



TÉLÉCOM SUDPARIS  
École de l'institut Mines-Télécom  
9 Rue Charles Fourier, 91000 Évry

KERDONCUFF Tanguy

---

# Recherche en Deep Learning pour la segmentation semantique d'images

---

**Résumé :** Ce compte rendu présente des recherches effectuées lors d'un stage de fin d'étude en Deep Learning. Le domaine d'application étant la segmentation d'images de ville.

**Mots clefs :** Recherche, Intelligence Artificielle, Machine learning, Réseau de Neurones, Deep learning, Segmentation d'images, Voitures autonomes, Mathématiques, Python, Pytorch.

Élève : [Kerdoncuff Tanguy](#)

Encadrant au laboratoire Hubert Curien : [Rémi Emonet](#)

Encadrant à Télécom SudParis : [Emmanuel Monfrini](#)

# Table des matières

<b>1</b>	<b>Introduction</b>	<b>1</b>
<b>2</b>	<b>Dans la peau d'un enseignant-chercheur</b>	<b>1</b>
2.1	Laboratoire Hubert Curien . . . . .	2
2.1.1	Rapide présentation du laboratoire . . . . .	2
2.1.2	Une équipe de choc . . . . .	2
2.2	Un équilibre entre cours et recherche . . . . .	3
2.2.1	Un rythme de vie agité . . . . .	3
2.2.2	Cours de modélisation de systèmes mécaniques en Python . . . . .	4
2.3	Deux analyses poussées sur la recherche . . . . .	5
2.3.1	Les articles scientifiques . . . . .	5
2.3.2	Les différentes thèses et les profils associés . . . . .	6
2.4	En dehors du laboratoire . . . . .	8
2.4.1	Conférence d'Apprentissage . . . . .	8
2.4.2	Tournoi de foot . . . . .	10
2.5	Apport du stage . . . . .	10
<b>3</b>	<b>Segmentation d'image avec un réseau de neurones : GridNet</b>	<b>11</b>
3.1	La segmentation d'image . . . . .	12
3.1.1	CityScapes : un jeu de données . . . . .	12
3.1.2	Un critère de performance : $mIoU$ . . . . .	13
3.2	Une architecture : GridNet . . . . .	16
3.2.1	GridNet généralise les réseaux convolutifs classiques . . . . .	16
3.2.2	Code de GridNet en Python . . . . .	18
3.2.3	Quelques résultats et analyses . . . . .	19
3.3	Les avancées théoriques de ce stage . . . . .	25
3.3.1	Fonction de perte : cross entropy, fonction de coût : $mIoU$ . . . . .	25
3.3.2	Première relaxation continue du problème . . . . .	26
3.3.3	L'extension de Lovasz . . . . .	30
3.3.4	Transformation de l'extension de Lovasz . . . . .	32
3.3.5	Conclusion des approximations de Lovasz . . . . .	33
<b>4</b>	<b>Conclusion</b>	<b>34</b>

# 1 Introduction



FIGURE 1 – Un exemple de segmentation d’images.

Le but initial de mon stage était de comprendre GridNet [1], une architecture de réseaux de neurones adaptée à la segmentation sémantiques d’images. Cette architecture a été inventée par [Damien Fourure](#). Il a ensuite fallu écrire le code associé, en Python, grâce à la bibliothèque Pytorch. L’image 1 est un exemple de segmentation d’images de ville avec GridNet. Une fois ce travail effectué l’objectif était de modifier cette architecture pour arriver à de meilleurs résultats. Je me suis particulièrement intéressé aux fonctions de pertes utilisées lors de l’apprentissage du réseau de neurones.

Ce rapport s’articulera autour de deux grandes parties. D’une part une analyse de la vie dans un laboratoire de recherche. D’autre part une partie plus technique qui fera une rapide explication de la segmentation par le machine learning et qui expliquera les avancées apportées par le stage. Les deux sous-parties (3.1 et 3.2) se veulent abordable et intéressante même pour des personnes n’ayant pas beaucoup de connaissances en mathématiques ou en machine learning. La dernière sous-partie (3.3) servira plutôt de récapitulatif complet et rigoureux des avancées théoriques de ce stage.

# 2 Dans la peau d’un enseignant-chercheur

Un stagiaire ingénieur en entreprise est vraiment plongé dans le travail d’un vrai ingénieur : rendez-vous, problème technique, projet... Du moins cela fut le cas lors de mon expérience de stage de deuxième année. En revanche un stage en laboratoire s’apparente plus au travail d’un doctorant que d’un enseignant-chercheur. Néanmoins je vais essayer dans cette première partie de décrire certains aspects du métier d’enseignant-chercheur.

## 2.1 Laboratoire Hubert Curien

Je vais rapidement présenter le laboratoire pour définir le cadre de mon stage puis je présenterai l'équipe avec qui j'ai travaillé.

### 2.1.1 Rapide présentation du laboratoire

Le laboratoire Hubert Curien est situé à Saint-Etienne et est rattaché à l'université Jean-Monnet, au CNRS et à l'Institut d'Optique Graduate School. Je faisais parti de la thématique Data Intelligence constituée d'une quarantaine de personnes (doctorants et stagiaires compris) qui travaillent sur des problèmes de machine learning et de data mining. Dans cette sous-catégorie qui étudie le machine learning on retrouve : le transfert learning, la détection de fraude, le métrique learning, le traitement de la langue et enfin la vision par ordinateur. C'est sur ce dernier domaine que je vais travailler. On arrive finalement à une dizaine de personnes que je vais côtoyer quotidiennement et qui travaillent sur des problématiques de machine learning.

### 2.1.2 Une équipe de choc

Une rapide présentation des gens que j'ai côtoyés s'impose. Tout en haut de la pyramide il y a Marc, directeur adjoint du laboratoire, mais toujours en lien étroit avec le groupe Data Intelligence. Vient alors Amaury, à la tête du groupe Data Intelligence, avec qui je n'ai que très peu discuté à cause d'une longue absence pour problèmes médicaux. Ensuite, on trouve les enseignants-chercheurs : Baptiste, Émilie et Rémi, mon encadrant. Ces deux derniers sont dans le même bureau à côté du mien avec aussi Emmanuel un post-doc avec qui je me suis rapidement bien entendu. On peut alors descendre dans la hiérarchie avec les doctorants de mon bureau, Nam, Jordan, Léo, Valentina et Guillaume. Et pour finir il y a Rémi (un autre) et moi-même qui sommes stagiaire de M2. La pyramide continue lorsqu'il s'agit de stagiaires, les M2 ont le droit d'être dans la même salle que les doctorants, tandis que les stagiaires de M1 sont dans une autre salle (pour des raisons de place). Ceci fait à mon avis une grande différence, j'aurais passé un stage nettement moins plaisant si j'avais été coupé du reste de l'équipe en étant dans une autre salle. Il y aura même Fabien, stagiaire de L3 qui rejoindra l'équipe au milieu de mon stage. Un peu à l'écart de cette hiérarchie, on trouve Thomas qui aide à monter divers projets d'informatique et qui m'a appris à utiliser les clusters sur lesquels on effectue nos calculs.

## 2.2 Un équilibre entre cours et recherche

Après ce bref résumé de la situation, je vais tenter d'expliquer en quoi la vie au laboratoire est souvent moins monotone que celle en entreprise. Tout d'abord en attaquant l'aspect du peu de contraintes imposées puis en parlant des cours qui ont une place importante au sein du laboratoire.

### 2.2.1 Un rythme de vie agité

On va essayer de comprendre ici de quoi est fait l'emploi du temps d'un enseignant-chercheur et de le comparer au mien lors de mon stage et aussi, dans une certaine mesure, à l'emploi du temps classique d'un employé d'entreprise. Tout d'abord un sujet récurrent, les horaires. Sur ma convention, j'ai des horaires précis à faire, mais mon encadrant de stage m'a expliqué que cela n'avait pas beaucoup d'importance tant que le travail était bien fait. Tout le monde peu venir à peu près quand il veut. Néanmoins, en pratique, la plupart des gens arrivent entre 8 h et 10 h et repartent entre 16 h 30 et 19 h et de manière assez régulière. La particularité de ce laboratoire est qu'il est en zone à régime restrictif (ZRR) pour protéger les recherches techniques et scientifiques. Cela implique une fermeture totale du laboratoire à 21 h et une ouverture à 7 h.

Mais les horaires au laboratoire sont encore plus déformés à cause des cours qui doivent être dispensés à l'université. Les enseignants-chercheurs du laboratoire s'occupent en grande partie des cours de L1, L2, L3 d'informatique et aussi de trois masters, dont un en machine learning. Je parlerai plus en détails des cours dans la sous-partie suivante, car cela me semble être un point important du travail dans le laboratoire.

Passons maintenant à l'organisation du travail de recherche. Pour moi, cela s'organisait autour de 3 grandes activités. Tout d'abord lire des articles pour comprendre l'état de l'art, les nouvelles idées, ce qui évite de tout réinventer. Il y a ensuite une part de réflexion : essayer de trouver des idées pertinentes, les creuser pour voir si elles sont réalisables et en discuter avec d'autres (dans mon cas, il s'agit surtout mon encadrant de stage). Enfin, coder un algorithme pour tester les idées. Évidemment, cet enchaînement présenté de manière structuré est bien souvent mis à mal, et on oscille en permanence entre les trois. Un enseignant-chercheur (ou un doctorant) a, en plus des trois items précédents, d'autres obligations. Il doit donner de nombreux cours, écrire des articles scientifiques et parfois aider les conférences à trouver les meilleurs articles pour les publier. Ce travail est celui des "reviewers" dont on reparlera plus tard. Ce qui m'amène à aborder le sujet des conférences et des workshop (sorte de mini-conférence sur la journée). J'ai pu participer à une conférence à Rouen et à un workshop à Lyon. Enfin, il ne faut pas oublier les réunions, notamment d'équipes, qui se font de tous les mois et qui permettent surtout d'expliquer l'avancement des financements pour le laboratoire. Pas toujours facile de

comprendre avec 4 acronymes inconnus à chaque phrase, mais cela viendra sans doute avec le temps.

### 2.2.2 Cours de modélisation de systèmes mécaniques en Python

Pour parler plus spécifiquement des cours, je vais plutôt présenter le cours auquel j'ai participé. J'ai donc aidé mon tuteur de stage, Rémi, à donner des TP de python à des L1 physiciens. L'objectif étant de les aider à créer une simulation d'un mouvement physique d'un objet. Chaque groupe avait son propre projet personnel. Un exemple classique est le mouvement d'objets autour de planètes dont la simulation est simple et est assez amusante à voir. C'est donc un cours un peu particulier, plutôt une aide aux projets que des TP classiques.

Le moins que l'on puisse dire c'est que je n'étais pas de trop, surtout lorsque s'approche la deadline. Chaque groupe à des questions à poser, même si la plupart du temps cela se résume à "ça marche pas". Le principal problème résidait dans le fait que les élèves ne savaient pas quoi faire en cas d'erreur. Il faut arriver à trouver la ligne qui pose le problème dans le code, puis si on ne comprend pas immédiatement, essayer d'afficher des informations pour voir ce qui se passe mal. Au début, j'aids les élèves en résolvant leurs bugs informatiques et en expliquant ce qui n'allait pas. Or, il m'a semblé que ce n'était pas la bonne attitude, il vaut plutôt leur expliquer que faire dans ces situations-là. Il faut que l'élève comprenne par lui-même que faire pour résoudre l'erreur. Comprendre comment résoudre les erreurs est crucial, cela permet de former des élèves indépendants et autonomes.

J'ai remarqué aussi que l'informatique était un exemple de matière très difficile à maîtriser. Alors que j'ai plus ou moins 5 ans d'expérience en python et en langage de programmation en général, j'ai quelques fois eu des difficultés pour résoudre les problèmes. Il y a une grande différence avec d'autres matières dans lesquelles il "suffit" d'avoir des connaissances brutes. En effet en informatique, il faut arriver à comprendre la logique de celui qui a créé le programme ce qui peut prendre du temps surtout lorsque cette personne n'a pas les raisonnements classiques de l'informatique (j'y reviendrais concernant mon propre code dans la deuxième partie). La pire des situations est quand le programme renvoie des valeurs, mais juste pas celles attendues, il n'y a donc pas d'erreur syntaxique et il faut comprendre tout le code pour savoir où est le problème.

Ce passage de l'autre côté du bureau fut très intéressant. J'ai pu m'intéresser aux méthodes d'enseignement et à la manière de bien expliquer. Avec l'aide d'internet et d'un livre gentiment prêté par Rémi, j'ai pu comprendre un certain nombre d'aspects, même s'ils restent assez souvent théorique. Une théorie que j'ai pu lire est qu'une personne trop forte dans un domaine avait du mal à expliquer ce sujet car tout lui semblait évident. Il est alors très difficile de revenir aux fondamentaux et surtout aux intuitions utiles pour comprendre le problème. Finalement,

selon cette théorie, je pense avoir le bon niveau pour enseigner le python à des débutants. Pas trop expert, mais avec des connaissances suffisantes pour résoudre leurs erreurs.

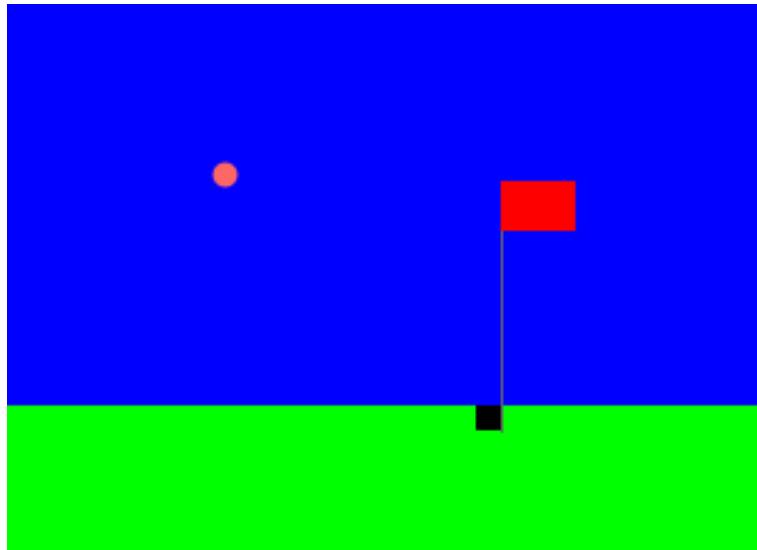


FIGURE 2 – Exemple de simulation physique d’élèves de L1. Tir d’une balle de golf avec un angle et une vitesse initial fixé par l’utilisateur

Après cette parenthèse sur l’enseignement, retournons à la recherche avec deux analyses spécifiques.

