

Département Génie électrique et informatique
Filière génie informatique

**Proposition d'un système du smart
parking basé sur l'intelligence artificiel
et le computer vision**

Réalisé par :
El Hassak Mohamed

Sous la direction de :
Pr. Boulaalam Abdelhak

TABLE DES MATIERES

| | |
|--|----|
| Introduction | 4 |
| Chap1 : Les fonctionnalités du smart parking | 6 |
| 1. Connaître la disponibilité des places | 6 |
| 2. Reconnaissance automatique des plaques | 7 |
| 3. Reconnaissance faciale..... | 8 |
| 4. La collecte et le traitement de données..... | 9 |
| Chap2 : Smart Parking : Prototype..... | 11 |
| 1. Le fonctionnement global du Smart Parking..... | 11 |
| 2. Les méthodologies..... | 13 |
| a. Computer vision: | 13 |
| b. Détection des véhicules avec YOLO..... | 14 |
| c. Reconnaissance optique de caractères OCR..... | 17 |
| d. Système de messagerie distribué avec KAFKA | 20 |
| Chap3 : Mise en œuvre une implémentation du prototype..... | 23 |
| 1. Environnement de travail | 23 |
| 2. Implémentation..... | 25 |
| Conclusion..... | 28 |
| Bibliographique..... | 29 |

LISTE DES FIGURES

| | |
|---|----|
| FIGURE 1 : DETECTION DES PLACES LIBRES..... | 6 |
| FIGURE 2: RECONNAISSANCE AUTOMATISEE DES PLAQUES..... | 7 |
| FIGURE 3 : RECONNAISSANCE FACIALE..... | 8 |
| FIGURE 4 : LA COLLECTE, LE TRAITEMENT ET L'AFFICHAGE DES DONNEES..... | 9 |
| FIGURE 5 : LE FLOW DE MODELE | 11 |
| FIGURE 6 : AFFICHAGE L'ETAT DU PARKING..... | 12 |
| FIGURE 7 : BOUNDING BOXES AUTOUR UNE VOITURE | 14 |
| FIGURE 8 : EXEMPLE DETECTION PAR R-CNN..... | 15 |
| FIGURE 9 : EXEMPLE D'OBJETS DETECTES PAR YOLO..... | 15 |
| FIGURE 10 : ENCODING IMAGE YOLO3 | 16 |
| FIGURE 11 : DIVISER IMAGE EN CELLULES-YOLO..... | 16 |
| FIGURE 12 : BOUNDING BOXES AUTOUR IMAGE | 16 |
| FIGURE 13 : AVANT ET APRES LA SUPPRESSION NON-MAX..... | 17 |
| FIGURE 14 : LES ETAPES DU SYSTEME OCR | 18 |
| FIGURE 15 : DETECTION DU TEXTE-OCR..... | 18 |
| FIGURE 16 : SEGMENTATION DU TEXTE -OCR | 19 |
| FIGURE 17 : CLASSIFICATION DU CARACTERES-OCR..... | 19 |
| FIGURE 18 : ARCHITECTURE D'UN SYSTÈME KAFKA..... | 21 |
| FIGURE 19 : PUBLIER DES MESSAGES VERS UN TOPIC | 22 |
| FIGURE 20 : TEST SYSTÈME RAP-ENTREE..... | 25 |
| FIGURE 21 : AFFICHAGE DES DONNÉES DANS LE CLIENT..... | 26 |
| FIGURE 22 : TEST SYSTEME RAP-SORTIE | 26 |
| FIGURE 23 : HISTORIQUE DE PARKING | 27 |

Introduction

Etudiant de 5eme année génie informatique à l'ENSA de Fès, on nous a proposé un projet qui nous permette de mettre en pratique nos connaissances et nos compétences professionnelles à travers l'étude du thème du smart parking comme étant une solution aux plus gros problèmes de conduite zones urbaines.

L'organisation de ce projet était évidente en tant que problème émergent nécessitant des solutions innovantes ; Le stationnement n'était pas un problème au début, car il y avait suffisamment de places de stationnement dans la rue, mais le taux de voitures a continué d'augmenter jusqu'à ce que le problème de stationnement commence à se poser. Comme récemment, de nombreuses plaintes ont été formulées aux universités par des étudiants concernant ce problème, en raison du manque d'espace, de l'incapacité d'augmenter le nombre de places et la disponibilité.

Alors comment peut-on résoudre ce problème ? Et pourquoi le smart parking est-il la solution la plus appropriée ?

Le stationnement intelligent a été développé pour fournir et trouver un parking vide et réduire les problèmes de stationnement illégal et son effet négatif. Depuis plusieurs années, les villes constatent que les conducteurs confrontent de réels problèmes avec l'inadaptation des délais du stationnement. Cela entraîne une multitude de conséquences, notamment des embouteillages, des accidents de la circulation fréquents et une perte de temps massive. Un problème de pollution se pose également, les automobilistes qui tournent dans la ville pour chercher une place de parking polluent la ville sans s'en rendre compte.

On a également confronté à d'autres problèmes, comme trouver une place dans un parking et du problème d'identification des voitures, comme l'espace est vaste ainsi que la présence de plusieurs véhicules qui peuvent être identiques.

Pour répondre à ces nombreux problèmes, on aura besoin d'un système dans lequel les automobilistes pourraient utiliser des appareils intelligents ou leurs smartphones pour trouver de

l'espace disponible, s'identifier comme abonnés ou se renseigner sur d'autres véhicules qui sont déjà stationnés. Alors nous allons d'abord expliquer en détail.

dans le premier chapitre les différentes fonctionnalités du smart parking, c'est-à-dire qu'est ce qui doit être mis en place pour répondre à ces problèmes. Ensuite, le second chapitre sera consacrée aux technologies utilisées par le smart parking pour rendre le stationnement des automobilistes plus simple ainsi ce qui concerne l'étude de son fonctionnement ; Nous aborderons également ce problème dans notre école .Enfin le troisième chapitre comprend la partie pratique de notre travail qui sera consacrée à une petite implémentation afin de mettre en œuvre notre prototype.

Chap1 : Les fonctionnalités du smart parking

Dans cette partie, nous allons développer les fonctionnalités du smart parking. Quelles en sont les utilités et ses caractérisations.

