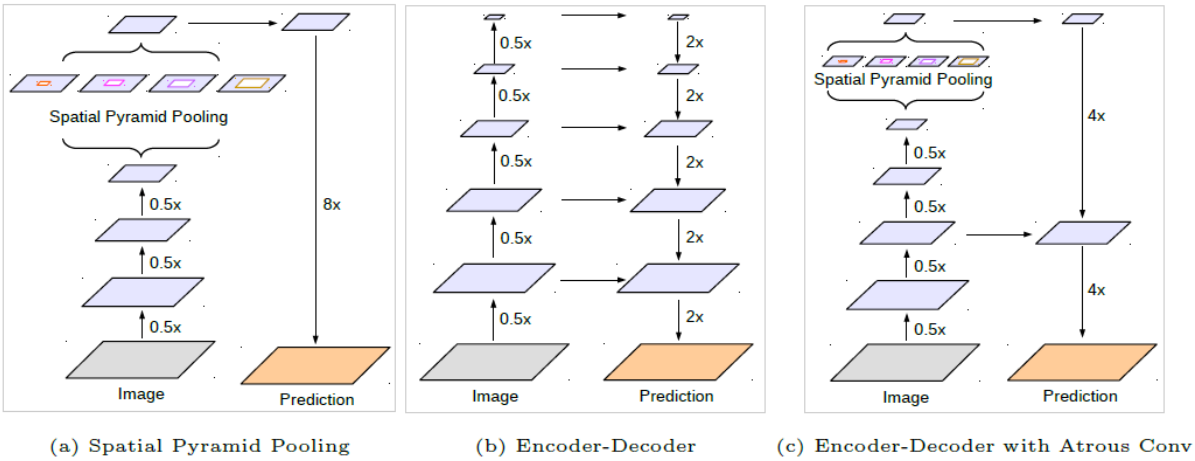
  - Atrous ou Dilated convolution

FCN32 par exemple qui downsample par 32 perd beaucoup d'information et a du mal à sortir de beaux outputs détaillés. La déconvolution par 32 est couteuse en calcul et mémoire.
La convolution dilatée augmente la taille du filtre en ajoutant des 0 entre. Elle augmente le champs de vision du filtre en élargissant le contexte. Si D=1, nous avons un filtre de convolution normal. Avec D=2 par exemple, nous avons un filtre 5x5 avec le même nombre de paramètre qu'un filtre 3x3.
Dans le cas de Deeplab on downsample par 8 seulement. Pour l'inférence Deeplab utilise un upsampling bilinéaire peu couteux contrairement à la déconvolution

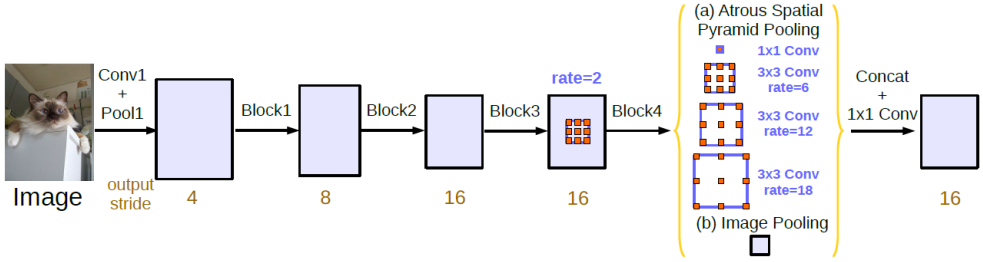


- ASPP : Atrous ou Dilated Spatial Pyramidal Pooling -Deeplab V2

Spatial Pyramidal Pooling est un concept permettant de capter l'information multi échelle d'une feature map sans contrainte de taille en entrée.
Dans le cas de l'ASPP, l'input est convoluée avec différents taux de dilatation et les sorties sont fusionnées, concaténées ensemble. L'image à segmenter peut-être de n'importe quelle taille