## 2.3 Deux analyses poussées sur la recherche

Nous verrons ici deux aspects de la recherche que j’ai choisi de détailler. D’une part la publication d’articles scientifiques et d’autre part le cas des doctorants.

### 2.3.1 Les articles scientifiques

Dans cette sous-partie, je vais essayer d’apporter un regard critique sur la publication d’articles scientifiques. Ces publications sont des éléments essentiels pour un chercheur, c’est une forme de reconnaissance de son travail. La publication d’un article dans une conférence est même nécessaire pour l’obtention d’une thèse. Le nombres de publication dans des grandes conférences est l’un des critères utilisés pour connaître l’efficacité d’un chercheur.

Historiquement, les documents scientifiques étaient contenus dans des livres. Avant l’invention de l’imprimerie, ces livres n’étaient pas beaucoup partagés et les scientifiques découvraient et redécouvriraient les mêmes choses en s’appuyant très peu sur des recherches récentes. Avec l’arrivée de l’imprimerie puis d’internet, beaucoup d’articles scientifiques sont devenus accessibles à tous. Il a fallu trouver comment garder les bons et rejeter les moins bons (ceux qui contiennent des erreurs ou ne sont pas honnêtes). Ce qui nous amène au système actuel. Je

tiens à préciser que ce qui suit ne concerne que les articles autour de l'informatique et des mathématiques. Des conférences ont été créées dans lesquelles on peut essayer de soumettre un article que l'on a écrit. Plusieurs autres chercheurs du même domaine (appelés reviewers, anglicisme assez étrange), lisent l'article et émettent un avis concernant sa publication. Pour des questions d'objectivité d'analyse, la plupart du temps, les reviewers ne connaissent pas l'auteur de l'article et inversement. C'est ce qu'on appelle le double-aveugle.

Le système en place me semble excellent et je n'ai pas vraiment mieux a proposer. Néanmoins, j'ai entendu quelques critiques. La première est que pour qu'un article soit publié, il faut avant tout qu'il soit bien écrit, le contenu scientifique passe en second plan. Ce qui peut d'une part être frustrant pour les personnes ne sachant pas bien rédiger et pas très pertinent du point de vu du progrès scientifique. Néanmoins un article avec de très belles découvertes mais incompréhensible n'est pas forcément intéressant pour la communauté. Un juste milieu semble nécessaire. Une autre critique est que les reviewers ne passent pas toujours beaucoup de temps sur les articles. Un article publié par Léo (doctorant au laboratoire) a reçu un avis très négatif par un des reviewers, car selon lui, il n'y avait pas de lien vers le code. Alors que le lien était dans l'article et tout à fait visible. Une solution, au moins pour régler le second problème serait peut-être de payer les chercheurs qui examinent les articles, ce qui permettrait qu'ils y consacrent plus de temps. Mais cela impliquerait probablement de payer pour la publication d'article, ce qui n'est pas forcément un bon point. Toujours en lien avec ce manque de temps accordé à l'analyse d'article, on peut parfois voir le cas inverse. C'est-à-dire un article qui est trompeur, erroné, mais qui est pourtant accepté. Fabien (le stagiaire de Léo) m'a montré un article qui trichait lors de la présentation des résultats. Certaines lignes dans le code servaient à faire : "si l'algorithme A est meilleur que le nôtre, utiliser A, sinon utiliser le nôtre". Les résultats étaient donc toujours meilleurs ou au moins équivalents à l'algorithme A... Évidemment, ce passage n'était pas expliqué dans l'article et les auteurs ont arrêté de répondre aux messages de Fabien lorsqu'il a posé cette question (alors qu'ils répondaient avant). Il manque clairement de la rigueur ou de l'honnêteté scientifique dans ce cas présent. Il faut tout de même garder en tête que ce genre de situation est marginal et que la plupart des publications sont tout à fait honnête et juste. Comment les articles ne s'écrivent pas tous seul, il me semble cohérent de parler maintenant des doctorants.

### 2.3.2 Les différentes thèses et les profils associés

Les doctorants ont un rôle très important dans les laboratoires. Ce sont globalement eux qui effectuent le travail de recherche. Les Maîtres de conférence, Professeurs... (que l'on va continuer à appeler chercheurs pour des questions de simplicité) sont principalement là pour donner des idées, des directions à creuser. Ils participent en revanche nettement plus lors de l'écriture des articles, chose qui semble être assez délicate. Dans tous les cas le doctorant a un rôle central à

jouer et c'est de lui dont nous allons parler ici.

Je distingue pour ma part 3 types de thèses. Tout d'abord il y a les thèses CIFRE, c'est-à-dire en partenariat entre une entreprise et un laboratoire ou une école. Cela permet d'obtenir des financements plus facilement, mais ce partenariat n'est pas toujours facile à gérer pour les doctorants. D'un côté, on a le laboratoire qui pousse à écrire des articles et à chercher de nouvelles manières de faire ou à trouver des résultats scientifiques. De l'autre, on a l'entreprise qui veut très souvent un algorithme qui fonctionne, des résultats plus concrète. Ce qui amène certains problèmes, comme Jordan (un thésard de l'équipe) prenait des jours de congé auprès de l'entreprise pour pouvoir venir au laboratoire pour écrire son article. Les avantages majeurs que je trouve aux thèses CIFRE sont le fait de travailler en groupe et donc moins seul mais aussi d'obtenir une expérience professionnelle en entreprise qui peut s'avérer très enrichissante. Ensuite on trouve les thèses dite académiques, qui peuvent avoir des financements non-publics, mais qui sont toujours gérées uniquement par un laboratoire. Je pense qu'on peut séparer ces thèses en deux catégories, d'une part les thèses qui ont un financement pour un objectif précis qui doit être étudié. C'est le cas de Damien Fourure ancien doctorant qui a créé l'architecture sur laquelle j'ai travaillé pendant mon stage et qui devait travailler sur ces sujets-là. D'autre part, il y a les thèses plus libres, qui ont moins de contrainte et d'objectifs, mais qui on parfois le défaut de se faire avec moins d'encadrement car les encadrants ne sont pas toujours des experts dans les domaines choisis.

Un autre sujet qui me semble pertinent concerne les différents profils des doctorants. Le machine learning étant à la frontière des mathématiques et de l'informatique, il y a des profils assez variés. Guillaume, par exemple, est un mathématicien. Nous avons à plusieurs reprises résolu des problèmes de mathématiques ensemble, parfois juste pour le plaisir, sans lien direct avec notre travail. Ce profil, assez peu orienté vers l'informatique lui a posé des problèmes lorsqu'il a dû écrire beaucoup de code pour faire tourner ses algorithmes. Il m'a affirmé qu'il valait mieux être un bon codeur pour faire une thèse ici car il est assez simple de demander l'aide sur la partie mathématique, alors que devenir un bon codeur s'apprend avec beaucoup de pratique et on peut rarement demander de l'aide à quelqu'un qui n'a pas écrit le programme. Ce qui nous amène au profil opposé, avec Léo et Kevin, qui ne jurent que par l'informatique. Ce fut notamment frappant au début du stage de Fabien (le stagiaire L3 de Léo), qui malgré son absence de connaissances dans le machine learning et ces 3 années scolaires de moins, a du rapidement expliquer à Léo différents problèmes mathématiques dans les articles.

Je tenté d'expliquer ma compréhension de deux sujets de la recherche qui m'ont semblé important, je vais maintenant parler d'activités pas directement reliées à mon stage mais qui participent à la vie du laboratoire et de l'équipe.

## 2.4 En dehors du laboratoire

Ici je vais parler de deux évènements un peu particuliers, l'un directement lié à mon travail. L'autre qui concerne plus la vie du laboratoire.

### 2.4.1 Conférence d'Apprentissage

Un évènement important durant mon stage fut de participer à la CAp (Conférence d'Apprentissage) à Rouen. Cette conférence est un point de repère pour tous les Français qui travaillent sur des méthodes d'apprentissages. Elle est légèrement différente des autres conférences, les chercheurs publient rarement uniquement à CAp, elle regroupe des papiers qui ont déjà été publiés ou qui ont été soumis à d'autres conférences plus reconnues. Il ne s'agit donc pas d'une conférence qui essaye de récupérer de bons articles, mais plutôt comme une manière de rassembler les Français qui travaillent sur les domaines d'apprentissage. Cette conférence a lieu chaque année et donc une bonne partie de l'équipe et moi-même sommes partis à Rouen pour ces trois jours de conférence.

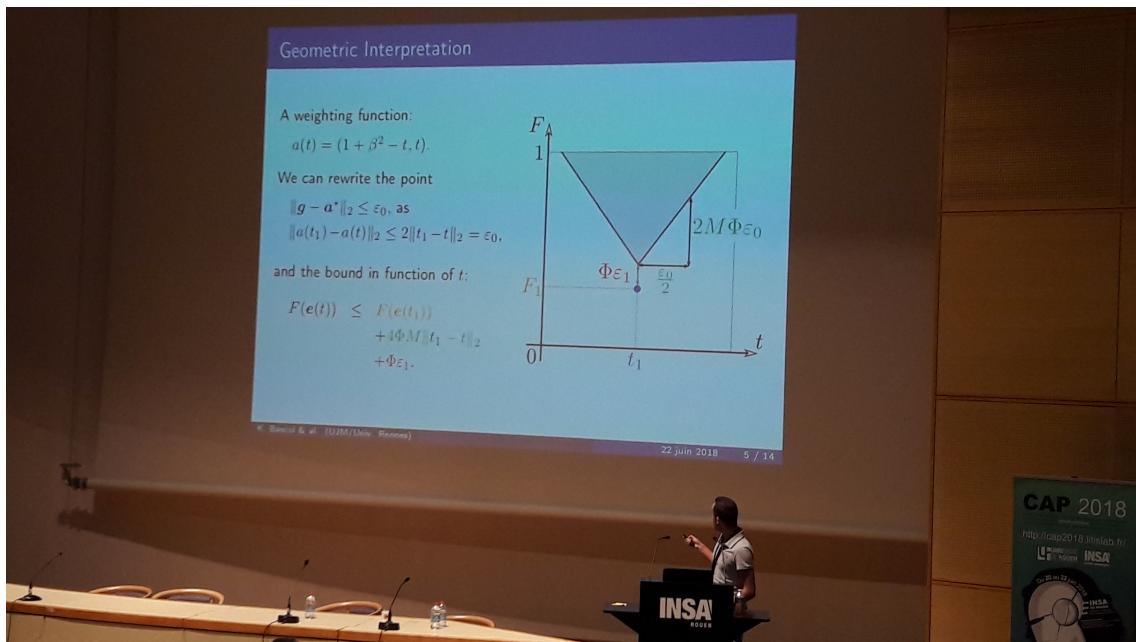


FIGURE 3 – Présentation de Guillaume l'un des doctorants du laboratoire.

Le principe d'une conférence c'est avant tout d'écouter des personnes présenter leurs recherches. La photo 3 nous décrit typiquement à quoi peut ressembler une présentation. La major partie de la journée était réservée à ces présentations. Pas facile de suivre tout ce qui est raconté, il est souvent question de domaine très pointu et je manque pas mal de connaissances générales sur ces sujets. Mais il semblerait qu'il devienne de plus en plus facile de comprendre les conférences. L'idée n'est pas forcément de tout comprendre, mais surtout d'avoir une vision de ce qui est fait et de creuser les domaines qui semblent intéressants. C'est pourquoi en fin de

journée, les personnes qui avaient présenté un article devaient désormais présenter un poster. Nous pouvions alors librement aller voir les sujets qui nous intéressaient et parler directement avec les auteurs. Cette période a été pour moi la plus intéressante et c'est certainement de ces échanges directs que je retiendrai le plus de choses. Je n'ai rien présenté durant cette conférence, je n'avais rien trouvé de suffisamment pertinent pour écrire un article.

La conférence ne s'arrête pas aux simples présentations scientifiques. Il y a énormément de rencontres et de team building durant cette petite semaine. Notamment avec les sorties dans les bars ou restaurants mais aussi les activités proposées par la conférence, une visite de la ville et un repas de gala au musée des beaux-arts de Rouen. Il faut tout de même noter que cet esprit d'équipe est moins importante qu'en entreprise, ce qui est important c'est surtout d'avoir une bonne ambiance. Il y a certes toujours des échanges sur les questions techniques, mais il y a moins d'obligations d'interactions qu'en entreprise, le travail est souvent plus individuel.

La photo 4 du groupe durant le gala au musée des beaux-arts. La photo a été évidemment transformée. Certains algorithmes capturent le style d'une image pour ensuite transformer une autre image et la rapprocher du style de l'image précédente. Ici, Rémi et Emmanuel ont appliqué d'une part le style du premier plan de la photo sur le tableau en arrière-plan, mais aussi le style du tableau sur le premier plan. C'est donc pour moi un exemple qui lie parfaitement le côté technique de la conférence au côté social.



FIGURE 4 – Photo de l'équipe transformée avec un changement de style artistique.

### 2.4.2 Tournoi de foot

Un autre évènement qui a eu lieu un peu après la conférence de Rouen, le tournoi de foot annuel. L'idée étant de faire jouer les différentes équipes du laboratoire les unes contre les autres. L'équipe a pris très au sérieux ce tournoi, en organisant deux entraînements avant. Finalement, ce tournoi se réduisit à un match vu le nombre de joueurs, les informaticiens contre les physiciens. Malgré le comptage des points approximatif, je dirais les physiciens ont gagné 8 à 7. J'aimerais pouvoir dire que cela rentre dans la catégorie team-building, mais je n'en suis pas tout à fait sûr. En effet, le foot ne rassemble pas énormément. Avec des niveaux de jeu et des conditions physiques très hétérogène, il y a des frustrations dans le jeu. De plus il y a une certaine opposition avec l'équipe d'en face, les physiciens, qui jouaient un peu agressif et pas très fair-play. Tout s'est évidemment bien passé, mais je ne pense pas que jouer au foot soit une activité qui lie particulièrement les gens ensemble. En parlant de foot, il me semble opportun de noter qu'il y a eu des projections des matchs de la France au laboratoire lors de la coupe du monde.

J'ai précédemment expliqué des points qui me semblaient intéressants concernant la vie en laboratoire, je vais maintenant rapidement développer ce que ce stage m'a apporté.

## 2.5 Apport du stage

Je vais commencer par les apports sur mes choix de carrière. Ce stage avait pour but de me lancer sur une thèse. On m'avait vivement conseillé de commencer par un stage de 6 mois pour voir si tout se passait bien et si j'avais vraiment envie de faire de la recherche pendant 3 ans. Ce qui est chose faite puisque j'ai été recruté pour une thèse intitulé "Machine Learning Theoretical Framework in a Learning to Learn context" qui s'inscrit dans le cadre du projet TADALOT (Anomaly Detection by Machine Learning using Atypical Losses and Transfer-learning). C'est probablement le plus grand apport de ce stage. Bien que ce stage m'a confirmé cette envie de faire de la recherche, il m'a fait changer d'avis sur le type de thèse. J'avais fait des demandes puis décliné deux stages (toujours dans le domaine du machine learning) car ils me semblaient trop théoriques. J'avais envie de développer un algorithme de Deep Learning, algorithme qui fasse quelque chose de très complexe avec une grosse base de données. Finalement, après l'expérience de ce stage, j'ai envie de me tourner vers des problèmes un peu plus théoriques.

