RÉPUBLIQUE ALGÉRIENNE DÉMOCRATIQUE ET POPULAIRE
MINISTÈRE DE L'ENSEIGNEMENT SUPÉRIEUR ET DE LA RECHERCHE SCIENTIFIQUE
UNIVERSITÉ MHAMED BOUGARA BOUMERDES

FACULTÉ DES SCIENCES DE L'INGÉNIEUR
DEPARTEMENT INGÉNIERIE DES SYSTÈMES ÉLECTRIQUE
SPÉCIALITÉ RÉSEAUX ET TÉLÉCOMMUNICATION

MÉMOIRE DE FIN D'ÉTUDE POUR L'OBTENTION D'UN MASTER
SPÉCIALITÉ RÉSEAUX ET TÉLÉCOMMUNICATIONS

THÈME

ESTIMATION D'ÉTAT DE TRAFIC ROUTIER EN TEMPS RÉEL

Réalisé par :
- BENTAYEB Islam Chafa
- BOUHRAOUA Abd El Halim

Promoteur :
- Dr.BENMOUSSA yahia

Encadreurs
- Dr.BENDJOUDI Ahcène
- Dr.YAHIAOUI Saïd

Membres de jury :
-Mr.BAICHE karim Président
-Dr.RIAHLA mohamed amine Examinateur
-Pr.HAMADOUCH mohamed Examinateur

2017/2018

Remerciements

Avant tout propos, Merci au bon Dieu pour nous avoir donné le courage, la patience et la force dans les moments difficiles pour réaliser ce travail.

Nos remerciements s'adressent à Dr.Benmoussa qui a accepté d'être notre promoteur de ce projet, ainsi à nos encadreurs Dr.Yahiaoui et Dr.Bendjoudi, leur expérience ainsi que leur grande connaissance dans le domaine ont joué un rôle important dans la conception de ce travail. Nous profitons de cette occasion pour vous exprimer notre profonde gratitude tout en vous témoignant notre respect.

Nous remercions également les membres du jury pour nous avoir honorés par leur présence et pour avoir accepté d'évaluer ce modeste travail. Veuillez trouver ici le témoignage de notre respect le plus profond.

Nous exprimons nos sincères remerciements à nos familles et nos amis pour avoir cru en nous et pour nous avoir soutenus pendant tout le long de ce mémoire.

Enfin, nous ne manquerions pas de remercier nos parents, qui se sont dépensés pour nous sans compter, nous leur devons notre reconnaissance pour tous leurs sacrifices et nous prions le bon Dieu de les bénir, en espérant qu'ils soient toujours fiers de nous.

Et bien sûr merci à vous qui prenez la peine de nous lire.

Table de matière

Table de figures

Liste des tableaux

Introduction générale

Description de la problématique

Le trafic routier en Algérie, particulièrement dans les grandes villes, connaît une croissance importante et permanente. Selon l'ONS (Office National des Statistiques), le parc national automobile a augmenté d'un demi-million de véhicule en seulement deux ans (entre 2014 et 2016). Il compte, à présent, plus de 5.9 millions de véhicules. [1]

De ce fait, la congestion routière est devenue une réalité quotidienne dans la vie de la plupart des conducteurs algériens et leurs provoque : stress, fatigue et recours à des pratiques dangereuses qui augmentent le risque et le nombre d'accidents de la circulation.

Le temps passé dans les embouteillages est majoritairement considéré comme « perdu », n'étant utilisé ni pour le travail ni pour les loisirs. Cette perte a un coût économique et écologique très important, par exemple les retards de livraison qui peuvent être dramatiques pour les entreprises, la consommation supplémentaire en carburant qui pollue de plus en plus l'atmosphère, ... etc.

On peut donc conclure que la congestion du trafic routier a des conséquences économiques, sociales, sanitaires et écologiques qui constitue un réel problème pour la société algérienne, ce qui nécessite la recherche de solutions pouvant être mises en œuvre rapidement, tel que : l'estimation de la densité ainsi que l'estimation de la vitesse du trafic routier en temps réel.

Contexte de l'étude

L'estimation en temps réel du trafic routier a bénéficié, ces dernières années, d'importantes avancées dans les technologies de détection. Actuellement grâce aux données mobiles envoyées par des appareils à l'intérieur des véhicules, mesurées par le système de localisation du réseau de téléphonie mobile, par les GPS à bord, ou par radio-identification on peut élaborer un estimé du trafic sur l'ensemble du réseau routier mais conditionné par la participation du conducteur en acceptant de partager ses données.

Malheureusement en Algérie l'obtention de l'intégration de ces données mobiles reste impossible. En plus, cette solution n'est pas pratique au niveau des routes peu fréquentées à cause de l'indisponibilité d'informations. Cependant, les caméras de trafic combinées avec les algorithmes de vision par ordinateur permettent d'identifier les véhicules qui passent à un certain endroit et estimer leur vitesse sans nécessité le consentement de ces usagers. Et cela permet de construire un estimé du trafic, sans besoin de nouvelles infrastructures coûteuses.

Etablissement d'accueil

Nous avons mené notre étude au niveau du Centre de Recherche sur l'Information Scientifique et Technique (CERIST) qui est un Établissement Public à caractère Scientifique et Technique (EPST) il a été créé en 1985 par le décret 85-56 du 16 mars 1985 et qui est sous tutelle du Ministère de l'Enseignement Supérieur et de la Recherche Scientifique.

Le CERIST est l'un des organismes de recherche scientifique qui a accumulé un capital d'expérience important en matière de formation continue dans le domaine de l'information scientifique et technique. A ce titre, il est notamment chargé de :

- Mener toute activité de recherche relative à la création, la mise en place et le développement du système national d'information scientifique et technique,
- Promouvoir la recherche dans les domaines des sciences et des technologies de l'information et de la communication et de participer à leur développement,
- Contribuer à la coordination et à la mise en œuvre des programmes nationaux d'information scientifique et technique dans un cadre concerté et en liaison avec les secteurs concernés,
- Contribuer à l'édification et à la promotion de la société de l'information par la mise en place et le développement de réseaux sectoriels d'information thématiques notamment le réseau académique et de recherche, et d'assurer leur connexion avec les réseaux similaires à l'étranger ainsi que par le développement et la généralisation des techniques d'information et de communication dans les activités d'enseignement supérieur,
- Participer à la modernisation du système documentaire universitaire national par la mise en place notamment de bibliothèques virtuelles,

- Réunir les éléments nécessaires à la constitution de bases de données nationales dans les domaines des sciences et de la technologie et en assurer la diffusion,
- Promouvoir la recherche en matière de sécurité de l'information et des réseaux.

Objectifs du mémoire

A présent, malgré les nombreuses recherches dans le domaine de la vision par ordinateur, l'analyse et la compréhension automatique du trafic reste un challenge. Le capteur vidéo semble être le choix d'avenir parce qu'il offre toutes les données nécessaires à la réalisation de nombreuses fonctions (aussi bien de comptage de véhicules que de la détection d'incidents). C'est un capteur très flexible et économiquement intéressant. Cependant, les processus de traitement chargés d'analyser ce flux vidéo sont des systèmes complexes et difficiles à développer.

Le but de ce travail est la conception et la réalisation d'un système d'estimation de l'état du trafic routier en temps réel avec pour entrées les flux vidéo obtenus des caméras du trafic, et en sortie l'état estimé en temps réel du trafic routier. Ce système doit être capable de détecter un véhicule puis l'identifier d'une façon unique pour calculer par la suite sa vitesse moyenne.

Plan du mémoire

Notre mémoire est structuré en 5 chapitres. Tour d'abord, dans le premier chapitre on va présenter des notions de base sur la classification, les différents types d'images et leurs caractéristiques ainsi que les difficultés et les domaines d'application de la classification. Le deuxième chapitre est consacré à la description des réseaux de neurones convolutifs, leurs architectures et la présentation de quelques systèmes de classification basé sur les réseaux de neurones. Ensuite, dans le troisième chapitre on présente les différents modules sur lesquels notre système d'estimation du trafic routier est construit. Quant au quatrième chapitre, il va être consacré à la présentation des différentes bibliothèques et plateformes utilisées pour la réalisation de notre système, et Pour conclure, dans le dernier chapitre on discute les différents résultats obtenus et on termine par une conclusion générale.

Chapitre I

Etat de l'art

I État de l’art

I-1 Introduction

La dernière décennie a vu une importante généralisation de la vidéo-surveillance, en particulier dans l’espace public. Avec la multiplication des caméras se pose la question de leur efficacité. Il est en effet très difficile pour un opérateur humain de gérer un nombre conséquent de caméras. De plus, même avec un nombre très restreint de caméras, les opérateurs ne peuvent maintenir en continu leur attention et manquent donc un nombre conséquent d’événements d’intérêt. Pour ces raisons il est très important d’automatiser autant que possible les tâches de vidéo-surveillance.

I-2 La Vidéosurveillance

La vidéosurveillance est un procédé technique qui peut être utilisé à différentes fins : surveiller son domicile, sécuriser une entreprise, garder un œil sur son bébé et même assurer la sécurité dans les villes via la vidéosurveillance urbaine. [2]

I-2-1 Vidéosurveillance urbaine

L'objectif de ce type d'installation étant la prévention des risques de délinquance et la protection des personnes :

- à proximité ou devant les lieux, bâtiments et installations publics ;
- sur la voie publique.

La vidéosurveillance urbaine peut être utilisée :

- pour la protection des biens et des personnes dans les lieux particulièrement exposés à la délinquance (vols, agressions) ;
- pour la défense nationale et en prévention d'actes terroristes ;
- pour le contrôle du trafic routier et la constatation d'infractions au Code de la route.

I-2-2 Les composants d'un système de vidéosurveillance

Un système de vidéosurveillance est composé des éléments suivants :

I-2-2-1 Les caméras de vidéosurveillance

Elles peuvent être intérieures ou extérieures. C'est l'étanchéité du caisson qui détermine cette fonction. Les caméras extérieures sont habituellement dotées d'une casquette qui protège l'objectif des projections d'eau.

On distingue également les caméras simples de celles à vision nocturne. Les premières ne sont fonctionnelles qu'en plein jour ou dans des locaux éclairés, les secondes peuvent fonctionner de jour comme de nuit. La sensibilité des caméras infrarouges est variable, certains modèles fonctionnent dans l'obscurité totale, d'autres nécessitent un seuil minimal d'illumination.

L'un des critères principaux des caméras de vidéosurveillance est leur résolution. Au-delà de 700px de résolution horizontale, on parle de caméra HD.

I-2-2-2 Un enregistreur de vidéosurveillance

L'enregistreur permet de stocker les images et de piloter et programmer le système. Les modèles les plus simples stockent les images et permettent de les visualiser.

I-2-2-3 La connectique de vidéosurveillance

La connectique permet de relier les caméras à l'enregistreur. Il s'agit de câble blindé conçu pour acheminer les signaux vidéo de la caméra vers l'enregistreur. Dans le cas des systèmes PPOE ce même câble permet l'alimentation électrique des caméras.

I-3 Notion de base sur l'image

I-3-1 Définition d'une image

Une image est une représentation planaire d'une scène ou d'un objet situé en général dans un espace tridimensionnel, elle est issue du contact des rayons lumineux provenant des objets formants la scène avec un capteur (caméra, scanner, rayons X, ...). Il ne s'agit en réalité que d'une représentation spatiale de la lumière.

L'image est considérée comme un ensemble de points auquel est affectée une grandeur physique (luminance, couleur). Ces grandeurs peuvent être continues (image analogique) ou bien discrètes (images digitales). Mathématiquement, l'image représente une fonction continue IF, appelée fonction image, de deux variables spatiales représentée par IF(x, y). Cette fonction quantifie l'intensité lumineuse de n'importe quel point dans l'image .Dans une image en noir et blanc, l'intensité est le niveau de gris : plus un point est sombre, plus son niveau de gris est faible. [3]

La fonction Image peut se représenter sous la forme suivante :

$$\begin{array}{rccc} I : & \mathbb{R}^2 & \rightarrow & \mathbb{R} \\ & (x,y) & \mapsto & I(x,y) \end{array}$$

Dans ce cas, l'intensité d'un point désigne sa couleur. Celle-ci peut être perçue comme un mélange des trois couleurs primaires (rouge, vert et bleu). Ainsi, une image ne correspond non plus à une seule fonction, mais à trois : on associe à chaque point son intensité de rouge, de vert et de bleu. Ces trois valeurs sont respectivement notées $r(x,y)$, $g(x,y)$ et $b(x,y)$ et stockées dans un vecteur colonne de taille trois, de sorte que l'image puisse être représentée comme une fonction vectorielle :

$$\begin{array}{rccc} I : & \mathbb{R}^2 & \rightarrow & \mathbb{R}^3 \\ & (x,y) & \mapsto & \begin{bmatrix} r(x,y) \\ g(x,y) \\ b(x,y) \end{bmatrix} \end{array}$$

I-3-2 Caractéristiques d'une image numérique

L'image est un ensemble structuré d'informations caractérisé par les paramètres suivants :

I-3-2-1 Un pixel

Contraction de l'expression anglaise « Picture Eléments », le pixel est le plus petit point de l'image, c'est une valeur numérique représentative des intensités lumineuses.

