Reconnaissance visuelle robuste par réseaux de neurones dans des scénarios d'exploration robotique. Détecte-moi si tu peux !

Joris Guerry

NNT : 2017SACLX080

THÈSE DE DOCTORAT
DE L'UNIVERSITÉ PARIS-SACLAY
PRÉPARÉE À L'ECOLE POLYTECHNIQUE

Ecole doctorale n°573
Interfaces : approches interdisciplinaires, fondements, applications et innovation
Spécialité de doctorat : Informatique

par

M. JORIS GUERRY

Reconnaissance visuelle robuste par réseaux de neurones dans des scénarios d'exploration robotique.

Détecte-moi si tu peux !

Thèse présentée et soutenue à l'École Polytechnique, le 20 novembre 2017.

Composition du Jury :

M.	THIERRY CHATEAU	Professeur Université Blaise Pascal	(Rapporteur)
M.	DAVID FILLIAT	Professeur ENSTA Paristech, U2IS	(Directeur de thèse)
M.	YANN GOUSSEAU	Professeur Télécom ParisTech	(Président)
M.	BERTRAND LE SAUX	Ingénieur de recherche ONERA, DTIS	(Encadrant)
M.	FABIEN MOUTARDE	Professeur Mines ParisTech	(Rapporteur)
M.	ROGER REYNAUD	Professeur Université Paris-Sud	(Examinateur)
M.	CHRISTIAN WOLF	Maitre de conférence INRIA	(Examinateur)

Remerciements

Mon papa m'a dit un jour : "Fils, je continuerai de travailler tant que tu n'auras pas fini tes études."

Moi : "Papa, je vais faire un doctorat".

Papa : "...."

Je souhaite tout d'abord remercier l'ensemble des membres du jury. C'est vous qui m'avez conféré le titre de docteur et par la même occasion conclu ces années d'études si particulières. Tout a commencé 4 ans avant, en dernière année des Arts et Métiers où j'ai pu suivre l'enseignement de David sur la vision par ordinateur. Malgré la moins bonne note de mon bulletin j'ai trouvé tout ça très intéressant et donc j'ai postulé en thèse sur un sujet de Bertrand avec David comme directeur de thèse. Le jour de l'entretien je suis arrivé en costume, au milieu de l'été, à la gare de Palaiseau et je me suis dirigé à pied vers l'ENSTA. Je ne savais pas à ce moment là que Saclay était en fait le plateau de Saclay, je suis donc arrivé trempé de sueur à l'entretien. Première difficulté de la thèse, loin d'être la dernière.

Comme bon Gadz'Arts que je suis, j'avais laissé les epsilons et les dérivées partielles en prépa pour faire du marteau pilon et de la fonderie pendant trois ans. Aux Arts et Métiers un petit nombre c'est 0, un grand nombre c'est 1000 et si vous divisez 10 par 3 et bien ça fait 3. Mais ce fut un plaisir de renouer avec des mathématiques qui tâchent. Je crois pouvoir dire d'ailleurs que ma thèse a changé plus que mes connaissances, elle a changé ma façon d'aborder certains problèmes, et même des problèmes de tous les jours.

Alors des problèmes j'en ai rencontré pas mal, surtout au début de ma thèse quand les résultats avaient du mal à tomber. Mais c'est pas grave, les copains sont là pour vous remonter le moral ! Guillaume, Calum, Hélène et David pour citer ceux qui ont commencé avec moi. Merci pour les franches rigolades (et les moments gênants #Calum). Encore merci Guillaume pour m'avoir permis de développer mes compétences d'échéquiste et d'y perdre beaucoup de temps (Hicham en conviendra). Merci Hicham pour tes réponses à tout, en learning comme en pause café ! Merci Maxime, Maxime et Maxime... Merci à Nicolas pour les échanges techniques, Anthelme pour les énigmes, Marcela pour ton 2nd degré, David S. pour les gateaux, Sémy pour tes tournées, Isabelle pour ta gentillesse. Merci aux stagiaires qui apportent toujours cette fraîcheur qui détend l'atmosphère (Juliette, Oriane, Martin, Xavier, Soufiane). Et puis le mercredi c'est panzani donc merci Pierre !

Je souhaite également remercier toutes les personnes qui m'ont aidé pour la construction d'ONERA.ROOM : Aurélien, Martial, Anthelme, Hélène, Alexandre E., Julien, et bien sûr Betrand (et les différents figurants !) ; ainsi que toutes les personnes qui ont été disponibles pour échanger sur des problématiques plus "Deep Learning" : Adrien, Alexandre B., Nicolas,

Maxime B., Stéphane...

Je remercie tout particulièrement mes encadrants, Bertrand et David qui ont su me recadrer quand je papillonais d'idées en idées. David, je te remercie pour avoir éveillé ma curiosité sur un domaine comme la vision par ordinateur, pour tous ces conseils que tu as pu m'apporter tout au long de ces trois années et de ta prise de recul permanente sur mes propositions. Bertrand, je te remercie également pour tes conseils mais surtout pour ta présence. Toujours disponible et à l'écoute tu as su me motiver quand j'en avais besoin et me laisser autonome de manière générale, ce qui a fait de ma thèse une excellente expérience humaine et professionnelle. En particulier nos derniers travaux ont été l'occasion de travailler véritablement ensemble, ainsi qu'avec Alexandre. C'était une réussite !

Alexandre, c'est toujours un plaisir de réfléchir et d'échanger avec toi, ne change pas !

Puis est venu le moment de la rédaction du manuscrit, où Elise, Fabrice, Alexandre et Guy m'ont proposé leur aide. Merci pour ça. Cette aide que l'on m'a proposée témoigne de la bonne ambiance qui régnait à l'ONERA, notamment en salle de pause. On pouvait y jouer aux échecs, discuter de voyage avec Philippe, observer toutes les couleurs du spectre visible avec Frédéric et ses chemises, toucher les biceps de Fabrice et parler de pleins d'autres choses et échanger avec Pauline, Valentina, Flora, Robin, Elyse, Patrick, Gilles, Alain, Annie... L'ONERA c'est aussi le café en étant toujours bien reçu chez Françoise !

Et puis quand on est doctorant ça arrive de rester tard au bureau et de discuter avec Aurélien qui s'en va : "T'es encore là toi ? Mais tu sais que t'es en thèse seulement ?" <rires+larmes> C'est également Fabrice et Elise qui s'arrêtent pour discuter : ça va me manquer !

Je voudrais enfin remercier mes parents qui m'ont toujours poussé à faire des études et qui m'en ont donné les moyens ainsi que mon frère Olivier et ma soeur Laurianne qui m'ont montré l'exemple. C'est également mon frère qui est mon exemple en terme d'humour, c'est donc auprès de lui que vous pouvez vous plaindre.

Il reste quelqu'un que je n'ai pas cité, qui m'accompagne pourtant depuis longtemps maintenant. Susana, tu étais là quand j'avais besoin de toi, pour me remonter le moral, pour me faire rire, pour corriger mon manuscrit ou pour m'écouter te parler de réseaux de neurones dans la voiture en rentrant du boulot. Tu m'as supporté pendant tout ce temps et tu m'as rendu la vie plus facile. Ça n'aurait tout simplement pas été possible sans toi. Donc un grand merci !

Une histoire s'achève...

21. dxe1=Q# .

Table des matières

Table des matières v

Liste des figures vii

Liste des tableaux xi

I Motivation et état de l'art 1

1 Introduction 3
1.1 Contexte . . . 4
1.2 Problématiques et Contributions . . . 5
1.3 Organisation du manuscrit . . . 7

2 Etat de l'art : reconnaissance visuelle 9
2.1 Introduction . . . 10
2.2 Les tâches de la vision par ordinateur . . . 10
2.3 Les jeux de données . . . 20
2.4 Conclusion . . . 26
2.5 Références . . . 26

3 État de l'art : Apprentissage profond 31
3.1 Introduction . . . 32
3.2 Les réseaux de neurones convolutionnels . . . 37
3.3 La reconnaissance visuelle avec l'apprentissage profond . . . 45
3.4 Les données multi-modales en apprentissage profond . . . 53
3.5 Conclusion . . . 56
3.6 Références . . . 57

II Contributions 63

4 Sélection d'algorithmes de classification 65
4.1 Introduction . . . 66
4.2 Preuve de concept . . . 68
4.3 Cas 1 : usage de plusieurs jeux de données . . . 72
4.4 Cas 2 : usage d'un jeu de données unique . . . 76
4.5 Conclusion du chapitre . . . 82
4.6 Références . . . 83

5 Approche multi-modale pour la détection **85**
5.1 Introduction . . . 86
5.2 Les données multi-modales . . . 86
5.3 Détection multi-modale . . . 96
5.4 Résultats de détection . . . 100
5.5 Conclusion du chapitre . . . 110
5.6 Références . . . 111

6 Approche multi-vue pour la segmentation **115**
6.1 Introduction . . . 116
6.2 La méthode : SnapNet-R . . . 117
6.3 Application mono-vue : SUNRGBD et NYUDv2 . . . 118
6.4 Application multi-vue : 3DRMS Challenge . . . 128
6.5 Conclusion du chapitre . . . 135
6.6 Références . . . 135

III Conclusions et perspectives **139**

7 Conclusion et Perspectives **141**
7.1 Conclusion . . . 142
7.2 Perspectives . . . 146
7.3 Références . . . 147

8 Liste complète des références **149**

A Figures annexes **I**

B Liste des acronymes **V**

C Glossaire **VII**

D Liste complète des références **IX**

Liste des figures

1.1 Exemples de robots mobiles 4

2.1 Différentes tâches de la vision par ordinateur 11
2.2 La tâche de classification. 11
2.3 Situations complexes de la vision par ordinateur 12
2.4 Illustration *EigenFaces* 13
2.5 Illustration d'un *Deformable Part Model* 14
2.6 Exemple de courbes précision-rappel 15
2.7 Méthode de la fenêtre glissante 17
2.8 *Non Maximum Suppression* (*NMS*) 18
2.9 *Intersection over Union* (*IoU*) 19
2.10 Exemples d'images du jeu de données CIFAR10 21
2.11 Exemples d'images du jeu de données Caltech101 21
2.12 Extrait de *Image-Net.org* 22
2.13 Exemples du jeu de données NYUDv2 23
2.14 Exemple du jeu de données SUN3D 24
2.15 Exemple du jeu de données RGBD People 25
2.16 Exemple du jeu de données ONERA.ROOM 25
2.17 Exemple du jeu de données *InOutdoor RGBD People* 25

3.1 Neurone artificiel d'un perceptron 32
3.2 Perceptron multi-couche 33
3.3 Exemples de descente de gradient 36
3.4 Exemples de filtres de convolution 37
3.5 Illustration de l'opération de convolution 3x3 38
3.6 Max pooling avec un filtre 2x2 et un pas de 2 39
3.7 Structure de CNNs : AlexNet vs. VGG 42
3.8 Exemples de micro-architectures 43
3.9 Zoo de réseaux de neurones de l'Institut Asimov 44
3.10 Illustration du *RCNN* 46
3.11 Illustration du *Fast-RCNN* 47
3.12 Illustration du *Fast-RCNN* 48
3.13 Illustration d'un "Fully Convolutionnal Network 50
3.14 Illustration d'un auto-encodeur 50
3.15 Réseau de segmentation : SegNet 51
3.16 Réseau de segmentation : U-net 51
3.17 Illustration du *Fast-RCNN* 52
3.18 Exemples de la représentation 3D PANORAMA 53
3.19 Exemple de l'encodage *HHA* 55
3.20 Fusion avec le *Gating Network* 55

3.21 Fusion avec le *FuseNet* 56

4.1 Illustration de la stratégie globale de sélection 67
4.2 Illustration du réseau sélecteur 70
4.3 Performances de *AlexNet* et LeNet sur SUN3D 71
4.4 Comparaison de la fusion par sélection avec l'oracle 71
4.5 Précisions cumulées des méthodes sur les jeux de test 73
4.6 Précisions des méthodes pour plusieurs classes sélectionnées 74
4.7 Stratégie de sélection sur *ImageNet* 77
4.8 Architecture *Dofeann* 79
4.9 Architecture de partitionnement 81

5.1 Encodage de la carte de profondeur 87
5.2 Exemple de mouvemnent du challenge *3D Hand Gesture Recogniton* 91
5.3 Exemple d'encodage pour le *workshop SHREC* 91
5.4 Matrices de confusion *SHREC 2017 Challenge* 93
5.5 Variation de luminosité dans le jeu de données NYUDv2 "Eclipse" 94
5.6 Influence de la luminosité en segmentation sémantique sur NYUDv2 95
5.7 Evaluation de l'influence de la luminosité sur les performances de segmentation 95
5.8 Aperçu global de la méthode. 96
5.9 Illustration de l'architecture du *Gating Network* 96
5.10 Illustration du fonctionnement du *Gating Network* 97
5.11 Illustration de la méthode idéale de détection 97
5.12 Illustration globale du *Faster-RCNN* 97
5.13 Illustration globale du *Faster-R&M_CNN* 98
5.14 Les architectures de fusion du Faster-RCNN 100
5.15 Prédictions sur *RGBD People* 101
5.16 Exemples de prédictions de nos méthodes multimodales 102
5.17 Robby : le robot de l'ONERA utilisé pour les acquisitions d'ONERA.ROOM 103
5.20 Fusion en X sur ONERA.ROOM, situation 1 105
5.21 Fusion en X sur ONERA.ROOM, situation 2 106
5.22 Fusion en X sur ONERA.ROOM, situation 3 106
5.23 Vidéo de ONERA.ROOM en ligne 107
5.18 Influence de la diminution de luminosité sur la fusion en X 108
5.19 Courbes Précision-Rappel (IoD RGBD People & ONERA.ROOM) 109

6.1 Exemple d'image *RGB-D* vue en 3D 116
6.2 Processus complet de SnapNet-R 117
6.3 Exemples d'une mauvaise stratégie d'observation sur des données mono-vues 119
6.4 Aperçu du jeu de données SUNRGBD 120
6.5 Filtrage du "meshing" pour l'estimation d'agencement 121
6.6 Illustration de la stratégie multi-vue pour l'estimation d'agencement 121
6.7 Exemple d'estimation d'agencement avec SnapNet 122
6.8 Stratégie d'observation sur des données mono-vues 123
6.9 Répartition des objets dans SUNRGBD 124
6.10 Résultats qualitatifs de segmentation sur SUNRGBD 126
6.11 Aperçu du jeu de données *3DRMS Challenge* 129
6.12 Les différentes méthodes de reconstruction 3D 129
6.13 Résultats qualitatifs du challenge *3DRMS* 2017 133
6.14 Résultats de la reconstruction sur le challenge 3DRMS 2017 1/2 134

6.15 Résultats de la reconstruction sur le challenge 3DRMS 2017 2/2 134

A.1 Structure du *CNN* LeNet I
A.2 Structure du *CNN* AlexNet II
A.3 Structure du *CNN* SqueezeNet III
A.4 Structure du *CNN* Dofeann IV

Liste des tableaux

2.1 Comparaison des jeux de données *RGB-D* 26

3.1 Résumé des méthodes de détection par *CNN* 48
3.2 Méthodes RGB-D pour la détection d'objets. 56

4.1 Temps de calcul pour différentes architectures CNN 67
4.2 Pourcentage de bonnes classifications communes entre les méthodes. 75
4.3 Evaluation de la précision moyenne des méthodes sur l'ensemble des jeux de données. L'oracle désigne la méthode résultant de la sélection du meilleur classifieur disponible à chaque image. Les réseaux de confiance (ou RCs) désignent les différentes architectures utilisées pour prédire le meilleur classifieur à utiliser pour une image donnée. 76
4.4 Précision et temps de calcul des réseaux classifieurs sur *ImageNet* 78
4.5 Complémentarité des réseaux classifieurs sur *ImageNet* 78
4.6 Résultats du réseau de confiance *Dofeann* sur *ImageNet* 80
4.7 Résultats de l'approche par clustering sur *ImageNet* 82

5.1 Détection d'objets avec *Faster RCNN* sur le jeu de données NYUDv2 89
5.2 Comparaison des encodages sur de la segmentation sémantique 90
5.3 Résultats au *workshop SHREC* 92
5.4 Comparaison des performances au *workshop SHREC* 93
5.5 *EER* sur *RGBD People* LUBER et al. [2011] pour différents détecteurs. 101
5.6 Performance de nos détecteurs sur *InOutdoor RGBD People* 103
5.7 Performance de nos détecteurs sur ONERA.ROOM 104

6.1 Résultats de l'estimation d'agencement sur le jeu de données SUNRGBD. . . . 122
6.2 Résultats quantitatifs de segmentation sur SUNRGBD 125
6.3 Résultats quantitatifs de SnapNet-R sur NYUDv2 127

Première partie

Motivation et état de l’art

Chapitre 1

Introduction

> *Le robot intelligent n'existe pas, en tout cas pas encore. Peut-être émergera-t-il dans quelques décennies, mais au prix d'importants progrès non seulement technologiques mais aussi qualitatifs et théoriques. Pour l'instant nous ne savons produire que des robots très spécialisés.*
>
> Cédric Villani, Interview Le Monde, 2017

Sommaire

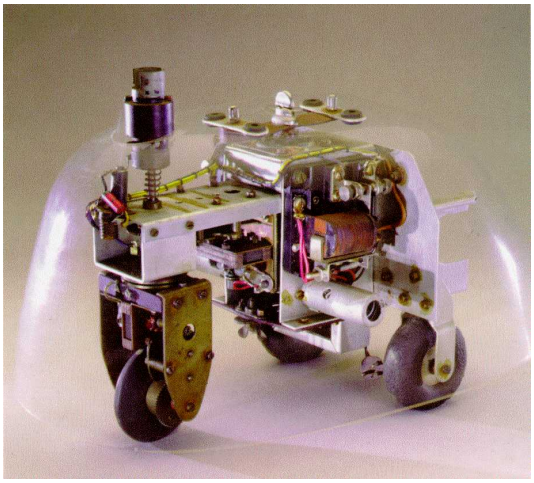
1.1 Contexte **4**
1.2 Problématiques et Contributions **5**
1.3 Organisation du manuscrit **7**

1.1 Contexte

De la Tortoise de Grey Walter (1950) 1.1(a) à Handle de Boston Dynamics (2016) 1.1(c) en passant par Asimo de Honda (2000) 1.1(b), les robots mobiles sont de plus en plus capables d'évoluer dans le monde et d'interagir avec leur environnement.

(c) Handle (Boston Dynamics 2016)

(b) Asimo (Honda 2000)



(a) Tortoise (Grey Walter 1950)

FIGURE 1.1 – Exemples de robots mobiles

Pour pouvoir aller plus loin, plus longtemps et pour augmenter leur panel de compétences les robots doivent témoigner d'une grande adaptabilité. Cette capacité à modifier leur comportement et de s'ajuster au milieu est essentielle. Monter des escaliers nous semble une activité banale, et pourtant nous sommes, nous humains, sensibles à une petite différence dans la taille des marches au point de pouvoir trébucher. Dans le cadre d'interactions sociales il est primordiale de reconnaître sans faute des personnes ; pour la conduite d'un véhicule il est nécessaire de différencier un obstacle sur route ou de discerner les feux tricolores malgré leur variabilité déjà grande à l'échelle d'un pays ; savoir lire des caractères manuscrits provenant de personnes différentes ; et tout cela de jour comme de nuit, de loin comme de près, de face, de derrière ou de trois quarts. Plus que des objets ce sont parfois des concepts que nous apprenons. Ils résultent de l'accumulation de connaissances "multi-sens", contextualisées et qui évoluent dans une représentation riche du monde qui nous entoure. L'ensemble de ces mécanismes acquis consciemment et inconsciemment au fil des années permet à l'homme une grande robustesse dans la reconnaissance de ces concepts. Un robot, quant à lui, est limité par sa modélisation du monde, elle-même limitée par sa puissance de calcul, de mémoire, de capteurs, de systèmes d'action, d'échange social, *etc*. Chaque problématique est complexe

et forme un pan entier de la science d'aujourd'hui. Concevoir un robot est donc une tâche de conciliation technique, généralement contrainte par un cahier des charges précis avec des limites budgétaires fortes. Le roboticien se doit donc de rivaliser d'ingéniosité dans ses choix de solutions techniques (parfois au dépend de la qualité matérielle si l'intégration logicielle le permet). Pour cela, la nature est souvent source d'inspiration et quoi de mieux que de regarder l'homme pour concevoir un robot qui évoluera dans le monde des hommes.

Des cinq sens, la vision est une source très riche d'informations pour analyser le monde environnant sous réserve de réussir à en extraire le contenu. La vision permet à l'homme de planifier ses actions, reconnaître à distance des objets, se localiser... C'est exactement dans ce contexte que se situe cette thèse : la reconnaissance visuelle par un robot. Comme la robotique, c'est une science de compromis. Puissance de calcul, mémoire, capteurs, modélisation... Nous avons plus particulièrement cherché à répondre à ce besoin de robustesse qu'un robot doit pouvoir fournir pour faire face à des scénarios complexes en terme de conditions d'acquisition de l'image. Nous fixons dans la section suivante les problématiques globales de ces travaux de thèse.

1.2 Problématiques et Contributions

Le cadre général de ce travail peut être formalisé par les questions suivantes :

1. Comment percevoir ?
2. Comment reconnaître ?
3. Comment s'adapter ?
4. Quelles sont les contraintes et avantages liés à la robotique ?

La problématique n°1 concerne l'aspect capteur visuel, acquisition de l'information brute et pré-traitements : données couleurs (RGB) en 2D, données de profondeur (Depth) pour de la 2.5D, nuage de points (3D), encodage de l'information (c'est-à-dire traduction, transformation, filtrage...), *etc*.

La problématique n°2 concerne les méthodes à utiliser pour modéliser des classes et mesurer une similarité entre les modèles et les observations.

Les problématiques n°3 et n°4 sont étroitement liées : en effet la robotique impose des scénarios, des variations, et de manière générale des contraintes, qui nécessitent des algorithmes pouvant faire preuve d'adaptabilité (par fusion, usage de la redondance, ou encore connaissances *a priori*). Par exemple, les conditions d'observations peuvent changer entre la phase d'apprentissage et la phase d'utilisation (caméras différentes, luminosités différentes, angles de vue, ...) ce qui entraîne généralement une perte de performance. Le terme domaine est employé pour désigner les différentes distributions statistiques représentants les données : les méthodes employées doivent donc être capables d'effectuer de l'adaptation de domaine.

Ce travail de thèse en plus de vouloir apporter des stratégies robustes de reconnaissance visuelle dans le cadre de scénario robotique, se donne aussi comme objectif d'étudier le spectre des possibilités offert par l'apprentissage dit "profond". Le terme d'apprentissage profond est parfois abusif et difficile à quantifier tant les structures ont pu évoluer durant les dernières années (passant par exemple du LeNet à 7 couches, jusqu'au macro-réseaux comme le ResNet à 1000 couches). C'est ainsi que nous proposons diverses contributions à base de réseaux de neurones artificiels, appliqués à des données visuelles robotiques.

Bien que l'embarquabilité ne soient pas au coeur de nos travaux, nous avons gardé en tête des considérations de temps de calcul pour assurer de l'interactivité, c'est-à-dire

des traitements de plusieurs images par seconde. De nombreux travaux plus bas niveau s'intéressent à l'implémentation *hardware* de méthodes que nous utiliserons. Ce n'est pas dans le cadre de nos travaux.

Durant cette thèse nous avons plus précisement abordé les problématiques suivantes :

— Dans nos premiers travaux nous nous sommes intéressés à la sélection d'algorithmes pour la classification d'objets. L'objectif essentiel étant de déterminer s'il est possible d'utiliser des réseaux de neurones pour sélectionner à la volée le classifieur qui s'appliquera le mieux sur les données parmi un ensemble de méthodes disponibles. Cette approche par sélection s'oppose dans son principe au concept de fusion. Pour garder un intérêt face aux méthodes ensemblistes il est nécessaire de créer un réseau sélecteur économe en ressources mais tout de même discriminant vis-à-vis des méthodes de classification.

— À la suite de l'étude de sélection par réseaux de neurones il nous a semblé judicieux d'intégrer une seconde modalité dans le processus de reconnaissance visuelle : en l'occurence, une carte de profondeur. Certaines situations se prêtent en effet tout particulièrement à l'usage d'une modalité ou d'une autre (caméra classique couleur en plein soleil et caméra "carte de profondeur" active en environnement obscur par exemple). Cet ajout d'une source d'information supplémentaire vient naturellement s'intégrer dans le cadre du projet INACHUS [1] avec des problématiques de Search and Rescue où les lieux d'explorations peuvent être coupés électriquement et donc non éclairés voire enfumés. Nous avons particulièrement travaillé sur la détection de personnes vis-à-vis de cette problématique.

— Enfin, après avoir travaillé en couleur (2D), puis en couleur+carte de profondeur (2.5D), nous avons décidé d'orienter nos travaux vers l'usage de l'information spatiale (3D), en particulier pour la segmentation sémantique. Nous proposons ainsi une étude sur l'usage d'une stratégie multi-vue appliquée à des données *RGB-D* vue comme des nuages de points colorés. Ce genre d'approche peut s'appliquer à la suite d'un *SLAM* dans un environnement reconstruit en 3D ou bien directement sur une vue unique *RGB-D* où la variation du point d'observation peut apporter une information discriminante pour la tâche désirée.

1. http://www.inachus.eu

1.3 Organisation du manuscrit

Ce manuscrit est organisé en deux parties principales, la première traitant des motivations de cette thèse et de l'ensemble des études existantes utiles à la présentation de notre travail, la seconde étant consacrée à la présentation de nos contributions, les conclusion et les perspectives formant une 3ème partie :

Partie I

— Chapitre 2 : Nous présentons l'ensemble des connaissances nécessaires vis-à-vis de la reconnaissance visuelle ainsi que les méthodes historiques de ce domaine.

— Chapitre 3 : Nous abordons à part les méthodes connexionnistes, plus généralement appelée "apprentissage profond". Nous tâcherons d'orienter cette présentation vis-à-vis des contraintes du monde de la robotique mobile.

Partie II

— Chapitre 4 : Nous nous focalisons tout d'abord sur des méthodes à une modalité (**2D**). Notre première contribution s'intéresse à la manière de choisir la meilleure méthode au sein d'une liste disponible en fonction de la situation courante. Nous avons décidé d'utiliser des réseaux de neurones artificiels pour prendre cette décision et proposons différentes structures pour remplir ce rôle.

— Chapitre 5 : Nous étudions ici les différentes façons de fusionner l'information *RGB-D*[2] (**2.5D**) au moyen de réseaux de neurones artificiels. Tout d'abord en encodant l'information de profondeur de manière à en tirer le meilleur parti, puis en utilisant des réseaux de neurones multimodes pour la détection de personnes.

— Chapitre 6 : Ce dernier chapitre de contribution cherche à gagner une dimension supplémentaire dans l'espace des données d'entrée en considérant l'aspect spatial des images *RGB-D*. Nous appliquons ici une stratégie multi-vue dans un environnement **3D**. Cette stratégie s'applique également à des données obtenues directement sous forme de nuages de points 3D (acquis par laser ou par *SLAM* par exemple).

Partie III

— Chapitre 7.1 : Pour conclure nous donnons la liste des points à retenir de ces travaux et une liste de recommandations à prendre en compte pour l'usage de l'apprentissage profond dans un cadre robotique.

— Chapitre 7.2 : Enfin, nous abordons les améliorations envisageables et les perspectives intéressantes offertes par l'état final de cette étude.

2. L'usage du terme anglais est très courant pour désigner des images en trois canaux (Rouge, Vert, Bleu) et un quatrième canal contenant l'information de profondeur (*Depth*).

Chapitre 2

Etat de l'art : reconnaissance visuelle

> *Computer vision is concerned with the automatic extraction, analysis and understanding of useful information from a single image or a sequence of images. It involves the development of a theoretical and algorithmic basis to achieve automatic visual understanding.*
>
> The British Machine Vision Association and Society for Pattern Recognition, http://www.bmva.org/visionoverview, 2017

Sommaire

2.1 Introduction 10
2.2 Les tâches de la vision par ordinateur 10
2.2.1 La classification 11
2.2.2 La détection d'objet 16
2.2.3 La segmentation sémantique 19
2.3 Les jeux de données 20
2.3.1 Les jeux de données RGB 20
2.3.2 Les jeux de données *RGB-D* 22
2.4 Conclusion 26
2.5 Références 26

2.1 Introduction

Si la vision par ordinateur pourra à terme englober des approches très variées de perception machine (comme pour l'homme dont la vision est enrichie par de nombreuses autres informations) elle consiste le plus souvent en l'extraction et le traitement de l'information contenue dans une image (et par extension dans une vidéo). Nous n'étudierons pas la physique des capteurs visuels et les pré-traitements bas-niveaux. La matière première de notre étude est l'image numérique, au sens d'un tableau de valeurs discrètes issues de divers capteurs.

Une image peut être extraite d'une caméra, ou d'un capteur plus complexe. Elle peut être en noir & blanc ou bien multispectrale, volontairement floue, infrarouge, *etc*. Voici quelques exemples d'application : la reconnaissance de caractère pour le traitement automatisé des chèques bancaires ou du courrier, la détection de visage sur l'écran d'un appareil photo, la ré-identification sur les réseaux sociaux ou pour les systèmes biométriques, la reconnaissance des émotions pour les interfaces homme-machine ou les portiques de sécurité dans les aéroports, l'analyse du bon déroulement d'une chaîne de conception, la détection de défaut en production ou de fissures sur des ouvrages, la reconstruction d'environnement et la localisation, l'analyse de situation ou de comportement, le suivi de cible, et bien plus encore. En particulier, la vision par ordinateur permet à un robot mobile de percevoir le monde qui l'entoure, de s'y localiser, d'éviter des obstacles, sous réserve de savoir ce qu'est un obstacle.

Pour permettre une comparaison des stratégies proposées par la communauté ou bien simplement pour tester la répétabilité de certains traitements il est nécessaire de construire des jeux de données invariants. C'est pourquoi nous présentons dans un premier temps les tâches de la vision par ordinateur abordées dans cette thèse ainsi que les jeux de données associés et acceptés par la communauté. Nous présentons ensuite les méthodes historiques appliquées sur les problématiques de ces jeux de données.

2.2 Les tâches de la vision par ordinateur

La vision par ordinateur s'est structurée autour de grands défis à relever. Nous nous sommes particulièrement intéressés à :

- la classification
- la détection (localisation + classification)
- la segmentation sémantique (classification de chaque pixel d'une image)

voir Figure 2.1

Nous détaillerons ces tâches dans cette section. D'autres tâches y sont intimement liées, comme par exemple et de façon non exhaustive :

- La différenciation et segmentation d'instances (chaise n°1 versus chaise n°2)
- La ré-identification (chaque classe est un unique objet)
- Le "captioning" (description des images, au service des aveugles par exemple)

Il est important de souligner que ces différentes tâches sont conceptuellement liées entre-elles. Elles ont toutes pour objectif d'attribuer des classes : que ce soit à l'image entière, à une partie de l'image ou à un pixel de l'image. La classification est donc au cœur de notre étude.

En ce qui concerne les séquences d'images (ou vidéos) d'autres défis apparaissent, liés à la temporalité des données ou à leurs proximités spatiales :

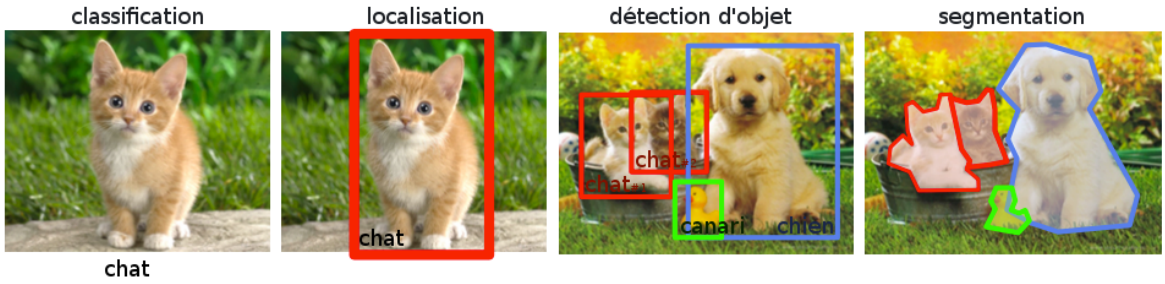


FIGURE 2.1 – Différentes tâches de la vision par ordinateur. La dernière colonne sur la segmentation fait la différence entre les deux instances de chat, ce qui est une tâche supplémentaire non triviale par rapport à la segmentation seule.

— Le suivi d'objet dans des vidéos (*tracking*),
— L'appariement d'images (pour l'exploitation de données –*data mining*– par exemple),
— La localisation et cartographie simultanées – ou *Simultaneous Localization And Mapping* (*SLAM*),
— La reconnaissance d'activité ("jouer au piano" *versus* "sauter en l'air").

2.2.1 La classification

Nous faisons ici référence à la classification automatique. Elle consiste à attribuer pour une entrée donnée dans l'espace des images une ou plusieurs catégories, non nécessairement exclusives.

L'espace des images

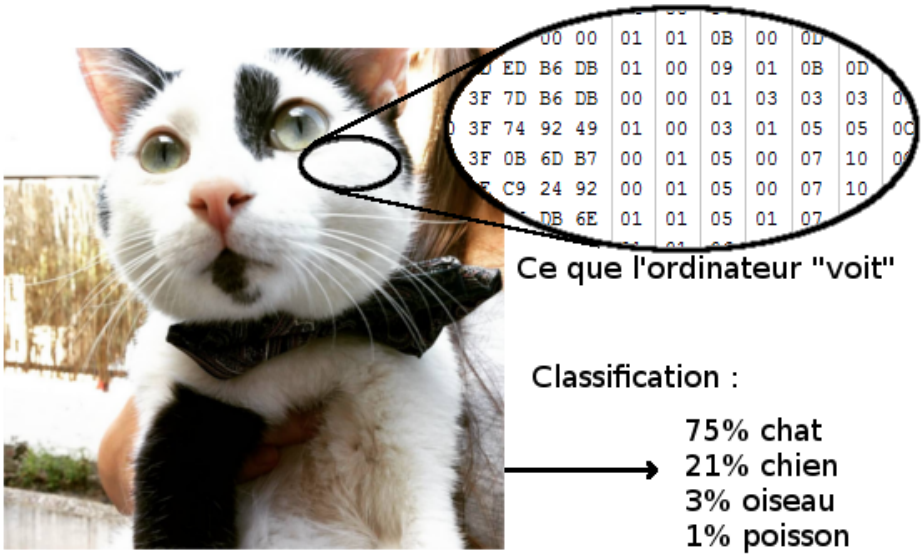


FIGURE 2.2 – La tâche de classification.

Définissons tout d'abord l'espace des images. Nous pouvons définir une image par son intensité I en fonction du pixel observé en position (x, y) : $I(x, y)$ (voir Figure 2.2). Dans le

cas d'une image rouge, vert, bleue – ou *Red, Green, Blue* (*RGB*) – chaque canal possède une intensité I_c correspondante. Nous dénotons la largeur de l'image –*width*– par w et sa hauteur –*height*– par h. Le nombre de canaux (1 pour une image en niveaux de gris, 3 pour une image *RGB*) sera noté d pour *depth* (profondeur). Ainsi pour des dimensions données (w, h, d) une image est un point dans un espace en $w * h * d$ dimensions. Dans le cas d'une image dite "*8bits*" (où l'intensité est stockée par une valeur variant de 0 à 255), il existe 256 possibilités pour chaque composante, c'est-à-dire 256^{w*h*d} images différentes. Les caméras que nous avons utilisés dans ces travaux produisent des images de taille (640, 480, 3). Cela signifie qu'il existe environ $10^{2219433}$images potentielles. Bien sûr ce nombre d'images est surestimé par rapport aux images que nous sommes susceptibles de rencontrer. De par les *a priori* que nous connaissons sur le monde (le ciel est en haut, les objets sont sur le sol, les plantes sont vertes, *etc*.) ce nombre d'images potentiellement existantes diminue drastiquement. Nous avons donc comme données d'entrée brutes, des vecteurs de grandes dimensions qui évoluent dans un espace très peu dense –*sparse*. Et pourtant, nous cherchons à différencier des images entre-elles pour leur attribuer certaines classes.

Séparabilité des classes et espace de représentation

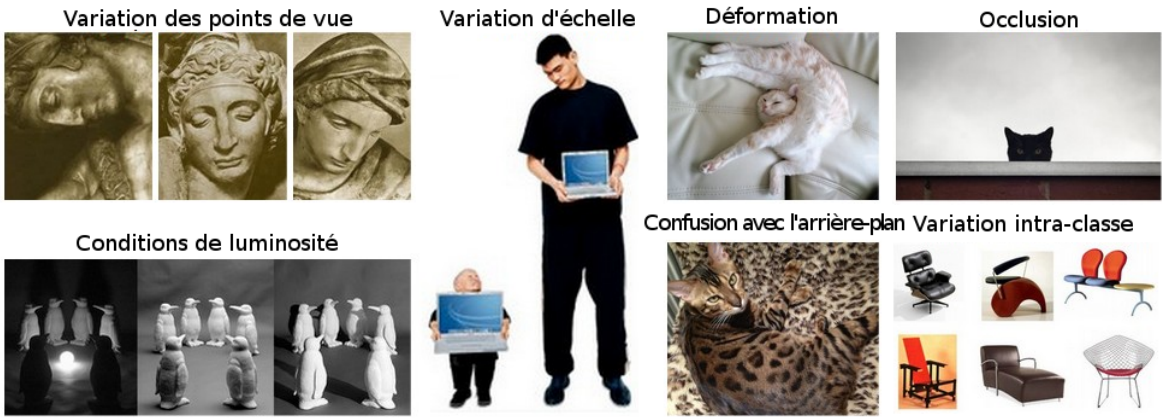


FIGURE 2.3 – Situations complexes de la vision par ordinateur (traduit d'une présentation de Andrew Zisserman, VGG, Oxford).

Le problème de la classification peut être vu comme un problème de séparation des données en groupes, parfois appelés *clusters* : le groupe des chiens, le groupe des chats, *etc*. La séparabilité des données venant d'images est particulièrement complexe à cause des différents problèmes illustrés dans la Figure 2.3. Il faut donc travailler dans un espace de représentation des données qui permet de faire cette différence. Après obtention d'un espace de représentation permettant la séparabilité des données il faut y définir des partitions et être capable de mesurer l'appartenance à ces partitions.

Cependant, en raison du fléau de la dimension –ou *curse of dimensionality* [BELLMAN, 1961]– il est impossible en pratique de différencier des points directement dans l'espace d'origine de nos données. Il est donc nécessaire de réduire les dimensions du problème. Pour cela, il nous est possible d'utiliser des méthodes de réduction de dimensionalité ou de définir artisanalement des indices de description.

Réduction de la dimensionnalité La première méthode de réduction de dimensionnalité consiste à décomposer l'espace de travail selon les axes où la variance est maximale : c'est la décomposition en composante principale –ou *Principal Component Analysis* (*PCA*). Il est

ensuite possible de déterminer le nombre de composantes désirées et de conserver les axes ayant la plus grande variance, ce qui est équivalent à conserver les axes qui supportent le plus d'information. Cette stratégie de composante principale a été appliquée par TURK et PENTLAND [1991] et leurs célèbres "EigenFaces" (voir Figure 2.4)

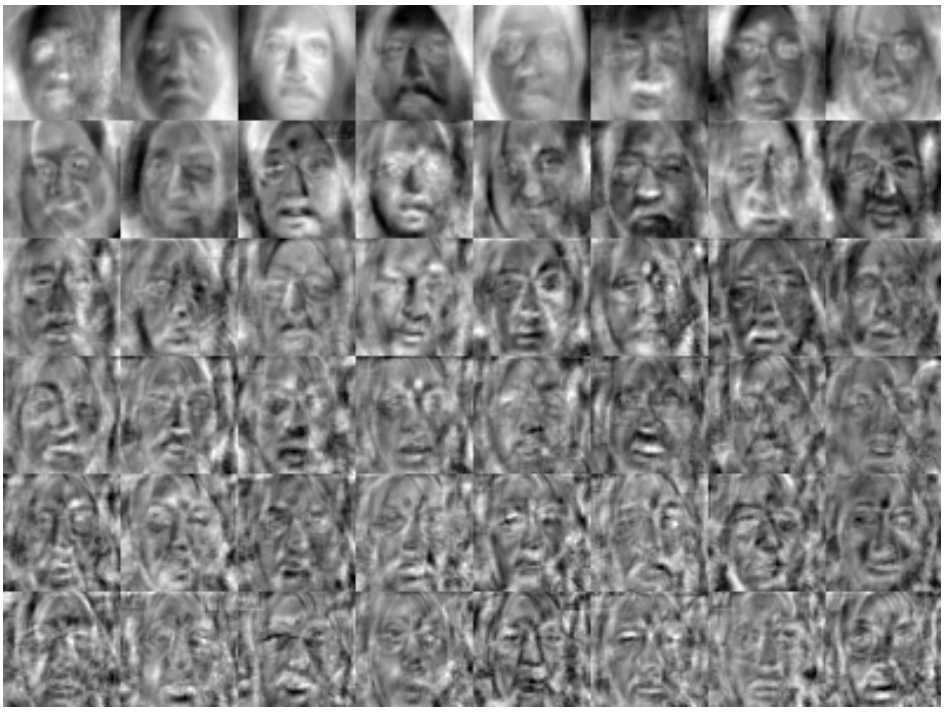


FIGURE 2.4 – Exemple d'*Eigenfaces* de TURK et PENTLAND [1991]. Tout visage peut se décomposer comme une somme pondérée des visages ci-dessus et d'une perte qui dépend d'un nombre de dimension choisi pour la *PCA*.

Il existe également des alternatives à la *PCA* qui réduisent la dimension de l'espace d'origine en utilisant des indices définis "à la main". C'est le cas de l'approche par histogramme de gradients orientés – ou *Histogram of Oriented Gradients* (*HOGs*) – [DALAL et TRIGGS, 2005]. En calculant le gradient de l'image nous faisons ressortir les contours qui sont caractéristiques des formes et donc des classes d'objets. Dans cet espace de *HOGs*, FELZENSZWALB et al. ont défini des modèles déformables composés de différentes parties agencés entre-elles qui peuvent se déformer et ainsi s'adapter aux angles de vue, aux orientations et aux objets déformables. L'exemple de l'article FELZENSZWALB et al. [2010] est disponible dans la Figure 2.5. Dans cette catégorie d'approches, pouvons également citer les *SIFT* "*Scale-invariant feature transform*" LOWE [1999], les *SURF* "*Speeded Up Robust Features*" BAY et al. [2006], les "local binary patterns" de OJALA et al. [2002] ou encore les méthodes à base de "*bag of visual words*" comme dans YANG et al. [2007].

Il est également possible de définir des indices arbitrairement comme le pourcentage de pixels rouges, le nombre de correspondances avec un certain motif ou l'histogramme des variances d'une sous-division de l'image. Ces indices "*handcrafted*" nous permettent d'injecter directement des *a priori* humainement compréhensibles dans notre tâche de séparation des données. Cette approche a été utilisée dans WANG et al. [2014] où des caractéristiques définies par l'homme (morphologiques, statistiques et texturales) sont couplées avec des caractéristiques extraites de réseaux de neurones (voir Chapitre 3 sur l'apprentissage profond). Nous avons également appliqué ce principe dans le Chapitre 5 (Section 5.2.1) pour orienter et donc faciliter l'entraînement d'un réseau de neurones.

Une implémentation globale dans un scénario d'exploration par drone effectuée à l'ONERA

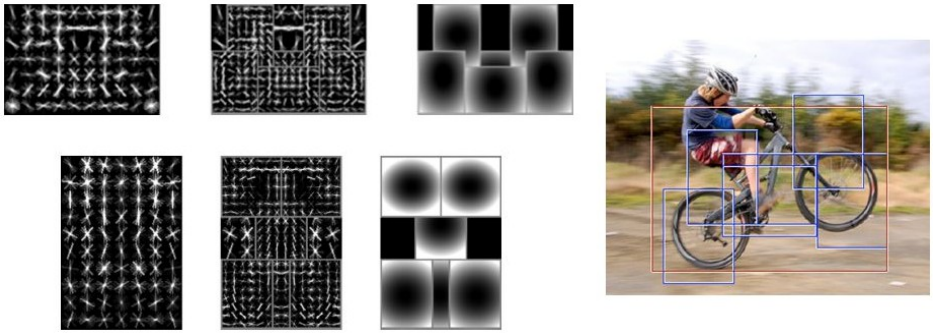


FIGURE 2.5 – Illustration d'un *Deformable Part Model*

est présentée dans SANFOURCHE et al. [2014]. Ils utilisent les "local binary patterns" et des *HOGs* sur des données acquises par approche *SLAM*. Une composante intéressante de ces travaux est l'interaction homme-machine qui permet à un utilisateur de la station-sol de désigner des exemples positifs et négatifs pour apprendre de nouvelles classes au cours du scénario. Bien que non employée dans nos travaux, cette approche interactive est très intéressante vis-à-vis de l'adaptation de domaine. En effet, les données d'entraînements hors-ligne sont généralement légèrement différentes des données en situation réelle. L'opérateur peut donc corriger ce biais en direct si la méthode de classification le permet. En ce qui concerne l'apprentissage de modèles d'objets à la volée pour une reconnaissance nous pouvons citer les travaux de KALAL et al. [2012]. Bien qu'orientés vers une problématique de *tracking* vidéo, ces travaux apportent une solution avec exécution temps-réel pour apprendre un modèle d'objet avec un seul exemple et ajouter des composantes à ce modèle en fonction de l'apparence de l'objet tracké qui peut évoluer.

Apprentissage de métrique et *clustering* Une fois projetées dans un espace de représentation exploitable nous pouvons tenter de différencier nos données entre elles. Pour cela nous pouvons avoir recours à diverses méthodes de partitionnement – ou *clustering*. Une première solution consiste à apprendre une distance entre les objets qui aura tendance à éloigner les objets de classes différentes et à rapprocher les objets d'une même classe. Cette discipline est également appelée *metric learning*. Nous invitons le lecteur intéressé à regarder une étude comparative sur les méthodes de *metric learning* : KULIS et al. [Metric learning : A survey 2013]. Une fois une distance déterminée dans l'espace de travail, il est possible de regrouper les objets d'une même classe en déterminant leurs centroïdes, en utilisant des méthodes de type k-moyennes FORGY [1965].

Une autre approche consiste à déterminer les hyperplans qui séparent les classes entre elles (approche par machine à vecteurs de support – ou *Support Vector Machine* (*SVM*) – et sa variante à noyau). De nombreuses autres approches existent, voir par exemple XU et WUNSCH [Survey of clustering algorithms 2005]. Nous abordons les approches par apprentissage profond dns le Chapitre 3.

Evaluation d'une classification

Pour évaluer la qualité d'une méthode de classification nous définissons la précision pour chaque classe :

$$\mathrm{P} = \frac{TP}{\mathrm{N}} = \frac{TP}{TP + FP} \tag{2.1}$$

où

— N désigne le nombre d'images proposées.

— *True Positive* (*TP*), désigne les propositions d'une méthode correctement classifiées comme vraies.

— *False Positive* (*FP*), désigne les propositions d'une méthode classifiées comme vraies alors qu'elles sont fausses.

— *True Negative* (*TN*), désigne les propositions d'une méthode correctement classifiées comme fausses.

— *False Negative*(*FN*), désigne les propositions d'une méthode classifiées comme fausses alors qu'elles sont vraies.

Nous définissons également le rappel comme la capacité à trouver les images contenant une classe donnée :

$$\mathrm{R} = \frac{TP}{\mathrm{N}_p} = \frac{TP}{TP + FN} \tag{2.2}$$

où

— N_p désigne le nombre total d'images contenant la classe considérée.

Analyser un seul de ces indices n'est pas suffisant. Dans un jeu de données constitué pour moitié de voitures et pour moitié de motos, une méthode qui propose systématiquement des motos aura 100% de rappel sur les motos (elles sont bien toutes proposées pour l'évaluation) mais le rappel sur les voitures sera de 0%. De plus, la précision globale de la méthode sera de 50%. C'est pourquoi ces deux indices sont généralement regroupés ensemble dans une courbe paramétrique précision-rappel qui permet d'évaluer la précision et le rappel en fonction d'un certain paramètre de la méthode (voir Figure 2.6)

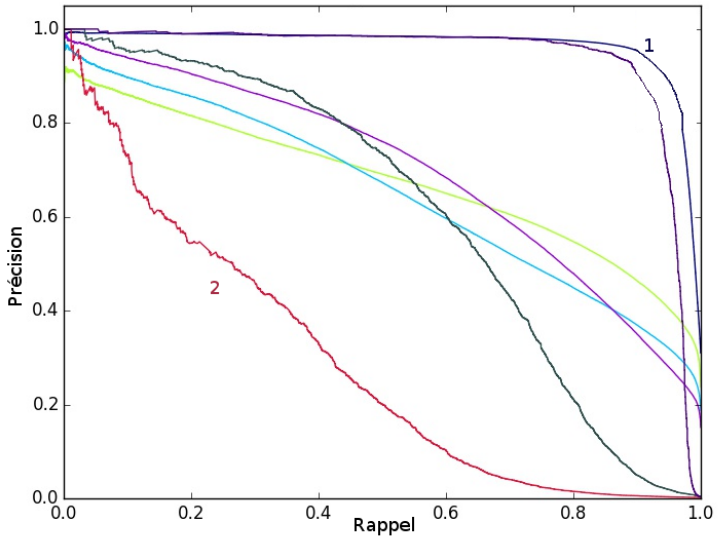


FIGURE 2.6 – Exemple de courbes précision-rappel. La courbe 1 correspond au classifieur le plus performant au sens où même pour un rappel élevé la précision reste élevée également. La courbe 2 montre une chute de précision lorsque le rappel augmente.

En pratique, la courbe précision-rappel n'est pas très pratique pour classer les méthodes entre elles et nous préférons calculer des indices scalaires qui tentent de résumer les courbes, par exemple le "map" de EVERINGHAM et al. [2010].

Nous définissons également la précision moyenne (PM) comme la moyenne des précisions par classe. Elle permet de contrebalancer une disproportion des classes d'un jeu de données en accordant autant d'importance à chaque classe.

2.2.2 La détection d'objet

La détection d'objet consiste à définir une zone –très souvent rectangulaire– dans une image (localisation) et de lui attribuer une classe d'objet (classification). Nous parlerons de patch pour définir cette zone. La tâche de classification considère un patch et répond à la question "Est-ce que l'objet de la classe considérée est dans ce patch ?". La tâche de localisation quant à elle, a pour objectif de définir les dimensions et la position du patch dans une image.

Etant donné que nous travaillons à classifier des patchs, il faut proposer des patchs au classifeur, donc nous avons besoin d'une stratégie de proposition de patchs.

Une fois les patchs obtenus nous pourrons appliquer un processus de classification expliqué dans la section précédente.

Stratégies de proposition de patchs

Il existe plusieurs approches pour faire de la détection. La plus classique d'entre elles consiste à appliquer un classifieur partout dans l'image, sur des patchs de formes et d'échelles variées. Si le score du classifieur est suffisant (e.g supérieur à un certain seuil), alors on peut valider le patch (localisation induite). Ce genre d'approches possède un inconvénient majeur : la proportion de pixels appartenant à des objets recherchés est souvent faible par rapport à la surface de l'arrière plan. Il faut donc utiliser des classifieurs performants face à un biais en proportion (on parle du *class-imbalance problem* [JAPKOWICZ et STEPHEN, 2002], qui sera une problématique clé dans le Chapitre 6). La stratégie de proposition de patchs est donc critique vis-à-vis de la complexité de la tâche du classifieur ainsi que du nombre de patchs proposés et donc du temps de calcul et des ressources mémoires.

a) Stratégie par fenêtre glissante L'approche de référence pour la localisation d'objet est basée sur la classification. En effet, il est possible d'appliquer une stratégie de fenêtre glissante – ou *Sliding window* (FG) – (voir Figure 2.7) sur l'image entière qui propose une liste arbitrairement pré-determinée de patchs. Les patchs peuvent varier en taille et en forme et la stratégie de parcourt sur l'image peut être exhaustive (allant de pixel en pixel) ou bien partiellement lacunaire en effectuant des pas supérieurs à un pixel (on parle parfois du *stride*). Cette approche présente l'avantage d'être aussi exhaustive qu'on le souhaite, cependant elle peut proposer trop de patchs qui nécessitent tous d'être traités par la méthode de classification.

Cette méthode a été appliquée par VIOLA et al. [2005]. VIOLA et al. emploient une méthode de détection de visage basée sur des descripteurs locaux dont une vidéo illustrative (fenêtre glissante + descripteur) est disponible (voir HARVEY [2010]).

b) Les autres stratégies Il existe d'autres méthodes opérant une sélection des patchs proposés plus subtile pour gagner en temps de calcul, quitte, en général, à perdre en rappel. Pour donner quelques exemples :

— la "*Selective Search*" de UIJLINGS et al. [2013],

— l' "*Objectness*" de ALEXE et al. [2012],

— les "category-independent object proposals" de ENDRES et HOIEM [2010],

FIGURE 2.7 – Méthode de la fenêtre glissante

— les "constrained parametric min-cuts (CPMC)" de CARREIRA et SMINCHISESCU [2012],

— le "multi-scale combinatorial grouping" de ARBELÁEZ et al. [2014].

Nous n'avons pas fait usage de ces méthodes dans ces travaux de thèse car nous nous sommes orientés vers une stratégie de proposition de patchs par réseau de neurones (voir Chapitre 3, Section 3.3.2). Il est cependant intéressant d'y reconnaître de nombreux concepts à l'origine du *Region Proposal Network* de REN et al. que nous détaillerons dans le Chapitre "État de l'art : Apprentissage profond".

c) La classification au service de la localisation Les stratégies précédemment évoquées sont totalement agnostiques au classifieur. Pourtant savoir quel objet nous cherchons peut orienter la recherche. Par exemple si nous cherchons une voiture sur la route il est possible de se concentrer sur des rectangles horizontaux pour économiser du temps de calcul. Le Chapitre "État de l'art : Apprentissage profond" (page 31) introduit une stratégie dans laquelle la méthode de classification vient apporter un *a priori* sur la forme et permet d'améliorer la localisation. Nous utiliserons d'ailleurs cette stratégie dans le Chapitre "Approche multi-modale pour la détection" (page 85).

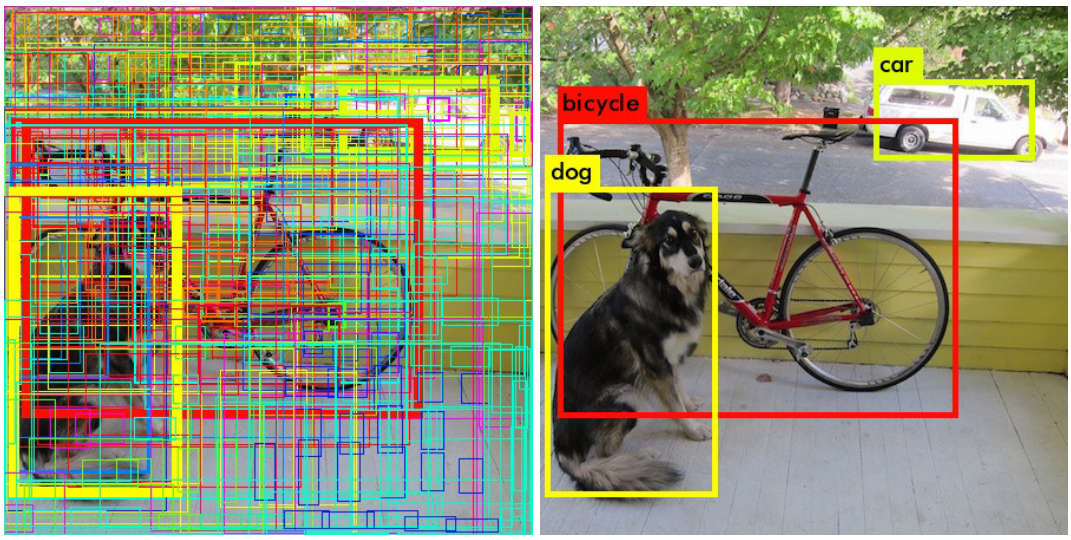
Sélection des meilleurs patchs

Une fois que les patchs ont été proposés et classifiés il est possible que plusieurs patchs soient retenus pour un même objet, variant légèrement en position, en forme ou en échelle (voir Figure 2.8(a)). Cette situation demande de filtrer les détections supplémentaires (comptabilisés comme des fausses détections par la totalité des métriques de détection). Pour choisir la meilleure détection candidate, l'algorithme de suppression des non-maximums – ou *Non Maximum Suppression (NMS)* – est massivement utilisé. Il consiste à :

1. ranger les détections selon le score du classifieur, du plus grand au plus petit,

2. mettre de côté la première détection,
3. pour chaque détection restante, supprimer cette détection si elle s'intersecte trop avec les détections mises de côté (aire de l'intersection divisée par l'aire de l'union supérieure à un certain seuil,généralement 0,5 ou 0,6),
4. recommencer à l'étape 2 avec les détections restantes.

Le résultat de cet algorithme est disponible sur la Figure 2.8 (couplé à un seuillage des scores dans la Figure 2.8(b)).



(a) Détections avant d'appliquer une *NMS* [REDMON et FARHADI, 2016].

(b) Détections retenues après l'application d'une *NMS* [REDMON et FARHADI, 2016].

(c) Détections retenues avant et après l'application d'une *NMS* [PRISACARIU et al., 2009]

FIGURE 2.8 – Illustration de la *NMS*

Evaluation d'une détection

Pour évaluer la qualité d'une détection il faut à la fois évaluer la qualité de la classification ainsi que la qualité de la localisation.

Pour **la classification**, l'évaluation dépend du problème, mais dans le cas le plus courant nous cherchons à donner une classe parmi une liste de classes exclusives. L'évaluation est donc binaire : 1 si la classe est correctement attribuée, 0 si la classe est incorrecte. Nous utilisons donc les mêmes indices de mesure que la classification d'image. Une détection redondante est marquée comme fausse et influence le score de précision.

Pour **la localisation**, nous cherchons à mesurer le recoupement entre un patch proposé et le patch de la vérité terrain. Nous utilisons pour cela l'intersection divisée par l'union – ou *Intersection over Union* (IoU) –. L'IoU permet de mesurer l'intersection d'une proposition

tout en sanctionnant les débordements (quelques exemples de valeurs sont disponibles dans la Figure 2.9). Nous pouvons ensuite calculer l'IoU moyen sur l'ensemble des détections.

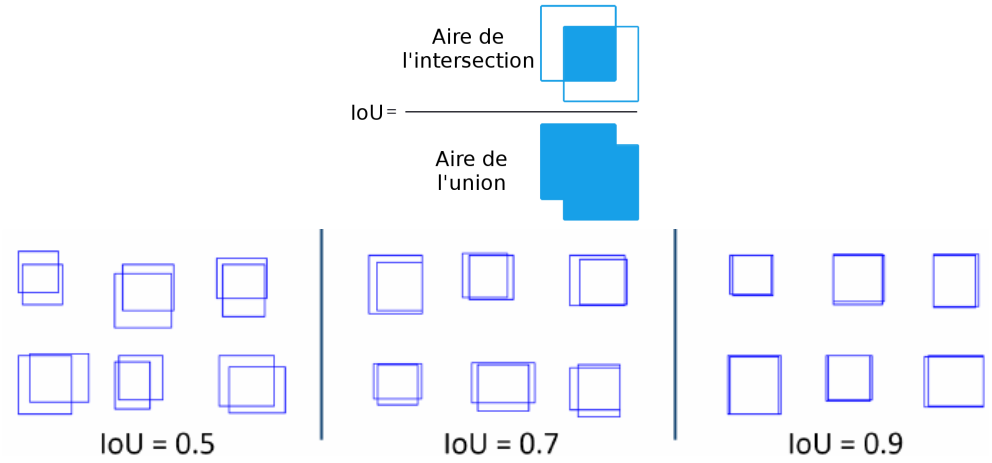


FIGURE 2.9 – Illustration de l'IoU

2.2.3 La segmentation sémantique

La segmentation est un processus qui vise à regrouper des pixels d'une image entre-eux selon un ou plusieurs critères. La segmentation sémantique regroupe les méthodes de segmentation qui attribuent *in fine* une étiquette de classe aux régions segmentées.

La segmentation sémantique 2D Le problème de segmentation sémantique est souvent étudié dans un cadre probabiliste relationnel. En effet, l'image est décomposée selon une certaine technique (par patch de formes quelconques, grille, pixel, détection de contour, points d'intérêts + expansion, ...) et ensuite les algorithmes utilisés tentent de définir des relations d'appartenance aux classes pour ces zones. Ces relations sont définies selon les caractéristiques propres à chaque zone de la décomposition (par des méthodes de classification) et/ou par contamination des voisins de façon itérative pour maximiser une fonction objective. LARLUS et al. [2010] proposent un exemple de méthode pour la segmentation sémantique.

La segmentation sémantique 3D La compréhension de la scène visuelle est une capacité clé pour permettre aux robots intelligents d'évoluer et d'interagir dans leur environnement. Après des décennies de progrès, nous avons atteint un point où l'acquisition de scènes 3D complexes détaillées est maintenant possible de plusieurs façons : des scanners laser précis, des capteurs rouge, vert, bleue, profondeur – ou *Red, Green, Blue, Depth map* (*RGB-D*) – et des techniques de reconstruction basé sur de le *SLAM*. Ces types de reconstruction sont suffisants pour une simple navigation et un évitement de collision, mais une nouvelle étape est nécessaire pour que le robot agisse à dessein : une analyse sémantique des données 3D.

En 3D, concevoir les caractéristiques les plus discriminantes pour l'entraînement d'un classificateur supervisé a longtemps été une approche standard. Par exemple, CHARANIYA et al. [2004] concatène des caractéristiques comme la hauteur normalisée ou la luminance. Des descripteurs génériques capables de représenter les points et leurs environs ont également été proposés : par exemple, les histogrammes des éléments de points rapides (*FPFH*) de RUSU et al. [2009] ou les histogrammes de signature de TOMBARI et al. [2010].

Nous détaillons la segmentation sémantique 2D et 3D par réseau de neurones dans le chapitre suivant : 3.

Évaluation d'une segmentation

Tout comme la classification et la détection, différentes métriques existent pour mesurer la qualité d'une segmentation.

Avec C l'ensemble des classes et X les points à classifier. Avec $X_c \in X$ les points classifiés avec l'étiquette $c \in C$. Nous notons X_c^* les points appartenant réellement à la classe c : la vérité terrain.

Précision Globale La Précision Globale (P) correspond à la proportion de points correctement labellisés.

$$P = \frac{1}{|X|} \sum_{c \in C} |X_c \cap X_c^*| \tag{2.3}$$

Cet indice ne prend pas en compte le problème de disproportion dans les classes présentes mais donne une bonne idée du comportement global du classifieur.

