Cycle de formation des ingénieurs en informatique

Rapport de stage

Thème :

plateforme smart pour la détection des agressions verbales

Réalisé par :

Montassar nawara

Encadrée par :

MMe Yemna Sayeb

Période : de 15/07/2025 à 31/08/2025

I. Introduction générale

Dans notre société moderne, caractérisée par une vie quotidienne rythmée par les avancées technologiques et les interactions constantes, la question de la sécurité publique demeure cruciale. Bien que nous appartenions à une génération considérée comme l'une des plus avancées sur le plan intellectuel et technologique, des phénomènes de violence continuent à se manifester, que ce soit dans l'espace public ou privé.

La violence, sous ses différentes formes, reste une réalité préoccupante. Elle peut se traduire par des agressions physiques, verbales, des vols, des harcèlements ou toute autre atteinte aux droits fondamentaux des individus. Ces comportements agressifs témoignent d'un non-respect des lois et des valeurs humaines qui fondent la vie en société.

Historiquement, l'homme a toujours cherché à vivre en communauté pour assurer sa sécurité et celle des siens. Le regroupement urbain et l'organisation sociale ont pour objectif d'instaurer une paix durable. Cependant, malgré les efforts déployés par les États, les autorités locales et les forces de l'ordre, les actes d'agression persistent dans nos villes et nos quartiers.

Il ne s'agit pas là d'événements rares ou exceptionnels : chacun d'entre nous a déjà été témoin, victime ou au moins informé d'un acte de violence survenu dans son environnement proche. Cette banalisation de l'agression soulève une interrogation fondamentale : comment peut-on, à défaut d'éradiquer totalement ces actes, protéger les victimes potentielles et assurer que les agresseurs soient rapidement identifiés et traduits en justice ?

Le défi est de taille. De nombreux cas d'agressions se terminent en drames humains, et d'autres ne sont jamais résolus faute de preuves ou en raison de la fuite de l'agresseur. Ce constat met en évidence une nécessité impérieuse : renforcer les outils de détection, de prévention et d'intervention rapide face à ces actes.

Parallèlement, notre époque est marquée par une explosion des données et un développement fulgurant de l'intelligence artificielle. Cette évolution offre aujourd'hui des opportunités inédites pour relever le défi de la sécurité dans l'espace public. Grâce à l'analyse de données massives (big data), à la reconnaissance vocale et visuelle, et à la puissance des algorithmes intelligents, il devient possible de concevoir des systèmes capables de détecter automatiquement des situations à risque.

C'est dans cette optique que notre projet s'inscrit : proposer une solution technologique basée sur l'intelligence artificielle permettant la détection en temps réel des agressions verbales dans les espaces publics. Ce système reposerait sur l'usage de capteurs sonores et visuels implantés dans les zones sensibles des villes.

Dans un premier temps, ces capteurs seraient capables d'identifier une agression verbale à travers l'analyse acoustique ou visuelle. Ensuite, en une fraction de seconde, les données captées seraient transmises à un système centralisé d'analyse qui évaluerait la gravité de la situation.

Selon le niveau de menace détecté, le système pourrait alors déclencher différentes réponses : envoi automatique d'un signal aux patrouilles les plus proches, activation d'effets sonores ou lumineux pour dissuader l'agresseur, ou encore enregistrement de la scène pour usage ultérieur par les autorités compétentes.

L'objectif final est de garantir une intervention rapide, dissuader les comportements agressifs, et renforcer la sécurité des citoyens dans les zones urbaines à forte densité. Ce projet s'inscrit donc à la croisée entre innovation technologique et responsabilité sociale.

En somme, notre démarche vise à mobiliser les outils modernes d'analyse et d'intelligence artificielle pour répondre à une problématique humaine ancienne : la violence urbaine. En combinant capteurs intelligents, traitement en temps réel, et systèmes d'alerte, nous espérons proposer une solution efficace, rapide et éthique. Protéger les individus vulnérables, prévenir les drames et renforcer la justice sont les moteurs de notre initiative. Ce projet n'est pas seulement technologique, il est profondément humain. Il reflète notre engagement envers une société plus sûre et solidaire.

II. Étude quelques travaux similaires

Dans un premier temps, il est essentiel d'examiner des travaux de recherche existants dans le domaine, à l'échelle mondiale, afin d'identifier les approches déjà proposées pour répondre à des problématiques similaires à la nôtre. Cette revue de littérature nous permettra :

- D'analyser les idées développées dans des projets antérieurs
- De repérer les points forts et les éléments compatibles avec notre propre solution
- D'utiliser certains projets comme base de référence pour construire et améliorer notre approche.

L'objectif est donc de nous appuyer sur ces travaux pour consolider notre démarche, éviter de repartir de zéro, et garantir que notre projet soit à la fois innovant et ancré dans une réalité technologique et scientifique déjà explorée.

1. Définition des concepts fondamentaux

Avant d'aborder les projets existants, il convient de bien définir les notions clés qui structurent notre sujet. Cela facilitera la compréhension des travaux étudiés et permettra de mieux situer notre projet dans le paysage technologique actuel.

1.1. Agressions verbales

Les agressions verbales désignent tout acte de communication violent ou menaçant, qui vise à nuire psychologiquement à une personne. Elles incluent les insultes, cris, menaces, propos dégradants ou discriminatoires, et peuvent provoquer un sentiment d'insécurité important, même en l'absence de contact physique.

1.2. Détection et analyse de données

Il s'agit de l'ensemble des techniques permettant de collecter, traiter et interpréter des données issues de différentes sources (audio, vidéo, etc.) dans le but d'identifier des situations suspectes ou à risque. Ces analyses peuvent être basées sur des algorithmes de machine learning, de traitement du signal ou de reconnaissance d'événements.

1.3. Signal

Le signal, dans le contexte de notre projet, désigne l'alerte transmise automatiquement lorsqu'une agression est détectée. Ce signal peut être dirigé vers les autorités (ex. : patrouille de police) ou vers des dispositifs dissuasifs (sirènes, lumières, messages audio...).

1.4. Détection sonore

La détection sonore repose sur l'analyse de l'environnement acoustique afin d'identifier des anomalies vocales comme des cris, des insultes ou des bruits violents. Des microphones intelligents associés à des algorithmes d'analyse permettent de reconnaître ces événements en temps réel.

1.5. Détection visuelle

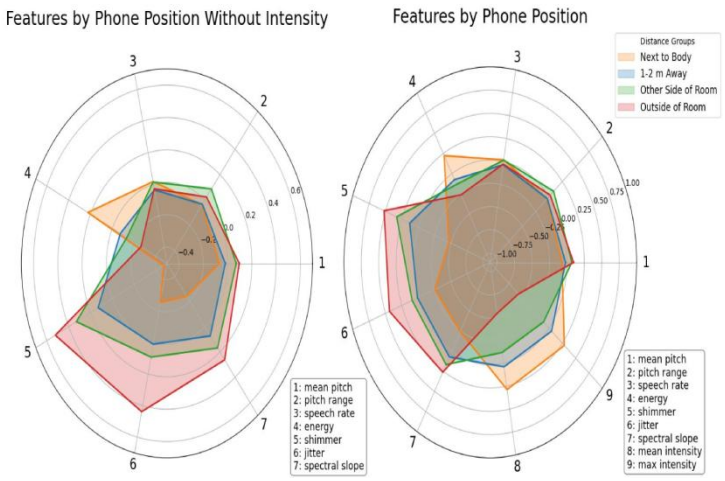
La détection visuelle utilise des caméras et des techniques de vision par ordinateur pour analyser les comportements humains, les expressions faciales ou les mouvements suspects. Couplée à la détection sonore, elle renforce la fiabilité du système en apportant une confirmation visuelle des incidents détectés.