1. Connaître la disponibilité des places

La connaissance des disponibilités des places est essentielle dans le Smart Parking. En effet, l'utilisateur, pour trouver une place facilement, doit savoir où sont les places libres. Quand un automobiliste arrive pour chercher une place de parking, habituellement il tourne en rond jusqu'à trouver une place de libre, ce qui entraîne des embouteillages, l'encombrement des rues et la pollution de la ville. Si l'utilisateur connaît la localisation des places libres, il peut en réserver une et aller directement à cette dernière. Cela évite d'errer de ne pas trouver une place et permet un gain de temps considérable. La solution est par exemple de poser des caméras autour le parking indique si la place est prise ou non. Les technologies utilisées seront précisées dans le chapitre 2.

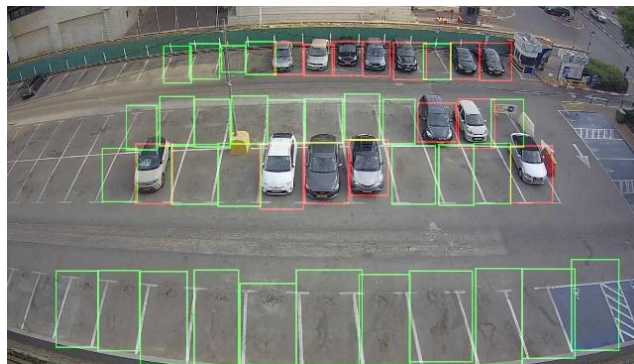


Figure 1 : détection des places libres

Les automobilistes doivent être informés des disponibilités des places. Pour cela, il existe des sites qui indiquent les stationnements libres (pays étranger). Les utilisateurs savent ainsi où se trouve les emplacements disponibles. Ils peuvent également réserver leurs places pour éviter que d'autres automobilistes leur prennent. On trouve également des applications pour les téléphones portables, ce qui est plus facile d'utilisation pour les automobilistes quand ils sont en déplacement dans une

autre ville par exemple. Une fois arrivés dans un lieu, ils peuvent directement chercher sur leurs mobiles les disponibilités des places afin de trouver facilement un stationnement.

Par exemple, l'application **Apila**¹, disponible sur iPhone, Android et Smartphone, permet de trouver une place. Cette application, qui est communautaire, indique les places de stationnement libres dans la rue. Elle n'utilise aucune technologie comportant des capteurs ou autre mais simplement ses utilisateurs ! En effet, quand quelqu'un libère une place, il l'indique sur l'application, ce qui met à jour l'emplacement des places disponibles. Peu coûteuse, cette solution n'est néanmoins.

2. Reconnaissance automatique des plaques

La reconnaissance automatisée des plaques est une technologie destinée à identifier **les plaques d'immatriculation**² de véhicules via l'utilisation de techniques de **reconnaissance optique de caractères**³.



Figure 2: reconnaissance automatisée des plaques

¹ Apila est une application actuellement proposée en version bêta 2.0 sur l'App Store et bêta 1.0 sur Google Play.

² est une plaque portant une combinaison unique de chiffres ou de lettres (pour une zone géographique donnée), destinée à identifier facilement un véhicule terrestre

³ En anglais *optical character recognition* (OCR), désigne les procédés informatiques pour la traduction d'images de textes imprimés ou dactylographiés en fichiers de texte

La lecture automatique de plaques minéralogiques peut être utilisée aussi bien pour fichier les images capturées par les caméras que le texte de la plaque d'immatriculation, avec la possibilité de stocker une photographie du conducteur.

3. Reconnaissance faciale

Un système de reconnaissance faciale est une application logicielle visant à reconnaître une personne grâce à son visage de manière automatique. C'est un domaine de **la vision par ordinateur**⁴ consistant à reconnaître automatiquement une personne à partir d'une image de son visage. Il s'agit d'un sujet particulièrement étudié en vision par ordinateur, avec de très nombreuses publications et brevets, et des conférences spécialisées.

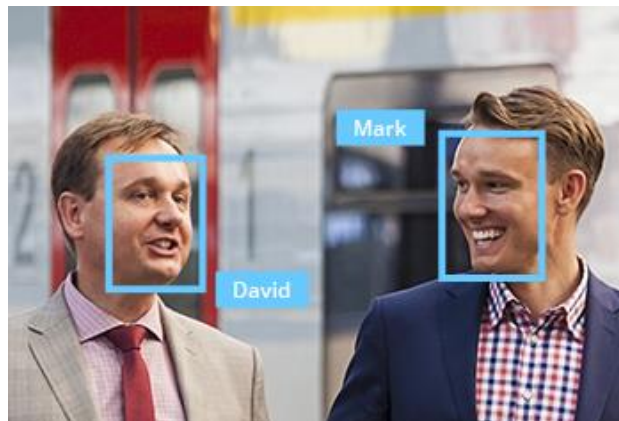


Figure 3 : reconnaissance faciale

La reconnaissance faciale a plusieurs applications en vidéo surveillance, biométrie, robotique, indexation d'images et de vidéos, recherche d'images par le contenu, etc. Ces systèmes sont généralement utilisés à des fins de sécurité pour déverrouiller ordinateur/mobile/barrière, mais aussi en domotique. Ils sont appréciés car considérés comme peu invasifs, en comparaison avec les

⁴ La vision par ordinateur (*computer vision* en anglais) est une branche de l'intelligence artificielle dont le principal but est de permettre à une machine d'analyser, traiter et comprendre une ou plusieurs images prises par un système d'acquisition (ex : caméras)

autres systèmes biométriques (empreintes digitales, reconnaissance de l'iris...). Le fonctionnement de ces systèmes se base sur une ou plusieurs caméras pour reconnaître l'utilisateur.

Par exemple, Dans notre système ou les utilisateurs sont les étudiants, les profs et les employés de l'établissement chacune a sa propre espace de stationnement, pour assurer cette opération et pour garantir la sécurité de l'espace, il est nécessaire de savoir l'identité des personnes entrants.

4. La collecte et le traitement de données

Nous considérons les systèmes de stationnement intelligents réels où les données d'occupation des parkings sont collectées à partir des caméras installées au terrain et envoyées aux serveurs backend pour un traitement et une utilisation ultérieure pour les applications. Notre objectif est de rendre ces données utiles aux utilisateurs finaux, tels que les gestionnaires de parking. À cette fin, nous concoctons et validons des algorithmes. Nous analysons d'abord les statistiques des données de stationnement réelles, en affichants l'état de parking de stationnement. Nous considérons ensuite un algorithme de machine Learning Ceux-ci sont réglés selon une approche supervisée ou non supervisée utilisant notre données collectées, Notre approche serve aussi à faire des prédictions sur les données à fin d'améliorer encore le travail de système.

