Ecole Publique d'Ingénieurs en 3 ans

Rapport

# PROJET 2A
# RECONNAISSANCE CLAVIER

Le 02 avril 2024

**Cédric Willaume**
**Mohamed-Abderrahmane Bedda**
**Abdelmalek Belghomari**
**Haykel Sriha**
**Winnie Kamtchueng**

Encadrants : **Christophe Rosenberger**
**Tanguy Gernot**


www.ensicaen.fr

# TABLE DES MATIERES

# TABLE DES FIGURES

# 1. Introduction

Dans le monde numérique actuel, la sécurité des informations est d'une importance cruciale. Les avancées technologiques ont permis une utilisation généralisée des dispositifs électroniques, des smartphones aux ordinateurs portables et cela en passant par les claviers et les pavés tactiles. Cependant, cette prolifération de technologies n'est pas sans risque. Les données sensibles peuvent être compromises à tout moment, que ce soit par des pirates informatiques ou des méthodes plus subtiles et méconnues. Car toute donnée est une information, il suffit de savoir l'exploiter.

# 2. Présentation du projet

## 2.1. Contexte

Dans le cadre de nos études en deuxième année en informatique à l'ENSICAEN, il nous a été confié par Christophe Rosenberger et Tanguy Gernot, un projet touchant au domaine de la FORENSIC sur l'analyse des émissions sonores des claviers. Le contexte de notre projet est profondément ancré dans les défis contemporains de la cybersécurité et de la lutte contre la cybercriminalité.

Notre projet est un support non négligeable pour des services de sécurité par exemple pour protéger l'accès à des données sensibles. Il peut aussi permettre de surveiller l'activités des utilisateurs d'un poste de travail. Il est un biais utile à l'authentification et à la prévention des attaques biométriques.

## 2.2. Objectifs

Le projet se divise en deux axes principaux : l'identification de l'utilisateur et l'identification des touches tapées à partir de l'analyse des émissions sonores du clavier. Par la suite, nous envisageons de créer un site web permettant à l'utilisateur de saisir un texte et de recevoir les résultats des deux algorithmes selon sa préférence.

# 3. Etat de l'art

A l'amorce de notre projet, nous avons commencé par nous documenter sur des études précédemment réalisées, et nous avons ainsi réalisé un état de l'art.

En passant en revue les travaux existants, nous avons pu nous renseigner sur les pistes explorées par d'autres chercheurs. Cette exploration nous a fourni une base solide, permettant de nous guider dans nos propres recherches.

## 3.1. Méthodologies utilisées dans des travaux antérieurs

L'examen approfondi des études précédentes nous a permis de découvrir une diversité de méthodologies employées pour la reconnaissance des frappes de clavier. Parmi celles-ci, il y avait des méthodes de Deep Learning avec des réseaux de neurones profonds (CoAtNet), permettant d'extraire et d'analyser les caractéristiques des frappes de clavier à partir de données audios.

De plus, des techniques de traitement de signal ont été largement utilisées pour analyser les motifs acoustiques des frappes de clavier, notamment en utilisant des transformées de Fourier ou des coefficients cepstraux de fréquence de Mel appelés « MFCC » (Mel Frequency Cepstral Coefficients). Ces techniques ont permis de caractériser et analyser efficacement les signaux des frappes de clavier, facilitant ainsi leur classification et leur identification. Avec cela, plusieurs études ont utilisé des modèles de Machine Learning traditionnels pour reconnaître les frappes de clavier à partir des caractéristiques précédentes.

