

République Algérienne Démocratique et Populaire

Ministèredel’EnseignementSupérieuretdelaRechercheScientifique

#### UniversitédesSciencesetdelaTechnologieHouariBoumediene

Facultéd’Electroniqueetd’Informatique

DépartementInformatique

Mémoire de Licence

Filière: Informatique Spécialité:Académique

**Conception et développement d’une application de gestion d’un établissement scolaire en optimisant certaines tâches par les commandes vocales**

**Sujet Proposé par:**

**M (e) :**A.I AMROUS

**Présenté par :**

##### ZAIT Fouad

Binôme n° : 067/ 2021

# Remerciements

##### Avant tout, nous remercions Dieu tout-puissant, qui nous a bénis et permis d’atteindre là ou nous sommes aujourd'hui. Louange à Dieu qui nous a donné la volonté, la détermination et la capacité de mener à bien notre travail.

Nous remercions particulièrement madame Amrous Imen, qui nous a fait honneur en acceptant notre demande d'encadrement, grâce à sa patience, sa disponibilité et surtout ses judicieux conseils, qui ont contribué à alimenter nos réflexions.

Enfin, nous adressons nos sincères remerciements et notre gratitude à nos parents qui ont toujours été là pour nous et qui nous ont encouragé et soutenu pour mener à bien notre travail, on remercie également nos collègues, amis et tous ceux qui nous ont aidés.

**Sommaire**

Introduction générale 1

[Chapitre I :Introductionà la reconnaissance de la parole 2](#_TOC_250039)

1. [Introduction 2](#_TOC_250038)
2. [Généralité sur la parole 2](#_TOC_250037)
   1. [Définition 2](#_TOC_250036)
   2. [Domaines d’application de la reconnaissance vocale 2](#_TOC_250035)
   3. Les problèmes rencontrés dans la reconnaissance de la parole 3
3. [Système de la reconnaissance automatique de la parole (RAP) 3](#_TOC_250034)
   1. [Définition 3](#_TOC_250033)
   2. [La structure générale d’un système de reconnaissance automatique de la parole 4](#_TOC_250032)
   3. [Extraction de paramètres acoustiques 4](#_TOC_250031)
4. [Les paramètres MFCC 7](#_TOC_250030)
   1. [Modélisation des paramètres acoustiques 9](#_TOC_250029)
   2. [Décodage de la parole 9](#_TOC_250028)
5. [Conclusion 10](#_TOC_250027)

[Chapitre II : Conception 11](#_TOC_250026)

1. [Introduction 11](#_TOC_250025)
2. [Les différentes tâches de notre application 11](#_TOC_250024)
3. [Conception de notre système vocal de saisie de notes 15](#_TOC_250023)
   1. [L'Outil CMU Sphinx 15](#_TOC_250022)
4. [Architecture générale de notre système de RAP 16](#_TOC_250021)
   1. [Préparation de données 16](#_TOC_250020)
   2. [Entraînement (l’apprentissage) 21](#_TOC_250019)
   3. [Décodage 22](#_TOC_250018)
   4. [Evaluation de performance 22](#_TOC_250017)
5. [Conclusion 24](#_TOC_250016)

[Chapitre III : Développement de l’application 25](#_TOC_250015)

[.1 Introduction 25](#_TOC_250014)

1. [Présentation de l’environnement de développement 25](#_TOC_250013)
   1. [Environnement matérielle 25](#_TOC_250012)
   2. [Environnement Logiciel 25](#_TOC_250011)
   3. [Langages de développement 26](#_TOC_250010)
2. [Implémentation de notre système de reconnaissance vocale sous Android 26](#_TOC_250009)
   1. [Pocket sphinx 27](#_TOC_250008)
3. [Le déroulement et la présentation visuelle de notre application « digital school » 27](#_TOC_250007)
   1. [L’interface de l’administration 28](#_TOC_250006)
   2. [L’interface de l’enseignant 32](#_TOC_250005)
   3. [L’interface de l’étudiant 34](#_TOC_250004)
4. [Conclusion 35](#_TOC_250003)

[Conclusion générale 35](#_TOC_250002)

[Bibliographie 35](#_TOC_250001)

[Webographie 35](#_TOC_250000)

Figure 2-1 la reconnaissance de la parole chez l’humain. 2

Figure 3-1 Système de reconnaissance automatique de la parole 4

Figure 3-2 Echantillonnage d’un signal vocal 5

Figure 3-3 Préaccentuation d’un signal vocal 5

Figure 3-4 Segmentation du signal vocal en trames. 6

Figure 4-1 Étapes du calcul des paramètres MFCC 8

Figure 4-1 Le schéma général de système RAP avec l’outil Sphinx Train. 16

Figure 4-2 Le fichier de configuration pour l’extraction de paramètre acoustique 18

Figure 4-3 Grammaire utilisée pour la reconnaissance des notes. 19

Figure 4-4 Extraits defichiers digit\_train.fileids. 20

Figure 4-5 Un Extraits de fichier digit\_train.transcription 21

Figure 4-6 Fichier des sons superflus. 21

Figure 4-7 Effet du nombre d’états des HMM sur le taux de reconnaissance 23

Figure 4-1 interface de l’inscription. 28

Figure 4-2 interface principale de l’administrateur 29

Figure 4-3 l’ajout des classes aux étudiants et aux enseignants. 29

Figure 4-4 l’ajout des classes aux étudiants et aux enseignants. 30

Figure 4-5 l’envoi d’un message aux étudiants et aux enseignants. 30

Figure 4-6 l’envoi de la date de paiement aux étudiants et aux enseignants. 31

Figure 4-7 l’envoi de l’emploi du temps aux étudiants et aux enseignants. 32

Figure 4-8 l’interface principale de l’enseignant 32

Figure 4-9 la tache de la saisie vocale des notes. 33

Figure 4-10 la suite des autres taches de l’enseignant 33

Figure 4-11 l’interface de l’étudiant 34

Figure 4-12 visualisation de l’interface de l’étudiant (suite) 35

Tableau 2-1 L’utilisation de la reconnaissance vocale dans quelques domaines. 3

Tableau 4-1 Paramètres d’enregistrements utilisés pour la création de la base de données audio. 17

Tableau 4-2 Symboles de phonèmes. 18

Tableau 4-3 Extrait du fichier de dictionnaire phonétique 19

Tableau 4-4 Effet du nombre de paramètres MFCC sur le taux de reconnaissance 22

Tableau 4-5 Taux de reconnaissance moyen pour des locuteurs des deux sexes. 24

Tableau 3-1 Les principales méthodes de Pocket Sphinx. 27

##### Notre travail propose une application Android pour gérer un établissement scolaire ainsi que la prise en compte de ses nombreuses spécificités. Notre application permet de garantir une interaction permanente entre trois acteurs principaux : l’administration, les enseignants et les élèves. Elle permet d’automatiser et alléger beaucoup de tâches et rendre l’interaction de ces acteurs principaux plus interactive. Par exemple : les enseignants peuvent rédiger leurs remarques quotidiennement dans une application sur le téléphone, et les parent auront accès à tout moment à ces informations (devoirs à faire, bulletins de notes, absence).

De même, notre application intègre un module saisie vocale des notes, ceci va permettre à l’enseignant de saisir ces notes d’une manière plus rapide. Notre système de reconnaissance vocale est conçu à l’aide de l’outil CMU Sphinx4 qui est un projet Open Source de l’Université Carnegie Mellon, il est basé sur les modèles de Markov caché (HMM) et les paramètres (MFCC).

**Mot-clé:** Reconnaissance vocale, RAP, HMM, MFCC, Android.

##### De plus en plus, la technologie de la reconnaissance de la parole fait son chemin vers des applications réelles. Elle est déployée dans pas mal d’applications de différents domaines tel que la dictée automatique (transcriptions médicales, légales ou d'affaires), les applications d’aides pour les handicapes (distributeur automatique de billets pour personnes aveugles, etc.

