Université de Yaoundé I

Département d’informatique – INFO L3

RAPPORT DU PROJET DE INF3721 RECONNAISSANCE DES CHIFFRES DE 0 À 9

Membres du groupe 2 :

|  |  |
| --- | --- |
| **NOMS ET PRENOMS** | **MATRICULE** |
| **MAKOU FOKOUA PATRICIA** | **21T2446** |
| **MEKA MOÏSE CHRISTIAN JUNIOR** | **21T2561** |
| **TCHEMMOE SOREL STEPHANIE** | **21T2626** |
| **RUDY TCHAMBA TCHAPTCHE ITIEL** | **21T2981** |
| **ATEUGONG TAKAN STIVINSTON** | **21T2491** |
| **VOUKENG TEDONKEMWA ERDI DÉSIRE** | **21T2427** |
| **FOSSI TALLA JORDANE** | **20U2878** |

Sous la supervision de : Dr MELATAGIA

Table des matières

[INTRODUCTION 2](#_Toc171011758)

[I. PROCESSUS DE DÉVELOPPEMENT DU MODELE 3](#_Toc171011759)

[COLLECTE DES DONNÉES 3](#_Toc171011760)

[PRÉTRAITEMENT DES DONNÉES 3](#_Toc171011761)

[DIVISION DES DONNEES EN TRAIN-VALID-TEST 4](#_Toc171011762)

[II. DEPLOIEMENT DU MODELE 6](#_Toc171011763)

[CONCLUSION 7](#_Toc171011764)

[ANNEXE 8](#_Toc171011765)

[CHOIX TECHNOLOGIQUES 8](#_Toc171011766)

# **INTRODUCTION**

Le cours de classification automatique supervisée et non supervisée s’est achevé avec la réalisation de projets d’apprentissage automatique par groupe de sept (7) à réaliser sur une période allant du 21 mai au 8 juillet 2024. Dans le cadre de ces projets, notre groupe a eu la charge de mettre en place et déployer un modèle de classification des audios de chiffres (0 à 9) en utilisant le Mel Frequency Cepstral Coefficients (MFCC) et le perceptron multicouche (MLP).

Le développement de ce modèle de classification s’est fait en suivant une approche itérative partant de la collecte de données jusqu’à l’évaluation du modèle en passant par le traitement des données, l’entrainement et la validation du modèle.

Le présent rapport décrit le processus de réalisation de ce projet, en montrant comment s’est déroulé chaque étape, en détaillant les choix technologiques faits et ainsi que les résultats obtenus lors des tests.

# **PROCESSUS DE DÉVELOPPEMENT DU MODELE**

## **COLLECTE DES DONNÉES**

Pour constituer notre ensemble de données, chaque membre du groupe a enregistré 10 étudiants distincts. Chaque étudiant a prononcé les chiffres de 0 à 9 à trois reprises. Ainsi, nous avons obtenu un dataset comprenant 2100 enregistrements audio, au format .wav, annotés lors de l’enregistrement. En effet, chaque nom de fichier audio enregistré contient le chiffre correspondant.

## **PRÉTRAITEMENT DES DONNÉES**

Sachant que les performances du modèle dépendent en grande partie de la qualité des données sur lesquelles est basée son apprentissage, nous avions effectué des prétraitements sur les audios pour extraire les impuretés.

Pour la réalisation de cette tâche, nous avons utilisé les bibliothèques librosa et noisereduce. Voici en détail les étapes que nous avons suivies pour la préparation de nos données :

* **Chargement des Audios :**

Nous avons utilisé la bibliothèque ***librosa*** pour charger chaque fichier audio au format .wav dans notre environnement de travail. À chaque enregistrement nous avons récupéré l’étiquette du chiffre correspondant.

* **Réduction des Bruits de Fond :**

Ayant collecté les enregistrements audios à l’université, ils contenaient des bruits indésirables (sons ambiants). Nous avons utilisé la bibliothèque ***noisereduce*** pour réduire ces bruits et améliorer la qualité des signaux audio.

* **Suppression des Silences :**

Les débuts et fins d’enregistrements peuvent contenir des moments de silence. Avec ***librosa***, nous avons détecté et supprimé ces parties inutiles et ne conserver que le segment pertinent.

