THÈSE

En vue de l'obtention du

DOCTORAT DE L'UNIVERSITÉ DE TOULOUSE

Délivré par : *l'Université Toulouse - Jean Jaurès*

Présentée et soutenue le *Date de défense (15/06/2022)* **par :**

Julien Biau

Nouvelles approches évolutionnaires pour des algorithmes interprétables de vision par ordinateur pour les systèmes embarqués

---

JURY

Stéphane Doncieux	Professeur d'Université	Président du Jury
Hervé Luga	Professeur d'Université	Membre du Jury
Georges Zissis	Professeur d'Université	Membre du Jury
Cyril Fonlupt	Professeur d'Université	Membre du Jury
Jenny Benois-Pineau	Professeur d'Université	Membre du Jury

École doctorale et spécialité :
MITT : Domaine STIC : Intelligence Artificielle

Unité de Recherche :
REVA (UMR 5055)

Directeur(s) de Thèse :
Hervé Luga et *Georges Zissis*

Rapporteurs :
Stéphane Doncieux et *Cyril Fonlupt*

Remerciements

Je voudrais tout d'abord remercier Yves Le-henaff, président de la société Kawantech qui m'a permis de réaliser cette thèse en acceptant qu'elle ait lieu dans le cadre de ma fonction au sein de la société Kawantech.

Ensuite, je tiens à remercier Hervé Luga et Georges Zissis pour avoir accepté d'encadrer cette thèse en m'aiguillant pendant quatre années pour que je puisse la mener à son terme.

Je tiens aussi à remercier très sincèrement Sylvain Cussat-Blanc qui a été disponible à chaque fois que nécessaire pour m'orienter dans mes choix et m'aider à progresser tout au long de ma démarche de recherche scientifique.

Et pour finir, je tiens à remercier Marie Martinez et Martine Biau qui ont eu la difficile tâche de relire et corriger cette thèse.

Abstract

In recent years, the emergence of deep neural networks have made it possible to go beyond the state of the art in terms of image analysis. They are used today in many fields, from commerce to health and industry. However their precision has a cost: a very high computing power making their use on embedded electronic cards very complex or even impossible. This thesis aims to propose effective alternative solutions to overcome this lock. A new method for extracting and categorizing moving objects in videos is proposed. This method running on an electronic card embedded in a luminaire was developed by proposing a two-stage pipeline: the first phase allows the creation of a set of data from videos and the learning of an LSTM model while the second uses this same model to categorize objects. The use of metadata instead of images has here allowed a drastic reduction in the computational cost required for equivalent precision. An adaptation of the framework CGP-IP has been created optimized for genetic improvement allowing better conservation of the graph structure. It has enabled image filters made by experts to obtain better precision in a lower number of generations. In order to better control the necessary computing power on limited processors, the framework CGP-IP-GI has been modified with a multi-objective algorithm of the NSGA-II type. It thus allowed on a filter developed by experts to divide the necessary computing power by five for a loss of precision of only five percent. The works presented have therefore made it possible to address several objectives: From a computational point of view, they propose algorithms making it possible to obtain state-of-the-art results on processors with limited computing power. From a point of view of the exploitability of the results, the use of genetic programming has allowed the development of applications and interpretable image filters. Finally, we have shown that a multi-objective algorithm makes it possible to modulate the computing power while obtaining convincing results.

Résumé

Depuis quelques années, l'émergence des réseaux de neurones profonds a permis d'outrepasser l'état de l'art en matière d'analyse d'images. Ils sont aujourd'hui utilisés dans de multiples domaines, du commerce à la santé en passant par l'industrie. Seulement leurs précisions ont un coût : une puissance de calcul très élevée rendant leur utilisation sur des cartes électroniques embarquées très complexe voire impossible. Ces travaux de thèse visent à proposer des solutions alternatives efficaces permettant de lever ce verrou. Une nouvelle méthode d'extraction et de catégorisation d'objets en mouvement dans les vidéos est proposée. Cette méthode s'exécutant sur une carte électronique embarquée dans un luminaire a été développée en proposant un pipeline à deux étages : la première phase permet la création d'un ensemble de données issues de vidéos et l'apprentissage d'un modèle LSTM tandis que la seconde utilise ce même modèle pour catégoriser les objets. L'utilisation de méta données en lieu et place des images a permis ici une réduction drastique du coup de calcul nécessaire pour une précision équivalente. Une adaptation du *framework* CGP-IP a été créée et optimisée pour l'amélioration génétique, permettant une meilleure conservation de la structure du graphe. Elle a permis, sur des filtres à images réalisés par des experts, d'obtenir une meilleure précision en un nombre inférieur de générations. Afin de mieux contrôler la puissance de calcul nécessaire sur des processeurs limités, le *framework* CGP-IP-GI a été modifié avec un algorithme multiobjectif de type NSGA2. Il a ainsi permis, sur un filtre développé par des experts, de diviser la puissance de calcul nécessaire par cinq pour une perte de la précision de seulement cinq pour cent. Les travaux présentés ont donc permis d'adresser plusieurs objectifs. D'un point de vue computationnel, ils proposent des algorithmes permettant d'obtenir des résultats de l'état de l'art sur des processeurs à puissance de calcul limité. D'un point de vue de l'exploitabilité des résultats, l'utilisation de la programmation génétique a permis le développement d'applications et de filtres d'images interprétables. Enfin nous avons montré qu'un algorithme multiobjectif permet de moduler la puissance de calcul tout en obtenant des résultats probants.

Contents

Chapter 1

Contexte industriel

1.1 La société

Kawantech est une société dans le domaine de la Smart City. La Smart City regroupe différents services pour digitaliser la ville. Kawantech travaille dans ce domaine depuis 2011. L'ensemble des services fournis sont construits autour d'un unique produit : Kara. Il permet de contrôler l'éclairage des luminaires, surveiller des places de parking ainsi que produire des statistiques sur les flux de déplacements (piétons, voitures, etc..).

1.1.1 Le produit : Kara

Le Kara est un produit étanche composé d'une carte électronique et d'un capteur vidéo à installer dans la coque du luminaire. La carte électronique est de type multimédia. Elle est composée d'un processeur et de plusieurs périphériques permettant de communiquer (tel que le Wifi, le Bluetooth, le Dash7, LoR,etc. . .). Le système d'exploitation utilisé est un Linux de type embarqué. Tous les services fournis par le Kara sont pris en charge dans un logiciel développé en C++ qui est exécuté sur le Linux. Pour réduire sa consommation, la carte électronique est conçue avec des composants de type embarqué. Le processeur est un Arm 32Bit qui fonctionne à 1Ghz avec 1Go de mémoire vive.

1.1.2 Les services logiciels implantés dans le Kara

Le premier service fourni par le Kara est la gestion de l'éclairage. Grâce à lui, la puissance de l'éclairage est régulée en fonction de la circulation dans la rue. Il permet ainsi une

économie de 65% de l'énergie consommée pour le fonctionnement du luminaire. Il réduit ainsi la facture d'électricité de 65% et de fait, atténue les perturbations de la lumière artificielle sur la faune et la flore.

Le deuxième service développé est le Smart Parking. Il permet de définir des places dans la rue et de surveiller si ces places sont disponibles ou occupées. C'est un service bien moins couteux qu'un produit concurrent installé dans le sol d'autant plus s'il est couplé au service d'éclairage. Toutes les informations sur l'état des places sont transmises à nos serveurs et peuvent être consultées par nos clients.

Le dernier service développé est le comptage. Il permet d'analyser des flux de personnes ou de véhicules dans la rue. Les informations sur les flux sont transmises à nos serveurs qui peuvent également être consultées par nos clients. Des statistiques sur les flux permettent de les analyser en fonction de différents critères tel que la date, le climat, des événements et ainsi prédire les mouvements de flux à venir.

Pour respecter la vie privée des personnes dans la rue, le Kara ne sauvegarde aucune vidéo ni image provenant de son capteur vidéo. Le capteur vidéo est un capteur spécialisé pour la vision de nuit. Le flux vidéo est en noir et blanc et sa résolution est de 600 pixels par 400 pixels.

Le cœur de notre application utilise différents algorithmes pour détecter ce qui se déplace dans la rue. Nous obtenons une image différentielle en soustrayant l'image actuelle avec la précédente. Nous nettoyons cette image en utilisant plusieurs fonctions issues de la librairie OpenCV [1] telles que *Erode* ou *Dilate* qui permettent de réduire le bruit. L'image résultante est ainsi composée de blobs représentant des objets en mouvement. Nous catégorisons ensuite ces blobs pour pouvoir les transférer aux différents services.

Tous ces services sont extrêmement consommateurs de puissance de calcul et d'énergie. Parce que nous utilisons une électronique embarquée, notre puissance de calcul est limitée et notre logiciel et nos algorithmes doivent être optimisés au maximum.

1.2 Objectifs de la thèse

La société est composée d'une solide équipe technique dont une des spécialités est le traitement du signal et de l'image. L'évolution des performances des algorithmes d'intelligence

[1] https://opencv.org/

artificielle et l'arrivée des réseaux de neurones profonds dans l'industrie a levé plusieurs questionnements dans la société.

En effet, le principal produit de la société Kara a pour principale tâche de catégoriser des objets en mouvement afin d'adapter la puissance lumineuse en conséquence. Pour réaliser cette tâche, il est ainsi possible d'utiliser des algorithmes de détection des objets dans une image. Dans notre contexte où les images proviennent de vidéos, il existe depuis le début des années 2000 de nombreux algorithmes pour analyser des vidéos. Enfin, il est aussi possible d'utiliser la vidéo pour obtenir des informations temporelles que l'on pourra utiliser dans des algorithmes sur les séries temporelles. Dans ces trois domaines, détection des objets dans les images, classification des vidéos et séries temporelles, l'arrivée des réseaux de neurones a rebattu les cartes avec des performances largement supérieures à celles des algorithmes traditionnels. La surperformance des réseaux de neurones profonds par rapport aux algorithmes traditionnels est dû à une augmentation constante du nombre d'objets et d'images dans les ensembles de données combinée à l'utilisation de cartes graphiques puissantes qui permet ainsi d'apprendre à partir de centaines de milliers de données avec des temps d'apprentissage toujours plus courts. Dans notre contexte de carte embarquée, nous n'avons pas accès à de puissantes cartes graphiques, bien au contraire, notre puissance de calcul est très limitée.

Il était devenu urgent de gagner en compétence au sein de la société pour pouvoir mieux comprendre les enjeux de l'intelligence artificielle ainsi que ses apports dans notre industrie. C'est dans cette optique que cette thèse a démarré et doit apporter plusieurs réponses au delà d'une montée en compétence en intelligence artificielle : une solution équivalente ou avec de meilleurs résultats à la nôtre avec de l'intelligence artificielle est-elle envisageable ? quelles sont les limites et les contraintes des réseaux de neurones profonds ? est-il possible de faire de l'intelligence artificielle sur des processeurs embarqués (sans processeur graphique ou de calcul dédié) ? Pour cela, plusieurs travaux vont être réalisés tel que : des mesures de performance en terme de puissance de calcul nécessaire mais aussi de précision lors d'utilisation de réseau de neurone sur des cartes électroniques embarquées, la création d'un pipeline automatisé pour extraire des méta datas depuis des objets en mouvement afin de les catégoriser avec un LSTM, l'utilisation de la programmation génétique pour augmenter l'efficacité de filtres à images désignés par des experts tout en limitant les ressources processeurs.

Chapter 2

Etat de l'art

Le chapitre est découpé en quatre sections qui présentent l'état de l'art dans les domaines suivants : la détection des objets dans une image dans la section 2.1, la détection des objets dans une vidéo dans la section 2.2, la dimension temporelle dans la section 2.3.

2.1 La détection d'objets

La détection d'objets est une catégorie importante de la vision assistée par ordinateur qui vise à retrouver des instances d'objets de différentes classes dans une image digitale. L'objectif de la détection d'objets est de développer des modèles informatiques ou des techniques qui permettent de localiser et de catégoriser des objets dans une image. Ces informations sont nécessaires pour les applications de vision assistée par ordinateur. La détection d'objets est un socle pour différentes tâches qui en découlent telle que la segmentation d'images, l'étiquetage d'images, le suivi d'objets dans l'image, etc. L'arrivée des réseaux de neurones profonds depuis quelques années a considérablement accéléré la recherche sur ce sujet. Les algorithmes de détection d'objets sont actuellement utilisés dans différents domaines tels que les voitures autonomes, la vision des robots, la vidéo surveillance, etc. La figure 2.1 présente le nombre croissant de publications qui concernent la détection d'objets depuis les années 2000.

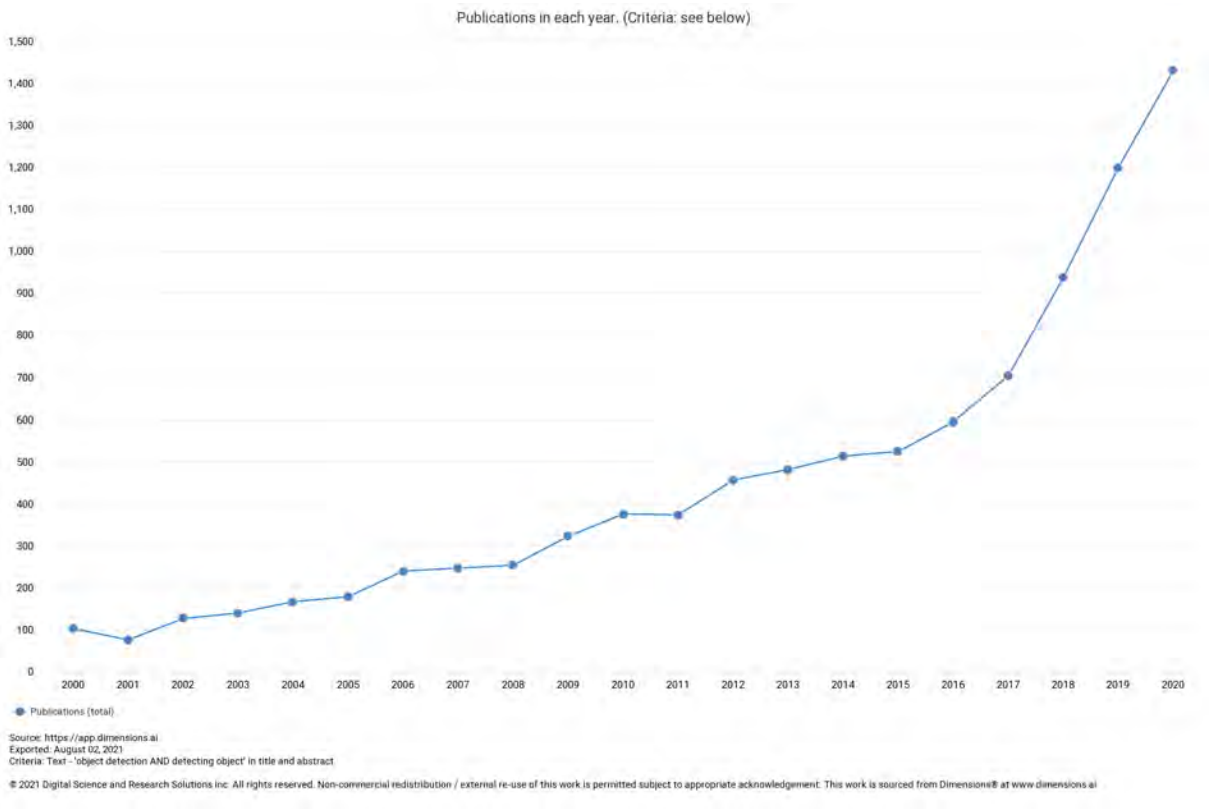


Figure 2.1: Le nombre croissant de publications sur la détection d'objet

Nous allons d'abord étudier les ensembles de données pour la catégorisation d'objets de la vie courante dans la section 2.1.1 ainsi que les métriques utilisées sur les précédents ensembles de données pour mesurer la performance des détecteurs dans la section 2.1.2. Nous étudierons ensuite les travaux concernant la détection d'objets dans la section 2.1 au cours des 20 précédentes années, en commençant par les détecteurs classiques dans la section 2.1.3 et enfin les détecteurs à base de réseaux de neurones profonds dans les sections 2.1.4 et 2.1.5.

	train		validation		trainval	
	images	**objets**	**images**	**objets**	**images**	**objets**
VOC-2007	2 501	6301	2 510	6 307	5 011	12 608
VOC-2012	5 717	13609	5 823	13 841	11 540	27 450
ILSVRC-2014	456 567	478 807	20 121	55 502	476 688	534 309
ILSVRC-2017	456 567	478 807	20 121	55 502	476 688	534 309
MS-COCO-2015	82 783	604 907	40 504	291 875	123 287	896 782
MS-COCO-2018	118 287	860 001	5 000	36 781	123 287	896 782
OID-2018	1 743 042	14 610 229	41 620	204 621	1 784 662	14 814 850

Table 2.1: Exemple d'ensemble de données réputées

2.1.1 Les jeux de données

Construire de larges ensembles de données sans qu'ils ne soient biaisés est essentiel pour le développement des algorithmes de vision assistée par ordinateur. Dans la détection des objets, plusieurs ensembles de données reconnues et testées ont été créés dans les 10 dernières années : PASCAL VOC Challenge [42, 43], ImageNet Large Scale Visual Recognition Challenge [153], MS-COCO Détection Challenge [100].

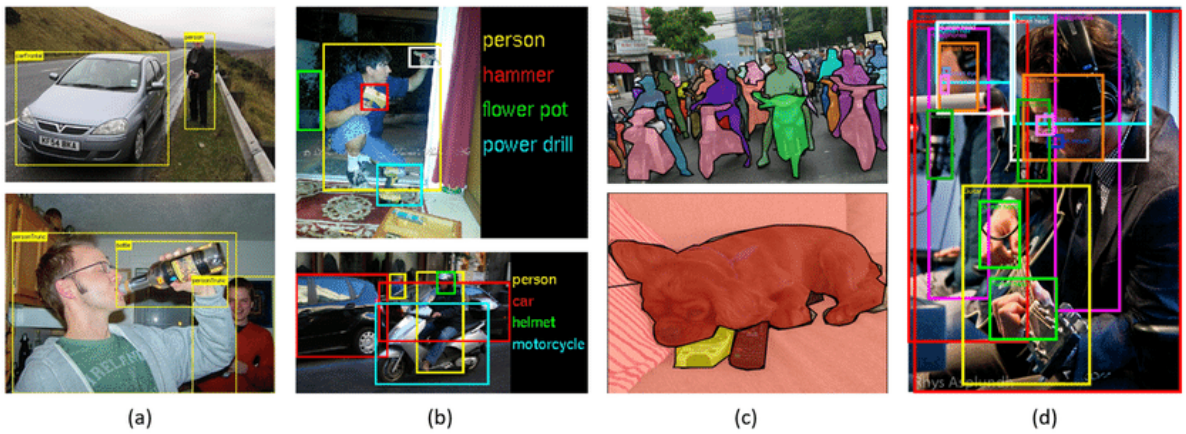


Figure 2.2: Exemples d'images extraites de (a) PASCAL-VOC07, (b) ILSRVRC, (c) MS-COCO et (d) Open Images

Pascal VOC

De 2005 à 2012, PASCAL Visual Object Classes (VOC)[1] [42, 43] a été l'une des plus importantes compétitions au début de la communauté de la vision assistée par ordinateur. Il y a plusieurs tâches dans PASCAL VOC, telles que la classification, la détection d'objets, la segmentation et la détection d'actions. Deux versions sont principalement utilisées : VOC07 (5000 images avec 12000 annotations) et VOC12 (11000 images avec 27000 annotations). 20 classes d'objets que l'on rencontre dans la vie de tous les jours sont annotées dans ces ensembles de données : personne, oiseau, chat, vache, chien, cheval, mouton, avion, vélo, bateau, bus, voiture, moto, train, bouteille, chaise, table, plante en pot, canapé, télé.

[1] http://host.robots.ox.ac.uk/pascal/VOC/

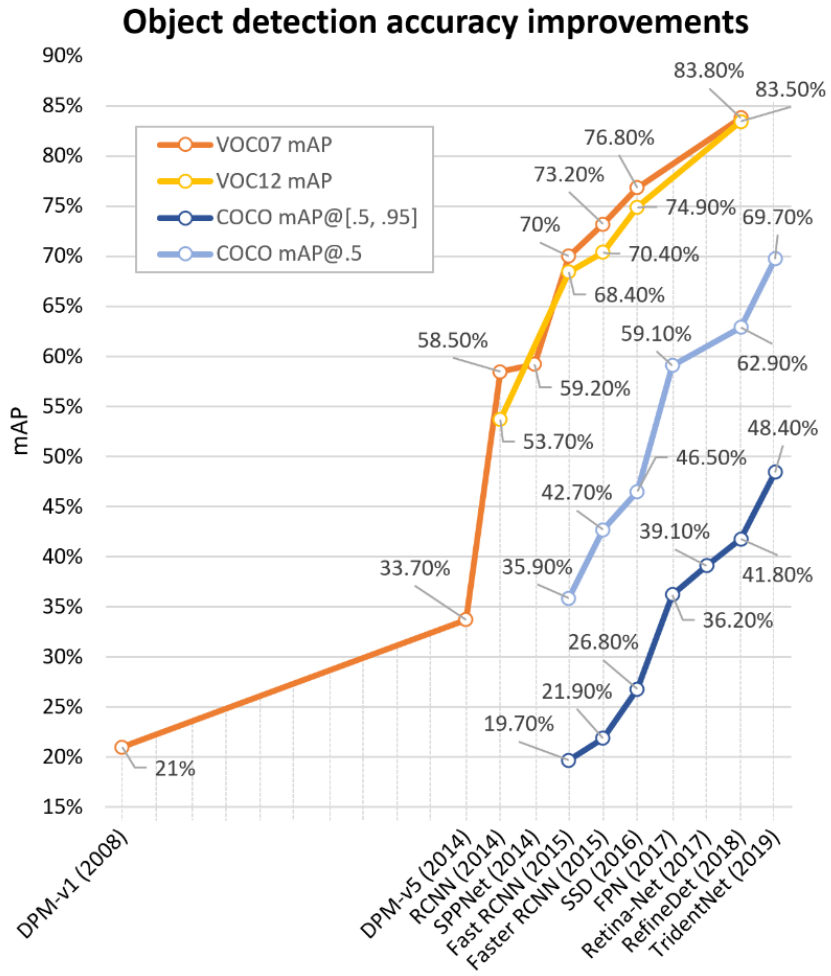


Figure 2.3: L'amélioration de la précision sur les ensembles de données VOC07, VOC12 et MS-COCO de 2008 à 2018 (Source [195])

ImageNet Large Scale Visual Recognition Challenge ILSVRC

ILSVRC[2] [153] a grandement contribué à l'avancée de l'état de l'art dans la détection d'objets génériques. ILSVRC était une compétition organisée chaque année entre 2010 et 2017. Il utilise les images ImageNet[3] et contient 200 classes d'objets visuels, 517 000 images annotées avec 534 000 objets.

MS-COCO

MS-COCO[4] [100] est à ce jour l'ensemble de données le plus complet pour la détection d'objets. La compétition annuelle basée sur cet ensemble de données se tient depuis 2015. Il a moins de classes que ILSVRC, par exemple, MS-COCO17 contient 164 000 images

[2]http://image-net.org/challenges/LSVRC/
[3]https://www.image-net.org/
[4]http://cocodataset.org/

annotées avec 897 000 objets parmi 80 classes. Si on compare MS-COCO avec VOC et ILSCRC, son évolution est d'avoir une localisation au pixel près plutôt qu'avec des boîtes de coordonnées, ce qui permet une meilleure localisation. De plus, MS-COCO contient plus de petits objets (moins de 1% de la surface de l'image) et avec des densités d'objets plus élevées. Tout comme ImageNet à son époque, MS-COCO est devenu de facto un standard pour la communauté autour de la détection des objets.

Open Images

La compétition Open Images Detection (OID)[5] [92] démarre en 2018. Elle est composée de deux tâches : la première est la détection d'objets standards, alors que la deuxième est la détection de la relation visuelle entre une paire d'objets, par exemple, la relation entre une (a) bière sur une (b) table ou une (a) personne tenant une (b) guitare. L'ensemble de données pour la détection d'objets est composé de 1 910 000 images avec 15 440 000 objets annotés pour 600 classes.

Ensemble de données pour les autres tâches de détection

En plus de la détection d'objets généraux, les 20 dernières années ont connu le développement d'applications de détection dans différents domaines comme la détection des piétons, la détection des visages, la détection du texte, la détection des signaux de circulation. La Table 2.2 liste le plus populaire pour chaque domaine. La Table 2.3 liste le plus récent pour chaque domaine.

[5]https://storage.googleapis.com/openimages/web/index.html

Ensemble de données	**année**	**description**	**#cités**
INRIA [26]	2005	L'un des premiers ensembles de données de piétons. Introduits par la publication HOG [26]. http://pascal.inrialpes.fr/data/human/	24 705
FDDB [84]	2010	2 800 images et 5 000 visages extraits depuis Yahoo! avec des occlusions, différentes poses, images flous, etc. http://vis-www.cs.umass.edu/fddb/index.html	531
ICDAR [108]	2003	ICDAR2003 est un des premiers ensembles de données public de détection de texte. ICDAR 2015 et 2017 sont des évolutions [87, 158]. http://rrc.cvc.uab.es/	530
LISA [123]	2012	L'un des premiers ensembles de données de panneaux de signalisation. 6 600 images vidéo, 7 800 instances de 47 panneaux de signalisation américains. http://cvrr.ucsd.edu/LISA/lisa-traffic-sign-dataset.html	325

Table 2.2: 4 ensembles de données les plus cités

Ensemble de données	**année**	**description**	**#cités**
EuroCity [13]	2018	Le plus large ensemble de données de piétons issu de 31 villes de 12 pays d'Europe. 238 000 instances depuis 47 000 images	1
WildestFaces [188]	2018	68 000 images vidéo et 2 200 plans de 64 célébrités combattant dans des scénarios sans contraintes.	2
COCOText[168]	2016	Le plus large ensemble de données de détection de texte. Construit depuis MS-COCO, 63 000 images et 173 000 textes annotés. https://bgshih.github.io/cocotext/	69
BSTL [7]	2017	Le plus large ensemble de données de feu de signalisation. 5 000 images statiques, 8 300 images vidéo, et 24 000 instances de feu de signalisation. https://hci.iwr.uni-heidelberg.de/node/6132	21

Table 2.3: 4 ensembles de données les plus récents

Les statistiques de ces ensembles de données sont dans la table 2.1. La Figure 2.3 montre l'évolution de la précision sur les ensembles de données VOC07, VOC12 et MS-COCO de 2008 à 2018. La Figure 2.2 montre quelques exemples d'images de ces ensembles

de données.

2.1.2 Les métriques

Comment pouvons-nous évaluer l'efficacité d'un détecteur d'objet ? Cette question a eu différentes réponses au cours du temps. Au début, il n'y avait pas de critère d'évaluation accepté pour la détection d'objet. Par exemple, au début de la recherche des piétons [26], *false positives per window* (FPPW) était souvent utilisé comme métrique.

$$FPPW = \frac{\text{number of False Positive}}{\text{number of windows}} \tag{2.1}$$

Cependant FPPW présente des défauts et échoue à prédire la performance d'une image complète dans certains cas [38]. En 2009, l'ensemble de données Caltech [38] sur la détection des piétons est créé et la nouvelle métrique d'évaluation est FPPW à *false positives per image* (FPPI).

$$FPPI = \frac{\text{number of False Positive}}{\text{number of images}} \tag{2.2}$$

Dans les années suivantes, la méthode la plus fréquente d'évaluation des détecteurs d'objets est *Average precision* (AP) qui a été initialement introduite avec VOC07 2.1.1. AP est définie comme la précision moyenne de détection. Pour comparer la performance pour toutes les catégories, *mean AP* (mAP) moyenné sur toutes les classes est utilisé comme métrique finale de la performance. Pour mesurer la précision de localisation de l'objet, *Intersection over Union* (IoU) est utilisé pour vérifier que la localisation prédite et la localisation réelle sont inférieures à un seuil prédéfini à 0.5. IoU est devenu de facto la métrique pour la détection des objets depuis plusieurs années. Après 2014, suite à la popularité des ensembles de données MS-COCO 2.1.1, les chercheurs ont commencé à accorder plus d'attention à la précision des boîtes de localisation. A la place d'utiliser un seuil IoU fixe, MS-COCO AP est moyenné avec différents seuils IoU entre 0.5 et 0.95. Ce changement de métrique a encouragé plus de précision dans la localisation des objets. Récemment, il y a eu des développements approfondis dans l'évaluation de l'ensemble de données Open Images, en considérant les groupes de boîtes de localisations et la hiérarchie non complète. Certains chercheurs ont proposé quelques métriques alternatives *localisation recul precision* [131]. Malgré ces récents changements, mAP est encore la métrique

la plus fréquemment utilisée pour la détection d'objets.

Au fil des années, les ensembles de données n'ont cessé de croitre aussi bien en nombre d'images qui les composent qu'en nombre d'objets référencés sur la totalité de l'ensemble. Cette croissance constante a permis l'essor des réseaux de neurones profonds.

On peut classer la détection d'objets en deux périodes successives, la première avant 2014 étant la période des détecteurs d'objets classiques et la deuxième après 2014 étant la période des détecteurs d'objets à base d'apprentissage profond comme présenté sur la figure 2.4.

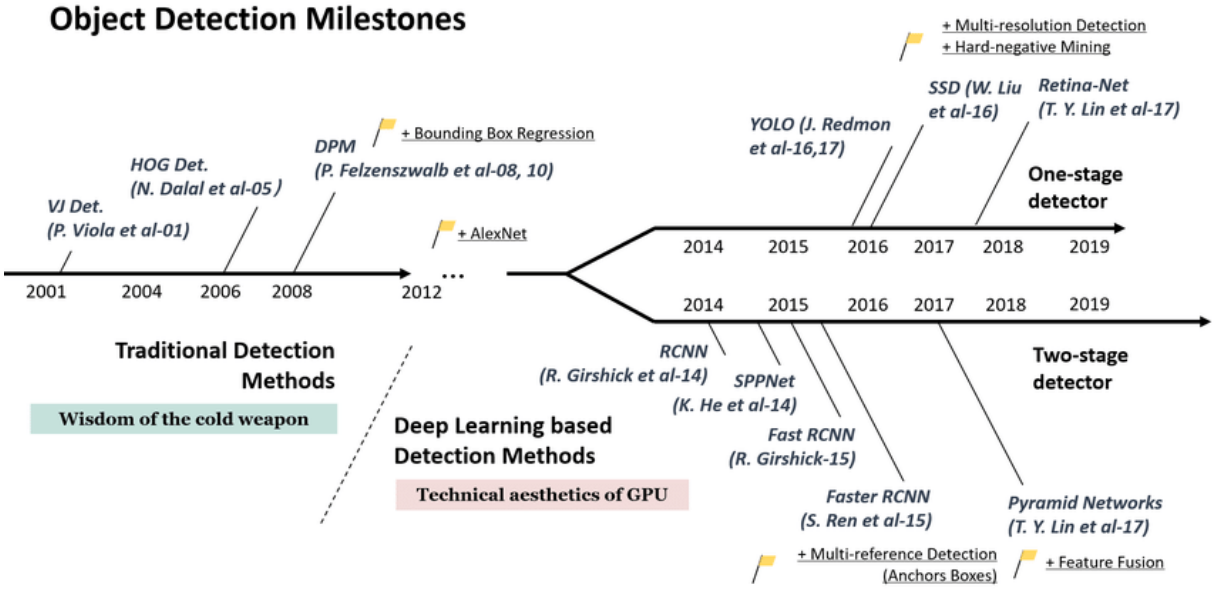


Figure 2.4: Les étapes clefs de l'évolution de la détection d'objet (Source [195])

2.1.3 Les détecteurs d'objets traditionnels

Au début des années 2000, la plupart des algorithmes de détections d'objets étaient basés sur des caractéristiques extraites à la main. Nous allons passer en revue les trois algorithmes majoritairement utilisés.

Les détecteurs Viola Jones

P. Viola et M. Jones ont réalisé le premier détecteur en temps réel de visages humains sans contraintes [170, 171]. Sur un processeur Pentium III 700Mhz, il était entre 10 et 100 fois plus rapide que les autres détecteurs avec la même précision. Il utilise la technique la plus directe qui consiste à avoir une fenêtre glissante sur l'image qui parcourt toutes les

positions et tailles possibles à la recherche d'un visage humain. Même si le processus parait simple, la puissance de calcul nécessaire n'était pas disponible à l'époque. Pour cela, ils ont dû réduire considérablement le temps de calcul nécessaire en utilisant 3 techniques différentes :

- *integral image* : c'est une méthode de calcul pour accélérer le *box filtering* et le processus de convolution. Elle rend la complexité de calcul de chaque fenêtre dans le détecteur indépendante de la taille de fenêtre [134, 135, 124].
- *feature selection* : les auteurs ont utilisé l'algorithme Adaboost [51] pour sélectionner un petit ensemble de caractéristiques qui sont principalement utiles pour la détection de visages à partir d'un vaste ensemble de caractéristiques aléatoires (environ 180 000 dimensions).
- *detection cascades* : un paradigme de détection en plusieurs étapes a été introduit dans le détecteur pour réduire la charge de calcul en réduisant les calculs sur les fenêtres d'arrière-plan et en se concentrant sur les visages.

Les détecteurs de HOG

Le descripteur de caractéristiques HOG (Histogram of Oriented Grdients) a été proposé en 2005 par N. Dalal et B. Trigges [26]. HOG est considéré comme une avancée significative pour la transformation de caractéristiques à échelle invariante [106, 107] et le contexte de forme. HOG peut être utilisé pour détecter une variété de classes, bien qu'à l'origine il était fait pour la détection de piétons. Afin de détecter des objets de tailles différentes, le détecteur HOG retaille l'image d'origine plusieurs fois tout en gardant la taille de la fenêtre de détection inchangée. Le détecteur HOG a été une base pour plusieurs autres détecteurs d'objets tels que les modèles *part-based* [46, 47] et les modèles *Exemplar-SVMs* [111] et pour une large variété d'applications de vision assistée par ordinateur pendant plusieurs années.

Deformable Part-based Model DPM

DPM a été le gagnant du Pascal VOC 2.1.1 en 2007/2008 et 2009. DPM a été proposé par P. Felzenszwalb en 2008 [46]. DPM est une évolution du détecteur HOG sur lequel un ensemble d'améliorations tels que les modèles *part-based* et les modèles *grammar* ont été

amenées par R. Girshick [47, 48, 56]. Le DPM suit la philosophie de détection du *divide and conquer*, où l'entrainement peut simplement être considéré comme l'apprentissage d'une façon simple de décomposer un objet, et son inférence peut être considérée comme un ensemble de détections de différentes parties d'un objet. Un détecteur DPM typique se compose d'un filtre racine et de plusieurs filtres de parties. A la place de spécifier manuellement la configuration des filtres des parties (taille et localisation), une méthode d'apprentissage supervisé est développée dans DPM où toutes les configurations de filtres de parties peuvent être apprises automatiquement comme variables latentes. Pour accélérer la détection, R. Girshick a développé une technique pour compiler les modèles de détections en un seul plus rapide qui implémente une architecture en cascade. Cela a permis de multiplier par 10 la vitesse d'exécution sans perdre en précision.

2.1.4 Les détecteurs CNN à deux étages

La détection d'objets atteint un plateau qui durera 3 ans. C'est en 2012 que nous assistons au retour des CNN [91] qui vont connaitre une amélioration nette de leur performance grâce à l'utilisation de cartes graphiques pour l'entrainement. En 2014, R. Girshirck et al. proposent *Regions with CNN* RCNN pour la détection d'objets [58, 57]. A partir de ce moment-là, la détection d'objets va évoluer à un rythme frénétique.

Avec l'utilisation des réseaux de neurones profonds, la détection d'objets peut être scindée en deux catégories :

- **les détecteurs à deux étages** qui divisent le processus de détection en proposition de région et en étape de classification. Dans la première étape, les modèles proposent d'abord plusieurs *region of interest* (RoI), en utilisant des boîtes de référence (ancres). Dans la deuxième étape, les propositions sont classées et leurs localisations sont affinées.

- **les détecteurs à un étage** contiennent un seul réseau *feed-forward* entièrement convolutionnel qui fournit directement les cadres de délimitation et la classification des objets.

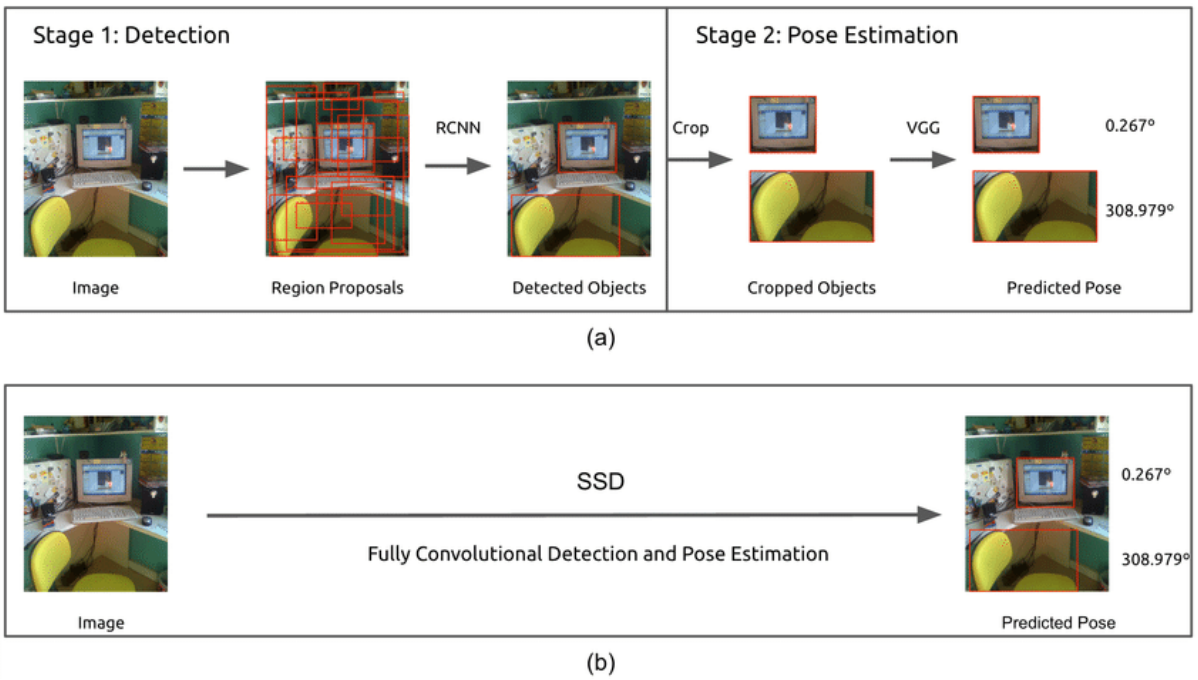


Figure 2.5: (a) CNN à 2 étages ; (b) CNN à 1 étage (Source [140])

RCNN

RCNN utilise la recherche sélective pour extraire un ensemble de propositions d'objets sur des délimitations d'objets peu nombreuses et précises car (1) un objet dont l'emplacement n'est jamais généré ne peut pas être reconnu et (2) l'apparence et le contexte immédiat à proximité sont les plus efficaces pour la reconnaissance d'objets [154]. Ensuite, chaque proposition est découpée avec une hauteur et largeur fixes pour pouvoir être utilisée dans un CNN entrainé sur ImageNet 2.1.1 (par exemple AlexNet [91]) qui en extrait les caractéristiques. Un classificateur linéaire SVM est utilisé pour prédire la présence d'un objet dans chaque région et classer cet objet dans sa catégorie. RCNN va permettre une amélioration significative de la précision moyenne en passant de 33.7% à 58.5% sur l'ensemble de données VOC07 2.1.1. Même si RCNN a réussi une avancée majeure, le revers de la médaille est évident : le calcul redondant d'extractions des caractéristiques sur un grand nombre de régions proposées qui se chevauchent (environ 2000 par image) engendre une réduction de la vitesse de détection. SPPNet [75] va régler ce problème.