Précsion moyenne La précision moyenne (PM) est calculée avec les précisions de chaque classe.

$$PM = \frac{1}{|C|} \sum_{c \in C} \frac{|X_c \cap X_c^*|}{\#X_c^*} \tag{2.4}$$

Intersection sur l'Union moyenne (mIoU) Par rapport à la précision moyenne, la $mIoU$ pénalise également les mauvaises prédictions.

$$mIoU = \frac{1}{|C|} \sum_{c \in C} \frac{|X_c \cap X_c^*|}{|X_c \cup X_c^*|} \tag{2.5}$$

2.3 Les jeux de données

Comme indiquée dans l'introduction du chapitre, la communauté se base essentiellement sur des jeux de données disponibles pour se comparer sur les différentes tâches. Bien que nous n'ayons pas encore abordée l'aspect multi-modal de nos travaux nous présenterons également dans cette section les jeux de données *RGB-D* pour regrouper l'ensemble des jeux de données dans une seule et même partie.

2.3.1 Les jeux de données RGB

CIFAR10/CIFAR100 Cifar10 et Cifar100 sont des jeux de données historiques pour la tâche de classification. Ils consistent en un regroupement en 10 ou 100 classes de petites images de 32x32 pixels centrés sur l'objets d'étude. C'est un jeu léger en espace mémoire, homogène dans sa répartition inter-classe et très pratique pour l'expérimentation.

Disponible sur la page du jeu de données `https://www.cs.toronto.edu/~kriz/cifar.html` il y a 50 000 images d'entraînement et 10 000 images de test. Les images appartiennent toutes à des classes mutuellement exclusives (*airplane, automobile, bird, cat, deer, dog, frog, horse, ship, truck*). Quelques exemples sont disponibles dans l'image de la Figure 2.10.

FIGURE 2.10 – Exemples d'images du jeu de données CIFAR10 (extrait de http://cs231n.github.io/)

Caltech101/Caltech256 Les jeux de données Caltech101 et Caltech256 sont des jeux de données avec respectivement 101 et 256 classes d'images RGB de toutes résolutions. Ils sont téléchargeables aux adresses :

Caltech101 : http://www.vision.caltech.edu/Image_Datasets/Caltech101/

Caltech256 : http://www.vision.caltech.edu/Image_Datasets/Caltech256/

Des détails sur leur acquisition et leur contenu sont disponibles dans FEI-FEI et al. [2006] et GRIFFIN et al. [2007] et quelques exemples sont présentés dans la Figure 2.11.

FIGURE 2.11 – Exemples d'images du jeu de données Caltech101

ImageNet Le jeu de donnée ImageNet est le jeu de données 2D RGB de référence. Disponible à l'adresse http://www.image-net.org il offre plusieurs millions d'images de toutes tailles, de toutes sources, avec des milliers de classes d'objets. Le challenge classique de détection sur ce jeu de donnée est effectué sur un sous-ensembe d'un million d'images réparties en 1000 classes (environ 1000 images par classe). La sémantisation du jeu de données est ordonnée grâce à l'architecture WordNet [MILLER, 1995] où l'ensemble des mots sont reliés dans un graphe hiérarchique. Un exemple est disponible dans la Figure 2.12.

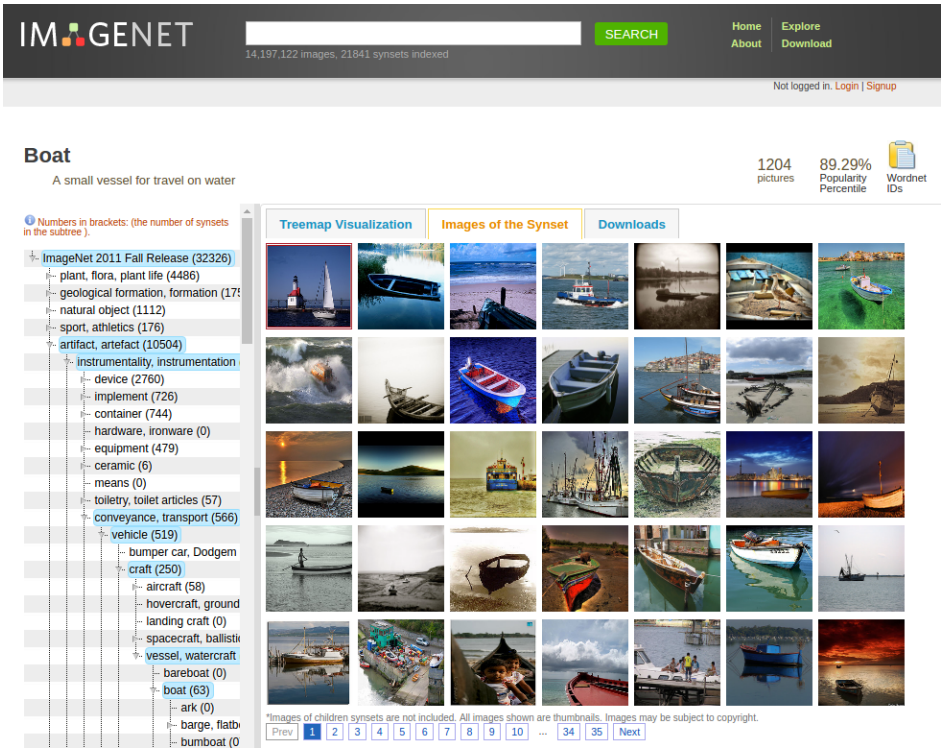


FIGURE 2.12 – Capture d'écran du site http://www.image-net.org/synset?wnid=n02858304 où *n02858304* correspond au wordnet "boat"⊂"vessel"⊂"craft"⊂"vehicle"⊂"transport"⊂"instrumentation"⊂"artefact"

Cette structure WordNet a d'ailleurs été utilisée par REDMON et FARHADI [2016] pour apprendre à classifier en plus des 1000 classes d'ImageNet des classes sans exemple RGB, seulement en utilisant les relations hiérarchiques entre les mots.

2.3.2 Les jeux de données *RGB-D*

Grâce à la Microsoft Kinect d'abord, puis à toutes les caméras *RGB* (Intel RealSense ou Asus Xtion pour n'en citer que quelques-unes), de nombreuses applications sont apparues dans la communauté robotique et surtout dans le domaine de la vision par ordinateur. En effet, le canal de profondeur (la *Depth*) donne accès à des informations géométriques 3D. Cette tendance a été permise par le faible coût de ces appareils et leur facilité d'utilisation.

Ces applications, pour la plupart, nécessitent une phase d'apprentissage, et nécessitent donc un ensemble de données d'entraînement. C'est dans ce contexte que plusieurs jeux de données, composés d'images *RGB-D*, ont été partagés afin de permettre à la communauté de tester et de comparer leurs méthodes.

Disponible avec des caractéristiques variées nous avons été particulièrement intéressés par cinq d'entre eux pour usage dans nos travaux.

RGB-D : les classiques

NYU Depth v2 Généralement appelé **NYUDv2** [SILBERMAN et al., 2012], ce fut l'un des premiers jeux de données *RGB-D*. Il est devenu une référence pour les données *RGB-D* comme MNIST [LECUN et al., 1998] l'est pour la classification d'images 2D. Il ne contient que 1449 images le rendant facile à utiliser. GUPTA et al. ont proposé une redéfinition des 800 classes originales en 40 macro-classes adoptées par tous. Il est disponible à l'adresse `http://cs.nyu.edu/~silberman/datasets/nyu_depth_v2.html` (exemple dans la Figure 2.13).

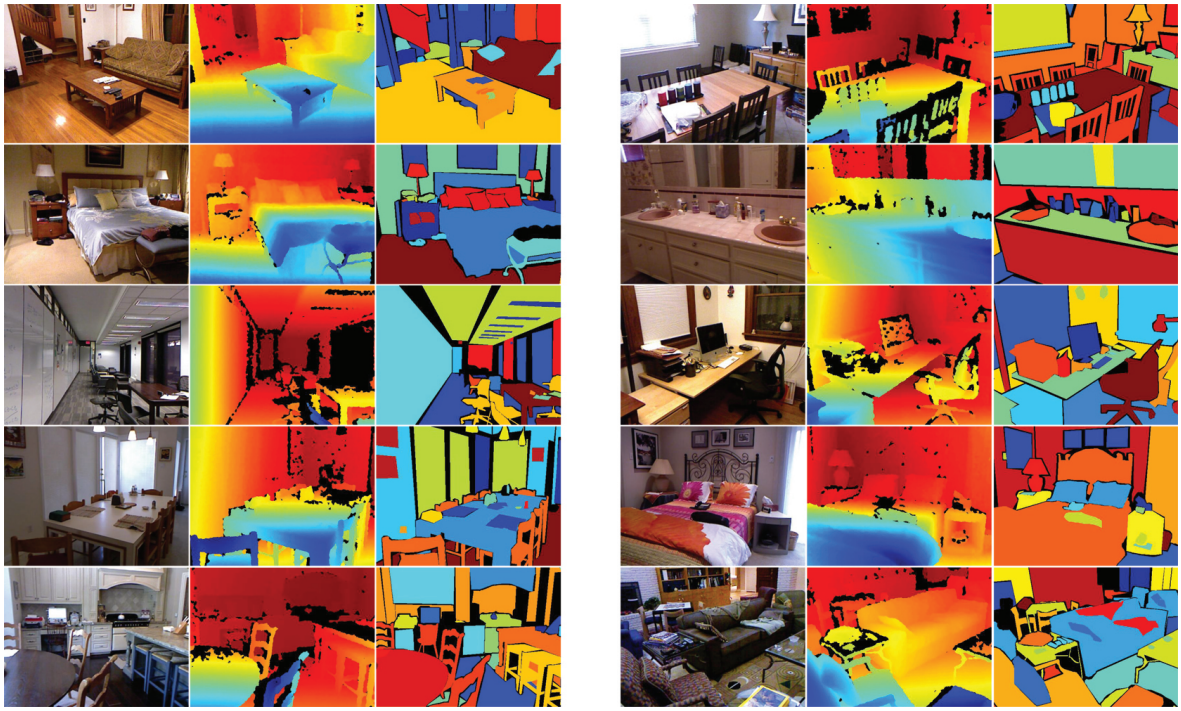


FIGURE 2.13 – Exemples du jeu de données NYUDv2 extrait du site de SILBERMAN et al. : `http://cs.nyu.edu/~silberman/datasets/nyu_depth_v2.html`

SUN3D Le jeu de donnée *RGB-D* **SUN3D** est proposé par l'université de Princeton [XIAO et al., 2013]. Il propose notamment des reconstructions 3D de nombreuses scènes pour effectuer diverses tâches comme la segmentation sémantique, de la détection où de l'estimation d'agencement. Nous utiliserons ce jeu de données dans le Chapitre 4 traitant de la sélection d'algorithmes. Les auteurs de *SUN3D* ont également proposé une interface web d'annotation visible dans la Figure 2.14.

SUNRGBD **Sun-RGBD** est un dataset acquis avec des capteurs *RGB-D* bas-coûts SONG et al. [2015]. Il est le résultat de l'accumulation de plusieurs jeux de données *RGB-D* :

- NYUDv2 SILBERMAN et al. [2012]
- Berkeley B3DO JANOCH et al. [2013]
- SUN3D XIAO et al. [2013]

Il atteint une taille de 10335 images. Complété avec différentes annotations 2D et 3D sur l'ensemble du jeu de données il peut être utilisé pour l'entraînement et l'évaluation d'algorithmes sur de nombreuses tâches comme la classification de scènes, la segmentation

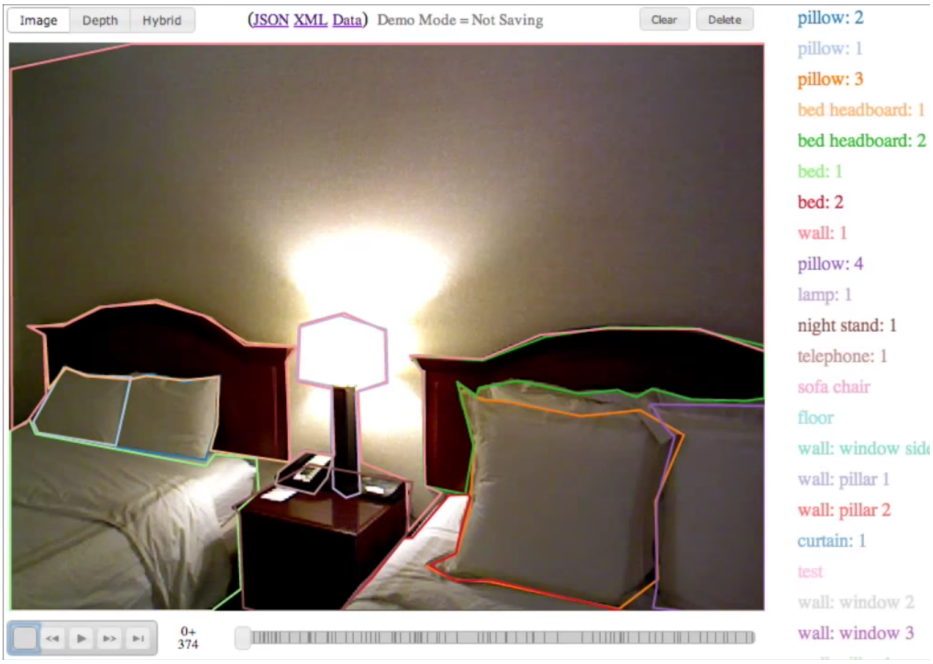


FIGURE 2.14 – Exemple du jeu de données SUN3D dans son interface d'annotation

sémantique (2D et 3D), l'estimation d'agencement ou la détection d'objets (2D et 3D). Il est composé de 37 classes (par exemple *table, bed, chairs, curtain, fridge, mirror, sink, floor, wall, etc.*).

RGB-D non-statiques

NYUDv2 et SUNRGBD sont néanmoins essentiellement tournés vers des classes statiques (mur, table, lit...) et chaque image étiquetée est indépendante des autres. Étant donné que dans le cadre de nos travaux nous abordons des scénarios de robotique mobile nous nous sommes également intéressés à des jeux de données avec séquences vidéos et des objets mobiles. En particulier les jeux de données qui proposent de la détection de personne.

RGBD People dataset C'est le cas de **RGBD People dataset** SPINELLO et ARRAS [2011] qui offre trois séquences vidéos d'un même panorama dans un hall d'université avec l'ensemble des personnes étiquetées et agrémentées d'un niveau d'occlusion. Ce jeu de données est utilisé dans le Chapitre "Approche multi-modale pour la détection" et un exemple est donné dans la Figure 2.15.

ONERA.ROOM Dans le cadre des scénarios exploratoires envisagés, nous prévoyons de faire face à de grandes variations de luminosité. Il est important de garder à l'esprit que les caméras *RGB-D* listées ci-dessus se réfèrent à des dispositifs dits actifs. Cela signifie que pour obtenir la carte de profondeur, les caméras recourent à l'émission d'un champ infrarouge particulier. Cette méthode offre l'avantage de fonctionner sans une source de lumière externe (dans l'obscurité donc) mais peut être fortement bruitée par le soleil (c'est-à-dire à l'extérieur). Au début de cette étude, aucun ensemble de données n'était disponible pour étudier l'influence de ces variations qui peuvent parfois être brutales. C'est pourquoi nous avons commencé à produire l'ensemble de données ONERA.ROOM que nous aborderons spéciquement dans le Chapitre "Approche multi-modale pour la détection".

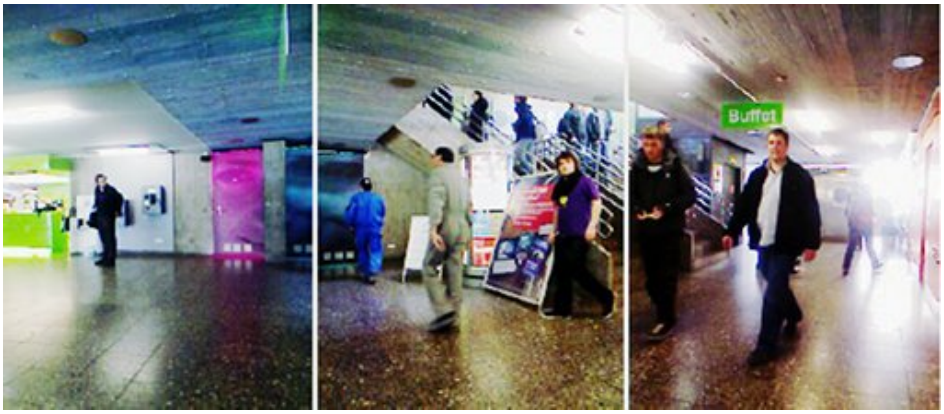


FIGURE 2.15 – Exemple du jeu de données *RGB-D* People extrait de http://www2.informatik.uni-freiburg.de/~spinello/RGBD-dataset.html

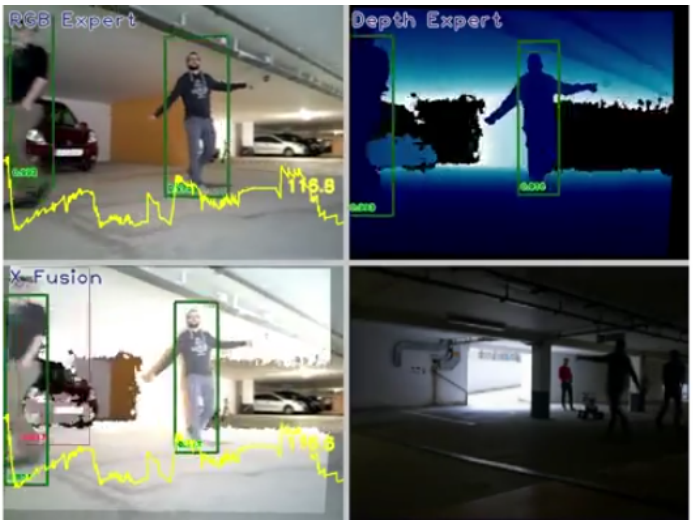


FIGURE 2.16 – Exemple du jeu de données ONERA.ROOM

InOutPeople RGBD dataset D'autres travaux ont également produit un jeu de données *RGB-D* complexe en 2016 : **InOutPeople** MEES et al. [2016]. Il offre des séquences vidéos avec des variations de luminosité, des situations en extérieur, du flou de bougé, du flou cinétique, le tout observé depuis un robot mobile terrestre. Toutes les personnes visibles dans une des modalités sont étiquetées. Nous utiliserons également ce jeu de données dans le Chapitre "Approche multi-modale pour la détection" et un exemple est donné dans la Figure 2.17.

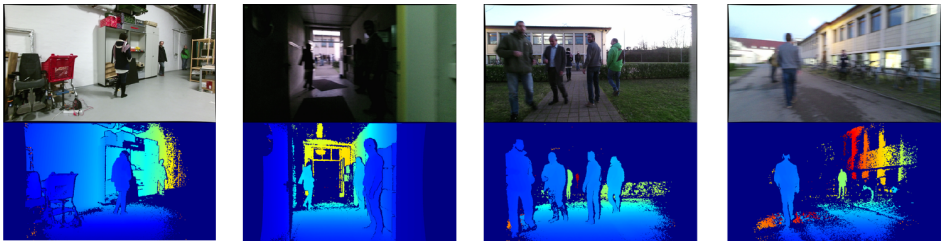


FIGURE 2.17 – Exemple du jeu de données *InOutdoor RGBD People* extrait de http://adaptivefusion.cs.uni-freiburg.de/#dataset

TABLEAU 2.1 – Comparaison des jeux de données *RGB-D*

Dataset	nb images annotées	résolution	capteur	nb classes	video disponible
NYU Depth v2	1449	640*480	Kinect 1 (portée à la main)	800+	oui (non labellisée)
SunRGBD	10,335	640*480	Kinect v1/v2, xtion, realsense	37	non
RGBD People	3000+	640*480	Kinect 1 (statique)	1 (personne)	oui
InOutPeople	8605	640*480	Kinect 2 (robot mobile)	1 (personne)	oui
ONERA.ROOM*	35379	640*480	Kinect v1 (robot mobile)	10+	oui

* Jeu de données développé par nos soins présenté dans le Chapitre "Approche multi-modale pour la détection".

Résumé Nous résumons l'ensemble de ces informations dans le Tableau 2.1. Pour obtenir des informations sur les autres jeux de données *RGB-D* disponibles nous orientons le lecteur vers FIRMAN [2016] qui compare les jeux en profondeur et propose d'excellentes perspectives d'évolution pour les jeux de données futurs.

2.4 Conclusion

Le domaine de la vision par ordinateur est un domaine riche qui travaille sur des données en grandes dimensions. Pour extraire de l'information de ces données, les chercheurs ont recours à des représentations intermédiaires, supervisées ou non, qui facilitent la prise de décisions. En plus de savoir traduire l'information (au sens de la projeter dans un espace de travail exploitable) il faut savoir définir des modèles reconnaissables.

Parmi les approches présentées nous retenons tout particulièrement l'approche par descripteurs *HOGs* couplée à des modélisations des classes sous forme de *Deformable Part Model* et une classification par *SVM* que nous utiliserons dans certaines de nos expérimentations.

Nous avons également présenté plusieurs jeux de données qui permettent à la communauté de comparer leurs méthodes et sur lesquelles nous évaluerons nos approches. Parmi eux se trouve *ImageNet* dont la complexité vient mettre en défaut les méthodes classiques de sémantisation. De même, les données *RGB-D* apportent une nouvelle dimension dans la source d'information qui complexifie le travail.

C'est pourquoi nous nous intéressons dans le Chapitre "État de l'art : Apprentissage profond" et dans nos travaux à des méthodes plus récentes à base de réseaux de neurones artificiels. Ces méthodes sont aujourd'hui appliquées dans de nombreux domaines (vision, langage naturel, traduction, régression...) et offrent des performances comparables voire supérieures à celles de l'Homme.

2.5 Références

ALEXE, B., T. DESELAERS et V. FERRARI. 2012, «Measuring the objectness of image windows», IEEE transactions on pattern analysis and machine intelligence, vol. 34, n° 11, p. 2189–2202. 16

ARBELÁEZ, P., J. PONT-TUSET, J. T. BARRON, F. MARQUES et J. MALIK. 2014, «Multiscale combinatorial grouping», dans Proceedings of the IEEE conference on computer vision and pattern recognition, p. 328–335. 17

BAY, H., T. TUYTELAARS et L. VAN GOOL. 2006, «Surf : Speeded up robust features», Computer vision–ECCV 2006, p. 404–417. 13

BELLMAN, R. 1961, Adaptive control process : a guided tour. 12

CARREIRA, J. et C. SMINCHISESCU. 2012, «Cpmc : Automatic object segmentation using constrained parametric min-cuts», IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 34, nº 7, p. 1312–1328. 17

CHARANIYA, A. P., R. MANDUCHI et S. K. LODHA. 2004, «Supervised parametric classification of aerial Lidar data», dans CVPR/W, IEEE, p. 30–30. 19

DALAL, N. et B. TRIGGS. 2005, «Histograms of oriented gradients for human detection», dans Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on, vol. 1, IEEE, p. 886–893. 13

ENDRES, I. et D. HOIEM. 2010, «Category independent object proposals», Computer Vision–ECCV 2010, p. 575–588. 16

EVERINGHAM, M., L. GOOL, C. K. WILLIAMS, J. WINN et A. ZISSERMAN. 2010, «The pascal visual object classes (voc) challenge», International journal of computer vision, vol. 88, nº 2. 15

FEI-FEI, L., R. FERGUS et P. PERONA. 2006, «One-shot learning of object categories», IEEE transactions on pattern analysis and machine intelligence, vol. 28, nº 4, p. 594–611. 21

FELZENSZWALB, P. F., R. B. GIRSHICK, D. MCALLESTER et D. RAMANAN. 2010, «Object detection with discriminatively trained part-based models», Pattern Analysis and Machine Intelligence, IEEE Transactions on. 13

FIRMAN, M. 2016, «Rgbd datasets : Past, present and future», dans Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops, p. 19–31. 26

FORGY, E. 1965, «Cluster analysis of multivariate data : efficiency versus interpretability models», Biometrics, vol. 61, nº 3, p. 768–769. 14

GRIFFIN, G., A. HOLUB et P. PERONA. 2007, «Caltech-256 object category dataset», . 21

GUPTA, S., R. GIRSHICK, P. ARBELÁEZ et J. MALIK. 2014, «Learning rich features from rgb-d images for object detection and segmentation», dans European Conference on Computer Vision, Springer, p. 345–360. 23

HARVEY, A. 2010, «Sliding window video», URL https://vimeo.com/12774628, [En ligne ; Page disponible le 10-septembre-2017]. 16

JANOCH, A., S. KARAYEV, Y. JIA, J. T. BARRON, M. FRITZ, K. SAENKO et T. DARRELL. 2013, «A category-level 3d object dataset : Putting the kinect to work», dans Consumer Depth Cameras for Computer Vision, Springer, p. 141–165. 23

JAPKOWICZ, N. et S. STEPHEN. 2002, «The class imbalance problem : A systematic study», Intelligent data analysis, vol. 6, nº 5, p. 429–449. 16

KALAL, Z., K. MIKOLAJCZYK et J. MATAS. 2012, «Tracking-learning-detection», Pattern Analysis and Machine Intelligence, IEEE Transactions on. 14

KULIS, B. et al.. 2013, «Metric learning : A survey», Foundations and Trends® in Machine Learning, vol. 5, nº 4, p. 287–364. 14

LARLUS, D., J. VERBEEK et F. JURIE. 2010, «Category level object segmentation by combining bag-of-words models with dirichlet processes and random fields», International Journal of Computer Vision, vol. 88, nº 2, p. 238–253. 19

LECUN, Y., L. BOTTOU, Y. BENGIO et P. HAFFNER. 1998, «Gradient-based learning applied to document recognition», Proc. of the IEEE, vol. 86, nº 11, p. 2278–2324. 23

LOWE, D. G. 1999, «Object recognition from local scale-invariant features», dans Computer vision, 1999. The proceedings of the seventh IEEE international conference on, vol. 2, Ieee, p. 1150–1157. 13

MEES, O., A. EITEL et W. BURGARD. 2016, «Choosing smartly : Adaptive multi-modal fusion for object detection in changing environments», dans Intelligent Robots and Systems (IROS), 2016 IEEE/RSJ International Conference on, IEEE, p. 151–156. 25

MILLER, G. A. 1995, «Wordnet : a lexical database for english», Communications of the ACM, vol. 38, nº 11, p. 39–41. 22

OJALA, T., M. PIETIKAINEN et T. MAENPAA. 2002, «Multiresolution gray-scale and rotation invariant texture classification with local binary patterns», IEEE Transactions on pattern analysis and machine intelligence, vol. 24, nº 7, p. 971–987. 13

PRISACARIU, V., I. REID et al.. 2009, «fasthog-a real-time gpu implementation of hog», Department of Engineering Science, vol. 2310, nº 9. 18

REDMON, J. et A. FARHADI. 2016, «Yolo9000 : better, faster, stronger», arXiv preprint arXiv :1612.08242. 18, 22

REN, S., K. HE, R. GIRSHICK et J. SUN. 2015, «Faster r-cnn : Towards real-time object detection with region proposal networks», dans Advances in neural information processing systems, p. 91–99. 17

RUSU, R. B., N. BLODOW et M. BEETZ. 2009, «Fast point feature histograms (fpfh) for 3d registration», dans Robotics and Automation, 2009. ICRA'09. IEEE International Conference on, IEEE, p. 3212–3217. 19

SANFOURCHE, M., B. LE SAUX, A. PLYER et G. LE BESNERAIS. 2014, «Cartographie et interprétation de l'environnement par drone», dans Colloque scientifique francophone Drones et moyens légers aéroportés d'observation. 14

SILBERMAN, N., D. HOIEM, P. KOHLI et R. FERGUS. 2012, «Indoor segmentation and support inference from rgbd images», Computer Vision–ECCV 2012, p. 746–760. 23

SONG, S., S. P. LICHTENBERG et J. XIAO. 2015, «SUN RGB-D : A RGB-D scene understanding benchmark suite», dans CVPR, Boston, USA, p. 567–576. 23

SPINELLO, L. et K. O. ARRAS. 2011, «People detection in rgb-d data», dans Intelligent Robots and Systems (IROS), 2011 IEEE/RSJ International Conference on, IEEE, p. 3838–3843. 24

TOMBARI, F., S. SALTI et L. DI STEFANO. 2010, «Unique signatures of histograms for local surface description», dans ECCV, Springer, Hersonissos, Crete, p. 356–369. 19

TURK, M. A. et A. P. PENTLAND. 1991, «Face recognition using eigenfaces», dans Computer Vision and Pattern Recognition, 1991. Proceedings CVPR'91., IEEE Computer Society Conference on, IEEE, p. 586–591. 13

UIJLINGS, J., K. VAN DE SANDE, T. GEVERS et A. SMEULDERS. 2013, «Selective search for object recognition», International journal of computer vision. 16

VIOLA, P., M. J. JONES et D. SNOW. 2005, «Detecting pedestrians using patterns of motion and appearance», International Journal of Computer Vision, vol. 63, n° 2, p. 153–161. 16

WANG, H., A. CRUZ-ROA, A. BASAVANHALLY, H. GILMORE, N. SHIH, M. FELDMAN, J. TOMASZEWSKI, F. GONZALEZ et A. MADABHUSHI. 2014, «Mitosis detection in breast cancer pathology images by combining handcrafted and convolutional neural network features», Journal of Medical Imaging, vol. 1, n° 3, p. 034 003–034 003. 13

XIAO, J., A. OWENS et A. TORRALBA. 2013, «SUN3D : A database of big spaces reconstructed using sfm and object labels», dans 2013 IEEE International Conference on Computer Vision, Sidney, Australia, p. 1625–1632. 23

XU, R. et D. WUNSCH. 2005, «Survey of clustering algorithms», IEEE Transactions on neural networks, vol. 16, n° 3, p. 645–678. 14

YANG, J., Y.-G. JIANG, A. G. HAUPTMANN et C.-W. NGO. 2007, «Evaluating bag-of-visual-words representations in scene classification», dans Proceedings of the international workshop on Workshop on multimedia information retrieval, ACM, p. 197–206. 13

Chapitre 3

État de l'art : Apprentissage profond

> “ *Features matter.* ”
>
> Ross Girshick, GUPTA et al., 2014

Sommaire

3.1 Introduction 32
3.1.1 Les neurones artificiels 32
3.1.2 Entraîner un réseau de neurones 33
3.2 Les réseaux de neurones convolutionnels 37
3.2.1 Les différentes couches d'un *CNN* 37
3.2.2 Les architectures de *CNNs* 40
3.3 La reconnaissance visuelle avec l'apprentissage profond 45
3.3.1 Classification par réseaux de neurones 45
3.3.2 Détection par réseaux de neurones 46
3.3.3 Segmentation sémantique par réseau de neurones 49
3.3.4 NMS par apprentissage profond 53
3.4 Les données multi-modales en apprentissage profond 53
3.5 Conclusion 56
3.6 Références 57

3.1 Introduction

L'apprentissage profond – ou *Deep Learning (DL)* – est une branche de l'apprentissage automatique. Remis au goût du jour par KRIZHEVSKY et al. en 2012, cette technologie traite des réseaux de neurones artificiel – ou *Artificial Neural Networks (ANNs)* –. Cette appellation vient d'un parallèle direct avec le fonctionnement des neurones biologiques. En effet, un neurone biologique, dans une version simplifiée, est connecté à la sortie d'autres neurones. Ces neurones d'entrée transmettent tous leur potentiel d'activation au neurone de sortie. Si le potentiel total est suffisant alors le neurone de sortie s'active également, changeant son potentiel d'action. Auparavant, ce domaine était intitulé le connectionisme [LECUN, 1987].

3.1.1 Les neurones artificiels

Le perceptron

Le modèle connexionniste le plus basique est le perceptron, composé d'un ou plusieurs neurones artificiels (voir Figure 3.1).

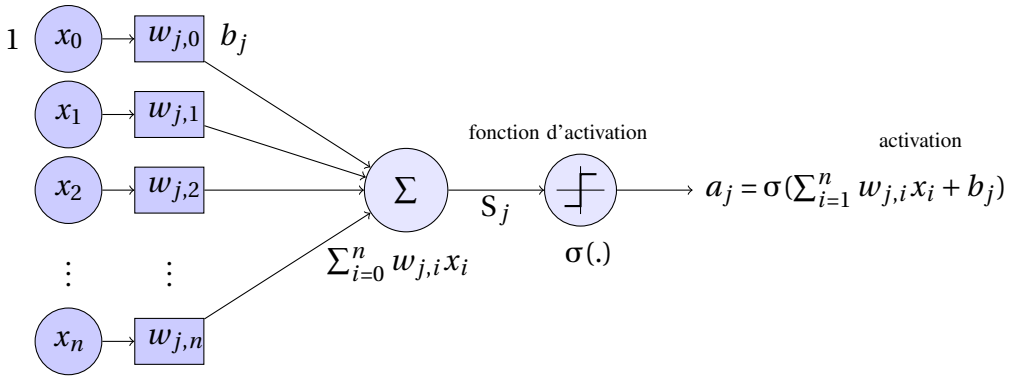


FIGURE 3.1 – Schéma de fonctionnement d'un neurone artificiel de perceptron.

Le perceptron calcule les activations a_j de ces neurones, c'est-à-dire pour un neurone j et un vecteur d'entrée x de taille n, il effectue la somme pondérée par n poids $w_{j,i\in[\![1:n]\!]}$, ajoute un biais b_j et applique une fonction d'activation σ :

$$a_j = \sigma(\sum_{i=1}^{n} w_{j,i}x_i + b_j) \tag{3.1}$$

Perceptron Multi-Couche

Le vecteur des N_0 activations de la couche 0, $A_0 = \{a_j\}_{j\in[\![1:N_0]\!]}$, peut ensuite être considéré comme le vecteur d'entrée d'un nouveau perceptron. Cette accumulation de perceptrons forme ce que l'on appelle un perceptron multi-couches – ou *Multi Layer Perceptron (MLP)* – (voir Figure 3.2).

Cette structure par couche donne ainsi tout son sens à l'apprentissage "profond" dans le sens où un réseau peut se définir par sa profondeur (c'est-à-dire le nombre de couches).

Approximation universelle

En 1989, GYBENKO a montré qu'un perceptron multi-couche pouvait approximer de nombreuses fonctions dans $\mathbb{R}^m$ en proposant le théorème suivant :

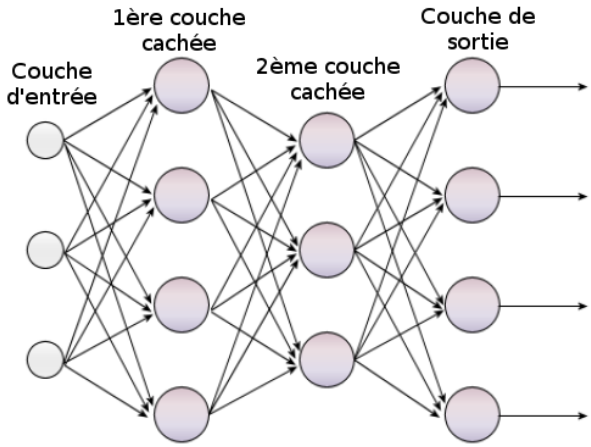


FIGURE 3.2 – Schéma d'un perceptron multi-couche WIKIPEDIA [2017]

.

Théorème 1. *Soit* ϕ *une fonction non constante, bornée, continue, et croissante. Soit* I_m *l'hypercube unitaire de dimension* m *:* $[0,1]^m$*. L'espace des fonctions continues dans* I_m *est dénoté par* $C(I_m)$*. Alors, pour tout* $\epsilon > 0$*, il existe un entier* N *tel que pour toute fonction* $f \in C(I_m)$ *il existe des constantes réelles* $v_i, b_i \in \mathbb{R}$ *et des vecteurs réels* $w_i \in \mathbb{R}^m$*, où* $i = 1, ..., N$ *tel que nous pouvons définir :*

$$F(x) = \sum_{i=1}^{N} v_i \phi(w_i^T x + b_i) \tag{3.2}$$

comme une approximation de f*, autrement dit :*

$$|F(x) - f(x)| < \epsilon \tag{3.3}$$

pour tout $x \in I_m$*.*

Cependant cette étude n'a pas démontré qu'il était possible de concevoir des algorithmes pour apprendre ces paramètres. Il existe donc un vide théorique sur la capacité de nos algorithmes à entraîner un réseau de neurones. Nous y reviendrons dans la Sous-section 3.1.2.

3.1.2 Entraîner un réseau de neurones

La descente de gradient stochastique

L'entraînement d'un réseau de neurones consiste à minimiser son erreur de prédiction. Pour cela on définit $\mathscr{L}$ une fonction de perte –ou *Loss function*– entre la vérité terrain et la prédiction. Cette fonction de perte dépend de tous les paramètres $\theta = \{w_{j,i}, b_j\}$ du réseau de neurones. Si nous notons F l'opération du réseau de neurones, x l'échantillon et y l'étiquette associée, alors nous cherchons :

$$\theta^* = \arg\min_{\theta} \Big(\sum_{\forall (x,y)} \mathscr{L}(x, y, \theta) \Big) \tag{3.4}$$

Cependant, optimiser une somme de plusieurs milliers de termes avec plusieurs millions de paramètres (typiquement le cas d'un réseau de neurones de type VGG [SIMONYAN et ZISSERMAN, 2014] sur le jeu de données ImageNet [DENG et al., 2009] et ses 1000 classes) est relativement complexe. Il faut utiliser ce qui est appelé un M-estimateur [HUBER, 1981]

(pour *Maximum-likelihood type estimator*), qui nécessite de connaître la dérivée de $\mathscr{L}$ par rapport à l'entrée x. De part la profondeur des réseaux de neurones et la non-linéarité de leurs fonctions d'activation, obtenir cette dérivée partielle est un travail fastidieux, si ce n'est impossible. Il nous est cependant possible de définir l'erreur d'une couche donnée en fonction de l'erreur de la couche suivante et de partir ainsi de l'erreur de la couche finale, jusqu'à la couche initiale.

Démonstration :

Pour une couche (*layer*) indicée l, notons $o_{l,j}$ la sortie (*output*) du neurone j, $n_{l,j}$. Et notons $\mathrm{S}_{l,j}$ la somme pondérée des sorties de la couche $l-1$ avec les poids $w_{l,j,i}$ du neurone $n_{l,j}$.

$$o_{l,j} = \sigma(\mathrm{S}_{l,j}) = \sigma(\sum_k w_{l,j,k} o_{l-1,k}), \text{ où } \sigma \text{ désigne la fonction d'activation.} \tag{3.5}$$

En exprimant la dérivée partielle de la fonction de perte par rapport à un paramètre $w_{l,j,i}$ nous obtenons :

$$\begin{aligned} \frac{\partial \mathscr{L}}{\partial w_{l,j,i}} &= \frac{\partial \mathscr{L}}{\partial o_{l,j}} \frac{\partial o_{l,j}}{\partial w_{l,j,i}} \\ &= \frac{\partial \mathscr{L}}{\partial o_{l,j}} \frac{\partial o_{l,j}}{\partial \mathrm{S}_{l,j}} \frac{\partial \mathrm{S}_{l,j}}{\partial w_{l,j,i}} \end{aligned} \tag{3.6}$$

Développons chaque terme ci-dessous :

$$\begin{aligned} \frac{\partial \mathscr{L}}{\partial o_{l,j}} &= \sum_{m \in \mathrm{M}} \left(\frac{\partial \mathscr{L}}{\partial \mathrm{S}_{l+1,m}} \frac{\partial \mathrm{S}_{l+1,m}}{\partial o_{l,j}}\right) \\ &= \sum_{m \in \mathrm{M}} \left(\frac{\partial \mathscr{L}}{\partial o_{l+1,m}} \frac{\partial o_{l+1,m}}{\partial \mathrm{S}_{l+1,m}} \frac{\partial \mathrm{S}_{l+1,m}}{\partial o_{l,j}}\right) \\ &= \sum_{m \in \mathrm{M}} \left(\frac{\partial \mathscr{L}}{\partial o_{l+1,m}} \frac{\partial \sigma(\mathrm{S}_{l+1,m})}{\partial \mathrm{S}_{l+1,m}} \frac{\partial \mathrm{S}_{l+1,m}}{\partial o_{l,j}}\right) \\ &= \sum_{m \in \mathrm{M}} \left(\frac{\partial \mathscr{L}}{\partial o_{l+1,m}} \sigma'(\mathrm{S}_{l+1,m}) \frac{\partial \mathrm{S}_{l+1,m}}{\partial o_{l,j}}\right) \\ &= \sum_{m \in \mathrm{M}} \left(\frac{\partial \mathscr{L}}{\partial o_{l+1,m}} \sigma'(\mathrm{S}_{l+1,m}) \frac{\partial \sum_k (w_{l+1,m,k} o_{l,k})}{\partial o_{l,j}}\right) \\ &= \sum_{m \in \mathrm{M}} \left(\frac{\partial \mathscr{L}}{\partial o_{l+1,m}} \sigma'(\mathrm{S}_{l+1,m}) \sum_k \left(w_{l+1,m,k} \frac{\partial o_{l,k}}{\partial o_{l,j}}\right)\right) \\ &= \sum_{m \in \mathrm{M}} \left(\frac{\partial \mathscr{L}}{\partial o_{l+1,m}} \sigma'(\mathrm{S}_{l+1,m}) \sum_k (w_{l+1,m,k} \delta_{k,j})\right) \\ &= \sum_{m \in \mathrm{M}} \left(\frac{\partial \mathscr{L}}{\partial o_{l+1,m}} \sigma'(\mathrm{S}_{l+1,m}) w_{l+1,m,j}\right) \end{aligned} \tag{3.7}$$

$$\begin{aligned} \frac{\partial o_{l,j}}{\partial \mathrm{S}_{l,j}} &= \frac{\partial \sigma(\mathrm{S}_{l,j})}{\partial \mathrm{S}_{l,j}} \\ &= \sigma'(\mathrm{S}_{l,j}) \end{aligned} \tag{3.8}$$

$$\begin{aligned}\frac{\partial S_{l,j}}{\partial w_{l,j,i}} &= \frac{\partial \sum_k w_{l,j,k} o_{l-1,k}}{\partial w_{l,j,i}} \\ &= \sum_k \frac{\partial w_{l,j,k}}{\partial w_{l,j,i}} o_{l-1,k} \\ &= \sum_k \delta_{k,i} o_{l-1,k} \\ &= o_{l-1,i} \end{aligned} \tag{3.9}$$

avec M qui désigne l'ensemble des neurones de la couche $l+1$ qui utilisent la sortie du neurone $n_{l,j}$, et $\delta_{i,j}$ désigne le symbole de Kronecker (vaut 1 si i=j, 0 sinon).

Ainsi, en injectant les équations 3.7, 3.8 et 3.9 dans l'équation 3.6 nous obtenons :

$$\frac{\partial \mathscr{L}}{\partial w_{l,j,i}} = \sum_{m \in M} \left(\frac{\partial \mathscr{L}}{\partial o_{l+1,m}} \sigma'(S_{l+1,m}) w_{l+1,m,j}\right) \sigma'(S_{l,j}) o_{l-1,i} \tag{3.10}$$

L'équation 3.10 nous permet de définir de façon itérative le gradient de l'erreur sur chaque paramètre $w_{l,j,i}$, couche par couche, en commençant par la couche finale pour laquelle nous avons l'erreur à disposition.

Ce processus de rétropropagation du gradient est couplé à un formalisme stochastique nous permettant de traiter les exemples d'entraînements un par un ou par petits ensembles. Nous parlons alors de descente de gradient stochastique – ou *Stochastic Gradient Descent (SGD)* –. Nous pouvons ainsi mettre les poids à jour avec la formule suivante :

$$\Delta w_{l,j,i} = -\eta \frac{\partial \mathscr{L}}{\partial w_{l,j,i}} \tag{3.11}$$

$$w_{l,j,i} = w_{l,j,i} + \Delta w_{l,j,i} = w_{l,j,i} - \eta \frac{\partial \mathscr{L}}{\partial w_{l,j,i}} \tag{3.12}$$

où η désigne le taux d'apprentissage.

Le taux d'apprentissage permet de pondérer la modification des paramètres. Généralement initialisé à 0,001 [1] il permet de converger vers un certain état de stabilité dans les valeurs des paramètres puis peut-être réduit pour affiner l'apprentissage. L'équation 3.12 peut-être vue comme un "correcteur proportionnel" à l'erreur du paramètre et η le gain du correcteur proportionnel. En poursuivant l'analogie d'un correcteur, il est possible d'ajouter un "correcteur dérivé" [RUMELHART et al., 1988] en ajoutant un terme d'inertie à la mise à jour :

$$\Delta w_{l,j,i} = -\eta \frac{\partial \mathscr{L}}{\partial w_{l,j,i}} + \alpha \Delta w_{l,j,i} \tag{3.13}$$

$$w_{l,j,i} = w_{l,j,i} + \Delta w_{l,j,i} = w_{l,j,i} - \eta \frac{\partial \mathscr{L}}{\partial w_{l,j,i}} + \alpha \Delta w_{l,j,i} \tag{3.14}$$

Le concept de rétropropagation du gradient date de 1960. Il fut utilisé en premier au service des réseaux de neurones par RUMELHART et al. [1988] et LECUN et al. [1998]. La *SGD* avec *momentum* est aujourd'hui l'algorithme le plus utilisé pour l'entraînement de réseaux de neurones même s'il existe d'autres méthodes (qui utilisent notamment les dérivées partielles

1. Le choix de la valeur d'initialisation du taux d'apprentissage est le résultat de l'expérience acquise sur des entraînements qui ont convergé.

de 2ème ordre de l'erreur). La *SGD* comporte des inconvénients notables : comme illustré sur la Figure 3.3, deux points de départs initiaux peuvent nous mener dans deux minimums locaux différents dans l'exemple donné. Cependant les travaux de KAWAGUCHI [2016] ont montré récemment (voir leur théorème 2.3) que les minimums locaux de la fonction d'erreur sont des minimums globaux et que tout point critique (c'est-à-dire dont le gradient est nul) est un point selle.

En ce qui concerne la fonction de perte elle-même il est possible d'utiliser différentes fonctions que nous aborderons dans les différentes tâches de la vision par ordinateur : sous-sections "Classification par réseaux de neurones", "Détection par réseaux de neurones" et "Segmentation sémantique par réseau de neurones")

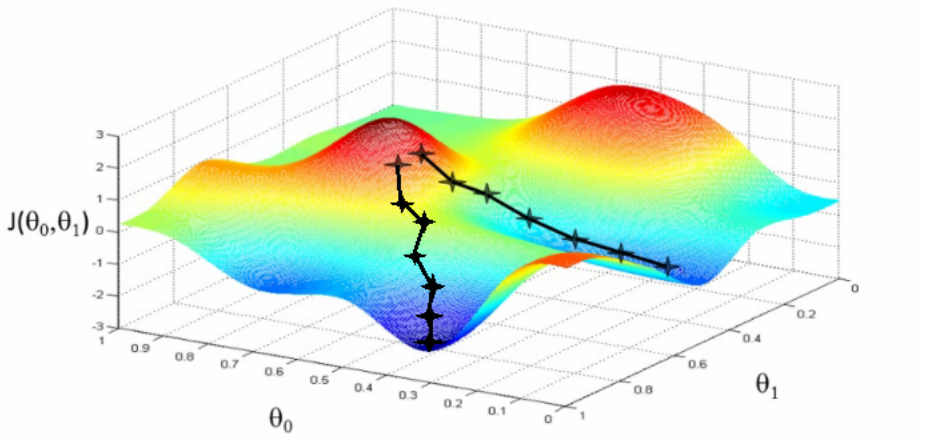


FIGURE 3.3 – Illustration de la descente de gradient sur une fonction de potentiel J arbitraire qui dépend de deux variables, Θ_0 et Θ_1

.

Le sur-apprentissage

Lorsqu'on entraîne un réseau de neurones nous séparons systématiquement les données d'entraînement des données de test. Cette séparation a pour objectif de simuler une situation réelle, c'est-à-dire d'avoir des données étiquetées pour l'entraînement et des données inconnues pour le test. De fait, il existe une différence statistique entre les données d'entraînement et les données de test. Idéalement cette différence est minime, si les exemples d'entraînement sont suffisamment nombreux et variés pour englober la distribution de l'ensemble de test. Néanmoins, la différence de domaine entre les deux partitions de données peut entraîner un problème de sur-apprentissage. En effet, le modèle entraîné peut se sur-spécialiser sur les données d'entraînement et ne plus être capable de généraliser pour inférer sur les données de test.

Pour corriger ce problème majeur de l'apprentissage profond il existe différentes solutions :

— arrêter l'entrainement avant convergence stricte,

— appliquer des méthodes d'élagage –suppression des paramètres d'influence négligeable– au travers de la méthode *Optimal brain damage* (*OBD*) LECUN et al. [1990] ou *Optimal brain surgeon* (*OBS*) HASSIBI et STORK [1993],

— retoucher au taux d'apprentissage η en le rendant plus grossier après une phase de convergence ("*Fine to Coarse*"),

— dégrader les poids (*weight decay*) en ajoutant un terme de régularisation dans la fonction de perte ($\lambda \sum_i w_i$) qui empêche les poids d'avoir chacun une trop grande valeur absolue,
— utiliser certaines couches spécifiques (*pooling*, *batch normalization* – ou Normalisation du batch (*BN*) – : voir Sous-section 3.2.1),
— décimer aléatoirement les poids à l'entraînement et au test (*dropout*) pour forcer de la redondance dans les paramètres,
— faire de l'augmentation des données (*data augmentation*) en modifiant légèrement les données d'entrée (ajouter du bruit, inverser horizontalement, déformer).

3.2 Les réseaux de neurones convolutionnels

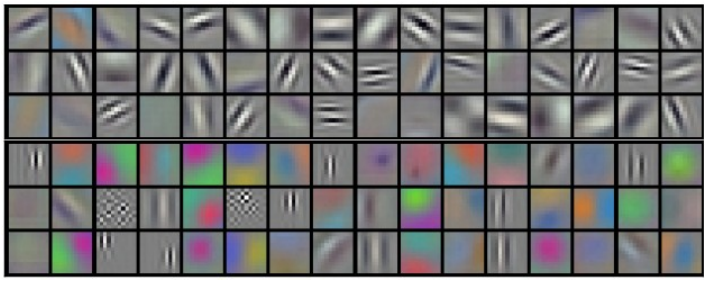


FIGURE 3.4 – Exemples de filtres de convolution : les 96 filtres de la première couche d'AlexNet [KRIZHEVSKY et HINTON, 2009].

Les réseaux de neurones convolutionnels – ou *Convolutional Neural Networks* (*CNNs*) –, à l'instar des perceptrons multi-couches, possèdent des neurones qui calculent une activation en fonction de l'entrée. Cependant, les *CNNs* ont vocation à travailler sur des images, qui plus est, des images complexes, résolues, en couleurs, *etc*. Il existe donc des *a priori* géométriques sur l'information disponible. C'est pourquoi nous contraignons les opérations de chaque neurone à de petites convolutions localisées. Cette opération est illustrée dans la Figure 3.5.

3.2.1 Les différentes couches d'un *CNN*

La couche convolutionnelle Une seule convolution permet d'obtenir toute une carte d'activation de la même dimension que l'image d'entrée, pour laquelle il aurait fallut $w * h * d$ neurones avec chacun $w * h * d$ connections dans un perceptron multi-couche. En effet, une convolution, par exemple de taille $(w = 3, h = 3, d = 3)$ peut être appliqué partout dans l'image, à chaque pixel. Une convolution ne possède pas autant de degrés de liberté qu'un neurone de perceptron, cependant elle bénéficie immédiatement de l'aspect géométrique de l'image, simplifiant ainsi sa tâche. En effet, les connexions des perceptrons ne prennent pas en compte l'ordre des pixels ni leurs positions relatives. La convolution s'appliquant partout dans l'image elle permet pour une carte d'activation d'avoir une invariance du traitement par translation. Ce qui ouvre la voie à la tâche de détection et améliore la robustesse en classification. Nous pouvons de plus appliquer autant de convolutions que désiré et obtenir plusieurs cartes d'activations pour une même entrée. C'est le cas du réseau de neurones AlexNet [KRIZHEVSKY et HINTON, 2009] qui applique sur l'image d'entrée 96 filtres de convolution de taille $(w = 11, h = 11, d = 3)$ (voir Figure 3.4) Nous reparlerons des structures de réseaux de neurones dans la sous-section 3.2.2.

Cette approche par filtres convolutionnels constitue la brique de base d'un *CNN* : nous parlons de couche convolutionnelle. La première couche de tout *CNN* permet de passer d'un

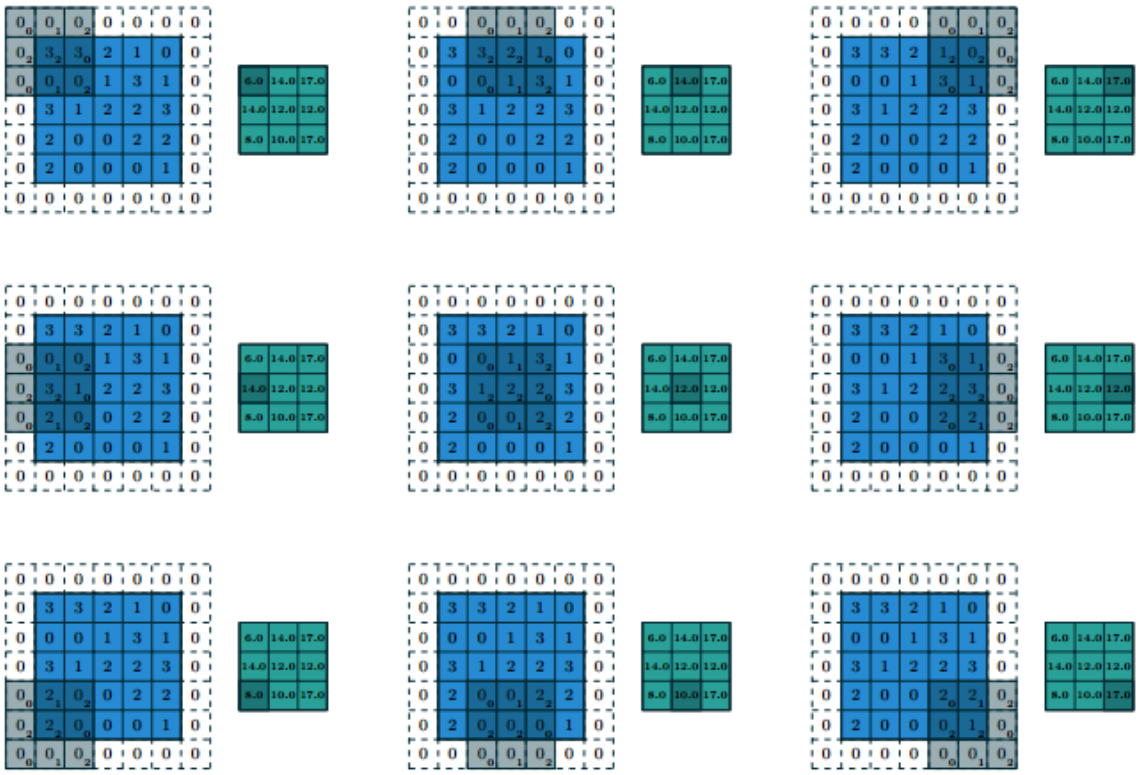


FIGURE 3.5 – Illustration de l'opération de convolution. Le noyau de convolution est ici de taille 3x3 pixels et s'applique partout sur l'image. L'usage des *GPU* en permet une implémentation optimisée qui rend l'opération extrêmement rapide. L'image est extraire de http://deeplearning.net/software/theano/tutorial/conv_arithmetic.html

tenseur 3D de profondeur 1 ou 3 (image en nuance de gris ou en RGB) à un tenseur 3D de profondeur égale au nombre de filtres convolutionnels. En effet, la profondeur d=3 (pour chaque canal de couleur) est particulièrement appréciée pour le stockage d'images numériques étant donné que cela correspond au fonctionnement de l'œil humain. Cependant, d'autres canaux peuvent être privilégiés comme le fait la quadrichromie et ses quatre canaux *CMYK* (*cyan, magenta, yellow, key*) idéaux pour les périphériques d'impression ou d'affichage. De la même façon, nous pouvons voir un *CNN* comme un enchaînement de traitements qui permettent de changer la représentation des données. Chaque couche de convolutions projette le tenseur 3D d'entrée dans un nouvel espace. Ce genre d'opération est linéaire et la composition d'opérations linéaires est assimilable à une seule opération linéaire. C'est tout l'intérêt de la fonction d'activation σ qui est, elle, non linéaire, et qui a lieu sur chaque carte d'activation, rendant le processus entier complètement non-linéaire.

Cependant, si l'espace d'origine nécessite des noyaux de convolutions avec une profondeur de 3, dans le cas du AlexNet [KRIZHEVSKY et HINTON, 2009], les noyaux de convolution de la deuxième couche nécessitent une profondeur de 96. Dans le cas du ResNet101 HE et al. [2016a] les noyaux de convolution atteignent même une profondeur de 2048. Cela a plusieurs conséquences directes : il faut stocker en mémoire ces paramètres et les valeurs des tenseurs 3D calculés à chaque couche lors de la phase d'entraînement, les couches suivantes doivent lire dans un espace de grande dimension, les GPU doivent être capables de gérer cette charge mémoire, le nombre d'exemples nécessaire pour l'entraînement augmente. Ceci a une influence sur la taille maximale du batch qui peut-être gérée par le réseau sur une architecture matérielle donnée.

Le *pooling* Malheureusement le stockage des tenseurs intermédiaires nécessaires lors de la phase d'apprentissage pour la rétro-propagation du gradient est une véritable problématique pratique. De plus, les filtres convolutionnels font ressortir des zones de forte activation qui sont mélangées aux zones faiblement activées. C'est dans ce cadre qu'a été introduit l'opération de *pooling*, utilisée notamment par CIREŞAN et al. [2011]; KRIZHEVSKY et HINTON [2009]; LECUN et al. [1998]. Elle consiste à choisir un représentant d'une zone spatiale en fonction d'un certain critère. Dans sa version la plus utilisée –le *max-pooling*– l'opération prend la valeur maximale d'une zone spatiale de taille 2x2. Cette opération a pour conséquence directe de diminuer la dimension spatiale du tenseur par 2 (voir illustration dans la Figure 3.6). C'est un sous-échantillonnage sélectif. Une conséquence directe du sous-échantillonnage est la perte d'information. Cependant cette perte d'information reste minime car nous gardons le terme le plus activé par rapport au filtre précédent. De plus, cette dégradation de la source permet d'éviter le surapprentissage –ou *over-fitting*.

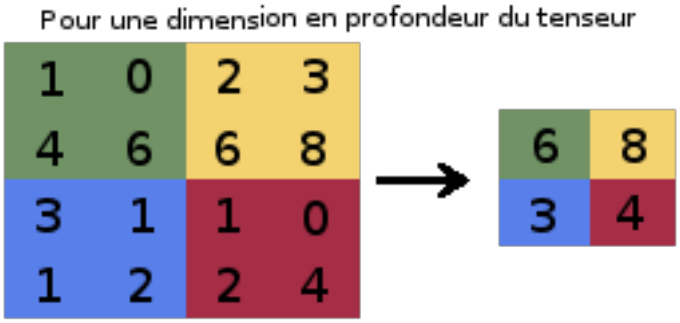


FIGURE 3.6 – Max pooling avec un filtre 2x2 et un pas (*stride*) de 2. WIKIPEDIA [2017]

La normalisation des batchs Nous commençons par définir quelques termes utiles à la compréhension des notions à venir :

Batch : Anglicisme pour "paquet, regroupement, série". Le terme *batch* désigne un regroupement, sous entendu de tenseurs. En pratique les images constituantes d'un batch (tenseurs 3D) sont concaténées dans un tenseur selon une nouvelle dimension (tenseur 4D). Les bibliothèques de *Deep Learning* sont généralement optimisées pour traiter directement des batchs ce qui permet d'augmenter la vitesse de traitement (à l'entraînement comme au test). Un avantage direct concerne l'apprentissage stochastique : la distribution des données observée à chaque itération est ainsi plus homogène que si un seul exemple avait été utilisé.

Batch normalisation : *batch normalization* – ou normalisation du batch (BN) – est une méthode pour normaliser la distribution statistique des données d'un batch. Cette opération peut-être appliquée sur chaque tenseur 4D intermédiaire entre les différentes couches d'un *CNN*. Introduite par IOFFE et SZEGEDY [2015], elle résulte du constat que chaque itération peut présenter un batch avec une distribution très différente de la précédente itération et ainsi rendre l'entraînement plus complexe. La *BN* permet ainsi d'homogénéiser chaque *batch*, d'accélérer l'entraînement, de dépasser les performances d'un modéle similaire sans *BN*, d'être moins sensible aux paramètres initiaux et de commencer l'entraînement avec un taux d'apprentissage plus élevé.

Elle présente cependant un inconvénient relatif à la puissance de calcul : il est nécessaire d'entraîner le réseau de neurones avec des batchs constitués de plusieurs images pour que la normalisation soit efficace. En pratique, nous cherchons à utiliser des batchs de taille 8,

16, 32, ..., 256, ce qui permet bien d'utiliser la *BN*. Cependant, certains modèles trop lourds pour les moyens mémoires à disposition ne permettent que des batchs de taille 1. Dans cette situation la *BN* n'est pas recommandée.

En effet, supposons que x_j fasse référence au batch de taille m en entrée de l'opération d'indice $_j$. La variable x_0 correspond donc au tenseur d'entrée du réseau de neurones : $\dim(x_0) = (m, c, h, w)$. Au moment de l'entraînement la *BN* remplace le tenseur d'activation x_j par sa normalisation $\hat{x}_j$ repondérée : $y_j = \mathrm{BN}(x_j)$:

$$\mu_j \leftarrow \frac{1}{m}\sum_{i=1}^{m} x_{j,i} \tag{3.15}$$

$$\sigma_j^2 \leftarrow \frac{1}{m}\sum_{i=1}^{m}(x_{j,i} - \mu_j)^2 \tag{3.16}$$

$$\hat{x}_{j,i} \leftarrow \frac{x_{j,i} - \mu_j}{\sqrt{\sigma_j^2 + \epsilon}} \tag{3.17}$$

$$y_{j,i} = \mathrm{BN}(x_{j,i}, \gamma_j, \beta_j) \equiv \gamma_j \hat{x}_{j,i} + \beta_j \tag{3.18}$$

ϵ permet d'assurer une certaine stabilité numérique en évitant de mettre le dénominateur à 0. γ_j et β_j sont des paramètres de la *BN* appris par *SGD* comme les paramètres d'une convolution.

Au moment du test il faut remplacer la moyenne et la variance du batch (μ_j, σ_j^2) par la moyenne et la variance de toutes les données d'entraînement $(\mathrm{E}[x_j], \mathrm{Var}[x_j])$. Cependant cette moyenne et cette variance globale change du fait de l'entraînement. C'est pourquoi elles sont calculés par moyenne glissante avec (μ_j, σ_j^2). Cette moyenne glissante ne serait pas stable dans le cas d'un batch de taille trop faible car ce batch ne serait pas suffisamment représentatif de la distribution globale des données.