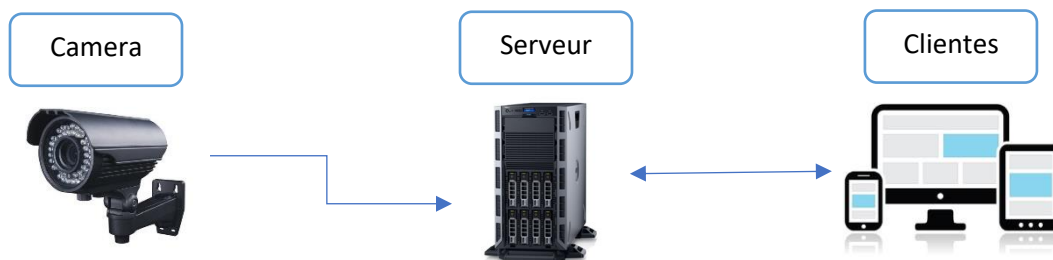


Figure 4 : la collecte, le traitement et l'affichage des données

La partie logicielle du système tourne sur un matériel (ordinateur) et peut être liée à d'autres applications ou bases de données. Elle commence par utiliser une série de techniques de manipulation d'image pour détecter, normaliser et agrandir l'image de la plaque d'immatriculation,

et enfin la reconnaissance optique de caractères pour extraire les caractères alphanumériques de la plaque. Typiquement, des ordinateurs sont rassemblés en batterie dans une **ferme de serveurs**⁵ (*server farm*) pour traiter les grandes charges de travail, dans ce cas on peut penser à Un système de traitement de données tous distribués sur plusieurs machines.

⁵ De cluster, de groupement de serveurs ou de ferme de calcul (*computer cluster* en anglais) pour désigner des techniques consistant à regrouper plusieurs ordinateurs indépendants appelés nœuds (*node* en anglais)

Chap2 : Smart Parking : Prototype

1. Le fonctionnement global du Smart Parking

On a ici le principe général du fonctionnement d'un smart parking, proposé par nous, un modèle qui respect les fonctionnalités les plus importants dans un parking (voir chapitre 1), dans lequel on essayer d'implémenter un ensemble de techniques de la vision par ordinateur, à savoir la détection, la localisation des objets dans une image, avec un traitement qui en temps réel afin de mettre le parking intelligent et automatisé.

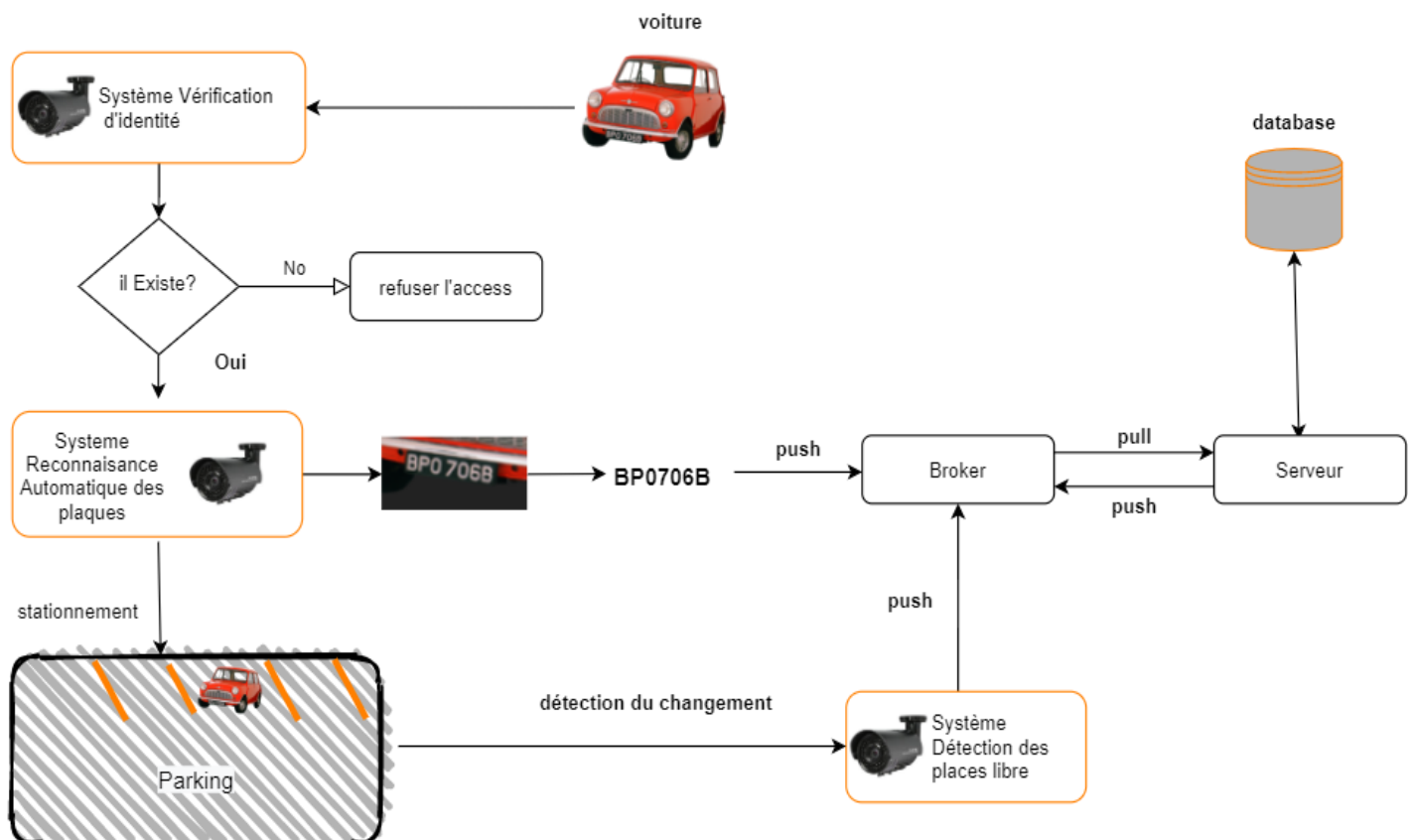


Figure 5 : le flow de modelé

Ce procédé est assez simple et peut se résumer en quatre étapes principales :

- ✚ Une voiture abonnée, équipée d'une plaque d'immatriculation, vient se garer sur une place de stationnement où on y trouve un barrage avec une caméra qui analyse en temps réel le visage de chauffeur et vérifie si est un membre de l'établissement, si oui la barrière se déverrouille.
- ✚ La deuxième étape c'est la reconnaissance automatique de plaque, en détectant le numéro d'immatriculation puis l'envoyer vers un serveur pour le traitement.
- ✚ Quand la voiture rentre à l'espace parking, des caméras sont installés pour détecter la disponibilité des places et transmet les informations en temps réel au serveur via des brokers.
- ✚ Le serveur général renvoie les informations vers les abonnés qui sont transmises sur un téléphone smart phone. Ainsi ces derniers sont informés de la disponibilité des places de stationnement.