Dans ce travail nous proposons la conception et le développement d’une application Android pour gérer un établissement scolaire ainsi que la prise en compte de ses nombreuses spécificités. Notre application permet de garantir une interaction permanente entre trois acteurs principaux : l’administration, les enseignants et les élèves. Elle permet d’automatiser et alléger beaucoup de tâches et rendre l’interaction de ces acteurs principaux plus interactive. Par exemple : les enseignants peuvent rédiger leurs remarques quotidiennement dans une application sur le téléphone, et les parent auront accès à tout moment à ces informations (devoirs à faire, bulletins de notes, absence).

##### De même, notre application intègre un module saisie vocale des notes, ceci va permettre à l’enseignant de saisir ces notes d’une manière plus rapide. Notre système de reconnaissance vocale est conçu à l’aide de l’outil CMU Sphinx4 qui est un projet Open Source de l’Université Carnegie Mellon, il est basé sur les modèles de Markov caché (HMM) et les paramètres (MFCC).

Ce mémoire s’articule autour de trois chapitres qui sont organisés suivant la description ci- dessous :

##### Dans le premier chapitre nous avons présenté le concept général de la reconnaissance de la parole, ses domaines et ses différents problèmes. Ainsi que les principales étapes de réalisation de ce dernier qui sont : l'extraction des paramètres, la modélisation et l’identification.

Le deuxième chapitre présente l’outil CMU Sphinx4 qu’on a utilisé pour la réalisation de notre système RAP, en décrivant aussi les différentes phases de conception de notre application. Nous avons clôturé ce chapitre par une série de tests qui nous a permet de sélectionner les meilleurs résultats du système.

Le troisième chapitre est consacré à l’implémentation de notre application sous Android en utilisant le système de reconnaissance vocale qu’on a réalisé. Nous allons présenter d’abord l’environnement de développement. Ensuite en achèvera le chapitre par une présentation visuelle de quelques interfaces de notre application.

### Chapitre I : Introduction à la reconnaissance de la parole

### Introduction :

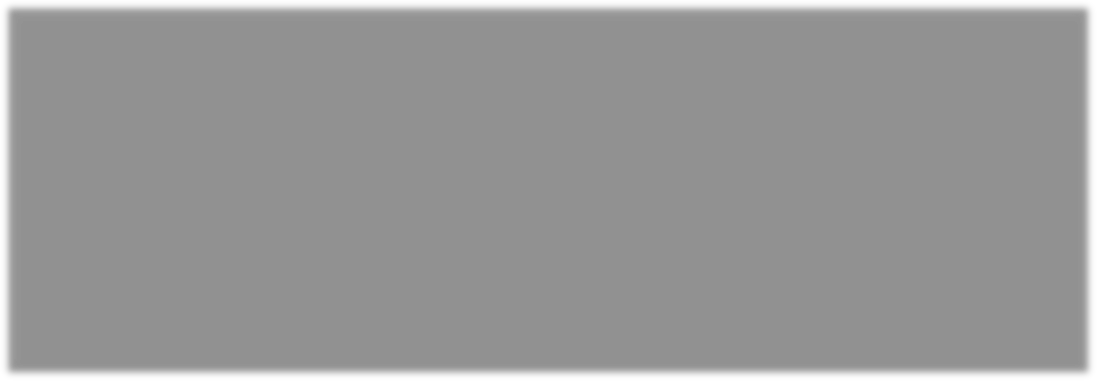
La reconnaissance vocale est l’une des techniques informatiques qui permet d’identifie les mots prononcés par l’être humain à l’aide d’un microphone et qui sera interpréter par un ordinateur ou un autre type de machine, cela signifie parler à votre machine et qu’elle reconnaisse correctement la parole puis elle la transforme en phrases textuelles.

Dans ce chapitre nous avons présenté le domaine de la reconnaissance vocale, nous avons commencé par définir la notion de la parole, ensuite nous passons à détailler et expliquer comment le système de la reconnaissance vocale s’exécute.

### Généralité sur la parole :

#### Définition :

La parole est un moyen de communication très efficace et naturel utilisé par l'humain permet l’expression de la pensée et de la communication entre les humains (Figure1.1).



**Figure 2-1 la reconnaissance de la parole chez l’humain.**

#### Domaines d’application de la reconnaissance vocale :

Aujourd’hui le système de la reconnaissance vocale est utilisé dans plusieurs domaines, il s’est répandu dans diverses applications. Le tableau suivant mentionne les différents domaines d’utilisation de système de reconnaissance vocal.

###### Tableau 2-1 L’utilisation de la reconnaissance vocale dans quelques domaines.

|  |  |
| --- | --- |
| **Domaine** | **Utilisation de la reconnaissance de la parole** |
| **La sécurité** | * La signature vocale |
| **L’industrie** | * L’inspection industrielle où les mains sont occupées par d'autres tâches |
| **Enseignement** | * Accessibilité des étudiants souffrant d’un handicap physique ou d’un tremble du langage. * Permettre aux élèves souffrant de difficultés d’apprentissage de faire des progrès considérables dans les domaines de la lecture, de l’écriture et de l’orthographe ! |
| **La santé** | * Saisies d’informations médicales et finalisation des comptes rendus en toute mobilité, pour une plus grande liberté des médecins ! * Les équipements destinés aux personnes handicapées. * Rapports de radiologie, lettres légales, etc. |
| **L’aviation** | * Commandes diverse |

* 1. **Les problèmes rencontrés dans la reconnaissance de la parole :**

Cette technique comme elle a beaucoup d’avantages elle ne cause aussi pas mal de problèmes. On doit faire attention aux critères suivants pour avoir un système qui a un taux d’erreur faible :

**- Le bruit persistant lors de l’enregistrement :** le bruit entraine un problème d’entrée et nous pouvons l’éviter en utilisant le système dans un endroit très calme ou bien on fait appel aux méthodes de filtrages. [site-filtrage].

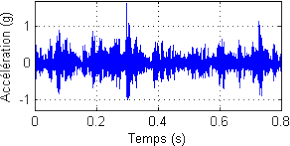
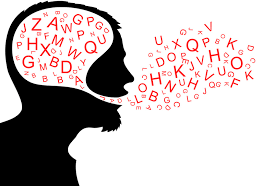
***-* La variabilité du signal de la parole *:*** la variation du signal de la parole dépond de plusieurs facteurs, comme la vitesse de la parole, le sexe de locuteur, etc.

- **La taille du vocabulaire :** plus le vocabulaire est large, plus il est difficile de l’identifie.

### Système de la reconnaissance automatique de la parole (RAP) :

#### Définition :

Un système de reconnaissance automatique de la parole est un système informatique qui analyse la voix humaine capté par un microphone et la convertit en texte lisible, c’est pour cela nous utilisons des méthodes numérique et probabiliste ainsi que des connaissances sur la langue à reconnaitre pour décider ce que le locuteur a le plus probablement dit.



Système de reconnaissance

vocale

Les mots prononcés

Signal obtenu

Texte reconnu

**Figure 3-1**Système de reconnaissance automatique de la parole.

#### La structure générale d’un système de reconnaissance automatique de la parole :

On peut décomposer schématiquement la structure d’un système de reconnaissance automatique de la parole(RAP) en trois modes principaux : l'extraction des paramètres, la modélisation et l’identification

**L’extraction des paramètres** : consiste à obtenir un ensemble de paramètres pertinents à partir du signal vocal. Ce calcul est réalisé en deux phases : la mise en forme de signal suivi par le calcul de paramètres proprement dit.

**Module Modélisation :** c’est une étape très importante dans le système RAP, elle dépond des vecteurs sonores obtenu à l’étape précédente afin de créer des modèles statistiques et de les enregistrer dans une base de modèles d’apprentissage pour les utilise à l’étape d’identification.

**Module d’identification :** cette étape dépend de la comparaison entre le signal du haut-parleur avec les modèles statistiques issus de l’étape précédente, afin de déterminer le texte associé au signal entrant.

#### Extraction de paramètres acoustiques :

L’extraction de paramètres acoustiques c’est une étape dont le but d’extraire un ensemble de paramètres pertinents et retirer l’information non pertinente mais avant ça, elle passe par deux phases : la mise en forme de signal (la phase de prétraitement de la parole) suivi par le calcul de paramètres proprement dit qu’on va bien détailler et les expliquer à la suite.

