**R**épublique **A**lgérienne **D**émocratique et **P**opulaire
**M**inistère de L'**E**nseignement **S**upérieur et de la
**R**echerche **S**cientifique

**U**niversité **M**ouloud **M**ammeri De **T**izi-**O**uzou

**F**aculté De **G**énie **E**lectrique Et D'informatique
**DEPARTEMENT D'AUTOMATIQUE**

# Mémoire de Fin d'Etude
# de MASTER ACADEMIQUE

Spécialité : AUTOMATIQUE et Informatique Industrielle

Présenté par
Nacerdine LARBI

Mémoire dirigé par **Sadia ALKAMA**

# Segmentation d'images avec le Deep Learning

*Mémoire soutenu publiquement le 27 septembre 2018 devant le jury composé de :*

**M. Ouerdia CHILALI**
Présidente

**M. Malika AIT AIDER**
Examinatrice

**M. Farida DORBANE**
Examinatrice

# Remerciements

Avant tout développement de ce travail, il apparaît opportun de commencer par des remerciements. A ceux qui m'ont beaucoup appris au cours de mon cursus, et même à ceux qui ont juste eu la gentillesse d'en faire un moment profitable et très agréable.

Je tiens à exprimer mon immense gratitude à **Mme Alkama**, ma promotrice qui m'a accompagné tout au long de cette expérience professionnelle avec beaucoup de patience et de pédagogie.

Je remercie également tous les membres de ma famille et ma fiancée pour leurs aides précieuses, leurs soutient et encouragements.

Tous mes remerciements vont aussi aux membres du jury ayant accepté d'analyser et de corriger mon travail.

Je tiens aussi, à remercier vivement tout les enseignants que j'ai eu durant ma formation en particulier ceux du département d'automatique.

Ainsi que tous les ami(e)s qui ont colorés mes journées et soutenus tout au long des périodes difficiles

Enfin, je remercie l'ensemble des personnes que j'ai été amené à côtoyer durant ma formation, pour les conseils qu'ils ont pu me prodiguer.

**LARBI Nacerdine**

# Sommaire

## ❖ Chapitre3 : TESTS

**Introduction générale**

Nous vivons dans un monde numérique, où les informations sont stockées, traitées, indexées et recherchées par des systèmes informatiques, ce qui rend leur récupération une tâche rapide et pas cher. Au cours des dernières années, des progrès considérables ont été réalisés dans le domaine de la classification d'images. Ce progrès est dû aux nombreux travaux dans ce domaine et à la disponibilité des bases d'images internationales qui ont permis aux chercheurs de signaler de manière crédible l'exécution de leurs approches dans ce domaine, avec la possibilité de les comparer à d'autres approches qui utilisent les mêmes bases.

Dans la fin des années 80 Yan le Cun a développé un type de réseau particulier qui s'appelle le réseau de neurone convolutionnel, ces réseaux sont une forme particulière de réseau neuronal multicouche dont l'architecture des connexions est inspirée de celle du cortex visuel des mammifères. En 1995, Yan le cun et deux autres ingénieurs ont développé un système automatique de lecture de chèques qui a été déployé largement dans le monde. À la fin des années 90, ce système lisait entre 10 et 20 % de tous les chèques émis aux États-Unis. Mais ces méthodes étaient plutôt difficiles à mettre en œuvre avec les ordinateurs de l'époque, et malgré ce succès, les réseaux convolutionnels et les réseaux neuronaux plus généralement ont été délaissés par la communauté de la recherche entre 1997 et 2012.

En 2011 et 2012 trois événements ont soudainement changé la situation. Tout d'abord, les GPU (Graphical Processing Unit) capables de plus de mille milliards d'opérations par seconde sont devenus disponibles pour un prix moins cher. Ces puissants processeurs spécialisés, initialement conçus pour le rendu graphique des jeux vidéo, se sont avérés être très performants pour les calculs des réseaux neuronaux. Deuxièmement, des expériences menées simultanément à Microsoft, Google et IBM avec l'aide du laboratoire de Geoff Hinton ont montré que les réseaux profonds pouvaient diminuer de moitié les taux d'erreurs des systèmes de reconnaissance vocale. Troisièmement plusieurs records en reconnaissance d'image ont été battus par des réseaux de neurones convolutionnels. L'événement le plus marquant a été la victoire éclatante de l'équipe de Toronto dans la compétition de reconnaissance d'objets « ImageNet ». La diminution des taux d'erreurs était telle qu'une véritable révolution. Du jour au lendemain, la majorité des équipes de recherche en parole et en vision ont abandonné leurs méthodes préférées et sont passées aux réseaux de neurones convolutionnels et autres réseaux neuronaux. L'industrie d'Internet a immédiatement saisi l'opportunité et a commencé à investir massivement dans des équipes de recherche et développements en Deep Learning (apprentissage profond).

Dans notre projet on va utiliser le Deep Learning basé sur les réseaux de neurones convolutionnels les plus répandus pour faire une segmentation d'image entrainé avec la base d'images CamVid et par la suite on va tester le réseau formé avec d'autre images.

Pour ce faire, nous avons structuré notre mémoire en trois chapitres :

- Dans le premier chapitre on va présenter les notions de base de la segmentation d'images.
- Le deuxième chapitre est consacré au Deep Learning.
- Dans la troisième chapitre, on va montrer la partie expérimentale de notre travail et on discute les différents résultats obtenus.
- On termine par une conclusion générale.

## 1.1 Introduction

Le traitement d'images est une discipline de l'informatique et des mathématiques appliquées qui étudie les images numériques et leurs transformations, dans le but d'améliorer leur qualité ou d'en extraire de l'information. La segmentation d'images est une étape importante et primordiale dans le processus de traitement et d'analyse d'images qui a pour but de partitionner une image en régions homogènes et regrouper des pixels ayant des propriétés communes suivant des critères prédéfinis. La figure 1.1 représente un exemple de segmentation d'images .

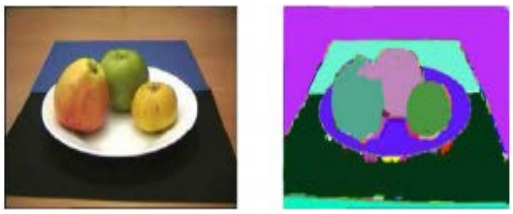


Figure 1.1 : Segmentation d'une image couleur

Sur cette figure, on voit bien que chaque objet de l'image se voit attribuer une couleur, d'où la séparation en régions dites homogènes. Cependant on voit qu'il peut y avoir des défauts de reconnaissance et donc y avoir des confusions entre les régions comme c'est le cas ici entre la pomme du milieu et celle se situant à sa gauche car elles ont une zone en violet commune.

## 1.2 Définition de la segmentation d'images

La segmentation est une des étapes critiques de l'analyse d'images qui conditionne la qualité des mesures effectuées ultérieurement. C'est généralement une première étape d'un traitement plus complexe comme la reconnaissance de formes. Elle permet de cerner les formes des objets sur lesquels doit porter l'analyse.de délimiter des régions (l'intérêt et de les extraire du fond). Une bonne méthode de segmentation sera celle qui permettra d'arriver à une bonne interprétation. Elle devra donc avoir simplifié l'image sans pour autant en avoir trop réduit le contenu.

La segmentation est une décomposition de l'image I en n régions $R_i$ tel que :

1. $\cup_i R_i = I$
2. $R_i \cap R_j = \emptyset$
3. P($R_i$) = vrai
4. P($R_i \cap R_j$ ) = $faux$ .

En conclusion La segmentation consiste à:
-Regrouper les pixels de l'image qui partagent une même propriété pour former des régions homogènes.
-Répartir l'ensemble de pixels de l'image en différents groupes.
-découper l'image en région. Une région est caractérisée par contours et par

homogénéité (par exemple, même couleur).
-partitionner une image en un ensemble de régions connexes et disjointes.
-la recherche de zones de l'image possédant des attributs communs, comme la luminosité, la couleur ou plus rarement la texture.

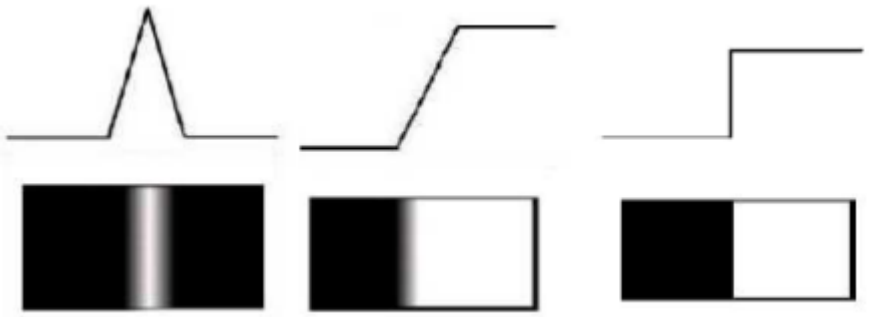
### 1.3 Différentes approches de segmentation

Généralement, les méthodes de segmentation sont regroupées en trois approches chacune ayant des avantages et ses domaines d'application et elles sont par fois complémentaires , ces approches sont :

1. Segmentation basée sur les contours (en anglais : edge-based segmentation)
2. Segmentation basée sur les régions (en anglais : régions-based segmentation).
3. Segmentation en utilisant la classification .

#### 1.3.1 approches contour

L'approche contour consiste à identifier les changements entre les régions. En général, un élément de contours est un point de l'image appartenant à la frontière de deux ou plusieurs objets ayant des niveaux de gris différents..La figure suivant montre quelques modèles de contours.



Toit Rampe Escalier

Figure 1.2 : Quelques modèles de contours

- Marche d'escalier : le contour est net (contour idéal).
- Rampe : le contour est plus flou.
- Toit : il s'agit d'une ligne sur un fond uniforme

Plusieurs méthodes ont été adaptées pour la détection des contours , on distingue principalement les méthodes dérivatives, les méthodes analytiques et les contours déformable.

Les méthodes dérivatives sont les plus utilisées pour détecter des transitions d'intensité par différenciation numérique première ou deuxième dérivé(voir figure 1.3). A chaque position, un opérateur est appliqué afin de détecter les transitions significatives au niveau de l'attribut

de discontinuité choisi. Le résultat est une image binaire constituée de points de contours et de points non-contours.

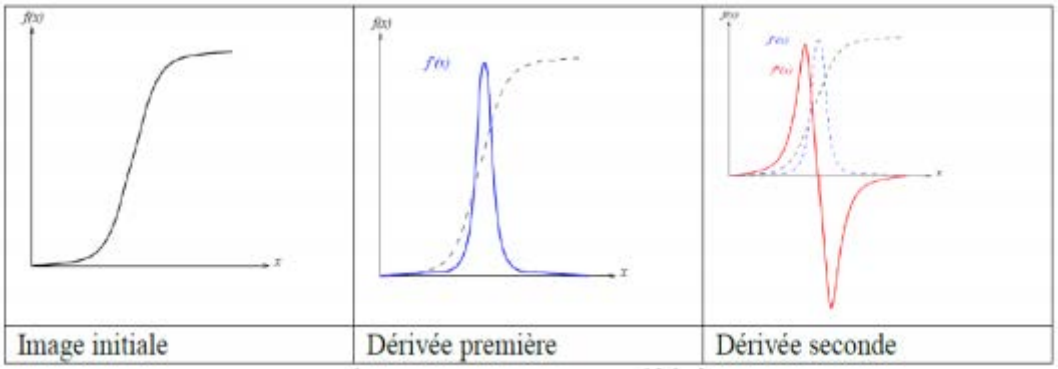



Figure 1.3 : contour et ses dérivées.

Ils existe plusieurs opérateurs de gradient (première dérivée), parmi eux il ya les masques de Robert,de Prewitt et de Sobel **[ 1 ].**
La figure ci dessous fournie une image à laquelle sont appliqués ces operateurs .


Figure 1.4 : Détection de contour par les différents filtres

La figure suivante représente la détection de contour en utilisant l'opérateur Laplacien **[1]** qui est basé sur la seconde derivée.

Figure 1.5 : image originale (à gauche), contour détecté par Laplacien

Les Méthodes analytiques consistent à trouver un filtre optimal satisfaisant les 3 contraintes suivantes :

- **Une bonne détection :** faible probabilité d'oublier un vrai point de contour et une faible probabilité de marquer un point image comme contour alors qu'il ne l'est pas.

- **Une bonne localisation :** les points contours doivent être le plus près possible de leur position réelle dans l'image.

- **Une réponse unique :** à un contour unique : un point de contour ne doit être détecté qu'une seule fois par le filtre mis en œuvre.

Le détecteur de contour de Canny est le plus utilisé **[2].**
Au filtre de Canny, Deriche a proposé un autre filtre (condition initiale différente) qui permet une simplification de son implémentation..