La couche de produit matriciel ou "*fully-connected*" Une couche qui effectue un produit matriciel avec son tenseur d'entrée est appelée une couche *fully-connected* car chaque composante de la sortie est la somme pondérée des paramètres d'une ligne de la matrice de poids avec l'entrée, chaque composante est donc entièrement connectée au tenseur d'entrée. Cette couche est directement comparable à un perceptron où chaque neurone correspond à une composante de la sortie. Il est important de voir que cette opération est équivalente à un changement de base entre le tenseur d'entrée et le tenseur de sortie. La base de sortie n'ayant pas forcément le même nombre de paramètres que la base d'entrée, cette couche peut être assimilée à une opération de projection. C'est généralement la dernière couche d'un *CNN* de classification, projetant l'avant dernier tenseur dans un espace de taille égale au nombre de classes.

3.2.2 Les architectures de *CNNs*

Nous ne parlerons ici que des architectures pour la classification par réseau de neurones convolutionnels. Les architectures pour les autres tâches seront abordées dans les sections correspondantes.

Les architectures classiques

Les *CNNs* ont démontré aujourd'hui leur grande capacité à résoudre des problèmes de classification sémantique (He et al. [2016a]; Iandola et al. [2016]; Simonyan et Zisserman

[2014]; ZHAO et al. [2016], *etc.*). Cette branche de l'apprentissage automatique bénéficie du développement récent des unités de traitement graphique – ou *Graphical Processing Units (GPUs)* – et de leur utilisation massive de calculs parallèles qui réduisent drastiquement le temps requis pour l'entraînement et le test.

A la suite de l'architecture **LeNet** proposée par LECUN et al. [1998]) et **AlexNet** KRIZHEVSKY et al. [2012]), beaucoup d'autres architectures ont suivi. Parmi elles nous pouvons citer le **GoogleNet** SZEGEDY et al. [2015] et les **VGG-*like* nets** SIMONYAN et ZISSERMAN [2014].

Des réseaux comme **LeNet**, **AlexNet** et **VGG19** présentent une structure sérielle agrégeant des blocs de {convolution, pooling, fonction d'activation} avant de finir sur une couche *fully-connected* (voir Figure 3.7).

Plus particulièrement, le réseau **VGG19** enchaîne des blocs où la réduction spatiale due au *max-pooling* est "compensée" par la profondeur des cartes d'activations qui double : $\frac{\text{largeur spatiale}}{2}$, profondeur sémantique $\times 2$. C'est un des réseaux les plus utilisés, notamment comme base structurelle pour des architectures plus complexes (voir Section 3.3.3).

Les macro-architectures

Le monde des *GPUs* étant en constante évolution, les possibilités offertes deviennent très grandes. La complexité des structures et leur profondeur n'a cessé d'augmenter. Cependant, il devient difficile pour des réseaux plus profonds de définir couche par couche la structure. En effet, il est difficile de discerner l'influence d'une couche sur une autre tant le nombre de paramètres devient conséquent. C'est dans ce contexte que sont nées les macro-architectures qui agencent des blocs prédéfinis d'opération –ou micro-architectures– dont les rôles respectifs sont compris.

Complexité des opérations Le GoogleNet de SZEGEDY et al. [2015] est constitué de blocs d'***Inception***, qui permettent d'effectuer en parallèle des opérations à différent niveau de précision sur un même tenseur d'entrée (voir Figure 3.8(a)). Dans la même logique, le SqueezeNet de IANDOLA et al. [2016] utilise des blocs de contraction de l'information (voir Figure 3.8(c)) permettant de réduire le nombre global de paramètres nécessaires : SqueezeNet obtient les mêmes performance que le célèbre AlexNet avec 50 fois moins de paramètres ce qui est très intéressant en terme d'embarquabilité du modèle, entraînement, test et définition. IANDOLA et al. [2016] est d'ailleurs à l'origine du terme de micro-architecture pour désigner les blocs élémentaires d'opérations.

Profondeur des réseaux D'autre part, il a été mis en avant un phénomène de "disparition du gradient" [HOCHREITER et al., 2001]. Plus il y a de couches dans un réseau plus l'information est filtrée (et donc son contenu se dilue), de même lors de la rétropropagation du gradient : l'information de l'erreur se propage couche par couche, avec de petites valeurs et peut devenir négligeable au bout d'un certain nombre de couches. Ce problème a pu être corrigé par HE et al. [2016a] qui furent les premiers à proposer des réseaux de neurones de 34, 50, 101 ou 152 couches. Une implémentation de leurs travaux est même monté à 1001 couches [HE et al., 2016b]. L'opération du bloc de base de cette architecture consiste à concaténer l'information de l'entrée du bloc avec sa sortie (le résidu). Cette opération permet de garder l'information d'entrée tout en augmentant la dimension sémantique du tenseur 3D. Ce bloc est illustré sur la Figure 3.8(b).

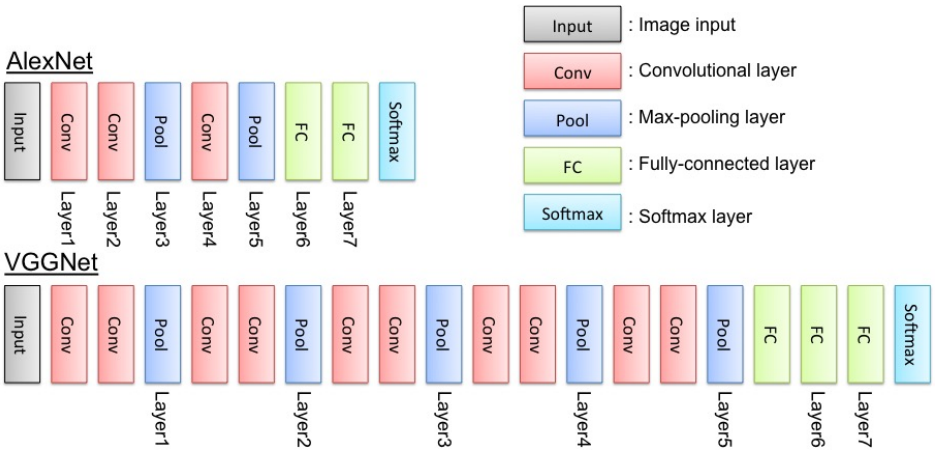


FIGURE 3.7 – Structures du réseau de neurones AlexNet et d'un réseau VGG (source :KATAOKA et al. [2015])

Les autres architectures

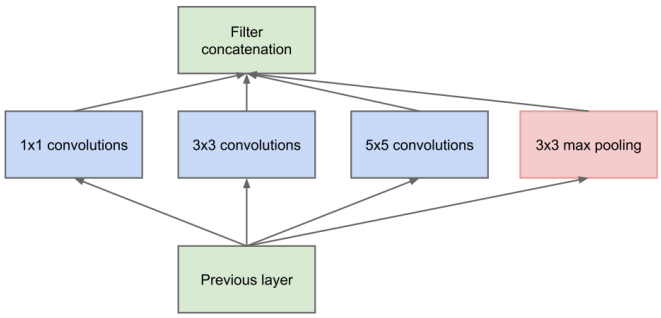
Les réseaux récurrents Aux réseaux de neurones convolutionnels sériels présentés ci-dessus nous pouvons ajouter les réseaux de neurones convolutionnels récurrents (souvent appelés *RNNs* pour *Recurrent Neural Networks*). Pour un certain neurone le tenseur de sortie peut être utilisé comme une autre entrée de la même opération pour l'itération suivante, ou, l'entrée principale du réseau de neurones peut accepter en plusieurs itérations différents tenseurs d'une même séquence. Ce genre de structure [GERS et al., 1999; HOCHREITER et al., 2001] offre l'avantage de pouvoir traiter des phrases, des vidéos, des stratégies de fenêtre glissante, *etc*., en apportant une connaissance temporelle des données. Nous ne faisons pas usage de cette technologie dans les travaux de cette thèse, mais leurs performances pour les tâches séquentielles en font une perspective de recherche très intéressante.

Le reste Il existe de très nombreuses architectures de réseaux de neurones, qui ne sont pas forcément convolutionnels. L'institut Asimov (une entreprise ayant pour vocation de mettre l'apprentissage profond au service de l'industrie "créative") propose un zoo des réseaux de neurones sur son site web (voir Figure 3.9).

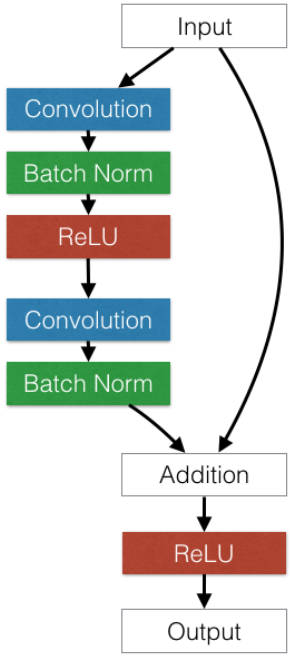
Parmi elles nous pouvons citer les machines de Boltzmann qui permettent de faire des entraînements non-supervisés avec une représentation interne latente des données [ACKLEY et al., 1985]. Les machines de Boltzmann ont également été employées pour initialiser des auto-encodeurs, mais cette pratique est devenue marginale. En effet aujourd'hui la plupart des initialisations de réseaux de neurones se font avec des réseaux de neurones similaires pré-entraînés sur le jeu de données ImageNet qui est très varié (1 000 000 d'images réparties en 1 000 classes).

Dans les autres architectures nous pouvons également citer PointNet de SU et al. [2017] qui n'utilise que des couches "*fully-connected*" sous forme de *MLP* pour traiter directement des ensembles de points 3D dans un tenseur de forme (N,3), T = [x1,y1,z1 ;x2,y2,z2 ;... ;xN,yN,zN]. Ceci permet de travailler sans avoir à ordonner les données ou les représenter dans l'espace et voxeliser le nuage ni même avoir recours à du maillage.

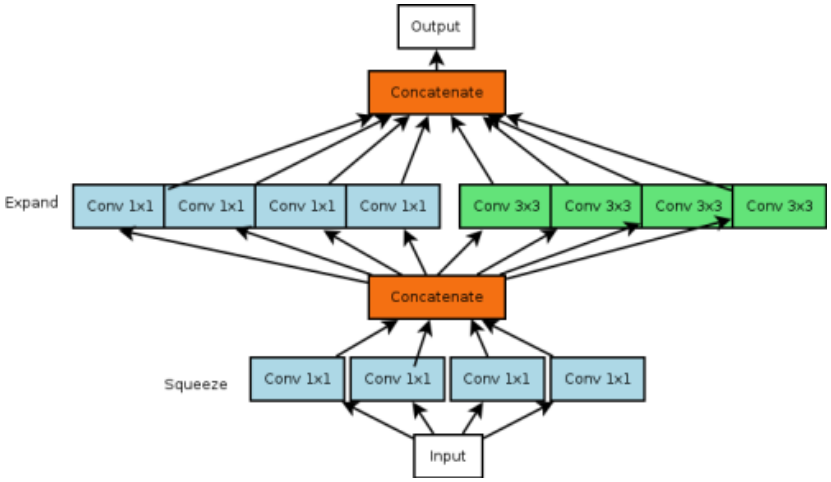
Définir une nouvelle architecture Il est relativement difficile aujourd'hui de définir une nouvelle structure qui n'est pas une variante des structures énoncées ci-dessus. En 30 ans le nombre de structures sérielles pour la classification peuvent être regroupées en très peu de classes avec des variantes négligeables. Certains études se focalisent plutôt sur la comparaison



(a) "*Inception Block*"



(b) "*Residual Block*"



(c) "*Squeeze Block*"

FIGURE 3.8 – Exemples de micro-architectures

des micro-architectures entre-elles SZEGEDY et al. [2017] ou l'inférence même des hyper-paramètres d'une structure SMITHSON et al. [2016].

Dans la suite du manuscrit, les réseaux de neurones feront toujours référence à des réseaux de neurones convolutionnels sériels (acycliques), sauf indication contraire.

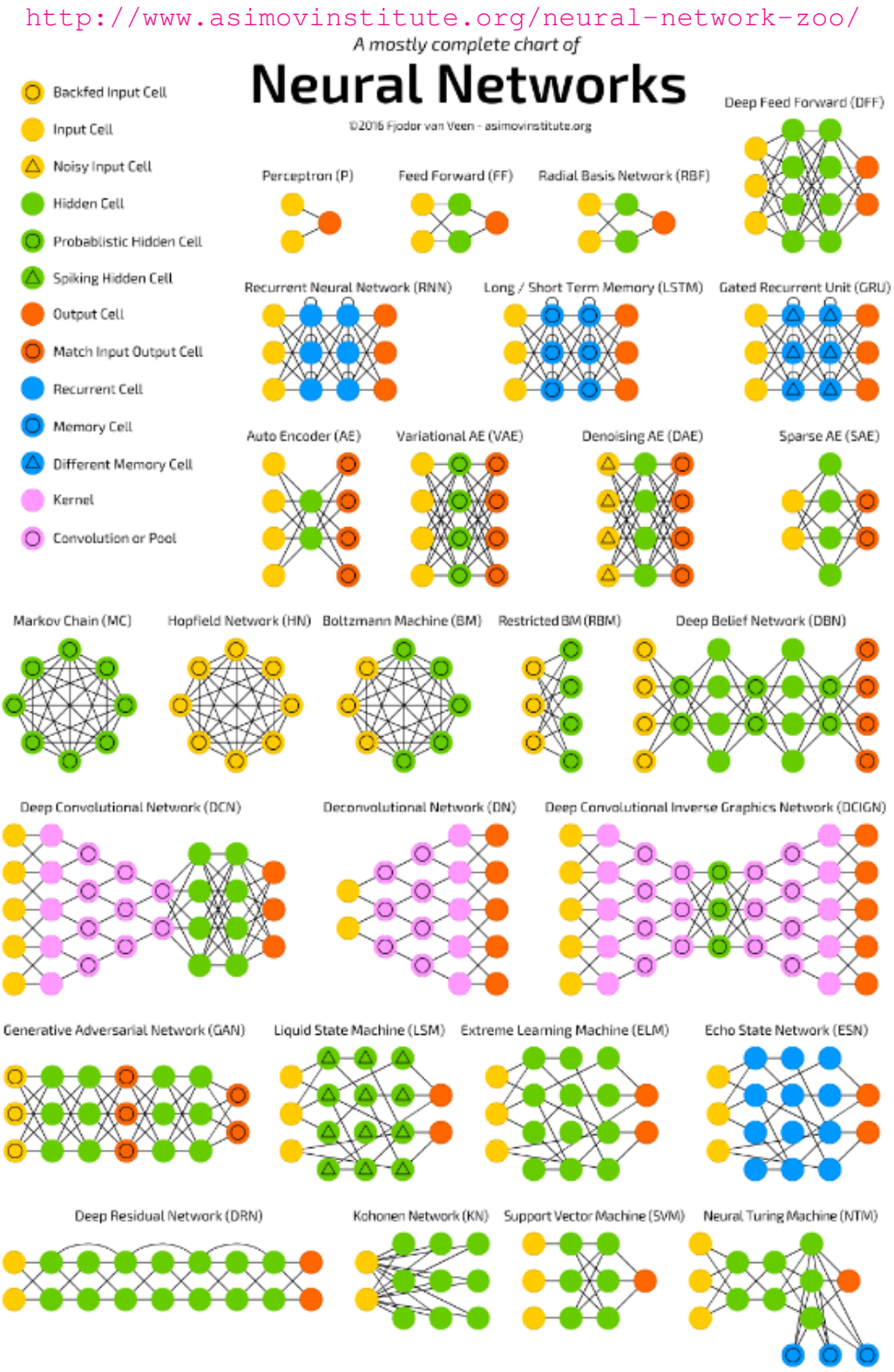


FIGURE 3.9 – Zoo de réseaux de neurones de l'Institut Asimov

3.3 La reconnaissance visuelle avec l'apprentissage profond

Un réseau de neurones est donc une fonction non linéaire qui transforme un tenseur en un autre tenseur. Cette fonction est entraînée en lui montrant des exemples entrée-sortie et en indiquant l'erreur sur la sortie prédite.

Cette technologie peut ainsi être appliquée à différents problèmes de la reconnaissance visuelle. La problématique est alors de définir quel est le tenseur d'entrée, quel est le tenseur de sortie désiré, et comment calculer une erreur sur la prédiction.

3.3.1 Classification par réseaux de neurones

Dans le cadre de la classification (sans localisation), l'objectif est de projeter le tenseur image dans un tenseur de probabilités d'appartenance aux classes considérées.

Pour cela les valeurs de sortie de la dernière couche du réseau de neurones (T) sont traitées par une couche dite de "*SoftMax*". Cette opération (voir Équation 3.19) applique une certaine normalisation des valeurs pour obtenir une distribution de probabilités sur les classes (p_{T}). Puis, il nous est possible de calculer une erreur entre les probabilités estimées et la vérité terrain en utilisant la régression logistique polytomique ou multinomiale dont la fonction de perte $\mathscr{L}$ est définie dans l'équation 3.20.

$$p_{\mathrm{T}} = \mathrm{softmax}(\mathrm{T}) = \frac{e^{\mathrm{T}_i}}{\sum_k e^{\mathrm{T}_k}} \tag{3.19}$$

$$\mathscr{L} = \frac{-1}{\mathrm{N}} \sum_{n=1}^{\mathrm{N_C}} \log(\hat{p}_n, l_n) \tag{3.20}$$

où l_n désigne la $n^{\text{ème}}$ étiquette (*label*) appartenant à $\{0, 1\}$ et T désigne un tenseur sur lequel nous appliquons l'opération de *SoftMax*. Autrement dit l est le vecteur de la vérité terrain et vaut 1 pour la classe n.

Il existe de plus une implémentation stable du calcul du gradient de cette fonction d'erreur exécutant à la fois la normalisation probabiliste ainsi que le calcul de l'erreur. Elle est implémentée dans l'ensemble des bibliothèques de *deep-learning* (par exemple *SoftMaxLogisticLoss*).

Il est souvent attribué à KRIZHEVSKY et HINTON [2009] et son réseau de classification entraîné sur ImageNet DENG et al. [2009] le regain d'intérêt pour l'apprentissage profond.

La classification par réseau de neurones est aujourd'hui la meilleure méthode de classification sur de nombreux jeux de données d'image :

- WAN et al. [2013] sur MNIST (0.21% d'erreur)
- GRAHAM [2014] sur CIFAR-10 (97% de précision) là où une méthode à base de descripteurs HOG + SVM à noyau [YANG et al., 2012] atteint les 55% de précision.
- CLEVERT et al. [2015] sur CIFAR-100 (75.72% de précision)
- Le challenge annuel ImageNet pour la classification de scène RUSSAKOVSKY et al. [2015] est également remportée par l'*Hikvision Research Institute* de *ShanghaiTech University* faisant usage de réseaux de neurones convolutionnels [2].

HE et al. [2016a] ont également atteint d'excellentes performances sur de nombreuses tâches de classification avec leur architecture ResNet et leur très grand nombre de couches (152 par exemple).

2. voir http://image-net.org/challenges/LSVRC/2016/results

3.3.2 Détection par réseaux de neurones

Dans le cas de la détection d'objets par réseau de neurones il est possible d'appliquer les mêmes stratégies de proposition de patchs qu'en vision par ordinateur classique (fenêtre glissante, proposition par saillance, stratégie des "5 crops", etc.) puis d'appliquer un réseau de neurones de classification sur les patchs. À nouveau, la localisation est directement définie par le patch.

Ce genre de stratégies est aussi exhaustif que l'est la stratégie de proposition de patchs, cependant ce n'est pas forcément efficace en terme de temps de calcul ou de ressource mémoire. De plus, comme énoncé dans le chapitre précédent, cela nécessite d'entraîner le réseau de neurones à différencier les classes désirées entre elles ainsi que vis-à-vis du fond de l'image (qui peut être fortement variable).

Ainsi, plusieurs problématiques se dessinent pour la détection d'objets :

1. Peut-on économiser des ressources en profitant de l'aspect convolutionnel des *CNNs* ?
2. Peut-on prédire une localisation avec un *CNN* ?
3. Peut-on traiter une image entière d'un seul coup ? (localisation et classification)

Nous présentons désormais les travaux de Ross Girshick *et al.* qui témoignent parfaitement de l'évolution des *CNNs* au service de la détection d'objets. Par ailleurs ils forment la base de l'algorithme qui sera présenté dans le Chapitre "Approche multi-modale pour la détection".

Le *RCNN* Le *RCNN* GIRSHICK et al. [2014] pour "*Regions with CNN features*" utilise un *CNN* comme une fonction de description (au même titre que les *HOGs*). En utilisant toujours une stratégie de proposition de patchs comme entrée et une *SVM* comme classifieur, cette méthode permet de proposer des patchs avec une étiquette. Par nature, le *RCNN* peut utiliser n'importe quelle stratégie de proposition de patchs en entrée. GIRSHICK et al. utilisent la "*Selective Search*" de UIJLINGS et al. [2013]. Le *RCNN* est illustré dans la Figure 3.10.

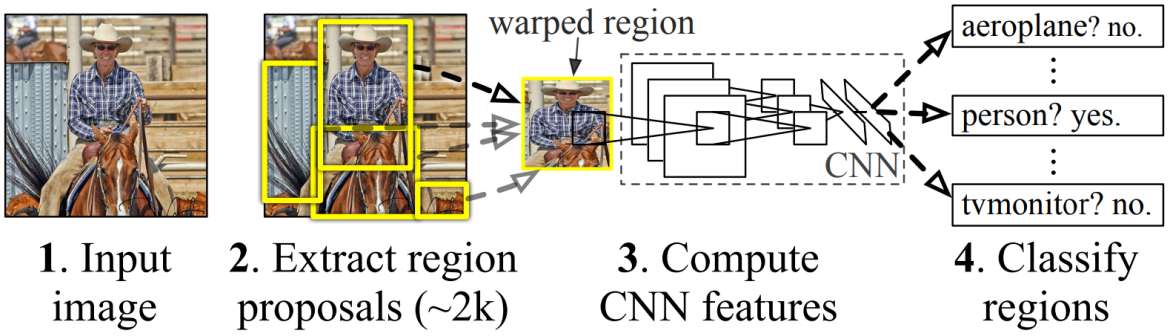


FIGURE 3.10 – Fonctionnement du *Regions with CNN* [GIRSHICK et al., 2014]

Le *RCNN* obtient lors de sa sortie en 2013 des scores sans précédent en détection au challenge *ILSVRC2013 detection dataset* RUSSAKOVSKY et al. [2015] (31,4% de mAP vs. 24.3% précédemment établi par OverFeat SERMANET et al. [2013]). Un avantage immédiat du *RCNN* est de pouvoir classifier les patchs en utilisant des batchs, et ainsi accélérer le processus de traitement.

le *Fast-RCNN* La seconde version du RCNN, appelée le Fast-RCNN [GIRSHICK, 2015], répond partiellement aux trois problématiques de la détection par réseau de neurones convolutionnels.

1. Il projette en une seule fois toute l'image d'entrée (pas seulement quelques patchs) dans un espace de caractéristiques grâce aux premières couches d'un réseau de neurones (typiquement jusqu'à la 5ème couche de convolutions d'un VGG16). Il y a donc une mise en commun des calculs pour tous les patchs potentiels. C'est intéressant car les techniques de propositions de patchs (énoncées dans le chapitre précédent) couvrent généralement toute l'image, avec des chevauchements.
2. Il calcule toujours des patchs avec une méthode indépendante (par exemple *Selective Search*).
3. Il utilise ensuite une couche appelée "*Roi-Pooling*" qui vient extraire les patchs proposées à l'étape 2 dans l'espace obtenu à l'étape 1 (*RoI* pour *Region of Interest*).
4. Il classifie ces patchs de caractéristiques avec la seconde partie du classifieur pour leur attribuer une étiquette (la classification par SVM a été remplacée par une classification par réseau de neurones).
5. Il utilise une couche "*fully-connected*" sur les caractéristiques du patch à la dernière couche du classifieur pour faire de la régression de localisation. À cette fin il régresse un paramètre d'élargissement, d'agrandissement ainsi qu'un décalage vertical et un décalage horizontal. Ceci permet d'améliorer la forme du patch et sa position. Comme indiquée précédemment, ce genre de tâche est parfaitement valide dans un réseau de neurones sous réserve de bien définir la fonction de perte : c'est généralement la perte euclidienne qui est utilisée pour régresser vers des valeurs réelles dans $]-\infty, +\infty[$.

Un schéma extrait de l'article original est disponible en Figure 3.11.

A la vue de cette méthode, une étape reste encore à intégrer dans le *CNN* : la proposition de patchs.

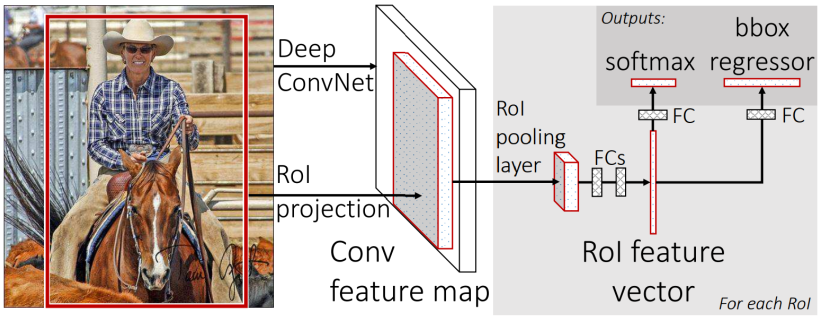


FIGURE 3.11 – Fonctionnement du *Fast-RCNN* [GIRSHICK, 2015]

le *Faster-RCNN* La même année, le Fast-RCNN est devenu le Faster-RCNN [REN et al., 2015] en y incorporant une stratégie interne de proposition de patchs : le *Region Proposal Network* ou *RPN*. Comme visible sur la Figure 3.12, le *RPN* profite également de l'espace de caractéristiques global (calculé sur toute l'image) pour y classifier un nombre prédéfini de patchs sans essayer d'inférer une des classes finales. Il effectue cette opération avec un petit *CNN* qui calcule un score d'*objectness*, que nous pourrions traduire par l'estimation qu'un patch soit un objet, à l'opposée qu'un patch soit de l'arrière-plan. De plus, le *RPN* effectue également une régression de localisation pour proposer des patchs d'objets relocalisés au classifieur. Il y a donc une double amélioration des localisations : une première qui utilise des *a priori* sur le fait d'être un objet, et une deuxième (à la fin du classifieur) qui utilise des *a priori* propres à chaque classe.

TABLEAU 3.1 – Résumé des méthodes de détection par *CNN* et résultats sur Pascal VOC 2007 EVERINGHAM et al. [2010]

Méthode	Images par seconde	mAP	Référence
RCNN	0,1	62%	GIRSHICK et al. [2014]
Fast-RCNN	10	70,0%	GIRSHICK [2015]
Faster-RCNN	17	73,2%	REN et al. [2015]
Yolo	45	63,4%	REDMON et al. [2016]
SSD	59	74,3%	LIU et al. [2016]
Yolo v2	67	76,8%	REDMON et FARHADI [2016]
RCNN OHEM	n.a	78.9%	SHRIVASTAVA et al. [2016]

Le *Faster-RCNN* devient ainsi le réseau de neurones le plus rapide pour de la détection d'objets et fixe à l'époque le nouvel état de l'art sur le challenge de référence PASCAL VOC detection [EVERINGHAM et al., 2010].

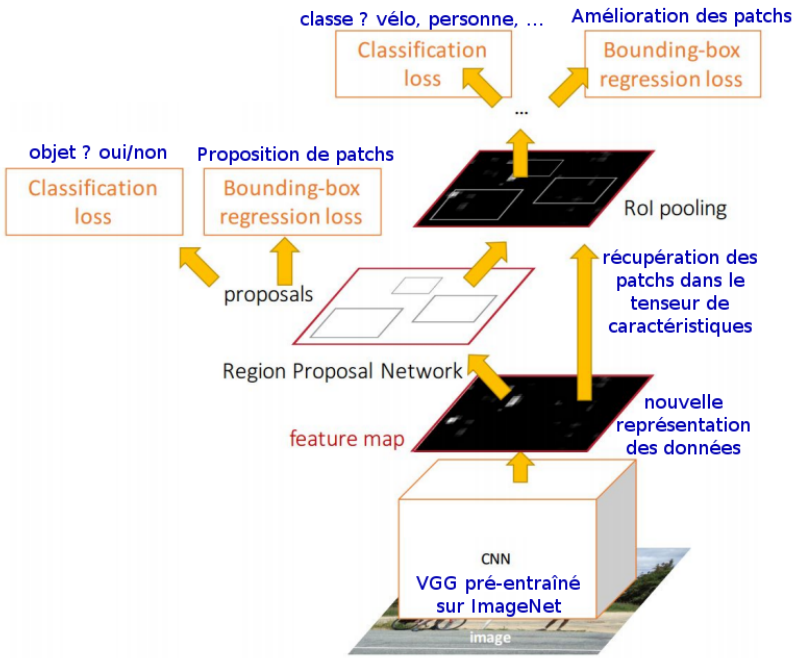


FIGURE 3.12 – Fonctionnement du *Faster-RCNN* [REN et al., 2015]

Le *RCNN* a également continué à évoluer, notamment pour la segmentation que nous abordons dans la section suivante. Une implémentation du *Fast-RCNN* avec du *Online Hard Example Mining* (*OHEM*) –ou recherche en ligne d'exemples difficiles– a permis d'augmenter le score de cette méthode sur le challenge de *Pascal VOC 2012* (voir Tableau 3.1).

Ce qui est intéressant avec la structure du *Faster-RCNN* c'est qu'elle est très modulaire. Ceci est le résultat de son évolution dans les travaux de GIRSHICK. Chaque brique peut être remplacée, supprimée ou déplacée. C'est un avantage majeur pour le développement de nouvelles solutions comme nous le verrons dans le Chapitre 5 traitant notamment de la détection de personnes dans des données multi-modales.

En utilisant les mêmes concepts de régression incorporés dans le *CNN* mais en agençant différemment la structure, d'autres travaux ont pu augmenter les scores et les vitesses de calcul : voir Tableau 3.1.

Nous citons notamment YOLO et sa version 2 où REDMON et al. n'utilise pas de *RPN* intermédiaire. C'est directement le dernier tenseur qui a pour objectif de produire des patchs (x,y,w,h,p_{objet}) et des scores pour chaque classe (c_i) pour chaque cellule d'une sous-division de l'image d'entrée. Le tenseur de sortie est donc de la forme $W_{\text{division}} * H_{\text{division}} * (N_{\text{patch}} * 5 + N_{\text{classes}})$. Comparé au *RPN* qui ne propose pas les patchs avec une "*objectness*" trop faible, YOLO ne s'embarasse pas de cette étape. De plus, le RPN se contraint à des formes pré-determinées à rechercher (agnostiques aux classes) alors que YOLO est libre de sa proposition et Yolo V2, lui, utilise des *a priori* sur les formes du jeu de données. Une vidéo illustrant les performances de YOLO v2 est disponible à l'adresse de l'autheur : https://pjreddie.com/darknet/yolo/.

L'approche SSD de LIU et al. est très proche de YOLO. Ils font en plus l'usage d'une approche multi-échelle en réutilisant des tenseurs intermédiaires qui permet d'améliorer l'invariance en taille du détecteur.

3.3.3 Segmentation sémantique par réseau de neurones

La classification sémantique consiste à attribuer une étiquette avec une résolution pixellique. Cette tâche peut également être appliquée aux espaces 3D en associant à chaque point de l'espace une étiquette.

La **segmentation sémantique 2D** est devenue populaire dans les années 2000 avec des challenges sur les jeux de données CamVid BROSTOW et al. [2009] ou Pascal VOC EVERINGHAM et al. [2010]. La **segmentation sémantique 3D** s'est répandue avec l'apparition de scanners laser qui permettent l'acquisition de larges nuages de points. Cette technologie est notamment utilisée pour la cartographie urbaine HAALA et al. [1998] et bien sûr la robotique ANGUELOV et al. [2005]; FILLIAT et al. [2012].

Segmentation 2D

Réseaux purement convolutionnels En **2D**, les réseaux de neurones purement convolutionnels – ou *fully-convolutional networks* (FCN)– de LONG et al. [2015] ont été une étape importante dans le domaine de la segmentation en produisant une prédiction dense avec des *CNNs*. LONG et al. utilisent une propriété intéressante qui relie les couches de convolutions et les couches "fully-connected" :

1. *conv→fc* : il est possible de voir une couche de convolutions comme le résultat d'une couche "*fully-connected*" où les poids de la couche "*fully-connected*" seraient essentiellement peuplés de 0 et de répétitions décalées des filtres convolutionnels.
2. *fc→conv* : il est possible de voir un neurone de sortie d'une couche *fully-connected* comme un neurone de sortie d'une convolution de même surface d'action que la *fully-connected*. Dans le cas d'une *fully-connected* entre un tenseur dans $\mathbb{R}^m$ et un tenseur dans $\mathbb{R}^n$, l'opération est assimilable à n convolutions de m paramètres.

En utilisant cette astuce, LONG et al. ont pu convertir toutes les structures classiques de réseau de neurones de classification (*CNN+fc+fc*) en réseaux de neurones de prédiction dense, donc de segmentation sémantique. Il faut pour cela légèrement modifier la fonction d'erreur qui doit appliquer une régression logistique multinomiale par pixel sémantique généré.

Cette approche purement convolutionnelle peut ainsi s'appliquer sur une image entière tout en bénéficiant à chaque couche de l'optimisation des processus de convolution sur *GPU*.

Cependant, la dégradation de l'information par les couches de *pooling* et par le filtrage des convolutions rend la prédiciton dense peu résolue et mène à une perte de performance une fois

la prédiction remise à l'échelle d'entrée. Pour cela, LONG et al. ré-injectent les tenseurs post-*pooling* à la fin du réseau pour permettre d'agrandir progressivement les cartes de prédictions en précisant où était l'information sélectionnée par l'opération de max-pooling.Ce concept de réinjection des tenseurs intermédiaires est désormais omniprésent dans le monde de la segmentation sémantique. Une belle illustration de *FCN* est proposée par TAI et al. [2016] dans la Figure 3.13

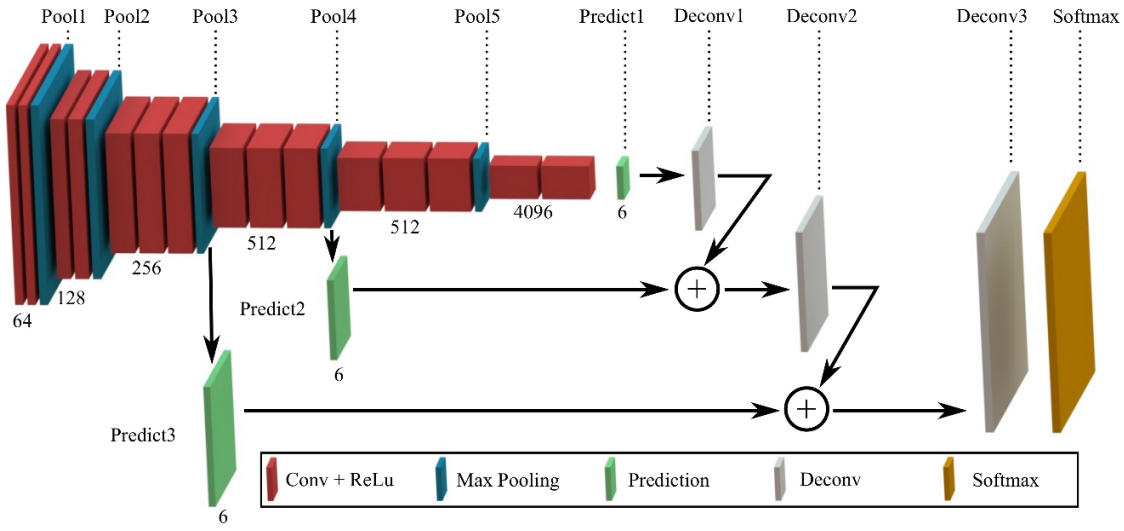


FIGURE 3.13 – Illustration d'un "Fully Convolutionnal Network" [TAI et al., 2016] qui réinjecte les sorties des couches de *pooling* pour dilater la carte de prédiction basse résolution.

Structures "Encodeur+Décodeur" Sans forcément produire une structure purement convolutionnelle, il est possible d'utiliser une structure de la forme encodeur+décodeur – ou *Encoder + Decoder (ED)* –. La première structure *E+D* fut utilisée en apprentissage non supervisée. Elle consiste à définir comme tenseur de sortie d'un réseau de neurones le même tenseur que l'entrée (voir Figure 3.14). L'objectif est de reconstruire le signal d'entrée. Cependant, les couches intermédiaires peuvent produire des tenseurs qui reposent dans des espaces de dimensions moindres. Cette situation force le réseau à sélectionner l'information utile pour la reconstruction. Le terme de compression est parfois employé pour désigner cette spécificité des auto-encodeurs. Cette structure non supervisée est parfois utilisée comme base de paramètres pour l'entraînement d'un *CNN*, en retirant le décodeur, ou d'un *E+D*, en changeant la sortie désirée.

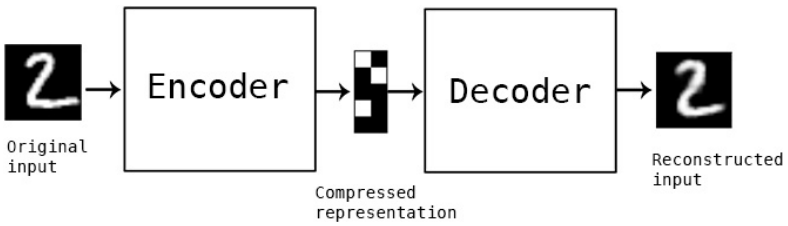


FIGURE 3.14 – Illustration d'un auto-encodeur. Image extraite du blog de Keras, un célèbre *framework* de *Deep Learning* : https://blog.keras.io/building-autoencoders-in-keras.html.

Cette structure d'*E+D* est conceptuellement idéale pour la segmentation sémantique. En

effet, elle permet de produire directement une carte de prédiction de la taille de l'image comme dans RAZAKARIVONY et JURIE [2014].

De plus, les méthodes les plus performantes aujourd'hui utilisent la réinjection de tenseurs intermédiaires de l'encodeur dans le décodeur. C'est notamment le cas de SegNet BADRINARAYANAN et al. [2015] et de sa version bayésienne KENDALL et al. [2015]. À l'instar du "Fully-Convolutional Network" de LONG et al., l'information des couches de *pooling* est réinjectée dans le décodeur. Cependant cette fois-ci ce sont directement les indices des maximums retenus par le *max-pooling* qui sont propagés, afin de reconstruire progressivement l'image en ayant conscience d'où venait l'information. Ce concept est illustré sur la Figure 3.15.

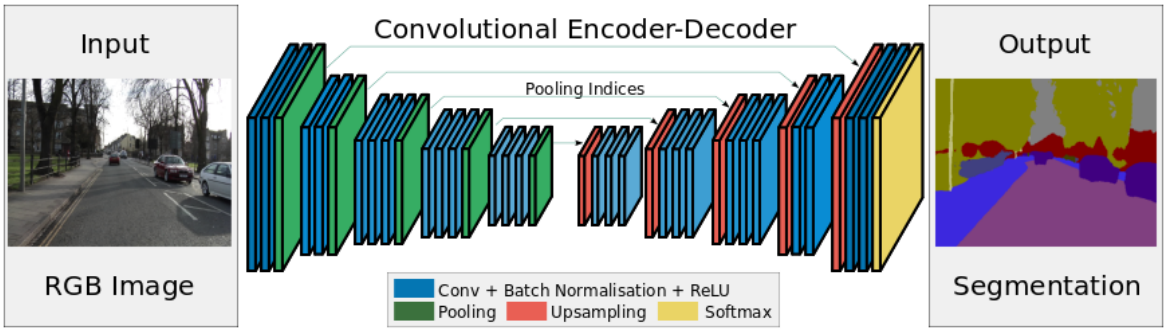


FIGURE 3.15 – Illustration du réseau de segmentation sémantique SegNet[BADRINARAYANAN et al., 2015].

Une autre structure de référence est la structure U-Net [RONNEBERGER et al., 2015], qui injecte entièrement les tenseurs intermédiaires *post-pooling* dans le décodeur selon la configuration illustrée dans la Figure 3.16

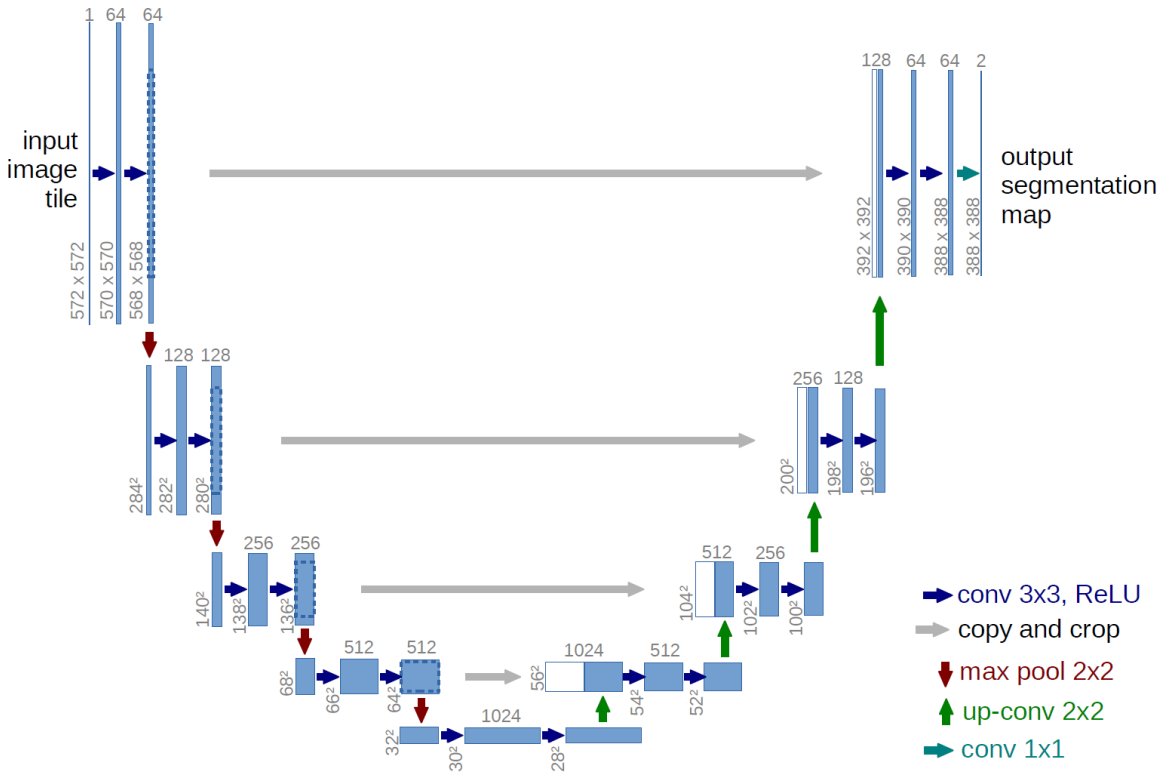


FIGURE 3.16 – Illustration du réseau de segmentation sémantique U-net[RONNEBERGER et al., 2015].

L'état de l'art dans le monde de la segmentation sémantique est aujourd'hui basé sur un mélange entre des réseaux de neurones et ce qui est appelé des champs aléatoires conditionnels – ou *Conditional Random Fields* (CRFs) –. Les *CRFs* permettent d'apporter un cadre probabiliste conditionnel sur les variables considérées. Cet outil est régulièrement utilisé en fin de processus de segmentation pour améliorer une carte de prédiction connaissant les relations conditionnelles entre les classes. En effet, les *CRFs* peuvent par exemple apprendre qu'une table est sur le sol et qu'il y a peu de chance qu'elle soit traversée par un pot de fleurs. Ceci permet notamment d'améliorer les contours des objets. LIN et al. [2017] utilisent une architecture convolutionnelle multi-échelle et définissent d'autres réseaux de neurones pour remplacer des fonctions internes au calcul d'un *CRF*. C'est un processus long, complexe à mettre en place, impossible à entraîner en une seule étape mais qui réalise aujourd'hui les meilleures performances de segmentation sémantique sur de nombreux challenges. Nous en reparlons dans le Chapitre 6.

Nous présentons également le *Mask-RCNN* HE et al. [2017], voir Figure 3.17, qui étend le concept de *RCNN* en ajoutant un module de segmentation. Le *RCNN* et ses déclinaisons illustrent bien la capacité d'un réseau de neurones à s'adapter à diverses tâche tout en utilisant une représentation intermédiaire commune des données.

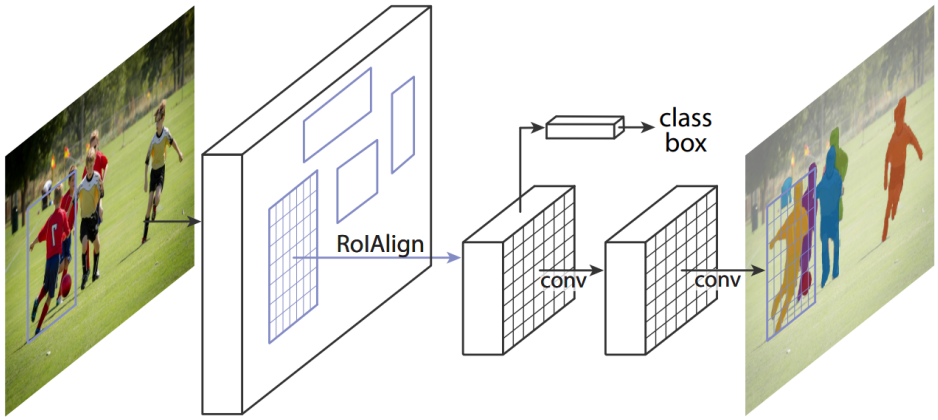


FIGURE 3.17 – Fonctionnement du *Mask-RCNN*, extrait de HE et al. [2017]

Segmentation 3D

En utilisant un cadre d'apprentissage profond, les représentations et les classifieurs pour la segmentation 3D sont appris en un seul bloc. Dans ce domaine, trois approches concourent.

Les méthodes à base de voxel Premièrement, les méthodes **voxel-based** utilisent une voxelisation de l'espace 3D pour créer des tenseurs 3D afin d'alimenter un réseau neuronal convolutif 3D (CNN) [LAI et al., 2014; MATURANA et SCHERER, 2015; WU et al., 2015], principalement pour la classification d'objets. À la suite de cette façon d'encoder la 3D, qui peut être coûteuse en termes de calcul, HACKEL et al. [2017] ont proposé d'utiliser des tenseurs locaux multiéchelle 3D pour caractériser les points 3D réels. Dans la tâche d'étiquetage sémantique de voxel (qui est légèrement différente de l'étiquetage ponctuel) du benchmark ScanNet [DAI et al., 2017], le réseau de référence prédit les étiquettes pour une colonne complète de voxels à la fois en utilisant le voisinage des voxels.

Les stratégies multi-vues Deuxièmement, la stratégie **multi-vue** consiste à appliquer des réseaux neuronaux sur des tenseurs 2D qui sont des collections d'images de la scène.

Pour extraire et classer les objets, le *CNN* "multi-vue" (*MVCNN*) [Su et al., 2015] prend plusieurs images autour d'un objet maillé en 3D et effectue une classification d'image à l'aide d'un réseau de neurones.

La représentation PANORAMA [Sfikas et al., 2017] projettent les points du nuage sur des cylindres orientés suivant les 3 directions principales du volume (voir Figure 3.18).

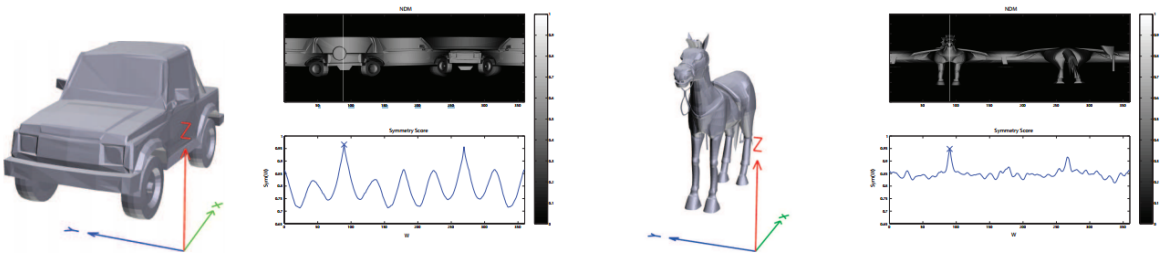


FIGURE 3.18 – Exemples de la représentation 3D PANORAMA [Sfikas et al., 2017]

SnapNet [Boulch et al., 2017] prend de façon aléatoire de nombreuses vues tout autour de la scène, crée des images virtuelles *RGB* et composite (distance, entropie, angle de la normale à la surface avec le sol) puis traitent ces images avec deux U-Net [Ronneberger et al., 2015] et un module de correction residuel [Audebert et al., 2016]. L'encodage "composite", renommé *DNA* (*Depth, Noise, Angle*) sera ré-employé dans le Chapitre 5 lors d'une comparaison des différents encodages.

Les stratégies par ensemble de points Troisièmement, certaines méthodes (**point-based**) fonctionnent directement sur des ensembles de points non-ordonnés. Pour cela elles utilisent des architectures avec des couches *fully-connected* et des couches de *pooling* plutôt que des couches convolutionnelles. Ainsi, PointNet Su et al. [2017] peut prédire des classes pour la forme 3D entière ou effectuer une segmentation sémantique d'une scène 3D. Cependant, il manque la capacité de capturer le contexte local à différentes échelles. Pour contourner cet inconvénient et améliorer la généralisation sur des scènes complexes, PointNet ++ Qi et al. [2017] introduit un réseau complémentaire plus bas niveau.

3.3.4 NMS par apprentissage profond

Comme nous venons de le signaler avec la méthode de Lin et al. [2017] et leurs *CRFs*, la tendance est à remplacer les "briques-méthodes" existantes par des réseaux de neurones. Cette perspective permet d'injecter facilement de la non-linéarité et se prête particulièrement aux traitements de données géométriques comme des images.

C'est dans ce même concept, que Hosang et al. [2017] ont proposé récemment de remplacer la *NMS* par des réseaux de neurones. Comme indiqué dans ce chapitre sur l'apprentissage profond, le problème revient à définir une entrée, une sortie désirée et une fonction d'erreur sur la sortie prédite. Hosang et al. ont introduit un réseau de neurones pour faire de la *NMS* ainsi qu'une fonction de perte qui pénalise la proposition de plusieurs détections pour une même instance.

3.4 Les données multi-modales en apprentissage profond

L'usage d'une modalité comme la carte de profondeur permet d'être moins dépendant aux couleurs et aux textures des objets. Cela permet de se concentrer également sur la forme

géométrique globale et la séparation d'instances d'objet. Cependant, l'utilisation de la carte de profondeur dans les réseaux de neurones est un problème qui n'a pas encore de solution triviale. BADRINARAYANAN et al. reportait même en 2015 avec le SegNet de meilleurs résultats que l'état de l'art sur un jeu de données *RGB-D* en utilisant uniquement l'information des canaux couleurs.

Les approches *RGB-D* suivent généralement le même modèle : définir une structure de réseau multi-modale et encoder la carte de profondeur par prétraitement pour faciliter l'apprentissage du réseau. En effet, les données de carte de profondeur sont très bruitées et peuvent théoriquement se prêter à des inférences géométriques, comme démontré par SILBERMAN et al. [2012]. Il faut donc d'abord les filtrer pour supprimer les artefacts : les objets brillants, trop éloignés, trop proches ou réfléchissants, peuvent ne pas répondre aux stimulus des caméras *RGB-D*. Ensuite, nous pouvons transformer l'information de la carte de profondeur et en extraire d'avance certaines caractéristiques spatiales.

Les travaux présentés dans FILLIAT et al. [2012] proposent de transformer l'information *RGB-D* aux travers de trois représentations (histogrammes locaux de couleurs, *3D Surflets* [WAHL et al., 2003], *SURF*) et utilisent ensuite un réseau de neurones pour travailler sur la concaténation de ces représentations. Pour effectuer la détection ils effectuent un traitement de l'information volumique obtenue par le biais de divers capteurs (dont le LIDAR) et y applique une stratégie de segmentation par croissance des zones 3D.

LENZ et al. [2015] proposent trois solutions différentes pour fusionner l'information de plusieurs modalités : la concaténation des canaux au début du réseau, entraîner chaque modalité indépendemment et fusionner les tenseurs de fin d'experts, ou déterminer par apprentissage des sous-ensembles de tenseurs de chaque expert à concaténer ensemble. Cette dernière méthode donne de meilleurs résultats sur leurs données.

Dans une implémentation multi-modale du RCNN [GIRSHICK et al., 2014], appelée *Depth-RCNN* [GUPTA et al., 2014], l'encodage de la carte de profondeur au format HHA permet d'améliorer significativement les résultats. **HHA** Désigne un encodage de la carte de profondeur. L'information est retraduite en trois canaux : **H**auteur, disparité **H**orizontale et **A**ngle de la normale à la surface avec le vecteur gravité.

Le HHA n'est qu'un changement de représentation, passant d'une information géométrique 2D (qui possède de l'information 3D) à une représentation 3 canaux ayant chacun un sens distinct, toujours géométriques mais indépendants.

Un exemple d'encodage HHA est présenté dans la Figure 3.19.

LONG et al. ont confirmé les avantages de l'encodage HHA en adaptant le *Fully Convolutional Network* (*FCN*) [LONG et al., 2015] aux données *RGB-D*. Il a mis en parallèle deux *FCN* (un expert *RGB* et un expert *Depth*). Il a ensuite simplement fusionné avec une somme les cartes de prédiction de chaque expert.

Plus récemment, EITEL et al. [2015] ont introduit une "approche double-flux" comparable au *FCN* mais plus subtile. Les deux experts (*RGB* et *Depth*) sont fusionnés par deux couches "*fully-connected*" successives en fin des réseaux. Comme décrit dans leur article, ils ont choisi de ne pas utiliser l'encodage HHA, long à calculer, mais ont choisi d'effectuer un rendu de l'intensité de la carte de profondeur avec la carte de couleur "jet" (voir Figure 5.1(h) (page 87)). Ils ont par la suite amélioré leurs travaux sur la détection multi-modale en introduisant dans MEES et al. [2016] un "gating network" qui pondère les sorties de chaque expert (voir Figure 3.20). Ce faisant, le réseau global est capable de donner plus d'importance à l'un de ses experts en considérant des tenseurs intermédiaires des dernières couches. Cette approche est également connue sous le nom de *mixture of deep network experts* (*MoDe*).

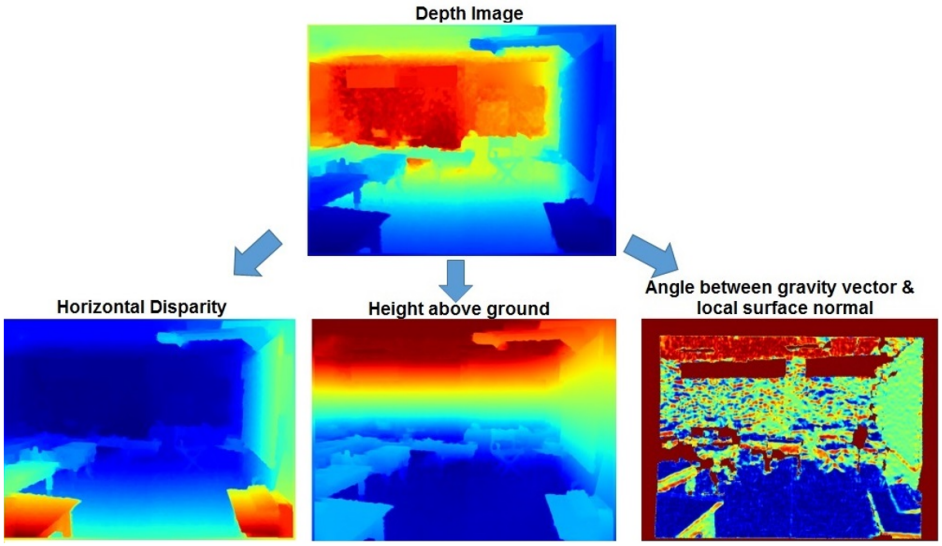


FIGURE 3.19 – Exemple de l'encodage *HHA* extrait de RAJ et al. [2015]

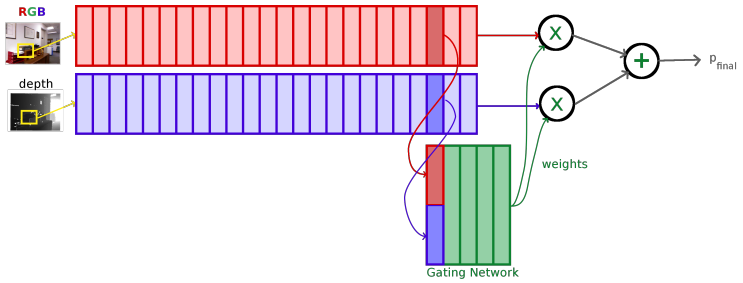


FIGURE 3.20 – Fusion avec le *Gating Network* [MEES et al., 2016]

En plus du *HHA* et du "*Jet rendering*" il existe d'autres encodages potentiels pour la carte de profondeur. Nous pouvons citer en plus le *DHA* de HANDA et al. [2016] qui remplace la disparité horizontale par la carte de profondeur.

Dans la catégorie fusion tardive nous pouvons également citer AUDEBERT et al. [2016] qui emploie un module comparable au Gating Network. Au lieu de prédire des facteurs de pondération pour chaque expert AUDEBERT et al. utilisent des tenseurs des dernières couches pour prédire contextuellement un delta à ajouter à la moyenne des tenseurs de prédiction : $p_{\text{final}} = 0.5p_{\text{RGB}} + 0.5p_{+}\delta$. L'objectif de δ est de venir accentuer certaines probabilités en fonctions des tenseurs intermédiaires observés.

Une autre stratégie multi-modale a été proposée par HAZIRBAS et al. [2016]. Appelée FuseNet (voir Figure 3.21), elle consiste à utiliser une structure encodeur-décodeur sur les données *RGB* dans laquelle nous injectons les tenseurs intermédiaires d'un autre encodeur spécialisé sur la *Depth*. Un inconvénient de cette méthode est que si l'activation résultante des filtres convolutifs n'est pas suffisante pour juste une seule des modalités, il est possible que l'activation totale ne soit pas suffisante pour le macro-réseau, bloquant localement les informations. Par exemple, si l'image *RGB* est trop sombre, seul l'expert *Depth* devrait s'exprimer, ce qui n'est pas possible ici. C'est un inconvénient majeur des structures hybrides (à l'opposée des structures avec fusion tardive ou "late fusion").

L'architecture de FuseNet a évolué vers une méthode profitant de l'aspect temporel des données dans les travaux de MA et al. [2017]. Le jeu de données NYUDv2 (voir Sous-Section 2.3) est composé de vidéos *RGB-D* pour lesquelles seulement quelques images sont étiquetées. C'est sur ce sous-ensemble étiqueté que les méthodes sont généralement comparées. Mais

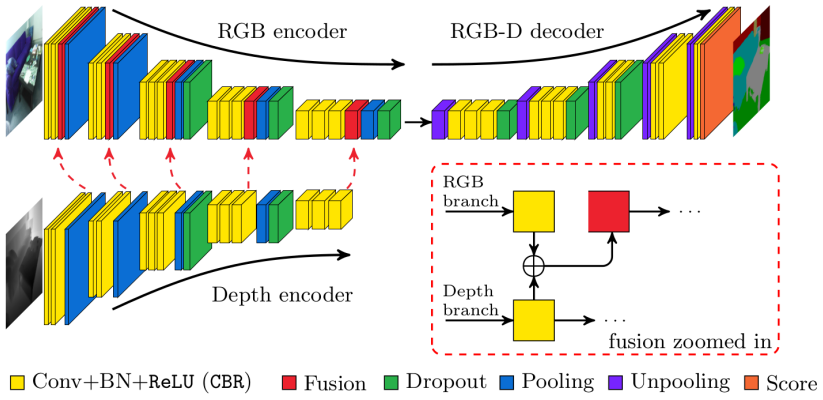


FIGURE 3.21 – Fusion avec le *FuseNet* [HAZIRBAS et al., 2016]

TABLEAU 3.2 – Méthodes *RGB-D* pour la détection d'objets.

Méthode	Encodage	fusion	Détails
Depth-RCNN [GUPTA et al., 2014]	HHA	"*late fusion*"	concaténation des tenseurs des experts *Depth* et *RGB* + SVM
FCN [LONG et al., 2015]	HHA	"*late fusion*"	somme des sortie des experts *Depth* et *RGB*
Dual Stream [EITEL et al., 2015]	jet colormap	"*late fusion*"	Fusion par *"fully-connected"*
MoDe [MEES et al., 2016]	jet colormap	"*late fusion*"	pondération avec un *Gating network*
Residual Correction [AUDEBERT et al., 2016]	N.A.	"*late fusion*"	accentuation de la moyenne avec un résidu
FuseNet [HAZIRBAS et al., 2016]	*Depth* brute	"*hybrid fusion*"	injection de tenseurs intermédiaires

MA et al. proposent une méthode qui profite des images non-étiquetées adjacentes aux images étiquetées. En utilisant une approche par *SLAM* ils peuvent reprojeter toute l'information dans le repère étiqueté et ainsi améliorer l'entraînement et donc le test sur le challenge de segmentation sémantique de NYUDv2. Cette approche profite concrètement de l'aspect robotique du jeu de données. Nous en reparlerons dans le Chapitre 6 traitant de la segmentation sémantique 3D.

En utilisant le formalisme introduit par VALADA et al. [2016], nous pouvons résumer ces dernières approches dans le Tableau 3.2.

Pour terminer, nous pouvons également citer STEDER et al. [2009] qui calcule des caractéristiques sur des nuages de points obtenus par traitement de la carte de profondeur. Bien que non "multi-modale", cette approche pourrait être considérée comme un expert pour une méthode multi-modale indépendante des structures *CNNs* de base (comme le *Gating Network* ou la *Residual Correction*).

3.5 Conclusion

Les méthodes à base de réseaux de neurones offrent de nombreux avantages. Elles peuvent partir directement des données brutes et travailler avec une seule structure à la reprojection de

l'information dans un espace de représentation intéressant, à la classification, à la détection, à la segmentation, à la regression... Ce grand pouvoir d'adaptation se fait au prix d'un grand nombre de paramètres internes et impose des exigences matérielles, logicielles et temporelles fortes. En particulier, les réseaux de neurones convolutionnels possèdent un profond aspect géométrique dans leur façon de traiter l'information. Il faut garder cette notion à l'esprit pour définir sa tâche et comprendre son fonctionnement interne. Nous nous intéressons à ces solutions car ce sont aujourd'hui les méthodes de sémantisation les plus performantes.

Il existe de nombreuses architectures différentes de réseaux de neurones. Cette diversité nous mène à une première réflexion : Quelle méthode choisir ? Ne sont-elles pas complémentaires ? Si oui, comment les coupler pour améliorer les performances recherchées ? Nous abordons cette problématique de fusion dans le Chapitre "Sélection d'algorithmes de classification".