Si le bit est la plus petite unité d'information que peut traiter un ordinateur, le pixel est le plus petit élément que peuvent manipuler les matériels et logiciels sur l'image. [4]

I-3-2-2 La résolution

La résolution d'une image correspond au niveau de détail qui va être représenté sur cette image. C'est le nombre de pixels par unité de longueur dans l'image à numériser. Elle est en dpi (dots per inch) ou en ppp (points par pouce). [5]

Plus le nombre de pixels est élevé par unité de longueur de l'image à numériser, plus la quantité d'information qui décrit l'image est importante et plus la résolution est élevée. [6]

I-3-2-3 Dimension

L'image numérique se présente sous forme d'une matrice dont les éléments sont des pixels. La dimension d'une image est le nombre de lignes de cette matrice multiplié par le nombre de colonnes, et qui nous donne le nombre total de pixels dans cette image. [4]

I-3-2-4 Taille du fichier image

C'est le nombre de pixels multiplié par le nombre d'octets nécessaires au codage d'un seul pixel. Les images brutes codent en général chaque pixel sur un octet, Par contre les images de très haute qualité sont codées sur 3, 4, voire 6 octets par pixel. Ces fichiers sont donc très volumineux et subissent une compression pour le stockage et la transmission. [7]

Pour une image de taille MxN, où chaque pixel est codé sur k bits, l'espace mémoire nécessaire pour le stockage de cette image est donné par :

$$S = k \times M \times N$$

I-3-2-5 Types des images

- **Images binaire**

Une image binaire est une matrice rectangulaire dont les éléments valent 0 ou 1. Lorsque l'on visualise une telle image, les zéros sont représentés par des noirs et les uns par des blancs.

- **Image d'intensité**

Une image d'intensité est une matrice dont laquelle chaque élément est un réel compris entre zéro et un.

On parle aussi d'image en niveaux de gris car les valeurs comprises entre 0 et 1 représentent les différents niveaux de gris.

- **Image couleur RGB**

Pour représenter la couleur d'un pixel, il faut donner trois nombre qui correspondent au dosage des trois couleurs de base : rouge, vert et bleu (RGB). On peut ainsi représenter une image couleur par trois matrices, chaque matrice correspondant à une couleur de base.

I-3-3 Les différents formats d'images

On peut classer les images en deux formats :

I-3-3-1 Format vectorielle

Dans une image vectorielle les données sont représentées par des formes géométriques simples qui sont décrites d'un point de vue mathématique. Par exemple, un cercle est décrit par une information du type (cercle, position du centre, rayon). Ces images sont essentiellement utilisées pour réaliser des schémas ou plans.

I-3-3-2 Format matricielle

Une image matricielle est formée d'un tableau de points ou pixels. Plus la densité des points sont élevée, plus le nombre d'informations est grand et plus la résolution de l'image est élevée. Corrélativement la place occupée en mémoire et la durée de traitement seront d'autant plus grandes.

I-3-4 Types des traitements d'Images

On désigne l'ensemble des opérations sur les images numériques, qui transforment une image en une autre image. Il existe différents types de traitement d'images:

I-3-4-1 Traitement ponctuel

En chaque point on applique une formule de modification du niveau de gris indépendamment de la localisation géométrique de ce pixel dans l'image (sa position), c.à.d. le nouveau niveau

de gris dépend seulement de l'ancien niveau de gris du pixel considéré.

I-3-4-2 Traitement local

Le nouveau niveau de gris en un point dépend de l'ancien niveau de gris du pixel considéré et de ses voisins.

I-3-4-3 Traitement global

Le nouveau niveau de gris en un point dépend de l'ancien niveau de gris de tous les pixels de l'image.

I-3-5 Objet

Les objets physiques sont les objets du monde réel qui apparaissent dans les scènes observées par les caméras. Les objets physiques sont divisés en deux types : les objets de contexte et les objets mobiles.

I-3-5-1 Les objets de contexte

Sont des objets physiques qui sont habituellement statiques (p. ex. les murs). Dans le cas où ils ne sont pas statiques, leurs mouvements peuvent être prédits par les informations contextuelles p.ex. Les chaises, les portes sont des objets de contexte.

I-3-5-2 Les objets mobiles

Sont des objets physiques qui peuvent être perçus dans les scènes par leurs mouvements. [8] Il est cependant difficile de prédire leurs mouvements P. ex. les personnes, les véhicules.

I-3-6 Caractéristiques, ou zones d'intérêt d'une image

En vision par ordinateur, le terme de ***(local) features*** désigne des zones intéressantes de l'image numérique. Ces zones peuvent correspondre à des contours, des points ou des régions d'intérêt. A chaque *feature* détectée est associé un vecteur, appelé **descripteur** (***feature descriptor*** ou ***feature vector***), qui, comme son nom l'indique, décrit la zone concernée. [9]

I-3-7 La vidéo

La vidéo est une succession d'images animées défilant à une certaine cadence afin de créer une illusion de mouvement pour l'œil humain. [8]

I-4 Traitement d'image : différentes approches

I-4-1 Difficultés de la reconnaissance d'objet

I-4-1-1 Eclairage

Les conditions d'éclairages peuvent différer au cours de la journée, de plus, les conditions météorologiques peuvent affecter l'éclairage de l'image, de même, des images in-door ou out-door de même objet peuvent avoir différents conditions d'éclairage.

Quel que soit les conditions d'éclairage, le système doit être capable de reconnaître l'objet.

I-4-1-2 Positionnement

L'objet peut être capturé par la caméra par des angles différents, donc le système doit prendre ça en considération.

I-4-1-3 Rotation

Le système doit être capable de gérer la difficulté de la rotation, comme montré dans la « figure 1 », la lettre « A » peut apparaître dans n'importe quelle forme, mais, l'orientation de cette lettre ou de l'image, ne doit pas affecter la reconnaissance de l'objet.

Figure 1 Caractère A avec rotation

I-4-1-4 Occlusion

La condition lorsque l'objet dans une image n'est pas complètement visible est appelée occlusion. [6]

I-4-1-5 L'échelle

La modification de la taille de l'objet dans l'image ne doit pas affecter l'exactitude du système de reconnaissance d'objet.

Un système de reconnaissance d'objets doit prendre en considérations toutes les difficultés indiquées ci-dessus pour être efficace et robuste.

I-4-2 Détection d'objet

Pour extraire une information, les systèmes de vision font appel à des méthodes très diversifiées de traitement d'image et d'apprentissages.

La détection et le suivi d'objet jouent un rôle important dans de nombreuses applications de vision artificielle, telle que la classification vidéo, la surveillance et le routage robotique autonome. L'utilisation d'informations dans une seule image est la méthode la plus utilisée pour la détection d'objets. Bien que certaines méthodes de détections d'objets utilisent les informations temporelles calculées à partir de l'analyse d'une séquence d'image afin de réduire le nombre de fausses détections et augmenter la précision. [10]

La première étape d'un système de vision artificielle consiste à prendre une ou plusieurs images de l'objet à inspecter à partir d'un dispositif d'acquisition. Les images acquises sont numérisées et traitées. Suivant le problème posé, des méthodes de traitement d'images sont utilisées afin d'extraire l'information pertinente. Une décision est ensuite prise par rapport à des critères prédéfinis et une action est effectuée. Par exemple dans notre projet, après détection d'une voiture le premier réseau fait appel au deuxième réseau pour extraire la plaque d'immatriculation de cette voiture.

I-4-2-1 La segmentation

En analyse d'images, on fait souvent la distinction entre traitement bas niveau et haut niveau. Les premiers travaillent sur les valeurs attachés aux pixels de l'image sans faire le lien entre la réalité qu'elles représentent. Tandis que les secondes opèrent sur des entités symboliques constituant une interprétation de la réalité extraite de l'image. La segmentation est un traitement bas niveau qui consiste à diviser l'ensemble de pixels en régions connexes, homogènes et différentes de ses voisins. Ici, on ne cherche pas à déterminer ce que les régions représentent. La qualité de l'interprétation d'une image dépend fortement de celle de la

segmentation. Malgré la grande diversité de méthodes, les résultats de segmentation restent moyens et varient beaucoup en fonction de la technique choisie. Une méthode de segmentation générale et automatique est difficile à concevoir étant donnés les différentes type de régions pouvant être présentes dans une image.

La segmentation peut s'appuyer sur une détection préalable des contours présents dans l'image, c'est ce qu'on appelle l'approche frontière, ou sur une agrégation des zones homogènes de l'image, il s'agit alors d'une approche région. Dans les méthodes les plus modernes on profite des deux approches conjointes en les faisant coopérer.

I-4-2-1-1 L'approche frontière

On parle souvent ou également de détection de contour. Un contour correspond de façon très intuitive à une zone linéaire de l'image qui sépare deux régions de niveaux de luminance différents. Pour mériter le qualificatif de contour, les régions doivent avoir une certaine étendue et la linéarité de la zone frontière doit être raisonnablement régulière. Ces critères très approximatifs sont couramment utilisés par le système visuel humain pour discerner un vrai contour d'une zone perturbée, nous dirons plutôt bruitée, située dans une région homogène. En vision artificielle, la grande difficulté est la prise en compte du caractère approximatif de cette notion de contour qui prend son véritable sens dans la globalité du processus mis en œuvre par le système visuel humain dans la sélection de ces lignes frontières.

Si on considère l'image simple de la figure 2, il est facile d'identifier des contours caractérisés par un saut de luminance important, par exemple au bord du toit, la figure 3 montre le résultat obtenue après application de l'une des méthodes utilisée dans ce domaine (gradien, laplacien…).

Figure 2 Image « maison »

Figure 3 Image maison après application de l'une des méthodes de détection de contours

L'approche frontière exposée dans le paragraphe précédent permet d'obtenir des images en niveaux de gris où les contours correspondent aux zones les plus claires. Il reste à binariser ces images pour que seules les lignes des contours soient blanches, le fond restant parfaitement noir. Les principales difficultés sont la variabilité du niveau de gris dans les zones contours et l'absence de critère pour repérer le niveau de gris moyen. Les lignes de contours doivent présenter une certaine continuité. On adopte ainsi souvent une démarche en deux étapes. Dans la première on cherche à extraire les pixels les plus clairs qui appartiennent probablement aux contours. Dans un deuxième temps, parmi ceux-ci, on sélectionne ceux qui forment des lignes continues.

I-4-2-1-2 L'approche région

Les méthodes de l'approche région cherchent à regrouper directement des pixels ayant une propriété commune, l'ensemble des regroupements de pixels définis à la fin une segmentation de l'image. Les plus importantes sont celles qui précèdent par croissance de région et par division/fusion de région.

La segmentation par croissance de région vise à regrouper les pixels adjacents de l'image dont les attributs varient de façon négligeable. Il faut choisir un prédicat ou critère d'uniformité P à partir de lequel la croissance de région s'effectuera.

Les méthodes du type divisions et fusion comportent, comme leur nom l'indique, deux étapes. Dans la première, l'image est divisée récursivement jusqu'à ce que toutes les régions vérifient le prédicat P. dans la deuxième étape, les régions adjacentes sont regroupées, tant que les

régions résultantes vérifient P.

I-4-2-2 La classification

Contrairement à la segmentation d'images, ici on cherche à identifier ce que représente chacune des régions segmentées. Dans ce domaine on procède par l'attribution des pixels de l'image à des classes connues à priori (classification supervisée) ou à des classes inconnues (classification non supervisée).

I-4-2-2-1 Classification supervisée

L'objectif de la classification supervisée est principalement de définir des règles permettant de classer des objets dans des classes à partir de variables qualitatives ou quantitatives caractérisant ces objets. On dispose au départ d'un échantillon dit d'apprentissage dont le classement est connu. Cet échantillon est utilisé pour l'apprentissage des règles de classement.

On utilise souvent un deuxième échantillon indépendant, dit de validation ou de test pour étudier la fiabilité de ces règles et le évaluer en cas de sous apprentissage.

Principe

On dispose d'éléments déjà classé: voiture, maison, vélo …. Etc. et on veut classer un nouvel élément dans l'une de ces classes

I-4-2-2-2 Classification non supervisée

Procède de la façon contraire, c'est à dire ne nécessitent aucun apprentissage et aucune tâche préalable d'étiquetage manuel. Elle consiste à représenter un nuage des points d'un espace quelconque en un ensemble de groupes appelé Cluster. Un Cluster est une collection d'objets qui sont similaires entre eux et qui sont dissemblables par rapport aux objets appartenant à d'autres groupes.

Principe

On dispose d'éléments non classés (exemple voiture) et on veut regrouper tous les éléments qui appartient à cette classe dans la même classe.

I-4-2-3 Classification des images et l'apprentissage machine (approche classique)

Jusqu'à récemment, les systèmes de reconnaissance des images classiques étaient composés de deux blocs: un extracteur de caractéristiques (feature extractor en anglais), suivi d'un classifieur entrainable simple. L'extracteur de caractéristiques est programmé «à la main», et transforme le tableau de nombres représentant l'image en une série de nombres, un vecteur de caractéristiques, dont chacun indique la présence ou l'absence d'un motif simple dans l'image. Ce vecteur est envoyé au classifieur, dont un type commun est le classifieur linéaire. Ce dernier calcule une somme pondérée des caractéristiques: chaque nombre est multiplié par un poids (positif ou négatif) avant d'être sommé. Si la somme est supérieure à un seuil, la classe est reconnue. Les poids forment une sorte de «prototype» pour la classe à laquelle le vecteur de caractéristiques est comparé. Les poids sont différents pour les classifieurs de chaque catégorie, et ce sont eux qui sont modifiés lors de l'apprentissage. Les premières méthodes de classification linéaire entrainable datent de la fin des années cinquante et sont toujours largement utilisées aujourd'hui. [11]

I-4-2-3-1 Algorithmes utilisés pour la classification:

I-4-2-3-1-1 KNN

La méthode de k voisins les plus proches en anglais KNN (k-nearest neighbor) est une méthode de l'apprentissage supervisé dédié à la classification. Cet algorithme figure parmi les plus simples algorithmes d'apprentissage artificiel. L'objectif de l'algorithme est de classer les exemples non étiquetés sur la base de leur similarité avec les exemples de la base d'apprentissage.