* **Normalisation des Audios :**

La normalisation permet d’ajuster l’amplitude des signaux audio pour les rendre comparables. Avec ***librosa***, nous avons normalisé chaque enregistrement afin d’éviter des variations excessives.

* **Extraction des MFCC :**

Les MFCC sont des caractéristiques fréquentielles extraites à partir des enregistrements audios. Nous avons utilisé la bibliothèque ***librosa*** pour calculer les MFCC de chaque audio avec l’hyperparamètre **n\_mfcc=40**.

Les matrices des MFCC résultantes n’ayant pas les mêmes dimensions, pour un souci d’uniformisation de la dimension des entrés du modèle, la moyenne suivant les colonnes de chaque matrice MFCC a été calculé. Obtenant ainsi une liste des vecteurs de spectres représentant les propriétés acoustiques.

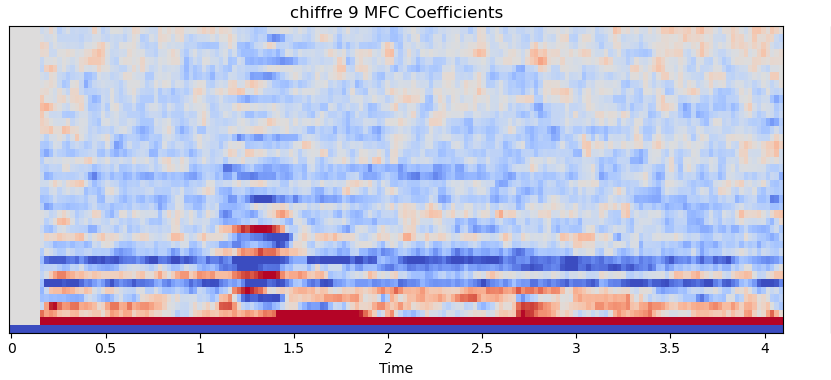


Figure 1 : Spectre résultant de l’extraction des MFCC d’un audio

En somme, ce prétraitement a permis d’obtenir des données propres et adaptées à l’entraînement de notre modèle de reconnaissance des chiffres.

* **Encodage des étiquettes (classes) :**

Pour réaliser l’encodage des étiquettes de chaque audio, nous avons utilisé le **one hot encoding**.

## **DIVISION DES DONNEES EN TRAIN-VALID-TEST**

Notre ensemble de données constitué des 2100 audios de chiffres a été séparé en ensemble d’entrainement, de Validation et de test suivant les proportions 64%, 16% et 20% respectivement.

**CONSTRUCTION DU MODÈLE**

* **Choix de l’Architecture :**

L’architecture du MLP choisi est la suivante :