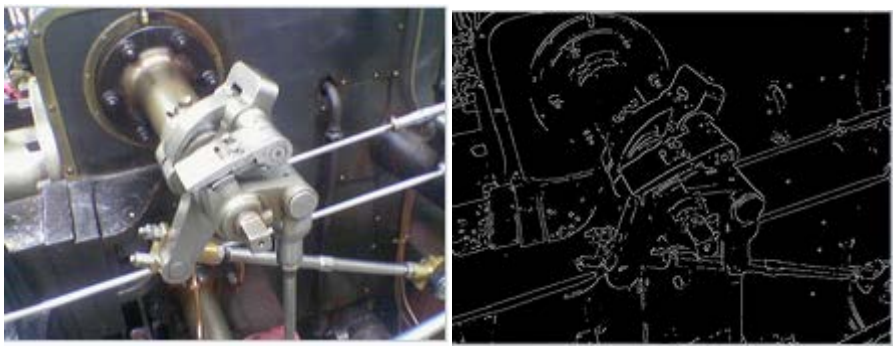


Figure 1.6 Exemple d'application du filtre de Canny

Les modèles déformables, introduits par Kasssont sont aussi connus sous les noms de « snakes » ou « contours actifs ».

L'intérêt principal des contours actifs est de détecter des objets dans une image en utilisant les techniques d'évolution de courbes. L'idée est de partir d'une courbe initiale, généralement un carré ou un cercle, et de la déformer jusqu'à obtenir le contour de l'objet.

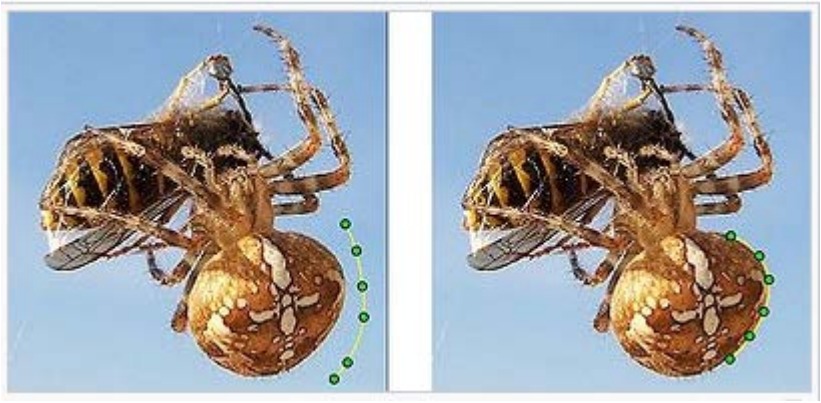


Figure 1.7 : Exemple d'un *snake* qui épouse le corps de l'araignée

### 1.3.2 Approche région

La segmentation d'image par l'approche région consiste à découper l'image en régions. Les pixels adjacents sont regroupés en régions distinctes selon un critère d'homogénéité ou de similarité donnée. Ce critère peut être, par exemple, le niveau de gris, couleur, texture…etc.

Un processus de groupement est répété jusqu'à ce que tous les pixels dans l'image soient inclus dans des régions. Cette approche vise, donc, à segmenter l'image en se basant sur des propriétés intrinsèques des régions.

Il existe plusieurs méthodes telles que la segmentation par croissance de région, par division de région, et par fusion de région que nous présentons ci-dessous.

#### 1.3.2.1 Croissance de région (région growing)

Cette technique consiste à faire progressivement accroître les régions autour de leur point de départ.

Le principe de l'agrégation de pixel est le suivant : on choisit un germe (Le point de départ est le choix d'un ensemble de pixels appelés « germes ») et on fait croitre ce germe tant que des pixels de son voisinage vérifient le test d'homogénéité. Lorsqu'il n'y a plus de pixels candidats dans le voisinage, on choisit un nouveau germe et on itère le processus.

Figure 1.8 : Croissance progressive des régions.

Parmi les avantages de cette technique, nous pouvons citer :

- La simplicité et la rapidité de la méthode.
- La segmentation d'objet à topologie complexe.
- La préservation de la forme de chaque région de l'image.

Cependant, il existe plusieurs inconvénients comme :

.

- Une mauvaise sélection des germes ou un choix du critère de similarité mal adapté peuvent entraîner des phénomènes de sous-segmentation1ou de sur-segmentation.
- Il peut y avoir des pixels qui ne peuvent pas être classés.

**1.3.2.2 Segmentation par fusion de régions (Merge)**

Les techniques de réunion (region merging) sont des méthodes ascendantes où tous les pixels sont visités. Pour chaque voisinage de pixel, un prédicat P est testé. S'il est vérifié les pixels correspondants sont regroupés dans une région.

Les inconvénients de cette méthode se situent à deux niveaux :

- Cette méthode dépend du critère de fusion qui peut influer sur le résultat final de la segmentation.
- Elle peut introduire l'effet de sous-segmentation.

**1.3.2.3 Segmentation par division de régions (Split)**

La division consiste à partitionner l'image en régions homogènes selon un critère donné. Le principe de cette technique est de considérer l'image elle-même comme région initiale, qui par la suite est divisée en régions. Le processus de division est réitéré sur chaque nouvelle région (issue de la division) jusqu'à l'obtention de classes homogènes

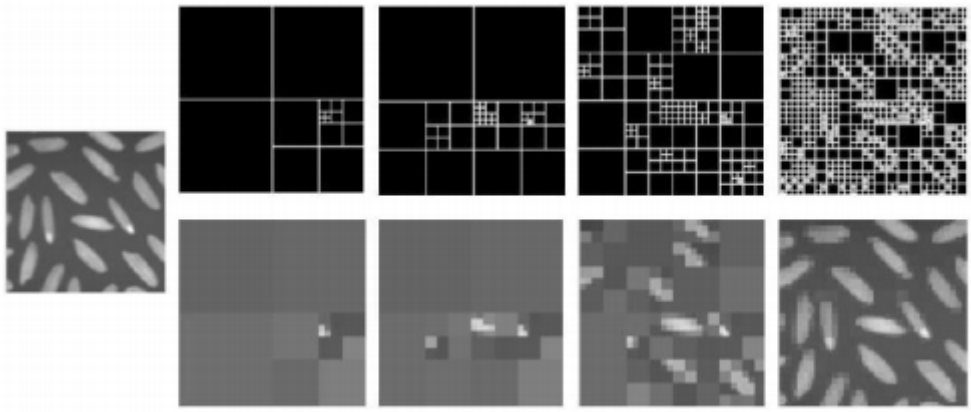


Figure 1.9: Décompositions successives des blocs.

Cette méthode présente un inconvénient majeur qui est la sur-segmentation. Toutefois, ce problème peut être résolu en utilisant la méthode de division-fusion que nous présentons dans ce qui suit.

**1.3.2.4 Segmentation par division-fusion (Split and Merge)**

Ces méthodes combinent les deux méthodes décrites précédemment, la division de l'image en de petites régions homogènes, puis la fusion des régions connexes et similaires au sens d'un prédicat de regroupement. Deux régions seront fusionnées si elles répondent aux critères de similarité des niveaux de gris et d'adjacence de régions .On s'arrête quand le critère de fusion n'est plus vérifié.

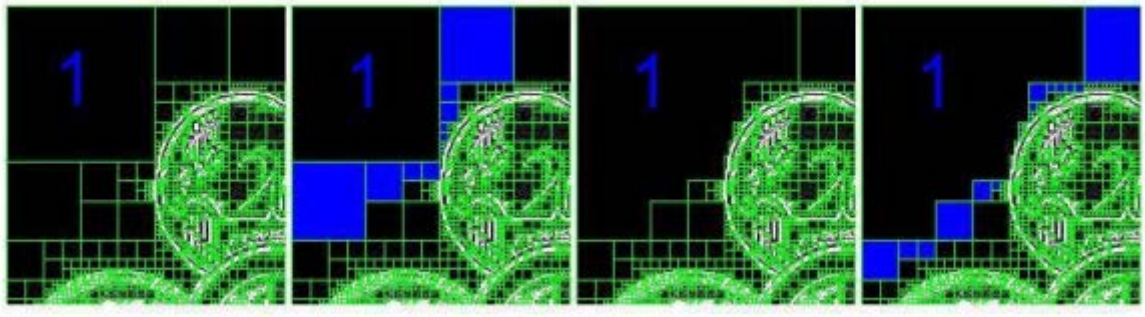



Figure 1.10 **:** Agrégation itérative des blocs similaires au bloc 1.

Les inconvénients de cette méthode se situent à trois niveaux :

- Les limites des régions obtenues sont habituellement imprécises et ne coïncident pas exactement aux limites des objets de l'image.

- La difficulté d'identifier les critères pour agréger les pixels ou pour fusionner et diviser les régions.

### 1.3.3 La segmentation basée sur la classification des pixels

De nombreux travaux sur la classification des pixels qui consiste à affecter à chaque pixel de l'image une classe qui définit les régions à extraire de l'image .

Un classifieur désigne tout outil de reconnaissance qui pour un vecteur reçu en entrée, donne des informations sur sa classe d'appartenance. Cet outil peut s'écrire sous la forme d'une fonction, qui à l'aide de descripteur d'un vecteur x à reconnaître, qui à l'aide de descripteur d'un vecteur x à reconnaître, attribue à x la classe $C_i$ parmi k classe possible (i ∈[1,k]) Nous pouvons alors définir un classifieur par la relation suivante où l'ensemble K = $(C_1, \cdots, C_k)$

$$e : x \in R^n \rightarrow K \qquad (1.1)$$

La classification des pixels peut étre supervisées ou non supervisée.

#### 1.3.3.1 Classification de pixels non supervisée

La classification de pixels non supervisée appelée aussi classification de pixels sans apprentissage consiste à découper l'espace de représentation en zones homogènes selon un critère de vraisemblance entre les individus. Cette approche est utilisée pour effectuer une classification de pixels en aveugle c'est-à-dire sans connaissance a priori sur l'image et ne nécessite donc pas de phase d'apprentissage.

#### 1.3.3.2Classification de pixels supervisée

La classification de pixels supervisée appelée aussi classification de pixels avec apprentissage consiste à définir une fonction de discrimination effectuant un découpage de l'espace de représentation à partir d'une connaissance a priori de l'image. Ce type de classification nécessite la création d'une base d'apprentissage faisant intervenir une segmentation de référence.

## 1.4 Quelques algorithmes de segmentation d'images

### 1.4.1 Algorithmes de classification de pixels non-supervisée

Nous allons à présent présenter trois des algorithmes de classification de pixels non-supervisée à savoir :

1. L'algorithme des k-moyennes .
2. L'algorithme des C-moyennes floues .
3. L'algorithme de Fisher.

### 1.4.1.1 Algorithme des k-moyennes

L'un des algorithmes les plus connu, pour la classification est l'algorithme K -means largement adopté en traitement d'images vu sa simplicité de mise en œuvre et sa capacité à fournir une bonne approximation de la segmentation recherchée. C'est un algorithme itératif qui minimise la somme des distances entre chaque pixel et le centre des classes. Ces centroïdes sont initialement placés le plus loin possible les uns des autres afin d'optimiser la qualité des résultats obtenus. Le principe de cet algorithme consiste à échanger des pixels entre deux classes jusqu'à ce que la somme des distances intra classes ne puisse plus diminuer. Le résultat idéal serait un ensemble de classes compacts et clairement séparés. Néanmoins cette méthode nécessite comme unique paramètre un nombre de classes K prédéfini a priori par l'utilisateur **[3].**

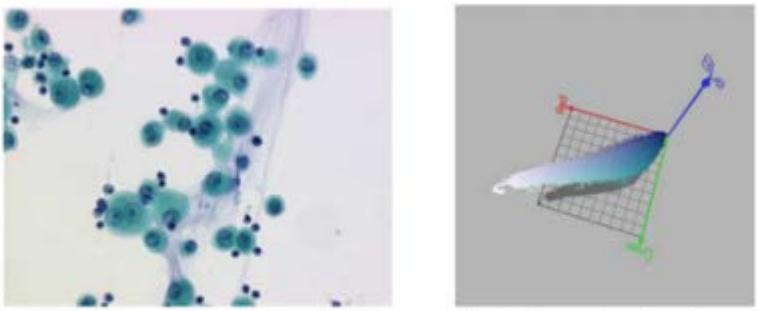
La figure 1.9 présente le résultat obtenu de l'application de l'algorithme K-means sur une image test où le nombre de classe K a été choisi arbitrairement. Sur cette image, nous remarquons qu'un mauvais choix de la valeur de K conduira à un résultat qui n'a pas de rapport avec l'image originale.

