

الجمهورية الجزائرية الديمقراطية الشعبية  
République Algérienne démocratique et populaire

وزارة التعليم العالي و البحث العلمي  
Ministère de l'enseignement supérieur et de la recherche scientifique

جامعة سعد دحلب البليدة  
Université SAAD DAHLAB de BLIDA

كلية التكنولوجيا  
Faculté de Technologie

قسم الإلكترونيك  
Département d'Électronique



## Mémoire de Master

Filière: Électronique  
Spécialité: Instrumentation

présenté par:

**Manel MORSLI**

&

**Bouchra SAIFI**

# Détection d'anomalies pour les bruits de machine

Proposé par: **Mr. Farid YKHLEF**

Année Universitaire: 2022-2023

Nous tenons à remercier au premier lieu **ALLAH**, le tout puissant qui nous a donné la force, la patience et la volonté pour terminer ce travail.  
Alhamdulillah.

Nous voudrions vraiment remercier notre promoteur Mr. **YKHLEF**, pour ses avis compétents, sa disponibilité, son soutien, et sa patience. Nous sommes très reconnaissantes de tout ce que vous avez fait.

Nos remerciements vont aussi à toutes celles et ceux qui ont contribué de près ou de loin à la réalisation de ce modeste travail.

Avec amour et émotion, je souhaite dédier ces quelques mots à mes êtres chers qui ont accompagné chaque étape de mon parcours, et qui ont forgé la personne que je suis aujourd'hui.

À ma chère maman, **Samia GACEM**, mon ange qui, bien que partie de ce monde l'année passée, continue de briller dans mon cœur comme une étoile éternelle. Tes encouragements, ton amour inconditionnel et ta bienveillance ont façonné l'individu que je suis devenu. Ce mémoire est dédié à toi, ma source d'inspiration éternelle.

À mon cher père, **Toufik MORSLI**, même si tu n'es plus parmi nous physiquement, ton essence reste gravée en moi, éternelle et unique. Tu étais mon guide, mon héros et mon exemple de force inébranlable. Cette dédicace est un hommage vibrant à ton amour inconditionnel et à l'héritage précieux que tu as laissé derrière toi.

À mon frère précieux, **Merouane**, tu as été bien plus qu'un simple frère pour moi. Tu as été mon pilier de confiance, mon bras droit et mon soutien inconditionnel. À travers les hauts et les bas de cette aventure, ta présence solide a été ma source de réconfort.

Mes chers petits frères, **Mazen et Mohamed**, vous êtes les joyaux de ma vie. Vos sourires lumineux et votre innocence pure ont illuminé mon parcours. Même si vous êtes encore jeunes, votre soutien et votre enthousiasme m'ont motivé à donner le meilleur de moi-même.

À mes chères amies, **Khaoula et Ghizlane**, votre amitié a été un cadeau précieux tout au long de ce parcours. Votre soutien inébranlable ont illuminé mes journées les plus sombres.

À mon binôme, **Bouchra**, nous avons formé une équipe harmonieuse et complémentaire, surmontant les obstacles avec détermination et créativité. Ta présence et ton engagement ont été essentiels à notre réussite commune.

À mon amie chère, **Meriem**, cette dédicace est une marque de reconnaissance et d'affection envers notre amitié précieuse.

À ma précieuse famille **MORSLI et GACEM**, cette dédicace est un hommage vibrant à l'amour et au soutien indéfectibles que vous m'avez prodigués tout au long de ma vie.

**Manel**

Je dédie cet ouvrage

À mes très chers parents qui m'ont toujours poussé et motivé dans mes longues études. Ce projet de fin d'études représente donc l'aboutissement du soutien et des encouragements qu'ils m'ont prodigués tout au long de ma scolarité.

C'est un moment de plaisir de dédier cet ouvrage à mes belles-soeurs: **Fatouma** et sa petite fille **Maria**, **Nesrine** et mon frère **Mohamed Yacine**.

Et finalement, à mes amis qui n'ont jamais cessé de me soutenir, spécifiquement mon binôme **Manel**, ta perspicacité et tes compétences ont enrichi notre travail et m'ont inspiré à donner le meilleur de moi-même.

Que Dieu vous donne santé, bonheur, courage et surtout réussite.

**Bouchra**

---

**ملخص:** الهدف من هذا المشروع هو الكشف عن الاختلالات في أصوات الآلات التي يمكن أن تكون مؤشرة جدا على مشاكل الأداء المحتملة.

نقترح منهجية تستخدم تقنيات معالجة الإشارات و خوارزمية التعلم الآلي لتحديد الأنماط الغير الطبيعية في بيانات أصوات الآلات تلقائيا.

**كلمات المفاتيح:** التعلم العميق, الاختلالات, الخصائص.

---

**Résumé:** L'objectif de ce projet était de détecter les anomalies dans les bruits de machine, qui peuvent être très indicatrices de problèmes potentiels de performance.

Nous proposons une méthodologie qui utilise des techniques de traitement du signal et des algorithmes du machine learning pour identifier automatiquement les motifs anormaux dans les données de bruit de machine.

**Mots clés:** Apprentissage profond, Anomalies, Caractéristiques.

---

**Abstract:** The goal of this project was to detect the anomalies in machine noises which can be very indicative of potential performance issues.

We propose a methodology that utilizes signal processing techniques and machine learning algorithms to automatically identify abnormal patterns in machine noise data.

**Keywords:** Deep learning, Anomalies, features.

---

## Listes des acronymes et abréviations

<i>ANN</i>	<b>Réseau neuronal artificiel</b>
<i>CNN</i>	<b>Réseau neuronal convolutif</b>
<i>DCT</i>	<b>Transformée en cosinus discrète</b>
<i>HMM</i>	<b>Modèles de Markov cachés</b>
<i>K-NN</i>	<b>Voisin le plus proche K</b>
<i>LIV</i>	<b>Soupape d'entrée de fuite</b>
<i>LOV</i>	<b>Soupape de sortie de fuite</b>
<i>LSTM</i>	<b>Mémoire à Long Terme-Court Terme</b>
<i>MSE</i>	<b>Erreur quadratique moyenne</b>
<i>MFCC</i>	<b>Coefficient Cepstraux en Fréquence Mel</b>
<i>MLP</i>	<b>Perceptron multicouche</b>
<i>NRV</i>	<b>Clapet anti-retour</b>
<i>OOB</i>	<b>Programmation orientée objet</b>
<i>PCA</i>	<b>Analyse des composants principaux</b>
<i>ReLU</i>	<b>Unité Linéaire Rectifiée</b>
<i>RNN</i>	<b>Réseau neuronal récurrent</b>
<i>RL</i>	<b>Apprentissage par renforcement</b>
<i>STFT</i>	<b>Transformée de Fourier à court terme</b>
<i>SGD</i>	<b>Descente de gradient stochastique</b>
<i>SL</i>	<b>Apprentissage supervisé</b>
<i>SVM</i>	<b>Machine à vecteurs de support</b>
<i>USL</i>	<b>Apprentissage sans surveillance</b>
<i>VAE</i>	<b>Encodeur automatique variationnel</b>

# Table des matières

Introduction générale .....	12
-----------------------------	----

## Chapitre 1: Notions de bases et concepts utiles

Introduction .....	15
1.1 Prise de son .....	15
1.1.1 Définition .....	15
1.1.2 Quelques techniques de prise de son .....	15
1.2 Représentation du signal audio .....	16
1.2.1 Représentations du signal audio .....	16
1.3 La transformée de Fourier (STFT) .....	18
1.4 L'extraction de caractéristiques .....	20
1.4.1 techniques de l'extraction de caractéristiques .....	20
1.4.2 Types de caractéristiques .....	21
1.5 La classification .....	21
1.5.1 Méthodes de classification .....	22
1.5.2 Algorithmes de classification .....	22
1.6 Évaluation .....	23
1.6.1 Quelques techniques de l'évaluation .....	24
1.6.2 Matrice de confusion .....	24
1.6.3 Métriques (Metrics) .....	25
Conclusion .....	26

## Chapitre 2: Deep Learning pour la surveillance de l'état des machines

Introduction .....	27
2.1 Bases du Machine learning .....	27
2.1.1 Définition .....	27
2.1.2 Types du machine learning .....	28
2.2 Deep learning .....	30
2.2.1 Définition .....	30
2.2.2 Deep neural networks .....	31
2.2.3 Le perceptron multicouche (MLP) .....	32
2.2.4 Fonction d'activation .....	34
2.2.5 Hyper-paramètres du réseau de neurones .....	37
2.2.6 Auto-encodeur .....	38
2.2.7 Convolutional Neural Networks (CNNs) .....	40
Conclusion .....	42

## Chapitre 3: Mise en place de la solution

Introduction .....	43
3.1 Python .....	43
3.1.1 Définition .....	43
3.1.2 Histoire et origines .....	43
3.1.3 Que nous offre Python ? .....	43
3.1.4 Programmation orientée objet (OOP) .....	44
3.1.5 Nature interprétée et interactive .....	45

3.1.6 Philosophie Open-source .....	45
3.2 Construire le modèle CNN .....	46
3.2.1 Collecte et prétraitement des données .....	46
3.2.2 Conception de l'architecture .....	47
3.2.3 Entraînement .....	48
3.2.4 Visualisation des résultats de L'entraînement .....	49
3.3 Fonctionnement et test .....	50
3.3.1 L'entraînement du CNN .....	50
3.3.2 L'évaluation du CNN .....	51
3.3.3 Résultat de la prédiction .....	53
Conclusion .....	58
Conclusion générale .....	59
Bibliographie .....	61

## Liste des figures

Figure 1.1. La représentation temporelle d'un signal audio	17
Figure 1.2. La représentation fréquentielle d'un signal audio	17
Figure 1.3. La représentation en spectrogramme d'un signal audio	18
Figure 1.4. Exemple de l'un des métriques d'évaluation (Accuracy)	26
Figure 1.5. Exemple de l'un des métriques d'évaluation (Precision)	26
Figure 2.1. Exemple en comment le SL fonctionne	28
Figure 2.2. Exemple en comment le USL fonctionne	29
Figure 2.3. Exemple en comment le RL fonctionne	29
Figure 2.4. Architecture d'un Deep neural network	31
Figure 2.5. Architecture d'un MLP	34
Figure 2.6. La courbe de la fonction Sigmoidé	35
Figure 2.7. La courbe de la fonction tangente hyperbolique	35
Figure 2.8. La courbe de la fonction ReLU	36
Figure 2.9. La courbe de la fonction Softmax	36
Figure 2.10. Exemple d'optimisations hyper-paramètres	37
Figure 2.11. Architecture d'un Auto-encodeur	39
Figure 2.12. Architecture d'un CNN	41
Figure 2.13. Architecture d'un CNN	42
Figure 3.1. Les classes de la Dataset	46
Figure 3.2. Prétraitement de données	46
Figure 3.3. Création du CNN	47
Figure 3.4. Compilation, l'entraînement et l'évaluation du modèle	48
Figure 3.5. Tracer les courbes de la perte et les métriques	49
Figure 3.6. L'entraînement du CNN (Epoch N° 00 $\Rightarrow$ Epoch N° 10)	50

Figure 3.7. L'entraînement du CNN (Epoch N° 11 ⇒ Epoch N° 20) .....	50
Figure 3.8. Les valeurs finales de métriques de l'évaluation .....	51
Figure 3.9. La courbe de la perte (Loss) .....	51
Figure 3.10. La courbe de l'exactitude (accuracy) .....	51
Figure 3.11. La courbe de la Précision (Precision) .....	52
Figure 3.12. La prédiction et le spectrogramme d'un Audio normal .....	53
Figure 3.13. La prédiction et le spectrogramme d'un Audio (une faute "Bearing") ..	53
Figure 3.14. La prédiction et le spectrogramme d'un Audio(une faute "Flywheel")..	54
Figure 3.15. La prédiction et le spectrogramme d'un Audio (une faute "LIV") .....	54
Figure 3.16. La prédiction et le spectrogramme d'un Audio (une faute "LOV") .....	55
Figure 3.17. La prédiction et le spectrogramme d'un Audio (une faute "NRV") .....	55
Figure 3.18. La prédiction et le spectrogramme d'un Audio (une faute "Piston") ....	56
Figure 3.19. La prédiction et le spectrogramme d'un Audio(une faute "Riderbelt") ..	56

## Introduction Générale

Dans les environnements industriels, les bruits émis par les machines sont omniprésents et fournissent des informations cruciales sur leur état de fonctionnement.

Cependant, au sein de cette cacophonie, des anomalies se glissent souvent, posant un défi majeur pour les opérateurs et les équipes de maintenance.

Le projet intitulé “**Détection d'anomalies pour les bruits de machine**” aborde un problème fondamental auquel sont confrontées les industries du monde entier. La difficulté réside dans la détection précise et rapide des anomalies parmi les divers sons émis par les machines. Ces anomalies peuvent être le résultat de vibrations inhabituelles, de variations de fréquences ou de motifs atypiques, signalant des dysfonctionnements potentiels, des usures mécaniques ou des pannes imminentes.

L'impact des anomalies non détectées peut être considérable, entraînant des arrêts de production imprévus, des retards et des coûts de réparation élevés. Les approches traditionnelles de détection d'anomalies, telles que les inspections manuelles périodiques ou les méthodes de maintenance préventive, présentent des limitations. Elles sont souvent chronophages, sujettes aux erreurs humaines et ne permettent pas une détection précoce des problèmes.

Les anomalies dans les bruits de machine peuvent avoir différentes origines, comme des problèmes de palier, des défauts de volant moteur, des soupapes de sortie de fuite ou des dommages aux composants...etc. Ces anomalies peuvent être subtiles et difficiles à détecter à l'oreille humaine, mais elles laissent des empreintes acoustiques spécifiques qui peuvent être exploitées à l'aide de techniques avancées de traitement du signal.

La complexité de la détection des anomalies dans les bruits de machine réside dans la diversité des machines, des environnements industriels et des types d'anomalies possibles. Chaque machine a sa propre signature sonore, influencée par sa conception, son état de fonctionnement et son environnement. De plus, les bruits de fond, les interférences environnementales et les variations de charge de travail peuvent masquer les anomalies et rendre leur identification difficile.

Face à ce défi, le projet "Détection d'anomalies pour les bruits de machine" vise à développer une solution innovante pour identifier ces anomalies. Il s'agit de mettre au point un système automatisé qui puisse rapidement et précisément distinguer les schémas normaux des bruits émis par les machines des schémas anormaux, afin de prévenir les pannes et les dysfonctionnements.

Ce projet revêt une importance capitale pour les industries de divers secteurs, tels que la fabrication, l'énergie et les transports. Une détection précoce des anomalies dans les bruits de machine permettrait d'éviter les coûts liés aux arrêts de production, d'optimiser les processus de maintenance et d'améliorer l'efficacité opérationnelle. De plus, cela contribuerait à renforcer la sécurité en identifiant les risques potentiels liés aux anomalies sonores et en prenant des mesures appropriées dans les plus brefs délais.

Les méthodes classiques, telles que l'inspection visuelle ou l'analyse manuelle des enregistrements sonores, se révèlent souvent inefficaces et subjectives. Elles nécessitent une expertise approfondie et une attention constante, ce qui limite leur applicabilité dans des environnements industriels complexes où de nombreuses machines sont en fonctionnement simultané.

Cependant, grâce aux avancées en matière de traitement du signal, du machine learning et de l'intelligence artificielle, de nouvelles opportunités émergent pour relever ce défi. Les techniques de reconnaissance de motifs et du machine learning peuvent être appliquées pour analyser les caractéristiques acoustiques des bruits de machine, identifier les schémas anormaux et distinguer les anomalies des sons anormaux avec une précision accrue.

Le développement d'algorithmes sophistiqués et de modèles du machine learning spécifiquement conçus pour la détection des anomalies dans les bruits de machine ouvre des perspectives prometteuses. En entraînant ces modèles avec des données d'apprentissage représentatives et en les adaptant aux spécificités de chaque industrie, il devient possible d'obtenir des systèmes de détection performants et fiables.

L'objectif ultime de ce projet est de fournir aux industries un outil précieux pour la maintenance prédictive et la gestion proactive des équipements. En détectant les anomalies dans les bruits de machine dès leur apparition, les entreprises peuvent anticiper les problèmes potentiels, planifier les interventions de maintenance de manière plus efficace et réduire les coûts associés aux réparations d'urgence.

En conclusion, le projet "**Détection d'anomalies pour les bruits de machine**" représente une réponse essentielle à une problématique majeure rencontrée par les industries. En améliorant la détection des anomalies dans les bruits de machine, il vise à prévenir les pannes coûteuses, à optimiser la performance des équipements et à renforcer la sécurité des opérations industrielles. Dans les sections suivantes, nous explorerons en détail les généralités sur la détection des anomalies sonores et Deep Learning pour la surveillance de l'état des machines, ainsi que la solution suggérée à la problématique mentionnée avant et son résultat final.

## Introduction

Dans ce chapitre, nous commençons par présenter quelques techniques de prise de son, en poursuivant par une discussion sur les représentations d'un signal audio et La transformée de Fourier, l'extraction de caractéristiques (temporelles, spectrales et perceptives). Ensuite on discute de la classification est ses algorithmes utilisés ainsi que l'évaluation et ses mesures.

### 1.1 Prise de son

#### 1.1.1 Définition

C'est l'ensemble des techniques et des processus utilisés pour enregistrer des sons, que ce soit pour la musique, la radio, la télévision ou tout autre type de production audio. Elle consiste à capter des sources sonores à l'aide de microphones et d'équipements d'enregistrement audio, en ajustant le gain, le niveau d'enregistrement, l'égalisation et les effets, afin de créer un enregistrement de qualité supérieure. La qualité de la prise de son est essentielle pour obtenir une production audio claire, précise et authentique. [1]

#### 1.1.2 Quelques techniques de prise de son

- **Enregistrement direct:** Cette technique consiste à capturer le son directement à partir de la source sonore. Elle est souvent utilisée pour enregistrer des voix, des instruments de musique ou tout autre son qui nécessite une clarté maximale. L'enregistrement direct minimise les bruits ambiants et les réflexions indésirables. [1]
- **Enregistrement d'ambiance:** Cette technique vise à capturer l'atmosphère sonore d'un environnement spécifique. Elle est souvent utilisée dans les domaines du cinéma, de la télévision et des jeux vidéo pour créer une sensation d'immersion. L'enregistrement d'ambiance est réalisé en plaçant des microphones à différents endroits pour capturer les sons naturels de l'environnement. [1]
- **Prise de son en extérieur:** Lors de la prise de son en extérieur, il est essentiel de prendre en compte les bruits ambiants tels que le vent, le trafic ou les foules. Des techniques

spécifiques sont utilisées pour minimiser ces bruits indésirables, comme l'utilisation de bonnettes anti-vent sur les microphones, l'éloignement des sources de bruit et l'utilisation de microphones directionnels pour cibler spécifiquement la source sonore. [1]

- **Prise de son en studio:** Dans un environnement de studio contrôlé, plusieurs techniques sont utilisées pour obtenir une prise de son de haute qualité. Cela peut inclure l'utilisation de cabines insonorisées pour enregistrer des voix ou des instruments, le positionnement précis des microphones pour obtenir le meilleur son possible, ainsi que l'utilisation de techniques de re-recording pour ajouter des effets spéciaux ou mélanger plusieurs pistes audio. [1]
- **Enregistrement en multi-microphones:** L'enregistrement en multi-microphones implique l'utilisation de plusieurs microphones pour capturer différents éléments sonores de manière séparée. Cette technique est couramment utilisée dans les situations d'enregistrement en direct, les concerts ou les enregistrements d'orchestres, où chaque instrument ou source sonore est enregistré individuellement pour un contrôle ultérieur pendant le mixage. [1]
- **Prise de son binaurale:** La prise de son binaurale est une technique qui vise à reproduire de manière réaliste l'expérience auditive humaine en utilisant un système de deux microphones placés de manière similaire à nos oreilles. Cette technique permet de créer une sensation d'espace et de localisation précise du son lors de l'écoute avec des écouteurs. [1]

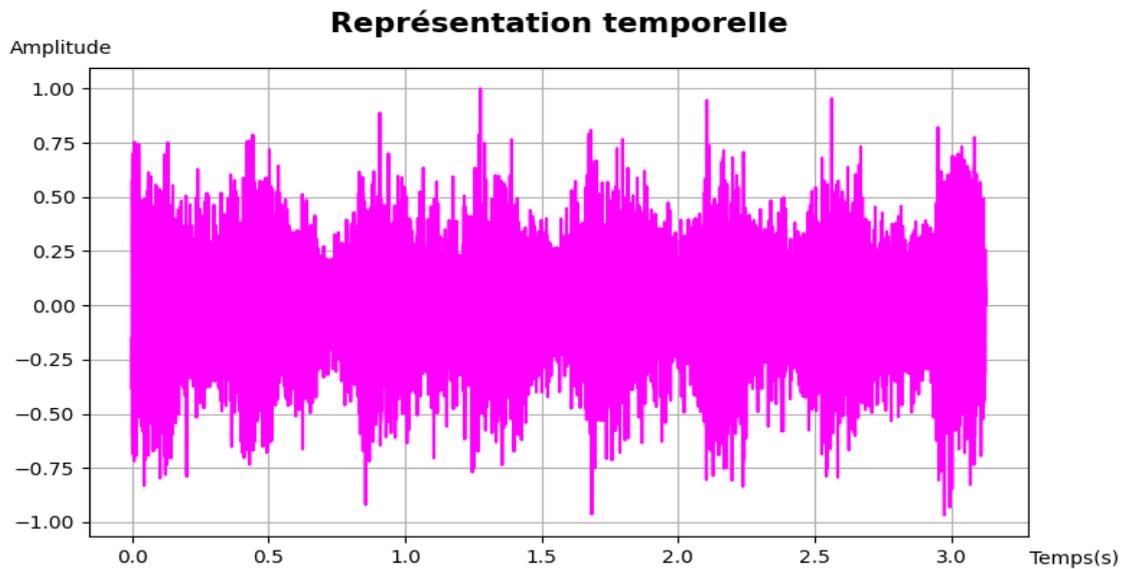
## 1.2 Représentation du signal audio

C'est la manière de visualiser les caractéristiques acoustiques du son en utilisant des graphiques, des images ou des spectres. Les différentes représentations du signal audio peuvent aider à comprendre la structure, la composition et la qualité d'un enregistrement sonore.