2. travaux similaires

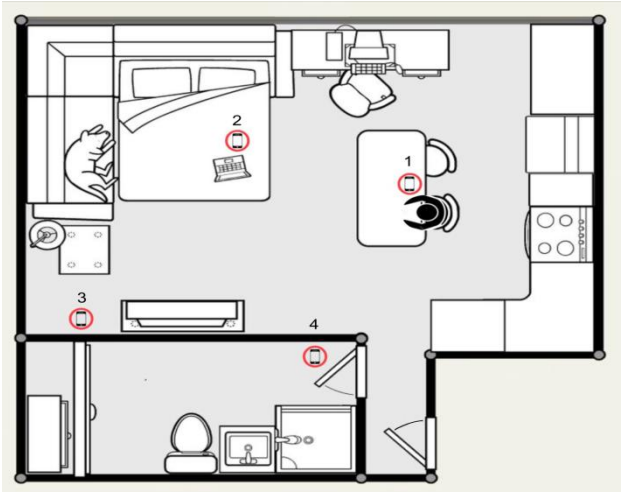
2.1. Étude de article « BERSting at the Screams: A Benchmark for Distanced, Emotional and Shouted Speech Recognition **»**

2.1.1. Présentation

Des chercheurs de la **Simon Fraser University** (Canada), **SUPMICROTECH/CNRS** (France) et **Enchanted Tools** ont développé le corpus **BERSt**, un benchmark vocal inédit destiné à évaluer la reconnaissance automatique de la parole dans des conditions extrêmes : cris, émotions fortes, distance.



2.1.2. Mission



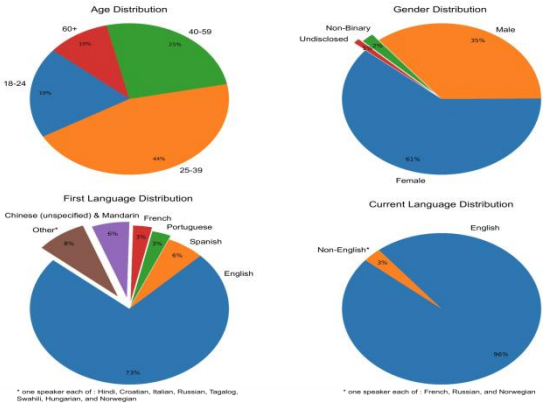
Ce travail vise à combler un manque dans les bases de données vocales actuelles, en fournissant une ressource dédiée à l'étude de la **parole émotionnelle, criée ou distante**, enregistrée dans des environnements réalistes avec des smartphones.

- Il s'agit de tester plusieurs aspects :
- Reconnaissance automatique de la parole (ASR),
- Détection de cris,
- Reconnaissance des émotions dans la voix (SER)

2.1.3. Technologies utilisées

- **Corpus vocal** :

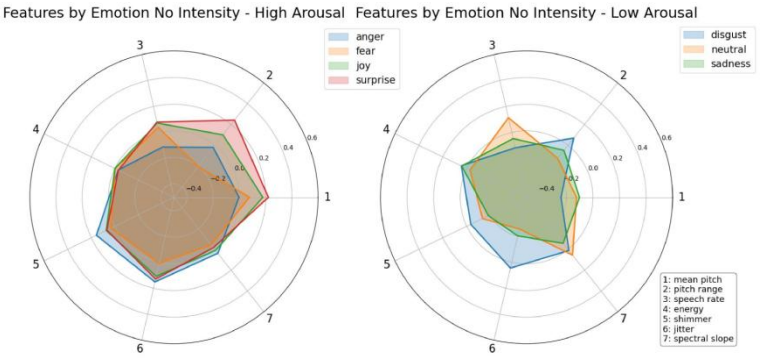
• 4526 clips (~4 heures),
• 98 locuteurs,
• 7 émotions différentes (joie, peur, colère, etc.),
• Enregistrements faits avec **smartphones** dans **98 configurations d'environnement et de positionnement** (19 positions de téléphone).



- **Modèles et outils testés** :

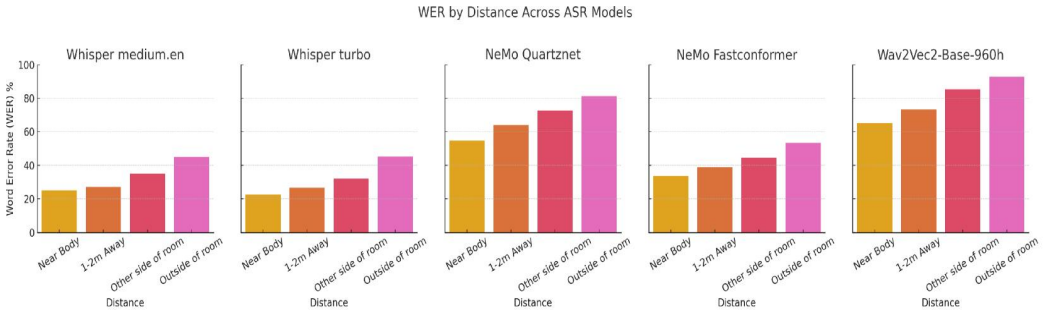
• ASR : Whisper Turbo, SpeechBrain, etc.
• SER : DAWN, Wav2Small
• Analyse fine des performances selon la distance, le type de cri ou d'émotion.

MODEL	ARCHITECTURE	# PARAM.	DATASET	# H
Whispermedium.en	sequence-to-sequence	769 M	OpenAI custom**	680,000
Whisper-turbo	sequence-to-sequence*	798 M	OpenAI custom**	680,000
Ne Mo Quartz net	CNN† with residual connections	18.9 M	Libri Speech, Common Voice	3,500
Ne Mo Fast Conformer Trans ducer	Auto regressive conformer	114 M	NeMo ASRSet	24,500
Wav2Vec2-Base960h	CNN-based encoder transformer	94.4 M	LibriSpeech	960



2.1.4. Avantages :

- Enregistrements **réalistes** (domiciles, smartphones, situations naturelles).
- Corpus **riche et unique** combinant distance + cri + émotion.
- Permet de mesurer **la robustesse des modèles IA** face à des données complexes et bruitées.
- Benchmark ouvert, utilisable pour évaluer d'autres modèles de détection vocale



2.1.5. Conclusion

Le corpus BERSt révèle des performances médiocres :

- WER (Word Error Rate) ~ 30 % pour Whisper turbo ;
- SER ~ 32 % de précision non pondérée (DAWN) Cela démontre l'écart important entre les capacités des modèles actuels et leurs performances en conditions réelles, confirmant la nécessité de corpus de ce type et l'importance de poursuivre les recherches pour améliorer la robustesse de l'ASR et SER en contexte urbain.

https://arxiv.org/html/2505.00059v1?

https://www.aimodels.fyi/papers/arxiv/bersting-screams-benchmark-distanced-emotional-shouted-speech?

A. Apport pour nos projet

Ce travail apporte une contribution essentielle à notre problématique, car il **cible exactement les mêmes contraintes** :

- reconnaissance vocale en **situation tendue (cris, agressions verbales)**,
- en **environnement urbain réel**,
- avec des **dispositifs simples comme des smartphones ou capteurs acoustiques**.

Il nous fournit une **base technique, méthodologique et de comparaison** précieuse pour construire ou entraîner un système de détection d'agressions vocales dans l'espace public.
Nous pouvons ainsi :

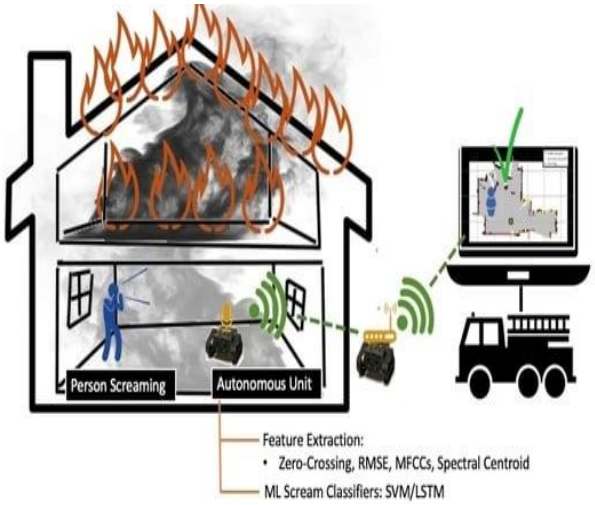
- utiliser les **résultats comme point de départ**,

- tester nos propres algorithmes sur le corpus BERSt,
- ou **étendre la méthode** en y ajoutant la **réaction automatique (alerte, patrouille)**, absente dans leur approche.