(a) Spatial Pyramid Pooling (b) Encoder-Decoder (c) Encoder-Decoder with Atrous Conv



- CRF : Conditional Random Fields

L'opération de pooling aide à réduire le nombre de paramètres mais apporte une propriété d'invariance : Le réseau n'est pas affecté par de faibles translations de l'input. Cette propriété rend les limites grossières et non clairement définis.
Le CRF essaye d'améliorer le résultat en prenant en compte les labels des pixels avoisinants

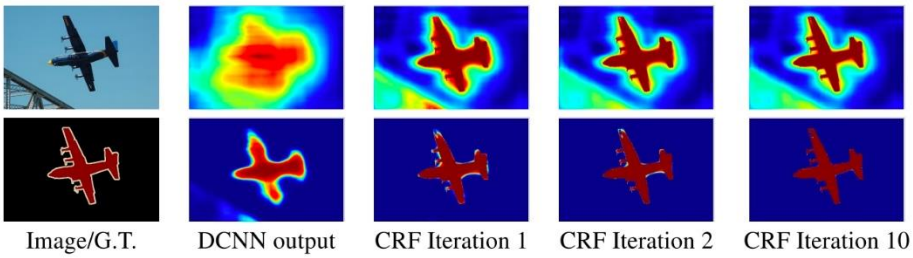


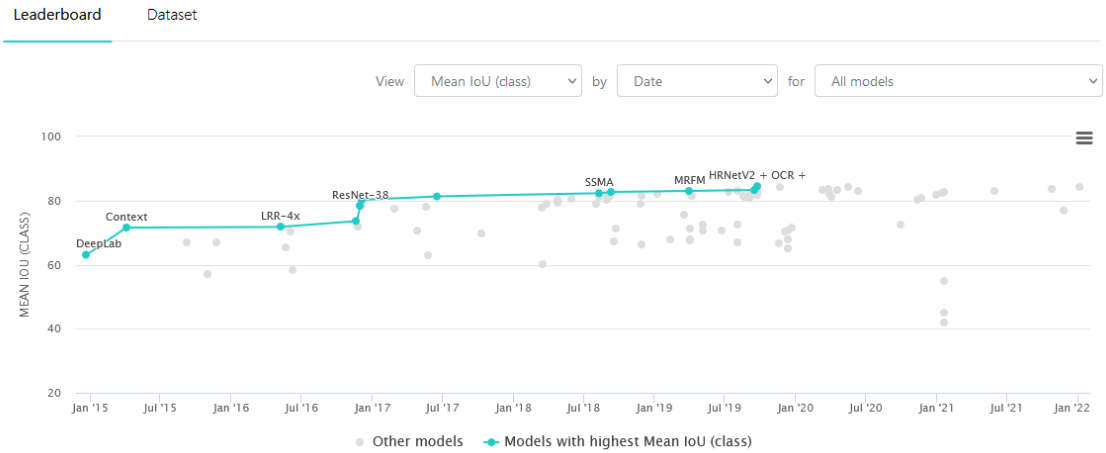
Figure 2: Score map (input before softmax function) and belief map (output of softmax function) for Aeroplane. We show the score (1st row) and belief (2nd row) maps after each mean field iteration. The output of last DCNN layer is used as input to the mean field inference. Best viewed in color.

Deeplab V3+ suggère d'avoir un decodeur au lieu d'un up sampling bilineaire x16

C. Etat de l'art de la segmentation sémantique

On peut suivre l'évolution de l'état de l'art via les performances jusqu'à aujourd'hui de différents modèles entrainés sur différents jeux de données comme cityscape par exemple sur le lien suivant :

https://paperswithcode.com/sota/semantic-segmentation-on-cityscapes



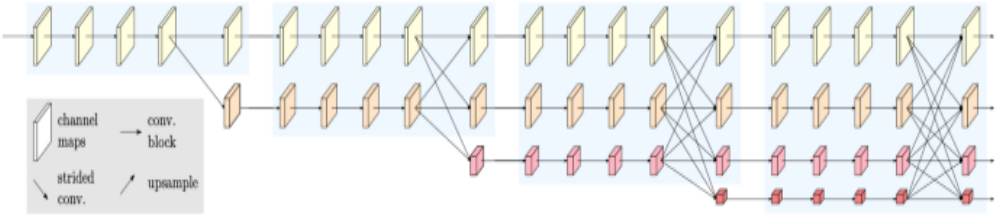
L'évolution historique est la suivante :