### 1.2.1 Représentations du signal audio

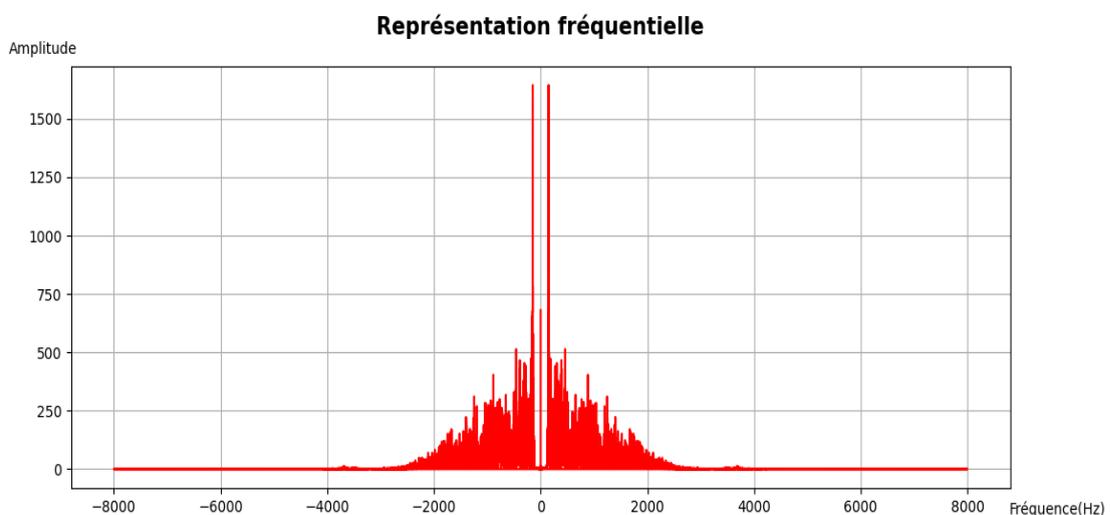
- **Représentation temporelle:** Le signal audio est représenté en fonction du temps. L'axe horizontal représente le temps et l'axe vertical représente l'amplitude du signal. Cette représentation est utile pour visualiser les variations d'amplitude dans le signal en fonction du temps. La représentation temporelle est souvent utilisée pour l'analyse de la

parole et de la musique, car elle permet de visualiser les variations de volume et les rythmes. [2]



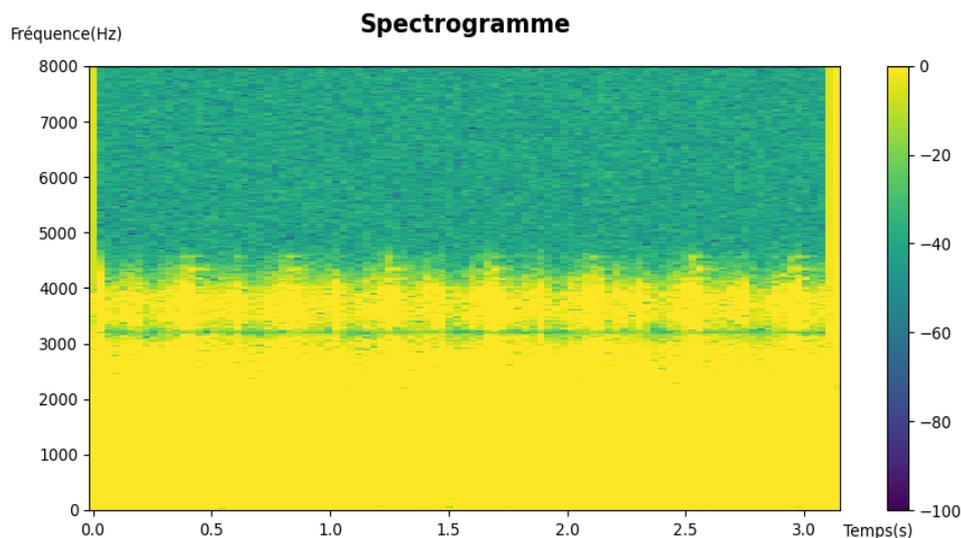
**Figure 1.1.** La représentation temporelle d'un signal audio.

- **Représentation fréquentielle:** Le signal audio est représenté en fonction de la fréquence. L'axe horizontal représente la fréquence et l'axe vertical représente l'amplitude du signal. Cette représentation est utile pour visualiser la distribution des différentes fréquences dans le signal. La représentation fréquentielle est souvent utilisée pour l'analyse de la musique et des sons complexes, car elle permet de visualiser les harmoniques et les différences de tonalité. [2]



**Figure 1.2.** La représentation fréquentielle d'un signal audio.

- **Représentation en spectrogramme:** Le spectrogramme est une représentation bidimensionnelle qui combine les informations temporelles et fréquentielles du signal. Dans cette représentation, le signal audio est divisé en petits segments de temps, et pour chaque segment, la distribution des différentes fréquences est représentée. Les couleurs sont souvent utilisées pour représenter l'amplitude du signal dans chaque segment. Cette représentation est utile pour l'analyse de signaux audio complexes, car elle permet de visualiser les variations temporelles et fréquentielles du signal en même temps. [2]



### 1.3 La transformée de Fourier (STFT)

C'est une technique de traitement de signal qui permet d'analyser les variations de fréquence d'un signal audio dans le temps. [3]

La STFT consiste à diviser le signal audio en petits segments de temps, appelés fenêtres, et à calculer la transformée de Fourier pour chaque fenêtre. En utilisant une fenêtre de taille fixe, la STFT fournit une représentation spectrale locale et temporelle du signal. [3]

Cette représentation spectrale peut être affichée sous la forme d'un spectrogramme. [3]

La STFT peut être utilisée pour détecter des anomalies dans un signal audio en comparant la distribution spectrale de chaque fenêtre à une distribution de référence. [3]

Les variations significatives de la distribution spectrale par rapport à la référence peuvent indiquer la présence d'une anomalie. [3]

La STFT est largement utilisée dans de nombreux domaines, notamment dans l'analyse de la parole, la reconnaissance de la parole, la musique et l'audio en général. Elle est souvent utilisée en combinaison avec d'autres techniques de traitement de signal pour extraire des informations pertinentes du signal audio. [3]

On définit la STFT par cette équation mathématique: [3]

$$STFT(t, f) = \int X(\tau) w(\tau - t) e^{(-j2\pi f\tau) d\tau}$$

Avec:

- $STFT(t, f)$  est la valeur de la STFT à l'instant de temps  $t$  et à la fréquence  $f$ .
- $x(t)$  est le signal d'origine dans le domaine temporel.
- $w(\tau - t)$  est la fenêtre temporelle appliquée au signal  $x(t)$ . Elle est centrée autour du temps et est typiquement une fonction de fenêtrage symétrique.
- $e^{(-j2\pi f\tau) d\tau}$  est la fonction de l'exponentielle complexe utilisée pour représenter les différentes fréquences.
- $\tau$  est une variable d'intégration utilisée pour parcourir le signal dans le domaine temporel.

#### □ Étapes de calcul de la STFT

- Diviser le signal en segments temporels: Tout d'abord, le signal est découpé en segments de temps courts et généralement chevauchants. La taille de ces segments est déterminée par la fenêtre temporelle choisie. Les segments chevauchants permettent une analyse plus précise des variations fréquentielles. [3]
- Appliquer la fenêtre temporelle: Avant d'appliquer la transformée de Fourier, chaque segment temporel est généralement multiplié par une fenêtre, telle que la fenêtre de Hamming. Cela permet de réduire les effets de bord et d'assurer une meilleure représentation spectrale. [3]

- Calculer la transformée de Fourier: Une fois que les segments ont été fenêtrés, la transformée de Fourier est appliquée à chaque segment. Cela convertit le signal du domaine temporel en un spectre fréquentiel. [3]
- Obtenir le spectre temps-fréquence: En résultat, on obtient une représentation temps-fréquence du signal. Chaque segment de temps est associé à un spectre fréquentiel qui montre la distribution des fréquences présentes dans ce segment spécifique. [3]

## 1.4 L'extraction de caractéristiques

L'extraction de caractéristiques est une étape importante dans le traitement de signal audio pour la détection d'anomalies. Elle permet de réduire la dimensionnalité des données en sélectionnant les informations les plus pertinentes du signal audio qui peuvent aider à identifier des modèles ou des motifs caractéristiques dans les données.

Ces caractéristiques peuvent être des mesures telles que la fréquence, l'énergie, le tempo, le spectre de fréquences, les coefficients de Mel Cepstral (MFCC), ou encore des mesures plus complexes telles que des paramètres de modulation. [4]

### 1.4.1 Techniques de l'extraction de caractéristiques

- **Méthodes statistiques:** Elles exploitent des mesures statistiques telles que la moyenne, la variance, l'écart-type, la covariance, etc., pour caractériser les propriétés des données. [4]
- **Transformées mathématiques:** Des transformées comme la transformée de Fourier, la transformée en ondelettes, la transformée en cosinus discrète (DCT), etc., peuvent être appliquées pour extraire des informations dans le domaine fréquentiel, temporel ou spatial. [4]
- **Méthodes basées sur des modèles:** Ces méthodes utilisent des modèles mathématiques ou statistiques pour représenter les données. Par exemple, les modèles de mélange gaussien, les modèles cachés de Markov (HMM), les réseaux de neurones, etc. [4]
- **Descripteurs de formes:** Ils sont utilisés pour extraire des caractéristiques géométriques ou structurelles des objets ou des données. Par exemple, les moments de Hu, les descripteurs de texture, les histogrammes de couleurs, etc. [4]

- **Méthodes d'apprentissage automatique non supervisées:** Des techniques telles que l'analyse en composantes principales (PCA), la réduction de dimensionnalité, la méthode du voisin le plus proche, etc., sont utilisées pour extraire des caractéristiques significatives sans étiquettes de classe préalables. [4]

## 1.4.2 Types de caractéristiques

- **Caractéristiques temporelles:** Elles sont basées sur l'amplitude du signal audio en fonction du temps (les variations temporelles des données), elles fournissent des informations sur la dynamique et les propriétés temporelles des signaux (**exemples:** La moyenne, La variance, L'énergie). [5]
- **Caractéristiques spectrales:** Ces caractéristiques sont basées sur la distribution des fréquences du signal audio (les variations de fréquence dans les données), elles sont souvent utilisées pour l'analyse des signaux audio et des données de vibration (**exemples:** Le spectre de puissance, La bande passante). [5]
- **Caractéristiques perceptives:** basées sur la façon dont l'oreille humaine perçoit le signal audio (domaines où les signaux sont perçus par les sens humains), elles sont conçues pour capturer les aspects perceptifs et subjectifs des données (**exemples:** Le niveau sonore, Le pitch, La mélodie, Le timbre et la rugosité). [5]

On note que le choix des caractéristiques à extraire dépendra des objectifs du projet et des techniques d'analyse que nous souhaitons utiliser. Une fois que les caractéristiques sont extraites, elles peuvent être utilisées comme entrée pour des algorithmes de classification pour détecter les anomalies dans le signal audio.

## 1.5 La classification

La classification est une technique d'analyse de données qui consiste à assigner des étiquettes ou des catégories prédéfinies à des observations en fonction de certaines caractéristiques. Dans le cadre de la détection d'anomalies pour les bruits de machine, la classification peut être utilisée pour identifier différents types de bruits de machines ou pour détecter des anomalies spécifiques. [6]

### 1.5.1 Méthodes de classification

- **Les classificateurs basés sur des règles:** Ces classificateurs utilisent des règles prédéfinies pour déterminer si un segment de signal est normal ou anormal. Par exemple, un système expert médical peut utiliser des règles pour diagnostiquer une maladie en fonction des symptômes présentés par un patient. Les règles peuvent être exprimées sous la forme de conditions (Si... Alors...).
- **Les classificateurs basés sur des seuils:** Ces classificateurs utilisent des seuils de tolérance pour déterminer si un segment de signal est normal ou anormal. Les seuils peuvent être déterminés empiriquement ou par des techniques statistiques.
- **Les classificateurs basés sur le machine learning:** Ces classificateurs utilisent des algorithmes du machine learning pour apprendre à partir de données d'entraînement et pour prédire si un segment de signal est normal ou anormal. Les algorithmes du machine learning les plus couramment utilisés pour la détection d'anomalies incluent les réseaux de neurones, les machines à vecteurs de support et les arbres de décision.

Le choix de l'algorithme de classification dépendra des caractéristiques extraites, des données d'entraînement disponibles et des performances souhaitées pour la détection d'anomalies. Il est également important de prendre en compte les coûts associés à la classification erronée de segments de signal, car cela peut entraîner des conséquences graves pour la maintenance préventive des machines.

### 1.5.2 Algorithmes de classification

- **Arbres de décision:** Les arbres de décision sont des modèles de classification basés sur une structure d'arbre où chaque nœud représente une décision basée sur une caractéristique particulière. Les arbres de décision sont faciles à interpréter et à visualiser, et ils peuvent gérer des données à la fois catégoriques et numériques. [6]
- **Machines à vecteurs de support (SVM):** Les SVM sont des algorithmes de classification qui trouvent un hyperplan optimal pour séparer les différentes classes d'un ensemble de données. Ils sont efficaces pour traiter des données avec de grandes dimensions et sont couramment utilisés pour la classification binaire. [6]

- **Réseaux de neurones:** Les réseaux de neurones sont des modèles de classification inspirés du fonctionnement du cerveau. Ils sont constitués de couches de neurones artificiels interconnectés qui apprennent à partir des données. Les réseaux de neurones profonds, tels que les réseaux de neurones convolutionnels (CNN) et les réseaux de neurones récurrents (RNN), sont particulièrement efficaces pour la classification d'images et de séquences respectivement. [6]
- **Naïve Bayes:** Les classificateurs naïfs de Bayes sont basés sur le théorème de Bayes et supposent une indépendance conditionnelle entre les caractéristiques. Malgré leur simplicité, ils peuvent être très performants pour la classification de textes et la classification de documents. [6]
- **K-plus proches voisins (K-NN):** Le classificateur K-NN attribue une étiquette à un échantillon en se basant sur les étiquettes des échantillons les plus proches dans l'espace des caractéristiques. La proximité est généralement mesurée en utilisant une mesure de distance, telle que la distance euclidienne. Le choix de la valeur k détermine le nombre de voisins à considérer. [6]
- **Forêts aléatoires:** Les forêts aléatoires sont des ensembles d'arbres de décision où chaque arbre est formé sur un sous-ensemble aléatoire des données d'entraînement et des caractéristiques. Les prédictions sont ensuite combinées par vote majoritaire ou moyennage pour la classification finale. [6]
- **Classificateurs bayésiens:** Les classificateurs bayésiens utilisent le théorème de Bayes pour estimer la probabilité d'appartenance à une classe donnée. Ils peuvent être basés sur différentes distributions de probabilité, telles que la distribution normale (classificateur bayésien gaussien) ou la distribution de Bernoulli (classificateur bayésien naïf). [6]

Ces méthodes de classification sont largement utilisées et ont fait leurs preuves dans différents domaines. Cependant, il est important de choisir la méthode appropriée en fonction de la nature des données, de la complexité du problème et des contraintes spécifiques de l'application.

## 1.6 Évaluation

L'évaluation permet de mesurer les performances du système de détection d'anomalies et de déterminer s'il fonctionne efficacement pour détecter les anomalies dans le signal audio. Les

données d'évaluation sont généralement séparées en deux ensembles : l'ensemble d'apprentissage, utilisé pour entraîner le modèle, et l'ensemble de test, utilisé pour évaluer les performances du modèle.