2.2. Étude de article « An Initial Machine Learning-Based Victim's Scream Detection»

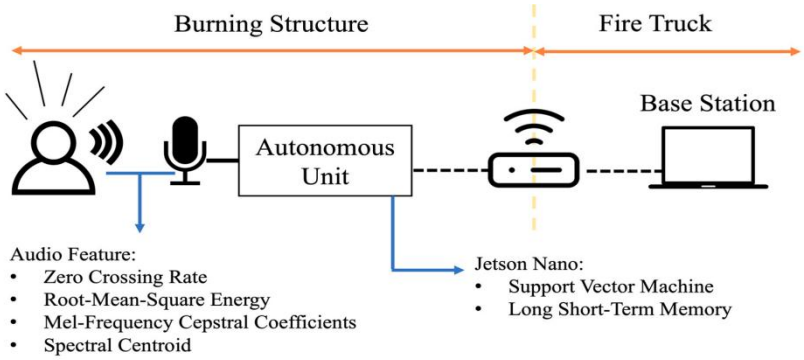
2.2.1. Présentation

L'article propose un système automatisé de détection des cris de victimes lors d'incendies, utilisant des techniques de machine learning. Les auteurs comparent deux modèles : SVM et LSTM, appliqués à des signaux audio pour localiser rapidement les personnes piégées. Les résultats montrent que le SVM offre de meilleures performances en temps réel grâce à sa faible complexité. Le système est conçu pour être embarqué dans un véhicule autonome de secours. Cette solution vise à améliorer la réactivité des interventions d'urgence en conditions extrêmes.



2.2.2. Mission

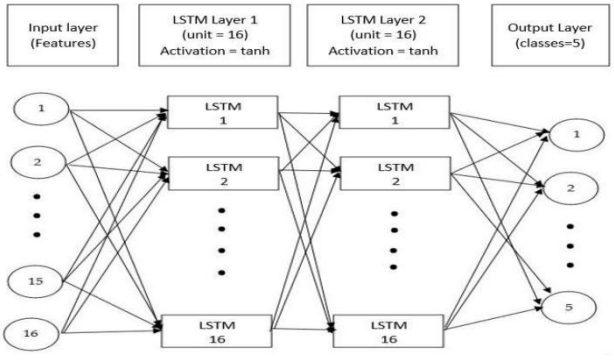
Le but de cette recherche est de détecter les cris de détresse des victimes piégées dans des sites en feu afin d'accélérer les opérations de secours.
Le système vise à capter les signaux audio (cris) dans des environnements extrêmes et de faible visibilité, ce qui représente un grand défi pour les secouristes.



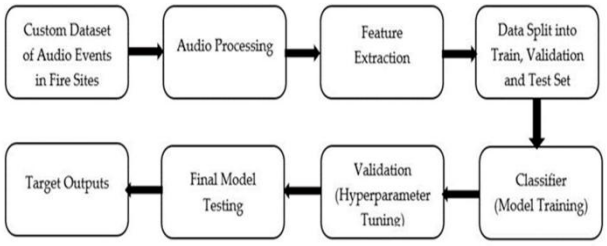
2.2.3. Technologies

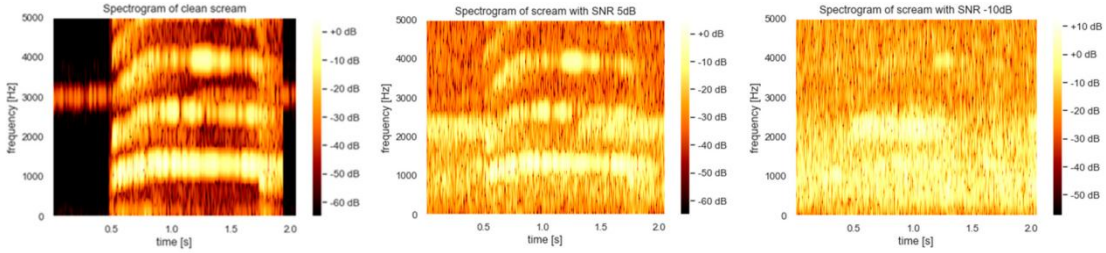
- **Méthodes de machine learning :**

  - SVM (Support Vector Machines)
  - LSTM (Long Short-Term Memory)



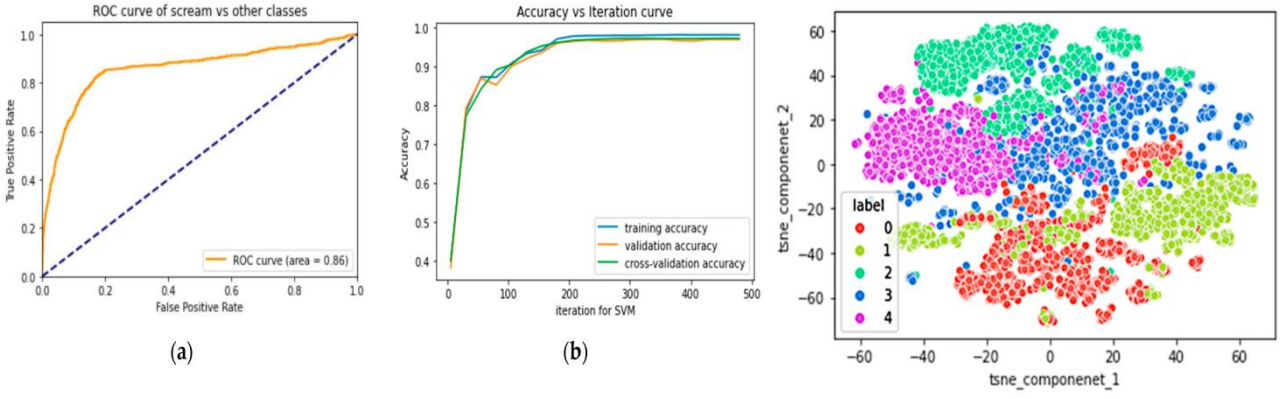
- **Traitement audio** pour l'extraction des caractéristiques des cris (features)
- **Système embarqué autonome (AESV)** pour l'implémentation en situation réelle





2.2.4. Avantages :

- Très bonne précision dans la détection des cris de victimes
- SVM, plus léger que LSTM, est adapté au déploiement en temps réel
- Le système fonctionne sans vision (utile en cas de fumée dense)
- Contribue à la rapidité des secours dans des situations critiques



2.2.5. Conclusion

L'étude montre que l'audio est un vecteur efficace pour la détection de détresse dans les incendies. L'approche par SVM est prometteuse pour une intégration sur des robots ou drones de sauvetage.

LSTM reste pertinent mais est plus complexe. Ce système peut être crucial pour réduire les délais de sauvetage.

B. Apport pour nos projet

Cet article apporte un **appui technique solide** sur deux points clés de projet :

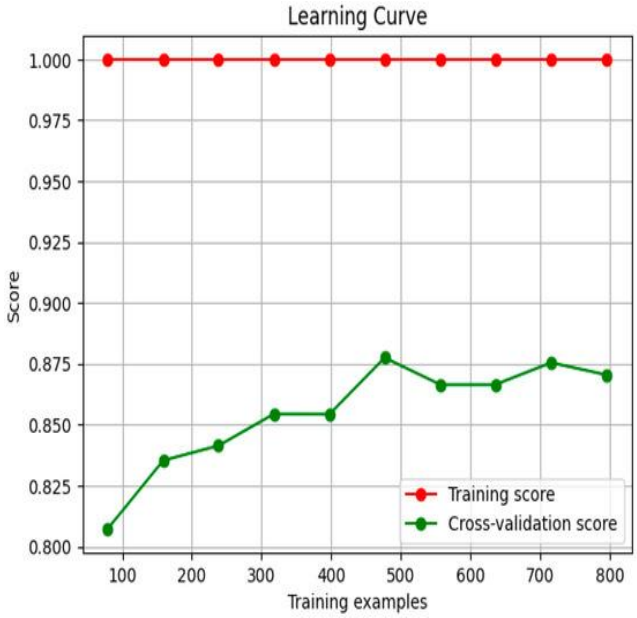
- Il valide l'approche **audio/machine learning** pour détecter des signaux d'alerte humains.
- Il prouve la **faisabilité en temps réel**, ce qui soutient ton idée de déclenchement automatique (signal vers les patrouilles).

https://www.mdpi.com/2076-3417/11/18/8425

2.3. Étude de article « Enhancing the Prediction of Episodes of Aggression in Patients with Dementia Using Audio-Based Detection: A Multimodal Late Fusion Approach with a Meta-Classifier »

2.3.1. Présentation

L'article propose une méthode innovante pour anticiper les épisodes d'agressivité chez les patients atteints de démence à l'aide d'une détection basée sur l'audio. Il s'agit d'une approche multimodale combinant plusieurs types de signaux (audio, contexte) avec un méta-classifieur utilisant une fusion tardive.