L'usage de représentations internes intermédiaires comme sources d'information pour des opérations complémentaires comme dans les architectures *RCNNs* offre des possibilités combinatoires intéressantes pour la détection multi-modale que nous étudions dans le Chapitre "Approche multi-modale pour la détection". À la lumière des contributions sur les données *RGB-D*, l'encodage est une problématique importante pour faciliter le travail d'une méthode de sémantisation basée sur la carte de profondeur. C'est pourquoi nous étudierons également cet aspect dans le Chapitre "Approche multi-modale pour la détection".

Enfin, les méthodes d'apprentissage profond ont particulièrement évolué vis-à-vis de la tâche de segmentation sémantique avec les architectures encodeur-décodeur comme SegNet ou FuseNet. L'information sémantique produite par ces méthodes est parfois considérée comme de plus haut niveau car précise au pixel près. Nous évaluons quelques méthodes de segmentation sémantique dans le Chapitre "Approche multi-modale pour la détection" et utilisons l'expérience acquise au sein d'une approche multi-vue dans le Chapitre "Approche multi-vue pour la segmentation". La stratégie multi-vue consiste à générer des images virtuelles 2D d'une représentation 3D de l'information. Elle se prête donc particulièrement à l'usage des réseaux de neurones convolutionnels 2D. Cela permet d'importer nos connaissances des outils 2D sur des nuages de points 3D d'une part et permet d'aborder les données RGB-D sous leur aspect 2.5D d'autre part.

3.6 Références

ACKLEY, D. H., G. E. HINTON et T. J. SEJNOWSKI. 1985, «A learning algorithm for boltzmann machines», Cognitive science, vol. 9, n° 1, p. 147–169. 42

ANGUELOV, D., B. TASKARF, V. CHATALBASHEV, D. KOLLER, D. GUPTA, G. HEITZ et A. NG. 2005, «Discriminative learning of markov random fields for segmentation of 3d scan data», dans Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on, vol. 2, IEEE, p. 169–176. 49

AUDEBERT, N., B. LE SAUX et S. LEFÈVRE. 2016, «Semantic segmentation of earth observation data using multi-modal and multi-scale deep networks», dans Asian Conference on Computer Vision, Springer, p. 180–196. 53, 55, 56

BADRINARAYANAN, V., A. KENDALL et R. CIPOLLA. 2015, «SegNet : A Deep Convolutional Encoder-Decoder Architecture for Image Segmentation», arXiv preprint arXiv :1511.00561. URL http://arxiv.org/abs/1511.00561. 51, 54

BOULCH, A., B. LE SAUX et N. AUDEBERT. 2017, «Unstructured point cloud semantic labeling using deep segmentation networks», Eurographics/3DOR. 53

BROSTOW, G. J., J. FAUQUEUR et R. CIPOLLA. 2009, «Semantic object classes in video : A high-definition ground truth database», Pattern Recognition Letters, vol. 30, n° 2, p. 88–97. 49

CIREŞAN, D. C., U. MEIER, J. MASCI, L. M. GAMBARDELLA et J. SCHMIDHUBER. 2011, «High-performance neural networks for visual object classification», arXiv preprint arXiv :1102.0183. 39

CLEVERT, D.-A., T. UNTERTHINER et S. HOCHREITER. 2015, «Fast and accurate deep network learning by exponential linear units (elus)», arXiv preprint arXiv :1511.07289. 45

DAI, A., A. X. CHANG, M. SAVVA, M. HALBER, T. A. FUNKHOUSER et M. NIESSNER. 2017, «Scannet : Richly-annotated 3d reconstructions of indoor scenes», dans CVPR, Honolulu, Hawai, USA, p. –. 52

DENG, J., W. DONG, R. SOCHER, L.-J. LI, K. LI et L. FEI-FEI. 2009, «Imagenet : A large-scale hierarchical image database», dans Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on, IEEE, p. 248–255. 33, 45

EITEL, A., J. T. SPRINGENBERG, L. SPINELLO, M. RIEDMILLER et W. BURGARD. 2015, «multi-modal deep learning for robust rgb-d object recognition», dans Intelligent Robots and Systems (IROS), 2015 IEEE/RSJ International Conference on, IEEE, p. 681–687. 54, 56

EVERINGHAM, M., L. GOOL, C. K. WILLIAMS, J. WINN et A. ZISSERMAN. 2010, «The pascal visual object classes (voc) challenge», International journal of computer vision, vol. 88, n° 2. 48, 49

FILLIAT, D., E. BATTESTI, S. BAZEILLE, G. DUCEUX, A. GEPPERTH, L. HARRATH, I. JEBARI, R. PEREIRA, A. TAPUS, C. MEYER et al.. 2012, «Rgbd object recognition and visual texture classification for indoor semantic mapping», dans Technologies for Practical Robot Applications (TePRA), 2012 IEEE International Conference on, IEEE, p. 127–132. 49, 54

GERS, F. A., J. SCHMIDHUBER et F. CUMMINS. 1999, «Learning to forget : Continual prediction with lstm», . 42

GIRSHICK, R. 2015, «Fast r-cnn», dans Proceedings of the IEEE international conference on computer vision, p. 1440–1448. 46, 47, 48

GIRSHICK, R., J. DONAHUE, T. DARRELL et J. MALIK. 2014, «Rich feature hierarchies for accurate object detection and semantic segmentation», dans Proceedings of the IEEE conference on computer vision and pattern recognition, p. 580–587. 46, 48, 54

GRAHAM, B. 2014, «Fractional max-pooling», arXiv preprint arXiv :1412.6071. 45

GUPTA, S., R. GIRSHICK, P. ARBELÁEZ et J. MALIK. 2014, «Learning rich features from rgb-d images for object detection and segmentation», dans European Conference on Computer Vision, Springer, p. 345–360. 31, 54, 56

GYBENKO, G. 1989, «Approximation by superposition of sigmoidal functions», Mathematics of Control, Signals and Systems, vol. 2, nº 4, p. 303–314. 32

HAALA, N., C. BRENNER et K.-H. ANDERS. 1998, «3D urban GIS from laser altimeter and 2D map data», International Archives Photogramm. Remote Sens., vol. 32, p. 339–346. 49

HACKEL, T., N. SAVINOV, L. LADICKY, J. D. WEGNER, K. SCHINDLER et M. POLLEFEYS. 2017, «Semantic3d.net : A new large-scale point cloud classification benchmark», ISPRS Annals of Photogrammetry, Remote Sensing and Spatial Information Science, vol. IV-1/W1. 52

HANDA, A., V. PATRAUCEAN, V. BADRINARAYANAN, S. STENT et R. CIPOLLA. 2016, «Understanding real world indoor scenes with synthetic data», dans Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, p. 4077–4085. 55

HASSIBI, B. et D. G. STORK. 1993, «Second order derivatives for network pruning : Optimal brain surgeon», dans Advances in neural information processing systems, p. 164–171. 36

HAZIRBAS, C., L. MA, C. DOMOKOS et D. CREMERS. 2016, «Fusenet : Incorporating depth into semantic segmentation via fusion-based cnn architecture», dans Asian Conference on Computer Vision, Springer, p. 213–228. 55, 56

HE, K., G. GKIOXARI, P. DOLLÁR et R. GIRSHICK. 2017, «Mask r-cnn», arXiv preprint arXiv :1703.06870. 52

HE, K., X. ZHANG, S. REN et J. SUN. 2016a, «Deep residual learning for image recognition», dans Proceedings of the IEEE conference on computer vision and pattern recognition, p. 770–778. 38, 40, 41, 45

HE, K., X. ZHANG, S. REN et J. SUN. 2016b, «Identity mappings in deep residual networks», dans European Conference on Computer Vision, Springer, p. 630–645. 41

HOCHREITER, S., Y. BENGIO, P. FRASCONI et J. SCHMIDHUBER. 2001, «Gradient flow in recurrent nets : the difficulty of learning long-term dependencies», . 41, 42

HOSANG, J., R. BENENSON et B. SCHIELE. 2017, «Learning non-maximum suppression», arXiv preprint arXiv :1705.02950. 53

HUBER, P. 1981, «J. 1981. robust statistics», New York : John Wiley. 33

IANDOLA, F. N., M. W. MOSKEWICZ, K. ASHRAF, S. HAN, W. J. DALLY et K. KEUTZER. 2016, «Squeezenet : Alexnet-level accuracy with 50x fewer parameters and< 1mb model size», arXiv preprint arXiv :1602.07360. 40, 41

IOFFE, S. et C. SZEGEDY. 2015, «Batch normalization : Accelerating deep network training by reducing internal covariate shift», dans International Conference on Machine Learning, p. 448–456. 39

KATAOKA, H., K. IWATA et Y. SATOH. 2015, «Feature evaluation of deep convolutional neural networks for object recognition and detection», arXiv preprint arXiv :1509.07627. 42

KAWAGUCHI, K. 2016, «Deep learning without poor local minima», dans Advances in Neural Information Processing Systems, p. 586–594. 36

KELLEY, H. J. 1960, «Gradient theory of optimal flight paths», Ars Journal, vol. 30, nº 10, p. 947–954. 35

KENDALL, A., V. BADRINARAYANAN et R. CIPOLLA. 2015, «Bayesian segnet : Model uncertainty in deep convolutional encoder-decoder architectures for scene understanding», arXiv preprint arXiv :1511.02680. 51

KRIZHEVSKY, A. et G. HINTON. 2009, «Learning multiple layers of features from tiny images», . 37, 38, 39, 45

KRIZHEVSKY, A., I. SUTSKEVER et G. E. HINTON. 2012, «Imagenet classification with deep convolutional neural networks», dans Advances in neural information processing systems, p. 1097–1105. 32, 41

LAI, K., L. BO et D. FOX. 2014, «Unsupervised feature learning for 3D scene labeling», dans ICRA, IEEE, p. 3050–3057. 52

LECUN, Y. 1987, Modèles connexionnistes de l'apprentissage, thèse de doctorat, These de Doctorat, Universite Paris 6. 32

LECUN, Y., L. BOTTOU, Y. BENGIO et P. HAFFNER. 1998, «Gradient-based learning applied to document recognition», Proc. of the IEEE, vol. 86, nº 11, p. 2278–2324. 35, 39, 41

LECUN, Y., J. S. DENKER et S. A. SOLLA. 1990, «Optimal brain damage», dans Advances in Neural Information Processing Systems 2, édité par D. S. Touretzky, Morgan-Kaufmann, p. 598–605. URL http://papers.nips.cc/paper/250-optimal-brain-damage.pdf. 36

LENZ, I., H. LEE et A. SAXENA. 2015, «Deep learning for detecting robotic grasps», The International Journal of Robotics Research, vol. 34, nº 4-5, p. 705–724. 54

LIN, G., C. SHEN, A. VAN DEN HENGEL et I. REID. 2017, «Exploring context with deep structured models for semantic segmentation», IEEE Transactions on Pattern Analysis and Machine Intelligence. 52, 53

LIU, W., D. ANGUELOV, D. ERHAN, C. SZEGEDY, S. REED, C.-Y. FU et A. C. BERG. 2016, «Ssd : Single shot multibox detector», dans European conference on computer vision, Springer, p. 21–37. 48, 49

LONG, J., E. SHELHAMER et T. DARRELL. 2015, «Fully convolutional networks for semantic segmentation», dans Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, p. 3431–3440. 49, 50, 51, 54, 56

MA, L., J. STUECKLER, C. KERL et D. CREMERS. 2017, «Multi-view deep learning for consistent semantic mapping with rgb-d cameras», dans arXiv :1703.08866, p. –. 55, 56

MATURANA, D. et S. SCHERER. 2015, «Voxnet : A 3D convolutional neural network for real-time object recognition», dans IROS, Hamburg, Germany, p. 922–928. 52

MEES, O., A. EITEL et W. BURGARD. 2016, «Choosing smartly : Adaptive multi-modal fusion for object detection in changing environments», dans Intelligent Robots and Systems (IROS), 2016 IEEE/RSJ International Conference on, IEEE, p. 151–156. 54, 55, 56

QI, C. R., L. YI, H. SU et L. J. GUIBAS. 2017, «Pointnet++ : Deep hierarchical feature learning on point sets in a metric space», arXiv preprint arXiv :1706.02413. 53

RAJ, A., D. MATURANA et S. SCHERER. 2015, «Multi-scale convolutional architecture for semantic segmentation», . 55

RAZAKARIVONY, S. et F. JURIE. 2014, «Autoencodeurs discriminants pour la détection de cibles faiblement résolues», dans Reconnaissance de formes et intelligence artificielle (RFIA) 2014. 51

REDMON, J., S. DIVVALA, R. GIRSHICK et A. FARHADI. 2016, «You only look once : Unified, real-time object detection», dans Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, p. 779–788. 48, 49

REDMON, J. et A. FARHADI. 2016, «Yolo9000 : better, faster, stronger», arXiv preprint arXiv :1612.08242. 48

REN, S., K. HE, R. GIRSHICK et J. SUN. 2015, «Faster r-cnn : Towards real-time object detection with region proposal networks», dans Advances in neural information processing systems, p. 91–99. 47, 48

RONNEBERGER, O., P. FISCHER et T. BROX. 2015, «U-Net : Convolutional networks for biomedical image segmentation», dans MICCAI, Munich, ISBN 978-3-319-24574-4, p. 234–241, doi :10.1007/978-3-319-24574-4_28. URL http://dx.doi.org/10.1007/978-3-319-24574-4_28. 51, 53

RUMELHART, D. E., G. E. HINTON, R. J. WILLIAMS et al.. 1988, «Learning representations by back-propagating errors», Cognitive modeling, vol. 5, n° 3, p. 1. 35

RUSSAKOVSKY, O., J. DENG, H. SU, J. KRAUSE, S. SATHEESH, S. MA, Z. HUANG, A. KARPATHY, A. KHOSLA, M. BERNSTEIN, A. C. BERG et L. FEI-FEI. 2015, «ImageNet Large Scale Visual Recognition Challenge», International Journal of Computer Vision (IJCV), vol. 115, n° 3, doi :10.1007/s11263-015-0816-y, p. 211–252. 45, 46

SERMANET, P., D. EIGEN, X. ZHANG, M. MATHIEU, R. FERGUS et Y. LECUN. 2013, «Overfeat : Integrated recognition, localization and detection using convolutional networks», arXiv preprint arXiv :1312.6229. 46

SFIKAS, K., T. THEOHARIS et I. PRATIKAKIS. 2017, «Exploiting the PANORAMA Representation for Convolutional Neural Network Classification and Retrieval», dans Eurographics Workshop on 3D Object Retrieval, Lyon, France, p. –. 53

SHRIVASTAVA, A., A. GUPTA et R. GIRSHICK. 2016, «Training region-based object detectors with online hard example mining», dans Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, p. 761–769. 48

SILBERMAN, N., D. HOIEM, P. KOHLI et R. FERGUS. 2012, «Indoor segmentation and support inference from rgbd images», Computer Vision–ECCV 2012, p. 746–760. 54

SIMONYAN, K. et A. ZISSERMAN. 2014, «Very deep convolutional networks for large-scale image recognition», arXiv preprint arXiv :1409.1556. 33, 40, 41

SMITHSON, S. C., G. YANG, W. J. GROSS et B. H. MEYER. 2016, «Neural networks designing neural networks : Multi-objective hyper-parameter optimization», dans Computer-Aided Design (ICCAD), 2016 IEEE/ACM International Conference on, IEEE, p. 1–8. 43

STEDER, B., G. GRISETTI, M. VAN LOOCK et W. BURGARD. 2009, «Robust on-line model-based object detection from range images», dans Intelligent Robots and Systems, 2009. IROS 2009. IEEE/RSJ International Conference on, IEEE, p. 4739–4744. 56

SU, H., , C. QI, K. MO et L. GUIBAS. 2017, «Pointnet : Deep learning on point sets for 3d classification and segmentation», dans CVPR, Honolulu, Hawai, USA, p. –. 42, 53

SU, H., S. MAJI, E. KALOGERAKIS et E. LEARNED-MILLER. 2015, «Multi-view convolutional neural networks for 3D shape recognition», dans ICCV, p. 945–953. 53

SZEGEDY, C., S. IOFFE, V. VANHOUCKE et A. A. ALEMI. 2017, «Inception-v4, inception-resnet and the impact of residual connections on learning.», dans AAAI, p. 4278–4284. 43

SZEGEDY, C., W. LIU, Y. JIA, P. SERMANET, S. REED, D. ANGUELOV, D. ERHAN, V. VANHOUCKE et A. RABINOVICH. 2015, «Going deeper with convolutions», dans Proceedings of the IEEE conference on computer vision and pattern recognition, p. 1–9. 41

TAI, L., Q. YE et M. LIU. 2016, «Pca-aided fully convolutional networks for semantic segmentation of multi-channel fmri», arXiv preprint arXiv :1610.01732. 50

UIJLINGS, J., K. VAN DE SANDE, T. GEVERS et A. SMEULDERS. 2013, «Selective search for object recognition», International journal of computer vision. 46

VALADA, A., G. L. OLIVEIRA, T. BROX et W. BURGARD. 2016, «Deep multispectral semantic scene understanding of forested environments using multi-modal fusion», dans International Symposium on Experimental Robotics, Springer, p. 465–477. 56

WAHL, E., U. HILLENBRAND et G. HIRZINGER. 2003, «Surflet-pair-relation histograms : a statistical 3d-shape representation for rapid classification», dans 3-D Digital Imaging and Modeling, 2003. 3DIM 2003. Proceedings. Fourth International Conference on, IEEE, p. 474–481. 54

WAN, L., M. ZEILER, S. ZHANG, Y. L. CUN et R. FERGUS. 2013, «Regularization of neural networks using dropconnect», dans Proceedings of the 30th international conference on machine learning (ICML-13), p. 1058–1066. 45

WIKIPEDIA. 2017, «Réseau neuronal convolutif — Wikipédia, l'encyclopédie libre», URL https://goo.gl/jQfwmi, [En ligne ; Page disponible le 12-août-2017]. 33, 39

WU, Z., S. SONG, A. KHOSLA, F. YU, L. ZHANG, X. TANG et J. XIAO. 2015, «3D shapenets : A deep representation for volumetric shapes», dans CVPR, Boston, USA, p. 1912–1920. 52

YANG, W., Y. WANG, A. VAHDAT et G. MORI. 2012, «Kernel latent svm for visual recognition», dans Advances in neural information processing systems, p. 809–817. 45

ZHAO, H., J. SHI, X. QI, X. WANG et J. JIA. 2016, «Pyramid scene parsing network», arXiv preprint arXiv :1612.01105. 41

Deuxième partie

Contributions

Chapitre 4

Sélection d'algorithmes de classification par réseau de neurones

> *Choisir ! c'est l'éclair de l'intelligence. Hésitez-vous ?*
>
> Honoré de Balzac, L'Illustre Gaudissart, II, 1833

Sommaire

4.1 Introduction 66
4.2 Preuve de concept 68
4.2.1 La méthode 68
4.2.2 Résultats 69
4.3 Cas 1 : usage de plusieurs jeux de données 72
4.3.1 Les méthodes 72
4.3.2 Les résultats par méthodes 73
4.3.3 Réseau de confiance 75
4.4 Cas 2 : usage d'un jeu de données unique 76
4.4.1 Approche 1 : en partant des méthodes 77
4.4.2 Approche 2 : en partant des données 80
4.5 Conclusion du chapitre 82
4.6 Références 83

4.1 Introduction

Pour permettre aux robots d'évoluer dans un environnement et d'y interagir il est nécessaire de leur permettre de reconnaître les objets qui les entourent. Pour cela la vision par ordinateur a massivement recours à l'apprentissage statistique (*machine learning*) qui consiste à entraîner le robot en lui présentant une grande quantité d'images-exemples des objets à reconnaître. Face aux très nombreuses méthodes existantes, choisir la méthode la plus efficace dans un contexte donné peut se révéler difficile. De plus, pour un même contexte, différentes méthodes peuvent être complémentaires. Il peut donc être intéressant d'essayer de tirer parti de plusieurs méthodes simultanément.

Robustesse par fusion Pour faire face à la haute variabilité des objets qu'un robot peut rencontrer (variabilité intra-classe ou de conditions d'observations) il est possible de recourir à de nombreuses techniques de fusion (voir [A survey of image classification methods and techniques for improving classification performance LU et WENG, 2007]). Ces stratégies de fusion permettent d'améliorer les performances de classification, ce qui est notamment montré dans CARUANA et al. [2004] [1]. Cependant, elles imposent au minimum le coût de calcul cumulé de chaque méthode employée, alourdissant le processus.

Limitation mémoire Ces dernières années les approches par réseaux de neurones ont permis d'améliorer significativement les performances DENG et al. [2009]; KRIZHEVSKY et HINTON [2009] mais restent encore longues et délicates à entraîner et requièrent de grandes ressources de calcul. Même s'il est possible de charger en mémoire GPU plusieurs réseaux de neurones légers en même temps, la tendance est à augmenter le nombre de paramètres des réseaux. En effet, les réseaux se complexifient en structure et augmentent en profondeur, limitant souvent l'usage à un seul réseau à la fois, voire deux. Pour donner un ordre de grandeur, un ResNet [HE et al., 2016] à 152 couches avec un *batch* de taille 4 et des images en entrée de 224x224 pixels utilise 5Gb de mémoire GPU en phase d'entraînement. En phase de test ce besoin diminue mais reste de l'ordre du Gigabits, rendant possible d'utiliser en parallèle 5 d'entre eux sur une carte graphique de type NVIDIA GTX 1080 (2016) qui possède 8Gb de mémoire. Une autre évolution récente des travaux en apprentissage profond consiste à utiliser des images entières et avec une grande résolution. C'est le cas dans REDMON et FARHADI [2016] qui effectue de la détection d'objets. Ceci implique également un usage plus grand de mémoire GPU.

Bien que l'usage d'un réseau de neurones soit rapide une fois chargé dans le GPU pour faire de la classification (voir les temps de classification dans le Tableau 4.1), il faut plusieurs secondes pour charger les paramètres du réseau en mémoire la première fois. Ce temps de chargement des paramètres est particulièrement gênant si nous devons tester plusieurs méthodes à base de réseau de neurones pour prendre une seule décision dans des situations où le temps est critique (par exemple la détection de piétons pour une voiture autonome). A chaque changement de méthode il faudra attendre le temps de chargement en mémoire avant de pouvoir faire l'inférence.

Sélection plutôt que fusion Dans le cadre ce chapitre, nous abordons la question de savoir s'il est possible d'utiliser de nombreuses méthodes complémentaires sans avoir à toutes les tester pour économiser nos ressources. Pour cela nous nous demandons s'il n'est pas possible de définir pour une image donnée son affinité avec les méthodes disponibles.

1. CARUANA et al. traitent du regroupement de méthodes au sein d'une grande liste de méthodes disponibles pour améliorer les performances sur plusieurs tâches.

TABLEAU 4.1 – Temps de calcul pour différentes architectures de réseaux de neurones et erreur sur le challenge de classification d'*ImageNet* 2012 ILSVRC, Ce tableau est extrait du site https://github.com/jcjohnson/cnn-benchmarks où beaucoup plus de détails sont disponibles.

Réseau	Nombre de couches	Erreur (Top-1)	Erreur (Top-5)	Vitesse (ms)
AlexNet	8	42,90	19,80	14,56
Inception-V1	22	-	10,07	39,14
VGG-16	16	27,00	8,80	128,62
VGG-19	19	27,30	9,00	147,32
ResNet-18	18	30,43	10,76	31,54
ResNet-34	34	26,73	8,74	51,59
ResNet-50	50	24,01	7,02	103,58
ResNet-101	101	22,44	6,21	156,44
ResNet-152	152	22,16	6,16	217,91
ResNet-200	200	21,66	5,79	296,51

Pour définir l'affinité entre une image et des méthodes il faut projeter cette image dans un espace de représentation adéquat. Cependant, la grande variabilité des données et des méthodes rend cet espace difficile à définir. C'est pourquoi nous nous tournons vers l'usage d'un réseau de neurones artificiels. Ainsi nous pouvons utiliser ce réseau de neurones pour sélectionner la méthode qui présente la plus grande affinité avec l'image. En d'autres mots : comment utiliser un réseau de neurones pour choisir le meilleur classifieur au sein d'une liste de méthodes disponibles ? Ce concept est illustré dans la Figure 4.1. En particulier nous nous intéressons à l'usage de plusieurs réseaux de neurones pour effectuer la tâche de classification.

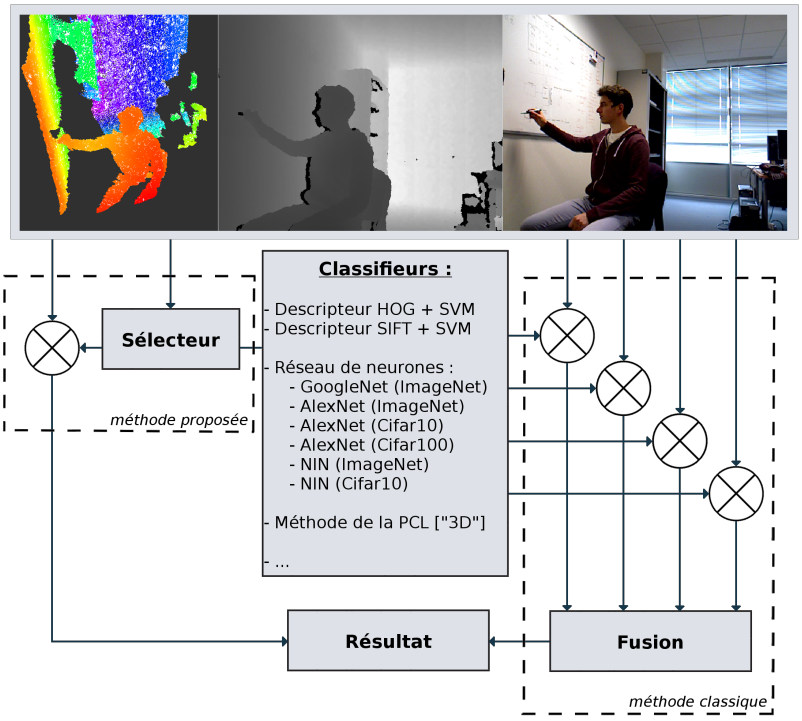


FIGURE 4.1 – Illustration de la stratégie globale de sélection. Notre exemple est composé avec une image *RGB-D* bien que nous n'en fassions pas usage dans ce chapitre.

4.2 Preuve de concept

Nous avons tout d'abord effectué une expérience d'implémentation sur un jeu de données de petite taille (SUN3D [XIAO et al., 2013]) afin de montrer les gains potentiels d'une telle approche.

4.2.1 La méthode

En pratique nous avons voulu faire de la classification de patchs dans des images *RGB* de SUN3D. Pour cela nous avons entraîné deux réseaux de neurones différents sur l'ensemble d'entraînement de SUN3D :

— Réseau 1 : LeNet [LECUN et al., 1998]

— Réseau 2 : *AlexNet* [KRIZHEVSKY et HINTON, 2009]

Ce choix a été guidé par la différence qui existe entre les deux méthodes : le LeNet est léger avec peu de paramètres, l'*AlexNet* possède beaucoup plus de paramètres et pouvait être implémenté sur notre configuration matérielle du moment.

Pour effectuer la sélection nous avons utilisé un troisième réseau : un *AlexNet* à nouveau. Son rôle est de prédire quelle est le meilleur réseau à utiliser (1 ou 2 ?) étant donnée l'image à classifier.

Nous séparons le processus en plusieurs phases :

Sur la partition d'entraînement :

1. Entraînement du réseau 1
2. Entraînement du réseau 2
3. Évaluation du réseau 1
4. Évaluation du réseau 2
5. Entraînement du réseau sélecteur avec les résultats des deux réseaux de classification (voir Figure 4.2(a))

Sur la partition de test

6. Test du réseau sélecteur (voir Figure 4.2(b))
7. Utilisation du réseau sélectionné par le réseau sélecteur pour chaque image

L'étape 5 demande de choisir sous quelle forme nous présentons la vérité terrain pour le réseau de confiance :

1. Vecteur binaire : 1 pour les méthodes correctes, 0 pour les méthodes incorrectes
2. Vecteur binaire exclusif : 1 pour la meilleure méthode, 0 pour les autres
3. Vecteur des scores de chaque réseau pour la classe de l'image

Nous reparlerons de la troisème possibilité dans la section suivante. En ce qui concerne les deux représentations binaires nous avons rejeté la représentation 2 "binaire exclusif". En effet, cette représentation sanctionne le sélecteur quand il ne propose pas la meilleure méthode alors que la proposition peut être tout de même valable. Nous utiliserons donc la première représentation avec la notation suivante que nous retrouvons dans la Figure 4.2 :

— image : $x_i \in \mathrm{X}$

— étiquette : $y_i \in \{0,1\}^n$ où n est le nombre de méthodes

Nous pouvons voir le réseau sélecteur comme un réseau de classification qui cherche à attribuer pour une image donnée une classe parmi la liste suivante : "Image qui peut-être classifiée par la méthode 1", "Image qui peut-être classifiée par la méthode 2", *etc*... Cependant ces classes ne sont pas exclusives : une image peut potentiellement être classifiée par plusieurs méthodes.

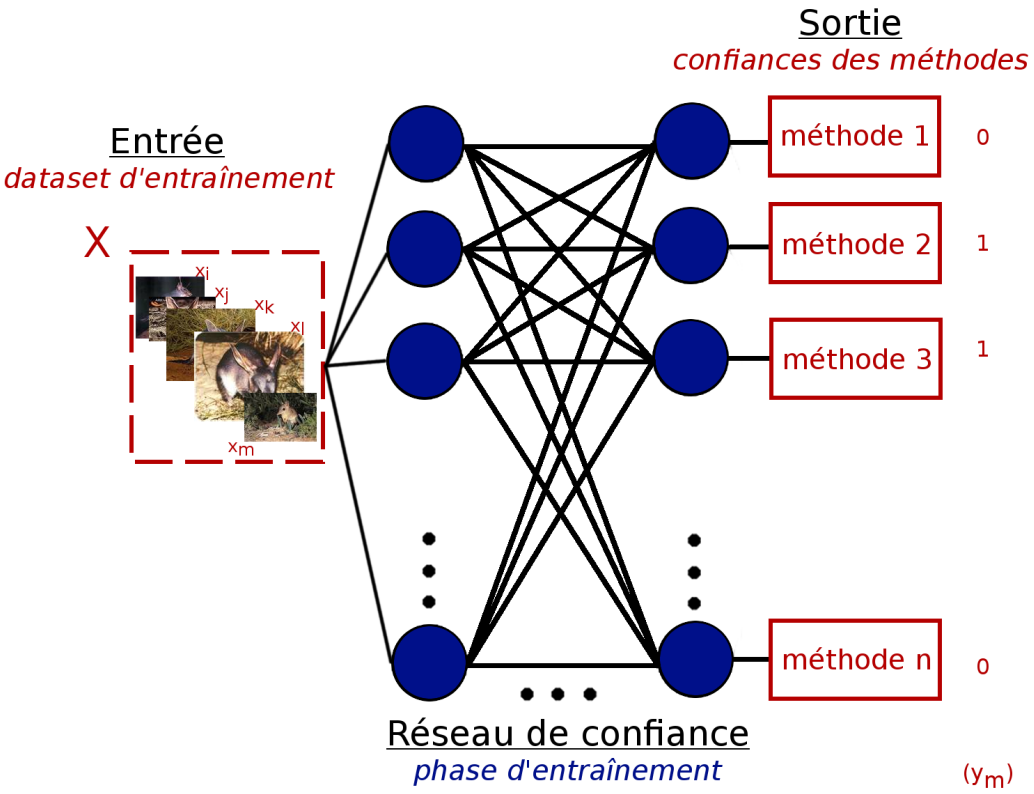
Nous devons également définir une fonction de perte pour entraîner le réseau sélecteur. Dans notre première implémentation, nous entraînons le réseau sélecteur avec des vecteurs binaires non-exclusifs et une fonction de perte de type "SoftMaxWithLoss" comme définie dans le Chapitre "État de l'art : Apprentissage profond". C'est donc une situation de regression logistique. C'est un fonctionnement classique de classifieur. De cette façon le réseau sélecteur apprend à prédire un vecteur contenant les probabilités que chaque méthode fonctionne. Nous pouvons ainsi voir le réseau sélecteur comme un réseau qui prédit une confiance en nos méthodes.

4.2.2 Résultats

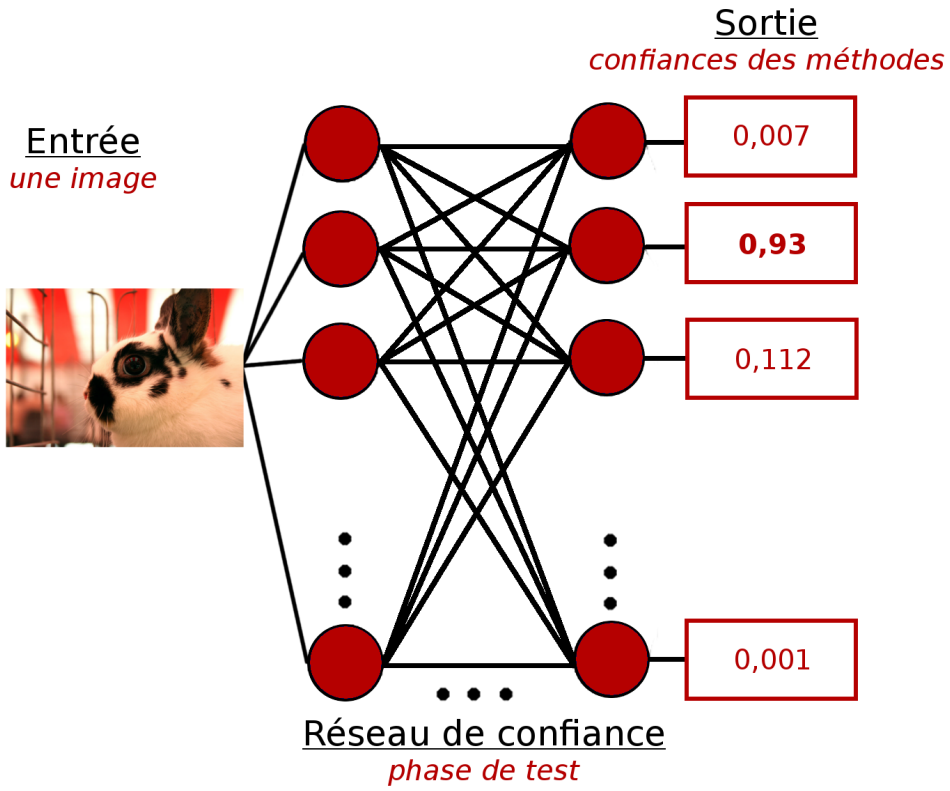
Les résultats sont illustrés dans la Figure 4.3. Comme nous pouvons le constater avec la Figure 4.3(c) les deux méthodes classifient toutes les deux 38,9% des images mais 12,9% des images ne sont classifiées que par *AlexNet* et 15,4% des images ne sont classifiées que par LeNet. Cette situation est particulièrement propice à l'usage d'une méthode de sélection ou de fusion.

Comme nous pouvons le voir dans la Figure 4.4(a), un oracle qui saurait choisir parfaitement quand utiliser *AlexNet* et quand utiliser LeNet pourrait atteindre une précision de 67,2%. Notre méthode atteint dans ces conditions une précision de 59,2%. Nous n'atteignons pas la performance optimale de l'oracle, mais nous améliorons la performance de classification de 4.9 points par rapport au réseau 1 (LeNet) et de 7,4 points par rapport au réseau 2 (*AlexNet*).

Dans cette situation, notre méthode s'avère efficace pour effectuer de la sélection de classifieurs. Nous présenterons dans la suite du chapitre des situations plus complexes de sélection (plus de 10 classifieurs disponibles, des jeux de données d'entraînements différents) ainsi que des approches différentes pour effectuer la sélection.

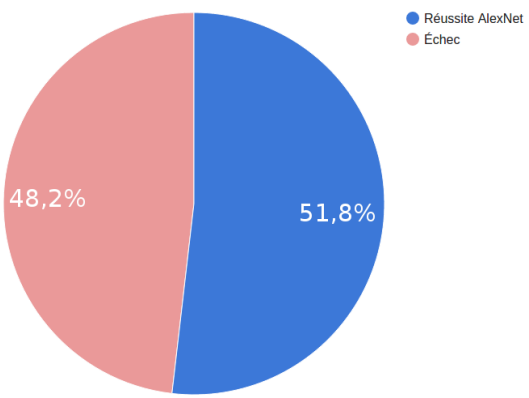


(a) Réseau de confiance à l'entraînement

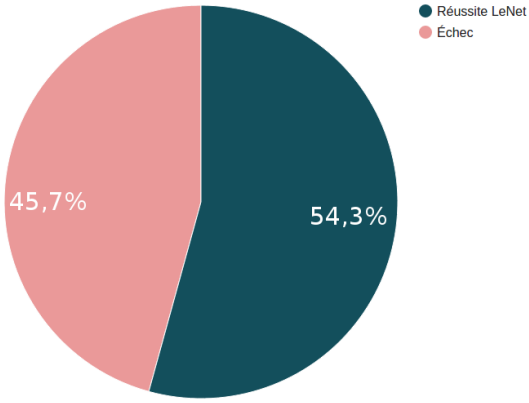


(b) Réseau de confiance au test

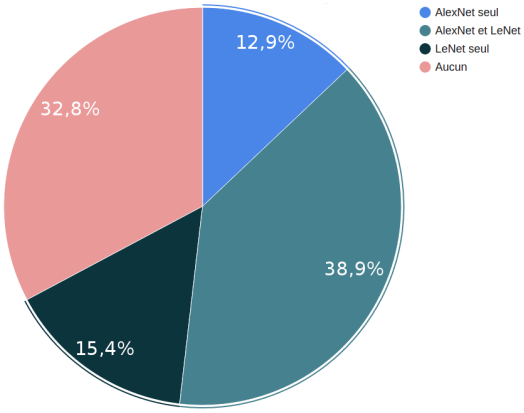
FIGURE 4.2 – Illustration du réseau sélecteur



(a) Performance du *AlexNet* sur SUN3D

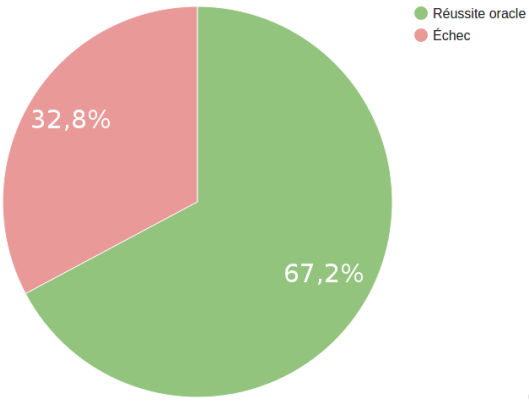


(b) Performance du LeNet sur SUN3D

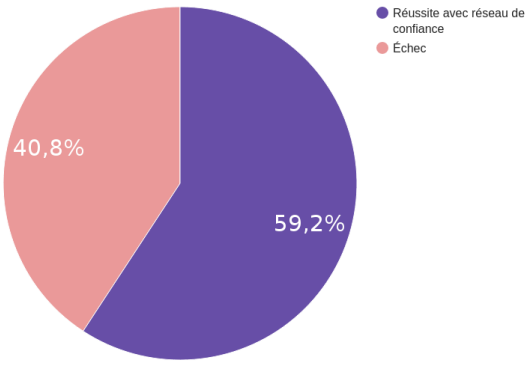


(c) Complémentarité des méthodes sur SUN3D

FIGURE 4.3 – Performances de *AlexNet* et LeNet sur SUN3D



(a) Performance de l'oracle sur SUN3D



(b) Performance du réseau de confiance sur SUN3D

FIGURE 4.4 – Comparaison de la fusion par sélection avec l'oracle

4.3 Cas 1 : usage de plusieurs jeux de données

Nous nous intéressons ici à une situation particulière de sélection de méthodes : nous utilisons un ensemble de méthodes différentes qui ont été entraînées sur des jeux de données différents avec des ensembles de classes différents et des encodages de l'information différents (*RGB* ou nuances de gris). Cette expérience a plusieurs objectifs : voir si un réseau sélecteur peut travailler sur un ensemble de jeux de données hétérogènes, voir si le réseau sélecteur est capable d'utiliser un ensemble de méthodes de types variés (HOG+SVM *versus* *CNN*) et par déduction : est-il possible d'effectuer du transfert d'apprentissage seulement par sélection ?

4.3.1 Les méthodes

Nous utilisons les méthodes suivantes :

1. **svm_cifar10** : Architecture HOG+SVM entraînée sur le jeu de données CIFAR10
2. **svm_cifar100** : Architecture HOG+SVM entraînée sur le jeu de données CIFAR100
3. **svm_caltech101** : Architecture HOG+SVM entraînée sur le jeu de données Caltech101
4. **svm_caltech256** : Architecture HOG+SVM entraînée sur le jeu de données Caltech256
5. **svm_sun3d** : Architecture HOG+SVM entraînée sur le jeu de données SUN3D
6. **svm_mountain_forest** : Architecture HOG+SVM entraînée sur le jeu de données Mountain/Forest [2]
7. **bvlc_*AlexNet*** : Architecture *CNN* *AlexNet* entraînée sur le jeu de données *ImageNet*
8. **bvlc_*GoogleNet*** : Architecture *CNN* *GoogleNet* entraînée sur le jeu de données *ImageNet*
9. ***VGG*_cnn_s** : Architecture *CNN* *VGG*11 entraînée sur le jeu de données *ImageNet*
10. ***SqueezeNet*** : Architecture *CNN* *SqueezeNet* entraînée sur le jeu de données *ImageNet*
11. **ln_cifar10_gs** : Architecture *CNN* LeNet entraînée sur le jeu de données CIFAR10 en nuance de gris (*greyscale*)
12. **ln_cifar10_rgb** : Architecture *CNN* LeNet entraînée sur le jeu de données CIFAR10
13. **an_cifar10_gs** : Architecture *CNN* *AlexNet* entraînée sur le jeu de données CIFAR10 en nuance de gris (*greyscale*)
14. **an_cifar10_rgb** : Architecture *CNN* *AlexNet* entraînée sur le jeu de données CIFAR10
15. **ln_cifar100_gs** : Architecture *CNN* LeNet entraînée sur le jeu de données CIFAR100 en nuance de gris (*greyscale*)
16. **ln_caltech101_gs** : Architecture *CNN* LeNet entraînée sur le jeu de données Caltech101 en nuance de gris (*greyscale*)
17. **ln_caltech256_gs** : Architecture *CNN* LeNet entraînée sur le jeu de données Caltech256 en nuance de gris (*greyscale*)
18. **ln_sun3d_gs** : Architecture *CNN* LeNet entraînée sur le jeu de données SUN3D en nuance de gris (*greyscale*)
19. **an_sun3d_rgb** : Architecture *CNN* *AlexNet* entraînée sur le jeu de données SUN3D

2. Petit jeu de données de classification avec deux classes : des forêts et des montagnes

Cette liste est composée de méthode à base de *CNNs* ainsi que de méthodes à base de HOG+SVM pour varier les styles de méthodes disponibles et espérer une certaine complémentarité au moment de la classification par sélection. De plus, une seule méthode ne sera pas capable de classifier toutes les données de tous les ensembles de test. Par exemple, les méthodes entraînées sur CIFAR10 ne connaissent même pas les classes de SUN3D.

Les données utilisées pour l'entraînement correspondent aux partitions d'entraînement des différents jeux de données. Les méthodes sont testées sur toutes les partitions de test après transformation des images au format utilisé par les méthodes. Par exemple la méthode an_cifar10_gs a été entraînée sur la partition d'entraînement de CIFAR10 convertie en nuance de gris. Elle sera testée par exemple sur la partition de test de Caltech256 également convertie en nuance de gris et redimensionnée à la taille d'entrée du réseau.

4.3.2 Les résultats par méthodes

Nous proposons ici une analyse des résultats de classification en trois points :

— Comment les méthodes réagissent sur les différents jeux de données ?
— Comment les méthodes réagissent face aux différentes classes ?
— Comment les méthodes se complètent sur l'ensemble des données ?

Corrélation méthode/jeu de données

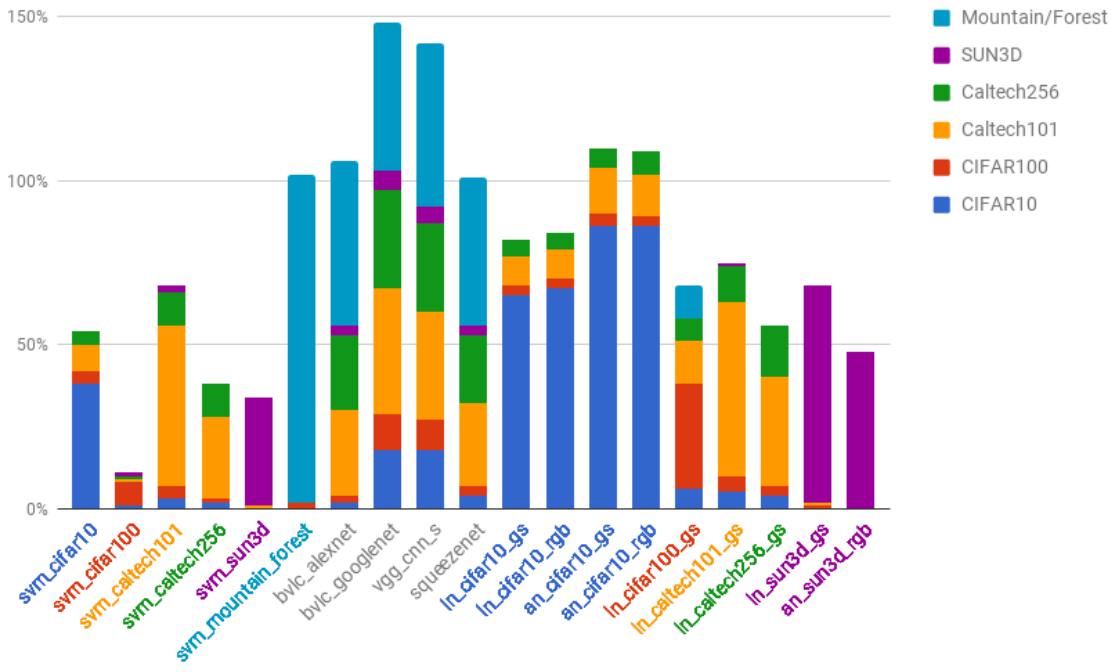


FIGURE 4.5 – Précisions cumulées des méthodes sur les jeux de test

Le graphique de la Figure 4.5 présente les réussites de classification cumulées des méthodes sur chaque jeu de données disponibles. Nous avons coloré les noms des méthodes en fonction des données d'entraînement utilisées.

— La première constatation était prévisible : les méthodes fonctionnent mieux sur les jeux de données observés à l'entraînement (c'est particulièrement visible avec les méthodes {ln,an}_cifar10_{gs,rgb}.)

— Les méthodes *CNN* ont de meilleurs performances que les méthodes HOG+SVM entraînées sur les mêmes données (voir méthode svm_cifar_100 *versus* méthode ln_-cifar100_gs).

— Les méthodes qui utilisent la couleur sont légèrement meilleures que les méthodes utilisant l'image en nuances de gris (voir méthodes ln_cifar10_{gs,rgb})

— Les méthodes entraînées sur *ImageNet* (écrites en gris) sont polyvalentes et performent mieux que toutes les autres méthodes.

Corrélation méthode/classe

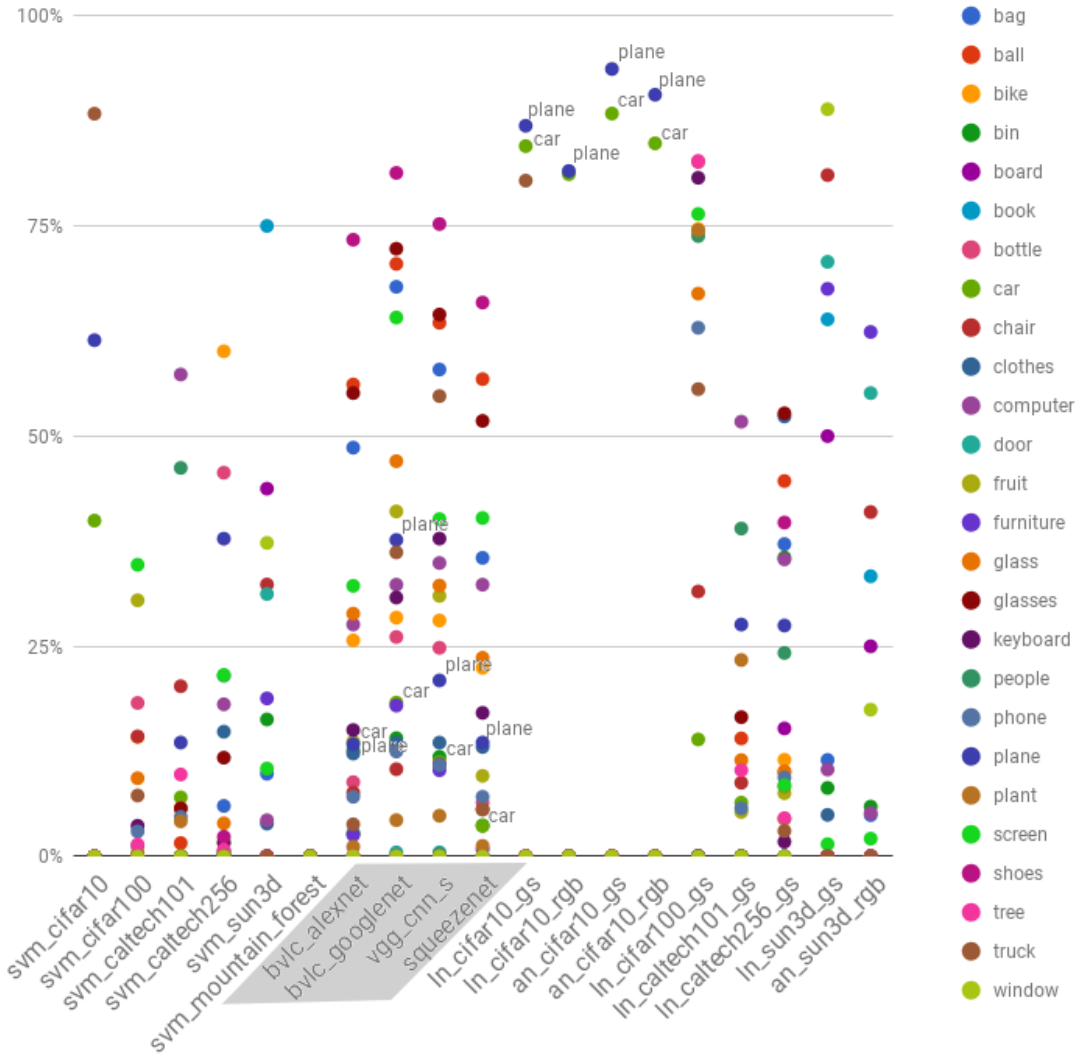


FIGURE 4.6 – Précisions des méthodes sur les jeux de test pour plusieurs classes sélectionnées

Le graphique de la Figure 4.6 présente les réussites de classification des méthodes sur une sélection de classes disponibles.

— Comme nous avions déjà pu le voir sur la Figure 4.5 les méthodes entraînées sur *ImageNet* sont polyvalentes et parviennent à classifier de nombreuses classes (voir bvlc_*AlexNet*, bvlc_*GoogleNet*, *VGG*_cnn_s et *SqueezeNet* sur la Figure 4.6).

— Les méthodes entraînées sur CIFAR10 obtiennent de meilleurs performances pour les classes "*plane*" et "*car*" que les méthodes entraînées sur *ImageNet*.

Ces deux premiers points illustrent le compromis polyvalence/spécialisation que nous rencontrons lorsque nous construisons notre ensemble de méthodes.

Corrélation méthode/méthode

TABLEAU 4.2 – Pourcentage de bonnes classifications communes entre les méthodes. Le tableau se lit "La méthode A dans la colonne de gauche a également classifié XX des images classifiées par la méthode B en tête de colonne." (Les colorisations rouge et bleue servent dans l'analyse du tableau.)

	svm_cifar10	svm_cifar100	svm_caltech101	svm_caltech256	svm_sun3d	svm_mountain_forest	bvlc_*AlexNet*	bvlc_*GoogleNet*	*VGG*_cnn_s	*SqueezeNet*
svm_cifar10	100,0%	9,1%	11,6%	27,8%	0,0%	0,0%	14,1%	27,7%	30,8%	16,5%
svm_cifar100	1,9%	100,0%	3,6%	4,5%	4,6%	5,5%	4,5%	4,3%	4,7%	4,4%
svm_caltech101	4,9%	7,1%	100,0%	50,1%	4,3%	0,0%	16,3%	10,6%	10,0%	15,4%
svm_caltech256	7,4%	5,7%	31,8%	100,0%	3,0%	0,0%	15,1%	10,3%	8,6%	14,7%
svm_sun3d	0,0%	1,4%	0,7%	0,7%	100,0%	0,0%	1,3%	0,8%	0,7%	0,9%
svm_mountain_forest	0,0%	1,6%	0,0%	0,0%	0,0%	100,0%	2,7%	1,0%	1,1%	2,3%
bvlc_*AlexNet*	6,2%	9,5%	17,1%	25,1%	8,8%	20,0%	100,0%	33,5%	35,8%	59,4%
bvlc_*GoogleNet*	30,7%	23,0%	28,2%	43,0%	12,8%	19,1%	84,7%	100,0%	67,8%	77,1%
*VGG*_cnn_s	30,6%	22,0%	23,7%	32,3%	10,5%	18,5%	81,0%	60,8%	100,0%	70,9%
SqueezeNet	7,7%	9,8%	17,2%	26,1%	6,6%	17,7%	63,4%	32,6%	33,4%	100,0%
ln_cifar10_gs	89,3%	8,0%	20,0%	26,9%	0,0%	0,0%	19,4%	41,0%	43,3%	26,7%
ln_cifar10_rgb	88,1%	8,3%	18,7%	24,9%	0,0%	0,0%	18,3%	40,1%	42,3%	25,8%
an_cifar10_gs	92,7%	8,4%	23,2%	30,8%	0,0%	0,0%	23,8%	44,2%	46,4%	31,1%
an_cifar10_rgb	91,4%	8,4%	22,5%	30,4%	0,0%	0,0%	22,4%	43,1%	45,3%	29,7%
ln_cifar100_gs	18,8%	86,5%	33,7%	27,1%	17,8%	60,5%	19,0%	28,0%	30,2%	21,4%
ln_caltech101_gs	8,0%	5,8%	60,2%	54,4%	3,7%	0,0%	25,1%	17,5%	16,4%	23,2%
ln_caltech256_gs	8,6%	6,3%	35,7%	59,8%	1,8%	0,0%	30,6%	18,9%	18,3%	27,5%
ln_sun3d_gs	0,0%	2,8%	2,3%	0,1%	75,6%	0,0%	1,4%	0,8%	0,8%	0,8%
an_sun3d_rgb	0,0%	1,1%	1,0%	0,1%	47,5%	0,0%	0,9%	0,5%	0,5%	0,5%

	ln_cifar10_gs	ln_cifar10_rgb	an_cifar10_gs	an_cifar10_rgb	ln_cifar100_gs	ln_caltech101_gs	ln_caltech256_gs	ln_sun3d_gs	an_sun3d_rgb
svm_cifar10	41,1%	41,2%	40,6%	40,8%	8,7%	13,5%	13,4%	0,0%	0,0%
svm_cifar100	0,8%	0,8%	0,8%	0,8%	8,4%	2,1%	2,1%	4,8%	3,1%
svm_caltech101	3,8%	3,7%	4,3%	4,2%	6,5%	42,4%	23,1%	8,0%	5,7%
svm_caltech256	3,3%	3,1%	3,6%	3,6%	3,3%	24,3%	24,5%	0,3%	0,4%
svm_sun3d	0,0%	0,0%	0,0%	0,0%	0,5%	0,4%	0,2%	40,8%	41,2%
svm_mountain_forest	0,0%	0,0%	0,0%	0,0%	1,7%	0,0%	0,0%	0,0%	0,0%
bvlc_*AlexNet*	3,9%	3,7%	4,6%	4,4%	3,9%	18,6%	20,8%	5,2%	5,2%
bvlc_*GoogleNet*	20,9%	20,8%	21,5%	21,3%	14,3%	32,8%	32,5%	7,5%	7,4%
*VGG*_cnn_s	19,8%	19,7%	20,2%	20,1%	13,9%	27,5%	28,2%	6,6%	6,7%
SqueezeNet	5,8%	5,6%	6,4%	6,2%	4,6%	17,7%	20,0%	3,1%	3,3%
ln_cifar10_gs	100,0%	94,0%	91,1%	91,2%	8,6%	26,6%	19,7%	0,0%	0,0%
ln_cifar10_rgb	92,7%	100,0%	89,8%	90,4%	8,6%	25,4%	18,4%	0,0%	0,0%
an_cifar10_gs	95,8%	95,8%	100,0%	96,3%	9,2%	29,7%	22,0%	0,0%	0,0%
an_cifar10_rgb	94,1%	94,6%	94,5%	100,0%	8,7%	28,5%	21,3%	0,0%	0,0%
ln_cifar100_gs	8,6%	8,7%	8,7%	8,5%	100,0%	27,5%	19,5%	15,4%	12,5%
ln_caltech101_gs	7,3%	7,0%	7,7%	7,5%	7,5%	100,0%	31,7%	4,2%	3,7%
ln_caltech256_gs	5,8%	5,5%	6,2%	6,1%	5,8%	34,5%	100,0%	0,8%	0,8%
ln_sun3d_gs	0,0%	0,0%	0,0%	0,0%	0,9%	0,9%	0,2%	100,0%	80,0%
an_sun3d_rgb	0,0%	0,0%	0,0%	0,0%	0,4%	0,5%	0,1%	49,8%	100,0%

Le Tableau 4.2 permet d'observer la corrélation entre plusieurs méthodes. Par exemple la méthode ln_cifar10_gs classifie correctement 89,3% des images qui sont correctement classifiées par la méthode svm_cifar10. Cette corrélation est due au même jeu de données d'entraînement utilisé pour l'apprentissage. Réciproquement, la méthode svm_cifar10 classifie correctement 41,1% des images qui sont correctement classifiées par la méthode ln_cifar10_gs. Cela signifie que malgré l'usage du même jeu d'entraînement ces méthodes sont complémentaires. En effet, aucune des méthodes ne classifie 100% des bonnes classifications de l'autre méthode.

4.3.3 Réseau de confiance

Au vu des complémentarités des différentes méthodes illustrées dans la section précédente il nous semble envisageable d'appliquer notre stratégie de sélection de méthode.

Nous avons entraîné plusieurs architectures différentes (basées sur *LeNet, AlexNet et SqueezeNet*) pour prédire quelle méthode utiliser pour une image donnée. L'ensemble d'entraînement du réseau de confiance est l'union des partitions d'entraînements des jeux de

données [3]. Nous avons appelé ces architectures des réseaux de confiance (ou RCs). Nous proposons les architectures des différents RCs en annexe (Figure A.1 (page I), Figure A.2 (page II), Figure A.3 (page III))

TABLEAU 4.3 – Evaluation de la précision moyenne des méthodes sur l'ensemble des jeux de données. L'oracle désigne la méthode résultant de la sélection du meilleur classifieur disponible à chaque image. Les réseaux de confiance (ou RCs) désignent les différentes architectures utilisées pour prédire le meilleur classifieur à utiliser pour une image donnée.

Méthode	Précision moyenne
Oracle	**77.8%**
RC : *SqueezeNet* 256x256	**43.0%**
RC : *AlexNet* 256x256	23.7%
RC : LeNet 32x32	36.3%
svm_cifar10	9.0%
svm_cifar100	1.8%
svm_caltech101	11.3%
svm_caltech256	6.3%
svm_sun3d	5.7%
svm_mountain_forest	17.0%
bvlc_*AlexNet*	17.7%
bvlc_*GoogleNet*	**24.7%**
*VGG*_cnn_s	23.7%
SqueezeNet	16.8%
ln_cifar10_gs	13.7%
ln_cifar10_rgb	14.0%
an_cifar10_gs	18.3%
an_cifar10_rgb	18.2%
ln_cifar100_gs	11.3%
ln_caltech101_gs	12.5%
ln_caltech256_gs	9.3%
ln_sun3d_gs	11.3%
an_sun3d_rgb	8.0%

Comme nous pouvons le voir dans le Tableau 4.3, le réseau de confiance basé sur l'architecture *CNN SqueezeNet* permet d'atteindre une précision moyenne de 43%. Bien que 34,8 points derrière un oracle théorique cette méthode nous permet de dépasser le bvlc_*GoogleNet* et sa moyenne de 24,7%. Comme expliqué dans le Chapitre "État de l'art : Apprentissage profond", le *SqueezeNet* est une architecture très légère ce qui rend particulièrement intéressant ce résultat : le réseau de confiance ne demande pas trop de ressources et permet un gain conséquent en précision moyenne.

4.4 Cas 2 : usage d'un jeu de données unique

Dans le Cas 1, le réseau sélecteur pouvait trouver des différences entre les jeux de données pour décider quelle méthode utiliser. Dans le Cas 2, nous utilisons un seul jeu de données, le

3. Les classes à prédire correspondent à l'union des classes de tous les datasets. Les réseaux entraînés sur ImageNet peuvent proposer 1000 classes différentes mais seulement un sous-ensemble est concerné dans notre cas.

jeu de données *ImageNet*. Les données sont statistiquement plus proches que dans le Cas 1 et surtout les méthodes que nous utilisons sont adaptées à ces données car entraînées dessus. Dans cette situation plus subtile, l'usage d'un réseau sélecteur est-il toujours possible ? Sa tâche devient plus complexe : il doit trouver des sous-espaces latents non-exclusifs dans le jeu de données qui présentent des affinités avec les méthodes en se basant uniquement sur les résultats de classification.

De part la taille de *ImageNet* nous avons restreint les classes à la liste suivante : *bag, bike, car, chair, computer, keyboard, person, phone, screen, shelf, table*. Ce choix est justifié par la possibilité d'observer les objets en conditions réelles, là où nous produisons notre propre jeu de données *RGB-D* : voir Section "ONERA.ROOM" (page 103).

4.4.1 Approche 1 : en partant des méthodes

Notre première approche, similaire à la méthode présentée dans les sections précédente, consiste à utiliser 4 réseaux de neurones différents entraînés sur *ImageNet* :

— *SqueezeNet*,

— *GoogleNet*,

— *AlexNet*,

— *VGG*16,

puis d'utiliser un réseau de confiance entraîné à la lumière des performances de chaque réseau de classification. Nous rappelons l'objectif de notre approche de sélection dans la Figure 4.7.

