**Smart City : Accident Detection with IoT and Deep Learning**

1. **Généralité sur la détection d’accident utilisant des algorithmes d'intelligence Artificielle**
   1. **Statistiques sur les accidents de la route au maroc**
   2. **IA et accidents monitoring**

Les technologies de l’IA et de l’IoT ouvrent une nouvelle ère prometteuse des villes dites intelligentes. La vie de milliards de personnes se voit transformée grâce à cette avancée technologique que ce soit dans le domaine du transport, de la communication… . Un sujet qui préoccupe au plus au point les autorités et la population actuellement est la mise en place d’un système de détection d'accident intelligent. Ainsi, plusieurs approches sont utilisées pour la mise en place d’un tel système. La migration vers une détection plus intelligente grâce au développement du big data et aux algorithmes de l'intelligence artificielle permettra de réduire considérablement le nombre de décès par accident de circulation. Car dès la détection d’un accident la plateforme intelligente enverra un message aux autorités compétentes les plus proches pour qu’ils puissent intervenir à temps. La seule difficulté réside au niveau de la mise en place d’un tel système.

1. **Exigences de détection d’accident**

Une infrastructure de détection automatique d’accident nécessite plusieurs composantes:

* La plateforme IOT: son rôle est de collecter les données de circulation plus particulièrement le flux vidéo et les envoyer à la plateforme de smart.
* Le système intelligents : il a plusieurs rôles:
  + Détecter tous les véhicules dans le flux videos envoyé par la plateforme IoT
  + Classer ses véhicules en différentes catégories: bus, camion, piétons, motos …
  + Extraction des informations temporelles tels que les mouvements, la vitesse, la trajectoire, l’accélération afin d’effectuer des opérations de tracking et d’analyse de la trajectoire. En appliquant un certain algorithme sur ces opérations on peut faire de la détection d’accidents.
* L’ensemble de l’infrastructure doit opérer en temps réel.

Système fonctionnel en cas de ciel nuageux, pluie, crépuscule, la nuit …

1. **Etat de l’art algorithme de détection**

Nous verrons ici les différentes approches utilisées dans la littérature pour faire de la détection d’accident.

Xiaohui Huang, Pan He, Anand Rangarajan et Sanjay Ranka proposent une architecture de réseaux convolutifs à deux flux pour la détection des accidents en temps réel. Ce réseau convolutif se subdivise en:

* Un réseau de flux spatial pour la détection des véhicules de façon individuelle et des probables régions d’accidents sur une image donnée. L’objectif ici est de détecter les caractéristiques d’un objet en se basant sur une méthode de détection d’objet.
* Le réseau flux temporel exploite certaines caractéristiques de mouvement des objets détectés par le réseau à flux spatial afin d'effectuer un suivi d’objets multiples et par la suite générer les trajectoires des objets ciblés.

La combinaison des caractéristiques d’apparences(flux spatial) et de mouvement(flux temporel) nous permet ainsi grâce à un algorithme de prédire un accident. Pour mettre en place une telle architecture on a besoin de maîtriser trois éléments:

* La détection d’objet
* le Tracking d’objets multiple
* Et la détection d’accidents

1. **Détection d’objet**

La détection d’objet possède plusieurs applications:

* La vision par ordinateur
* La reconnaissance faciale
* Le suivi d’objets et la vidéosurveillance….

Il faut noter ici que la détection d’objet est différente sur certains aspects de classification d’image. Détecter un objet sur une image c’est donner sa classe et ses informations de localisations sur l’image.

On distingue 03 méthodes de détection d’objets:

1. **Approche traditionnelle basée sur l’apprentissage automatique**

* La sélection de région: Utilisation de fenêtres coulissantes pour sélectionner les zones d’intérêts
* Extraction de features: On extrait ici les features des régions d’intérêts à l’aide de fonctions comme le HOG.
* utilisation d’un classificateur: SVM

**Inconvénients**

* **Demande une grande sollicitation du CPU** : Car les fenêtres coulissantes utilisées pour la sélection des régions d’intérêts sont redondantes et donc gourmandes en temps CPU.
* **Non robustes :** À cause de la diversité des objets, de la déformation, des conditions d’éclairage, ….