Le principe de cet algorithme de la classification est très simple. On lui fournit :

- Un ensemble de données d'apprentissage D.

- Une fonction de distance d.

- Un entier K

Pour tout nouveau point de test x, pour lequel il doit prendre une décision, l'algorithme recherche dans D, les K points les plus proches de x au sens de la distance d, et attribut x à la classe qui est la plus fréquente parmi ces K voisins.

Dans l'exemple suivant, on a 3 classes, et le but de trouver la valeur de la classe de l'exemple inconnu x.

On prend la distance Euclidienne et k=5 voisins.

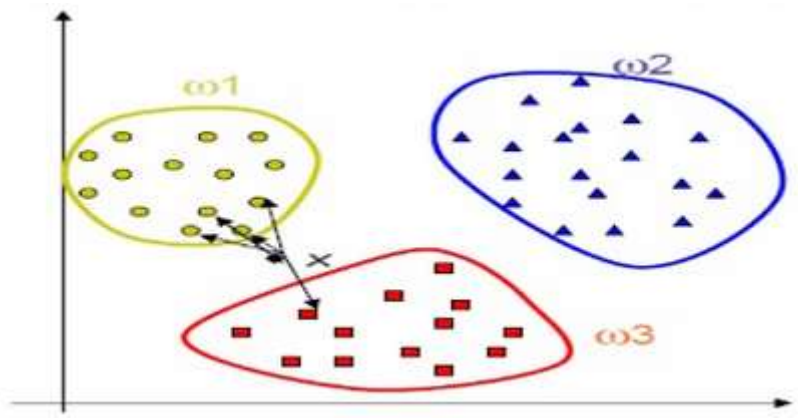


Figure 4 Exemple de fonctionnement de l'algorithme KNN

On remarque que parmi les 5 voisins les plus proches, 4 appartiennent à ω1 et 1 appartient à ω3 donc x appartiendra à la classe ω1.

I-4-2-3-1-2 SIFT – Scale Invariant Feature Transforme :

SIFT est un algorithme qui a été publié par David Lowe en 1999, cet algorithme consiste à chercher des points caractéristiques (features) sur une photo qui seront décrit chacun par des coordonnées (x et y), une orientation et une échelle. L'idée générale de SIFT est de trouver des points-clés qui sont invariants à plusieurs transformations: rotation, échelle et illumination. Ensuite, une comparaison est faite entre les deux images en utilisant ces points-clés.

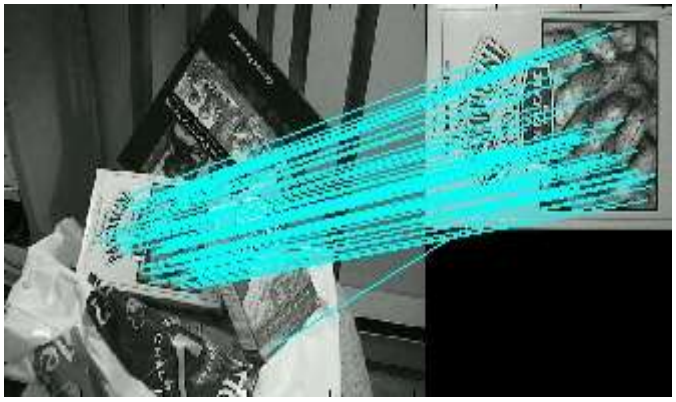


Figure 5 Comparaison entre deux images en utilisant l'algorithme SIFT

I-4-2-3-1-3 SURF- Speed Up Robust Features:

Surf est un algorithme qui s'appuie largement sur SIFT avec des améliorations qui le rend plus rapide et plus robuste.

I-4-2-4 Classification des images et les réseaux de neurones

Le problème de l'approche classique de la reconnaissance des images est qu'un bon extracteur de caractéristiques est très difficile à construire, et qu'il doit être repensé pour chaque nouvelle application.

C'est là qu'intervient l'apprentissage profond ou deep learning en anglais. C'est une classe de méthodes dont les principes sont connus depuis la fin des années 1980, mais dont l'utilisation ne s'est vraiment généralisée que depuis 2012, environ.

L'idée est très simple: le système entrainable est constitué d'une série de modules, chacun représentant une étape de traitement. Chaque module est entrainable, comportant des paramètres ajustables similaires aux poids des classifieurs linéaires. Le système est entraîné de bout en bout: à chaque exemple, tous les paramètres de tous les modules sont ajustés de manière à rapprocher la sortie produite par le système de la sortie désirée. Le qualificatif profond vient de l'arrangement de ces modules en couches successives.

I-4-3 Application de la reconnaissance d'objets

I-4-3-1 Reconnaissance biométrique

La technologie biométriques utilise des traits physiques ou comportementaux humains pour reconnaître tout individu à des fins de sécurité et d'authentification. [12]

La biométrie est l'identification d'un individu basée sur des caractéristiques telles que les empreintes digitales, la géométrie de la main, la rétine de l'iris, l'ADN ...Etc. [6]

I-4-3-2 Surveillance

Les objets peuvent être reconnus et suivis pour divers systèmes de vidéosurveillance. La reconnaissance d'objet est nécessaire pour que la personne suspecte ou le véhicule soit suivi.

I-4-3-3 Inspection industrielle

Des pièces de machine peuvent être reconnues en utilisant la reconnaissance d'objet et peuvent être surveillées pour être avertis en cas d'un dysfonctionnement ou des dommages.

I-4-3-4 Robotique

La recherche de robots autonomes est l'un des problèmes les plus importants de ces dernières années. Comme par exemple la compétition de football des robots humanoïdes est très populaire, les joueurs s'appuient très fortement sur leurs systèmes de vision lorsqu'ils sont dans des environnements dynamiques et imprévisibles.

Le système de vision peut aider le robot à collecter diverses informations sur l'environnement comme: la position de la balle, positions des obstacles à éviter ...etc.

I-4-3-5 Médicale

La reconnaissance d'objet peut être utilisée dans le domaine médical comme pour la détection des tumeurs dans les images IRM, la détection du cancer de la peau...etc. [6]

I-4-3-6 Système de véhicules intelligents

Des systèmes de véhicules intelligents (comme les voitures autonomes lancées récemment par Uber) sont nécessaire pour la détection et la reconnaissance des panneaux de signalisation, en particulier pour la détection et le suivi de véhicules et aussi pour éviter la collision avec les piétons ou avec d'autres voitures.

Conclusion

Ce chapitre a été consacré à la présentation des notions de bases sur les images ainsi de la classification, ses domaines d'application et les différentes difficultés qu'un système de reconnaissance d'objet doit surmonter.

Chapitre II

Réseaux de neurones convolutifs

II Réseaux de neurones convolutifs

II-1 Introduction

Les réseaux de neurones artificiels sont des réseaux neuronaux multicouches entièrement connectés qui ressemblent à la figure ci-dessous :

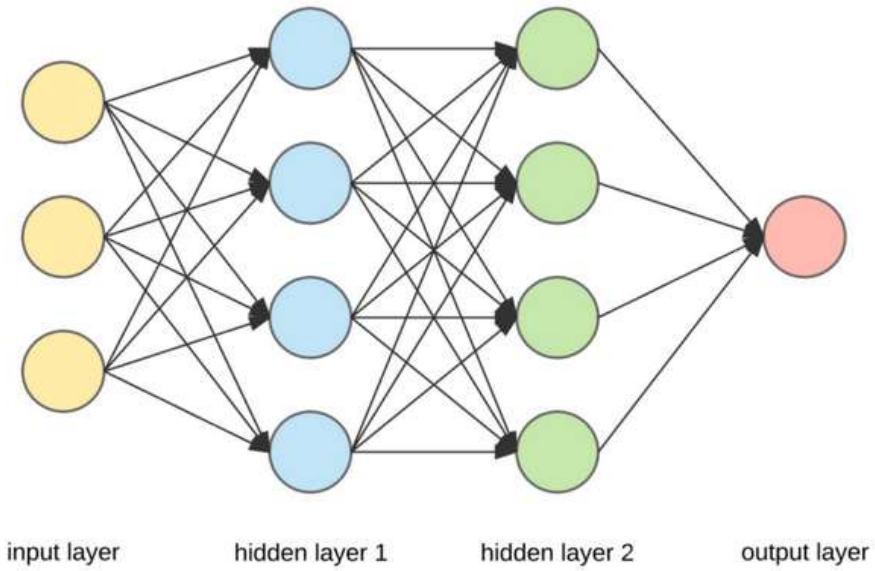


Figure 6 Architecture des réseaux de neurones

Ils sont composés d'une couche d'entrée, de plusieurs couches cachées et d'une couche de sortie. Chaque nœud d'une couche est connecté à tous les autres nœuds de la couche suivante. Plus il y a de couches cachées plus le réseau est profond.

Si nous zoomons sur l'un des nœuds cachés ou de sortie, ce que nous allons rencontrer est le suivant :

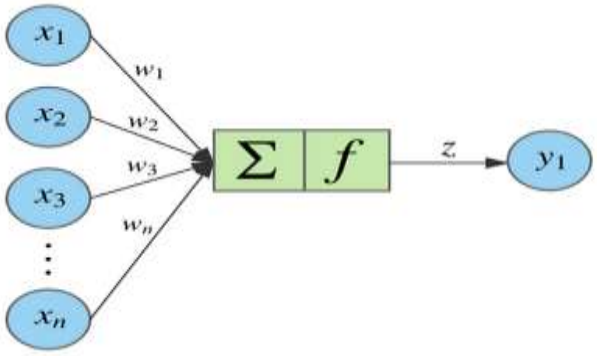


Figure 7 Zoom sur un nœud de réseaux de neurones

Un nœud donné prend la somme pondérée de ses entrées, et passe à travers une fonction d'activation non linéaire. C'est la sortie du nœud, qui devient alors l'entrée d'un autre nœud dans la couche suivante. Entraîner un réseau neuronal signifie d'apprendre les poids associés à chaque classe.

La somme pondérée de ses entrées passe par une fonction d'activation non linéaire. Il peut être représenté comme un produit scalaire vectoriel, où n est le nombre d'entrées pour le nœud.

$$z = f(x \cdot w) = f\left(\sum_{i=1}^{n} x_i w_i\right)$$

II-2 Architecture d'un CNN:

Tous les modèles CNN suivent une architecture similaire, comme la montre la figure ci-dessous.

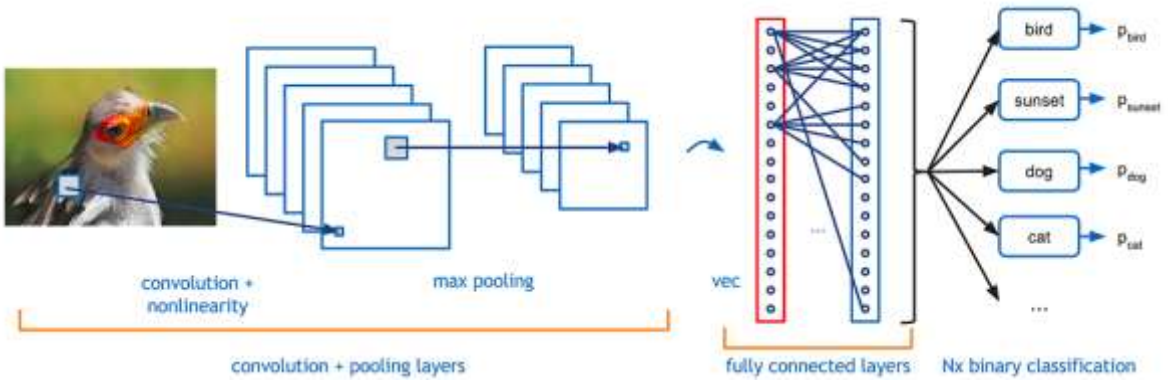


Figure 8 Architecture d'un réseau de neurones convolutifs

En entrée on donne une image avec laquelle nous travaillons. Nous effectuons une série de convolutions + opérations de pooling, suivies d'un certain nombre de couches entièrement connectées.

II-2-1 La couche de convolution

Le bloc de construction principale de CNN est la couche convolutionnelle. La convolution est une opération mathématique utilisée pour extraire les caractéristiques de l'image d'entrée (les bords, couleurs, courbes ...). Comme un détecteur de courbe, le filtre aura une structure de pixels dans laquelle il y aura des valeurs numériques plus élevé le long de la zone qui forme une courbe. [13]

Exemple :

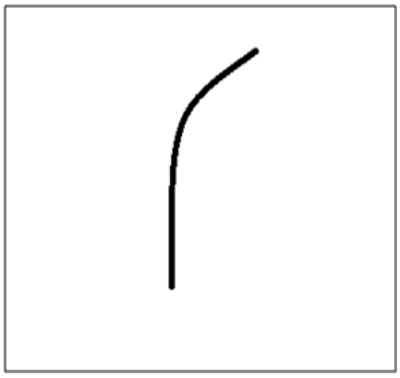


Figure 9 Une courbe

0	0	0	0	0	30	0
0	0	0	0	30	0	0
0	0	0	30	0	0	0
0	0	0	30	0	0	0
0	0	0	30	0	0	0
0	0	0	30	0	0	0
0	0	0	0	0	0	0

Figure 10 La représentation du filtre en pixels

La convolution est appliquée sur les données d'entrée en utilisant un filtre de convolution pour produire une carte de caractéristiques.