Les performances du modèle peuvent être évaluées en comparant les prédictions du modèle aux étiquettes de vérité terrain (ou labels) sur l'ensemble de tests. [7]

### 1.6.1 Quelques techniques de l'évaluation

- **Validation croisée (Cross-Validation):** La validation croisée est une technique qui permet d'estimer les performances d'un modèle en utilisant l'ensemble de données disponible. Elle consiste à appliquer une division sur les données en plusieurs ensembles (plis) pour l'entraînement et la validation. Le modèle est entraîné sur une partie des données et évalué sur les autres parties. Cela permet d'obtenir plusieurs estimations des performances du modèle, en réduisant le biais introduit par une seule division des données. La validation croisée k-fold est l'une des variantes les plus courantes, où les données sont divisées en k ensembles de taille égale. [7]
- **Ensemble d'apprentissage et ensemble de tests (Train-Test Split):** Dans cette technique, les données sont divisées en un ensemble d'apprentissage et un ensemble de tests. Le modèle est entraîné sur l'ensemble d'apprentissage et évalué sur l'ensemble de tests. Cette approche est simple à mettre en œuvre, mais elle peut être sensible à la composition des données et peut donner des estimations biaisées des performances du modèle. [7]
- **Évaluation sur des données indépendantes:** Dans certains cas, il peut être judicieux de diviser les données en ensembles d'entraînement, de validation et de test distincts. L'ensemble d'entraînement est utilisé pour entraîner le modèle, l'ensemble de validation est utilisé pour ajuster les paramètres du modèle et l'ensemble de test est utilisé pour évaluer les performances finales du modèle. Cela permet de simuler une situation où le modèle est utilisé sur de nouvelles données non vues. [7]

### 1.6.2 Matrice de confusion

La matrice de confusion est un tableau qui montre le nombre de prédictions correctes et incorrectes effectuées par le système de détection d'anomalies. Elle est généralement représentée sous la forme d'une matrice à deux dimensions, avec les prédictions réelles en

colonnes et les prédictions du système de détection d'anomalies en lignes. La matrice de confusion peut être utilisée pour calculer des métriques telles que la précision, le rappel et le score F1. [8]

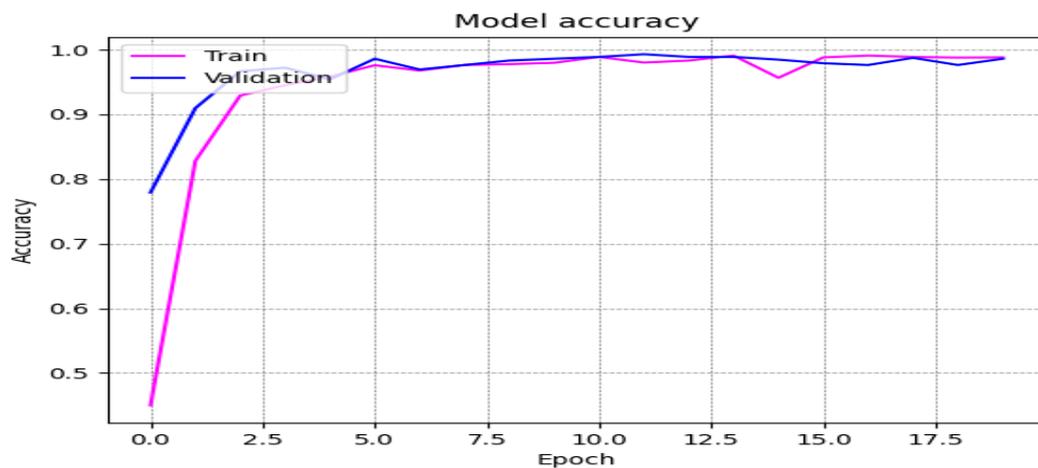
### 1.6.3 Métriques (Metrics)

Ce sont des indicateurs quantitatifs utilisés pour évaluer les performances d'un modèle de classification. Elles permettent de mesurer différents aspects de la qualité des prédictions du modèle. [9]

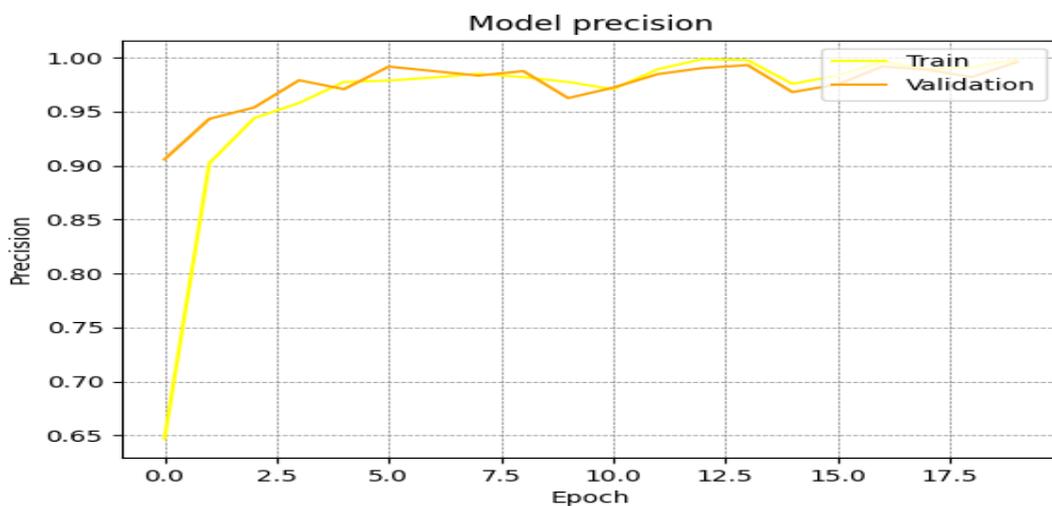
#### □ Métriques d'évaluation couramment utilisées

- **Précision (Precision):** La précision est le nombre de prédictions correctes faites par le système de détection d'anomalies, divisé par le nombre total de prédictions effectuées. Elle mesure la capacité du système à identifier correctement les anomalies. [9]
- **Rappel (Recall):** Le rappel est le nombre de prédictions correctes faites par le système de détection d'anomalies, divisé par le nombre total d'anomalies dans le signal audio. Il mesure la capacité du système à identifier toutes les anomalies présentes dans le signal audio. [9]
- **Score F1:** Une mesure de la précision et de la rappel en classification binaire, qui combine les deux métriques en un seul score. Le score F1 est calculé en prenant en compte la moyenne harmonique de la précision et du rappel pour évaluer la performance globale du système de détection d'anomalies.
- **Exactitude (Accuracy):** L'exactitude mesure la proportion d'échantillons correctement classés parmi tous les échantillons, elle est une mesure globale de la performance du modèle, mais elle peut être trompeuse en cas de déséquilibre de classe. [9]

Il est important de prendre en compte les coûts associés aux faux positifs et aux faux négatifs dans l'évaluation des performances d'un système de détection d'anomalies pour les bruits de machine. En général, il est préférable de minimiser les faux négatifs, car cela permet de détecter toutes les anomalies présentes dans le signal audio, même si cela peut entraîner des coûts plus élevés en termes de temps et de ressources pour la maintenance préventive des machines.



**Figure 1.4.** Exemple de l'un des métriques d'évaluation (Accuracy).



**Figure 1.5.** Exemple de l'un des métriques d'évaluation (Precision).

## Conclusion

Dans ce chapitre on a parlé de la prise de son, comment on représente un signal, la STFT, l'extraction des caractéristiques, la classification et l'évaluation.

Dans le chapitre suivant on va discuter du machine et deep learning, les réseaux de neurones, l'auto-encodeur et finalement le CNN.

## Introduction

Dans ce chapitre, nous commençons par présenter quelques bases du machine learning, en poursuivant par une discussion sur le deep learning, les deep neural networks, le MLP . Ensuite on discute d'auto-encodeur, les fonctions d'activation et les hyper-paramètres d'un réseau de neurones. Finalement, on explique l'architecture et quelques applications du CNN.

## 2.1 Bases du Machine learning

### 2.1.1 Définition

Le Machine Learning, qui fait partie de l'intelligence artificielle, permet aux ordinateurs d'acquérir des connaissances et d'améliorer leurs performances sans qu'il soit nécessaire de les programmer explicitement pour chaque tâche. Le but du machine learning est d'analyser des données et de trouver des schémas ou des modèles à partir desquels des prédictions peuvent être faites. Ces prédictions peuvent être utilisées pour prendre des décisions ou automatiser des tâches. [10]

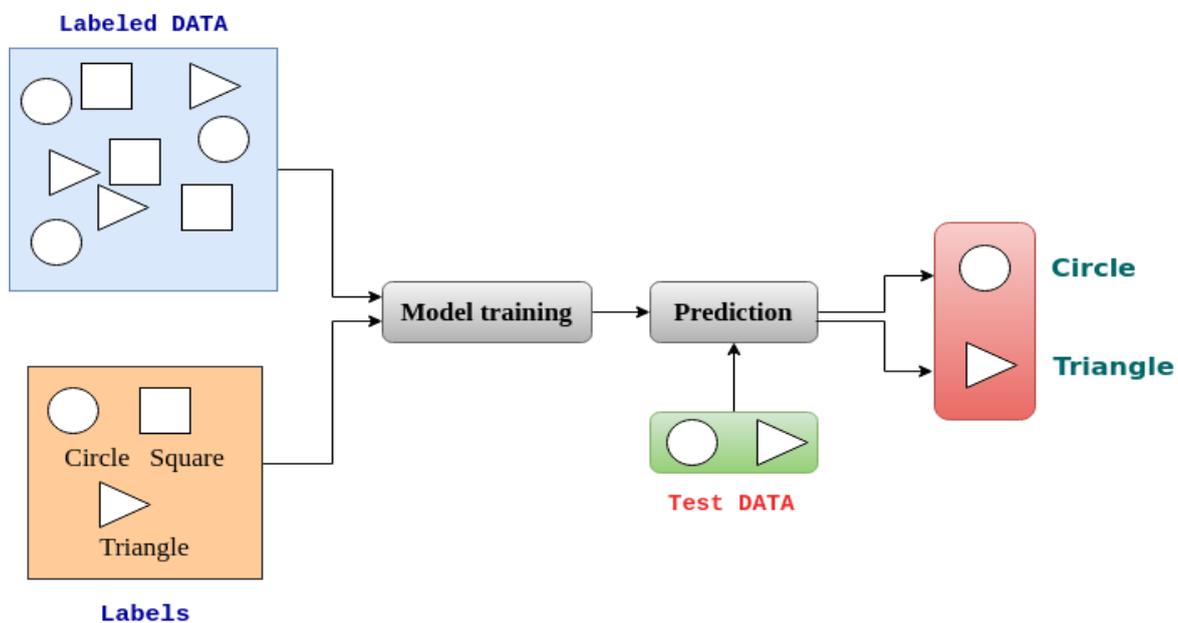
Le machine learning implique une série d'étapes pour entraîner un ordinateur ou une machine à apprendre à partir de données. Le processus commence par la collecte de données à partir de différentes sources. Ensuite, les données collectées sont prétraitées en les nettoyant, en les transformant et en les organisant. Des caractéristiques pertinentes sont ensuite extraites ou sélectionnées à partir des données. Un modèle du machine learning approprié est choisi en fonction du problème et des caractéristiques des données. Le modèle est ensuite entraîné sur les données prétraitées, en ajustant ses paramètres internes pour apprendre les motifs et les relations. Après l'entraînement, le modèle est évalué à l'aide de mesures pour évaluer ses performances. Si nécessaire, le modèle est optimisé en affinant ses paramètres. Une fois entraîné et évalué, le modèle peut faire des prédictions ou des inférences sur de nouvelles données non vues. Enfin, le modèle est déployé dans un environnement de production et surveillé pour garantir des résultats précis et fiables. Tout au long du processus, des itérations

peuvent être effectuées pour améliorer les performances du modèle en ré-examinant des étapes telles que le prétraitement, la sélection du modèle et l'optimisation. [10]

Le Machine Learning utilise des algorithmes qui permettent aux ordinateurs de s'entraîner à partir de données. Les algorithmes du Machine Learning sont classés en trois catégories principales : **Supervised learning**, **Unsupervised learning** et **Reinforcement learning**. [10]

### 2.1.2 Types du machine learning

- **Supervised learning(SL)**: Ce type est utilisé pour résoudre des problèmes de classification et de régression. Dans le supervised learning, un modèle est entraîné sur un ensemble de données étiquetées, c'est-à-dire des données qui ont déjà été catégorisées ou étiquetées. Le modèle peut ensuite être utilisé pour prédire les étiquettes de nouvelles données. Les exemples courants de problèmes de classification incluent la détection de spam dans les e-mails et la reconnaissance d'images. Les exemples courants de problèmes de régression incluent la prédiction des prix de l'immobilier et la prédiction des performances des étudiants. [10]



**Figure 2.1.** Exemple en comment le SL fonctionne.

- **Unsupervised learning(USL)**: Cela est pour la détection de modèles dans des données non étiquetées. Les algorithmes du unsupervised learning sont souvent utilisés pour la segmentation de données ou la réduction de dimensionnalité. Les exemples courants du

unsupervised learning incluent la classification automatique de documents, la segmentation de clientèle et la détection d'anomalies. [10]

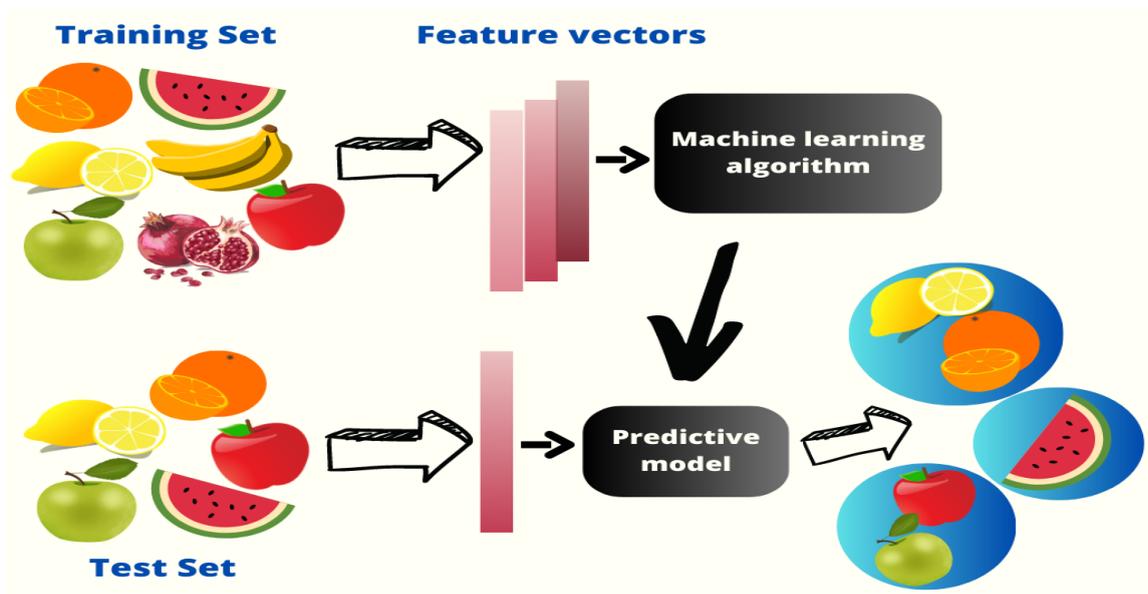


Figure 2.2. Exemple en comment le USL fonctionne.

- **Reinforcement learning(RL):** Le RL est pour les problèmes d'apprentissage interactifs. Dans le reinforcement learning, un agent apprend à prendre des décisions en interagissant avec un environnement. L'agent reçoit des récompenses pour les bonnes décisions et des pénalités pour les mauvaises décisions. Les exemples courants de le reinforcement learning incluent les jeux, la robotique et la gestion des ressources naturelles. [10]

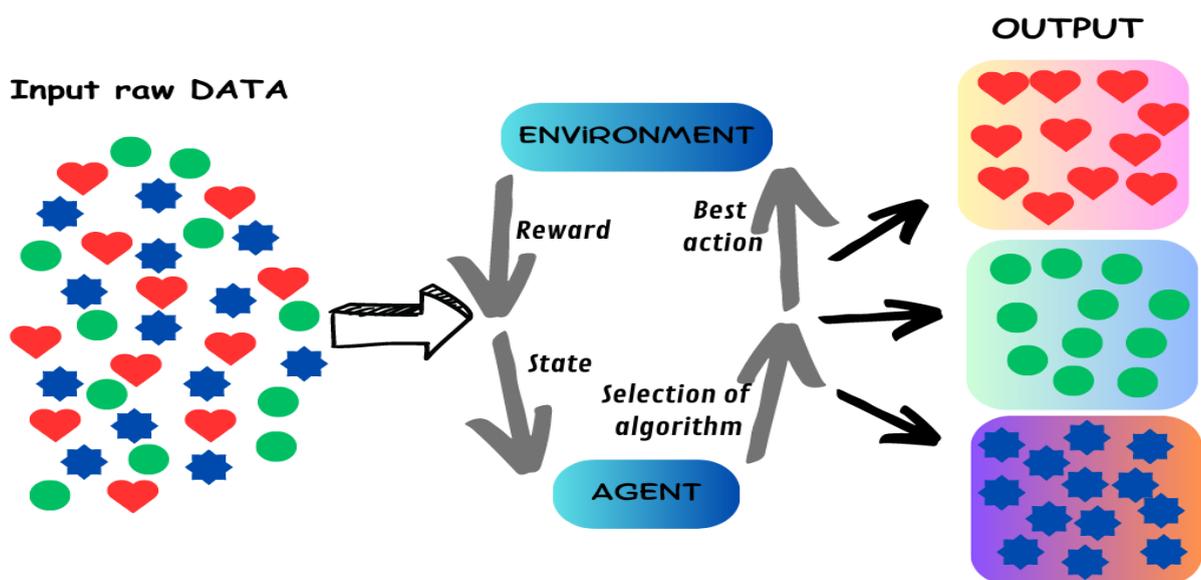


Figure 2.3. Exemple en comment le RL fonctionne.

## 2.2 Deep learning

### 2.2.1 Définition

Le deep learning est une branche spécifique du domaine "Machine Learning" qui utilise des deep neural networks pour apprendre à partir de données non structurées ou non étiquetées. Le Deep Learning permet de résoudre des problèmes complexes en utilisant des architectures de deep neural networks composées de multiples couches. Ces couches sont conçues pour extraire des caractéristiques des données d'entrée, et chaque couche traite les caractéristiques extraites par la couche précédente. [11]

Le processus du deep learning comprend plusieurs étapes clés. Tout d'abord, il faut collecter et préparer les données en rassemblant des ensembles de données pertinents et en les nettoyant pour assurer leur qualité. Ensuite, la construction du modèle intervient, impliquant la conception d'une architecture de réseau de neurones profonds avec des couches interconnectées qui effectuent des transformations sur les données. Une fois le modèle construit, il passe à l'étape cruciale de l'entraînement, où il est ajusté en modifiant les poids et les biais des neurones pour minimiser une fonction de coût ou d'erreur à l'aide de techniques d'optimisation telles que la rétropropagation du gradient. [11]

Après l'entraînement, le modèle est évalué et validé pour vérifier sa performance en utilisant un ensemble de données de validation distinct. Si nécessaire, des ajustements sont effectués pour améliorer les résultats, tels que des modifications de l'architecture, l'ajout de régularisation ou l'augmentation des données d'entraînement. Une fois que le modèle a été validé, il est testé sur un ensemble de données de test indépendant pour évaluer ses performances réelles. Si les résultats sont satisfaisants, le modèle est prêt à être déployé pour effectuer des prédictions sur de nouvelles données. [11]

Après le déploiement, il est possible d'effectuer un réglage fin et une optimisation continue en fonction des performances réelles et des retours d'expérience, en ajustant les hyper-paramètres du modèle ou en utilisant des techniques telles que l'apprentissage par transfert ou l'ensemble d'apprentissage actif. Ce processus itératif de formation, d'évaluation, d'ajustement et de déploiement se poursuit jusqu'à ce que le modèle atteigne les performances souhaitées pour la tâche donnée. Le deep learning a été appliqué avec succès dans divers domaines tels que la vision par ordinateur, le traitement du langage naturel et la reconnaissance vocale. [11]

## 2.2.2 Deep neural networks

Les deep neural networks sont des réseaux de neurones artificiels utilisés pour le deep learning dans le domaine du Deep Learning. Ils sont construits à partir de plusieurs couches de neurones interconnectés. Chaque couche traite les entrées de la couche précédente pour produire une sortie qui sera utilisée par la couche suivante. [12]

Les deep neural networks peuvent apprendre des représentations hiérarchiques de données à partir de couches profondes, et sont utilisés pour des tâches telles que la reconnaissance d'images, la reconnaissance vocale et la traduction automatique. [12]

Cependant, leur entraînement nécessite une grande quantité de données et de puissance de calcul, et leur interprétation peut être difficile en raison de leur complexité. [12]

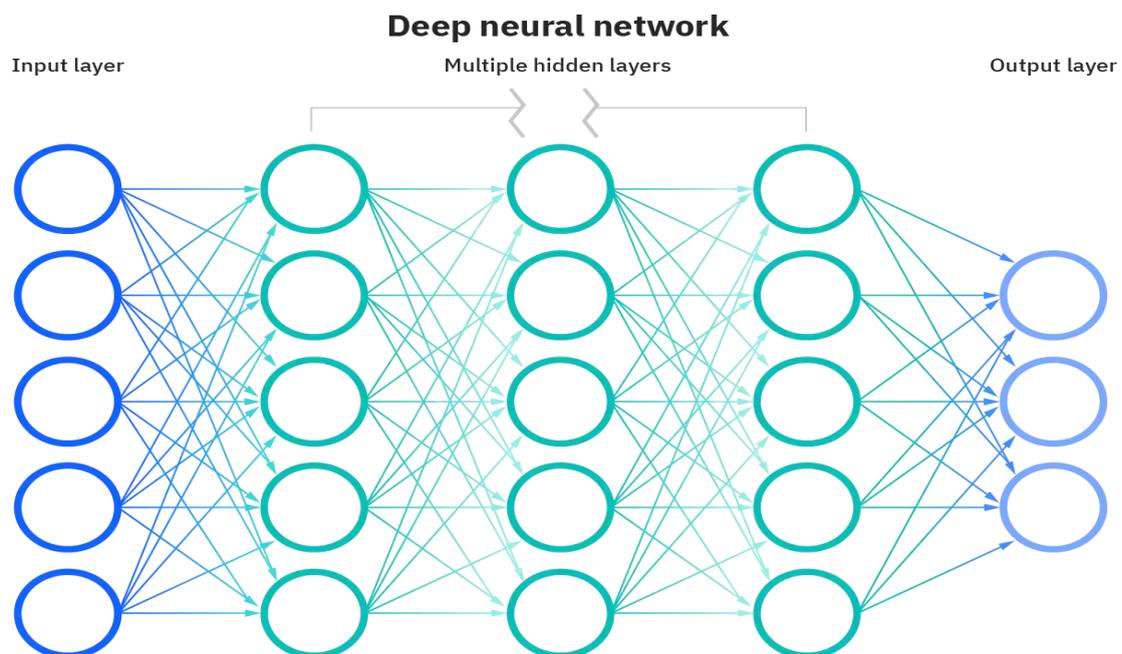


Figure 2.4. Architecture d'un Deep neural network.