2.3.2. Mission

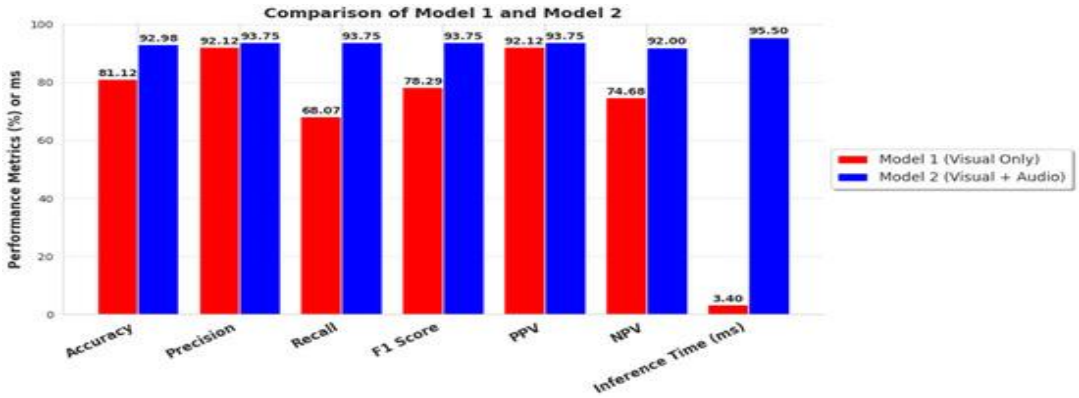
L'objectif est de prédire de manière proactive les comportements agressifs afin de permettre une intervention précoce dans les établissements de soins, assurant ainsi une meilleure sécurité pour les patients comme pour les soignants.

2.3.3. Technologies

- Détection et traitement de signaux audio
- Apprentissage automatique (machine learning)
- Approche de **fusion tardive multimodale**
- Méta-classifieur pour combiner différentes sources de données

2.3.4. Avantages

- Amélioration de la précision de la prédiction par rapport aux approches univariées
- Utilisation de données non intrusives (audio)
- Adaptabilité du système dans un cadre médical sensible
- Potentiel d'intégration dans des systèmes de surveillance automatisés



2.3.5. Conclusion

Cette recherche démontre la pertinence de l'audio comme indicateur comportemental dans des contextes médicaux. En combinant plusieurs sources de données et en exploitant des modèles avancés, le système permet une meilleure anticipation des crises d'agressivité, contribuant ainsi à des environnements de soins plus sûrs et plus réactifs.

C. Apport pour nos projet

Le lien avec nos projet est direct et pertinent : dans les deux cas, l'objectif est de détecter **des actes d'agression à partir de signaux sonores** pour déclencher une **réaction rapide**. Tandis que l'article cible un cadre médical (patients atteints de démence), le projet se concentre sur **l'espace public urbain**, mais l'enjeu est similaire : **protéger des personnes vulnérables par la détection automatique d'un comportement verbal agressif**. Les approches technologiques (capteurs sonores, traitement temps réel, machine learning) sont comparables et renforcent la validité de ta démarche.

https://www.mdpi.com/2076-3417/15/10/5351

4. Conclusion générale des travaux similaires

L'analyse des trois travaux similaires met en lumière l'évolution significative des technologies de détection basées sur le son et l'intelligence artificielle pour répondre à des problématiques critiques liées à la sécurité et à la santé. Le premier article a posé les bases d'une reconnaissance vocale avancée dans des contextes bruyants et émotionnellement chargés, en proposant un référence rigoureux pour la détection de cris et de discours agressifs. Le second a démontré la faisabilité d'un système de détection automatisé de cris dans des environnements extrêmes comme les incendies, en comparant les performances de modèles SVM et LSTM. Enfin, le troisième a étendu cette logique au domaine médical, en anticipant les comportements agressifs chez des patients vulnérables à l'aide d'une approche multimodale audio-centrée.

Ces travaux confirment la **pertinence, la faisabilité technique et l'impact sociétal** des systèmes de détection acoustique intelligente. Chacun, dans son contexte, souligne la nécessité d'une réponse rapide, précise et non intrusive aux comportements agressifs ou aux situations de danger.

Ainsi, notre projet s'inscrit dans cette continuité, en adaptant les approches existantes à l'espace public urbain. Il vise à offrir une solution innovante de détection en temps réel des agressions verbales, contribuant à renforcer la sécurité des citoyens grâce à l'exploitation éthique et efficace de l'intelligence artificielle.

III. recherches et analyses de Conception sur les études sur sonore

1. Analyse de technique

Dans cette partie, nous allons explorer les bibliothèques disponibles dans différents langages de programmation, principalement Python, qui permettent d'analyser des signaux audio en temps réel, de traiter et structurer les données sonores, ainsi que de manipuler, enregistrer et stocker ces données de manière efficace. L'objectif est de disposer d'un socle technologique solide pour développer un système intelligent de détection d'agressions verbales dans des environnements urbains.

1.1. Étude des bibliothèques de manipulation sonore

1.1.1. Pydub (Python)

Pydub est une bibliothèque Python simple et efficace pour la manipulation de fichiers audio. Elle permet de lire, découper, combiner, convertir et exporter des sons dans divers formats (MP3, WAV, FLAC, etc.).

a) Utilité

traitement de base, conversion de formats, effets simples.

b) Limite

pas adaptée pour du traitement audio en temps réel ou des analyses acoustiques complexes.

1.1.2. Librosa (Python)

Librosa est une bibliothèque puissante pour l'analyse audio et la musique. Elle permet d'extraire des caractéristiques audio telles que les MFCC, le tempo, le pitch, les spectrogrammes, etc.

a) Utilité

idéale pour le machine learning audio, la reconnaissance vocale, la classification des émotions ou cris.

b) Limite

nécessite une bonne maîtrise des concepts audio pour une exploitation optimale.

1.1.3. Sounddevice (Python)

Sounddevice permet d'enregistrer ou de lire des flux audio en temps réel à partir du microphone ou des haut-parleurs. Elle est compatible avec NumPy pour un traitement direct des données sous forme de tableaux

a) Utilité

enregistrement en temps réel, interface simple avec le matériel.

b) Complémentaire
Librosa ou Pydub pour l'analyse après enregistrement.

1.1.4. OpenSMILE (C++/CLI via Python bindings)

OpenSMILE est un outil très puissant pour l'extraction de caractéristiques audio utilisées en reconnaissance d'émotions, analyse vocale, stress ou agressivité.
a) Utilité
extraction de très haut niveau de descripteurs vocaux pour les modèles ML.
b) Utilisé
echerche académique, projets de santé mentale, systèmes de détection émotionnelle.

1.1.5. SpeechRecognition (Python)

Cette bibliothèque permet de convertir l'audio en texte en utilisant divers moteurs (Google, Sphinx, etc.).
a) Utilité
econnaissance de mots ou phrases agressives à partir de la voix.
b) Intégration
très utile pour le NLP combiné avec des modèles de classification.

1.1.6. Web Audio API (JavaScript)

API native des navigateurs web pour la capture, manipulation et analyse du son directement dans le navigateur.
a) Utilité
utile pour des applications web embarquant de la détection ou des capteurs sonores. Mais elle est pour la navigateur uniquement

1.1.7. TarsosDSP (Java)

TarsosDSP est une bibliothèque Java utilisée pour l'analyse du son et la reconnaissance musicale. Elle permet l'extraction de pitch, MFCC, etc.
a) Utilité
alternative Java pour des projets Android ou embarqués.