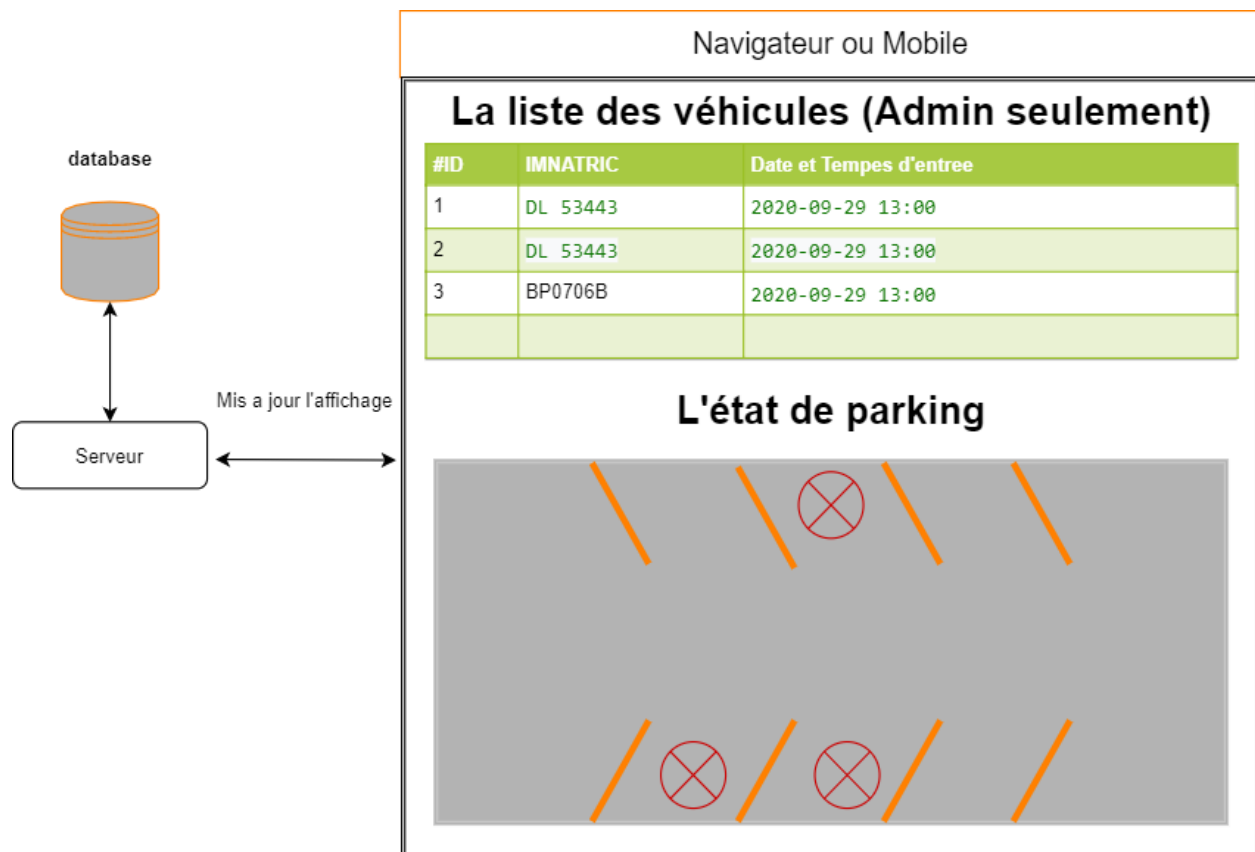


Figure 6 : affichage l'état du parking

Il faut savoir pour mettre en pratique ce modèle on va utiliser les différents aspect et techniques de **Deep Learning**⁶, et plus précisément le computer vision.

2. Les méthodologies

a. Computer vision:

Désigne les différentes techniques permettant aux ordinateurs de voir et de comprendre le contenu d'images. Il s'agit d'une sous-catégorie d'intelligence artificielle et de Machine Learning.

Le domaine de la Computer Vision regroupe de multiples techniques issues de divers champs d'ingénierie ou d'informatique. De manière générale, les différentes méthodes ont pour but de reproduire la vision humaine. Pour comprendre le contenu des images, les machines doivent être capables d'en extraire une description : un objet, une description, un modèle 3D...

Certains systèmes de vision par ordinateur peuvent aussi nécessiter un traitement de l'image, à savoir une simplification ou une augmentation de son contenu. En guise d'exemple, on peut citer la normalisation des propriétés photométriques de l'image, le cropping de ses contours ou la suppression du « bruit » tels que les artefacts numériques induits par un faible éclairage.

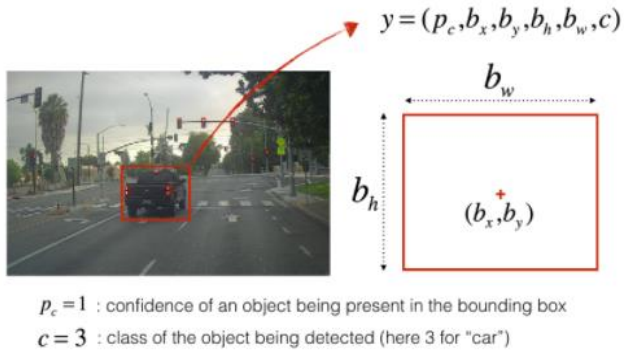
Nous examinerons les problèmes de vision par ordinateur suivants où l'apprentissage en profondeur a été utilisé :

Classification d'image, classification d'image avec localisation, détection d'objets, segmentation d'objets, Transfert de style d'image, colorisation d'image, reconstruction d'image, super-résolution d'image, synthèse d'image et d'autres problèmes.

⁶ Le deep learning (ou apprentissage profond) est un ensemble de méthodes d'*apprentissage automatique* conçues sur la base de *réseaux de neurones profonds*, visant à mimer la « profondeur » des couches d'un cerveau.

Dans notre cas, notre système doit faire plusieurs tâches de la vision par ordinateur, à savoir la détection des objets (voitures, plaques), la localisation c'est-à-dire dessiner les boîtes englobantes⁷ autour des voitures détectées, l'identification des caractères de la plaque (OCR), etc....