- **Types du Deep neural network**
- **Artificial Neural Networks(ANN):** Ce sont les réseaux de neurones les plus simples, dans lesquels chaque neurone est connecté à tous les neurones de la couche suivante. [13]
- **Réseaux de neurones multicouches(MLP):** Ce sont composés de plusieurs couches de neurones entièrement connectés. Chaque neurone est connecté à tous les neurones de la couche suivante, formant ainsi une structure hiérarchique. [13]

- **Convolutional Neural Networks(CNN)**: Ces réseaux sont conçus pour traiter des données structurées telles que des images, des vidéos et des signaux audio. Les couches de convolution extraient des caractéristiques à différentes échelles et résolutions à partir des entrées. [13]
- **Recurrent Neural Networks(RNN)**: Sont conçus pour traiter des données séquentielles telles que du texte, de la parole ou des séquences temporelles. Les couches récurrentes permettent aux réseaux de mémoriser des informations à partir de séquences précédentes. [13]
- **LSTM**: Sont une variante des réseaux récurrents, conçus pour mieux gérer les problèmes de longue dépendance temporelle. [13]
- **Autoencoder Neural Networks** : Ces réseaux sont désignés pour apprendre une représentation compressée de données d'entrée, en utilisant une architecture de réseau qui force le modèle à répliquer l'entrée à travers une couche "code" de taille inférieure. [13]

### 2.2.3 Le perceptron multicouche (MLP)

Le MLP est un type de réseau de neurones artificiels qui consiste en plusieurs couches de neurones interconnectés. Les neurones d'une couche traitent les entrées de la couche précédente et produisent une sortie qui sera utilisée par la couche suivante. [13]

Le MLP est un réseau de neurones feedforward, c'est-à-dire que les données circulent dans une seule direction, de l'entrée vers la sortie. La première couche du MLP est appelée couche d'entrée et reçoit les données d'entrée, tandis que la dernière couche est appelée couche de sortie et produit les sorties du réseau. [13]

Les couches intermédiaires entre la couche d'entrée et la couche de sortie sont appelées couches cachées, car les sorties de ces couches ne sont pas directement observables. Les MLP peuvent avoir plusieurs couches cachées, et chaque couche peut avoir un nombre différent de neurones. [13]

Le MLP est souvent utilisé pour des tâches de classification et de régression. Pour l'entraînement du MLP, on utilise un algorithme du supervised learning, tel que la rétropropagation du gradient. L'objectif de l'entraînement est de trouver les poids des connexions entre les neurones qui minimisent une fonction de perte, telle que l'erreur

quadratique moyenne pour les tâches de régression ou l'entropie croisée pour les tâches de classification. [13]

Le fonctionnement du MLP peut être décrit comme suit. Tout d'abord, les poids et les biais des neurones du MLP sont initialisés de manière aléatoire ou selon une méthode spécifique. Chaque connexion entre les neurones est attribuée un poids, qui sera ajusté lors de l'apprentissage pour influencer la sortie du neurone. [13]

Lorsqu'une donnée d'entrée est présentée au MLP, elle se propage de manière séquentielle à travers les différentes couches du réseau, dans un processus appelé propagation avant (Forward Propagation). Chaque neurone de la couche d'entrée reçoit les valeurs d'entrée et les transmet, pondérées par les poids, à la couche suivante. Chaque neurone de la couche suivante effectue une somme pondérée de ses entrées et applique une fonction d'activation non linéaire, telle que la fonction sigmoïde ou la fonction ReLU, pour produire sa sortie. [13]

La propagation avant se poursuit jusqu'à ce que les valeurs atteignent la couche de sortie. Les sorties des neurones de cette couche représentent les prédictions ou les valeurs calculées par le modèle pour la tâche donnée. [13]

Une fois les prédictions obtenues, l'erreur entre ces prédictions et les valeurs attendues est calculée à l'aide d'une fonction de coût, telle que l'erreur quadratique moyenne (Mean Squared Error, MSE) ou l'entropie croisée (Cross-Entropy Loss). Cette erreur est utilisée pour évaluer la performance du modèle et guider la mise à jour des poids et des biais lors de l'apprentissage. [13]

La rétropropagation du gradient (Backpropagation) est alors utilisée pour déterminer le gradient de l'erreur par rapport aux poids et aux biais du réseau. Ce gradient est ensuite utilisé pour ajuster les poids et les biais afin de minimiser l'erreur. Cela se fait généralement à l'aide d'un algorithme d'optimisation, tel que la descente de gradient stochastique (Stochastic Gradient Descent, SGD) ou ses variantes, en mettant à jour les paramètres dans la direction opposée au gradient. [13]

Le processus d'apprentissage se poursuit par des itérations successives sur plusieurs exemples d'entraînement. À chaque itération, les étapes de propagation avant, de calcul de l'erreur et de rétropropagation du gradient sont répétées, permettant ainsi au réseau d'ajuster progressivement ses poids et ses biais. Le nombre d'itérations dépend de la taille du jeu de données et des critères d'arrêt prédéfinis. [13]

Une fois que le MLP a été entraîné, il peut être utilisé pour effectuer des prédictions sur de nouvelles données, en passant simplement les données d'entrée à travers le réseau et en observant la sortie de la couche de sortie. [13]

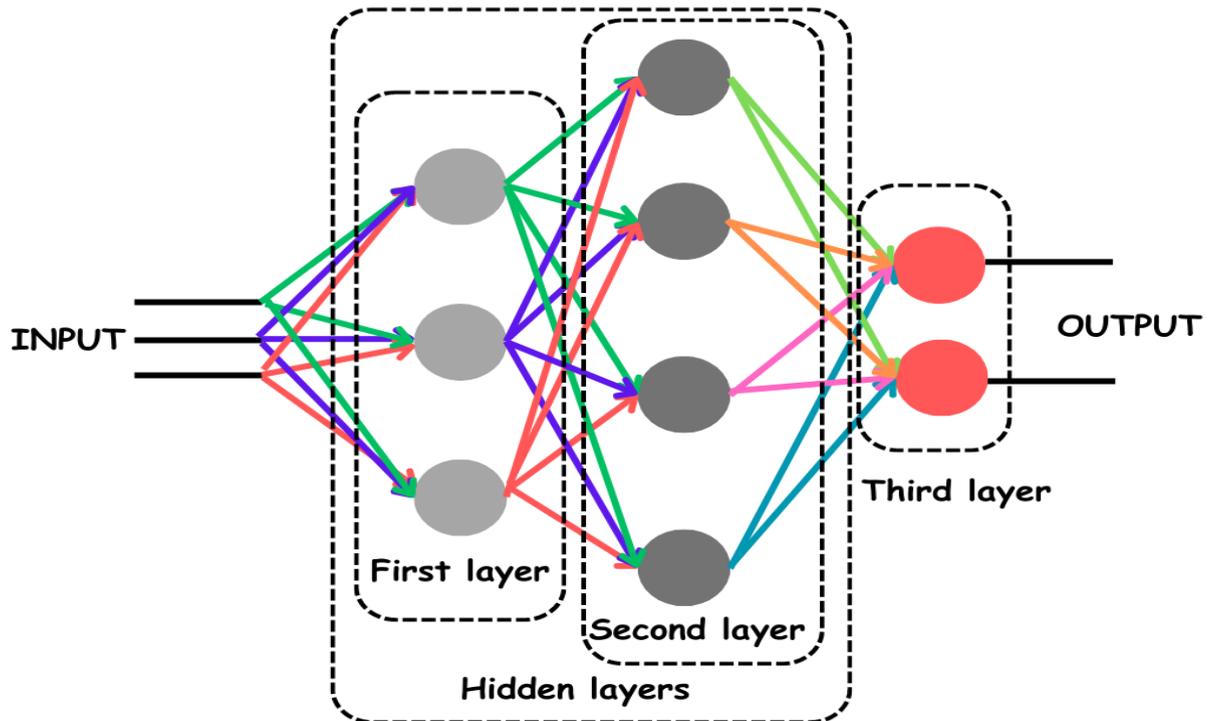


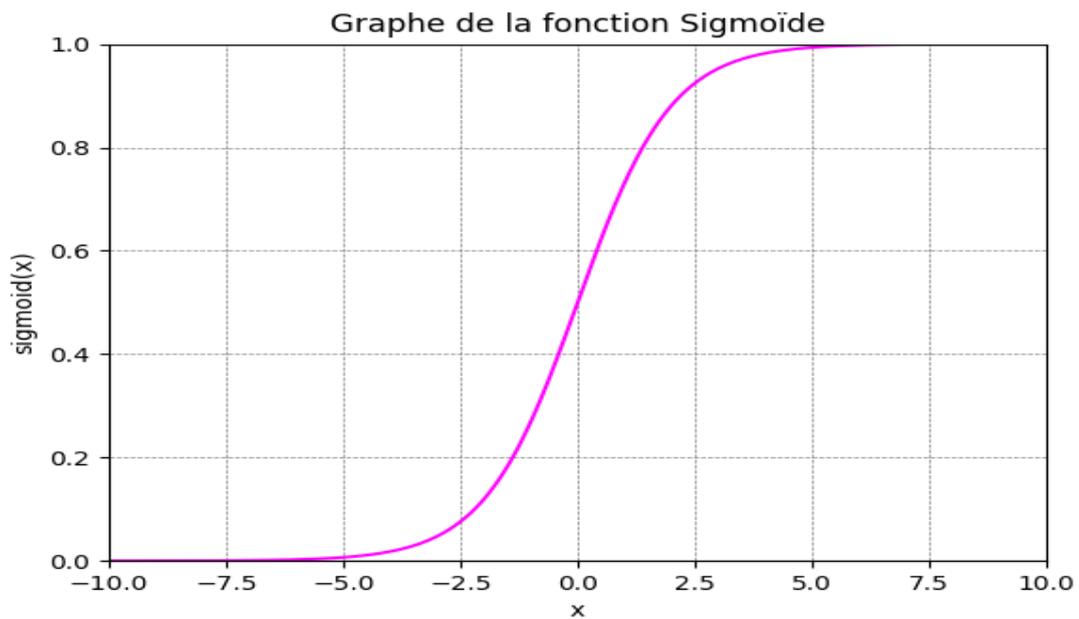
Figure 2.5. Architecture d'un MLP.

#### 2.2.4 Fonction d'activation

La fonction d'activation est une fonction mathématique appliquée à la sortie d'un neurone dans un réseau de neurones. Elle permet d'introduire une non-linéarité dans la sortie du neurone, permettant ainsi au réseau de modéliser des relations non-linéaires entre les entrées et les sorties. En effet, sans une fonction d'activation, la sortie d'un neurone ne serait qu'une combinaison linéaire des entrées, ce qui ne permettrait pas de capturer des relations complexes. [14]

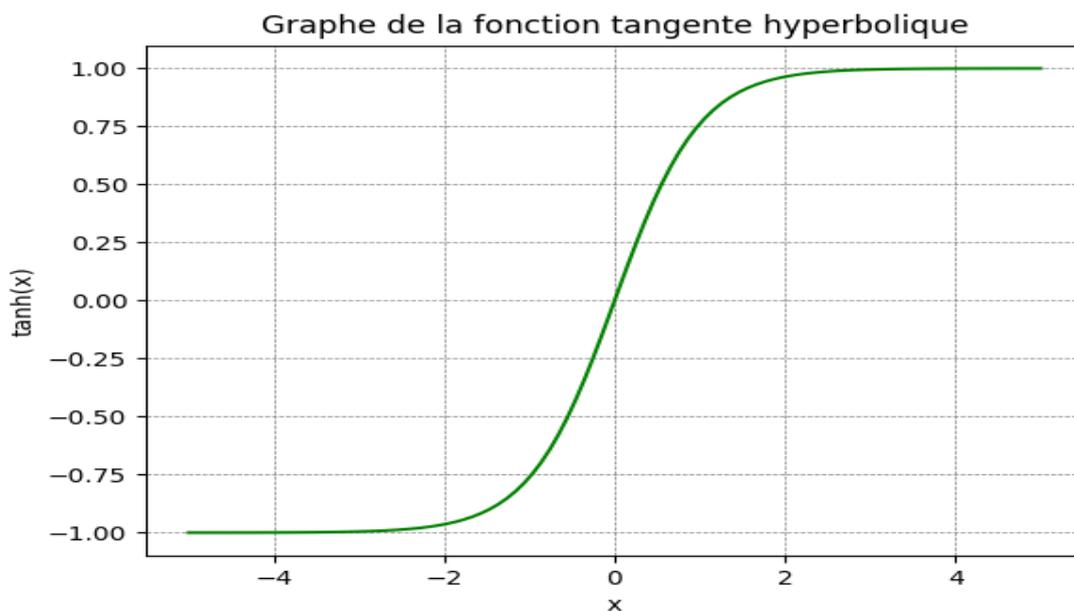
##### □ Types de fonction d'activation

- **Fonction Sigmoidale:** La fonction sigmoïde est une fonction non linéaire qui a une forme en S. Elle est définie comme  $f(x) = 1 / (1 + e^{-x})$ . La sortie de cette fonction est toujours comprise entre 0 et 1. La fonction sigmoïde est souvent utilisée comme fonction d'activation pour les couches cachées des réseaux de neurones. [14]



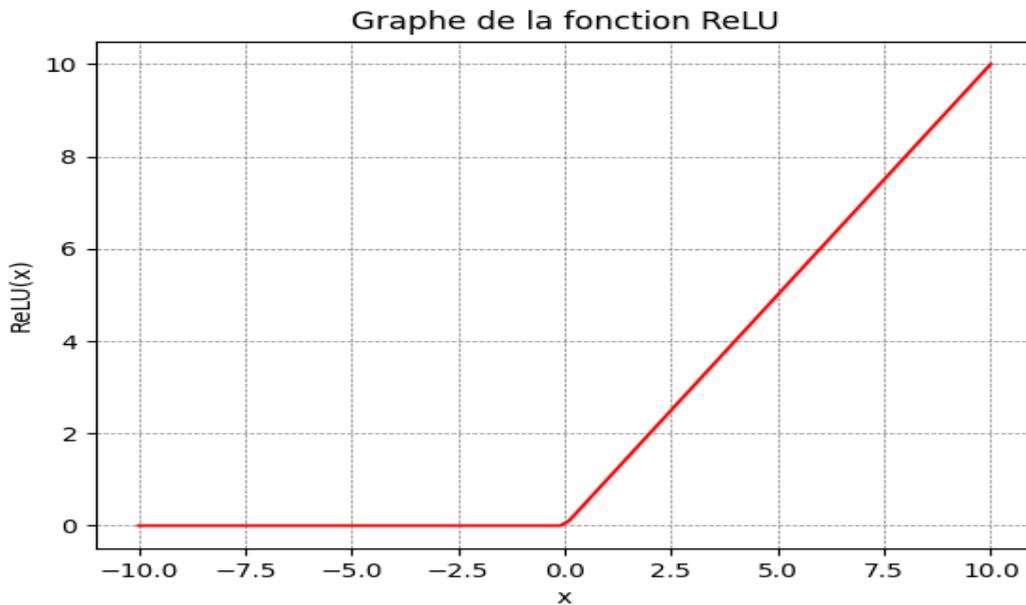
**Figure 2.6.** La courbe de la fonction Sigmoide.

- **Fonction tangente hyperbolique:** La fonction tangente hyperbolique est similaire à la fonction sigmoïde, mais elle a une plage de sortie de -1 à 1. Elle est définie comme  $f(x) = (e^x - e^{-x}) / (e^x + e^{-x})$ . La fonction tangente hyperbolique est également souvent utilisée comme fonction d'activation pour les couches cachées des réseaux de neurones. [14]



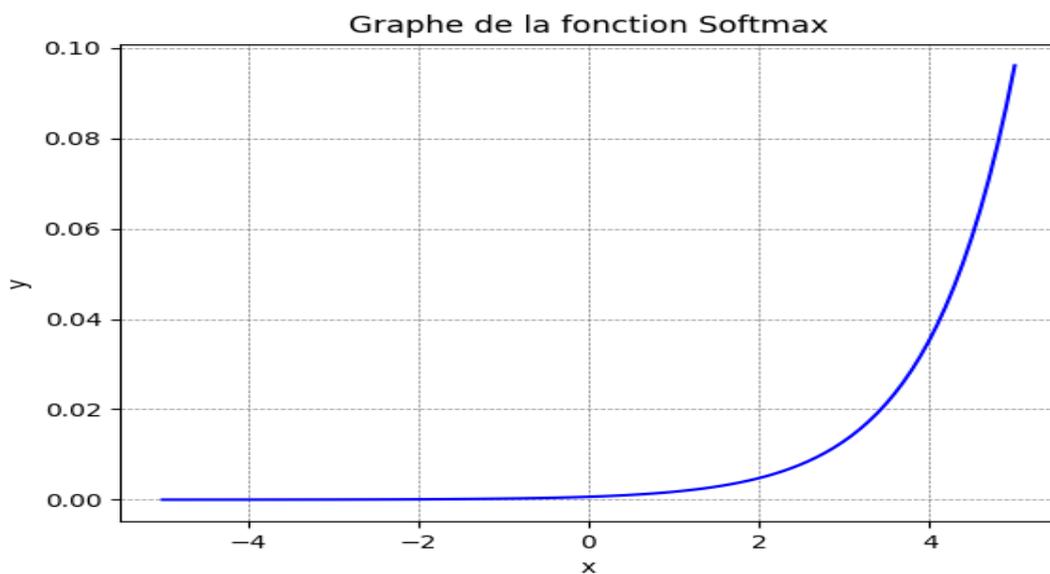
**Figure 2.7.** La courbe de la fonction tangente hyperbolique.