Pour expliquer cette étape importante des CNN on utilisera un exemple :

1	1	1	0	0
0	1	1	1	0
0	0	1	1	1
0	0	1	1	0
0	1	1	0	0

Figure 11 Image d'entrée

1	0	1
0	1	0
1	0	1

Figure 12 Un filtre

Sur le côté gauche c'est l'image en entrée de la couche de convolution et sur la droite c'est le filtre de convolution, également appelé le noyau, ceci est appelé une convolution 3x3 en raison de la forme du filtre.

Une opération de convolution est effectuée en glissant le filtre sur l'entrée. À chaque endroit, une opération de multiplication matricielle élément par élément est effectuée suivie par une addition des résultats, et le résultat final est stocké dans la carte de caractéristiques.

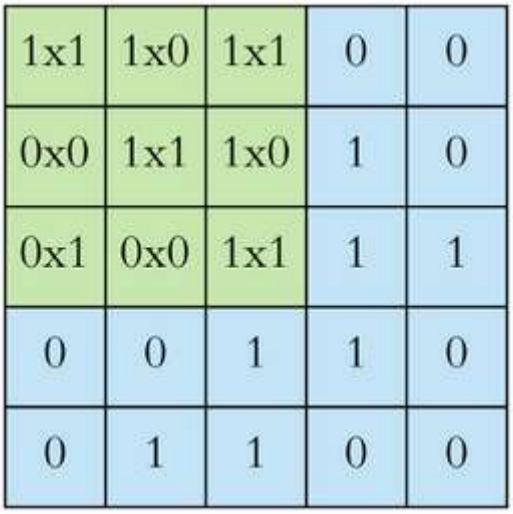


Figure 13 Entrée x filtre

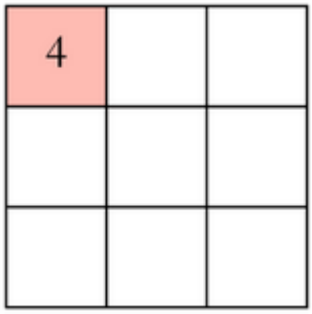


Figure 14 Carte de caractéristiques

La zone verte où l'opération de convolution a lieu s'appelle le champ réceptif. En raison de la taille du filtre, le champ réceptif est également de taille 3x3.

Ici, le filtre est en haut à gauche, la sortie de l'opération de convolution "4" est montrée dans la carte de caractéristiques résultante. Le filtre est ensuite glisser vers la droite, et la même opération sera effectuée, en ajoutant également ce résultat à la carte de caractéristiques. Ce processus se répète jusqu'à ce que le filtre passe par tous les blocs de l'image d'entrée.

Le résultat final est montré dans la figure 15

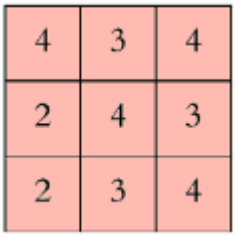


Figure 15 La carte de caractéristique finale

L'exemple précédent est une opération de convolution représentée en 2D à l'aide d'un filtre 3x3. Mais en réalité ces convolutions sont réalisées en 3D, une image est représentée comme une matrice 3D avec des dimensions de hauteur, largeur et profondeur, où la profondeur correspond aux canaux de couleur RGB. Dans ce cas le filtre de convolution a une hauteur et une largeur spécifique (3x3 ou 5x5), et il doit avoir la même profondeur que l'entrée pour couvrir toute l'image.

Plusieurs convolutions sont effectuées sur l'image d'entrée, chacune utilise un filtre différent pour extraire des différentes caractéristiques et chaque résultat est stocké dans une carte de caractéristiques distincte. Les cartes de caractéristiques sont mises les unes sur les autres ce qui devient la sortie final de la couche de convolution.

Dans un deuxième exemple plus réel on a une image de 32x32x3 et on utilise un filtre de taille 5x5x3. Lorsque le filtre est à un emplacement particulier, il couvre un petit volume de l'entrée, et une opération de convolution est effectuée sur cette zone. La seule différence est que cette fois la somme de la matrice multipliée en 3D au lieu de 2D, mais le résultat est toujours un scalaire. Le filtre est glissé sur l'entrée et la convolution est effectuée en stockant le résultat dans une carte de caractéristiques. Cette carte est de taille 32x32x1, comme le montre la *figure 16*

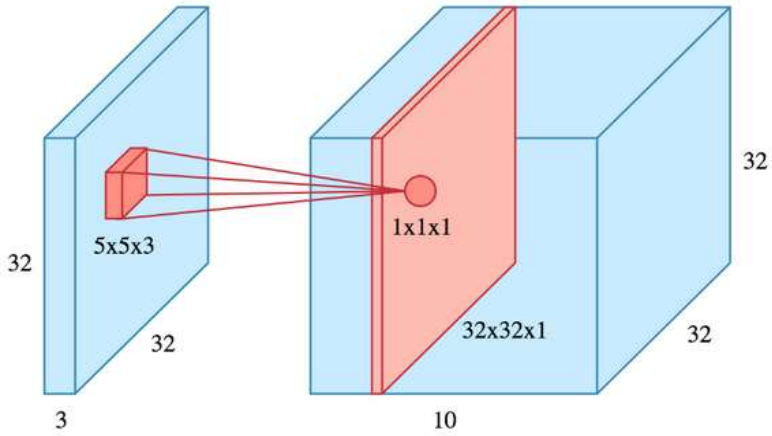


Figure 16 Filtre 3D

Le nombre de cartes de caractéristiques est égale au nombre de filtre utilisées (si on a 10 filtres différents, nous aurions 10 carte de caractéristiques de taille 32x32x1). [14]

II-2-1-1 Le pas (stride)

Stirde spécifie combien le filtre de convolution est déplacé à chaque étape. Par défaut, la valeur est 1. Plus le pas est petit, plus les champs se chevauchent et plus le volume de la sortie sera grand.

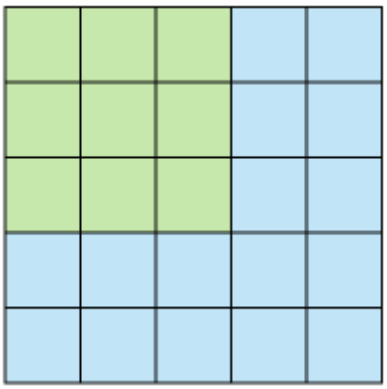
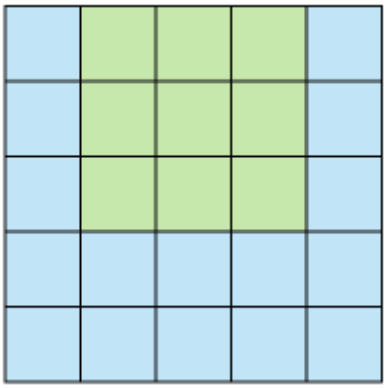


Figure 17 Convolution avec un pas = 1



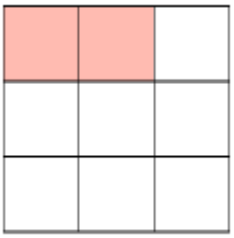
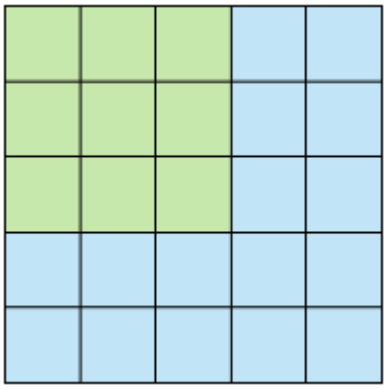


Figure 18 Résultat

On peut avoir de plus grands pas si on veut mois de chevauchement entre les champs réceptifs. Cela rend la carte de caractéristiques plus petite.



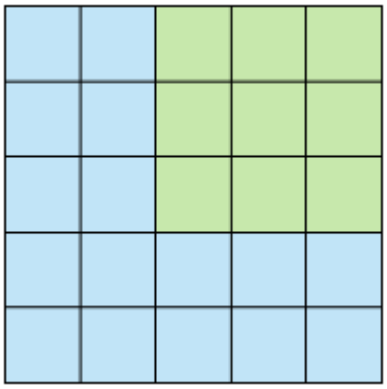


Figure 19 Convolution avec un pas =2

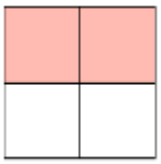


Figure 20 Résultat

II-2-1-2 La marge (padding ou dit encore remplissage)

La marge permet de contrôler la dimension spatiale de volume de sortie. En particulier, il est parfois souhaitable de conserver la même surface que celle du volume d'entrée.

Quand un filtre de 5x5x3 est appliqué à une image de 32x32x3, le résultat sera une carte de 28x28x3, et la taille diminue plus vite quand d'autres filtre sont appliqués, ce qui permet de perdre trop d'informations de l'image d'origine, dans ce cas, si on veut conserver la même taille d'entrée tout en appliquant un filtre de 5x5x3, il faut utiliser un remplissage de 0 de 2 couches, ce qui résulte une entrée de 36x36x3.

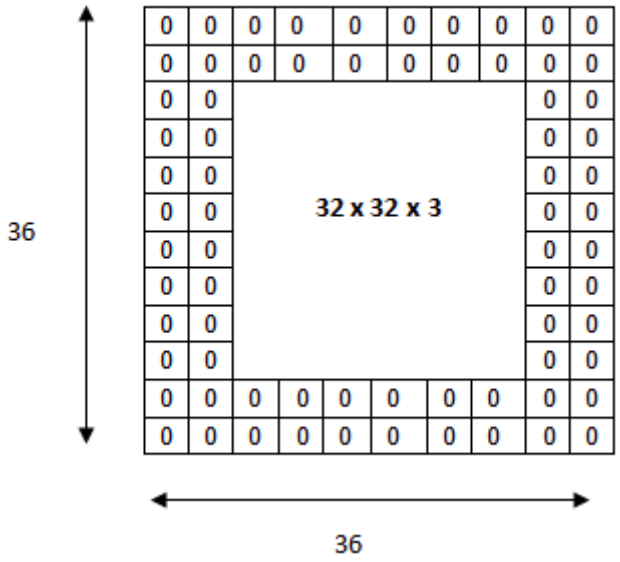


Figure 21 Exemple de remplissage par 0 [14]

La formule pour calculer la taille de sortie pour une couche de convolution :

H= ((W – K + 2P) / S) + 1 [14]

Où :

H : La hauteur et la largeur de la sortie (hight/width)

W : La hauteur et la largeur de l'entrée

K : la taille du filtre

P : padding (remplissage par 0)

S : stride (le pas)

- Si on veut avoir la taille de la sortie égale à la taille de l'entrée, on calcule la marge de la façon suivante :

P = (K – 1) / 2

II-2-2 Couche de correction (ReLU) :

Un élément important dans l'ensemble du processus est l'unité rectifiée linéaire ou ReLu. Les mathématiques derrière ce concept sont assez simples: chaque fois qu'il y a une valeur négative dans un pixel, on la remplace par un 0. F(x)= max (0, x)

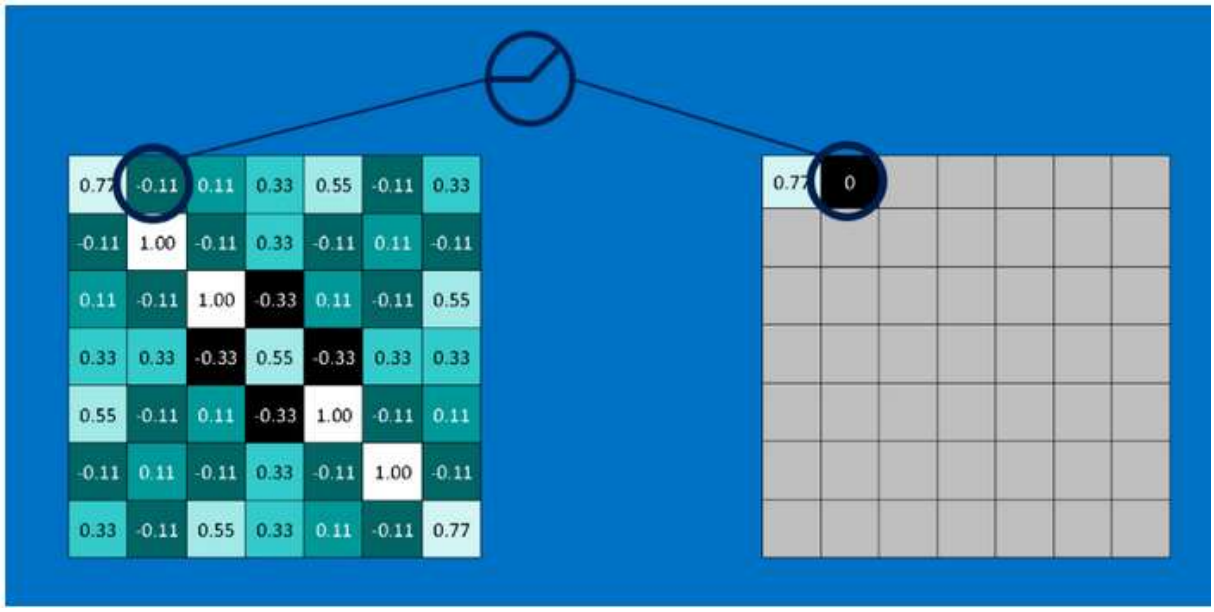


Figure 22 Exemple du résultat de la couche ReLu

II-2-3 La couche Pooling

Le Pooling est une méthode permettant de prendre une large image et d'en réduire la taille tout en préservant les informations les plus importantes qu'elle contient. En effet, il suffit de faire glisser une petite fenêtre pas à pas sur toutes les parties de l'image et de prendre la valeur maximale de cette fenêtre à chaque pas. On pratique, souvent une fenêtre de 2 ou 3 pixels de côté est utilisée et une valeur de 2 pixels pour ce qui est la valeur d'un pas. Cette opération réduit la hauteur et la largeur d'une image mais tout en gardant la profondeur intacte. [14]

Après avoir procédé au pooling, l'image n'a plus qu'un quart du nombre de ses pixels de départ. Parce qu'il garde à chaque pas la valeur maximale dans la fenêtre, il préserve ainsi les meilleurs caractéristiques de cette fenêtre.