Figure 1.11 : (a) l'image originale, (b) l'image classée par K-means avec K=3, (c) l'image classée par k-means avec K=5

Les résultats de classification établis par l'algorithme des k-moyennes peuvent fluctuer selon les paramètres d'entrés qui sont **[4]** :

### 1.4.1.2. Algorithme des C-moyennes floues (FCM)

L'algorithme des C-moyennes floues (Fuzzy C-means) diffère peu de l'algorithme des k-moyennes si ce n'est qu'il est basé sur une classification floue. Il n'associe pas directement une classe Ci à un pixel x de l'image mais plutôt un degré d'appartenance à une classe (compris entre 0 et 1). Il est basé sur l'optimisation d'un critère quadratique où la somme des écarts quadratiques intra-classes doit être minimale **[4].**



(a) Image initiale (b) Représentation dans l'espace RGB

Figure 1.12 : Représentation d'une image dans l'espace RGB

Les deux paramètres d'entrées de cet algorithme devant être définis au préalable sont respectivement le paramètre m et le nombre de classes k.

### 1.4.1.3 Algorithme de Fisher

L'algorithme de Fisher consiste à effectuer une classification de pixels de l'image en k classes, en utilisant le partitionnement d'un histogramme de niveaux de gris en k classes disjointes tel que la somme des variances des classes soit minimale. Dans le cadre de la couleur, il convient d'appliquer cet algorithme séparément sur les trois composantes de l'espace de représentation couleur.

### 1.4.2 Algorithmes de classification de pixels supervisée

Nous allons à présent présenter quatre algorithmes de classification de pixels supervisée à savoir :

1. Algorithme des k-plus proches voisins
2. Algorithme de Bayes
3. Algorithme des Machines à support de vecteurs
4. Algorithme des Réseaux de Neurones Multi Couches

### 1.4.2.1 Algorithme des k-plus proches voisins

La méthode des k-plus proches voisins (k−P P V ) est une méthode d'estimation non paramétrique de densité **[5, 6].** Elle consiste à rechercher à partir d'une base d'apprentissage et d'une distance dénie sur l'espace des données les K plus proches voisins d'un élément. Ce qui revient à calculer l'estimation de densité $r_i$ de la classe $C_i$ au point x déni par la relation suivante :

$$r_i(x) = \frac{k_i(x)}{n_i \times V(x)} \qquad (1.8)$$

Où $k_i(x)$ est le nombre de points de $C_i$ appartenant aux k − P P V de $x$ , $n_i$ est le cardinal de la classe $C_i$ et V(x) le volume de la plus petite boule contenant le k − P P V de x.

La probabilité d'appartenance à une classe de cet élément x est alors proportionnelle au nombre d'exemples de cette classe parmi ses k −P P V . La classe de l'élément x est alors celle ayant la plus grande probabilité. Les résultats dépendant fortement de la valeur de k, il convient de choisir une valeur minimisant le risque d'erreur déterminé à partir d'une base de tests.

La figure 1.13 illustre une classification pixellaire effectuée par l'algorithme des k − P P V sur deux images d'une base de tests.

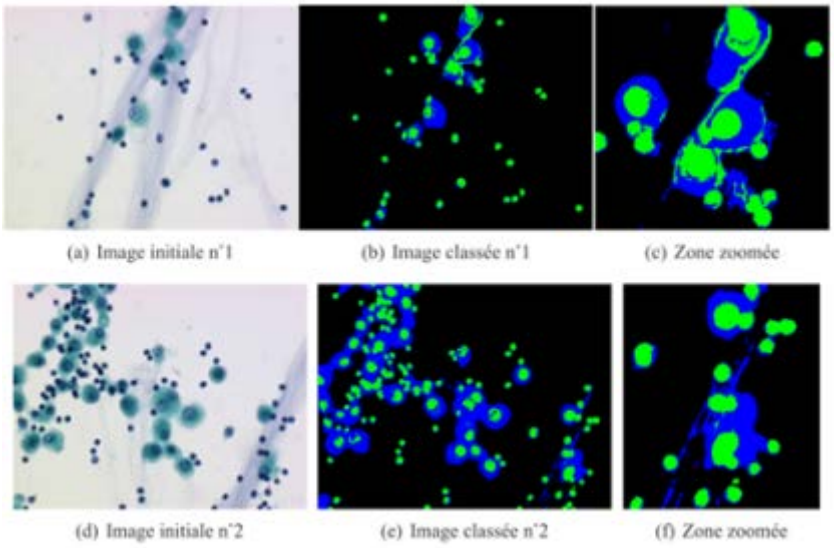



Figure1.13 : Classification pixellaire par l'algorithme des k − P P V

### 1.4.2.2 Algorithme de Bayes

Cet algorithme de classification est basé sur la théorie de décision Bayesienne **[7].** La méthode utilisée est une approche statistique supposant que le problème de classification peut être exprimé en termes probabilistes. L'hypothèse de base est généralement que la fonction de densité de probabilité d'un pixel x d'appartenir à une classe $C_i$ est de forme gaussienne.

L' algorithme de Bayes cherche à déterminer pour chaque élément x la classe $C_i$ qui maximise la probabilité de contenir cet élément.

$$f(x, C_i) = -\frac{1}{2}(x-\mu_i)^T \sum_i^{-1}(x-\mu_i) - \frac{1}{2}\log \left| \sum_i \right| - log\, p_i + \frac{k}{2} log \quad 2\Pi \quad (1.9)$$

où k est le nombre de classes, $\mu_i$ la moyenne des éléments de la classe $C_i$, $\Sigma_i$ la matrice de variance-covariance et $p_i$ la probabilité a priori de la classe $C_i$ .

La figure 1.14 illustre une classification pixellaire effectuée par l'algorithme de Bayes sur deux images d'une base de tests.

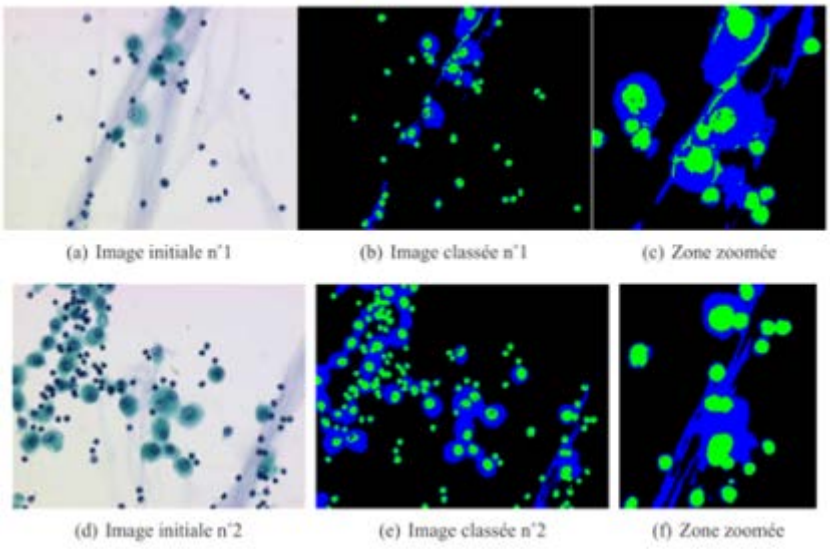



Figure 1.14 : Classification pixellaire par l'algorithme de Bayes

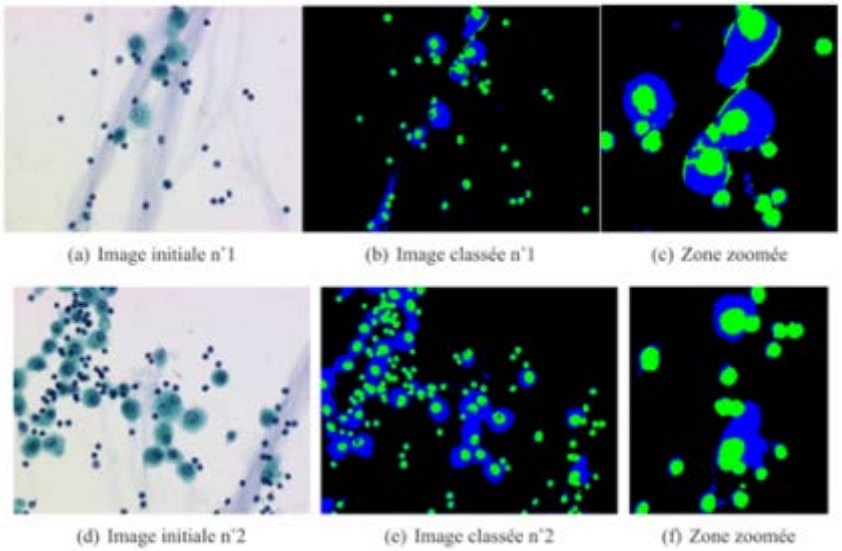



Figure 1.15 : Classification pixellaire par l'algorithme des SV M

### 1.4.2.3 Algorithme des Réseaux de Neurones Multi Couches

Les réseaux de neurones multicouches (MLP : Multi Layer Perceptron) sont utilisés depuis de nombreuses années dans le domaine de la classification étant donné leurs bons résultats. L'idée principale des MLP est de grouper des neurones par couche et de connecter complètement les neurones des couches adjacentes. Typiquement, les couches sont organisées de la façon suivante :

une couche d'entrée (paramètres caractérisant un objet), une ou plusieurs couches cachées (augmentant les possibilités d'apprentissage), et une couche de sortie (fournissant la classe trouvée pour un objet)

La phase d'apprentissage consiste à modifier les poids reliant les neurones de façon à ce que la classe en sortie correspond à celle de l'objet présenté en entrée. Cette modification est effectuée par un algorithme de rétro-propagation **[8]** . Afin d'obtenir une bonne généralisation, il reste deux paramètres à régler : la durée de l'apprentissage et le nombre de neurones cachés. Ils sont choisis de façon à minimiser le risque d'erreur déterminé à partir d'une base de tests.

La figure 1.16 illustre une classification pixellaire effectuée par l'algorithme des Réseaux de Neurones Multi Couches sur deux images d'une base de tests.

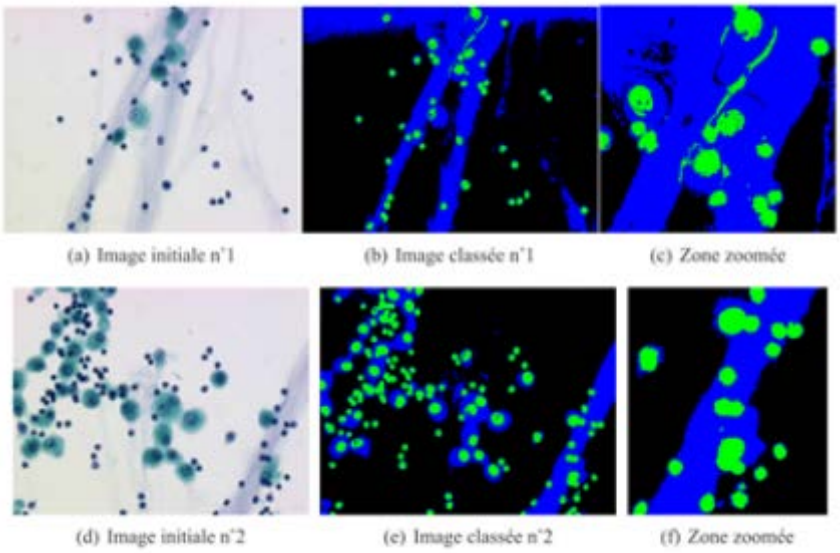



Figure 1.16 : Classification pixellaire par l'algorithme des RNMC

### 1.5 Segmentation sémantique d'images

La segmentation sémantique associe une étiquette ou une catégorie à chaque pixel d'une image. Elle permet de reconnaître un ensemble de pixels qui forment des catégories distinctes.