Évidemment, les apports du stage ne s'arrêtent pas ici, j'ai beaucoup appris tout au long de ce stage sur différents domaines techniques. Tout d'abord la partie informatique, interagir avec un cluster à distance, lancer des programmes dessus... Je me sens nettement plus à l'aise dans mon environnement Linux. J'ai aussi appris comment coder un gros projet seul, j'estime que j'ai écrit approximativement 3 800 lignes (quelques duplicates et copie de code en ligne). J'ai

appris (à mes dépens) des règles essentielles pour structurer des gros projets (classes, fichiers différents, commentaires, nom de variable...). En mathématiques, j'ai eu aussi de gros rappels sur les bases qui sont, à mon avis, essentiels. Pas forcément énormément de nouveautés, mais des choses qu'il faut revoir pour ne pas les oublier (j'ai passé une bonne partie de ma deuxième semaine à comprendre ce qu'était que la dérivée sous contrainte). Enfin sur le domaine du machine learning, j'ai évidemment compris l'état de l'art du Deep Learning, mais aussi celui des domaines tels que le transfert learning (qui était au cœur de mon projet Cassiopée de deuxième année, sans que je le sache) ou encore le métrique learning. Ce qui m'amène justement sur la partie technique ce stage, avec comme fil conducteur, la segmentation d'une image.

### 3 Segmentation d'image avec un réseau de neurones : Grid-Net

Cette courte partie expliquera les bases nécessaires du machine learning (et mes notations) pour une meilleure compréhension de ce qui suit.

En machine learning on a des couples de données  $(X_1 \dots X_N, Y_1 \dots Y_N)$ . Le but étant de prédire  $Y_n$  (par exemple le nombre de voitures) en fonction de  $X_n$  (une image) à l'aide d'une fonction que l'on appellera  $f$ . On compare la prédiction  $f(X_n) = \hat{y}_n$  avec  $Y_n$  à l'aide d'une fonction de perte : on utilisera dans la suite la notation  $loss(f(X_n), Y_n)$ . Ensuite, dans le cadre des réseaux de neurones, la fonction  $f$  peut être décomposée en plusieurs fonctions, que l'on appelle couches :  $f(X_n) = (f_1 \circ f_2 \circ f_3 \circ \dots \circ f_l)(X_n)$ . Il faut donc trouver une bonne fonction  $f$  qui prédit correctement  $y$ .  $f$  dépend de paramètres que l'on notera  $\theta$ . L'objectif devient de trouver les bons paramètres  $\theta$  pour minimiser l'erreur 3.1.

$$\sum_{i=1}^n loss(f_\theta(X_i), Y_i) \tag{3.1}$$

On choisit les meilleurs paramètres  $\theta$  en regardant des exemples et en changeant ces paramètres via diverses méthodes, j'utiliserais uniquement la méthode de la descente de gradient. Un point important du machine learning est la généralisation, c'est-à-dire que sur les  $N$  couple, on va en utiliser que une partie pour apprendre les meilleurs paramètres  $\theta$  possible et une autre partie pour tester. Un bon algorithme de machine learning est un algorithme qui généralise bien à ces nouveaux exemples qui lui sont inconnu. On peut maintenant revenir à quelque chose de plus concret en commençant par une explication de la segmentation d'image.

### 3.1 La segmentation d'image

#### 3.1.1 CityScapes : un jeu de données

Je vais commencer par définir la segmentation d'images. Plus précisément, la segmentation sémantique multi-classes car il existe plusieurs types de segmentation. On part d'une image, dans notre cas une photo de ville et pour chaque pixel de l'image l'algorithme doit dire à quel objet il appartient. On a 19 classes qui sont par exemple : voiture, piéton, bus, vélo, arbre, route... Une image vaut mieux que mille mots, voici Figure 5 et 6 un exemple de segmentation parfaite. Chaque pixel de l'image a été attribué à une classe (ensuite transformée avec des couleurs pour la rendre plus compréhensible pour l'humain).



FIGURE 5 – Exemple de photo de la base de données Cityscapes [2]

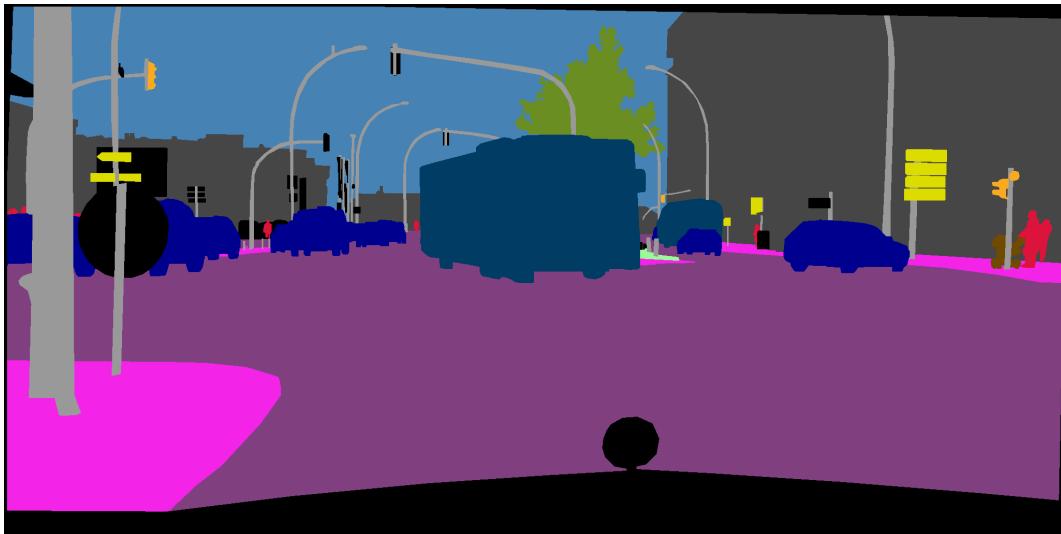


FIGURE 6 – Photo entièrement segmentée de la base de donnée Cityscapes [2]

On comprend que cela ne va pas être une tâche facile, les méthodes dites classiques d'analyse d'images vont être compliquées à mettre en oeuvre vu la diversité de ce qui se trouve sur l'image.

Il semble difficile de définir des règles pour la détection de chaque objet. L'approche qui va être choisie sera le Deep Learning, c'est une sous-méthode de l'apprentissage automatique. On va avoir accès à une énorme base de données nommée Cityscapes. Comme son nom l'indique, c'est une base de données avec des images de villes comme sur la Figure 5. Il y a 5 000 images de taille 1024 par 2048 (donc un peu plus de 10 milliards de pixels) qui sont annotées et mises à disposition du public. Annoté signifie qu'une segmentation a été faite par un humain, c'est ce qu'on considérera comme vrai. Il peut avoir quelques ambiguïtés, notamment entre les arbres et le ciel, mais toutes les images ont été annotées suivant les mêmes règles. Je vais utiliser 3 000 de ces images (6 milliards de pixels) pour entraîner mon réseau de neurones, puis 500 (1 milliard de pixels) pour voir s'il a bien réussi à généraliser le problème sur de nouvelles images. Les 1 500 restantes sont gardées par les créateurs du jeu de données pour évaluer correctement les différents algorithmes. Il y a énormément de détails sur ces images. Si on observe bien la partie gauche de l'image, on peut voir une tache rouge, qui correspond à une personne. Sur la photo initiale, cette personne est difficilement distinguable. Les zones en noir sur la segmentation ont été supprimées, c'est-à-dire que l'on ne regarde pas ces zones. Ceci a été fait pour deux raisons, d'une part, il est inutile de toujours prédire le devant de notre voiture, d'autre part certains objets sont en trop petite quantité et donc ont été supprimés pour réduire le nombre de classes. On voit notamment une poussette en marron à droite de l'image, qui sera elle aussi supprimée dans la suite.

Lorsque l'on voit ce jeu de données, on ne peut que faire le lien avec les voitures autonomes. Les photos sont prises d'une voiture et correspondent bien à la vision humaine de la route. On voit la proximité entre segmenter l'image et conduire une voiture. Ce n'est pas suffisant, mais déjà un bon début de reconnaître la position des autres voitures, des piétons... On pourrait probablement utiliser mon travail (ou d'autre du même genre) pour les voitures autonomes. Bien entendu les recherches présentées sont utilisables dans de nombreux autres contextes, la cartographie par satellite, la détection de tumeur chez un malade, la détection de missile... et bien d'autres encore.

On a donc fixé les idées et l'objectif à atteindre : l'algorithme prend en entrée une image et doit la ressortir segmentée. Pour savoir si on a bien réussi la segmentation, il faut un critère, et on va voir dans la partie suivante que ce critère n'est pas forcément évident à choisir.

### 3.1.2 Un critère de performance : $mIoU$

Je commence par préciser ce qui a été dit précédemment concernant les fonctions de pertes. Il faut bien distinguer les fonctions de coûts (ou critère de performance), qui sont utilisées pour tester si un algorithme est bon, des fonctions de pertes qui sont utilisées pour effectuer la descente de gradient. Les premières sont rarement dérivables, c'est pourquoi on est obligé de changer de fonctions pour faire la descente de gradient, on utilise les fonctions de pertes. Dans

cette partie nous allons nous intéresser au critère de performance utilisé comme référence par les créateurs de Cityscapes : la *mIoU*.

Je vais expliquer dans cette partie comment on compare deux segmentations, pour savoir si la segmentation prédictive est proche de la vraie segmentation. La méthode intuitive est appelée l'accuracy, elle consiste à comparer chaque pixel des deux images et de compter le nombre d'erreurs. On obtient donc un pourcentage de valeur correctement prédictives. Ce raisonnement n'est pas forcément pertinent pour des données déséquilibrées. Pour simplifier considérons que l'objectif soit de détecter un très petit objet sur une image, et que cet objet soit présent sur 1% de l'image. Un algorithme qui indiquerait la non présence de l'objet sur chaque pixel aurait 99% d'accuracy, ce qui est excellent alors que il ne fait absolument pas le travail attendu vu qu'il ne prédit jamais la position de l'objet ! Il faut donc trouver autre chose.

Le critère qui a été retenu est la moyenne des Intersection over Union (mIoU). Il est défini rigoureusement dans l'équation 3.2, en notant  $N$  le nombre de pixels sur une image,  $K$  le nombre de classes, pour tout  $k \in \llbracket 1, K \rrbracket$  et pour tout  $n \in \llbracket 1, N \rrbracket$   $Y_n^k \in \{0, 1\}$  qui indique si le pixel  $n$  appartient à la classe  $k$ . De même on définit pour tout  $k \in \llbracket 1, K \rrbracket$  et pour tout  $n \in \llbracket 1, N \rrbracket$   $\hat{Y}_n^k \in \{0, 1\}$  qui indique si l'algorithme pense que le pixel  $n$  appartient à la classe  $k$ .

$$mIoU = \frac{1}{k} \sum_{k=1}^K \frac{\sum_{n=1}^N Y_n^k \hat{Y}_n^k}{\sum_{n=1}^N \hat{Y}_n^k + \sum_{n=1}^N Y_n^k - \sum_{n=1}^N Y_n^k \hat{Y}_n^k} \quad (3.2)$$

On peut le définir plus intuitivement comme dans l'équation 3.3. Pour chaque classe on compte le nombre d'éléments prédicts comme étant de cette classe et étant bien de cette classe (Vrai Positif) et on le divise par la somme entre les erreurs (Faux Positif and Vrai Negatif) et le terme du numérateur. Les Faux Positif sont les pixels prédicts comme étant de cette classe alors qu'ils appartiennent à une autre classe. Les Vrai Negatif sont les pixels prédicts comme étant d'une autre classe alors qu'ils appartiennent à cette classe. Ce qui permet d'avoir une valeur entre 0 et 1 pour chaque classe. Si l'IoU d'une classe  $k$  vaut 1, alors c'est qu'il n'y a pas d'erreur et on a juste  $\frac{VraiPositif_k}{VraiPositif_k}$  et si l'IoU vaut 0 c'est qu'il n'y a eu aucune bonne prédiction  $VraiPositif_k = 0$ .

$$mIoU = \frac{1}{k} \sum_{k=1}^K \frac{VraiPositif_k}{FauxPositif_k + VraiNegatif_k + VraiPositif_k} \quad (3.3)$$

Ceci change complètement la donne par rapport à l'accuracy. En effet pour obtenir une bonne valeur de l'IoU il faut trouver les pixels de cette classe, sinon on obtient toujours 0. Néanmoins il ne faut pas en trouver trop, car sinon les erreurs seront importantes et donc l'IoU pas très élevé. La figure 7 est un exemple de matrice représentant les prédictions en fonction du vrai label pour les 19 classes présentes.

		Matrice de confusion																		
		Real class																		
		Prediction																		
road	63210	610	8	4	6	6		6	2	16		40	2	183	2	3	1	2	15	
sidewalk	472	5740	54	8	23	27		1	12	61		42	1	34		1	1	1	39	
building	4	68	25315	118	48	136	10	70	726	3	68	116	8	122	6	3	3		42	
wall	26	89	182	417	69	7		1	87	7		47		10					24	
fence	5	30	365	70	546	22		15	65	7		12	1	17					48	
pole	2	42	410	11	28	1309	10	20	181	5	7	36	1	33	1	1	1	1	24	
traffic light			62			9	155	2	31			1	1							
traffic sign	2	1	137		13	15	2	693	32			11	1	9	1				2	
vegetation	13	56	445	12	31	92	12	18	22620	278	54	40	5	59	1	2	1	2	15	
terrain	20	91	5	3	4	3			83	553		1	1	9					4	
sky			88			4	1	1	64		3521									
person	21	19	83	3	2	20		1	26			2315	52	87		1		1	39	
rider	1	1	9		1	1			6			75	245	10				3	30	
car	54	7	90	1	9	10		13	34	2		25	8	11256	33	9		13	28	
truck	217		36	1		2	1	3	8		4	3	1	188	316	2				
bus	8	1	30		4	7		1	13			1		114	67	448	58			
train	1		61	1		1			2					5	1	11	114			
motorcycle		1	4		1	1			3			12	13	21				71	29	
bicycle	3	13	34		4	7		1	18	2		28	45	23				13	998	

FIGURE 7 – Matrice de confusion avec notamment les *VraiPositif<sub>moto</sub>* (71), *VraiNegatif<sub>moto</sub>* ( $1+4+1+1+3+12+13+21+29 = 85$ ) et les *FauxPositif<sub>moto</sub>* ( $2+1+1+2+1+3+13+13 = 36$ ) associé à la détection de la classe moto. On a donc  $IoU_{moto} = \frac{71}{71+36+85} = 0.37$ . Les cases grisées n'ont aucune prédiction associée. Les prédictions sont heureusement principalement sur la diagonale. Les valeurs de la matrice correspondent au nombre de pixels sur une image.