Si vous souhaitez que le détecteur d'objets reconnaisse 80 classes, vous pouvez représenter



l'étiquette de classe c soit sous la forme d'un entier de 1 à 80, soit sous la forme d'un vecteur à 80 dimensions (avec 80 nombres) dont un composant est 1 et le reste est égal à 0.

Figure 7 : bounding boxes autour une voiture

b. Détection des véhicules avec YOLO

Notre objectif est de détecter une voiture dans une image ou dans un stream vidéo en temps réel, puis designer bounding boxes autour la voiture.

Pour cela on va utiliser des algorithmes de deep Learning, algorithmes basés sur la classification. Ils sont mis en œuvre en deux étapes. Tout d'abord, ils sélectionnent les régions d'intérêt dans une image. Deuxièmement, ils classent ces régions à l'aide de réseaux de neurones convolutifs.

Cette solution peut être lente car nous devons exécuter des prédictions pour chaque région sélectionnée. Un exemple largement connu de ce type d'algorithme est le réseau neuronal

⁷ Les boîtes englobantes (*Boundig boxes* en anglais) sont des boîtes imaginaires qui se trouvent autour d'objets dont la collision est vérifiée, comme les piétons sur ou à proximité de la route, d'autres véhicules et des panneaux

convolutif basé sur la région (R-CNN) et ses cousins Fast-RCNN, Faster-RCNN et le dernier ajout à la famille : Mask-RCNN. Un autre exemple est RetinaNet.

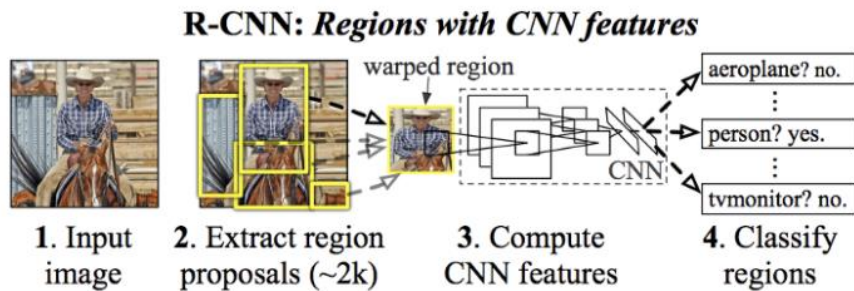


Figure 8 : exemple détection par R-CNN

Pour résoudre ce problème des algorithmes basés sur la régression sont définie - au lieu de sélectionner des parties intéressantes d'une image, ils prédisent des classes et des bounding boxes pour l'image entière en une seule exécution de l'algorithme. Les deux exemples les plus connus de ce groupe sont les algorithmes de la famille YOLO (You Only Look Once) et SSD (Single Shot Multibox Detector). Ils sont couramment utilisés pour la détection d'objets en temps réel car, en général, ils échangent un peu de précision contre de grandes améliorations de la vitesse.

YOLO : "You Only Look Once" (YOLO) est un algorithme populaire car il atteint une grande

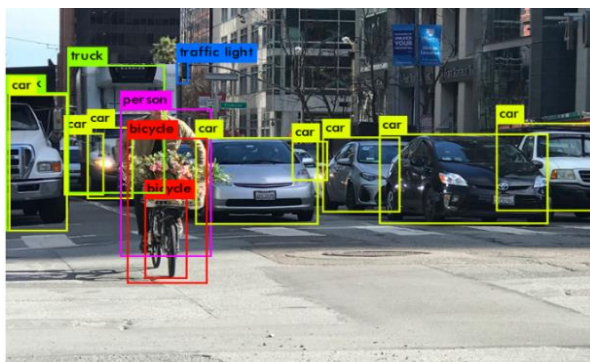


Figure 9 : exemple d'objets détectés par YOLO

précision tout en étant capable de fonctionner en temps réel. Cet algorithme "ne regarde qu'une seule fois" l'image en ce sens qu'il ne nécessite qu'un seul passage de propagation vers l'avant à travers le réseau pour faire des prédictions. Après la suppression non-max, il génère ensuite les objets reconnus avec les bounding boxes.

Comme nous l'avons mentionné ci-dessus, lorsque nous travaillons avec l'algorithme YOLO, nous ne recherchons pas de régions intéressantes dans notre image qui pourraient potentiellement contenir un objet.

Au lieu de cela, nous divisons notre image en cellules, généralement en utilisant une grille 19×19 (YOLO3). Chaque cellule est responsable de la prédiction de 5 boîtes englobantes (au cas où il y aurait plus d'un objet dans cette cellule). Par conséquent, nous arrivons à un grand nombre 1805 de bounding boxes pour une image.