1. **Approche basée sur l’apprentissage profond centré sur des propositions par région**

On utilise ici les réseaux convolutifs pour extraire les caractéristiques des propositions de régions d’intérêts pour produire plusieurs propositions d’objets. On utilise ensuite un classificateur comme le SVM pour affiner ces propositions. Ces méthodes sont dites méthodes en deux étapes. Le principe est le suivant:

* Collecter des images d’entrées
* Générer un certain nombre de propositions de régions
* Extraire les fonctionnalités à l’aide d'un réseau convolutif
* Classer les régions à l’aide d’un classificateur robuste comme SVM

Les réseaux convolutifs utilisés pour extraire les features à partir des propositions des zones d’intérêts:

* R-CNN
* Fast R-CNN
* Faster R-CNN

1. **Approche basé sur l’apprentissage profond de bout en bout**

On les appelle les méthodes en une étape. Elles ne nécessitent pas de propositions de régions. Parmi ces méthodes on peut retenir:

* SSD
* YOLO

Yolo utilise un réseau convolutif pour prédire des boîtes englobantes et les probabilités de classes correspondantes. La vitesse de détection est de 45 frames par seconde. L’inconvénient de YOLO c’est son incapacité à détecter des objets de très petites tailles.

1. **Architectures**
   1. **Architecture de détection d’accidents en utilisant les audio des crash et le bruit de dérapages de pneus.**

L’architecture permet la détection des accidents de route rien qu’en utilisant les audio des crashs d’accidents et du bruit de dérapage des pneus lors d’un accident. Le système sera ajouté dans un autre système déjà existant. L’objectif est d’améliorer la performance de détection des accidents de la route.

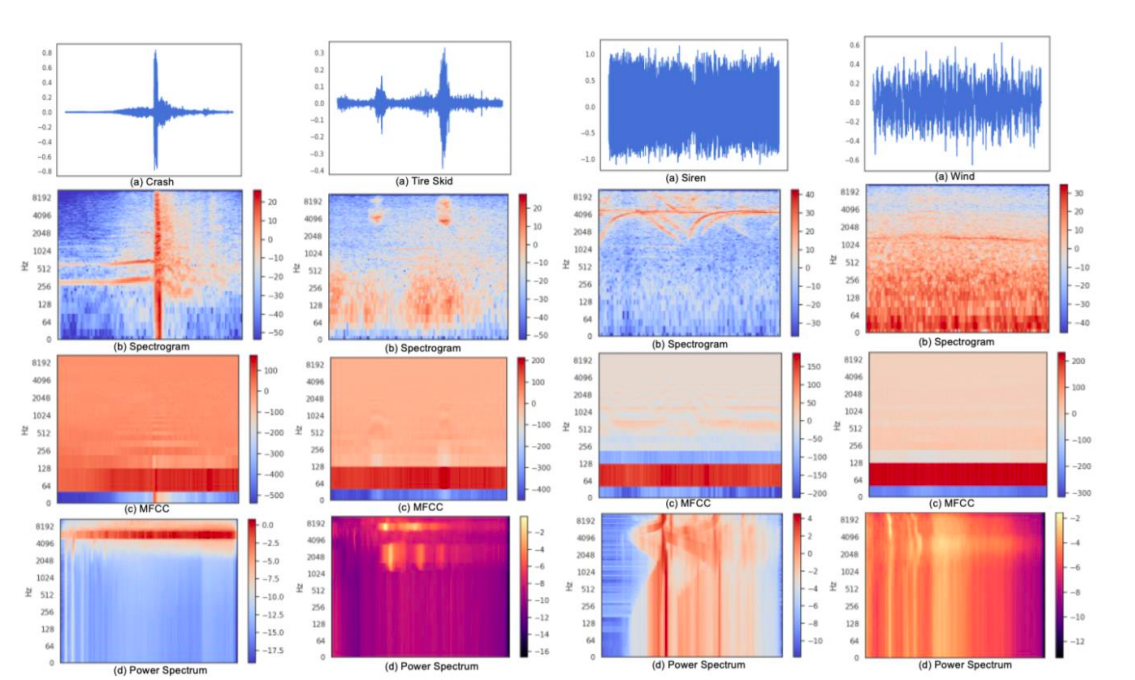
Importants audio parameters:

* Mel Frequency Cepstral Coefficients(MFCC)
* Short Time Fourier Transform
* Power Spectrum or Power density spectrum(PSD)

Le système d’acquisition des audios peut être implémenté comme une partie de la plateforme IoT.

1. **Feature Extraction**

On s’intéresse ici à 3 types de caractéristiques:



1. **Le MFCC**

Il est très utilisé en reconnaissance vocale. Les différentes étapes à suivre pour obtenir le MFCC:

* Divisé le signal en des images spécifiées
* Obtenir l’amplitude du spectre
* Prendre le logarithme de cette amplitude
* Converting to mel-Spectrum
* Taking the Discrete Fourier Transform

La fréquence d’échantillonnage est de 44.1KHz

Coefficients du MFCC = 1000 ce qui nous donne la dimension (430, 120)

1. **Short Time Fourier Transforme**

* x(i) : Signal audio échantillonné
* w(i) : représente la fenêtre
* Fe = 5.49 KHz
* Window width = 860 sample

1. **Power Spectrum or Power Density Spectrum(PSD)**

La PSD décrit comment l’énergie du signal change en fonction de la fréquence.