L'état de l'art, nous a donc amené à construire un tableau (figure 1), classifiant chacune des approches étudiées. Ce tableau offre une vue d'ensemble sur la méthode utilisée, et son efficacité dans la reconnaissance des frappes de clavier.

| Paper | Year | Principle | Accuracy(percentage) |
|---|---|---|---|
| A Practical Deep Learning-Based Acoustic Side Channel attack on keyboards | 2023 | CoAtNet | 93% |
| Analyse de la dynamique de frappe au clavier sonore pour l'identification, le profilage et l'extraction du texte saisi | 2022 | SVM/MFCC | 96% |
| Don't skype & type | 2017 | MFCC | 91% |
| Don't skype & type | 2017 | LF | 100% |
| Reconnaissance de saisie sur clavier par analyse acoustique | 2011 | Intercorrelation/DFT | 99% |
| Keyboard Acoustic Emanations Revisited | 2009 | MFCC/HMM | 87% without any noise |
| Keyboard Acoustic Emanations: An Evaluation of strong passwords and typing styles | Unknown | Tim-Frq | 82.69% |

*Figure 1 Tableau de comparaison de différents modèles de reconnaissance sonore du clavier*

### 3.2. Première approche méthodologique choisie

Après rédaction de notre état de l'art scientifique, nous avons naïvement opté pour une approche combinant l'utilisation des coefficients MFCC ainsi qu'un modèle de classification, le réseau de neurones. Notre approche vise ainsi à fournir une solution robuste et simple pour la reconnaissance des frappes de clavier à partir de données audios. Cette approche aura été remise en question par la suite afin de ne pas se baser que sur un choix naïf.

## 4. Gestion de projet

### 4.1. Organisation du travail

L'organisation de notre projet a été pensée à la suite du premier rendez-vous avec notre tuteur et client. Nous avons dressé un diagramme de GANTT de novembre à avril, reprenant tous les grands points de notre projet. Cela nous a permis de délimiter les tâches du projet, comme le développement de la démonstration web, la création de la base de données d'audios, l'entrainement des différents modèles ainsi que les différentes approches et méthodes que nous pourrions utiliser. Nous avons donc décidé d'une organisation par rôles comme celui de développeurs, d'architecte et de chef de projet afin de partager le travail et être plus efficace sur les différents aspects du projet. Nous nous organisions à l'aide de nos canaux de discussions et nous partagions notre code à chaque changement. Au cours de l'année, nos différents sprints ont été pensé pour être efficace au maximum afin de fournir agilité et robustesse à chaque étape.

### 4.2. Outils utilisés

Dans le cadre de notre projet, nous avons eu recours à plusieurs outils ; pour la communication et le partage de documents, nous avons utilisé les plateformes de messagerie instantanée telles que WhatsApp et Teams.

Nos tuteurs (et clients) ayant fournis un canal de discussion déjà prêt sur Discord, c'est ce que nous avons privilégié pour échanger avec eux, pour une collaboration efficace en tout temps.

Enfin, pour le partage et la gestion du code source, nous avons opté pour l'utilisation des serveurs GitLab de l'ENSICAEN. Cette plateforme de développement collaboratif basée sur Git, offre les fonctionnalités avancées de contrôle de version et de gestion de projet dont nous avions besoin. Utiliser la version de l'ENSICAEN nous permettait d'être sûr que tous les membres de l'équipe ainsi que nos tuteurs avaient déjà un compte sur la plateforme.

# 5. Conception

Après avoir étudié le sujet et défini les contours du projet, nous entamons la phase de conception proprement dite. Dans un premier temps, nous aborderons la création de la base de données, puis nous détaillerons le travail effectué sur la chaîne opérationnelle, pour enfin nous pencher sur la réalisation de la démonstration web.

## 5.1. Base de données

Pour qu'une base de données destinée à l'entraînement d'un modèle soit considérée comme valide, elle doit être suffisamment riche et équilibrée, respectant un nombre identique d'occurrences pour chaque touche (ici, label dans le vocabulaire de l'apprentissage). Après avoir tenté, sans succès, de trouver des enregistrements de frappes de touches sur internet, nous avons pris l'initiative de construire notre propre base de données. Trois membres de notre équipe ont contribué à la base de données pour la reconnaissance de touches, tandis que quatre membres du groupe ont participé à celle dédiée à la reconnaissance de personnes. Afin de répondre aux critères énoncés, chaque participant a dû taper chaque lettre de l'alphabet ainsi que la touche espace de manière répétée mais lente, en raison des limitations de notre système de segmentation pour le modèle de reconnaissance de lettres. Pour le modèle de reconnaissance de personnes, chaque participant a saisi une liste de 30 phrases de manière spontanée, afin que nous puissions capter leur comportement typique à l'égard de l'utilisation du clavier.