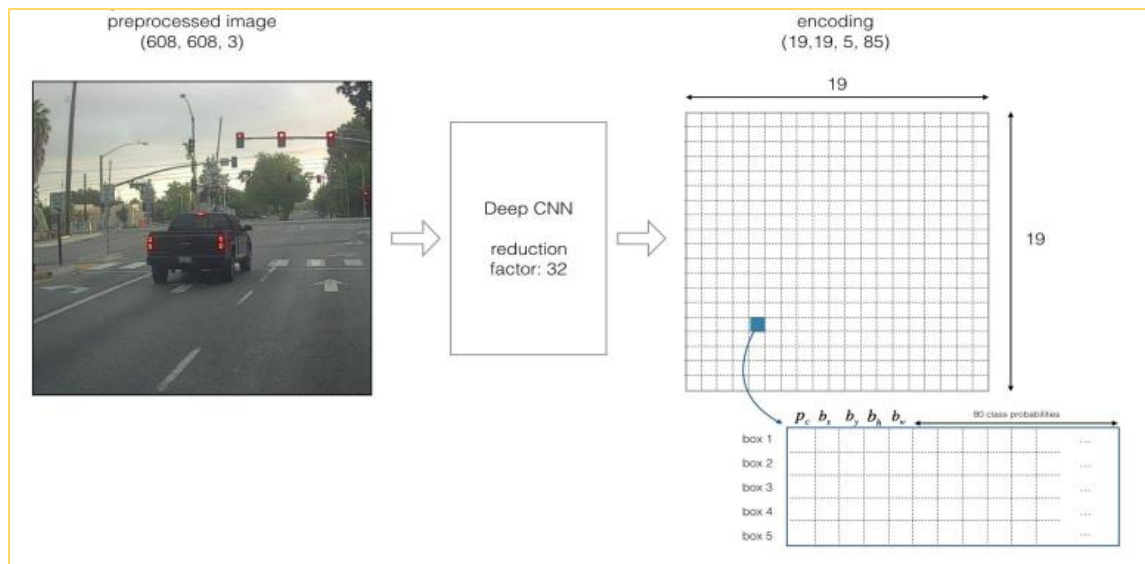


Figure 10 : encoding image yolo3

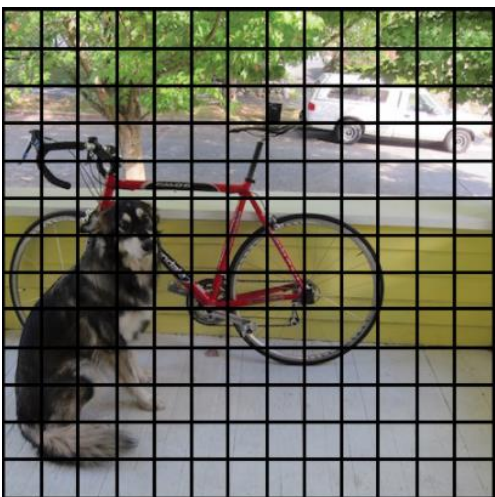


Figure 11 : diviser image en cellules-YOLO



Figure 12 : bounding boxes autour image

La plupart de ces cellules et cadres de délimitation ne contiennent pas d'objet. Par conséquent, nous prédisons la valeur p_c , qui sert à supprimer les boîtes avec une faible probabilité d'objet et les bounding boxes avec la zone partagée la plus élevée dans un processus appelé **suppression non-max**.⁸



Figure 13 : Avant et après la suppression non-max

c. Reconnaissance optique de caractères OCR

La reconnaissance optique de caractères, ou OCR, définit le processus de conversion mécanique ou électronique d'images numérisées d'un texte manuscrit, tapé ou imprimé en texte codé par machine.

Pensez à tout type de numéro de série ou de code composé de chiffres et de lettres que vous avez besoin de numériser. En utilisant l'OCR, vous pouvez transformer ces codes en sortie numérique. La technologie utilise différentes techniques. De manière très simplifiée, l'image prise sera prétraitée et les caractères extraits et reconnus.

Les étapes d'un système OCR :

⁸ Suppression non-max est une technique dans la vision par ordinateur pour éliminer les bounding boxes avec une faible probabilité

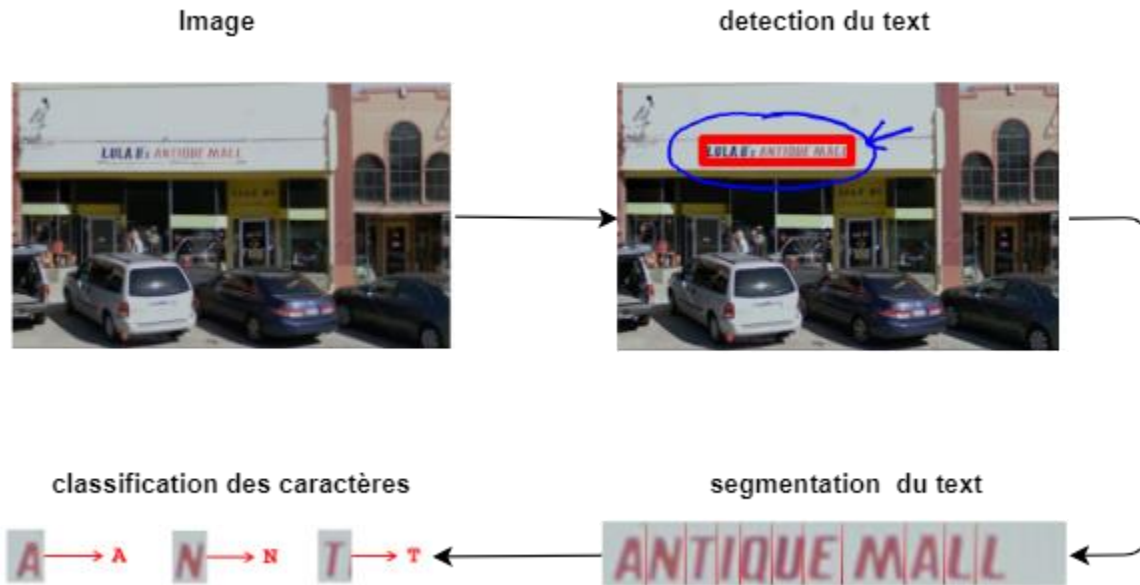


Figure 14 : les étapes du système OCR

Afin de réaliser une OCR photo, voici ce que nous pouvons faire. Nous pouvons d'abord parcourir l'image et trouver les régions où il y a du texte et de l'image. Voici donc un exemple de texte et d'image que le système OCR photo peut trouver.



Figure 15 : détection du texte-OCR

Deuxièmement, étant donné le rectangle autour de cette zone de texte, nous pouvons alors effectuer une segmentation des caractères, où nous pourrions prendre cette zone de texte qui dit "Antique Mall" et essayer de la segmenter dans les emplacements des caractères individuels.



Figure 16 : segmentation du texte -OCR

Et enfin, après avoir segmenté en caractères individuels, nous pouvons ensuite lancer un feu croisé, qui regarde les images des caractères visuels, et essaie de comprendre que le premier caractère est un **A**, le deuxième est un **N**, le troisième est un **T**, et ainsi de suite, de sorte qu'en faisant tout cela, vous pouvez alors comprendre le contenu de cette phrase et de même pour certains des autres mots qui apparaissent dans cette image.



Figure 17 : classification du caractères-OCR

En plus, il existe des systèmes photo OCR qui font des choses encore plus complexes, comme un peu de correction orthographique à la fin. Donc, si, par exemple, votre système de segmentation et de classification de caractères vous indique qu'il voit le mot **ne1 t oyage**. Ensuite, vous savez, une sorte de système de correction orthographique pourrait vous dire que c'est probablement le mot « **nettoyage** », et votre algorithme de classification de caractères vient de confondre le 1 avec un t. Mais aux fins de ce que nous voulons faire ici, ignorons cette dernière étape et concentrons-nous simplement sur le système qui effectue ces trois étapes de détection de texte, de segmentation de caractères et de classification de caractères.

En plus de code d'immatriculation, d'autres informations comme la date le temps, sont envoyés vers le serveur, pour cela on peut penser à utiliser un système de messagerie distribué.

d. Système de messagerie distribué avec KAFKA

Apache Kafka est une plate-forme de diffusion (streaming) distribuée développée en Scala et Java.