SPPNet

En 2014, K. He et al. ont proposé Spatial Pyramid Pooling Networds (SPPNet) [75]. Les précédents modèles CNN requièrent une taille d'image fixe (224x224 pixels pour AlexNet

[91] par exemple). La principale contribution de SPPNet est l'introduction d'une couche Spatial Pyramidal Pooling (SPP), qui permet au CNN de générer une représentation de taille fixe quelle que soit la taille de la région, sans avoir à la retailler. Avec SPPNet, la carte des caractéristiques peut être obtenue sur l'image entière en une seule fois, ce qui permet d'éviter d'avoir à recalculer les caractéristiques. SPPNet est 20 fois plus rapide que RCNN sans perte de précision sur la détection. Néanmoins, il y a encore quelques inconvénients : l'entrainement est toujours sur plusieurs étages et SPPNet règle avec précision ses couches entièrement connectées alors qu'il ignore ses précédentes couches. Fast RCNN [55] va résoudre ces problèmes.

Fast RCNN et Faster RCNN

En 2015, Fast RCNN [55] est proposé par R. Girshick. Il s'agit d'une amélioration de RCNN et SPPNet [58, 75] qui introduit une nouvelle couche *ROI Pooling* qui permet de partager les calculs sur toutes les propositions plutôt que de faire les calculs pour chaque proposition indépendamment. Sur VOC07 2.1.1, Fast RCNN augmente la précision moyenne de 58.5% (RCNN) à 70% tout en multipliant la vitesse de détection par 200. La même année, S. Ren et al. ont proposé Faster RCNN [145, 146] peu de temps après Fast RCNN. Faster RCNN est le premier détecteur de réseaux de neurones profonds en temps réel. Sa principale contribution est dans l'introduction de Region Proposal Network (RPN) qui permet la proposition de régions à un coût presque nul. De RCNN à Faster RCNN, la plupart des blocs d'un système de détection d'objets ont été graduellement intégrés dans un seul système complet.

Feature Pyramid Networks FPN

En 2017, T.-Y. Lin et al. ont proposé Feature Pyramid Networks (FPN) [98] sur les bases de Faster RCNN. Avant FPN, la plupart des détecteurs basés sur des réseaux de neurones profonds ne détectaient que sur la dernière couche du réseau. Bien que les caractéristiques dans les couches les plus profondes d'un CNN soient bénéfiques pour la catégorisation, ce n'est pas propice à la localisation des objets dans l'image. Une architecture descendante avec des connexions latérales est développée dans le FPN pour construire des sémantiques de haut niveau à toutes les tailles. FPN, utilisé dans un système Faster RCNN, atteint les meilleurs résultats actuels sur l'ensemble de données MSCOCO 2.1.1. FPN est devenu

fondamental dans la construction des blocs des derniers détecteurs d'objets.

2.1.5 Les détecteurs CNN à un étage

You Only Look Once (YOLO)

YOLO a été proposé par R. Joseph et al. en 2015. Il a été le premier détecteur à un étage dans l'ère des réseaux de neurones profonds [144]. YOLO est extrêmement rapide, sa version rapide s'exécute à 155 images par seconde sur VOC07 2.1.1 avec une précision moyenne de 52.7%, alors que sa version évoluée s'exécute à 45 images par seconde avec une précision moyenne de 63.4% sur VOC07 2.1.1 et 57.9% sur VOC12 2.1.1. Sa philosophie est d'appliquer un unique réseau de neurones sur l'image complète. Le réseau divise l'image en plusieurs régions et prédit la boite de placement et la probabilité pour chaque région simultanément. R. Joseph proposera deux nouvelles versions qui augmenteront la précision tout en gardant une vitesse de traitement rapide [142, 143]. YOLO souffre malgré tout d'une chute dans la précision des coordonnées des boîtes de placement en comparaison aux détecteurs à deux étages surtout sur les objets de petite taille.

Single Shot MultiBox Detector (SSD)

SSD [101] a été proposé par W. Liu et al. en 2015. La principale contribution de SSD est l'introduction des techniques de détections multi-références et multi-résolutions qui ont significativement amélioré la précision des détections sur un détecteur d'un étage, spécialement sur les petits objets. SSD a l'avantage en termes de précision et de vitesse de détection. La principale différence entre SSD et n'importe lequel des précédents détecteurs est qu'il est capable de détecter des objets de différentes tailles sur différentes couches du réseau, là où ses prédécesseurs ne le pouvaient que sur les dernières couches.

Retina Net

Malgré leur vitesse élevée et leur simplicité, les détecteurs à un étage ont suivi la précision des détecteurs à deux étages pendant plusieurs années. T.-Y. Lin et al. ont découvert les raisons et ont proposé RetinaNet en 2017 [99]. Ils affirment que l'extrême déséquilibre des classes arrière-plan et premier plan, rencontré durant l'entraînement des détecteurs denses est la principale cause. Pour cela, une nouvelle fonction de perte nommée *focal loss* a été

introduite dans RetinatNet en modifiant la fonction standard entropie croisée pour que le détecteur se concentre davantage sur les exemples mal classés pendant l'entraînement. *Focal loss* a permis aux détecteurs à un étage d'avoir une précision comparable à celles des détecteurs à deux étages tout en maintenant une vitesse élevée.

***Fully Convolutional One-Stage* (FCOS)**

FCOS a été proposé en 2019 par Tian et al., c'est un détecteur d'objets convolutionnels en 1 étape pour prédire la classe des objets pixel par pixel. [163]. FCOS n'utilise pas d'algorithme de proposition de *region of interest* ce qui lui permet de réduire le temps de calculs ainsi que le nombre de paramètres. Il utilise un réseau de neurones entièrement convolutionnel, c'est à dire sans couche *Dense* [105]. FCOS prédit pour chaque pixel du premier plan un vecteur 4D contenant la localisation de l'objet.

Cette étude des détecteurs d'objets traditionnels et à base de réseaux de neurones profonds a montré la sur performance dans la catégorisation des objets des réseaux de neurones profonds qui ne cessent de s'améliorer depuis plusieurs années. Même si les performances qu'ils atteignent sont suffisantes pour les utiliser sur notre carte embarquée, le puissance de calcul nécessaire est un problème car celle dont nous disposons est extrêmement limitée. Dans notre contexte, les images que nous utilisons sont extraites d'un flux vidéo. Les algorithmes sur les vidéos permettent-ils d'atteindre une meilleure précision de catégorisation ?

2.2 Historique des travaux sur les vidéos

Depuis des années, l'usage de plus en plus grandissant de la vidéo sur Internet mais aussi au travers des réseaux sociaux a fait naitre de nouvelles problématiques et renforcé l'intérêt de la recherche dans ce domaine. Régulièrement, des enquêtes sont publiées pour mettre à jour l'état de l'art et l'avancée des travaux de recherche dans ce domaine [14, 157, 9, 29, 136, 1, 149]. Tout comme pour la détection d'objets dans une image, l'arrivée des réseaux de neurones profonds va permettre une amélioration nette sur les travaux de vidéos. Nous allons d'abord étudier les travaux avant l'apparition des réseaux de neurones profonds dans la section 2.2.1 au travers des approches basées sur le texte,

des approches basées sur l'audio et enfin les approches visuelles. Nous étudierons par la suite les approches avec réseaux de neurones profonds dans la dernière section 2.2.2.

2.2.1 Avant l'arrivée des réseaux de neurones profonds

La plupart des recherches sur la classification de vidéos ont porté sur des vidéos entières en les classant dans différentes catégories telles que des films ou certains types de sports [90] en 1999 mais aussi sur les journaux télévisés ou l'éducation [44] en 2004. Certaines recherches ont porté sur la classification de segments de vidéo pour permettre d'identifier si ils sont violents [127] ou effrayants [125]. En 2001, des travaux ont été menés pour découper des journaux télévisés en différents segments [193]. Afin de pouvoir classifier des vidéos, 3 types de caractéristiques peuvent être utilisées : texte, audio et visuelle.

Approches basées sur le texte

Cette méthode consiste à produire du texte depuis la vidéo puis à l'analyser pour le classifier. Le texte peut être soit 1) visible à l'image ou 2) extrait de la bande sonore [28]. Dans la première catégorie, le texte visible à l'image est extrait en utilisant un *optical character recognition* (OCR) [71] tel qu'un score lors d'un match, le numéro des joueurs sur leurs maillots, du sous-titrage ou de la description audio, etc. Un OCR permet d'extraire, depuis une image, un fichier texte comprenant les textes trouvés dans l'image en plusieurs phases : préanalyse de l'image pour éventuellement améliorer la qualité de l'image, segmentation en lignes et en caractères, reconnaissance proprement dite des caractères, post-traitement utilisant des méthodes linguistiques et contextuelles pour réduire le nombre d'erreurs de reconnaissance et enfin génération du format de sortie. Dans la deuxième catégorie, le texte est extrait depuis la bande audio en utilisant un moteur de reconnaissance vocale [174]. Le moteur de reconnaissance vocale permet de retranscrire automatiquement un son sous la forme d'un texte en 3 étapes : l'ordinateur capte le signal sonore et le numérise sous forme de vecteurs acoustiques (la modélisation du signal), l'ordinateur, via des calculs statistiques, déduit quel est le phonème (plus petite unité distinctive du sens) le plus probable (la lecture du signal) et l'ordinateur reconstruit un mot à partir de l'étape précédente et, en comparant avec un lexique de base, choisit le mot le plus approprié selon la syntaxe de la phrase (le choix du mot). Cette méthode est principalement utilisée pour proposer des sous-titres ou de la description audio. Ils sont

placés sur l'écran dans la langue choisie et permettent une meilleure compréhension. Les avantages d'une approche basée sur le texte est la facilité à comprendre la relation entre les caractéristiques extraites et le résultat produit.

Approches basées sur l'audio

L'approche basée sur l'audio est plus utilisée que celle sur le texte dans la recherche car elle nécessite moins de puissance de calcul et donc moins de temps de traitement car la taille des données en entrée est bien plus réduite. De plus, le stockage de l'audio et de ses caractéristiques nécessite moins d'espace que pour le texte ou la vidéo. Pour traiter l'audio, le signal est échantillonné à une fréquence particulière et, à partir de chaque échantillon, des caractéristiques physiques ou perceptibles sont extraites pour examen [104, 103]. Ces fenêtres d'échantillonnage peuvent se chevaucher dans certains cas. Les caractéristiques appropriées du signal échantillonné sont extraites en fonction des exigences de l'application. Les caractéristiques audio peuvent être classées en caractéristiques physiques ou en caractéristiques perceptibles [17].

Caractéristiques physiques

Aussi appelées caractéristiques bas niveau du signal, les valeurs d'amplitude du signal échantillonné sont directement utilisées pour calculer ses valeurs caractéristiques. Elles peuvent être : *root mean square* (RMS) [125], *zero crossing rate* (ZCR) [35], *Short Time Energy* (STE), *Spectral Roll-off*, *Spectrum Centroid*, *Spectral Flux*, *Fundamental Frequency*, *Mel-Frequency Cepstral Coefficient* (MFCC) [147].

Caractéristiques perceptibles

Cette méthode mesure les caractéristiques perceptives du son basées sur le système perceptif humain [179]. L'être humain interprète la piste audio en fonction de la perception de ce qu'il a entendu. Les caractéristiques sont le timbre, le volume et le ton.

Les caractéristiques audio sont extraites des signaux audio extraits des vidéos. Il est ainsi possible de classer les vidéos en fonction des différentes valeurs de ses caractéristiques. Par exemple, une voix féminine a une valeur de ton plus élevée que la voix masculine, la musique a une amplitude continue plus élevée que la parole.

Approches visuelles

Cette approche est la plus proche de celles de l'être humain. Elle peut être combinée avec l'approche texte et l'approche audio [14]. Les caractéristiques visuelles sont extraites des images de vidéo ou des plans de vidéo. La plupart des chercheurs ont utilisé des approches basées sur les plans de la vidéo car c'est la méthode la plus simple pour découper une vidéo. Ce n'est pourtant pas si simple d'extraire un plan d'une vidéo avec des méthodes automatiques [59]. Les caractéristiques visuelles sont souvent basées soit sur la couleur, soit sur le mouvement, soit la durée des plans. Elles peuvent fournir des informations sur les lumières, l'action, le fond d'écran et le rythme de la vidéo.

Caractéristiques basées sur la couleur

Chaque image d'une vidéo est composée de pixels appartenant à un espace colorimétrique [141]. Les deux espaces colorimétriques les plus utilisés sont RGB (Red, Green, Blue) et HSV (Hue Saturation Value). La distribution de la couleur dans une image est souvent représentée par un histogramme des couleurs. Il représente le nombre de pixels dans une image pour chaque couleur possible [164, 186]. La plupart du temps, les histogrammes sont utilisés pour comparer deux images. Mais l'inconvénient est que nous ne pouvons pas trouver le pixel exact pour une couleur particulière. Un autre problème peut être que les images de différentes scènes peuvent avoir des conditions d'éclairage différentes nécessitant un prétraitement pour réduire l'impact des conditions d'éclairages sur les images.

Caractéristiques basées sur la prise de vue

Il faut découper la vidéo en différentes séquences en cherchant les transitions entre les séquences. On utilise pour cela différentes techniques tel que *hard cut* qui représente un changement brusque de plan et donc d'intensité de couleur, *faded* qui représente une disparition du plan sur fond noir, *disolve* qui représente une transition entre la dernière image d'un plan et la première du plan suivant. La méthode la plus simple pour trouver des prises de vues consiste à calculer la différence d'histogrammes de couleur entre les images [2, 83, 175].

Caractéristiques basées sur les objets

Dans cette approche, les objets sont d'abord identifiés, ensuite les caractéristiques sont extraites des objets précédemment identifiés. Par exemple, les visages peuvent être détectés à partir d'une vidéo, puis des caractéristiques telles que le teint, la texture, la taille, la position sont extraites. Cette approche est la moins utilisée car il est difficile de

détecter et identifier des objets dans une vidéo [175, 187].

2.2.2 Réseaux de neurones profonds

Après avoir permis une percée dans la classification d'images, les réseaux de neurones profonds ont fait leur apparition pour la reconnaissance d'activité basée sur la vidéo. Bien que de grands progrès aient été réalisés et que des résultats de l'état de l'art aient été obtenus, le niveau de précision de classification des vidéos n'a pas encore atteint le niveau de précision de classification d'images. En plus des informations spatiales, la vidéo contient également des informations temporelles, ce qui est également important pour la compréhension du contenu vidéo. Des recherches considérables ont été menées sous deux aspects principaux : l'apprentissage supervisé et l'apprentissage non supervisé selon que les données sont étiquetées ou non.

Supervisé

Deux modèles représentent la grande majorité des recherches : *two-stream* CNN et 3D CNN que nous allons étudier par la suite.

***two-stream* CNN**

Proposé en 2014 [159], il modélise séparément les informations statiques et dynamiques du flux vidéo. Les caractéristiques statiques sont extraites des images vidéo et les caractéristiques dynamiques sont extraites des données de flux optique (Figure 2.6). Des chercheurs ont aussi travaillé sur un ensemble d'outils pour des réseaux de neurones profonds multimodaux utilisant différentes sources d'informations telles que les images, l'audio et les informations textuelles [162].

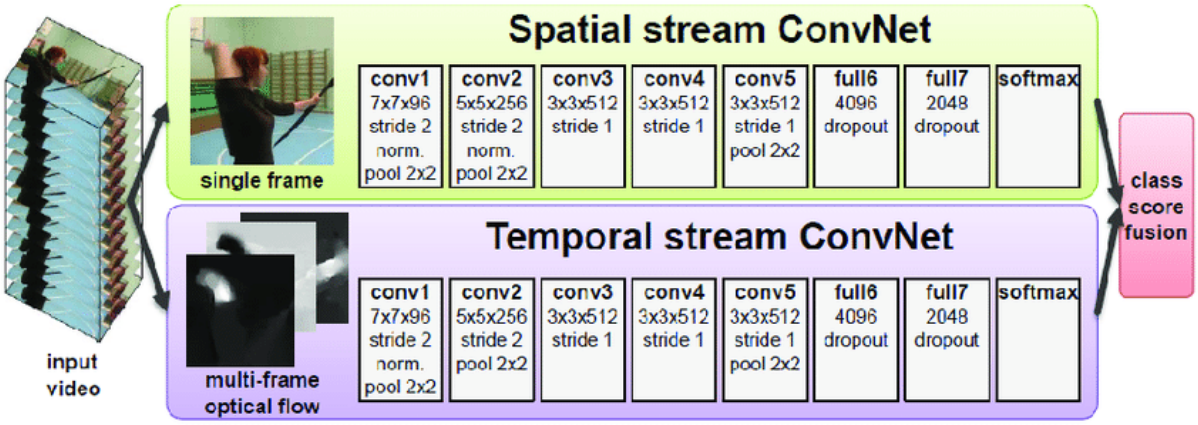


Figure 2.6: Architecture d'un modèle *two-stream* CNN (Source [155])

La précision du modèle *two-stream* CNN a dépassé celle des modèles avec des caractéristiques manuelles pour la première fois sur les ensemble de données UCF101[6] et HMDB51[7]. Une évolution du modèle *two-stream* CNN nommé *temporal segment networks* (TSN) [173] va diviser la vidéos en plusieurs segments, chaque segment est échantillonné et classé, et les résultats de la classification finale sont obtenus en fusionnant les scores de classification de chaque segment, réduisant ainsi la puissance de calcul nécessaire. Une autre évolution du *two-stream* CNN introduit un mécanisme d'attention [138], lors du processus d'extractions des caractéristiques statiques et dynamiques, les régions clés et les images clés sont situées séparément pour obtenir des caractéristiques plus représentatives pour améliorer la précision.

3D CNN

3D CNN est une extension d'un modèle CNN dans un espace tri-dimensionnel proposé en 2013 qui empile plusieurs images dans un cube et utilise un noyau de convolution 3D pour pouvoir modéliser des informations spatio-temporelles [85]. Dans cette structure, chaque carte de caractéristiques de la couche de convolution est connectée à plusieurs images adjacentes dans la couche supérieure, de sorte que les caractéristiques obtenues contiennent des informations de mouvement (Figure 2.7).

[6]ensemble de données de reconnaissance d'action depuis des vidéos, collectées à partir de YouTube, comportant 101 catégories d'action. https://www.crcv.ucf.edu/data/UCF101.php

[7]ensemble de données de vidéos provenant de diverses sources composé de 6 849 clips vidéo de 51 catégories d'action. https://paperswithcode.com/dataset/hmdb51

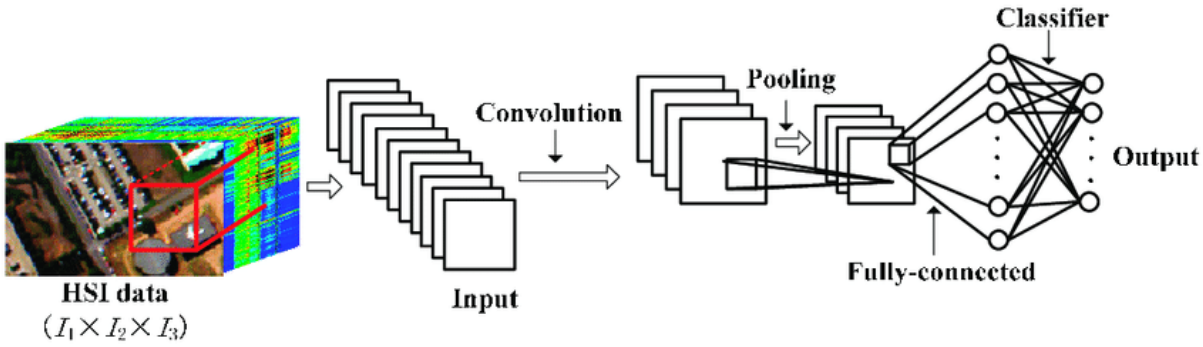


Figure 2.7: Architecture d'un modèle 3D CNN (Source [102])

Inspiré par l'utilisation réussie des réseaux de neurones récurrents dans les domaine du traitement du langage pour le traitement des informations de séquence, les chercheurs ont appliqué des réseaux de neurones récurrents pour traiter les données vidéo. La sortie du modèle CNN est connectée sur l'entrée d'un modèle *Long short term memory* (LSTM) en séries temporelles pour la modélisation de séries temporelles, ce qui confirme la faisabilité de l'utilisation du LSTM pour réaliser l'agrégation des caractéristiques vidéo [128]. En plus des méthodes ci-dessus, les chercheurs ont également combiné certains modèles pour obtenir de meilleurs résultats de classification tels que le modèle I3D [18], combinaison du modèle *two-sources* CNN avec le modèle 3D CNN. Bien que la convolution 3D puisse extraire des informations temporelles par elle-même, l'ajout d'informations sur le flux optique peut améliorer les performances.

Long-term Temporal CNN

Le modèle Long-term Temporal CNN a été proposé en 2017 par Gül Varol et al. [167]. Ce modèle utilise des réseaux de neurones avec des convolutions temporelles à long terme pour apprendre de la représentation des actions dans la vidéo. Il permet une amélioration de la précision atteignant 92.7% sur UCF101 et 67.2% sur HMDB51. Le réseau a 5 couches convolutionnelles spacio-temporelles avec des filtres de cartes de réponse en 64, 128, 256, 256 et 256, suivies de 3 couches entièrement connectées de tailles 2048, 2048 et le nombre de classes. *Dropout* est utilisé pour les deux premières couches entièrement connectées. Les couches entièrement connectées sont activées avec ReLU. La couche Softmax à la fin du réseau génère des scores de classes.

Siamese Spatio-Temporal Convolutional Neural Network

Le modèle SSTCNN a été proposé en 2020 par Pierre-Etienne Martin et al. [113]. Le modèle est constitué de 2 branches avec 3 couches de convolutions 3D avec des filtres

de cartes de réponse en 30, 60 et 80, suivies par une couche entièrement connectée de taille 500. Les deux branches sont combinées à l'aide d'une interpolation bilinéaire des caractéristiques, réunies dans une deuxième couche entièrement connectée de taille 21 (correspondant au nombre de classes considérées). Une couche Softmax est finalement appliquée à la fin pour obtenir un score de classement. Le modèle est évalué sur l'ensemble de données TTStroke-21, composé de vidéos de parties de tables de tennis. Ce modèle a permis d'atteindre une précision de 91,4% sur l'ensemble de données TTStroke-21.

Three-Stream 3D/1D CNN

Le modèle Three-Stream 3D/1D CNN a été proposé en 2021 par Pierre-Etienne Martin et al. [114]. L'utilisation d'informations sur les positions, fusionnée avec des informations de couleurs RVB et le flux optique permet une augmentation des performances (jusqu'à 18%) dans le classement commun ainsi que dans la tâche de segmentation. L'architecture est inspirée des travaux sur *Twin Spatio-Temporal Convolutional Neural Network - T-STCNN with attention mechanisms* [112] qui prend en entrée les valeurs du flux optique et RVB à travers deux branches en utilisant des convolutions 3D et des mécanismes d'attention. Par rapport aux travaux précédents, le réseau compte trois branches et prend en entrée des coordonnées articulaires supplémentaires. De plus l'étape de la fusion est adaptée pour fusionner les trois modalités. Comme le montre la Figure 2.8, les réseaux réalisent une convolution 3D (spatio-temporelles) et une convolution 1D (temporelles).

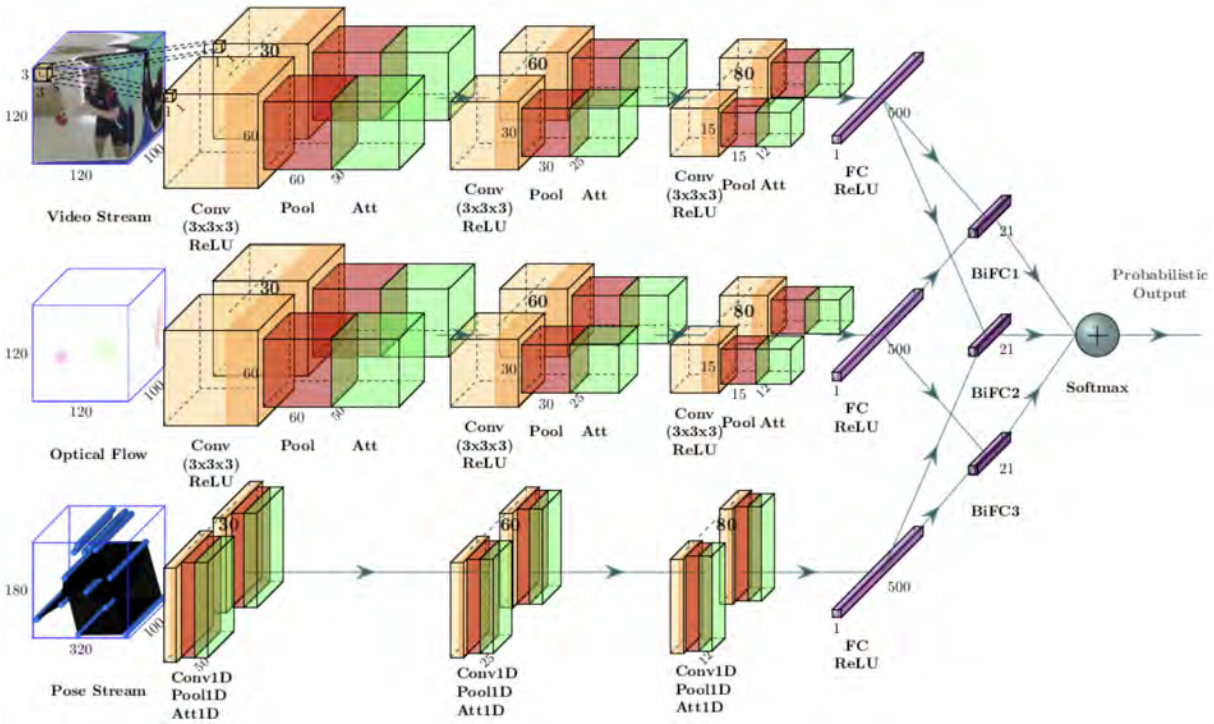


Figure 2.8: Architecture d'un modèle Three-Stream 3D/1D CNN (Source [114])

Multiview Transformers

Le modèle a été proposé en 2022 par Shen Yan et al. [185]. Le modèle est basé sur l'extension pour la vidéo des transformateurs de vision ViViT [6]. Le modèle construit différentes "vues" de la vidéo d'entrée en extrayant des jetons de *tubelets* spatio-temporels de dimensions variables. Ces jetons sont ensuite traités par un transformateur multivues, qui intègre des connexions latérales pour fusionner efficacement les informations provenant de plusieurs tailles. Le modèle a été évalué sur les ensembles de données Kinetics [88], Moments in Time [126], Epic-Kitchens-100 [27] et Something-Something V2 [61]. Cette approche fonctionne mieux, en terme de compromis précision/calcul qu'en augmentant la profondeur des architectures actuelles à vue unique. De plus, ce modèle obtient des résultats de l'état de l'art sur six ensembles de données de classification vidéo populaires.

Non supervisé

La classification non supervisée, également connue sous le nom d'analyse de cluster, consiste à mesurer la similitude des caractéristiques, et les attributs des catégories sont déterminés par interprétation visuelle ou analyse des clusters après la classification [183]. Par rapport aux caractéristiques obtenues par la méthode d'apprentissage supervisé, les caractéristiques obtenues par la méthode non supervisée sont plus faibles, mais peuvent

utiliser un plus grand nombre de vidéos pour l'entrainement [189]. Il est difficile d'étendre un modèle de classification supervisé avec de nouvelles catégories. La méthode *zero-shot* permet de traiter ce problème [54, 190]. C'est une méthode de prédiction basée soit sur les attributs d'objets ou soit sur les vecteurs de mots. Il permet d'établir le lien entre les étiquettes connues et celles non connues en mesurant les similarités dans le même espace entre les caractéristiques d'une nouvelle catégorie par rapport à l'ensemble des caractéristiques issues du modèle de classification supervisé. La précision des modèles n'est pas la seule priorité, des recherches ont été menées pour essayer de réduire la taille sans pour autant diminuer les performances. Ce modèle peut être réalisé en appliquant à plusieurs reprises des opérations matricielles de faible poids à toutes les images de la vidéo [96]. Ainsi un gain en mémoire vive est réalisé mais les opérations de calcul sur les flottants restent gourmands en ressources. Des chercheurs ont entrainé un modèle *teacher-student*, un ordinateur *teacher* avec une grande capacité de calcul qui regarde toutes les images de la vidéo est utilisé pour former un modèle *student* qui ne regarde qu'une petite fraction des images de la vidéo, réduisant de 90% le temps de calcul sur les flottants et sans dégradation des performances sur la précision [8].

L'étude des travaux dans la classification des vidéos a montré l'utilisation de différents modèles qui permettent de classer un segment ou une vidéo complète. Ces modèles n'ont que peu d'intérêt dans notre contexte où nous souhaitons catégoriser des objets et non des vidéos complètes. Pour autant, ces modèles utilisent la dimension temporelle fournie par la vidéo pour mieux la catégoriser. Nous allons étudier ces différents modèles spécifiques pour les séries temporelles non plus dans le contexte vidéo mais dans un contexte général.

2.3 La dimension temporelle

La catégorisation et la prédiction de séries temporelles est un sujet sur lequel les chercheurs travaillent depuis des décennies [110]. Les séries temporelles peuvent être définies comme tout ce qui est observé séquentiellement dans le temps. Par exemple tendances des ventes, cours boursiers, prévisions météorologiques, etc. Ces observations peuvent être des nombres, des étiquettes, des couleurs et bien plus encore. De plus, la durée d'espacement entre chaque observation peut être régulière ou irrégulière. D'autre part, le temps peut être

discret ou continu [70, 109]. Tout comme dans le domaine de la classification d'images ou de vidéos, l'arrivée des réseaux de neurones profonds a permis une grande percée. Nous allons d'abord étudier les modèles classiques 2.3.1 et ensuite les méthodes basées sur les réseaux de neurones profonds 2.3.2.

2.3.1 Les modèles classiques

Modèle *Autoregressive Moving Average* (ARMA)

C'est un modèle adapté à la modélisation de séries temporelles à un seul paramètre proposé en 1990. Il est combiné à l'aide d'un modèle AR(p) [165] et d'un modèle MA(q). Un modèle AR(p) utilise la relation de dépendance entre une observation et son décalage par rapport à d'autres observations. Un modèle MA(q) utilise la relation de dépendance entre une observation et une erreur résiduelle d'un modèle de moyenne mobile appliqué aux observations en décalage.

Modèle *Autoregressive Integrated Moving Average* (ARIMA)

Modèle proposé en 1994, il est le modèle le plus populaire et le plus fréquemment utilisé en modélisé de séries temporelles [165, 191]. L'hypothèse de ce modèle est basé sur le fait que la série temporelle est linéaire et suit une distribution statistique connue.

Modèle *Seasonal Autoregressive Integrated Moving Average* (SARIMA)

Le modèle ARIMA est destiné aux données non stationnaires et non saisonnières. [165]. Il est généralement utilisé pour traiter des problèmes de saisonnalité. Dans le modèle, la différenciation saisonnière des valeurs correctes est utilisée pour éliminer les valeurs non saisonnière dans la série.

2.3.2 Les modèles basés sur des réseaux de neurones profonds

Le modèle *Convolutional Neural Network* (CNN)

Les réseaux de neurones convolutifs (CNN), inspirés des études du cortex visuel chez les mammifères, est un type de réseau de neurones acycliques. Motivé par le succès de ses architectures (CNN) dans différents domaines tels que la classification d'image 2.1.4 et de vidéo 2.2.2, les chercheurs ont commencé à les adopter pour l'analyse des séries

temporelles [53, 21]. La convolution sur une série temporelle permet l'application et le glissement d'un filtre.

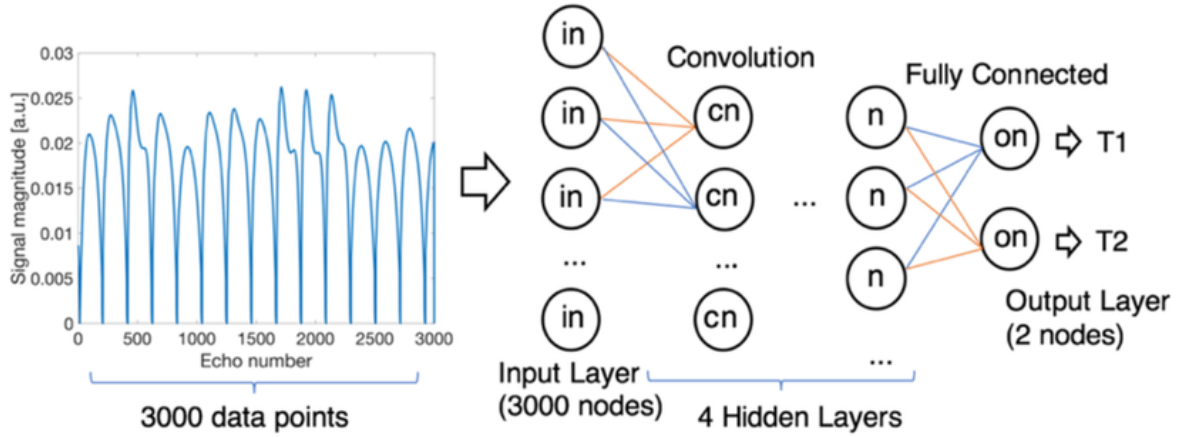


Figure 2.9: Architecture d'un CNN (Source [81])

Le modèle *Recurrent Neural Network* (RNN)

Les modèles RNN sont adaptés aux données à variables multiples de séries temporelles, capables de capturer des dépendances temporaires sur des périodes variables [133, 19, 132, 63]. Les RNN ont été utilisés dans de nombreuses applications de séries temporelles, telles que la reconnaissance [62] en 2014, la prévision de la charge électrique [172] et la pollution de l'air [60] en 2003.

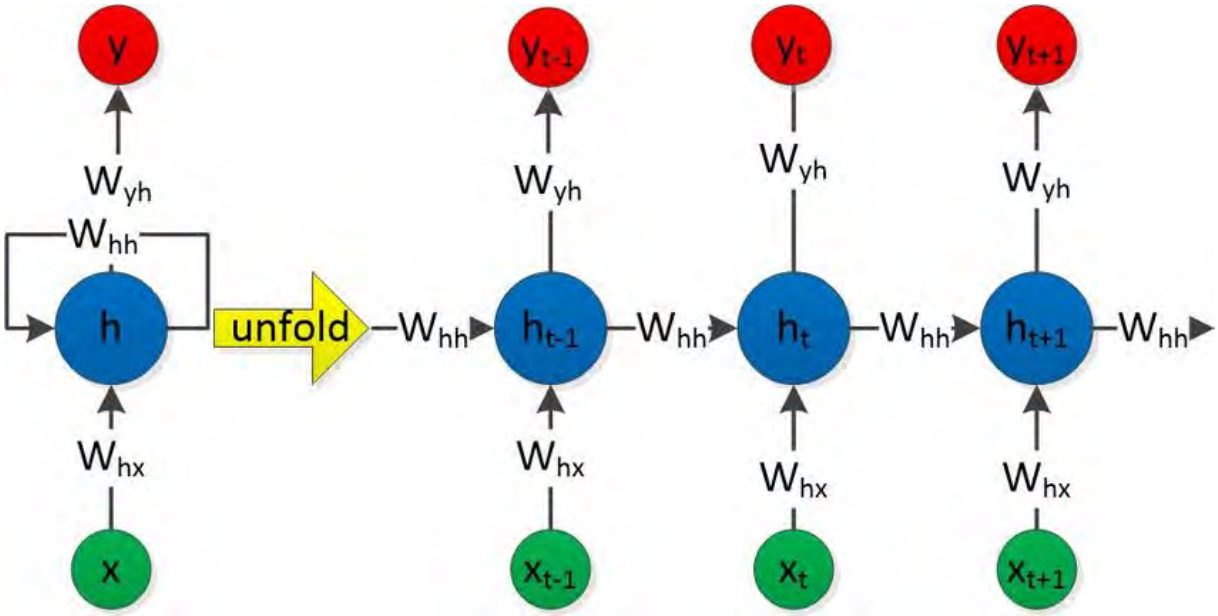


Figure 2.10: Architecture d'un RNN (Source [192])

Le modèle RNN a un état interne qui est renvoyé à son entrée. Il utilise les informations actuelles sur l'entrée et la prédiction de la dernière entrée. Ils ont une entrée et une sortie de taille fixe. L'algorithme de rétropropagation peut être adapté pour former un réseau récurrent en étalant le réseau à travers le temps et en restreignant certaines des connexions pour garder le même poids à tout moment [151]. Le modèle RNN peut encoder les informations antérieures dans la couche cachée pendant le processus d'apprentissage afin que les données de la série temporelle puissent être apprises efficacement [184].