###### Mise en forme du signal vocal (prétraitement de la parole) :

Cette étape dépend du traitement des signaux obtenus par le microphone pour les adapter à l’étape d’extraction d’information audio afin de calculer les coefficients acoustiques [Amrous- 2015], en appliquant une série d’opérations :

* + - 1. **Echantillonnage :** L’échantillonnage c’est la conversation de signal à temps continue a(t) en signale à temps discret a(nTe) c’est à dire produit une suite de valeurs discrètes nommées échantillon.



Echantillonnage

###### Figure 3-2 Echantillonnage d’un signal vocal.

* + - 1. **Préaccentuation :** En utilisant les échantillons précédents, la préaccentuation du signal est utilisée pour normaliser les valeurs obtenues en appliquant l’équation suivante sur chaque échantillon : (1.1) [Rabiner-1993]:

*sp* *n*  *se* *n* *se* *n* 1

###### (1.1)

Avec 0.9   1et *n* est l’instant de l’échantillon.



Préaccentuation

###### Figure 3-3 Préaccentuation d’un signal vocal.

* + - 1. **Segmentation :** Comme les méthodes du traitement de signal utilisées dans l’analyse du signal s’appliquent sur des signaux stationnaires (indépendant de temps) [Fohr-2006]. Et on sait que les signaux de parole sont des signaux non stationnaires et pour résoudre ce problème il faut découper le signale rentrant en une suite de segments ou trames d’une durée considérablement petite. La durée de ces segments est d’environ 20 à 30 ms, avec un chevauchement entre deux trames consécutives d’environ 10 à 15 ms.



Segment 1

Segment2 segment n

###### Figure 3-4 Segmentation du signal vocal en trames.

* + - 1. **Fenêtrage :** La segmentation du signal en trames produits des discontinuités aux frontières des trames pour régler ce problème, des fenêtres de pondération sont utilisées. Parmi les fenêtres les plus utilisées dans la littérature est la fenêtre de Hamming [Harris-1978]:

*s f* *n*  *w**n* *sp* *n*

###### (1.2)

Avec

*w**n*  0.54  0.46 cos 2 *n* 

0  *n*  *N* 1

###### (1.3)

 *N*  1





Où : N est le nombre d'échantillons de chaque trame.

###### Calcul de paramètres acoustiques :

Après que la mise en forme du signal vocale est prête nous passons à l’étape de calcule de paramètres acoustiques ou la transformation de chaque trame fenêtrée du signal en un vecteur acoustique constitué d’un ensemble de paramètres.

Pour le calcul de ces paramètres il existe plusieurs méthodes, nous citons :

* La méthode PLP *(Perceptual Linear Prediction*) [Hermansky-1990].
* La méthode MFCC (*Mel Frequency Cepstral Coefficients)* [Debyeche-2007]*.*
* La méthode LPC (*Linear Prediction Coding*) [Davis-1980].
* La méthode LPCC (*Linear Prediction Cepstral Coefficients*) [Jankovski-1995].

Nous avons choisi d’utilisé les paramètres MFCC, étant donné que c’est les paramètres les plus utilisé dans la littérature.

### Les paramètres MFCC :

Pour l’obtention d’un vecteur des coefficients MFCC nous passons par les étapes suivantes :

###### Le calcul de la transformée de Fourier discret :

Chacune des trames, de N valeurs est convertie par La transformé de Fourier discrète (*Discret Fourier Transformée****, DFT)***du domaine temporel au domaine fréquentiel selon la formule suivante :

*S**K* 

*N* 1

 *s f*



*n*0

*n**e*

2 *j* *Kn N*

0  *K*  *N* 1**(1.4)**

Avec 0 K  N-1

N est le nombre d’échantillons.

Sf (n) un signal discret 0 ≤ n ≤ N

###### Transformation en échelle de Mel :

L’échelle Mel est une échelle linéaire aux bases fréquences et logarithmique en hautes fréquence. Au cours de cette étape une conversion d’une échelle linéaire à une échelle Mel est nécessaire pour le calcul des paramètres MFCC, et cela se fait selon la formule suivante :

*m*  2595log

1  *f* 

10 

##### 



700 

###### (1.5)

Ou f est la fréquence linéaire.

###### Clacul de Log d’energie :

Après la phase de transformation en échelle Mel, on calcule le logarithme du spectre d’amplitude de chaque filtre selon l’équation suivante :

*S**m*  ln *N* 1 *S**K* *H*



*m*

*k* 0

*K* 





0  *m*  *M*

###### (1.6)

Où Hm est l’amplitude du *mième*filtre

###### DCT inverse :

Afin d’obtenir les coefficients MFCC à partir des logs d’énergies de l’étape précédente on applique la transformée en cosinus discrète inverse (*Discret* Cosinus *Transform)* qui consiste à convertir le spectre logarithmique de Mel en échelle temporelle en utilisant la formule suivante :

*M* 1

 *n*(*m*  1 ) 

2

 

###### (1.7)

*cn*   *S**m*cos *M*  0  *n*  *M*

*m*0  





##### 



###### Calcul des caractéristiques dynamiques des MFCC :

Afin d’améliorer considérablement les taux de reconnaissance en ajoutant les dérivées première *c* et

seconde

*c* des coefficients MFCCs.

Le vecteur MFCC dans ce cas est représenté comme suit :

*ok* 

 *c* 

 

*k*

 *ck* 



*k* 

###### (1.8)

Où :

*c* 

*iéme*

* *ck* : le vecteur MFCC de la *k* trame.
* *ck*  *ck*  2  *ck* 2 : Première dérivée des MFCC, calculée à partir des vecteurs de MFCC de la (

*kiémé*  2 ) trame et ( *Kiéme*  2 ) trame.

* *cK*  *ck* 1  *ck* 1 : Seconde dérivée des MFCC.

La figure 1 résume les étapes du calcul des paramètres MFCC :



Mise en forme du

signal

DF

Log énergie

Transformation en échelle

Mel

MFCC

DCT

|  |
| --- |
|  |
|  |
|  |
|  |
|  |

Une fois les deux phases principales d’extraction de paramètres acoustique sont terminées, ces derniers vont être modélisés comme c’est illustré dans ce qui suit.

#### Modélisation des paramètres acoustiques :

La modélisation est l’une des étapes importantes dans le système de reconnaissance vocale, elle permet de modéliser les unités acoustiques prononcées. Pour cela on a plusieurs techniques : les réseaux de neurones ANN[Robinson et Fallside-1991 ; Robinson-1994], les SVM [Vapnik- 1998], ainsi que, les modèles de Markov cachées, ou HMM (Hidden Markov Models) [Baker-1975 ; Jelinek-1976 ; Rabiner-1989]. Ces derniers sont l’une des modélisations les plus utilisées, c’est ainsi que nous avons choisis de les appliquer dans notre travail.

###### Définition d’un HMM :

Formellement, Un modèle de Markov est défini par (Q, π, A, B) [Barrlaut-08],tels que :

* Q est un ensemble (fini) d’états, ce dernier démarre de l'état initial 1, à l'instant *t*=1 et arrive à l'état final *N* à l'instant *t=T.*
* π le vecteur des probabilités initiales :

*i*  *P*(*q*1  *i*)

1  *i*  *N*

###### (1.9)

* *A*  *a*  la matrice de transition où l’élément *a* représente la probabilité de transition de l'état *i* à l'état *j* .

*ij ij*

*aij*  *P*(*qt* 1 

*j* / *qt*  *i*)

,1  *t*  *T* ,

1  *i*, *j*  *N* **(1.10)**

La somme des probabilités de transitions entre un état *i* et tous les autres états doit être égale 1,

*j ij* ,

 *a*

 1

*N*

1  *i*  *N* **(1.11)**

* *B*  *bi* (*ot* )est la matrice de probabilité de la distribution des observations *ot*

*i* :

à l'instant *t* et à l’état

*bi* (*ot* )  *p*(*ot* / *qt*

 *i*) **(1.12)**

#### Décodage de la parole :

Soit *O= o1 o2……oT* une suite d'observations acoustiques, (*M1, M2,…. MR*) : sont les modèles possible, *M bes*t

,1 *best*  *R* le modèle le plus probable connaissant la suite d’observations acoustiques *O* :

*M best*  arg max

*M*

*P*(*M* / *O*) **(1.13)**

Grâce à la règle de Bayes, il est possible d'écrire :

*P*(*M* / *O*)  *P*(*O* / *M* )*P*(*M* ) **(1.14)**

*P*(*O*)

Avec :

* *P* (*O/M*) : la probabilité du modèle acoustique.
* *P*(*M*) : la probabilité du modèle de langage*.*
* *P(O)* : la probabilité d'occurrence de la suite d'observations acoustiques *O*.