Le pooling permet d'avoir un gros gain en puissance de calcul. Cependant, en raison de la réduction agressive de la taille de la représentation (et donc la perte d'information associée), la tendance actuelle est d'utiliser de petit filtres (2x2).

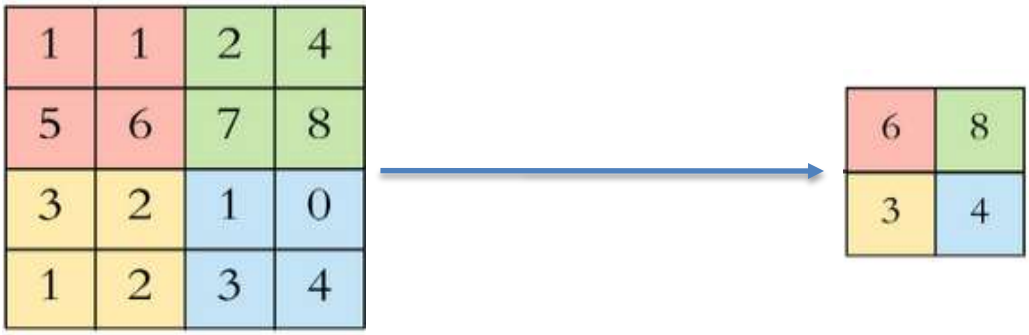


Figure 23 Résultat du pooling 2x2 avec un pas de 2

II-2-4 Couche entièrement connectées (FC)

Après l'extraction des caractéristiques en utilisant une succession de couches de convolutions et de pooling, nous devons classer les données dans diverses classes, et ceci est fait en utilisant un réseau neuronal entièrement connecté (FC).

Pour connecter la sortie de la couche Pooling à la couche FC, nous devons aplatir cette sortie dans un seul tenseur. Par exemple, si on a 100 canaux de 2x2 matrice comme sortie de la couche Pooling, cela signifie que nous devons aplatir toutes ces données dans un seul vecteur avec une seule colonne et 2x2x100=400 lignes.

II-3 Les différents systèmes de détection d'objets par les CNN

II-3-1 R-CNN (Region Convolutional Neural Network)

R-CNN est le 1ér modèle qui a effectué une recherche par régions pour la classification des objets. La classification par RCNN passe par des étapes simples :

- Analyser l'image d'entrée pour les objets possibles en utilisant un algorithme appelé « selective search », générant ainsi ~2000 régions
- Pour chaque région, on utilise un réseau de neurone convolutif (CNN) pour extraire les caractéristiques.
- Nous appliquons ensuite un classificateur (SVM) pour identifier les objets, et toutes les régions avec le même objet sont combinées pour afficher finalement un rectangle

qui entoure l'objet détecté.

La figure 22 Montre le schéma global de R-CNN

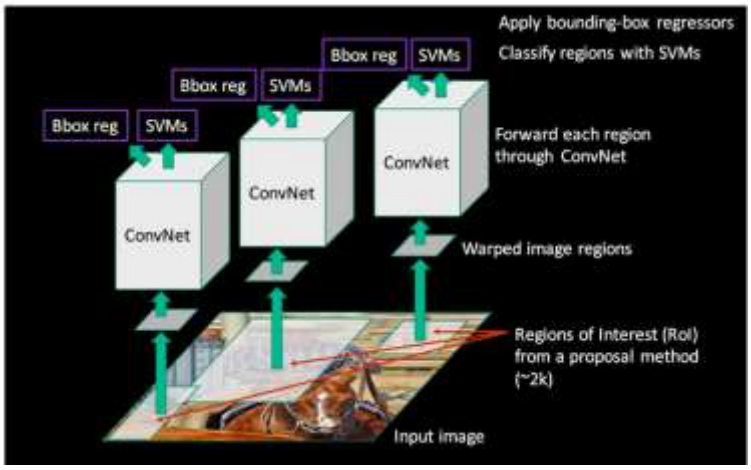


Figure 24 Architecture des R-CNN

R-CNN est lent à l'entraînement et à la détection. Car il répète 2 000 fois la couche de convolution pour extraire les caractéristiques des régions. Il prend aussi beaucoup d'espace mémoire lors de l'entraînement

II-3-2 Fast R-CNN

Le descendant immédiat de R-CNN était FAST-R-CNN, il ressemblait à l'original, mais sa vitesse de détection est améliorée:

Fonctionnement

- Utiliser un algorithme de recherche sur l'image pour générer les zones intéressantes.
- Passer l'image entière par un réseau de neurone pour avoir une carte de caractéristiques
- Pour chaque région obtenue dans l'étape « 1 », utiliser la carte de caractéristiques obtenue dans l'étape 2, et une couche entièrement connectées pour obtenir les coordonnées de l'objet détecté.

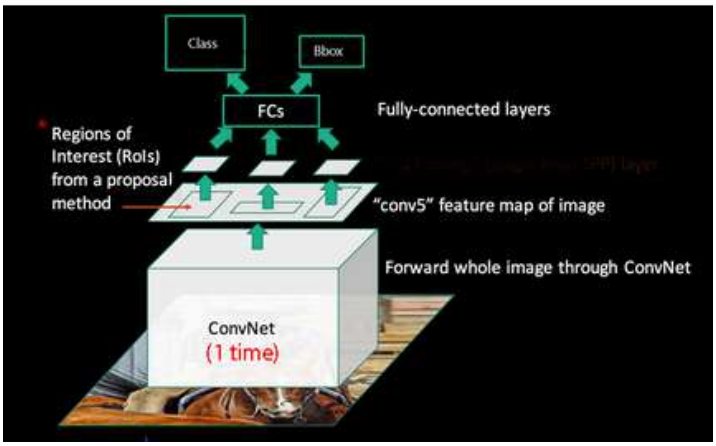


Figure 25 Architecture de Fast R-CNN

II-3-3 Faster R-CNN

Fonctionnement

- Exécuter un réseau de neurone sur l'image entière pour obtenir une carte de caractéristiques.

- Exécuter l'algorithme de recherche des zones d'intérêts en parallèle

- Pour chaque région obtenue dans l'étape « 1 », utiliser la carte de caractéristiques obtenue dans l'étape 2, et plusieurs couche entièrement connectées pour obtenir les coordonnées de l'objet détecté.

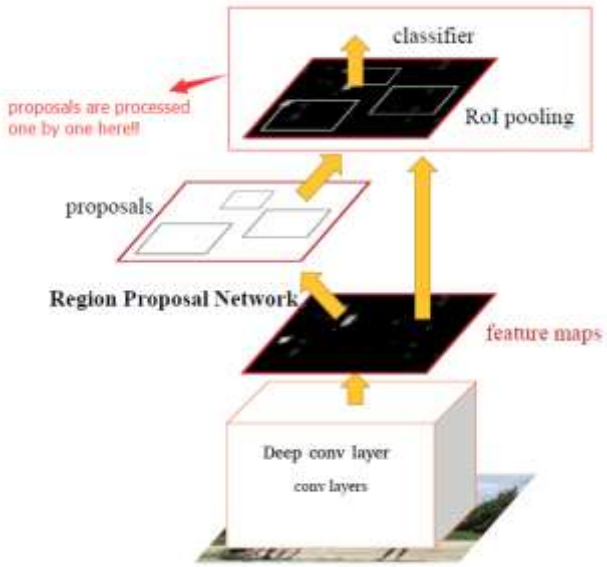


Figure 26 Architecture de Faster R-CNN

La seule différence entre Fast et Faster R-CNN c'est que Faster R-CNN résout le problème de goulot d'étranglement qui est dû au lancement d'algorithme de recherche des zones intéressantes et la couche de convolution en série.

Conclusion

Dans ce chapitre on a présenté les réseaux de neurones convolutifs et leurs différentes couches qui sont capable d'extraire les caractéristiques des images et classifier les objets présents dedans. On a aussi présenté quelques systèmes de reconnaissance d'objet basé sur les réseaux de neurones ainsi que le fonctionnement de chacun entre eux.

Chapitre III

Conception

III-Conception

III-1 Introduction

Il y a plusieurs critères à prendre en compte pour concevoir un système d'estimation de trafic routier, pour notre part le système est basé sur des données sous forme de vidéos issue des caméras routières, cela nous mène à construire un système capable de traiter les vidéo.

D'abord, le système doit reconnaître un véhicule sur une scène et l'identifier d'une façon unique, pour qu'on puisse calculer sa vitesse moyenne de déplacement entre deux endroits. Le système doit être performant en terme précision. En plus, le temps de reconnaissance d'une plaque d'immatriculation doit être rapide de quelques millisecondes. Enfin, le système doit être robuste et fonctionnel de jour comme de nuit, sous différentes conditions d'illumination et selon différentes situations climatiques.

Dans ce chapitre, nous allons exposer l'approche que nous avons suivie pour la conception de notre système d'estimation de l'état du trafic routier. Nous commencerons par une description générale du système en citant notamment les différentes unités qui le composent. Puis, nous expliquerons chaque unité de façon détaillée.

III-2 Description générale du système

Notre système reçoit comme entrée les flux vidéo et donne comme sortie l'état du trafic routier. Il est composé de deux unités principales, à savoir: **unité d'analyse des vidéos et d'extraction d'information** et **unité de calcul et de prédiction de l'état du trafic routier**. Chaque unité est composée de plusieurs modules.

La figure 28 montre la description générale du système

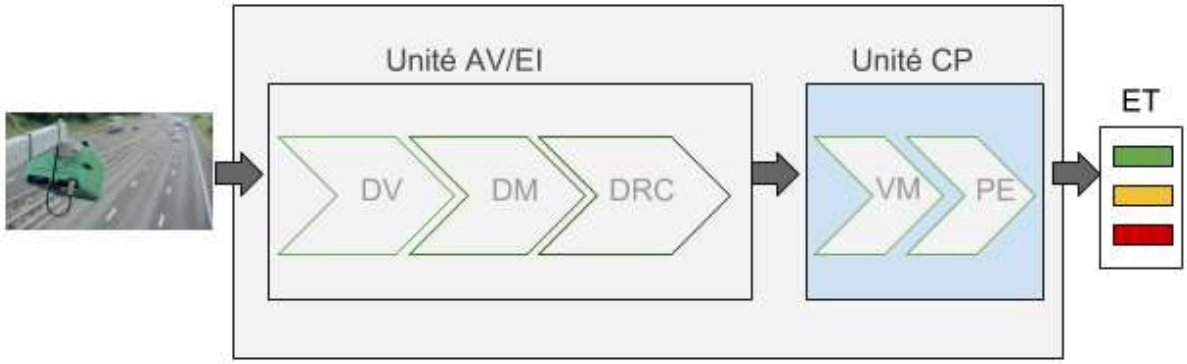


Figure 27 Architecture générale du système

AV/EI : unité d'analyse vidéo et extraction d'information

CP : unité de calcul et de prédiction de l'état du trafic

DV : Module de détection de véhicule

DM : Module de détection de plaque d'immatriculation

DRC : Module de détection et d'identification de chiffres

VM : Module calcul vitesse moyenne

PE : Module prédiction de l'état du trafic

ET : L'état du trafic routier

Le code couleur indique l'état du trafic :

- **Vert** : le trafic est normal.
- **Orange** : le trafic est moyennement fluide.
- **Rouge** : le trafic est ralenti. Plus le rouge est foncé, plus le trafic est ralenti.

III-2-1 Unité d'analyse des vidéos et d'extraction d'information

Cette unité analyse les flux vidéo envoyés par les caméras de surveillances pour détecter les véhicules passants puis les identifier de façon unique tout en enregistrant le temps de leurs passages. Ceci se fait en trois tâches principales effectuées par ces trois modules:

1- Module de détection de véhicule

2- Module de détection de plaque d'immatriculation

3- Module de détection et d'identification de chiffres

La sortie de chaque module donne l'entrée du module qui le suit , le premier module a comme sortie l'image de véhicule détecté qui est passé au deuxième module pour localiser la plaque d'immatriculation qui a son tour produit l'entrée de troisième module qui détecte et reconnaît le caractère dans la plaque d'immatriculation.

Le temps de passage des véhicules est enregistré lors de la détection du véhicule par le premier module pour être utilisé dans le calcul de leurs vitesses.

III-2-1-1 Module détection de véhicule

Ce premier module de la première unité a pour rôle principal la détection des véhicules sur les routes puis les enregistrer sous forme d'une image sur un répertoire commun avec le deuxième module tout en enregistrant le temps de passage de chaque véhicule.