Figure 7 on trouve une matrice de confusion associé à la prédiction de classe, la valeur dans la case  $(i, j)$  de la matrice de confusion correspondra aux nombres de pixels prédis comme étant de la classe  $i$  et étant de la classe  $j$  d'après la vérité terrain.

Ce critère semble donc assez pertinent, je reviendrai sur cette question lorsqu'on parlera des différents résultats obtenus.

## 3.2 Une architecture : GridNet

GridNet est une architecture de réseau de neurones convolutifs qui a été inventée par Damien Fourure et Rémi Emonet. Ce réseau est adapté à la segmentation sémantique. Je vais tout d'abord vous expliquer l'intérêt de GridNet puis je parlerai de ce que j'ai fait dans mon stage. Cette partie n'a pas vocation à définir rigoureusement ce qu'est un réseau convolutif, mais donnera seulement des intuitions.

### 3.2.1 GridNet généralise les réseaux convolutifs classiques

Commençons par une rapide explication des réseaux convolutifs, ces types de réseaux traillent sur des zones de l'image (typiquement  $3 \text{ pixels} \times 3 \text{ pixels}$ ). A chaque couche, on a la forme de l'image qui est préservée avec des calculs locaux (on garde la structure spatiale de l'image) : les pixels en haut à gauche de l'image d'entrée n'influenceront que les pixels en haut à gauche de la sortie (Figure 8).

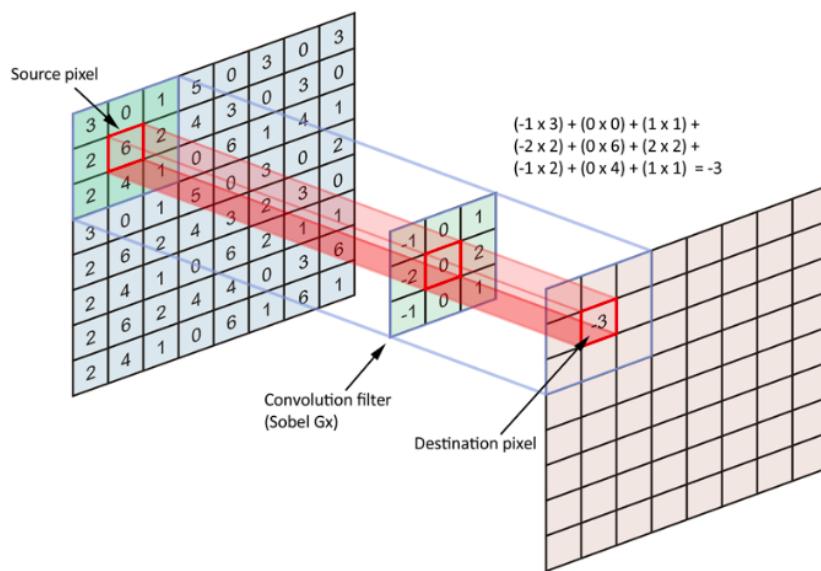


FIGURE 8 – Un exemple de convolution. Le filtre de convolution (de taille  $3 \times 3$  ici) étant le même pour tous les pixels de cette couche, on réduit grandement le nombre de paramètres par rapport à une architecture complètement connectée. Ce sont les paramètres de ce filtre de convolution qui vont être modifiés pour répondre au problème. Image prise sur le site <http://openresearch.ai/t/network-in-network/39>

Il est important de remarquer qu'en jouant avec les paramètres de la couche de convolution, on peut réduire ou augmenter la taille de l'image de sortie par rapport à celle d'entrée. Par exemple en décalant le filtre de convolution de deux pixels par deux pixels au lieu de un par un. La Figure 9 nous explique ce phénomène. La taille de l'image est progressivement réduite puis ré-augmentée pour atteindre la taille de l'image initiale. Ceci permet d'obtenir des informations générales concernant la situation. L'information est compressée dans les couches du milieu, on

espère qu'elles contiennent les informations importantes de l'image.

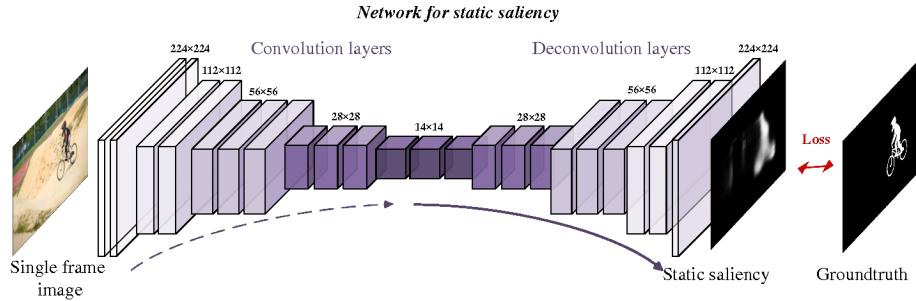


FIGURE 9 – Un réseau classique complètement convolutif. A gauche l'image d'entrée, puis la réduction de dimension au milieu et enfin la restitution de l'image qui a été segmentée. On compare ensuite cette image à la vérité terrain (Groundtruth en anglais). Image prise sur l'article <https://perso.univ-st-etienne.fr/fod07375/pdf/gridNetwork.pdf> [1]

Passons maintenant au détail de GridNet. La Figure 10 représente en détail la structure des 3 couches utilisées dans ce réseau. Les couches vertes ne modifient pas la taille de l'image, elles la transforment juste. La couche rose diminue la taille de l'image par deux tandis que la jaune multiplie cette taille de l'image par deux. On espère que ce chemin (diminution et augmentation) captera les informations globales sur l'image tandis que le chemin par une couche verte gardera les détails de l'image.

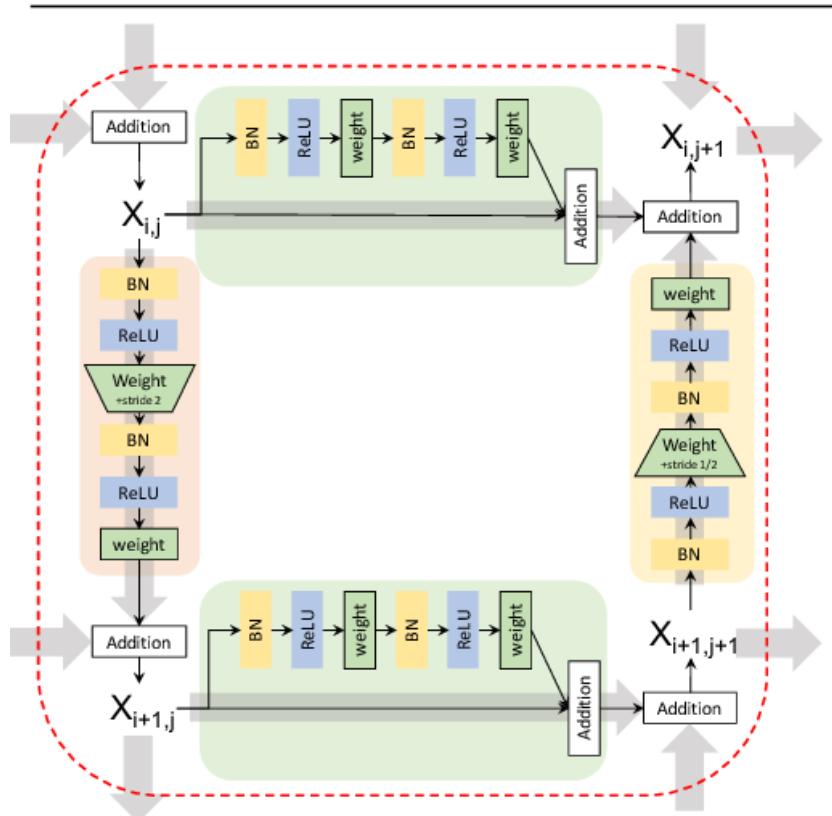


FIGURE 10 – Sous réseau de GridNet. Chaque couche est composée de deux normalisations, de deux fonctions d'activations ReLU et de deux convolutions. Image prise dans l'article <https://perso.univ-st-etienne.fr/fod07375/pdf/gridNetwork.pdf> [1]

Pour donner un exemple concret en lien avec le jeu de données Cityscapes, on peut imaginer que la couche du bas comprenne que la photo concerne une image de bus et pas de voiture. La couche du haut n'aura pas de compréhension abstraite et arrivera juste à différentier les objets les uns des autres. La fusion de ces deux informations va permettre une définir le bon objet, mais aussi d'avoir des contours précis. Cette vision est très grossière et très simplifiée, mais donne l'idée derrière cette réduction de dimension.

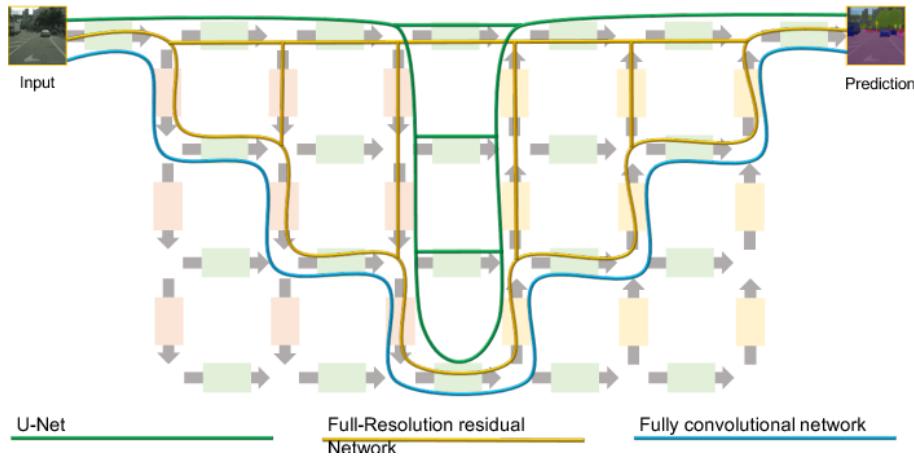


FIGURE 11 – Représentation de GridNet avec les différents réseaux convolutif qu'il généralise. Image prise dans l'article <https://perso.univ-st-etienne.fr/fod07375/pdf/gridNetwork.pdf> [1]

GridNet va étendre ce principe, en diminuant la taille de l'image plusieurs fois et en augmentant le nombre de convolutions qui ne modifient pas la taille de l'image. Figure 11 on trouve une architecture possible de GridNet. En suivant le code couleur et les flèches on remarque que toutes les images sur une même ligne sont de même dimension.

L'apport supplémentaire de la Figure 11 est la comparaison avec d'autres réseaux classiques. Par exemple le réseau **Fully convolutional network** est celui représenté précédemment Figure 9. Les deux autres réseaux sont aussi très classiques pour de la segmentation sémantique. On peut voir GridNet comme une généralisation de différents réseaux. On peut noter que pour des raisons de limitation de mémoire et de temps de calcul, on ne peut généraliser que les petits réseaux **U-net** ou autres. Pour un **Fully convolutional network** avec  $n$  couches, on a approximativement besoin de  $\frac{n}{2} \times \frac{n-2}{2}$  couches pour créer un réseau GridNet équivalent, ce qui devient très vite déraisonnable quand  $n$  devient grand.

### 3.2.2 Code de GridNet en Python

J'ai donc codé GridNet avec le package Pytorch. Ce package spécialement développé pour créer des architectures de Deep Learning. Ce code avait déjà été implémenté en Lua par Damien Fourure. Mais je n'ai pas simplement transformé un code d'un langage à un autre, et ceci pour plusieurs raisons. Tout d'abord la façon de structurer le code en Lua m'a semblé assez différente

de celle de Python. Ensuite le code en Lua étant peu commenté il m'était assez difficile de tout comprendre (il était toutefois bien organisé). Je me suis assez vite rendu compte qu'à partir du moment où j'avais compris la structure, je pouvais coder l'algorithme sans prendre appui sur le code en Lua. C'est ce que j'ai fait et je me suis permis de prendre quelques libertés. J'ai changé légèrement l'architecture des couches qui me semblait étrange et quelques paramètres. Je ne savais pas si quelqu'un allait reprendre mon code un jour, néanmoins j'ai essayé de clarifier au maximum en commentant la plupart des lignes et surtout en indiquant le rôle de chaque fonction. J'ai dû quelques fois faire de l'orienter objet, mais cela n'est pas toujours très intuitif ni intéressant pour les réseaux de neurones.

J'ai assez vite eu des premiers résultats. Mais ils n'étaient pas très bons (0.4 de  $mIoU$  contre 0.7 pour le meilleur résultat de Damien). J'ai essayé de modifier des choses pour ré-atteindre les résultats précédents, sans succès. Il y a notamment eu de nombreux essais avec une sorte de pondération des classes (j'en reparlerai dans la partie suivante). J'ai peu à peu regretté de ne pas avoir suivi complètement le code en Lua. Le principal problème pour débugger ce genre d'algorithme est qu'il faut attendre 4 jours pour obtenir un résultat même après une légère modification ! Finalement après plusieurs semaines d'essais infructueux et un désespoir grandissant, mon encadrant Rémi Emonet a regardé mon code, en un quart d'heure la question était réglée. J'avais confondu deux paramètres, qui n'ont rien à voir. J'avais mis un paramètre qui pénalisait les poids de mes couches de convolution et les forçait à être très petit. Cette régularisation n'étant pas adaptée, j'ai réussi à obtenir presque 0.7 de  $mIoU$  en l'enlevant.

J'ai découvert dans ce stage comment lancer des programmes sur un serveur. Se connecter à distance en ssh, utiliser un environnement virtuel sur le serveur et enfin faire tourner mon code. La gestion de la puissance de calcul est peut-être très basique, mais très intéressante. Le programme lancé va se mettre dans une file d'attente et à chaque fois qu'une machine de calcul se libère, c'est le premier programme de la file d'attente qui se lance. La parallélisation des calculs peut aussi se faire sur le serveur : il suffit de demander plusieurs machines et faire quelques modifications dans le code. Au lieu d'apprendre sur 4 images à la fois, on peut apprendre sur 16 images si 4 machines sont libres.