## 5.2. Chaîne opérationnelle

Maintenant que nous avons une base de données établie qui servira à l'entraînement du modèle, nous pouvons entamer le processus de traitement des fichiers audios pour tenter d'obtenir une prédiction de l'utilisateur et de sa saisie. Ce processus implique plusieurs étapes intermédiaires, allant de la segmentation des données au choix du modèle adapté, avant d'arriver à l'amélioration de la prédiction. Nous détaillerons donc chaque étape de manière progressive, en mettant en lumière les étapes intermédiaires cruciales. Nous ne distinguerons pas vraiment la prédiction de la saisie de la prédiction de l'utilisateur, car les deux chaînes opérationnelles se ressemblent énormément ; la différence étant que pour la reconnaissance utilisateur, il n'est pas nécessaire de segmenter les audios.

### 5.2.1. Segmentation des audios

La segmentation des audios a pour but d'isoler chaque frappe de touche, permettant de pouvoir mieux l'analyser, et ainsi la prédire. L'approche que nous avons adoptée pour la

segmentation, s'appuie sur une analyse approfondie du signal audio. Cette méthode se divise en plusieurs étapes clés :

- Préparation du Signal Audio

La première étape consiste à lire le signal audio à partir d'un fichier audio de format .wav, afin de récupérer les données d'amplitude du signal ainsi que le taux d'échantillonnage. Ensuite, nous procédons à un pré-traitement visant à éliminer certaines parties des audios, souvent vides ou bruitées en fin de signal.

- Définition du Seuil Dynamique

Afin d'identifier avec précision les moments de frappe des touches, nous mettons en place un processus de seuillage adaptatif qui correspond à une amplitude au-delà de laquelle un son est considéré comme un bruit d'une touche. Ce seuil est déterminé à partir de la moyenne des amplitudes qui excèdent un seuil bas prédéfini, ajustée par des coefficients fixés afin d'optimiser la sensibilité de la détection selon les particularités de l'audio.

- Identification et Sélection des Pics

Nous utilisons ensuite la fonction find_peaks de la bibliothèque Python scipy.signal, configurée avec le seuil dynamique établi et une distance minimale fixe entre les pics. Cela évite de détecter plusieurs pics pour une seule frappe de touche ; car lorsque l'on tape sur une touche, deux pics d'intensité sont générés, l'un correspondant à l'appui sur la touche, et l'autre à son relâchement. Nous ne voulons pas que ces deux pics soient considérés comme n'appartenant pas à la même touche. Cette approche garantit que chaque pic détecté correspond fidèlement à une frappe de touche dans l'enregistrement.

- Segmentation et Sauvegarde

Après l'identification des pics, le signal audio est segmenté de façon à ce que chaque segment représente une frappe de touche. Ces segments sont extraits en se basant sur les positions des pics détectés, de manière à centrer les pics au sein des segments. Ceci permet de prendre en compte à la fois les moments d'appui et de relâchement des touches.

### 5.2.2. MFCC

Après la segmentation, nous adoptons l'approche d'extraction des coefficients cepstraux de fréquence de Mel (MFCC) pour chaque segment isolé. Les MFCC sont choisis pour leur capacité éprouvée à capturer de manière efficace les caractéristiques spectrales essentielles des sons, ce qui en fait une ressource utile dans le domaine de la reconnaissance vocale et la classification des sons. En utilisant la fonction feature.mfcc() de la bibliothèque librosa, un outil puissant pour

l'analyse audio sous Python, nous avons calculé les MFCC de chaque segment correspondant à une frappe de touche, puis moyenné ces coefficients pour obtenir un vecteur de caractéristiques unique par touche. Les vecteurs résultants sont par la suite enregistrés dans un fichier CSV, servant comme entrée de notre modèle de reconnaissance.