Résumé de cette section

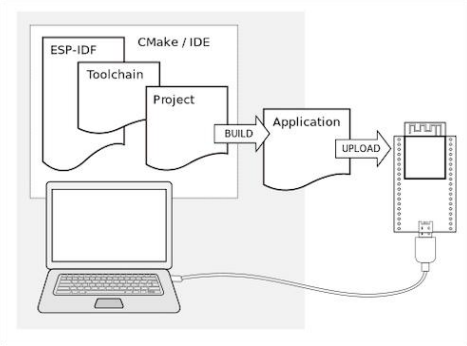
La combinaison de bibliothèques comme **Librosa**, **OpenSMILE**,**SpeechRecognition,** et **Sounddevice** offre une base technologique robuste pour capturer, analyser et interpréter les sons suspects dans l'espace urbain. Ces outils sont essentiels pour développer des modèles de détection d'agressions verbales, en temps réel ou différé,

et pour assurer une précision dans l'identification du stress, de la colère ou de la violence dans la voix.

1.2. Conception de étude sonore

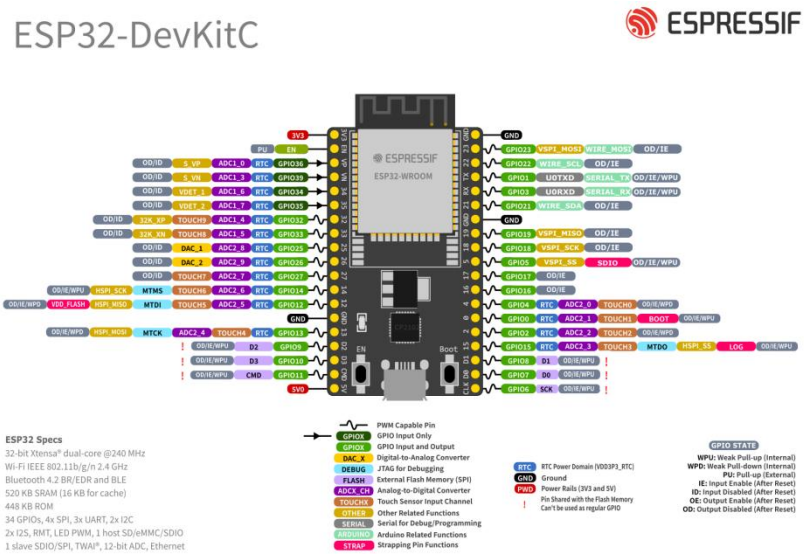
1.2.1. Cartes embarquées (microcontrôleurs)

A. ESP32



- **Définition** : Microcontrôleur 32 bits avec Wi-Fi et Bluetooth intégrés.
- **Langage de programmation** : C/C++ (via Arduino IDE ou ESP-IDF), MicroPython.
- **Avantages** :

  ✓ Supporte l'**I2S** pour microphones numériques.
  ✓ Faible consommation d'énergie.
  ✓ Connectivité sans fil native.



Utilité pour nos projet : Idéal pour placer des capteurs sonores autonomes, capables de transmettre le signal en temps réel via Wi-Fi à un serveur centra

B. Arduino

- **Définition** : Microcontrôleur simple, très utilisé pour les prototypes.
- **Langage de programmation** : C/C++ avec l'IDE Arduino.

- **Avantages** :

  ✓ Facilité d'utilisation et large communauté.
  ✓ Compatible avec de nombreux capteurs analogiques.

Limites :

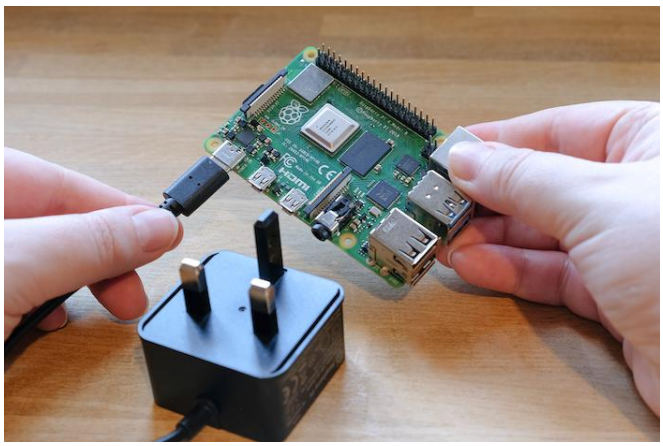
✧ Pas de connectivité native (sauf si ajout de module Wi-Fi).
✧ Mémoire limitée, peu adapté au traitement audio complexe.

Utilité pour nos projet : utile en phase de prototypage ou pour des traitements simples (seuil sonore, transmission série...).

C. Raspberry Pi (3/4/Zero)

- **Définition** : Nano-ordinateur avec OS complet (Raspberry Pi OS).
- **Langages de programmation** : Python, C++, Node.js, etc.
- **Avantages** :

  ✓ Supporte les bibliothèques audio avancées comme pyaudio, librosa, pydub.
  ✓ Capable d'exécuter des modèles de machine learning légers.
  ✓ Supporte les connexions USB, I2S, ou Jack 3.5mm pour microphone.



Utilité pour nos projet : point de traitement central capable de recevoir des données de plusieurs capteurs (ESP32), analyser le son, détecter des agressions, et transmettre des alertes.

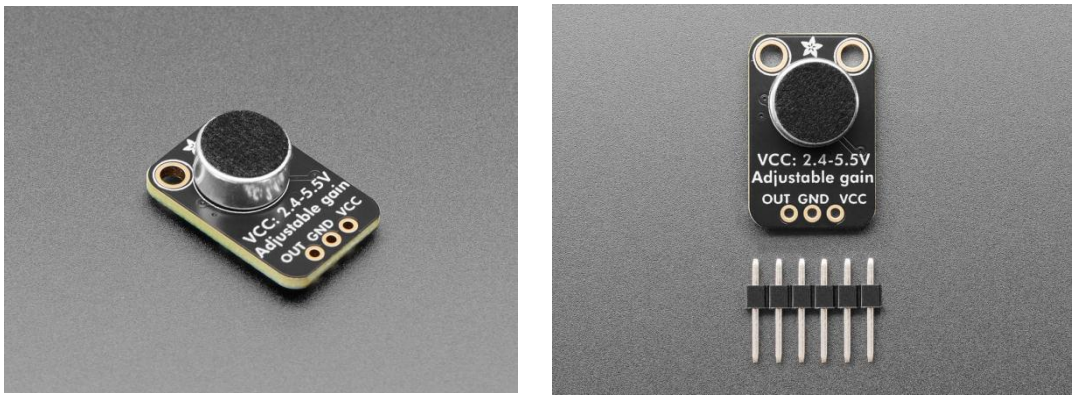
1.2.2. Les capteur sonore

La détection fiable des agressions verbales dans un environnement urbain repose avant tout sur l'acquisition d'un signal sonore de bonne qualité. Le choix du capteur est donc une étape critique dans la conception de notre système.

A. Types de capteurs sonores

Il existe plusieurs types de microphones adaptés à différents usages. Voici les plus pertinents pour notre projet :

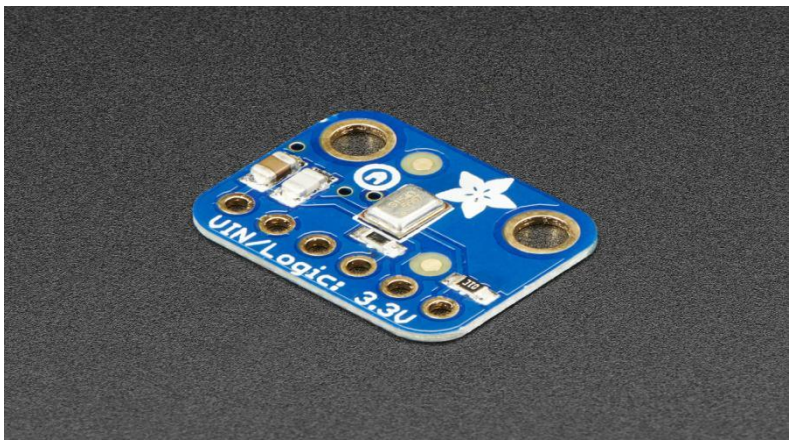
- **Microphones électret** : très utilisés pour les projets embarqués. Peu coûteux, compacts, mais sensibles aux bruits parasites.
  Exemple :



Adafruit Electret Microphone Amplifier

- **Microphones MEMS (MicroElectroMechanical Systems)** : compacts, résistants et précis, ils offrent une bonne sensibilité et sont idéaux pour une intégration dans des environnements urbains.