- 2017: PSPnet avec un mean Iou de 80.3%
- 2017: DeeplabV3 avec un mean IoU de 81.3%
- 2018: SSMA (Self supervised Model adaptation for multi modal segmentation) avec un mean IoU de 82.3%
    - La multi modal segmentation permet de combiner les features map de plusieurs modalités comme une photo classique et une photo thermique par exemple ou un lidar

- 2019 : HRNetV2+OCR+ avec un mean IoU de 84.5%
  https://towardsdatascience.com/hrnet-explained-human-pose-estimation-sematic-segmentation-and-object-detection-63f1ce79ef82

  https://arxiv.org/pdf/1909.11065v6.pdf

HRNET pour high resolution Network maintient une représentation haute résolution en connectant en parallèle les fortes aux faibles résolutions

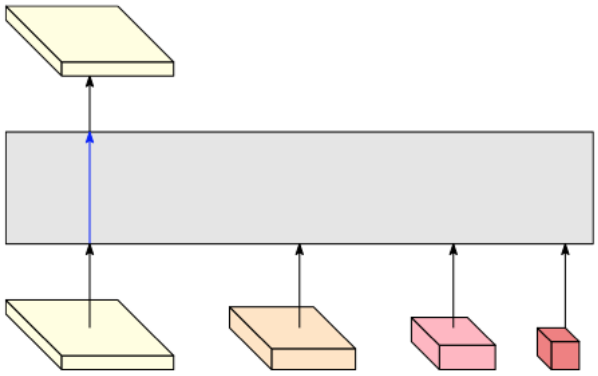


Chaque bloc bleu représente un multi résolution. Le channel jaune représente la plus forte résolution alors que la rouge représente la plus faible. Un bloc multi résolution est un groupe qui divise les channels d'entrée en plusieurs sous channels et en y faisant des convolutions classiques séparément. Chaque bloc se termine par une full connection au prochain bloc.

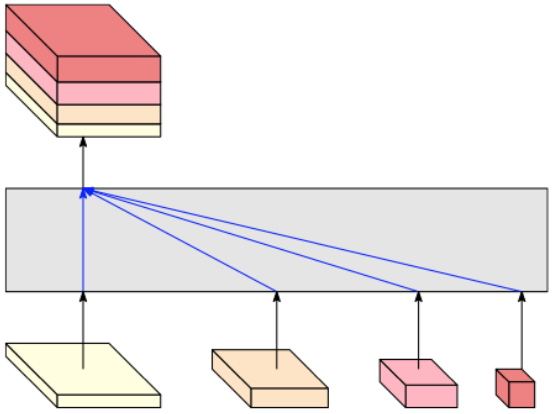
Pour quelles raisons HRNet fonctionne mieux que VGGNet, Restnet ou DenseNet ?

a) Les réseaux convolutionnels classiques travaillent en séries et les hautes résolutions sont retrouvés à partir de faibles résolutions, ce qui induit une perte d'informations.
b) L'approche HRNet permet de garder la haute résolution à travers tout le réseau
c) D'autres approches agrègent hautes résolutions et faibles résolutions up samplées mais HRNet répète la multi résolution