La séparation d'images en deux classes est un exemple simple de segmentation sémantique. Par exemple, à la figure 1.17, une image présentant une personne à la plage est associée à une version montrant les pixels de l'image segmentés en deux classes distinctes : la personne et l'arrière-plan

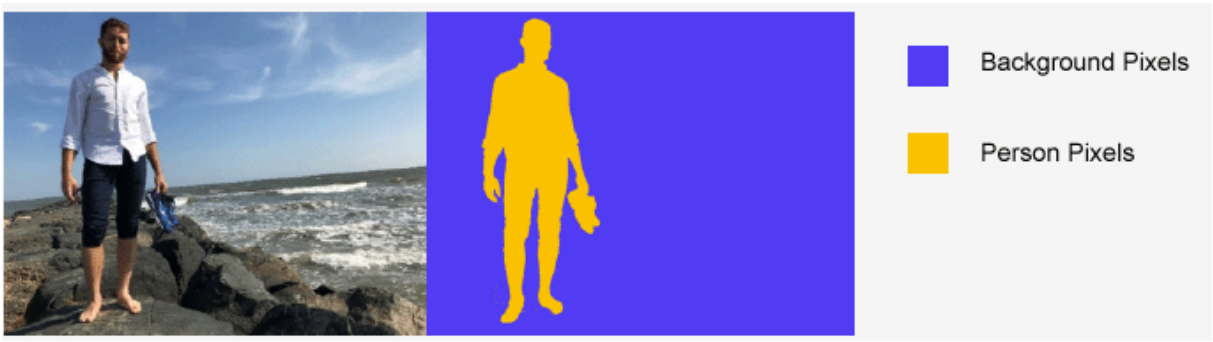



Figure 1.17 : Image et étiquette des pixels

La segmentation sémantique étiquette les pixels d'une image c'est ce qui la rend utile dans des applications de divers domaines :

**Conduite autonome :** pour identifier un parcours conduisible pour les véhicules en distinguant la route des obstacles tels que les piétons, trottoirs, poteaux et autres véhicules.

**Contrôles industriels :** pour détecter les défauts dans des matériaux, comme le contrôle des composants électroniques.

**Imagerie satellite :** pour identifier les montagnes, les rivières, les déserts et autres terrains.

**Imagerie médicale :** pour analyser et détecter les anomalies cancéreuses dans les cellules.

**Vision robotique :** pour identifier les objets et le terrain et s'y déplacer.

## 1.6 Conclusion

Dans ce chapitre, nous avons définis la segmentation d'images en développant les différentes approches.

Dans le chapitre suivant nous allons détailler la technique de l'apprentissage profond (Deep learning) qui sera utilisée pour la segmentation d'images sémantique .

## 2.1 introduction

Ces derniers temps, le Deep Learning (apprentissage profond) attire beaucoup l'attention, et pour cause : le niveau de performance atteint est tout simplement extraordinaire . Les algorithmes de Deep Learning surpassent les capacités humaines en ce qui concerne la classification des images .Grâce au Deep Learning, les voitures autonomes sont capables de reconnaître un panneau stop ou de différencier un piéton d'un lampadaire. Cette méthode permet également de contrôler vocalement des smartphones, des tablettes, des téléviseurs ou des haut-parleurs mains libres.

## 2.2 Historique et domaine d'application

Depuis 2012, les algorithmes à base de Deep Learning semblent prêts à résoudre bien des problèmes : reconnaitre des visages comme le propose DeepFace, vaincre des joueurs de go( le jeu de go est un jeu de stratégie, très répandu en extrême-orient, qui se joue à deux et qui consiste à former des territoires en posant des pions, ou pierres, sur un plateau. Le but est de marquer plus de points que l'adversaire en créant plus de territoires que lui et/ou en capturant ses pierres), ou de poker ou bientôt permettre la conduite de voitures autonomes ou encore la recherche de cellules cancéreuses.

Les fondements de ces méthodes ne sont pas si récents : le Deep Learning a été formalisé en 2007 à partir des nouvelles architectures de réseaux de neurones dont les précurseurs sont McCulloch et Pitts en 1943 **[9].** Suivront de nombreux développements comme les réseaux de neurones convolutifs de Yann Le Cun et Yoshua Bengio en 1998 **[10]** et les réseaux de neurones profonds qui en découlent en 2012 et ouvrent la voie à de nombreux champs d'application comme la vision, le traitement du langage ou la reconnaissance de la parole, ce développement c'est fait récemment ces nouvelles techniques de Machine Learning qui utilise des données massives (big data) ainsi que aux grande capacité de capacités de calcul notamment grâce aux processeurs graphiques. Preuve que chaque domaine irrigue les autres, c'est pour pouvoir utiliser les immenses promesses du Deep Learning que Google a mis au point les accélérateurs TPU(Tensor Processing Unit, un type de processeur dédié au calcul d'apprentissage des réseaux de neurones ; Time Processing Unit, une technique d'optimisation sur les processeurs Motorola. Thermoplastic polyurethane, en français polyuréthane thermoplastique).

## 2.3 Définition du Deep Learning

Le Deep Learning est un ensemble de méthodes d'apprentissage automatique (Machine Learning) tentant de modéliser avec un haut niveau d'abstraction des données grâce à des architectures articulées de différentes transformations non linéaires .Ces techniques ont permis des progrès importants et rapides dans les domaines de l'analyse du signal sonore ou visuel et notamment de la reconnaissance faciale, de la reconnaissance vocale, de la vision par

ordinateur, du traitement automatisé du langage. Le Deep Learning est une méthode de Machine Learning qui consiste à enseigner à des ordinateurs ce dont les humains sont naturellement capables .

Le Deep Learning apprend à un modèle informatique comment réaliser des tâches de classification directement à partir d'images, de textes ou d'audio. Les modèles de Deep Learning peuvent atteindre un niveau de précision exceptionnel,. L'entraînement des modèles s'effectue via un vaste ensemble de données labellisées et d'architectures de réseaux de neurones qui contiennent de nombreuses couches.

Le Deep Learning est une branche particulière du Machine Learning. Un processus de Machine Learning commence par l'extraction manuelle de caractéristiques pertinentes à partir d'images. En s'appuyant sur ces caractéristiques, un modèle qui catégorise les objets de l'image est ensuite créé. Dans un processus de Deep Learning, l'extraction de caractéristiques pertinentes à partir d'images est automatique. En outre, le Deep Learning effectue un apprentissage « de bout en bout » : à partir de données brutes, un réseau se voit assigner des tâches à accomplir (une classification, par exemple) et apprend comment les automatiser.

une autre différence majeure est le fait que les algorithmes de Deep Learning évoluent avec les données, tandis que le Shallow Learning (apprentissage peu profond) converge. Le Shallow Learning désigne les méthodes de Machine Learning dont la progression s'arrête à partir d'un certain niveau de performance après l'alimentation du réseau en exemples supplémentaires et en données d'apprentissage.

Un des avantages majeurs des réseaux de Deep Learning réside dans leur capacité à continuer à s'améliorer en même temps que le volume de vos données augmente.

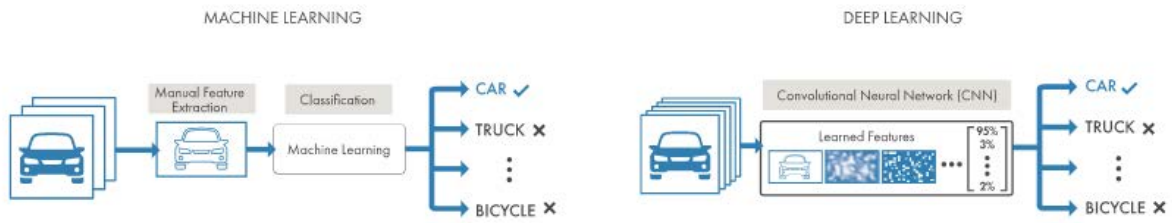



Figure 2.1 : Comparaison de méthodes de classification de véhicules de Machine Learning (gauche) et de Deep Learning (droite).

Pour classer des images avec le Machine Learning, les choix de caractéristiques et de classificateur doivent être effectués manuellement. Avec le Deep Learning, l'extraction de caractéristiques et le processus de modélisation sont automatiques.

### 2.4 Types du Deep Learning

Il existe 3 types de réseaux de Deep Learning qui sont :

- Les réseaux de neurones convolutionnels
- Les autoencodeurs
- La machine de Botlzmann

#### 2.4.1 Les réseaux de neurones convolutionnels

Les réseaux de neurones convolutionnels sont à ce jour les modèles les plus performants pour classer des images. Désignés par l'acronyme CNN, de l'anglais Convolutional Neural Network, ils comportent deux parties bien distinctes. En entrée, une image est fournie sous la forme d'une matrice de pixels. La couleur est représentée par une troisième dimension, de profondeur 3 pour représenter les couleurs fondamentales [Rouge, Vert, Bleu].
La première partie d'un CNN est la partie convolutive. Elle fonctionne comme un extracteur de caractéristiques des images. Une image est passée à travers une succession de filtres, ou noyaux de convolution, créant de nouvelles images appelées cartes de convolutions. Certains filtres intermédiaires réduisent la résolution de l'image par une opération de maximum local. Au final, les cartes de convolutions sont mises concaténées en un vecteur de caractéristiques, appelé code CNN.

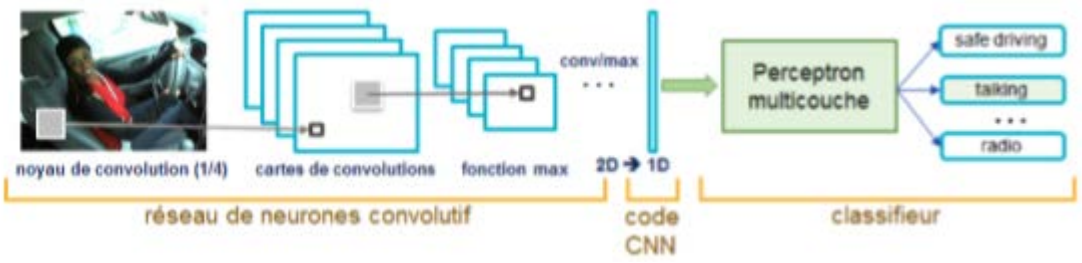



Figure 2.2 : architecture standards d'un réseau de neurone convolutionnel

Ce code CNN en sortie de la partie convolutive est ensuite branché en entrée d'une deuxième partie, constituée de couches entièrement connectées (perceptron multicouche). Le rôle de cette partie est de combiner les caractéristiques du code CNN pour classer l'image.
La sortie est une dernière couche comportant un neurone par catégorie. Les valeurs numériques obtenues sont généralement normalisées entre 0 et 1, de somme 1, pour produire une distribution de probabilité sur les catégories.

Les réseaux de neurones convolutionnels sont basés sur le perceptron multicouche(MLP), et inspirés du comportement du cortex visuel des vertébrés. Bien qu'efficaces pour le traitement d'images, les MLP ont beaucoup de mal à gérer des images de grande taille, ce qui est dû à la croissance exponentielle du nombre de connexions avec la taille de l'image.

Par exemple, si on prend une image de taille 32x32x3 (32 de large, 32 de haut, 3 canaux de couleur), un seul neurone entièrement connecté dans la première couche cachée du MLP aurait 3072 entrées (32*32*3). Une image 200x200 conduirait ainsi à traiter 120 000 entrées par neurone ce qui, multiplié par le nombre de neurones, devient énorme.

Les CNN visent à limiter le nombre d'entrées tout en conservant la forte corrélation « spatialement locale » des images naturelles. Par opposition aux MLP, les CNN ont les traits distinctifs suivants :

1. 'Volumes 3D de neurones'. La couche de neurones n'est plus simplement une surface (perceptron), mais devient un volume avec une profondeur. Si on considère un seul champ récepteur du CNN, les n neurones associés (sur la profondeur) forment l'équivalent de la première couche d'un MLP.

2. 'Connectivité locale'. Grâce au champ récepteur qui limite le nombre d'entrées du neurone, tout en conservant l'architecture MLP, les CNN assurent ainsi que les filtres produisent la réponse la plus forte à un motif d'entrée spatialement localisé, ce qui conduit à une représentation parcimonieuse de l'entrée. Une telle représentation occupe moins d'espace en mémoire. De plus, le nombre de paramètres à estimer étant réduit, leur estimation (statistique) est plus robuste pour un volume de données fixé (comparé à un MLP).

3. 'Poids partagés' : Dans les CNN, les paramètres de filtrage d'un neurone (pour un champ récepteur donné) sont identiques pour tous les autres neurones d'un même noyau (traitant tous les autres champs récepteurs de l'image). Ce paramétrage (vecteur de poids et biais) est défini dans une « carte de fonction ». Cela signifie que tous les neurones dans une couche de convolution donnée détectent exactement la même caractéristique. En multipliant les champs récepteurs, il devient possible de détecter des éléments indépendamment de leur position dans le champ visuel, ce qui induit une propriété d'invariance par translation.

Ensemble, ces propriétés permettent aux réseaux de neurones convolutionnels d'obtenir une meilleure généralisation (en termes d'apprentissage) sur des problèmes de vision. Le partage de poids permet aussi de réduire considérablement le nombre de paramètres libres à apprendre, et ainsi les besoins en mémoire pour le fonctionnement du réseau. La diminution de l'empreinte mémoire permet l'apprentissage de réseaux plus grands donc souvent plus puissants.

.