- **Fonction ReLU:** La fonction ReLU (Rectified Linear Unit) est définie comme  $f(x) = \max(0, x)$ . Elle est simple et rapide à calculer, et est souvent utilisée comme fonction d'activation pour les couches cachées des réseaux de neurones profonds. [14]



**Figure 2.8.** La courbe de la fonction ReLU.

- **Fonction Softmax :** La fonction softmax est utilisée comme fonction d'activation pour la couche de sortie des réseaux de neurones lorsqu'il s'agit de résoudre des problèmes de classification multiclasse. Elle prend un vecteur de valeurs en entrée et renvoie un vecteur de probabilités normalisé. [14]



**Figure 2.9.** La courbe de la fonction Softmax.

## 2.2.5 Hyper-paramètres du réseau de neurones

Ce sont des paramètres de configuration qui ne sont pas appris à partir des données, contrairement aux paramètres de poids et de biais. Les hyper-paramètres déterminent la structure et le comportement du réseau de neurones et doivent être choisis avec soin pour garantir des performances optimales. [15]

Les exemples d'hyper-paramètres incluent le nombre de couches et de neurones dans chaque couche, la fonction d'activation, le taux d'apprentissage, le nombre d'itérations d'entraînement, le régularisateur, le type de perte, le type d'optimiseur, etc. Le choix des valeurs optimales pour ces hyper-paramètres est crucial pour obtenir des résultats satisfaisants lors de l'entraînement d'un réseau de neurones. Cela peut être réalisé à travers des méthodes d'optimisation telles que la recherche aléatoire, la recherche en grille, l'optimisation bayésienne, etc. [15]

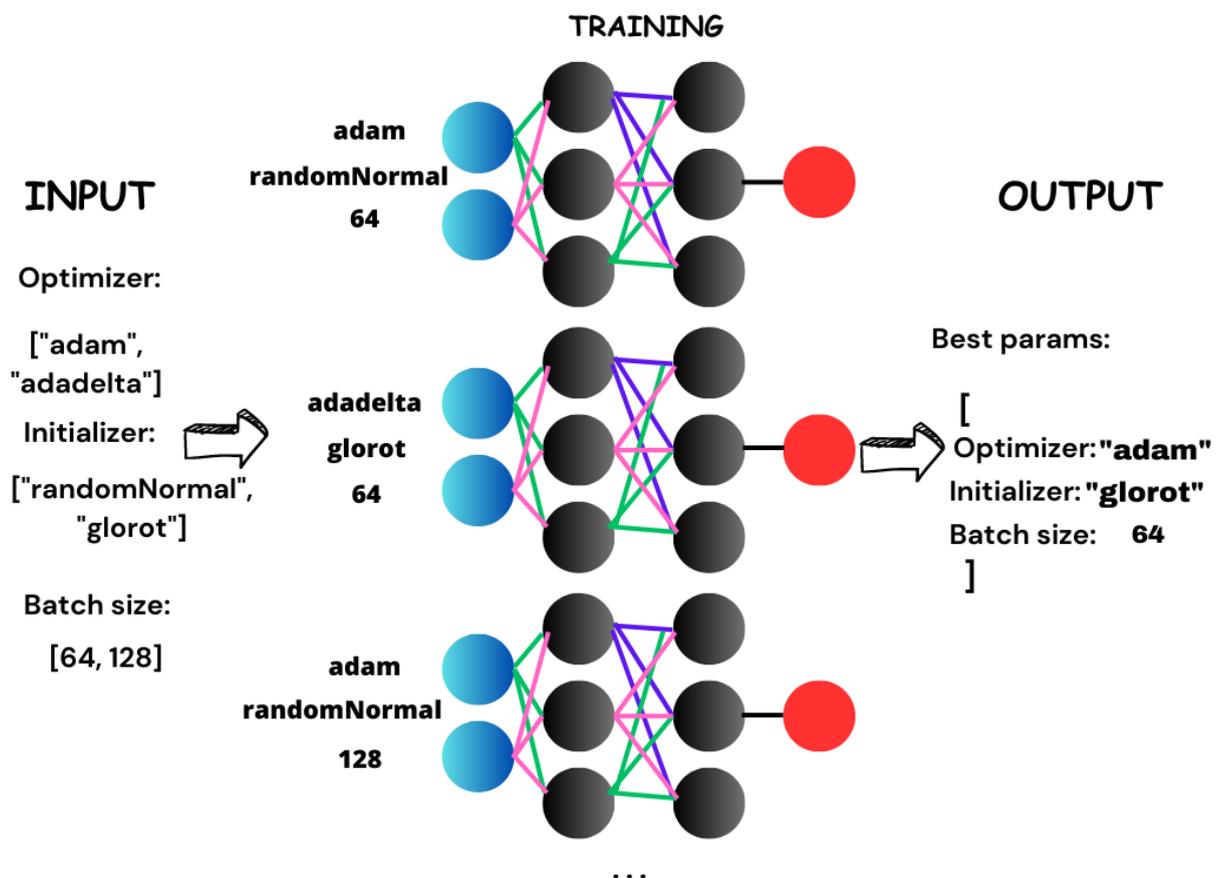


Figure 2.10. Exemple d'optimisations hyper-paramètres.

## □ Stratégies de recherche d'Hyper-paramètres

- **Recherche par grille:** Cette méthode consiste à définir une grille prédéfinie de valeurs pour chaque hyper-paramètre et à évaluer le modèle pour chaque combinaison possible. Cela permet d'explorer systématiquement l'espace des hyper-paramètres, mais peut devenir coûteux en termes de temps de calcul lorsque le nombre d'hyper-paramètres et les valeurs à tester sont élevés. [15]
- **Recherche aléatoire:** Contrairement à la recherche par grille, la recherche aléatoire sélectionne les valeurs des hyper-paramètres de manière aléatoire dans un espace défini. Cette approche permet d'explorer rapidement différentes combinaisons d'hyper-paramètres sans la contrainte d'une grille prédéfinie. Cependant, elle peut ne pas être aussi exhaustive que la recherche par grille. [15]
- **Optimisation bayésienne:** Cette méthode utilise des modèles probabilistes pour estimer l'incertitude de performance en fonction des hyper-paramètres. Elle utilise ensuite ces estimations pour guider la sélection des hyper-paramètres à évaluer. L'optimisation bayésienne permet d'explorer efficacement l'espace des hyper-paramètres en se concentrant sur les régions les plus prometteuses. [15]
- **Optimisation par essais de particules:** Cette méthode s'inspire du comportement des essaims d'oiseaux pour explorer l'espace des hyper-paramètres. Les particules représentent différentes combinaisons d'hyper-paramètres et se déplacent dans l'espace de recherche en utilisant des règles de mise à jour basées sur la performance. L'optimisation par essais de particules peut être efficace pour les espaces de recherche complexes. [15]
- **Recherche en boîte noire:** Cette approche consiste à utiliser des algorithmes d'optimisation basés sur le gradient, tels que la descente de gradient stochastique, pour ajuster les hyper-paramètres du modèle. Cependant, cela peut être coûteux en termes de calcul et nécessite souvent une évaluation fréquente du modèle. [15]

### 2.2.6 Auto-encodeur

L'auto-encodeur est un type de réseau de neurones artificiels non supervisé qui apprend à reconstruire une entrée à partir d'une version compressée de cette entrée. L'objectif principal de l'auto-encodeur est de réduire la dimensionnalité de l'entrée tout en préservant les informations importantes qui la caractérisent. Les auto-encodeurs sont souvent utilisés pour la

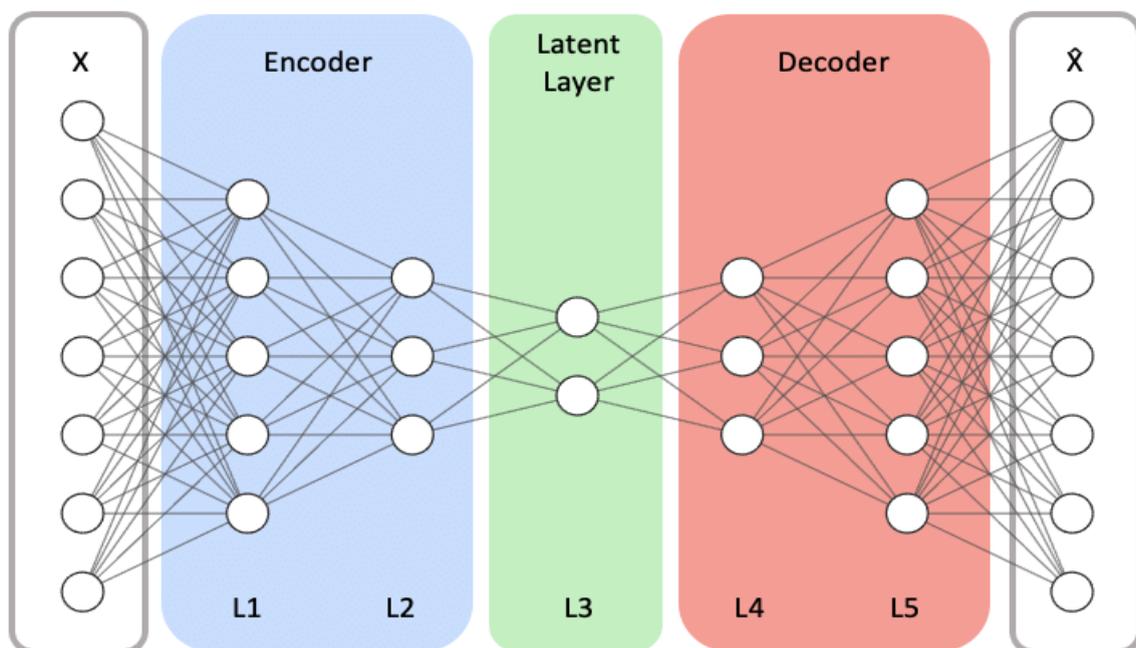
compression de données, la détection d'anomalies, la génération de données, la restauration d'images ou la réduction de bruit. [16]

Un auto-encodeur se compose généralement de deux parties:

- **L'encodeur (bottleneck):** L'encodeur prend en entrée les données d'entrée et les transforme en une représentation compressée, appelée "latent space". [16]
- **Le décodeur:** Il prend cette représentation compressée et tente de la reconstruire en une sortie qui ressemble le plus possible à l'entrée originale. [16]

Pendant l'entraînement, l'auto-encodeur ajuste les poids de l'encodeur et du décodeur pour minimiser l'erreur de reconstruction. [16]

L'une des variantes les plus populaires de l'auto-encodeur est le Variational Autoencoder (VAE), qui utilise un modèle de probabilité pour apprendre la distribution des données en sortie. Les VAE peuvent être utilisés pour générer de nouvelles données à partir de la distribution apprise, ce qui peut être utile dans des applications telles que la génération de visages, la génération de musique, etc. [16]



**Figure 2.11.** Architecture d'un Auto-encodeur.

## □ **Fonctionnement**

Le fonctionnement repose sur la compression des données d'entrée dans une représentation latente de dimension réduite, suivie d'une reconstruction précise à partir de cette représentation. Initialement, les données sont encodées en extrayant les caractéristiques importantes à l'aide d'une couche d'encodage. Ensuite, ces données sont comprimées pour obtenir une représentation latente compacte. Par la suite, un processus de décodage est effectué pour reconstruire les données d'origine à partir de la représentation latente, utilisant une couche de décodage. Le but de l'auto-encodeur est de minimiser la différence entre les données d'entrée et leur reconstruction, en ajustant les poids et les biais du réseau lors de l'apprentissage. En comprimant les données dans une représentation latente, l'auto-encodeur capture les informations essentielles et élimine les détails non importants. Cela permet d'effectuer des tâches telles que la réduction de dimensionnalité ou la détection d'anomalies. [16]

L'auto-encodeur est un outil précieux dans l'apprentissage non supervisé, permettant de découvrir des structures sous-jacentes dans les données sans nécessiter d'étiquettes d'apprentissage. [16]

### **2.2.7 Convolutional Neural Networks (CNNs)**

Les CNNs sont une classe de réseaux de neurones artificiels souvent utilisés pour la reconnaissance d'images, la classification d'images, la segmentation d'images et la détection d'objets. Les CNNs ont été introduits en 1998 par Yann LeCun et ses collègues pour la reconnaissance de chiffres manuscrits. [17]

Ils sont conçus pour exploiter la structure spatiale des données d'entrée, telles que les images. Contrairement aux réseaux de neurones à couches denses, qui traitent l'entrée comme un vecteur de caractéristiques plates, les CNNs sont capables de reconnaître des motifs locaux en utilisant des filtres de convolution qui sont appliqués sur l'image. Les filtres de convolution sont des matrices de poids qui effectuent une opération de multiplication matricielle entre l'image et les filtres pour produire une carte de caractéristiques. [17]

Les réseaux de neurones convolutifs sont souvent constitués de plusieurs couches, telles que les couches de convolution, les couches de pooling et les couches entièrement connectées. [17]

## □ L'architecture du CNN

- **Les couches de convolution (Convolutional layers):** Chaque couche applique une série de filtres de convolution à l'entrée pour extraire des caractéristiques à différentes échelles et niveaux d'abstraction. [17]
- **Les couches de pooling (Pooling layers):** Ces couches réduisent la taille spatiale des cartes de caractéristiques en agrégeant les informations les plus importantes tout en préservant les caractéristiques les plus saillantes. [17]
- **Les couches entièrement connectées (Fully connected layers):** Ces couches prennent en entrée les cartes de caractéristiques obtenues après la dernière couche de pooling et les transforment en une représentation vectorielle de la classe de sortie. [17]

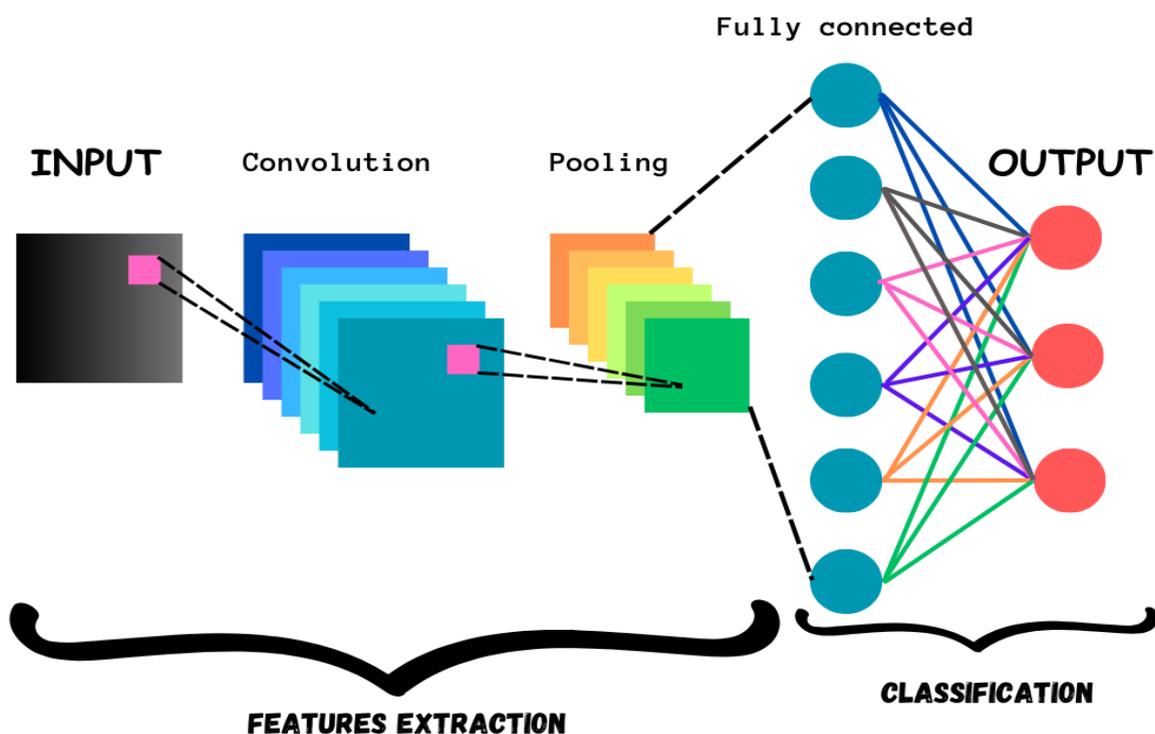


Figure 2.12. Architecture d'un CNN.

## □ Fonctionnement

En commençant par la convolution est appliquée en utilisant des filtres pour détecter des motifs spécifiques dans l'image. Les filtres effectuent des opérations de multiplication et de sommation pour extraire les caractéristiques importantes de l'image, telles que les contours et

les textures. Ensuite, des couches de pooling sont utilisées pour réduire la dimensionnalité de la sortie de la convolution en regroupant les valeurs voisines. Cela permet de conserver les caractéristiques les plus saillantes tout en réduisant la taille de la représentation. Après chaque couche de convolution ou de pooling, une fonction d'activation non linéaire, comme la fonction ReLU, est appliquée pour introduire de la non-linéarité dans le réseau. Enfin, des couches entièrement connectées effectuent la classification en produisant des scores de classe ou des probabilités associées à chaque classe. Pendant l'entraînement, les poids et les biais du réseau sont ajustés itérativement à l'aide de la rétropropagation du gradient et d'un algorithme d'optimisation pour minimiser une fonction de perte. Les CNNs sont extrêmement puissants pour la reconnaissance d'images en raison de leur capacité à capturer les motifs locaux et à exploiter la structure spatiale des données. [17]

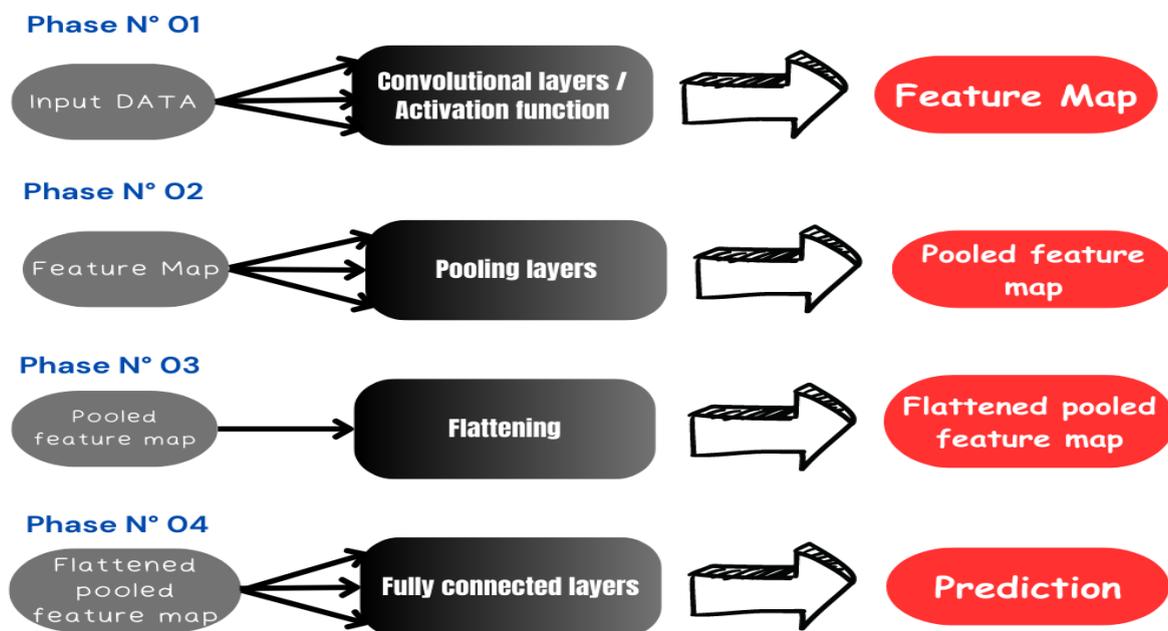


Figure 2.13. Étapes de construction d'un CNN.

## Conclusion

Dans ce chapitre on a parlé du machine learning et ses types, le deep learning, les deep neural networks, le MLP, les fonctions d'activation et les hyper-paramètres d'un réseau de neurones. On a bien expliqué l'architecture d'un CNN et que fait chaque couche.

Dans le chapitre suivant on aura une discussion sur l'entraînement du CNN, la prédiction et les résultats obtenus.

## Introduction

Dans ce dernier chapitre on va présenter comment on a entraîné notre CNN ainsi que sa performance, Ensuite nous présenterons les différents résultats obtenus et la prédiction finale.