### 3.2.3 Quelques résultats et analyses

Cette partie comportera beaucoup d'exemples de segmentation. L'organisation sera toujours la même, la photo prise initialement, la segmentation avec GridNet puis la segmentation optimale (vérité terrain) et quelques commentaires sur ces trois images. Il faut garder en tête que les images que je vais présenter sont très bien réussies. Ces images ont été prises soit dans l'ensemble de validation soit dans l'ensemble d'apprentissage. Enfin elles ont toutes été faites avec une fonction de perte particulière : la cross entropy (ce point deviendra important lors de la partie suivante).

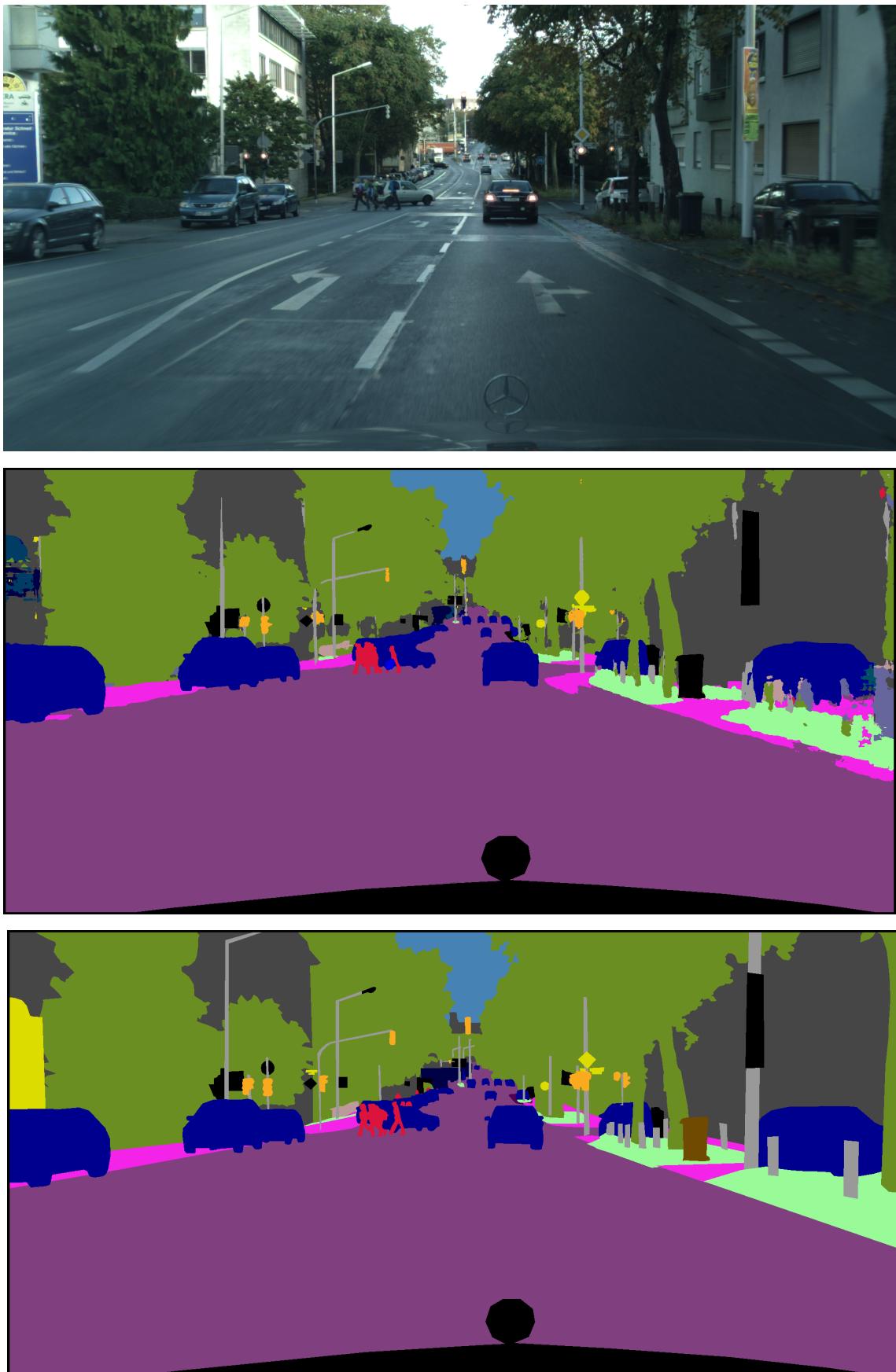


FIGURE 12 – Une photo de Darmstadt en Allemagne, la segmentation avec GridNet et la vérité terrain. Base de données d'entraînement.

Le premier exemple Figure 12 est particulièrement bien segmentée. Il y a de nombreuses remarques à faire sur cette première image. Tout d'abord on remarque que globalement l'image est plus brouillon sur la segmentation de GridNet, tout est moins uniforme, en particulier sur la droite de l'image. Ceci est encore plus marqué pour les segmentations pas très réussies. Cette non précision sur les contours fait perdre un peu de performance mais pas énormément. Si on regarde au contraire à gauche de l'image, on observe un panneau d'information qui est coupé. Il est censé être segmenté en jaune. C'est souvent en bord d'image que GridNet fait de grosses erreurs de segmentation, en effet il n'y a pas de contexte à gauche il est donc difficile de comprendre cette zone. On voit bien que GridNet n'est pas très sûr de lui, il prédit un peu de bus, de camion, de végétation, de bâtiment et quelques traits jaunes qui correspondent aux panneaux d'informations. Ici c'est une erreur dramatique au regard de la fonction de coût ( $IoU_{panneau} = \frac{VraiPositif}{FauxPositif + VraiNegatif + VraiPositif}$ ). Les autres éléments de signalisation sont très petits par rapport à la pancarte à gauche. Ils sont globalement bien détectés mais cela donnera tout de même une  $IoU$  très faible pour cette classe. On va avoir très peu de *VraiPositif* et beaucoup de *VraiNegatif*, ce qui va donner une  $IoU$  proche de 0 pour cette classe là. Un dernier commentaire que l'on peut faire sur cette série d'images est la bonne segmentation même sur des objets très distants. Les voitures sont notamment très bien détectées dans le fond de l'image. Ceci est en parti dû à un traitement des images. Les images doivent être de taille 400 pixels par 400 pixels, parfois on découpe juste l'image avec les bonnes dimensions. Mais on découpe aussi l'image en disons 800 pixels par 800 pixels puis on diminue sa qualité pour obtenir 400 pixels par 400 pixels. On obtient donc des objets qui sont 4 fois plus petits qu'avec la première méthode. Cela permet d'augmenter la diversité du jeu de données mais aussi de permettre une compréhension moins dépendantes des dimensions des objets.

Regardons cette deuxième série d'images 13. Encore une fois très peu d'erreurs ici. Les parties droite et centrale sont très bien segmentées. En revanche il y a de nombreuses imprécisions à gauche. La voiture est déformée (ceci est dû aux vitres, on en reparlera dans le prochain exemple), les personnes aussi, le trottoir n'est pas parfait. Mais le plus étonnant est cette grosse tâche verte en haut à gauche, alors qu'il n'y a qu'un bâtiment pour la vérité terrain. Si on regarde la vraie image, on se rend compte qu'il y a bien un arbre à cet endroit là. GridNet ne s'est donc pas trompé ! C'est la "vérité" terrain qui est fausse. Cela pose la question de savoir comment ces objets ont été segmentés, normalement par un humain, mais il semble très étrange qu'un humain ait pu ne pas voir cet arbre... Cela peut être dû à des consignes mal expliquées. La raison reste obscure, mais il ne faut pas faire une confiance aveugle en la vérité terrain. Deux problèmes apparaissent avec ce type d'erreur. La première possibilité est que l'image soit dans la base d'entraînement, dans ce cas on poussera l'algorithme à reconnaître des bâtiments lorsqu'il y a des arbres, ce qui est assez problématique. L'autre possibilité est que cette image soit dans la base de test, dans ce cas l' $IoU$  de la végétation et des bâtiments vont baisser car il y a des soi-disant "fausses" prédictions. Il ne faut pas espérer atteindre une  $mIoU$  de 1, si l'algorithme arrive un jour à avoir 0.95 (ou peut-être 0.99) il sera plus performant que l'humain

et donc il ne sera plus pertinent de l'évaluer sur un jeu de données moins bon que lui. On peut aussi envisager qu'aucun algorithme n'atteigne de telles performances justement à cause des incohérences dans le jeu de données. Néanmoins, je tiens à préciser que c'est le seul exemple flagrant que j'ai trouvé, les autres images que j'ai regardé sont tout à fait cohérentes.

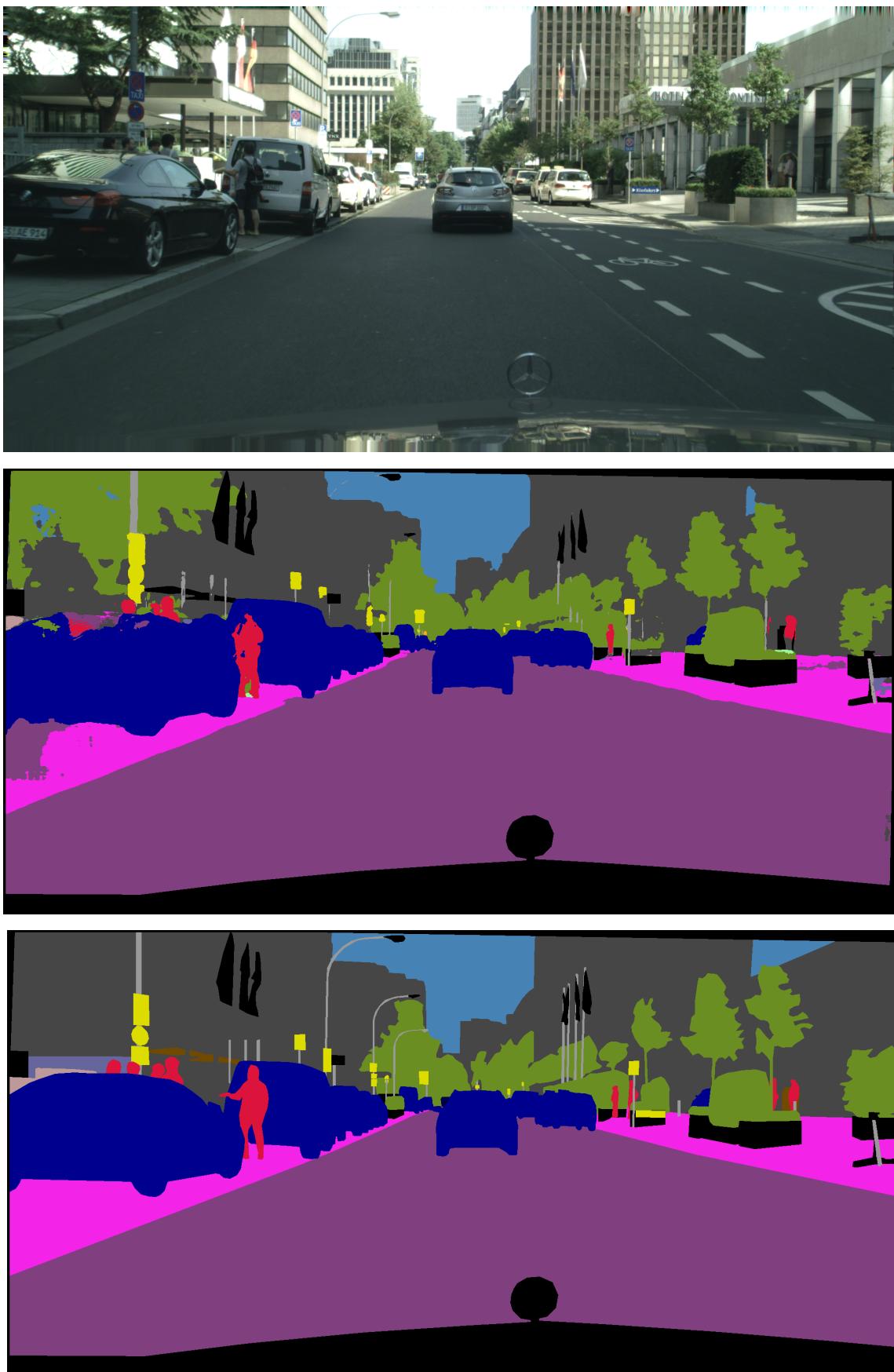


FIGURE 13 – Une photo de Frankfurt en Allemagne, la segmentation avec GridNet et la vérité terrain. Base de données de test.

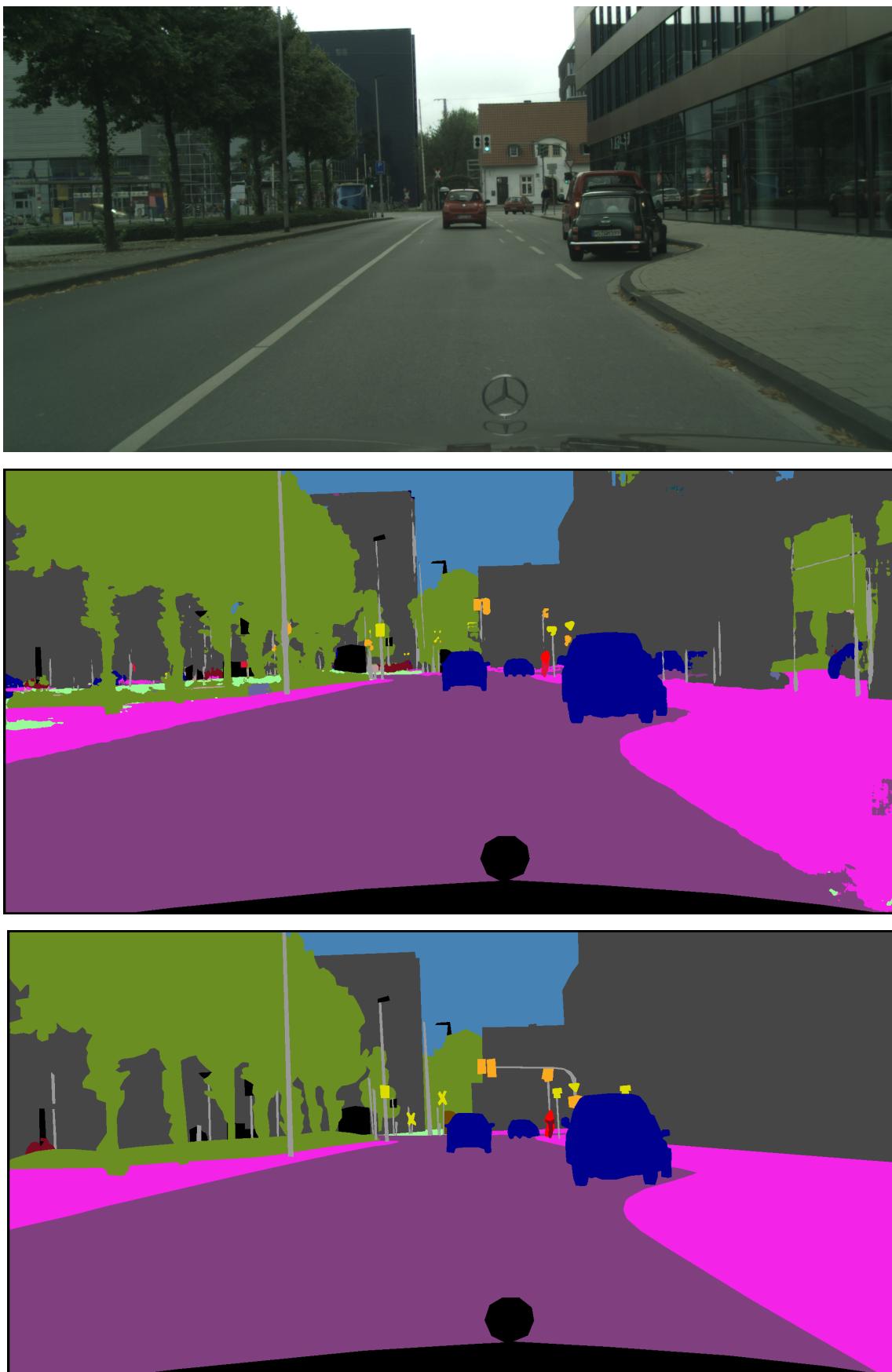


FIGURE 14 – Une photo de Münster en Allemagne, la segmentation avec GridNet et la vérité terrain. Base de données d'entraînement.