Le modèle *Long Short-Term Memory* (LSTM)

Le modèle LSTM, proposé en 1997, s'est avéré efficace pour traiter divers problèmes tels que la parole et la reconnaissance de l'écriture car il a la capacité d'apprendre à partir des entrées que le délai entre les évènements importants soit long ou court [80]. Les unités LSTM se composent d'une cellule et de trois portes : la porte d'oubli, la porte d'entrée et la porte de sortie. La figure 2.11 montre l'architecture d'une unité LSTM. La cellule permet de maintenir un état aussi longtemps que nécessaire. Cette cellule consiste en une valeur numérique que le réseau peut piloter en fonction des situations

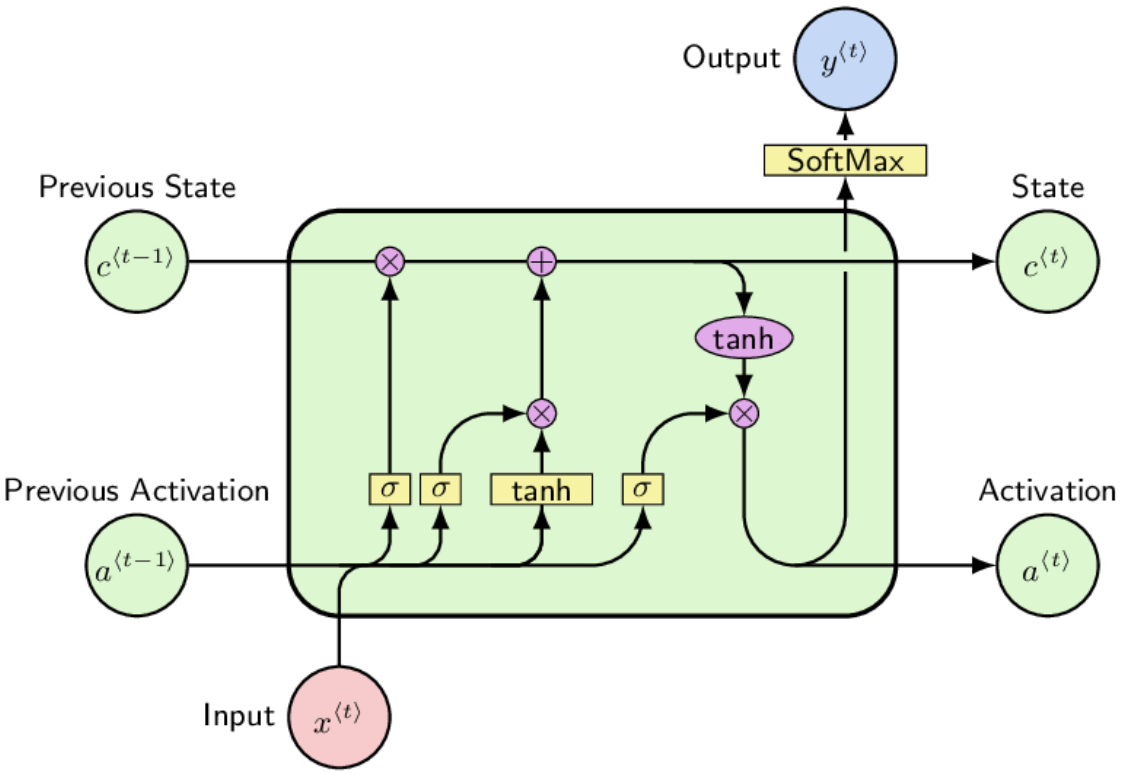


Figure 2.11: Architecture d'un LSTM (Source [52])

Le modèle *Gated Recurrent Unit* (GRU)

Le modèle GRU est très similaire au modèle LSTM à la différence qu'il n'y a que deux portes au lieu de trois : la porte *reset* et la porte *update* [22]. La porte *reset* détermine comment une nouvelle entrée et la précédente mémoire sont combinées. La porte *update* détermine la quantité d'informations qui doit être gardée de la précédente entrée. Parce que le modèle GRU a moins de paramètres que le modèle LSTM, il sera plus rapide et nécessitera moins de données. Cependant, si la quantité de données est grande, le modèle LSTM pourrait obtenir un meilleur résultat.

Le modèle *Restricted Boltzmann Machines* (RBM)

Paul Smolensky a introduit ce modèle en 1986 sous le nom "Harmonium" [152]. Le modèle RBM est un réseau de neurones stochastique [49]. Depuis 2002, c'est devenu l'un des modèles d'apprentissage de réseaux de neurones profonds les plus populaires en raison de sa capacité à apprendre en mode supervisé et non supervisé [78]. Il est utilisé dans une large gamme d'applications telles que la réduction de dimension [79], la prédiction de problèmes [95] et la représentation d'apprentissage [24].

Le modèle *Deep Belief Networks* (DBN)

Le modèle DBN est basé sur des méthodes génératives qui permettent de modéliser toutes les variables d'un problème sous la forme d'une distribution de probabilité incluant les informations sur la structure des données grâce à tous les outils de modélisation non-supervisés. Une modèle DBN se compose de plusieurs modèles RBM empilés entrainés couche par couche dans un environnement non supervisé pour obtenir des performances robustes [77]. L'apprentissage dans un modèle DBN est effectué couche par couche, chacune étant exécutée comme un modèle RBM formé au-dessus de la couche précédemment formée (les DBN sont un ensemble de couches RBM qui sont utilisées dans la phase de pré-apprentissage et deviennent ensuite un réseau acyclique lors du réglage fin) [130].

Le modèle *Deep Autoencoder* (AE)

Le modèle AE est un réseau de neurones d'apprentissage non supervisé formé pour reproduire son entrée à sa sortie [76] proposé en 2006. Il est composé de deux réseaux

symétriques DBN qui ont généralement quatre ou cinq couches peu profondes. Le premier DBN représente la moitié du codage du réseau, et le deuxième représente la moitié de décodage. Le modèle a une couche cachée h, qui définit un code utilisé pour représenter son entrée.

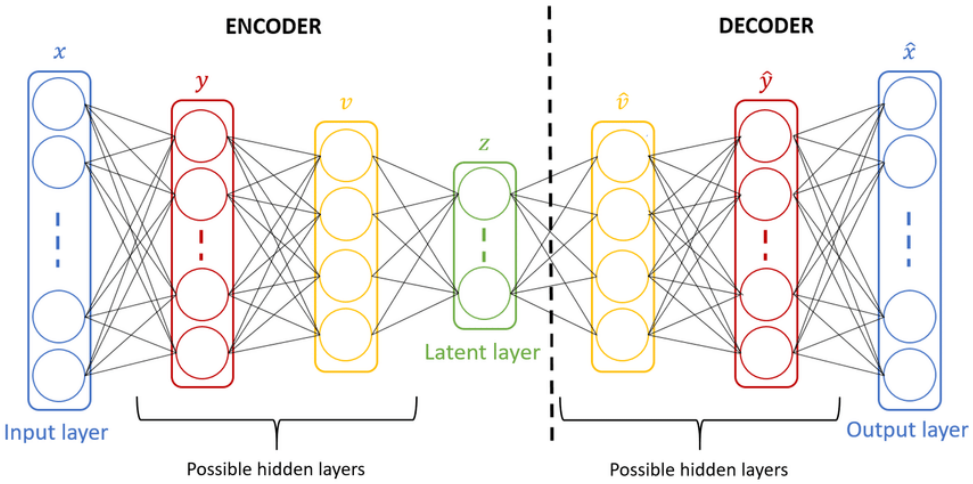


Figure 2.12: Architecture d'un AE (Source [148])

Cette étude a une fois de plus mis en avant les performances des réseaux de neurones face aux algorithmes traditionnels dans la prédiction des séries temporelles. Dans notre contexte, l'information temporelle est primordiale car elle va permettre d'obtenir des informations supplémentaires telles que la vitesse d'un objet ou sa provenance, facilitant ainsi sa catégorisation. De plus, les réseaux de neurones récurrents n'utilisent en entrée que très peu d'informations à la différence des réseaux de neurones profonds sur les images permettant ainsi une utilisation en temps réel sur notre carte embarquée à faible puissance de calcul.

2.4 Positionnement des travaux de thèse

La plateforme matérielle retenue par le directeur technique est la carte NXP IMX6. Plusieurs raisons ont appuyé ce choix tel que sa consommation, son tarif, sa disponibilité, sa plage de température d'utilisation, sa taille, etc. L'ensemble d'outil logiciel retenu est TensorFlow pour ses performances, sa communauté ainsi que sa version légère embarquée qui permettra de réduire la consommation en puissance de calcul et mémoire des modèles sur la carte NXP IMX6. Bien qu'il soit aujourd'hui possible d'atteindre des

taux de précision proches de la perfection avec des réseaux de neurones profonds, son utilisation sur des images en temps réel reste encore trop gourmande en puissance de calcul pour notre plateforme. Seulement 2 détections à la minute seraient envisageables alors que nous en souhaitons à minima 10 par seconde. Nous n'avons pas trouvé d'algorithme qui nous permette de tirer parti des avancées de l'intelligence artificielle sur notre plateforme matérielle tout en répondant à nos contraintes temps réel.

Afin de combler cette lacune et atteindre notre objectif, nous avons développé une architecture 3 qui, plutôt que d'utiliser des images en entrée (représentant donc une énorme quantité d'informations et nécessitant une puissance de calcul adéquate), va étudier les informations des objets en mouvement (taille, position, précédentes positions) permettant ainsi de classifier en temps réel les personnes en mouvement sur des architectures matérielles embarquées. Le chapitre 4 traitera d'une évolution d'un ensemble d'outil (CGP-IP) pour favoriser l'évolution génétique sur un filtre de traitement de l'image créé par des experts du traitement du signal. Le chapitre 5 proposera une évolution des travaux réalisés dans le chapitre 4 mais dans un contexte multi-objectifs afin de pouvoir maintenir un niveau de précision acceptable tout en réduisant drastiquement les temps de calcul du filtre de traitement de l'image. Pour finir, nous présenterons dans le chapitre 6 nos différentes conclusions sur les travaux menés ainsi que les pistes à suivre dans l'avenir.

Chapter 3

Détection des humains et véhicules dans une vidéo

La société Kawantech souhaite faire évoluer son logiciel de détection d'objets en mouvement, embarqué sur ses cartes électroniques pour utiliser des algorithmes d'intelligence artificielle. Les cartes électroniques sont situées dans des luminaires et permettent une gestion intelligente de la puissance du luminaire en fonction des personnes et des véhicules en mouvement sous le luminaire.

Comme étudié dans l'état de l'art, les réseaux de neurones profonds qui offrent une précision élevée, nécessitent une puissance de calcul conséquente qui ne permet pas une utilisation en temps réel sur les cartes embarquées de Kawantech.

Des travaux récents ont été menés sur des cartes électroniques embarquées avec des *Graphics Processing Unit* GPU permettant l'utilisation de réseaux de neurones profonds tel qu'un système de détection d'objets basés sur un réseau de neurones profonds MobileNet-SSD sur une plateforme Nvidia Jetson TK1[1] [129], un système de détection d'objets en temps réel basé sur un réseau de neurones profonds pour des cartes électroniques embarquées avec un GPU NVidia ou Intel pour une utilisation dans des drones [82] et enfin un système de détection et traçage d'objets basé sur des réseaux de neurones profonds pour des cartes électroniques embarquées à base de NVidia Jetson TX2[2] pour des applications *Internet of Things* IOT [12]. De plus, des travaux récents existent avec des algorithmes basés sur l'extraction de caractéristiques tel qu'un système temps réel

[1]https://developer.nvidia.com/embedded/jetson-tk1-developer-kit
[2]https://www.nvidia.com/fr-fr/autonomous-machines/embedded-systems/jetson-tx2/

de détection des piétons basé sur un algorithme HOG dans des images thermiques pour des cartes électroniques embarquées [150] ou bien un algorithme robuste d'extractions de caractéristiques pour des applications sur des cartes électroniques embarquées à base de FPGA [3].

Nous proposons une nouvelle approche pour un système autonome qui combine un réseau de neurones profonds avec un réseau de neurones récurrents. Ce système est composé de deux applications et sera capable d'apprendre les spécificités de son environnement pour catégoriser les objets en mouvement avec une puissance de calcul limitée. Le diagramme du système autonome est représenté sur la Figure 3.1.

L'application principale va tout d'abord détecter les objets en mouvement dans le flux vidéo en appliquant des filtres d'images décrits dans la Section 3.1 et va ensuite catégoriser les objets détectés précédemment à l'aide d'un réseau de neurones. Cette application de catégorisation est décrite dans la Section 3.3.

Chaque luminaire a un point de vue différent qui varie en fonction de sa hauteur, de l'angle de la caméra mais aussi de son environnement (un carrefour, une place, une voie d'autoroute, etc.). Afin de pouvoir fonctionner correctement dans chaque environnement, le système doit d'abord apprendre les spécificités de son environnement final, pour ensuite pouvoir catégoriser les objets en déplacement. Pour cela, une application d'apprentissage décrite dans la Section 3.2 est nécessaire à la première mise en route du luminaire une fois son installation réalisée sur la voie publique.

Chaque objet en mouvement est assigné à une trace. Une trace correspond au chemin parcouru par un objet (vélo, voiture, bus, camion, etc) depuis la première fois où il a été identifié dans le flux vidéo. Une trace est une liste énumérant la position et la taille d'un objet pour chaque image où l'objet est en mouvement. Une trace peut donc identifier une personne ou un véhicule en mouvement. Une trace permet de connaitre l'historique d'un objet depuis son apparition à l'image. L'utilisation des traces et non pas seulement des blobs permet de disposer d'informations supplémentaires primordiales telles que la destination, la vitesse et la provenance des objets qui permettront l'entrainement d'un réseau de neurones récurrents.

Les résultats et les paramètres du système utilisé sur 3 expérimentations différentes seront présentés dans la Section 3.4.

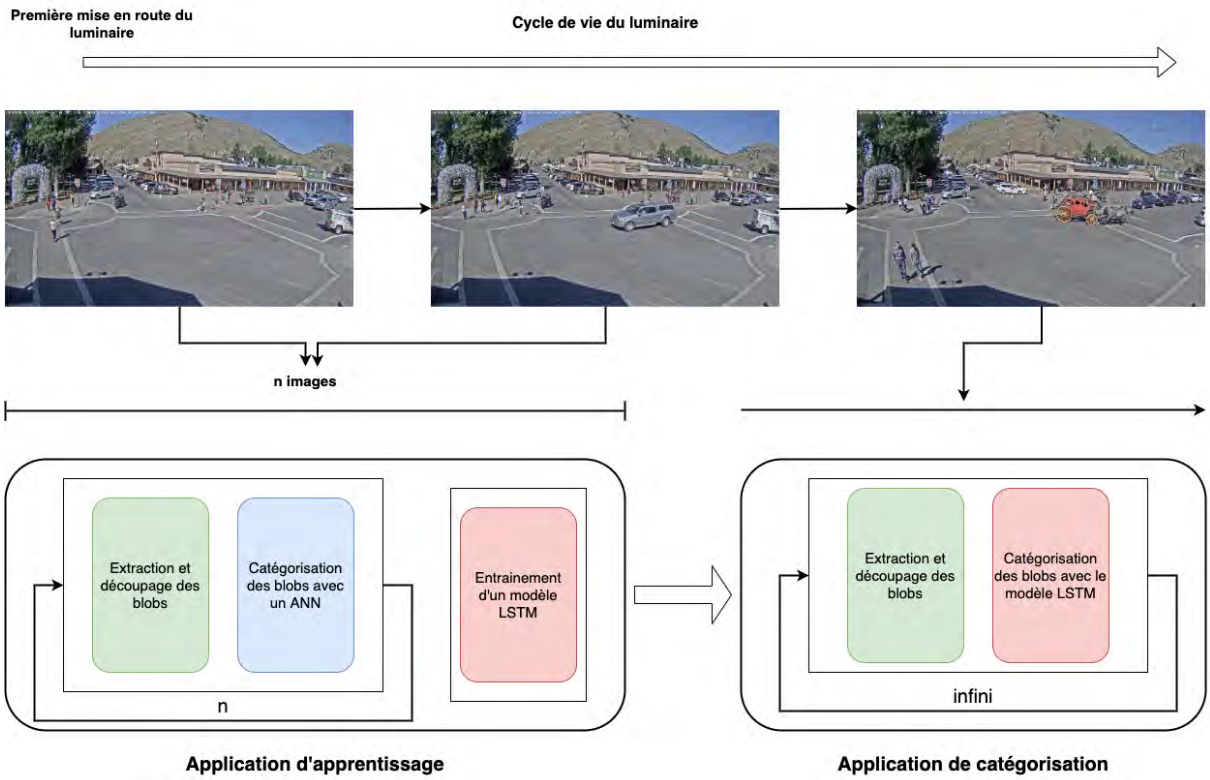


Figure 3.1: Diagramme du système complet

3.1 Fonctions communes

Le pipeline d'extraction des blobs en mouvement et de la mise à jour de la liste des traces est commun aux deux applications. Il est réalisé au travers de 5 fonctions communes aux deux applications : extraction des blobs en mouvement de la vidéo, filtrage et découpage des blobs extraits et mise à jour de la liste des traces.

3.1.1 Extraction des blobs en mouvement

La fonction permet d'extraire les blobs en mouvement de la vidéo. Le flux vidéo est fourni par un driver système. L'image précédente (Figure 3.2.A) est soustraite à l'image actuelle (Figure 3.2.B) pour obtenir une image différentielle (Figure 3.2.C). Dans la Figure 3.2.C, l'arrière plan est représenté par des pixels noirs alors que les zones grises et blanches correspondent aux blobs en mouvement depuis la précédente image.

$$image = image_{precedente} - image_{courante} \tag{3.1}$$

A : Image précédente B : Image actuelle C : Image différentielle

Figure 3.2: Blobs en mouvement depuis la précédente image

3.1.2 Extraction et filtrage des blobs depuis l'image

La fonction permet de supprimer les bruits de l'image différentielle en supprimant les zones dont l'aire est trop réduite.

Algorithm 1: Suppression des blobs d'aire trop réduite

```
Data: blobs = liste des blobs détectés
Result: liste des blobs maintenus
for i ← 0 to blobs.length do
    aire = blobs[i].width * blobs[i].height;
    if aire<min_area_blob then
        delete blobs[i];
    end
end
```

La fonction *findCoutours* de la librairie OpenCV permet de récupérer la liste des contours depuis une image. Elle est utilisée pour récupérer la liste des blobs. Chaque blob est défini par ses coordonnées ainsi que sa taille. L'aire du blob est calculé et le blob est supprimé de la liste si l'aire est inférieure à **min_area_blob** pixels (Algorithme 1). **min_area_blob** est un paramètre du système. La valeur de l'aire minimum d'un blob tient compte de la hauteur du luminaire, de son angle de vue et de la résolution de la vidéo. Il y a deux raisons pour lesquelles les blobs de trop petite taille sont supprimés. La première est que si le blob (donc l'objet) est au premier plan, une petite aire signifie que l'objet est de taille très réduite ce qui exclut des personnes ou objets de taille supérieure (vélo, voiture, etc.). La seconde est que si le blob est en arrière plan, une petite aire représente un objet dont les détails seront trop réduits pour pouvoir le catégoriser correctement.

3.1.3 Découpage des blobs

Comme le montre la Figure 3.3, deux objets en mouvement bien distincts dans un environnement en trois dimensions peuvent ne sembler qu'un seul blob en mouvement lorsqu'on

les réduit dans un environnement en deux dimensions.

Image originale Blobs

Figure 3.3: 2 personnes en mouvement reconnues comme un seul objet en mouvement

Le problème d'occlusion est un problème récurent sur les systèmes vidéos de surveillances ou de suivis des personnes. Si deux objets en mouvement ou plus sont inclus dans un même blob, les informations de celui-ci telles que sa position et sa taille sont erronées et produiront une mauvaise classification par le réseau de neurones récurrents. Depuis le début des années 2000, de nombreux travaux ont été entrepris pour proposer différentes solutions telles qu'un réseau bayésien dynamique proposé en 2003 par Ying Wu et al. [182], un *framework* utilisant des caractéristiques simples telles que la position et la taille fournies par les boites englobantes proposé en 2005 par Yongtae Do [37] ou encore deux nouvelles techniques proposées par Yongtae Do en 2008 [36] dont la première utilise un modèle pour distinguer les êtres humains entre eux et un réseau de neurones probabiliste est utilisé dans la deuxième.

Nous proposons une approche nouvelle basée sur les propriétés convexes des polygones extraits des objets en mouvement. La grande majorité des objets en mouvement (voitures, camions, bus, piétons, vélos, motos) que nous souhaitons suivre ont une forme géométrique rectangulaire comme pour les camions avec dans certains cas des extrémités convexes

comme pour les voitures. Si deux objets en mouvement sont fusionnés dans le même blob, le polygone formé sera alors composé de deux polygones de taille inférieure. Notre algorithme vise à rechercher si l'un des bords du polygone est convexe privilégiant ainsi l'hypothèse d'une occlusion de deux blobs entre eux. Dans les cas où l'occlusion est complète ou presque, notre algorithme échouera à partager les deux blobs.

Pour cela, une nouvelle image ayant la taille du blob ainsi que le contenu du blob est créée. Les projections sur l'axe des abscisses et sur l'axe des ordonnées du blob permettent d'identifier si le blob est composé d'une seule ou de plusieurs masses.

Processus sur l'axe des abscisses

La courbe *projectionX* représentant la somme des intensités en niveau de gris d'une même colonne x est tracé sur l'image. Elle correspond à la courbe rouge sur la Figure 3.4.

$$projectionX(x) = \sum_{y=0}^{h-1} p(x, y) \tag{3.2}$$

avec :

h la hauteur de l'image

p(x,y) l'intensité en niveau de gris du pixel en (x,y)

S'il n'y a qu'un seul objet dans l'image, la courbe *projectionX* ne va présenter qu'un seul sommet. Pour calculer le nombre de sommets dans la courbe, la courbe convergente *projectionXconv* à la courbe *projectionX* est calculée en la traçant sur l'image à l'aide le fonction *convexHull* de la libraire OpenCV. Elle correspond à la courbe bleue sur la Figure 3.4.

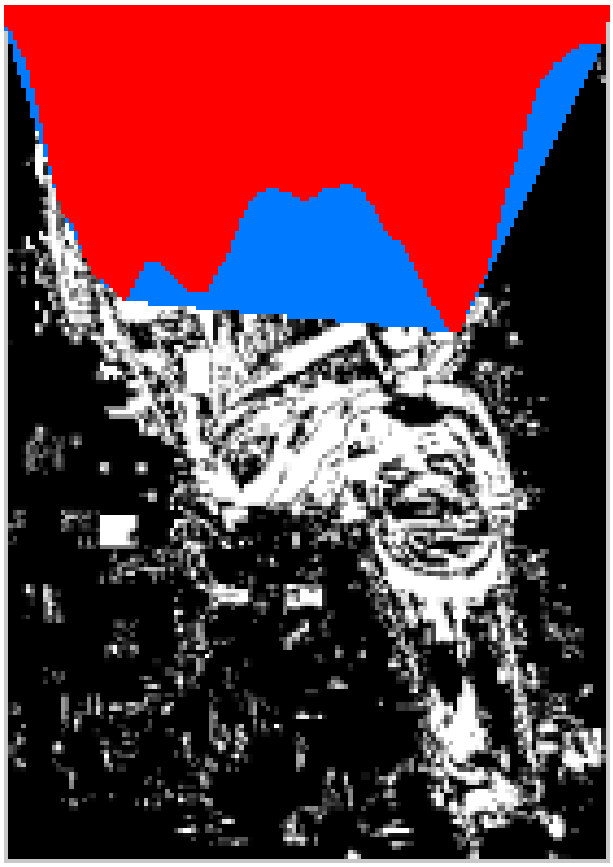


Figure 3.4: Courbes ProjectionX et ProjectionX convexe

(3.3)

$$differenceX(x) = \sum_{x=0}^{l-1}(projectionXconv(x) - projectionX(x)) \qquad (3.4)$$

avec :

l la largeur de l'image

La courbe *projectionX* est soustraite à la courbe *projectionXconv* pour obtenir la courbe *differenceX*.

Sur l'axe des abscisses, si le maximum de *differenceX* est supérieur à **threshold_projectionX** de *projectionX*, il y a au moins deux objets, il faut donc découper en 2 blobs avec comme séparateur une ligne verticale pour x qui correspond au maximum de *differenceX*. **threshold_projectionX** est un paramètre du système.

Processus sur l'axe des ordonnées

La courbe *projectionY* représentant la somme des intensités en niveaux de gris d'une même ligne y est tracé sur l'image.

$$projectionY(y) = \sum_{x=0}^{l-1} p(x, y) \tag{3.5}$$

l la largeur de l'image

p(x,y) l'intensité en niveaux de gris du pixel en (x,y)

S'il n'y a qu'un seul objet dans l'image, la courbe *projectionY* ne va présenter qu'un seul sommet. Pour calculer le nombre de sommets dans la courbe, la courbe convergente *projectionYconv* à la courbe *projectionY* est utilisée en la traçant sur l'image à l'aide la fonction *convexHull* de la libraire OpenCV.

$$differenceY(y) = \sum_{y=0}^{h-1} (projectionYconv(y) - projectionY(y)) \tag{3.6}$$

avec :

h la hauteur de l'image

La courbe *projectionY* est soustraite à la courbe *projectionYconv* pour obtenir la courbe *differenceY*.

Sur l'axe des ordonnées, si le maximum de *differenceY* est supérieur à **threshold_projectionY** de *projectionY*, il y a au moins deux objets, il faut donc découper en 2 blobs avec comme séparateur une ligne horizontale pour y qui correspond au maximum de *differenceY*. **threshold_projectionY** est un paramètre du système.

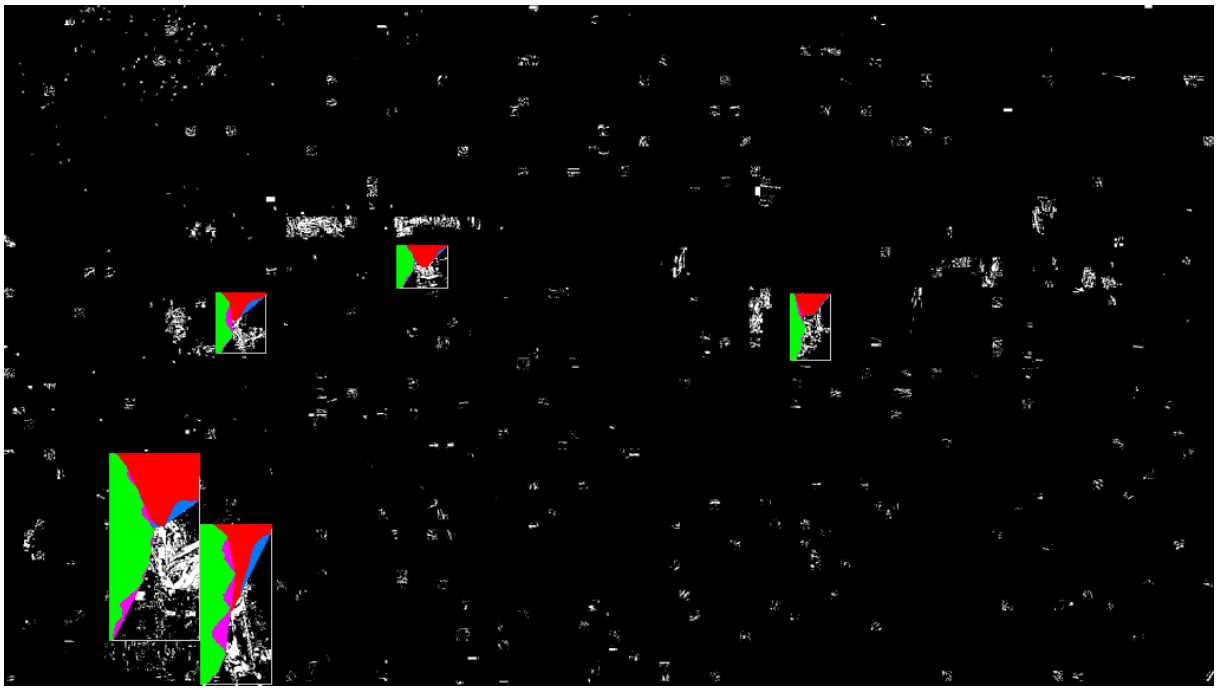
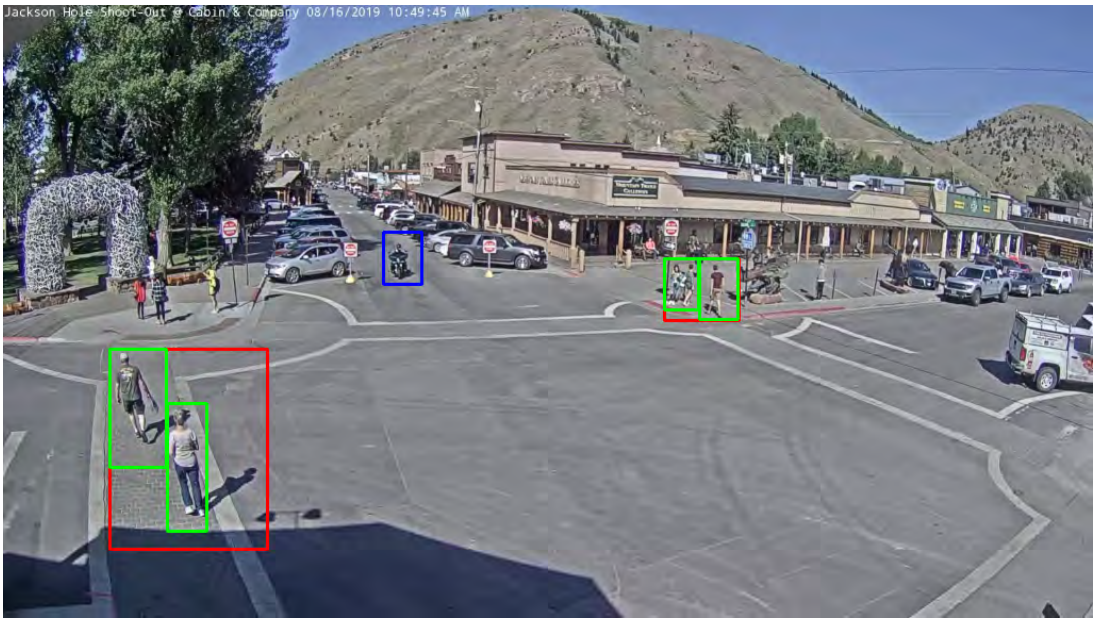


Figure 3.5: Image obtenue une fois chaque blob découpé

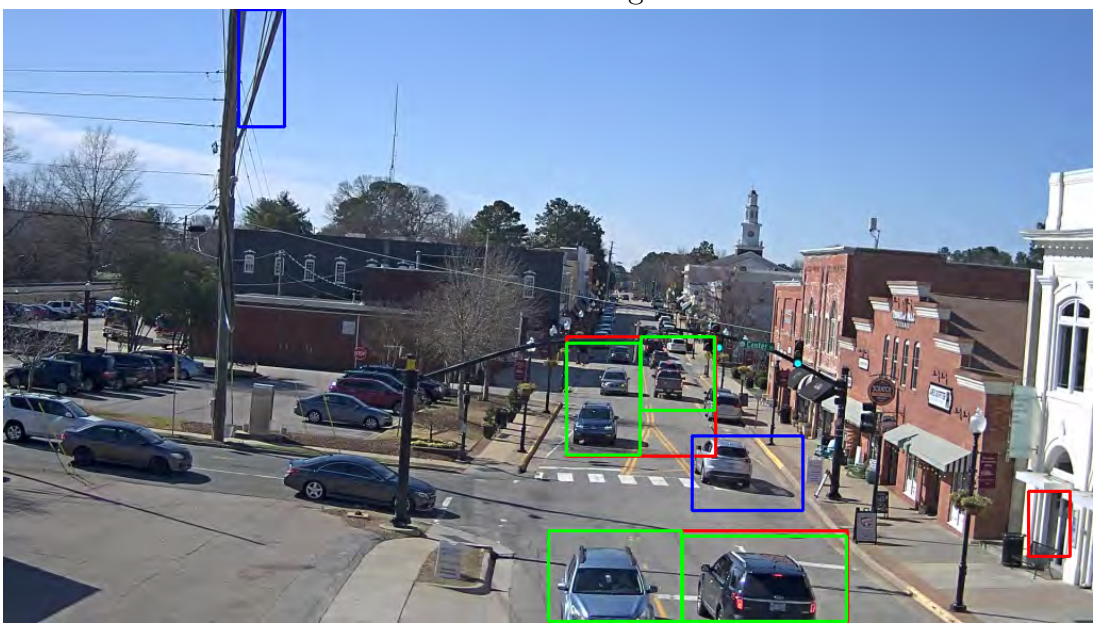
Lors du **Processus sur l'axe des abscisses** ou **Processus sur l'axe des ordonnées**, si un blob est découpé en deux blobs, la même fonction est appliquée aux deux nouveaux blobs pour savoir s'il est composé d'un unique objet ou de plusieurs. Une fois le processus récursif terminé, la liste de blobs définis par les coordonnées de leur centre (x et y) ainsi que par leur largeur et leur hauteur est complète. La Figure 3.5 représente le résultat du processus de découpe sur l'axe des abscisses et sur l'axe des ordonnées pour chaque blob en mouvement de d'aire supérieure à **min_area_blob**.

Résultats

Le processus de découpage des blobs permet de découper des blobs lorsque ils sont trop proches. Pour fonctionner correctement, il requiert des amas de blobs. Pour cela, 9 images avec des blobs proches ont été sélectionnées (2 images sont représentées sur la Figure 3.6 et les 7 autres sur les Figures A.1 à A.7 en Annexe A). Les résultats des deux méthodes **classique** et **développée** ainsi que la comparaison avec la méthode **manuelle** (comptabilisé visuellement par un homme) sont présentés dans le Tableau 3.1.



Caméra 1 - Image 1



Caméra 2 - Image 3

Figure 3.6: Les blobs en rouge sont ceux détectés par la **méthode classique** et supprimés par la **méthode développée**. Les blobs en bleu sont ceux détectés par la **méthode classique** et maintenus par la **méthode développée**. Les blobs en vert sont ceux détectés par la **méthode développée**.

	Caméra 1			Caméra 2			Caméra 3		
Image	**1**	**2**	**3**	**1**	**2**	**3**	**1**	**2**	**3**
Méthode classique									
Nombre d'objets	3	3	4	6	7	5	2	2	5
Temps d'exécution	0.057	0.065	0.049	0.072	0.057	0.046	0.036	0.226	0.034
Méthode développée									
Nombre d'objets	5	5	5	6	7	6	3	3	6
Temps d'exécution	0.310	0.378	0.175	0.471	0.582	0.457	0.058	0.074	0.120
Méthode manuelle									
Nombre d'objets	7	7	7	7	8	8	3	3	6
Temps d'exécution	4	4	4	4	4	4	4	4	4

Table 3.1: Nombre d'objets et temps d'exécution en secondes pour la **méthode classique** et pour la **méthode développée** ainsi qu'avec la **méthode manuelle**

Sur les 9 images sélectionnées, la **méthode développée** permet de découper sur chaque image 1 ou 2 amas de blobs en blob de taille réduite représentant ainsi plus précisément des objets en mouvement mais elle ne permet pas pour autant d'obtenir le bon nombre d'objets comme le permet la **méthode manuelle**. Le temps d'exécution de la **méthode développée** est proportionnel aux nombres de blobs re découpés. Le temps d'exécution de la **méthode développée** est compris entre **2** et **10** fois le temps d'exécution de la **méthode classique** pour une augmentation du nombre de blobs de **10%** à **50%**. La **méthode développée** permet ainsi un découpage plus fin des blobs même si celui-ci ne correspond pas à la réalité du terrain mais si rapproche améliorant ainsi les performances de l'**application d'apprentissage** et l'**application de catégorisation**. Une étude plus approfondie permettrait de confirmer ces résultats préliminaires.

3.1.4 Mise à jour des traces à partir des blobs

Un blob en mouvement doit appartenir à une trace. Si il a déjà été identifié dans une précédente image, il faut mettre à jour la trace correspondante sinon il faut créer une nouvelle trace.

Pour cela une liste de traces va être mise à jour à chaque nouvelle image. Un filtre de Kalman est utilisé pour prédire où doit évoluer la trace (donc la position de son prochain blob) en fonction de la position sur les précédentes images du blob.

Pour chaque trace, la liste des blobs en mouvement est passée en revue pour voir si un blob peut être rattaché à une trace (Algorithme 2). Lorsqu'un blob est rattaché à une trace, il est ainsi supprimé de la liste des blobs en mouvement. Pour être rattaché à une

trace, un blob en mouvement doit correspondre aux critères suivants :

- le centre du blob doit être à l'intérieur de l'aire du blob prédit par le filtre de Kalman
- la distance en pixels entre le centre du blob et le centre du blob prédit doit être inférieur à **max_distance_center** (**max_distance_center** est un paramètre du système)

Après avoir parcouru la liste des traces, la liste des blobs restants est parcourue. Effectivement, il est possible qu'un objet soit divisé en plusieurs blobs, il faut vérifier qu'un blob restant ne fasse pas partie d'une trace ayant précédemment trouvé un meilleur candidat. Dans ce cas, il est ajouté à la trace. Sinon c'est qu'il s'agit d'une nouvelle trace.

Algorithm 2: Mise à jour de la liste des traces

```
Data: blobs = liste des blobs en mouvement
traces = liste des traces
for i ← 0 to traces.length do
    for j ← 0 to blobs.length do
        inside = centerInside(blobs[j],traces[i]);
        distance = distanceCentre(blobs[j],traces[i]);
        if inside and distance < max_distance_center then
            updateTrace(traces[i],blobs[j]);
            delete blobs[i];
        end
    end
end
for i ← 0 to blobs.length do
    for j ← 0 to traces.length do
        inside = centerInside(blobs[i],traces[j]);
        if inside then
            updateTrace(traces[j],blobs[i]);
            delete blobs[i];
        end
    end
end
for i ← 0 to blobs.length do
    creationNouvelleTrace(blobs[i]);
end
```

Après la mise à jour d'une trace avec l'ajout d'un blob en mouvement, différentes étapes sont nécessaires pour assurer la cohérence de la trace :

- mise à jour du filtre de Kalman

- vérification d'une dérive induite par le filtre de Kalman
- garder le blob d'origine ou celui prédit par le filtre de Kalman

3.1.5 Garder le blob d'origine ou celui prédit par le filtre de Kalman

Parce que les blobs sont très sensibles au bruit généré par le flux vidéo, leurs positions et leurs tailles peuvent être légèrement différentes d'une image à l'autre. Lors de l'ajout d'un blob à une trace, il est nécessaire de vérifier si le blob correspond correctement à celui prédit ou s'il est trop différent.

- la distance en pixels entre les centres du blob et du blob prédit doit être inférieure à **max_distance_center**
- le ratio entre l'aire du blob prédit et l'aire du blob doit être compris entre **min_area_ratio_kalman** et **max_area_ratio_kalman** (**min_area_ratio_kalman** et **max_area_ratio_kalman** sont des paramètres du système)

Si ces deux conditions sont réunies, le blob est ajouté tel quel à la trace, sinon c'est le blob prédit qui est ajouté à la trace.

Lorsqu'un objet en mouvement change de direction, le filtre de Kalman a une légère latence. Pour éviter de perdre une trace suite à un changement de direction, lorsque les **nb_kalman_check** précédents blobs utilisés sont ceux prédits par le filtre, une vérification est faite pour s'assurer que la direction des blobs prédits est la même que la direction des blobs extraits du flux vidéo. **nb_kalman_check** est un paramètre du système.

3.2 Application d'apprentissage

Elle permet au système d'apprendre les caractéristiques propres à son environnement (Figure 3.7). Sa durée d'exécution est fonction du nombre d'images traitées par le capteur vidéo que l'on souhaite pour la phrase d'apprentissage. La précision de l'apprentissage est liée aux nombres d'images, un plus grand nombre d'images va permettre une meilleure précision.

Pour cela, l'application va récupérer **n** images du flux vidéo, **n** étant le nombre d'images souhaitées pour l'apprentissage. Chaque image sera traitée avec les fonctions suivantes :

- *Extraction de l'image différentielle depuis la vidéo*
- *Extraction et filtrage des blobs depuis l'image*
- *Découpage des blobs*
- **Catégorisation et correspondance des blobs**
- *Mise à jour des traces à partir des blobs*

Une fois les **n** images souhaitées extraites, l'extraction du flux vidéo s'arrête.