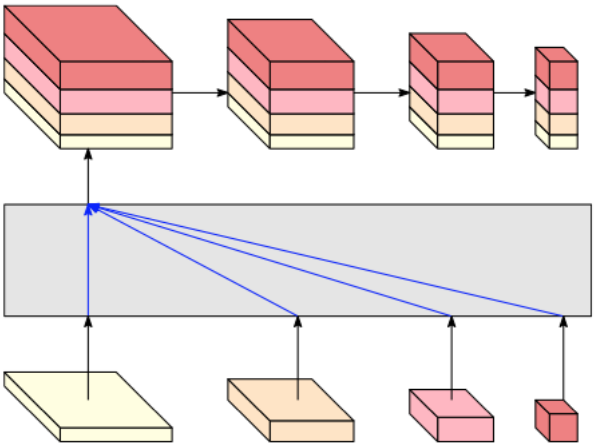
HRNetV1 ne gardait en bout de réseau que la résolution haute résolution et perdait une part d'information :



HRNetV2 a réglé le problème en upsamplant les résolutions plus faibles puis en concaténant afin de garder toute l'information :



HRNetV2+ , construit pour la détection d'objet, rajoute un average pooling :



Au final cette approche architecturale permet les hautes résolutions, améliorent la précision et les connexions sémantiques

OCR Object Contextual Representations For Semantic Segmentation:

OCR est une agrégation pondérée de toutes les représentations des régions d'objet. C'est une étape tardive permettant d'améliorer la performance de la segmentation sémantique. Les principales étapes de L'OCR sont les suivantes :

a) Diviser une petite zone de pixels en un jeu grossier de régions d'objet défini par leur classes issues d'un réseau de neurones profond type RestNet ou HRNet.
b) Ensuite on agrège tous les pixels appartenant à un objet
c) On augmente la représentation de chaque pixel avec l'aide la représentation contextuelle de l'objet (OCR). L'OCR est une agrégation pondérée de toutes les représentations de la région de l'objet avec des poids calculés en accord avec la relation entre chaque pixel et les régions d'objet.

(a) ASPP (b) OCR

Pour schématiser, dans le cas de l'ASPP (Dilated Spatial Pyramidal Pooling) de deeplab, le contexte de pixel rouge est une zone comprenant arrière-plan de l'objet et objet (filtres de convolutions dilatés), alors que pour l'OCR, le contexte ne correspond qu'a l'objet

- 2022 : Lawin Transformer Mean IoU : 84.4% (se rapproche de HRNetV2+OCR+)
  https://arxiv.org/pdf/2201.01615v1.pdf
  Les représentations multi échelle sont cruciales pour la segmentation sémantique. Les ViT , les transformers pour la vision sont puissants en matière de classification d'image. Ils sont également puissants en matière de segmentation d'image mais ont un certain cout de calcul.Pour réduire le cout, l'utilisation de HVT (hierarchical vision transformer) a émergé , mais jusqu'à présent le principal problème des ViT résidait dans un manque d'information contextuel multi échelle.
  Lawin Transformer introduit Le **LawinASPP (large window attention spatial pyramidal pooling)**

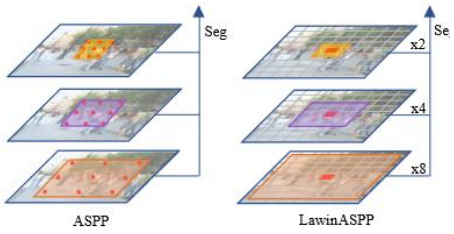


Figure 1. Difference between LawinASPP and ASPP. In ASPP, atrous convolution with different dilation rates captures representations at multple scalees. In contrast, LawinASPP replaces atrous convolution with our proposed *large window attention*. The red window represents the query area. The yellow, orange and purple windows represent the context area with different spatial sizes.

