**Reconnaissance Acoustique en Milieu Urbain pour Personnes Malentendantes**

**DEPOORTERE Antoine, RAJAONA Cbastien**

<https://github.com/Cbastienr/BirdCLEF24_Sub/blob/main/Dataset.py>

Epitech Digital School, Paris - France

## 1. Introduction

Les environnements urbains, riches en diversité sonore, représentent un défi particulier pour les personnes malentendantes en raison de la multitude de bruits ambiants et de signaux acoustiques. La capacité à identifier et à discriminer ces sons est essentielle pour la sécurité et la qualité de vie. Pour répondre à cette problématique, ce projet vise à développer un système de reconnaissance acoustique capable de détecter et d'identifier divers sons urbains. Financé par le "Ministère du bien vivre en bonne santé", ce projet s'appuie sur des techniques avancées de deep learning, en particulier le framework PyTorch, afin de créer un modèle performant de reconnaissance sonore4. L'objectif est d'intégrer ce modèle dans des appareils auditifs, améliorant ainsi leur efficacité et fournissant un soutien auditif amélioré aux personnes malentendantes.

## 2. Objectif

L'objectif principal de cette étude est de développer un modèle de reconnaissance sonore capable de classer efficacement divers sons urbains à partir du dataset UrbanSound8K1. Ce modèle sera ensuite étendu pour inclure la reconnaissance des chants d'oiseaux à l'aide du dataset BirdCLEF 20248, permettant ainsi une application plus large dans le domaine de la surveillance de la biodiversité.

Une image contenant texte, carte, capture d’écran

Description générée automatiquement

Pour atteindre ces objectifs, nous allons d'abord nous concentrer sur le développement et l'entraînement du modèle en utilisant PyTorch pour créer un réseau de neurones convolutifs (CNN)3 destiné à la classification des sons urbains. Une évaluation rigoureuse des performances du modèle sera mise en place par le biais de techniques de validation croisée, afin de mesurer la précision, le rappel et le F1-score du modèle. Une fois que le modèle de base sera établi et testé, nous l'adapterons et l'étendrons pour inclure la reconnaissance des chants d'oiseaux en utilisant le dataset BirdCLEF 2024. Cette extension vise à diversifier les capacités du modèle et à démontrer son applicabilité dans des contextes variés.

En parallèle, nous explorerons l'intégration du modèle sur une plateforme embarquée ESP32. Cette étape impliquera l'optimisation du modèle pour fonctionner efficacement sur un matériel à ressources limitées, permettant ainsi une détection sonore en temps réel. Enfin, toutes les méthodologies utilisées, ainsi que les résultats obtenus, seront minutieusement documentées et feront l'objet d'une publication scientifique. Cet article de recherche visera à partager nos découvertes et à contribuer aux connaissances existantes dans le domaine de la reconnaissance sonore.

## 3. Méthodologie

### 3.1 Initiation avec un Classifieur Acoustique avec PyTorch

Le dataset UrbanSound8K contient 8732 extraits sonores répartis en dix catégories : air\_conditioner, car\_horn, children\_playing, dog\_bark, drilling, engine\_idling, gun\_shot, jackhammer, siren, et street\_music1. Les fichiers audios, au format WAV, sont organisés en 10 plis pour une validation croisée rigoureuse, ce qui facilite l’évaluation de la performance des modèles de classification acoustique de manière robuste et fiable.

Pour classifier ces sons urbains, un modèle de réseau de neurones convolutifs (CNN) a été développé en utilisant PyTorch4. Les techniques de prétraitement des données, telles que la transformation de Fourier et les spectrogrammes, ont été employées pour convertir les données audios en formats utilisables par les modèles de deep learning2. Ces transformations permettent de représenter les caractéristiques temporelles et fréquentielles des sons, essentielles pour l’analyse par les réseaux de neurones.