Comme *P(O)* est une entité qui reste constante pour tous modèles, l’équation précédente peut être réduite à :

*M best*  arg max

*M*

*P*(*O* / *M* )*P*(*M* )

**(1.15)**

### Conclusion :

Au long de ce chapitre nous avons présenté une introduction générale à la reconnaissance de la parole, ses problématiques et ses difficultés, ainsi que ses différents domaines d’applications. Ensuite nous avons présenté la structure générale de SRAP, en expliquant ces trois principaux modules : l’extraction de paramètre, modélisation acoustique et identification.

Le chapitre suivant présente notre conception pour créer une application de gestion d’une école qui va inclure un module de saisie de notes par commandes vocales.

### Chapitre II : Conception

### Introduction :

Dans ce chapitre nous allons présenter les différentes phases de conception de notre application de gestion d’un établissement scolaire. Nous mettons plus l’accent sur la conception de système de reconnaissance automatique de la parole qui permet la saisie automatique de notes. Ce système est basé sur les modèles de Markov cachés (HMM) et nous l’avons conçu en utilisant l’open source CMU Sphinx.

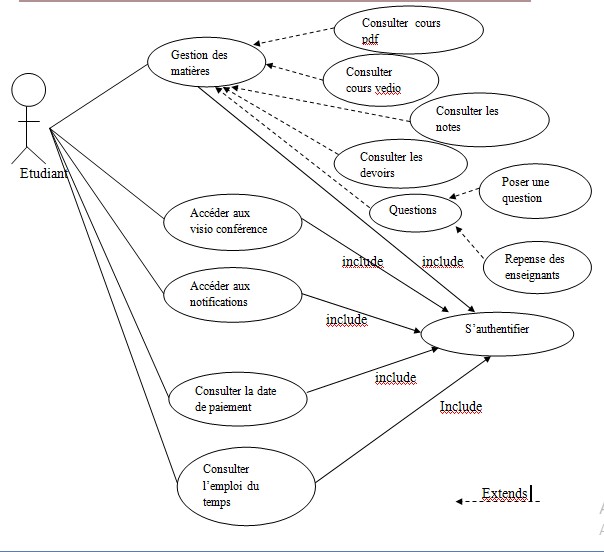
### Les différentes tâches de notre application :

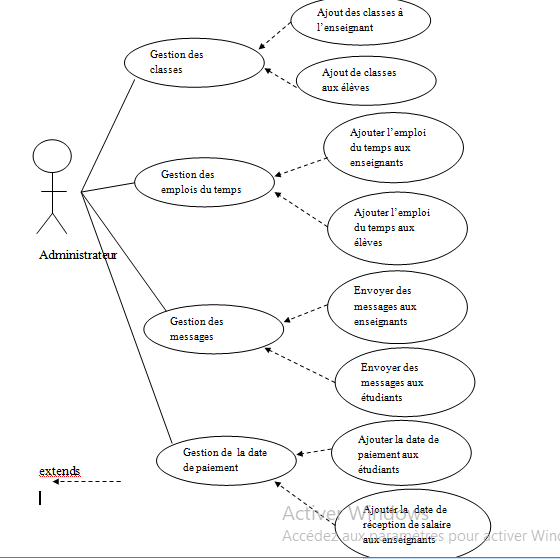
On a conçu 2 modules pour notre application, d'abord les élèves et les enseignants devront s'inscrire et attendre l'acceptation de l'inscription de la part de l'administration.

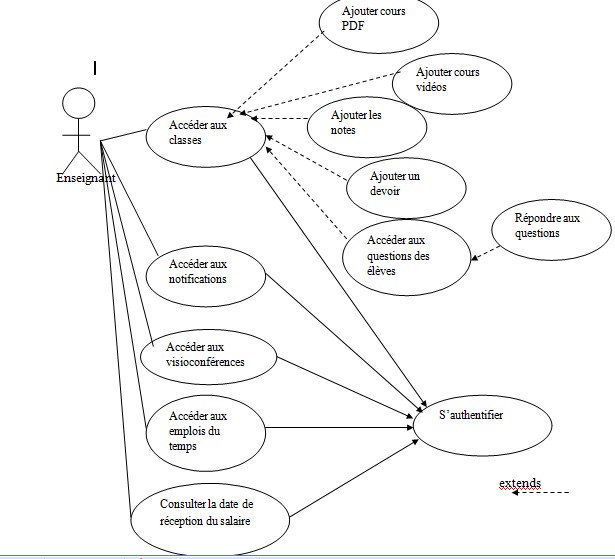
**L'administration** : Quand l'administrateur accédera à son interface il pourra leur effectuer leurs classes, envoyer des messages qui seront des notifications pour les élèves et les enseignants, définir leurs emploie du temps et définir la date pour procéder au paiement pour les étudiants et pour les enseignants ca sera la date pour venir se présenter à l'administration pour recevoir leurs salaires.

**L'enseignant :**il aura accès à ses classes ajouté par l'administration , en choisissant une classe attribué il pourra envoyer des cours PDF , des cours vidéos, des devoirs ,saisir des notes par des commandes vocaux et il pourra aussi voir les questions des élèves et leurs répondre , il aura accès aussi au visio conférence(skype,webex,zoom,google meet), il pourra aussi recevoir les messages de l'administrateur qui seront des notifications , il pourra consulter son emploie du temps et la date pour venir se présenter à l'administration pour recevoir son salaire .

**L'élève:** En choisissant la matière qu'il veut, il aura accès au cours PDF, vidéos et devoirs envoyés par l'enseignant de cette matière, il pourra questionner l'enseignant et consulter des éventuelles réponses , il pourra également voir sa note ajouté par son enseignant , il pourra consulter son emploie du temps , sa date de paiement , accéder au visio conférence et recevoir des notifications de l'administration .







Comme nous l’avons expliqué au début de chapitre, notre application doit contenir un module qui permet aux enseignants la saisie vocale de leurs notes. Pour créer un tel module, nous avons conçu tout un système de reconnaissance de la parole qui permet la reconnaissance de notes. Ce qui vaut dire, les chiffres de zéro à vingt, la virgule, le chiffre vingt-cinq et le chiffre soixante-quinze.

### Conception de notre système vocal de saisie de notes :

Dans cette partie, nous allons détailler les différentes étapes qui nous ont permet de concevoir notre système de RAP, à base de l’outil CMU Sphinx. En commençant par les étapes de préparation de données audio et linguistique, ensuite nous allons expliquer comment nous avons entrainé ce système, et puis l’évalué pour tester sa robustesse.

#### L'Outil CMU Sphinx :

CMU Sphinx est une série de bibliothèques et d'outils de développement de reconnaissance vocale.

Le CMU Sphinx comprend entre autres les outils suivants :

* + - **Sphinx 4 :** Une réécriture complète du Sphinx en Java. Il offre à la fois la précision et la rapidité.
    - **Sphinx train :** Une suite d'outils qui permet de créer le modèle acoustique.
    - **CMU-Cambridge Language Modeling Toolkit :** Une suite d'outils qui permet de créer le modèle de langage.

###### Présentation de CMU sphinx4 :

C’est un projet open source de l’université Carnegie Mellon (CMU) permet à des groupes de recherche avec des budgets modestes de développer et de conduire des applications de recherches dans le domaine de la reconnaissance vocal. CMU Sphinx 4 est une librairie de classes et d’outils disponible en langage de programmation Java. Il utilise des HMM continus et fournit une grande flexibilité, exactitude et vitesse. Sphinx-4 est un système modulaire, flexible, accepte différentes grammaires et langues.

