République Tunisienne
Ministère de l'Enseignement Supérieur
et de la Recherche Scientifique
Université de Tunis
ensit Ecole Nationale Supérieure d'Ingénieurs de Tunis

	Mini Projet 1er SEMESTRE **A-U : 2024-2025**	**Date de création :**
	Classe : 3GInfo NTS **Matière : Machine Learning** **Code autorisé : les labs ipynb de nos TPs**	
Date de remise : 12/12/2024	Enseignant : Dr. Mehrez Boulares	
	Durée :3 sem **Barème :**	**Nbre de pages : 05**

Mini-Projet : Création d'un Système de Classification Audio Inspiré de Shazam à l'Aide des CNN

Contexte

Shazam est une application célèbre qui identifie des chansons à partir d'un court extrait audio. Ce projet vise à créer un modèle de classification audio similaire, capable de reconnaître des morceaux de musique ou des genres musicaux en utilisant des réseaux de neurones convolutionnels (CNN). L'accent sera mis sur la conversion des données audio en représentations adaptées, telles que des spectrogrammes, et l'entraînement d'un modèle CNN pour effectuer la classification.

Objectifs

1. **Extraction de caractéristiques :** Convertir les fichiers audio en spectrogrammes ou en représentations temps-fréquence comme les *Mel-spectrogrammes*.
2. **Construction d'un modèle CNN :** Concevoir un réseau de neurones convolutionnel adapté à l'analyse des spectrogrammes.
3. **Entraînement et évaluation :** Entraîner le modèle sur un ensemble de données audio (par exemple, une base de chansons ou de genres musicaux) et évaluer ses performances.
4. **Implémentation d'une application pratique :** Créer une interface utilisateur simple permettant de soumettre un extrait audio pour classification.

Données à Utiliser

- **Données disponibles :** Utilisez des bases de données publiques comme GTZAN Genre Dataset ou Freesound, etc…
- **Prétraitement :** Les fichiers audio seront transformés en spectrogrammes ou en *Mel-frequency cepstral coefficients* (MFCCs).

Tâches à Réaliser

1. **Analyse des données audio :** Étudier la structure des fichiers audio et comprendre les concepts de spectrogramme et MFCC.
2. **Prétraitement :**
    - Charger les fichiers audio avec des bibliothèques comme `Librosa` ou `PyDub`.
    - Générer des spectrogrammes ou MFCCs et les normaliser pour les rendre exploitables par les CNN.
3. **Conception du modèle CNN :**
    - Construire un CNN adapté à l'analyse des spectrogrammes.
    - Ajouter des couches pour capturer les caractéristiques pertinentes des données audio.
4. **Entraînement et Validation :**
    - Diviser les données en ensembles d'entraînement, de validation, et de test.
    - Utiliser une fonction de perte adaptée et optimiser le modèle.
5. **Mise en application :**
    - Implémenter une interface simple permettant de charger un fichier audio, générer son spectrogramme, et prédire la classe correspondante.

Livrables

1. Code complet du projet (prétraitement, modèle CNN, évaluation, et application).
2. Rapport détaillé comprenant :
    - La méthodologie utilisée.
    - Les résultats obtenus (précision, courbes de validation, etc.).
    - Les limites et améliorations possibles.
3. Une démonstration pratique (par exemple, une petite application ou un notebook interactif).

Outils Recommandés

- **Langage de programmation :** Python
- **Bibliothèques :**
    - Manipulation audio : `Librosa`, `Soundfile`, `PyDub`
    - Modélisation : `TensorFlow`, `Keras`, `PyTorch`
    - Visualisation : `Matplotlib`, `Seaborn`

Extensions (optionnelles)

1. Ajouter une détection des segments audio pertinents dans des enregistrements plus longs.
2. Étendre le projet à la reconnaissance d'artistes ou de locuteurs.
3. Entraîner le modèle sur des données de bruit pour tester sa robustesse.