Un modèle LSTM est utilisé pour catégoriser un blob en fonction de sa taille et sa position à partir des données récoltées dans les traces. Un modèle LSTM permet de bénéficier de la dimension temporelle. Celle-ci va permettre de prendre en comptre une trace dans son ensemble : sa provenance, sa destination et sa vitesse. Ces informations sont très importantes pour catégoriser correctement un objet. Toutes les voitures se déplacent sur la route, les personnes sur le trottoir. Voiture et vélo n'ont pas la même vitesse. En effet, chaque type d'objet a des caractéristiques qui lui sont propres : la vitesse de déplacement d'une voiture est plus élevée que celle d'un vélo et d'un piéton, un piéton circule sur un trottoir alors qu'une voiture circule sur une route.

L'application va traiter les traces avec les fonctions suivantes :

- **Augmentation de l'ensemble de données**
- **Transformation en séries temporelles**

Une fois toutes les traces traitées, TensorFlow [3] et Keras[4] sont utilisés pour construire et entrainer notre modèle LSTM. Le modèle utilise en entrée les séries temporelles issues de la fonction *Transformation en séries temporelles* et la catégorie de la série temporelle en sortie. Il est composé de :

- une couche LSTM avec M unités
- une couche dense avec N unités
- une couche drop out avec un coefficient O

[3]https://www.tensorflow.org/
[4]https://keras.io/

La phase d'entrainement comporte 3 paramètres :

- nombre d'epoch
- taille du batch d'apprentissage
- optimiseur à utiliser : Adam ou SGD

Une fois le modèle entrainé, l'application d'apprentissage n'est plus exécutée et le modèle est utilisé dans l'application de catégorisation décrite dans la section 3.3.

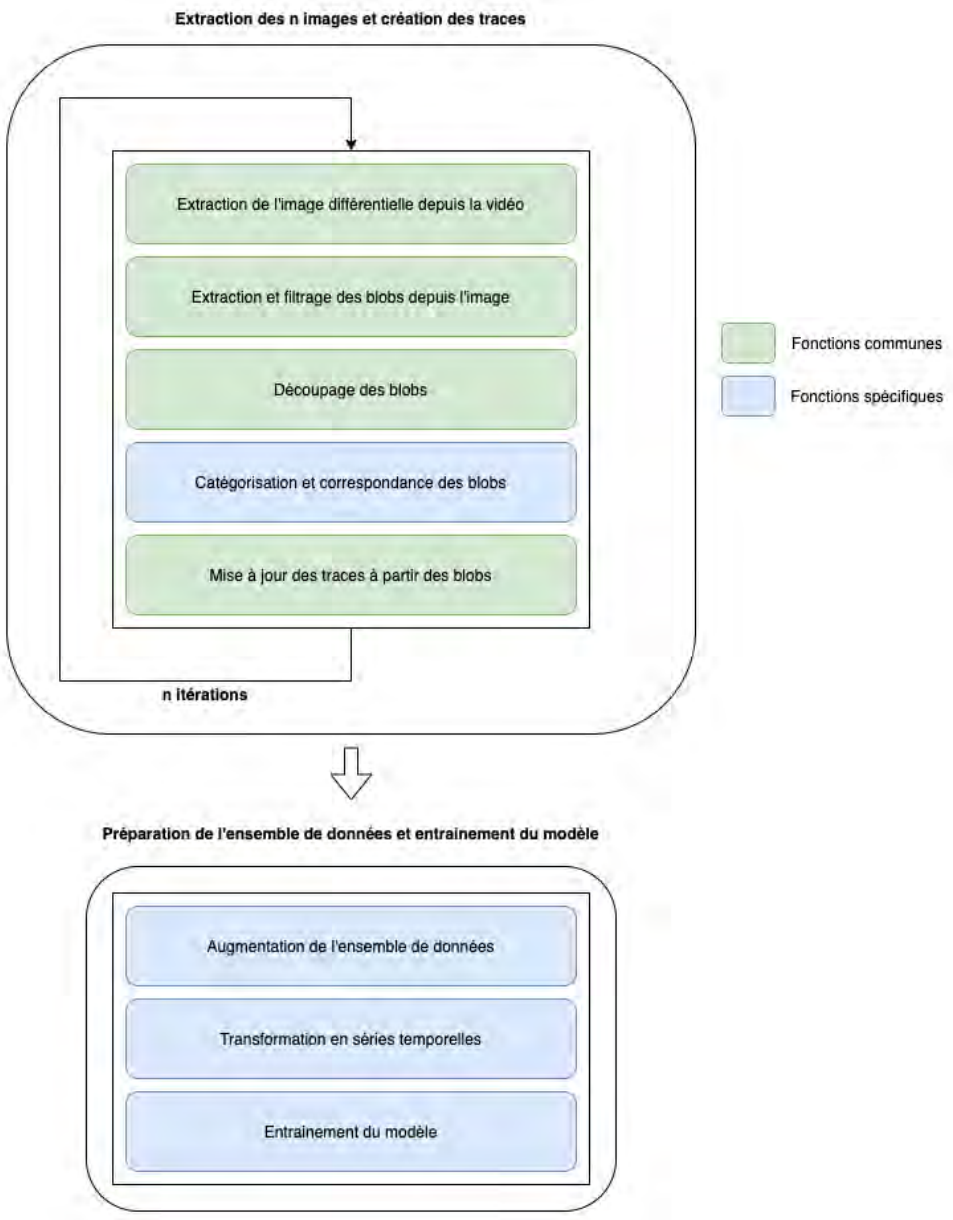


Figure 3.7: Synoptique de l'application d'apprentissage

3.2.1 Catégorisation et correspondance des blobs

Mask Region based Convolutional Neural Networks (Mask RCNN) [74] est utilisé pour catégoriser tous les blobs extraits du flux vidéo. Mask RCNN est construit sur le modèle *Feature Pyramid Network* (FPN) qui catégorise chaque pixel de l'image dans une des classes qu'il a apprises. Mask RCNN a été entrainé sur l'ensemble de données MS Coco et il permet d'identifier plusieurs centaines de classes différentes. Seules les classes qui seront utilisées sont gardées, à savoir les voitures, les camions, les vélos et les personnes (Figure 3.8).

Figure 3.8: Détection de classes d'objets avec le modèle Mask RCNN

Mask RCNN utilise une image en entrée et produit une liste d'objets trouvés sur l'image en entrée, correspondants aux classes qu'il a apprises. Pour chaque objet, sa position, sa taille et son indice de confiance sont fournis. Les objets dont l'indice de confiance est inférieur à **min_maskrcnn_confidence** sont supprimés. **min_maskrcnn_confidence** est un paramètre du système.

La fonction va ensuite passer en revue la liste des blobs en mouvement obtenue à la suite de la fonction *Découpage des blobs* pour trouver parmi les objets détectés par Mask RCNN celui qui correspond le mieux, selon les critères suivants :

- la distance entre le centre du blob et le centre de l'objet doit être inférieure à la moyenne des largeurs du blob et de l'objet

- le centre de l'objet doit se situer à l'intérieur de l'aire du blob
- le ratio entre l'aire de l'objet et l'aire du blob doit être compris entre **min_area_ratio_maskrcnn** et **max_area_ratio_maskrcnn** (**min_area_ratio_maskrcnn** et **max_area_ratio_maskrcnn** sont des paramètres du système)

Lorsqu'un objet détecté par Mask RCNN correspond à un blob en mouvement, il est retiré de la liste des objets détectés par Mask RCNN pour ne pas être réassigné. Les blobs en mouvement n'ayant aucune correspondance avec un objet détecté par Mask RCNN sont supprimés de la liste des blobs en mouvement (Algorithme 3). Le résultat de la fonction est visible sur la Figure 3.9.

Algorithm 3: Correspondance des objet détectés avec les blobs en mouvement

```
Data: blobs = liste des blobs en mouvement
objets = liste des objets détectés par Mask RCNN
for i ← 0 to blobs.length do
    for j ← 0 to objets.length do
        inside = centerInside(blobs[i],objets[j]);
        distance = distanceCentre(blobs[i],objets[j]);
        ratio = calculRatio(blobs[i],objets[j]);
        if inside and distance and min_area_ratio_maskrcnn < radio <
         max_area_ratio_maskrcnn then
            updateBlob(blobs[i],objets[j]);
            delete objets[j];
        end
    end
end
for i ← 0 to blobs.length do
    if no blobs[i].classe then
        delete blobs[i];
    end
end
```

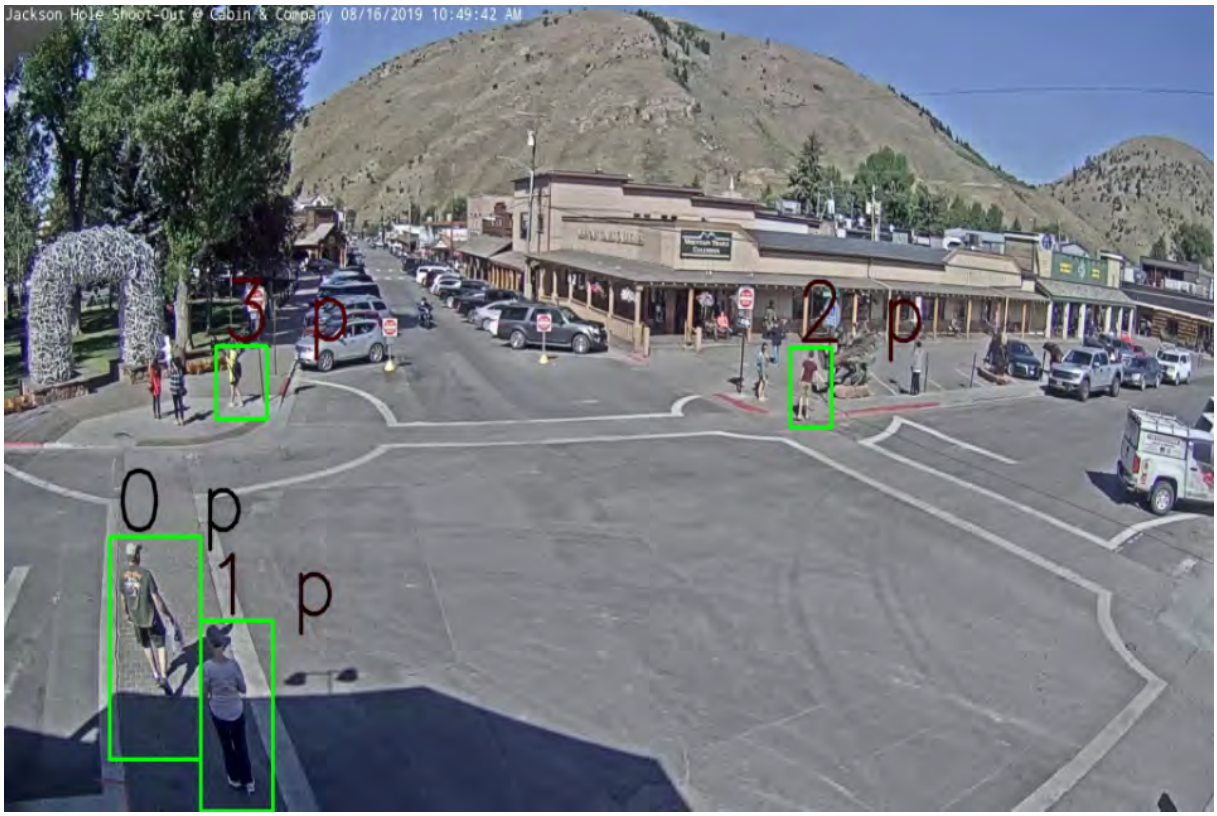


Figure 3.9: Objets en mouvement et leurs classes prédites par le modèle Mask RCNN

3.2.2 Augmentation de l'ensemble de données

L'augmentation de l'ensemble de données est une technique incontournable qui consiste à produire de nouvelles entités légèrement différentes à partir de celles qui existent déjà dans l'ensemble de données. Cette technique permet de faciliter l'apprentissage tout en réduisant l'*overfitting*.

La première étape de la fonction est de supprimer les traces qui comportent différentes catégories. Cela peut se produire lorsque un objet passe devant un autre dans le flux vidéo, les deux traces concernées vont se retrouver perturbées lorsque l'objet de l'une masquera l'objet de l'autre.

Comme il peut y avoir une quantité de traces différentes pour chaque catégorie, il est nécessaire d'équilibrer pour que chaque classe ait le même nombre de traces. Une quantité de traces trop différentes entre les catégories engendrerait un *overfitting* sur la catégorie la plus représentée. Pour éviter cela, il faut augmenter les catégories les moins représentées. Pour générer des traces augmentées, des traces existantes sont légèrement modifiées, par exemple, en changeant de quelques pixels la taille ou le centre des blobs.

4 coefficients compris entre 0.9 et 1.1 (diminution ou augmentation de 10%) vont être tirés aléatoirement pour chaque nouvelle trace augmentée. Ils représente la largeur, la hauteur et le centre en x et y du blob et seront appliqués à tous les blobs de la série. La vitesse n'est volontairement pas modifiée car le processus est plus complexe mais cela aurait pu être réalisable. Une fois toutes les catégories en nombre égal, une nouvelle série d'augmentations est réalisée pour augmenter la taille de l'ensemble des données et rendre le modèle plus performant. Par exemple, à partir de 5 minutes de vidéo, l'équivalent d'une heure de traces augmentées est générée aidant le modèle à mieux généraliser.

3.2.3 Transformation en séries temporelles

La dernière étape avant de pouvoir entrainer le modèle est de convertir toutes les traces de longueur différentes en séries temporelles de taille fixe (Figure 3.10). Une trace est décomposée en séries temporelles de **nb_els_time_serie** éléments en utilisant une fenêtre glissante. **nb_els_time_serie** est un paramètre du système. Une trace de courte durée fournira moins de séries temporelles qu'une trace de longue durée. Il est possible de calculer le nombre de séries temporelles créées depuis une trace avec la Formule 3.7.

$$\text{nb\_series} = \text{nb\_blobs} - \textbf{nb\_els\_time\_serie} \tag{3.7}$$

avec :

nb_series le nombre de series temporelles pour une trace

nb_blobs le nombre de blobs dans la trace

nb_els_time_serie le nombre d'éléments dans la fenêtre glissante

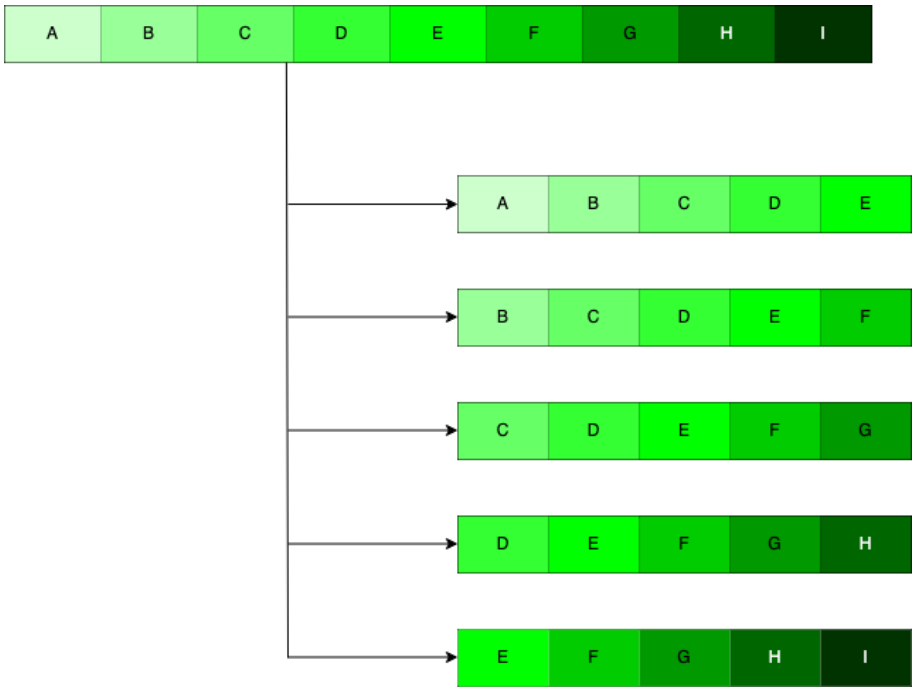


Figure 3.10: Conversion de données brutes en séries temporelles de 5 éléments

3.3 Application de catégorisation

L'application de catégorisation est exécutée une fois que l'application d'apprentissage a terminé d'entrainer le modèle LSTM. Le modèle LSTM est utilisé pour catégoriser les traces. Le processus est le suivant Figure 3.11 :

- *Extraction de l'image différentielle depuis la vidéo*
- *Extraction et filtrage des blobs depuis l'image*
- *Découpage des blobs*
- *Mise à jour des traces à partir des blobs*
- Utilisation du modèle LSTM pour catégoriser la trace

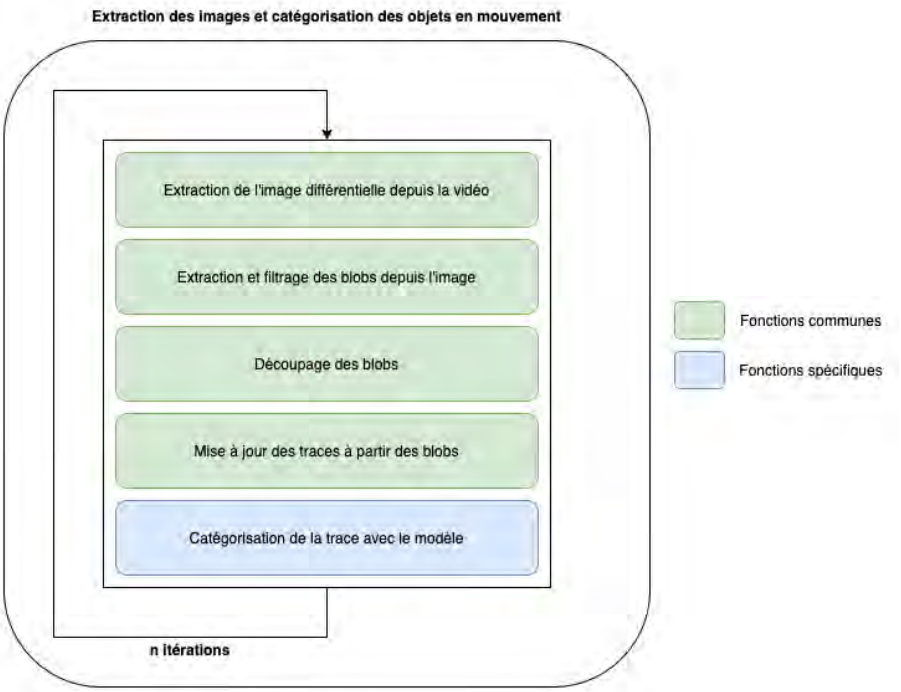


Figure 3.11: Synoptique de l'application de catégorisation

3.4 Expérimentations et résultats

Nous avons évalué le système sur 3 vidéos :

- Figure 3.12 est extraite de la caméra 1 et représente un carrefour de grande taille
- La Figure 3.13 est extraite de la caméra 2 et représente un carrefour de taille moyenne
- La Figure 3.14 est extraite de la caméra 3 et représente un parking

Ainsi les différents objets à détecter et à suivre seront analysés avec des propriétés différentes (position, vitesse, taille, angle), ce qui permettra de tester l'adaptation de la méthode aux différents environnements.

Figure 3.12: Caméra 1 - Carrefour de grande taille

Figure 3.13: Caméra 2 - Carrefour de taille moyenne

Figure 3.14: Caméra 3 - Parking

Les paramètres du système ont été choisis de manière empirique, ils dépendent de la résolutions de la caméra, la hauteur à laquelle elle est située ainsi que l'angle de vision. Les paramètres du système sont identiques pour les trois expériences. Il serait intéressant de réaliser une étude sur l'ensemble des paramètres du système pour les trois expériences différentes afin de déterminer leurs influences sur les résultats.

- **min_area_blob** : 250
- **threshold_projectionX** : 0.66
- **threshold_projectionY** : 0.66
- **max_distance_center** : 40
- **min_area_ratio_kalman** : 0.8
- **max_area_ratio_kalman** : 1.2
- **nb_kalman_check** : 5
- **min_maskrcnn_confidence** : 0.7
- **min_area_ratio_maskrcnn** : 0.25

- **max_area_ratio_maskrcnn** : 4
- **nb_els_time_serie** : 5

Afin d'évaluer quels sont les meilleurs paramètres à utiliser pour l'entrainement du modèle LSTM, 972 configurations vont être entrainées avec les paramètres suivants :

- LSTM M unités : 50 ou 100 ou 200
- dense N unités : 50 ou 100 ou 200
- nombre d'epoch : 50 ou 100
- batch size : 16 ou 32 ou 40
- coefficient drop out O : de 0.1 à 0.9 avec des pas de 0.1
- optimisation : Adam ou SGD

Pour chaque expérimentation, les 5 configurations de paramètres donnant la meilleure précision sont retenues et présentées dans la Table 3.2 pour l'expérimentation 1, la Table 3.3 pour l'expérimentation 2 et la Table 3.4 pour l'expérimentation 3.

# LSTM	# DENSE	epochs	batch size	dropout	optimizer	accuracy
100	**50**	**50**	**16**	**0.3**	**Adam**	**0.95794**
200	100	50	16	0.3	Adam	0.95589
100	100	50	16	0.2	Adam	0.95396
50	200	50	16	0.1	Adam	0.94992
200	50	50	16	0.4	Adam	0.94973

Table 3.2: Les 5 configurations de paramètres donnant la meilleure précision pour l'expérimentation 1

# LSTM	# DENSE	epochs	batch size	dropout	optimizer	accuracy
200	**200**	**100**	**16**	**0.2**	**Adam**	**0.98856**
200	50	100	32	0.3	Adam	0.98840
200	200	100	32	0.2	Adam	0.98776
200	200	100	32	0.8	Adam	0.98744
200	200	100	16	0.4	Adam	0.98679

Table 3.3: Les 5 configurations de paramètres donnant la meilleure précision pour l'expérimentation 2

# LSTM	# DENSE	epochs	batch size	dropout	optimizer	accuracy
200	**200**	**100**	**16**	**0.4**	**Adam**	**0.98391**
200	100	100	16	0.7	Adam	0.97306
100	200	100	16	0.2	Adam	0.97178
200	100	100	32	0.2	Adam	0.96980
200	50	50	16	0.6	Adam	0.96933

Table 3.4: Les 5 configurations de paramètres donnant la meilleure précision pour l'expérimentation 3

Les paramètres statistiquement majoritaires pour l'ensemble des 15 configurations ayant la meilleure précision sont :

- pour le nombre d'unités LSTM : 200
- pour le nombre d'unités Dense : 200
- pour le nombre d'épochs : 100
- pour la taille du batch : 16
- pour le coefficiant de dropout : 0.2
- optimisation : Adam

C'est donc l'ensemble des paramètres cité précédemment qui sera utilisé en production.

3.5 Conclusion préliminaire

Le système présenté obtient une précision moyenne de 97% basé sur le modèle Mask RCNN pour les 3 expériences. La réelle différence entre le système proposé et le modèle Mask RCNN n'est pas sur la précision mais sur le temps de calcul. Sur l'architecture d'entrainement du système, un mono processeur Intel de 2Ghz, Mask RCNN nécessite 2000ms pour catégoriser les objets alors que le modèle LSTM ne nécessite que 20ms. La quantité de données à traiter en entrée est bien plus réduite pour le modèle LSTM que pour le modèle Mask RCNN car il utilise des métadonnées et non pas une image complète comme le modèle Mask RCNN. Le modèle LSTM entrainé nécessite très peu de puissance de calcul pour une précision légèrement inférieure, il est donc utilisable sur des architectures à puissance de calcul limitée telles que des cartes électroniques embarquées.

Chapter 4

Amélioration de filtre sur image en utilisant CGP

Dans ce chapitre, nous allons travailler sur l'amélioration de l'extraction des blobs en mouvement qui est actuellement un filtre simple construit par un des employés de Kawantech utilisant très peu de fonctions. Pour cela, nous allons améliorer génétiquement le filtre à image au travers d'une stratégie évolutionnaire. Une adaptation de CGP-IP pour l'amélioration génétique va être présentée et testée dans trois expérimentations. Ce chapitre a donné lieu à une publication [11].

4.1 Introduction

La construction automatisée de filtre d'image en utilisant le *machine learning* a permis le développement de nouvelles applications. L'algorithme couramment utilisé pour ce type de problème est un réseau neuronal convolutif qui apprend une séquence de filtres paramétrés et est composé de millions de paramètres [72]. Bien que les performances de ces méthodes sur de larges ensembles de données soient impressionnantes, surpassant la performance humaine dans les tâches de catégorisation visuelle [73], un sérieux obstacle dans l'application de ces méthodes est leur manque d'interprétabilité. De plus, ces systèmes sont souvent utilisés pour remplacer d'anciens systèmes créés par des experts sans pour autant profiter de leurs expertises.

La programmation génétique offre une alternative séduisante pour ces processus de *machine learning*. En combinant des fonctions de traitement de l'image de bas et haut

niveau, il est possible de construire un filtre compréhensible et interprétable. L'ensemble des fonctions peuvent être choisies par un expert pour répondre à des contraintes de lisibilité ou de performances et peuvent être construites en utilisant des librairies existantes comme OpenCV[1] que nous utiliserons dans ce chapitre. De plus, des filtres d'images existants peuvent être utilisés comme point de départ pour les optimiser. L'amélioration génétique de logiciel a démontré que l'évolution, à partir des programmes créés par des humains, améliore leur efficacité et corrige des erreurs [5]. Cela permet d'obtenir un programme final qui a de meilleures performances et moins d'erreurs que l'original mais qui reste interprétable et compréhensible par des experts.

Nous utilisons Cartesian Genetic Programming (CGP) [118], une forme populaire de programmation génétique de graphes, pour améliorer un filtre d'image existant ou créer de nouveaux filtres. Avec CGP, les programmes sont représentés comme des graphes de fonctions, qui permettent d'encoder les filtres existants en modifiant le graphe du programme. Nous proposons de nouveaux opérateurs génétiques spécifiques pour l'amélioration génétique avec CGP. Nous démontrerons que l'insertion de nœuds dans un graphe peut améliorer l'évolution en se basant sur la précision du filtre. Nous allons pour cela étudier un CGP standard, un CGP avec une population de départ créé par un expert, ainsi que les opérateurs de mutations proposés sur trois différents ensembles d'images pour de la segmentation : deux existants sur des précédents travaux avec CGP et un nouveau sur le trafic urbain. Notre évolution permet de rapidement améliorer un filtre d'image créé par un expert avec des connaissances basiques. Selon nous, ce sont les premiers travaux avec CGP pour de l'amélioration génétique.

Ce chapitre est structuré de la manière suivante. Dans la section 4.2, nous présentons Cartesian Genetic Programming et ses applications pour le traitement des images et ensuite l'amélioration génétique. Dans la section 4.3, nous décrivons les opérateurs d'ajout et de suppression de nœuds pour l'amélioration génétique avec CGP. Nous présentons les tâches de traitement de l'image et les paramètres expérimentaux dans la section 4.5. Nous comparons l'évolution des différents processus avec des populations prises au hasard ainsi que celles issues des filtres des experts et en utilisant les opérateurs proposés. Enfin, dans la section 4.6, nous discutons des possibles applications de cette méthode et définissons les futurs axes de recherche.

[1] https://opencv.org/

4.2 Travaux existants

Ce travail porte sur deux parties de la littérature de la programmation génétique : Cartesian Gentic Programming [118], plus spécifiquement son application au traitement de l'image et une étude sur de nouveaux opérateurs génétiques, et l'optimisation génétique, optimisation de programmes conçus par des experts grâce à l'évolution.

4.2.1 Cartesian Genetic Programming

Cartesian Genetic Programming (CGP) est une forme de Programmation Génétique (GP) dans laquelle les programmes sont représentés par des graphes directs, souvent acycliques, indexés par des coordonnées cartésiennes. CGP a été inventé par Miller et Thomson [118, 119] pour faire évoluer des circuits électroniques, mais a depuis été appliqué à de multiples domaines [120]. CGP est utilisé pour faire évoluer des réseaux de neurones [89], pour de la détection d'objets dans du traitement de l'image [67], et pour de la réduction de bruit dans du traitement de l'image [86]. Ces bénéfices incluent la neutralité des nœuds, qui sont les parties codées du génome qui ne contribuent pas au programme interprété, la réutilisation des nœuds et une taille fixe de représentation qui limite la taille des programmes [117]. Miller a publié en 2019 un état des lieux de CGP [121].

Avec CGP, les nœuds fonctionnels, définis par un ensemble de gènes évolués, connectent les entrées du programme à d'autres nœuds actifs au moyen de leurs coordonnées cartésiennes. La sortie du programme correspond soit à un nœud interne soit à une des entrées. Les nœuds CGP sont disposés dans une grille de R lignes et de C colonnes. Un nœud est autorisé à se connecter à n'importe lequel des précédents nœuds en se limitant au paramètre L qui précise le nombre maximum de colonnes qui peuvent être traversées (Figure 4.1). Dans ce travail, [121], $R = 1$ signifie que tous les nœuds sont dans une seule ligne.

Le génotype CGP consiste en une liste de nœud de gènes. Chaque nœud dans le génome porte sa fonction, les coordonnées des entrées (Connection 0 et Connection 1), et des paramètres optionnels pour la fonction du nœud. Enfin, la fin du génome code les nœuds qui donnent la sortie finale du programme. En remontant depuis les nœuds de sortie, une fonction peut être dérivée pour chaque sortie du programme, offrant une représentation concise et lisible du programme.

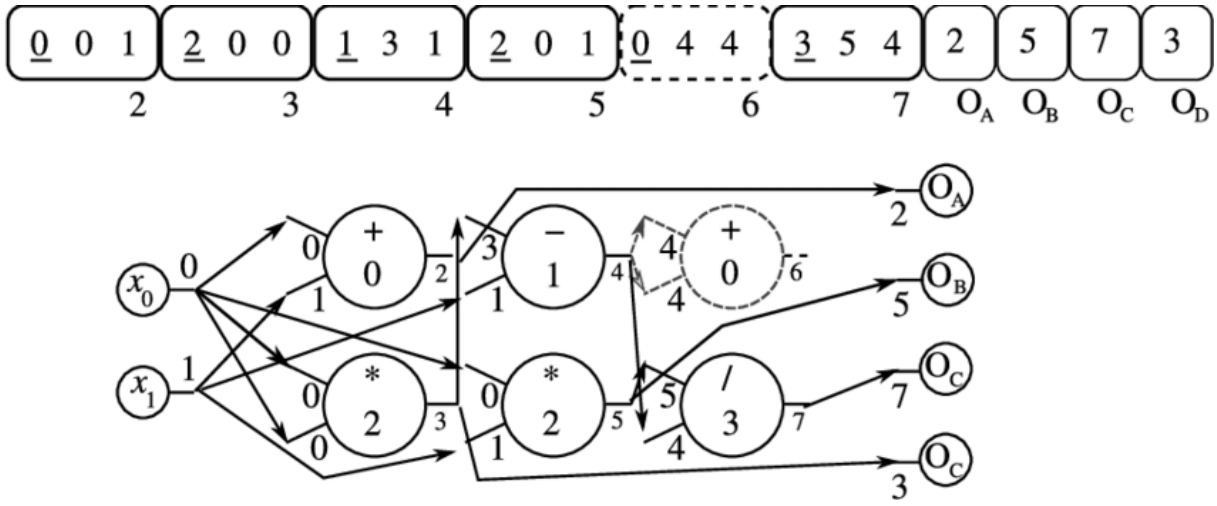


Figure 4.1: Le génotype CGP est un ensemble de 4 fonctions mathématiques : addition (0), soustraction (1), multiplication (2), division (3) (Source [116])

Les gènes avec CGP sont optimisés en utilisant une stratégie évolutionnaire 1+λ. Une population d'individus λ est générée au hasard à partir d'un parent et évaluée sur un problème de test. L'évaluation est effectuée en décodant le programme à partir du génotype individuel et en l'appliquant à un problème spécifique tel que la segmentation d'image comme dans ce chapitre. Le meilleur individu basé sur son évaluation est retenu pour la prochaine génération. Une opération de mutation est appliquée à cet individu pour créer de nouveaux individus λ. Avec CGP, l'opération de mutation échantillonne au hasard un sous-ensemble de nouveaux gènes à partir d'une distribution uniforme. Cette nouvelle population est évaluée et le meilleur individu est retenu pour la prochaine génération. Ce processus itératif s'exécute tant qu'une condition d'arrêt n'est pas rencontrée (généralement un certain niveau de précision ou un nombre de générations).

4.2.2 Cartesian Genetic Programming pour le traitement de l'image

Un choix important lors de l'utilisation de CGP est l'ensemble des fonctions accessibles. Dans l'application originale pour la création de carte électronique, les fonctions étaient des portes logiques tel que AND et NOR. Les applications de CGP aux jeux vidéos [178] utilisent un ensemble de fonctions mathématiques telles que $x + y$, $x * y$, et $cos(x)$ avec des entrées x et y. L'ensemble des fonctions doit être défini de telle manière que chaque sortie d'une fonction d'un nœud soit une entrée valide pour les autres fonctions.

CGP pour le traitement de l'image (CGP-IP) est une adaption de CGP qui utilise des fonctions de traitement de l'image et qui applique le programme directement sur les images [66]. Les entrées et sorties des fonctions évoluées sont des images qui permettent la cohérence entre chaque fonction de nœud. Chaque fonction de nœud est définie par une image d'une taille fixe en entrée et une image de la même taille en sortie. CGP-IP a précédemment été utilisé avec un ensemble de 60 fonctions [64] provenant de la librairie OpenCV.

Dans des précédents travaux, [66], CGP-IP utilise un algorithme de distribution de la population par île. Dans cette méthode, plusieurs populations sont en compétition à l'intérieur d'îles indépendantes avec une stratégie évolutionnaire $1 + \lambda$. Un paramètre de migration permet la synchronisation du meilleur individu sur l'ensemble de toutes les îles. Le modèle d'île a été démontré comme une alternative à l'algorithme génétique et aide à préserver la diversité génétique [177]. Son utilisation avec CGP-IP a prouvé une amélioration par rapport à une stratégie évolutionnaire $1 + \lambda$.

Les individus CGP-IP sont évalués en appliquant le filtre évolué à un ensemble d'images, les comparant aux images attendues et calculant une métrique représentant la différence entre la sortie du filtre évolué avec les images attendues telle que l'erreur moyenne ou le coefficient de corrélation de Matthew (MCC) [115]. Dans ce chapitre, nous utilisons MCC, qui mesure la qualité de la classification binaire et qui a prouvé être particulièrement bien adapté pour la classification avec CGP [68]. Les calculs sont basés sur la matrice de confusion, qui est l'ensemble des vrais positifs (TP), faux positifs (FP), vrais négatifs (TN) et faux négatifs (FN).

$$mcc = \frac{TP * TN - FP * FN}{\sqrt{(TP + FP)(TP + FN)(TN + FP)(TN + FN)}} \tag{4.1}$$

Un MCC avec un score de 1 correspond à la classification parfaite, 0 à une classification au hasard et -1 à une classification parfaite mais totalement inversée. Notre fonction d'évaluation pour cette évolution est définie telle quelle :

$$fitness = 1 - mcc. \tag{4.2}$$

Dans ce chapitre, l'évolution est utilisée pour *minimiser* la fonction d'évaluation en cherchant un programme avec un MCC élevé.

4.2.3 Amélioration Génétique

Genetic improvement (GI) est un champ de recherche logicielle relativement récent qui applique des méthodes d'optimisation évolutionnaires pour améliorer les logiciels existants. En utilisant du code source développé par un humain comme point de départ, le GI cherche l'espace des variations du programme créé en appliquant des opérateurs de mutation. La richesse de cet espace dépend de la puissance et de l'expressivité des opérateurs de mutation, qui peuvent modifier du code existant en changeant des fonctions ou des paramètres, en ajoutant du code au programme ou dans certains cas en en supprimant. Sur la précédente décennie, le champ de GI s'est étendu et les recherches actuelles sur le GI démontrent plusieurs applications potentielles. GI a été utilisé pour corriger des erreurs logicielles [5, 94], pour considérablement augmenter les performances des applications logicielles [94, 176], pour porter une application d'une plateforme à une autre [93], pour transplanter des fonctionnalités de code entre plusieurs versions d'un système [139], pour ajouter des fonctionnalités [69] et plus récemment pour améliorer la gestion de la mémoire [181] et la gestion de l'énergie [15].

Dans la plupart des méthodes, appliquer GI à un programme existant est réalisé en encodant le programme existant dans un arbre GP et en calculant le génome correspondant. Les opérateurs de mutation GP sont appliqués au programme encodé pour générer des programmes alternatifs. A cet effet, l'encodage et les opérateurs du programme doivent être définis à la fois pour être adaptés au programme initial à améliorer et avec des fonctions supplémentaires pour permettre une évolution pour améliorer le graphe fonctionnel. La fonction d'évaluation utilisée pendant la phase d'évolution du programme peut être basée sur différentes métriques, telles que la taille du programme, son efficacité, sa pertinence pour des cas de test donnés et autres [5, 93, 176].

Dans ce chapitre, nous proposons des opérateurs pour GI avec CGP. A notre connaissance, c'est la première utilisation de GI pour CGP car la majorité de la littérature sur GI utilise une représentation du programme sous forme d'arbre plutôt qu'un graphe. Les opérateurs proposés d'insertion et de suppression d'un nœud sont similaires à des opérateurs existants pour GI avec une représentation en arbre mais sont étudiés ici dans le cadre de l'évolution des graphes.

4.3 Genetic Improvement avec CGP-IP

Dans cette section, nous introduisons nos opérations d'insertion et de suppression créées pour améliorer le GI basé sur CGP. Lors de la phase d'évolution dans un CGP, des mutations sont appliquées au hasard sur les gènes de nœuds existants qui correspondent à des connexions, des fonctions ou des paramètres. Les génomes sont de taille constante et ajouter ou supprimer des nœuds fonctionnels à l'intérieur d'un graphe peut être difficile pour permettre à l'évolution de réussir. Pour cela, de précédents travaux ont proposé un génome s'auto- modifiant [65] qui utilise des fonctions qui peuvent ajouter ou supprimer des nœuds mais seulement lors de l'exécution du graphe. Nous proposons les opérateurs d'insertion et de suppression pour changer la taille du graphe pendant l'évolution. Ces opérateurs sont créés afin de maintenir le sous graphe actif du programme, par exemple lorsqu'ils ne sont pas destructifs.

La mutation consiste à appliquer un de ces trois opérateurs suivants : insertion d'un nœud, suppression d'un nœud ou mutation des paramètres standards à l'aide d'une distribution uniforme. Les opérateurs de nœuds ont un taux de mutation configurable r_{ins} et r_{del} correspondant à la probabilité d'applications des opérateurs. Si l'un de ces opérateurs est appliqué, il sera la seule mutation autorisée, sinon la mutation des paramètres standard s'applique. Dans ce chapitre, $r_{ins} = 0.1$ et $r_{del} = 0.1$ pour toutes les expérimentations.