L’architecture du modèle CNN que nous avons conçue se compose de plusieurs blocs de convolution suivis de couches de pooling, de flattening (aplatissement) et de couches linéaires. Les blocs de convolution sont utilisés pour extraire des caractéristiques importantes des données audio. Chaque bloc comprend une couche de convolution, une fonction d'activation ReLU (Rectified Linear Unit) et une couche de pooling. La couche de convolution applique des filtres (ou noyaux) à l'entrée pour détecter des motifs spécifiques dans les données. Notre modèle comprend quatre couches de convolution avec des nombres croissants de filtres (16, 32, 64, 128), chacune ayant une taille de noyau de 3x3. La fonction d'activation ReLU introduit de la non-linéarité dans le modèle, ce qui permet au réseau de mieux apprendre des motifs complexes. La couche de MaxPooling réduit la dimensionnalité des données en prenant la valeur maximale dans chaque fenêtre de taille spécifiée (2x2 dans notre cas), ce qui permet de réduire le nombre de paramètres et de contrôler le surapprentissage3.

Après les couches de convolution et de pooling, les données sont aplaties en un vecteur 1D, préparant ainsi les données pour les couches entièrement connectées (linéaires). La couche linéaire finale prend le vecteur aplati et produit les sorties du modèle. Dans notre cas, nous avons une seule couche linéaire avec 10 sorties correspondant aux 10 classes de sons dans le dataset UrbanSound8K. La fonction softmax est utilisée à la fin pour convertir les scores de la couche linéaire en probabilités, facilitant ainsi l'interprétation des résultats. En résumé, cette architecture est conçue pour identifier les motifs dans les données audio et classer les sons en utilisant une combinaison de convolutions, d'activations non linéaires, de pooling et de couches linéaires. Elle permet de capturer et d'analyser les caractéristiques complexes des sons urbains, offrant ainsi une solution robuste pour la reconnaissance sonore.

#### Hypothèses

Nous postulons que le modèle CNN développé à l'aide de PyTorch sera capable de classifier les sons urbains avec une précision supérieure à 80% sur le dataset UrbanSound8K. Cette hypothèse repose sur la robustesse des architectures de réseaux de neurones convolutifs pour la reconnaissance de motifs complexes dans les données audio3. Nous anticipons que la technique de validation croisée en 10 plis utilisée avec le dataset UrbanSound8K permettra au modèle de généraliser efficacement à des données non vues, minimisant ainsi le risque de surapprentissage. Cette méthodologie rigoureuse devrait offrir une évaluation précise des performances du modèle sur des ensembles de données diversifiés. Les techniques de prétraitement des données, telles que la transformation de Fourier et les spectrogrammes, offriront des représentations suffisamment riches des caractéristiques temporelles et fréquentielles des sons. Ces représentations sont essentielles pour permettre une classification précise par le modèle CNN, en capturant les nuances des signaux audio2. En ce qui concerne l'architecture du modèle, nous prévoyons que les quatre couches de convolution avec des nombres croissants de filtres (16, 32, 64, 128) permettront de capturer des motifs de plus en plus complexes dans les données audio, améliorant ainsi la performance de classification. Cette hiérarchie croissante de filtres est conçue pour identifier des caractéristiques de plus en plus abstraites. L'utilisation de la fonction d'activation ReLU (Rectified Linear Unit) devrait introduire une non-linéarité suffisante dans le modèle pour qu'il puisse apprendre des motifs complexes. Cette capacité à introduire de la non-linéarité est cruciale pour améliorer la précision de la classification. De plus, les couches de MaxPooling réduiront efficacement la dimensionnalité des données sans perte significative d'information. Cette réduction de la dimensionnalité aidera à contrôler le surapprentissage et à améliorer la robustesse du modèle, en conservant les caractéristiques les plus importantes des données. Enfin, nous anticipons que la couche linéaire finale avec 10 sorties sera adéquate pour produire des probabilités correctes pour chaque catégorie de sons, grâce à l'utilisation de la fonction softmax. Cette étape finale facilitera l'interprétation des résultats et devrait permettre une classification précise des différentes classes sonores. Nous espérons également que l'architecture CNN proposée sera suffisamment optimisée pour permettre une classification sonore en temps réel, ouvrant ainsi des possibilités d'applications pratiques en environnements urbains. Ces hypothèses, une fois testées, permettront de valider ou d'ajuster notre approche pour améliorer le modèle et obtenir des résultats optimaux pour la reconnaissance sonore urbaine.