###### Sphinx Train :

Est l’outil crée par CMU pour le développement des modèles acoustiques. C’est un ensemble de programmes et documentations pour réaliser et construire des modèles acoustiques pour n’importe quelle langue.

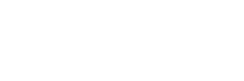
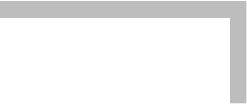
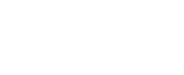
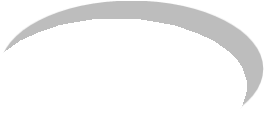
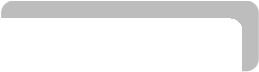
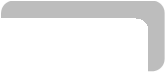
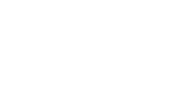
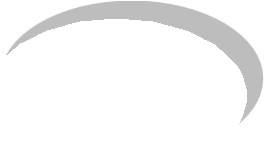
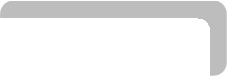
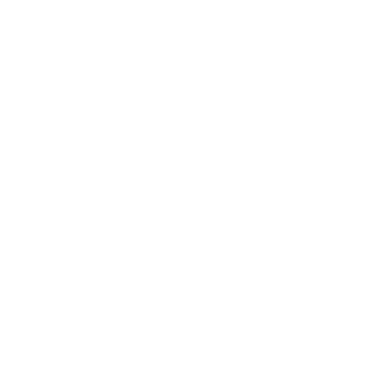
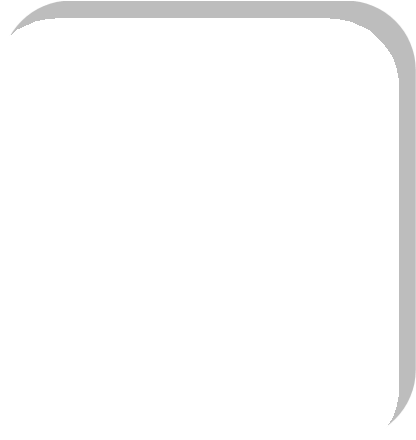
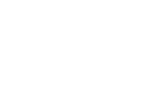
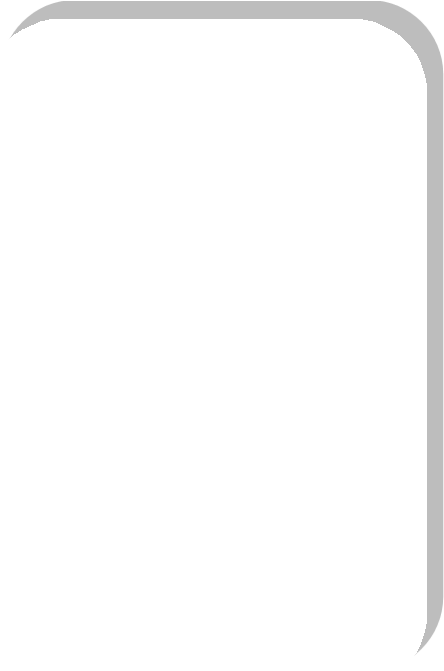
Les différentes librairies qui composent Sphinx Train :

* + - * ActivePerl : L’outil pour éditer des scripts pour Sphinx Train et permet de travailler dans un Unix-like environnement pour Windows plateforme [site-0].
      * Microsoft Visual Studio : Pour compiler les sources en C afin de produire les Exécutables.

### Architecture générale de notre système de RAP :

L’architecture générale de notre système RAP est basée sur le regroupement d’un ensemble de données d’entrées appelé la base de données de traitement et les traiter avec l’outil Sphinx Train pour créer les modèles acoustiques. La figure suivante illustre le schéma général de ce système avec l’outil Sphinx Train.

**Figure 4-1 Le schéma général de système RAP avec l’outil Sphinx Train.**



Les données

acoustiques

Sphinxtrain

Les modèles

acoustiques

**Les fichiers de**

**configuration :**

RAP

-Le dictionnaire

-Liste des phonèmes

-Le fichier de transcription

Le modèle

de langage

Grammaire

syntaxique

Texte reconnu

#### Préparation de données :

Dans la conception notre système de reconnaissance nous avons deux types de données d’entrées. D’abord nous allons présenter l’ensemble de données acoustiques que nous avons préparé soit manuellement, ou en utilisant des scripts. Ensuite nous allons présenter les différents fichiers de données linguistiques.

###### Préparation de données acoustiques :

Nous avons commencé par enregistrer une base de données audio, en suite nous avons traité cette dernière pour extraire ses paramètres acoustiques. Le détail de ces traitements est présenté dans ce qui suit

###### Création de données audio :

Notre corpus est constitué de chiffres de ‘zéro’ jusqu’à ‘vingt’, ainsi que les chiffres : 25, 75, et le mot ‘virgule’. Ces mots sont utilisés pour décrire n’importe quelle note. Il contient aussi le mot ‘envoyer’ pour sauvegarder la note de l’élève dans notre base de données principale.

Le corpus est composé de vingt répétitions de chaque mot par 20 locuteurs. Dans ce tableau nous résumons les paramètres d’enregistrement que nous avons adapté.

###### Tableau 4-1 Paramètres d’enregistrements utilisés pour la création de la base de données audio.

|  |  |
| --- | --- |
| Paramètre | Valeur |
| Echantillonnage | 16 kHz, 16 bits |
| Format audio | Wav |
| Corpus | 23 chiffres + le mot virgule+ le mot ‘envoyer’ |
| Répétition | 20fois pour chaque chiffre |
| Locuteur | 20 locuteurs |
| Durée totale de base de données | 10 000s |

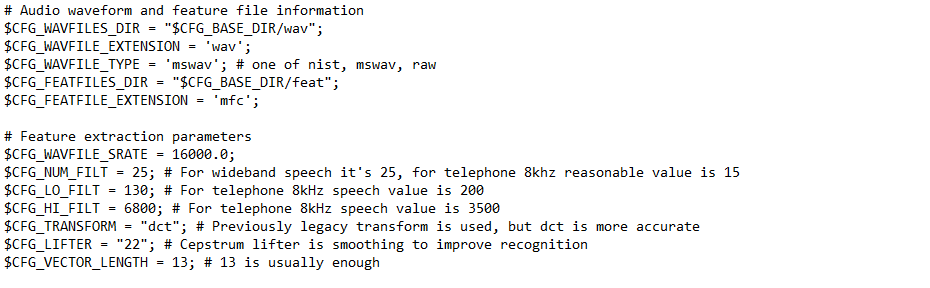
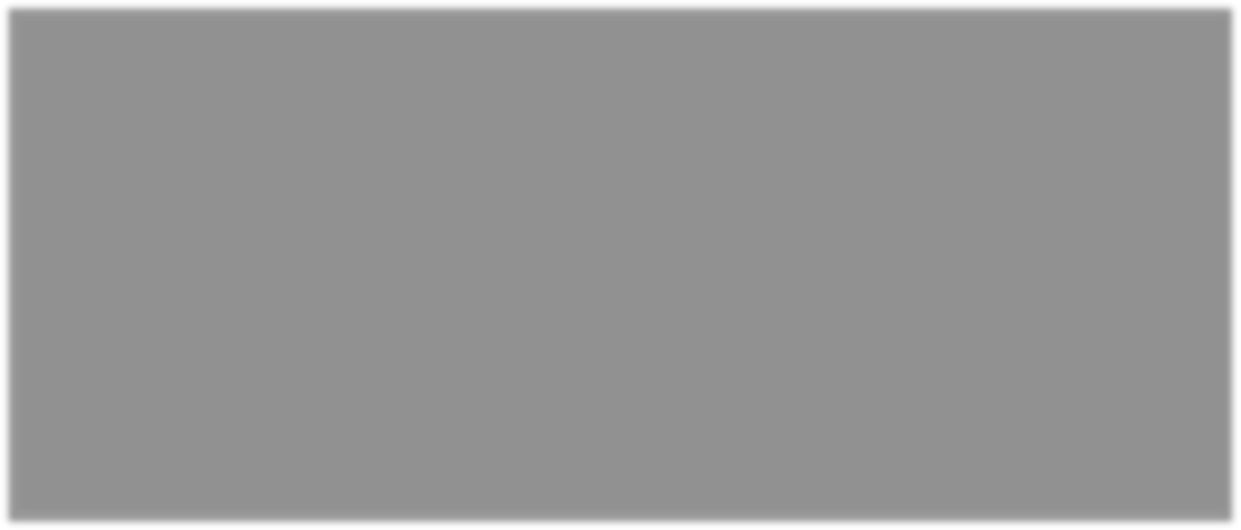
**Codification du corpus :**