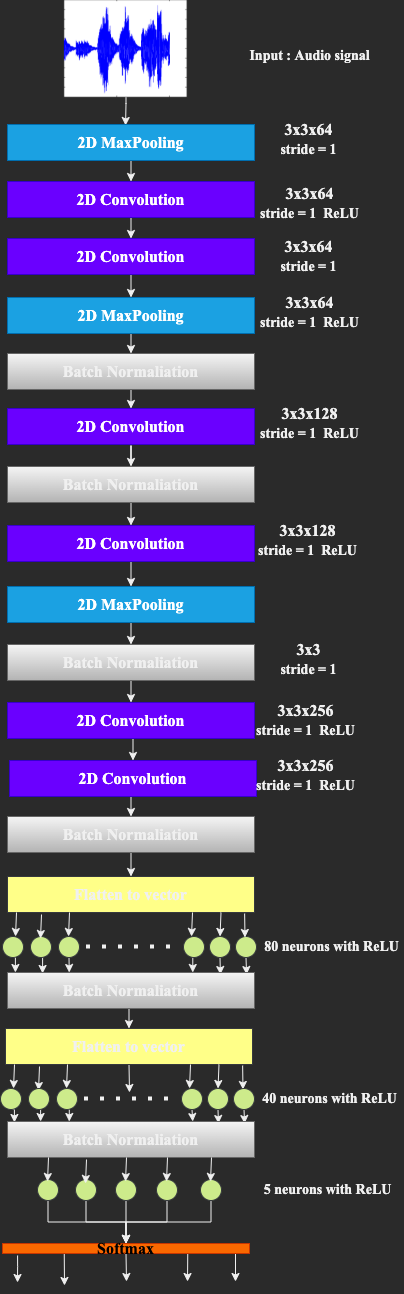
* Fe = 22.1KHz
* number of filter = 128
* Window 0.7S

1. **Data Augmentation**

Nous allons apporté Trois types de modifications sur nos audios:

* **Background Noise:** On va mixer l’audio avec un bruit de fond dont les amplitudes minimum et maximums sont connus (resp 0.001 et 0.015)
* **Time Stretch** : A sound sample can be sped up or slowed using this feature without changing the pitch.
* **Pitch shift** : On modifie le pitch de l’audio
* **Shift** : On va décaler les échantillons de l’audio dans le temps.

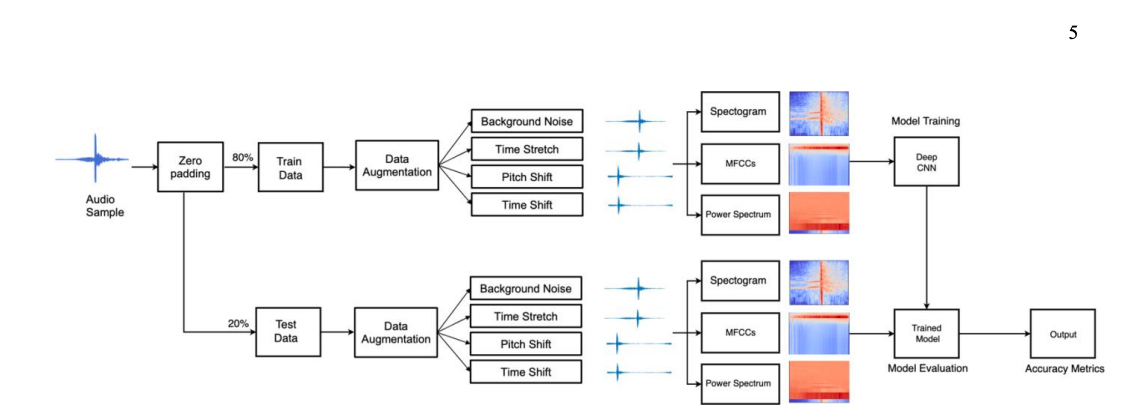
1. **Model Architecture**
2. **CNN architecture**



**Le modèle possède 5 couches séparées à chaque fois par une couche de batch normalisation. Cette couche permet d'accélérer la convergence du modèle proposé. On distingue donc:**