### 3.2 Développement avec un Jeu de Données Supplémentaire

Le dataset BirdCLEF 2024 comprend des enregistrements audios de chants d'oiseaux provenant des Ghâts occidentaux en Inde, une région reconnue pour sa riche biodiversité mais également soumise à des changements environnementaux rapides8. Le défi de ce projet consiste à classifier automatiquement les espèces d'oiseaux à partir de ces enregistrements sonores, permettant ainsi de mieux comprendre et surveiller la biodiversité dans cette région.

L'approche technique repose sur l'utilisation de techniques avancées de traitement du signal et de réseaux de neurones, notamment les réseaux convolutifs (CNN) et récurrents (RNN). Ces techniques seront mises en œuvre pour développer un modèle capable de reconnaître les espèces d'oiseaux à partir de leurs chants3. La validation croisée et l'optimisation des hyperparamètres seront essentielles pour améliorer la précision et la robustesse du modèle, en garantissant une performance élevée sur des données inédites. Pour le prétraitement et l'enrichissement des données, les enregistrements audios seront transformés en spectrogrammes, qui sont des représentations visuelles des fréquences sonores au fil du temps. Ces spectrogrammes serviront d'entrée aux modèles de deep learning. En outre, des techniques de data augmentation, telles que l'ajout de bruit, le changement de vitesse et le pitch shifting, seront utilisées pour augmenter la robustesse du modèle9. Ces techniques permettent de simuler différentes conditions d'enregistrement et de rendre le modèle plus résistant aux variations naturelles des chants d'oiseaux.

### 3.3 Développement Embarqué sur ESP32

Pour optimiser le modèle PyTorch afin qu'il fonctionne efficacement sur un microcontrôleur ESP32, il est essentiel d'adapter le modèle en tenant compte des ressources limitées de ce type de matériel. Cela implique de réduire la complexité du modèle tout en maintenant une précision acceptable. Des techniques comme la quantification et la simplification des architectures de réseaux de neurones peuvent être utilisées pour atteindre cet objectif10. L'intégration matérielle nécessite la configuration et la connexion d'un microphone à l'ESP32 pour permettre la détection sonore en temps réel. Cela inclut le choix d'un microphone compatible, l'établissement des connexions nécessaires, et la mise en place du logiciel de traitement du signal sur l'ESP32 pour capturer et prétraiter les données audios avant de les transmettre au modèle de reconnaissance. Le développement et les tests impliquent la conversion du modèle initialement développé en PyTorch au format TensorFlow Lite, compatible avec l'ESP3210. Une fois converti, le modèle sera déployé sur l'ESP32, et des tests de performance seront réalisés pour évaluer la précision et la réactivité du modèle en conditions réelles. Ces tests permettront d'identifier et de corriger les éventuels problèmes de latence et de précision, assurant ainsi que le système fonctionne de manière fiable et efficace dans des situations pratiques.

### 3.4 Implémentation du modèle SSAST (Self-Supervised Audio Spectrogram Transformer)

L'implémentation du modèle SSAST pour l'analyse des signaux audio a été effectuée en vue d'évaluer ses performances dans le cadre de la compétition Kaggle BirdCLEF 20247. Le modèle SSAST, spécialisé dans la reconnaissance de motifs audio complexes, a été adapté et entraîné sur les enregistrements de chants d'oiseaux fournis par le dataset BirdCLEF 2024, ainsi que sur le dataset UrbanSound8K pour assurer une base de comparaison robuste. Le modèle SSAST (Self-Supervised Audio Spectrogram Transformer) est une architecture avancée de deep learning conçue pour l'analyse et la reconnaissance de motifs audio à partir de spectrogrammes7.