Les enregistrements sont codifiés de façon simple et compréhensible. La structure de codification de corpus est représentée comme suit :

* + - * + pour les nombres de 0 à 20 et 25 50 75 :**Loc\_n° du locuteur\_nombre.wav**
        + pour les mots virgule et envoyer :**Loc\_n°du locuteur\_code mot.wav**

###### Extraction de paramètres acoustique :

Cette étape consiste à extraire l’ensemble des paramètres acoustiques qui caractérisent chaque fichier audio. Le signal parole sera transformé en une série de vecteurs de caractéristiques (feature vectors) comprenant les coefficients MFCC[Varela-2003]. L’extraction de ces paramètres se fait en utilisant un fichier de configuration, qui mentionne les différents paramètres nécessaires à l’algorithme d’extraction de MFCC, mentionné en chapitre 1. La figure suivante représente le fichier de configuration.



###### Figure 4-2 Le fichier de configuration pour l’extraction de paramètre acoustique.

* + 1. **Conception des données linguistiques :**

Afin de réaliser l’entrainement de notre système de RAP. Certains fichiers de données linguistiques doivent être préparés selon une syntaxe bien précise. Ces fichiers sont les suivants :

###### Liste des phonèmes :

Afin de créer le dictionnaire phonétique nous avons besoins de décrire le contenu phonétique de chaque mot de dictionnaire. Le tableau ci-dessous représente les phonèmes utilisés dans notre cas.

###### Tableau 4-2 Symboles de phonèmes.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **phonème** | **symbole** | **Phonème** | **symbole** | **phonème** | **symbole** | **phonème** | **symbole** |
| œ̃ | un | A | a | ɥ | hu | u | y |
| d | d | K | qu | n | n | ʁ | r |
| ø | eux | s | s | œ | eu | ɔ | o |
| t | t | ẽ | ingt | f | f | ɛ̃ | in |
| r | re | i | i | ɔ̃ | on | v | v |
| w | ois | ɛ | ep | z | z | ɡ | g |

* + - 1. **Le dictionnaire phonétique :**

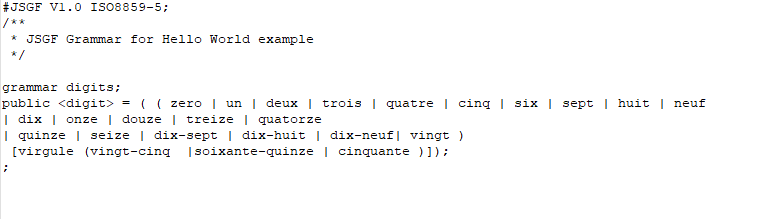
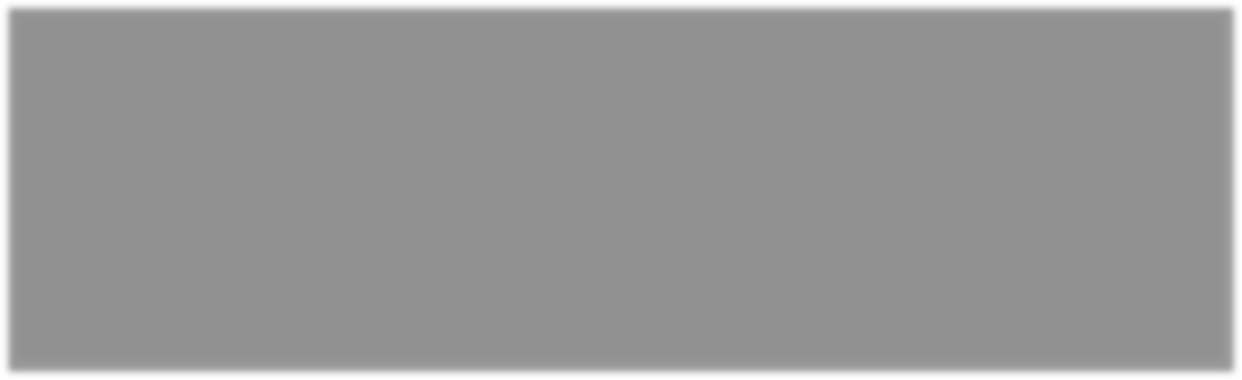
Il permet de donner une représentation phonétique pour chaque mot. Le tableau suivante montre une partie fichier dictionnaire de notre application.

###### Tableau 4-3 Extrait du fichier de dictionnaire phonétique.

|  |  |
| --- | --- |
| **Mot du dictionnaire** | **Décomposition phonétique** |
| Un | œ̃ |
| deux | d ø |
| quatre | k a t r |
| dix-sept | d i s s ɛ t |
| vingt | v ẽ |
| Virgule | v i ʁ ɡ y l |

* + - 1. **Modèle de langage (grammaire) :**

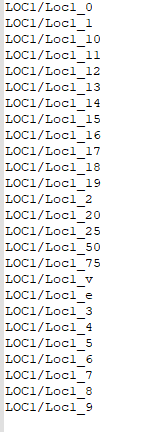
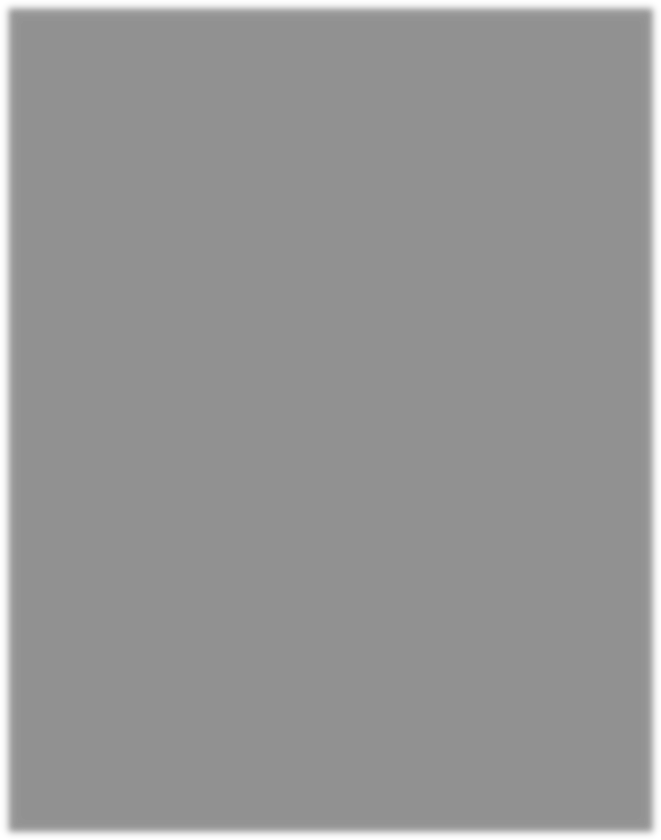
C’est un modèle qui définit l’usage des mots dans une application. Chaque mot dans le modèle de langage doit être définit dans le dictionnaire de prononciation. Le choix d’un modèle de langage dépend de l’usage de l’application. Dans notre cas la grammaire syntaxique est créée manuellement avec le Java Speech Grammar Format (JSGF)



###### Figure 4-3 Grammaire utilisée pour la reconnaissance des notes.

* + - 1. **Liste des fichiers :**

C’est un fichier texte listant la liste de tous les enregistrements audio. Il est créé pour l’utilisé dans la phase d’entrainement de notre système RAP. Nous avons créé ce fichier d’une manière automatique en citant dans les noms de fichiers qui se trouvent dans le dossier de la base de données audio enregistrée. La figure suivante est un exemple de notre fichier«**digit\_train.fileids** ».