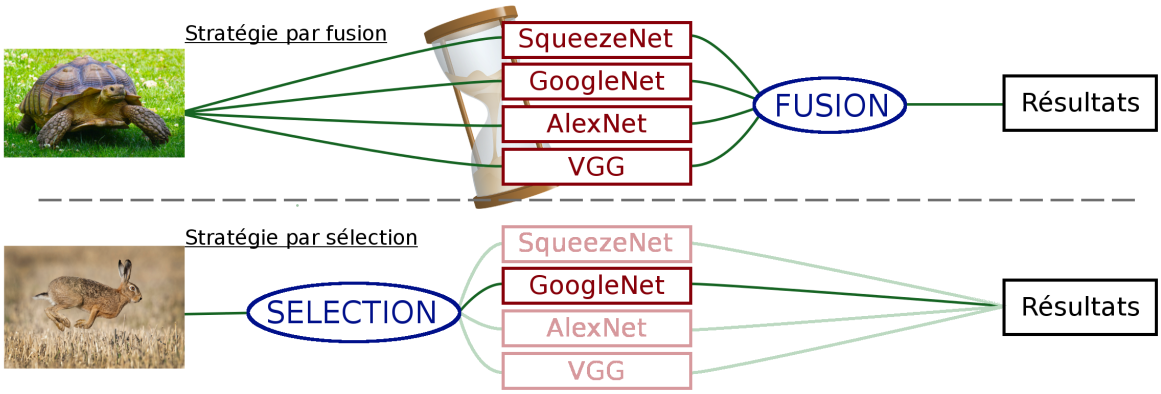


FIGURE 4.7 – Stratégie de sélection sur *ImageNet*

Résultats des classifieurs

Le Tableau 4.4 montre le score de précision de chaque méthode sur le jeu de données ImageNet ainsi que les temps de passe d'une image dans chaque réseau. Nous présentons également les performances d'un oracle théorique qui saurait choisir la meilleure méthode et montre un gain potentiel de performance significatif.

TABLEAU 4.4 – Précision et temps de calcul des réseaux classifieurs sur *ImageNet*

Méthodes	Top-1 score	Temps de calcul
AlexNet	83.2%	25 ms
GoogleNet	82,30%	40 ms
VGG	83.9%	44 ms
SqueezeNet	80.4%	60 ms
Oracle	93,2%	60 ms

TABLEAU 4.5 – Complémentarité des réseaux classifieurs sur *ImageNet*. Ici les valeurs correspondent aux pourcentages d'images correctement classifiées par les deux méthodes. Par exemple, 73,1% des images ont été correctement classifiées par *SqueezeNet* et *GoogleNet*. Les valeurs de la diagonale correspondent aux scores des méthodes.

AlexNet	83,2			
GoogleNet	76,7	82,3		
SqueezeNet	74,4	73,1	80,4	
VGG	78,2	77,4	74,4	83,9
	AlexNet	*GoogleNet*	*SqueezeNet*	*VGG*

Plus précisément, le Tableau 4.5, montre que 73,1% des images ont été correctement classifiées par *SqueezeNet* et *GoogleNet*. Cependant, *GoogleNet* a pu classifer 9,2% d'images en plus, soit une précision de 82,3% et *SqueezeNet* a pu classifier 7,3% d'images en plus soit une précision de 80,4%. Cela signifie que 9,2+7,3=16,5% des images sont classifiées exclusivement par *GoogleNet* ou *SqueezeNet*. C'est exactement dans ce genre de situation qu'une approche par sélection a de l'intérêt : les méthodes sont complémentaires. Cette situation se retrouve également dans la section précedente, dans le Tableau 4.2 où nous observons par exemple une complémentarité sur le jeu de données CIFAR10.

Architecture du sélecteur

Dans cette situation, nous n'avons pas réutilisé un réseau de confiance basé sur l'architecture *SqueezeNet*. En effet, un tel réseau n'était pas capable de différencier suffisamment efficacement les données pour les associer aux classifieurs. Cette situation est particulièrement difficile car les données sont très variées et les méthodes sont conceptuellement proches (ce sont toutes des *CNNs*).

Nous avons donc défini une nouvelle architecture de sélection plutôt que de réutiliser des réseaux déjà existants construit pour de la classification. Notre hypothèse consiste à dire que les filtres convolutionnels des méthodes sont sensibles à certaines sous-variétés de nos images. Etant donné que toutes les méthodes que nous utilisons ici sont capables de classifier les mêmes classes, ce n'est pas par rapport aux classes que nous cherchons à les différencier. Nous avons donc cherché à construire un réseau de neurones capable de trouver des différences entre les images indépendamment des classes. Autrement dit, plutôt que de chercher des descripteurs locaux pour classifier les images, nous cherchons des descripteurs globaux pour caractériser les images. Ceci est effectué par l'architecture présentée dans la Figure 4.8 appelée "*Dofeann*" pour *Domain features neural network*.

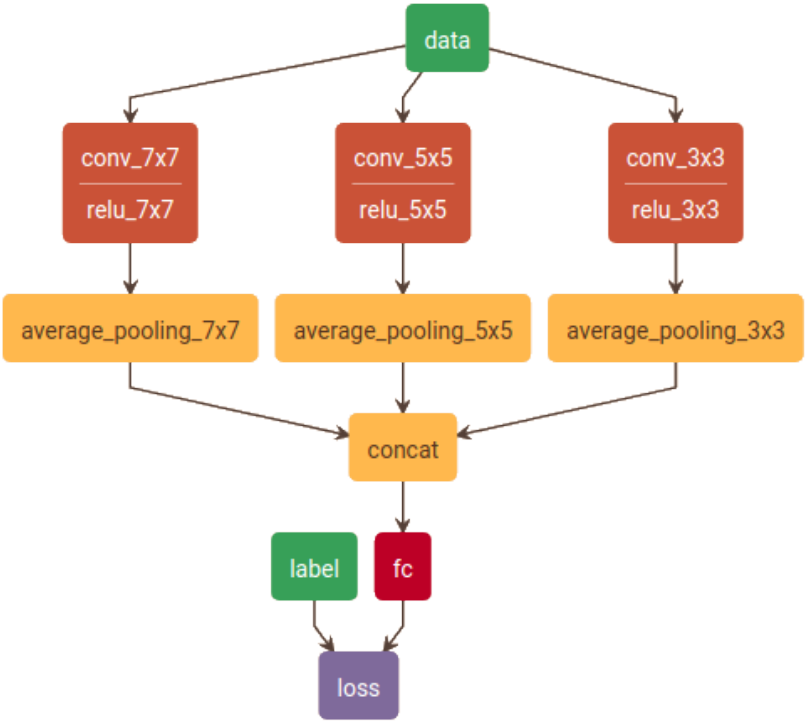


FIGURE 4.8 – Architecture *Dofeann* pour la sélection de méthode selon des descripteurs globaux.

L'architecture *Dofeann* a pour objectif d'être légère. Elle est constituée de trois branches qui appliquent respectivement 20 filtres convolutionnels de taille 7x7, 5x5 et 3x3. Ceci résulte en 60 cartes d'activations dont nous calculons la moyenne par carte pour obtenir un vecteur de 60 scalaires. Ce vecteur caractérise la sensibilité moyenne de l'image par rapport à 60 descripteurs locaux. Nous projetons ensuite ce vecteur de taille 60 dans un espace de taille 4 par une couche "*fully-connected*". Ceci nous permet d'entraîner le réseau avec une fonction de perte euclidienne avec les étiquettes que nous définissons par dans la section suivante.

Formalisation de la vérité-terrain pour le sélecteur

En regardant les performances de nos 4 méthodes il est possible de séparer les images en $2^4 = 16$ partitions :

1. les images classifiées par aucune méthodes : [0,0,0,0]
2. les images classifiées par la méthode 1 : [0,0,0,1]
3. les images classifiées par la méthode 2 : [0,0,1,0]
4. les images classifiées par la méthode 3 : [0,1,0,0]
5. les images classifiées par la méthode 4 : [1,0,0,0]
6. les images classifiées par les méthodes 1 et 2 : [0,0,1,1]

   ...

15. les images classifiées par les méthodes 1 et 2 et 3 : [0,1,1,1]
16. les images classifiées par toutes les méthodes : [1,1,1,1]

Nous pouvons ainsi attribuer à chaque image une étiquette des performances des classifeurs sous forme binaire (par exemple [0,1,1,0]). Cette première représentation nous permet d'appliquer la fonction de perte de regression logistique.

Nous avons également envisagé une seconde représentation sous la forme des performances de classification pour la classe de l'image [méthode 1 : 0.38, méthode 2 : 0.56, méthode 3 : 0.97, méthode 4 : 0.21]. Cette seconde représentation permet d'introduire la capacité de chaque méthode à classifier une image donnée indépendamment de sa décision finale. Même si la première classe proposée par une méthode n'est pas la bonne, il est possible que la seconde classe soit correcte et avec un score très élevé. Par exemple une prédiction [chat : 0.89, chien : 0.05, lion : 0.88, ...] pour une méthode X sur une image de lion aurait dans la première représentation l'étiquette 0 (car mauvaise classificaiton) alors que dans la seconde représentation elle aurait l'étiquette 0,88. Ce qui permettrait de ne pas sanctionner le réseau sélecteur s'il venait à proposer cette méthode. En pratique cette représentation de la vérité terrain ne permettait pas la convergence du réseau sélecteur. Nous ne présenterons que des résultats avec la première représentation binaire.

En pratique, plus de 70% des images peuvent être classifiées par 3 ou 4 méthodes. Ces exemples sont moins discriminants vis-à-vis des méthodes à employer que les images classifiées par une seule méthode. Nous avons donc pondéré l'erreur d'apprentissage en fonction du nombre de méthodes disponibles pour la classification avec les coefficients qui suivent :

— (1 1 1 1) → 0.0
— (0 1 1 1) → 0.5
— (0 0 1 1) → 2.0
— (0 0 0 1) → 5.0
— (0 0 0 0) → 0.0

TABLEAU 4.6 – Résultats du réseau de confiance *Dofeann* sur *ImageNet*

Méthodes	Top-1 score	Temps de calcul cumulés
AlexNet	83.2%	41.98 s
GoogleNet	82,30%	67.16 s
VGG	**83.9%**	73.88 s
SqueezeNet	80.4%	100.74 s
Oracle	**93.2%**	476.43 s
RC *Dofeann* + méthodes choisies	83,80%	129,4 s

Nous obtenons après entraînement les résultats du Tableau 4.6. La stratégie proposée ici n'est pas concluante. En effet, le réseau de confiance *Dofeann* couplé aux méthodes qu'il propose ne permettent pas de dépasser les performances du meilleur réseau tout seul (le *VGG*). Cependant cette approche reste potentiellement intéressante : si nous parvenons à dépasser les performances du meilleur classifieur seul avec la stratégie de sélection alors nous aurons une méthode performante avec un temps de calcul inférieur au temps de calcul de l'oracle (et par conséquent inférieur aux temps de calcul des méthodes de fusion).

4.4.2 Approche 2 : en partant des données

Dans notre première approche nous cherchions à différencier les images en fonction des performances des méthodes. Nous nous sommes alors confrontés à une situation difficile pour séparer les données. Nous avons donc décidé d'aborder le problème sous un autre angle : séparer d'abord les données en se basant uniquement sur l'information qu'elles contiennent, puis entraîner des méthodes pour chaque partition.

Pour cela nous appliquons le protocole suivant :

1. Sélection d'un sous-ensemble de 20 filtres non corrélés de la première couche d'un VGG ([7x7x20]) et calcul de la moyenne d'activation sur les images

   image [256x256x3]→ vecteur [20]

2. Décomposition par *PCA*

   vecteur [20] → vecteur [10]

3. Partitionnement des données dans cet espace de taille [10] avec la méthode des k-moyennes (k=4)

   vecteur [10] → vecteur [4]

4. Entraînement du réseau de partitionnement avec des étiquettes exclusives d'appartenance aux clusters.
5. Usage du réseau de partitionnement pour produire les clusters finaux
6. Entraînement d'un réseau de classification pour chaque cluster final

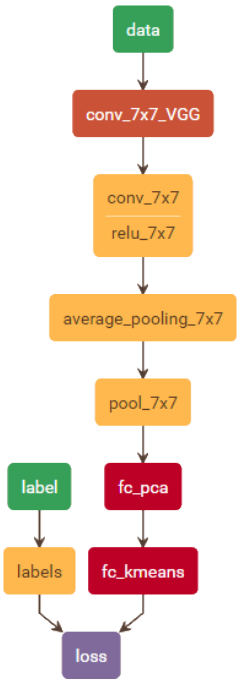


FIGURE 4.9 – Architecture de partitionnement.

La Figure 4.9 présente notre architecture complète de partitionnement. Elle utilise 20 des premiers filtres de convolution d'un VGG16, calcule la moyenne d'activation des images pour construire un vecteur sur lequel nous appliquons une *PCA* et enfin un partitionnement par la méthode des k-moyennes. La *PCA* est appliquée au travers d'un produit matriciel ce qui permet de l'incorporer directement dans le *CNN* sous forme d'une couche "*fully-connected*". La phase d'apprentissage sur le réseau de partitionnement permet d'apprendre la dernière couche du réseau qui correspond au partitionnement par la méthode des k-moyennes. Nous pouvons dire que nous apprenons le résultat de la méthode des k-moyennes avec une couche "*fully-connected*".

Finalement notre réseau de partitionnement prend une image en entrée et produit un vecteur d'appartenance aux clusters appris. Il n'y a pas besoin de retomber exactement sur les clusters obtenus à l'étape 3. L'essentiel c'est que les réseaux de classifications soient des

experts sur les clusters de l'étape 5. L'analogie entre le réseau de partition et les experts de clusters avec le réseau de confiance et les méthodes de classifications est directe. Pour évaluer cette méthode il suffit de prendre chaque image de test, voir à quel cluster elle appartient en utlisant le réseau de partitionnement puis de lui appliquer la méthode experte du cluster. Les résultats de cette approche testée avec des *SqueezeNets* comme experts sont disponibles dans le Tableau 4.7.

TABLEAU 4.7 – Résultats de l'approche par clustering sur *ImageNet*

Méthodes	Top-1 score
SqueezeNet entraîné sur tout *ImageNet*	80.4%
Oracle	**81.1%**
Réseau de partitionnement + experts choisis	77.4%

Cette seconde approche ne s'avère pas non plus concluante. Cette tendance semble illustrer encore une fois que les méthodes qui font face aux plus d'exemples à l'entraînement seront plus à même de généraliser sur l'ensemble de test que des experts sur-spécialisés. Cependant il est important de bien définir la source d'erreur : ce sont les réseaux de confiance ou de partitionnement qui ne permettent pas d'atteindre des performances utiles. En effet, si nous regardons les oracles théoriques dans le Tableau 4.6 ou dans le Tableau 4.7 ils sont meilleurs que toutes les autres méthodes. Cette problématique reste donc ouverte.

4.5 Conclusion du chapitre

Notre première observation concerne l'oracle théorique. Dans l'ensemble des situations étudiées les méthodes sont toujours complémentaires. Il y a donc un gain de performance possible pour les stratégies de fusion ou de sélection à même d'exploiter les données.

Nous avons mis en avant deux situations où la stratégie de sélection est efficace. Premièrement, dans un cas simple avec seulement deux méthodes différentes, et deuxièmement, dans un cas où les domaines d'apprentissages sont très variés. Cela signifie que le réseau sélecteur a pu trouver un espace de représentation des données dans lequel les affinités avec les méthodes étaient mesurables donc séparables.

Cependant, dans l'étude sur ImageNet avec 4 *CNNs* différents la tâche s'est avérée trop complexe. Malgré une complémentarité démontrée des méthodes, nous ne parvenons pas à prédire correctement celles qu'il faut utiliser. Dans cette situation les méthodes de classification ont été entraînées sur le même domaine d'images. Cela implique que le réseau sélecteur ne peut pas utiliser les différences de domaines en tant que critères de séparabilité comme dans la situation précédente. Il doit trouver une relation entre les images et les structures internes des méthodes. Autrement dit, le sélecteur doit être capable de trouver des descripteurs discriminants propres à chaque architecture convolutionnelle de classification. Hors, ces architectures sont très profondes et avec de nombreuses non-linéarités. Cette tâche, *a posterori*, n'est pas réalisable avec une structure de réseau sélecteur aussi simple que *Dofeann*. Il faudrait peut-être complexifier l'architecture de sélection au risque de perdre l'intérêt de cette approche...

Par ailleurs, les travaux présentés dans ce chapitre ont permis de mettre en avant les difficultés d'utilisation de l'apprentissage profond. Bien que non indiqué dans nos résultats, l'usage de ces méthodes est particulièrement chronophage et demande de bien poser les objectifs avant de lancer les entraînements.

A la lumière des performances du concept de réseau sélecteur nous avons décidé d'orienter la suite de nos travaux vers les données *RGB-D*. En effet, ces données présentent naturellement des différences de domaine que nous pouvons utiliser pour prédire quelles méthodes il faut employer. C'est le sujet du chapitre suivant : "Approche multi-modale pour la détection".

Dissémination

• Ces travaux ont fait l'objet d'une présentation dans le *workshop* "Apprentissage profond pour la perception et la robotique" au congrès sur la Reconnaissance des Formes et l'Intelligence Artificielle (RFIA) 2016.

Sélection d'algorithmes de classification par réseau de neurones
Joris Guerry, Bertrand Le Saux, David Filliat

4.6 Références

CARUANA, R., A. NICULESCU-MIZIL, G. CREW et A. KSIKES. 2004, «Ensemble selection from libraries of models», dans Proceedings of the twenty-first international conference on Machine learning, ACM, p. 18. 66

DENG, J., W. DONG, R. SOCHER, L.-J. LI, K. LI et L. FEI-FEI. 2009, «Imagenet : A large-scale hierarchical image database», dans Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on, IEEE, p. 248–255. 66

HE, K., X. ZHANG, S. REN et J. SUN. 2016, «Deep residual learning for image recognition», dans Proceedings of the IEEE conference on computer vision and pattern recognition, p. 770–778. 66

KRIZHEVSKY, A. et G. HINTON. 2009, «Learning multiple layers of features from tiny images», . 66, 68

LECUN, Y., L. BOTTOU, Y. BENGIO et P. HAFFNER. 1998, «Gradient-based learning applied to document recognition», Proc. of the IEEE, vol. 86, n° 11, p. 2278–2324. 68

LU, D. et Q. WENG. 2007, «A survey of image classification methods and techniques for improving classification performance», International journal of Remote sensing, vol. 28, n° 5, p. 823–870. 66

REDMON, J. et A. FARHADI. 2016, «Yolo9000 : better, faster, stronger», arXiv preprint arXiv :1612.08242. 66

XIAO, J., A. OWENS et A. TORRALBA. 2013, «SUN3D : A database of big spaces reconstructed using sfm and object labels», dans 2013 IEEE International Conference on Computer Vision, Sidney, Australia, p. 1625–1632. 68

Chapitre 5

Approche multi-modale pour la détection d'objets

> *This is video recognition. This is a camera seeing what's going on.*
>
> Bill Gates
> speaking about the Kinect™ camera,
> CNet News Interview, 2009

Sommaire

5.1 Introduction 86
5.2 Les données multi-modales 86
5.2.1 Carte de profondeur (*D*) : étude des encodages 86
5.2.2 Carte des couleurs (RGB) : Etude de l'influence de la luminosité . 94
5.3 Détection multi-modale 96
5.3.1 Construction de l'approche 96
5.3.2 Architectures finales des méthodes 98
5.4 Résultats de détection 100
5.4.1 Résultats sur *RGBD People* 101
5.4.2 Résultats sur *InOutdoor RGBD People* 102
5.4.3 ONERA.ROOM 103
5.5 Conclusion du chapitre 110
5.6 Références 111

5.1 Introduction

Comme nous avons pu le voir dans le Chapitre "Sélection d'algorithmes de classification", l'usage de plusieurs méthodes peut permettre d'améliorer les performances. Pour pouvoir bénéficier de la complémentarité des méthodes disponibles il faut cependant être capable de discerner des domaines propres à chaque méthode. Or cette séparation est immédiate avec des données provenant de modalités différentes et nous nous sommes donc orientés vers cette problématique.

Dans le monde de la robotique nous pouvons trouver des capteurs *RGB-D* qui offrent directement ce genre de données : une modalité *RGB* et une modalité *Depth* (ou carte de profondeur). Comme expliqué dans le Chapitre "État de l'art : Apprentissage profond", la carte de profondeur possède des propriétés géométriques proches des images couleurs (comme les contours ou les dimensions) mais ne contient pas la même information. Globalement insensible aux couleurs et aux textures, la *Depth* contient de l'information sur le volume et les distances. Exploitée avec succès par GUPTA et al. [2014] après encodage en *HHA*, d'autres méthodes ont également proposé diverses façon de traiter la carte de profondeur avant de l'injecter dans un réseau de neurones pour en faciliter l'extraction d'information. Cette tendance présente l'inconvénient d'alourdir le processus de traitement [1]. Nous avons donc décidé d'étudier l'intérêt de l'encodage de la carte de profondeur.

Les caméras *RGB-D* ont la particularité d'être actives pour obtenir la carte de profondeur en émettant un champ infrarouge. Ce comportement a deux conséquences : une mauvaise, de jour ce capteur est bruité par le rayonnement infrarouge du soleil, une bonne, il fonctionne dans le noir. Ainsi, quand le robot qui embarque la caméra *RGB-D* est en extérieur de jour il peut utiliser son capteur *RGB*, quand il fait nuit ou que l'environnement est trop sombre il peut utiliser le capteur *Depth* et dans les autres situations les deux capteurs fonctionnent ensemble. Nous étudions donc dans ce chapitre comment faire travailler ensemble des méthodes de sémantisation *RGB* et *Depth* pour que le robot soit toujours capable d'analyser son environnement.

5.2 Les données multi-modales

Avec l'avènement de capteurs d'image et de profondeur peu coûteux, de nombreuses solutions ont été proposées pour l'extraction et la fusion des informations de couleur et de profondeur. Dans cette section, nous menons une étude approfondie de ces approches sur deux tâches : la détection d'objet et la segmentation sémantique. Nous évaluons et comparons les différents encodages de profondeur de référence dans un cadre d'apprentissage profond pour déterminer le plus efficace. Nous mettons également en avant la complémentarité de la fusion de la couleur et de la profondeur face à des fortes variations de luminosité.

5.2.1 Carte de profondeur (*D*) : étude des encodages

L'information contenue dans le canal de profondeur ne comporte pas la même signification que les informations contenues dans les canaux *RGB*. Elle possède une dynamique différente, à la fois spatialement mais aussi en intensité. De plus c'est une mesure très bruitée de part le processus même de mesure mais aussi à cause de l'environnement qui peut contenir des surfaces réfléchissantes, brillantes, semi-transparentes, etc. D'autre part, l'évolution constante

1. Les encodages de la carte de profondeur utilisent généralement un passage par une représentation 3D gourmande en ressources pour estimer les normales des surfaces.

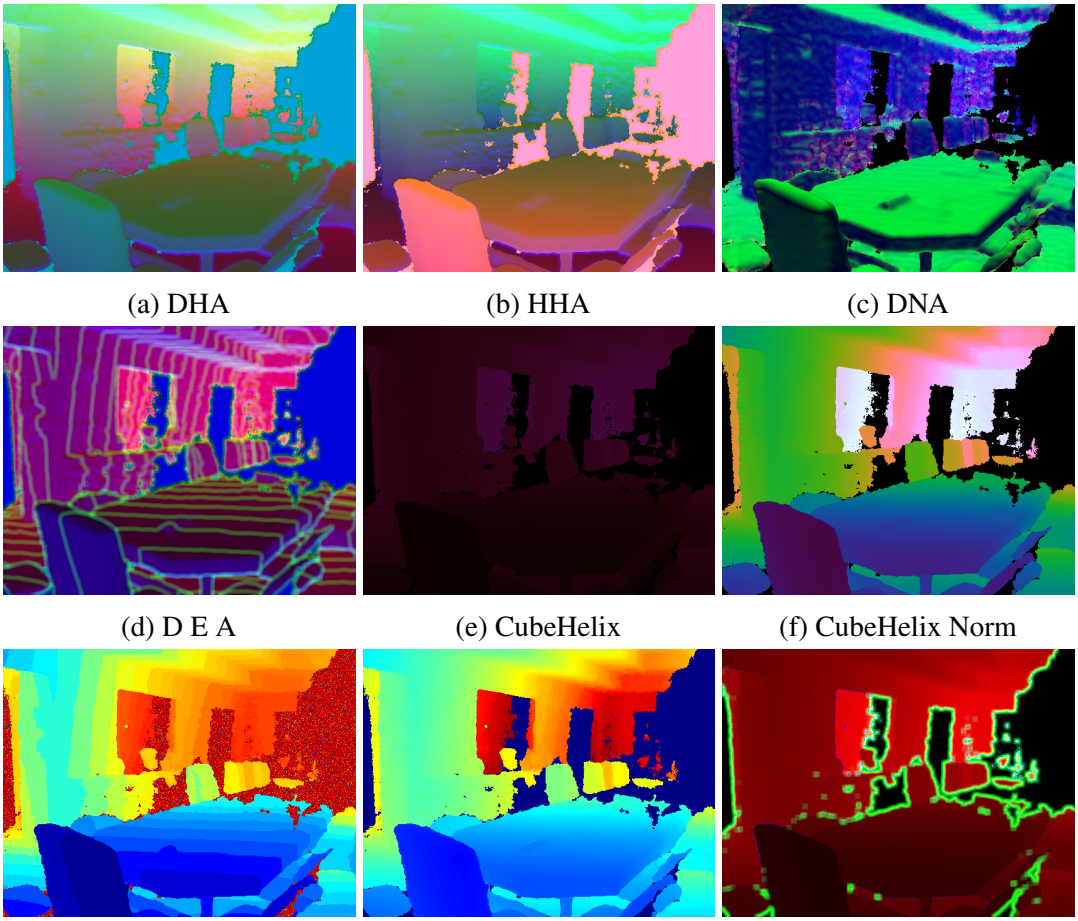
des performances des méthodes convolutionnelles purement *RGB* rattrapent régulièrement les méthodes *RGB-D* comme BADRINARAYANAN et al. [2015] en segmentation sémantique sur SUNRGBD avec leur SegNet. C'est pourquoi deux problèmes connexes sont encore d'actualité :

— Comment prétraiter la carte de profondeur pour faciliter l'extraction d'information par les réseaux de neurones ? (objet de cette sous-section, 5.2.1)

— Comment fusionner cette information issue de la carte de profondeur avec l'information contenue dans l'image couleur ? (voir Section 5.3)

Comme nous avons pu le voir dans la Section 3.4, l'encodage de la carte de profondeur peut jouer un rôle primordial dans l'usage des données *RGB-D* . Cependant, de nombreuses références [BOULCH et al., 2017; EIGEN et FERGUS, 2015; GUPTA et al., 2014; HANDA et al., 2016; LONG et al., 2015; MEES et al., 2016] utilisent des encodages différents pour la carte de profondeur. Cette situation nous laisse devant un vaste choix lorsque nous souhaitons également utiliser la carte de profondeur. En particulier dans les scénarios d'exploration robotique envisagés où la modalité *RGB* pourrait être complètement indisponible ou inutilisable.

C'est pourquoi nous avons décidé de comparer nous-mêmes les différents algorithmes d'encodages.



(a) DHA (b) HHA (c) DNA

(d) D E A (e) CubeHelix (f) CubeHelix Norm

(g) HisteqR Jet (h) Jet Norm (i) HisteqF E Lsd

FIGURE 5.1 – Illustration des différents encodages appliqués à la carte de profondeur sur le jeu de données NYUDv2.

Les encodages Nous avons défini plusieurs nouvelles procédures d'encodage de la carte de profondeur :

- Égalisation d'histogramme avec plusieurs variantes :
  - *Front* : pixels des artefacts remplacés par des 0 (**HistEq F**, Figure 5.1(i))
  - *Back* : pixels des artefacts remplacés par des 255 (**HistEq B**)
  - *Random* : pixels des artefacts remplacés par des valeurs aléatoires entre 0 et 255 (**HistEq R**, Figure 5.1(g))
- Estimation du bruit au travers de la matrice de covariance du nuage de points 3D (abbrévié en **N**, voir Figure 5.1(c))
- Entropie locale sur 9x9 pixels (abbréviée en **E**, voir Figure 5.1(d))
- La normalisation des valeurs (**Norm**)
- La normalisation des premiers 95% des valeurs, dû à la considération qu'au delà les valeurs de profondeurs sont des artefacts d'acquisitions non nuls (**Focus**).
- Rendu en CubeHelix [GREEN, 2011] : ce rendu offre l'avantage de garder une variation linéaire de l'intensité en fonction de la profondeur (**Cub**, voir Figure 5.1(e))
- La variance locale sur une zone de 3x3 pixels –ou *Local Standard Deviation* (**Lsd**)

Ces nouvelles stratégies d'encodages s'ajoutent aux stratégies précédemment existantes définies dans l'état de l'art, Section 3.4 :

- HHA [GUPTA et al., 2014] (voir Figure 5.1(a))
- Jet [MEES et al., 2016] (voir Figure 5.1(h))
- DNA [BOULCH et al., 2017] (voir Figure 5.1(c))
- DHA [HANDA et al., 2016] (voir Figure 5.1(a))

Tâche de détection Nous effectuons tout d'abord une expérience de détection sur NYUDv2 avec l'architecture *Faster RCNN* de REN et al. [2015]. Dans l'architecture *Depth-RCNN* de GUPTA et al. [2014] les patchs obtenus sur la modalité *RGB* sont utilisés dans l'architecture multimodale (l'information couleur est au service de la détection et l'information *Depth* ne sert qu'à la sémantisation). Au contraire, nous avons choisi d'utiliser chaque modalité indépendamment les unes des autres. Cette contrainte nous semble primordiale étant donné que nous considérons des scénarios avec une perte complète de l'information couleur.

Les résultats de cette expérience sont présentés dans le Tableau 5.1. Les informations sur la couleur restent la meilleure source de décision. Nous n'avons pas réussi à reproduire les tendances de GUPTA et al. [2014] où l'encodage *HHA* a dépassé les performances de la couleur mais il reste néanmoins le meilleur encodage en profondeur pour la tâche de détection. D'autre part, les rendus, qu'il s'agisse de Jet ou de CubeHelix, ne permettent pas un gain significatif par rapport à la contribution de l'égalisation et de la focalisation de l'histogramme. En effet, ces derniers rattrapent presque le *HHA* (-0,5 points). En outre, le stockage de l'information en 16 bits n'améliore pas les performances, au contraire. C'est pourquoi nous étudierons uniquement les encodages de canaux 8 bits pour le reste de ce chapitre.

Tâche de segmentation Nous utilisons le *Fully Convolutional neural networks* (FCN) de LONG et al. [2015] pour effectuer la tâche de segmentation sémantique sur NYUDv2. En particulier, nous employons le modèle *FCN32s* basé sur *VGG16*. Ils rapportent des résultats comparables aux nôtres en utilisant les macro-classes de GUPTA et al. [2014]. Comme nous

TABLEAU 5.1 – Détection d'objets avec *Faster RCNN* sur le jeu de données NYUDv2 : comparaison de plusieurs encodages de la carte de profondeur.

Source	PM
RGB ($3 * 8$ *bits*)	**20.75%**
Carte de profondeur	
Depth (*8 bits*)	15.11%
Depth (*16 bits*)	14.59%
Focus (*8 bits*)	**16.41%**
Focus (*16 bits*)	15.14%
Égalisation d'histogramme "*Random*" (*8 bits*)	16.09%
Égalisation d'histogramme "*Front*" (*8 bits*)	**16.41%**
Égalisation d'histogramme "*Back*" (*8 bits*)	14.34%
Rendu de la carte de profondeur	
Jet map ($3 * 8$ *bits*)	15.29%
Cubehelix map ($3 * 8$ *bits*)	15.6%
HHA [GUPTA et al., 2014]	**16.94%**
Rendu de la carte de profondeur + Focus	
Jet map + Focus ($3 * 8$ *bits*)	14.76%
Cubehelix map + Focus ($3 * 8$ *bits*)	16.39%

pouvons le voir dans le Tableau 5.2, l'information RGB permet toujours d'effectuer une meilleure sémantisation que la carte de profondeur (68.64% de précision globale *versus* 51,2%). Cependant, dans ce genre d'environnement (NYUDv2 est essentiellement constitué de scène d'intérieur), les encodages DHA et HHA ne parviennent pas à dépasser le RGB (-5points). Un autre point intéressant c'est que l'encodage "Jet + Norm", utilisé par MEES et al. [2016], est le meilleur encodage par rendu si nous considérons la précision moyenne et l'IoU moyenne mais il reste 7 points derrière l'encodage *HHA*. Bien que le fossé était plus faible sur la tâche de détection, il semble que la segmentation sémantique bénéficie beaucoup du canal comportant les angles des surfaces. C'est en accord avec les remarques de SILBERMAN et al. [2012] (l'auteur de NYUDv2) qui a montré l'intérêt des indices géométriques pour faciliter l'inférence de classes sémantiques.

Un mot sur le challenge SHREC 2017

Le challenge Nous profitons de cette section sur les encodages pour introduire nos résultats au challenge *3D Hand Gesture Recognition* du workshop *3D Shape Retrieval Contest* (SHREC) de *Eurographics* 2017. Bien qu'éloigné de notre domaine d'application initial nous cherchons ici à caractériser des séquences vidéos de mouvements. Nous faisons donc face à une problématique d'encodage de l'information vidéo. La longueur des séquences peut varier de 15 à 150 images. Les mouvements sont regroupés en 14 classes ou 28 si nous considérons que la main est ouverte ou fermée. Bien que le squelette [2] de la main soit également fourni nous avons opté pour une solution purement basée sur la carte de profondeur.

2. positions des articulations pour chaque image

TABLEAU 5.2 – Comparaison des encodages de la carte de profondeur sur la tâche de segmentation sémantique de NYUDv2 avec un FCN32s de LONG et al. [2015].

Encodage	P	PM	mIoU
RGB			
RGB	68.64%	51.3%	37,3%
Encodages avec traitements géométriques			
DHA HANDA et al. [2016]	**64.2%**	43.7%	31.3%
HHA GUPTA et al. [2014]	64.1%	**44.5%**	**31.7%**
DNA BOULCH et al. [2017]	62.4%	42.1%	29.4%
HisteqF E A	61.9%	39.7%	27.8%
Rendus et Normalisation seulement			
CubeHelix	**59.8%**	29.8%	20.8%
CubeHelix+Norm	59.7%	35.5%	24,6%
HisteqR+Jet+Norm	59.6%	36.6%	25.4%
Jet+Norm EITEL et al. [2015]	59.5%	**37.4%**	**25.7%**
HisteqB+Jet+Norm	59.3%	36.4%	25.0%
HisteqF+Jet+Norm	59.1%	36.8%	25.2%
HisteqR+Cub+Norm	58.9%	34.3%	23.6%
HisteqF E S	58.6%	32.2%	22.1%
HisteqF+Cub+Norm	58.6%	33.8%	23.3%
HisteqB+Cub+Norm	58.2%	33.5%	23.1%
HisteqR+Jet	57.6%	33.0%	22.5%
HisteqB+Jet	57.3%	32.7%	22.2%
D+Norm	55.3%	21.1%	14.0%
Jet	54.5%	20.8%	13.9%
D	51.2%	16.3%	10.3%

FIGURE 5.2 – Exemple de mouvement du challenge *3D Hand Gesture Recogniton* : "Swipe left". Image extraite de http://www-rech.telecom-lille.fr/shrec2017-hand/

Comme énoncé dans le Chapitre "État de l'art : Apprentissage profond", la difficulté avec une nouvelle tâche pour un réseau de neurones est de définir ce que nous mettons en entrée, ce que nous désirons en sortie, quelle fonction de perte utiliser et quelle structure de réseau choisir.

Notre intuition vient du fait qu'il est possible de deviner un mouvement rien qu'en regardant un sous ensemble d'images de la séquence : une image du début, une image du mileu et une image de la fin par exemple. En se basant sur cette observation nous avons pensé à simplement concaténer les cartes de profondeur les unes sur les autres, dans l'ordre temporel. Ainsi chaque canal représente une image-clé et un tenseur représente un mouvement. Un exemple de tenseur avec 3 images-clés est proposée dans la Figure 5.3.

FIGURE 5.3 – Exemple d'encodage pour le *workshop SHREC*. Le mouvement representé ici est un pincement avec seulement deux doigts. La main part du haut (rouge), se referme assez rapidement en descendant (vert) et fini vers le bas complètement pincée (bleu)

Comme nous pouvons le voir sur la Figure 5.3 la zone de l'image correspondant à

l'opérateur est essentiellement blanche car il ne bouge pas. Ce genre de représentation est un autre exemple d'injection d'*a priori* humain dans la méthode de classification (comme annoncé dans le Chapitre 2 sur la reconnaissance visuelle).

Une fois la représentation choisie pour nos données nous retombons sur le formalisme de la classification par réseau de neurones. Nous utilisons donc un *VGG11* avec une couche de *SoftMaxWithLoss* qui tente de prédire la classe du mouvement.

Cependant l'entraînement est plus subtil. À l'image de MNIST, le jeu de donnée sur les chiffres, il est plus facile d'apprendre d'abord à différencier les 10 chiffres avant de différencier les nombres pairs des nombres impairs. Cet apprentissage multi-tâche se retrouve également dans le *Faster-RCNN* [REN et al., 2015] qui entraîne de façon alternative la tâche du *rpn* et celle du classifieur.

De la même façon, sur ce challenge nous avons procédé par un entraînement itératif avec plusieurs objectifs différents :

1. Tout d'abord nous avons appris à classifer les 14 classes (ce qui donne de meilleurs résultats que d'apprendre directement les 28 classes).

2. Nous avons ensuite utilisé le même réseau jusqu'à la couche de convolutions 5 pour apprendre une seconde tâche : est-ce que la main est ouverte ou est-ce que la main est fermée ? Ceci est directement reliée au problème à 28 classes : $class_{28} = 2 * class_{14} - \delta$, où δ correspond à 1 si la main est fermée, 0 sinon.

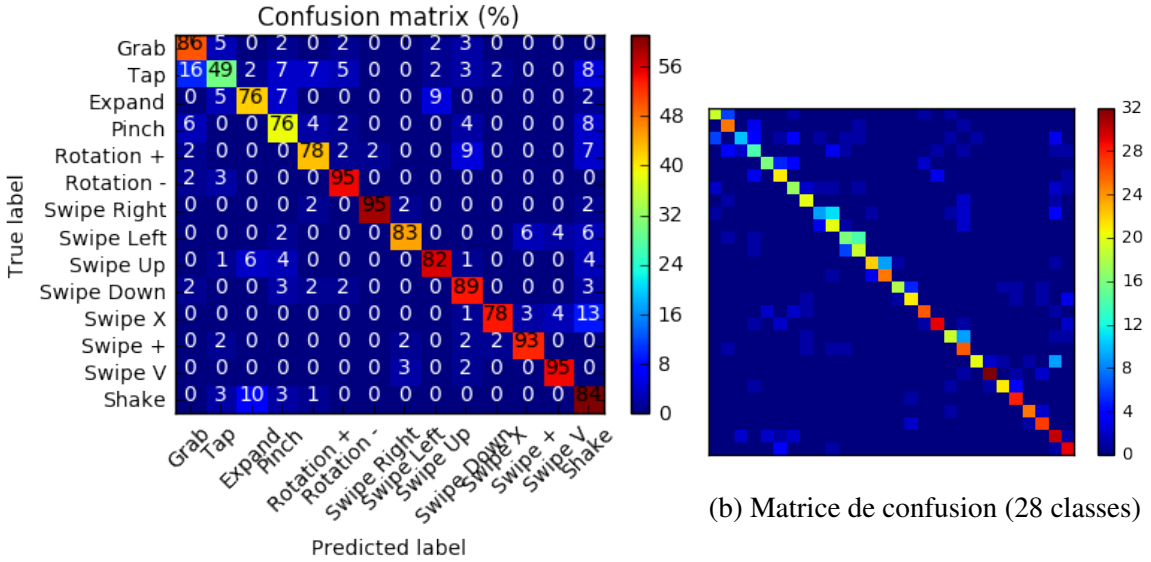
3. Nous pouvons désormais utiliser la prédiction des 14 classes en parallèle de la prédiction de l'état de la main pour prédire les 28 classes avec une couche "fully-connected" qui prend comme entrée toutes ces prédictions.

4. Désormais le réseau a convergé vers une situation où il sait prédire les 28 classes. Nous avons pu donc retirer le matériel interne propre au problème des 14 classes et de l'état de la main pour ne garder qu'une seule et unique structure qui prédit directement les 28 classes, ce qui permet d'alléger la structure globale et d'améliorer légèrement les résultats.

Nous avons essayé différents ratios d'échantillonnage allant de 3 images-clés par séquences, puis 5 et jusqu'à 10 images-clés. C'est l'approche avec 5 images-clés qui a donné les meilleurs résultats présentés dans le Tableau 5.3.

TABLEAU 5.3 – Précision sur le jeu de test du challenge *3D Hand Gesture Recogniton*

Tâche	P	Temps de traitement moyen
28 classes	71.9%	236ms
14 classes (depuis 28 classes)	82.9%	236ms
14 classes (prédiction directe)	81.0%	81ms



(a) Matrice de confusion (14 classes)

(b) Matrice de confusion (28 classes)

FIGURE 5.4 – Matrices de confusion sur les problèmes à 14 classes (gauche) et 28 classes (droite).

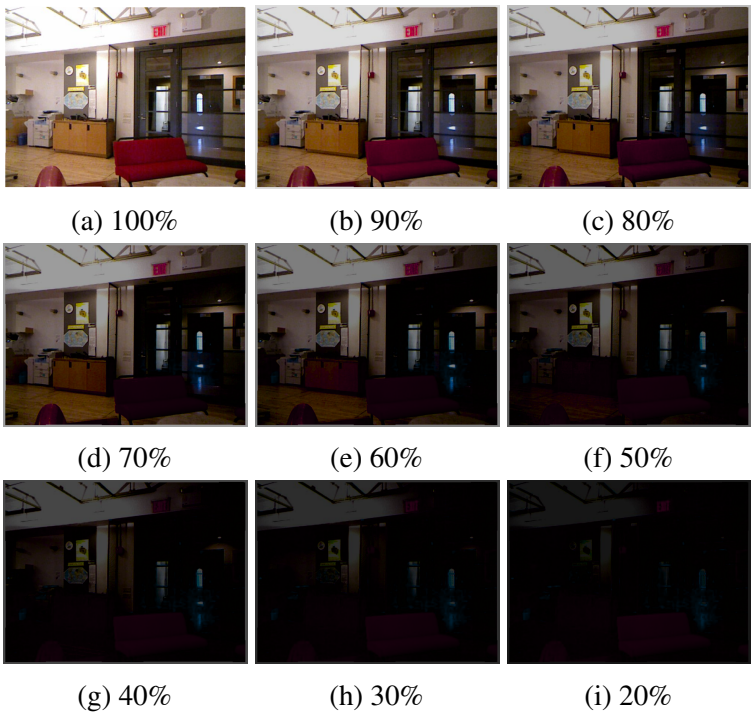
La matrice de confusion sur le problème à 28 classes (Figure 5.4(b)) montre que le réseau fait encore des erreurs pour prédire si la main est ouverte ou fermée mais parvient tout de même à trouver le mouvement : nous pouvons voir des petits blocs de 4 carrés le long de la diagonale, les mouvements sont rangés dans le même ordre que la matrice de confusion à 14 classes, doublés pour différencier l'état de la main.

TABLEAU 5.4 – Comparaison des performances sur le jeu de test du challenge *3D Hand Gesture Recogniton*

	Précision	
Méthode	14 gestes	28 gestes
Guerry *et al.*	82.90	71.90
DE SMEDT et al. [2016]	88.24	81.90
OHN-BAR et TRIVEDI [2013]	83.85	76.53
OREIFEJ et LIU [2013]	78.53	74.03
DEVANNE et al. [2015]	79.61	62.00

Ces résultats ont été proposés au challenge SHREC 2017. En comparaison de la méthode récente de DE SMEDT et al. [2016] qui fait usage de l'information du squelette de la main nous faisons moins bien (voir Tableau 5.4). Il est cependant intéressant de voir ce qu'une méthode purement réseau de neurones convolutifs et utilisant uniquement la carte de profondeur est capable de faire. Bien que non autorisées, les données *RGB* seraient certainement très utiles pour faire cette inférence. De plus il existe d'autres structures de réseaux de neurones particulièrement adaptées à des données séquentielles comme les *Long Short-Term Memory Network* de HOCHREITER et SCHMIDHUBER [1997] qui pourraient directement traiter les séquences, image par image.

5.2.2 Carte des couleurs (RGB) : Etude de l'influence de la luminosité



(a) 100% (b) 90% (c) 80%

(d) 70% (e) 60% (f) 50%

(g) 40% (h) 30% (i) 20%

FIGURE 5.5 – Variation de luminosité dans le jeu de données NYUDv2 "Eclipse"

Afin d'étudier l'influence des variations de luminosité nous avons produit une version artificiellement peu éclairée du jeu de données NYUDv2 appelée NYUDv2 "Eclipse" (voir Figure 5.5). Nous l'avons dupliqué en filtrant l'intensité lumineuse de 10% à chaque fois. Pour ce faire, nous avons projeté l'information *RGB* dans l'espace CIE L*u*v* où nous seuillons la luminance à i% avant de revenir dans l'espace *RGB*, ce qui donne le sous-dataset RGBi%. Nous avons entraîné un modèle RGBi% par sous-dataset {RGB100%, RGB80%, RGB60%, RGB40%, RGB20% } et évaluons tous les modèles sur tous les sous-datasets. La précision moyenne de chaque modèle est tracée dans la Figure 5.6.

Nous avons également entraîné d'autres modèles sur des ensembles de sous-datasets (RGB80%-100%, RGB60%-100%, etc.) et traçons le meilleur modèle avec des points bleus : RGB10%-100%. Ces résultats confirment le leitmotif du *Deep Learning* : "The more data the better". Cet entraînement sur l'ensemble de sous-datasets RGB10%-100% peut être interpreté comme de l'augmentation de données mais n'est pas capable de battre chaque modèle sur leur propre sous-dataset d'entraînement. Par conséquent, ce modèle d'ensemble est mieux fait pour s'adapter aux variations de luminosité mais ne maximise pas la performance potentielle.

La précision moyenne d'un modèle "*late-fusion*" de FCN32s avec l'encodage DHA est représentée dans le graphique de la Figure 5.7. Cette fusion permet à l'expert *Depth* de venir contrecarrer la perte de performance de l'expert couleur sur la baisse de luminosité. La fusion des deux fait toujours au moins aussi bien que l'expert *Depth* tout seul. De plus, les performances globales multimodales sont meilleures que l'expert RGB100% testé à 100% de luminosité.

Ces résultats nous permettent de mettre en avant l'influence de la luminosité et l'intérêt d'utiliser la carte de profondeur pour rester robuste en conditions difficiles. Nous en reparlerons plus loin dans ce chapitre avec des données réelles grâce à l'introduction du nouveau jeu de données ONERA.ROOM.

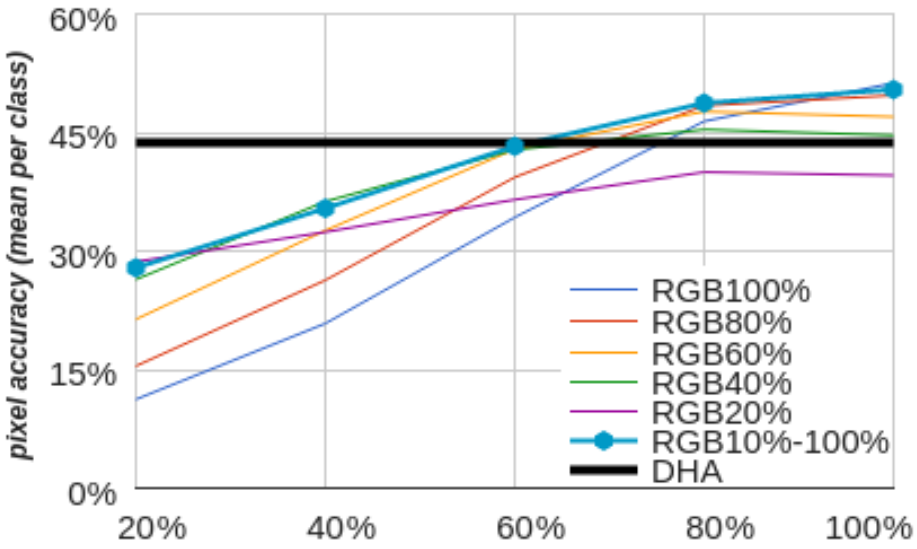


FIGURE 5.6 – Evaluation de l'influence de la luminosité sur la tâche de segmentation sémantique de NYUDv2. Les courbes colorées correspondent aux pourcentages de luminosités observés lors de l'entraînement. La courbe pointillée bleue représente la performance du modèle entraîné sur l'ensemble de NYUDv2 "Eclipse". La courbe horizontale noire correspond à la performance du classifeur *Depth* qui est indépendant aux variations de luminosité.

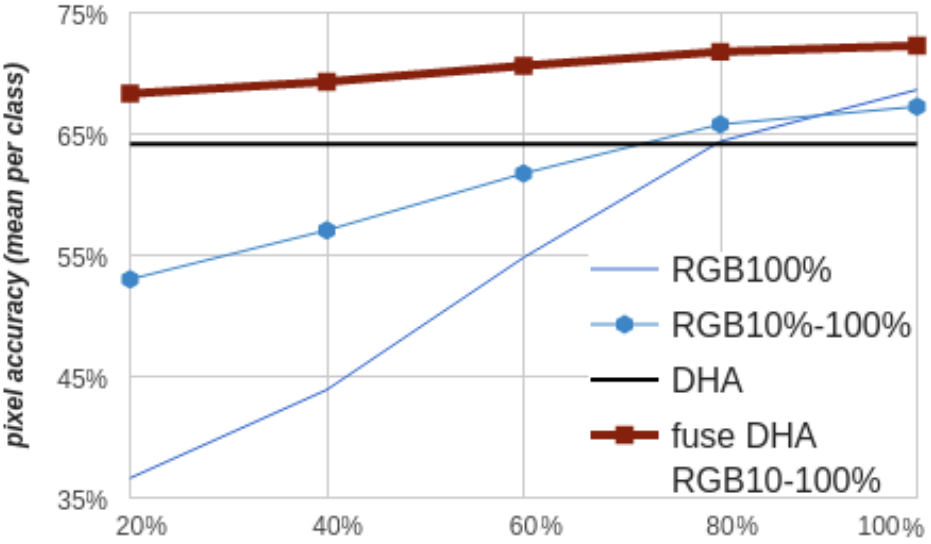


FIGURE 5.7 – Evaluation de l'influence de la luminosité sur les performances de segmentation. Les courbes de couleur correspondent au pourcentage de luminosité lors de l'entraînement des modèles et indiquent la précision moyenne. La courbe en pointillée bleue représente la performance du modèle entrainé sur l'ensemble de NYUDv2 "Eclipse" et la courbe noire représente la performance du classifieur *Depth*, indépendant aux variations de luminosité. Enfin, la courbe rouge représente la fusion par somme du modèle *Depth* et du modèle RGB10-100%.

5.3 Détection multi-modale

Après avoir mis en l'avant l'intérêt du couplage de données RGB avec des cartes de profondeur, nous nous sommes intéressés à la problématique de détection de personnes. Dans ce contexte, l'usage d'un encodage lourd et complexe comme HHA n'est pas préconisé. En effet, nous ne chercons pas à différencier des objets dont les angles des surfaces principales sont caractéristiques (comme les murs, les tables ou le sol dans NYUDv2). C'est pourquoi dans la suite de ce chapitre nous ne faisons pas usage d'un encodage canonique de la carte de profondeur. Cependant, comme nous pouvons le voir dans le Tableau 5.2, le processus de normalisation joue un role primordial (cf. Jet versus Jet+Norm). Nos données seront donc normalisées.

5.3.1 Construction de l'approche

Les travaux de MEES et al. ont proposé une méthode très intéressante pour la fusion de données mutlimodales : le *Gating Network*. Il se base sur des tenseurs de réseaux de neurones experts pour effectuer une somme pondérée avec les prédictions expertes. L'architecture de cette méthode est illustrée dans la Figure 5.9 et son fonctionnement dans différentes situations dans la Figure 5.10 .

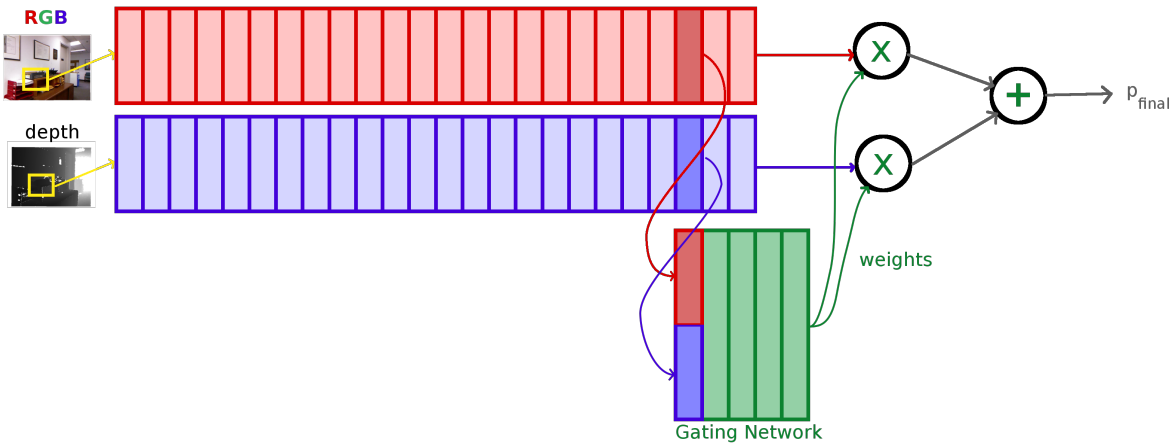


FIGURE 5.8 – Aperçu global de la méthode.

FIGURE 5.9 – Illustration de l'architecture du *Gating Network* de MEES et al. [2016]

Cette méthode possède plusieurs inconvénients. Tout d'abord, le réseau entier est traversé pour calculer les tenseurs haut-niveaux nécessaires à la pondération des experts. De plus, la totalité du procédé doit être effectué pour chaque patch.

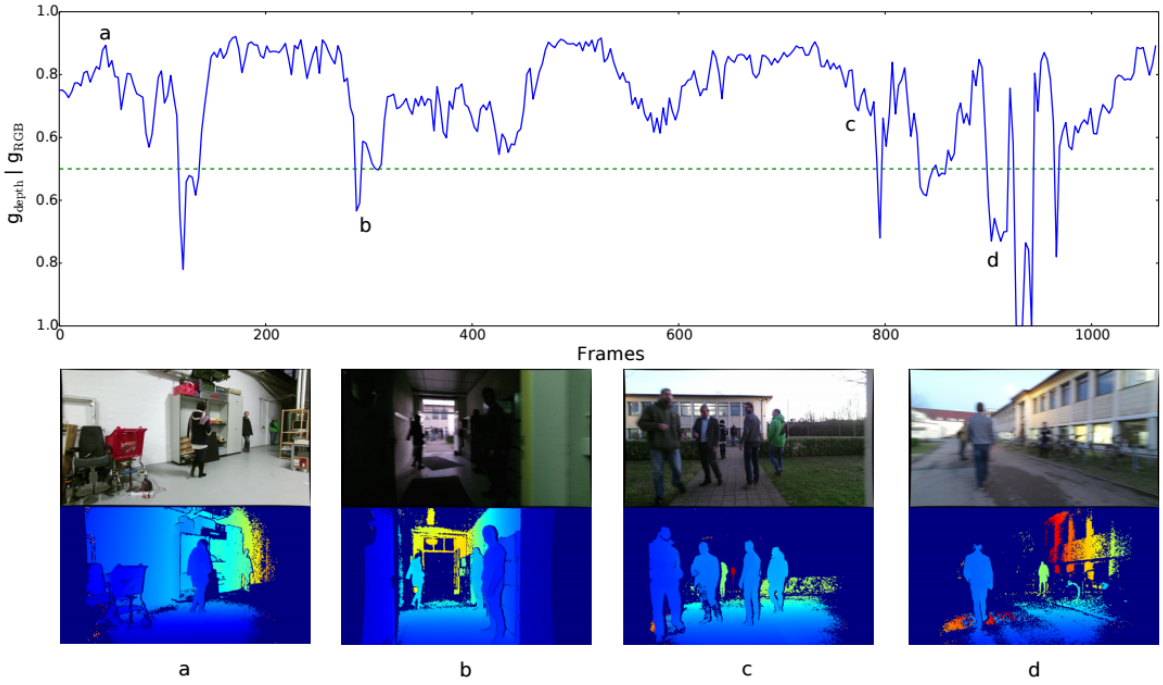


FIGURE 5.10 – Illustration du fonctionnement du *Gating Network* (extrait de MEES et al. [2016]). Le *Gating Network* propose un coefficient de pondération qui indique s'il faut plus pondérer les sorties de l'expert *RGB* ou de l'expert *Depth*.

Notre objectif initial est de se rapprocher des performances de cette méthode tout en effectuant de la sélection plutôt que de la fusion, c'est-à-dire en limitant au maximum les calculs faits sur des zones ou des modalités qui n'ont pas d'information utile. Ce que nous venons d'expliquer se traduit par le canevas suivant (Figure 5.11) :

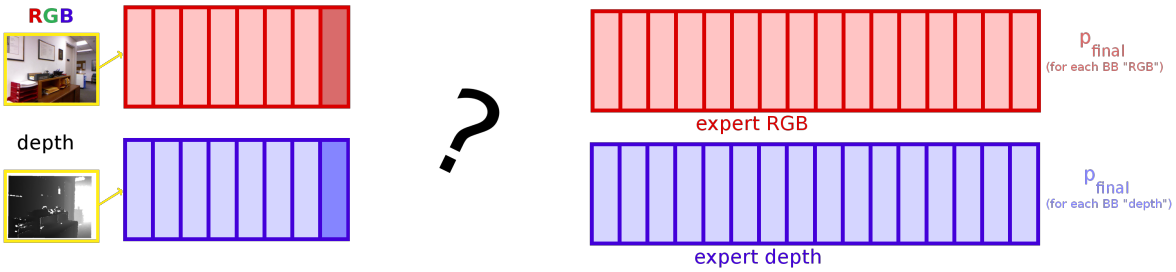


FIGURE 5.11 – Illustration de la méthode idéale de détection

Ce canevas prend toute l'image en entrée, possède un extracteur de caractéristiques léger sur l'entrée et propose des patchs pertinents, classifiés en sortie par deux experts. En modalité *RGB* il existe diverses solutions correspondantes à ce schéma, dont le *Faster-RCNN* REN et al. [2015] représenté dans la Figure 5.12 :

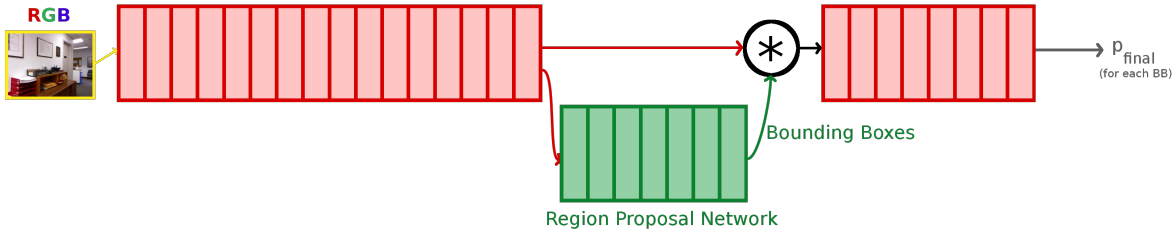


FIGURE 5.12 – llustration globale du *Faster-RCNN* de REN et al. [2015]

Cette méthode présente plusieurs avantages. Tout d'abord elle prend toute l'image en entrée. Elle fait de la proposition de patch interne et la régression de la position (avec le *RPN*). De plus, c'est une stratégie économe en temps et en ressources car elle utilise des tenseurs calculés sur toute l'image pour le *RPN* et pour le classifieur. Enfin c'est une structure qui s'adapte à toute architecture de *CNN* (car le *RPN* est un *CNN*). En s'inspirant de la méthode *Faster-RCNN* et du *Gating Network* nous proposons donc un *Faster R&M_CNN* (pour *Faster Region & Method CNN*) qui fait à la fois la proposition de patchs et la proposition de la méthode à utiliser. Le concept est illustré dans la Figure 5.13 :

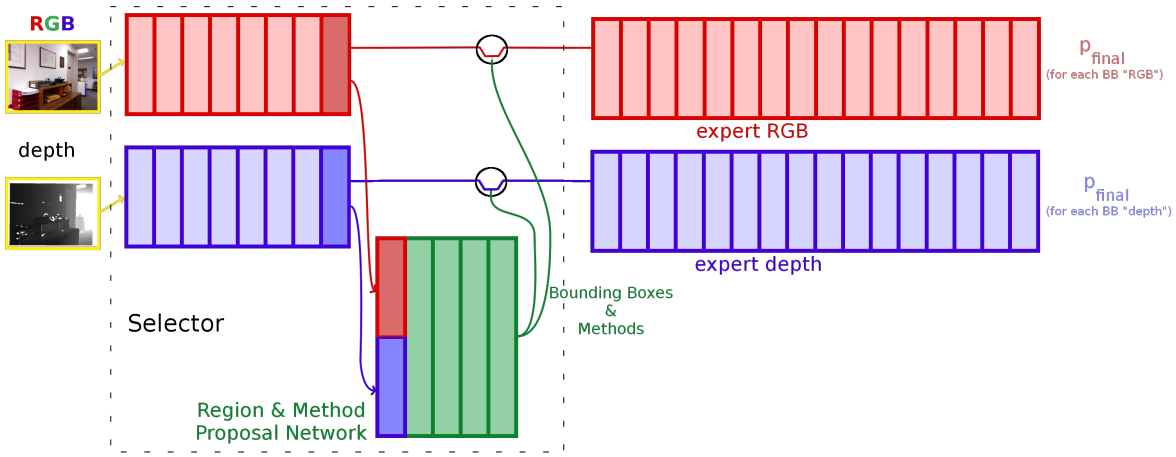


FIGURE 5.13 – Illustration globale du *Faster-R&M_CNN*

Cette proposition de méthode offre tous les avantages du *Faster-RCNN* pour des données multimodales.

Le *Faster-R&M_CNN* possède une partie qui est assimilable au rôle du sélecteur mentionné dans le chapitre précédent. Il est entouré en pointillé dans la Figure 5.13.

Ce formalisme facilite la réflexion quant à la position du réseau de proposition (en vert dans la figure). Dans le cas où le réseau est après la convolution numéro 5 d'un *VGG16* nous retombons dans le cas de deux *Faster-RCNNs* en parallèle qui échangent leurs détections. Dans le cas où nous plaçons le réseau de proposition dans les toutes dernières couches alors nous retombons sur une situation comparable au *Gating Network* de MEES et al.. Enfin, si nous avançons le réseau de proposition en amont des experts alors nous retombons sur le formalisme du chapitre précédent qui tentait de faire de la sélection.

Cette représentation du problème nous a orienté vers la solution médiane : mettre deux *Faster-RCNNs* en parallèle, un pour chaque modalité, et les laisser communiquer au niveau du *RPN*. Nous présentons cette architecture avec des variantes dans la sous-section qui suit.

5.3.2 Architectures finales des méthodes

Les idées de la section précédente nous ont conduit à proposer plusieurs approches pour la fusion de l'information des modalités *RGB* et *Depth*. Tout d'abord, nous construisons des experts propres à chaque modalité en utilisant un *Faster-RCNN* REN et al. [2015]. Nous montrons dans les résultats que cette architecture s'adapte très bien aux données de profondeur.