Une image contenant capture d’écran, Tracé, ligne

Description générée automatiquement

Pendant l'entraînement, le modèle est soumis à des tâches d'apprentissage auto-supervisé telles que la prédiction de patchs masqués du spectrogramme ou l'identification de la séquence temporelle des patches. Ces tâches aident le modèle à apprendre des représentations riches des données audio. Après l'apprentissage auto-supervisé, le modèle sera ajusté pour classer les chants d'oiseaux. Pour évaluer les performances du modèle SSAST, des comparaisons sont effectuées avec les modèles développés précédemment. Ces comparaisons se concentrent sur des métriques clés telles que la précision, le rappel et le F1-score. Les évaluations sont menées sur les jeux de données UrbanSound8K et BirdCLEF afin de déterminer l'efficacité du modèle SSAST dans différents contextes et types de données sonores. Les résultats de l'implémentation et de l'évaluation du modèle SSAST seront minutieusement documentés. Un article de recherche sera rédigé pour détailler les méthodologies employées, les résultats obtenus, et les conclusions tirées de ces expériences. Cette documentation comprendra une analyse approfondie des performances du modèle, des graphiques illustrant les résultats comparatifs, et des discussions sur les implications des découvertes pour la reconnaissance sonore et la classification des chants d'oiseaux. La publication de cet article visera à contribuer aux connaissances existantes dans le domaine et à partager les avancées réalisées au cours de ce projet.

## 4. Expériences

### 4.1 Préparation des Données

Pour préparer les données audios, nous avons utilisé des techniques de transformation pour convertir les enregistrements en spectrogrammes. Le dataset BirdClef2024 a été transformé en utilisant la transformation MelSpectrogram de torchaudio pour extraire des caractéristiques pertinentes des signaux audio2.

### 4.2 Entraînement du Modèle

Le modèle CNN développé a été entraîné en utilisant les données prétraitées. Nous avons utilisé des techniques de régularisation comme le dropout pour éviter le surapprentissage et avons optimisé les hyperparamètres pour améliorer la performance du modèle3.

## 4.3 Évaluation

Le modèle a été évalué en utilisant une validation croisée sur 10 plis pour garantir la robustesse des résultats. Les métriques d'évaluation incluent la précision, le rappel et le F1-score.

## 5. Résultats

Epoch 1/10, Loss: 5.1904

Epoch 2/10, Loss: 5.1872

Epoch 3/10, Loss: 5.1862

Epoch 4/10, Loss: 5.1923

Epoch 5/10, Loss: 5.1923

Epoch 6/10, Loss: 5.1930

Epoch 7/10, Loss: 5.1929

Epoch 8/10, Loss: 5.1933

Epoch 9/10, Loss: 5.1931

Epoch 10/10, Loss: 5.1921

Predictions: [44, 62, 44, 62, 44, 44, 44, 62, 62, 44]

Actuals: [88, 59, 2, 59, 24, 132, 90, 76, 44, 33]

## 6. Discussions

#### Interprétation des Pertes Constantes

Les pertes élevées et constantes suggèrent que le modèle n'apprend pas efficacement. Ce phénomène peut être attribué à plusieurs facteurs. Premièrement, un taux d'apprentissage inapproprié, qu'il soit trop faible ou trop élevé, peut empêcher le modèle de converger correctement. Un taux trop faible ralentit l'apprentissage, tandis qu'un taux trop élevé peut conduire à des oscillations et empêcher le modèle de trouver un minimum optimal.

Il peut être intéressant de segmenter les données, pour diminuer la durée des epochs et entraîner plus rapidement notre modèle.