###### Figure 4-4 Extraits defichiers digit\_train.fileids.

* + - 1. **Transcription textuelle :**

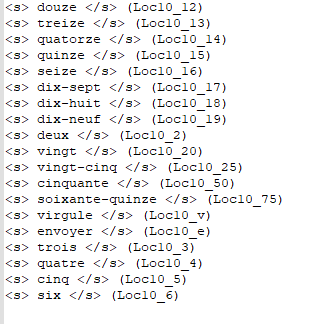
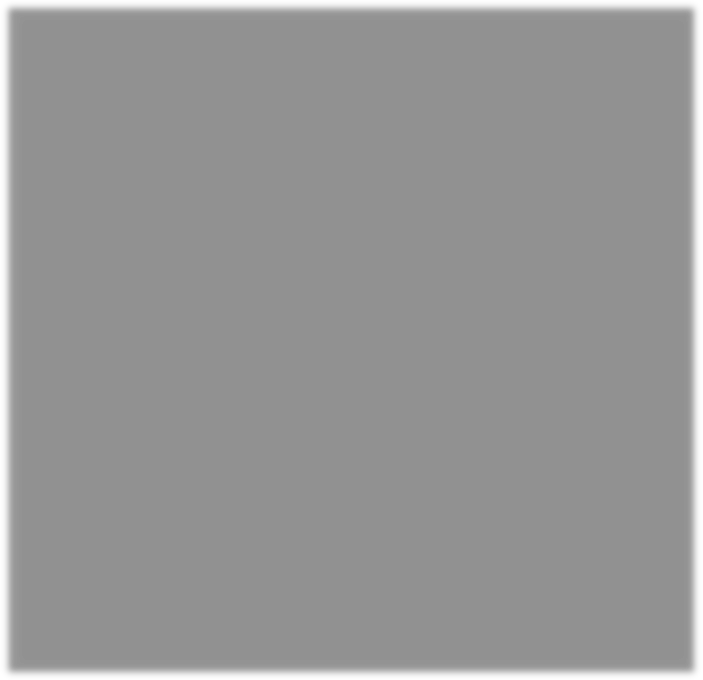
L’étape d’entrainement nécessite, en plus des fichiers audio, la transcription textuelle de chaque fichier. Pour cela,nous avons conçu un script Perl qui permet de remplir ce fichier de façon automatique en se basant sur la codification d’enregistrement que nous avons adopté. L’algorithme utilisé est annoncé comme suit :

###### Début

* Ouvrir le fichier **«digit\_train.Transcription : Transcription textuelle »**.
* Ouvrir le fichier **«digit\_train.fileids : Liste fichiers ».**
* Mettre dans une structure de donnée un code pour chaque mot de vocabulaire.
* Pour chaque **ligne** de **« Liste fichiers »** :
  + Extraire les deux derniers caractères.
  + Comparer les deux caractères extraits avec les codes des mots de vocabulaire.
  + Ecrire le mot correspondant dans le fichie**« digit\_train.Transcription »**.
  + Ajouter <s> et </s> avec les noms des fichiers entre parenthèse.

###### Fin.

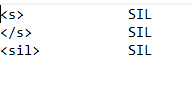
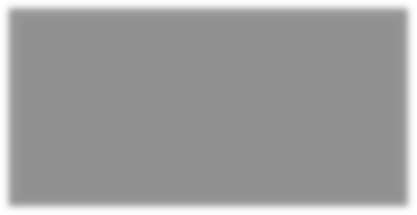
Nous précisons que la syntaxe <s> mot <s/> (nomFichierWav) est une exigence de l’outil CMU-SPHINX. Un Extraits de fichier digit\_train.transcription est illustré dans la figure suivante :



###### Figure 4-5 Un Extraits de fichier digit\_train.transcription .

* + - 1. **Liste des sons superflus :**

C’est un fichier qui contient les codes utilisé dans la transcription textuelle, alors qu’ils ne représentent pas de la parole (les rires, les sonneries du téléphone, le silence, la respiration). Dans notre application nous avons juste considéré le silence. La figure suivante décrit un exemple de notre fichier utilisé :



**Figure 4-6 Fichier des sons superflus.**

#### Entraînement (l’apprentissage) :

Une fois nous avons terminé la préparation de tous les fichiers nécessaires, nous entamons l’étape d’entrainement des modèles HMMs. Cette étape est réalisée à travers les outils de « sphinx train ». Durant la phase d’apprentissage, chaque unité acoustique ou phonème est représentée par un modèle statistique décrivant la distribution de ces données. Ces données statistiques se composent principalement la moyenne et la covariance, ainsi que la matrice de transition des HMMs.

Après l’apprentissage, le modèle acoustique devrait avoir les fichiers suivants :

* + - mdef
    - feat.params
    - mixture\_weights
    - means
    - noisedict
    - Transition\_matrices
    - variances

#### Décodage :

Le rôle du décodage est de transformer le signal acoustique, en une suite d'unités linguistique. Cette phrase est réalisée en se basant sur les modèles acoustiques crées lors de l’étape d’entrainement ainsi que sur le modèle de langage.

#### Evaluation de performance :

Dans le but d’évaluer les performances de notre système, nous avons effectué un ensemble de tests. Ces derniers sont menés sur un certain nombre de propriétés différentes du modèle acoustique, parmi eux nous mentionnons les suivants :

* + - Le choix de nombre de paramètres MFCC.
    - Le nombre d’état de modèle HMM.
    - Le choix du sexe de locuteur.

###### Influence du nombre de paramètres MFCC :

Afin d’obtenir le meilleur taux de reconnaissance nous avons étudié l’influence du nombre de coefficients des paramètres MFCC en variant la taille de vecteurs de ces derniers.Les résultats sont mentionnés dans le tableau suivant :

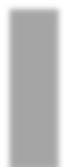
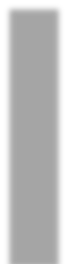
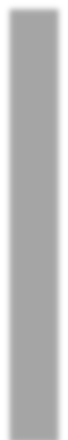
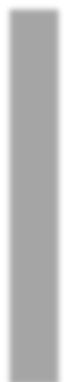
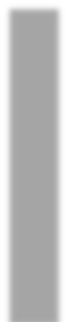
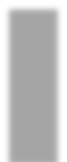
###### Tableau 4-4 Effet du nombre de paramètres MFCC sur le taux de reconnaissance.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Nombre de coefficients MFCC** | 8 | 9 | 10 | 11 | 12 | 13 | 14 |
| **Taux de reconnaissance** | 88.9% | 89.55% | 91.23% | 92.11% | 92.98% | 91.69% | 90.77% |

D’après l’observation du taux de reconnaissance nous avons remarqué que les MFCC avec 12 paramètres ont le taux de reconnaissance le plus élevé.

###### Influence du nombre d’états des HMMs :

Pour atteindre la valeur maximale de taux de reconnaissance nous avons aussi étudié l’effet du changement du nombre d’états par HMM sur notre système. La figure ci-dessous montre les différents résultats obtenus.



**taux de reconnaissace**

97.00%

96.00%

95.00%

94.00%

93.00%

92.00%

taux de reconnaissace

91.00%

90.00%

89.00%

88.00%

87.00%

état1

état2

état3

état4

état5

état6

###### Figure 4-7 Effet du nombre d’états des HMM sur le taux de reconnaissance.

D’après les résultats de la figure 2.7 nous remarquons que le taux de reconnaissance atteindre une valeur maximale de 95.65% lorsqu’on fixe le nombre d’état à quatre, puis se détériorent à partir de cinq états.

###### 4.4.1 Influence de performance par a port au choix du sexe de locuteur :

Après avoir déterminé une combinaison assez bien des paramètres du modèle acoustique, nous avons fait des tests pour diffèrent locuteurs des deux sexes. Les résultats sont mentionnés dans le tableau suivant :

###### Tableau 4-5 Taux de reconnaissance moyen pour des locuteurs des deux sexes.

|  |  |  |
| --- | --- | --- |
| **Locuteurs** | **Nombre de locuteurs** | **Taux de reconnaissance** |
| **Male** | 3 | 92.95% |
| **femelle** | 2 | 90.30% |

Vu la taille de notre corpus d’apprentissage qui est relativement petit notre résultat sont satisfaisants.