Exemple :



SPH0645LM4H MEMS I2S Microphone

- **Microphones omnidirectionnels avec AOP intégré** : ils peuvent capter le son sur 360°, ce qui est utile pour des espaces publics ouverts.

B. Conexion

Les capteurs sonores peuvent être connectés à une carte via différents types d'interfaces :

- **Analogique (A0)** : sortie de tension continue à connecter à une entrée analogique de la carte (ESP32, Arduino).
- **Numérique (I2S, I2C, SPI)** : plus précis et adapté pour des données audio temps réel.

https://www.adafruit.com/product/1063

https://learn.adafruit.com/adafruit-i2s-mems-microphone-breakout

https://docs.espressif.com/projects/esp-idf/en/latest/esp32/

https://www.raspberrypi.com/documentation/

Exemple de câblage :

- Le **micro MEMS I2S** se connecte via I2S (Data In, Clock, Word Select).
- Le **micro analogique** se connecte directement sur une broche analogique + GND + VCC.

C. Utilisation dans nos projet

Dans notre système, les capteurs seront disposés dans des zones sensibles (rues, stations, places publiques). Ils doivent capter les sons en continu, filtrer les bruits ambiants, et transmettre uniquement les signaux suspects vers l'unité centrale de traitement (ESP32, Raspberry Pi, etc.).
Des algorithmes de prétraitement pourront être embarqués (filtrage, détection de seuil, compression audio...).

1.3. Base de donnée

La base de données constitue un élément central de notre système, permettant de **stocker**, **organiser** et **interroger** les données audio, les métadonnées, les résultats de détection, ainsi que les historiques d'événements. Plusieurs systèmes de gestion de

bases de données (SGBD) peuvent être envisagés selon les besoins du projet : relationnel, NoSQL, hébergé ou local.

1.3.1. MySQL

- **Type** : Base de données relationnelle open-source.
- **Langage de requête** : SQL.
- **Avantages** :

  - ✓ Large adoption et bonne documentation.
  - ✓ Adapté pour la gestion de données structurées (utilisateurs, événements, métadonnées...).

Cas d'utilisation dans nos projet :Stockage des profils utilisateurs, journaux d'activité, logs d'alerte, etc.

https://dev.mysql.com/

1.3.2. PostgreSQL

- **Type** : Base de données relationnelle avancée, orientée objet.
- **Langage de requête** : SQL (avec extensions comme JSONB, PostGIS...).
- **Avantages** :

  - ✓ Très robuste pour les traitements complexes.
  - ✓ Support natif des données JSON, idéal pour stocker des résultats hybrides (audio + méta).

https://www.postgresql.org/docs/

Cas d'utilisation dans nos projet : Centralisation des événements audio avec stockage des analyses statistiques ou modèles prédictifs.

1.3.3. MongoDB

- **Type** : Base de données NoSQL orientée documents.
- **Langage de requête** : BSON/JSON.
- **Avantages** :

  - ✓ Flexible pour les données non structurées ou semi-structurées.
  - ✓ Haute scalabilité, facile à déployer en cloud.

https://www.mongodb.com/docs/

Cas d'utilisation dans nos projet :

Stockage de fichiers audio encodés, configurations des capteurs, historiques d'analyses sonores.

1.4. Logiciels et environnement de travail

Pour le développement, le test, et le déploiement du système de détection des agressions verbales, un environnement de travail bien structuré est essentiel. Ce dernier inclut des plateformes de versionnement, d'hébergement, de simulation et de gestion des services cloud.

1.4.1. AWS (Amazon Web Services)

Rôle : Plateforme cloud utilisée pour héberger les bases de données, les API, les fichiers audio, ou déployer des modèles IA.

Services pertinents :

- **EC2** : machines virtuelles.
- **S3** : stockage des fichiers (audio dans nos cas).
- **Lambda** : fonctions serverless pour l'analyse en temps réel.

Avantage : Scalabilité, haute disponibilité, sécurité renforcée.

https://aws.amazon.com/fr/

1.4.2. GitHub

Rôle : Plateforme de gestion de code source.

Fonctionnalités :

- Contrôle de version (Git).
- Collaboration en équipe.
- Déploiement automatisé via GitHub Actions.

Avantage : Large communauté, intégration CI/CD, visibilité du projet.

https://github.com/

1.4.3. Server pre-local

Définition :
Serveur intermédiaire entre le développement local et la production.
Utilité :

- Tester le déploiement final avec configuration proche de la prod.
- Simuler les charges utilisateurs et vérifier la stabilité de l'API ou du système de détection.

Hébergement possible :
Machine virtuelle, container Docker, ou cloud privé.

1.5. Conclusion

Le système pourra exploiter **PostgreSQL ou MongoDB** selon la structure des données audio.
L'environnement de développement s'appuiera sur **GitHub/GitLab pour le code**, **serveur local pour le test**, et éventuellement **AWS pour le déploiement final**.
Ces choix offrent une architecture fiable, évolutive, et collaborative pour le bon déroulement du projet.

1.6. Outils d'analyse de données

L'analyse des données audio est une étape fondamentale dans notre projet, car elle permet de transformer des signaux sonores bruts en **données exploitables** pour l'entraînement de modèles d'intelligence artificielle. Cette phase comprend la structuration des données, leur transfert sécurisé, et le choix d'architectures de machine learning adaptées aux caractéristiques temporelles et fréquentielles du signal vocal.

1.6.1. Mise en forme des données sonores

Avant l'entraînement, les données audio doivent être **prétraitées et converties** dans un format adapté :

Extraction de caractéristiques acoustiques : **MFCCs** (Mel Frequency Cepstral Coefficients), **Spectrogrammes**, **Chroma features**, etc.

Normalisation des durées ou intensités.

Segmentation de séquences pertinentes (cris, mots-clés...).

Encodage dans des formats compatibles avec les frameworks IA : .npy, .csv, ou TFRecord (pour TensorFlow).

Bibliothèques utiles :

- librosa : pour l'extraction audio en Python.

- torchaudio : pour le traitement audio avec PyTorch.
- soundfile : pour lire/écrire les fichiers audio.

1.6.2. Transfert sécurisé et connecté des données vers le serveur

Une fois les données extraites et structurées, elles doivent être **transmises à un serveur de traitement** :

- **Connexion API REST** : via Flask, FastAPI, ou Node.js pour recevoir les fichiers audio et stocker leurs méta-informations.
- **Connexion locale directe** : si l'analyse est effectuée à bord (edge computing) avant synchronisation.
- **Sécurité** : authentification par tokens (JWT), transferts chiffrés HTTPS.
- **Optimisation** : compression des fichiers audio, découpage par lots, base de données adaptée (MongoDB ou PostgreSQL avec liens aux fichiers sur S3).

Modèles d'Pour entraîner notre système à reconnaître des agressions verbales, nous utilisons des **modèles d'apprentissage automatique spécialisés dans le traitement du son**, capables de capter les variations de fréquence, d'intensité et de ton dans les cris ou insultes.

1.6.3. Modèles d'analyse des données audio

a. CNNs (Convolutional Neural Networks)

Utilité : très efficace pour extraire des caractéristiques spatiales sur les spectrogrammes.