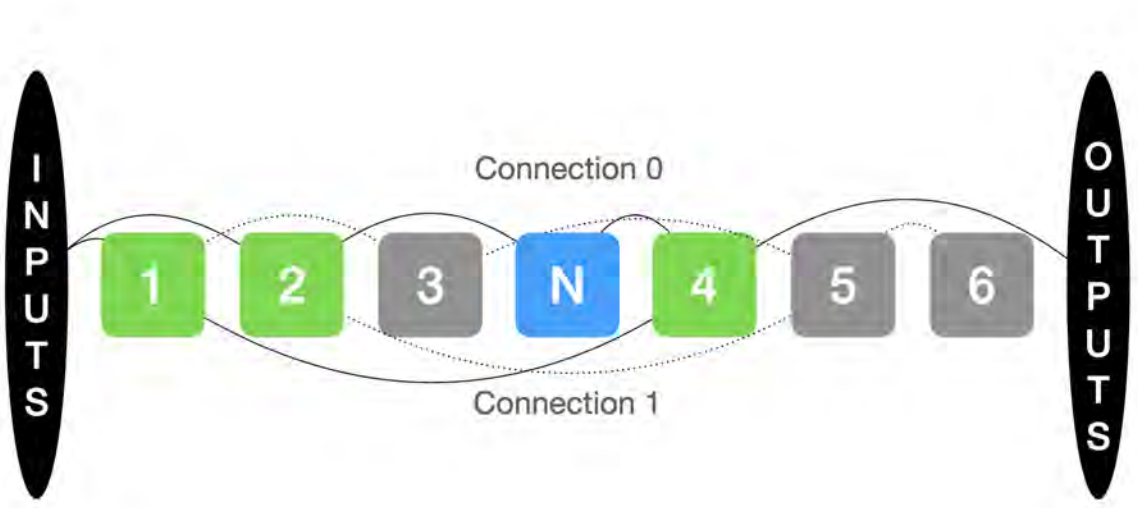


Figure 4.2: Un graphe avec un nœud N inséré à l'index 4

4.3.1 Insertion d'un nœud

L'opérateur d'insertion d'un nœud ajoute un nouveau nœud entre deux nœuds connectés dans le graphe actif d'un individu CGP. Pour permettre l'insertion d'un nœud, nous avons changé la taille maximale possible du graphe d'un individu CGP $R * C$ et plus précisément le nombre de colonnes C dans notre cas, en adaptant cette valeur tout au long de l'évolution. Pour préserver la structure du programme, les connexions aux autres nœuds dans le génome sont ajustés après l'insertion d'un nœud. Comme décrit dans l'Algorithme 4 et illustré dans la Figure 4.2, les connexions de tous les nœuds après le nœud inséré sont augmentées de 1. Cela permet de préserver les connexions existantes du graphe et insère simplement le nouveau nœud entre deux nœuds choisis au hasard. Nous avons étudié deux fonctions d'insertion : en utilisant une fonction d'identité (NOP), qui ne change pas immédiatement le programme du graphe, ou en insérant une fonction au hasard qui peut changer le programme du graphe.

Algorithm 4: Insertion d'un nœud avec une fonction au hasard

```
Data: nodes is an array containing all nodes
NOP_insertion is a boolean
Result: node inserted at position index
index = getRandomActiveNode();
nodes.insert(index,copyNode(nodes[index]));
nodes[index+1].conn0 = 1;
nodes[index+1].conn1 = nodes[index+1].conn1 + 1;
if NOP_insertion then
    // set a NOP function
    nodes[index].function = NOP;
else
    // set a Random function
    nodes[index].function = getRandomFunction();
end
for i ← index to nodes.length do
    if i − nodes[i].conn0 < index then
        nodes[i].conn0 = nodes[i].conn0 + 1;
    end
    if i − nodes[i].conn1 < index then
        nodes[i].conn1 = nodes[i].conn1 + 1;
    end
end
for i ← 0 to outputs.length do
    if nodes.length − outputs[i] < index then
        outputs[i] = outputs[i]+1;
    end
end
```

4.3.2 Suppression d'un nœud

L'opérateur de suppression d'un nœud supprime un nœud d'un graphe actif d'un individu CGP, comme montré dans la Figure 4.3. Comme pour l'opérateur d'insertion de nœuds, le reste du génome est corrigé pour s'assurer que les autres parties du graphe ne soient

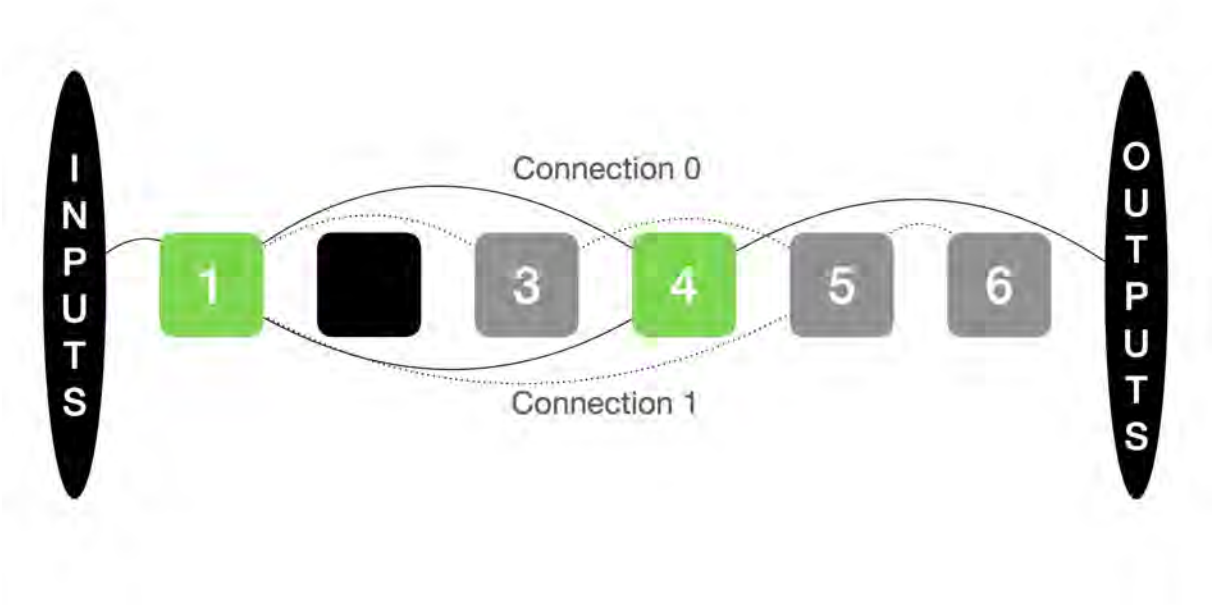


Figure 4.3: Un graphe avec une suppression du nœud à l'index 2

pas impactées. Spécifiquement, comme décrit dans l'Algorithme 5, tous les nœuds connectés au nœud supprimé sont à la place connectés à la Connexion 0 du nœud supprimé et chaque nœud après le nœud supprimé a ses connexions décrémentées de 1.

Algorithm 5: Suppression d'un nœud

```
Data: nodes is an array containing all nodes
Result: node deleted at position index in the graph
if active_nodes.length > 1 then
    index = getRandomActiveNode();
    for i ← 0 to outputs.length do
        if nodes.length − outputs[i] == index then
            outputs[i] = outputs[i]+nodes[index].conn0;
        end
    end
    for i ← 0 to active_nodes.length do
        if i − nodes[i].conn0 == index then
            nodes[i].conn0 = nodes[i].conn0 + nodes[index].conn0;
        end
    end
    for i ← index to nodes.length do
        if i − nodes[i].conn0 < index then
            nodes[i].conn0 = nodes[i].conn0 - 1;
        end
        if i − nodes[i].conn1 < index then
            nodes[i].conn1 = nodes[i].conn1 - 1;
        end
    end
    nodes.remove(index);
end
```

Ces opérateurs peuvent être bénéfiques pour l'évolution de CGP en agrandissant la topologie durant l'évolution plutôt qu'en cherchant dans un graphe de taille maximum fixe. Cependant, nous les étudierons dans le cas de GI, où des filtres d'images existants peuvent être améliorés grâce aux opérateurs d'insertion et de suppression de nœuds.

4.4 Expérimentations

Pour évaluer les opérateurs d'insertion et de suppression dans le contexte du GI, nous étudions l'effet d'utiliser un filtre fabriqué par un expert comme point de départ et en utilisant les opérateurs de mutations proposés :

baseline: standard CGP-IP. Le chromosome de départ est généré aléatoirement. Les opérateurs d'insertion et de suppression sont désactivés.

fixed size: standard CGP-IP. Le chromosome de départ est exclusivement composé de fonctions extraites depuis un filtre expert et est positionné au début du graphe. Les nœuds inactifs sont aléatoirement définis après les nœuds actifs du graphe. Les opérateurs d'insertion et de suppression sont désactivés.

fixed size with NOP: standard CGP-IP. Le chromosome de départ est exclusivement composé de fonctions extraites depuis un filtre expert et est positionné au début du graphe. Un nœud avec une fonction NOP est inséré entre chaque nœud actif. Par exemple, si le filtre contient 10 nœuds actifs, le génome initial sera composé de 20 nœuds actifs (10 nœuds avec des fonctions expertes et 10 nœuds avec des NOP). Les nœuds inactifs sont aléatoirement définis après les nœuds actifs du graphe. Les opérateurs d'insertion et de suppression sont désactivés.

adapting with NOP: Le génome initial est construit comme pour la méthode **fixed size**, par exemple avec un individu expert. Les opérateurs d'insertion et de suppression sont activés. Si une insertion se produit, seulement des fonctions NOP sont insérées.

adapting with random: Le génome initial est construit comme pour la méthode **fixed size**. Les opérateurs d'insertion et de suppression sont activés. Si une insertion se produit, la fonction du nœud inséré est choisie aléatoirement dans la librairie de fonctions.

adapting, no expert: Le chromosome de départ est généré aléatoirement. Les opérateurs d'insertion et de suppression sont activés. Si une insertion se produit, la fonction du nœud inséré est choisie aléatoirement dans la librairie de fonctions.

Ces configurations permettent des études détaillées et indépendantes des deux améliorations proposées pour CGP-IP dans ce chapitre : amélioration génétique d'un filtre d'images existant et des opérateurs de mutations structurelles d'insertion et de suppression de nœuds. La configuration **fixed size** isole le bénéfice possible de s'appuyer sur des filtres d'images, avec une distinction dans la façon dont le filtre expert est codé dans le génome de départ. Les trois configurations 'adaptive' permettent l'étude des opérateurs d'insertion

et de suppression de nœuds, en particulier pour leur utilisation en GI. Encore une fois, la distinction entre information génétique aléatoire supplémentaire et NOP est faite. Pour l'opérateur d'insertion de nœuds, la différence détermine si le phénotype fonctionnel d'un individu CGP-IP est modifié par l'insertion de nœuds (**adapting with random**) ou si la mutation d'insertion est seulement structurelle (**adapting with NOP**). Finalement, la configuration **adapting, no expert** permet une étude indépendante des avantages des opérateurs d'insertion et de suppression de nœuds lors du démarrage à partir de gènes aléatoires, comme avec la configuration **baseline**.

Pour ces travaux, nous avons privilégié le nombre d'itérations et de configurations pour chaque expérience plutôt que la taille de l'ensemble de données. Ainsi, ces travaux représentent le calcul de la précision de 576 000 individus, ce qui correspond à 24 jours de calcul sur un processeur Intel(R) Xeon(R) Gold 6230 CPU @ 2.10GHz.

4.4.1 Paramètres CGP-IP

Nous utilisons les paramètres pour CGP-IP suivants:

- R: le nombre de lignes dans CGP est 1
- C: le nombre de colonnes dans CGP est de 50 pour toutes les expérimentations mais peut évoluer avec les opérateurs d'insertion et de suppression
- r_{mut}: le taux de mutation pour chaque gène est de 0.25
- r_{ins}: le taux de l'opérateur d'insertion pour chaque gène est de 0.1
- r_{del}: le taux de l'opérateur de suppression pour chaque gène est de 0.1
- Le nombre d'îles est de 4
- λ: la taille de la population sur chaque île est de 4
- Intervalle de synchronisation entre chaque île : le nombre de générations avant que les îles ne comparent leur précision pour se mettre à jour avec le meilleur chromosome est de 20
- Nombre de générations: 1000 ou 2000 (dépend de l'expérimentation)

Chaque nœud du graphe est encodé avec 8 paramètres (voir la Table 4.1). L'allèle fonction représente un index dans la liste des fonctions de traitement de l'image. Le deuxième allèle, Connexion 0, est la connexion avec un précédent nœud où la sortie est utilisée pour l'entrée de la fonction. Le troisième allèle, Connexion 1, est la connexion avec un précédent nœud où la sortie est utilisée pour l'entrée de la fonction (toutes les fonctions n'utilisent pas Connexion 1). Les quatrième, cinquième et sixième allèles, Paramètre 0, 1 et 2, sont des nombres réels qui sont les premier, second et troisième paramètres de la fonction. Ces allèles ne sont pas forcément utilisés car toutes les fonctions n'ont pas trois paramètres. Par exemple, les paramètres Gabor Filter sont seulement utilisés avec les fonctions Gabor. Durant le processus d'évolution, les mutations peuvent se produire sur l'index de la fonction, sur une connexion ou sur un des paramètres.

Paramètre	**Type**	**Intervalle**
Fonction	INT	nombre de fonctions
Connexion 0	INT	nombre de nœuds/entrées
Connexion 1	INT	nombre de nœuds/entrées
Paramètre 0	REAL	$[-\infty, \infty]$
Paramètre 1	INT	[-16, 16]
Paramètre 2	INT	[-16, 16]
Gabor Filter Freq.	INT	[0, 16]
Gabor Filter Orien.	INT	[-8, 8]

Table 4.1: Paramètres d'un nœud

4.4.2 Les fonctions de traitement de l'image

L'ensemble des fonctions dans ce chapitre est commun à celui de CGP-IP [66]. Cependant, depuis la précédente publication [97], de nouvelles fonctions ont été ajoutées à la librairie OpenCV. C'est pour cela que nous avons mis à jour la liste des fonctions de traitement de l'image déjà existante avec les nouvelles fonctions *Watershed* et *Distance Transform* afin d'utiliser l'ensemble des fonctions disponibles dans la librairie OpenCV.

4.4.3 Ensemble de données

Dans ce chapitre, nous souhaitons construire un filtre d'image qui fournit une classification binaire d'une image en entrée, permettant la reconnaissance de différents types d'objets. Nous utilisons trois différents ensemble d'images: des images prises sur Mars par le rover

Spirit utilisées dans la publication [97], un ensemble d'images similaires mais prises sur la Lune et enfin un ensemble d'images du trafic urbain.

Mars

L'ensemble d'images de Mars est basé sur 5 images extraites parmi 1449 images qui composent le panorama McMurdo (Figure 4.4) prise par le rover Spirit sur Mars[2].

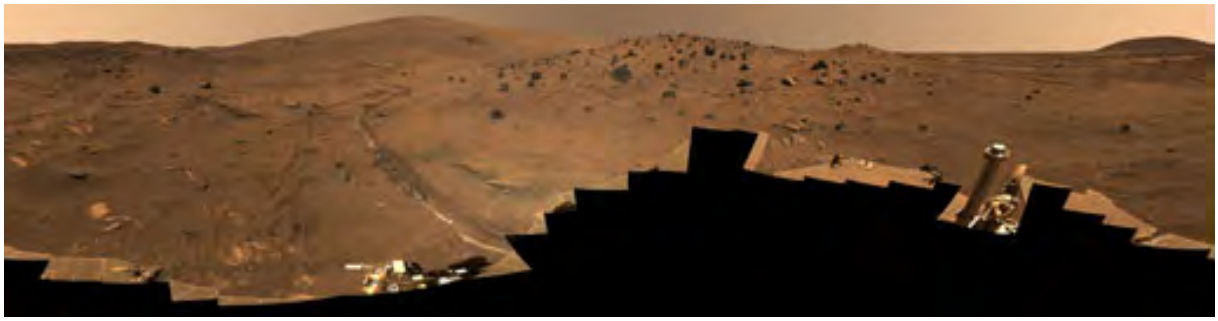
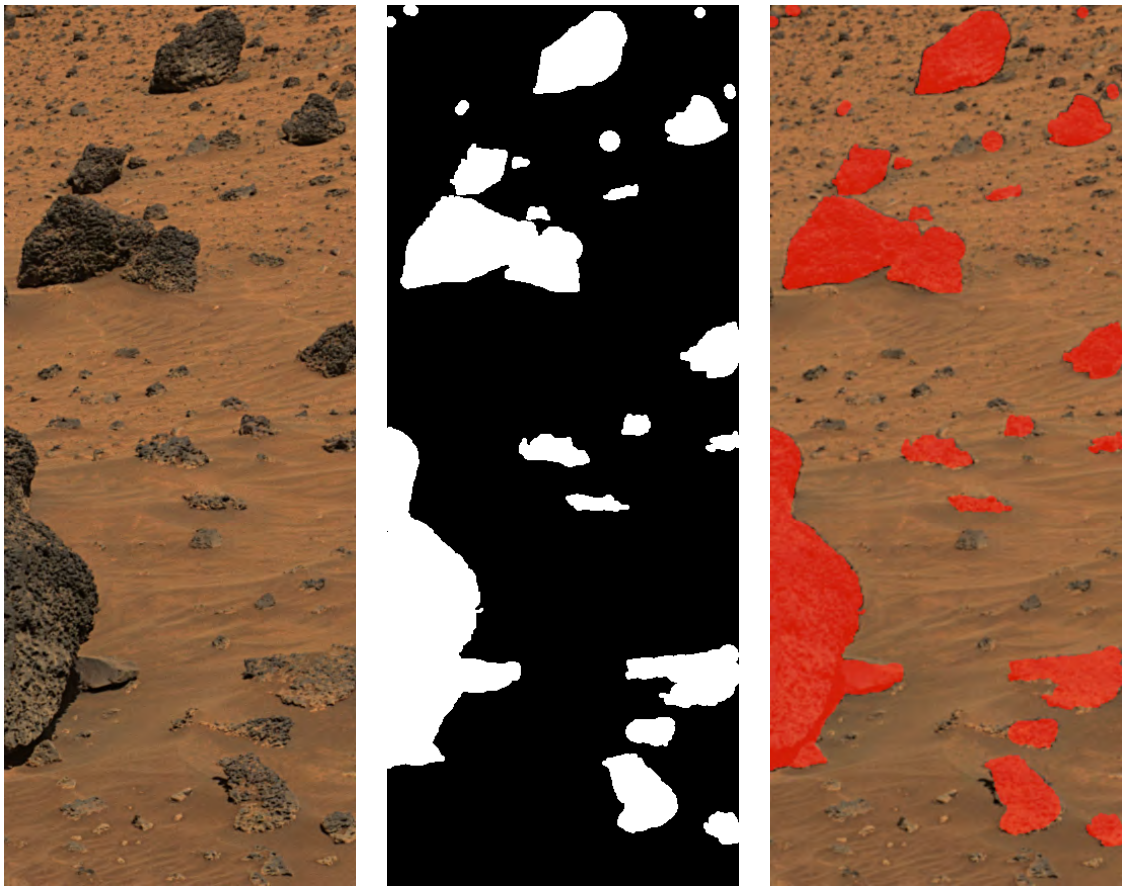


Figure 4.4: Image complète prise depuis le rover Spirit sur Mars

L'objectif de cet ensemble de données est d'extraire l'emplacement des rochers depuis une image. L'image en entrée (Table 4.5.A) a une résolution de 347x871 pixels et affiche des rochers sur le terrain Martien. L'image en sortie est un masque binaire (Table 4.5.B), qui identifie les pixels sur un rocher par 1 et les autres par 0.

Pour cet ensemble de données, nous utilisons un filtre proposé par [97], qui est décrit dans le Listing 4.1 et affiché dans la Figure 4.6. Ce filtre a été généré en utilisant CGP-IP sur le même ensemble de données et a déjà une précision élevée, ce qui nous permet de vérifier si une amélioration à l'aide de nos nouveaux opérateurs est possible. Pour cet ensemble de données, nous utilisons 1000 générations pour l'évolution sur 6 différents essais.

[2]http://pancam.sese.asu.edu/mcmurdo_v2.html



A : Input B : Output C : Overlay

Figure 4.5: Un exemple depuis l'ensemble de données d'images de Mars. L'objectif est d'identifier l'emplacement des rochers dans l'image.

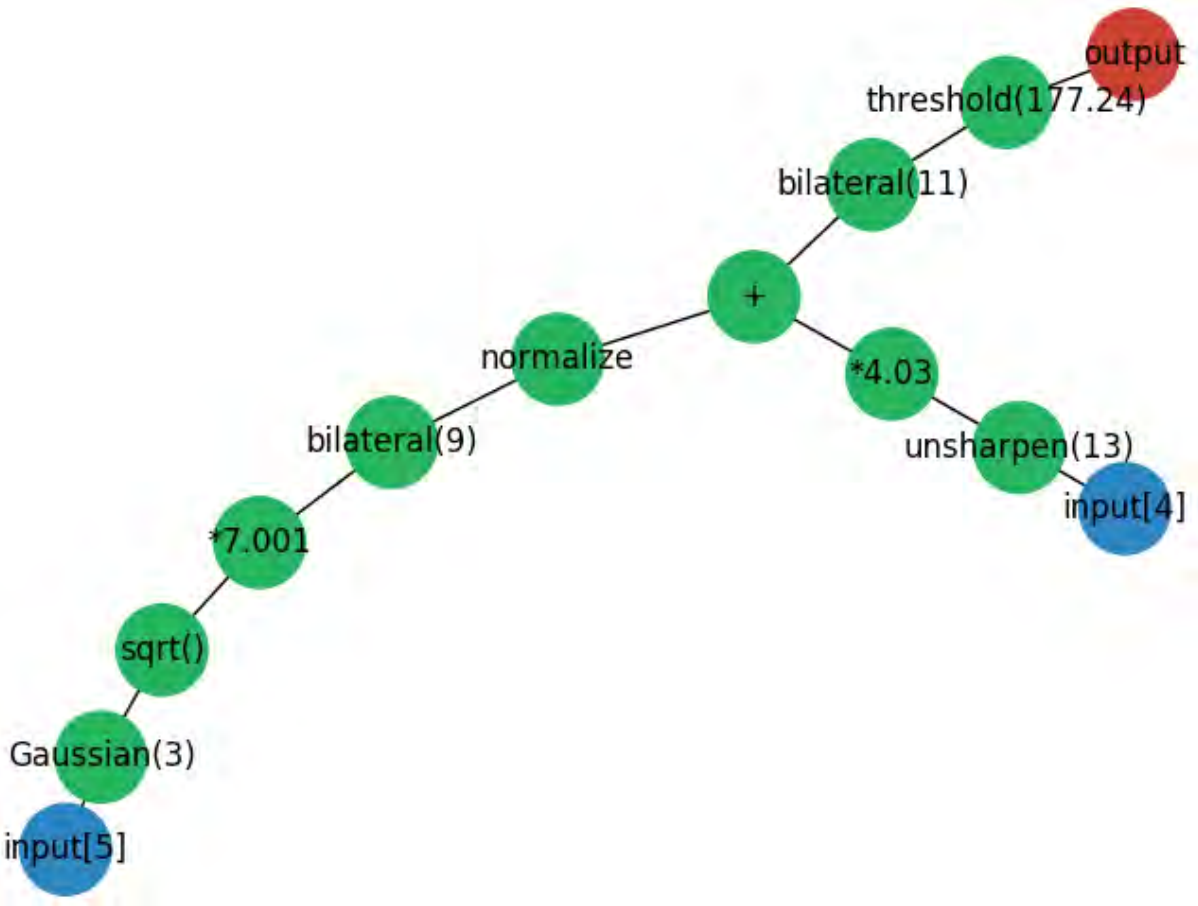


Figure 4.6: Graphe du gène évolué de Mars avec CGP-IP [97]

```python
def base_chromosome(input):
# input is composed of R[0], G[1], B[2], H[3], S[4], V[5]
node0 = cv2.GaussianBlur(input[5],(3,3))
node1 = np.sqrt(node0)
node2 = input[4]
node3 = cv2.unsharpen(node2,13)
node4 = node1*7.001
node5 = cv2.bilateralfilter(node4,9)
node6 = normalize(node5)
node7 = node3*4.03
node8 = node7 + node6
node9 = cv2.bilateralfilter(node8,11)
node10 = cv2.threshold(node9,177.24,255)
return node10
```

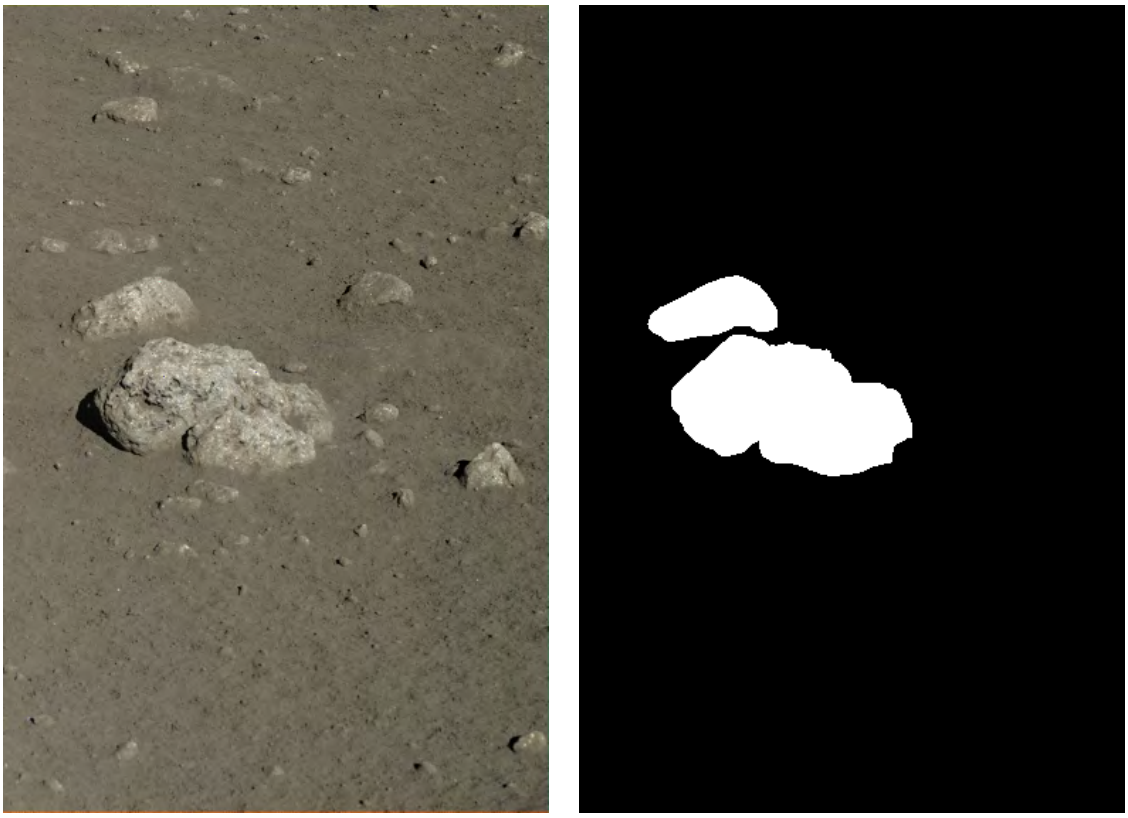
Listing 4.1: Encodage du gène de base de Mars en python

La Lune

L'ensemble de données Lune[3] est basé sur 5 images extraites parmi 9,766 rendus réalistes de paysages rocheux sur la Lune et leurs segmentations équivalentes (les 3 classes sont le ciel, les petits rochers et les larges rochers). Cet ensemble de données a été créé par Romain Pessia et Genya Ishigami du Space Robotics Group[4], Keio University, Japan. Tout comme pour l'ensemble de données Mars, l'objectif est d'extraire l'emplacement des rochers depuis une image. L'image en entrée (Table 4.7.A) a une résolution de 720x480 pixels et affiche des rochers sur le terrain Martien. L'image en sortie est un masque binaire (Table 4.7.B), qui identifie les pixels sur un rocher par 1 et les autres par 0.

Nous utilisons le même filtre que proposé par [97] et utilisé sur l'ensemble de données Mars (Figure 4.6). Cela nous permet d'étudier l'adaptation d'un filtre depuis un ensemble de données à un autre, où la taille ainsi que les couleurs des images changent et où l'environnement des images en sortie est différent. Pour cet ensemble de données, nous utilisons 2000 générations pour l'évolution sur 6 différents essais.

[3]https://www.kaggle.com/romainpessia/artificial-lunar-rocky-landscape-dataset
[4]http://www.srg.mech.keio.ac.jp/index_en.html



A : Input B : Output

Figure 4.7: Un exemple de l'ensemble de données d'images de la Lune

Trafic urbain

Dans cette expérience, nous appliquons CGP-IP pour identifier les objets en mouvement dans un environnement urbain. L'objectif de l'ensemble de données est de construire un filtre qui extrait et suit des objets spécifiques dans la vidéo. Pour cela, le filtre doit trouver les objets qui se sont déplacés d'une image à la suivante. L'ensemble de données a été créé avec des vidéos du trafic urbain [5]. Nous utilisons des vidéos d'une durée de 5 minutes en couleur RGB 16bit et avec une résolution de 1024x576 pixels. Les images sont converties en nuances de gris pour les entrées (Figure 4.9.A et Figure 4.9.B). L'image en sortie (Figure 4.9.C) a été généré avec Mask-RCNN [74] et identifie les larges objets tels que les piétons, les vélos, les voitures, etc.

Le filtre de départ utilisé dans cet ensemble de données a été créé par des ingénieurs. Il fonctionne en soustrayant les deux images en entrée (Figure 4.9.A et Figure 4.9.B) et en appliquant des fonctions d'érosion et de dilatation pour réduire le bruit dans l'image. Ce filtre est détaillé dans le Listing 4.2 et affiché dans la Figure 4.8. Pour cet ensemble

[5]https://camstreamer.com/live/streams/14-traffic

de données, nous utilisons 2000 générations pour l'évolution sur 6 différents essais.

```
def base_chromosome(input1,input2):
# input1 is composed of R[0], G[1], B[2]
# input2 is composed of R[0], G[1], B[2]
node0 = input1[0] + input1[1]
node1 = node0 + input1[2]
node2 = node1 / 3
node3 = input2[0] + input2[1]
node4 = node3 + input2[2]
node5 = node4 / 3
node6 = node5 - node2
node7 = cv2.threshold(node6,50,255)
node8 = cv2.dilate(node7)
node9 = cv2.dilate(node8)
return node9
```

Listing 4.2: Python encoding of the Urban Traffic gene base

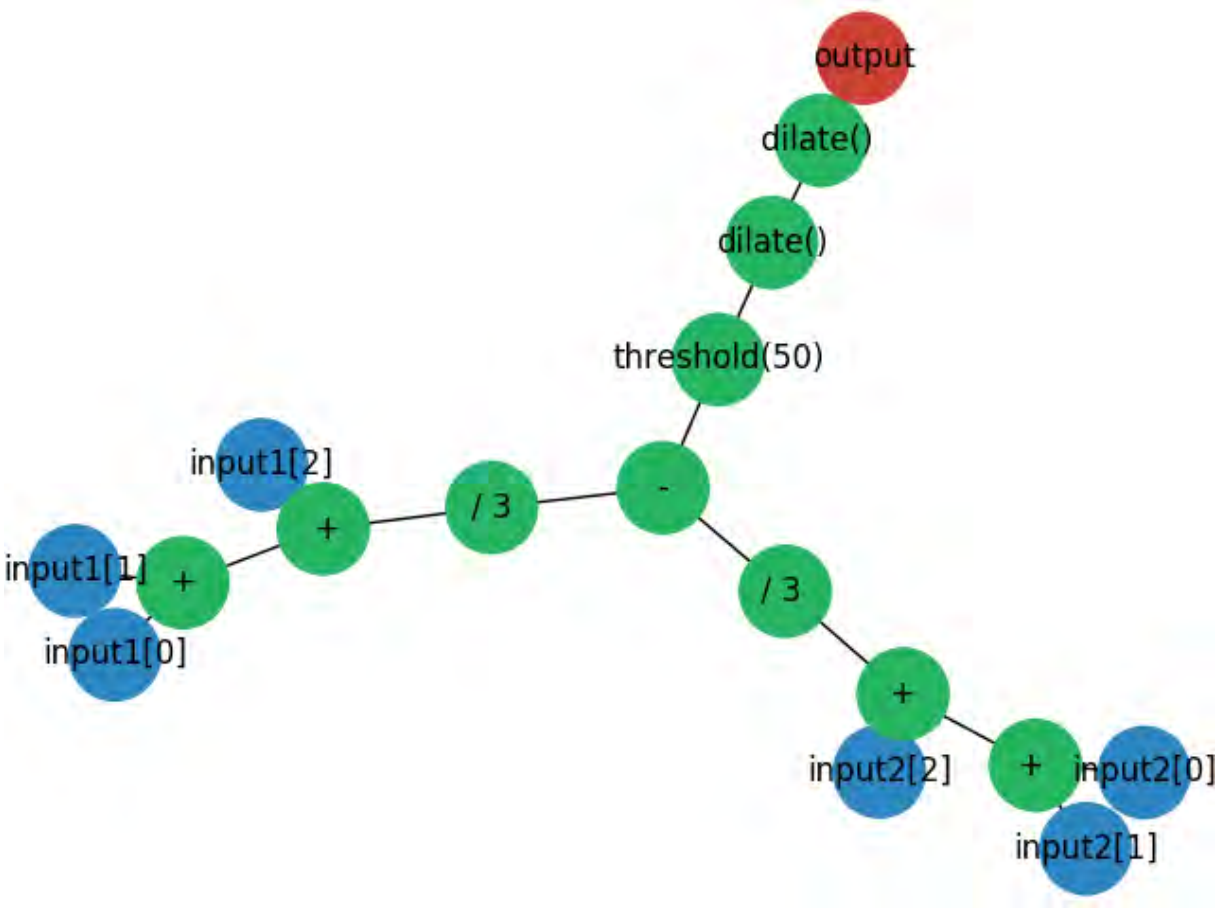


Figure 4.8: Graphe d'un gène pour l'ensemble de donnée du trafic urbain créé par un expert

A : Input 1 B : Input 2 C : Output

Figure 4.9: Exemple de l'ensemble de données d'images du trafic urbain

4.5 Résultats

Dans cette section, nous présentons les résultats obtenus sur ces trois ensembles de données. CGP-IP a été capable de construire des filtres d'images qui classifient correctement les objets désirés dans chaque cas, mais les expériences avec les nouveaux opérateurs montrent un bénéfice net comparé aux versions standards.

La Figure 4.10 montre l'évolution sur 2000 générations sur l'ensemble de données Mars. **Adapting with random** converge plus rapidement et avec une meilleure classification que les autres méthodes. Les autres configurations montrent peu de différences. **Adapting with NOP** termine avec la plus mauvaise précision, inférieure à **adapting with random** et **fixed size with NOP**. Ceci démontre que l'ajout de nœuds avec fonction NOP n'est pas avantageux. L'ajout de fonctions aléatoires qui changent le phénotype du programme est préférable.

La similarité des résultats sur l'ensemble de données Mars n'est pas surprenant sachant que l'individu de départ était extrait d'un précédent CGP-IP. Toutefois, il est à noter que **baseline** et **adapting, no expert**, qui n'utilisent pas ce point de départ, ont convergé pour obtenir les mêmes résultats que les autres individus.

La Figure 4.11 affiche la taille du graphe actif pour l'ensemble de données de Mars. **baseline**, **fixed size** et **fixed size with NOP** croit doucement pour arriver à 5/6 noeuds actifs. Pour **adapting with NOP**, le nombre de nœuds actifs croit constamment tout au long de l'évolution. **Adapting with random** et **adapting, no expert** converge vers la même taille grâce à l'utilisation des mêmes opérateurs. Il est évident que les méthodes avec insertion de nœuds produisent des graphes plus larges même si un opérateur de suppression est aussi présent.