Le choix du nombre de coefficients MFCC par fichier audio joue un rôle crucial dans la performance d'un modèle de machine learning. Un nombre excessif de MFCC peut ralentir le modèle, tandis qu'un nombre insuffisant peut entraîner un manque d'informations pertinentes pour l'analyse. Dans le cadre de notre projet de reconnaissance de touches, nous avons opté pour une approche initialement arbitraire en choisissant 20 MFCC, une décision basée sur les résultats satisfaisants obtenus lors de l'évaluation des audios testés.

Cependant, pour la reconnaissance de personnes, où les fichiers audios peuvent varier en longueur, nous avons adopté une méthode plus systématique. Cette approche consiste à segmenter l'audio en plusieurs parties selon leur taille et à extraire un nombre fixe de MFCC de chaque segment. Nous avons développé un script pour ajuster dynamiquement le nombre de segments, et de MFCC par segment, puis évalué la performance du modèle pour chaque configuration. Les résultats de ces évaluations ont été compilés et présentés sous forme de tableau.

| Nombre de MFCCs | 5 | 10 | 15 | 20 | 25 | 30 | 35 | 40 | 45 | 50 |
|---|---|---|---|---|---|---|---|---|---|---|
| Nombre de Segments | | | | | | | | | | |
| 5 | 0.750000 | 0.750000 | 0.750000 | 0.833333 | 0.833333 | 0.833333 | 0.833333 | 0.916667 | 0.916667 | 0.833333 |
| 10 | 0.833333 | 0.833333 | 0.916667 | 0.833333 | 0.916667 | 0.833333 | 0.833333 | 0.916667 | 0.916667 | 0.916667 |
| 15 | 0.916667 | 0.916667 | 0.916667 | 0.916667 | 0.916667 | 0.916667 | 0.916667 | 0.916667 | 0.916667 | 0.916667 |
| 20 | 1.000000 | 1.000000 | 1.000000 | 1.000000 | 1.000000 | 1.000000 | 1.000000 | 1.000000 | 1.000000 | 0.916667 |
| 25 | 1.000000 | 1.000000 | 0.916667 | 0.916667 | 0.916667 | 0.916667 | 1.000000 | 0.916667 | 0.833333 | 1.000000 |
| 30 | 0.916667 | 0.916667 | 0.916667 | 0.916667 | 0.916667 | 1.000000 | 0.916667 | 0.916667 | 0.916667 | 0.916667 |
| 35 | 0.916667 | 0.916667 | 0.916667 | 0.916667 | 0.916667 | 0.916667 | 0.916667 | 0.916667 | 0.833333 | 0.916667 |
| 40 | 0.916667 | 0.916667 | 0.916667 | 0.916667 | 0.916667 | 0.916667 | 0.916667 | 0.916667 | 0.916667 | 0.916667 |
| 45 | 0.916667 | 0.916667 | 0.916667 | 0.916667 | 0.916667 | 0.916667 | 0.916667 | 0.916667 | 0.916667 | 0.916667 |
| 50 | 0.916667 | 0.916667 | 0.916667 | 0.916667 | 0.916667 | 0.916667 | 0.916667 | 0.916667 | 0.916667 | 0.916667 |

*Figure 2 : Tableau de la performance du modèle en fonction du nombre de MFCCs et du nombre de segments*

L'analyse des résultats a confirmé notre hypothèse : avec un faible nombre de MFCC et de segments, la performance du modèle diminue. Inversement, augmenter excessivement ces paramètres conduit également à une baisse de performance. Finalement, nous avons identifié que la configuration optimale pour notre modèle de reconnaissance de personnes était de 20 segments et 20 MFCC par segment, totalisant 400 MFCC. Cette configuration a permis d'atteindre une performance d'évaluation parfaite de 1.0, selon notre tableau d'évaluation.

Cette démarche méthodique souligne l'importance d'un choix réfléchi des paramètres dans le développement de modèles de reconnaissance audio efficaces. Elle démontre également

l'avantage de l'expérimentation et de l'analyse empirique dans l'optimisation des performances de tels modèles.