Une plate-forme de streaming possède trois fonctionnalités clés :

- ✚ Permettre aux applications clientes Kafka de publier et s'abonner à des flux d'enregistrements. Similaires à une file d'attente de messages ou à un système de messagerie d'entreprise comme les Brokers JMS (**ActiveMQ**⁹) ou AMQP (**RabbitMQ**¹⁰).
- ✚ Permet de stocker les flux d'enregistrements de manière durable et tolérante aux pannes.
- ✚ Permet de traiter les flux d'enregistrements au fur et à mesure qu'ils se produisent (Real Time Stream Processing)

Kafka a quatre API principales :

- ✚ **Producer API** : Permet à une application de publier un flux d'enregistrements vers un ou plusieurs Topics (Sujets) Kafka.
- ✚ **Consumer API** : Permet à une application de s'abonner à un ou plusieurs Topics et de traiter le flux d'enregistrements qui lui sont transmis.
- ✚ **Streams API** : Permet à une application d'agir en tant que processeur de flux, en
 - Consommant un flux d'entrée provenant d'un ou plusieurs Topics
 - Transformant efficacement les flux d'entrée en flux de sortie
 - Produisant un flux de sortie vers un ou plusieurs Topics en sortie.
- ✚ **Connector API** : Permet de créer et d'exécuter des producteurs ou des consommateurs réutilisables qui connectent des topics Kafka à des applications ou des systèmes de données existants. Par exemple, un connecteur vers une base de données relationnelle peut capturer chaque modification apportée à une table.

⁹ Apache ActiveMQ est un courtier de messages open source écrit en Java avec un client Java Message Service complet.

¹⁰ RabbitMQ est un logiciel d'agent de messages open source qui implémente le protocole Advanced Message Queuing, mais aussi avec des plugins Streaming Text Oriented Messaging Protocol et Message Queuing Telemetry Transport. Le serveur RabbitMQ est écrit dans le langage de programmation Erlang.

Kafka fournit des API clients pour différents langages : Java, C++, Node JS, .Net, PHP, Python, etc...

Les composants principaux du système basé sur KAFKA.

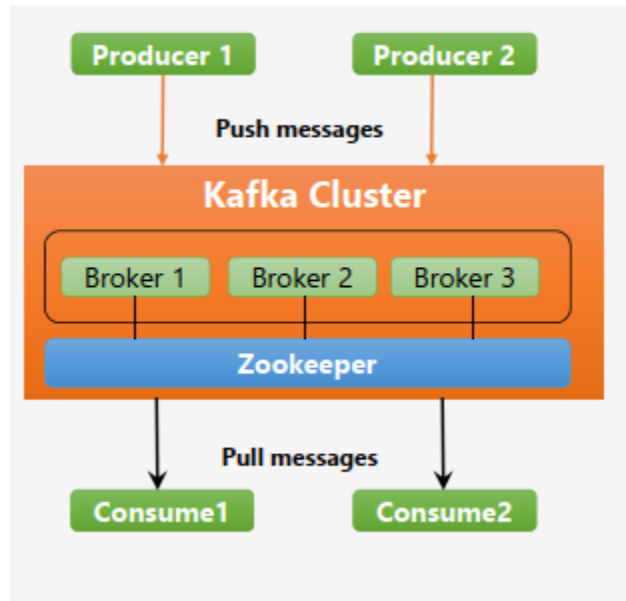


Figure 18 : architecture d'un système KAFKA

Kafka Brokers :

- ✚ Pour maintenir l'équilibre de la charge, le cluster Kafka est généralement composé de plusieurs Brokers dont un est élu comme Leader.
- ✚ Cependant, ils sont sans état, c'est pourquoi ils utilisent **ZooKeeper** pour conserver l'état du cluster.
- ✚ Une instance de Kafka Broker peut gérer des centaines de milliers de lectures et d'écritures par seconde.
- ✚ Chaque Broker peut gérer des To de messages.

Zookeeper:

- ✚ Kafka broker utilise ZooKeeper pour la gestion et la coordination.

- ✚ Il l'utilise également pour informer le producteur et le consommateur de la présence d'un nouveau broker dans le système Kafka ou d'une défaillance du broker dans le système Kafka.

Producers:

- ✚ Les producteurs de Kafka transmettent les données aux Brokers.
- ✚ Ces données sont à destination des Topics réparties en partitions dans les brokers du cluster
- ✚ Le producteur est responsable du choix de l'enregistrement à affecter à quelle partition du cluster.

Consumers:

- ✚ Consomme les messages à partir des Topics.
- ✚ En utilisant l'offset de la partition de Kafka Broker, le consommateur Kafka retient combien de messages ont été consommés parce que les Brokers Kafka sont sans état.
- ✚ Les utilisateurs peuvent rembobiner ou passer à n'importe quel point d'une partition.
- ✚ Chaque enregistrement publié dans un Topic est remis à une instance de consommateur au sein de chaque groupe de consommateurs abonné.

Dans notre cas, on prend l'exemple du système d'identification des plaques (Producer) qui nous donne une chaine de caractère contient le code d'immatriculation de la voiture, le but est d'envoyer ce code plus d'autres informations vers un broker qui a une sorte de boîte à lettre (Topic) dont laquelle les messages (enregistrements) sont publiés, puis le serveur qui joue dans ce cas le rôle d'un consommateur qui peut s'abonner au topic pour lire ces messages (Figure 17).

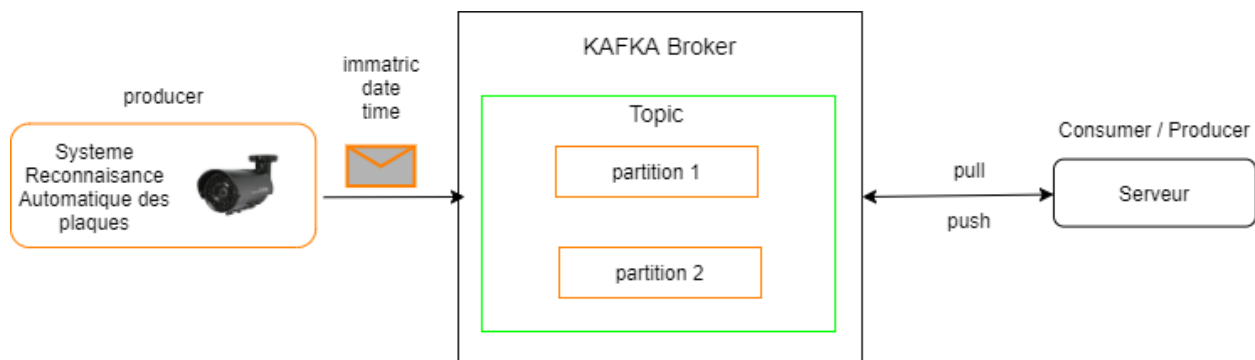


Figure 19 : publier des messages vers un topic

Chap3 : Mise en œuvre une implémentation du prototype

Ce chapitre sera consacré pour la présentation des outils et des technologies utilisés pour le développement des applications basées sur le AI et deep Learning, et nous finirons par quelques interfaces graphiques d'une partie de l'implémentation.