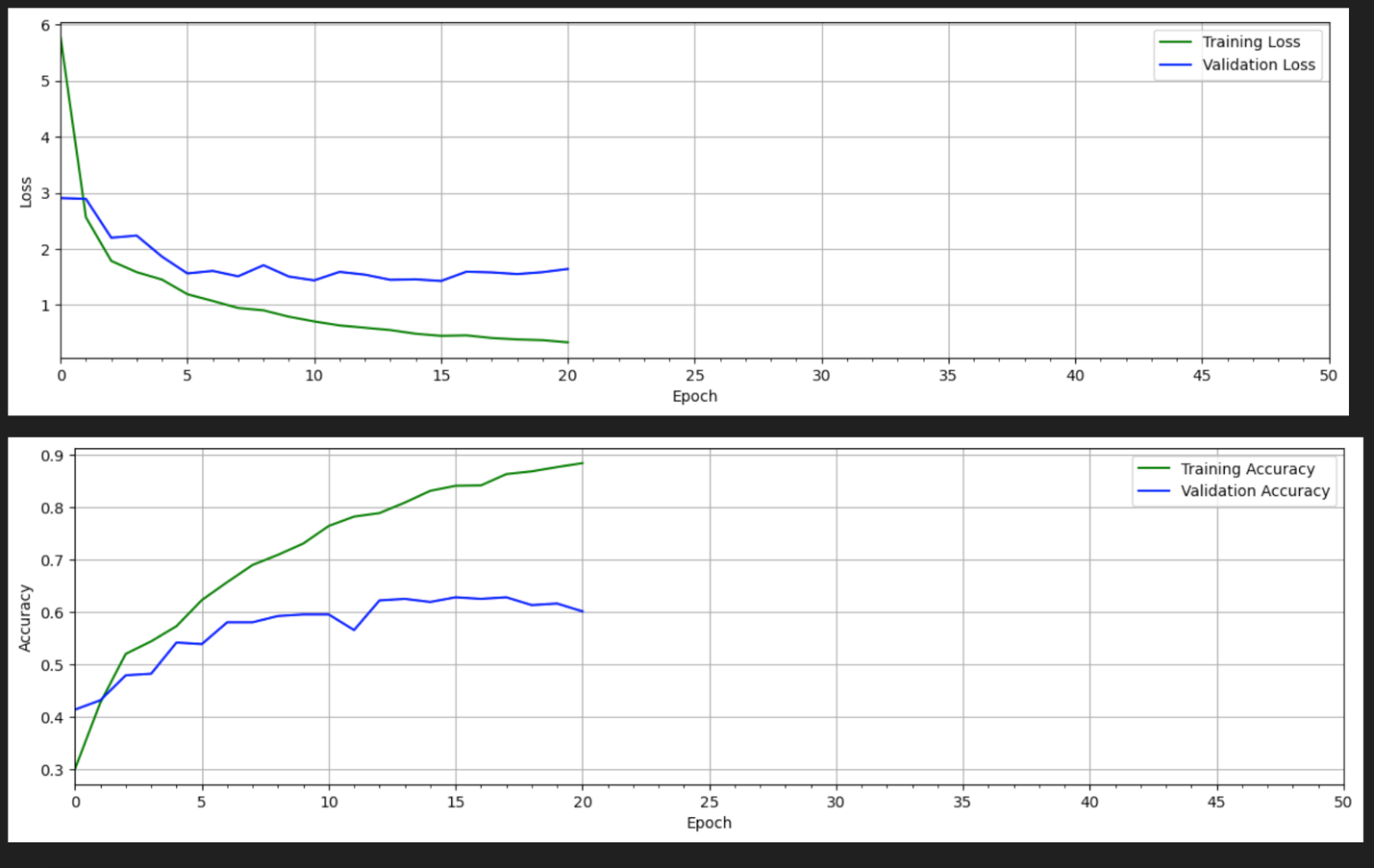
* Une couche d’entré de 128 neurones avec pour activation **ReLU ;**
* Une couche cachée de 128 neurones avec pour activation **ReLU ;**
* Une Troisième couche dense de 10 neurones qui représentera notre couche de sortie avec pour activation **Softmax**.
* **Fonction de Coût et Optimisation :**

Nous avons utilisé la fonction de coût **categorical\_crossentropy** car nos sorties sont des catégories à plus de deux classes et comme nous avions encoder nos données plus haut avec one hot nous ne pouvons donc pas utiliser le ***sparse\_categorical\_crossentropy*** car il est adapté aux catégories encodées par des chiffres (0, 1, 2, ...) ou ***binary\_crossentropy*** qui est adapté à la classification binaire.

La méthode d’optimisation utilisée ici était ***Adam*** *avec un taux d’apprentissage de 0,002*.

* **Entraînement et Validation :**

Le modèle a été entraîné sur l’ensemble d’entraînement et validé sur l’ensemble de validation, avec 80 époques et une taille de batch de 64. Pour éviter le surapprentissage lors de cet entrainement, on utilise un “***Early* *Stopping***” avec une attende de 5 épochs. Le figure 2 ci-dessous nous présente l’évolution de cet apprentissage.



**Figure 2 :** Évolution suivant les épochs de fonction de perte et du pourcentage de réussite des données du trainset et du validset lors l’entrainement du modèle

* **Évaluation du Modèle :**

Nous avons utilisé comme métriques l’**accuracy,** de **précision,** de **rappel** et de **f1score** pour évaluer la performance du modèle sur les données de test. Les graphiques ci-dessous nous présente les résultats de cette évaluation.