### 5.2.3. Choix du modèle

Après avoir caractérisé les audios grâce aux MFCC, la prochaine étape consiste à chercher un modèle de classification approprié pour classer les audios. Pour cela, nous avons utilisé l'application Orange, qui permet de tester efficacement différents modèles de classification, tout en laissant la liberté de changer les paramètres rapidement.

Dans la figure 3, on peut voir le workflow utilisé permettant de chercher le meilleur modèle de classification d'émission sonore du clavier. Parmi les différents modèles testés, il y a le k-Nearest Neighbors (kNN), l'adaBoost, le decision Tree, le support Vector Machine (SVM), ainsi que le Neural Network (le réseau de neurones).

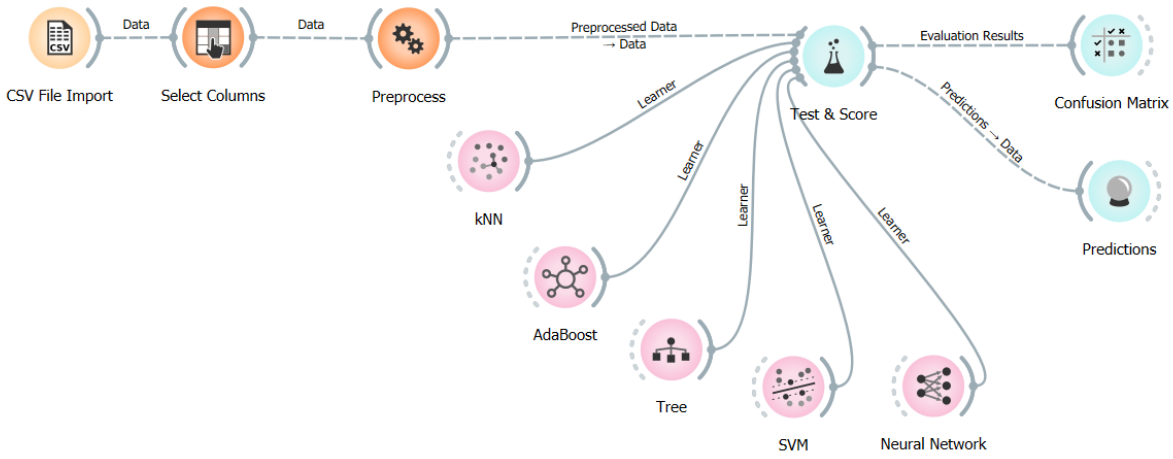


*Figure 3 Workflow Orange permettant de tester différents modèles de classification*

En entrée du workflow, nous importons le fichier csv rempli des MFCC calculés précédemment. Chaque ligne du fichier csv est mise en correspondance avec un label, qui est la lettre correspondant à ces MFCC, ce qui permet l'entraînement du modèle, ainsi que de l'évaluation de ses performances.

Par tâtonnements, on modifie les paramètres des modèles de classification, et on change les méthodes de pré-traitement, afin que la précision augmente. Et finalement, on trouve que le meilleur modèle est le réseau de neurones avec une précision de 73.6%. Les paramètres utilisés sont ceux de la figure 4. En pré-traitement une méthode de normalisation des MFCC est utilisée. Pour le réseau de neurones, 500 neurones dans les couches cachées sont utilisés, ainsi qu'une fonction d'activation ReLu, un solveur Adam, un coefficient de régularisation de 0.0001, et un nombre maximal d'itérations de 200.