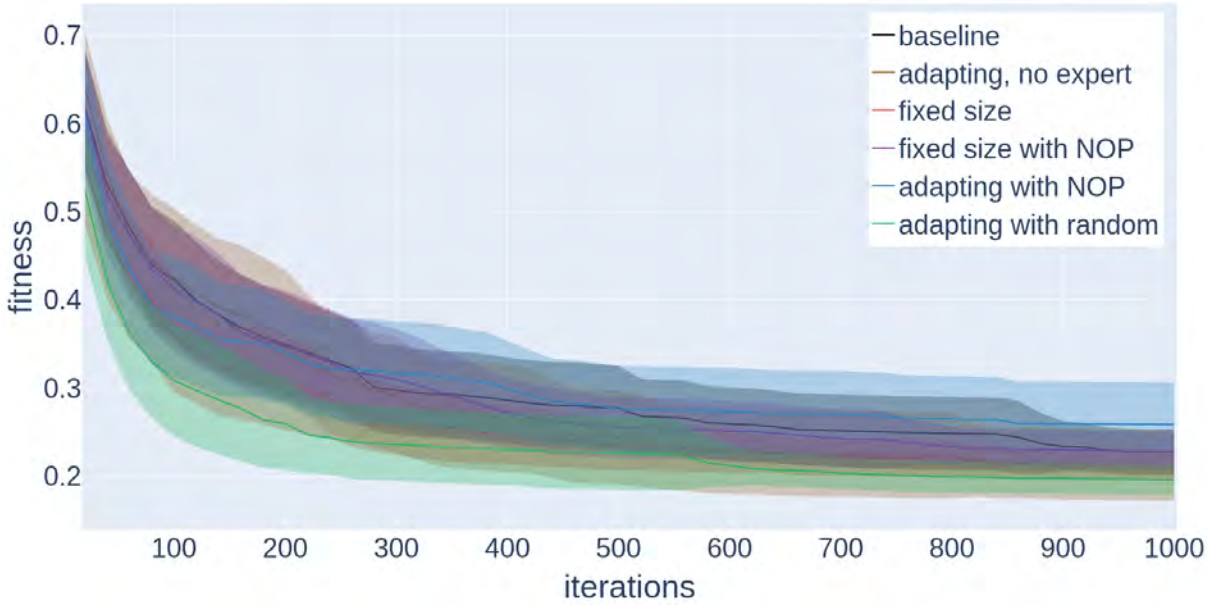


Figure 4.10: Moyenne et écart-type de la précision pour l'ensemble de données de Mars sur 40 séries

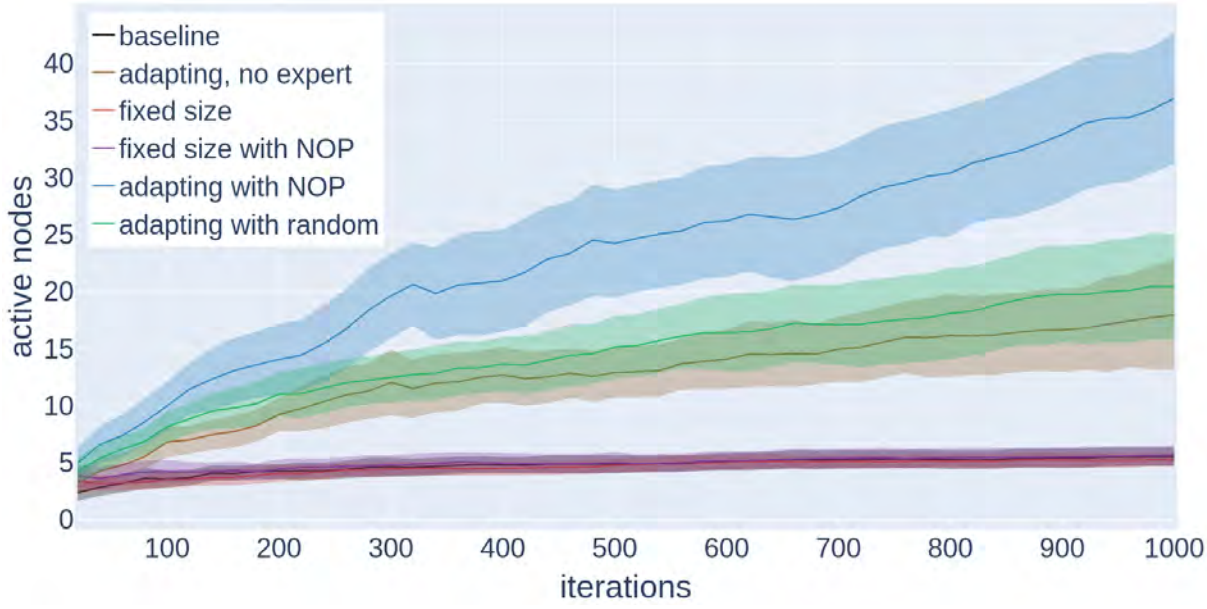


Figure 4.11: Moyenne et écart-type du nombre de nœuds actifs pour l'ensemble de données de Mars sur 40 séries

La Figure 4.12 affiche l'évolution au travers de 2000 générations sur l'ensemble de données Lune. **Adapting with random** performe mieux et plus vite que les cinq autres méthodes avec un écart-type et une valeur t-test p inférieure à $1e^{-5}$. **Baseline** démarre plus doucement que les cinq autres méthodes avec un plus grand écart-type et la plus mauvaise précision. Bien que **adapting, no expert** converge vers la même précision que les autres méthodes, le bénéfice de démarrer avec un filtre expert est démontré au

travers de la bonne performance de **baseline**. Il est également à noter que l'expert utilisé dans cet ensemble de données a été initialement formé sur l'ensemble de données Mars, démontrant qu'un filtre peut être transféré d'une expérience à une autre.

La Figure 4.13 montre que **baseline**, **fixed size** et **fixed size with NOP** croissent doucement pour atteindre 7 nœuds actifs. Pour cet ensemble de données, **adapting with random** croise plus que **adapting with NOP**. Les méthodes **adapting with random** et **adapting with NOP** atteignent des tailles de graphes actifs considérablement apres 2000 générations, surpassant même la taille initiale de 50 nœuds.

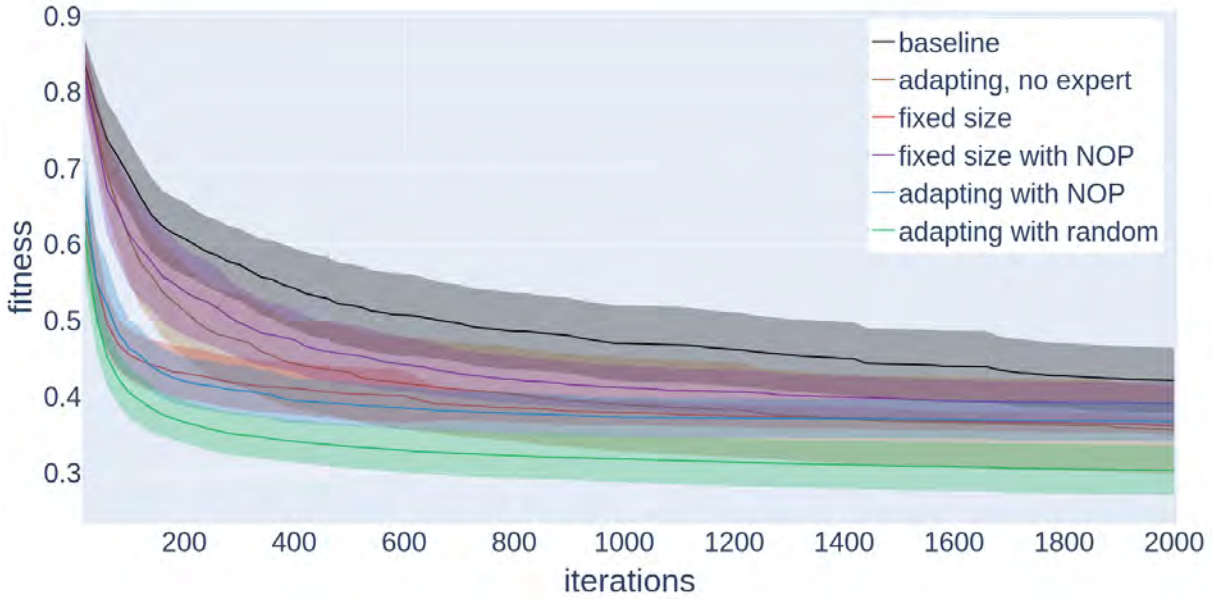


Figure 4.12: Moyenne et écart-type de la précision pour l'ensemble de données de la Lune sur 40 séries

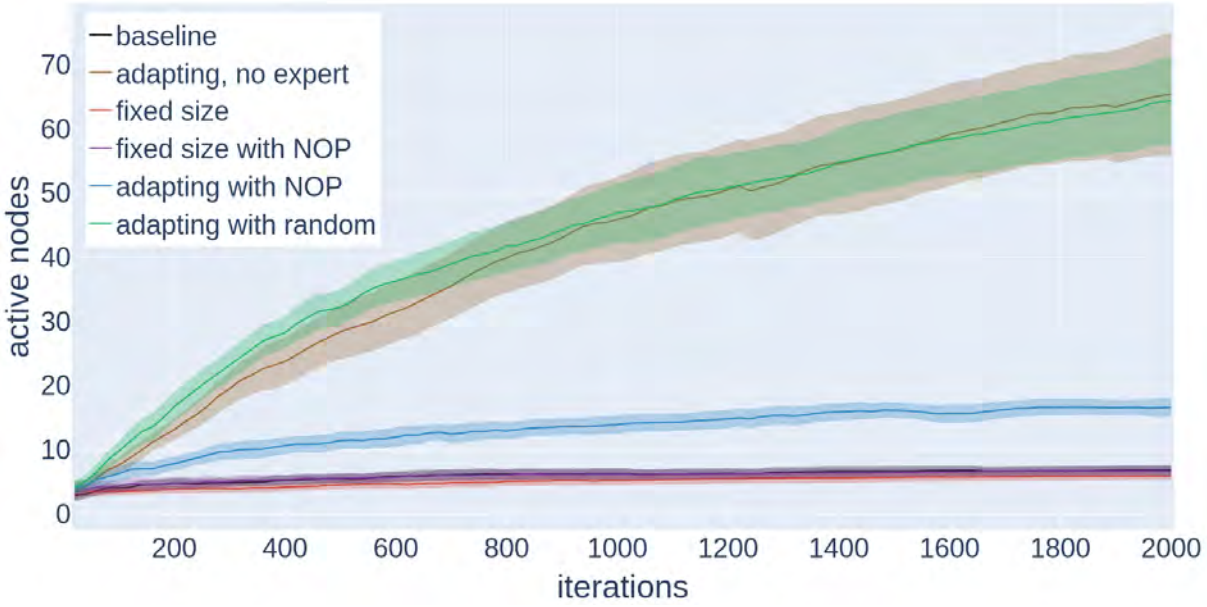


Figure 4.13: Moyenne et écart-type du nombre de nœuds actifs pour l'ensemble de données de la Lune sur 40 séries

La Figure 4.14 montre l'évolution de la fonction de précision après 2000 itérations sur l'ensemble de données du trafic urbain. **Adapting with random** surpasse et converge plus vite que **baseline**, **fixed size**, **fixed size with NOP** et **adapting with NOP** (valeur p du ttest $<1e^{-5}$ à 300 itérations mais aussi à 2000 itérations). De plus, l'écart-type sur 40 séries est le plus faible comparé aux autres méthodes. Cela signifie que **adapting with random** performe mieux que les autres méthodes mais aussi qu'il produit des solutions de qualité similaire indépendamment du caractère aléatoire du processus évolutif. **Adapting, no expert** performe plus lentement que **adapting with random** mais atteint une meilleure précision après 700 itérations et continue de s'améliorer. Cela démontre le potentiel désavantage de démarrer avec un individu créé par un expert, qui est une convergence précoce basée sur cet individu et un manque d'exploration par rapport à une initialisation aléatoire.

La Figure 4.15 montre que **fixed size** et **fixed size with NOP** réduisent la taille à 7 nœuds actifs. Comme dans l'ensemble de données Lune, **adapting with random** continue d'augmenter tout au long de l'évolution même s'il y a une étape de compression claire visible autour de 500 générations pour **adapting, no expert** qui peut démontrer l'avantage de l'opérateur de suppression.

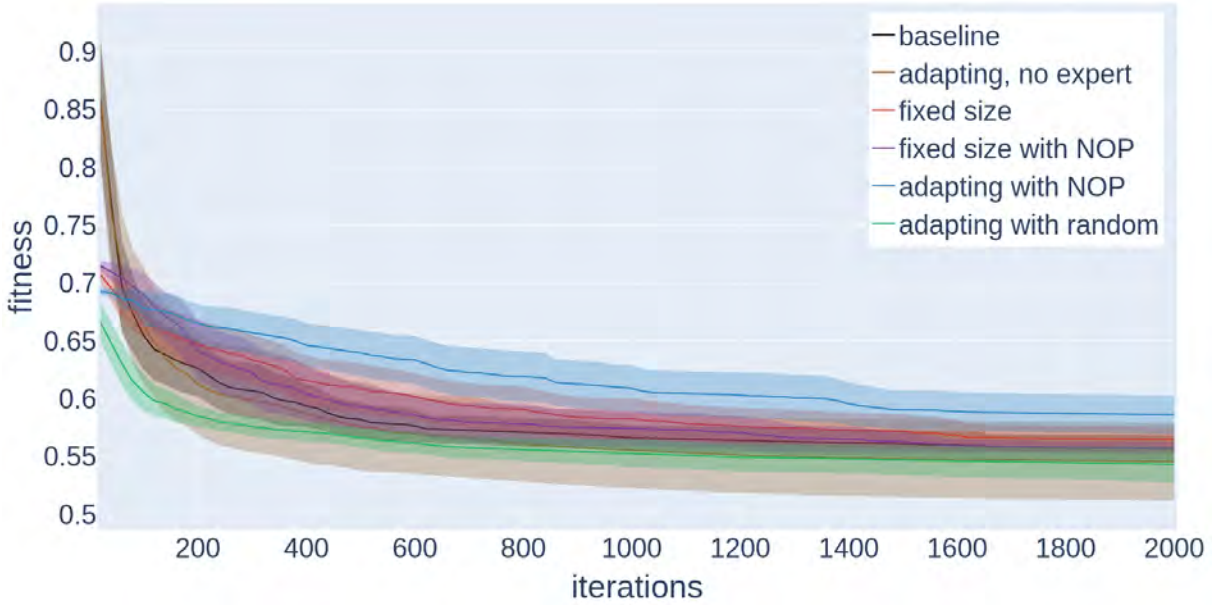


Figure 4.14: Moyenne et écart-type de la précision pour l'ensemble de données du trafic urbain sur 40 séries

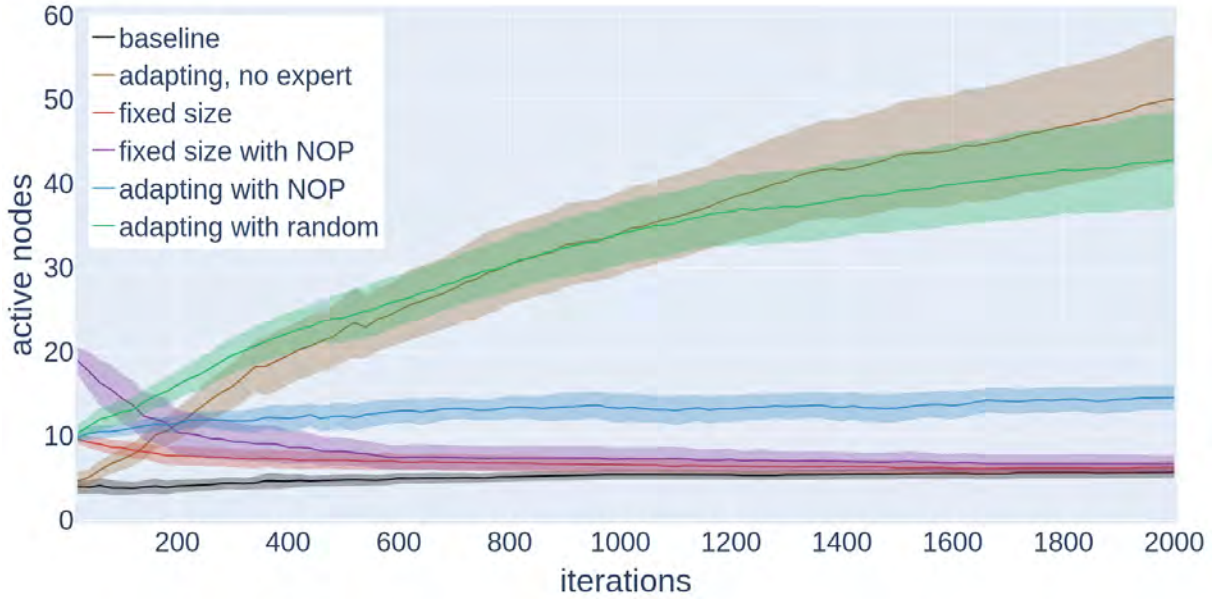


Figure 4.15: Moyenne et écart-type du nombre de nœuds actifs pour l'ensemble de données du trafic urbain sur 40 séries

Pour chaque expérimentation (Table 4.2), **adapting with random** performe mieux et plus vite que les CGP-IP standard avec un écart-type faible et une valeur p de t-test $<1e^{-5}$. Notre évolution avec ces fonctions aléatoires surpasse CGP-IP standard (**baseline**) et **fixed size with NOP** dans chaque cas. **Adapting with NOP** converge un peu plus vite et termine avec une précision plus faible. Comme attendu, **baseline** converge plus lentement avec un plus grand écart-type que les autres. Les opérateurs de mutation pro-

	Lune	Mars	Trafic urbain
Baseline	0.42 (0.09)	0.23 (0.05)	0.56 (0.02)
Adapting, no expert	0.36 (0.12)	0.21 (0.08)	0.55 (0.07)
Fixed sized	0.36 (0.02)	0.21 (0.04)	0.56 (0.02)
Fixed size with NOP	0.39 (0.05)	0.23 (0.04)	0.56 (0.01)
Adapting with NOP	0.37 (0.05)	0.26 (0.1)	0.59 (0.03)
Adapting with random	**0.3 (0.07)**	**0.2 (0.03)**	**0.54 (0.03)**

Table 4.2: Moyenne et écart-type pour chaque méthode sur chaque ensemble de données

posés (insertion de nœuds avec des fonctions NOP ou des fonctions aléatoires) augmente considérablement le nombre de nœuds actifs dans le graphe contrairement à CGP-IP standard où le nombre de nœuds actifs est relativement stable. Avec l'ensemble de données du trafic urbain, **adapting, no expert** performe mieux que **adapting with random** après 700 itérations ce qui signifie qu'il est important de partir d'un chromosome efficace sinon l'amélioration génétique va être dépassée par l'évolution classique sur un chromosome aléatoire.

4.6 Conclusion

Dans ce chapitre, nous avons proposé des opérateurs de mutations pour CGP-IP à utiliser dans le contexte du GI. Notre algorithme ajoute de nouvelles fonctions dans le graphe tout en gardant ses connexions existantes intactes. Nous avons testé CGP-IP avec ces nouveaux opérateurs sur 3 ensembles de données et nous avons montré qu'il surpasse constamment CGP-IP standard en augmentant la vitesse de convergence et la précision finale sur tous les ensembles de données.

Nos méthodes ont l'intérêt de légèrement mais constamment augmenter le nombre de nœuds actifs. Cela permet à l'évolution d'accéder à de nouveaux espaces de recherche permettant de mener à une meilleure précision. Cette conclusion est alignée avec les précédents travaux sur la croissance progressive des réseaux de neurones [161] ou les réseaux de régulation de gènes [25]. Par exemple, l'algorithme NeuroEvolution of Augmenting Topologies (NEAT) [161] démontre qu'augmenter la complexité du programme tout au long de la recherche peut améliorer l'optimisation. Cela est cohérent avec les résultats de ce chapitre, que ce soit à partir d'un filtre d'images créé par un expert ou bien d'une population initiale tirée aléatoirement.

Une piste à explorer avec cette méthode est la réduction de la complexité des graphes

au cours du temps. Les opérateurs d'insertion de nœuds sont clairement bénéfiques pour l'évolution mais ils réintroduisent le problème du gonflement dans CGP [122] alors que CGP visait à éviter ce problème [166]. Effectivement le nombre de nœuds actifs ne cesse de croitre alors que la précision reste stable lors des dernières itérations. Nous prévoyons d'étudier différents taux de mutation pour voir si les larges graphes peuvent être réduits lors de l'évolution.

En pratique, cette méthode peut être utilisée pour converger plus rapidement vers une meilleure solution en utilisant soit un filtre efficace créé par un expert ou depuis un chromosome aléatoirement généré. Cela permet l'application de cette méthode qui s'appuie sur les pipelines de traitement d'images existants, en particulier ceux qui utilisent CGP-IP.

Chapter 5

Optimisation multiobjectifs de filtres d'images

Lors de l'amélioration d'un filtre dans le précédent Chapitre 4, l'augmentation de la précision du filtre a entrainé une augmentation de son temps d'exécution qui est un élément critique sur un système temps réel. Dans ce chapitre, une approche multiobjectifs va être développée afin de prendre en compte le couple précision du filtre et temps d'exécution du filtre afin d'obtenir une solution qui améliore la précision tout en gardant un temps d'exécution du filtre acceptable. Ce chapitre est structuré de la manière suivante. L'état de l'art sur les algorithmes génétiques multiobjectifs est présenté dans la section 5.1. La section 5.2 décrit l'adaptation de CGP-IP-GI pour intégrer NSGA2 afin de réaliser du multiobjectifs. La section 5.3 présente l'expérience sur laquelle les travaux ont étés étudiés ainsi que ses résultats dans la section 5.4. Enfin, la section 5.5 présente les conclusions préliminaires. Ce chapitre a été proposé pour une publication pour la conférence GECCO 2022 [1].

5.1 Travaux précédents

Nous allons tout d'abord étudier les précédents travaux dans le domaine des problèmes à objectif multiples et tout particulièrement dans le domaine des algorithmes génétiques.

[1] https://gecco-2022.sigevo.org/

5.1.1 Algorithmes génétiques multiobjectifs

A la différence d'un problème mono objectif où il existe une solution optimale, un problème multiobjectifs va présenter une multitude de solutions plus ou moins performantes sur plusieurs critères. L'amélioration d'un objectif dans un problème multiobjectifs peut entrainer la dégradation des autres objectifs. Les algorithmes génétiques multiobjectifs proposent donc un ensemble de solutions performantes appartenant au front Pareto.

Le front Pareto

Une solution peut être la meilleure, la pire ou égale à d'autres solutions en fonction des valeurs de ses objectifs. Une solution non dominée signifie qu'une solution n'est pas la pire dans aucun des objectifs mais est meilleure que les autres solutions dans au moins un objectif. Les solutions Pareto-Optimale sont les solutions non dominées par une autre solution dans l'espace de recherche [30].

Dominance

Soit $\mathbf{u}$ = (u1, u2, ..., um) et $\mathbf{v}$ = (v1, v2, ..., vm) deux vecteurs de décision. $\mathbf{v}$ domine $\mathbf{u}$ ($\mathbf{u} \leq \mathbf{v}$) pour un problème de minimisation si et seulement si :

$$\begin{cases} \forall i \in 1, ..., n \quad f_i(\mathbf{v}) \leq f_i(\mathbf{u}) \\ \exists i \in 1, ..., n \quad f_i(\mathbf{v}) < f_i(\mathbf{u}) \end{cases} \tag{5.1}$$

L'ensemble de toutes les solutions dominantes est appelé Front de Pareto. Le véritable but de l'optimisation multiobjectifs est de trouver ou d'approcher le front de Pareto et de permettre une répartition équitable sur ce front (Figure 5.1).

Multi-Objective Genetic Algorithm (MOGA)

MOGA a été développé par Fonseca et Fleming [50]. Des groupes sont formés en commençant par les solutions non dominées dans le premier groupe et les solutions dominées dans le deuxième groupe. Dans chaque groupe, la proximité des solutions entre elles est calculée, puis le paramètre de précision (σ_{share}) est calculé ([50] §4.1). Le paramètre de précision (σ_{share}) permet d'appliquer MOGA dans divers problèmes d'optimisation. Cependant, dans cette technique, donner la même valeur de précision à différentes solutions sur le même front peut amener l'algorithme à rechercher les solutions sur le mauvais front.

***Non-dominated Sorting Genetic Algorithm* (NSGA)**

NSGA a été développé par Srivas et Deb [160]. La méthode commence par trier aléatoirement les solutions obtenues en fonction de leur dominance. Comme dans la méthode MOGA, la proximité entre les solutions dans chaque sous-population est calculée. Les solutions sont sélectionnées avec la méthode de la roulette. Dans cette méthode, la chance que les solutions de la 1ère sous-population non dominée soient sélectionnées est plus élevée. De nouvelles solutions sont obtenues en appliquant un croisement et une mutation sur les solutions sélectionnées et l'algorithme continue de chercher des solutions jusqu'à ce que le critère d'arrêt soit atteint. La méthode NSGA classe les solutions en fonction de leur dominance et attribue des valeurs de précision. NSGA a été utilisé dans une étude visant à obtenir une puissance maximale, une efficacité thermique et une perte de charge minimale dans la conception de moteurs thermiques Stirling [4].

***Elitist Non-dominated Sorting Genetic Algorithm* (NSGA2)**

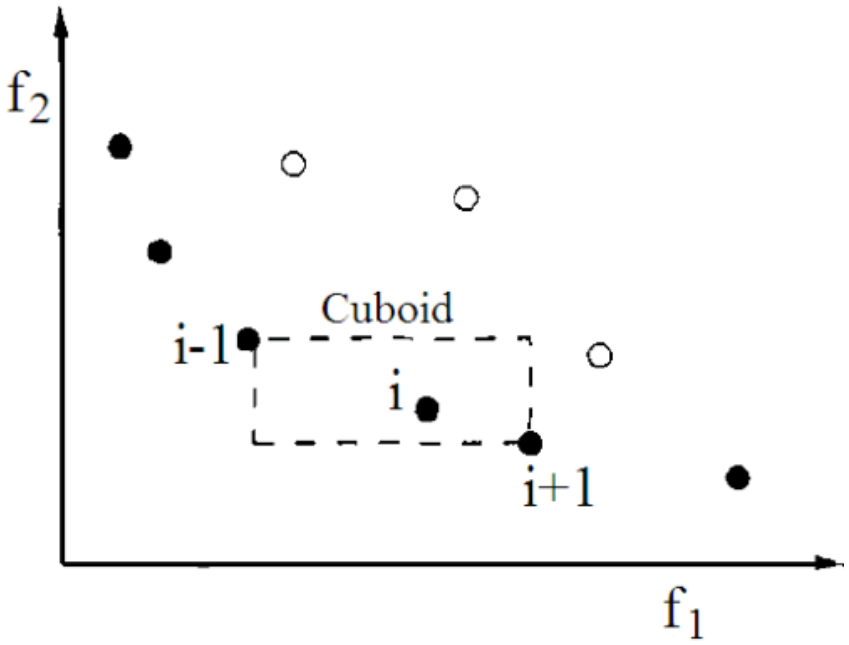


Figure 5.1: Calcul de la distance de *crowding*. Les points marqués dans des cercles pleins sont des solutions du front Pareto [31]

NSGA 2 a été développé par Deb et Goel [34, 33]. Cet algorithme est similaire à la méthode NSGA; cependant, les paramètres utilisés dans NSGA ne sont pas utilisés dans NSGA2. L'algorithme sélectionne les résultats les plus isolés avec la distance de *crowding* (Figure 5.1), le paramètre de précision (σ_{share}) n'a plus besoin d'être calculé. Cependant,

les points extrêmes doivent être conservés à chaque génération et, par conséquent, se voir attribuer une distance de *crowding* à l'infini. Par conséquent, l'algorithme ne perd jamais les solutions optimales sur le front de Pareto trouvées jusqu'à l'étape en cours. Le mécanisme de sélection de solution est utilisé pour limiter la taille de la population ; cependant, dans ce cas, l'algorithme peut perdre sa caractéristique de proximité avec la solution optimale. Puisque le nombre de solutions où existent les premières solutions non dominées n'est pas supérieur au nombre de populations principales, toutes les solutions de cet ensemble sont sélectionnées.

***Strength Pareto Evolutionary Algorithm* (SPEA)**

SPEA a été développé par Zitzler et Thiele [41]. Dans cet algorithme, la diversité dans la sélection des solutions optimales de Pareto est assurée par l'analyse de clustering. Le calcul des proximités est facile et aucun paramètre supplémentaire n'est requis. Dans l'algorithme, le paramètre N_t, qui correspond à la taille des populations dans lesquelles les solutions importantes seront recherchées, doit être identifié. De plus, l'équilibre entre la taille de la population principale N et N_t doit être assuré afin d'obtenir de bons résultats de l'algorithme. N_t ne doit être ni trop grand ni trop petit. S'il est trop grand, l'algorithme perd beaucoup de temps avec des solutions proéminentes et peut ne pas générer d'autres solutions et converger vers des solutions optimales. S'il est trop petit, les solutions de l'ensemble proéminent ne sont pas suffisamment utilisées et l'algorithme peut rechercher excessivement les solutions en dehors de la zone où existent des solutions optimales.

***Strength Pareto Evolutionary Algorithm 2* (SPEA2)**

SPEA2 est une évolution de la méthode SPEA, proposé par Zitzler et al. [194]. SPEA2 introduit un meilleur mécanisme de notation, une technique d'estimation de l'intensité et une gestion des archives (communauté secondaire) développée par rapport à la méthode SPEA. SPEA2 utilise une stratégie d'affectation de précision à grain fin en utilisant des informations d'intensité. De plus, la taille de l'archive stockant en externe les individus non dominés est stable. Si le nombre d'individus non dominés est inférieur à la taille de l'archive précédemment identifiée, l'archive est remplie d'individus non dominés. En outre, la technique de regroupement utilisée lorsque la surface non dominée dépasse la taille de l'archive a été remplacée par une méthode alternative de réduction des effectifs

qui présente des caractéristiques similaires mais ne perd pas les points du front de Pareto. Enfin, une autre différence par rapport à SPEA est que seuls les membres de l'archive sont utilisés dans le processus de sélection.

5.2 Implémentation multiobjectifs dans CGP-IP

Nous allons utiliser l'algorithme NSGA2 afin de réaliser une optimisation multiobjectifs. Pour cela, il est nécessaire d'adapter CGP-IP-GI en modifiant son algorithme évolutionnaire et en utilisant un front de Pareto comme fonction d'évaluation.

5.2.1 Algorithme évolutionnaire $\mu + \lambda$ avec $\mu > 1$

CGP-IP-GI utilise un algorithme évolutionnaire $\mu+\lambda$ avec $\mu = 1$ afin de générer λ enfants à partir de $\mu = 1$ parent. Pour que le front de Pareto soit représentatif, il est nécessaire de pouvoir utiliser $\mu > 1$. La phase d'évolution est modifiée pour pouvoir générer λ enfants depuis μ parents en utilisant l'Algorithme 6.

Algorithm 6: Algorithme évolutionnaire de μ parents en λ enfants

Data: parents est un tableau contenant μ parents

λ correspond au nombre d'enfants

Result: enfants contient les λ enfants

$nb\_enfants = 0$;

while $\lambda > 0$ **do**

  enfants[$nb\_enfants$] = evolve(parents[$nb\_enfants\%parents.length$]);

  $\lambda = \lambda - 1$;

  $nb\_enfants = nb\_enfants + 1$;

end

5.2.2 Adaptation de NSGA2 pour CGP-IP-GI

Afin de préserver plusieurs candidats après chaque phase de sélection, il est nécessaire de remplacer la fonction d'évaluation ne prenant en compte qu'un seul objectif et ne gardant que la meilleure solution par un algorithme NSGA2 qui va prendre en compte deux objectifs et permettre de sélectionner et retenir un ensemble de solutions. L'algorithme NSGA2 se décompose en deux phases : la première élimine les solutions dominées et la

deuxième garde les solutions les plus isolées en les triant.

Sélection des solutions non dominées

Les solutions n'appartenant pas au front de Pareto ou solutions dominées sont supprimées. Seules les solutions non dominées sont retenues (Algorithme 7). Une solution x ∈ E domine si x' ∈ E si :

$$\forall i, f_i(x) \leq f_i(x') et \exists i, f_i(x) < f_i(x') \tag{5.2}$$

avec $f_i(x)$ la fonction de l'objectif **i**

Algorithm 7: Suppression des solutions dominées

```
Data: solutions est un tableau contenant toutes les solutions
Result: frontpareto contient les solutions sur le front Pareto
solutions.sort([fitness,duration]);
for i ← 0 to solutions.length − 1 do
    front = True;
    for j ← i + 1 to solutions.length do
        if solutions[i].duration≥solutions[j].duration then
            front = False;
        end
    end
    if front then
        frontpareto.push(solutions[i])
    end
end
frontpareto.push(solutions[-1]);
```

Tri par distance de *crowding*

Si le nombre de solutions non dominées est supérieur au nombre de parents μ, il est nécessaire de faire un tri et de n'en retenir que μ. Les points extrêmes doivent être conservés à chaque génération et, par conséquent, se voir attribuer une distance de *crowding* infinie. Pour chaque solution entre les extrémités, la distance de *crowding* entre la solution

précédente et la solution suivante est calculée. Les solutions sont triées en fonction de la distance de *crowding.* et les μ solutions ayant la plus grande distance sont conservées (Algorithme 8).

Algorithm 8: Tri des solutions par distance de *crowding* et sélection de μ solutions

Data: solutions est un tableau contenant toutes les solutions

Result: parents contient les μ solutions à utiliser comme parents

for $i \leftarrow 1$ **to** $solutions.length - 1$ **do**

    solutions[i].crowding = abs(solutions[i-1].fitness - solutions[i+1].fitness) +
     abs(solutions[i-1].duration - solutions[i+1].duration)

end

parents.push(solutions[0]);

parents.push(solutions[-1]);

solutions.sort(crowding);

for $i \leftarrow 0$ **to** $\mu - 2$ **do**

    parents.push(solutions[i]);

end

5.2.3 Synchronisation des îles

CGP-IP-GI utilise une répartition d'individus qui évoluent sur différentes iles. Les îles sont synchronisées à un intervalle fixe. La synchronisation consiste à prendre le meilleur des individus de toutes les îles et de l'implanter comme individu parent sur toutes les îles. Le processus de synchronisation est adapté pour utiliser l'algorithme NSGA2. Lors de la synchronisation, tous les individus de chaque île sont réunis et seuls les individus appartenant au front Pareto sont gardés. Si le nombre d'individus retenus sur le font Pareto est supérieur à μ (nombre de parents), un tri par distance d'isolement est appliqué pour ne garder que les individus les plus isolés. Les μ individus restants sont implantés à la place des précédents parents sur chaque île.

5.3 Expérience

Pour évaluer l'adaptation de CGP-IP-GI en multiobjectifs, nous allons l'étudier sur un ensemble de données utilisées au cours du chapitre précédent.

5.3.1 Paramètres CGP-IP

Nous utilisons les paramètres pour CGP-IP-GI suivants:

- R: le nombre de lignes dans CGP est 1
- C: le nombre de colonnes dans CGP est de 50 pour toutes les expérimentations mais peut évoluer avec les opérateurs d'insertion et de suppression
- r_{mut}: le taux de mutation pour chaque gène est de 0.25
- r_{ins}: le taux de l'opérateur d'insertion pour chaque gène est de 0.1
- r_{del}: le taux de l'opérateur de suppression pour chaque gène est de 0.1
- Le nombre d'îles est de 4
- μ: le nombre de parents sur chaque île est de 4
- λ: la taille de la population sur chaque île est de 8
- Intervalle de synchronisation entre chaque île : le nombre de générations avant que les îles ne comparent leur précision pour se mettre à jour avec le meilleur chromosome est de 100
- Le nombre de générations est 5000

Chaque nœud du graphe est encodé avec 8 paramètres (voir la Table 5.1). L'allèle fonction représente un index dans la liste des fonctions de traitement de l'image. Le deuxième allèle, Connexion 0, est la connexion avec un précédent nœud où la sortie est utilisée pour l'entrée de la fonction. Le troisième allèle, Connexion 1, est la connexion avec un précédent nœud où la sortie est utilisée pour l'entrée de la fonction (toutes les fonctions n'utilisent pas Connexion 1). Les quatrième, cinquième et sixième allèles, Paramètre 0, 1 et 2, sont des nombres réels qui sont les premier, second et troisième paramètres de la fonction. Ces allèles ne sont pas forcément utilisés car toutes les fonctions n'ont pas trois paramètres. Par exemple, les paramètres Gabor Filter sont seulement utilisés avec les fonctions Gabor. Durant le processus d'évolution, les mutations peuvent se produire sur l'index de la fonction, sur une connexion ou sur un des paramètres.

Paramètre	**Type**	**Intervalle**
Fonction	INT	nombre de fonctions
Connexion 0	INT	nombre de nœuds/entrées
Connexion 1	INT	nombre de nœuds/entrées
Paramètre 0	REAL	$[-\infty, \infty]$
Paramètre 1	INT	[-16, 16]
Paramètre 2	INT	[-16, 16]
Gabor Filter Freq.	INT	[0, 16]
Gabor Filter Orien.	INT	[-8, 8]

Table 5.1: Paramètres d'un nœud

5.3.2 Ensemble de données : Trafic urbain

Dans cette expérience, nous appliquons CGP-IP pour identifier les objets en mouvement dans un environnement urbain. L'objectif de l'ensemble de données est de construire un filtre qui extrait et suit des objets spécifiques dans la vidéo. Pour cela, le filtre doit trouver les objets qui se sont déplacés d'une image à la suivante. L'ensemble de données a été créé avec des vidéos du trafic urbain [2]. Nous utilisons des vidéos d'une durée de 5 minutes en couleur RGB 16bit et avec une résolution de 1024x576 pixels. Les images sont converties en nuances de gris pour les entrées (Figure 4.9.A et Figure 4.9.B). L'image en sortie (Figure 4.9.C) a été généré avec Mask-RCNN [74] et identifie les larges objets tels que les piétons, les vélos, les voitures, etc.

Le filtre de départ utilisé dans cet ensemble de données a été créé par des ingénieurs. Il fonctionne en soustrayant les deux images en entrée (Figure 4.9.A et Figure 4.9.B) et en appliquant des fonctions d'érosion et de dilatation pour réduire le bruit dans l'image. Ce filtre est détaillé dans le Listing 4.2 et affiché dans la Figure 4.8.

5.3.3 Fonctions des objectifs

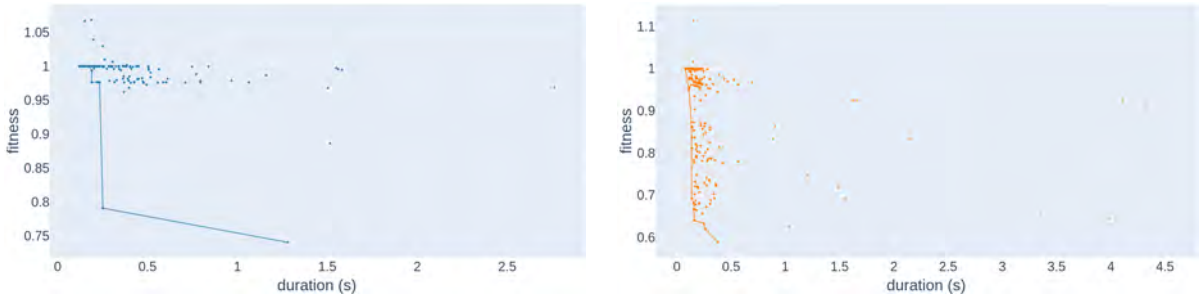
Deux objectifs vont être optimisés en parallèle. Le premier objectif est la précision du résultat. La fonction de l'objectif précision est : 1 - MCC. Son calcul est présenté dans la section 4.2.2, un résultat de la fonction objectif plus petit indique une meilleure précision, l'objectif est donc de le minimiser. Le deuxième objectif est le temps d'exécution du calcul. La fonction objectif pour le temps de calcul est mesurer avec la fonction Python pro-

[2] https://camstreamer.com/live/streams/14-traffic

cess_time [3] qui retourne le temps de calcul nécessaire pour exécuter la fonction d'évaluation de la précision. Un résultat de la fonction objectif pour le temps de calcul plus petit indique un temps de calcul réduit, l'objectif est donc de le minimiser.

5.4 Résultats

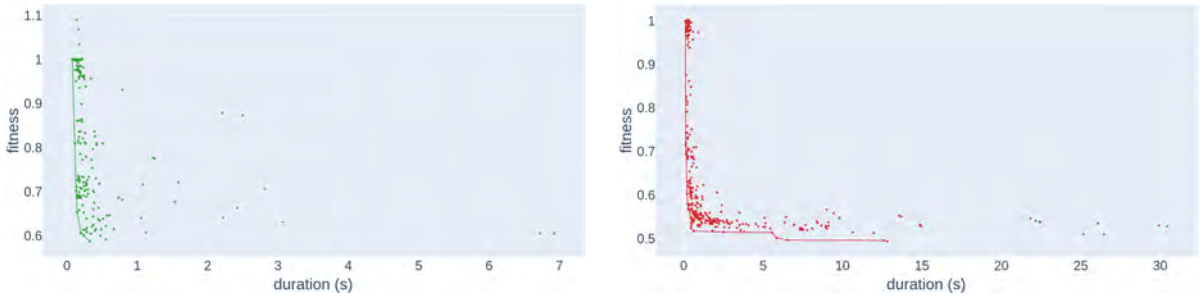
Dans cette section, nous présentons les résultats obtenus sur notre ensemble de données lors de 5000 itérations sur 40 séries. L'utilisation d'un algorithme multiobjectifs a permis d'obtenir un ensemble de solutions performantes autant pour un objectif que pour deux.