De nouveau comparons les deux segmentations, notamment la partie droite. La vérité terrain indique qu'il y a uniquement un mur, tandis que GridNet prédit des arbres et une voiture. Si on observe la vraie image on se rend compte qu'il y a des vitres, avec évidemment le reflet d'une voiture et d'un arbre. Les vitres sont très mal gérées par GridNet, il distingue le reflet d'autres objets dans la vitre. Ce cas me semble différent de l'exemple précédent, ici il n'y a pas d'erreurs de la vérité terrain selon moi. Si on voit ici un algorithme qui permettrait de conduire des voitures autonomes, définir des vitres comme des murs me semble pertinent. Mais il reste intéressant de remarquer que l'algorithme ne comprend pas le concept de vitre et donc segmente comme s'il n'y avait pas de réflexion.

Cette analyse des résultats nous a permis de comprendre un peu plus l'objectif de ce type d'algorithme et cela peut aussi donner des idées d'améliorations. Pour moi la principale idée qui ressort est qu'il faut quelque chose de moins brouillon, faire des limites plus nettes et moins mélanger les classes. Un article expliquait l'intérêt de rajouter un coût à chaque fois qu'un pixel d'une classe n'était pas entouré de la même classe. C'est une direction que j'aurais pu prendre, mais je me suis tourné vers autre chose. Ce qui m'amène justement aux avancées de mon stage sur le plan théorique et pratique.

### 3.3 Les avancées théoriques de ce stage

Les directions à prendre pour améliorer ou modifier GridNet ont été multiples. L'architecture aurait pu s'adapter au multi-view (plusieurs données d'entrée), au multi-output (plusieurs objectifs à la fois). J'aurais pu essayer sur d'autres jeux de données, et pourquoi pas faire des compétitions. Je me suis finalement intéressé principalement aux fonctions de perte utilisées pour apprendre au réseau de neurones. Plus simplement, quelle est la fonction que l'on minimise. La partie qui va suivre est de loin la plus technique et elle me servira aussi personnellement pour clarifier mes idées sur cette partie de mon stage.

#### 3.3.1 Fonction de perte : cross entropy, fonction de coût : $mIoU$

La question que je me suis posé lors de mon stage est la suivante : pourquoi minimiser la cross entropy (Equation 3.5) lors de l'apprentissage du réseau de neurones alors que les performances finales sont calculées sur l'IoU (Equation 3.4). J'utiliserais les notations précédentes que je complète ici. On notera  $X_n = (X_n^1, \dots, X_n^K) \in \mathbb{R}^K$  la sortie du réseau de neurones pour tout élément  $n$ . Mais aussi  $\underline{X}_n = (\underline{X}_n^1, \dots, \underline{X}_n^K) = (\frac{e^{X_n^k}}{\sum_{j=1}^K e^{X_n^j}})_{k \in [1, K]} \in [0, 1]^K$  qui correspond à une sorte de normalisation permettant d'interpréter les résultats en tant que probabilité d'appartenance à chaque classe. Cette fonction est appelée *softmax*. On peut ensuite créer  $\hat{Y} = (\hat{Y}_n^1, \dots, \hat{Y}_n^K)$  avec  $\hat{Y}_n^k = 1$  si  $k = argmax_{j \in [1, n]}(X_n^j)$  et 0 sinon.  $\hat{Y}$  est donc la prédiction. On notera le  $-$  devant les deux équations (3.4, 3.5) ce qui permet de parler de minimisation de l'erreur au lieu

de maximisation du gain.

$$mIoU(Y, \hat{Y}) = -\frac{1}{k} \sum_{k=1}^K \frac{\sum_{n=1}^N Y_n^k \hat{Y}_n^k}{\sum_{n=1}^N \hat{Y}_n^k + \sum_{n=1}^N Y_n^k - \sum_{n=1}^N Y_n^k \hat{Y}_n^k} = -\frac{1}{k} \sum_{k=1}^K \frac{I_k(Y, \hat{Y})}{U_k(Y, \hat{Y})} \quad (3.4)$$

$$crossEntropy(Y, \underline{X}) = -\frac{1}{N} \sum_{n=1}^N \sum_{k=1}^K Y_n^k \log(\underline{X}_n^k) \quad (3.5)$$

Pas évident de voir les similitudes à priori, on peut néanmoins noter un point commun très important, le minimum est atteint au même point, si et seulement si  $\underline{X} = \hat{Y} = Y$ .

Parlons un peu de la fonction de cross entropy, et pourquoi elle est utilisée classiquement lors de la segmentation d'image. Tout d'abord notons que c'est une fonction convexe en fonction de  $\underline{X}$  (à  $Y$  fixé). Ensuite, elle est très proche de l'accuracy dont on a parlé précédemment, il suffirait juste d'enlever le logarithme (et le  $-$  devant l'expression). L'ajout de ce logarithme va justement permettre de pénaliser fortement les erreurs graves et au contraire de faiblement pénaliser lorsqu'il y a peu d'erreur de prédiction. Si la bonne classe est  $k$  pour un pixel  $n$  et que  $\underline{X}_n^k \rightarrow 0$  alors  $crossEntropy \rightarrow +\infty$ . A l'inverse  $\log(1) = 0$ . Enfin, on peut remarquer que la *crossEntropy* est séparable, c'est-à-dire que la *crossEntropy* pour un pixel ne dépend pas des autres pixels, on moyenne simplement la *crossEntropy* pour chaque pixel. Avec la *mIoU* on ne peut en revanche pas regarder l'erreur pixel par pixel. Ceci va poser un premier problème lorsque l'on va essayer de minimiser directement la *mIoU* au lieu d'utiliser la *crossEntropy*.

### 3.3.2 Première relaxation continue du problème

Il m'a semblé opportun d'utiliser les termes "relaxation continue" car on veut initialement minimiser un système discret, mais on le passe sous forme continue pour pouvoir le résoudre (dérivé et donc descente de gradient impossible dans le cas discret).

Pour minimiser directement la *mIoU* il faut commencer par passer du discret au continu pour pouvoir dériver cette fonction de perte, c'est-à-dire arrêter d'utiliser  $\hat{Y} \in \{0, 1\}^{N \times K}$  mais au contraire utiliser  $\underline{X} \in \llbracket 0, 1 \rrbracket^{N \times K}$ . Il est important que lorsque  $\underline{X} \in \{0, 1\}^{N \times K}$  les deux fonctions aient la même valeur. Ce problème est bien posé mais possède énormément de solutions, on cherche une fonction définie sur  $\llbracket 0, 1 \rrbracket^N$  et on impose uniquement la valeur de  $2^N$  points et la dérivalibilité. Mais on aimerait en plus une fonction qui fonctionne bien avec une descente de gradient, par exemple une fonction convexe. On va commencer simplement par créer *mIoU<sub>c</sub>* (c pour continue) qui découle naturellement de *mIoU*, en gardant la même fonction et en remplaçant  $\hat{Y}$  par  $\underline{X}$  (Equation 3.6).

$$mIoU(Y, \underline{X}) = -\frac{1}{k} \sum_{k=1}^K \frac{\sum_{n=1}^N Y_n^k \underline{X}_n^k}{\sum_{n=1}^N \underline{X}_n^k + \sum_{n=1}^N Y_n^k - \sum_{n=1}^N Y_n^k \underline{X}_n^k} = -\frac{1}{k} \sum_{k=1}^K \frac{I_k(Y, \underline{X})}{U_k(Y, \underline{X})} \quad (3.6)$$

On a donc trouvé une fonction qui nous convient, elle est égale à la  $mIoU$  quand  $\underline{X} \in \{0, 1\}^{N \times K}$  et dérivable. En revanche elle a un problème (que l'on retrouvera pour toutes les fonctions suivantes), elle n'est pas séparable. En pratique cela pose des problèmes, en effet  $N$  vaut parfois des valeurs assez faibles lorsque l'on calcule la  $mIoU_c$  :  $400 \times 400$  (taille de l'image)  $\times 4$  (nombre d'images). Alors que si on avait calculer avec toute la base de données d'exemples (au lieu de 4 images), on aurait  $1024 \times 2048$  (taille totale des images)  $\times 3\,000$  (nombre total d'images). Le gradient associé à un pixel va être influencé par les autres pixels de l'image, mais aussi par les autres pixels des autres images. Cela peut poser problème notamment si par malchance une classe est très peu présente sur les 4 images choisies, le gradient sera particulièrement fort alors que les erreurs sur cette classe ne sont peut-être pas très graves si on regarde sur tout le jeu de données. Ce qui m'amène justement à parler des gradients, et de la différence entre ceux de la *crossEntropy* et de la  $mIoU_c$ . Il me paraît souvent intéressant de comparer les valeurs des gradients, car c'est cela qui compte vraiment pour modifier les paramètres  $\theta$  du réseau correctement. On fixe  $n \in \llbracket 1, N \rrbracket$  un pixel et  $k \in \llbracket 1, K \rrbracket$  une classe, le gradient de la  $mIoU_c$  est celui de l'équation 3.7 presque sûrement.

$$\frac{\partial mIoU_c}{\partial \underline{X}_n^k}(Y, \underline{X}) = \begin{cases} -\frac{1}{K} \frac{1}{U_k(Y, \underline{X})} & \text{si } Y_n^k = 1 \\ \frac{1}{K} \frac{I_k(Y, \underline{X})}{(U_k(Y, \underline{X}))^2} & \text{si } Y_n^k = 0 \end{cases} \quad (3.7)$$

$$\frac{\partial crossEntropy}{\partial \underline{X}_n^k}(Y, \underline{X}) = \begin{cases} -\frac{1}{N} \frac{1}{\underline{X}_n^k} & \text{si } Y_n^k = 1 \\ 0 & \text{si } Y_n^k = 0 \end{cases} \quad (3.8)$$

J'ai décidé de ne montrer ici que le gradient par rapport à  $\underline{X}$  sans aller plus loin, on aurait pu regarder par rapport à  $X$ . La règle des chaînes permet de voir le gradient des paramètres du réseau de neurones comme une multiplication de dérivée partielle, dont la dernière serait celle correspondant à la fonction de perte. Il est donc pertinent d'étudier ces deux fonctions car ce sont les seules qui vont changer. Néanmoins l'étude n'est pas complète, on pourrait notamment penser à la fonction *softmax* qui force chaque valeur à être entre 0 et 1 et qui influence énormément les gradients.

La *crossEntropy* n'influence que la probabilité de la classe réelle du pixel. La dérivée est inversement proportionnelle en  $\underline{X}_n^k$ , plus il y a d'erreurs dans la prédiction, plus la dérivée sera forte. En revanche la  $mIoU_c$  influence toutes les classes pour tous les pixels. De plus ce gradient est à peu près indépendant de  $\underline{X}_n^k$ , ce pixel  $n$  n'a pas plus d'influence que les autres et c'est le même pour toute la classe  $k$ .

J'ai codé cette fonction de perte et appris des nouveaux paramètres  $\theta$  pour mon réseau Grid-Net. Les résultats sont sensiblement les mêmes qu'avec la *crossEntropy*, globalement un  $mIoU$  un peu au-dessus de 0.65. L'écart entre l'ensemble d'apprentissage et de test est légèrement plus faible (le réseau généralise mieux). Mais le changement n'est pas très significatif.

On voit sur la Figure 15 un exemple de segmentation qui montre les quelques différences d'approche entre les deux fonctions de pertes. Ici avec la  $mIoU_c$  on a une grande précision sur certaines zones, tel que les feux, par contre on a des résultats vraiment très brouillon en haut à gauche. Mais ceci n'est vraiment pas très grave pour l' $IoU$ , car les classes ciel, bâtiment ou trottoir sont très présentes donc quelques pixels en moins ne sont vraiment pas grave. Les deux segmentations restent tout de même assez proches. On peut de nouveau noter un panneau jaune qui a été segmenter par les deux fonctions de perte, mais pas par la vérité terrain, encore une fois c'est une erreur de la base de données, le panneau est très clairement présent.

