Modèle Unet :

Cours CNN : <https://www.youtube.com/watch?v=DRWmP5ezyy0>

Cours UNET + code example : <https://www.youtube.com/watch?v=PrZ3r33gewQ&t=1508s>

Entrée :

Img + vrai masque (labelisé)

Sortie :

Masque prédit

Une image contenant diagramme, ligne, Tracé, capture d’écran

Description générée automatiquement

Architecture Modèle U-NET :

* Réseaux de convolution : voir shema.

Les réseaux de neurones convolutif (CNN) désigne une sous-catégorie des réseaux de neurones, **ils sont spécialement conçu pour traiter des images en input.**

Une image contenant texte, voiture, capture d’écran, véhicule

Description générée automatiquement

Qu’est-ce que la convolution ?

**Ex-image niveau de gris :**

Une image contenant texte, Police, capture d’écran, diagramme

Description générée automatiquement

Nous avons une image en niveau de gris avec chaque pixel de 0 à 255 (intensité du gris), on va prendre des carrée ici 3x3 et les multiplier à un noyau kernel 3x3, la multiplication se fait terme par terme, et on prend donc la somme. Puis nous déplaçons le carré orange et poursuivons pour construire notre plan de convolution. Ensuite nous normalisons le résultats pour mettre les valeur négative à zero.

Pour des images RGB, nous utilisons un kernel de taile 3x3x3 :   
  
Une image contenant capture d’écran, texte, conception

Description générée automatiquement

Le principe ensuite reste le même.

**Paramètre padding dans la convolution :** Une image contenant capture d’écran, ligne, Rectangle, conception

Description générée automatiquement

Le padding est utilisé pour avoir la même taille de l’entré en sortie , matrice de même taille que l’entré, cela en rajouter des pixel tout autour de l’image.

Cependant si padding = « valid », nous ne pouvons donc pas forcement ici avec une matrice image orange de 5x5 et un noyau de 3x3, nous ne pouvons pas obtenir un résultat de même taille que l’entré.

**2 ème paramètre c’est le stride , pas de déplacement de la fenêtre :** Une image contenant capture d’écran, conception

Description générée automatiquement

Ici le stride est de 2x2 (en bleu). En utilisant un stride supérieur à un cela permet de génerer des images de petites tailles.

Ici avec un pas de déplacement de 2 ( 2pixel), nous obtenons un plan de convolution de x plus petit que l’image.

**Après la convolution et la normalisation il y a le POOLING :**

En entré on a les features map, et nous les redimensionnons sans perdre en caractéristique importantes.

Nous utilisons des célules carré pour perdre le moins d’informations.

Une image contenant texte, Police, capture d’écran, ligne

Description générée automatiquement

But : Réduit le nombre de calcul et de paramètre dans le réseau, cela évite l’over-fitting (surapprentissage)

**Dernière étape : Convo/normalisation/pooling 🡪 la couche fully connected**

But : Prendre la totalité des valeurs de nos matrices précédemment calculé et à les empilés pour les exploités dans la couche d’entré de notre réseaux de neurones (fully connected)

Input : vecteur

Output : nouveaux vecteurs ( en utilisant une combinaison linéaire et peut etre une fonction d’activation ..)

La dernière couche FC permet de classifier l’image en entré de réseaux, elle renvoie un vecteur de taille n (n = nombre de classe dans le problème de classification) chaque éléments de vecteurs indique la probabilité de chaque classe.Une image contenant texte, voiture, véhicule, Véhicule terrestre

Description générée automatiquement

Vecteur 0.82 🡪 82% de chance de représenter une voiture.

BILAN ARCHITECTURE CNN :Une image contenant texte, Police, capture d’écran, logo

Description générée automatiquement

1. Image RGB
2. Construire des couches de convolution de même taille ou taille inférieur (padding)
3. Normalisation avec fonction reload
4. Réduire la taille des plans de convolutions : pooling
5. Procédure 1-4 répété plusieurs fois.
6. Flat layer = entré de notre réseaux de neurones complètement connecté (mettre à plat)