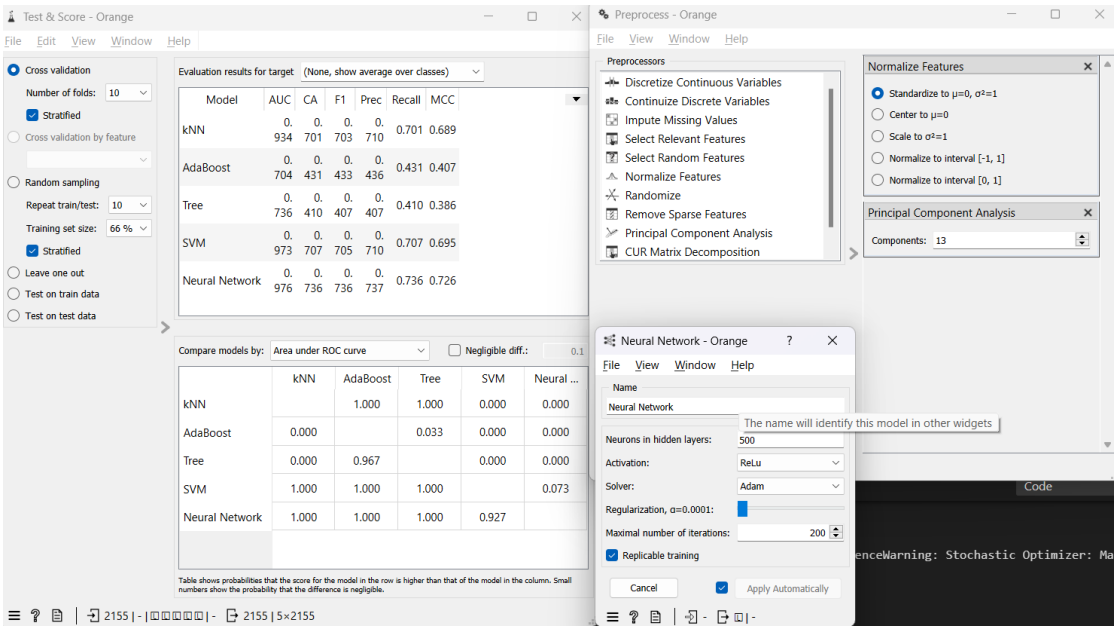


*Figure 4 Paramètres pour le réseau de neurones, et performances des modèles*

### 5.2.4. Amélioration de la prédiction

Pour améliorer les performances du modèle de reconnaissance de touches, notamment dans les cas où l'utilisateur saisit des phrases complètes plutôt que de simples mots de passe, nous avons envisagé l'intégration d'une fonctionnalité supplémentaire à l'exécutable. Cette fonction permet d'appliquer la distance de Levenshtein, en combinaison avec un dictionnaire, pour affiner les prédictions du modèle.

La distance de Levenshtein, également appelée « distance d'édition », est une mesure de la différence entre deux chaînes de caractères, correspondant au nombre minimum d'opérations nécessaires pour transformer l'une en l'autre. Ces opérations comprennent l'insertion, la suppression ou le remplacement d'un caractère. En utilisant cette distance, nous pouvons identifier les mots les plus similaires à ceux prédits par notre modèle, ce qui permet d'augmenter la précision globale de la reconnaissance.

Afin de corriger les mots en anglais, nous avons intégré la bibliothèque NLTK (Natural Language Toolkit), qui offre un accès à un vaste répertoire de mots anglais. Pour le français, en l'absence d'une bibliothèque Python dédiée, nous avons opté pour l'utilisation d'une liste de 20 000 mots français téléchargée préalablement. Cette approche nous permet de couvrir efficacement les deux langues et d'améliorer significativement la justesse des prédictions pour des saisies textuelles variées.

## 5.3. Démonstration Web

Un site web a été créé permettant à n'importe quel utilisateur de tester facilement la solution apportée par notre équipe à l'analyse des émissions sonores du clavier.

### 5.3.1. Présentation du site web

Le site web a été réalisé en HTML, CSS et JavaScript. Il communique également avec un fichier Python pour des traitements en back-end.

On y trouve trois sections fondamentales :

Premièrement deux boutons qui permettent de choisir le type de reconnaissance que l'on souhaite ; soit de l'utilisateur, soit de la saisie.