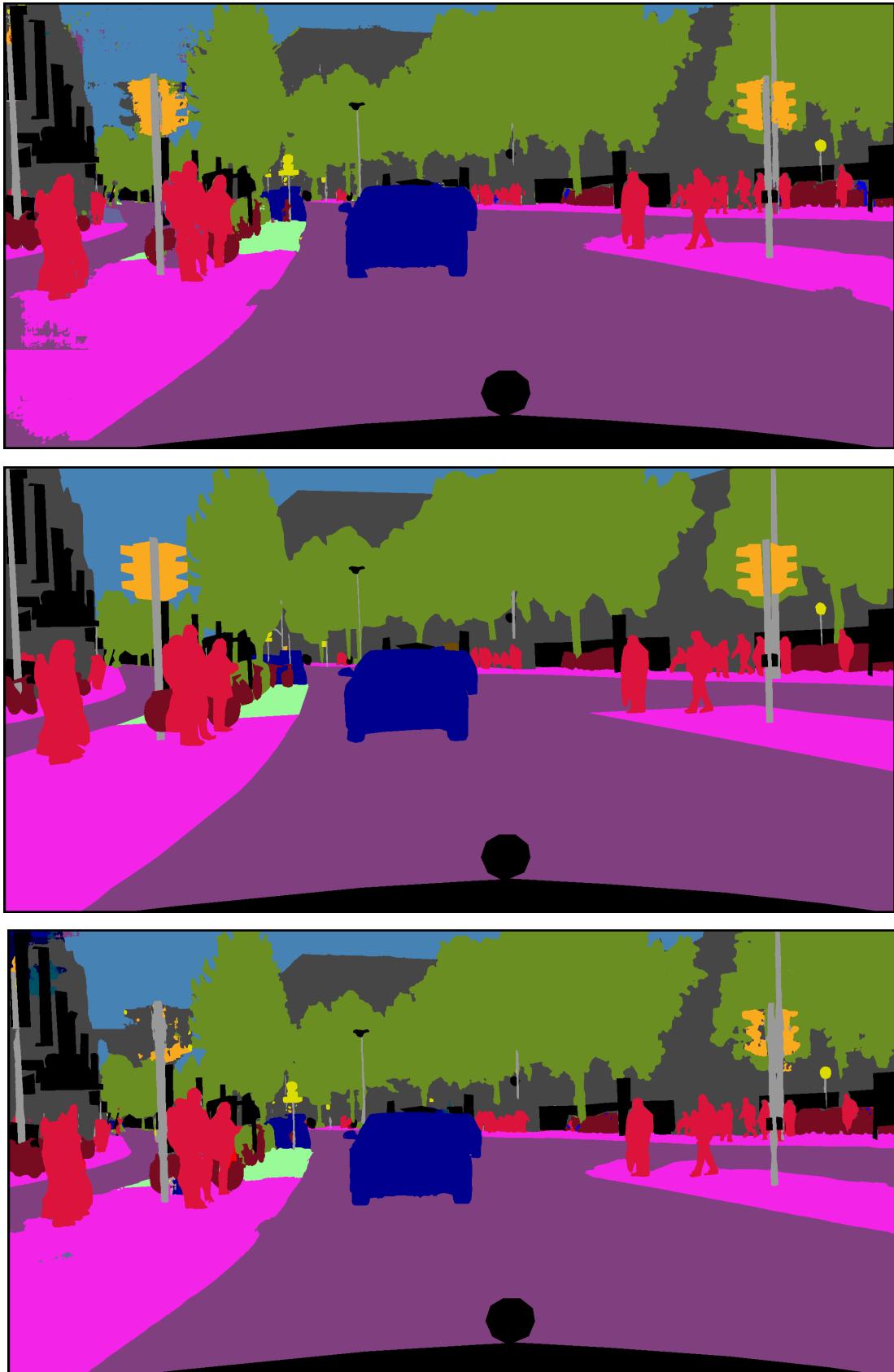


FIGURE 15 – Une segmentation de Munster en Allemagne avec la *crossEntropy*, la vrai segmentation la segmentation avec GridNet ( $mIoU_c$ ). Base de données d'entraînement de test.

### 3.3.3 L'extension de Lovasz

Durant mon stage, un article est paru qui essayait lui aussi d'approximer la  $mIoU$  pour en faire une fonction de perte [3]. L'idée était d'utiliser l'extension de Lovasz. Dans ce qui va suivre, les notations vont devenir un peu lourdes c'est pourquoi nous allons nous restreindre à l'étude de la  $mIoU$  à deux classes, donc la simple  $IoU$ . Ceci peut être immédiatement généralisable car la  $mIoU$  est une simple moyenne des  $IoU$  pour chaque classe. Pour saisir d'où vient cette extension, on définit ce qu'est une fonction submodular.

**Definition 3.1.** On note  $f : 2^{\{1, \dots, N\}} \rightarrow \mathbb{R}$  avec  $N \in \mathbb{N}$ .  $f$  est submodular si et seulement si  $\forall (S, T) \in (2^{\{1, \dots, N\}})^2$  avec  $S \subseteq T$  et  $\forall i \in \llbracket 1, N \rrbracket$ .

$$f(T \cup \{i\}) - f(T) \leq f(S \cup \{i\}) - f(S)$$

**Definition 3.2.** On note  $\forall i \in \llbracket 0, N \rrbracket$   $E_i = \begin{pmatrix} 1 \\ \vdots \\ 1 \leftarrow i \\ 0 \\ \vdots \\ 0 \end{pmatrix}$ .  $\forall \underline{X} \in \mathbb{R}^N$  on peut trouver  $(\lambda_i)_{i \in \llbracket 0, N \rrbracket} \in \mathbb{R}^N$  tel que  $\sum_{i=0}^N \lambda_i = 1$  et  $\sum_{i=0}^N \lambda_i E_i = \underline{X}$ . On définit l'extension de Lovasz de la fonction  $f$   $\forall \underline{X} \in \mathbb{R}^N$  :

$$\hat{f}(\underline{X}) = \sum_{i=0}^N \lambda_i f(E_i)$$

Les valeurs de  $\lambda$  dépendent implicitement de  $\underline{X}$ . On peut bien trouver ces coefficients  $\lambda$ , et on va même en donner une forme explicite possible. On réarrange le vecteur  $\underline{X}$  en le triant du plus grand au plus petit :  $\underline{X}_0 > \underline{X}_1 > \dots > \underline{X}_N > \underline{X}_{N+1}$  avec  $\underline{X}_0 = 1$  et  $\underline{X}_{N+1} = 0$ , on verra plus tard que ce changement d'ordre n'est pas problématique dans le cas qui nous concerne. On va poser  $\forall i \in \llbracket 0, N \rrbracket$   $\lambda_i = \underline{X}_i - \underline{X}_{i+1}$ , on a bien les deux conditions de la définition 3.2 qui sont respectées (on peut le vérifier en quelques lignes de calcul).

**Propriété 3.1.** L'extension de Lovasz d'une fonction est convexe si et seulement si cette fonction est submodular.

Cette définition passe sous silence un point important. Le premier théorème utilise des fonctions qui prennent des ensembles en argument alors que le second utilise des fonctions qui prennent leurs valeurs dans  $\{0, 1\}^N$ . Mais on peut régler ce problème en associant pour chaque ensemble  $S$  un vecteur avec des 0 à la position  $i$  si  $S$  ne contient pas  $i$  et 1 si  $S$  contient  $i$ .

La propriété 3.1 est le point de départ de l'article dont j'ai parlé précédemment. Avec l'IoU submodular on peut créer une approximation convexe. Sauf que contrairement à ce qui a été

dit dans l'article, l'IoU n'est pas submodular (démonstration en annexe). L'approximation de Lovasz reste pertinente néanmoins, mais n'est plus convexe. On définit donc l' $IoU_{lovasz}$ . On se souviendra que  $Y$  contient les labels associés aux prédictions  $\underline{X}$ , mais ne sont pas vraiment des paramètres de la fonction car ils sont toujours fixés. Ceci nous permet de préciser que modifier l'ordre de  $X$  n'importe pas car on modifie l'ordre de  $Y$  de la même manière. Ré-agencer les  $X$  selon l'ordre décroissant ne change donc rien, c'est uniquement pour simplifier l'écriture des équations.

$$IoU_{lovasz}(\underline{X}, Y) = \sum_{i=0}^N (\underline{X}_i - \underline{X}_{i+1}) IoU(E_i, Y) = \sum_{i=1}^N \underline{X}_i (IoU(E_i, Y) - IoU(E_{i-1}, Y)) \quad (3.9)$$

On a trouvé une approximation de l' $IoU$ . Il y a plusieurs interprétations possibles de cette fonction. On peut notamment la voir comme une somme pondérée des probabilités  $\underline{X}$ . Mais je préfère l'interprétation probabiliste qui donne immédiatement une image de cette approximation. En gardant en tête que les probabilités de sortie  $\underline{X}$  ont été triées par ordre décroissant, on peut définir  $IoU_{lovasz}$  comme dans l'équation 3.10

$$IoU_{lovasz}(\underline{X}, Y) = \mathbb{E}_\Delta(IoU(\{i | \underline{X}_i > \Delta\}, Y)) \quad \Delta \hookrightarrow U([0, 1]) \quad (3.10)$$

Encore une fois on jongle avec le lien entre ensemble et vecteur,  $\{i | \underline{X}_i > \Delta\}$  est équivalent à  $(1, \dots, 1, 0, \dots, 0)$  avec le dernier 1 à la  $i^{ieme}$  position. La démonstration de l'équation 3.10 va être pertinente pour la suite.

*Démonstration.* On prend  $\Delta \hookrightarrow U([0, 1])$ . D'après le théorème de transfert on a :

$$\mathbb{E}_\Delta(IoU(\{i | \underline{X}_i > \Delta\}, Y)) = \int_0^1 IoU(\{i | \underline{X}_i > a\}, Y) da \quad (3.11)$$

$$= \sum_{i=0}^N \int_{\underline{X}_i}^{\underline{X}_{i+1}} IoU(E_i, Y) da \quad (3.12)$$

$$= \sum_{i=0}^N (\underline{X}_i - \underline{X}_{i+1}) IoU(E_i, Y) \quad (3.13)$$

$$= IoU_{lovasz}(\underline{X}, Y) \quad (3.14)$$

□

On peut donc voir  $IoU_{lovasz}(\underline{X}, Y)$  comme la moyenne de la réalisation suivante : on choisit un nombre uniformément entre 0 et 1, on ne garde que les probabilités des pixels supérieurs à ce nombre et on retourne l' $IoU$  évalué avec 1 à la place des probabilités précédentes et 0 pour les

autres probabilités. Les probabilités proches de 1 apparaîtront alors très souvent dans l' $IoU$ , tandis que celle proche de 0 presque pas. On a une probabilité  $\underline{X}_i$  que l'indice  $i$  soit 1 dans l' $IoU$ .

Avec cette nouvelle forme, j'ai réappris le réseau GridNet. Les résultats n'ont malheureusement pas été meilleurs, toujours pareils, un peu au dessus de 0.65 d' $IoU$ . Ceci me permet de passer à la dernière partie, qui essaye de pousser plus loin l'idée de l'extension de Lovasz.

### 3.3.4 Transformation de l'extension de Lovasz

Vu que l' $IoU$  n'est pas submodular, il n'est pas forcément pertinent de garder la forme de l'extension de Lovasz. On va essayer de modifier légèrement cette extension. Si on reprend la formule avec l'espérance (3.10), on peut essayer de modifier la loi de  $\Delta$ . Regardons ce qu'il se passe si on prend une loi quelconque  $\mathcal{L}$  tel que  $\Delta$  soit nul hors de l'intervalle  $[0, 1]$ .

$$IoU_{lovasz}(\underline{X}, Y) = \mathbb{E}_{\Delta}(IoU(\{\underline{i} | \underline{X}_{\underline{i}} > \Delta\}, Y)) \quad \Delta \hookrightarrow \mathcal{L} \quad (3.15)$$

On note  $f_{\Delta}$  et  $F_{\Delta}$  la fonction de répartition associé à variable aléatoire  $\Delta$ . On a  $IoU_{lovasz}(\underline{X}, Y) = \sum_{i=0}^N (F_{\Delta}(\underline{X}_i) - F_{\Delta}(\underline{X}_{i+1})) IoU(E_i, Y)$ . On peut donc voir ce changement de la loi  $\Delta$  comme une modification de  $\underline{X}_i$ . On utilise plus directement  $\underline{X}_i$ , mais  $F_{\Delta}(\underline{X}_i)$ . Ceci est loin d'être absurde, on effectue souvent des modifications sur la sortie avant d'appliquer une fonction de perte. Par exemple on peut voir la *crossEntropy* comme l'application du logarithme puis l'utilisation de la fonction de perte associé à l'accuracy. La méthode de normalisation avec la fonction softmax est aussi une modification de la sortie. Si on revient à la définition 3.2, on garde deux propriété importante avec changement de  $\Delta$ . On pose  $(\lambda_i) = F_{\Delta}(\underline{X}_i) - F_{\Delta}(\underline{X}_{i+1})$ , on a toujours  $\sum_{i=0}^N \lambda_i = 1$  et quand  $\underline{X} \in \{0, 1\}^N$  on a  $IoU_{lovasz}(\underline{X}, Y) = IoU(\underline{X}, Y)$ . En revanche on a  $\sum_{i=0}^N \lambda_i E_i \neq \underline{X}$  et on se retrouve avec la modification de  $\underline{X}$  :  $\sum_{i=0}^N \lambda_i E_i = F_{\Delta}(\underline{X})$  (abus de notation qui signifie qu'on applique  $F_{\Delta}$  à chaque élément de  $\underline{X}$ ).

On peut même aller plus loin et de ne plus travailler avec des nombres entre 0 et 1 mais de rester sur  $\mathbb{R}$ . En effet le passage de  $X$  à  $\underline{X}$  (sorte de normalisation), n'est pas forcément utile. Si la sortie du réseau appartient à  $\mathbb{R}$ , on ne peut plus l'interpréter comme des probabilités d'appartenances aux classes. Plus la valeur de sortie est grande plus le réseau sera confiant dans le fait que le pixel est de la classe positive, inversement pour les négatifs. Pour cela, il suffit de ne plus normaliser avec la fonction *softmax* et d'appliquer l'approximation de l' $IoU$  avec une distribution de probabilité défini sur  $\mathbb{R}$ . On a trouvé une manière d'approximer l' $IoU$ , en se passant de cette fonction *softmax* et en utilisant n'importe quelle distribution.

Regardons maintenant la différentiabilité de cette fonction. On prend  $i \in \llbracket 1, N \rrbracket$ . Tout

d'abord en utilisant la formule 3.9, on a presque sûrement (cas particulier lorsque  $X_i = X_{i+1}$  ou  $X_i = X_{i-1}$ ) la dérivé suivante :

$$\frac{\partial IoU_{\text{Lovasz}}}{\partial \underline{X}_i}(Y, \underline{X}) = f_\Delta(\underline{X}_i)(IoU(E_i, Y) - IoU(E_{i-1}, Y)) \quad (3.16)$$

Cette formule ne fait pas intervenir la fonction de répartition  $F_\Delta$ , qui n'a pas à être connu. Notamment pour les fonctions de répartitions gaussiennes où l'on n'a pas de formule explicite de  $F_\Delta$ . On peut donc avoir une fonction de perte pas calculable explicitement mais pourtant avoir une dérivé connue et donc une descente de gradient possible.

Maintenant on pourrait se demander quelles distributions utiliser, et qu'elle est l'intuition derrière l'utilisation des différentes distributions. Pour se fixer les idées sur ces exemples de distributions, nous allons rester avec  $\underline{X}$  (valeur entre 0 et 1), mais cela n'est pas indispensable.