Ensuite, l'objectif de la fusion est de pouvoir s'adapter à des données réalistes dans lesquelles les conditions de luminosité peuvent complètement dégrader les performances de l'expert *RGB* (dans un environnement sombre, flou, enfumé, ...) ou de l'expert *Depth* (extérieur, longue distance, ...). C'est pourquoi les approches que nous proposons tentent de

garder les experts indépendants (c'est-à-dire d'éviter les architectures hybrides) tout en leur permettant de s'entraider les uns les autres.

Les architectures que nous proposons sont illustrées dans la Figure 5.14. L'opérateur "*" indique que nous utilisons une liste de patchs (proposé par le *RPN*) pour transformer un tenseur de caractéristiques globales en un batch de tenseurs de caractéristiques locales. L'opérateur "NMS" fait référence à la *Non Maximum Suppression*, récemment appelée "GreedyNMS" dans HOSANG et al. [2017] en opposition à la "Learnt NMS".

Nous appelons ces architectures des "*RCNN RGBD*". Voici les trois variantes que nous proposons :

— La **Fusion en U** est une version simple avec deux experts en parallèle. Les deux flux d'information sont uniquement fusionnés tout à la fin. La fusion est directement implémentée avec une *NMS*. Cette approche simple permet d'avoir un meilleur rappel que chaque expert pris seul car un réseau peut compléter les détections manquées de l'autre. Cependant, les fausses détections s'accumulent également, ce qui a pour conséquent de diminuer la précision de la méthode globale. Cette stratégie de fusion peut être appliquée à toutes les méthodes de vision par ordinateur qui proposent des détections.

— La **Fusion en X** est plus subtile. Ajouter une *NMS* commune après les *RPNs* permet de partager les propositions de détections avant la classification par les deux experts. Cette mise en commun des détections permet par exemple à l'expert *RGB* de classifier une détection proposée par l'expert *Depth*. De cette façon, les détections d'objets provenant des deux experts peuvent être classifiées dans l'espace des couleurs et dans l'espace des formes sans considération de l'origine de la détection. Les détections finales redondantes seront gérées par la *NMS* finale comme dans la Fusion en U.

— La **Fusion en Y** a pour objectif de n'utiliser qu'un seul *RPN*. Nous faisons ici l'hypothèse que le RPN *RGB-D* aura un espace plus riche pour détecter des objets. En pratique son entrée est constituée de la concaténation des sorties des couches de convolutions n°5 des réseaux de base (de type *VGG16*). Néanmoins, même si l'espace est plus riche, il est désormais plus grand. Cette ajout de complexité nécessite un entraînement plus long et plus d'exemples. La suite du réseau est constituée d'un seul expert *RGB-D* qui travaille sur le même tenseur que le RPN *RGB-D* .

Les architectures en U et en X sont structurellement indépendantes. Les entraînements des experts se font indépendamment l'un de l'autre. Ce n'est qu'au moment du test que les détections sont partagées. Cela permettrait d'ajouter d'autres modalités (en plus du *RGB* et de la *Depth*, comme le flux optique ou l'infrarouge...) sans avoir à ré-entraîner les premiers experts et sans trop changer la structure du *framework*. Au contraire, l'architecture en Y doit être entraînée avec toutes les données et nous ne pouvons pas y ajouter une nouvelle modalité sans ré-entraîner l'ensemble.

En détail, nous utilisons un réseau de neurones *VGG16* [CHATFIELD et al., 2014] comme base structurelle pour le *Faster-RCNN*. Il existe également des implémentations avec le "Residual Network" [HE et al., 2015] qui mène à des améliorations de détection sur la plupart des jeux de données mais à un coût computationnel très élevé. Chaque entraînement est effectué avec la descente de gradient stochastique, avec 10 000 itérations et un taux d'apprentissage constant de 0.001. Les poids du réseau sont initialisés avec un réseau pré-entraîné sur le jeu de données ImageNet [DENG et al., 2009]. Nous avons utilisé le *framework* Pytorch.

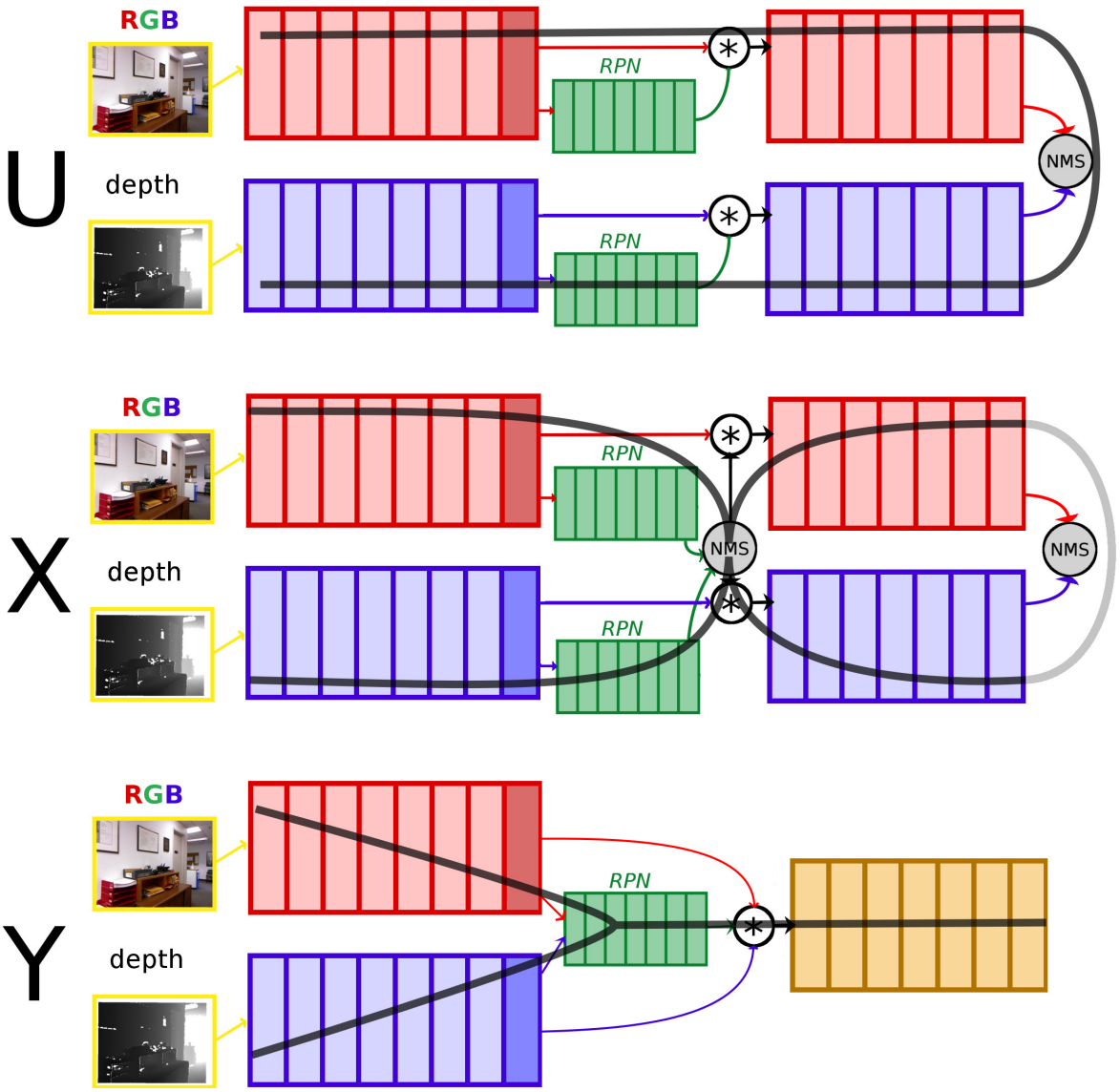


FIGURE 5.14 – Nos architectures de fusion, de haut en bas : fusion en U (*NMS* avec les sorties des classifieurs de chaque expert), fusion en X (*NMS* avec les sorties des *RPN* et *NMS* avec les sorties des classifieurs de chaque expert), fusion en Y (le *RPN* travaille sur la concaténation des tenseurs de chaque semi-expert).

5.4 Résultats de détection

Nous comparons nos méthodes à des méthodes de référence sur deux jeux de données *RGB-D* publiques : *RGBD People* et le plus récent *InOutdoor RGBD People* avec une caméra mobile. Nous évaluons également les performances de nos méthodes sur des expériences plus complexes de détection de personnes ce qui a aboutit à la création d'un nouveau jeu de données : ONERA.ROOM. Pour évaluer la qualité de nos méthodes nous utilisons les métriques standards de la détection de personnes que nous retrouvons également dans MEES et al. [2016] : Précision, Rappel, *Equal Error Rate* (*EER*), moyenne de Intersection-sur-l'Union (mIoU) et la moyenne harmonique la précision et du rappel (F1).

5.4.1 Résultats sur *RGBD People*

Nous testons tout d'abord nos méthodes sur *RGBD People* [SPINELLO et ARRAS, 2011]. Ce jeu de données est fait de plus de 3000 images *RGB-D* acquises avec 3 Kinect statiques positionnées à la verticale dans un grand hall d'université. Les conditions de luminosité sont stables. En suivant le même protocole que MEES et al. [2016] nous avons produit 5 partitionnements aléatoires (70% pour l'entraînement et 30% pour le test) nous donnons les résultats moyennés dans le Tableau 5.5 en ignorant les situations d'occlusions. Nous considérons une détection correcte si l'IoU avec la vérité terrain est supérieure à 0,6. Nous comparons également nos résultats à la méthode *HOD*[3] et *HGE*[4] de SPINELLO et ARRAS [2012].

TABLEAU 5.5 – *EER* sur *RGBD People* LUBER et al. [2011] pour différents détecteurs.

Méthode	Source	*EER*
HOD SPINELLO et ARRAS [2012]	*D*	56.3
HGE SPINELLO et ARRAS [2012]	*RGB-D*	87.4
Gating Net. MEES et al. [2016]	*RGBD-Optical flow*	89.3
Faster RCNN REN *et al. [2015]*	*D*	98.3
Faster RCNN REN *et al. [2015]*	RGB	98.4
U	*RGB-D*	98.4
Y	*RGB-D*	98.3
X	*RGB-D*	**98.6**

La méthode base, *Faster-RCNN* permet déjà un gain significatif par rapport à la méthode de MEES et al. [2016] (+9.1 points). La fusion en X vient légèrement améliorer le score (encore +0.2 points). Des exemples qualitatifs sont présentés dans la Figure 5.15.

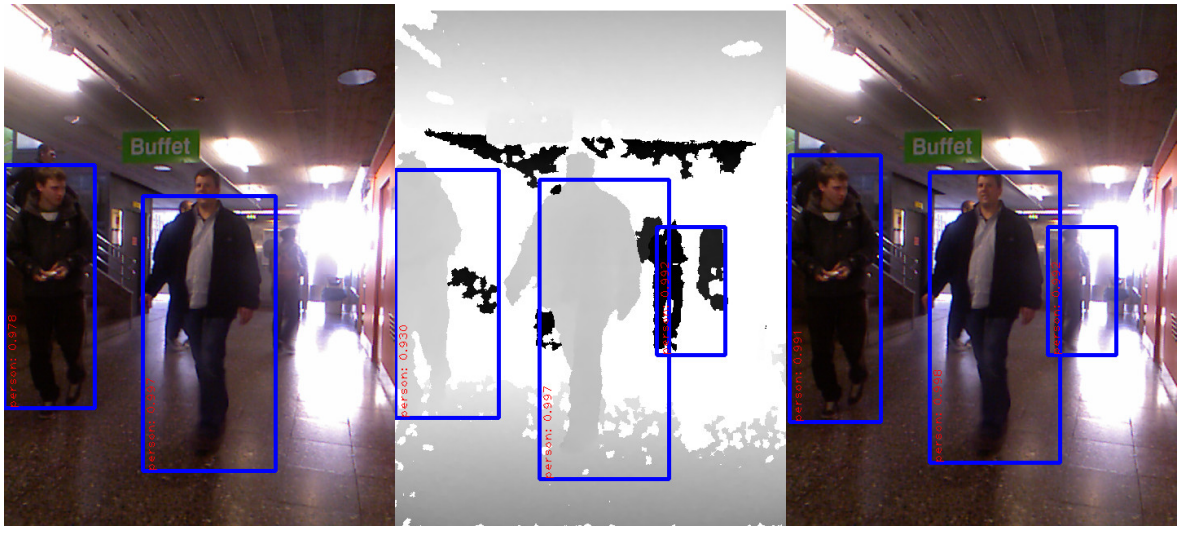


FIGURE 5.15 – Exemples de Prédictions sur *RGBD People*. Modèles entraînés et testés sur le jeu de données *RGBD People* LUBER et al. [2011]. De gauche à droite : expert *RGB*, expert *Depth*, fusion en X. L'expert *Depth* est parvenu à trouver l'ensemble des personnes dans l'image mais la fusion en X propose des boîtes englobantes qui sont mieux alignées avec la vérité terrain.

L'intérêt de la fusion *RGB-D* reste ici limité car les conditions d'acquisition sont bonnes,

3. *HOD* : *Histogram of Oriented Depths*
4. *HGE* : *Hierarchical Gaussian Process Mixtures of Experts*

ce qui conduit les méthodes RGB à être très performantes. Intéressons nous désormais à des jeux de données plus complexes.

5.4.2 Résultats sur *InOutdoor RGBD People*

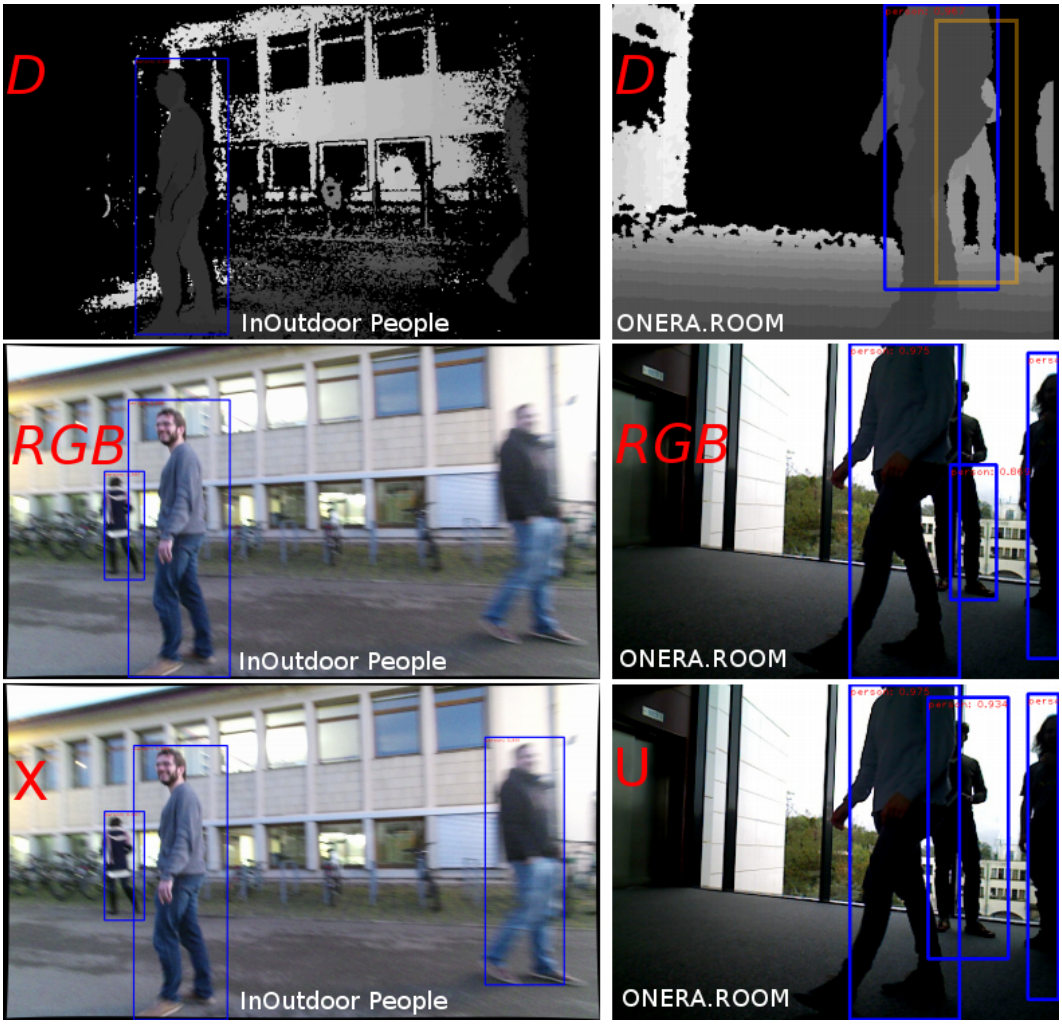


FIGURE 5.16 – Exemples de prédictions de nos méthodes multimodales. Modèles entraînés sur *InOutdoor RGBD People* MEES et al. [2016] et testés sur *InOutdoor RGBD People* et ONERA.ROOM.

La base de 8305 images *InOutdoor RGBD People* MEES et al. [2016] est obtenue par une caméra *RGB-D* embarquée sur un robot mobile qui navigue au milieu de personnes en mouvement. Il comporte des changements de luminosité brusques (intérieur/extérieur) et propose des séquences à différentes heures de la journée avec des variations de luminosité. Nous considérons une détection correcte si $IoU > 0,6$.

Les résultats sont donnés dans le tableau 5.6. Le Faster RCNN REN et al. [2015] seul, (*RGB* ou *D*) permet déjà d'obtenir un meilleur score que le précédent état de l'art MEES et al. [2016] avec un gain de 11,5 points sur la précision moyenne. Ce gain important peut en partie s'expliquer par un meilleur rappel (taux de détection) du *RPN*. Les stratégies de fusion permettent d'améliorer encore le score de 2,5 points en montant la précision moyenne à 94.4%. Un exemple de classification est disponible dans la colonne de gauche de la Figure 5.16. Il est intéressant de constater que la fusion en X (colonne de gauche, dernière ligne) possède une nouvelle détection. Ceci est permis grâce à la *NMS* intermédiaire, post-*RPN* : l'expert *Depth* a trouvé un objet qu'il n'a pas pu classifier mais l'a indiqué à l'expert *RGB*.

TABLEAU 5.6 – Performance sur le jeu de données *InOutdoor RGBD People* pour différents détecteurs. Les expériences étoilées correspondent à l'absence de luminosité (en pratique les pixels des canaux RGB sont à 0)

Méthode	Source	Précision/Rappel	PM	*EER*	mIoU	F1
Gating Net. MEES et al. [2016]	*RGB-D*	–/81.1	80.4	–	–	–
Faster RCNN REN et al. [2015]	RGB	**73.2**/94.2	91.9	90.1	**80.3**	**82.4**
Faster RCNN REN et al. [2015]	*D*	46.9/85.9	84.0	84.8	79.4	60.7
U	*RGB-D*	45.6/96.4	**94.4**	92.1	**80.3**	61.9
X	*RGB-D*	43.5/**96.5**	94.3	**92.4**	79.9	60.0
Y	*RGB-D*	59.4/93.3	90.2	90.1	80.2	72.6
U *	~~*RGB*~~-*D*	46.9/85.9	84.0	84.8	79.4	60.7
X *	~~*RGB*~~-*D*	45.9/85.9	84.1	84.8	79.4	59.9
Y *	~~*RGB*~~-*D*	n.a/0	n.a	n.a	0	n.a

L'expert *RGB*, lui, a pu classifier cette nouvelle détection. La fusion en U (colonne de droite) ne permet pas cet échange de détection pré-classification mais permet de réordonner, par score, l'ordre des patchs lors de la *NMS* finale : dans l'exemple ci-dessus le patch provenant de l'expert *Depth* est délaissé au profit du patch central de l'expert *RGB* ce qui permet à l'expert *Depth* de proposer un patch (en jaune) précédemment ignoré.

Nous notons que la stratégie Y a de moins bonnes performances que X ou U. En effet, l'entraînement est plus délicat, la tâche est plus compliquée car l'espace de caractéristiques est plus grand pour le *RPN* et l'unique classifieur.

De plus, conceptuellement ce type de réseau hybride ne peut pas gérer la perte d'une des sources comme le montre l'absence totale de détections pour Y* dans les expériences avec ablation de la modalité *RGB* du Tableau 5.6. Au contraire, U* et X* réagissent bien en égalisant les performances de l'expert *Depth* seul.

5.4.3 ONERA.ROOM

Création du dataset

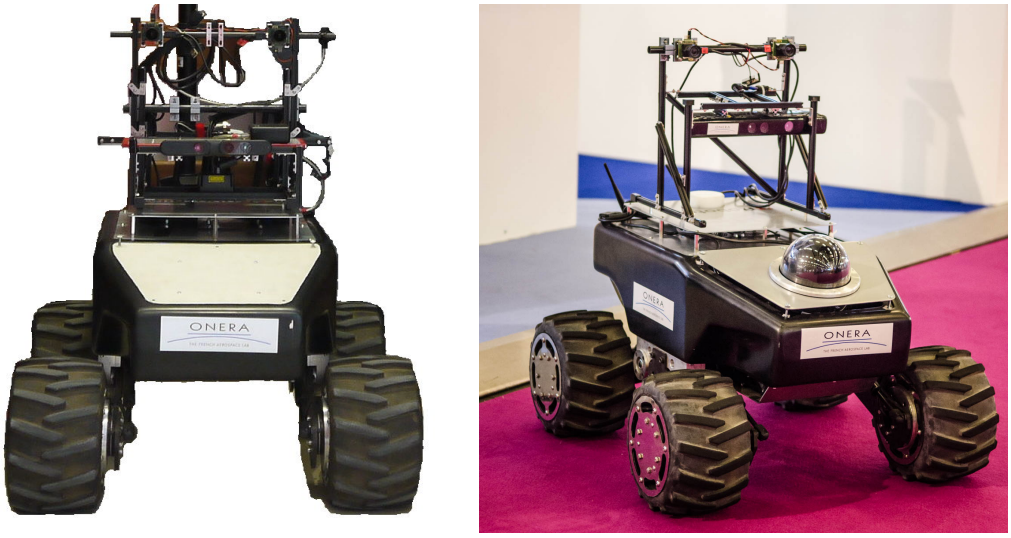


FIGURE 5.17 – Robby : le robot de l'ONERA utilisé pour les acquisitions d'ONERA.ROOM

ONERA.ROOM est un nouveau jeu de données comportant 27 séquences acquises par

divers capteurs RGB-D (Kinect v1, Xtion et RealSense) embarqués sur un robot mobile télécommandé (voir Figure 5.17). 23 séquences contenant des personnes ont été labellisées ce qui représente 27201 patchs de personnes répartis dans 35379 images. Orienté vers des scénarios robotiques pour la recherche et le sauvetage, le jeu de données ONERA.ROOM comporte des séquences acquises dans le noir, des séquences floutées par le mouvement du robot et des cas de personnes inconscientes au sol. Il comporte trois partitions de niveau de difficulté croissante : "Easy","Average" et "Hard", et est disponible publiquement à l'adresse http://jorisguerry.fr/ONERA.ROOM. L'ensemble de l'étiquetage a été effectué avec le logiciel Actanno V2 disponible à la même adresse. Ce logiciel est basé sur Actanno de WOLF et al. [2014], un logiciel d'étiquetage de données vidéos RGB que nous avons adapté aux données RGBD.

Résultats sur ONERA.ROOM

TABLEAU 5.7 – Performances sur le jeu de données ONERA.ROOM sur la partition "Easy" pour différents détecteurs entraînés sur *InOutdoor RGBD People* MEES et al. [2016]. Les expériences étoilées correspondent à l'absence de luminosité (en pratique les pixels des canaux RGB sont à 0)

Méthode	Source	Précision/Rappel	PM	*EER*	mIoU	F1
Faster RCNN REN et al. [2015]	*RGB*	61,0/96,1	91,2	**91,0**	**72,8**	74,6
Faster RCNN REN et al. [2015]	*Depth*	25,6/76,9	66,9	68,3	65,3	38,5
U	*RGB-D*	26,7/95,8	90,6	88,0	71,1	41,8
X	*RGB-D*	25,7/**96,6**	**91,3**	89,1	71,7	40,7
Y	*RGB-D*	**81,1**/92,1	87,1	90,3	71,7	**86,3**
U *	~~*RGB*~~*-D*	18,2/78,3	67,0	68,3	65,3	29,5
X *	~~*RGB*~~*-D*	25,0/77,0	66,7	68,1	65,3	37,8
Y *	~~*RGB*~~*-D*	n.a/0	n.a	n.a	0	n.a

Pour imposer une indépendance statistique entre les données d'entraînement et les données de test nous avons utilisé les modèles entraînés sur l'ensemble "*train*" de *InOutdoor RGBD People* pour les appliquer sur ONERA.ROOM. ONERA.ROOM comportant des séquences de difficultés variées, l'ensemble des résultats se réfère à la partition "*Easy*". Nous considérons une détection correcte si $IoU > 0,5$. Les tendances sont similaires aux expériences sur *InOutdoor RGBD People* (voir Figure 5.19 et Tableau 5.7). L'astérisque signifie que la source *RGB* était indisponible (image noire). Les fusions en U et en X se comportent bien face à ce défaut (cf. courbe fusion U * et fusion X * dans la Figure 5.19) qui retombent au niveau de performance de l'expert *Depth*. C'est une tendance que nous avons confirmée en diminuant progressivement la luminosité dans une acquisition spéciale d'ONERA.ROOM, illustrée dans la Figure 5.18. La fusion en X possède une meilleure précision moyenne que l'expert *RGB* (cf. tableau 5.7). Bien que l'*EER* de la fusion en Y soit bon, cette stratégie de fusion est incapable de faire la moindre détection en cas de perte de la source *RGB*. Il est important de souligner que nous n'avons pas cherché à entraîner l'architecture Y différement des autres architectures. Cependant, un entraînement avec plus d'images acquises dans le noir et de l'augmentation de données de type NYUDv2 "Eclipse" pourraient améliorer nettement les performances. Si nous regardons de plus près les valeurs de la Précision dans le Tableau 5.7 nous constatons que les performances des méthodes indépendantes (U et X) sont très faibles car elles proposent trop de fausses détections alors que la méthode Y est bien meilleure, au dessus-même du *Faster-RCNN RGB*.

Un exemple de classification est disponible dans la colonne de droite de la Figure 5.16. Les fusions en U et en X sont meilleures que l'expert *RGB* et aussi bonnes que l'expert *Depth*

en l'absence de lumière. Dans le cas d'une mission de secours une telle approche sera plus robuste aux conditions dégradées et imprévisibles.

Dans la Figure 5.20, la fusion en X sur ONERA.ROOM (colonne de droite) permet de faire une détection alors que les deux experts n'en proposent aucune (colonne de gauche et du milieu). De plus, la fusion en X parvient à différentier deux personnes très proches là où les experts *RGB* et *D* ne proposent qu'une seule détection.

Dans la Figure 5.21, la fusion en X sur ONERA.ROOM était sur le point de détecter une personne inconsciente au sol mais manque de précision dans le patch. Cependant c'est la seule méthode capable de détecter une personne accroupie (colonne de gauche), une personne à la limite de la carte de profondeur (colonne du milieu) et elle propose des meilleurs patchs (colonne de droite) que l'expert *Depth*.

Dans la Figure 5.22, la fusion en X sur ONERA.ROOM permet d'améliorer la proposition de patchs (colone de gauche et du milieu) et parvient même a détecter une personne inconsciente au sol (colonne de droite). Cependant, comme pour la Fusion en U, ces stratégies souffrent des *False Positive* (*FP*) – ou Faux(sse) Positif(ve) –s (voir colonne de gauche) car elles ne peuvent pas supprimer une détection proposée par un expert.

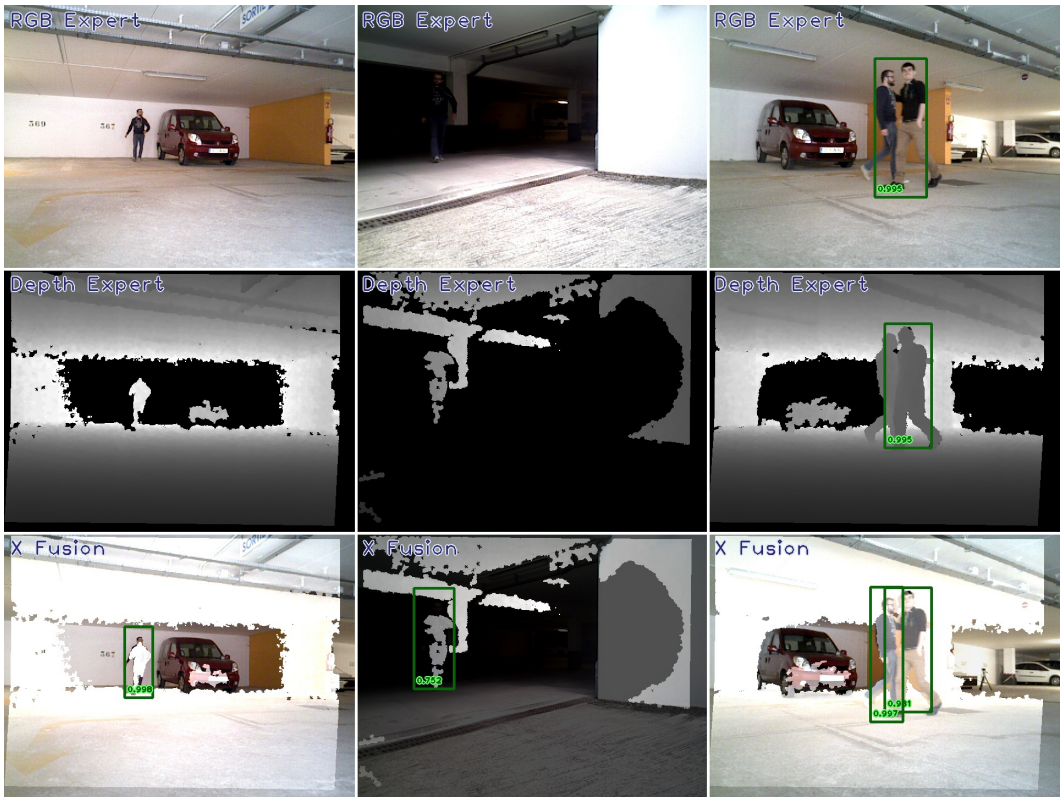


FIGURE 5.20 – Fusion en X sur ONERA.ROOM : dans le parking de l'ONERA.

FIGURE 5.21 – Fusion en X sur ONERA.ROOM : intervention avec des personnes au sol.

FIGURE 5.22 – Fusion en X sur ONERA.ROOM : évolution au milieu de personnes en mouvement.

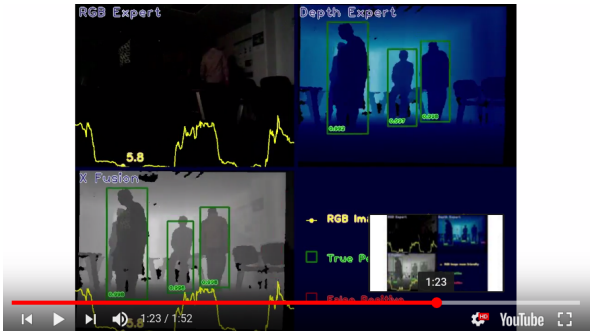


FIGURE 5.23 – Une vidéo présentant nos méthodes appliquées sur ONERA.ROOM est disponible en ligne à l'adresse http://jorisguerry.fr/ONERA.ROOM.

FIGURE 5.18 – Influence de la diminution de luminosité. Le dataset ONERA.ROOM propose une séquence statique dans laquelle le seul facteur qui change est l'intensité de la lumière ambiante. La courbe jaune indique la moyenne de l'intensité des pixels de l'image. La première colonne montre les bonnes détections (*True Positive* (*TP*) – ou Vrai(e) Positif(ve) –) de l'expert *RGB* ; la seconde colonne indique les *TP* de l'expert *Depth* et la troisième colonne montre les *TP* de la fusion en X. Les images de la dernière colonne sont faites de la moyenne de l'image RGB et de l'image *Depth* pour des questions d'esthétiques.

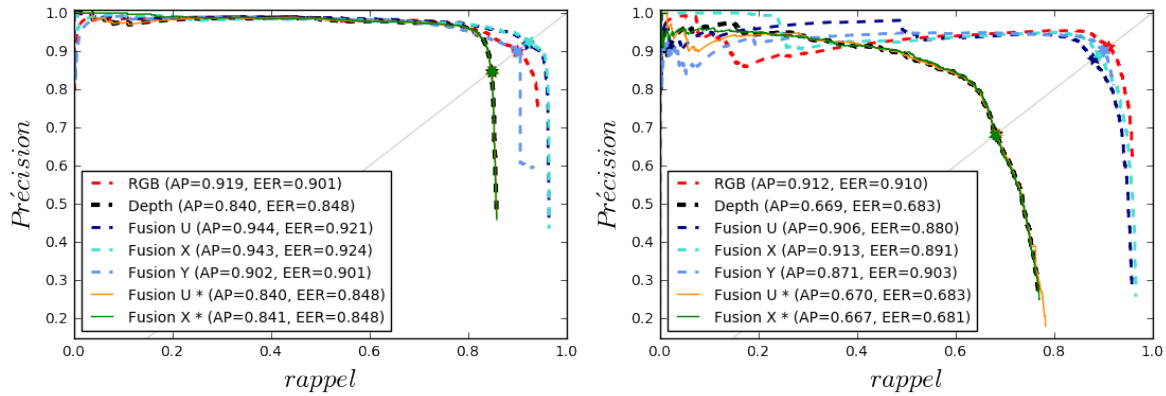


FIGURE 5.19 – Courbes Précision-Rappel sur les jeux de données *InOutdoor RGBD People* MEES et al. [2016] (à gauche) et ONERA.ROOM (à droite).

5.5 Conclusion du chapitre

Nous venons de présenter diverses approches pour utiliser des données multi-modales, spécifiquement les données *RGB-D*. Notre étude a tout d'abord mis en avant l'intérêt d'effectuer des encodages sur la carte de profondeur dans certaines situations et la chute de performance des experts *RGB* en cas de forte baisse de luminosité.

Nous avons ensuite proposé plusieurs stratégies – U, X et Y – pour fusionner des détecteurs de personnes basés sur l'architecture du *Faster-RCNN*. En plus de fusionner par *NMS* les détections finales (U), une de nos stratégies (X) partage les zones d'intérêts intermédiaires des *Faster-RCNNs* entre les experts pour compléter la phase de pré-détection. Cette configuration permet un gain de performance sans même modifier les architectures des réseaux et sans échanger de tenseurs de description grâce à une augmentation du rappel. Cette situation mène cependant aussi à l'échange des mauvaises détections et implique une baisse de la précision globale. Notre troisième architecture (Y) ne souffre pas d'une mauvaise précision, cependant, entraînée dans les mêmes conditions elle n'est pas capable de réagir face à une perte de modalité.

Nos méthodes présentent d'excellents résultats sur les jeux de données *RGBD People* et *InOutdoor RGBD People*, fixant de nouvelles références dans l'état de l'art de la détection de personnes sur des données *RGB-D*.

Enfin, ces travaux ont également mené à la réalisation d'un nouveau jeu de données robotiques *RGB-D* : ONERA.ROOM. Il permet d'évaluer les méthodes dans des situations critiques vis-à-vis des modalités *RGB-D* et proposent des séquences très variées avec plus de 35000 images annotées.

Dans ce chapitre, les données *RGB-D* ont été traitées comme des images 2D. Cependant, l'information volumique contenue dans la carte de profondeur n'a pas été vraiment exploitée. Même les encodages contenant un canal avec les angles des normales des surfaces au vecteur vertical ne font qu'un usage intermédiaire de l'aspect 3D. Dans le chapitre suivant, "Approche multi-vue pour la segmentation", nous utilisons cette information pour générer des vues alternatives de scènes *RGB-D*.

Dissémination

• Ces travaux ont fait l'objet d'une publication à l'*European Conference on Mobile Robotics* (*ECMR*) 2017 :

"Look At This One" Detection sharing between modality-independent classifiers for robotic discovery of people
Joris Guerry, Bertrand Le Saux, David Filliat

• Ces travaux ont fait l'objet d'une publication au colloque 2017 du Groupe d'Etudes du Traitement du Signal et des Images (GRETSI).

RCNN RGBD pour la détection de personnes en conditions difficiles
Joris Guerry, Bertrand Le Saux, David Filliat

• Ces travaux ont fait l'objet d'une publication commune au workshop *3D Shape Retrieval Contest* (SHREC) de *Eurographics* 2017.

SHREC'17 Track : 3D Hand Gesture Recognition Using a Depth and Skeletal Dataset
Q. De Smedt, H. Wannous, J.-P. Vandeborre, J. Guerry, B. Le Saux, D. Filliat

5.6 Références

BADRINARAYANAN, V., A. KENDALL et R. CIPOLLA. 2015, «SegNet : A Deep Convolutional Encoder-Decoder Architecture for Image Segmentation», arXiv preprint arXiv :1511.00561. URL http://arxiv.org/abs/1511.00561. 87

BOULCH, A., B. LE SAUX et N. AUDEBERT. 2017, «Unstructured point cloud semantic labeling using deep segmentation networks», Eurographics/3DOR. 87, 88, 90

CHATFIELD, K., K. SIMONYAN, A. VEDALDI et A. ZISSERMAN. 2014, «Return of the devil in the details : Delving deep into convolutional nets», dans British Machine Vision Conference, p. 6.1–6.12. 99

DE SMEDT, Q., H. WANNOUS et J.-P. VANDEBORRE. 2016, «Skeleton-based dynamic hand gesture recognition», dans Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops, p. 1–9. 93

DENG, J., W. DONG, R. SOCHER, L.-J. LI, K. LI et L. FEI-FEI. 2009, «Imagenet : A large-scale hierarchical image database», dans Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on, IEEE, p. 248–255. 99

DEVANNE, M., H. WANNOUS, S. BERRETTI, P. PALA, M. DAOUDI et A. DEL BIMBO. 2015, «3-d human action recognition by shape analysis of motion trajectories on riemannian manifold», IEEE transactions on cybernetics, vol. 45, n° 7, p. 1340–1352. 93

EIGEN, D. et R. FERGUS. 2015, «Predicting depth, surface normals and semantic labels with a common multi-scale convolutional architecture», dans Proceedings of the IEEE International Conference on Computer Vision, p. 2650–2658. 87

EITEL, A., J. T. SPRINGENBERG, L. SPINELLO, M. RIEDMILLER et W. BURGARD. 2015, «multi-modal deep learning for robust rgb-d object recognition», dans Intelligent Robots and Systems (IROS), 2015 IEEE/RSJ International Conference on, IEEE, p. 681–687. 90

GREEN, D. 2011, «A colour scheme for the display of astronomical intensity images», arXiv preprint arXiv :1108.5083. 88

GUPTA, S., R. GIRSHICK, P. ARBELÁEZ et J. MALIK. 2014, «Learning rich features from rgb-d images for object detection and segmentation», dans European Conference on Computer Vision, Springer, p. 345–360. 86, 87, 88, 89, 90

HANDA, A., V. PATRAUCEAN, V. BADRINARAYANAN, S. STENT et R. CIPOLLA. 2016, «Understanding real world indoor scenes with synthetic data», dans Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, p. 4077–4085. 87, 88, 90

HE, K., X. ZHANG, S. REN et J. SUN. 2015, «Deep residual learning for image recognition», arXiv preprint arXiv :1512.03385. 99

HOCHREITER, S. et J. SCHMIDHUBER. 1997, «Long short-term memory», Neural computation, vol. 9, n° 8, p. 1735–1780. 93

HOSANG, J., R. BENENSON et B. SCHIELE. 2017, «Learning non-maximum suppression», arXiv preprint arXiv :1705.02950. 99

LONG, J., E. SHELHAMER et T. DARRELL. 2015, «Fully convolutional networks for semantic segmentation», dans Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, p. 3431–3440. 87, 88, 90

LUBER, M., L. SPINELLO et K. O. ARRAS. 2011, «People tracking in RGBD data with on-line boosted target models», dans Intelligent Robots and Systems (IROS), 2011 IEEE/RSJ International Conference on, IEEE, p. 3844–3849. xi, 101

MEES, O., A. EITEL et W. BURGARD. 2016, «Choosing smartly : Adaptive multi-modal fusion for object detection in changing environments», dans Intelligent Robots and Systems (IROS), 2016 IEEE/RSJ International Conference on, IEEE, p. 151–156. 87, 88, 89, 96, 97, 98, 100, 101, 102, 103, 104, 109

OHN-BAR, E. et M. TRIVEDI. 2013, «Joint angles similarities and hog2 for action recognition», dans Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops, p. 465–470. 93

OREIFEJ, O. et Z. LIU. 2013, «Hon4d : Histogram of oriented 4d normals for activity recognition from depth sequences», dans Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, p. 716–723. 93

REN, S., K. HE, R. GIRSHICK et J. SUN. 2015, «Faster r-cnn : Towards real-time object detection with region proposal networks», dans Advances in neural information processing systems, p. 91–99. 88, 92, 97, 98, 101, 102, 103, 104

SILBERMAN, N., D. HOIEM, P. KOHLI et R. FERGUS. 2012, «Indoor segmentation and support inference from rgbd images», Computer Vision–ECCV 2012, p. 746–760. 89

SPINELLO, L. et K. O. ARRAS. 2011, «People detection in rgb-d data», dans Intelligent Robots and Systems (IROS), 2011 IEEE/RSJ International Conference on, IEEE, p. 3838–3843. 101

SPINELLO, L. et K. O. ARRAS. 2012, «Leveraging RGBD data : Adaptive fusion and domain adaptation for object detection», dans Robotics and Automation (ICRA), 2012 IEEE International Conference on, IEEE, p. 4469–4474. 101

WOLF, C., E. LOMBARDI, J. MILLE, O. CELIKTUTAN, M. JIU, E. DOGAN, G. EREN, M. BACCOUCHE, E. DELLANDRÉA, C.-E. BICHOT et al.. 2014, «Evaluation of video activity localizations integrating quality and quantity measurements», Computer Vision and Image Understanding, vol. 127, p. 14–30. 104

Chapitre 6

Approche multi-vue pour la segmentation sémantique

« La réalité n'est qu'un point de vue. »

Philip K. Dick, 1928-1982,

Sommaire

6.1 Introduction 116
6.2 La méthode : SnapNet-R 117
6.2.1 Le générateur d'images 2D virtuelles 117
6.2.2 La segmentation sémantique 118
6.2.3 La reprojection en 3D 118
6.2.4 L'évaluation de la performance 118
6.3 Application mono-vue : SUNRGBD et NYUDv2 118
6.3.1 Particularité des données mono-vues 119
6.3.2 Tâche 1 : Estimation d'agencement, génération de vues virtuelles 120
6.3.3 Tâche 1 : Estimation d'agencement, Résultats 121
6.3.4 Tâche 2 : Segmentation sémantique, génération de vues virtuelles 122
6.3.5 Tâche 2 : Segmentation sémantique, Résultats 123
6.4 Application multi-vue : 3DRMS Challenge 128
6.4.1 "Classif 2D" : Reconstruction basée sur la navigation et étiquetage direct 129
6.4.2 "Classif 2D-3D" : Reconstruction basée sur la navigation et étiquetage 3D 130
6.4.3 "Classif 3D" : Reconstruction hors-ligne et sémantisation 3D . . . 131
6.4.4 Résultats qualitatifs 131
6.5 Conclusion du chapitre 135
6.6 Références 135

6.1 Introduction

Nous venons d'étudier plusieurs stratégies pour utiliser des données *RGB-D*. Ces approches considéraient l'information de profondeur comme une carte 2D. Même si certains encodages comme l'encodage *HHA* traduisent l'information spatiale contenue dans la carte de profondeur nous ne profitons toujours pas du fait qu'une image *RGB-D* peut-être reprojetée en 3D. Nous parlons d'ailleurs parfois de données 2.5D (voir exemple dans la Figure 6.1).

Il existe de nombreuses approches qui considèrent directement l'information 3D comme entrée pour la sémantisation (voir l'état de l'art consacré à la segmentation de nuages de points 3D dans le Chapitre 3). En effet, il existe de plus en plus de jeux de données 3D mis à disposition de la communauté grâce à des acquisitions LIDAR notamment mais aussi en faisant usage d'approches de type *SLAM*. Notre jeu de données ONERA.ROOM propose également des reconstructions 3D basées sur l'algorithme *RGBD-SLAM* de ENGEL et al. [2014].

FIGURE 6.1 – Exemple d'image *RGB-D* vue en 3D

Il existe donc tout un panel de méthodes 3D que nous pouvons appliquer à nos données 2.5D. Parmi ces méthodes, les approches multi-vues permettent de faire le travail de sémantisation avec des techniques 2D. C'est pourquoi nous avons choisi ce genre de d'approche pour réutiliser l'expérience acquise sur les données 2D lors de nos premiers travaux. Nous allons pour cela définir la méthode SnapNet-R dans la Section 6.2. SnapNet-R a pour objectif d'aborder la problématique de segmentation sémantique 3D.

Nous étudierons alors deux cas d'application :

1. Section 6.3 : dans la continuité de nos travaux sur les données *RGB-D* nous appliquerons SnapNet-R comme méthode d'augmentation des données sur des données mono-vues comme les jeux de données NYUDv2 et SUNRGBD.
2. Section 6.4 : nous appliquerons également SnapNet-R en mode CNN multi-vue dans un cadre plus classique de la robotique mobile au travers du challenge "*3D reconstruction*

meets Semantic 2017 (*3DRMS*)"

6.2 La méthode : SnapNet-R

Nos travaux se basent sur la méthode SnapNet de BOULCH et al. [2017] qui a établi des résultats de référence pour des données urbaines et de télédétection telles que *Semantic3D* [HACKEL et al., 2017]. Nous proposons de nouvelles stratégies d'échantillonnage multi-vues et une meilleure intégration de l'image et de la géométrie. Mais surtout, nous montrons comment la reconstruction de la structure 3D et la sémantique 2D peuvent bénéficier mutuellement l'une de l'autre :

La 3D au service de la 2D : Instantanément, lorsque le robot obtient une seule capture *RGB-D* de son environnement, nous générons de nouvelles vues virtuelles qui sont cohérentes avec la structure 3D de la scène perçue. Cela améliore la segmentation sémantique 2D grâce à une augmentation des données d'entraînements ainsi qu'au moment de la prédiction avec une procédure de vote.

La 2D au service de la 3D : Les nuages de points 3D peuvent être très bruités. Au moment de la reconstruction depuis les images 2D acquises, l'ajout de carte sémantique 2D permet de filtrer les points a utiliser pour la reconstruction 3D. Ceci permet d'obtenir des nuages de points plus précis, qui seront à leur tour plus faciles à étiqueter avec des stratégies de segmentation sémantique 3D.

L'approche SnapNet consiste à indépendamment faire la segmentation sémantique 2D d'images virtuelles générées depuis la scène 3D puis de reprojeter efficacement la classe par pixel sur le point 3D correspondant. Il y a donc trois composantes essentielles à cette méthode : le générateur d'images 2D virtuelles, la méthode de segmentation sémantique 2D et le système de reprojection 3D. Ce "pipeline" de traitement est illustré dans la Figure 6.2

La nouvelle version de SnapNet que nous proposons a pour objectif de s'appliquer à des données d'origine robotique ce qui explique sa nouvelle dénomination : SnapNet-R(obot).

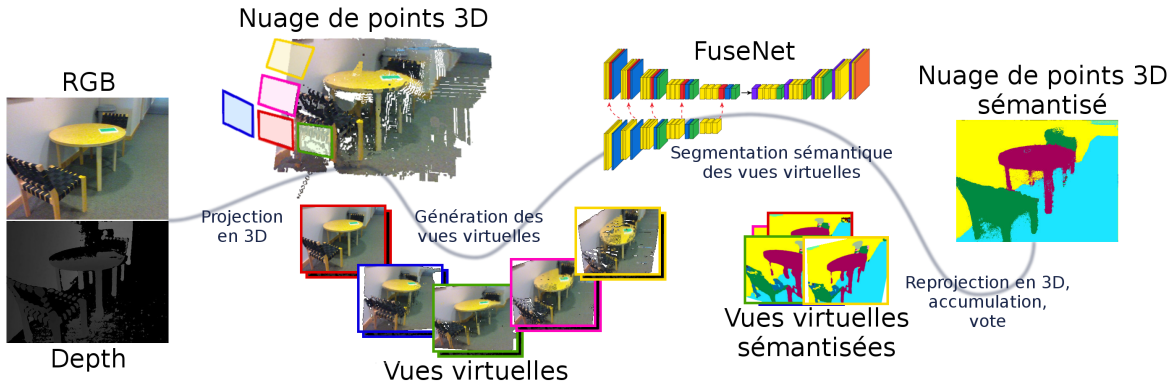


FIGURE 6.2 – Processus complet de SnapNet-R

6.2.1 Le générateur d'images 2D virtuelles

Le générateur d'images repose sur une stratégie d'observation de la scène 3D qui propose des points de vue. Ces points de vue doivent être adaptés à la scène considérée, nous y reviendrons dans les sections suivantes en détaillant les stratégies dans chaque cas.

De manière générale, le générateur doit prendre en compte l'échelle d'observation. Il faut en effet se rapprocher des objets de petites tailles (comme les voitures dans une scène urbaine) et être assez loin des grands objets (pour englober l'ensemble d'un bâtiment par exemple).

Cette distance caractéristique d'un objet permettra à la méthode de segmentation d'avoir des informations de contexte sur l'objet.

6.2.2 La segmentation sémantique

Pour la segmentation sémantique des images générées par la stratégie de génération d'images virtuelles, l'approche originale utilise deux SegNet [BADRINARAYANAN et al., 2015], un pour chaque modalité (caractéristiques *RGB* et images composites) et les fusionne par un module de correction résiduelle AUDEBERT et al. [2016]. Les caractéristiques géométriques sont fabriquées à la main : les normales, le bruit local et la profondeur vis-à-vis de la caméra virtuelle. Ainsi, ce protocole implique quatre étapes différentes : l'entraînement de l'expert *RGB*, la création de l'image composite, l'entraînement de l'expert composite et enfin l'entraînement du module de fusion.

Ici, nous remplaçons tous ces composants par un seul FuseNet SF5, introduit par HAZIRBAS et al. [2016], qui prend directement la carte *RGB* et la profondeur comme entrées et ne nécessite donc qu'une seule étape d'entraînement (pas de pré-traitement de l'information de profondeur, pas d'encodage, pas de correction résiduelle).

6.2.3 La reprojection en 3D

La reprojection de l'information sémantique 2D inférée par le FuseNet est ensuite reprojetée dans le nuage de point. Pour cela, BOULCH et al. utilisent une astuce numérique, encodant un identifiant unique par pixel sous la forme d'une couleur. En effet, un pixel est constitué de 3 canaux pouvant prendre 256 valeurs chacun. Un pixel peut donc être vu comme un nombre à 3 chiffres en base 256. Cette astuce permet de n'effectuer qu'une seule fois le calcul du passage de la 3D à la 2D. En l'appliquant également aux couleurs uniques nous connaissons dans l'image 2D quel est le point correspondant dans la scène 3D.

Nous n'avons pas changé cette composante de SnapNet pour produire SnapNet-R.

6.2.4 L'évaluation de la performance

L'évaluation sera détaillée dans chaque application à venir car elle dépend du contexte d'application : souhaite-t-on évaluer une segmentation 2D, une segmentation 3D, une reconstruction 3D ?

Néanmoins, que ce soit en 2D ou en 3D, la segmentation sémantique s'évalue toujours avec les métriques définies dans le Chapitre 2.

6.3 Application mono-vue : SUNRGBD et NYUDv2

Notre premier cas d'application concerne les scènes pour lesquelles nous n'avons qu'une seule vue *RGB-D* disponible. Pour une seule image, nous générons d'autres vues de la scène de sorte que les vues soient géométriquement compatibles avec la nature 3D de la scène observée. Ainsi, toutes les vues correspondent à ce qui serait vu de la scène d'un point de vue différent. Cela peut être considéré comme une augmentation de données, mais diffère de l'augmentation de données standard (telle que le *cropping*, les translations, les effets mirroirs, ...) car elle ne profite pas que des informations contenues dans le plan d'image 2D mais peut également extraire des connaissances grâce à la structure spatiale 3D au travers de l'ensemble des différentes apparences. Par exemple, par la variation des points de vue un

objet légèrement caché pourra se "détacher" des objets de premier plan et ainsi apparaître dans un contexte local différent de celui d'origine.

Nous étudierons l'intérêt de cette méthode sur deux tâches différentes :

1. L'estimation d'agencement
2. La segmentation sémantique

6.3.1 Particularité des données mono-vues

Une seule image *RGB-D* ne détermine l'aspect d'une scène que d'un seul point de vue. Hors, la couleur et la réflexion de la lumière infrarouge du capteur actif dépendent fortement de l'angle d'observation avec les surfaces observées et des sources de lumière de la pièce. Une scène peut donc fortement varier d'un point de vue à l'autre. Ceci nécessite une grande capacité d'adaptation aux conditions de luminosité pour le réseau de segmentation. De plus, la forme des objets et leur opacité rend inobservables des objets lorsqu'ils sont derrière d'autres objets.

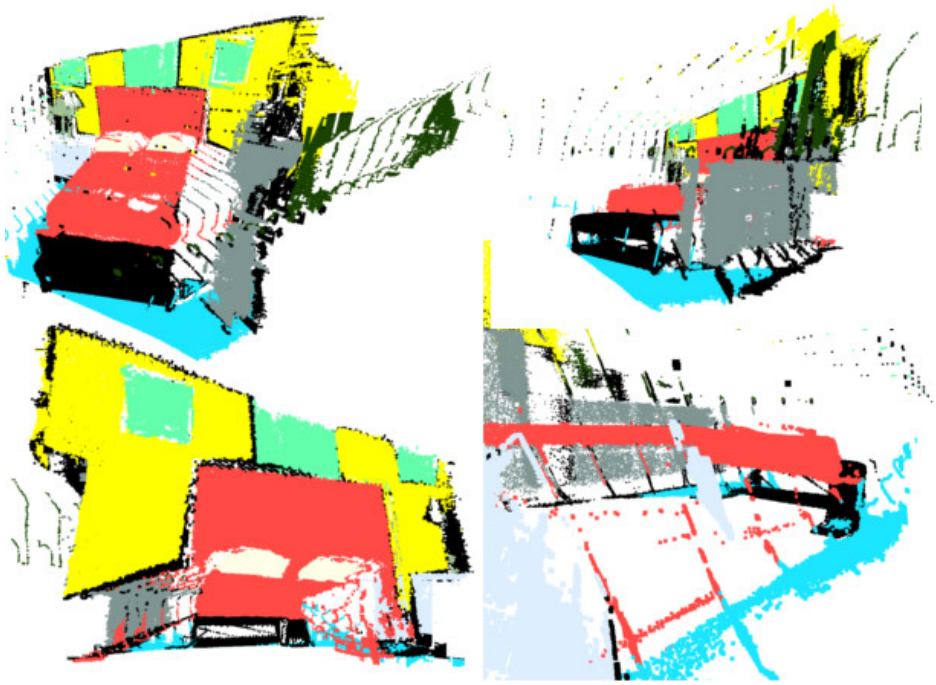


FIGURE 6.3 – Exemples d'une mauvaise stratégie d'observation sur des données mono-vues. En haut à gauche : un bon exemple proche du point de vue de la caméra, en haut à droite : vue au travers du meuble, en bas à gauche : vue de derrière et en bas à droite : vue de côté du lit.

Autrement dit, une seule observation *RGB-D* d'une scène donne accès à une représentation 3D très pauvre de la scène considérée. Par conséquent, une mauvaise stratégie d'observation peut mener à des vues artificielles inutiles, voire génératrices d'erreur (exemple dans la Figure 6.3)

Synthèse du nuage de points 3D En utilisant les caractéristiques intrinsèques des capteurs *RGB-D* il nous est possible d'aligner l'information *RGB* sur la carte de profondeur. Ensuite, en utilisant la matrice intrinsèque du capteur de profondeur nous pouvons reprojeter l'information couleur dans un espace 3D.

SUNRGBD et NYUDv2 L'ensemble de données SUNRGBD est composé uniquement d'images de vue unique (une vue d'ensemble est disponible dans la Figure 6.4). Cela implique une stratégie d'échantillonnage différente de celle d'une scène de nuage de points dense complète. En outre, l'approche SnapNet a d'abord été conçue pour fonctionner sur quelques grandes scènes et extraire autant d'images de chaque scène que vous le souhaitez. Mais SUNRGBD est composé de 5285 images d'entraînement et 5050 images de test où chacun d'entre eux est en fait une scène. Ainsi, nous ne pouvons pas générer des centaines d'échantillons dans chaque scène sans risquer de faire exploser le nombre final d'images générées. C'est pourquoi nous proposons diverses stratégies de génération propres aux données mono-vues avec moins de 20 vues par scène.

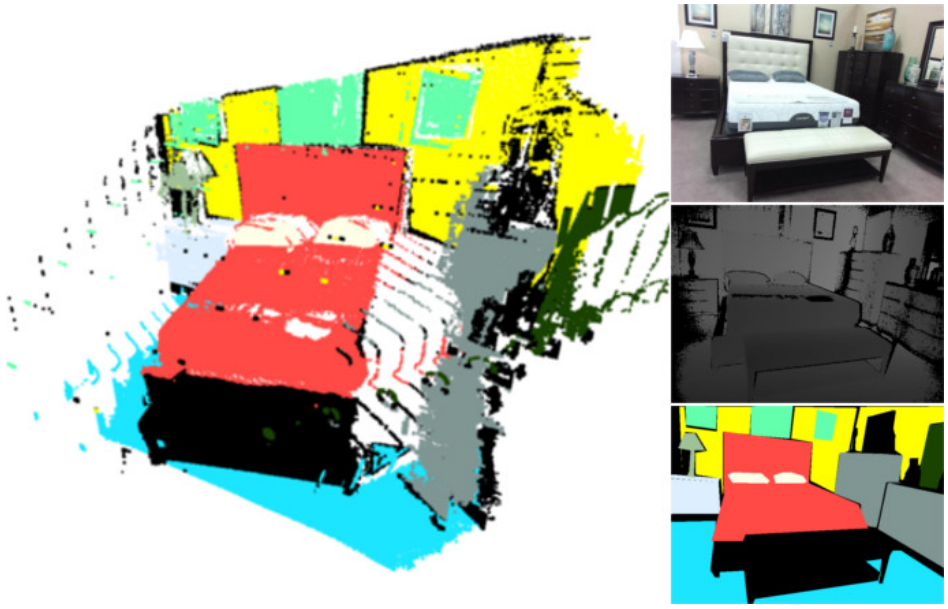


FIGURE 6.4 – Aperçu du jeu de données SUNRGBD : à gauche le nuage de point sémantisé et dans la colonne droite : l'image RGB, la carte de profondeur et la vérité terrain (voir Figure 6.10 pour la légende des étiquettes).

À propos des données bruitées Les capteurs de profondeur souffrent des surfaces brillantes comme les miroirs, les fenêtres, les écrans ou les objets en verre et la mesure peut être très bruitée. Cela entraîne des artefacts noirs sur la carte de profondeur et donc des trous dans les images *RGB* qui résultent de la re-projection 3D. C'est pourquoi nous avons appliqué des prétraitements d'*inpainting* : basés sur la méthode Navier-Stokes BERTALMIO et al. [2001] pour les images en profondeur et en utilisant la couleur moyenne des pixels voisins non-noirs sur les images *RGB* (avant entraînement et test).

6.3.2 Tâche 1 : Estimation d'agencement, génération de vues virtuelles

La première tâche concerne l'estimation d'agencement. Elle a pour but de produire une segmentation sémantique 3D avec les 3 classes suivantes : *wall, floor, stuff*. Elle a de l'intérêt notamment dans l'exploration robotique pour définir les limites d'explorations, éviter les obstacles ou faciliter la détection d'objets.

Dans notre première version de SnapNet-R nous faisons usage d'un maillage du nuage de points pour éviter dans les grandes scènes de "voir à travers les murs", à l'instar de la méthode SnapNet originale. Après avoir maillé le nuage de points issu d'une scène *RGB-D* mono-vue nous supprimons les triangles surfaciques dont le ratio entre le côté le plus grand et le côté le

plus petit est supérieur à un certain seuil (10 en pratique). Cette technique est illustrée dans la Figure 6.5.

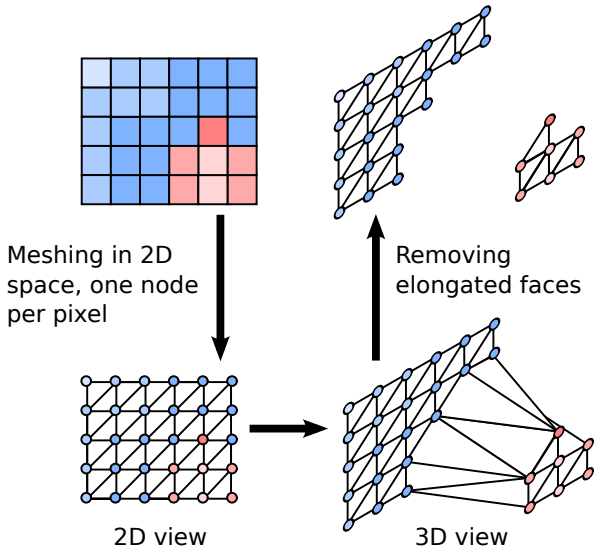


FIGURE 6.5 – Filtrage du "meshing" pour l'estimation d'agencement

Comme indiqué précédemment, un nuage de points 3D généré à partir des données *RGB-D* ne peut pas être échantillonné aléatoirement. De mauvais exemples de sélection aléatoire de points de vue dans l'ensemble de données SUNRGBD sont disponibles dans la Figure 6.3. Notre première stratégie produit 18 caméras virtuelles à 7 et 8 mètres de distance d'un point positionné à 6m devant la caméra, avec des angles d'azimuth de -20°, 0°, +20°et des angles d'élévation de 0°, 15°et 30°. Cette stratégie est illustrée dans la Figure 6.6.

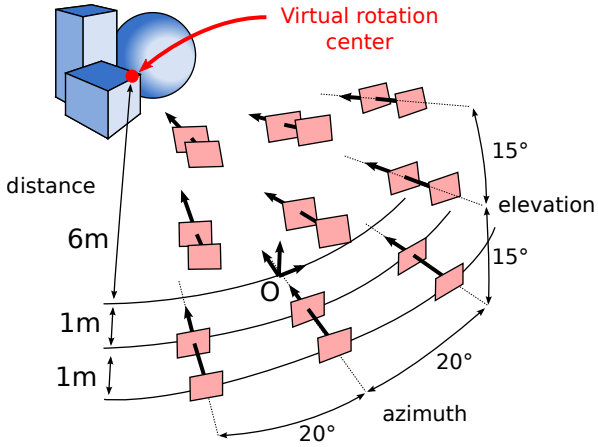


FIGURE 6.6 – Illustration de la stratégie multi-vue pour l'estimation d'agencement

6.3.3 Tâche 1 : Estimation d'agencement, Résultats

Des résultats qualitatifs sont disponibles dans la Figure 6.7 et nous nous comparons à d'autres méthodes dans le Tableau 6.1. Nous avons utilisé l'encodage "*Depth Composite*" de BOULCH et al. [2017] pour l'expert *Depth*. Dans la Figure 6.7, nous observons deux scènes : une salle de cours avec des rangées de chaise et une salle avec un bureau. Ces images montrent des situations de réussite de notre méthode. Comme nous pouvons le voir dans le Tableau 6.1, les performances globales de la méthode proposée sont faibles comparées à HANDA et al. [2016] mais reste néanmoins comparables, notamment sur la classe *floor*.