La détection des véhicules se fait en utilisant un réseau neuronal convolutif composé de 44 couches de convolution et 23 couches de pooling et une dernière couche entièrement connectée.

	Type	Filters	Size	Output
	Convolutional	32	3 × 3	256 × 256
	Convolutional	64	3 × 3 / 2	128 × 128
1×	Convolutional	32	1 × 1	
	Convolutional	64	3 × 3	
	Residual			128 × 128
	Convolutional	128	3 × 3 / 2	64 × 64
2×	Convolutional	64	1 × 1	
	Convolutional	128	3 × 3	
	Residual			64 × 64
	Convolutional	256	3 × 3 / 2	32 × 32
8×	Convolutional	128	1 × 1	
	Convolutional	256	3 × 3	
	Residual			32 × 32
	Convolutional	512	3 × 3 / 2	16 × 16
8×	Convolutional	256	1 × 1	
	Convolutional	512	3 × 3	
	Residual			16 × 16
	Convolutional	1024	3 × 3 / 2	8 × 8
4×	Convolutional	512	1 × 1	
	Convolutional	1024	3 × 3	
	Residual			8 × 8
	Avgpool		Global	
	Connected		1000	
	Softmax			

Table 1. **Darknet-53.**

Figure 28 Architecture du réseau de neurones

Une vidéo est composée de 25 frames ou plus, et le système par défaut traite et extrait les objets détectés dans toute la frame, ce qui veut dire qu'une voiture peut être détectée plusieurs fois. Pour diminuer la redondance d'information et du traitement il existe deux solutions :

- La première est de diminuer le nombre de frames à traitées par seconde
- La seconde est de traiter juste un segment pour chaque frame

La deuxième solution est meilleure que la 1ere, car dans la 1ere solution il se peut que le véhicule soit détecté juste après qu'il entre dans le champ de vision de la caméra, ce qui veut dire qu'il sera loin et la probabilité que le module de reconnaissance des chiffres les détecte dans la plaque d'immatriculation sera faible, c'est pour ça qu'on a choisi la 2éme solution. Dans ce cas toutes les voitures détectées qui ont la ligne du bas de l'encadrement dans la bande de traitement dans l'image seront enregistrés.

III-2-1-1-1 Calcul de la bande à traiter

Pour calculer la partie de l'image à traiter, il faut tout d'abord supposer la vitesse max (Vmax) des voitures sur la route où les caméras sont placées afin de calculer la distance qu'elles parcourent chaque seconde et ainsi la distance parcourue après chaque frame, pour faire en sorte de ne pas laisser aucune voiture passer par la caméra sans la capturer et de minimiser au maximum les redondances.

Calcul de la distance max parcourue par la voiture chaque seconde

Vmax (m) ---------> 3600 s

D1 ---------> 1 s

D1 (m/s) = (Vmax x 1) / 3600

Où :

D1: La distance maximale parcourue par la voiture chaque seconde

Calcul de la distance max parcourue par la voiture chaque frame

D1 ---------> NbreF

D2 ---------> 1 frame

D2 (m/frame) = (NbreF x D1)

Où:

D2 : La distance maximale parcourue par la voiture chaque frame

NbreF : Le nombre de frames enregistrées par la caméra chaque seconde

Calcul du nombre de pixels de la bande

Height (pixel) ---------> Y (m)

X ---------> D2

X= (Height x D2) / D2

Où:

Height : La dimension de la caméra en hauteur

Y: Le champe de vision de la caméra

X: La dimension de la bande à traiter en hauteur

Exemple

Pour calculer la partie de l'image à traiter on suppose dans cet exemple que la vitesse maximale égale à 150km/h et on suppose aussi que la caméra :

- Est une caméra HD avec une résolution 1080x720
- Enregistre 25 frames/seconde
- A un champ de vision de 10 mètres

Calcul de la distance max parcourue par la voiture chaque seconde

150 x 1000 m ---------> 3600 s

X ---------> 1 s

X= (150 x 1000 x 1) / 3600 = 41.66 m/s

Calcul de la distance max parcourue par la voiture chaque frame

41.66 m/s ---------> 25 frames

Y ---------> 1 frame

Y = (41.66 x 1)/25 = 1.66 mètres/frame

Calcul du nombre de pixels de la bande

720 pixels ---------> 10 mètres

Z ---------> 1.66 mètres

Z= (720 x 1.66)/10 = 119,52 pixels

Dans cet exemple la bande aura une résolution 1080x120

Figure 29 Exemple du segment à traiter

III-2-1-2 Module détection de plaque d'immatriculation

Après avoir détecté et enregistrer les véhicules, il va falloir les identifier d'une façon unique, et pour cela on a besoin d'un caractéristique unique pour chacun d'eux, la plaque d'immatriculation est l'objet idéal pour cela. Sur ce deuxième module on a comme entrée l'image représentant le véhicule et donne en sortie la plaque d'immatriculation, cela se fait en trois étape qui sont la détection, la coupure, l'enregistrement sur disque.

Un deuxième réseau neuronal convolutif est utilisé pour la détection de la plaque d'immatriculation des véhicules déjà enregistrée par le 1er module, il est composé de 44 couches de convolution et 23 couches de pooling et une dernière couche entièrement connectée, et donne comme sortie une image de la plaque d'immatriculation.

La figure 31 représente le module de détection de la plaque d'immatriculation

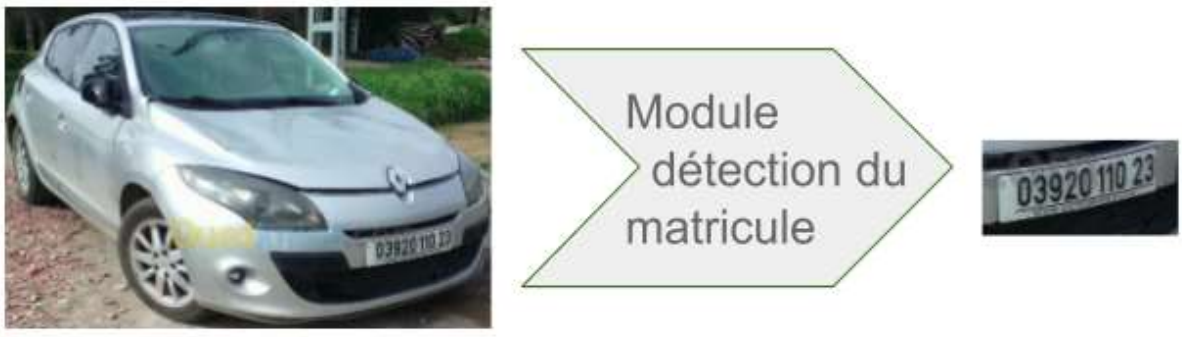


Figure 30 Module de détection de plaque d'immatriculation

III-2-1-3 Module détection et reconnaissance de caractère

Ce dernier module est responsable de la détection et la reconnaissance des chiffres sur l'image déjà enregistrées par le 2^{éme} module. Les caractères détectés sont désordonnés lors de la détection, un traitement d'ordonnancement est nécessaire avant de les enregistrer sur un fichier.

Un troisième et dernier réseau neuronal convolutif est utilisé pour la détection et la reconnaissance de caractère sur la plaque d'immatriculation, il est composé de 44 couches de convolution et 23 couches de pooling et une dernière couche entièrement connectée qui donne en sortie une chaîne des chiffres présent dans la plaque d'immatriculation et le temps de détection du véhicule sur un fichier texte qui peut être utilisé par l'unité de calcul et de prédiction.

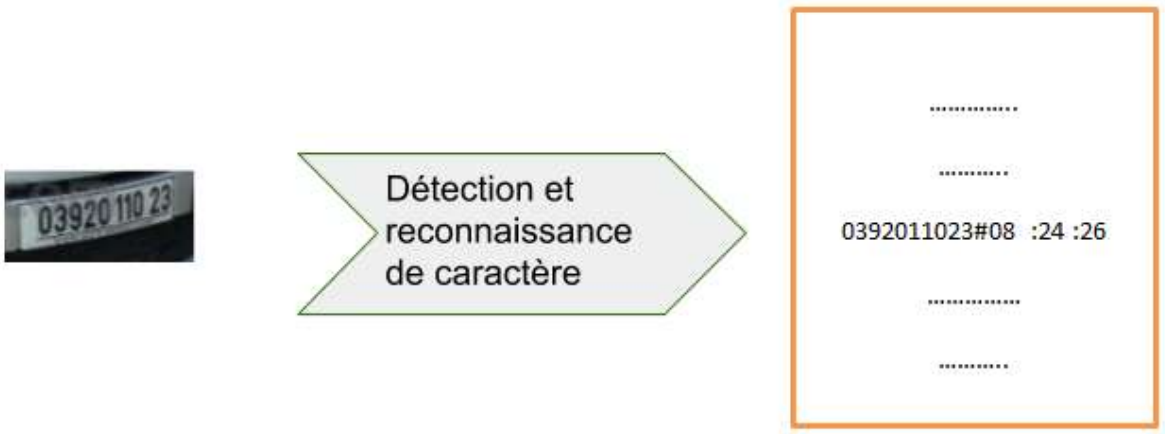


Figure 31 Module de détection et de reconnaissance de caractères

III-2-2 Unité de calcul et prédiction de l'état de trafic routier

Cette deuxième unité a deux tâche principale qui sont le calcul de la vitesse moyenne de déplacement des véhicules et de prédire l'état de la circulation selon la vitesse calculée et les répartit en trois classe : trafic normal, trafic moyennement fluide et trafic ralenti.

Cette unité reçoit comme entrée le fichier texte enregistré par le module de détection de plaque d'immatriculation et donne comme sortie une estimation de l'état de trafic routier. Elle est composée de deux modules, à savoir :

1- Module de calcul de vitesse moyenne

2- Module de prédiction de l'état du trafic routier

III-2-2-1 Module calcule de vitesse moyenne de déplacement

Ce module est responsable de calculer la vitesse moyenne de déplacement d'un véhicule entre deux endroit où il a été détecté, et pour cela il a besoin de quatre informations qui sont l'identifiant de véhicule et l'heur de passage d'un véhicule sur un point A qui est considéré comme le point de départ et un point B qui est considéré comme le point d'arrivée, et la distance entre ces deux points. L'heure de passage et l'identifiant de véhicule sont des informations issues de la sortie de la première unité, la distance entre les différent points sont

des données stable et disponible dans le système.

Le calcul effectué par ce module est sous la forme suivant :

Temps de la traversé / la distance entre le deux endroit = la vitesse moyenne

III-2-2-2 Module prédiction de l'état du trafic

Ce dernier module qui est le module de prédiction de l'état du trafic classifie les vitesses en trois classes : normal, moyennement fluide et ralenti, et cela en fonction de la vitesse de circulation des véhicule sur la route.

0 > vitesse moyenne > 20 ======> ralenti

20 > vitesse moyenne > 60 =======> moyennement fluide

Vitesse moyenne > 60 ========> normal

Conclusion

Ce chapitre à était consacré à la conception d'un système d'estimation d'état du trafic routier en temps réel basé sur les réseaux de neurones convolutifs. On a présenté les différents modules qui forment le système global ainsi que l'entrée/sortie de chaque module et la méthode utilisée pour la diminution de la redondance d'informations et du traitement

Chapitre IV

Réalisation

IV-Réalisation

IV-1 Introduction

Un bon système de détection et d'identification de véhicule doit être performant, rapide et robuste. Pour cela nous avons choisi d'utiliser les réseaux de neurones convolutifs qui sont de nos jours les modèles les plus performants pour la classification d'images.

Dans ce chapitre, on va définir les outils logiciels utilisés sur les différentes plateformes: *Grid*, *Cloud* …etc.

IV-2 Outils logiciels utilisés

Pour la réalisation de notre système d'estimation de l'état du trafic routier, on a choisie l'utilisation du framework YOLO, qui est un système de détection d'objet en temps réel open source et qui permet de traiter un nombre d'image suffisant pour la réalisation d'un système de détection d'objet en temps réel.

L'un des avantages de ce framework est la possibilité de l'installer avec différente bibliothèques (CUDA, CUDNN, OPENCV …) qui permettent l'utilisation d'une ou plusieurs cartes graphiques pour l'accélération du temps d'entraînement et de détection d'objets et ainsi augmenter le nombre d'images traitées par seconde.

IV-2-1 Yolo

YOLO est un système de détection d'objet en temps réel basé sur les réseaux de neurones convolutifs, capable de traiter 45 images par second. [15]

Les systèmes de détection antérieurs appliquent les classificateurs sur plusieurs emplacements dans la même image et les régions à haut score de l'image sont considérées comme des détections d'objets. Par contre, YOLO utilise une approche totalement différente, il applique un seul réseau de neurones sur l'image complète, ce réseau divise l'image en une grille de S x S cellules et prédit les probabilités pour chaque cellules en même temps d'où le nom YOLO

(You Only Look Once), et ce qui rend ce système très rapide par rapport aux autres systèmes. [16]

Fonctionnement

- YOLO divise l'image d'entrée en une grille de S x S cellules, chaque cellule de la grille est responsable de prévoir si un objet est présent dans cette cellule ou non

- Un seul réseau de neurones est exécuté sur l'image d'entrée

- Chaque cellule prévoit un encadrement de l'objet qu'elle a détecté (la position : x,y,w,h et la probabilité de cette prédiction) + une condition de la classe d'objet (exemple : s'il existe un objet dans cette cellule ça serait une voiture) [15]

- Combinaison des encadrements d'objets et les conditions de classes d'objets (classification)

- Afficher les encadrements des objets avec la classe à laquelle chaque objet appartient qui dépasse un seuil de probabilité donné (l'utilisateur peut changer ce seuil).