### 3.1 Python

#### 3.1.1 Définition

Python est un langage de programmation qui se distingue par sa haute abstraction, son interprétation directe du code et son approche orientée objet. Il a été créé par Guido van Rossum et a été initialement publié en 1991. Python se distingue par sa simplicité, sa lisibilité et son expressivité, ce qui en fait un langage accessible tant aux débutants qu'aux programmeurs expérimentés. [18]

#### 3.1.2 Histoire et origines

Python a été créé par Guido van Rossum, un informaticien néerlandais, à la fin des années 1980. La première version publique, Python 0.9.0, a été publiée en février 1991. Guido van Rossum visait à développer un langage de programmation facile à lire, à écrire et à comprendre. Le nom de Python a été inspiré de la série télévisée “Monty Python's Flying Circus”. Depuis sa création, Python a gagné en popularité grâce à sa simplicité et sa polyvalence. Il a évolué grâce aux contributions d'une communauté dévouée et passionnée de développeurs. Aujourd'hui, Python est largement reconnu comme l'un des langages de programmation les plus populaires et les plus utilisés, avec un écosystème florissant de bibliothèques, de frameworks et d'outils qui étendent ses capacités. [18]

#### 3.1.3 Que nous offre Python ?

- Simplicité et lisibilité: Python se distingue par sa syntaxe claire et sa simplicité d'utilisation. Son code est facile à lire, ce qui facilite la collaboration entre les développeurs et la maintenance du code. [18]

- Polyvalence: Python est un langage polyvalent qui peut être utilisé dans une large gamme de domaines, tels que le développement web, l'analyse de données, l'intelligence artificielle, l'apprentissage automatique, la robotique et bien plus encore. Il dispose d'une vaste bibliothèque standard et de nombreux packages tiers qui facilitent le développement d'applications dans différents domaines. [18]
- Grande communauté et écosystème actif: Python bénéficie d'une communauté de développeurs active et engagée. Il existe de nombreux forums, tutoriels, documentations et ressources en ligne qui facilitent l'apprentissage et la résolution des problèmes. De plus, la communauté Python crée et maintient régulièrement de nouvelles bibliothèques, frameworks et outils, enrichissant ainsi l'écosystème Python. [18]
- Productivité élevée: La syntaxe simple de Python, sa gestion automatique de la mémoire et sa large gamme de bibliothèques prêtes à l'emploi permettent aux développeurs de gagner du temps et d'être plus productifs. Python favorise un style de développement efficace et rapide, ce qui en fait un choix populaire pour les projets de toutes tailles. [18]
- Intégration facile: Python peut être facilement intégré avec d'autres langages de programmation. Il offre des interfaces pour de nombreux systèmes et bibliothèques, permettant aux développeurs de combiner les avantages de Python avec d'autres technologies. [18]
- Apprentissage et adoption faciles: Python est souvent recommandé comme premier langage de programmation pour les débutants. Sa syntaxe simple et lisible en font un excellent choix pour apprendre les concepts de base de la programmation. De plus, sa popularité signifie qu'il existe de nombreuses ressources d'apprentissage disponibles, notamment des cours en ligne, des tutoriels et des livres. [18]
- Portabilité: Python est compatible avec plusieurs plates-formes, y compris les principaux systèmes d'exploitation tels que Windows, macOS et Linux. Les programmes Python peuvent être exécutés sans modification sur différentes plates-formes, ce qui facilite le déploiement sur différents environnements. [18]

### **3.1.4 Programmation orientée objet (OOP)**

Python est un langage de programmation qui prend en charge pleinement la programmation orientée objet. La programmation orientée objet est un paradigme de programmation qui

permet d'organiser et de structurer le code en utilisant des objets, qui sont des instances de classes. [18]

L'utilisation de la programmation orientée objet en Python présente plusieurs avantages:

- Elle favorise la réutilisabilité du code en créant des classes génériques.
- Elle permet la modularité en divisant le code en modules autonomes.
- Elle facilite la représentation d'abstractions du monde réel grâce à l'utilisation d'objets.
- Elle prend en charge l'héritage et le polymorphisme, offrant ainsi une plus grande flexibilité et réutilisabilité du code. [18]

### **3.1.5 Nature interprétée et interactive**

Python est un langage de programmation interprété, ce qui signifie qu'il est exécuté ligne par ligne sans nécessiter de processus de compilation préalable. [18]

La nature interprétée et interactive de Python offre un développement rapide, une expérimentation facile, un débogage interactif, une facilité d'apprentissage et une grande portabilité. Les développeurs peuvent écrire et tester immédiatement leur code, expérimenter des instructions individuelles, obtenir des informations détaillées lors du débogage, faciliter l'apprentissage des débutants et exécuter des programmes Python sur différentes plates-formes sans modification supplémentaire. [18]

### **3.1.6 Philosophie Open-source**

Un langage de programmation open-source signifie que son code source est librement disponible et peut être modifié et distribué par la communauté des développeurs. Cette philosophie open-source présente plusieurs avantages:

- Collaboration communautaire: Python encourage la collaboration et le partage des connaissances, permettant aux développeurs de contribuer à son amélioration continue.
- Écosystème riche: Python bénéficie d'un vaste écosystème d'outils, bibliothèques et frameworks open-source qui étendent ses fonctionnalités et facilitent le développement dans différents domaines.
- Flexibilité et personnalisation: La nature open-source de Python offre aux développeurs une grande flexibilité pour adapter le langage à leurs besoins spécifiques et créer des solutions personnalisées.

- **Transparence et confiance:** La disponibilité du code source de Python favorise la transparence, permettant aux développeurs de comprendre en détail son fonctionnement et de vérifier sa qualité, sa sécurité et sa fiabilité.
- **Accessibilité:** Python étant open-source, il est accessible gratuitement à tous, éliminant les barrières financières et permettant à un large public d'apprendre et de développer avec le langage. [18]

## 3.2 Construire le modèle CNN

### 3.2.1 Collecte et prétraitement des données

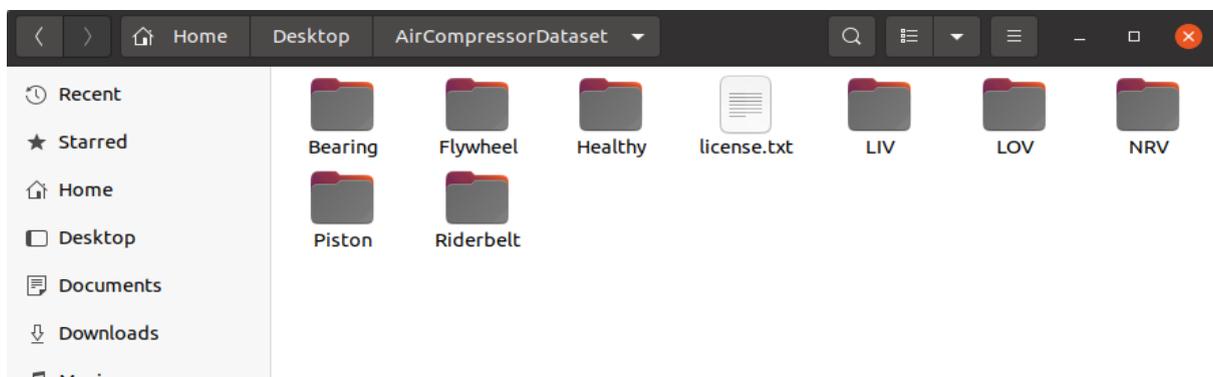


Figure 3.1. Les classes de la Dataset.

```

20 # Set the sampling rate for the audio files
21 sampling_rate = 16000
22
23 # Set the duration for the audio files (in seconds)
24 duration = 3
25
26 # Set the number of mel-frequency cepstral coefficients (MFCCs)
27 n_mfcc = 20
28
29 # Load the audio files and extract the MFCCs
30 X = []
31 y = []
32 max_len = 0
33 for i, cls in enumerate(classes):
34     class_directory = os.path.join(data_directory, cls)
35     for file in os.listdir(class_directory):
36         file_path = os.path.join(class_directory, file)
37         # Get the file name without extension
38         #file_name = os.path.splitext(os.path.basename(file_path))[0]
39         signal, sr = librosa.load(file_path, sr=sampling_rate, duration=duration)
40         mfccs = librosa.feature.mfcc(y=signal, sr=sr, n_mfcc=n_mfcc)
41         # Pad the feature matrix with zeros to match the largest matrix
42         if mfccs.shape[1] > max_len:
43             max_len = mfccs.shape[1]
44         X.append(mfccs)
45         y.append(i)
46         # Shift the audio forward by the specified amount of time
47         stretched_audio = librosa.effects.time_stretch(signal, rate=1.2)
48         shifted_audio = librosa.effects.pitch_shift(stretched_audio, sr=sampling_rate, n_steps=4)
49         mfccs = librosa.feature.mfcc(y=shifted_audio, sr=sr, n_mfcc=n_mfcc)
50         # Pad the feature matrix with zeros to match the largest matrix
51         if mfccs.shape[1] > max_len:
52             max_len = mfccs.shape[1]
53         X.append(mfccs)
54         y.append(i)

```

Figure 3.2. Prétraitement de données.

On fait le prétraitement des données audio en effectuant des étapes telles que l'augmentation de données, l'extraction de caractéristiques audio pertinentes et la division de l'ensemble de données en ensembles d'**entraînement** et de **test**.

L'augmentation de données qu'on l'utilise notamment dans le domaine de la vision par ordinateur et du traitement de l'audio. Elle consiste à appliquer des transformations aléatoires ou contrôlées aux données d'entraînement existantes afin de créer de nouvelles variations des exemples d'apprentissage.

### 3.2.2 Conception de l'architecture

```
76 # Define the CNN architecture
77 model = Sequential()
78 model.add(Conv2D(32, kernel_size=(3, 3), activation='relu', input_shape=(X_train.shape[1], X_train.shape[2], 1)))
79 model.add(MaxPooling2D(pool_size=(2, 2)))
80 model.add(Conv2D(64, kernel_size=(3, 3), activation='relu'))
81 model.add(MaxPooling2D(pool_size=(2, 2)))
82 model.add(Flatten())
83 model.add(Dense(128, activation='relu'))
84 model.add(Dropout(0.25))
85 model.add(Dense(len(classes), activation='softmax'))
```

**Figure 3.3.** Création du CNN.

Le processus d'obtention des MFCCs implique d'effectuer une série d'étapes de traitement du signal, y compris le cadrage du signal audio, le calcul de la transformée de Fourier...etc. Le résultat de ce processus est une représentation matricielle où les lignes correspondent à différentes bandes de fréquences et les colonnes représentent les trames ou segments de l'audio.

Étant donné que la représentation MFCC est intrinsèquement 2D, il convient d'appliquer des couches convolutives 2D pour capturer les modèles spatiaux et les dépendances dans les dimensions de fréquence et de temps.

Tant qu'on a utilisé une couche convolutive 2D, il est courant de la faire suivre d'une couche de regroupement 2D pour réduire les dimensions spatiales des cartes d'entités.

L'objectif principal du flattening est de transformer les informations multidimensionnels en un vecteur unidimensionnel afin de les utiliser comme entrée pour les couches entièrement connectées (fully connected layers) qui suivent.

On a utilisé une couche entièrement connectée (128 neurones) et elle pouvait sembler excessive dans la plupart des cas mais on a essayé d'assurer la bonne prédiction.

On note que l'utilisation d'un grand nombre de couches entièrement connectées peut poser quelques problèmes:

- **Le surapprentissage:** Le modèle s'adapte trop spécifiquement aux données d'entraînement et ne généralise pas bien aux nouvelles données
- **Temps d'entraînement:** Plus le modèle est complexe, plus le temps d'entraînement sera long.
- **Interprétabilité:** Rendre le modèle plus difficile d'analyser les poids et les activations des neurones pour comprendre comment le modèle prend des décisions.

La couche de sortie dépend du nombre de classes de notre dataset, dans notre cas on a 8 classes et donc 8 sorties.

### 3.2.3 Entraînement

```
87 # Compile the model
88 model.compile(loss='categorical_crossentropy', optimizer='adam', metrics=['accuracy', keras.metrics.Precision()])
89
90 # Train the model
91 history = model.fit(X_train, y_train, batch_size=32, epochs=20, verbose=1, validation_data=(X_test, y_test))
92
93 # Evaluate the model on the testing set
94 test_loss, test_acc, test_precision = model.evaluate(X_test, y_test)
```

**Figure 3.4.** Compilation, l'entraînement et l'évaluation du modèle.

On commence par la compilation pour configurer les éléments nécessaires pour l'entraînement:

- **loss:** On a choisi "categorical\_crossentropy" qui est utilisée dans les problèmes de classification multiclasse où les étiquettes sont encodées de manière one-hot.
- **optimizer:** On l'utilise pour ajuster les poids du modèle pendant l'entraînement afin de minimiser la fonction de perte, "adam" converge plus rapidement et de manière plus efficace.
- **metrics:** évaluer les performances du modèle pendant l'entraînement et l'évaluation, on a choisi que notre modèle soit évalué selon la précision et l'exactitude.

On passe vers l'entraînement qui prend quelque arguments:

- **X\_train & y\_train**: L'ensemble des données et les étiquettes d'entraînement.
- **batch\_size**: Le nombre d'exemples d'entraînement qui sont utilisés pour une seule mise à jour des poids du modèle.
- **epochs**: Le nombre d'itérations complètes que le modèle effectue sur l'ensemble des données d'entraînement.
- **verbose** (optionnel): Un paramètre qui contrôle la quantité d'informations affichées pendant l'entraînement.
- **validation\_data**: C'est un tuple contenant les données de validation et les étiquettes de validation.

Pour la dernière étape “l'évaluation”, La fonction “**evaluate**” calcule la perte (loss) et les métriques spécifiées lors de la compilation du modèle sur les données de test (la précision et l'exactitude dans notre cas).

### 3.2.4 Visualisation des résultats de L'entraînement

```
105 # Plot the loss over epochs
106 plt.plot(history.history['loss'], color="red")
107 plt.plot(history.history['val_loss'], color="green")
108 plt.title('Model loss')
109 plt.ylabel('Loss')
110 plt.xlabel('Epoch')
111 plt.legend(['Train', 'Validation'], loc='upper left')
112 plt.show()
113
114 # Ajouter une grille de lignes en pointillé dans le fond du graphique
115 plt.grid(color='gray', linestyle='--', linewidth=0.5)
116
117 # Plot the accuracy over epochs
118 plt.plot(history.history['accuracy'], color="magenta")
119 plt.plot(history.history['val_accuracy'], color="blue")
120 plt.title('Model accuracy')
121 plt.ylabel('Accuracy')
122 plt.xlabel('Epoch')
123 plt.legend(['Train', 'Validation'], loc='upper left')
124 plt.show()
125
126 # Ajouter une grille de lignes en pointillé dans le fond du graphique
127 plt.grid(color='gray', linestyle='--', linewidth=0.5)
128
129 # Plot the precision over epochs
130 plt.plot(history.history['precision'], color="yellow")
131 plt.plot(history.history['val_precision'], color="orange")
132 plt.title('Model precision')
133 plt.ylabel('Precision')
134 plt.xlabel('Epoch')
135 plt.legend(['Train', 'Validation'], loc='upper right')
136 plt.show()
```

**Figure 3.5.** Tracer les courbes de la perte et les métriques.

Cette étape n'est pas nécessaire pour construire un modèle CNN mais elle est importante pour visualiser la performance de notre CNN. Elle trace les courbes de la perte et les métriques d'évaluation qu'on a choisies.

Ces courbes peuvent déterminer si les paramètres qu'on a utilisés sont parfaits ou bien on doit changer quelques paramètres pour trouver la combinaison parfaite.

### 3.3 Fonctionnement et test

#### 3.3.1 L'entraînement du CNN

```
Epoch 1/20
90/90 [=====] - 5s 40ms/step - loss: 2.2783 - accuracy: 0.5760 - precision: 0.7416 - val_loss: 0.2796 - val_accuracy:
0.9056 - val_precision: 0.9322
Epoch 2/20
90/90 [=====] - 4s 43ms/step - loss: 0.2239 - accuracy: 0.9267 - precision: 0.9404 - val_loss: 0.1826 - val_accuracy:
0.9319 - val_precision: 0.9486
Epoch 3/20
90/90 [=====] - 3s 33ms/step - loss: 0.1763 - accuracy: 0.9424 - precision: 0.9481 - val_loss: 0.2689 - val_accuracy:
0.8944 - val_precision: 0.9044
Epoch 4/20
90/90 [=====] - 3s 33ms/step - loss: 0.0825 - accuracy: 0.9733 - precision: 0.9773 - val_loss: 0.0973 - val_accuracy:
0.9639 - val_precision: 0.9678
Epoch 5/20
90/90 [=====] - 2s 27ms/step - loss: 0.1091 - accuracy: 0.9615 - precision: 0.9667 - val_loss: 0.2189 - val_accuracy:
0.9194 - val_precision: 0.9320
Epoch 6/20
90/90 [=====] - 3s 29ms/step - loss: 0.0760 - accuracy: 0.9747 - precision: 0.9776 - val_loss: 0.0564 - val_accuracy:
0.9806 - val_precision: 0.9806
Epoch 7/20
90/90 [=====] - 2s 27ms/step - loss: 0.0431 - accuracy: 0.9878 - precision: 0.9882 - val_loss: 0.0540 - val_accuracy:
0.9833 - val_precision: 0.9847
Epoch 8/20
90/90 [=====] - 3s 28ms/step - loss: 0.0266 - accuracy: 0.9913 - precision: 0.9920 - val_loss: 0.0358 - val_accuracy:
0.9903 - val_precision: 0.9903
Epoch 9/20
90/90 [=====] - 3s 28ms/step - loss: 0.0658 - accuracy: 0.9802 - precision: 0.9819 - val_loss: 0.1636 - val_accuracy:
0.9444 - val_precision: 0.9483
Epoch 10/20
90/90 [=====] - 3s 28ms/step - loss: 0.0427 - accuracy: 0.9872 - precision: 0.9885 - val_loss: 0.0665 - val_accuracy:
0.9792 - val_precision: 0.9791
```

Figure 3.6. L'entraînement du CNN (Epoch N° 00 ⇒ Epoch N° 10).

```
Epoch 11/20
90/90 [=====] - 3s 32ms/step - loss: 0.0814 - accuracy: 0.9733 - precision: 0.9742 - val_loss: 0.0353 - val_accuracy:
0.9889 - val_precision: 0.9889
Epoch 12/20
90/90 [=====] - 3s 33ms/step - loss: 0.0409 - accuracy: 0.9844 - precision: 0.9850 - val_loss: 0.0838 - val_accuracy:
0.9694 - val_precision: 0.9694
Epoch 13/20
90/90 [=====] - 3s 30ms/step - loss: 0.0471 - accuracy: 0.9844 - precision: 0.9857 - val_loss: 0.0996 - val_accuracy:
0.9736 - val_precision: 0.9736
Epoch 14/20
90/90 [=====] - 3s 31ms/step - loss: 0.0335 - accuracy: 0.9892 - precision: 0.9896 - val_loss: 0.0351 - val_accuracy:
0.9847 - val_precision: 0.9847
Epoch 15/20
90/90 [=====] - 3s 29ms/step - loss: 0.0206 - accuracy: 0.9931 - precision: 0.9937 - val_loss: 0.0654 - val_accuracy:
0.9764 - val_precision: 0.9764
Epoch 16/20
90/90 [=====] - 2s 26ms/step - loss: 0.0310 - accuracy: 0.9889 - precision: 0.9889 - val_loss: 0.0515 - val_accuracy:
0.9792 - val_precision: 0.9819
Epoch 17/20
90/90 [=====] - 2s 25ms/step - loss: 0.0106 - accuracy: 0.9972 - precision: 0.9976 - val_loss: 0.0355 - val_accuracy:
0.9833 - val_precision: 0.9833
Epoch 18/20
90/90 [=====] - 2s 25ms/step - loss: 0.0078 - accuracy: 0.9972 - precision: 0.9976 - val_loss: 0.0656 - val_accuracy:
0.9833 - val_precision: 0.9833
Epoch 19/20
90/90 [=====] - 3s 30ms/step - loss: 0.0436 - accuracy: 0.9875 - precision: 0.9882 - val_loss: 0.1132 - val_accuracy:
0.9556 - val_precision: 0.9596
Epoch 20/20
90/90 [=====] - 2s 27ms/step - loss: 0.0410 - accuracy: 0.9858 - precision: 0.9871 - val_loss: 0.0405 - val_accuracy:
0.9875 - val_precision: 0.9875
```

Figure 3.7. L'entraînement du CNN (Epoch N° 11 ⇒ Epoch N° 20).