Exemple : classification de cris à partir d'un spectrogramme.

https://www.tensorflow.org/tutorials/audio/simple_audio?hl=fr

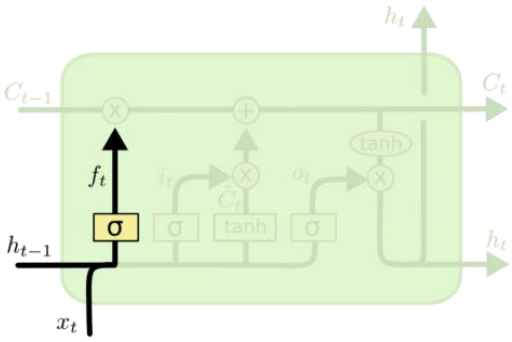
b. LSTM (Long Short-Term Memory)

Utilité : capte les dépendances temporelles dans un signal, idéal pour les séquences vocales longues.
Application : reconnaissance de phrases agressives, ou ton de la voix.

https://towardsdatascience.com/using-lstm-in-twitter-sentiment-analysis-a5d9013b523b/

https://towardsdatascience.com/neural-networks-for-real-time-audio-stateful-lstm-b534babeae5d/

https://github.com/Alec-Wright/CoreAudioML/blob/823a4727f4578aa434e715eae302e0e930576074/dataset.py



$$f_t = \sigma\left(W_f \cdot [h_{t-1}, x_t] \; + \; b_f\right)$$

c. ANNs (Artificial Neural Networks)

Utilité : modèles simples mais efficaces pour des jeux de données bien formatés (vecteurs MFCC...).
Limite : moins performants que CNN ou LSTM sur les séquences complexes.

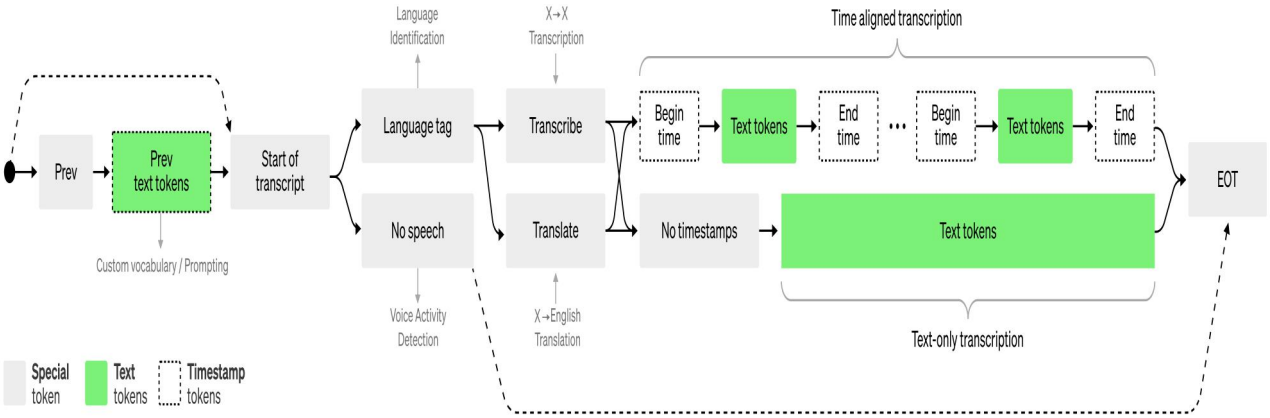
https://scikit-learn.org/stable/modules/neural_networks_supervised.html

d. CRNN (Convolutional Recurrent Neural Network)

Utilité : combine les avantages des CNN (extraction spatiale) et LSTM (séquence temporelle). Performant pour les tâches de détection de cris ou anomalies vocales.

https://arxiv.org/abs/1609.04243

e. Transformers audio (e.g., Wav2Vec 2.0, Whisper)



Utilité : architectures basées sur les Transformers pour transcrire ou classifier la parole directement.
Avantages : performances SOTA (state-of-the-art), peu de prétraitement nécessaire.

https://openai.com/index/whisper/

https://github.com/openai/whisper/blob/main/notebooks/LibriSpeech.ipynb

f. SVM (Support Vector Machine)

Utilité : modèle supervisé classique, très bon pour les petits jeux de données et la classification linéaire.
Application : utilisé dans certains projets de détection de cris en situation d'urgence.

https://scikit-learn.org/stable/modules/svm.html

g. Autoencoders / VAE

Utilité : apprentissage non supervisé pour la détection d'anomalies sonores ou la compression de données audio.
Application : détection automatique de sons suspects ou anormaux par reconstruction.

https://arxiv.org/abs/1711.00520

IV. Analyse et mise en place d une solution

Cette section vise à proposer une architecture fonctionnelle complète pour la détection d'agressions verbales à l'aide de capteurs sonores, d'un traitement intelligent du signal audio et d'un système décisionnel basé sur des modèles d'intelligence artificielle. L'ensemble du système s'articule autour de plusieurs modules interconnectés, depuis la détection sonore jusqu'à la prise de décision automatisée.

1. Analyse et proposition de solution

1.1. Analyse initiale

Le problème que nous cherchons à résoudre est la détection automatique d'agressions verbales dans des environnements publics (rue, transport, etc.). L'enjeu majeur réside dans la capacité à capter un signal sonore pertinent (cris, hausse de voix, etc.), à le traiter en temps réel, et à évaluer son niveau de dangerosité.

1.2. Schéma de solution

Lorsqu'une agression verbale se produit, le système commence par détecter un son dont l'intensité dépasse un seuil prédéfini (en dB) à l'aide de capteurs acoustiques. Ce signal est ensuite amplifié par un module d'amplification analogique pour améliorer la qualité d'analyse.

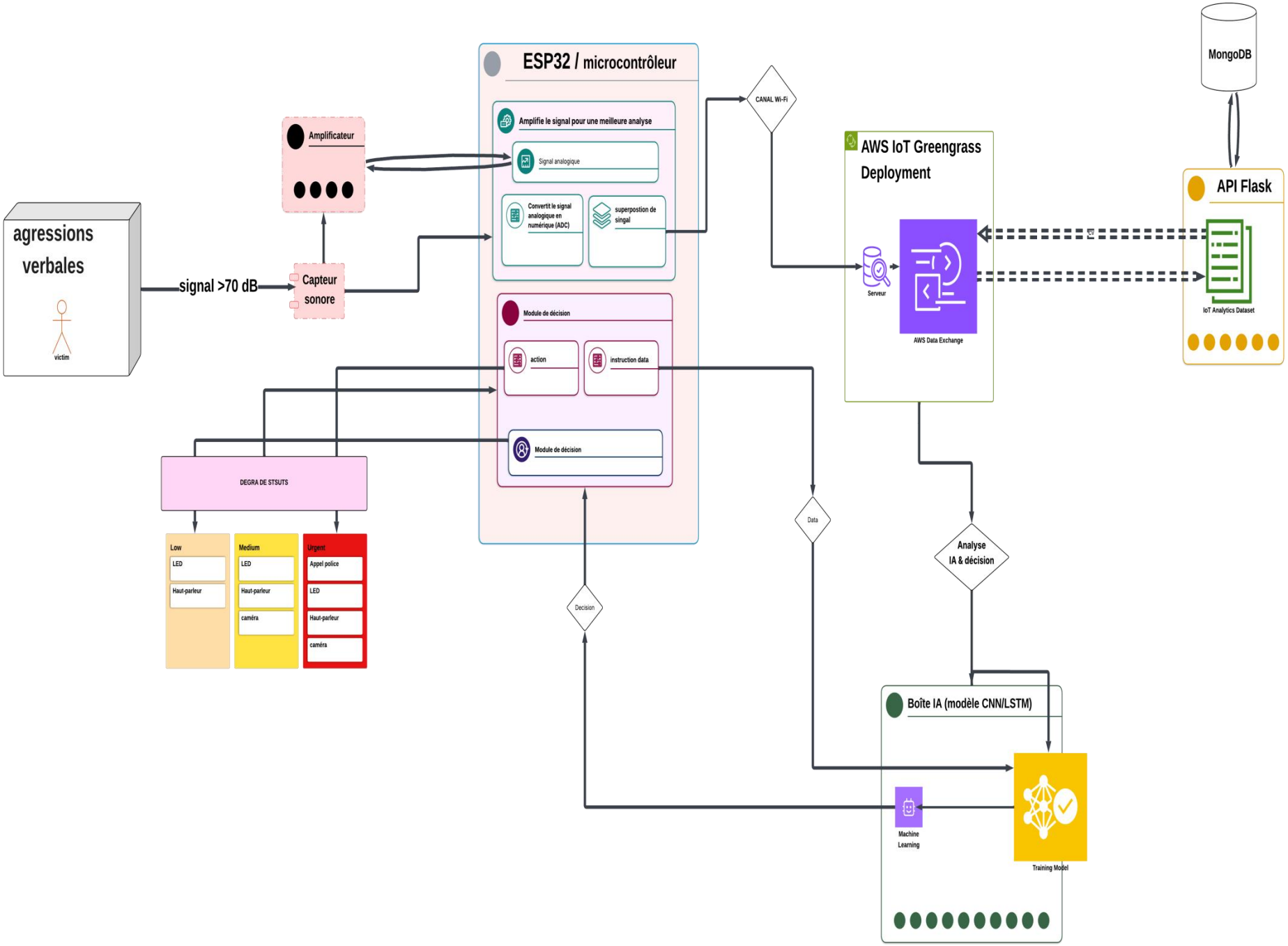
Ensuite, ce signal est transféré vers un microcontrôleur (ESP32 ou Arduino) doté de connectivité sans fil (Wi-Fi ou Bluetooth). Ce microcontrôleur transmet les données vers un serveur distant ou local. Le signal est alors pré-traité pour être converti en données numériques exploitables.