* **Couche 1 :** 
  + **64 filtres de tailles 3x3 avec une fonction d’activation ReLU**
  + **Ces Filtres sont précédés par une couche de Max Pooling qui permet de réduire la taille des cartes de caractéristiques afin de réduire le temps d'entraînement**
* **Couche 2 :**
  + **64 filtres de tailles 3x3 avec une fonction d’activation ReLU**
  + **Suivi d’une couche de Max Pooling avec un pool size de 3x3**
  + **Une couche de Batch-Normalization**
* **Couche 3 :**
  + **128 filtres de kernel 3x3 avec une fonction d’activation ReLU**
  + **Suivi d’une couche de batch-Normalisation**
* **Couche 4 :** 
  + **128 filtres de kernel 3x3 avec une fonction d’activation ReLU**
  + **suivi d’une couche de Pooling de dimension 3x3**
  + **Suivi d’une couche de batch normalisation**
* **couche 5:** 
  + **256 filtres de noyau 3x3 avec une fonction d’activation ReLU et un padding = ‘SAME’**
* **Couche 6:**
  + **256 filtres de noyau 3x3 avec une fonction d’activation ReLU et un padding = ‘SAME’**
  + **Suivi d’une couche de batch-normalisation**
* **Couche 7 :**
  + **80 neurones cachés avec une fonction d’activation ReLU**
  + **Ces neurones sont précédés par une couche flatten**
* **Couche 8 :**
  + **40 neurones cachés avec une fonction d’activation ReLU**
* **Couche 9 :** 
  + **5 couches de sorties avec une fonction d’activation SoftMax**

1. **Model Implementation FlowChart**

****

**Note sur TensorFlow2.0**

1. **Les réseaux de neurones**

Tensorflow nous permet de mettre en place un réseau de neurones. 03 éléments essentiels:

* Puissance de calcul
* Data
* Méthodologie

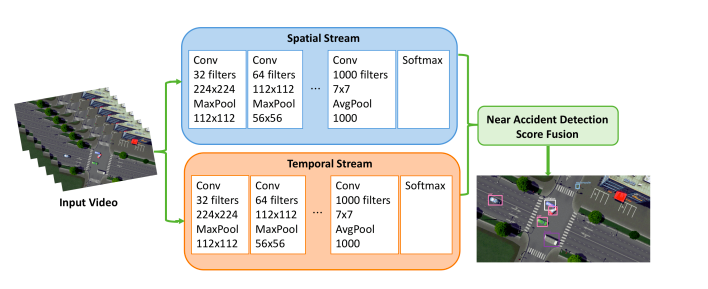
1. **Le perceptron**

Un perceptron c’est la modélisation d’un neurone qui permet de prédire une sortie. A Chaque entrée on associe un poids synaptique. Ceci donne une certaine importance à chaque entrée. Le perceptron est un modèle de classification linéaire.

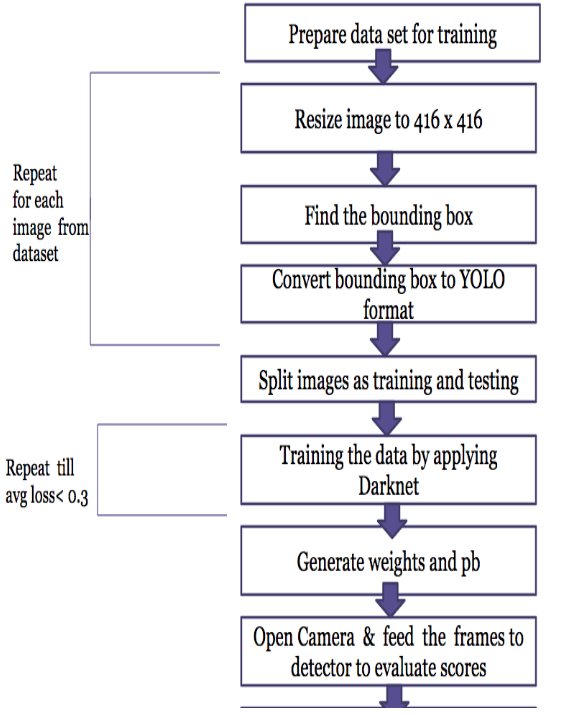
1. **L’algorithme de descente de gradient**

**L’algorithme de descente de gradient un algorithme d'optimisation. Il permet de réduire l’erreur de prédiction du neurone. Ce qui se passe c’est qu’on prend à chaque le gradient de l’erreur par rapport à chaque poids. Et ce gradient va nous renseigner dans quel sens il faut varier le poids afin de réduire l’erreur.**

* 1. **Architecture du système basé sur Yolo et DeepSort**

****

* + 1. **Flowchart de la détection spatiale avec Yolo**

****

* + 1. **Flowchart de la détection Temporelle avec DeepSort**
    2. **Algorithme de détection d’accident**