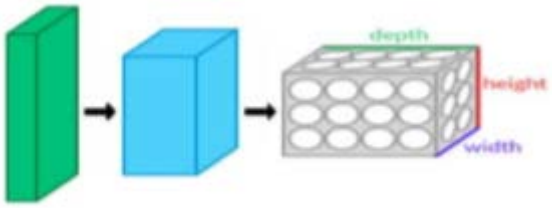



Figure 2.3 : Une couche du CNN en 3 dimensions. (Vert = volume d'entrée, bleu = volume du champ récepteur,gris = couche de CNN, cercles = neurones artificiels indépendants)

Une architecture CNN est formée par un empilement de couches de traitement indépendantes :

- La couche de convolution (CONV) qui traite les données d'un champ récepteur.
- La couche de pooling (POOL), qui permet de compresser l'information en réduisant la taille de l'image intermédiaire (souvent par sous-échantillonnage).
- La couche de correction (ReLU), souvent appelée par abus 'ReLU' en référence à la fonction d'activation (Unité de rectification linéaire).
- La couche "entièrement connectée" (FC), qui est une couche de type perceptron.
- La couche de perte (LOSS).

#### 2.4.1.1. Couche de convolution(CONV)

La couche de convolution est le bloc de construction de base d'un CNN. Trois paramètres permettent de dimensionner le volume de la couche de convolution la profondeur, le pas et la marge.

La profondeur de la couche est le nombre de noyaux de convolution (ou nombre de neurones associés à un même champ récepteur).

Le pas contrôle le chevauchement des champs récepteurs. Plus le pas est petit, plus les champs récepteurs se chevauchent et plus le volume de sortie sera grand.

La marge à 0 ou zero padding parfois, il est commode de mettre des zéros à la frontière du volume d'entrée. La taille de ce 'zero-padding' est le troisième hyper paramètre. Cette marge permet de contrôler la dimension spatiale du volume de sortie. En particulier, il est parfois souhaitable de conserver la même surface que celle du volume d'entrée.

Si le pas et la marge appliquée à l'image d'entrée permettent de contrôler le nombre de champs récepteurs à gérer (surface de traitement), la profondeur permet d'avoir une notion de volume de sortie, et de la même manière qu'une image peut avoir un volume, si on prend une profondeur de 3 pour les trois canaux RGB d'une image couleur, la couche de convolution va également présenter en sortie une profondeur. C'est pour cela que l'on parle plutôt de "volume de sortie" et de "volume d'entrée", car l'entrée d'une couche de convolution peut être soit une image soit la sortie d'une autre couche de convolution.

La taille spatiale du volume de sortie peut être calculée en fonction de la taille du volume d'entrée Wi la surface de traitement K (nombre de champs récepteurs), le pas S avec lequel ils sont appliqués, et la taille de la marge P La formule pour calculer le nombre de neurones du volume de sortie est :

$$W0 = \frac{W_i - K + 2P}{S} + 1 \qquad (2.1)$$

Si W0 n'est pas entier, les neurones périphériques n'auront pas autant d'entrée que les autres. Il faudra donc augmenter la taille de la marge (pour recréer des entrées virtuelles).

Souvent, on considère un pas S = 1, on calcule donc la marge de la manière suivante :

$$P = \frac{K-1}{2} \qquad (2.2)$$

si on souhaite un volume de sortie de même taille que le volume d'entrée. Dans ce cas particulier la couche est dite "connectée localement".

#### 2.4.1.2 Couche de pooling (POOL)

Un autre concept important des CNNs est le pooling, ce qui est une forme de sous-échantillonnage de l'image. L'image d'entrée est découpée en une série de rectangles de n pixels de côté ne se chevauchant pas (pooling). Chaque rectangle peut être vu comme une tuile. Le signal en sortie de tuile est défini en fonction des valeurs prises par les différents pixels de la tuile.

Le pooling réduit la taille spatiale d'une image intermédiaire, réduisant ainsi la quantité de paramètres et de calcul dans le réseau. Il est donc fréquent d'insérer périodiquement une couche de pooling entre deux couches convolutives successives d'une architecture CNN pour contrôler l'overfitting (sur-apprentissage). L'opération de pooling créait aussi une forme d'invariance par translation.

La couche de pooling fonctionne indépendamment sur chaque tranche de profondeur de l'entrée et la redimensionne uniquement au niveau de la surface. La forme la plus courante est une couche de mise en commun avec des tuiles de taille 2x2 (largeur/hauteur) et comme valeur de sortie la valeur maximale en entrée. On parle dans ce cas de « Max-Pool 2x2 ».

Il est possible d'utiliser d'autres fonctions de pooling que le maximum. On peut utiliser un « average pooling » (la sortie est la moyenne des valeurs du patch d'entrée), du « L2-norm pooling ». Dans les faits, même si initialement l'average pooling était souvent utilisé il s'est avéré que le max-pooling était plus efficace car celui-ci augmente plus significativement l'importance des activations fortes. En d'autres circonstances, on pourra utiliser un pooling stochastique.

Le pooling permet de gros gains en puissance de calcul. Cependant, en raison de la réduction

agressive de la taille de la représentation (et donc de la perte d'information associée),

la tendance actuelle est d'utiliser de petits filtres (type 2x2). Il est aussi possible d'éviter la couche de pooling mais cela implique un risque sur-apprentissage plus important.

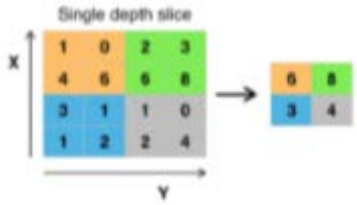



Figure 2.4 **:** Pooling avec un filtre 2x2 et un pas de 2

### 2.4.1.3 Couches de correction (RELU)

Il est possible d'améliorer l'efficacité du traitement en intercalant entre les couches de traitement une couche qui va opérer une fonction mathématique (fonction d'activation) sur les signaux de sortie.

La fonction RELU (abréviation de Unités Rectifié linéaires) :

$$\mathbf{F(x)=max(0,x)} \qquad (2.3)$$

Cette fonction force les neurones à retourner des valeurs positives.

### 2.4.1.4 Couche entièrement connectée (FC)

Après plusieurs couches de convolution et de max-pooling, le raisonnement de haut niveau dans le réseau neuronal se fait via des couches entièrement connectées. Les neurones dans une couche entièrement connectée ont des connexions vers toutes les sorties de la couche précédente. Leurs fonctions d'activations peuvent donc être calculées avec une multiplication matricielle suivie d'un décalage de polarisation.

### 2.4.1.5 Couche de perte (LOSS)

La couche de perte spécifie comment l'entrainement du réseau pénalise l'écart entre le signal prévu et réel. Elle est normalement la dernière couche dans le réseau. Diverses fonctions de perte adaptées à différentes tâches peuvent y être utilisées. La fonction Softmax permet de calculer la distribution de probabilités sur les classes de sortie.

### 2.4.2 Autoencodeur

Un autoencodeur est un réseau de neurones qui est formé pour répliquer son entrée à sa sortie. Auto-encodeur peut servir d'outils pour apprendre les réseaux neuronaux profond, , ce qui a pour objectif de compresser tout en gardant l'erreur $\|x - \hat{x}\|^2$ faible ou d'apprendre une représentation des données compacte mais riche en information.

Un autoencodeur de formation n'est pas surveillée en ce sens qu'aucune données étiquetées ne sont nécessaire. Le processus de formation repose toujours sur l'optimisation d'une

fonction objectif (aussi appelée fonction de coût ou d'erreur)

**La fonction de coût** mesure l'erreur entre l'entrée ***x*** et sa reconstruction à la sortie $\hat{x}$ .

Un autoencodeur est composé d'**un encodeur** et **un décodeur**. Le codeur et le décodeur peuvent avoir plusieurs couches, mais par souci de simplicité, on considère que chacun d'eux dispose d'une seule couche.

Si l'entrée d'un autoencodeur est un vecteur x $\in$ $\mathbb{R}^{D_x}$ l'encodeur mappe le vecteur x à un autre vecteur $z \in \mathbb{R}^{D^{(1)}}$ comme suit :

$$z = h^{(1)}\left(w^{(1)}x + b^{(1)}\right) \qquad (2.4)$$

où l'exposant (1) indique la première couche. $h^{(1)}$: $\mathbb{R}^{D^{(1)}} \longrightarrow \mathbb{R}^{D^{(1)}}$ est une fonction de transfert de l'encodeur, $w^{(1)} \in \mathbb{R}^{D^{(1)\times D_x}}$ est une matrice de poids, $b^{(1)} \in \mathbb{R}^{D^{(1)}}$ est un vecteur de polarisation ou de biais

Ensuite, le décodeur mappe la représentation codée $z$ dans une estimation du vecteur d'entrée originale, x, comme suit :

$$\hat{x} = h^{(2)}\left(w^{2}z + b^{(2)}\right) \qquad (2.5)$$

où l'exposant (2) représente la deuxième couche. $h^{(2)}$: $\mathbb{R}^{D_x} \longrightarrow \mathbb{R}^{D_x}$ est la fonction de transfert pour le décodeur, $w^{(1)} \in \mathbb{R}^{D_{x\times D^{(1)}}}$ est une matrice de poids , et $b^{(2)} \in \mathbb{R}^{D_x}$ est un vecteur de polarisation ou de biais.

Le poids du décodeur est liés au poids de l'encodeur (habituellement transposé ) .

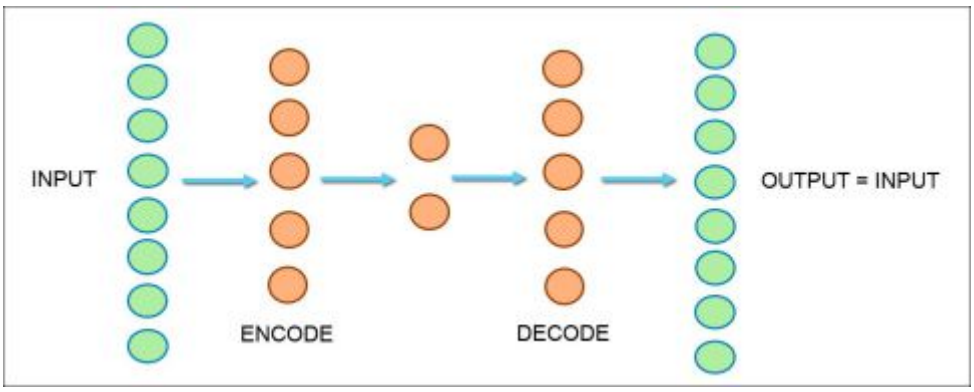



Figure 2.5: autoencodeur typique

| Option de fonction de transfert | Définition |
|---|---|
| 'logsig' | Fonction sigmoide logistique $$f(z) = \frac{1}{1+e^{-z}}$$ |
| 'satlin' | Fonction de transfert saturant linéaire $$f(z) = \begin{cases} 0, & if \quad z \leq 0 \\ z, & if \quad 0 \leq z \leq 1 \\ 1, & if \quad z \geq 1 \end{cases}$$ |

Table 2.1 : Fonction de transfert pour l'encodeur

| Option de fonction de transfert | Définition |
|---|---|
| 'logsig' | Fonction sigmoide logistique $$f(z) = \frac{1}{1+e^{-z}}$$ |
| 'satlin' | Fonction de transfert saturant linéaire $$f(z) = \begin{cases} 0, & if \quad z \leq 0 \\ z, & if \quad 0 \leq z \leq 1 \\ 1, & if \quad z \geq 1 \end{cases}$$ |
| 'purelin' | Fonction de transfert linéaire $f(z) = z$ |

Table 2.2 : Fonction de transfert pour le décodeur

#### 2.4.2.1 Entrainement d'un autoencodeur

Un Autoencodeur est entrainé de façon non supervisée, pour apprendre la représentation.