En termes de performance Lawin Transformer, fait un peu mieux que des PSPNET ou DeepLab V3 basés sur un réseau de base RESNET 101 et avec un cout de calcul moindre

Method	Backbone	FLOPs(G)↓	MS(%)↑
PSPNet [50]	ResNet101	2049	80.0
GCNet [5]	ResNet101	2203	80.7
PSANet [51]	ResNet101	2178	80.9
NonLocal [40]	ResNet101	2224	80.9
DeeplabV3 [9]	ResNet101	2781	80.8
CCNet [26]	ResNet101	2225	80.7
DANet [21]	ResNet101	2221	82.0
DNL [45]	ResNet101	2224	80.7
OCNet [49]	ResNet101	1820	81.6
DeeplabV3+ [10]	ResNet101	2032	82.2
SETR-PUP [52]	ViT-L*	—	82.2
Segmenter [36]	VIT-L*	—	81.3
SegFormer [43]	MiT-B5	1460	83.5
SegFormer†	MiT-B5	1460	83.7
Lawin	MiT-B5	1306	83.7
Lawin†	MiT-B5	1306	83.9
Lawin	Swin-L*	1797	84.2
Lawin†	Swin-L*	1797	**84.4**

Table 7. Performance Comparison on Cityscapes. The backbone marked with ⋆ indicates that it is pretrained on ImageNet22K. The method marked with † takes cropped input of 1024×1024.

D. Approche choisie

Après avoir tester des méthodes de machine Learning classiques, sur lesquelles nous ne reviendrons pas pour des raisons de performances assez éloignées des standards que nous pouvons obtenir aujourd'hui, s'est posée la question de savoir quelle approche deep learning nous allons tester et mettre en place.

Dans un premier temps, les images cityscapes étant des images extérieures, l'utilisation pour la partie encodeur d'un modèle pré entrainé ayant fait ses preuves sur imagenet nous semblait opportun. Nous aurions pu utiliser Resnet50 ou 101, efficientnet ou hrnet mais Vgg16 est assez populaire pour l'extraction de features et de plus assez léger, rendant la tache de transfer learning assez facile sur notre GPU.

Pour la partie décodeur, nous avons choisi Unet dans un premier temps pour son approche des skips connections qui permettent d'avoir des bords mieux définis comme déjà expliqué. Nous avons également testé le decodeur pspnet pour ajouter du contexte global.

Les FCN n'ont pas été testés car jugés trop simplistes pour notre tâche. En effet si les objets à segmenter ne sont ni gros ni en petit nombre, la perte d'information lors du upsampling est souvent trop importante pour obtenir une segmentation de qualité. Les dernières avancées deeplabv3, hrnetOCR et les tranformers n'ont pas été exploré faute de temps.

Concernant Les générateurs nous avons pu tester la mise en place d'augmentation lors de l'entrainement. L'augmentation sur le VGG-Unet n'a pas apporte de meilleurs résultats. Une des possibilités est que le test et le jeu de validation sont trop proches du jeu d'entrainement. Si nous dégradons fortement une image en contraste ou que nous prenons une image de scène routière ne venant pas de city scapes, les résultats de prédiction seront détériorés. Il ne faut aucun doute que si

le jeu de validation et que le jeu de test étaient également différents, l'augmentation aurait apporté davantage. Par ailleurs il faut veiller à contraindre les augmentations afin de ne pas trop les éloigner du jeu cible.

Voici les résultats que nous avons obtenu :

Modeles	type	temps d'entrainement	Validation Mean IoU	commentaires
Linear SVC	Machine Learning simple	sans gridsearch: 1 à 10 min pour 100 photos de 35K pixels fonction de la finesse du critère d'arret	0,17	résultats mauvais necessite plus d'entrainement necessite plus de feature engineering
Random Forest Classifier	Machine Learning simple	4,5 min pour 100 photos de 35K pixels	0,26	résultats mauvais necessite plus d'entrainement necessite plus de feature engineering
VGG PSPnet	Deep Learning	600 à 800 seconds par epoch de 2300 photos de 384x576 en fonction de l'augmentation 30 à 50 epochs 6 à 8h	0,54	résultats encourageants
VGG Unet	Deep Learning	600 à 800 seconds par epoch de 2300 photos de 256x512 en fonction de l'augmentation 30 à 50 epochs 6 à 8h	0,69 0,66 AVEC AUGMENTATION	bons resultats bors bien definis