Ces données sont ensuite analysées par un modèle d'intelligence artificielle hébergé sur le serveur. Le modèle estime la probabilité qu'il s'agisse d'une agression verbale réelle. En fonction de cette évaluation, une ou plusieurs actions sont déclenchées :

- ✓ Allumage d'un voyant LED,
- ✓ Emission d'un signal sonore via haut-parleur,
- ✓ Déclenchement d'un enregistrement vidéo,
- ✓ Envoi d'un message ou appel automatique vers les autorités locales.

https://lucid.app/lucidchart/b16608ea-8dcb-4c9f-8736-c57f6ade20b0/edit?viewport_loc=-691%2C-1598%2C4034%2C1598%2C0_0&invitationId=inv_dac62dfb-d4fd-46a9-ae1b-15470eb7e814

Schéma de solution



1.2.1. Schéma Partie backend

Composants principaux :

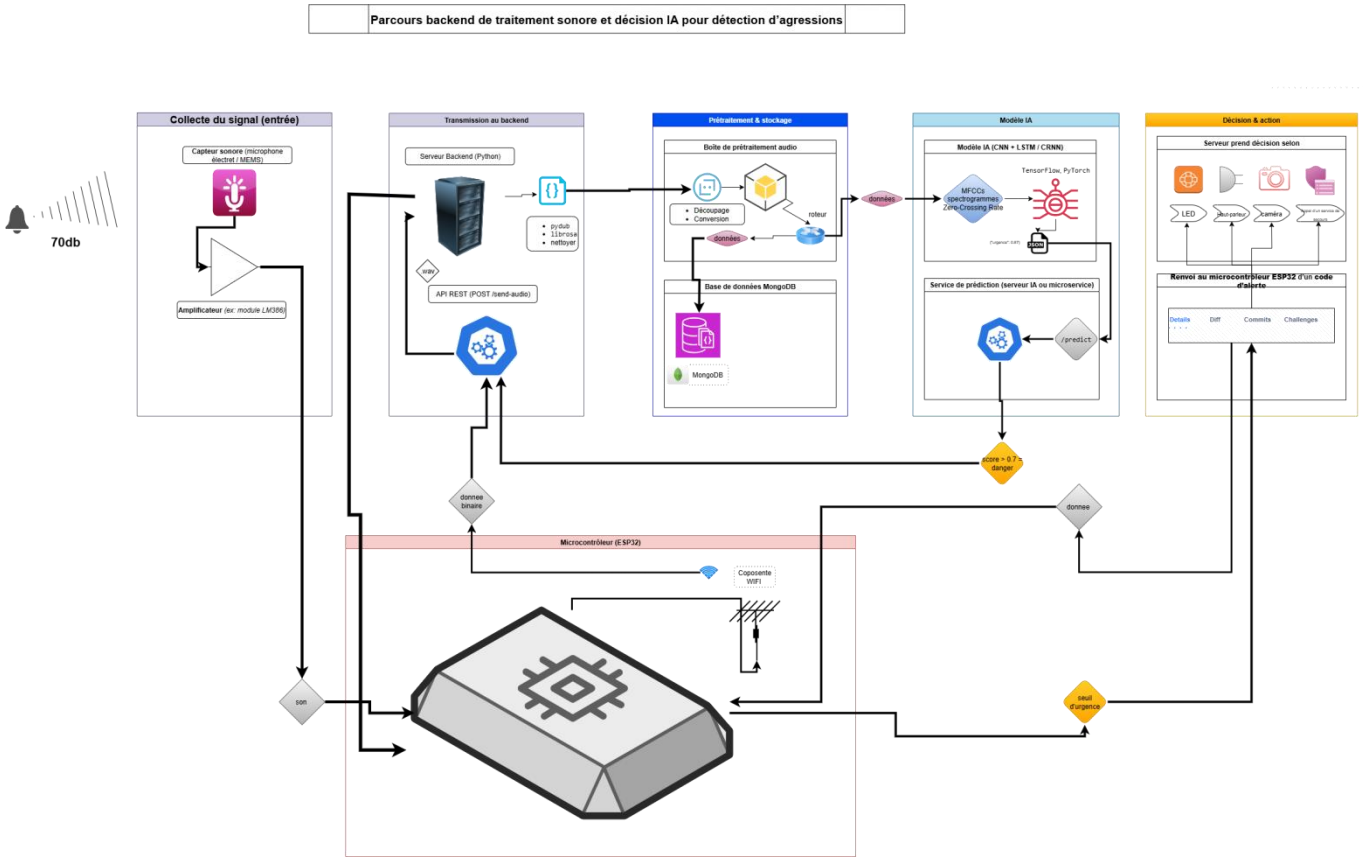
Serveur : (local ou AWS),

API REST : (Python + Flask ou FastAPI),

Base de données : MongoDB pour le stockage des événements détectés,

Environnement de traitement : (Docker ou Anaconda),

Librairies IA : TensorFlow, PyTorch.



Modèle IA utilisé :

CNN + LSTM + CRNN : pour combiner la reconnaissance de motifs spectrogrammiques et la dépendance temporelle.
Données entrantes sous forme de spectrogrammes log-mel ou MFCC.

1.2.2. Schéma Partie conception

Choix des composants :

Microprocesseur : ESP32, pour sa connectivité Wi-fi intégrée et sa faible consommation énergétique.
Capteurs audio : microphones électret ou MEMS, capables de détecter un signal >70 dB avec bonne sensibilité.

https://wokwi.com/projects/305569599398609473

Modules complémentaires :

Amplificateur audio (type MAX9814),
LED, haut-parleur, relais pour action de sortie,

1.3. Connexion entre capteurs et microprocesseur

L'intégration se fera via :

- Entrée analogique du Microcontrôleur pour le son
- Connexion série (UART, I2C) vers les modules complémentaires,
- Transmission via Wi-fi des données vers serveur/API.

1.3.1. Codage en C++ (ESP-IDF ou Arduino IDE)

Lecture des données audio via ADC.

Condition : si dB > seuil ⇒ capture + envoi via Wi-Fi.

API POST vers /analyse-son (traitée côté Python).

Gestion des GPIO pour action LED, haut-parleur, etc.

1.4. Bibliothèque et modèles

Exemples de bibliothèques à utiliser :

- **Pydub** / **Librosa** : traitement audio en Python,
- **Sounddevice** : enregistrement et lecture,
- **scikit-learn**, **Keras**, **TensorFlow**, **PyTorch** : entraînement IA,
- **OpenSMILE** : extraction d'attributs audio.

1.4.1. Analyse de précision via MATplotlib

MATLAB peut être utilisé pour la visualisation et la comparaison des performances du modèle (accuracy, F1-score, confusion matrix). On peut importer le modèle entraîné (.h5 ou .pb) et l'analyser avec des fonctions de classification.

1.5. Base de données

MongoDB : stocke les événements, les métadonnées (heure, lieu, intensité),

Intégration via **PyMongo**,

Test API : POST /event, **GET** /events?date=today.

1.6. Environnement de travail

GitHub : versioning et travail collaboratif,

AWS EC2 : déploiement du serveur,

Docker : pour contenir l'API + modèle IA,

VS Code / Jupyter pour développement local.

Conclusion du chapitre

Taw3 ba3ed hathi********** en 7 line max

1.7. Autre outil que nos aident

v. Bibliographie