### 2.4.3 Machines de Boltzmann

Les machines de Boltzmann restreintes (RBM pour Restricted Boltzmann Machines) sont des réseaux de neurones stochastiques génératifs qui ont été introduits par P. Smolensky **[11],** puis popularisés plus récemment par les travaux de G. E. Hinton qui a proposé un algorithme d'apprentissage rapide **[12],** ainsi qu'une version multi-couches appelée DBN (pour Deep Belief Network) **[13,14].**

Un modèle RBM standard se compose d'un ensemble de neurones connectés entre eux (que nous appellerons unités). Chaque unité fournit une décision en tenant compte de l'apport des autres unités. La Figure 2.6 – (a) illustre l'architecture standard des RBM : Une couche d'unités dites visibles (illustrées par des ronds blancs) connectée à une couche d'unités cachées (illustrées par des ronds gris). Les unités cachées fournissent des décisions binaires, alors que les unités visibles peuvent avoir des valeurs réelles. Les unités sont reliées par des connexions pondérées, dont les paramètres sont calculés durant l'apprentissage. A noter qu'il n'y a pas de connexions reliant les unités cachées (ou visibles) entre-elles. Cette restriction est d'ailleurs à l'origine du nom des RBM, par rapport aux machines de Boltzmann standards (c'est à dire non restreintes), pour lesquelles toutes les connexions sont autorisées.
Si nous notons par $v_i$ l'état d'activation de l'unité visible i, et par $c_j$ celui de l'unité cachée j, les RBMs assignent une probabilité pour chaque configuration jointe des unités visibles v et des unités cachées c :

$$p(v,c) = \frac{\exp[-E(v,c)]}{z} \qquad (2.6)$$

où E (v, c) est une fonction d'énergie , et Z est une constante de normalisation appelée fonction de partition.

un RBM peut être considéré comme un cas particulier d'un MRF( un champ aléatoire de Markov est un ensemble de variables aléatoires vérifiant une propriété de Markov relativement à un graphe non orienté, c'est un modèle graphique.) ayant une structure graphique et une fonction d'énergie spécifiques. Le terme d'énergie E (v,c) présent dans l'équation 2.6 est défini par :

$$E(v,c) = -\sum_{i,j} w_{ij}\, v_i c_j - \sum_i a_i v_i - \sum_j b_j c_j \qquad (2.7)$$

où $w_{ij}$ est le poids qui pondère la connexion entre i et j, $a_i$ est le biais de l'unité i et $b_j$ est celui de l'unité j.

Ce modèle est souvent entraîné par l'algorithme de divergence contrastive de Hinton et al. **[12],** qui se base sur un schéma d'auto-encodage, et qui vise à minimiser la fonction d'énergie exprimée par l'équation 2.7. Concrètement, et sans rentrer dans les détails, la mise à jour d'un poids $w_{ij}$ donné est exprimée, pour chaque exemple d'apprentissage, par :

$$\Delta w_{ij} = \epsilon * \left[ \langle v_i c_j \rangle_{donn\acute{e}es} - \langle v_i c_j \rangle_{reconstruction} \right] \quad (2.8)$$

Où $\epsilon$ est le taux d'apprentissage (learning rate en agnlais), et :

- Le terme $\langle v_i c_j \rangle_{donn\acute{e}es}$ désigne la fréquence avec laquelle l'unité visible i et l'unité cachée j sont activées mutuellement, quand le réseau est stimulé (au niveau de la couche visible) avec les données d'apprentissage. Les états d'activation des unités cachées sont dans ce cas obtenus par :

$$P = (c_j = 1|v) = \sigma\left(a_j + \sum_i w_{ij} v_i\right) \quad (2.9)$$

où $\sigma$ est une fonction sigmoïde.

- Le terme $\langle v_i c_j \rangle_{reconstruction}$ désigne la fréquence avec laquelle l'unité visible i et l'unité cachée j sont activées mutuellement, quand le réseau est stimulé (au niveau des couches cachées) avec des données reconstruites. Celles-ci étant obtenues en calculant la probabilité conditionnelle :

$$P = (v_i = 1|v) = \sigma\left(b_i + \sum_j w_{ij} c_j\right) \quad (2.10)$$

A noter que l'équation 2.10 est valable pour le cas binaire, et qu'une équation similaire, faisant intervenir une fonction Gaussienne, existe pour le cas des données réelles

Cette procédure est ainsi répétée pour tous les exemples d'apprentissage, jusqu'à la convergence, c'est à dire quand l'erreur de reconstruction est inférieure à un certain seuil.

un modèle RBM temporel (appelé tRBM, pour Temporal RBM), qui consiste simplement en un RBM classique dans lequel l'information temporelle relative aux instants passés est incorporée en tant que biais supplémentaire Figure 2.6-b).

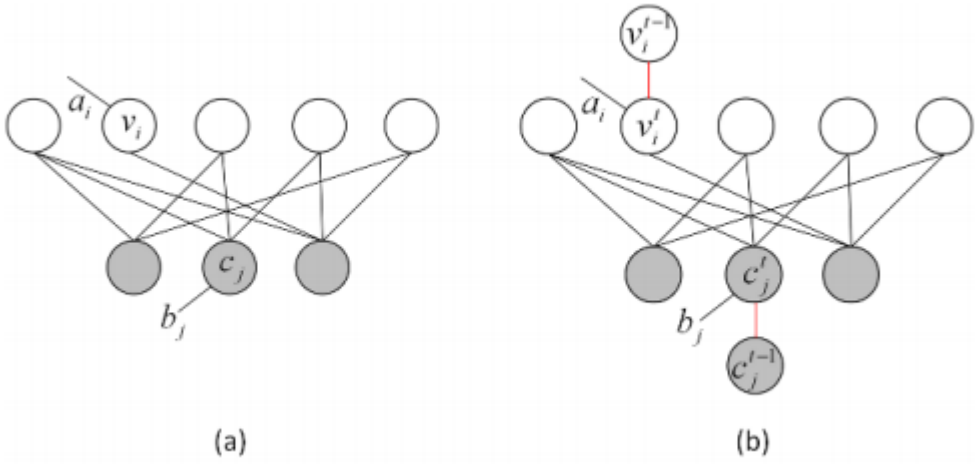



Figure 2.6 : – (a) Machine de Boltzmann restreinte **[15]-** (b) - Machine de Boltzmann restreinte temporelle **[16].**

**2.5 Utilisation du Deep Learning en traitement d'images**

Le Deep Learning et le traitement de l'image sont deux domaines d'un grand intérêt pour les universitaires et les professionnels de l'industrie. Les domaines d'application de ces deux disciplines varient largement, englobant des domaines tels que la médecine, la robotique, la sécurité et la surveillance.

Aucun apprentissage profond (Deep Learning) ne nuit le traitement d'image. Il faut d'énormes ensembles de données et de nombreuses ressources de calcul pour faire le Deep Learning. Il existe de nombreuses applications où il est souhaitable de pouvoir traiter des images avec moins de charge de calcul et des empreintes de mémoire plus petites et sans avoir accès à des bases de données volumineuses. Quelques exemples sont les téléphones mobiles, les tablettes, les caméras mobiles, les automobiles, le Deep Learning est en réussite en ce moment car il existe des résultats très impressionnants à la classification.

La classification est un problème parmi beaucoup d'autres traités par le traitement d'images, même s'il était vrai que le Deep Learning résoudrait tous les problèmes de classification, il y aurait beaucoup d'autres types de traitement d'image à faire :

- Réduction du bruit,
- enregistrement d'image
- calcul de mouvement
- morphing (procédé numérique consistant à transformer progressivement une image en une autre)
- les modes de fusion
- affinement(rendre une image plus nette)
- corrections et transformations optiques
- calcul de géométries
- estimation 3D
- modèles de mouvement 3D temporel
- vision stéréoscopique,

- compression et codage de données
- segmentation d'images
- suppression du flous de l'images)
- stabilisation de mouvement,
- infographie.

**2.6 Conclusion**

Dans ce chapitre, nous avons présenté le Deep Learning qui utilise les réseaux de neurones convolutionnels les plus répandus. Ces réseaux sont capables d'extraire des caractéristiques d'images présentées en entrée et de classifier ces caractéristiques,

le Deep Learning est l'une des méthode les plus remarquable pour le développement du traitement d'images tel que la segmentation d'images.
Dans le prochain chapitre, nous exploiterons le Deep Learning pour réaliser une segmentation sémantique.

### 3.1 Introduction

La segmentation sémantique est une segmentation qui permet d'associer une catégorie à chaque pixel d'une image Par exemple, un véhicule autonome doit identifier des véhicules, des piétons, des panneaux de signalisation, des trottoirs et autres éléments de l'environnement routier.

La segmentation sémantique nécessite d'avoir une base de données pour réaliser l'apprentissage.

Cette base de données comprend deux catégories de données :

La première catégorie comprend un ensemble d'images contenant les objets qu'on souhaite reconnaitre

La deuxième catégorie comprend les mêmes images de l'ensemble précédent mais celles-ci sont étiquetées.

Il est généralement assez difficile de réaliser sa propre base de données pour plusieurs raisons qui sont principalement :

- Il faut recueillir beaucoup d'images de mêmes catégories présentant les mêmes objets.
- Il faut étiqueter toutes ces images de manière manuelle.

Il existe plusieurs base de données qui peuvent être téléchargées et dans plusieurs domaines d'application.

Dans ce travail, nous utilisons la base de données Cambridge-driving (CamVid) contenant des images de l'environnement de la ville.

Avant de développer les différentes étapes pour réaliser la segmentation sémantique en utilisant cette base de données, nous allons d'abord détailler cette base.

Nous allons par la suite développer l'utilisation de l'exemple fourni par Matlab 2018 pour la segmentation sémantique.

### 3.2 La base de données utilisée

CamVid est un sous-ensemble de 701 images et de 701 images étiquetées de taille 960*720. La base de données Cambridge-driving (CamVid) est une collection d'images contenant des vues au niveau de la rue prises durant la conduite y compris les voitures, les piétons, les routes, animal, enfant, chariot roulant, cycliste, moto, camion, bus, train, ciel, tunnel, bâtiment, mur, arbre, végétation, trottoir, bloc de stationnement, poteau, cône de signalisation, pont, feu de circulation.

Chaque pixel de ces images est étiqueté par une des 32 classes sémantiques contenues dans les images. Ces 32 classes sont formées d'objets mobiles et d'objets immobiles.

Parmi les objets mobiles on trouve les animaux, les piétons, les enfants, les cyclistes, les motocycles et scooter, les voitures, les camions, les bus, les trains, les bagages et autres.

Parmi les objets immobiles, on trouve le ciel, les tunnels, bâtiments, les clôtures, les arbres, autres végétations, trottoirs, bloc de stationnement, poteaux et colonnes, feux de signalisation, panneaux de circulation, autres symboles, textes divers, passerelles, route, plafond des passages souterrain, ruelles, arrêts d'urgence, route piétonnes et autres .

Sur la figure ci-dessous, nous affichons 3 exemples d'images et d'images étiquetés prises de la base de données CamVid.

Pour télécharger la base de donnée CamVid il faut utiliser les instruction suivantes :

imageURL = ' http://web4.cs.ucl.ac.uk/staff/g.brostow/MotionSegRecData/files/701_StillsRaw_full.zip'

labelURL = ' http://web4.cs.ucl.ac.uk/staff/g.brostow/MotionSegRecData/data/LabeledApproved_full.zip'

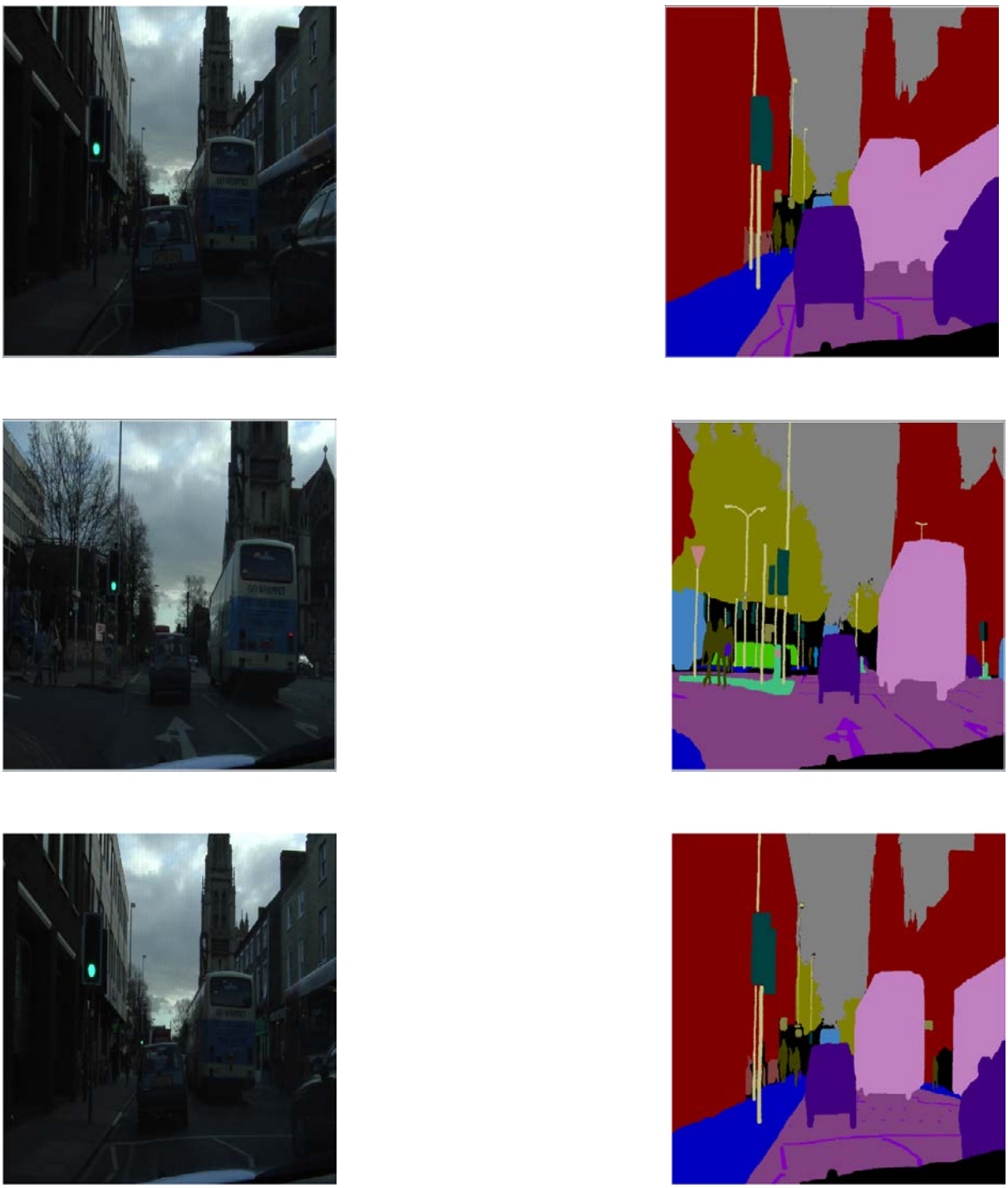


Figure 3.1 : Images de la base de données Camvid avec leurs étiquettes.