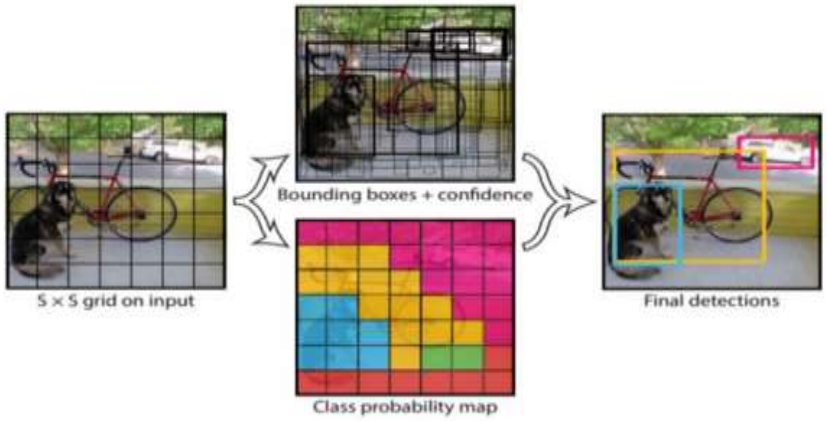


Figure 32 Etapes de la classification par YOLO

IV-2-2 OpenCV

Opencv (Opencv Source Computer Vision Library) est une bibliothèque graphique libre écrite en C et C++, initialement développée par Intel, spécialisée dans le traitement d'images en temps réel. Elle dispose d'une interface en python et Java. Elle prend en charge Windows, Linux, Mac OS, iOS et Android.

Un des buts d'OpenCV est d'aider les gens à construire rapidement des applications sophistiquées de vision à l'aide d'infrastructure simple de vision par ordinateur. La bibliothèque d'OpenCV contient plus de 500 fonctions. La figure 34 montre le logo d'OpenCV. [17]

Figure 33 Logo d'OpenCV

Dans notre projet openCV a été utilisée pour le traitement des vidéos.

IV-2-3 Cuda

CUDA (initialement l'acronyme de Compute Unified Device Architecture) est une technologie de GPGPU (General-Purpose Computing on Graphics Processing Units), c'est-à-dire utilisant un processeur graphique (GPU) pour exécuter des calculs généraux à la place du processeur (CPU). En effet, ces processeurs comportent couramment de l'ordre d'un millier de circuits de calcul fonctionnant typiquement à 1 GHz, ce qui représente un potentiel très supérieur à un processeur central à 4 GHz. [18]

Dans notre projet CUDA est utilisé pour accélérer le traitement d'images.

IV-2-4 Cudnn

Cudnn (Cuda Deep Neural Network), est une bibliothèque utilisée pour l'accélération de GPU pour les réseaux de neurones profonds, limité aux GPU fournie par NVIDIA. [19]

Dans notre projet CUDA est utilisé pour accélérer l'entraînement

IV-3 Plateformes utilisés

Les plateformes utilisées pour la réalisation de notre modèle sont :

IV-3-1 Ordinateur de bureau

Un ordinateur de bureau peut être utilisé pour l'apprentissage de modèle, mais il est extrêmement lent, il peut prendre jusqu'à 25 jour ou plus, pour atteindre une précision élevée pour la détection d'un seul objet.

IV-3-1-1 Configuration matériel

La configuration du matériel utilisé dans notre implémentation est :

- CPU i5 : 2.40 GHZ
- RAM de taille 4 GO
- Disque dur de taille 500 GO
- Système d'exploitation Linux Ubuntu version 16.04

IV-3-1-2 Déploiement du modèle

Six étapes sont nécessaires pour le déploiement du modèle :

1- Installation des outils logiciels

Opencv et yolo sont deux briques de base :

L'installation de Yolo se fait à l'aide d'un fichier Makefile est cela en exécutant la commande : $ make

L'installation de la bibliothèque Opencv se fait à partir de code source en deux étapes :

1-1 Téléchargement d'une version stable de la bibliothèque

Un lien vers le code source est disponible sur le site officiel :

https://github.com/opencv/opencv/archive/3.4.1.zip

1-2 Générer OpenCV à partir de code source à l'aide de l'outil CMake

Premièrement on crée un répertoire 'build' pour séparer les fichiers généré des fichiers source avec la commande : $ mkdir Build

Dans ce répertoire on exécute les commandes suivantes :

$ cmake -D CMAKE_BUILD_TYPE=Release -D CMAKE_INSTALL_PREFIX=/usr/local ..

$ make -j7

$ sudo make install

2-Preparation de dataset

La préparation de dataset se fait en trois étapes :

2-1 La collecte des données

Pour que le modèle atteint une précision élevée, plus de 300 image pour chaque classe (voiture, plaque d'immatriculation ...) est nécessaire.

Toutes les images ont été collectées de site d'ouedkniss.

2-2 L'annotation des images

L'annotation est une tâche primordiale à réaliser, ça consiste à encadrer les objets qui nous intéressent dans chaque image pour que le réseau de neurones puisse savoir leurs positions exactes et extraire les caractéristiques de ces zones uniquement. Un outil logiciel nous permet de faire l'annotation d'une manière facile : Yolo mark

La figure suivante montre un exemple pratique de notre annotation pour la détection des chiffres :

Figure 34 Exemple d'annotation

2-3 Séparation des données

La dernière étape est de séparer les images en 2 répertoires : 70% des images pour l'entraînement et 30% pour le test.

3- Modification des fichiers de configuration

Avant de lancer l'entraînement il faut tout d'abord modifier le fichier de configuration, pour le faire, les étapes suivantes sont nécessaires :

- Créer un fichier de configuration *yolo-obj.cfg* (contient le même contenu que le fichier *yolov3.conf*)
- Modifier le paramètre *batch* : de *batch=1* à *batch=64* (le batch c'est le nombre d'images utiliser lors de l'entraînement pour chaque itération)
- Modifier le paramètre *subdivision* : de *subdivision=1* à *subdivision=8* (divisé le traitement des 68 images en 8 tâches, ce qui veut dire que chaque itération est divisée en 8 tâches). Dans le cas où le GPU a assez de mémoire, le nombre de subdivisions peut être augmenté pour accélérer l'entraînement.
- Changer le nombre de classes dans les lignes 610, 696, 783 de *classes=80* à *classes=1* (pour le réseau qui détecte juste la voiture et le réseau qui détecte que la plaque d'immatriculation) et à *classes=10* (pour le réseau qui détecte le chiffres)
- Changer le nombre de filtre dans les lignes 603, 689, 776 de *filters=225* à *filters=18* (pour le réseau qui détecte juste la voiture et le réseau qui détecte que la plaque d'immatriculation) et à *filters=45* (pour le réseau qui détecte le chiffres). le nombre de

filtres = (le nombre de classes + 5) x 3

- Création d'un nouveau fichier *obj.names* dans le répertoire *data*, qui contient les noms de classes (chaque classe dans une ligne)
- Création d'un nouveau fichier *obj.data* qui contient :

-classes =<nombre de classes>

-train = data/train.txt

-test = data/test.txt

-names= data/obj.names

-backup = backup/

4 Lancé l'entraînement et suivre l'évolution

Après avoir préparé l'environnement, il ne reste qu'à lancer l'entraînement, pour cela on exécute yolo en lui donnons comme arguments le fichier de configuration et un weights initial avec la commande :

./darknet detector train data/obj.data yolo-obj.cfg darknet.conv.74

On peut lancer l'entraînement sur plusieurs GPUs

./darknet detector train data/obj.data yolo-obj.cfg darknet.conv.74 -gpus 0,1,2

Après chaque 100 itérations un nouveau poids sera enregistré dans le répertoire *backup/*

Quand arrêter l'entraînement ?

Habituellement 2 000 itérations pour chaque classe (objet) sont suffisantes, mais pour plus de précision on se base sur les informations affichées sur la console pendant l'entraînement.

Pendant l'entraînement on voit des lignes s'afficher comme suit :

Region Avg IOU: 0.798363, Class: 0.893232, Obj: 0.700808, No Obj: 0.004567, Avg Recall: 1.000000, count: 8 Region Avg IOU: 0.800677, Class: 0.892181, Obj: 0.701590, No Obj: 0.004574, Avg Recall: 1.000000, count: 8

9002: 0.211667, **0.060730 avg**, 0.001000 rate, 3.868000 seconds, 576128 images Loaded: 0.000000 seconds

- 9002: le nombre d'itérations
- 0.60730 avg: la perte moyenne (plus la valeur est petite, plus le poids est précis)

L'entraînement peut être arrêté lorsque la perte moyenne égale à 0.xxxxx avg, et que cette valeur ne diminue plus à de nombreuses itérations.

5- Tester le résultat

Après la phase d'entraînement on test le résultat pour voire le taux de reconnaissance correcte.

6- Déployer le modèle

A la fin on déploie le modèle en utilisant le « weights » résultant de l'entraînement

IV-3-2 Cloud

Le Cloud Computing a révolutionné toute l'industrie en démocratisant l'utilisation des Data Centers et en transformant le workflow de nombreuses entreprises. Nos ressources les plus importantes sont désormais hébergées dans le Cloud, en toute sécurité et avec une gestion simplifiée. Pour exploiter au mieux ces données dématérialisées, il nous faut une solution de calcul à hautes performances.

Les processeurs graphiques (GPU) constituent une puissante plateforme de calcul capable de transformer les ressources Big Data en renseignements exploitables. Ce sont les moteurs d'une nouvelle ère de l'intelligence artificielle. Nous pouvons désormais accéder dans le Cloud à de puissants services Deep Learning et IA accélérés par GPU - tout en profitant de capacités sans précédent.

Le Cloud Computing sur GPU est disponible à la demande sur les principales plateformes Cloud de l'industrie.

Figure 35 Logo des différentes entreprises qui fournissent du cloud computing

Floydhub est la plateforme que nous avons utilisée pour l'apprentissage de notre premier modèle. floydhub est une Paas (Platform-as-a-Service) pour l'entraînement et le déploiement d'un modèle sur le cloud.

IV-3-2-1 Configuration matériel

Floydhub propose deux pack, le premier est standard avec :

Tesla K80 · 12 GB Memory

61 GB RAM · 100 GB SSD

Et le deuxième est haute performance avec :

Tesla V100 · 16 GB Memory

61 GB RAM · 100 GB SSD

Pour notre utilisation nous avons choisi le pack standard, qui est compatible avec nos outils logiciel.

IV-3-2-2 Déploiement du modèle

Après avoir créé un compte et s'y connecter sur Floydhub il faut :

1- Préparer l'environnement

Floydhub nous fournis un outil *floyd-cli* qui nous permet de nous connecter à notre projet sur la plateforme depuis notre ordinateur local.

2- Uploader la dataset

Après la préparation de dataset on utilise floyd-cli pour l'envoyer au cloud avec la commande: $ floyd data upload.

3- Installer les outils logiciels

Cuda et Cudnn sont déjà pré installé sur la plateforme, il nous reste qu'à installer yolo et

opencv, et pour cela en refait les mêmes étapes expliquées précédemment pour un ordinateur de bureau.

4- Lancer l'entraînement

Depuis notre station de travail locale on lance l'entraînement avec la commande:

$ floyd run --gpu --data ichafa/datasets/train/1:/train

--data ichafa/datasets/test/1:/test

--data ichafa/datasets/weights/1:/weights "make && ./darknet detector train plat.data cfg/plat.cfg /weights/darknet53.conv.74"

--gpu : pour indiquer qu'on veut utiliser une carte graphique type Tesla K80 GPU.

--data : pour lier les dataset.

5- Télécharger le résultat et le tester

Le résultat de l'entraînement est un fichier « *.weights* » qu'on peut le récupérer sur la plateforme floydhub puis le testé sur notre ordinateur local pour voir le taux de précision.

IV-3-3 Grille informatique

Une grille informatique (en anglais, *grid*) est une infrastructure virtuelle constituée d'un ensemble de ressources informatiques potentiellement partagées, distribuées, hétérogènes, délocalisées et autonomes.

Notre travaille a était réalisé au niveau de Grid'5000, c'est un instrument scientifique flexible et de grande taille pour le support de la démarche expérimentale dans tous les domaines de l'informatique, en particulier pour les systèmes parallèles et distribués tels que les clouds, le HPC et les systèmes pour le big data.

Figure 36 Grid'5000

IV-3-3-1Configuration matériel

Grid'5000 fournit un accès à une grande quantité de ressources: 1000 nœuds, 8000 cœurs, regroupés en clusters homogènes et dotés de différentes technologies: Ethernet 10G, Infiniband, GPU, Xeon PHI.

Pour notre cas d'utilisation un seul nœud doté d'une carte graphique puissante était suffisant. Souvent nous avons travaillé avec une carte graphique de type GeForce GTX 1080 Ti.

IV-3-3-2 Déploiement du modèle

Les démarches que nous avons suivies après que nous nous sommes connectés à la grille sont :

1- Réserver un nœud

La première étape est de réserver un nœud, et cela se fait à l'aide de OAR qui est un outil de gestion de tâche et de ressource (anglais: scheduler)

La commande est :

$ oarsub -p "cluster='chifflet' and gpu!='NO'" -l nodes=1,walltime=2 -I -t deploy

2- Déployer une image sur le nœud

Après avoir réservé un nœud on a besoin de déployer une image, *Kadeploy* est un outil logiciel disponible sur la grille, qui nous permet de réaliser cette tâche.