#### ★ Commentaires

- Le modèle a été entraîné sur un total de 20 époques, ce qui représente le nombre de fois où l'ensemble des données d'entraînement a été passé dans le modèle.
- Au fur et à mesure que les époques avançaient, nous avons pu observer une amélioration continue des performances du modèle (Une exactitude et une précision augmentées, une perte diminuée).

→ La convergence du modèle a été atteinte après 08 époques comme le montrent des valeurs d'exactitude constamment élevées.. Cela indique que le modèle a appris à généraliser correctement sur l'ensemble de données de test. Cependant, l'entraînement sur 20 époques a été réalisé afin de garantir que le modèle ait eu suffisamment d'occasions de se spécialiser et d'optimiser ses performances sur l'ensemble de données.

### 3.3.2 L'évaluation du CNN

```
23/23 [=====] - 0s 10ms/step - loss: 0.0405 - accuracy: 0.9875 - precision: 0.9875  
Test loss: 0.040522314608097076  
Test accuracy: 0.987500011920929  
Test precision: 0.9874826073646545
```

Figure 3.8. Les valeurs finales de métriques de l'évaluation.

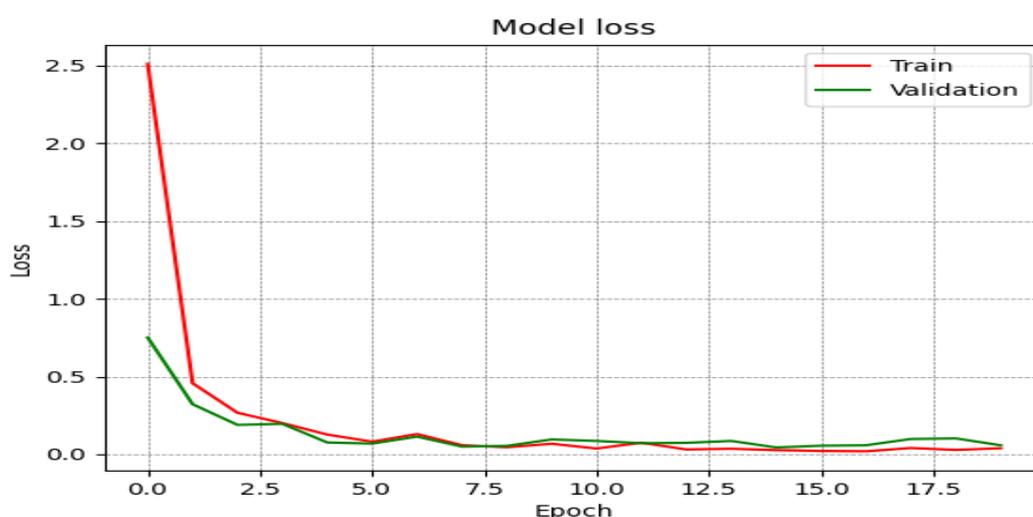


Figure 3.9. La courbe de la perte (Loss).

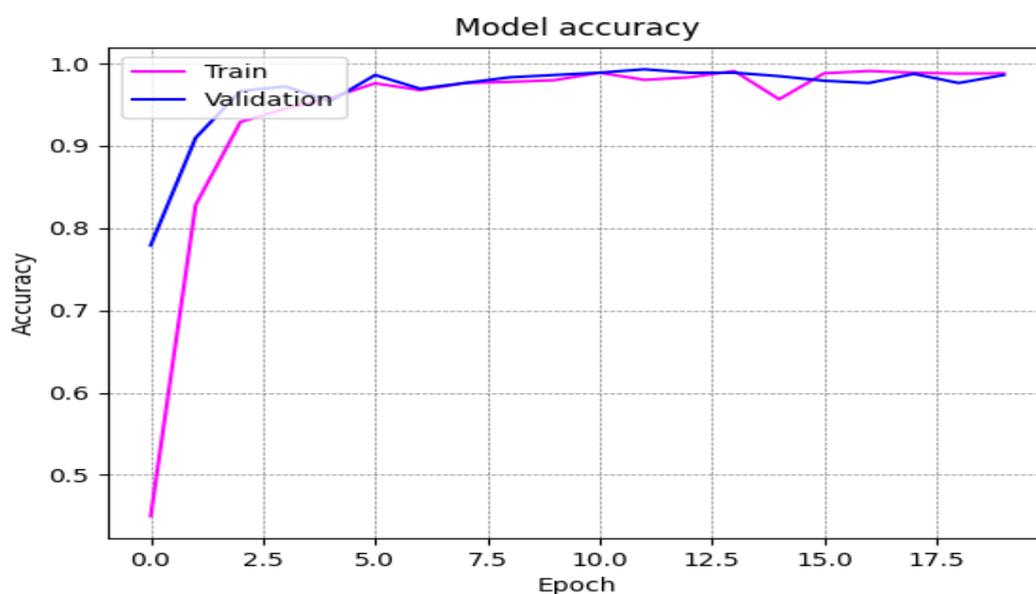
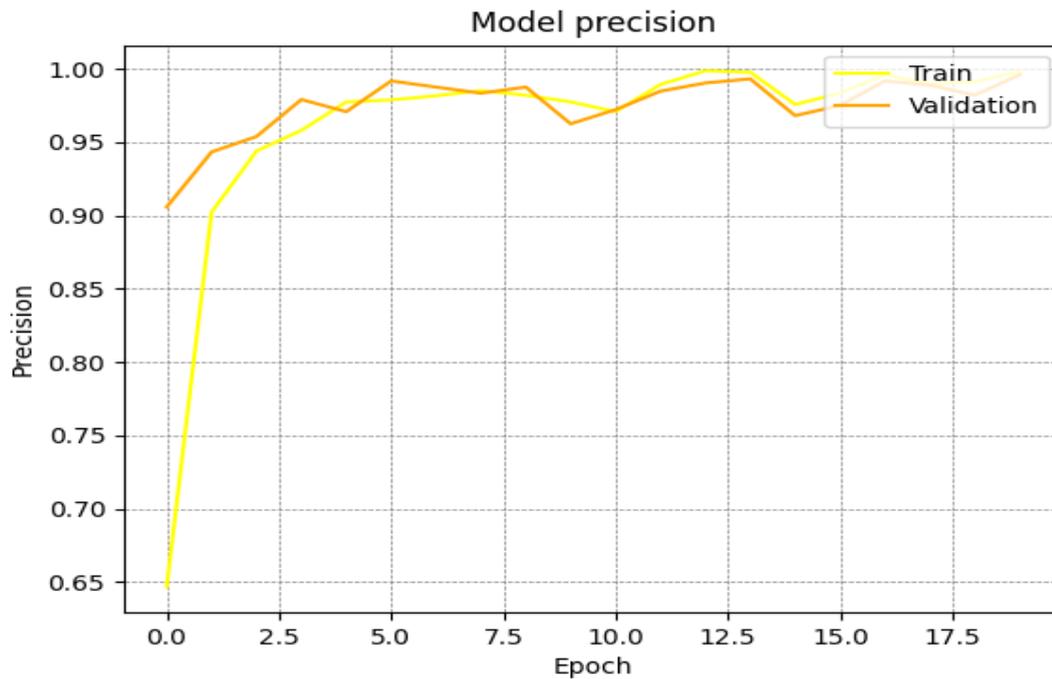


Figure 3.10. La courbe de l'exactitude (accuracy).



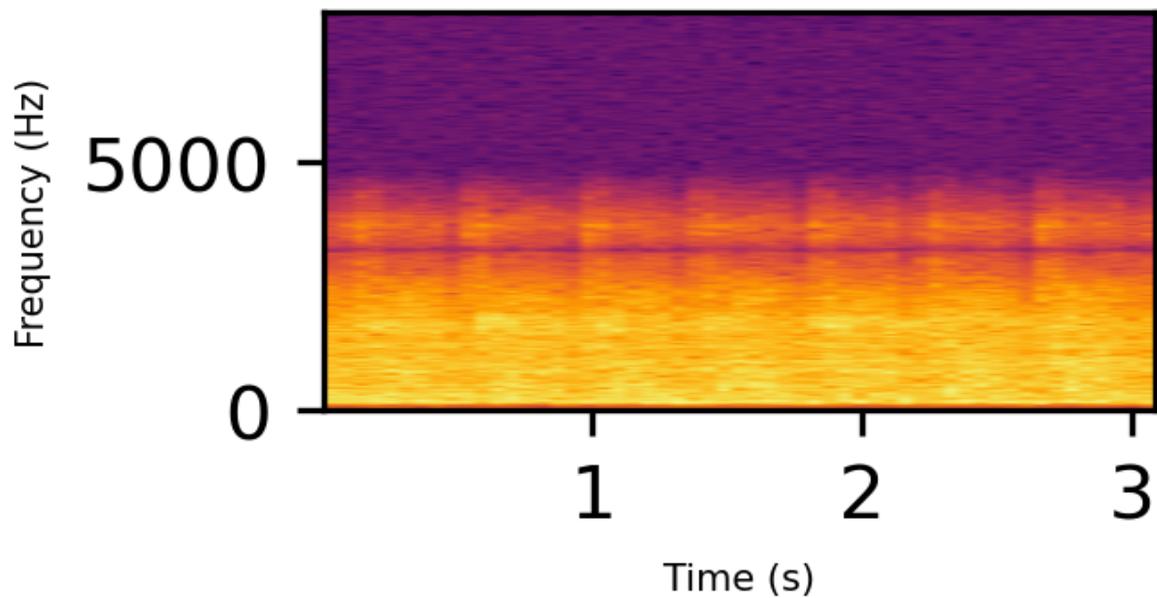
**Figure 3.11.** La courbe de la Précision (Precision).

### ★ Commentaires

- La courbe de perte (Loss) démontre une diminution constante au fil des époques, indiquant que notre modèle apprend à mieux ajuster les données d'entraînement, la perte a continué à diminuer progressivement jusqu'à atteindre 0,0405.
- Nous constatons une croissance constante de la courbe d'exactitude (Accuracy), ce qui signifie que notre modèle parvient à effectuer des prédictions plus précises à mesure qu'il s'entraîne davantage, l'exactitude a continué à augmenter progressivement jusqu'à atteindre 0,9875.
- La croissance régulière de la courbe de précision (precision) suggère que notre modèle parvient à minimiser les faux positifs et à fournir des résultats plus fiables pour les classes positives, la précision a continué à augmenter progressivement jusqu'à atteindre 0,9874.

### 3.3.3 Résultat de la prédiction

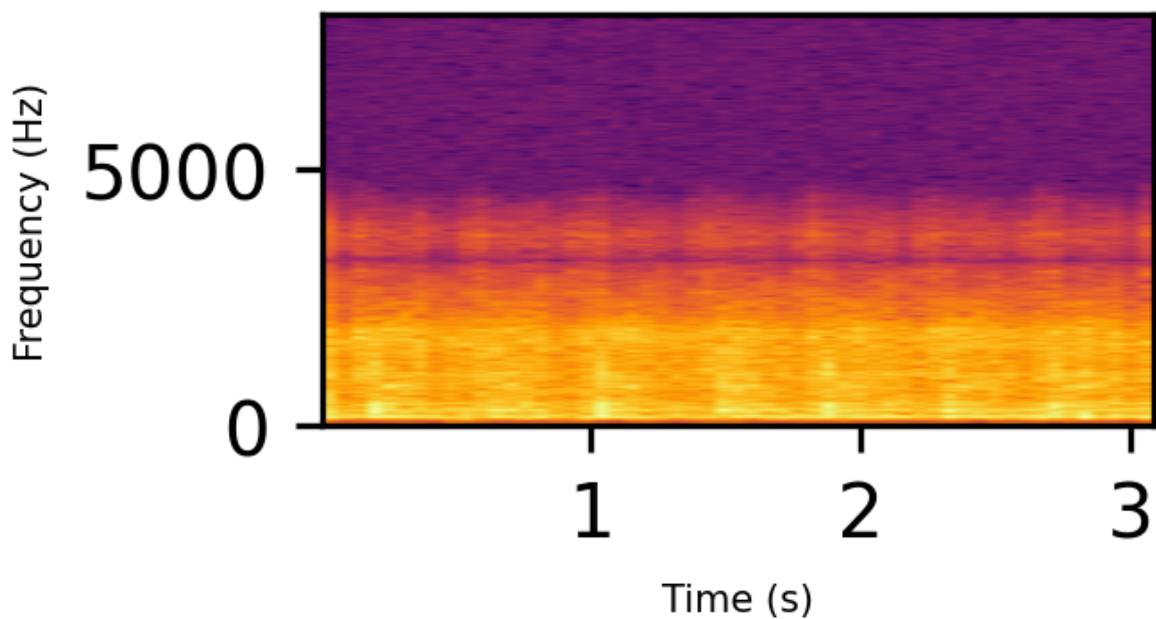
Le spectrogramme:



Prédiction: En bon état!

**Figure 3.12.** La prédiction et le spectrogramme d'un Audio normal.

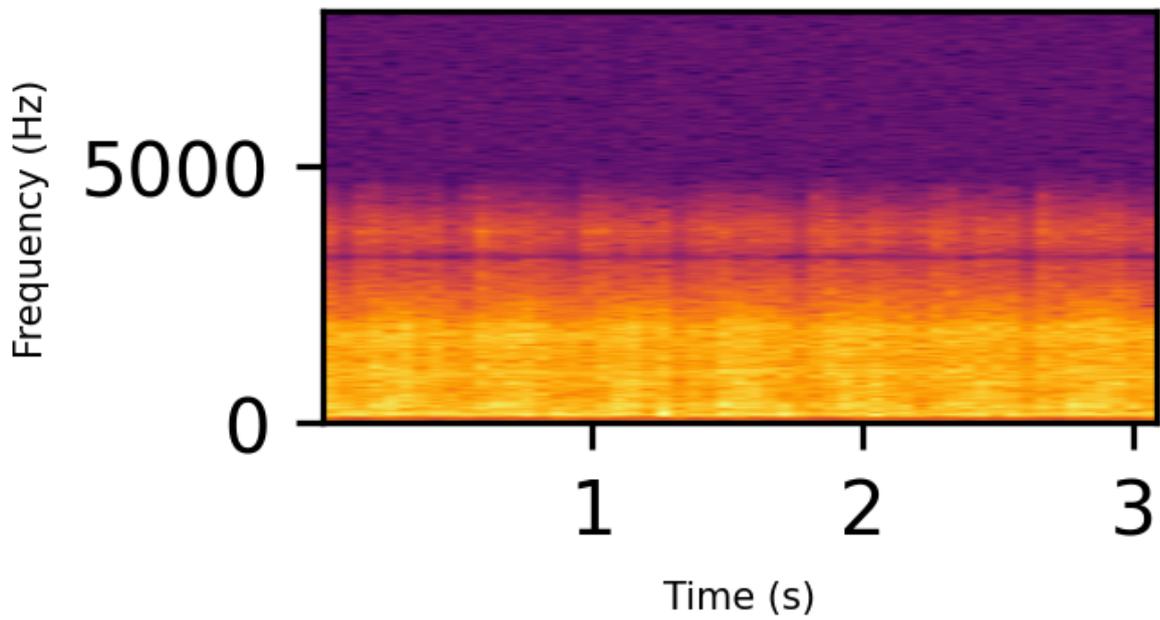
Le spectrogramme:



Prédiction: Défectueux! Il y a une faute de palier (Bearing)

**Figure 3.13.** La prédiction et le spectrogramme d'un Audio (une faute "Bearing").

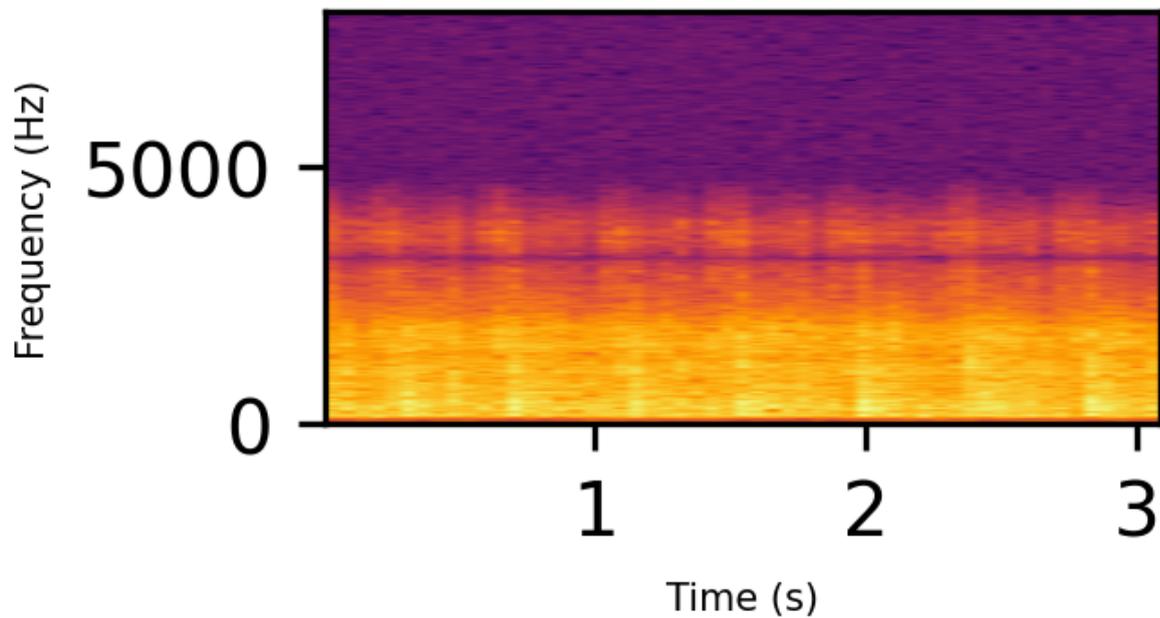
Le spectrogramme:



Prédiction: Défectueux! Il y a une faute de volant (Flywheel)

**Figure 3.14.** La prédiction et le spectrogramme d'un Audio (une faute "Flywheel").

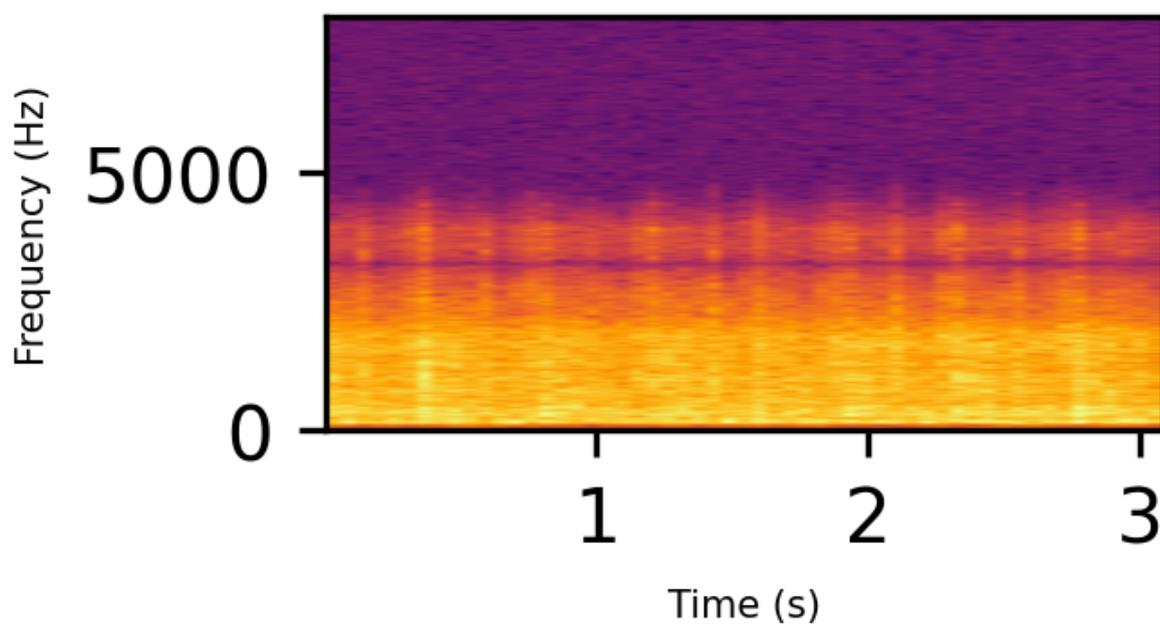
Le spectrogramme:



Prédiction: Défectueux! Il y a une faute de vanne d'admission de fuite (LIV)

**Figure 3.15.** La prédiction et le spectrogramme d'un Audio (une faute "LIV").

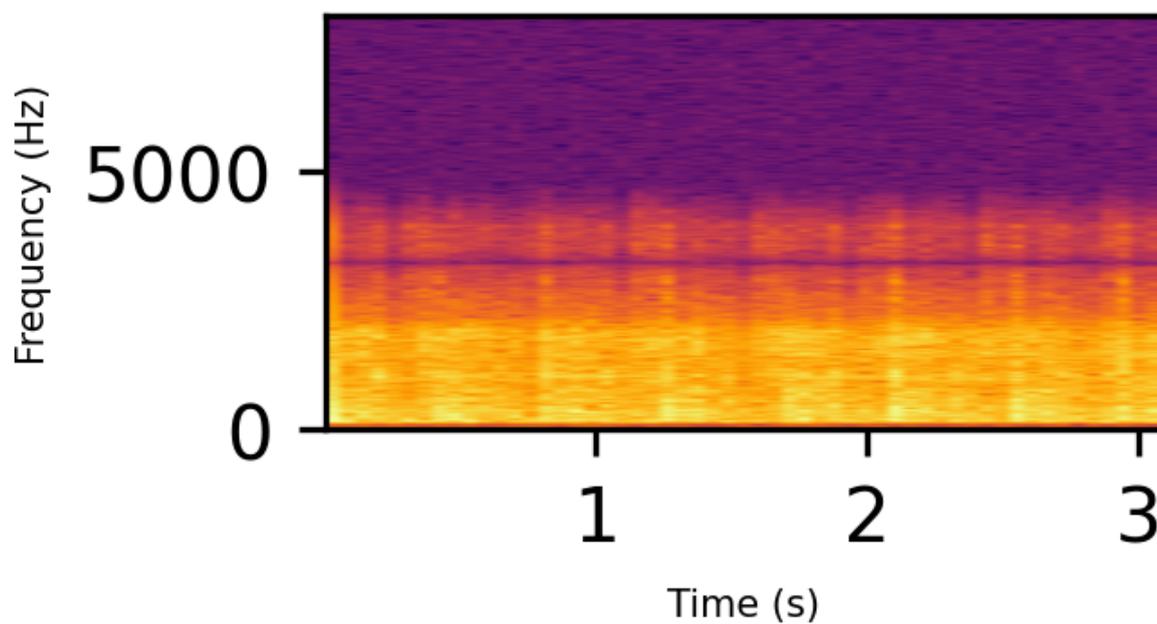
Le spectrogramme:



Prédiction: Défectueuse! Il y a une fuite de vanne de sortie de fuite (LOV)

**Figure 3.16.** La prédiction et le spectrogramme d'un Audio (une faute "LOV").

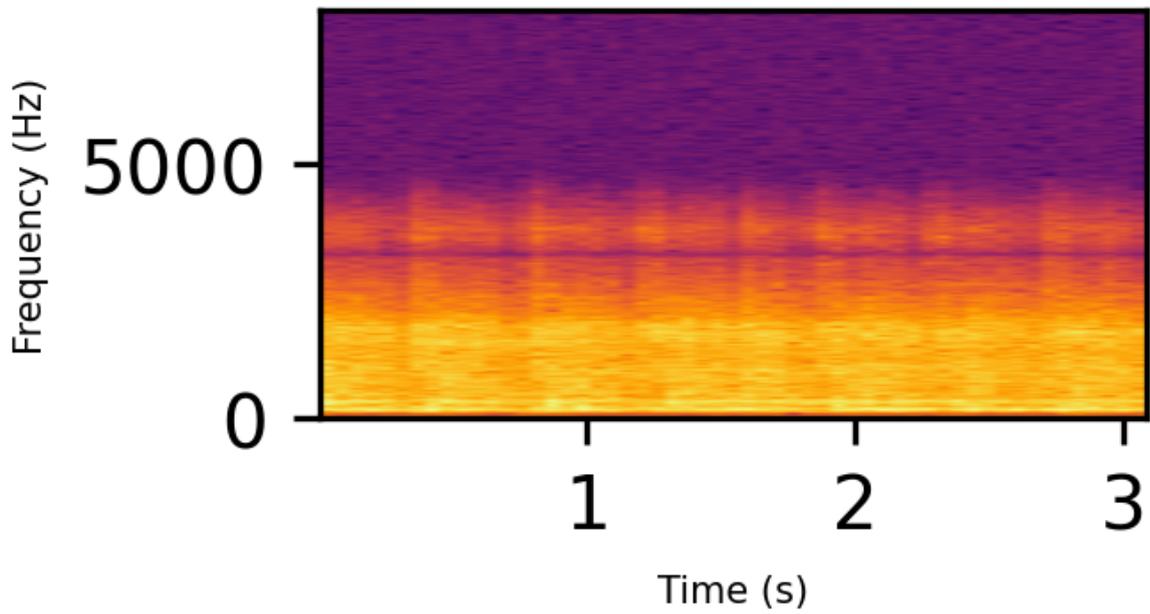
Le spectrogramme:



Prédiction: Défectueuse! Il y a une fuite du clapet anti-retour (NRV)

**Figure 3.17.** La prédiction et le spectrogramme d'un Audio (une faute "NRV").

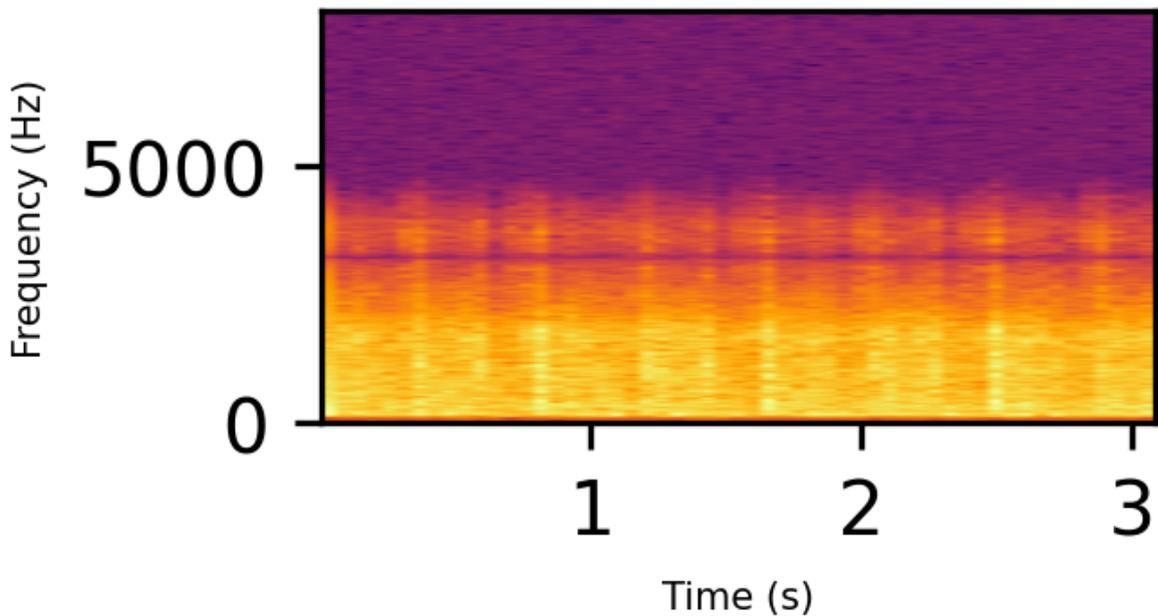
Le spectrogramme:



Prédiction: Défectueux! Il y a une faute de la ceinture pilote (Piston)

**Figure 3.18.** La prédiction et le spectrogramme d'un Audio (une faute "Piston").

Le spectrogramme:



Prédiction: Défectueux! Il y a une faute de courroie de cavalier (Rider belt)

**Figure 3.19.** La prédiction et le spectrogramme d'un Audio (une faute "Rider belt").

## ★ Commentaires

- Nous avons pu observer que notre modèle était capable de repérer de manière précise et fiable les anomalies présentes dans les données d'évaluation.
- Notre Dataset se compose de 8 états, comprenant un état "**Healthy**" et 7 états défectueux : Défaut de palier (**Bearing**), Défaut de volant moteur (**Flywheel**), Défaut de soupape d'admission de fuite (**LIV**), Défaut de soupape de sortie de fuite (**LOV**), Défaut de soupape de non-retour (**NRV**), Défaut de segment de piston (**Piston**) et Défaut de courroie de cavalier (**Rider belt**). Nous sommes heureux de vous annoncer que notre modèle a démontré une précision exceptionnelle en prédisant correctement chaque défaut à chaque fois.
- Nous avons tracé le spectrogramme avec chaque prédiction afin d'analyser la distribution des fréquences et les variations temporelles du signal, obtenant ainsi des informations précieuses pour des analyses et des traitements ultérieurs.
- Pour tracer le spectrogramme, nous avons utilisé une **colormap**, qui est une correspondance entre des valeurs numériques et des couleurs spécifiques utilisées dans la visualisation des données. Elle attribue des couleurs pour représenter différentes valeurs, permettant ainsi une représentation visuelle et une interprétation des motifs ou des variations des données.
- Dans notre cas, nous avons utilisé une colormap appelée "**Inferno**". Elle est conçue pour représenter des données continues avec une transition fluide des couleurs. Dans la colormap "**Inferno**", les valeurs plus basses sont représentées par des couleurs plus froides, telles que des violets profonds, tandis que les valeurs plus élevées sont représentées par des couleurs plus chaudes, comme des jaunes vifs et des oranges. Cette colormap est souvent utilisée pour mettre en évidence les variations d'intensité ou de magnitude dans les données, créant des visualisations à la fois esthétiques et informatives.
- Les valeurs plus basses peuvent correspondre à des amplitudes plus faibles ou à des composantes de fréquence moins intenses. Cela peut être interprété comme des parties plus calmes ou moins importantes du signal audio.

→ Les valeurs plus hautes peuvent correspondre à des amplitudes plus élevées ou à des composantes de fréquence plus intenses. Cela peut être interprété comme des parties plus fortes ou plus importantes du signal audio.

## **Conclusion**

En fin de ce chapitre on a vu comment l'entraînement du modèle a été fait, on a aussi obtenu des bons résultats concernant la détection d'anomalies.

Je suppose qu' avec un bon matériel et l'énorme développement de la technologie, on peut prendre ce thème vers un autre niveau.

## Conclusion Générale

Au cours de ce projet de recherche, notre objectif principal était de développer une méthode efficace de détection des anomalies de bruit des machines. Les anomalies dans le fonctionnement des machines peuvent entraîner des perturbations coûteuses, des temps d'arrêt imprévus et des réparations onéreuses. Il était donc essentiel de mettre au point une solution capable d'identifier rapidement et précisément ces anomalies, afin d'améliorer la maintenance prédictive et d'optimiser les opérations industrielles.

Nous avons identifié que les anomalies des machines constituaient une problématique majeure dans le domaine. Les méthodes traditionnelles de détection étaient souvent limitées par leur subjectivité, leur dépendance à l'expertise humaine et leur manque de précision. Afin de surmonter ces limitations, nous avons adopté une approche basée sur l'utilisation d'un modèle de réseaux de neurones convolutionnels (CNN) pour la prédiction des anomalies.

Notre modèle CNN a été développé en utilisant un ensemble de données représentatives comprenant des enregistrements de bruits de machines normaux et anormaux. Grâce à un processus du deep learning, le modèle a été capable de reconnaître les schémas caractéristiques des bruits de machines anormales et de faire des prédictions précises.

Les résultats de notre recherche ont été très encourageants. Notre modèle CNN a démontré une grande précision dans la détection des anomalies de bruit des machines, surpassant les méthodes traditionnelles. Les performances de prédiction ont été validées à travers une série de tests et d'évaluations, confirmant la fiabilité et l'efficacité de notre approche.

L'impact de notre travail est significatif pour l'industrie et la société en général. En détectant les anomalies de bruit des machines de manière précoce et précise, les entreprises peuvent réduire les temps d'arrêt, minimiser les coûts de réparation et améliorer la sécurité des opérations industrielles. Notre méthode ouvre la voie à une maintenance prédictive plus efficace et à une utilisation optimale des ressources.

Cependant, il est important de noter que notre recherche présente certaines limites. Notre modèle CNN a été entraîné sur un ensemble de données spécifique et son efficacité pourrait varier dans des contextes industriels différents. De plus, la détection des anomalies de bruit des

machines reste un domaine en constante évolution, avec de nouvelles techniques et approches qui émergent régulièrement.

Pour des recherches futures, il serait intéressant d'explorer d'autres architectures de modèles de deep learning, d'intégrer des techniques de traitement du signal avancées et de considérer des ensembles de données plus vastes et variés. De plus, des études supplémentaires pourraient être menées pour évaluer l'applicabilité de notre modèle dans différents environnements industriels et pour affiner les performances de prédiction.

En conclusion, notre projet de recherche a abouti au développement d'une méthode prometteuse pour la détection des anomalies de bruit des machines. Notre modèle CNN a démontré des performances remarquables et ouvre la voie à des améliorations significatives dans le domaine de la maintenance prédictive. L'adoption de cette approche innovante peut entraîner des économies substantielles pour les entreprises, tout en améliorant la fiabilité et l'efficacité de leurs opérations.

## Bibliographie

[1] Robert CAPLAIN, “Techniques de Prise de son”, Éditions Dunod, Juin 7, 2017. [Online].

Disponible:

<https://www.scribd.com/document/360072178/Techniques-de-Prise-de-Son>. [Consulté le 02 mai 2023].

[2] Antoine LORENZI, Benjamin CHAIX, “Représentation du son”, Voyage au centre de l'audition, Décembre 27, 2016. [Online]. Disponible:

<http://www.cochlea.eu/son/representation-du-son>. [Consulté le 04 mai 2023].

[3] “Short-Time Fourier Transforms”, université de Carnegie Mellon. [Online]. Disponible:

[https://course.ece.cmu.edu/~ece491/lectures/L25/STFT\\_Notes\\_ADSP.pdf](https://course.ece.cmu.edu/~ece491/lectures/L25/STFT_Notes_ADSP.pdf). [Consulté le 05 mai 2023].

[4] Kurzekar, P. K., Deshmukh, R. R., Waghmare, V. B., & Shrishrimal, “A Comparative Study of Feature Extraction Techniques for Speech Recognition System”, International Journal of Innovative Research in Science, Décembre 12, 2014. [Online]. Disponible:

[https://www.researchgate.net/publication/270898600\\_A\\_Comparative\\_Study\\_of\\_Feature\\_Extraction\\_Techniques\\_for\\_Speech\\_Recognition\\_System](https://www.researchgate.net/publication/270898600_A_Comparative_Study_of_Feature_Extraction_Techniques_for_Speech_Recognition_System). [Consulté le 06 mai 2023].

[5] Imad Rida, “feature extraction for temporal signal recognition:an overview”, Université de Normandie, Décembre 06, 2018. [Online]. Disponible:

<https://arxiv.org/pdf/1812.01780.pdf>. [Consulté le 08 mai 2023].

[6] Upasana, “Introduction to Classification Algorithms”, EDURKA!, Novembre 25, 2020. [Online].Disponible:

<https://www.edureka.co/blog/classification-algorithms/#classificationalgorithmsvsclusteringalgorithms>. [Consulté le 09 mai 2023].

[7] “Cross-Validation ou Validation croisée : définition et importance”, DataScientest, Mai 03, 2023. . [Online]. Disponible: <https://datascientest.com/cross-validation>. [Consulté le 10 mai 2023].

[8] “Matrice de confusion : qu’est-ce que c’est et comment l’utiliser ?”, DataScientest, Février 16, 2023. [Online]. Disponible: <https://datascientest.com/matrice-de-confusion>. [Consulté le 10 mai 2023].

[9] Sumeet Kumar Agrawal, “Metrics to Evaluate your Classification Model to take the right decisions”, Analytics Vidhya, Mai 17, 2023. [Online]. Disponible: <https://www.analyticsvidhya.com/blog/2021/07/metrics-to-evaluate-your-classification-model-to-take-the-right-decisions/>. [Consulté le 20 mai 2023].

[10] Guillaume Saint-Cirgue, “Comment fonctionne le Machine Learning ?”, MACHINE LEARNIA, Mars 11, 2019. [Online]. Disponible: <https://machinelearnia.com/comment-fonctionne-machine-learning/>. [Consulté le 12 mai 2023].

[11] Chandrahas Mishra, “Deep Machine Learning and Neural Networks: An Overview”, IJ-AI, Mai 23, 2017. [Online]. Disponible: [https://www.researchgate.net/publication/332578851\\_Deep\\_Machine\\_Learning\\_and\\_Neural\\_Networks\\_An\\_Overview](https://www.researchgate.net/publication/332578851_Deep_Machine_Learning_and_Neural_Networks_An_Overview). [Consulté le 13 mai 2023].

[12] Jürgen Schmidhuber, “Deep learning in neural networks: An overview”, Elsevier, Octobre 13, 2014. [Online]. Disponible: <https://www2.econ.iastate.edu/tesfatsi/DeepLearningInNeuralNetworksOverview.JSchmidhuber2015.pdf>. [Consulté le 14 mai 2023].

[13] Great Learning Team, “Types of Neural Networks and Definition of Neural Network”, Great Learning, Novembre 23, 2022. [Online]. Disponible: <https://www.mygreatlearning.com/blog/types-of-neural-networks/>. [Consulté le 15 mai 2023].

[14] Siddharth Sharma, Simone Sharma, “ACTIVATION FUNCTIONS IN NEURAL NETWORKS”, IJEAST, Avril, 2020. [Online]. Disponible: <https://www.ijeast.com/papers/310-316,Tesma412,IJEAST.pdf>. [Consulté le 17 mai 2023].

[15] Matthias Feurer, Frank Hutter, “Automated Machine Learning Methods, Systems, Challenges”, The Springer Series, Octobre, 2018. [Online]. Disponible: [https://www.researchgate.net/publication/333179372\\_Hyperparameter\\_Optimization](https://www.researchgate.net/publication/333179372_Hyperparameter_Optimization). [Consulté le 19 mai 2023].

[16] Dor Bank, Noam Koenigstein, Raja Giryes, “Autoencoders”, ResearchGate, Mars, 2020. [Online]. Disponible:

[https://www.researchgate.net/publication/339945889\\_Autoencoders](https://www.researchgate.net/publication/339945889_Autoencoders). [Consulté le 19 mai 2023].

[17] Jonas Teuwen, Nikita Moriakov, “Convolutional neural networks”, Université de Huston, 2020. [Online]. Disponible:

[https://www2.cs.uh.edu/~ceick/ai/CNN\\_Reading\\_Material.pdf](https://www2.cs.uh.edu/~ceick/ai/CNN_Reading_Material.pdf). [Consulté le 21 mai 2023].

[18] Vinothraj Thangarajah, “Python current trend applications-an overview”, ResearchGate, Octobre, 2019. [Online]. Disponible:

[https://www.researchgate.net/publication/344569950\\_Python\\_current\\_trend\\_applications-an\\_overview](https://www.researchgate.net/publication/344569950_Python_current_trend_applications-an_overview). [Consulté le 24 mai 2023].