- $\mathcal{N}(0.5, 0.5)$ . Tout d'abord il faut normaliser par une valeur différente de  $\sqrt{2\pi}0.5$  pour pouvoir avoir une intégrale entre 0 et 1 qui vaut 1. Ensuite on peut voir cette gaussienne comme une manière de pousser les éléments indécis ( $\underline{X}_i$  qui sont autour de 0.5) vers l'un des deux côtés. En effet d'après 3.16, la dérivé va être forte que lorsque  $\underline{X}_i$  est proche de 0.5. Ceci n'est pas forcément un bon choix car les pixels mal classé aux extrême auront un faible gradient aussi.
- $\mathcal{N}(0.5, 10)$ . Même situation que précédemment, mais avec une variance plus grande, on se rapproche plus de la loi uniforme.
- $\mathcal{N}(0.9, 0.5)$ . Ce type de gaussienne décenterer peut être pertinent si on ne veut pas avoir de faux positif. Les pixels négatifs qui sont classé proche de 1 auront un gradient très fort.
- $\frac{4}{3}\mathbf{1}_{[0,0.25]} + \frac{4}{6}\mathbf{1}_{[0.25,0.75]} + \frac{4}{3}\mathbf{1}_{[0.75,1]}$ . Distribution en escalier qui se concentre que sur les valeurs proche de 0 et de 1.

Ici on a raisonné en travaillant sur la distribution  $\Delta$ , mais on peut aussi choisir directement  $F_\Delta$  pour savoir quelle modification on va apporter à  $\underline{X}$ . On peut prendre la fonction exponentielle, une sigmoïde, une fonction affine. Au delà d'une modification de  $\underline{X}$ , l'interprétation n'est pas évidente. Mais il peut être intéressant de voir cette extension de l' $IoU$  sous cette forme.

### 3.3.5 Conclusion des approximations de Lovasz

Aucune expérience n'a été faite sur cette idée de modification de l'extension de Lovasz au moment de l'écriture de ce rapport. Les nombreuses modifications de la fonction de perte de GridNet n'ont pas changé grand chose au résultat. Que se soient les approximations de l' $IoU$  ou d'autres fonctions de perte que j'ai testé lors de ce stage. Pour réduire l'écart entre Grid-

Net est les autres algorithmes, il faudrait se concentrer sur d'autres domaines. Notamment le pré-apprentissage sur d'autres jeux de données (tel que ImageNet [4]), qui améliore remarquablement les résultats dans de nombreuses situations. Une expérimentation sur un autre jeu de donnée est en cours, pour essayer de détecter les différences entre toutes les approximations de l' $IoU$ , pour voir leurs points fort et faible dans un cas pratique.

Dans cette deuxième partie du rapport, nous avons compris la segmentation sémantiques d'images en utilisant l'exemple du jeu de donnée Cityscapes. Ensuite une architecture, GridNet sur laquelle j'ai travaillé pendant ce stage ainsi que quelques résultats visuels. Puis j'ai développé rigoureusement les extensions possibles de l' $IoU$  en me concentrant sur celle de Lovasz.

## 4 Conclusion

D'un point de vue technique les résultats de ce stage sont mitigés, une implémentation en Pytorch de GridNet qui fonctionne bien mais pas de progrès par rapport aux résultats précédents. Quelques découvertes intéressantes sur les manières de mieux approximé l' $IoU$ , mais encore une fois des résultats toujours similaires. Néanmoins j'ai bon espoir que mes recherches servent dans d'autres applications que la segmentation sémantique, car les fonctions de perte créées sont utilisables dès que l'on test les algorithmes avec la  $mIoU$ . Cela pourrait être une direction à prendre pour continuer ce stage. En revanche si l'objectif est d'améliorer l'état de l'art sur la segmentation d'image, je pense qu'il faut modifier l'architecture de GridNet. Cela peut notamment permettre un pré-apprentissage sur ImageNet. La deuxième solution est évidemment de tester plus de paramètres et d'architectures. J'ai gardé la même architecture et les mêmes hyper-paramètres tout au long de mon stage, si je veux obtenir de bons résultats, il faut regarder le plus de paramètre possible et le plus d'architecture possible. Mais des contraintes techniques au laboratoire font qu'il n'est pas possible de tester autant de programme en un temps raisonnable.

D'un point de vu plus personnel ce stage m'aura permis de découvrir le monde de la recherche. J'aurais aussi acquis et renforcé de nombreuses connaissance en mathématiques, informatique et machine learning. Ceci me semble au moins aussi important que d'avoir réussi à faire des découvertes. J'ai aussi passé 5 mois géniaux, entourer d'un groupe soudé, ce qui m'a totalement donné envie de continuer en thèse.

## Remerciements

Je vais essayer de rester succinct dans cette liste de remerciements même si beaucoup devrait avoir leur place.

Je remercie tout d'abord Amaury Habrard qui a répondu à ma candidature spontané. Puis surtout Rémi Emonet de m'avoir accepté en stage, puis de m'avoir guidé tout au long de ce stage et de toujours avoir été intéressé par la direction que je prenais. Je remercie aussi toute l'équipe pour l'accueil et les moments passés ensemble que se soit dans le laboratoire, autour d'une bière ou au milieu d'un match de foot. Je remercie Emmanuel Monfrini d'avoir accepté d'être mon encadrant de stage à Télécom SudParis mais aussi de m'avoir activement aidé dans ma recherche de stage et de thèse tout au long de l'année. Je remercie aussi Jérémie Jakubowicz avec qui j'ai aussi beaucoup échangé autour de ma recherche de thèse. Enfin je remercie ma compagne, en particulier pour m'avoir soutenu lors du moment crucial de mon choix de stage.

## Annexe

Nous allons ici démontrer rigoureusement que l' $IoU$  n'est pas submodular. Cette erreur à été faite dans [3] qui lui même cite un article dans lequel la démonstration est erronée [5]. On peut définir une fonction submodular de plusieurs façons. On rappel une des définitions d'une fonction submodular que l'on utilisera 4.1. On défini aussi l' $IoU$  de manière ensembliste 4.2.

**Definition 4.1.** *On note  $f : 2^{\{1, \dots, m\}} \rightarrow \mathbb{R}$  avec  $m \in \mathbb{N}$ .  $f$  est submodular si et seulement si  $\forall (A, B) \in (2^{\{1, \dots, m\}})^2$  avec  $A \subseteq B$  et  $\forall x \in \llbracket 1, m \rrbracket \setminus A$ .*

$$f(A \cup \{x\}) - f(A) \geq f(B \cup \{x\}) - f(B)$$

**Definition 4.2.** *On prend  $(A, Y) \in (2^{\{1, \dots, m\}})^2$ , et on note  $|A|$  la fonction qui retourne le nombre d'élément dans  $A$ .  $Y$  correspond à la vérité terrain et donc n'est pas une variable.*

$$IoU_Y(A) = \frac{|A \cap Y|}{|A \cup Y|}$$

On va démontrer que l' $IoU$  n'est pas submodular et on ne va pas se restreindre a un contre exemple mais plutôt montrer que l'inégalité est souvent pas vérifier (notamment avec les hypothèses si dessous). On prend  $(A, B, Y) \in (2^{\{1, \dots, m\}})^3$  avec  $A \subset B$  et  $|A \cap Y| = |B \cap Y| > 0$ . On notera  $a_b = |A \cap Y| = |B \cap Y|$ ,  $a_d = |A \cup Y|$ ,  $b_d = |B \cup Y|$ , on remarquera que  $a_d < b_d$  Montrons que  $\forall x \in 2^{\{1, \dots, m\}} \setminus (Y \cup B)$  on a  $R = (IoU_Y(A \cap \{x\}) - IoU_Y(A)) - (IoU_Y(B \cap \{x\}) - IoU_Y(B)) < 0$ . Soit  $x \in 2^{\{1, \dots, m\}} \setminus (Y \cup B)$

$$R = (IoU_Y(A \cup \{x\}) - IoU_Y(A)) - (IoU_Y(B \cup \{x\}) - IoU_Y(B)) \quad (4.1)$$

$$= \frac{|(A \cup \{x\}) \cap Y|}{|(A \cup \{x\}) \cup Y|} - \frac{|A \cap Y|}{|A \cup Y|} - \left( \frac{|(B \cup \{x\}) \cap Y|}{|(B \cup \{x\}) \cup Y|} - \frac{|B \cap Y|}{|B \cup Y|} \right) \quad (4.2)$$

$$= \frac{|A \cap Y|}{|A \cup \{x\} \cup Y|} - \frac{|A \cap Y|}{|A \cup Y|} - \left( \frac{|B \cap Y|}{|B \cup \{x\} \cup Y|} - \frac{|B \cap Y|}{|B \cup Y|} \right) \quad (4.3)$$

$$= \frac{ab_n}{a_d + 1} - \frac{ab_n}{a_d} - \left( \frac{ab_n}{b_d + 1} - \frac{ab_n}{b_d} \right) \quad (4.4)$$

$$= -ab_n \left( \frac{1}{(a_d + 1)a_d} - \frac{1}{(b_d + 1)b_d} \right) \quad (4.5)$$

$$(4.6)$$

Or on a successivement :

$$a_d < b_d \quad (4.7)$$

$$(a_d + 1)a_d < (b_d + 1)b_d \quad (4.8)$$

$$\frac{1}{(a_d + 1)a_d} > \frac{1}{(b_d + 1)b_d} \quad (4.9)$$

$$\frac{1}{(a_d + 1)a_d} - \frac{1}{(b_d + 1)b_d} > 0 \quad (4.10)$$

$$-ab_n \left( \frac{1}{(a_d + 1)a_d} - \frac{1}{(b_d + 1)b_d} \right) < 0 \quad (4.11)$$

Finalement  $R < 0$  sous toutes les hypothèses données sur  $x, A, B$ , donc l'IoU n'est pas submodular. On peut notamment remarquer que si on modifie la définition de d'une fonction submodular en rajoutant des valeurs absolue comme dans la définition 4.3, la preuve ne fonctionne plus et l'IoU a probablement cette nouvelle propriété (résultat qu'il faudrait néanmoins démontrer).

**Definition 4.3.** On note  $f : 2^{\{1, \dots, m\}} \rightarrow \mathbb{R}$  avec  $m \in \mathbb{N}$ .  $f$  est positive submodular si et seulement si  $\forall (A, B) \in 2^{\{1, \dots, m\}}^2$  avec  $A \subseteq B$  et  $\forall x \in [1, m]$ .

$$|f(A \cup \{x\}) - f(A)| \geq |f(B \cup \{x\}) - f(B)|$$

## Table des figures

1	Un exemple de segmentation d'images . . . . .	1
2	Exemple de simulation physique d'élèves de L1. Tir d'une balle de golf avec un angle et une vitesse initial fixé par l'utilisateur . . . . .	5

3	Présentation de Guillaume l'un des doctorants du laboratoire.	8
4	Photo de l'équipe transformée avec un changement de style artistique.	9
5	Exemple de photo de la base de données Cityscapes [2]	12
6	Photo entièrement segmentée de la base de donnée Cityscapes [2]	12
7	Matrice de confusion avec notamment les <i>VraiPositif<sub>moto</sub></i> (71), <i>VraiNegatif<sub>moto</sub></i> ( $1 + 4 + 1 + 1 + 3 + 12 + 13 + 21 + 29 = 85$ ) et les <i>FauxPositif<sub>moto</sub></i> ( $2 + 1 + 1 + 2 + 1 + 3 + 13 + 13 = 36$ ) associé à la détection de la classe moto. On a donc $IoU_{moto} = \frac{71}{71+36+85} = 0.37$ Les cases grisées n'ont aucune prédiction associée. Les prédictions sont heureusement principalement sur la diagonale. Les valeurs de la matrice correspondent au nombre de pixels sur une image.	15
8	Un exemple de convolution. Le filtre de convolution (de taille $3 \times 3$ ici) étant le même pour tous les pixels de cette couche, on réduit grandement le nombre de paramètres par rapport à une architecture complètement connectée. Ce sont les paramètres de ce filtre de convolution qui vont être modifiés pour répondre au problème. Image prise sur le site <a href="http://openresearch.ai/t/network-in-network/39">http://openresearch.ai/t/network-in-network/39</a>	16
9	Un réseau classique complètement convolutif. A gauche l'image d'entrée, puis la réduction de dimension au milieu et enfin la restitution de l'image qui a été segmentée. On compare ensuite cette image à la vérité terrain (Groundtruth en anglais). Image prise sur l'article <a href="https://perso.univ-st-etienne.fr/fod07375/pdf/gridNetwork.pdf">https://perso.univ-st-etienne.fr/fod07375/pdf/gridNetwork.pdf</a> [1]	17
10	Sous réseau de GridNet. Chaque couche est composée de deux normalisations, de deux fonctions d'activations ReLU et de deux convolutions. Image prise dans l'article <a href="https://perso.univ-st-etienne.fr/fod07375/pdf/gridNetwork.pdf">https://perso.univ-st-etienne.fr/fod07375/pdf/gridNetwork.pdf</a> [1]	17
11	Représentation de GridNet avec les différents réseaux convolutif qu'il généralise. Image prise dans l'article <a href="https://perso.univ-st-etienne.fr/fod07375/pdf/gridNetwork.pdf">https://perso.univ-st-etienne.fr/fod07375/pdf/gridNetwork.pdf</a> [1]	18
12	Une photo de Darmstadt en Allemagne, la segmentation avec GridNet et la vérité terrain. Base de données d'entraînement.	20
13	Une photo de Frankfurt en Allemagne, la segmentation avec GridNet et la vérité terrain. Base de données de test.	23
14	Une photo de Münster en Allemagne, la segmentation avec GridNet et la vérité terrain. Base de données d'entraînement.	24

15 Une segmentation de Munster en Allemagne avec la <i>crossEntropy</i> , la vrai segmentation la segmentation avec GridNet ( $mIoU_c$ ). Base de données d'entraînement de test. . . . .	29
---	----

## Références

- [1] Fourure Damien and Emonet Rémi and Fromont Elisa and Muselet Damien and Tréneau Alain and Wolf Christian "Residual Conv-Deconv Grid Network for Semantic Segmentation." Proceedings of the British Machine Vision Conference, 2017.
- [2] Cordts, Marius, et al. "[The cityscapes dataset](#)" CVPR Workshop on the Future of Datasets in Vision.
- [3] Matthew, Maxim Berman Amal Rannen Triki, and B. Blaschko. "[The Lovász-Softmax loss: A tractable surrogate for the optimization of the intersection-over-union measure in neural networks](#)" 2018.
- [4] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. "[ImageNet: A Large-Scale Hierarchical Image Database](#)" In CVPR 2009
- [5] Yu and M. B. Blaschko. "[The Lovasz hinge: A convex surrogate for submodular losses](#)". 2015