**Figure 3 :** Performance du modèle lors de la phase d’évaluation avec le testset

En somme, ce processus nous a permis de construire un modèle capable de reconnaître les chiffres prononcés dans des enregistrements audio. Néanmoins, les performances du modèle ne sont pas bonnes.

# **DEPLOIEMENT DU MODELE**

Le modèle précédemment développé a été déployé par le biais d’une api en utilisant la bibliothèque FastAPI.

En effet, nous avons créé une API avec deux (2) route :

* **docs/**: Pour la documentation sur l’API ;
* **model\_call/** Accessible avec la méthode post qui prend en entré le fichier audio du chiffre en format .wav, traite le fichier comme décrit plus haut, extrait le MFCC, fait passer le résultat dans le modèle et retourne le chiffre correspondant.

# **CONCLUSION**

Arrivé au terme de la réalisation de ce projet, il ressort que le modèle de classification MLP effectue de bonnes prédictions environ une fois sur deux et les chiffres obtenant une meilleure généralisation par le modèle sont 3, 6, 7 et 0. Durant cet exercice, nous nous sommes confrontés à de nombreux obstacle dont le principal était la taille du dataset. En effet lors de notre revu des précédents travails réalisés avec la classification des audios, la taille des ensembles de données était considérablement très grande par rapport à la nôtre (Ce qui pourrait être à l’origine de la faible performance du modèle). Aussi le MFCC retournant une matrice, résumer cette dernière en un vecteur pourrait causer une perte d’information. Pour résoudre ce problème, il est généralement conseillé des architectures plus complexes tels que les Réseaux de Neurone Récurrents (RNN).

# ANNEXE

## CHOIX TECHNOLOGIQUES

Lors de la conception de notre modèle, nous avons pris des décisions importantes concernant les outils et bibliothèques à utiliser. Voici les choix technologiques que nous avons faits :

1. **Google Colab (Colaboratory)**

**Rôle** : Google Colab est un environnement de notebook Jupyter basé sur le cloud qui permet d’exécuter du code Python.

**Utilisation** : Nous avons utilisé Google Colab pour développer et expérimenter notre modèle. Il offre une puissance de calcul gratuite et facilite le partage de notebooks.

1. **Librosa**

**Rôle** : Librosa est une bibliothèque Python spécialement conçue pour l’analyse des signaux audio.

**Utilisation** : Nous avons utilisé Librosa pour charger les fichiers audios, extraire les MFCC et effectuer des opérations de prétraitement.

1. **Noicereduce**

**Rôle** : Noisereduce est une bibliothèque qui permet de réduire les bruits de fond dans les enregistrements audio.

**Utilisation** : Nous avons appliqué Noisereduce pour améliorer la qualité de nos signaux audio en supprimant les interférences indésirables.

1. **IPython**

**Rôle** : IPython est un environnement interactif pour le développement et l’expérimentation.

**Utilisation** : Nous l’avons utilisé pour explorer nos données et écouter nos enregistrements.

1. **Audiomentations**

**Rôle** : Audiomentations est une bibliothèque Python dédiée à l’augmentation de données audio.

**Utilisation** : Nous avons appliqué des transformations audios telles que le changement de vitesse, l’ajout de bruit et la modification du pitch pour augmenter notre ensemble de données.

1. **TensorFlow**

**Rôle** : TensorFlow est une bibliothèque open-source pour l’apprentissage automatique et l’intelligence artificielle.

**Utilisation** : Nous avons construit notre modèle de reconnaissance des chiffres en utilisant TensorFlow pour définir l’architecture du MLP, l’entraîner et l’évaluer.

1. **Keras**

**Rôle** : Keras est une interface haut niveau pour TensorFlow, simplifiant la création et la formation de réseaux de neurones.

**Utilisation** : Nous avons utilisé Keras pour construire notre architecture MLP, ajouter des couches, définir les fonctions d’activation et compiler notre modèle.

En combinant ces outils, nous avons pu développer un modèle pour la reconnaissance des chiffres audio.

1. **FastAPI**

**Rôle** : FastAPI est une bibliothèque python largement utilisée pour développer aisément des API dans de nombreux domaines notamment pour le déploiement des modèles d’apprentissage artificiel.

**Utilisation** : Nous avons utilisé FastAPI pour construire une API pour déployer notre modèle.