Nous avons choisis de travailler avec ubuntu 16.04, la commande pour déployer l'image de cette distribution est :

$ kadeploy3 -f $OAR_NODE_FILE -e ubuntu1604-x64-base -k

3- Préparer l'environnement de travail

La préparation de l'environnement de travail se fait par l'installation et la configuration des différents outils logiciels sur le l'image déployée : yolo, opencv, cuda, cudnn. Et uploader les datasets nécessaires pour l'entraînement de réseau.

4- Lancer l'entraînement

L'étape qui suit la préparation de l'environnement et l'exécution de programme, on lance l'entraînement en donnant le fichier de configuration et un fichier weight initial.

5- Tester le résultat

L'étape finale est de tester le résultat pour voir le taux de précision, on relance l'entraînement si le taux de précision est faible.

Conclusion

Dans ce chapitre on a présenté les différentes bibliothèques et plateformes utilisées dans ce projet, les étapes à suivre pour l'entrainement d'un réseau de neurones en utilisant le framewok YOLO ainsi que les différentes étapes suivies pour la réalisation de notre système.

Chapitre V

Résultats

V Résultat

V-1 Introduction

Dans ce chapitre nous allons présenter les aspects performances de notre système sous différentes plateforme ainsi que les résultats obtenus en termes de précision et de temps de traitement pour chaque module de notre système d'estimation d'état de trafic routier

V-2 Aspects performances

Le tableau suivant montre les différents aspects de performance

	Temps pour 100 itérations	Temps d'entraînement (1seule classe -2000 itérations-)	Temps de détection
Ordinateur bureau	12 heures	25 jours	25 secondes
Cloud	12 minutes	10 heures	0.0029 second
Grille	9 minutes	8 heures	0.0022 second

Tableau 1 Comparaison entre les différentes plateformes utilisées

La vitesse d'entraînement et de détection d'objet dépend fortement des performances de la machine utilisée. Pour une machine sans GPU comme le montre le tableau 1 le temps d'entraînement peut prendre plus de 25 jours pour une seule classe et c'est très long et ce n'est surtout pas pratique quand le nombre de classes est élevé.

Le tableau 1 montre aussi que le type du GPU utilisée joue un rôle important pour réduction de ce temps. Pour une GPU Geforce 1080Ti, on gagne 3 minutes pour chaque 100 itérations, cette différence peut être remarquable quand le nombre de classe est élevé.

V-3 Comparaison entre YOLO et les autres systèmes

	Précision (base de données VOC -20classes-)	Nombre de frames/seconde	Vitesse de traitement d'une image
R-CNN	66.0	0.05	20s
Fast-R-CNN	70.0	0.5	2s
Faster-R-CNN	73.0	7	142ms
YOLO	69.0	45	22ms

Tableau 2 Comparaison entre YOLO et d'autres systèmes de détection d'objets [15]

La conception d'un système de détection d'objets en temps réel en utilisant les flux vidéo requiert une précision et une vitesse de traitement d'image élevée. Le tableau 2 fait la comparaison de ces 2 facteurs entre YOLO et différents systèmes de détection d'objet.

On remarque que la précision des différents systèmes est proche les unes des autres sauf pour le R-CNN qui a donné une faible précision avec un taux de 66%.

Bien que Fast-R-CNN et Faster-R-CNN ont donnés un taux de précision un peu plus élevé que YOLO, mais ils restent trop loin pour être utilisés dans des systèmes de détection d'objet en temps réel, car ils prennent beaucoup de temps pour traiter les images, le meilleure entre eux peut traiter 7 frames par seconde qui n'est pas pratique car une vidéo contient au moins 25 frames par seconde. Par contre YOLO peut traiter jusqu'à 45 frames par seconde ce qui fait de lui le meilleure système pour la détection d'objets en temps réel même avec un taux de précision de 69%, car la précision peut être augmentés en utilisant une plus grande base de donnée d'entraînement et en ajoutant le nombre d'itérations.

V-4 Précision des réseaux

V-4-1 Réseau de détection de la plaque d'immatriculation

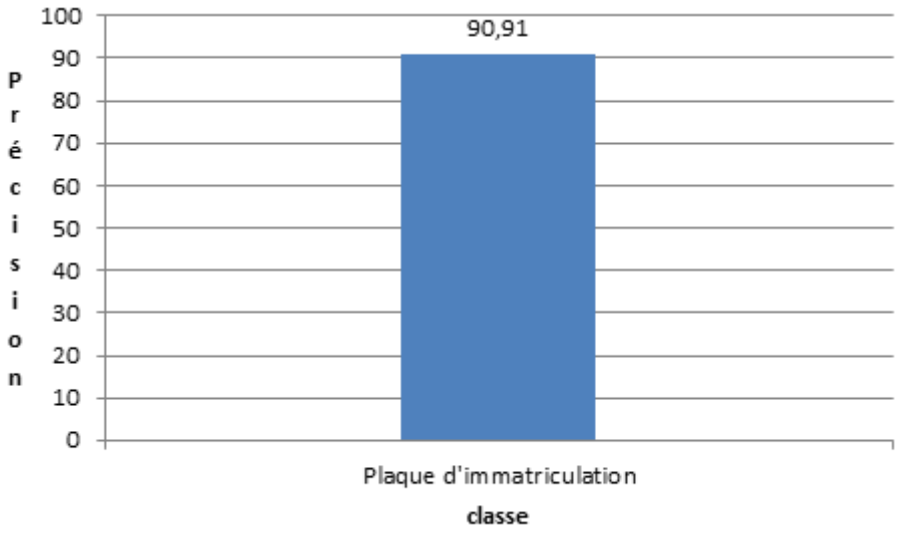


Figure 37 Précision du réseau de détection de la plaque d'immatriculation

V-4-2 Réseau de détection des chiffres

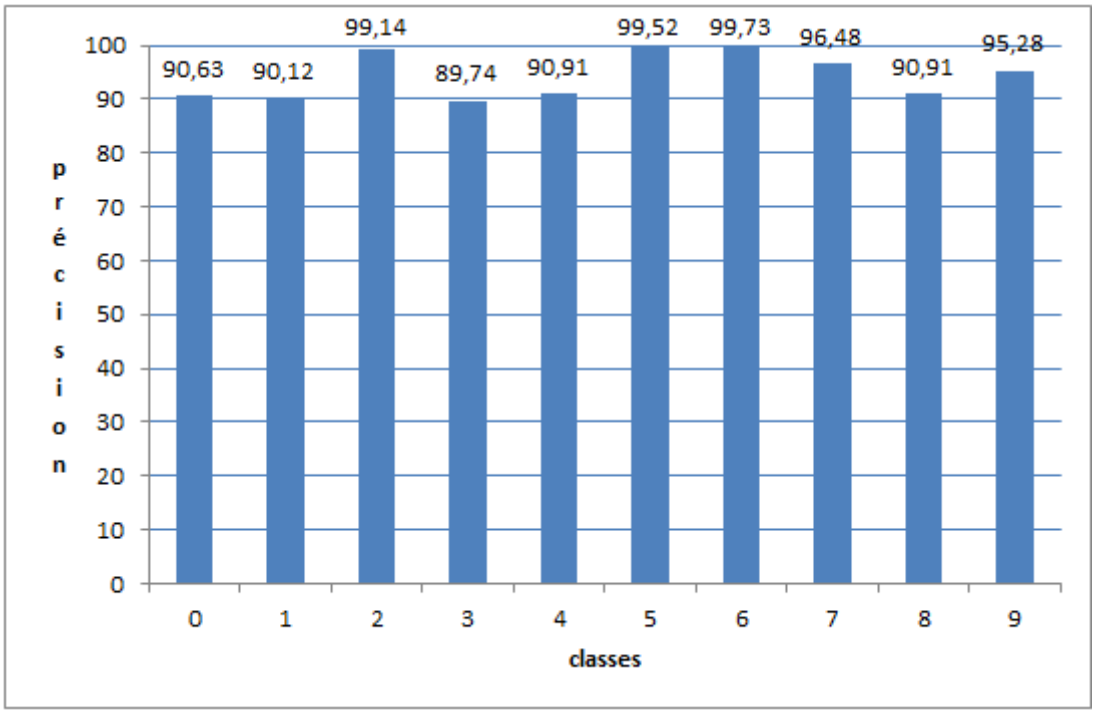


Figure38 Précision du réseau de reconnaissance des chiffres

Notre réseau de reconnaissance des chiffres a donné un taux de précision moyen élevé de 94.25%, qui est assez suffisant pour être utiliser dans notre système d'estimation de trafic routier.

V-5 Comparaison entre enregistrement de tous les objets détectés dans l'image et les objets détectés dans un segment

Pour une vidéo de 15 secondes, 30 frames/seconde où 21 voitures passent on a eu les résultats suivants :

	Nombre d'images enregistrées	Taux de redondances
Traitement de toute l'image	1032	4814,28 %
Traitement d'un segment	47	123,8 %

Tableau 3 Comparaison entre traitement de toute l'image et le traitement d'un segment

Le tableau 5 montre le taux de redondances entre le traitement de toute l'image et le traitement d'un seul segment de l'image. On remarque que même que le traitement d'un segment de chaque frame a donné une redondance de plus de 100% mais ça a pu diminuer le taux de redondance d'information de presque 39 fois que le traitement de toute l'image.

Conclusion générale

L'objectif initial de ce travail était de mettre en œuvre un système d'estimation d'état du trafic routier en temps réel. On a donc commencé par discuter les différentes solutions qui peuvent être utilisées GPS, radio-identification, caméra de surveillance. Ensuite on a parlé des notions fondamentales des réseaux de neurones convolutifs et on a présenté les différentes types de couches utilisées pour la classification d'image : couche convolutionnelle, pooling, couche de correction et la couche entièrement connectée.

On a rencontré quelques problèmes dans la phase de réalisation. L'utilisation d'un CPU a fait que le temps d'entrainement du réseau et de détection d'objet était très coûteux. Afin de régler ce problème on devait déployer notre système sur des machines performantes utilisant des cartes GPU avec des architectures modernes au lieu d'utiliser qu'une machine avec un CPU.

On a fait la comparaison entre les différents systèmes de détection d'objet basé sur les réseaux de neurones, et on a choisi l'utilisation de framework YOLO, pour le nombre important d'image qu'il est capable de traiter en une seconde et pour sa grande précision, ce qui était convenable pour la conception de notre système d'estimation d'état du trafic routier en temps réel.

Notre système à présent permet l'estimation d'état du trafic routier en un seul segment mais le travail reste en cours pour le déployer sur une grande échelle.

Le système peut être amélioré au futur pour non seulement estimer l'état du trafic routier mais aussi à des fins de sécurité comme la localisation et le suivi des véhicules volés.

Bibliographie

[1] http://www.ons.dz/IMG/pdf/NAT31-12-2016.pdf (site officiel de l'ONS)

[2] https://videosurveillance.ooreka.fr/comprendre/videosurveillance-urbaine le 28/5/2018

[3] Naciri H., Chaoui N, Conception et Réalisation d'un système automatique d'identification des empreintes digitales, Mémoire de PFE, Université de Tlemcen, 2003

[4] Mohamed Kamel BEN KADDOUR, Segmentation d'image par Coopération région - contours, Mémoire de PFE, Université de Tlemcen, 2016

[5] Christophe Alleau, " Transmettre et stocker de l'information", Poitier 2011

[6] Sukanya CM, Roopa Gukul, Vince Paul. A Survey On Object Recognition Methods, Computer science engeneering departement, calcul university, Kerala, India, janvier 2016.

[7]https://elearn.univ-ouargla.dz/2013-2014/courses/TI/document/Cours/Image_Chap01.pdf le 30/5/2018

[8] Thi-Lan le, "Indexation et recherche de vidéo pour la vidéosurveillance", université de NICE-SOPHIA ANTIPOLIS, 3 février 2009.

[9] https://openclassrooms.com/courses/classez-et-segmentez-des-donnees-visuelles/decouvrez-la-notion-de-features-dans-une-image le 31/5/2018

[10] Elgammal, A.Duraiswami, R.Harwood, D.Anddavis,2002.Background and forground modeling using nonparametric kernel density estimation dor visual surveillance. Proceedings of IEEE 90,7,11511163

[11] Mr Mokri Mohammed Zakaria, Classification des images avec les réseaux de neurones convolutionnels, Mémoire de PFE, Université de Tlemcen, 2016

[12] Hitesh A Patel, Darshak G Thakore, Moving Object Tracking Using Kalman Filter, International Journal Of Computer Science and Mobile Computing, April 2013

[13]https://medium.com/@CharlesCrouspeyre/comment-les-r%C3%A9seaux-de-neurones-%C3%A0-convolution-fonctionnent-b288519dbcf8 le 25/05/2018

[14]https://towardsdatascience.com/applied-deep-learning-part-4-convolutional-neural-networks-584bc134c1e2 le 25/05/2018

[15] https://www.youtube.com/watch?v=NM6lrxy0bxs (Présentation de YOLO par l'un de ses fondateurs)

[16] https://pjreddie.com/darknet/yolo/ (site officiel de YOLO) le 28/05/2018

[17] https://opencv.org/ (site officiel d'OpenCV) le 28/05/2018

[18] https://developer.nvidia.com/cuda-zone (site officiel de NVIDIA) le 28/05/2018

[19] https://developer.nvidia.com/cudnn (site officiel de NVIDIA) le 28/05/2018