A : Itération 1 B : Itérations 50

Figure 5.2: Front Pareto pour l'itération 1 et 50 sur 40 séries

La Figure 5.2 montre le front Pareto lors de l'itération 1 en bleu et de l'itération 50 en orange sur 40 séries. On peut noter une nette amélioration de la précision lors de l'évolution des itérations alors que la durée d'exécution a déjà atteint un plancher.



A : Itérations 100 B : Itérations 1000

Figure 5.3: Front Pareto pour l'itération 100 et 1000 sur 40 séries

La Figure 5.3 montre le front Pareto lors de l'itération 100 en vert et de l'itération 1000 en rouge sur 40 séries. On peut noter une amélioration de la précision entre l'itération 100 et l'itération 1000.

[3] https://docs.python.org/3/library/time.html

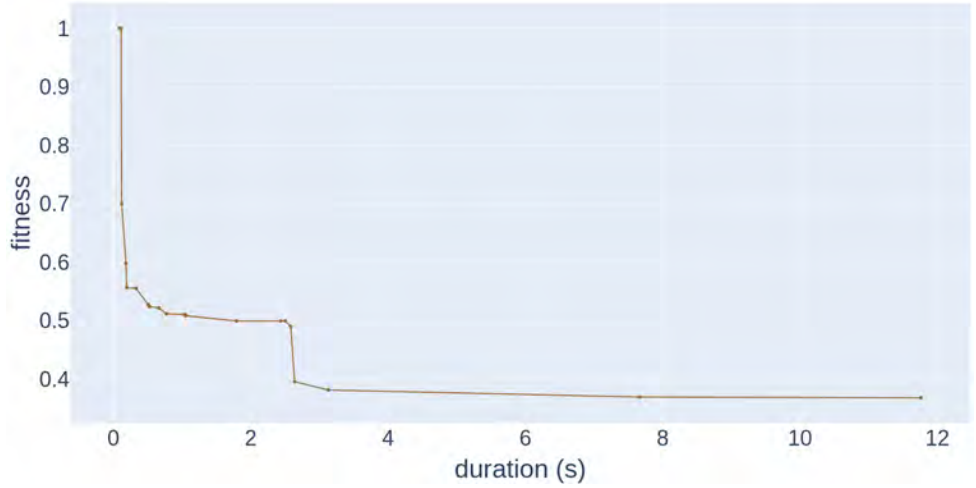
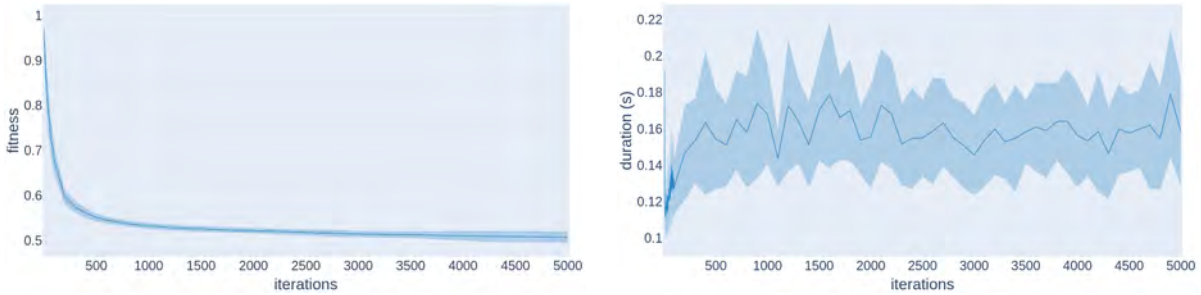


Figure 5.4: Front Pareto à l'itération 5000 sur 40 séries

La Figure 5.4 représente les solutions du front Pareto lors de l'itération 5000. Le front présenté couvre un large spectre de solutions en fonction des objectifs. La Table 5.2 présente les valeurs les plus représentatives du front Pareto.

Fitness	1.0	0.70	0.59	0.55	0.52	0.50	0.49	0.49	0.39	0.38	0.36	0.35
Duration	0.09	0.12	0.18	0.33	0.66	1.05	1.79	2.58	2.63	3.13	7.65	11.7

Table 5.2: Valeurs significatives du Front Pareto à l'itération 5000 sur 40 séries



A : Meilleure précision B : Durée la plus réduite

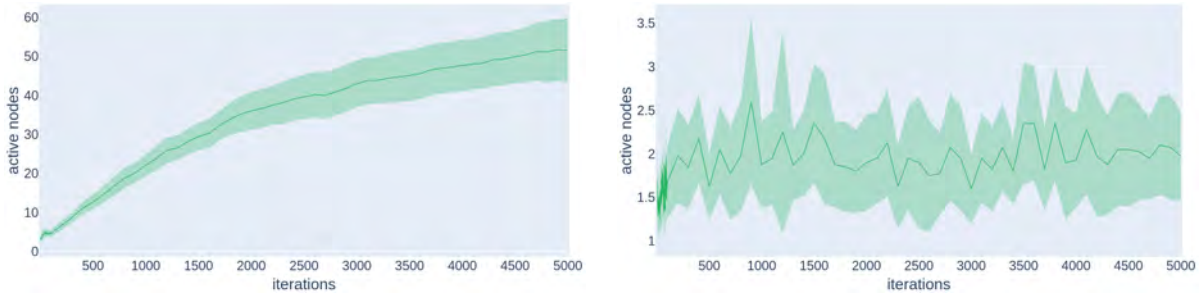
Figure 5.5: Moyenne et écart-type de la meilleure précision et de la durée la plus réduite sur 40 séries

La Figure 5.5.A présente l'évolution de la moyenne et de l'écart-type de la meilleure précision de chaque série sur 40 séries tout au long des 5000 itérations. On constate une nette amélioration de la précision jusqu'à l'itération 500, suivi ensuite d'une amélioration lente mais continue de la précision avec un écart-type qui augmente à partir de l'itération 3500 jusqu'à doubler à l'itération 5000.

La Figure 5.5.B présente l'évolution de la moyenne et de l'écart-type de la durée la plus réduite de chaque série sur 40 séries tout au long des 5000 itérations. On constate

une augmentation de la moyenne et de l'écart-type jusqu'à l'itération 250 pour ensuite se maintenir dans les mêmes niveaux jusqu'à l'itération 5000.

La comparaison de la Figure 5.5.A et la Figure 5.5.B révèle l'intrication de la précision avec la durée qui se stabilisent après l'itération 300.

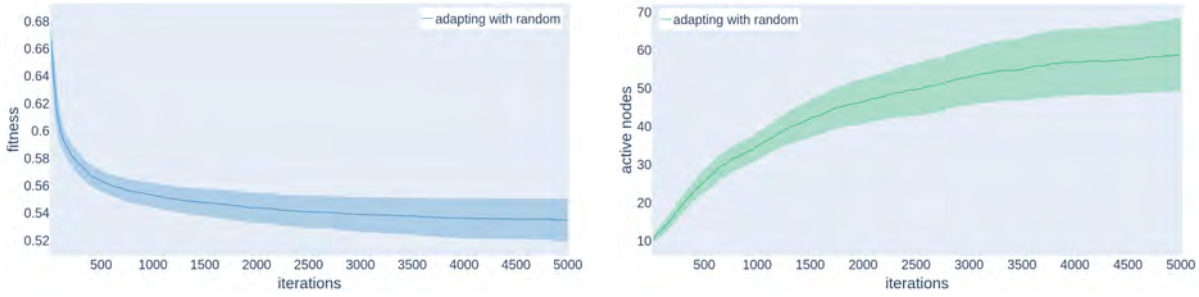


A : Meilleure précision B : Durée la plus réduite

Figure 5.6: Moyenne et écart-type du nombre de nœuds actifs pour la meilleure précision et la durée la plus réduite sur 40 séries

La Figure 5.6.A présente l'évolution de la moyenne et de l'écart-type du nombre de nœuds actifs pour la meilleure précision de chaque série sur 40 séries tout au long des 5000 itérations. On constate une augmentation constante de la moyenne et de l'écart-type du nombre de nœuds actifs tout au long des 5000 itérations.

La Figure 5.6.B présente l'évolution de la moyenne et de l'écart-type du nombre de nœuds actifs pour la durée la plus réduite de chaque série sur 40 séries tout au long de 5000 itérations. On constate une augmentation de la moyenne et de l'écart-type jusqu'à l'itération 200 pour ensuite se maintenir dans les mêmes niveaux jusqu'à l'itération 5000. La comparaison de la Figure 5.5 et la Figure 5.6 met en évidence l'amélioration des objectifs qui n'est possible qu'avec une augmentation du nombre de noeuds actifs.



A : Précision B : Nœuds actifs

Figure 5.7: Moyenne et écart-type de la précision et du nombre de nœuds actifs sur 40 pour [11]

La figure 5.7 provient d'une publication précédente [11]. La figure 5.7.A présente

l'évolution de la moyenne et de l'écart type de la précision sur 40 séries sur 5000 itérations. En le comparant à la Figure 5.5.A, ils ont en commun une amélioration constante et rapide jusqu'à l'itération 500 suivie d'une amélioration lente. Les amplitudes des écarts-types des deux figures sont similaires. L'adaptation multi-objectifs obtient à 5000 itérations une précision un peu meilleure (2%). La figure 5.7.B montre l'évolution de la moyenne et de l'écart type du nombre de nœuds actifs sur 40 séries sur 5000 itérations. En la comparant à la Figure 5.6.A, ils partagent la même évolution avec une augmentation constante du nombre de nœuds actifs et un écart-type similaire. À l'itération 5000, Figure 5.7. Une valeur moyenne est de 0,53, sa valeur minimale est de 0,44, sa valeur maximale est de 0,59 et son écart type est de 0,03. Même si la Figure 5.7.A et la Figure 5.5.A montrent que la précision moyenne est légèrement différente à 5000 itérations, la comparaison de leur meilleure précision montre que l'adaptation multi-objectif permet d'obtenir des individus avec une meilleure précision. La valeur p-value du Ttest entre la figure 5.5.A et la figure 5.7.A est inférieure à $1e^{-5}$.

Les résultats de l'expérience sur 40 séries lors de 5000 itérations présentent une amélioration régulière de la précision ainsi qu'une augmentation régulière du nombre de nœuds actifs. Le front Pareto obtenu après 5000 itérations propose un grand spectre de solutions avec une précision comprise entre 1.0 et 0.36 pour une durée qui varie entre 90 ms et 11.7 secondes. Il ainsi possible de choisir une solution en fonction de ces deux objectifs. Par exemple, même si la meilleure précision obtenue est de 0.36, il existe une solution proposant une précision moindre à 0.39 (soit 10% de précision en moins) mais qui présente une durée de 2.63 secondes au lieu de 11.7 secondes (soit une réduction de 78% du coût de calcul). De plus, la comparaison avec les résultats de travaux antérieurs a montré que l'utilisation d'un algorithme multi-objectifs ne réduisait pas la précision pendant 5000 itérations. Au contraire, garder un front de Pareto au lieu d'un individu unique au cours du processus d'évolution permet d'atteindre plus d'espace de recherche et conduit à trouver de meilleures solutions.

5.5 Conclusion préliminaire

Dans ce chapitre, nous avons adapté CGP-IP-GI pour pouvoir l'utiliser avec plusieurs objectifs. L'algorithme NSGA2 a été implanté et permet d'obtenir un ensemble de solutions

et ainsi de pouvoir en choisir une, en fonction des contraintes imposées.

Les résultats obtenus à partir de l'expérience avec cette adaptation multi-objectifs permettent de proposer un ensemble de solutions appartenant à un front de Pareto dans lequel une réduction minimale de la précision permet un gain considérable en temps de calcul nécessaire. De plus, garder un front de Pareto au lieu d'un individu unique pendant le processus d'évolution se traduit par un espace de recherche plus large, conduisant ainsi à des solutions efficaces.

Cette étude et sa comparaison avec les publications précédentes confirment qu'il est possible de réduire le temps de calcul nécessaire aux filtres d'image tout en conservant la précision, libérant ainsi de la puissance de calcul pour d'autres tâches. Cela peut être particulièrement utile pour des applications embarquées basées sur GP. De plus, les résultats confirment que l'algorithme NSGA-II est très efficace lorsqu'il est adapté dans les algorithmes génétiques pour l'amélioration génétique.

En pratique, cette évolution du cadre CGP-IP-GI permet de réaliser de l'amélioration génétique multi-objectifs en offrant une large gamme de solutions efficaces pour plusieurs objectifs. Le choix de la solution à retenir doit être fait par un humain qui peut sélectionner la solution la plus appropriée en fonction des différentes contraintes imposées.

Nous avons constaté que l'utilisation d'algorithmes multi-objectifs augmentait l'espace de recherche, conduisant ainsi à de meilleures solutions. Cette constatation est alignée avec de multiples travaux réalisés depuis plusieurs années sur la recherche divergente [39], la recherche de nouveauté et les algorithmes de qualité-diversité. Même si l'utilisation d'algorithmes multi-objectifs est bénéfique, elle ne réduit pas ni n'annule le problème du gonflement introduit avec les nouveaux opérateurs de mutations du chapitre précédent. Basée sur les travaux du chapitre précédent, une piste intéressante à explorer est l'utilisation d'algorithmes multi-objectifs pour contrôler l'augmentation du nombre de nœuds actifs et ainsi limiter l'effet non désirable de gonflement. Dans ce chapitre, le deuxième objectif choisi était de réduire le temps de calcul nécessaire. Il serait ainsi intéressant de contrôler l'évolution du nombre de nœuds actifs en essayant de les réduire dans le but de limiter l'effet de gonflement ou au contraire de les maximiser.

Chapter 6

Conclusions

Dans un contexte industriel tendu où l'innovation tient une place importante, il est important de connaître les limites de la technologie et les évolutions à suivre. Dans cette optique, la société Kawantech souhaite étudier l'utilisation de l'intelligence artificielle sur des cartes électroniques embarquées pour faire évoluer un produit fonctionnant avec des algorithmes *classiques*.

Une étude complète de l'état de l'art sur la détection d'objets dans les images ainsi que dans les vidéos est présentée dans le Chapitre 2. Le chapitre est complété par l'état de l'art de la catégorisation et la prédiction des séries temporelles.

Dans cette recherche, nous avons tout d'abord souhaité lever un premier verrou : comment utiliser des algorithmes d'intelligence artificielle sur une carte électronique embarquée avec une puissance de calcul limitée ?

Pour cela, nous avons développé dans le Chapitre 3 une application en deux phases. La première phase est une phase d'apprentissage qui utilise les réseaux de neurones profonds pour extraire des objets en mouvement, leurs classes et des caractéristiques propres. Un réseau de neurones récurrents est ensuite entrainé pour pouvoir identifier la classe d'un objet en mouvement à partir de ses caractéristiques. La deuxième phase est une phase de catégorisation. Le modèle de réseaux de neurones récurrents entrainé dans la première phase est utilisé pour catégoriser correctement les objets en mouvement. Cette application, en combinant la précision des réseaux de neurones profonds et récurrents à une réduction de la taille des données utilisées en entrée dans ces réseaux, permet de catégoriser en temps réel des objets en mouvement sur une carte électronique embarquée à faible puissance de calcul, répondant ainsi favorablement à la demande initiale de Kawantech.

L'utilisation de l'intelligence artificielle sur des processeurs embarqués à faible puissance de calcul n'est pas un sujet d'étude récent. Il existe de nombreuses recherches sur le sujet comme celles menées par Pearson et al. qui ont proposé en 2005 une implémentation sur un *Field Programmable Gate Arrays* FPGA des réseaux de neurones à impulsions [137] mais aussi par Farabet et al. qui ont proposé en 2009 une implémentation de CNN sur un FPGA pour du traitement en temps réel [45]. Ces travaux utilisent des processeurs FPGA pour faire fonctionner des réseaux de neurones profonds mais qui pourraient être utilisés pour d'autres tâches. Avec l'explosion récente des réseaux de neurones profonds, des processeurs spécifiques *Neural Processing Unit* NPU ont été développés pour les réseaux de neurones profonds et sont utilisés sur des cartes embarquées comme processeurs secondaires spécifiques à l'intelligence artificielle. Plusieurs travaux récents portent sur l'utilisation de NPU, comme ceux proposés par Chen et al. en 2021 sur la détection d'objets en temps réel [20] ou bien encore une étude sur l'efficacité des réseaux de neurones profonds sur des cartes embarquéss, proposée par Donggyu et al. en 2021 [23]. D'autres travaux ont porté sur le développement d'une architecture logicielle spécifique de réseaux de neurones profonds avec une réduction du nombre de paramètres comme proposé par Javad Shafiee et al. en 2017 [156] mais aussi Womg et al. en 2018 [180]. Notre approche a consisté à utiliser un réseau de neurones profonds précis mais très consommateur en puissance de calcul pour générer depuis un flux vidéo des métas données compactes qui vont permettre d'entrainer un réseau de neurones profonds sur des séries temporelles consommant donc peu de puissance de calcul. Il est ainsi possible, sur une carte embarquée à faible puissance de calcul, de faire tourner un algorithme d'intelligence artificielle pour de la détection et de la classification d'objet.

Nous avons ensuite tenté de lever le verrou suivant : comment améliorer un filtre de traitement d'images réalisé par des humains experts dans le domaine, en utilisant un algorithme d'intelligence artificielle ?

L'amélioration génétique le permet. Une adaptation du *framework* CGP-IP est développée et présentée dans le Chapitre 4. Elle consiste en l'ajout de deux nouveaux opérateurs qui interviennent lors de la phase de mutation et vont permettre d'insérer ou de supprimer un nœud dans le graphe actif sans en modifier la structure. Son utilisation dans le cadre de l'amélioration génétique a permis d'améliorer plus rapidement un filtre de traitement d'images réalisé par des experts du traitement du signal et d'atteindre une meilleure

précision.

Ce chapitre a fait l'objet d'une publication lors de le la conférence ECTA 2021 [1] [11]. CGP proposé par Julian F. Miller et Peter Thomsonen en 1997 [116] a permis de réaliser de multiples avancées en outrepassant régulièrement les performances de l'état de l'art [121]. CGP a été adapté dans différents *framework* tel que CGP-IP qui a, par exemple, permis la détection de rochers martiens sur des images en utilisant des fonctions de traitement de l'image [97]. A notre connaissance, nos travaux sont les premiers sur l'utilisation de CGP en amélioration génétique. Notre évolution CGP-IP-GI permet, en évolution génétique mais aussi en amélioration génétique, d'atteindre des solutions offrant une meilleure précision tout en diminuant le nombre de générations nécessaires. L'ajout de nœuds dans le graphe actif permet d'augmenter l'espace de recherche tout au long des multiples générations permettant ainsi d'atteindre de meilleures solutions plus rapidement. Cette conclusion est alignée avec les précédents travaux sur la croissance progressive des réseaux de neurones [161] ou les réseaux de régulation de gènes [25]. Dans le contexte industriel qui est le nôtre, CGP offre l'avantage d'être interprétable à l'inverse des solutions de réseaux de neurones profonds même si des travaux sont en cours pour mieux expliquer le phénomène *black box* des réseaux de neurones profonds [16].

Pour finir, nous avons levé le verrou suivant : existe-t-il en intelligence artificielle, des algorithmes permettant dans un environnement limité en puissance de calcul, de résoudre des problèmes en ayant plusieurs objectifs ?

Le Chapitre 5 propose une adaptation du travail précédent dans un contexte multiobjectif dont un des objectifs vise à augmenter la précision du programme et un deuxième objectif vise à limiter le temps d'éxécution du programme afin de réduire la consommation énergétique. La stratégie évolutionniste a été modifiée de $\mu = 1 + \lambda$ en $\mu > 1 + \lambda$ afin de pouvoir préserver un front de Pareto. NSGA2 a été implanté et permet de sélectionner les meilleurs individus du front de Pareto à garder pour la nouvelle génération. Il existe déjà des travaux pour implanter NSGA2 avec CGP [40]. L'algorithme propose ainsi plusieurs individus sur le front de Pareto et nécessite une intervention humaine pour choisir l'individu le plus adapté en fonction de ses résultats dans les différents objectifs. Sur notre ensemble de données, notre adaptation a permis de choisir le meilleur compromis entre précision et temps d'exécution du filtre de traitement de l'image, divisant ainsi

[1] https://ecta.scitevents.org/

le temps d'exécution par 4 pour une réduction de la précision de seulement 3%.

Un article est en cours de préparation et sera soumis à une conférence majeure tel que GECCO 2022 [2]. NSGA2 a été proposé en 2000 [32] et a depuis été utilisé et modifié régulièrement depuis plus de 20 ans [169]. Notre adaptation confirme les travaux précédents sur NSGA2 et valide son efficacité lorsqu'il est combiné avec CGP qui est, tout comme NSGA2, un algorithme évolutionniste. Dans un contexte industriel et encore plus sur des solutions embarquées qui souffrent d'un manque de puissance de calcul, l'utilisation du multiobjectif est salvatrice. Elle permet de produire des algorithmes tout en contrôlant la balance précision/temps d'exécution. Il est ainsi possible de contrôler et choisir le temps de puissance de calcul alloué à chaque algorithme permettant ainsi de prioriser certains algorithmes en fonction du niveau de précision attendu.

6.1 Perspectives

Un axe de recherche intéressant est lié à la parution récente d'un ouvrage sur l'utilisation de GP pour la classification des images [10]. Dans le Chapitre 3, il serait intéressant de remplacer la partie réseaux de neurones profonds LSTM par un individu généré par un CGP afin de pouvoir comparer les performances, autant en termes de puissance de calcul nécessaire que de précision obtenue. Un CGP classique serait utilisé et non CGP-IP-GI car les données en entrée sont des métas informations telles que la taille d'un objet, son placement dans l'image, etc... Cependant, il serait aussi possible de travailler directement depuis les images en entrée en utilisant CGP-IP-GI et ainsi économiser l'extraction des métas données. Un travail de mesure des performances en terme de puissance de calcul mais aussi de précision, est là aussi nécessaire afin de migrer vers la meilleure méthode si les résultats sont encourageants. Ces deux nouvelles méthodes nécessiteraient toujours une phase d'apprentissage pour adapter l'algorithme à l'environnement du luminaire. La dernière méthode serait donc de supprimer la phase d'apprentissage et de travailler directement sur des images en entrée, provenant de différents environnements, pour produire, en utilisant CGP-IP-GI, un individu qui permettrait la classification des objets en mouvement. Cette dernière méthode, si elle produisait des résultats comparables aux deux premières serait celle à privilégier, car en supprimant tous les réseaux de

[2] https://gecco-2022.sigevo.org/

neurones profonds, nous pourrions ainsi obtenir une application dite *glass box*.

Avec l'arrivée de la *big data*, les réseaux de neurones profonds ont connu un essor sans commune mesure à la hauteur de leurs performances. C'est aussi le cas dans le domaine de la *smartcity* ou la multiplicité des capteurs engendrent une énorme quantité de données que ce soient des mesures physiques, des vidéos, des bandes sonores, etc... Toutes ces données vont permettre la création de multiples réseaux de neurones profonds qui vont s'attaquer aux différentes problématiques de la *smartcity*. Pour autant, les réseaux de neurones profonds ont plusieurs défauts qui pourraient être corrigés avec l'utilisation d'algorithmes génétiques.

Leur premier défaut est d'être un système dit *black box*. En effet, il est difficile d'analyser les centaines de millions de paramètres qui composent un réseau de neurones profonds. C'est ainsi que des réseaux de neurones profonds peuvent fonctionner parfaitement sur des situations vues pendant leurs entrainements mais être faux sur des cas spécifiques. A l'inverse, les algorithmes génétiques tels que CGP permettent une interprétabilité de l'individu généré permettant une vérification de son fonctionnement avant même sa mise en production. De plus, même si ce n'est pas le cas de toutes les applications, dans certaines branches de la *smartcity* telles que la sécurité, l'interprétabilité des algorithmes est nécessaire pour qu'ils puissent être certifiés avec les normes en vigueur.

Le grand nombre de paramètres qui composent les réseaux de neurones profonds engendre un autre défaut : la puissance de calcul nécessaire pour les utiliser est très élevée aux risques d'avoir une fréquence d'utilisation très limitée. Par exemple, dans les applications de télésurveillance, la puissance de calcul nécessaire pour pouvoir traiter de multiples flux vidéos en temps réel est astronomique. Comme vu précédemment, l'utilisation des réseaux de neurones profonds sur des cartes électroniques embarquées à puissance de calcul limitée s'avère difficile. Soit le réseau de neurones profonds est rapide mais la quantité de données en entrée est limitée, soit il est précis avec une latence élevée. Les algorithmes génétiques permettent ainsi de générer des programmes moins gourmands en puissance de calcul surtout lorsqu'ils utilisent du multiobjectif. Il est alors possible de choisir le juste milieu entre précision des résultats et puissance de calcul nécessaire.

L'utilisation des algorithmes multiobjectifs dans le domaine des systèmes embarqués est un *must have*. Ils permettent un contrôle total sur les temps d'éxécution tout en maintenant un niveau de précision élevé. Leur utilisation sur des systèmes à puissance de

calcul limitée me semble inévitable et bénéfique.

Nous avons pu constater dans le Chapitre 4 et le Chapitre 5 que l'ajout de nouveaux opérateurs de mutation pour CGP avait réintroduit un effet indésirable de gonflement. Le gonflement des individus est un effet indésirable car il entraine un sur-apprentissage de l'ensemble de données d'entrainement qui se reflètera par une précision réduite sur des nouvelles données. De plus, le gonflement entraine la complexification de l'individu qui devient donc plus difficile à interpréter. Une piste intéressante suggérée dans le Chapitre 5 serait d'utiliser l'adaptation multi-objectifs de CGP pour contrôler le gonflement induit par l'ajout de nouveaux opérateurs du Chapitre 4 avec un objectif de contrôle du nombre de nœuds actifs. Ainsi l'adaptation multi-objectifs NSGA2 permettrait de limiter le gonflement et le sur-apprentissage.

Dans le domaine de la *smartcity* comme dans de nombreux autres, les algorithmes d'intelligence artificielle utilisés ne sont qu'un maillon d'un système complet. L'utilisation d'algorithmes génétiques moins gourmands en puissance de calcul permet le traitement de l'information au plus près de la source, autorisant ainsi la possibilité d'avoir des actions en temps réel, mais aussi réduisant la nécessité de bande passante sur les différents réseaux pour acheminer les données. Les coûts d'infrastructure des réseaux peuvent être revus à la baisse ainsi que la bande passante nécessaire.

Même si les algorithmes génétiques ne rivalisent pas encore avec les réseaux de neurones profonds dans tous les domaines, il existe aujourd'hui de nombreuses pistes à expérimenter pour améliorer leurs résultats et égaliser voire surpasser ceux des réseaux de neurones profonds. L'utilisation des algorithmes génétiques ne peut être que bénéfique dans le futur pour améliorer l'interprétabilité des algorithmes, évitant ainsi les systèmes *black box*. De plus, ils produisent des algorithmes nécessitant moins de puissance de calcul particulièrement adaptés aux cartes électroniques embarquées. L'utilisation du multiobjectif permet un contrôle précis de la répartition puissance de calcul/précision des résultats autorisant des actions en temps réel.

Bibliography

[1] Nayyer Aafaq et al. “Video description: A survey of methods, datasets, and evaluation metrics”. In: *ACM Computing Surveys* 52.6 (2019), pp. 1–28. ISSN: 15577341. DOI: 10.1145/3355390. arXiv: 1806.00186.

[2] Yousri Abdeljaoued, Touradj Ebrahimi, and C. Christopoulos. “A new algorithm for shot boundary detection”. In: *European Signal Processing Conference* 2015 (Mar. 2015).

[3] Abiel Aguilar-González, Miguel Arias-Estrada, and François Berry. “Robust feature extraction algorithm suitable for real-time embedded applications”. In: *Journal of Real-Time Image Processing* 14 (Mar. 2018). DOI: 10.1007/s11554-017-0701-8.

[4] Mohammad H. Ahmadi et al. “Application of the multi-objective optimization method for designing a powered Stirling heat engine: Design with maximized power, thermal efficiency and minimized pressure loss”. In: *Renewable Energy* 60 (2013), pp. 313–322. ISSN: 0960-1481. DOI: https://doi.org/10.1016/j.renene.2013.05.005. URL: https://www.sciencedirect.com/science/article/pii/S0960148113002504.

[5] A. Arcuri and X. Yao. “A novel co-evolutionary approach to automatic software bug fixin”. In: *CEC, pages 162–168* (2008).

[6] Anurag Arnab et al. “ViViT: A Video Vision Transformer”. In: *2021 IEEE/CVF International Conference on Computer Vision (ICCV)*. 2021, pp. 6816–6826. DOI: 10.1109/ICCV48922.2021.00676.

[7] K. Behrendt, L. Novak, and Rami Botros. “A deep learning approach to traffic lights: Detection, tracking, and classification”. In: *2017 IEEE International Conference on Robotics and Automation (ICRA)* (2017), pp. 1370–1377.

[8] Shweta Bhardwaj, Mukundhan Srinivasan, and Mitesh M. Khapra. *Efficient Video Classification Using Fewer Frames*. 2019. arXiv: 1902.10640 [cs.CV].

[9] Nirav Bhatt. "A survey on video classification techniques". In: *Journal of Emerging Technologies and Innovative Research* 2.3 (2015), pp. 607–610.

[10] Ying Bi, Bing Xue, and Mengjie Zhang. *Genetic Programming for Image Classification*. Springer International Publishing, 2021. DOI: 10.1007/978-3-030-65927-1.

[11] Julien Biau et al. "Improving Image Filters with Cartesian Genetic Programming". In: 2021, pp. 17–27. ISBN: 978-989-758-534-0.

[12] Beatriz Blanco-Filgueira et al. "Deep Learning-Based Multiple Object Visual Tracking on Embedded System for IoT and Mobile Edge Computing Applications". In: *IEEE Internet of Things Journal* 6.3 (2019), pp. 5423–5431. DOI: 10.1109/JIOT.2019.2902141.

[13] Markus Braun et al. "EuroCity Persons: A Novel Benchmark for Person Detection in Traffic Scenes". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* PP (Feb. 2019), pp. 1–1. DOI: 10.1109/TPAMI.2019.2897684.

[14] Darin Brezeale and Diane J. Cook. "Automatic video classification: A survey of the literature". In: *IEEE Transactions on Systems, Man and Cybernetics Part C: Applications and Reviews* 38.3 (2008), pp. 416–430. ISSN: 10946977. DOI: 10.1109/TSMCC.2008.919173.

[15] B. R. Bruce, J. Petke, and M. Harman. "Reducing energy consumption using genetic improvement". In: *GECCO* (2015).

[16] Vanessa Buhrmester, David Münch, and Michael Arens. *Analysis of Explainers of Black Box Deep Neural Networks for Computer Vision: A Survey*. 2019. arXiv: 1911.12116 [cs.AI].

[17] Juan José Burred and Alexander Lerch. "Hierarchical Automatic Audio Signal Classification". In: *Journal of the Audio Engineering Society (JAES)* 52 (July 2004), pp. 724–739.

[18] Joao Carreira and Andrew Zisserman. *Quo Vadis, Action Recognition? A New Model and the Kinetics Dataset*. 2018. arXiv: 1705.07750 [cs.CV].

[19] Zhengping Che et al. “Recurrent Neural Networks for Multivariate Time Series with Missing Values”. In: *Scientific Reports* 8 (Apr. 2018). DOI: 10.1038/s41598-018-24271-9.

[20] Long Chen et al. “Onboard Real-time Object Detection for UAV with Embedded NPU”. In: *2021 IEEE 11th Annual International Conference on CYBER Technology in Automation, Control, and Intelligent Systems (CYBER)*. 2021, pp. 192–197. DOI: 10.1109/CYBER53097.2021.9588193.

[21] Zhao Chen and Darvin Yi. “The Game Imitation: Deep Supervised Convolutional Networks for Quick Video Game AI”. In: *ArXiv* abs/1702.05663 (2017).

[22] Kyunghyun Cho et al. *Learning Phrase Representations using RNN Encoder-Decoder for Statistical Machine Translation*. 2014. arXiv: 1406.1078 [cs.CL].

[23] ChoiDonggyu et al. “A Study on the Efficiency of Deep Learning on Embedded Boards”. In: *The Journal of the Convergence on Culture Technology* 7.1 (Feb. 2021), pp. 668–673.

[24] Adam Coates, Andrew Ng, and Honglak Lee. “An Analysis of Single-Layer Networks in Unsupervised Feature Learning”. In: *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics*. Ed. by Geoffrey Gordon, David Dunson, and Miroslav Dudík. Vol. 15. Proceedings of Machine Learning Research. Fort Lauderdale, FL, USA: PMLR, Apr. 2011, pp. 215–223. URL: http://proceedings.mlr.press/v15/coates11a.html.

[25] Sylvain Cussat-Blanc, Kyle Harrington, and Jordan Pollack. “Gene regulatory network evolution through augmenting topologies”. In: *IEEE Transactions on Evolutionary Computation* 19.6 (2015), pp. 823–837.

[26] Navneet Dalal and Bill Triggs. “Histograms of Oriented Gradients for Human Detection”. In: vol. 1. July 2005, pp. 886–893. DOI: 10.1109/CVPR.2005.177.

[27] Dima Damen et al. “Rescaling Egocentric Vision: Collection Pipeline and Challenges for EPIC-KITCHENS-100”. English. In: *International Journal of Computer Vision (IJCV)* 130.1 (Jan. 2022), pp. 33–55. DOI: 10.1007/s11263-021-01531-2.

[28] Mittal Darji. “A REVIEW ONAUDIO FEATURES BASED EXTRACTION OF SONGS FROM MOVIES”. In: Feb. 2015.

[29] Mittal C Darji and Dipti Mathpal. "A Review of Video Classification Techniques". In: *International Research Journal of Engineering and Technology(IRJET)* 4.6 (2017), pp. 1888–1891. URL: https://irjet.net/archives/V4/i6/IRJET-V4I6593.pdf.

[30] Kalyan Deb. "Multiobjective Optimization Using Evolutionary Algorithms. Wiley, New York". In: Jan. 2001, pp. 1–5.

[31] Kalyanmoy Deb et al. "A fast and elitist multiobjective genetic algorithm: NSGA-II". In: *IEEE Trans. Evol. Comput.* 6 (2002), pp. 182–197.

[32] Kalyanmoy Deb et al. "A Fast Elitist Non-dominated Sorting Genetic Algorithm for Multi-objective Optimization: NSGA-II". In: *Parallel Problem Solving from Nature PPSN VI*. Ed. by Marc Schoenauer et al. Berlin, Heidelberg: Springer Berlin Heidelberg, 2000, pp. 849–858. ISBN: 978-3-540-45356-7.

[33] Tushar Deb Kalyanmoy and Goel. "A Hybrid Multi-objective Evolutionary Approach to Engineering Shape Design". In: *Evolutionary Multi-Criterion Optimization*. Ed. by Zitzler Eckart and Thiele Lothar, Deb Kalyanmoy and Coello Carlos Artemio, and Corne David. Berlin, Heidelberg: Springer Berlin Heidelberg, 2001, pp. 385–399. ISBN: 978-3-540-44719-1.

[34] Tushar Deb Kalyanmoy and Goel. "Controlled Elitist Non-dominated Sorting Genetic Algorithms for Better Convergence". In: *Evolutionary Multi-Criterion Optimization*. Ed. by Zitzler Eckart et al. Berlin, Heidelberg: Springer Berlin Heidelberg, 2001, pp. 67–81. ISBN: 978-3-540-44719-1.

[35] Phung Quoc Dinh, C. Dorai, and S. Venkatesh. "Video genre categorization using audio wavelet coefficients". In: 2002.

[36] Yongtae Do. "Dividing an image blob of two connected people using shape information". In: *2008 International Conference on Wavelet Analysis and Pattern Recognition*. Vol. 1. 2008, pp. 152–157. DOI: 10.1109/ICWAPR.2008.4635767.

[37] Yongtae Do. "Region Based Detection of Occluded People for the Tracking in Video Image Sequences". In: Sept. 2005, pp. 829–836. ISBN: 978-3-540-28969-2. DOI: 10.1007/11556121_102.

[38] Piotr Dollar et al. "Pedestrian detection: A benchmark". In: *2009 IEEE Conference on Computer Vision and Pattern Recognition*. 2009, pp. 304–311. DOI: 10.1109/CVPR.2009.5206631.

[39] Stephane Doncieux and Jean-Baptiste Mouret. "Beyond black-box optimization: a review of selective pressures for evolutionary robotics". In: *Evolutionary Intelligence* 7.2 (2014), pp. 71–93. DOI: 10.1007/s12065-014-0110-x. URL: https://doi.org/10.1007/s12065-014-0110-x.

[40] Antonio Miguel Batista Dourado and Emerson Carlos Pedrino. "Multi-Objective Cartesian Genetic Programming Optimization of Morphological Filters in Navigation Systems for Visually Impaired People". In: *Appl. Soft Comput.* 89.C (Apr. 2020). ISSN: 1568-4946. DOI: 10.1016/j.asoc.2020.106130. URL: https://doi.org/10.1016/j.asoc.2020.106130.

[41] Zitzler E. and Thiele L. "Multiobjective evolutionary algorithms: a comparative case study and the strength Pareto approach". In: *IEEE Transactions on Evolutionary Computation* 3.4 (1999), pp. 257–271. DOI: 10.1109/4235.797969.

[42] M. Everingham et al. "The Pascal Visual Object Classes (VOC) Challenge". In: *International Journal of Computer Vision* 88 (2009), pp. 303–338.

[43] Mark Everingham et al. "The Pascal Visual Object Classes Challenge: A Retrospective". In: *International Journal of Computer Vision* 111.1 (Jan. 2015), pp. 98–136. DOI: 10.1007/s11263-014-0733-5.

[44] Jianping Fan et al. "Semantic video classification and feature subset selection under context and concept uncertainty". In: July 2004, pp. 192–201. ISBN: 1-58113-832-6. DOI: 10.1109/JCDL.2004.1336120.

[45] Clément Farabet, Cyril Poulet, and Yann LeCun. "An FPGA-based stream processor for embedded real-time vision with Convolutional Networks". In: *2009 IEEE 12th International Conference on Computer Vision Workshops, ICCV Workshops*. 2009, pp. 878–885. DOI: 10.1109/ICCVW.2009.5457611.

[46] Pedro Felzenszwalb, David Mcallester, and Deva Ramanan. "A Discriminatively Trained, Multiscale, Deformable Part Model". In: vol. 8: June 2008. DOI: 10.1109/CVPR.2008.4587597.

[47] Pedro F. Felzenszwalb, Ross B. Girshick, and David A. McAllester. "Cascade object detection with deformable part models". In: *2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition* (2010), pp. 2241–2248.

[48] Pedro F. Felzenszwalb et al. "Object Detection with Discriminatively Trained Part-Based Models". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 32.9 (2010), pp. 1627–1645. DOI: 10.1109/TPAMI.2009.167.

[49] Asja Fischer and Christian Igel. "An Introduction to Restricted Boltzmann Machines". In: Jan. 2012, pp. 14–36. ISBN: 978-3-642-33274-6. DOI: 10.1007/978-3-642-33275-3_2.