TABLEAU 6.1 – Résultats de l'estimation d'agencement sur le jeu de données SUNRGBD.

Méthode	Source	*floor*		*wall*		*stuff*	
		P	mIoU	P	mIoU	P	mIoU
SceneNet DHA [HANDA et al., 2016]	D	71.1	~	89.2	~	~	~
SUNRGBD DHA [HANDA et al., 2016]	D	94.0	~	89.7	~	~	~
SnapNet-R	RGB	83.0	56.1	64.8	43.2	76.0	43.4
SnapNet-R	D	86.1	64.8	62.2	40.9	79.4	45.6
SnapNet-R	RGB-D	88.2	76.6	60.1	50.6	84.5	65.6

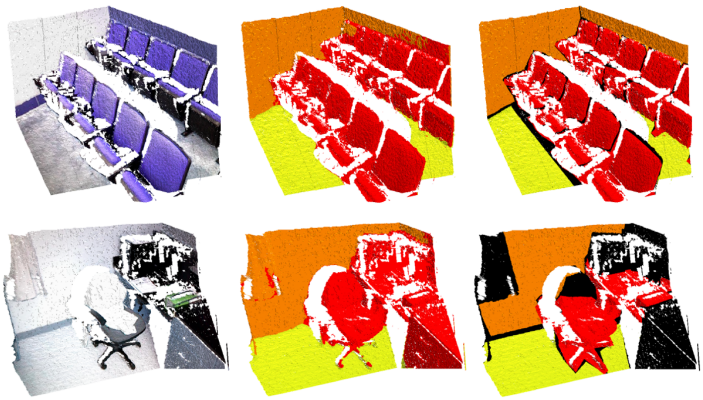


FIGURE 6.7 – Exemple d'estimation d'agencement avec SnapNet. Orange : *walls*, jaune : *floor*, rouge : *stuffs*, noir : *ignored*

Cette première expérience sur la tâche d'estimation d'agencement nous a permis de mettre en avant différentes lacunes de notre méthode. C'est en regardant les données générées par la première stratégie de génération que nous avons remarqué que les images virtuelles étaient mal cadrées et ignoraient certaines zones de l'espace. C'est pourquoi nous avons changé la stratégie de génération pour rester au plus près de la caméra d'origine.

6.3.4 Tâche 2 : Segmentation sémantique, génération de vues virtuelles

À la suite de la tâche 1 nous avons opté pour une nouvelle stratégie de génération de vues virtuelles. De plus, utiliser 18 vues par scène restait encore très conséquent pour l'entraînement. Nous avons donc choisi une stratégie d'échantillonnage qui génère seulement 5 images pour chaque scène, ce qui donne 26425 images pour l'entraînement et 25250 images pour le test. NYUDv2 étant partie intégrante de SUNRGBD nous lui avons appliqué les mêmes prétraitements et stratégie multi-vue.

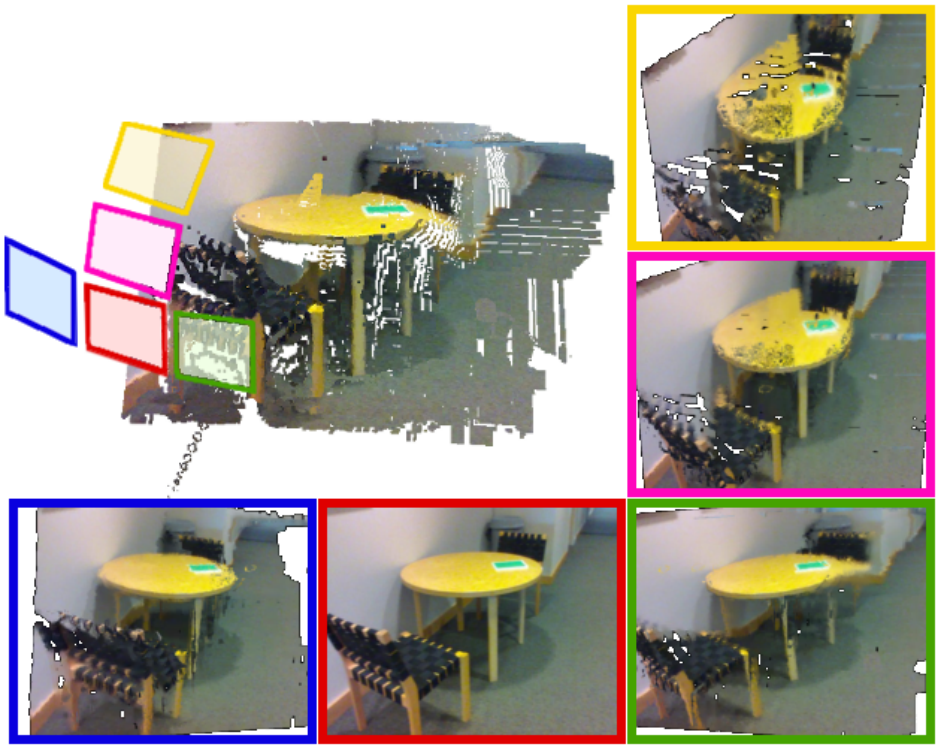


FIGURE 6.8 – Illustration de la stratégie d'observation sur des données mono-vues (scène extraite du dataset SUNRGBD). Les vraies distances et angles ne sont pas respectés pour des questions d'illustration.

Stratégie de sélection des points de vue Pour éviter les problèmes de cadrage et de zones involontairement censurées nous avons avons arbitrairement défini 5 positions de caméra :

- le premier point de vue est le point de vue original de la caméra,
- du premier point de vue, nous obtenons 2 points de vue supplémentaires en changeant l'angle d'azimut de +10°et -10°,
- du premier point de vue, nous obtenons 2 points de vue supplémentaires en changeant l'angle d'attitude de +10°et +20°.

Cette stratégie d'échantillonnage est illustrée dans la Figure 6.8. Aucun point de vue ascendant n'est choisi parce que les objets se tiennent principalement sur le sol et sont mieux visibles par dessus. En outre, les scènes sont des scènes d'intérieur donc les objets sont à portée de main et, par conséquent, regarder en haut pourrait nous les faire manquer (comme c'était le cas avec la stratégie de la tâche 1).

6.3.5 Tâche 2 : Segmentation sémantique, Résultats

Le "*class imbalance problem*"

Comme nous pouvons le voir sur la Figure 6.9 il existe un fort déséquilibre dans la répartition des classes des jeux de données SUNRGBD et NYUDv2. Bien que cette figure illustre la répartition des instances de chaque objet elle est également représentative de la distribution des pixels à laquelle il faudrait ajouter les pixels de classe "mur","sol" et "plafond" qui sont encore plus présentes que les pixels appartenant à des chaises. Pour ainsi dire, 70% des pixels sont des pixels appartenant à des murs.

Ce déséquilibre important est problématique dans l'entraînement d'un réseau de neurones. En effet, chaque fois que la fonction de perte est évaluée sur une prédiction il devrait y avoir

en moyenne 70% de prédiction de mur. Cette situation, si rien n'est fait pour le prendre en compte, mène à une situation où le réseau de neurones ne prédit que du mur et arrive ainsi à 70% de précision globale. C'est pourquoi nous utilisons également la précision moyenne (moyenne des précisons par classe). Pour que le réseau de neurones puisse se libérer de ce biais statistique nous appliquons des coefficients sur les composantes de la fonction de perte propres à chaque classe. Cette solution est également employée dans EIGEN et FERGUS [2015]; HAZIRBAS et al. [2016] en utilisant comme poids la fréquence médiane des classes divisée par la fréquence de la classe considérée.

$$w_i = \frac{\text{fréquence}_{\text{médiane}}(\text{C})}{\text{fréquence}_{c_i}} \tag{6.1}$$

Cette étape est primordiale pour assurer la convergence du réseau de neurones. Nous avons également essayé d'autres pondérations sans succès comme l'inverse de la fréquence de la classe, $\text{fréquence}_{\max}/\text{fréquence}_{c_i}$ ou $1+\text{fréquence}_{\max}/\text{fréquence}_{c_i}$. Cependant ces expériences ne furent que sommaires et mériteraient une étude approfondie.

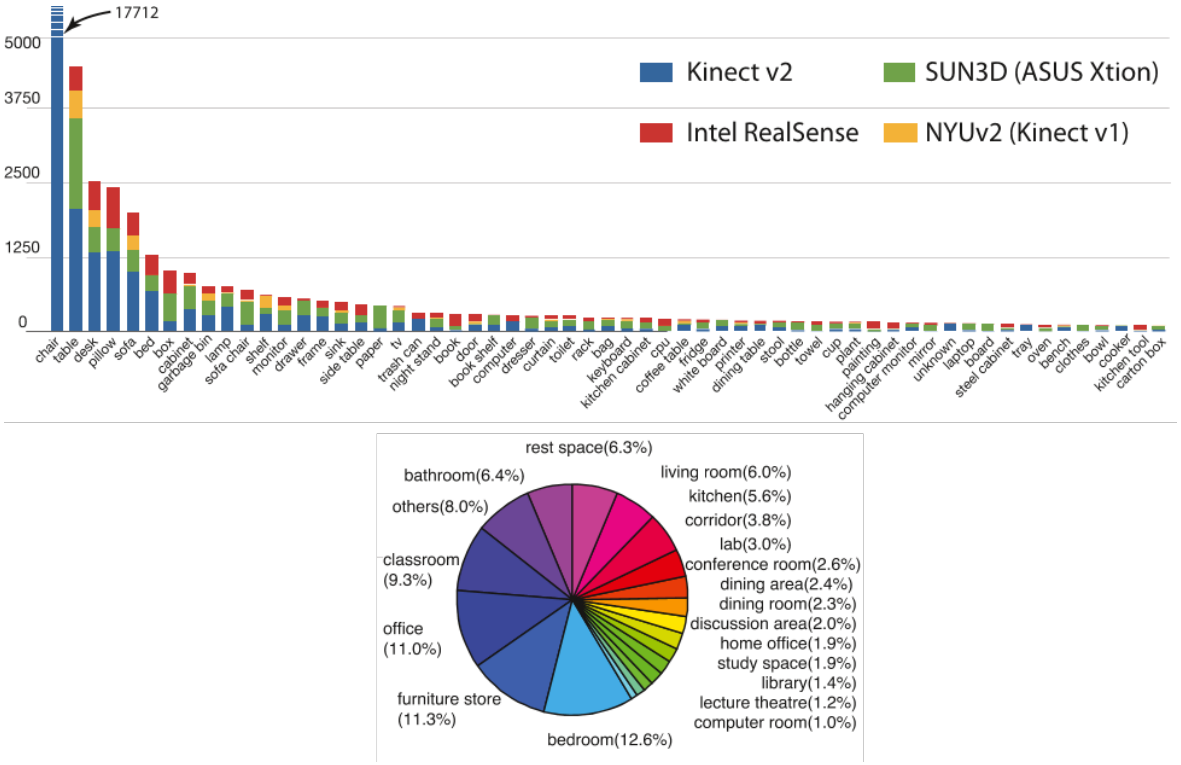


FIGURE 6.9 – Répartition des objets dans SUNRGBD. Image extraite de SONG et al. [2015]

SUNRGBD : segmentation sémantique 2D

La segmentation sémantique de SUNRGBD est une tâche difficile. La meilleure méthode à notre connaissance, le Context-CRF de LIN et al. [2017] n'atteint que les 42,3% de mIoU. Pour cela ils utilisent une architecture très lourde (brièvement présentée dans le Chapitre 3) couplée avec un CRF Dense KRÄHENBÜHL et KOLTUN [2011].

Les résultats obtenus sont montrés dans le Tableau 6.2. Précisons qu'ils ne font pas usage du moindre post-traitement de raffinement ou de *CRF*. Notre méthode consiste seulement à mettre dans un seul batch les 5 vues virtuelles de notre scène *RGB-D* redimensionnées en 224x224, effectuer la prédiction sémantique à basse résolution, étirer ces prédictions par

Expérience	Entraînement		Test		P	PM	mIoU
	Pré-proc.	Augm.	Pré-proc.	Augm.			
LSTM-CF [LI et al., 2016] (RGB)	✗	✗	✗	✗	–	48.1	–
FCN 8s [LONG et al., 2015] (RGB)	✗	✗	✗	✗	68.2	38.4	27.4
Bayesian SegNet [KENDALL et al., 2015] (RGB)	✗	✗	✗	✗	71.2	45.9	30.7
Context-CRF [LIN et al., 2017] (*RGB-D*)	✗	✗	✗	✗	**78.4**	53.4	**42.3**
*FuseNet SF5[HAZIRBAS et al., 2016] (*RGB-D*)	✗	✗	✗	✗	76.3	48.3	37.3
DFCN-DCRF [JIANG et al., 2017] (*RGB-D*)	✗	✗	✗	✗	76.6	50.6	39.3
*1 FuseNet SF5	✗	✗	✗	✗	76.88	52.61	39.17
1 FuseNet SF5	✗	✗	✗	✗	77.21	54.81	39.11
2	✗	✗	✓	✗	74.87	52.47	36.68
3	✗	✗	✓	✓	72.52	53.27	33.89
4	✓	✗	✗	✗	72.81	52.02	34.32
5	✓	✗	✓	✗	77.20	55.03	39.33
6	✓	✗	✓	✓	70.25	56.87	30.32
7	✓	✓	✗	✗	75.51	53.71	36.65
8	✓	✓	✓	✗	77.57	56.70	38.83
9 SnapNet-R	✓	✓	✓	✓	78.04	**58.13**	39.61
10** FusetNet SF5 (HD)	✗	✗	✗	✗	71.44	45.97	29.74
11** SnapNet-R (HD)	✓	✓	✓	✓	73.55	50.07	33.46

** Calculés à basse résolution (224x224) comme dans HAZIRBAS et al. [2016] au contraire des autres méthodes dont les résultats sont calculés à la résolution d'origine.*
***Nous avons également testé une stratégie Haute Résolution, utilisant des patchs de 224x244 pixels avec la résolution d'origine au lieu de redimensionner l'image.*

TABLEAU 6.2 – Résultats quantitatifs de segmentation sur SUNRGBD SONG et al. [2015]. Toutes les méthodes numérotées utilisent un FuseNet SF5 entraîné sur l'ensemble du jeu de données d'entraînement de SUNRGBD (à l'inverse de HAZIRBAS et al. [2016] qui se prive des images produites par le capteur *RealSense* d'Intel™pour des motifs de mauvaise qualité de la carte de profondeur). Nous avons reproduit les résultats du FuseNet SF5 dans l'expérience 1 que nous pouvons comparer à notre stratégie multi-vue dans l'expérience 9. Pour chaque métrique, la meilleure valeur est en gras et la deuxième meilleure valeur est soulignée.

l'algorithme des plus proches voisins, reprojeter toute l'information dans la scène *RGB-D* et finalement voter pour assigner une classe unique à chaque pixel.

Notre méthode complète correspond à l'expérience 9 dans le Tableau 6.2. Nous établissons un nouveau résultat de référence en précision moyenne sur la tâche de segmentation sémantique du jeux de données SUNRGBD. Nous atteignons 58,13% comparé aux 53.4% du Context-CRF LIN et al. [2017]. Nous obtenons également 78,04% de précision globale contre 78,4% pour le Context-CRF et 39,61% de mIoU (vs 42,3%), qui sont les seconds meilleurs résultats connus à ce jour sur ce jeu de données. Comparé au FuseNet SF5 seul, notre stratégie multi-vue permet un gain de performance sur toutes les métriques, particulièrement sur la précision moyenne, passant de 54,81% dans l'expérience 1 (52,61% dans l'article original HAZIRBAS et al. [2016]) à 58,13% dans l'expérience 9. Si nous venions à utiliser la méthode Context-CRF comme étape de segmentation dans la méthode SnapNet-R les résultats seraient probablement encore meilleurs pour chaque métrique.

Comme toute l'information passe nécessairement par une représentation 3D dans notre traitement nous avons dû pré-traiter l'information de profondeur pour ne pas perdre d'information *RGB*. Cela correspond aux opérations d'*inpainting* décrites dans la Section 6.3, paragraphe "à propos des données bruités". Pour déterminer l'influence de ces pré-traitements

nous présentons également les résultats de notre méthode avec et sans pré-traitements (voir Tableau 6.2). L'expérience 5 utilise le pré-traitement à l'entraînement et au test et mène déjà à une légère amélioration sur le FuseNet seul (expérience 1). Mais l'expérience 6 montre que la stratégie multi-vue utilisée uniquement lors du test réduit la mIoU (-9 points). Ceci implique que les images virtuelles générées par l'approche multi-vue sont légèrement différentes des images originales des scènes *RGB-D* ; il faut donc voir des images virtuelles à l'entraînement pour utiliser des images virtuelles au test.

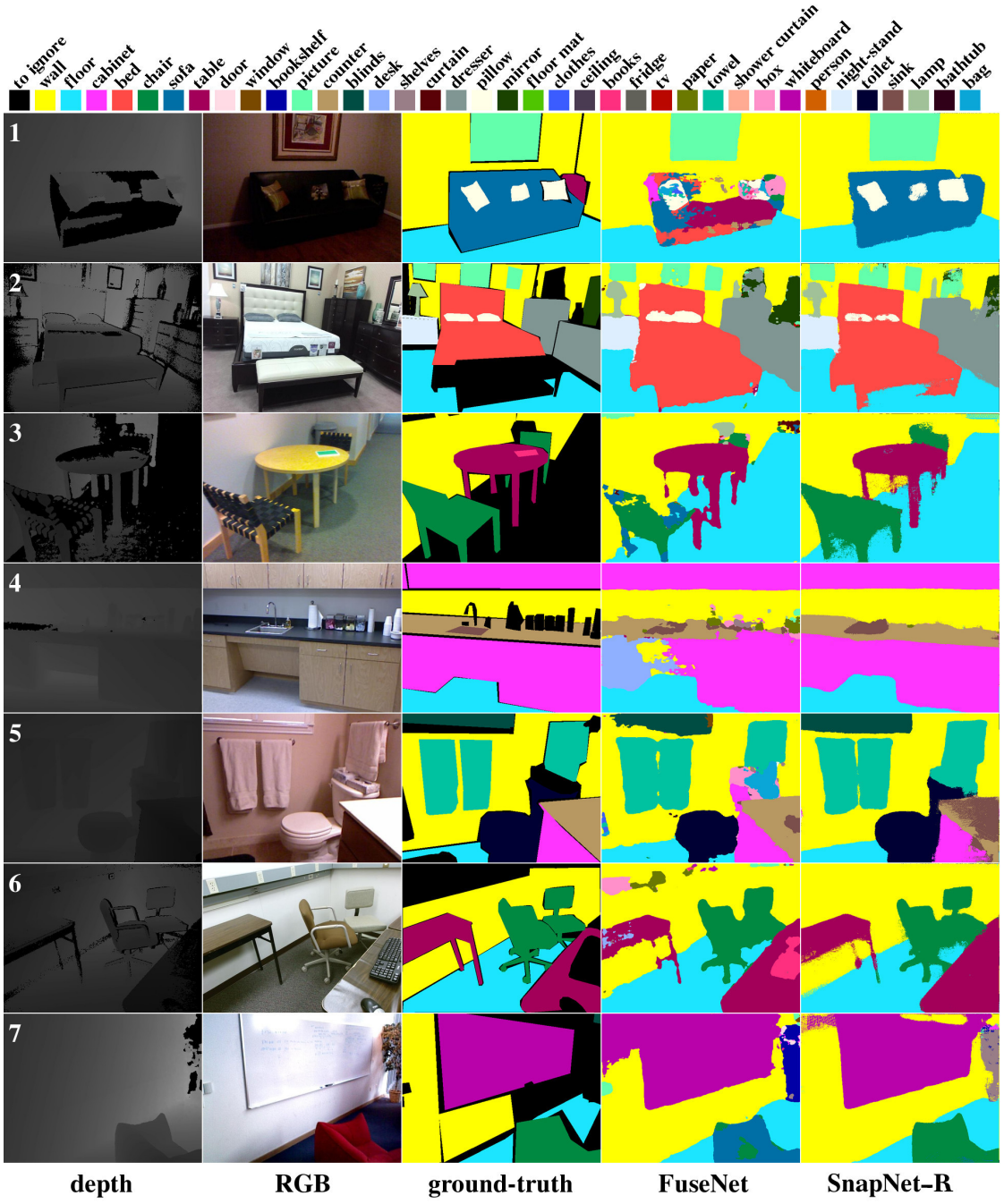


FIGURE 6.10 – Résultats qualitatifs de segmentation sur SUNRGBD SONG et al. [2015]. Les trois premières colonnes contiennent la carte de profondeur, l'image RGB et la vérité terrain. Les deux dernières colonnes présentent les résultats obtenus par le Fusenet SF5 HAZIRBAS et al. [2016] seul puis par SnapNet-R. De manière générale, les surfaces obtenues par SnapNet-R sont plus homogènes et les contours moins bruités que ceux de FuseNet seul.

Des résultats quantitatifs sont présentés dans la Figure 6.10. L'approche SnapNet-R est moins sensible aux occlusions (voir ligne 5), supposément grâce à l'aspect multi-vue. De plus, en regardant les chaises de la ligne 6 nous avons une segmentation plus nette des bords avec SnapNet-R qu'avec FuseNet seul. Les faibles résultats de la ligne 1 pour FuseNet peuvent s'expliquer par la qualité de la carte de profondeur. Dans cette situation spécifique SnapNet-R prend pleinement avantage de ses traitements d'*inpainting*. Il est important de noter que les données de SUNRGBD sont déjà traitées par une stratégie d'*inpainting* basique qui ne complète pas tous les artefacts. C'est ce qui est utilisé par FuseNet dans son implémentation originale de HAZIRBAS et al. [2016].

Nous avons également entraîné notre méthode sans faire de redimensionnement, directement à pleine résolution en utilisant des patchs de 224x224 pixels (expériences 10 et 11). Même si cette approche n'est pas capable d'atteindre les mêmes performances que l'expérience 9 nous observons les mêmes tendances par rapport à l'usage de la stratégie multi-vue.

NYUDv2 : segmentation sémantique 2D

Expérience	P	PM	mIoU
40 classes			
RCNN [GIRSHICK et al., 2014] (RGB-HHA)	60.3	35.1	28.6
FCN 16s [LONG et al., 2015] (RGB-HHA)	65.4	46.1	34.0
Eigen et al.[EIGEN et FERGUS, 2015](*RGB-D*-N)	65.6	45.1	34.1
Context-CRF [LIN et al., 2017] (*RGB-D*)	67.6	49.6	37.1
*FuseNet SF3[MA et al., 2017] (*RGB-D*)	66.4	44.2	34.0
*MVCNet-MP [MA et al., 2017](*RGB-D*)	**70.66**	51.78	**40.07**
FuseNet SF5 (*RGB-D*)	62.19	48.28	31.01
SnapNet-R (*RGB-D*)	69.20	**60.55**	38.33
13 classes			
Couprie et al.[COUPRIE et al., 2013] (*RGB-D*)	52.4	36.2	–
Hermans et al.[HERMANS et al., 2014] (*RGB-D*)	54.2	48.0	–
SceneNet (DHA)[HANDA et al., 2016] (DHA)	67.2	52.5	–
Eigen et al.[EIGEN et FERGUS, 2015] (*RGB-D*-N)	75.4	66.9	52.6
*FuseNet SF3 [MA et al., 2017] (*RGB-D*)	75.8	66.2	54.2
*MVCNet-MP [MA et al., 2017] (*RGB-D*)	79.13	70.59	59.07
FuseNet SF5 (*RGB-D*)	78.41	72.07	56.33
SnapNet-R (*RGB-D*)	**81.95**	**77.51**	**61.78**

* *Calculés à basse résolution (320x240) au contraire des autres méthodes dont les résultats sont calculés à la résolution d'origine.*

TABLEAU 6.3 – Résultats quantitatifs de SnapNet-R sur NYUDv2 SILBERMAN et al. [2012]. Pour chaque critère, la meilleure valeur est en gras, la deuxième meilleure valeur est soulignée.

Les résultats sur la base NYUDv2 dans le Tableau 6.3 montrent la même tendance que sur SUNRGBD. L'approche SnapNet-R réagit bien à une diminution du nombre de données à l'entraînement (7 fois moins). Sur le challenge de segmentation sémantique de 40 classes de NYUDv2 nous dépassons la méthode de référence de +8.77 points en précision moyenne et atteignons le deuxième meilleure score pour la précision globale et la mIoU. Concernant le challenge à 13 classes nous instaurons de nouveaux résultats de référence, atteignant les

81,95% en précision globale (+2.82), 77,51% en précision moyenne (+6.92) et 61,78% en mIoU (+2.71).

Par comparaison, les développements qui ont mené à la création de FuseNet HAZIRBAS et al. [2016] ont par la suite conduit au réseau cohérent multi-vue (*MVCNet*) MA et al. [2017] qui profite sur NYUDv2 des images acquises non étiquetées. Il déforme les prédictions sémantiques des images non étiquetées en les projetant dans une vue de référence adjacente commune pour laquelle les étiquettes sont connues. Cette association est effectuée sur la base d'une estimation de la trajectoire. Contrairement à *MVCNet*, notre méthode ne nécessite pas d'être entraînée sur des séquences vidéo pour extraire des images réalistes et ressemblantes, mais crée directement des vues virtuelles à partir des images *RGB-D* étiquetées. De plus, MVCNet utilise un entraînement de type "profondément supervisé" en appliquant une fonction de perte à chaque étape du décodeur de leur réseau.

6.4 Application multi-vue : 3DRMS Challenge

Après avoir appliqué l'approche SnapNet-R à des données *RGB-D* mono-vue nous nous intéressons à des scènes pour lesquelles nous possédons plusieurs vues réelles. De plus, dans le cadre du challenge *3DRMS* 2017, l'objectif n'est plus d'effectuer de la segmentation sémantique 2D comme sur SUNRGBD mais de produire un nuage de point 3D sémantisé.

Nous nous concentrons dans cette section sur le nuage de points reconstruit à partir d'une séquence d'images *RGB-D* ou de couples d'images stéréo, par exemple en utilisant une approche de type SLAM ENGEL et al. [2017]. Par rapport aux caméras mono-image *RGB-D*, nous exploitons la plus grande densité du nuage de points, ce qui nous permet de générer des points de vue très différents des positions des caméras d'origine. Nous retombons de fait sur le concept originel de SnapNet BOULCH et al. [2017] mais avec une nouvelle architecture de réseau et une procédure supplémentaire pour filtrer le nuage de points 3D avec sémantisation.

Le jeu de données du challenge 3DRMS 2017

Le **3DRMS Challenge** nous fournit un ensemble de séquences d'images capturées par une plate-forme robotique. Ce robot terrestre qui ressemble à une tondeuse est équipé de deux plates-formes stéréo complémentaires, produisant des images *RGB* et des images en niveaux de gris, ainsi qu'un capteur Leica Lidar pour l'acquisition d'un nuage de points 3D. L'ensemble de données contient 5 séquences d'images calibrées avec la position correspondante de la caméra : 4 pour l'entraînement (totalisant 108 vues) et 1 pour le test (avec 125 vues). Juste pour l'entraînement il y a également une vérité terrain sous la forme d'annotations sémantiques 2D pour les vues et une annotation sémantique 3D du nuage de points propre à la zone d'entraînement (voir Figure 6.11). Les différentes classes sont : *Unknown, Grass, Ground, Pavement, Hedge, Topiary, Rose, Obstacle, Tree, Background*.

Les différentes méthodes

Nous proposons trois méthodes de reconstruction et d'étiquetage différentes, chacune d'entre-elles correspondant à un cas d'utilisation robotique différent :

— **Classif 2D** : Segmentation sémantique des images réelles 2D puis projection en 3D (avec filtrage)

— **Classif 2D-3D** : Projection des images réelles 2D en 3D (avec filtrage) + SnapNet-R (3D→2D→3D)

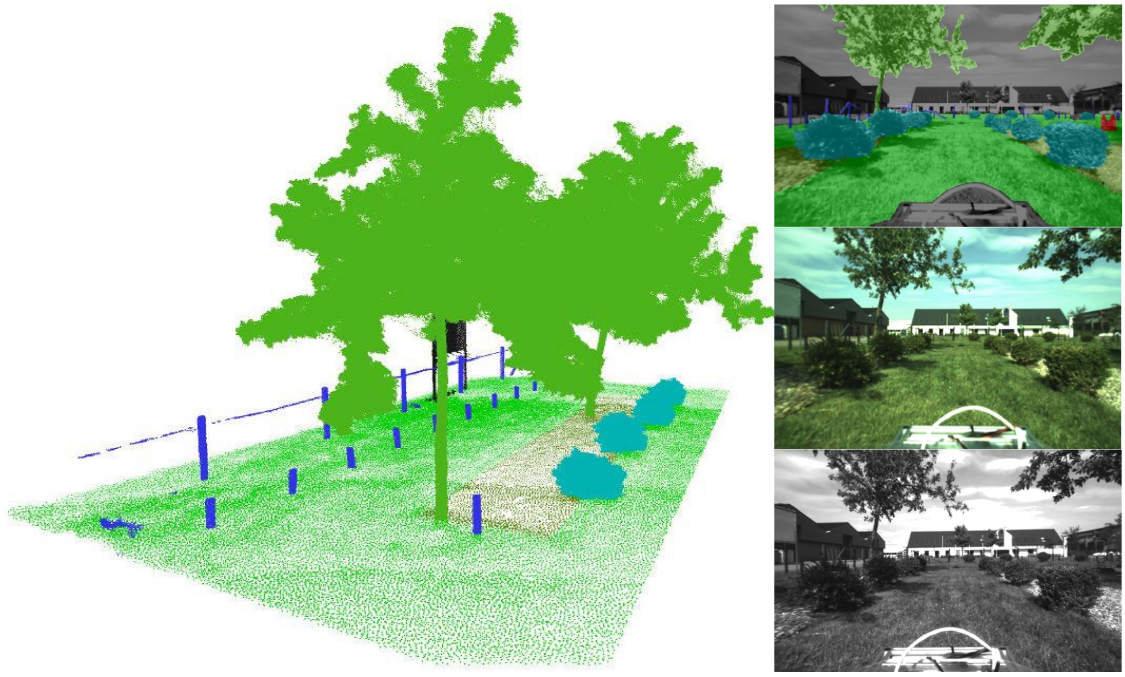


FIGURE 6.11 – Aperçu du jeu de données *3DRMS Challenge* : séquence d'entraînement "Boxhood Row" avec le nuage de point sémantisé et dans la colonne de droite : l'annotation sémantique sur l'image RGB, l'image "*front-left*" en *RGB* et l'image "*front-right*" en nuance de gris.

— **Classif 3D** : Construction d'un nuage de point global 3D avec l'ensemble des images + SnapNet-R (3D→2D→3D)

Ces méthodes sont illustrées dans la Figure 6.12.

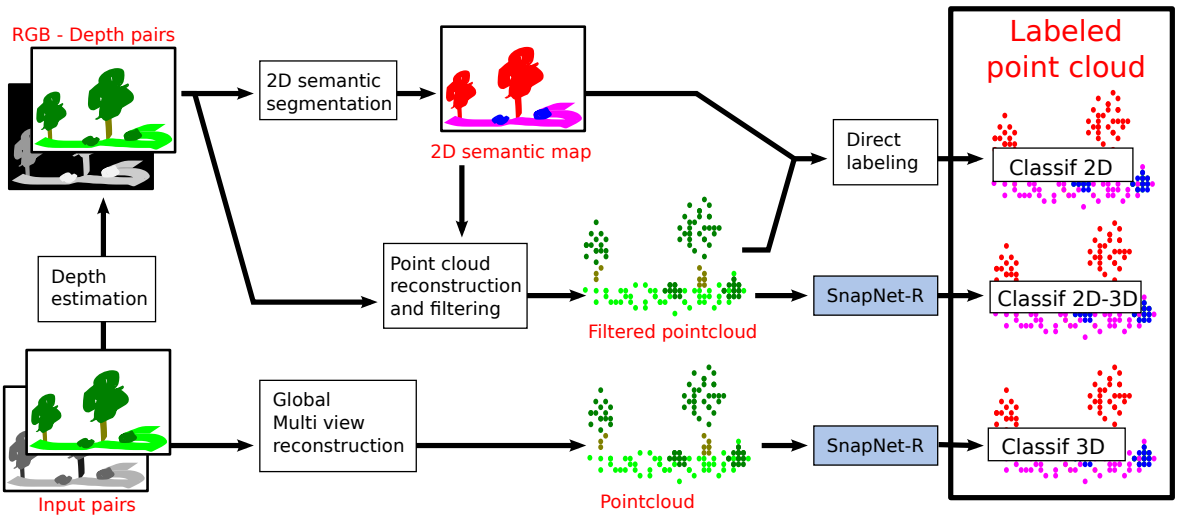


FIGURE 6.12 – Les différentes méthodes de reconstruction 3D appliquées pour le challenge *3DRMS* 2017.

6.4.1 "Classif 2D" : Reconstruction basée sur la navigation et étiquetage direct

La première méthode –**Classif 2D**– représente un type de reconstruction qui peut être utilisé dans la navigation robotique. Les étiquettes et les points sont estimés et accumulés pour chaque paire d'image stéréo acquise. Cette méthode se déroule en 4 étapes :

Estimation de profondeur Pour chaque paire de caméras, nous rectifions les images *RGB*/Grises et les images d'étiquettes de la vérité terrain. Ensuite, nous utilisons l'algorithme "*Efficient Large-scale Stereo Matching*" (*ELAS*) [GEIGER et al., 2010] afin de calculer la carte de disparité et d'ajouter un canal de profondeur aux images *RGB*, ce qui donne des données *RGB-D* brutes.

Segmentation sémantique 2D pour le fitrage Avec les données *RGB-D* de l'ensemble d'entraînement nous entraînons un Fusenet SF5 pour la segmentation sémantique 2D (avec les mêmes conditions que l'expérience 1 dans le Tableau 6.2). Malgré le faible nombre d'images d'entraînement, nous pouvons générer des cartes sémantiques grossières.

Reconstruction du nuage de points En tenant compte de la position globale donnée des caméras et des rectifications calculées, nous accumulons les points de toutes les images *RGB-D* dans un même système de coordonnées globales : une grille voxellique avec une taille de voxel de 0,01 unité de coordonnées. L'écartement entre les objectifs – ou *baseline* – des caméras d'acquisition est très petit (inférieur à 3cm). Ceci a pour conséquence de générer beaucoup de valeurs aberrantes dans l'accumulation des nuages de points (voir Figure 6.13(a)). Par conséquent, nous avons mis en place une stratégie de filtrage :

— D'abord, pour chaque image, nous filtrons les points trop proches ($< 1m$) ou trop loin ($> 3.5m$) de la caméra. Les points les plus proches sont principalement attribuables au véhicule embarqué utilisé pendant l'enregistrement et les plus loin sont trop bruités en raison de la petite *baseline*.

Ensuite, nous filtrons tous les points d'une même image selon deux filtres : basé sur les classes et basé sur la géométrie.

— Nous éliminons les points non étiquetés (classe "*Unknown*").

— Nous éliminons les points de l'arrière plan (classe "*background*").

— Nous éliminons les points qui sont dans une marge de 3 pixels entre deux objets de classes différentes.

— Nous calculons les normales en utilisant BOULCH et MARLET [2012] et rejetons les points qui sont trop alignés avec l'axe d'observation de la caméra (produit scalaire entre le vecteur normal et le vecteur de direction du centre de la caméra au point considéré $> 0,7$)

Une vue du résultat est présentée dans la Figure 6.13(b).

Étiquetage direct Comme dans SnapNet-R, l'étiquette à chaque point est calculée en votant à partir de l'accumulation de prédictions dans les différentes cartes sémantiques. Cette étape est comparable à la reprojection de SnapNet-R sauf que nous partons des images réelles et que les vraies positions des caméras sont disponibles.

6.4.2 "Classif 2D-3D" : Reconstruction basée sur la navigation et étiquetage 3D

La deuxième méthode –**Classif 2D-3D**– est une variante de la précédente. La reconstruction 3D est toujours effectuée en utilisant les positions de la caméra données par le jeu de données mais les cartes sémantiques 2D des images réelles sont utilisées seulement pour filtrer le nuage de points 3D comme pour la méthode "**Classif 2D**". Au moment de l'inférence, nous remplaçons "l'étiquetage direct" par SnapNet-R.

SnapNet-R : SnapNet-R permet de sémantiser le nuage de points 3D. Pour cela nous utilisons une autre stratégie de génération d'images virtuelles que celle appliquée sur SUNRGBD et NYUDv2. Nous positionnons la caméra virtuelle de sorte que :

— elle regarde toujours dans la direction de la scène, en particulier le centre de l'image correspond à un point 3D réel,

— elle soit toujours au dessus du niveau du sol,

— elle soit à une distance de 2 puis 5 mètres (deux images virtuelles pour un même point 3D → deux échelles).

L'étiquetage du nuage de points 3D est effectué différemment en fonction de l'étape d'entraînement et de test :

— à l'entraînement : le nuage de points est étiqueté avec la vérité terrain

— au test : le nuage de point est étiqueté avec la prédiction du FuseNet

Les classification sémantiques des images virtuelles sont alors transférées sur les points 3D ou elles sont fusionnées par la méthode de vote.

6.4.3 "Classif 3D" : Reconstruction hors-ligne et sémantisation 3D

La dernière approche –**Classif 3D**– est purement du post-traitement des données constitué de deux étapes :

— une reconstruction globale du nuage de points 3D

— une sémantisation du nuage de points 3D avec SnapNet-R

Reconstruction globale Nous utilisons le programme propriétaire "Agisoft Photoscan". En utilisant directement toutes les images pour la reconstruction il lui est possible de trouver des correspondances de points entre les différentes images, donc avec une plus grande "*baseline*" que l'approche stéréo. Cela donne un nuage de points avec moins d'artefacts (voir Figure 6.13(c)).

Nous utilisons le modèle SnapNet-R entraîné avec la méthode précédente pour effectuer la sémantisation.

6.4.4 Résultats qualitatifs

Au moment de la rédaction de ces lignes la vérité terrain de l'ensemble de test n'est pas encore disponible car elle est réservée à l'évaluation du challenge. De plus, en raison du faible nombre de données d'entraînements nous n'avons pas effectué de validation croisée en partionnant l'ensemble d'entraînement en plusieurs jeux "entraînement/validation" qui auraient trop bénéficié d'un sur-apprentissage. Nous avons donc effectué une analyse qualitative des résultats.

Reconstruction

La Figure 6.13 compare les nuages de points générés par nos trois méthodes (**Classif 2D**, **Classif 2D-3D** et **Classif 3D**).

Nous avons dû filtrer les données brutes avant d'utiliser SnapNet-R : les objets de petites tailles n'auraient pas été visibles autrement. De plus, la "*baseline*" étant petite, elle engendre des erreurs d'estimation de la carte de profondeur qui ne sont pas corrigées par l'accumulation. Cela produit plusieurs occurrences de petits objets comme les troncs d'arbustes.

Avec les données disponibles, l'approche globale a su produire un nuage de points plus lisse, avec moins d'artefacts que les méthodes "stéréo". Ceci est également observable sur la Figure 6.14 grâce à l'estimation des normales visibles avec les ombres qui traduisent les orientations locales.

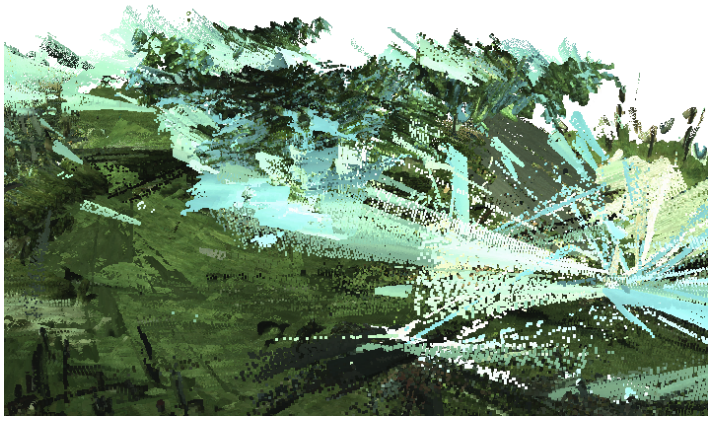
Sémantisation

Les résultats des Figures 6.14 et 6.15 représentent les classes prédites avec des couleurs arbitraires.

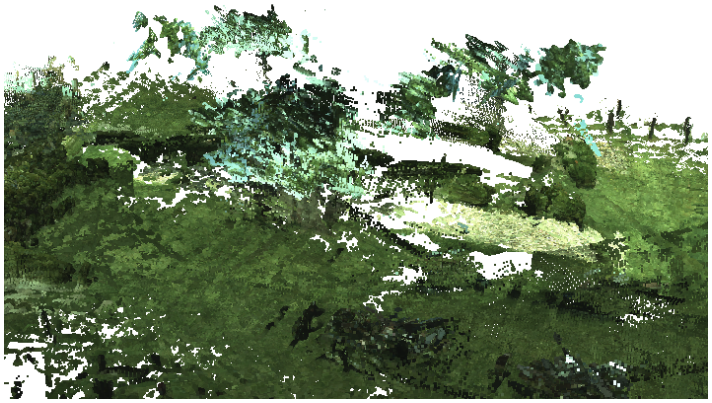
Comparée à la méthode **Classif 2D** (à gauche), SnapNet-R est plus précise dans son étiquetage (**Classif 2D-3D** au milieu). Ceci s'explique par les caméras virtuelles qui fournissent des vues inaccessibles au robot terrestre. Par exemple, des vues regardant le sol avec un peu plus d'angle d'incidence permettent d'appréhender la géométrie globale du dallage (en vert foncé) et des zones du terrain.

La méthode **Classif 3D** profite de la qualité de la reconstruction sur laquelle SnapNet-R s'applique aussi bien qu'en **Classif 2D-3D**. Il faudra attendre les résultats du Challenge SHREC 2017 pour avoir un aperçu quantitatif de la performance.

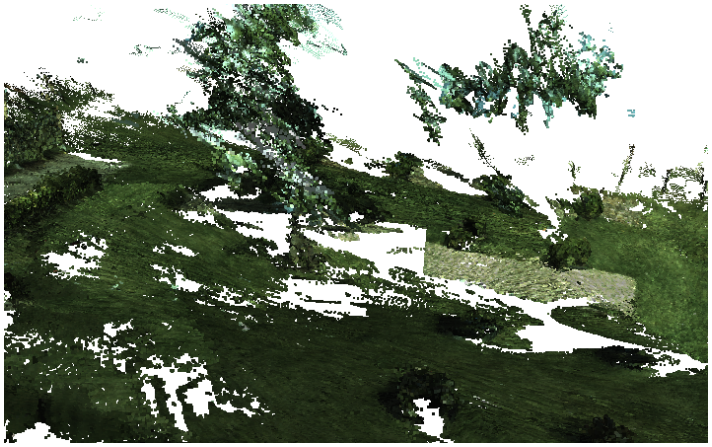
Cette expérience montre également la robustesse de notre approche face aux nuages de points 3D. Nous avons entraîné SnapNet-R sur les nuages de points issus des acquisitions 2.5D (et non pas le nuage de points issu du laser) : la régularisation induite par le réseau de neurones et le lissage de la prédiction avec l'accumulation réduit les étiquettes imparfaites due aux artefacts de reconstruction.



(a) Accumulation incrémentale sans filtrage



(b) Accumulation incrémentale avec filtrage



(c) Reconstruction globale

FIGURE 6.13 – Nuage de point du challenge *3DRMS* 2017 (ensemble de test)

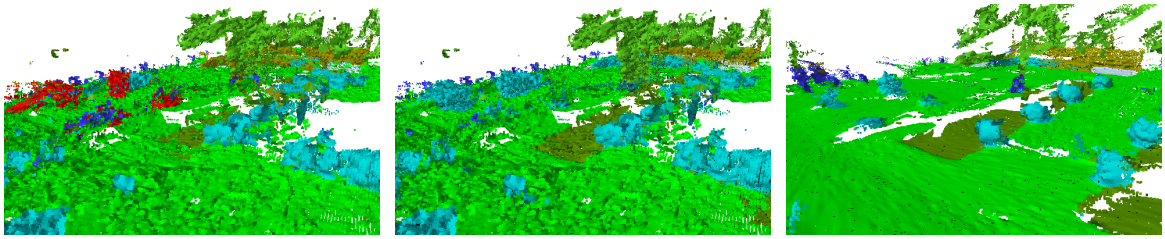


FIGURE 6.14 – Résultats de la reconstruction de l'ensemble de test du challenge *3DRMS* 2017. À gauche : sémantisation obtenue par la méthode **Classif 2D**, au milieu : sémantisation obtenue par la méthode **Classif 2D-3D**, à droite : sémantisation obtenue par la méthode **Classif 3D**.

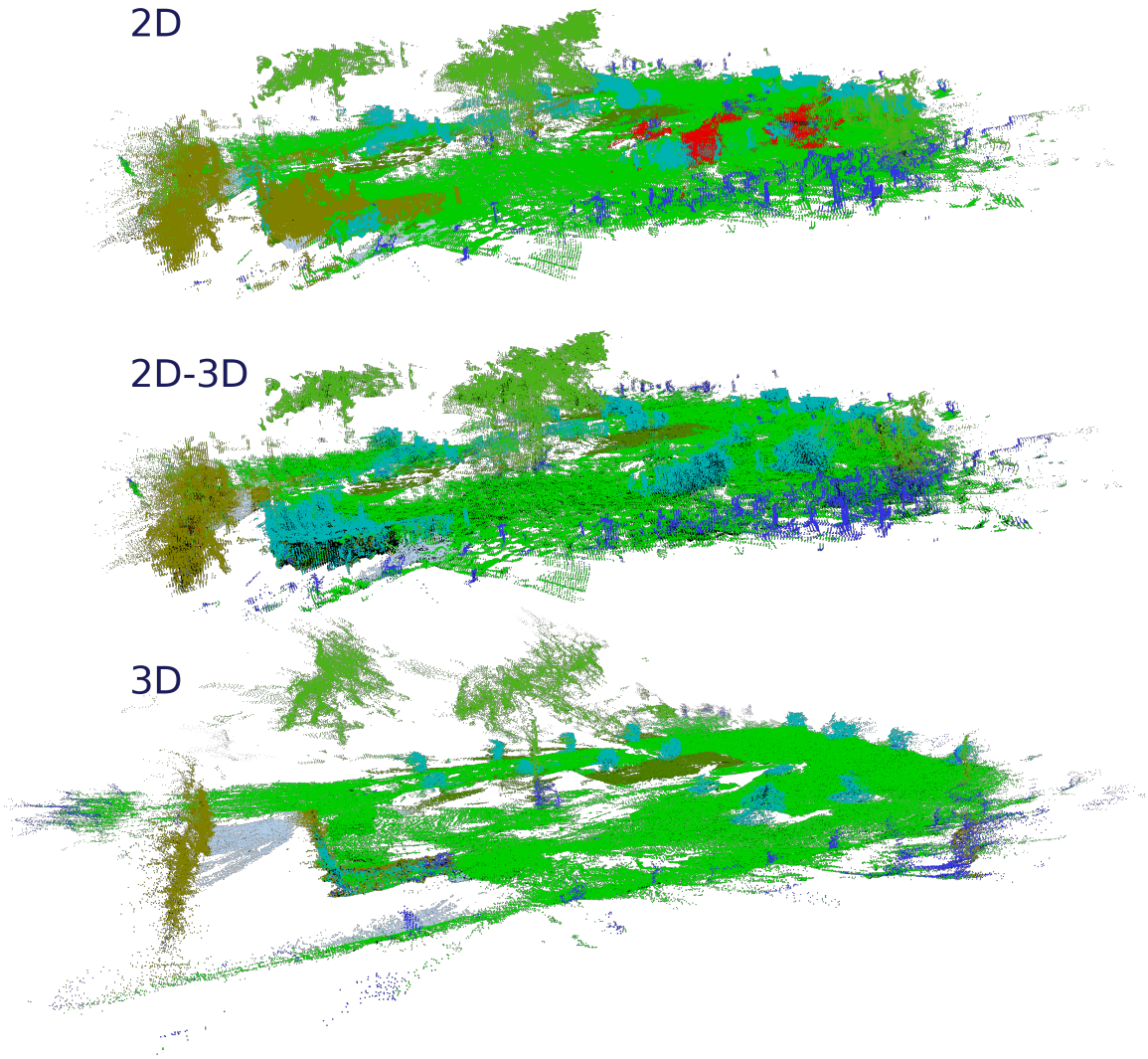


FIGURE 6.15 – Résultats de la reconstruction de l'ensemble de test du challenge 3DRMS 2017, 2ème point de vue. En haut : sémantisation obtenue par la méthode **Classif 2D**, au milieu : sémantisation obtenue par la méthode **Classif 2D-3D**, en bas : sémantisation obtenue par la méthode **Classif 3D**.

6.5 Conclusion du chapitre

Les travaux que nous venons de présenter confirment l'intérêt d'utiliser une stratégie multi-vue pour sémantiser des scènes 3D. Ce gain en performance provient tout d'abord du concept original de l'approche multi-vue. Par la multiplication des observations à l'entraînement nous sommes capables d'enrichir la capacité de sémantisation du réseau de segmentation. Par l'accumulation des prédictions au moment du test nous sommes capables de lisser les prédictions et d'observer certaines zones sous différents angles complémentaires.

Nous avons présenté ce concept au sein d'une architecture complète capable de s'adapter à des données produites depuis une seule acquisition *RGB-D* ou depuis une approche exploratoire (comme le *SLAM*). Elle intègre notamment une correction du *Class Imbalance Problem* qui est une étape essentielle pour utiliser la technologie des réseaux de neurones. De plus, l'évolution de SnapNet à SnapNet-R confirme l'intérêt du FuseNet de HAZIRBAS et al. [2016] pour les données multi-modales. Il faut cependant garder en tête les résultats du Chapitre 5 vis-à-vis des méthodes hybrides qui peuvent mal réagir en cas de perte d'une modalité. Il serait donc intéressant de poursuivre cette étude en intégrant nos approches RCNN RGB U ou X avec la méthode SnapNet-R.

En terme de performances, nous établissons de nouveaux résultats de référence sur les jeux de données SUNRGBD & NYUDv2 et présentons des résultats prometteurs pour le challenge 3DRMS 2017.

Dissémination

• Ces travaux ont fait l'objet d'une publication dans le journal *Computers and Graphics* 2017.

SnapNet : 3D point cloud semantic labeling with 2D deep segmentation networks
Alexandre Boulch, Bertrand Le Saux, Nicolas Audebert, Joris Guerry

• Ces travaux ont fait l'objet d'une publication au *workshop 3D Reconstruction meets Semantics* (*3DRMS*) à l'*International Conference on Computer Vision* (*ICCV*) 2017 :

SnapNet-R : Consistent 3D Multi-View Semantic Labeling for Robotics
Joris Guerry, Alexandre Boulch, Bertrand Le Saux, Julien Moras, Aurélien Plyer, David Filliat

6.6 Références

AUDEBERT, N., B. LE SAUX et S. LEFÈVRE. 2016, «Semantic segmentation of earth observation data using multi-modal and multi-scale deep networks», dans Asian Conference on Computer Vision, Springer, p. 180–196. 118

BADRINARAYANAN, V., A. KENDALL et R. CIPOLLA. 2015, «SegNet : A Deep Convolutional Encoder-Decoder Architecture for Image Segmentation», arXiv preprint arXiv :1511.00561. URL http://arxiv.org/abs/1511.00561. 118

BERTALMIO, M., A. L. BERTOZZI et G. SAPIRO. 2001, «Navier-stokes, fluid dynamics, and image and video inpainting», dans Computer Vision and Pattern Recognition, 2001. CVPR

2001. Proceedings of the 2001 IEEE Computer Society Conference on, vol. 1, IEEE, p. I–I. 120

BOULCH, A., M. DE LA GORCE et R. MARLET. 2014, «Piecewise-planar 3D reconstruction with edge and corner regularization», dans Computer Graphics Forum, vol. 33, p. 55–64. 118

BOULCH, A., B. LE SAUX et N. AUDEBERT. 2017, «Unstructured point cloud semantic labeling using deep segmentation networks», Eurographics/3DOR. 117, 121, 128

BOULCH, A. et R. MARLET. 2012, «Fast and robust normal estimation for point clouds with sharp features», dans Computer graphics forum, vol. 31, Wiley Online Library, p. 1765–1774. 130

COUPRIE, C., C. FARABET, L. NAJMAN et Y. LECUN. 2013, «Indoor semantic segmentation using depth information», arXiv preprint arXiv :1301.3572. 127

EIGEN, D. et R. FERGUS. 2015, «Predicting depth, surface normals and semantic labels with a common multi-scale convolutional architecture», dans Proceedings of the IEEE International Conference on Computer Vision, p. 2650–2658. 124, 127

ENGEL, J., V. KOLTUN et D. CREMERS. 2017, «Direct sparse odometry», IEEE transactions on pattern analysis and machine intelligence. 128

ENGEL, J., T. SCHÖPS et D. CREMERS. 2014, «Lsd-slam : Large-scale direct monocular slam», dans European Conference on Computer Vision, Springer, p. 834–849. 116

GEIGER, A., M. ROSER et R. URTASUN. 2010, «Efficient large-scale stereo matching», dans Asian conference on computer vision, Springer, p. 25–38. 130

GIRSHICK, R., J. DONAHUE, T. DARRELL et J. MALIK. 2014, «Rich feature hierarchies for accurate object detection and semantic segmentation», dans Proceedings of the IEEE conference on computer vision and pattern recognition, p. 580–587. 127

HACKEL, T., N. SAVINOV, L. LADICKY, J. D. WEGNER, K. SCHINDLER et M. POLLEFEYS. 2017, «Semantic3d.net : A new large-scale point cloud classification benchmark», ISPRS Annals of Photogrammetry, Remote Sensing and Spatial Information Science, vol. IV-1/W1. 117

HANDA, A., V. PATRAUCEAN, V. BADRINARAYANAN, S. STENT et R. CIPOLLA. 2016, «Understanding real world indoor scenes with synthetic data», dans Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, p. 4077–4085. 121, 122, 127

HAZIRBAS, C., L. MA, C. DOMOKOS et D. CREMERS. 2016, «Fusenet : Incorporating depth into semantic segmentation via fusion-based cnn architecture», dans Asian Conference on Computer Vision, Springer, p. 213–228. 118, 124, 125, 126, 127, 128, 135

HERMANS, A., G. FLOROS et B. LEIBE. 2014, «Dense 3d semantic mapping of indoor scenes from rgb-d images», dans Robotics and Automation (ICRA), 2014 IEEE International Conference on, IEEE, p. 2631–2638. 127

JIANG, J., Z. ZHANG, Y. HUANG et L. ZHENG. 2017, «Incorporating depth into both cnn and crf for indoor semantic segmentation», arXiv preprint arXiv :1705.07383. 125

KENDALL, A., V. BADRINARAYANAN et R. CIPOLLA. 2015, «Bayesian segnet : Model uncertainty in deep convolutional encoder-decoder architectures for scene understanding», arXiv preprint arXiv :1511.02680. 125

KRÄHENBÜHL, P. et V. KOLTUN. 2011, «Efficient inference in fully connected crfs with gaussian edge potentials», dans Advances in neural information processing systems, p. 109–117. 124

LI, Z., Y. GAN, X. LIANG, Y. YU, H. CHENG et L. LIN. 2016, «Lstm-cf : Unifying context modeling and fusion with lstms for rgb-d scene labeling», dans European Conference on Computer Vision, Springer, p. 541–557. 125

LIN, G., C. SHEN, A. VAN DEN HENGEL et I. REID. 2017, «Exploring context with deep structured models for semantic segmentation», IEEE Transactions on Pattern Analysis and Machine Intelligence. 124, 125, 127

LONG, J., E. SHELHAMER et T. DARRELL. 2015, «Fully convolutional networks for semantic segmentation», dans Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, p. 3431–3440. 125, 127

MA, L., J. STUECKLER, C. KERL et D. CREMERS. 2017, «Multi-view deep learning for consistent semantic mapping with rgb-d cameras», dans arXiv :1703.08866, p. –. 127, 128

SILBERMAN, N., D. HOIEM, P. KOHLI et R. FERGUS. 2012, «Indoor segmentation and support inference from rgbd images», Computer Vision–ECCV 2012, p. 746–760. 127

SONG, S., S. P. LICHTENBERG et J. XIAO. 2015, «SUN RGB-D : A RGB-D scene understanding benchmark suite», dans CVPR, Boston, USA, p. 567–576. 124, 125, 126

Troisième partie

Conclusions et perspectives

Chapitre 7

Conclusion et Perspectives

> *C'est le commencement qui est le pire, puis le milieu puis la fin ; à la fin, c'est la fin qui est le pire.*

Samuel Beckett, L'Innommable, 1953

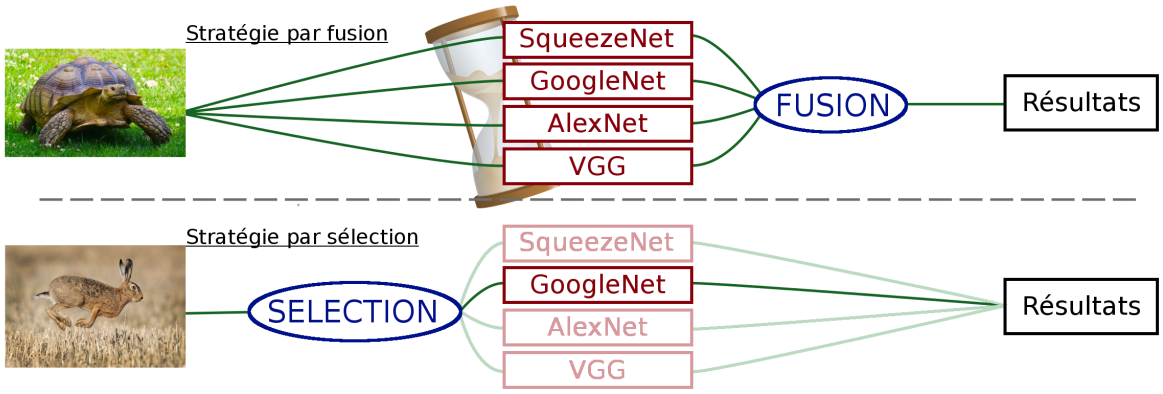
Sommaire

7.1 Conclusion **142**
7.2 Perspectives **146**
7.3 Références **147**

7.1 Conclusion

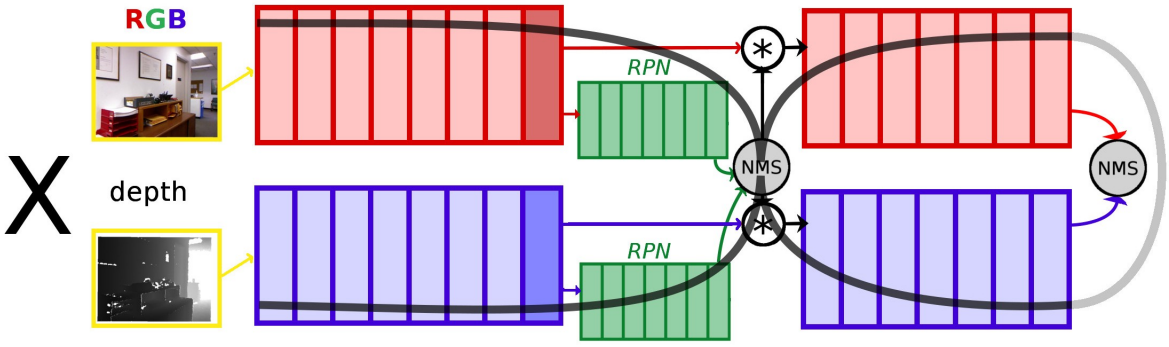
L'objectif principal de ce travail de thèse était de trouver comment améliorer les performances de reconnaissance visuelle d'un robot mobile. Nous avons fait ressortir différentes aspects de cet enjeu dans la Section 1.2 : la perception visuelle[i], la reconnaissance d'objets[ii], l'adaptation[iii] et le contexte robotique[iv]. Ces considérations ont été abordées de façon transverse dans les travaux présentés dans cette thèse.

Dans le Chapitre "Sélection d'algorithmes de classification" nous avons étudié le principe de sélection de méthodes pour la classification. Pour cela nous avons utilisé diverses architectures de réseaux de neurones pour déterminer le meilleur classifieur à utiliser dans une liste de méthodes disponibles sans connaître leurs prédictions respectives. Cette stratégie s'inscrit à la fois dans une logique d'adaptation contextuelle[iii] et d'économie des ressources[iv] en opposition aux algorithmes de fusion qui infèrent en se basant sur les prédictions de l'ensemble des classifieurs. Nous avons montré le potentiel certain de cette approche par une étude de la complémentarité des classifieurs[i] disponibles. Cependant, les performances de cette stratégie dépendent de la complexité à partitionner les données en des domaines propres à chaque classifieur. Dans un premier cas, avec des classifieur entraînés sur des images provenant de jeux de données variés, le sélecteur parvient correctement à attribuer des classifieurs aux images. Dans un second cas, en travaillant sur un jeu de données unique, le sélecteur que nous avons proposé ne parvient pas à séparer les données vis-à-vis des méthodes. Cette deuxième situation met en avant la complexité d'une telle tâche pour des données très similaires. C'est pourquoi nous avons orienté la suite de nos travaux vers des données naturellement séparables : les images *RGB-D*.