Objectif final : Développer une solution fonctionnelle inspirée de Shazam, utilisant des CNN pour classer ou identifier des extraits audio de manière automatique et précise.

L'algorithme de **Shazam** repose sur une méthode efficace et robuste de reconnaissance audio. Il s'agit d'un système basé sur l'extraction et la comparaison de *fingerprints* (empreintes numériques) des fichiers audio. Voici un résumé des étapes principales de l'algorithme :

---

1. Prétraitement Audio

- **Conversion en signal mono** : Les données audio sont converties en un signal mono pour simplifier le traitement.
- **Échantillonnage** : Le signal est échantillonné à une fréquence standard (typiquement 44,1 kHz) pour garantir la compatibilité des comparaisons.

---

2. Transformation en Domaine Fréquentiel (Spectrogramme)

- **Fenêtrage** : Le signal audio est divisé en segments courts (fenêtres) de taille fixe avec un recouvrement (par exemple, 50%).
- **FFT (Fast Fourier Transform)** : Chaque segment est transformé pour obtenir un spectrogramme représentant les fréquences dominantes dans le temps.
- **Réduction des dimensions** : Le spectrogramme peut être réduit à des plages de fréquences pertinentes, comme les bandes *Mel* ou log-scaled.

---

3. Extraction de Points Caractéristiques (Audio Fingerprinting)

- L'algorithme identifie les **points forts** (ou *peaks*) dans le spectrogramme, qui correspondent aux amplitudes locales maximales dans le domaine fréquence-temps.
- Ces points forts sont des combinaisons de fréquences et de temps qui restent stables face à des variations comme le bruit ou la compression.

---

4. Création d'un Hash Unique

- Les points forts sont combinés en paires pour générer des **hashes** uniques.
- Chaque *hash* encode des informations comme :
    - La fréquence du point fort 1.
    - La fréquence du point fort 2.
    - Le décalage temporel entre ces deux points.
    - Ces informations sont concaténées en une chaîne qui sert de signature unique pour une partie spécifique de l'audio.

5. Création d’une Base de Données de Fingerprints

- Les fichiers audio dans la base de données de Shazam sont prétraités pour générer leurs *fingerprints*.
- Chaque *fingerprint* est indexé avec une référence au fichier audio correspondant.

---

6. Recherche et Correspondance

- Lorsqu'un utilisateur soumet un extrait audio :
  - Les *fingerprints* de cet extrait sont générés à l’aide des étapes précédentes.
  - Ces *fingerprints* sont comparés à ceux stockés dans la base de données.
- Shazam utilise des techniques d'indexation rapide (comme des tables de hachage) pour effectuer cette comparaison efficacement.

---

7. Calcul des Correspondances

- Les correspondances sont analysées pour trouver la meilleure correspondance globale.
- Un score est attribué à chaque correspondance basée sur :
  - Le nombre de *hashes* communs entre l'extrait audio et un fichier de la base.
  - La cohérence temporelle des correspondances.

---

8. Retour du Résultat

- Une fois une correspondance trouvée, l’algorithme retourne le titre, l’artiste, et d’autres informations pertinentes.

Caractéristiques Clés de l’Algorithme

- **Robustesse** : Les points forts sont peu affectés par le bruit, les compressions ou les distorsions.
- **Efficacité** : Les *hashes* permettent une recherche rapide dans des bases de données massives.
- **Évolutivité** : Le système peut être étendu à des millions de morceaux en optimisant l'indexation.

Visualisation Simplifiée de l’Audio Fingerprint

Voici une illustration des étapes principales :

1. Un spectrogramme est généré à partir d'un fichier audio.
2. Les points forts sont extraits comme des pics d'énergie.
3. Les points sont reliés pour créer des combinaisons codées sous forme de *hashes*.

Ce système est efficace parce qu'il se concentre sur des éléments très spécifiques et stables de l'audio, réduisant ainsi la complexité des comparaisons et augmentant la précision.