En outre, l'architecture du modèle pourrait ne pas être adaptée à la tâche en question. Par exemple, le modèle pourrait nécessiter des ajustements au niveau de ses hyperparamètres, tels que le nombre de filtres, la taille des couches ou même l'ajout de couches supplémentaires pour améliorer sa capacité à capturer les motifs complexes des données. Les réseaux de neurones convolutionnels (CNN) nécessitent souvent des ajustements fins de leurs paramètres pour atteindre des performances optimales. Il est également important d'envisager l'utilisation de techniques de régularisation, comme le dropout, pour éviter le surajustement et améliorer la généralisation du modèle.

#### Analyse des Prédictions

Les prédictions fournies par le modèle ne correspondent pas adéquatement aux valeurs réelles, ce qui indique une faible performance du modèle. Cela peut être le résultat d'un sous-ajustement, où le modèle est trop simple pour capturer les complexités des données d'entraînement. Un modèle sous-ajusté n'est pas capable d'apprendre suffisamment des données pour faire des prédictions précises. Cette situation nécessite souvent des ajustements dans la complexité du modèle ou une augmentation de la quantité et de la qualité des données d'entraînement.

Il est également possible que le modèle ne soit pas correctement optimisé. L'utilisation d'optimiseurs comme Adam peut aider à trouver un minimum global, mais il est crucial d'ajuster les hyperparamètres tels que le taux d'apprentissage pour s'assurer que le modèle converge correctement. En outre, l'application de techniques de normalisation des données, comme la normalisation des spectrogrammes, est essentielle pour s'assurer que les entrées du modèle sont dans une gamme appropriée, ce qui facilite l'apprentissage.

## 7. Conclusion

Ce projet représente une opportunité unique d'appliquer des techniques avancées de deep learning à des problèmes réels d'acoustique urbaine, avec un impact direct sur la qualité de vie des personnes malentendantes. En participant à des compétitions internationales et en explorant des domaines tels que l'intégration matérielle et la recherche académique, les étudiants pourront acquérir des compétences précieuses et contribuer à des innovations significatives dans le domaine de la reconnaissance sonore.

## Références

1. Salamon, J., Jacoby, C., & Bello, J. P. (2014). "A Dataset and Taxonomy for Urban Sound Research." In *Proceedings of the 22nd ACM International Conference on Multimedia* (pp. 1041-1044). ACM.
2. Goodfellow, I., Bengio, Y., & Courville, A. (2016). *Deep Learning*. MIT Press.
3. Krizhevsky, A., Sutskever, I., & Hinton, G. E. (2012). "Imagenet classification with deep convolutional neural networks." In *Advances in neural information processing systems* (pp. 1097-1105).
4. Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., ... & Chintala, S. (2019). "PyTorch: An Imperative Style, High-Performance Deep Learning Library." In *Advances in Neural Information Processing Systems* (pp. 8024-8035).
5. Jing, L., & Tian, Y. (2020). "Self-supervised visual feature learning with deep neural networks: A survey." In *IEEE Transactions on Pattern Analysis and Machine Intelligence*.
6. Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., ... & Polosukhin, I. (2017). "Attention is all you need." In *Advances in Neural Information Processing Systems* (pp. 5998-6008).
7. Gong, Y., Chung, Y.-A., & Glass, J. (2021). "SSAST: Self-Supervised Audio Spectrogram Transformer." In *Proceedings of the 2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)* (pp. 7218-7222).
8. Joly, A., Goëau, H., Botella, C., Glotin, H., Vellinga, W.-P., Planqué, R., ... & Bonnet, P. (2016). "LifeCLEF 2016: multimedia life species identification challenges." In *Proceedings of CLEF* (pp. 286-310).
9. Shorten, C., & Khoshgoftaar, T. M. (2019). "A survey on image data augmentation for deep learning." In *Journal of Big Data, 6(1), 60*.
10. Al-Rfou, R., Alain, G., Almahairi, A., Angermueller, C., Bahdanau, D., Ballas, N., ... & Bengio, Y. (2016). "Theano: A Python framework for fast computation of mathematical expressions." In *arXiv preprint arXiv:1605.02688*.