[50] Carlos M. Fonseca and Peter John Fleming. "Genetic Algorithms for Multiobjective Optimization: FormulationDiscussion and Generalization". In: *ICGA*. 1993.

[51] Y. Freund and R. Schapire. "A Short Introduction to Boosting". In: 1999.

[52] Jennifer Joana Gago et al. *Sequence-to-Sequence Natural Language to Humanoid Robot Sign Language*. July 2019.

[53] John Cristian Borges Gamboa. *Deep Learning for Time-Series Analysis*. 2017. arXiv: 1701.01887 [cs.LG].

[54] Chuang Gan et al. "Recognizing an Action Using Its Name: A Knowledge-Based Approach". In: *International Journal of Computer Vision* 120 (2016), pp. 61–77.

[55] Ross Girshick. "Fast R-CNN". In: *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*. Dec. 2015.

[56] Ross Girshick, Pedro Felzenszwalb, and David McAllester. "Object Detection with Grammar Models". In: *Advances in Neural Information Processing Systems*. Ed. by J. Shawe-Taylor et al. Vol. 24. Curran Associates, Inc., 2011. URL: https://proceedings.neurips.cc/paper/2011/file/6faa8040da20ef399b63a72d0e4ab575-Paper.pdf.

[57] Ross Girshick et al. "Region-Based Convolutional Networks for Accurate Object Detection and Segmentation". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 38.1 (2016), pp. 142–158. DOI: 10.1109/TPAMI.2015.2437384.

[58] Ross Girshick et al. *Rich feature hierarchies for accurate object detection and semantic segmentation*. 2014. arXiv: 1311.2524 [cs.CV].

[59] Madzarov Gj et al. “Comparison of Automatic Shot Boundary Detection Algorithms Based On Color, Edges and Wavelets”. In: Oct. 2008.

[60] Pilar Gómez et al. “Local Maximum Ozone Concentration Prediction Using Soft Computing Methodologies”. In: *Systems Analysis Modelling Simulation* 43 (Aug. 2003), pp. 1011–1031. DOI: 10.1080/0232929031000081244.

[61] Raghav Goyal et al. *The ”something something” video database for learning and evaluating visual common sense*. 2017. DOI: 10.48550/ARXIV.1706.04261. URL: https://arxiv.org/abs/1706.04261.

[62] Alex Graves. *Generating Sequences With Recurrent Neural Networks*. 2014. aarXiv: 1308.0850 (cs.NE).

[63] Alex Graves. *Sequence Transduction with Recurrent Neural Networks*. 2012. arXiv: 1211.3711 [cs.NE].

[64] S. Harding, J. Leitner, and J. Schmidhuber. “Cartesian genetic programming for image processing”. In: *Genetic Programming Theory and Practice X* (2012).

[65] S. harding, J. Miller, and W. Banzhaf. “Self-modifying cartesian genetic programming”. In: *Natural Computing Series* (2011).

[66] Simon Harding, Jurgen Leitner, and Jurgen Schmidhuber. “Genetic Programming Theory and Practice”. In: *Journal of Intelligent and Robotic Systems* (2006).

[67] Simon Harding, Jürgen Leitner, and Jürgen Schmidhuber. “Cartesian Genetic Programming for Image Processing”. In: *Genetic Programming Theory and Practice X* (2013), pp. 31–44. DOI: 10.1007/978-1-4614-6846-2_3. URL: http://link.springer.com/chapter/10.1007/978-1-4614-6846-2%7B%5C_%7D3%7B%5C%%7D5Cnhttp://link.springer.com/10.1007/978-1-4614-6846-2.

[68] Simon Harding et al. “Mt-cgp: Mixed type cartesian genetic programming”. In: *Genetic and Evolutionary Computation Conference* (2012).

[69] M. Harman, Y. Jia, and W. Langdon. “Babel pidgin: Sbse can grow and graft entirely new functionality into a real world system”. In: *SSBSE Challenge, pages 247–252* (2014).

[70] Uwe Hassler. "Palma, W.: Time series analysis: Wiley, 2016, 616 pp., €113.20, ISBN: 978-1-118-63432-5". In: *Statistical Papers* 58 (Nov. 2016). DOI: 10.1007/s00362-016-0858-4.

[71] Alexander Hauptmann et al. "Video Classification and Retrieval with the Informedia Digital Video Library System." In: Jan. 2002.

[72] Kaiming He et al. "Deep residual learning for image recognition". In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2016, pp. 770–778.

[73] Kaiming He et al. *Delving Deep into Rectifiers: Surpassing Human-Level Performance on ImageNet Classification*. 2015. arXiv: 1502.01852 [cs.CV].

[74] Kaiming He et al. *Mask R-CNN*. 2018. arXiv: 1703.06870 [cs.CV].

[75] Kaiming He et al. "Spatial Pyramid Pooling in Deep Convolutional Networks for Visual Recognition". In: *Lecture Notes in Computer Science* (2014), pp. 346–361. ISSN: 1611-3349. DOI: 10.1007/978-3-319-10578-9_23. URL: http://dx.doi.org/10.1007/978-3-319-10578-9_23.

[76] G. E. Hinton and R. R. Salakhutdinov. "Reducing the Dimensionality of Data with Neural Networks". In: *Science* 313.5786 (2006), pp. 504–507. ISSN: 0036-8075. DOI: 10.1126/science.1127647. eprint: https://science.sciencemag.org/content/313/5786/504.full.pdf. URL: https://science.sciencemag.org/content/313/5786/504.

[77] Geoffrey Hinton, Simon Osindero, and Yee-Whye Teh. "A Fast Learning Algorithm for Deep Belief Nets". In: *Neural computation* 18 (Aug. 2006), pp. 1527–54. DOI: 10.1162/neco.2006.18.7.1527.

[78] Geoffrey E. Hinton. "Training Products of Experts by Minimizing Contrastive Divergence". In: *Neural Comput.* 14.8 (Aug. 2002), pp. 1771–1800. ISSN: 0899-7667. DOI: 10.1162/089976602760128018. URL: https://doi.org/10.1162/089976602760128018.

[79] Geoffrey E. Hinton and R. Salakhutdinov. "Reducing the Dimensionality of Data with Neural Networks". In: *Science* 313 (2006), pp. 504–507.

[80] Sepp Hochreiter and Jürgen Schmidhuber. "Long Short-term Memory". In: *Neural computation* 9 (Dec. 1997), pp. 1735–80. DOI: 10.1162/neco.1997.9.8.1735.

[81] Elisabeth Hoppe et al. "Deep Learning for Magnetic Resonance Fingerprinting: A New Approach for Predicting Quantitative Parameter Values from Time Series". In: *Studies in health technology and informatics* 243 (Jan. 2017), pp. 202–206.

[82] Sabir Hossain and Deok-jin Lee. "Deep Learning-Based Real-Time Multiple-Object Detection and Tracking from Aerial Imagery via a Flying Robot with GPU-Based Embedded Devices". In: *Sensors* 19.15 (2019). ISSN: 1424-8220. DOI: 10.3390/s19153371. URL: https://www.mdpi.com/1424-8220/19/15/3371.

[83] R.S. Jadon, Santanu Chaudhury, and K. K. Biswas. *Generic Video Classification: An Evolutionary Learning based Fuzzy Theoretic Approach*.

[84] Vidit Jain and Erik Learned-Miller. *FDDB: A Benchmark for Face Detection in Unconstrained Settings*. 2010.

[85] Shuiwang Ji et al. "3D Convolutional Neural Networks for Human Action Recognition". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 35.1 (2013), pp. 221–231. DOI: 10.1109/TPAMI.2012.59.

[86] Roman Kalkreuth, Gunter Rudolph, and Jorg Krone. "More efficient evolution of small genetic programs in Cartesian Genetic Programming by using genotypie age". In: *2016 IEEE Congress on Evolutionary Computation (CEC)*. IEEE, July 2016, pp. 5052–5059. ISBN: 978-1-5090-0623-6. DOI: 10.1109/CEC.2016.7748330. URL: http://ieeexplore.ieee.org/document/7748330/.

[87] Dimosthenis Karatzas et al. "ICDAR 2015 competition on Robust Reading". English. In: *13th International Conference on Document Analysis and Recognition, ICDAR 2015, Nancy, France, August 23-26, 2015*. 2015, pp. 1156–1160. DOI: 10.1109/ICDAR.2015.7333942.

[88] Will Kay et al. *The Kinetics Human Action Video Dataset*. 2017. DOI: 10.48550/ARXIV.1705.06950. URL: https://arxiv.org/abs/1705.06950.

[89] Gul Muhammad Khan, Julian F Miller, and David M Halliday. "Evolution of cartesian genetic programs for development of learning neural architecture." In: *Evolutionary computation* 19.3 (2011), pp. 469–523. ISSN: 1063-6560. DOI: 10.1162/EVCO_a_00043.

[90] V. Kobla, D. DeMenthon, and D. Doermann. "Identifying sports videos using replay, text, and camera motion features". In: *Electronic Imaging*. 1999.

[91] Alex Krizhevsky, Ilya Sutskever, and Geoffrey Hinton. "ImageNet Classification with Deep Convolutional Neural Networks". In: *Neural Information Processing Systems* 25 (Jan. 2012). DOI: 10.1145/3065386.

[92] Alina Kuznetsova et al. "The Open Images Dataset V4". In: *International Journal of Computer Vision* 128.7 (Mar. 2020), pp. 1956–1981. ISSN: 1573-1405. DOI: 10.1007/s11263-020-01316-z. URL: http://dx.doi.org/10.1007/s11263-020-01316-z.

[93] W. Langdon and M. Harman. "Evolving a cuda kernel from an nvidia template". In: *CEC, pages 1–8* (2010).

[94] W. B. Langdon and M. Harman. "Optimising existing software with genetic programming". In: *TEC, 19(1):118–135* (2015).

[95] Hugo Larochelle and Yoshua Bengio. "Classification Using Discriminative Restricted Boltzmann Machines". In: *Proceedings of the 25th International Conference on Machine Learning*. ICML '08. Helsinki, Finland: Association for Computing Machinery, 2008, pp. 536–543. ISBN: 9781605582054. DOI: 10.1145/1390156.1390224. URL: https://doi.org/10.1145/1390156.1390224.

[96] Joonseok Lee et al. "The 2nd YouTube-8M Large-Scale Video Understanding Challenge: Munich, Germany, September 8-14, 2018, Proceedings, Part IV". In: Jan. 2019, pp. 193–205. ISBN: 978-3-030-11017-8. DOI: 10.1007/978-3-030-11018-5_18.

[97] Jürgen Leitner et al. "Mars Terrain Image Classification using Cartesian Genetic Programming". In: *11th International Symposium on Artificial Intelligence, Robotics and Automation in Space (i-SAIRAS)* (2012).

[98] Tsung-Yi Lin et al. *Feature Pyramid Networks for Object Detection*. 2017. arXiv: 1612.03144 [cs.CV].

[99] Tsung-Yi Lin et al. "Focal Loss for Dense Object Detection". In: *2017 IEEE International Conference on Computer Vision (ICCV)*. 2017, pp. 2999–3007. DOI: 10.1109/ICCV.2017.324.

[100] Tsung-Yi Lin et al. “Microsoft COCO: Common Objects in Context”. In: *Computer Vision – ECCV 2014*. Ed. by David Fleet et al. Cham: Springer International Publishing, 2014, pp. 740–755. ISBN: 978-3-319-10602-1.

[101] Wei Liu et al. “SSD: Single Shot MultiBox Detector”. In: *Lecture Notes in Computer Science* (2016), pp. 21–37. ISSN: 1611-3349. DOI: 10.1007/978-3-319-46448-0_2. URL: http://dx.doi.org/10.1007/978-3-319-46448-0_2.

[102] Xuefeng Liu et al. “Hyperspectral Image Classification Based on Parameter-Optimized 3D-CNNs Combined with Transfer Learning and Virtual Samples”. In: *Remote Sensing* 10 (Sept. 2018), p. 1425. DOI: 10.3390/rs10091425.

[103] Z. Liu, J. Huang, and Y. Wang. “Classification TV programs based on audio information using hidden Markov model”. In: *1998 IEEE Second Workshop on Multimedia Signal Processing (Cat. No.98EX175)*. 1998, pp. 27–32. DOI: 10.1109/MMSP.1998.738908.

[104] Zhu Liu, Yao Wang, and Tsuhan Chen. “Audio Feature Extraction And Analysis For Scene Segmentation And Classification”. In: *Journal of VLSI Signal Processing* 20 (Apr. 1998). DOI: 10.1023/A:1008066223044.

[105] Jonathan Long, Evan Shelhamer, and Trevor Darrell. *Fully Convolutional Networks for Semantic Segmentation*. 2015. arXiv: 1411.4038 [cs.CV].

[106] D. Lowe. “Object recognition from local scale-invariant features”. In: *Proceedings of the Seventh IEEE International Conference on Computer Vision* 2 (1999), 1150–1157 vol.2.

[107] David G. Lowe. “Distinctive Image Features from Scale-Invariant Keypoints”. In: *Int. J. Comput. Vision* 60.2 (Nov. 2004), pp. 91–110. ISSN: 0920-5691. DOI: 10.1023/B:VISI.0000029664.99615.94. URL: https://doi.org/10.1023/B:VISI.0000029664.99615.94.

[108] S.M. Lucas et al. “ICDAR 2003 robust reading competitions”. In: *Seventh International Conference on Document Analysis and Recognition, 2003. Proceedings.* 2003, pp. 682–687. DOI: 10.1109/ICDAR.2003.1227749.

[109] Amal Mahmud and Ammar Mohammed. “A Survey on Deep Learning for Time-Series Forecasting”. In: Jan. 2021, pp. 365–392. ISBN: 978-3-030-59337-7. DOI: 10.1007/978-3-030-59338-4_19.

[110] Spyros Makridakis. "A Survey of Time Series". In: *International Statistical Review / Revue Internationale de Statistique* 44.1 (1976), pp. 29–70. ISSN: 03067734, 17515823. URL: http://www.jstor.org/stable/1402964.

[111] Tomasz Malisiewicz, Abhinav Gupta, and Alexei A. Efros. "Ensemble of exemplar-SVMs for object detection and beyond". In: *2011 International Conference on Computer Vision*. 2011, pp. 89–96. DOI: 10.1109/ICCV.2011.6126229.

[112] Pierre-Etienne Martin et al. "3D attention mechanism for fine-grained classification of table tennis strokes using a Twin Spatio-Temporal Convolutional Neural Networks". In: *25th International Conference on Pattern Recognition (ICPR2020)*. Milano, Italy, Jan. 2021. URL: https://hal.archives-ouvertes.fr/hal-02977646.

[113] Pierre-Etienne Martin et al. "Fine grained sport action recognition with Twin spatio-temporal convolutional neural networks". In: *Multimedia Tools and Applications* (Apr. 2020). DOI: 10.1007/s11042-020-08917-3. URL: https://hal.archives-ouvertes.fr/hal-02551019.

[114] Pierre-Etienne Martin et al. "Three-Stream 3D/1D CNN for Fine-Grained Action Classification and Segmentation in Table Tennis". In: *MMSports '21, October 20, 2021, Virtual Event*. Chengdu, China, Oct. 2021. DOI: 10.1145/3475722.3482793. URL: https://hal.archives-ouvertes.fr/hal-03353945.

[115] B. W. Matthews. "Comparison of the predicted and observed secondary structure of t4 phage lysozyme". In: *Biochimica et Biophysica Acta, 405(2):442–451* (1975).

[116] Julian Miller et al. "Designing Electronic Circuits Using Evolutionary Algorithms. Arithmetic Circuits: A Case Study". In: *Genetic Algorithms and Evolution Strategies in Engineering and Computer Science* (Oct. 1999).

[117] Julian F Miller. "What Bloat? Cartesian Genetic Programming on Boolean Problems". In: *2001 Genetic and Evolutionary Computation Conference Late Breaking Papers* (2001), pp. 295–302. URL: http://www.elec.york.ac.uk/intsys/users/jfm7/gecco2001Late.pdf.

[118] Julian F. Miller. "An empirical study of the efficiency of learning boolean functions using a cartesian ge- netic programming approach". In: *Proceedings of the Genetic and Evolutionary Computation Conference, volume 2, pages 1135–1142* (1999).

[119] Julian F. Miller and Peter Thomson. "Cartesian Genetic Programming". In: *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*. Vol. 1802. Springer, 2000, pp. 121–132. ISBN: 3540673393. DOI: 10.1007/978-3-540-46239-2_9. arXiv: arXiv:1011.1669v3. URL: http://link.springer.com/10.1007/978-3-540-46239-2%7B%5C_%7D9.

[120] Julian Francis Miller. *Cartesian genetic programming*. Springer, 2011.

[121] Julian Francis Miller. "Cartesian genetic programming: its status and future". In: *Genetic Programming and Evolvable Machines* (2019), pp. 1–40.

[122] Julian Francis Miller. "What bloat? Cartesian Genetic Programming on Boolean problems". In: 2003.

[123] Andreas Mogelmose, Mohan M. Trivedi, and Thomas B. Moeslund. "Vision based Traffic Sign Detection and Analysis for Intelligent Driver Assistance Systems: Perspectives and Survey". English. In: *I E E E Transactions on Intelligent Transportation Systems* 13.4 (Dec. 2012), pp. 1484–1497. ISSN: 1524-9050. DOI: 10.1109/TITS.2012.2209421.

[124] A. Mohan, Papageorgiou CP, and Tomaso Poggio. "Example-Based Object Detection in Images by Components". In: *Pattern Analysis and Machine Intelligence, IEEE Transactions on* 23 (May 2001), pp. 349–361. DOI: 10.1109/34.917571.

[125] S. Moncrieff, S. Venkatesh, and C. Dorai. "Horror film genre typing and scene labeling via audio analysis". In: *2003 International Conference on Multimedia and Expo. ICME '03. Proceedings (Cat. No.03TH8698)*. Vol. 2. 2003, pp. II–193. DOI: 10.1109/ICME.2003.1221586.

[126] Mathew Monfort et al. "Moments in Time Dataset: one million videos for event understanding". In: *CoRR* abs/1801.03150 (2018). arXiv: 1801.03150. URL: http://arxiv.org/abs/1801.03150.

[127] J. Nam, M. Alghoniemy, and A.H. Tewfik. "Audio-visual content-based violent scene characterization". In: *Proceedings 1998 International Conference on Image Processing. ICIP98 (Cat. No.98CB36269)*. Vol. 1. 1998, 353–357 vol.1. DOI: 10.1109/ICIP.1998.723496.

[128] Joe Yue-Hei Ng et al. *Beyond Short Snippets: Deep Networks for Video Classification*. 2015. arXiv: 1503.08909 [cs.CV].

[129] Milad Niazi-Razavi, Abdorreza Savadi, and Hamid Noori. "Toward real-time object detection on heterogeneous embedded systems". In: *2019 9th International Conference on Computer and Knowledge Engineering (ICCKE)*. 2019, pp. 450–454. DOI: 10.1109/ICCKE48569.2019.8964764.

[130] Henry Nweke et al. "Deep Learning Algorithms for Human Activity Recognition using Mobile and Wearable Sensor Networks: State of the Art and Research Challenges". In: *Expert Systems with Applications* 105 (Apr. 2018). DOI: 10.1016/j.eswa.2018.03.056.

[131] Kemal Oksuz et al. *Localization Recall Precision (LRP): A New Performance Metric for Object Detection*. 2018. arXiv: 1807.01696 [cs.CV].

[132] Hamid Palangi, Rabab Ward, and Li Deng. "Distributed Compressive Sensing: A Deep Learning Approach". In: *IEEE Transactions on Signal Processing* 64.17 (2016), pp. 4504–4518. DOI: 10.1109/TSP.2016.2557301.

[133] Hamid Palangi et al. "Deep Sentence Embedding Using Long Short-Term Memory Networks: Analysis and Application to Information Retrieval". In: *IEEE/ACM Transactions on Audio, Speech, and Language Processing* 24.4 (Apr. 2016), pp. 694–707. ISSN: 2329-9304. DOI: 10.1109/taslp.2016.2520371. URL: http://dx.doi.org/10.1109/TASLP.2016.2520371.

[134] C. Papageorgiou, M. Oren, and T. Poggio. "A general framework for object detection". In: *Sixth International Conference on Computer Vision (IEEE Cat. No.98CH36271)* (1998), pp. 555–562.

[135] Constantine Papageorgiou and Tomaso Poggio. "A Trainable System for Object Detection". In: *International Journal of Computer Vision* 38 (June 2000), pp. 15–33. DOI: 10.1023/A:1008162616689.

[136] Seetha Parameswaran. "a Review of Machine Learning Techniques Used". In: 12 (2017), pp. 64–69.

[137] M.J. Pearson et al. "Design and FPGA implementation of an embedded real-time biologically plausible spiking neural network processor". In: *International Conference on Field Programmable Logic and Applications, 2005.* 2005, pp. 582–585. DOI: 10.1109/FPL.2005.1515790.

[138] Yuxin Peng, Yunzhen Zhao, and Junchao Zhang. *Two-stream Collaborative Learning with Spatial-Temporal Attention for Video Classification.* 2017. arXiv: 1711.03273 [cs.CV].

[139] J. Petke et al. "Using genetic improvement and code transplants to specialise a c++ program to a problem class". In: *EuroGP, 137–149* (2014).

[140] Patrick Poirson et al. "Fast Single Shot Detection and Pose Estimation". In: (Sept. 2016).

[141] C. Poynton. "A technical introduction to digital video". In: 1996.

[142] Joseph Redmon and Ali Farhadi. *YOLO9000: Better, Faster, Stronger.* 2016. arXiv: 1612.08242 [cs.CV].

[143] Joseph Redmon and Ali Farhadi. *YOLOv3: An Incremental Improvement.* 2018. arXiv: 1804.02767 [cs.CV].

[144] Joseph Redmon et al. *You Only Look Once: Unified, Real-Time Object Detection.* 2016. arXiv: 1506.02640 [cs.CV].

[145] Shaoqing Ren et al. "Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks". In: *Advances in Neural Information Processing Systems.* Ed. by C. Cortes et al. Vol. 28. Curran Associates, Inc., 2015. URL: https://proceedings.neurips.cc/paper/2015/file/14bfa6bb14875e45bba028a21ed38046-Paper.pdf.

[146] Shaoqing Ren et al. "Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks". In: *IEEE transactions on pattern analysis and machine intelligence* 39.6 (June 2017), pp. 1137–1149. ISSN: 0162-8828. DOI: 10.1109/tpami.2016.2577031. URL: https://doi.org/10.1109/TPAMI.2016.2577031.

[147] M.J. Roach and John Mason. "Classification of Video Genre using Audio". In: (June 2001).

[148] Fanny Roche et al. "Autoencoders for music sound modeling: a comparison of linear, shallow, deep, recurrent and variational models". In: May 2019.

[149] Itsaso Rodríguez-Moreno et al. "Video activity recognition: State-of-the-art". In: *Sensors (Switzerland)* 19.14 (2019), pp. 1–25. ISSN: 14248220. DOI: 10.3390/s19143160.

[150] Sitapa Rujikietgumjorn and Nattachai Watcharapinchai. "Real-Time HOG-based pedestrian detection in thermal images for an embedded system". In: Aug. 2017, pp. 1–6. DOI: 10.1109/AVSS.2017.8078561.

[151] D. Rumelhart, Geoffrey E. Hinton, and R. J. Williams. "Learning representations by back-propagating errors". In: *Nature* 323 (1986), pp. 533–536.

[152] David E. Rumelhart and James L. McClelland. "Information Processing in Dynamical Systems: Foundations of Harmony Theory". In: *Parallel Distributed Processing: Explorations in the Microstructure of Cognition: Foundations*. 1987, pp. 194–281.

[153] Olga Russakovsky et al. "ImageNet Large Scale Visual Recognition Challenge". In: *International Journal of Computer Vision* 115 (Sept. 2014). DOI: 10.1007/s11263-015-0816-y.

[154] K. E. A. van de Sande et al. "Segmentation As Selective Search for Object Recognition". In: *IEEE International Conference on Computer Vision*. 2011. URL: https://ivi.fnwi.uva.nl/isis/publications/2011/vandeSandeICCV2011.

[155] Allah Bux Sargana, Plamen Angelov, and Zulfiqar Habib. "A Comprehensive Review on Handcrafted and Learning-Based Action Representation Approaches for Human Activity Recognition". In: *Applied Sciences* 7 (Jan. 2017), p. 110. DOI: 10.3390/app7010110.

[156] Mohammad Javad Shafiee et al. *Fast YOLO: A Fast You Only Look Once System for Real-time Embedded Object Detection in Video*. 2017. arXiv: 1709.05943 [cs.CV].

[157] Prashant Shambharkar et al. "A Survey on Classification of Videos using Data Mining Techniques". In: *Special Issue of International Journal of Computer Applications* 11 (2012), pp. 27–32.

[158] Baoguang Shi et al. "ICDAR2017 Competition on Reading Chinese Text in the Wild (RCTW-17)". In: Nov. 2017, pp. 1429–1434. DOI: 10.1109/ICDAR.2017.233.

[159] Karen Simonyan and Andrew Zisserman. *Two-Stream Convolutional Networks for Action Recognition in Videos*. 2014. arXiv: 1406.2199 [cs.CV].

[160] N. Srinivas and Kalyanmoy Deb. "Muiltiobjective Optimization Using Nondominated Sorting in Genetic Algorithms". In: *Evolutionary Computation* 2.3 (1994), pp. 221–248. DOI: 10.1162/evco.1994.2.3.221.

[161] Kenneth O. Stanley and Risto Miikkulainen. "Efficient Evolution of Neural Network Topologies". In: *Journal of Computing and Information Technology, 7:33–47* (2002).

[162] Haiman Tian et al. "Multimodal deep representation learning for video classification". In: *World Wide Web* 22 (May 2019). DOI: 10.1007/s11280-018-0548-3.

[163] Zhi Tian et al. "FCOS: Fully Convolutional One-Stage Object Detection". In: Oct. 2019, pp. 9626–9635. DOI: 10.1109/ICCV.2019.00972.

[164] B. T. Truong, S. Venkatesh, and C. Dorai. "Automatic genre identification for content-based video categorization". In: *Proceedings 15th International Conference on Pattern Recognition. ICPR-2000* 4 (2000), 230–233 vol.4.

[165] Granville Tunnicliffe Wilson. "Time Series Analysis: Forecasting and Control,5th Edition, by George E. P. Box, Gwilym M. Jenkins, Gregory C. Reinsel and Greta M. Ljung, 2015. Published by John Wiley and Sons Inc., Hoboken, New Jersey, pp. 712. ISBN: 978-1-118-67502-1". In: *Journal of Time Series Analysis* 37 (Mar. 2016), n/a–n/a. DOI: 10.1111/jtsa.12194.

[166] Andrew Turner and Julian Miller. "Cartesian Genetic Programming: Why No Bloat?" In: Apr. 2014. ISBN: 978-3-662-44302-6. DOI: 10.1145/2739482.2756571.

[167] Gül Varol, Ivan Laptev, and Cordelia Schmid. "Long-term Temporal Convolutions for Action Recognition". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 40.6 (June 2018), pp. 1510–1517. DOI: 10.1109/TPAMI.2017.2712608. URL: https://hal.inria.fr/hal-01241518.

[168] Andreas Veit et al. *COCO-Text: Dataset and Benchmark for Text Detection and Recognition in Natural Images*. 2016. arXiv: 1601.07140 [cs.CV].

[169] Shanu Verma, Millie Pant, and Vaclav Snasel. "A Comprehensive Review on NSGA-II for Multi-Objective Combinatorial Optimization Problems". In: *IEEE Access* 9 (2021), pp. 57757–57791. DOI: 10.1109/ACCESS.2021.3070634.

[170] Paul Viola and Michael Jones. "Rapid Object Detection using a Boosted Cascade of Simple Features". In: vol. 1. Feb. 2001, pp. I–511. ISBN: 0-7695-1272-0. DOI: 10.1109/CVPR.2001.990517.

[171] Paul Viola and Michael Jones. "Robust Real-Time Face Detection". In: *International Journal of Computer Vision* 57 (May 2004), pp. 137–154. DOI: 10.1023/B:VISI.0000013087.49260.fb.

[172] Walid and Alamsyah Alamsyah. "Recurrent Neural Network For Forecasting Time Series With Long Memory Pattern". In: *Journal of Physics: Conference Series* 824 (Apr. 2017), p. 012038. DOI: 10.1088/1742-6596/824/1/012038.

[173] Limin Wang et al. *Temporal Segment Networks for Action Recognition in Videos*. 2017. arXiv: 1705.02953 [cs.CV].

[174] Peng Wang, Rui Cai, and Shi-Qiang Yang. "A hybrid approach to news video classification multimodal features". In: *Fourth International Conference on Information, Communications and Signal Processing, 2003 and the Fourth Pacific Rim Conference on Multimedia. Proceedings of the 2003 Joint*. Vol. 2. 2003, 787–791 vol.2. DOI: 10.1109/ICICS.2003.1292564.

[175] G. Wei, L. Agnihotri, and N. Dimitrova. "TV program classification based on face and text processing". In: *2000 IEEE International Conference on Multimedia and Expo. ICME2000. Proceedings. Latest Advances in the Fast Changing World of Multimedia (Cat. No.00TH8532)*. Vol. 3. 2000, 1345–1348 vol.3. DOI: 10.1109/ICME.2000.871015.

[176] D. R. White, A. Arcuri, and J. A. Clark. "Evolutionary improvement of programs". In: *TEC, 15(4):515–538* (2011).

[177] Darrell Whitley, Soraya Rana, and Robert B. Heckendorn. "The island model genetic algorithm: On separability, population size and convergence". In: *Journal of Computing and Information Technology, 7:33–47* (1998).

[178] Dennis G Wilson et al. *Evolving simple programs for playing Atari games*. 2018. arXiv: 1806.05695 [cs.NE].

[179] Erling Wold et al. "Content-Based Classification, Search, and Retrieval of Audio". In: *IEEE MultiMedia* 3.3 (Sept. 1996), pp. 27–36. ISSN: 1070-986X. DOI: 10.1109/93.556537. URL: https://doi.org/10.1109/93.556537.

[180] Alexander Womg et al. "Tiny SSD: A Tiny Single-Shot Detection Deep Convolutional Neural Network for Real-Time Embedded Object Detection". In: *2018 15th Conference on Computer and Robot Vision (CRV)*. 2018, pp. 95–101. DOI: 10.1109/CRV.2018.00023.

[181] F. Wu et al. "Deep parameter optimisation". In: *GECCO* (2015).

[182] Ying Wu, Ting Yu, and Gang Hua. "Tracking appearances with occlusions". In: *2003 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2003. Proceedings.* Vol. 1. 2003, pp. I–I. DOI: 10.1109/CVPR.2003.1211433.

[183] KANG Xiaodong et al. "Unsupervised deep learning method for color image recognition". In: *Journal of Computer Applications* 35.9, 2636 (2015), p. 2636. DOI: 10.11772/j.issn.1001-9081.2015.09.2636. URL: http://www.joca.cn/EN/abstract/article_18542.shtml.

[184] Liwen Xu et al. "Long-Short-Term Memory Network Based Hybrid Model for Short-Term Electrical Load Forecasting". In: *Information* 9 (July 2018), p. 165. DOI: 10.3390/info9070165.

[185] Shen Yan et al. *Multiview Transformers for Video Recognition*. 2022. DOI: 10.48550/ARXIV.2201.04288. URL: https://arxiv.org/abs/2201.04288.

[186] Xun Yuan et al. "Automatic Video Genre Categorization using Hierarchical SVM". In: *2006 International Conference on Image Processing* (2006), pp. 2905–2908.

[187] Xun Yuan et al. "Automatic Video Genre Categorization using Hierarchical SVM". In: *2006 International Conference on Image Processing*. 2006, pp. 2905–2908. DOI: 10.1109/ICIP.2006.313037.

[188] Mehmet Kerim Yucel et al. *Wildest Faces: Face Detection and Recognition in Violent Settings*. 2018. arXiv: 1805.07566 [cs.CV].

[189] Chenrui Zhang and Yuxin Peng. *Better and Faster: Knowledge Transfer from Multiple Self-supervised Learning Tasks via Graph Distillation for Video Classification*. 2018. arXiv: 1804.10069 [cs.CV].

[190] Chenrui Zhang and Yuxin Peng. *Visual Data Synthesis via GAN for Zero-Shot Video Classification*. 2018. arXiv: 1804.10073 [cs.CV].

[191] Peter Zhang. "A neural network ensemble method with jittered training data for time series forecasting". In: *Information Sciences* 177 (Dec. 2007), pp. 5329–5346. DOI: 10.1016/j.ins.2007.06.015.

[192] Jian Zheng et al. "Electric Load Forecasting in Smart Grid Using Long-Short-Term-Memory based Recurrent Neural Network". In: Mar. 2017. DOI: 10.1109/CISS.2017.7926112.

[193] W. Zhu, C. Toklu, and S. Liou. "Automatic news video segmentation and categorization based on closed-captioned text". In: *IEEE International Conference on Multimedia and Expo, 2001. ICME 2001.* (2001), pp. 829–832.

[194] Eckart Zitzler, Marco Laumanns, and Lothar Thiele. "SPEA2: Improving the strength pareto evolutionary algorithm". In: 2001.

[195] Zhengxia Zou et al. *Object Detection in 20 Years: A Survey*. 2019. arXiv: 1905.05055 [cs.CV].

Appendix A

Détection des humains et véhicules dans une vidéo

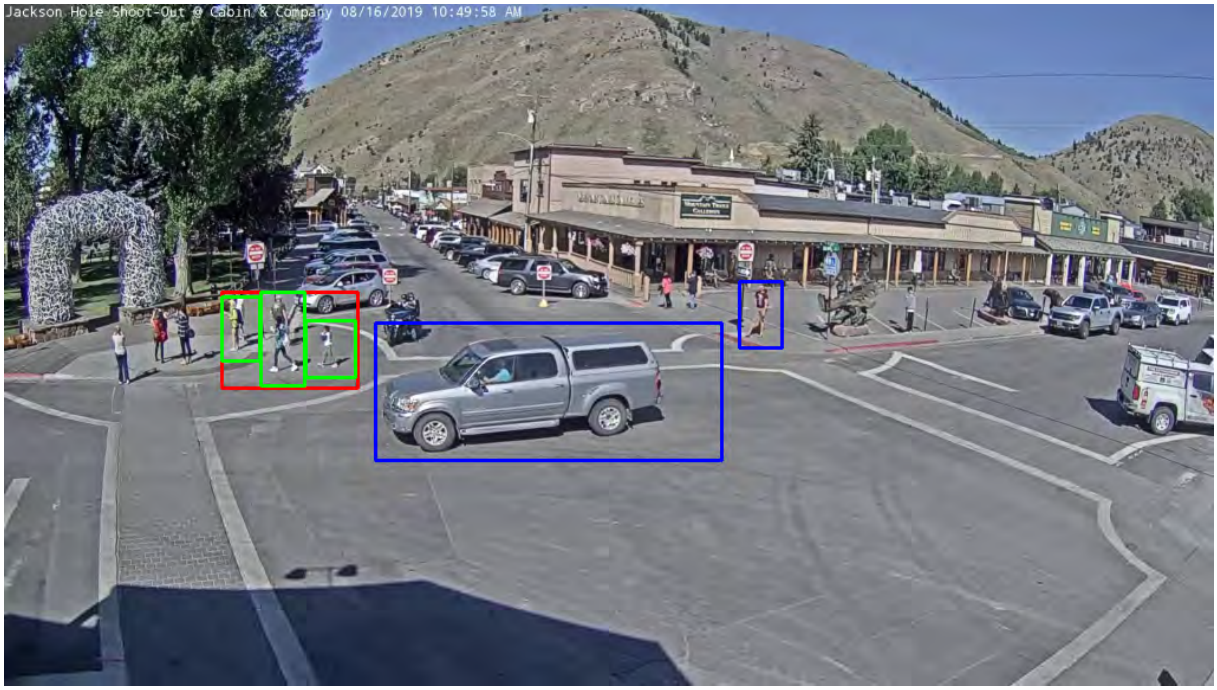


Figure A.1: Caméra 1 - Image 2

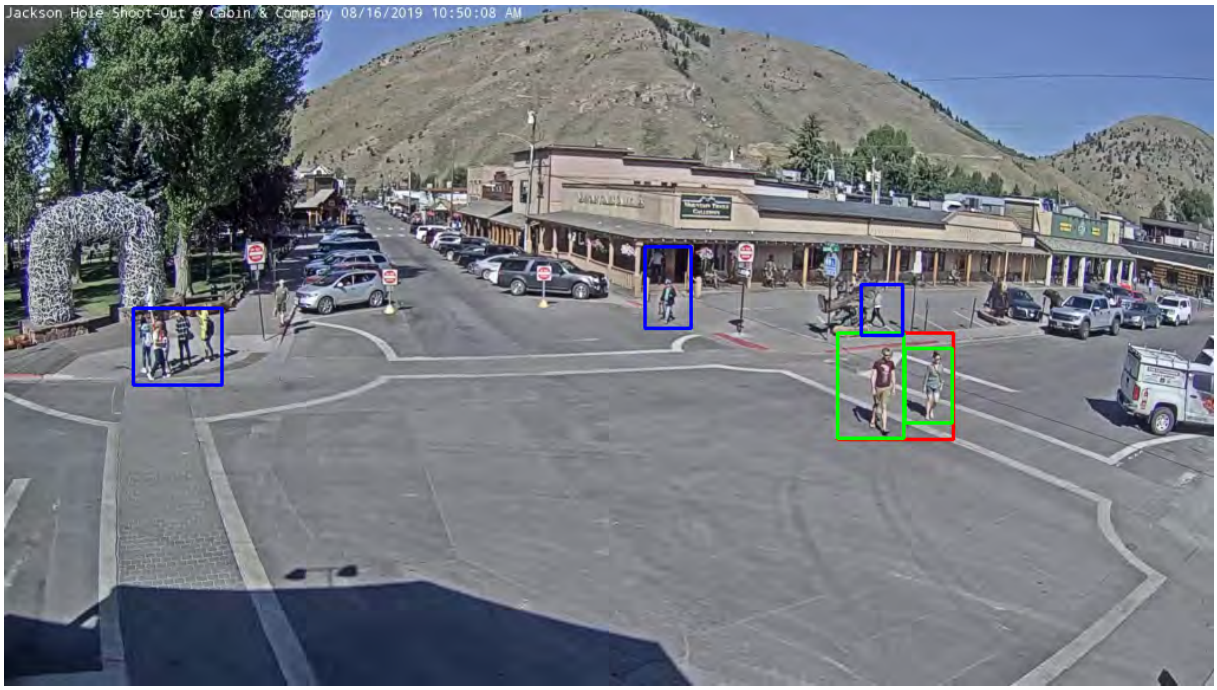


Figure A.2: Caméra 1 - Image 3

Figure A.3: Caméra 2 - Image 1

Figure A.4: Caméra 2 - Image 2

Figure A.5: Caméra 3 - Image 1

Figure A.6: Caméra 3 - Image 2

Figure A.7: Caméra 3 - Image 3