### 3.3 Apprentissage

Dans ce travail, un réseau de segmentation sémantique de 11 classes sur les 32 classes original de CamVid est formé.

Ces 11 classes sont : ciel, bâtiment, clôture, voiture, piéton, cycliste, arbre, symbole, route, trottoir et poteau.

L’architecture du réseau profond utilisé dans l’exemple fourni dans Matlab est de type convolutionnaire. C’est un réseau neuronal convolutionnaire (CNN) et il est noté SegNet. Il est formé par un empilement de couches : la couche de convolution, la couche de pooling, la couche de correction, la couche "entièrement connectée" et la couche de perte. Une implémentation inverse d'un CNN est ajoutée. Le processus de suréchantillonnage est effectué le même nombre de fois que le processus de sous-échantillonnage pour s'assurer que l'image finale possède les mêmes dimensions que l'image d'entrée. Pour établir une classification au niveau des pixels plutôt qu'au niveau de l'image entière, une couche de génération des classes des pixels est utilisée. Elle affecte chaque pixel à une certaine classe. Cela forme une architecture codeur/décodeur de 91 couches, qui permet la segmentation sémantique.

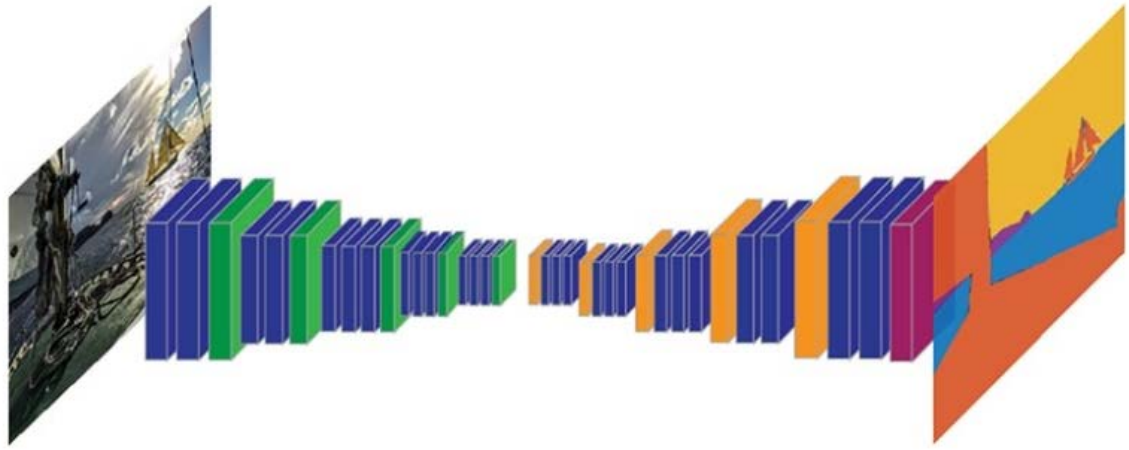


Figure 3.2: Réseau CNN exécutant des fonctions orientées image à chaque couche, puis sous-échantillonnant l'image à l'aide d'une couche de regroupement (en vert). Ce processus est répété plusieurs fois pour la première moitié du réseau. La génération de la première moitié de ce diagramme est suivie d'un nombre égal de couches de type « nonpooling » (en orange).

Ce réseau peut prendre environ 6 heures pour la formation et il peut encore durer plus longtemps selon le matériel utilisé. Pour exécuter l’exemple complet sans avoir à attendre pour la formation complète du réseau, il suffit d’introduire et exécuter ces instruction ci-dessous dans Matlab  pour le téléchargement du réseau pré-appris.

```
Pretrain= 'https://www.mathworks.com/supportfiles/vision/data/segnetVGG16CamVid.mat';
pretrainedFolder = fullfile(tempdir,'pretrainedSegNet');
pretrainedSegNet = fullfile(pretrainedFolder,'segnetVGG16CamVid.mat');
if ~exist(pretrainedFolder,'dir')
   mkdir(pretrainedFolder);
   disp('Downloading pretrained SegNet (107 MB)...');
   websave(pretrainedSegNet,pretrain);
end
```

**3.4 Test**

Pour le test, nous avons choisi 2 images qui font parties de la base de donnée CamVid, 2 images du même genre mais qui ne font pas parties de la base de données CamVid et 2 images différentes de la base de données Camvid. Le réseau formé de la segmentation sémantique de 11 classe prédéfinie peut segmenter une image de 1 à 11 classes à condition que ces classes existent dans les 11 classes prédéfinies du réseau formé.

Nous avons choisi de tester ce réseau formé de segmentation sémantique en segmentant ces images choisis en trois classe (ciel, voiture, route) ensuite en six classe (ciel, voiture, route, piétons, bâtiments, arbres)

Pour réaliser l'exécution du programme il faut d'abord charger le réseau formé comme indiqué précédemment et ensuite choisir et redimensionner l'image couleur à la taille 360*480. C'est cette taille qui est utilisée dans la couche d'entrée du réseau. Il faut ensuite définir les classes à détecter

Ces classes peuvent être définies par les instructions :

- Dans le cas de 3 classes :

```
classes = [
    "ciel"
    "voiture"
    "route"
        ];
```

- Dans le cas de 6 classes :

```
classes = [
      "ciel"
   "voiture"
   "route"
   "piéton"
   "bâtiment"
   "arbre"
       ];
```

Par la suite, la segmentation est réalisée en utilisant la commande :

C = semanticseg(I,net)

Avec :

I est l'image d'entrée

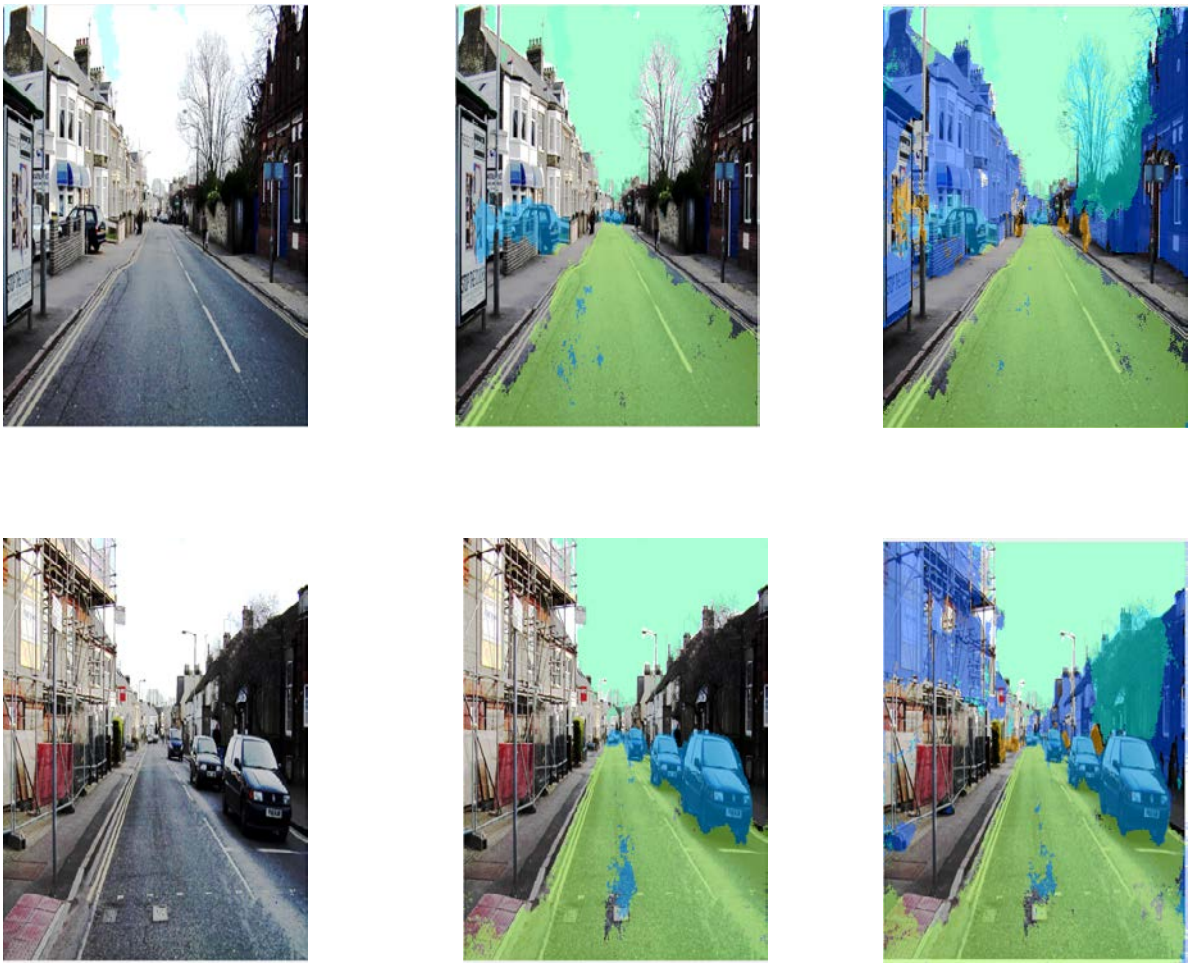
net est une variable qui représente le chargement du réseau formé.

Puis pour pouvoir fusionner l'image d'entrée I avec des couleurs différentes pour les étiquettes, il faut utiliser la commande :

B = labeloverlay(I,C)