*Figure 5 Boutons de choix de la prédiction souhaitée*

Ensuite, un espace qui permet de lancer l'enregistrement au fur et à mesure que l'on écrit sur un espace dédié. Cet enregistrement s'arrête automatiquement après 10 secondes. A la fin du timer, le fichier audio récupéré est renvoyé vers un fichier python qui se chargera de le traiter, en renvoyant le spectrogramme de l'audio, ainsi que la prédiction que l'on souhaitait.


*Figure 6 Aperçu de la zone de saisie de texte et d'enregistrement*

En cliquant sur le bouton "Voir les résultats", on est dirigé vers la prédiction qui est notre troisième partie. Dans cette partie est affichée :

- La chaîne initiale
- La prédiction (vert si elle est correcte, et rouge si elle est incorrecte)
- Le nombre de lettres correctes

- Le nombre total de lettres de la chaîne initiale
- La performance de l'algorithme


*Figure 7 Affichage de la prédiction dans le site web*

# 6. Conclusion

## 6.1. Travail réalisé

En fin de compte, nous avons réalisé ce que nous avions prévu de faire :

- La réalisation des différentes présentations (kick-off, mi-parcours et finale)
- État de l'art scientifique
- Collecte d'une base de données
- Segmentation des audios collectés
- Extraction des caractéristiques des audios de manière optimale
- Détermination du meilleur choix de la méthode de classification grâce à Orange
- Développement des modèles de reconnaissance de touches et d'utilisateurs
- Entraînement des modèles et analyse de résultats
- Développement de la démo web

## 6.2. Difficultés rencontrées

Toutefois, nous avons rencontré de nombreuses difficultés :

- Le manque d'une base de données suffisamment variée et riche pour entraîner suffisamment les modèles

Trouver une méthode optimale de segmentation ; la segmentation demande beaucoup de contraintes qui semblent propres à chaque audio (vitesse de frappe, intensité de la frappe…). Avoir une mauvaise segmentation induit dès le début le résultat de notre modèle en erreur, qui se doit de conserver l'intégrité de cette donnée primordiale.

- Difficulté d'organisation entre les membres de l'équipe ; il a été plutôt difficile de partager les tâches et de répartir le travail entre chaque membre du groupe, compte tenu de nos emplois du temps respectifs divergents.

- Bien que les modèles montrent d'excellentes performances lorsqu'ils sont évalués avec les fichiers audios de la base de données, nous avons observé une baisse significative de leur efficacité lors des tests via la démo web. Cette différence suggère que le modèle pourrait rencontrer des difficultés à s'adapter à des conditions variées. Ce problème pourrait être lié à plusieurs facteurs, comme la possibilité d'un entraînement pas suffisamment robuste.

### 6.3. Perspective

Dans une perspective d'amélioration de notre modèle, plusieurs axes se dégagent. Tout d'abord, il faudrait résoudre les problèmes cités précédemment concernant la base de données et la segmentation, ce qui pourrait grandement améliorer les performances de la reconnaissance sonore du clavier. Nous pourrions également éliminer certaines contraintes que nous nous étions imposées, comme l'enregistrement dans un environnement sans aucun bruit. En éliminant ces contraintes, nous pourrions explorer des approches plus flexibles, applicables à des cas de la vie réelle. Nous pourrions aussi explorer d'autres approches utilisant par exemple le Deep Learning. En prenant ces mesures, nous pourrions ouvrir de nouvelles possibilités pour améliorer la robustesse et la précision de notre modèle de reconnaissance de touches de clavier.

ENSI CAEN

ÉCOLE PUBLIQUE D'INGÉNIEURS
CENTRE DE RECHERCHE

Ecole Publique d'Ingénieurs en 3 ans

6 boulevard Maréchal Juin, CS 45053
14050 CAEN cedex 04

ministère éducation nationale enseignement supérieur recherche

Normandie Université

Cti Commission des titres d'ingénieur

cdefi Conférence des Directeurs des Écoles Françaises d'Ingénieurs

European Accreditation of Engineering Programmes EUR-ACE®

CONFÉRENCE DES GRANDES ÉCOLES