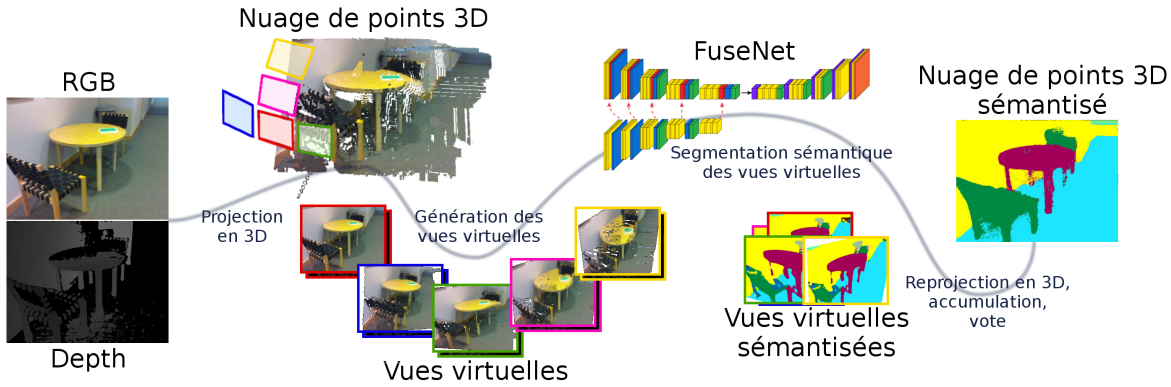
Afin d'exploiter au mieux les données *RGB-D* nous avons, dans le Chapitre "Approche multi-modale pour la détection", étudié les différentes façons d'encoder la carte de profondeur. Utiles dans certaines situations, notamment pour exploiter l'information d'orientation des surfaces, les meilleurs encodages sont coûteux en temps de calcul. Notre cadre applicatif, la détection de personnes, ne bénéficie pas particulièrement de cette étape de pré-traitement, que nous avons donc ignorée par la suite. D'autre part, nous avons mis en avant la chute de performance[iv] des détecteurs *RGB* lorsque la luminosité varie fortement (environnement sombre ou contre-jour), et des détecteurs *Depth* lorsque la lumière du soleil vient bruiter le champ infrarouge du capteur. Heureusement, ce champ infrarouge actif permet à la caméra *RGB-D* de fonctionner dans le noir. Les défauts du capteur sont ainsi incompatibles et par conséquent les modalités sont complémentaires[i]&[iii]. Nous avons donc proposé plusieurs architectures pour profiter de cet aspect combinatoire des informations couleur et carte de profondeur. Notre meilleure méthode, la fusion en X, utilise deux *Faster-RCNNs* experts en parallèle et profite des détections internes intermédiaires pour échanger de l'information entre les deux pipelines. Cette interactivité permet aux experts d'utiliser leurs espaces de

représentation propres pour classifier les détections de leur pair, permettant ainsi une double classification[ii] et une augmentation de la performance de détection. Nous proposons également l'architecture hybride Y qui pâtit fortement de la perte d'une modalité. Nos méthodes instaurent de nouvelles références dans l'état de l'art de la détection de personnes sur des données *RGB-D*.



Le Chapitre "Approche multi-modale pour la détection" fut également l'occasion d'introduire notre nouveau jeu de données *RGB-D* "ONERA.ROOM". Disponible en ligne, il est composé de plus de 35000 images de personnes annotées et contient des situations d'acquisitions complexes[iii]&[iv] pour permettre à l'ensemble de la communauté de tester la robustesse de leurs méthodes.

Enfin, dans le Chapitre "Approche multi-vue pour la segmentation" nous avons fait usage de l'information 3D contenue dans les images *RGB-D* au travers d'une méthode multi-vue. En considérant l'aspect mono-source des images 2.5D, nous avons défini une stratégie de génération de vues virtuelles cohérentes pour appliquer SnapNet-R, une variante pour la robotique d'une stratégie multi-vue éponyme. Nous utilisons les vues virtuelles pour entraîner et tester un réseau de neurones de segmentation sémantique. Cette approche permet d'augmenter les données d'entraînement et d'accumuler différentes prédictions lors du test. Nous obtenons de nouveaux résultats de référence sur les tâches de segmentation sémantique des jeux de données *SUNRGBD* & *NYUDv2*.



Ces travaux de thèses nous ont permi d'aborder de façon originale des données robotiques 2D, 2.5D et 3D avec des réseaux de neurones. Que ce soit pour la classification, la détection et la segmentation sémantique, nous avons non seulement validé nos approches sur des jeux de données difficiles, mais également amené l'état de l'art à un nouveau niveau de performance. Ces travaux ont notamment permis de faire ressortir deux problématiques inhérentes à l'apprentissage automatique.

La première concerne les distributions inégales d'images dans les jeux de données. Par exemple dans le Chapitre "Sélection d'algorithmes de classification", le réseau de confiance

qui doit choisir entre différentes méthodes entraînées sur différents jeux de données est influencé par la taille de CIFAR10 qui représente plus de 90% des images. Dans SUNRGBD et NYUDv2, utilisés dans les Chapitres "Approche multi-modale pour la détection" et "Approche multi-vue pour la segmentation", les pixels de la classe "wall", "floor" et "ceilings" représentent également plus de 80% des jeux de données. Il est très important d'avoir conscience de ce problème classique pour ne pas biaiser les méthodes d'apprentissage. En particulier, les réseaux de neurones étant entraînés par apprentissage stochastique, sont très sensibles à ce phénomène. Nous avons tenté différentes pondérations dans les fonctions de perte habituelles

Par ailleurs, les Chapitres "Approche multi-modale pour la détection" et "Approche multi-vue pour la segmentation" ont illustré les avantages et inconvénients des méthodes hybrides ou indépendantes. Appliquées sur des données *RGB-D*, les méthodes indépendantes (comme les architectures U et X) permettent de supporter la perte d'une modalité tout en offrant un caractère *plug and play* aux experts (autrement dit, nous pouvons changer facilement un des experts, sans toucher l'autre). Cette situation offre l'avantage et l'inconvénient de devoir entraîner chaque expert indépendamment : cela minimise les ressources nécessaires pour chaque entraînement mais nécessite plusieurs entraînements. Les méthodes hybrides (comme FuseNet et l'architecture Y) offrent, en situation normale, des performances équivalentes voire plus précises par rapport aux méthodes indépendantes. Elles sont construites pour utiliser au mieux la complémentarité des sources d'information. Cependant, comme nous pouvons l'observer dans HAZIRBAS et al. [2016], les tenseurs internes d'activations des différentes branches modales témoignent d'une répartition des tâches entre les modalités. Ceci implique une nécessité d'avoir toujours les deux modalités à disposition, ce qui n'est pas le cas en situation difficile comme montré dans le Chapitre "Approche multi-modale pour la détection". L'apprentissage quant à lui est unique donc théoriquement plus simple mais ne permet pas d'ajouter/retirer/remplacer une modalité sans ré-entraîner l'ensemble de la structure.

Dissémination

• Ces travaux ont fait l'objet d'une présentation dans le *workshop* "Apprentissage profond pour la perception et la robotique" au congrès sur la Reconnaissance des Formes et l'Intelligence Artificielle (RFIA) 2016.

Sélection d'algorithmes de classification par réseau de neurones
Joris Guerry, Bertrand Le Saux, David Filliat

• Ces travaux ont fait l'objet d'une publication à l'*European Conference on Mobile Robotics* (*ECMR*) 2017 :

"Look At This One" Detection sharing between modality-independent classifiers for robotic discovery of people
Joris Guerry, Bertrand Le Saux, David Filliat

• Ces travaux ont fait l'objet d'une publication au colloque 2017 du Groupe d'Etudes du Traitement du Signal et des Images (GRETSI).

RCNN RGBD pour la détection de personnes en conditions difficiles
Joris Guerry, Bertrand Le Saux, David Filliat

• Ces travaux ont fait l'objet d'une publication commune au workshop *3D Shape Retrieval Contest* (SHREC) de *Eurographics* 2017.

SHREC'17 Track : 3D Hand Gesture Recognition Using a Depth and Skeletal Dataset
Q. De Smedt, H. Wannous, J.-P. Vandeborre, J. Guerry, B. Le Saux, D. Filliat

• Ces travaux ont fait l'objet d'une publication dans le journal *Computers and Graphics* 2017.

SnapNet : 3D point cloud semantic labeling with 2D deep segmentation networks
Alexandre Boulch, Bertrand Le Saux, Nicolas Audebert, Joris Guerry

• Ces travaux ont fait l'objet d'une publication au *workshop 3D Reconstruction meets Semantics* (*3DRMS*) à l'*International Conference on Computer Vision* (*ICCV*) 2017 :

SnapNet-R : Consistent 3D Multi-View Semantic Labeling for Robotics
Joris Guerry, Alexandre Boulch, Bertrand Le Saux, Julien Moras, Aurélien Plyer, David Filliat

7.2 Perspectives

Pour terminer, nous abordons les améliorations envisageables et les perspectives intéressantes offertes par l'état final de notre étude.

U, X, Y, ... Concernant le Chapitre "Sélection d'algorithmes de classification", nous avons envisagé d'autres architectures pour la détection basée sur le *Faster-RCNN* [REN et al., 2015] :

— En U ou en X avec un seul Region Proposal Network *RGB-D*,

— En U, X ou Y avec de l'hybridation entre les deux branches initiales (comme dans FuseNet [HAZIRBAS et al., 2016]).

Cette extension serait sujette à la problématique des méthodes hybrides *versus* indépendantes. Cependant, seul le RPN *RGB-D* nécessiterait un entrainement spécifique pour faire face à la perte de modalité. Au même titre que le *Gating Network* de MEES et al. [2016], le RPN aurait un rôle contextuel et permettrait de vraiment retomber sur le formalisme introduit au début de ce chapitre : le *R&M_CNN* (*Region & Method CNN*).

SnapNet-R : "*Scene aware*" MA et al. [2017] proposent de projeter l'information d'images adjacentes des séquences vidéos de NYUDv2 dans un référentiel commun pour accumuler l'information voisine. Snapnet-R ne fait pas usage des images proches de la séquence car nous n'utilisons que des images virtuelles. Cependant, l'architecture de SnapNet-R est agnostique au fait que les images peuvent venir d'une même scène, ce qui nous permet d'ailleurs de mélanger les images virtuelles de scènes différentes dans les *batchs* d'entraînement.Nous pourrions donc utiliser une variante de l'encodage par accumulation tensorielle, présenté pour le challenge 3DRMS SHREC (voir Section 5.2.1), pour classifier un nuage de points avec plusieurs points de vue considérés comme un tout. Il faudrait pour cela définir une stratégie systématique d'observation (comme nous le faisons déjà et donc non-aléatoire) pour une région à classifier afin que le réseau de sémantisation puisse déterminer les relations géométriques entres les différentes vues. Une autre approche pour permettre à SnapNet-R de comprendre une scène dans son ensemble serait de produire un SnapLSTM-R, qui utiliserait un *Long Short-Term memory network* [LI et al., 2016] afin d'intégrer les informations de plusieurs acquisitions successives. Cette solution demanderait plus de travail.

SnapNet-R en conditions difficiles Les travaux de REN et al. sur le *Faster-RCNN* ont également mené à la création du *Mask-RCNN* [HE et al., 2017] qui complète la liste des modules de fin de réseaux :

— régression de la position et de la forme des patchs

— classification sémantique des patchs

— NOUVEAU : segmentation de l'objet dans le patch

Nous pouvons donc utiliser une fusion en X entre deux *Mask-RCNNs* mais cette fois-ci dans une tâche de segmentation et appliquer cette architecture au sein de la méthode SnapNet-R en lieu et place de FuseNet. Ceci permettrait d'avoir un SnapNet-R non hybride et donc plus robuste aux conditions *RGB-D* difficiles.

Formalisme Bayésien Le regain d'intérêt des réseaux de neurones artificiels a également été suivi par de nombreuses critiques sur l'aspect "boîte noire" de leur fonctionnement interne. Les travaux de ZEILER et FERGUS intitulés "*Visualizing and understanding convolutional networks*" ont permis de mieux appréhender le fonctionnement interne des *CNNs*, cependant,

comme le montre NGUYEN et al. [2015], les *CNNs* et leurs non-linéarités ont un processus d'inférence très sensible. Cette situation empêche l'usage des réseaux de neurones dans un cadre industriel strict avec maîtrise parfaite des comportements algorithmiques. Pour franchir cette limitation il serait intéressant de pouvoir caractériser statistiquement la qualité des prédictions. Cet aspect est déjà abordé dans l'article "*Bayesian SegNet : Model Uncertainty in Deep Convolutional Encoder-Decoder Architectures for Scene Understanding*" KENDALL et al. [2015]. Nous pensons que c'est une perspective de travail nécessaire pour permettre l'intégration de l'apprentissage profond dans des applications plus critiques. En particulier dans notre problématique, il serait intéressant d'évaluer la valeur des prédictions lorsque les conditions sont vraiment dégradées et ainsi améliorer la prise de décision. Que vaut la prédiction d'un *RCNN RGBD X* lorsqu'il regarde un objet en contre-jour dans une veranda ? La couleur est réduite à des contrastes très forts et la carte de profondeur est bruitée par la lumière du soleil...

Robotique mobile et limitation matérielle Comme indiqué dans l'introduction de nos travaux, nous ne nous sommes pas intéressés à la problématique d'embarquabilité de nos méthodes. Il existe tout de même certaines approches très intéressantes pour permettre l'usage de réseaux de neurones sur des modules de calcul basique comme des processeurs *ARM* (le *CPU* des *smartphones*, tablettes ou petits robots.). Nous pensons tout particulièrement aux réseaux de neurones binaires : le XNOR-Net RASTEGARI et al. [2016]. Cette approche propose d'utiliser des convolutions dont les paramètres sont des nombres binaires. Ceci résulte en un gain d'espace d'un facteur 32 par rapport aux nombres à virgules flottantes utilisés classiquement. De plus, cela permet d'accélérer le temps de calcul car les opérations sont plus simples (opérations booléennes) et permet d'obtenir des inférences temps réels sur *CPU*.

7.3 Références

HAZIRBAS, C., L. MA, C. DOMOKOS et D. CREMERS. 2016, «Fusenet : Incorporating depth into semantic segmentation via fusion-based cnn architecture», dans Asian Conference on Computer Vision, Springer, p. 213–228. 144, 146

HE, K., G. GKIOXARI, P. DOLLÁR et R. GIRSHICK. 2017, «Mask r-cnn», arXiv preprint arXiv :1703.06870. 146

KENDALL, A., V. BADRINARAYANAN et R. CIPOLLA. 2015, «Bayesian segnet : Model uncertainty in deep convolutional encoder-decoder architectures for scene understanding», arXiv preprint arXiv :1511.02680. 147

LI, Z., Y. GAN, X. LIANG, Y. YU, H. CHENG et L. LIN. 2016, «Lstm-cf : Unifying context modeling and fusion with lstms for rgb-d scene labeling», dans European Conference on Computer Vision, Springer, p. 541–557. 146

MA, L., J. STUECKLER, C. KERL et D. CREMERS. 2017, «Multi-view deep learning for consistent semantic mapping with rgb-d cameras», dans arXiv :1703.08866, p. –. 146

MEES, O., A. EITEL et W. BURGARD. 2016, «Choosing smartly : Adaptive multi-modal fusion for object detection in changing environments», dans Intelligent Robots and Systems (IROS), 2016 IEEE/RSJ International Conference on, IEEE, p. 151–156. 146

NGUYEN, A., J. YOSINSKI et J. CLUNE. 2015, «Deep neural networks are easily fooled : High confidence predictions for unrecognizable images», dans Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, p. 427–436. 147

RASTEGARI, M., V. ORDONEZ, J. REDMON et A. FARHADI. 2016, «Xnor-net : Imagenet classification using binary convolutional neural networks», dans European Conference on Computer Vision, Springer, p. 525–542. 147

REN, S., K. HE, R. GIRSHICK et J. SUN. 2015, «Faster r-cnn : Towards real-time object detection with region proposal networks», dans Advances in neural information processing systems, p. 91–99. 146

ZEILER, M. D. et R. FERGUS. 2014, «Visualizing and understanding convolutional networks», dans European conference on computer vision, Springer, p. 818–833. 146

Chapitre 8

Liste complète des références

ACKLEY, D. H., G. E. HINTON et T. J. SEJNOWSKI. 1985, «A learning algorithm for boltzmann machines», Cognitive science, vol. 9, n° 1, p. 147–169.

ALEXE, B., T. DESELAERS et V. FERRARI. 2012, «Measuring the objectness of image windows», IEEE transactions on pattern analysis and machine intelligence, vol. 34, n° 11, p. 2189–2202.

ANGUELOV, D., B. TASKARF, V. CHATALBASHEV, D. KOLLER, D. GUPTA, G. HEITZ et A. NG. 2005, «Discriminative learning of markov random fields for segmentation of 3d scan data», dans Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on, vol. 2, IEEE, p. 169–176.

ARBELÁEZ, P., J. PONT-TUSET, J. T. BARRON, F. MARQUES et J. MALIK. 2014, «Multiscale combinatorial grouping», dans Proceedings of the IEEE conference on computer vision and pattern recognition, p. 328–335.

AUDEBERT, N., B. LE SAUX et S. LEFÈVRE. 2016, «Semantic segmentation of earth observation data using multi-modal and multi-scale deep networks», dans Asian Conference on Computer Vision, Springer, p. 180–196.

BADRINARAYANAN, V., A. KENDALL et R. CIPOLLA. 2015, «SegNet : A Deep Convolutional Encoder-Decoder Architecture for Image Segmentation», arXiv preprint arXiv :1511.00561. URL http://arxiv.org/abs/1511.00561.

BAY, H., T. TUYTELAARS et L. VAN GOOL. 2006, «Surf : Speeded up robust features», Computer vision–ECCV 2006, p. 404–417.

BELLMAN, R. 1961, Adaptive control process : a guided tour.

BERTALMIO, M., A. L. BERTOZZI et G. SAPIRO. 2001, «Navier-stokes, fluid dynamics, and image and video inpainting», dans Computer Vision and Pattern Recognition, 2001. CVPR 2001. Proceedings of the 2001 IEEE Computer Society Conference on, vol. 1, IEEE, p. I–I.

BOULCH, A., M. DE LA GORCE et R. MARLET. 2014, «Piecewise-planar 3D reconstruction with edge and corner regularization», dans Computer Graphics Forum, vol. 33, p. 55–64.

BOULCH, A., B. LE SAUX et N. AUDEBERT. 2017, «Unstructured point cloud semantic labeling using deep segmentation networks», Eurographics/3DOR.

BOULCH, A. et R. MARLET. 2012, «Fast and robust normal estimation for point clouds with sharp features», dans Computer graphics forum, vol. 31, Wiley Online Library, p. 1765–1774.

BROSTOW, G. J., J. FAUQUEUR et R. CIPOLLA. 2009, «Semantic object classes in video : A high-definition ground truth database», Pattern Recognition Letters, vol. 30, n° 2, p. 88–97.

CARREIRA, J. et C. SMINCHISESCU. 2012, «Cpmc : Automatic object segmentation using constrained parametric min-cuts», IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 34, n° 7, p. 1312–1328.

CARUANA, R., A. NICULESCU-MIZIL, G. CREW et A. KSIKES. 2004, «Ensemble selection from libraries of models», dans Proceedings of the twenty-first international conference on Machine learning, ACM, p. 18.

CHARANIYA, A. P., R. MANDUCHI et S. K. LODHA. 2004, «Supervised parametric classification of aerial Lidar data», dans CVPR/W, IEEE, p. 30–30.

CHATFIELD, K., K. SIMONYAN, A. VEDALDI et A. ZISSERMAN. 2014, «Return of the devil in the details : Delving deep into convolutional nets», dans British Machine Vision Conference, p. 6.1–6.12.

CIREŞAN, D. C., U. MEIER, J. MASCI, L. M. GAMBARDELLA et J. SCHMIDHUBER. 2011, «High-performance neural networks for visual object classification», arXiv preprint arXiv :1102.0183.

CLEVERT, D.-A., T. UNTERTHINER et S. HOCHREITER. 2015, «Fast and accurate deep network learning by exponential linear units (elus)», arXiv preprint arXiv :1511.07289.

COUPRIE, C., C. FARABET, L. NAJMAN et Y. LECUN. 2013, «Indoor semantic segmentation using depth information», arXiv preprint arXiv :1301.3572.

DAI, A., A. X. CHANG, M. SAVVA, M. HALBER, T. A. FUNKHOUSER et M. NIESSNER. 2017, «Scannet : Richly-annotated 3d reconstructions of indoor scenes», dans CVPR, Honolulu, Hawai, USA, p. –.

DALAL, N. et B. TRIGGS. 2005, «Histograms of oriented gradients for human detection», dans Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on, vol. 1, IEEE, p. 886–893.

DE SMEDT, Q., H. WANNOUS et J.-P. VANDEBORRE. 2016, «Skeleton-based dynamic hand gesture recognition», dans Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops, p. 1–9.

DENG, J., W. DONG, R. SOCHER, L.-J. LI, K. LI et L. FEI-FEI. 2009, «Imagenet : A large-scale hierarchical image database», dans Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on, IEEE, p. 248–255.

DEVANNE, M., H. WANNOUS, S. BERRETTI, P. PALA, M. DAOUDI et A. DEL BIMBO. 2015, «3-d human action recognition by shape analysis of motion trajectories on riemannian manifold», IEEE transactions on cybernetics, vol. 45, n° 7, p. 1340–1352.

EIGEN, D. et R. FERGUS. 2015, «Predicting depth, surface normals and semantic labels with a common multi-scale convolutional architecture», dans Proceedings of the IEEE International Conference on Computer Vision, p. 2650–2658.

EITEL, A., J. T. SPRINGENBERG, L. SPINELLO, M. RIEDMILLER et W. BURGARD. 2015, «multi-modal deep learning for robust rgb-d object recognition», dans Intelligent Robots and Systems (IROS), 2015 IEEE/RSJ International Conference on, IEEE, p. 681–687.

ENDRES, I. et D. HOIEM. 2010, «Category independent object proposals», Computer Vision–ECCV 2010, p. 575–588.

ENGEL, J., V. KOLTUN et D. CREMERS. 2017, «Direct sparse odometry», IEEE transactions on pattern analysis and machine intelligence.

ENGEL, J., T. SCHÖPS et D. CREMERS. 2014, «Lsd-slam : Large-scale direct monocular slam», dans European Conference on Computer Vision, Springer, p. 834–849.

EVERINGHAM, M., L. GOOL, C. K. WILLIAMS, J. WINN et A. ZISSERMAN. 2010, «The pascal visual object classes (voc) challenge», International journal of computer vision, vol. 88, n° 2.

FEI-FEI, L., R. FERGUS et P. PERONA. 2006, «One-shot learning of object categories», IEEE transactions on pattern analysis and machine intelligence, vol. 28, n° 4, p. 594–611.

FELZENSZWALB, P. F., R. B. GIRSHICK, D. MCALLESTER et D. RAMANAN. 2010, «Object detection with discriminatively trained part-based models», Pattern Analysis and Machine Intelligence, IEEE Transactions on.

FILLIAT, D., E. BATTESTI, S. BAZEILLE, G. DUCEUX, A. GEPPERTH, L. HARRATH, I. JEBARI, R. PEREIRA, A. TAPUS, C. MEYER et al.. 2012, «Rgbd object recognition and visual texture classification for indoor semantic mapping», dans Technologies for Practical Robot Applications (TePRA), 2012 IEEE International Conference on, IEEE, p. 127–132.

FIRMAN, M. 2016, «Rgbd datasets : Past, present and future», dans Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops, p. 19–31.

FORGY, E. 1965, «Cluster analysis of multivariate data : efficiency versus interpretability models», Biometrics, vol. 61, n° 3, p. 768–769.

GEIGER, A., M. ROSER et R. URTASUN. 2010, «Efficient large-scale stereo matching», dans Asian conference on computer vision, Springer, p. 25–38.

GERS, F. A., J. SCHMIDHUBER et F. CUMMINS. 1999, «Learning to forget : Continual prediction with lstm», .

GIRSHICK, R. 2015, «Fast r-cnn», dans Proceedings of the IEEE international conference on computer vision, p. 1440–1448.

GIRSHICK, R., J. DONAHUE, T. DARRELL et J. MALIK. 2014, «Rich feature hierarchies for accurate object detection and semantic segmentation», dans Proceedings of the IEEE conference on computer vision and pattern recognition, p. 580–587.

GRAHAM, B. 2014, «Fractional max-pooling», arXiv preprint arXiv :1412.6071.

GREEN, D. 2011, «A colour scheme for the display of astronomical intensity images», arXiv preprint arXiv :1108.5083.

GRIFFIN, G., A. HOLUB et P. PERONA. 2007, «Caltech-256 object category dataset», .

GUPTA, S., R. GIRSHICK, P. ARBELÁEZ et J. MALIK. 2014, «Learning rich features from rgb-d images for object detection and segmentation», dans European Conference on Computer Vision, Springer, p. 345–360.

GYBENKO, G. 1989, «Approximation by superposition of sigmoidal functions», Mathematics of Control, Signals and Systems, vol. 2, n° 4, p. 303–314.

HAALA, N., C. BRENNER et K.-H. ANDERS. 1998, «3D urban GIS from laser altimeter and 2D map data», International Archives Photogramm. Remote Sens., vol. 32, p. 339–346.

HACKEL, T., N. SAVINOV, L. LADICKY, J. D. WEGNER, K. SCHINDLER et M. POLLEFEYS. 2017, «Semantic3d.net : A new large-scale point cloud classification benchmark», ISPRS Annals of Photogrammetry, Remote Sensing and Spatial Information Science, vol. IV-1/W1.

HANDA, A., V. PATRAUCEAN, V. BADRINARAYANAN, S. STENT et R. CIPOLLA. 2016, «Understanding real world indoor scenes with synthetic data», dans Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, p. 4077–4085.

HARVEY, A. 2010, «Sliding window video», URL https://vimeo.com/12774628, [En ligne ; Page disponible le 10-septembre-2017].

HASSIBI, B. et D. G. STORK. 1993, «Second order derivatives for network pruning : Optimal brain surgeon», dans Advances in neural information processing systems, p. 164–171.

HAZIRBAS, C., L. MA, C. DOMOKOS et D. CREMERS. 2016, «Fusenet : Incorporating depth into semantic segmentation via fusion-based cnn architecture», dans Asian Conference on Computer Vision, Springer, p. 213–228.

HE, K., G. GKIOXARI, P. DOLLÁR et R. GIRSHICK. 2017, «Mask r-cnn», arXiv preprint arXiv :1703.06870.

HE, K., X. ZHANG, S. REN et J. SUN. 2015, «Deep residual learning for image recognition», arXiv preprint arXiv :1512.03385.

HE, K., X. ZHANG, S. REN et J. SUN. 2016a, «Deep residual learning for image recognition», dans Proceedings of the IEEE conference on computer vision and pattern recognition, p. 770–778.

HE, K., X. ZHANG, S. REN et J. SUN. 2016b, «Identity mappings in deep residual networks», dans European Conference on Computer Vision, Springer, p. 630–645.

HERMANS, A., G. FLOROS et B. LEIBE. 2014, «Dense 3d semantic mapping of indoor scenes from rgb-d images», dans Robotics and Automation (ICRA), 2014 IEEE International Conference on, IEEE, p. 2631–2638.

HOCHREITER, S., Y. BENGIO, P. FRASCONI et J. SCHMIDHUBER. 2001, «Gradient flow in recurrent nets : the difficulty of learning long-term dependencies», .

HOCHREITER, S. et J. SCHMIDHUBER. 1997, «Long short-term memory», Neural computation, vol. 9, n° 8, p. 1735–1780.

HOSANG, J., R. BENENSON et B. SCHIELE. 2017, «Learning non-maximum suppression», arXiv preprint arXiv :1705.02950.

HUBER, P. 1981, «J. 1981. robust statistics», New York : John Wiley.

IANDOLA, F. N., M. W. MOSKEWICZ, K. ASHRAF, S. HAN, W. J. DALLY et K. KEUTZER. 2016, «Squeezenet : Alexnet-level accuracy with 50x fewer parameters and< 1mb model size», arXiv preprint arXiv :1602.07360.

IOFFE, S. et C. SZEGEDY. 2015, «Batch normalization : Accelerating deep network training by reducing internal covariate shift», dans International Conference on Machine Learning, p. 448–456.

JANOCH, A., S. KARAYEV, Y. JIA, J. T. BARRON, M. FRITZ, K. SAENKO et T. DARRELL. 2013, «A category-level 3d object dataset : Putting the kinect to work», dans Consumer Depth Cameras for Computer Vision, Springer, p. 141–165.

JAPKOWICZ, N. et S. STEPHEN. 2002, «The class imbalance problem : A systematic study», Intelligent data analysis, vol. 6, n° 5, p. 429–449.

JIANG, J., Z. ZHANG, Y. HUANG et L. ZHENG. 2017, «Incorporating depth into both cnn and crf for indoor semantic segmentation», arXiv preprint arXiv :1705.07383.

KALAL, Z., K. MIKOLAJCZYK et J. MATAS. 2012, «Tracking-learning-detection», Pattern Analysis and Machine Intelligence, IEEE Transactions on.

KATAOKA, H., K. IWATA et Y. SATOH. 2015, «Feature evaluation of deep convolutional neural networks for object recognition and detection», arXiv preprint arXiv :1509.07627.

KAWAGUCHI, K. 2016, «Deep learning without poor local minima», dans Advances in Neural Information Processing Systems, p. 586–594.

KELLEY, H. J. 1960, «Gradient theory of optimal flight paths», Ars Journal, vol. 30, n° 10, p. 947–954.

KENDALL, A., V. BADRINARAYANAN et R. CIPOLLA. 2015, «Bayesian segnet : Model uncertainty in deep convolutional encoder-decoder architectures for scene understanding», arXiv preprint arXiv :1511.02680.

KRÄHENBÜHL, P. et V. KOLTUN. 2011, «Efficient inference in fully connected crfs with gaussian edge potentials», dans Advances in neural information processing systems, p. 109–117.

KRIZHEVSKY, A. et G. HINTON. 2009, «Learning multiple layers of features from tiny images», .

KRIZHEVSKY, A., I. SUTSKEVER et G. E. HINTON. 2012, «Imagenet classification with deep convolutional neural networks», dans Advances in neural information processing systems, p. 1097–1105.

KULIS, B. et al.. 2013, «Metric learning : A survey», Foundations and Trends® in Machine Learning, vol. 5, n° 4, p. 287–364.

LAI, K., L. BO et D. FOX. 2014, «Unsupervised feature learning for 3D scene labeling», dans ICRA, IEEE, p. 3050–3057.

LARLUS, D., J. VERBEEK et F. JURIE. 2010, «Category level object segmentation by combining bag-of-words models with dirichlet processes and random fields», International Journal of Computer Vision, vol. 88, nº 2, p. 238–253.

LECUN, Y. 1987, Modèles connexionnistes de l'apprentissage, thèse de doctorat, These de Doctorat, Universite Paris 6.

LECUN, Y., L. BOTTOU, Y. BENGIO et P. HAFFNER. 1998, «Gradient-based learning applied to document recognition», Proc. of the IEEE, vol. 86, nº 11, p. 2278–2324.

LECUN, Y., J. S. DENKER et S. A. SOLLA. 1990, «Optimal brain damage», dans Advances in Neural Information Processing Systems 2, édité par D. S. Touretzky, Morgan-Kaufmann, p. 598–605. URL http://papers.nips.cc/paper/250-optimal-brain-damage.pdf.

LENZ, I., H. LEE et A. SAXENA. 2015, «Deep learning for detecting robotic grasps», The International Journal of Robotics Research, vol. 34, nº 4-5, p. 705–724.

LI, Z., Y. GAN, X. LIANG, Y. YU, H. CHENG et L. LIN. 2016, «Lstm-cf : Unifying context modeling and fusion with lstms for rgb-d scene labeling», dans European Conference on Computer Vision, Springer, p. 541–557.

LIN, G., C. SHEN, A. VAN DEN HENGEL et I. REID. 2017, «Exploring context with deep structured models for semantic segmentation», IEEE Transactions on Pattern Analysis and Machine Intelligence.

LIU, W., D. ANGUELOV, D. ERHAN, C. SZEGEDY, S. REED, C.-Y. FU et A. C. BERG. 2016, «Ssd : Single shot multibox detector», dans European conference on computer vision, Springer, p. 21–37.

LONG, J., E. SHELHAMER et T. DARRELL. 2015, «Fully convolutional networks for semantic segmentation», dans Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, p. 3431–3440.

LOWE, D. G. 1999, «Object recognition from local scale-invariant features», dans Computer vision, 1999. The proceedings of the seventh IEEE international conference on, vol. 2, Ieee, p. 1150–1157.

LU, D. et Q. WENG. 2007, «A survey of image classification methods and techniques for improving classification performance», International journal of Remote sensing, vol. 28, nº 5, p. 823–870.

LUBER, M., L. SPINELLO et K. O. ARRAS. 2011, «People tracking in RGBD data with online boosted target models», dans Intelligent Robots and Systems (IROS), 2011 IEEE/RSJ International Conference on, IEEE, p. 3844–3849.

MA, L., J. STUECKLER, C. KERL et D. CREMERS. 2017, «Multi-view deep learning for consistent semantic mapping with rgb-d cameras», dans arXiv :1703.08866, p. –.

MATURANA, D. et S. SCHERER. 2015, «Voxnet : A 3D convolutional neural network for real-time object recognition», dans IROS, Hamburg, Germany, p. 922–928.

MEES, O., A. EITEL et W. BURGARD. 2016, «Choosing smartly : Adaptive multi-modal fusion for object detection in changing environments», dans Intelligent Robots and Systems (IROS), 2016 IEEE/RSJ International Conference on, IEEE, p. 151–156.

MILLER, G. A. 1995, «Wordnet : a lexical database for english», Communications of the ACM, vol. 38, n° 11, p. 39–41.

NGUYEN, A., J. YOSINSKI et J. CLUNE. 2015, «Deep neural networks are easily fooled : High confidence predictions for unrecognizable images», dans Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, p. 427–436.

OHN-BAR, E. et M. TRIVEDI. 2013, «Joint angles similarities and hog2 for action recognition», dans Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops, p. 465–470.

OJALA, T., M. PIETIKAINEN et T. MAENPAA. 2002, «Multiresolution gray-scale and rotation invariant texture classification with local binary patterns», IEEE Transactions on pattern analysis and machine intelligence, vol. 24, n° 7, p. 971–987.

OREIFEJ, O. et Z. LIU. 2013, «Hon4d : Histogram of oriented 4d normals for activity recognition from depth sequences», dans Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, p. 716–723.

PRISACARIU, V., I. REID et al.. 2009, «fasthog-a real-time gpu implementation of hog», Department of Engineering Science, vol. 2310, n° 9.

QI, C. R., L. YI, H. SU et L. J. GUIBAS. 2017, «Pointnet++ : Deep hierarchical feature learning on point sets in a metric space», arXiv preprint arXiv :1706.02413.

RAJ, A., D. MATURANA et S. SCHERER. 2015, «Multi-scale convolutional architecture for semantic segmentation», .

RASTEGARI, M., V. ORDONEZ, J. REDMON et A. FARHADI. 2016, «Xnor-net : Imagenet classification using binary convolutional neural networks», dans European Conference on Computer Vision, Springer, p. 525–542.

RAZAKARIVONY, S. et F. JURIE. 2014, «Autoencodeurs discriminants pour la détection de cibles faiblement résolues», dans Reconnaissance de formes et intelligence artificielle (RFIA) 2014.

REDMON, J., S. DIVVALA, R. GIRSHICK et A. FARHADI. 2016, «You only look once : Unified, real-time object detection», dans Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, p. 779–788.

REDMON, J. et A. FARHADI. 2016, «Yolo9000 : better, faster, stronger», arXiv preprint arXiv :1612.08242.

REN, S., K. HE, R. GIRSHICK et J. SUN. 2015, «Faster r-cnn : Towards real-time object detection with region proposal networks», dans Advances in neural information processing systems, p. 91–99.

RONNEBERGER, O., P. FISCHER et T. BROX. 2015, «U-Net : Convolutional networks for biomedical image segmentation», dans MICCAI, Munich, ISBN 978-3-319-24574-4, p. 234–241, doi :10.1007/978-3-319-24574-4_28. URL http://dx.doi.org/10.1007/978-3-319-24574-4_28.

RUMELHART, D. E., G. E. HINTON, R. J. WILLIAMS et al.. 1988, «Learning representations by back-propagating errors», Cognitive modeling, vol. 5, n° 3, p. 1.

RUSSAKOVSKY, O., J. DENG, H. SU, J. KRAUSE, S. SATHEESH, S. MA, Z. HUANG, A. KARPATHY, A. KHOSLA, M. BERNSTEIN, A. C. BERG et L. FEI-FEI. 2015, «ImageNet Large Scale Visual Recognition Challenge», International Journal of Computer Vision (IJCV), vol. 115, n° 3, doi :10.1007/s11263-015-0816-y, p. 211–252.

RUSU, R. B., N. BLODOW et M. BEETZ. 2009, «Fast point feature histograms (fpfh) for 3d registration», dans Robotics and Automation, 2009. ICRA'09. IEEE International Conference on, IEEE, p. 3212–3217.

SANFOURCHE, M., B. LE SAUX, A. PLYER et G. LE BESNERAIS. 2014, «Cartographie et interprétation de l'environnement par drone», dans Colloque scientifique francophone Drones et moyens légers aéroportés d'observation.

SERMANET, P., D. EIGEN, X. ZHANG, M. MATHIEU, R. FERGUS et Y. LECUN. 2013, «Overfeat : Integrated recognition, localization and detection using convolutional networks», arXiv preprint arXiv :1312.6229.

SFIKAS, K., T. THEOHARIS et I. PRATIKAKIS. 2017, «Exploiting the PANORAMA Representation for Convolutional Neural Network Classification and Retrieval», dans Eurographics Workshop on 3D Object Retrieval, Lyon, France, p. –.

SHRIVASTAVA, A., A. GUPTA et R. GIRSHICK. 2016, «Training region-based object detectors with online hard example mining», dans Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, p. 761–769.

SILBERMAN, N., D. HOIEM, P. KOHLI et R. FERGUS. 2012, «Indoor segmentation and support inference from rgbd images», Computer Vision–ECCV 2012, p. 746–760.

SIMONYAN, K. et A. ZISSERMAN. 2014, «Very deep convolutional networks for large-scale image recognition», arXiv preprint arXiv :1409.1556.

SMITHSON, S. C., G. YANG, W. J. GROSS et B. H. MEYER. 2016, «Neural networks designing neural networks : Multi-objective hyper-parameter optimization», dans Computer-Aided Design (ICCAD), 2016 IEEE/ACM International Conference on, IEEE, p. 1–8.

SONG, S., S. P. LICHTENBERG et J. XIAO. 2015, «SUN RGB-D : A RGB-D scene understanding benchmark suite», dans CVPR, Boston, USA, p. 567–576.

SPINELLO, L. et K. O. ARRAS. 2011, «People detection in rgb-d data», dans Intelligent Robots and Systems (IROS), 2011 IEEE/RSJ International Conference on, IEEE, p. 3838–3843.

SPINELLO, L. et K. O. ARRAS. 2012, «Leveraging RGBD data : Adaptive fusion and domain adaptation for object detection», dans Robotics and Automation (ICRA), 2012 IEEE International Conference on, IEEE, p. 4469–4474.

STEDER, B., G. GRISETTI, M. VAN LOOCK et W. BURGARD. 2009, «Robust on-line model-based object detection from range images», dans Intelligent Robots and Systems, 2009. IROS 2009. IEEE/RSJ International Conference on, IEEE, p. 4739–4744.

SU, H., , C. QI, K. MO et L. GUIBAS. 2017, «Pointnet : Deep learning on point sets for 3d classification and segmentation», dans CVPR, Honolulu, Hawai, USA, p. –.

SU, H., S. MAJI, E. KALOGERAKIS et E. LEARNED-MILLER. 2015, «Multi-view convolutional neural networks for 3D shape recognition», dans ICCV, p. 945–953.

SZEGEDY, C., S. IOFFE, V. VANHOUCKE et A. A. ALEMI. 2017, «Inception-v4, inception-resnet and the impact of residual connections on learning.», dans AAAI, p. 4278–4284.

SZEGEDY, C., W. LIU, Y. JIA, P. SERMANET, S. REED, D. ANGUELOV, D. ERHAN, V. VANHOUCKE et A. RABINOVICH. 2015, «Going deeper with convolutions», dans Proceedings of the IEEE conference on computer vision and pattern recognition, p. 1–9.

TAI, L., Q. YE et M. LIU. 2016, «Pca-aided fully convolutional networks for semantic segmentation of multi-channel fmri», arXiv preprint arXiv :1610.01732.

TOMBARI, F., S. SALTI et L. DI STEFANO. 2010, «Unique signatures of histograms for local surface description», dans ECCV, Springer, Hersonissos, Crete, p. 356–369.

TURK, M. A. et A. P. PENTLAND. 1991, «Face recognition using eigenfaces», dans Computer Vision and Pattern Recognition, 1991. Proceedings CVPR'91., IEEE Computer Society Conference on, IEEE, p. 586–591.

UIJLINGS, J., K. VAN DE SANDE, T. GEVERS et A. SMEULDERS. 2013, «Selective search for object recognition», International journal of computer vision.

VALADA, A., G. L. OLIVEIRA, T. BROX et W. BURGARD. 2016, «Deep multispectral semantic scene understanding of forested environments using multi-modal fusion», dans International Symposium on Experimental Robotics, Springer, p. 465–477.

VIOLA, P., M. J. JONES et D. SNOW. 2005, «Detecting pedestrians using patterns of motion and appearance», International Journal of Computer Vision, vol. 63, n° 2, p. 153–161.

WAHL, E., U. HILLENBRAND et G. HIRZINGER. 2003, «Surflet-pair-relation histograms : a statistical 3d-shape representation for rapid classification», dans 3-D Digital Imaging and Modeling, 2003. 3DIM 2003. Proceedings. Fourth International Conference on, IEEE, p. 474–481.

WAN, L., M. ZEILER, S. ZHANG, Y. L. CUN et R. FERGUS. 2013, «Regularization of neural networks using dropconnect», dans Proceedings of the 30th international conference on machine learning (ICML-13), p. 1058–1066.

WANG, H., A. CRUZ-ROA, A. BASAVANHALLY, H. GILMORE, N. SHIH, M. FELDMAN, J. TOMASZEWSKI, F. GONZALEZ et A. MADABHUSHI. 2014, «Mitosis detection in breast cancer pathology images by combining handcrafted and convolutional neural network features», Journal of Medical Imaging, vol. 1, n° 3, p. 034 003–034 003.

WIKIPEDIA. 2017, «Réseau neuronal convolutif — Wikipédia, l'encyclopédie libre», URL https://goo.gl/jQfwmi, [En ligne ; Page disponible le 12-août-2017].

WOLF, C., E. LOMBARDI, J. MILLE, O. CELIKTUTAN, M. JIU, E. DOGAN, G. EREN, M. BACCOUCHE, E. DELLANDRÉA, C.-E. BICHOT et al.. 2014, «Evaluation of video activity localizations integrating quality and quantity measurements», Computer Vision and Image Understanding, vol. 127, p. 14–30.

WU, Z., S. SONG, A. KHOSLA, F. YU, L. ZHANG, X. TANG et J. XIAO. 2015, «3D shapenets : A deep representation for volumetric shapes», dans CVPR, Boston, USA, p. 1912–1920.

XIAO, J., A. OWENS et A. TORRALBA. 2013, «SUN3D : A database of big spaces reconstructed using sfm and object labels», dans 2013 IEEE International Conference on Computer Vision, Sidney, Australia, p. 1625–1632.

XU, R. et D. WUNSCH. 2005, «Survey of clustering algorithms», IEEE Transactions on neural networks, vol. 16, n° 3, p. 645–678.

YANG, J., Y.-G. JIANG, A. G. HAUPTMANN et C.-W. NGO. 2007, «Evaluating bag-of-visual-words representations in scene classification», dans Proceedings of the international workshop on Workshop on multimedia information retrieval, ACM, p. 197–206.

YANG, W., Y. WANG, A. VAHDAT et G. MORI. 2012, «Kernel latent svm for visual recognition», dans Advances in neural information processing systems, p. 809–817.

ZEILER, M. D. et R. FERGUS. 2014, «Visualizing and understanding convolutional networks», dans European conference on computer vision, Springer, p. 818–833.

ZHAO, H., J. SHI, X. QI, X. WANG et J. JIA. 2016, «Pyramid scene parsing network», arXiv preprint arXiv :1612.01105.

Annexe A

Figures annexes

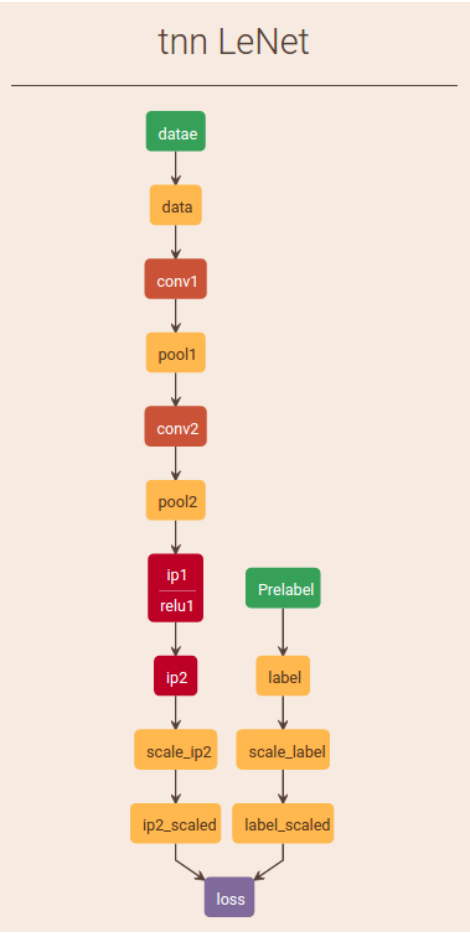


FIGURE A.1 – Structure du *CNN* LeNet

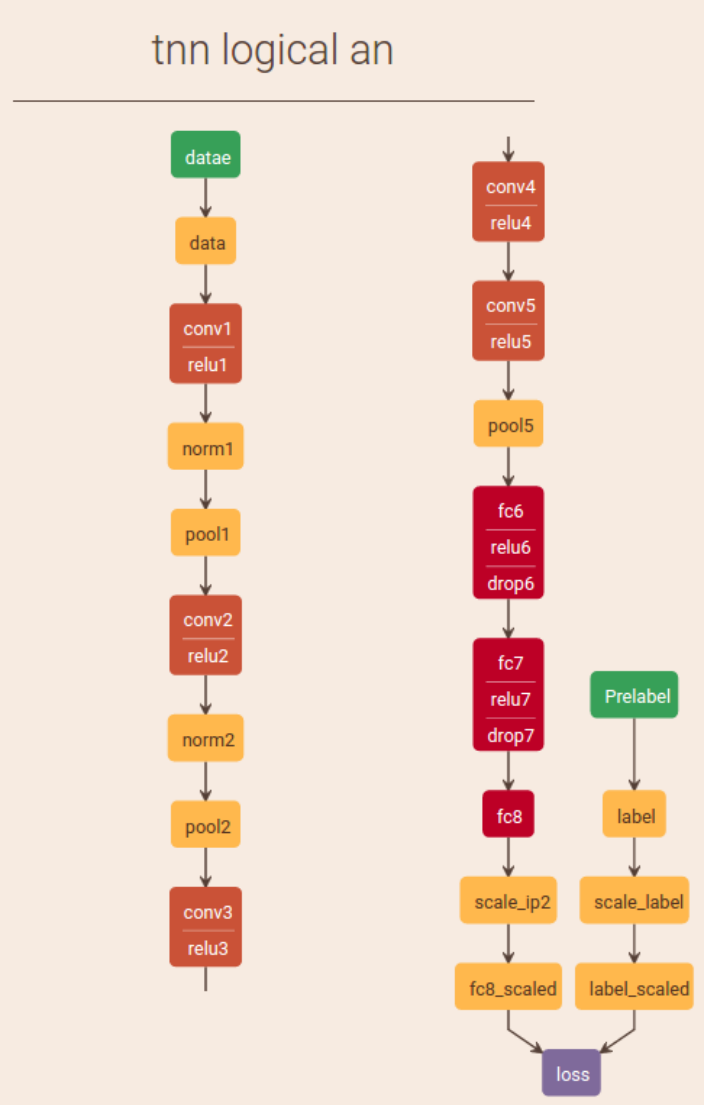


FIGURE A.2 – Structure du *CNN* AlexNet

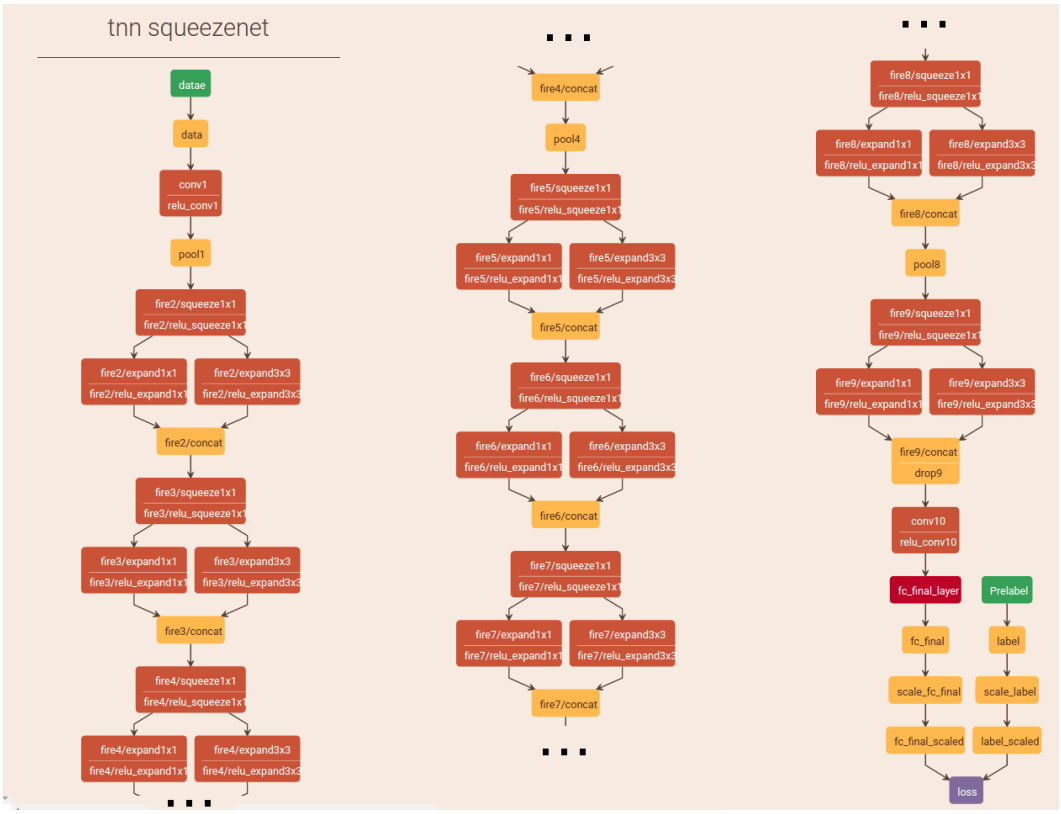


FIGURE A.3 – Structure du *CNN* SqueezeNet

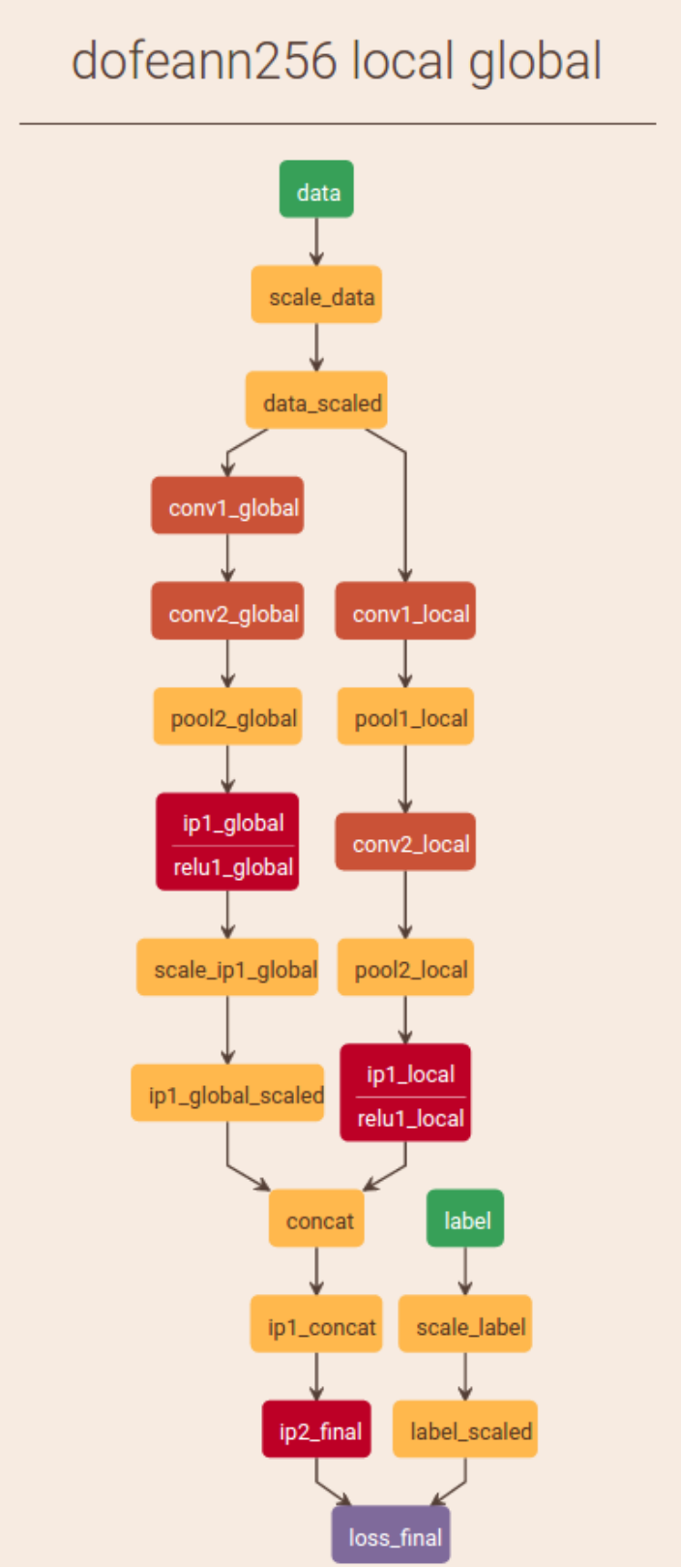


FIGURE A.4 – Structure du *CNN* Dofeann

Annexe B

Liste des acronymes

ANN réseau de neurones artificiel – ou *Artificial Neural Network*.. 32

BN *batch normalization* – ou normalisation du batch (BN) – est une méthode pour normaliser la distribution statistique des données d'un batch. Cette opération peut-être appliquée sur chaque tenseur 4D intermédiaire entre les différentes couches d'un *CNN*. Introduite par IOFFE et SZEGEDY [2015], elle résulte du constat que chaque itération peut présenter un batch avec une distribution très différente de la précédente itération et ainsi rendre l'entraînement plus complexe. La *BN* permet ainsi d'homogénéiser chaque *batch*, d'accélérer l'entraînement, de dépasser les performances d'un modéle similaire sans *BN*, d'être moins sensible aux paramètres initiaux et de commencer l'entraînement avec un taux d'apprentissage plus élevé.. 37, 39, 40

CNN réseau de neurones convolutionnels – ou *Convolutional Neural Network*. Également appelé réseau de neurones convolutifs.. 37–40, 46–50, 53, 56, 72–74, 76, 78, 81, 98, 146, 147, V

CRF champ aléatoire conditionnel – ou *Conditional Random Field*.. 52, 53, 124

DL L'apprentissage profond – ou *Deep Learning* (*DL*), est une branche de l'apprentissage automatique qui utilise des réseaux de neurones.. 32

E+D encodeur+décodeur – ou *Encoder* + *Decoder*.. 50

fenêtre glissante fenêtre glissante – ou *Sliding window* – est une méthode pour parcourir une image et proposer des patchs de façon arbitrairement pré-determinée.. 16, 42

FN *False Negative*(*FN*), désigne les propositions d'une méthode classifiées comme fausses alors qu'elles sont vraies.. 15

FP *False Positive* (*FP*), désigne les propositions d'une méthode classifiées comme vraies alors qu'elles sont fausses.. 14, 15, 105

GPU unité de traitement graphique – ou *Graphical Processing Unit*.. 41

HOGs histogramme de gradients orientés – ou *Histogram of Oriented Gradients*.. 13, 14, 46

IoU intersection divisée par l'union – ou *Intersection over Union* – est un indice de mesure de la qualité d'un recoupement, entre un patch proposé et un patch de la vérité terrain par exemple, ou entre une surface pixellique proposée et une certaine surface pixellique de la vérité terrain.. 18, 19

MLP perceptron multi-couches – ou *Multi Layer Perceptron*.. 32, 42

NMS suppression des non-maximums – ou *Non Maximum Suppression* – est une méthode permettant de sélectionner des patchs parmi d'autre patch en favorisant les meilleurs scores et supprimant les patchs ayant une intersection trop forte avec les meilleurs patchs.. 17, 18, 53, 99, 100

RGB rouge, vert, bleue – ou *Red, Green, Blue*.. 12, 22, 53–56, 68, 72, 86–88, 93, 94, 97–99, 108, 118–120, 125, 128, 130

RGB-D rouge, vert, bleue, profondeur – ou *Red, Green, Blue, Depth map* (*RGB-D*).. 19, 20, 22, 23, 25, 26, 54–57, 67, 77, 87, 99–103, 116–121, 124–128, 130, 135, 146

SGD descente de gradient stochastique – ou *Stochastic Gradient Descent*.. 35, 36, 40

SLAM localisation et cartographie simultanées – ou *Simultaneous Localization And Mapping* – est une méthode qui vise à se localiser dans une carte construire au fur et à mesure d'une exploration.. 11, 14, 19, 56, 116

SVM machine à vecteurs de support – ou *Support Vector Machine*.. 14, 46

TN *True Negative* (*TN*), désigne les propositions d'une méthode correctement classifiées comme fausses.. 15

TP *True Positive* (*TP*), désigne les propositions d'une méthode correctement classifiées comme vraies.. 14, 15, 108

Annexe C

Glossaire

auto-encodeur Structure d'encodeur+décodeur qui tente de reconstruire son entrée.. 50

batch Anglicisme pour "paquet, regroupement, série". Le terme *batch* désigne un regroupement, sous entendu de tenseurs. En pratique les images constituantes d'un batch (tenseurs 3D) sont concaténées dans un tenseur selon une nouvelle dimension (tenseur 4D). Les bibliothèques de *Deep Learning* sont généralement optimisées pour traiter directement des batchs ce qui permet d'augmenter la vitesse de traitement (à l'entraînement comme au test). Un avantage direct concerne l'apprentissage stochastique : la distribution des données observée à chaque itération est ainsi plus homogène que si un seul exemple avait été utilisé.. 39, 66

HHA Désigne un encodage de la carte de profondeur. L'information est retraduite en trois canaux : **H**auteur, disparité **H**orizontale et **A**ngle de la normale à la surface avec le vecteur gravité.. 54

image Le terme image désigne l'ensemble des pixels d'une image numérique.. 16

patch Anglicisme qui pourrait se traduire par "zone", le terme patch désigne une sous-partie d'une image (le plus souvent rectangulaire).. 16, V

rappel Le rappel est un indice qui mesure la capacité d'une méthode à trouver l'ensemble des entitées recherchées. En détection d'objet, une méthode qui propose des objets partout dans l'image aura un rappel de 100%, même s'il elle propose des mauvaises détections, car elle aura trouvé tous les objets. C'est pourquoi le rappel est généralement associé à la précision qui mesure la qualité des détections proposées. Il est possible d'étudier une méthode paramétrée avec une courbe paramétrique précision-rappel qui trace la précision en fonction du rappel pour un paramètre donné.. 16

surapprentissage Le terme surapprentissage désigne la situation où un algorithme se spécialise trop sur les données d'entraînement et n'est plus capable de généraliser. La conséquence du surapprentissage est une perte de performance sur l'ensemble de test.. 39

Annexe D

Liste complète des références

IOFFE, S. et C. SZEGEDY. 2015, «Batch normalization : Accelerating deep network training by reducing internal covariate shift», dans International Conference on Machine Learning, p. 448–456. V

université PARIS-SACLAY

ONERA
THE FRENCH AEROSPACE LAB

ÉCOLE POLYTECHNIQUE
UNIVERSITÉ PARIS-SACLAY

Titre : Reconnaissance visuelle robuste par réseaux de neurones dans des scénarios d'exploration robotique. Détecte-moi si tu peux !

Mots clés : classification, détection, segmentation sémantique, RGBD, réseaux de neurones

Résumé : L'objectif principal ce travail de thèse est la reconnaissance visuelle pour un robot mobile dans des conditions difficiles. En particulier nous nous intéressons aux réseaux de neurones qui présentent aujourd'hui les meilleures performances en vision par ordinateur. Nous avons étudié le principe de sélection de méthodes pour la classification d'images 2D en utilisant un réseau de neurones sélecteur pour choisir le meilleur classifieur disponible étant donnée la situation observée. Cette stratégie fonctionne lorsque les données peuvent être facilement partitionnées vis-à-vis des classifieurs disponibles, ce qui est le cas quand des modalités complémentaires sont utilisées. Nous avons donc utilisé des données RGB-D (2.5D) en particulier appliquées à la détection de personnes. Nous proposons une combinaison de réseaux de neurones détecteurs indépendants propres à chaque modalité (couleur & carte de profondeur) basés sur une même architecture (le *Faster RCNN*). Nous partageons des résultats intermédiaires des détecteurs pour leur permettre de se compléter et d'améliorer la performance globale en situation difficile (perte de luminosité ou bruit d'acquisition de la carte de profondeur). Nous établissons un nouvel état de l'art dans le domaine et proposons un jeu de données plus complexe et plus riche à la communauté (ONERA.ROOM). Enfin, nous avons fait usage de l'information 3D contenue dans les images RGB-D au travers d'une méthode multi-vue. Nous avons défini une stratégie de génération de vues virtuelles 2D cohérentes avec la structure 3D. Pour une tâche de segmentation sémantique, cette approche permet d'augmenter artificiellement les données d'entraînement pour chaque image RGB-D et d'accumuler différentes prédictions lors du test. Nous obtenons de nouveaux résultats de référence sur les jeux de données SUNRGBD et NYUDv2. Ces travaux de thèse nous ont permis d'aborder de façon originale des données robotiques 2D, 2.5D et 3D avec des réseaux de neurones. Que ce soit pour la classification, la détection et la segmentation sémantique, nous avons non seulement validé nos approches sur des jeux de données difficiles, mais également amené l'état de l'art à un nouveau niveau de performance.

Title : Robust visual recognition by neural networks in robotic exploration scenarios. Detect me if you can !

Keywords : classification, detection, semantic segmentation, RGBD, artificial neural networks

Abstract : The main objective of this thesis is visual recognition for a mobile robot in difficult conditions. We are particularly interested in neural networks which present today the best performances in computer vision. We studied the concept of method selection for the classification of 2D images by using a neural network selector to choose the best available classifier given the observed situation. This strategy works when data can be easily partitioned with respect to available classifiers, which is the case when complementary modalities are used. We have therefore used RGB-D data (2.5D) in particular applied to people detection. We propose a combination of independent neural network detectors specific to each modality (color & depth map) based on the same architecture (Faster RCNN). We share intermediate results of the detectors to allow them to complement and improve overall performance in difficult situations (luminosity loss or acquisition noise of the depth map). We are establishing new state of the art scores in the field and propose a more complex and richer data set to the community (ONERA.ROOM). Finally, we made use of the 3D information contained in the RGB-D images through a multi-view method. We have defined a strategy for generating 2D virtual views that are consistent with the 3D structure. For a semantic segmentation task, this approach artificially increases the training data for each RGB-D image and accumulates different predictions during the test. We obtain new reference results on the SUNRGBD and NYUDv2 datasets. All these works allowed us to handle in an original way 2D, 2.5D and 3D robotic data with neural networks. Whether for classification, detection and semantic segmentation, we not only validated our approaches on difficult data sets, but also brought the state of the art to a new level of performance.