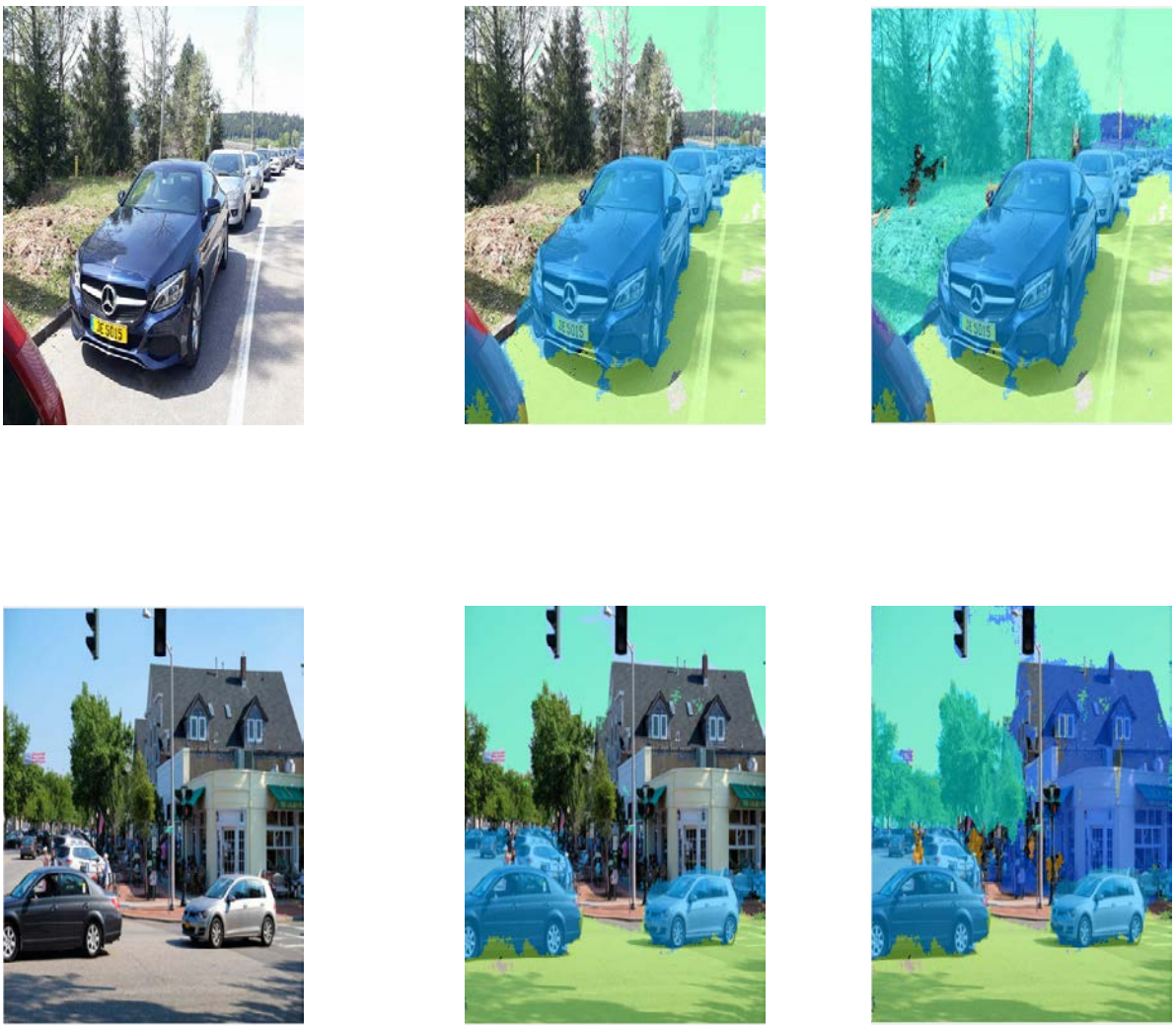
avec C est une matrice d'étiquettes.

Les résultats obtenus sur les 6 images de Tests sont fournis sur les figures ci-dessous:



a’) b’) c’)

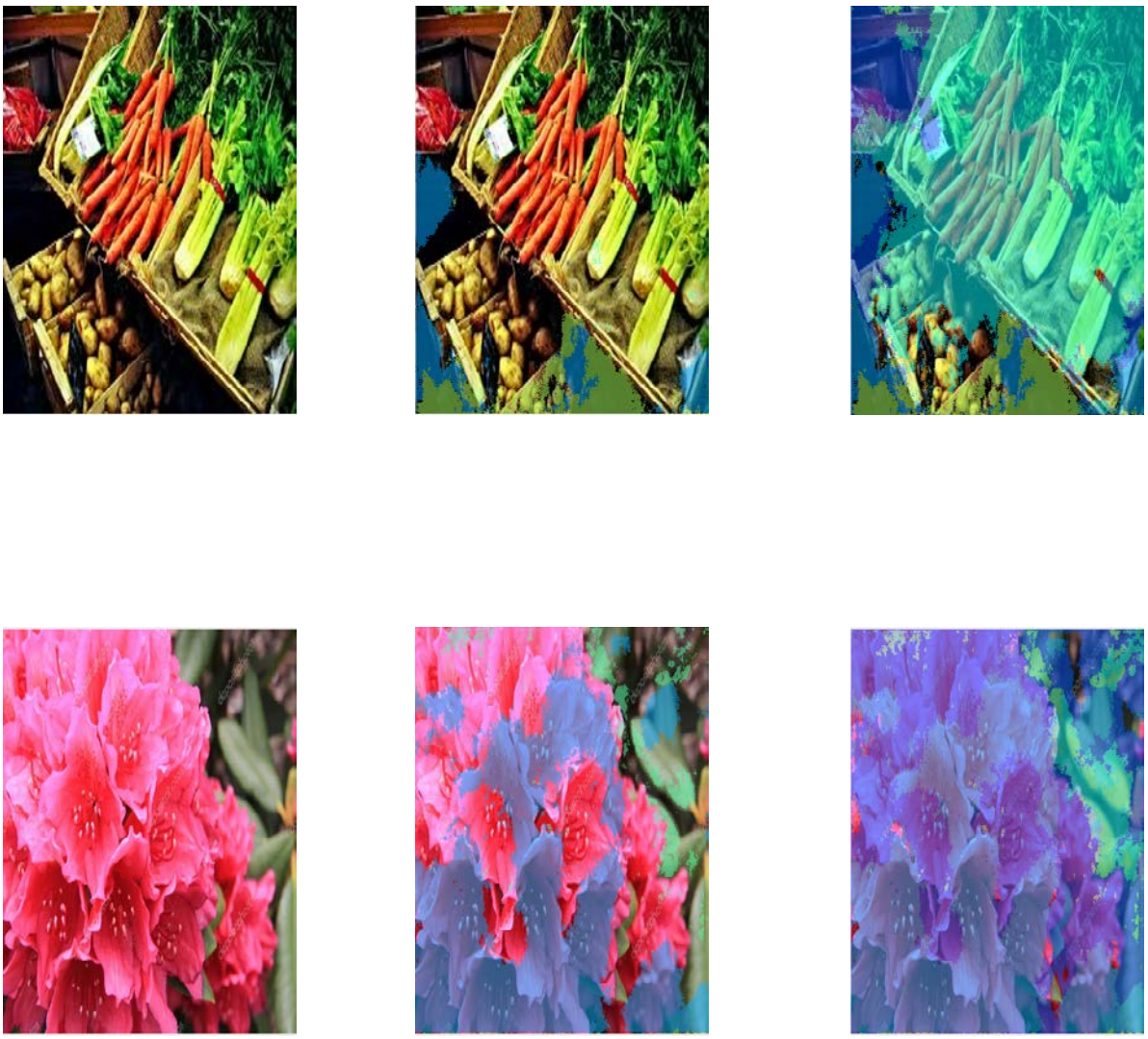
Figure 3.3: Résultat des images testées qui sont de la base de données CamVid
a’) images original
b’) segmentation sémantique en 3 classes
(Ciel en couleur cyan, Route en vert, Voiture en bleu)
C’) segmentation sémantique en 6 classes (Ciel en couleur cyan, Route en vert, Voiture en bleu, Piéton en jaune, Arbres en vert turquoise, Bâtiment en violet)



a’) b’) c’)

Figure 3.4 : résultat des images testées qui sont du même genre avec la base de données CamVid

a’) images original
b’) segmentation sémantique en 3 classes
(Ciel en couleur cyan, Route en vert, Voiture en bleu)
C’) segmentation sémantique en 6 classes (Ciel en couleur cyan, Route en vert, Voiture en bleu, Piéton en jaune, Arbres en vert turquoise, Bâtiment en violet)



a’) b’) c’)

Figure 3.5 : résultat des images testées qui sont totalement différentes de la base de données CamVid.

a’) images original
b’) segmentation sémantique en 3 classes
(Ciel en couleur cyan, Route en vert, Voiture en bleu)
C’) segmentation sémantique en 6 classes (Ciel en couleur cyan, Route en vert, Voiture en bleu, Piéton en jaune, Arbres en vert turquoise, Bâtiment en violet)

### 3.5 Interprétation

Pour les 2 images tirées de la même base de données Camvid (figure 3.3), nous constatons que dans le cas de la segmentation en 3 classes (figure 3.3.b) les 3 objets en question sont assez bien détectés. Quelques anomalies existent quand même comme de petites parties de la route qui ont été classées comme voiture. Les objets ne figurant pas dans les trois classes ne sont globalement pas classés comme les bâtiments et les arbres.

Ces derniers, lorsqu'ils sont inclus comme dans le cas de la segmentation en 6 classes, ils sont assez bien détectés. Nous pouvons constater cela sur la figure 3.3.c.

En ce qui concerne les deux images qui sont du même genre que celles de la base CamVid qui sont affichées en figure 3.4, nous pouvons tirer pratiquement les mêmes conclusions que précédemment.

Pour ce qui est des deux images totalement différentes de la base CamVid (figure3.5), nous constatons que les résultats de la segmentation sont mauvais.

On aurait pu penser que lorsqu'aucun des objets fournis pour la segmentation n'est présent, aucune classe ne sera détectée, or ce n'est pas le cas.

On remarque par exemple dans le cas de la deuxième image de la figure 3.5.b une partie des fleurs est classée comme voiture et dans la première image de la figure 3.5.c presque tous les légumes sont classés comme arbre.

### 3.6 Conclusion

Sachant que le temps de la formation du réseau et de l'étiquetage de sa propre base de données sont des inconvénients majeurs pour réaliser la segmentation sémantique nous avons préféré exploiter l'exemple de Matlab. Dans ce chapitre nous avons détaillé les différentes étapes pour réaliser la segmentation sémantique avec le Deep Learning en utilisant cet exemple. Nous avons introduit les réseaux de neurones convolutionnels utilisés en présentant les différents types de couches utilisées pour réaliser la classification.

**Conclusion générale**

Dans ce projet nous avons discuté des notions fondamentales de segmentation d'image et du Deep Learning basé sur les réseaux de neurones convolutionnels en particulier. Nous avons introduit ces réseaux de neurones convolutionnels en présentant les différents types de couches utilisées dans la classification: la couche convolutionnelle, la couche de rectification, la couche de pooling et la couche fully connected.

Nous avons exploité et fait fonctionner l'exemple de segmentation sémantique fournie par Matlab 2018.

Dans la phase d'implémentation, l'utilisation d'un CPU fait que le temps d'exécution soit trop couteu. Afin de régler ce problème il est nécessaire d'utiliser des réseaux de neurones convolutionnels profonds déployés sur un GPU au lieu d'un CPU.

Pour réaliser sa propre segmentation sémantique, il est nécessaire d'utiliser sa propre base étiquetée, Ceci est un travail fastidieux mais qui est nécessaire dans le cas d'un domaine d'utilisation bien défini au préalable.

# Bibliographie

**[1]** C.Houassine, segmentation d'images par une approche biomimétique hybride. université universite m'hamed bougara- boumerdes. 2012.

**[2]** J. Canny, A, computational approach to edge detection, IEEE trans. on pattern analysis and machine intelligence, vol. 8, n°6, pp. 679-698, novembre 1986.

**[3]** Sarah GHANDOUR. Segmentation d'images couleurs par morphologie mathématique : application aux images microscopiques. PhD thesis, Université de Toulouse III - Paul Sabatier, 2010.

**[4]** Mr Cyril Meurie. Segmentation d'images couleur par classication pixellaire ethiérarchie

de partitions. PhD thesis, Université de CAEN/BASSENORMANDIE, 2005

**[5]** D. Michie, D. Spiegelhalter, and C. Taylor. Machine learning, neural and statistical classication, 1994. vol. 6, pages 84106.

**[6]** R. Duda, P. Hart, and D. Stork. Pattern classication, 2001. Xiley Interscience 2e édition.

**[7]** J. Cocquerez and S. Philipp. Analyse d'images : ltrage et segmentation, 1995. Paris Masson

**[8]** J. Hérault and C. Jutten. Réseaux neuronaux et traitement du signal, 1994

**[9]** Warren McCulloch & Walter Pitts, *A Logical Calculus of Ideas Immanent in Nervous Activity*, 1943, Bulletin of Mathematical Biophysics 5:115-133.

**[10]** LeCun, Yann, L. Bottou, Yoshua Bengio et P. Haffner (nov. 1998). « Gradient based learning applied to document recognition ». Anglais. In : Proceedings of the IEEE

**[11]** P.K. Simpson : Fuzzy min-max neural networks. In IEEE International Joint Conference on Neural Networks, pages 1658–1669, 1991

**[12]** G.E. Hinton : Training products of experts by minimizing contrastive divergence. Neural computation, 14(8):1771–1800, 2002

**[13]** G.E. Hinton, S. Osindero et Y.W. Teh : A fast learning algorithm for deep belief nets. Neural computation, 18(7):1527–1554, 2006.

**[14]** G.E. Hinton et R.R. Salakhutdinov : Reducing the dimensionality of data with neural networks. Science, 313(5786):504–507, 2006

**[15]** P. Smolensky : Information processing in dynamical systems : foundations of harmony theory. In Parallel distributed processing : explorations in the microstructure of cognition, volume 1, pages 194–281. MIT Press, 1986.

**[16]** I. Sutskever et G.E. Hinton : Learning multilevel distributed representations for high-dimensional sequences. In International Conference on Artificial Intelligence and Statistics, pages 544–551, 2007