1. Environnement de travail

Pour développer des applications intelligentes de qualité, il est nécessaire d'utiliser des outils qui nous aider à construire facilement des modèles AI, cela élimine le besoin de comprendre tous les détails des algorithmes mathématiques derrière le ML / DL.



Python est le 4e langage le plus populaire selon l'index TIOBE et son usage est resté stable depuis une dizaine d'années. Python est un des langages principaux utilisés en *data analysis* (analyse de données) et en *machine learning* (apprentissage par la machine).

TensorFlow est un Framework d'apprentissage automatique (Machine Learning) développé par Google. Il est utilisable principalement en C++ et en Python, mais des portages sont disponibles pour la majorité des langages. Il nous permet de construire notre modèle de réseau de neurones facilement.





La bibliothèque Keras permet d'interagir avec les algorithmes de réseaux de neurones profonds et d'apprentissage automatique, qui facilite encore le travail avec des frameworks notamment Tensorflow, Theano, Microsoft Cognitive Toolkit ou PlaidML.

OpenCV (pour Open Computer Vision) est une bibliothèque graphique libre, initialement développée par Intel, spécialisée dans le traitement d'images en temps réel. La société de robotique Willow Garage et la société ItSeez se sont succédé au support de cette bibliothèque. Depuis 2016 et le rachat de ItSeez par Intel, le support est de nouveau assuré par Intel.

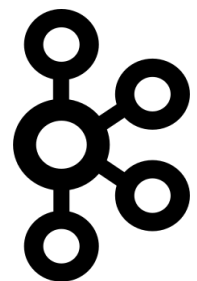


Tesseract est un logiciel de reconnaissance optique de caractères en plusieurs langues créé par Google.

Tesseract OCR

Pour la partie communication avec le serveur on utilise des middlewares comme kafka (voir chapitre 2).

Apache Kafka est une plate-forme de diffusion d'événements distribuée open source utilisée par des milliers d'entreprises pour des pipelines de données haute performance, des analyses de streaming, l'intégration de données et des applications critiques.



Pour mettre en œuvre un serveur qui va interagir avec la base de données et créer des services qui vont être consommés par les clients. C'est Spring Boot qui va se charger de faire ce travail.



Spring Boot is an open source Java-based framework used to create a micro Service. It is developed by Pivotal Team and is used to build stand-alone and production ready spring applications. This chapter will give you an introduction to Spring Boot and familiarizes you with its basic concepts

2. Implémentation

Dans cette partie nous prestons le résultat de l'implémentation du système de reconnaissance automatique des plaques qui joue un rôle principal dans le processus de notre prototype, à cause de manque de ressources de calcul de notre ordinateur qui utilise un CPU i5, on a utilisé des images au lieu d'un vidéo car ce dernier nécessite un très capacite de calcul, on parle ici d'un **GPU**¹¹.

Donc voilà le résultat de notre système via 2 étapes :

1. Détection la plaque dans l'image avec YOLO
2. Reconnaissance des caractères avec OCR



Figure 20 : test système RAP-entrée

¹¹ Une unité de traitement graphique qui effectue des calculs mathématiques rapides, principalement pour le rendu d'images.

L'affichage des données dans une application web réalisé par un Framework front end à savoir **Angular** ¹²:

liste des vehicules

| #NUM | IMMATRIC | Date Entree | Heure | Status |
|------|-------------|-------------|-------|------------|
| 1 | DL6C k 0805 | 03-10-2020 | 20:23 | Stationnee |

1

Figure 21 : affichage des données dans le client

On lance le même système pour la sortie pour détecter les véhicules sortants :

ALPN : Sortie du parking

Charger une image



plaque detecte:

DL6C K 0805

Numero d'immatric:

DL6C k 0805

Quiter

Figure 22 : test système RAP-sortie

¹² Angular est un Framework open source écrit en JavaScript qui permet la création d'applications Web et plus particulièrement de ce qu'on appelle des « Single Page Applications ».

Le véhicule s'ajoute aussi à l'historique de parking coloré par le rouge.

| liste des vehicules | | | | |
|---------------------|-------------|-------------|-------|------------|
| #NUM | IMMATRIC | Date Entree | Heure | Status |
| 1 | BALENO | 03-10-2020 | 20:41 | Stationnee |
| | DL6C k 0805 | 03-10-2020 | 20:23 | Out |
| 1 | | | | |

Figure 23 : historique de parking

Conclusion

L'objectif de ce projet est de proposer un exemple de parking intelligent. Un système dans ce sens qui utilise les outils de l'intelligence artificielle. Avec ce système, les usagers trouvent le meilleur espace libre, ce qui permet d'économiser du temps et de diminuer la pollution. Le parking se remplit efficacement et l'espace peut être utilisé correctement. La circulation devient plus fluide car il y aura moins de voitures qui circulent à la recherche d'une place de stationnement ouverte. La création du stationnement intelligent continue de se développer avec la croissance de nombre de voiture. Ainsi que le développement des technologies de computer vision qui continue d'être au cœur du développement du stationnement intelligent, les communications sans fil, l'analyse de données et les algorithmes avancés. Dans notre étude future, nous examinerons les aspects de sécurité de notre système ainsi que la mise en œuvre du système proposé à grande échelle dans le monde réel.

Bibliographique

Sites web :

<https://arxiv.org/pdf/1506.02640>

<https://pjreddie.com/darknet/yolo/>

<https://kafka.apache.org/>

<https://towardsdatascience.com/yolo-you-only-look-once-real-time-object-detection-explained-492dc9230006>

Cours en ligne :

<https://www.coursera.org/learn/machine-learning>

<https://www.coursera.org/learn/convolutional-neural-networks?specialization=deep-learning>

Chaines YouTube :

Siraj Raval : https://www.youtube.com/watch?v=4eIBisqx9_g

mohamedYoussfi : <https://www.youtube.com/watch?v=2YJEWPD8yk&t=4s>