### Conclusion :

Au cours de ce chapitre, nous avons présenté d’abord les différentes tâches de notre application, en précisant que c’est une application de gestion d’un établissement scolaire qui doit contenir un module de saisie automatique de notes. Ensuite, nous avons présenté à la présentation détaillée, de la conception de notre système de dictée vocale de notes. Ce dernier est conçu à base de l’outil CMU Sphinx. Cette conception à nécessité la préparation d’une base de données audio et linguistique, ainsi que d’autres fichiers nécessaires en respectant la syntaxe propre à notre outil de conception. Nous avions clôturé ce chapitre en effectuant des tests, pour évaluer notre système de reconnaissance conçu, afin de sélectionner les meilleurs résultats qui s’adaptent avec notre application.

Le chapitre suivant va présenter l’environnement de développement de notre application et le déploiement du système de reconnaissance conçu.

### Chapitre III : Développement de l’application

### Introduction :

Dans ce chapitre, après qu’on a terminé la conception de notre système nous avons procédé à la phase de développement ou nous avons présenté d’abord tous les outils (matériel, logiciel et langages) utilisés, ainsi que l’implémentation de notre système et le déroulement de la reconnaissance vocale dans un projet Android avec l’utilisation de la bibliothèque Pocket sphinx. Ensuite nous présentons un aperçu visuel sur les différentes interfaces de notre application.

### Présentation de l’environnement de développement :

#### Environnement matérielle :

Nous avons réalisé ce travail sur un pc portable HP dont la configuration est la suivante :

* + - Processeur Intel® core™ i7-8650U
    - CPU 1.90 GHz, RAM 16Gb
    - Système d’exploitation : Windows 10 professionnel

Pour le teste de l’application nous avons utilisé un appareil physique Samsung Galaxy A51, 8Go de mémoire vive, Système Android 11.

#### Environnement Logiciel :

###### Android studio :

Android studio [Site-1] est un environnement de développement spécialisé dans le développement d'applications Android. Il est basé sur IntelliJ IDEA, qui est un outil pour gérer l'installation du SDK Android sur le système.

###### Android SDK :

Android SDK (Software Development Kit) [Site-1] est un ensemble d’outils de développement. Il inclut un débogueur, des bibliothèques logicielles, un émulateur, de la documentation, des exemples de code et des tutoriels, que Google a met à la disposition des développeurs pour créer et développer des applications Android.

###### C:\Users\DELL\Pictures\Documents\Bluetooth Folder\images.pngAVD Android :

AVD Android (Android Virtual Device) [Site-1] est un émulateur qui permet de simuler des appareils android sur un ordinateur afin de tester des applications sur une variété d'appareils et de niveaux d'API Android sans besoin d'avoir d’un dispositif physique.

Comme ce dernier étant notoirement lents à démarrer et à s'exécuter, cela a poussé les développeurs sur Android studio à chercher d’autres émulateurs plus rapide comme : GenyMotien que nous avons utilisé.

###### Genymotion :

Genymotion [Site-2] est un émulateur Android rapide, simple à installer et puissant grâce à des widgets de capteurs conviviaux et des fonctionnalités d'interaction.Il permet de tester des applications Android sur une large gamme d’appareils virtuels.

#### Langages de développement :

###### Java :

Java est un langage de programmation orienté objet,créé par J. Gosling et P. Naughton présenté officiellement le 23 mai 1995 au SunWorld .C’est est à la base de la plupart des applications en réseau et il est

exploitée dans le monde entier, il permet de développer des applications portables de hautes performances sur une large gamme de plates-formes informatiques. [Site-3] [Site-4]

###### XML :

LelangageXML **(**eXtensible Markup Language) est un langage de balisage extensible qui définit un ensemble de règles pour l'encodage des documents dans un format lisible à la fois

par l'homme et par la machine. Il permet de décrire, stocker ou transférer des donnéesd’un programme à l’autre d’une façon structurée*.* [Site-5] [Site-6]

### Implémentation de notre système de reconnaissance vocale sous Android :

Afin d’exécuter notre système de reconnaissance vocale sous Android, nous avons utilisé la bibliothèque Pocket Sphinx qui permet de facilité l'utilisation de la fonctionnalité de reconnaissance vocale dans notre application.

#### Pocket sphinx :

Pocket Sphinx est un outil qui peut être intégré comme une bibliothèque dans un projet Android. Il permet de lire des données audio dans des blocs de mémoire à partir d'une source et de les transmettre au décodeur. [Site- 7]

Pour faciliter le traitement des modèles acoustiques des fonctions et des méthodes ont conçues pour ressembler au même flux de travail utilisé dans Pocket Sphinx. Le tableau 3.1 ci-dessous résume les principales fonctions proposées par cette bibliothèque :

**Tableau 3-1 Les principales méthodes de Pocket Sphinx.**

|  |  |
| --- | --- |
| **Les méthodes** | **Utilisation** |
| OnError() | Déclenchée en cas d’erreur pour le traiter ou quitter  l’application. |
| StartListning() | Appelée en cas ou le système à l’état prêt. |
| OnTimeOut() | Déclenchée lorsque le délai accordé à l’attente est  dépassé. |
| OnBeginingOfSpeech() | Effectuer l’enregistrement d’un signal de la parole. |
| OnEndOfSpeech() | Lancée lorsque l’utilisateur à arrêter de parler. |
| OnResult() | Déclencher une fois les résultats de reconnaissance  d’enregistrement d’entrée est prêts. |

### Le déroulement et la présentation visuelle de notre application « digital school » :

Notre application « digital school » est une application de la gestion d’un établissement scolaire qui permet de garantir l’interaction entre les acteurs principaux qui sont : l’administration, les étudiants et les enseignants. On a décomposé notre application en deux modules : l’administration et l’autre commune entre les étudiants et les enseignants.

Tout d’abord les étudiants et les enseignants doivent s’inscrire à l’application, ils devront remplie toutes leurs information en cliquant sur sign up pourqu’ils puissent se connecter et accéder à leurs interfaces. L’interface principale de l’inscription est montrée dans la figure suivante.



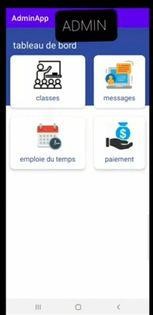
###### Figure 4-1 interface de l’inscription.

Une fois l’étudiant ou l’enseignant est inscrit, il sera enregistré dans la base de données, et s’il veut se connecter la prochaine il doit taper leur pseudo et leur mot de passe puis il peut accéder à leur interface

#### L’interface de l’administration :

L’administrateur a généralement quatre taches principales :

* + - Envoyer des messages aux étudiants ou aux enseignants.
    - Affecter des classes aux étudiants ou aux enseignants.
    - Envoyer les emplois du temps aux profs de même pour les étudiants.
    - Définir la date de paiement des profs aussi pour les étudiants.



###### Figure 4-2interface principale de l’administrateur.

.

Chacune de ces taches sont visualisées dans les figures suivantes :



###### Figure 4-3 l’ajout des classes aux étudiants et aux enseignants.

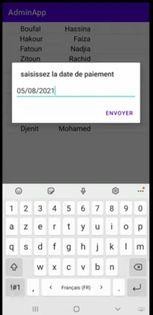


**Figure 4-4 l’ajout des classes aux étudiants et aux enseignants.**



###### Figure 4-5 l’envoi d’un message aux étudiants et aux enseignants.

L’administrateur pourra envoyer des messages aux étudiants et aux professeurs en cliquant sur l’étudiant ou l’enseignant qu’on veut lui envoyer le message et le message sera envoyer automatiquement.



###### Figure 4-6 l’envoi de la date de paiement aux étudiants et aux enseignants.

En précisant que pour l’étudiant sa sera la date pour procédé au paiement, mais pour les enseignants sa sera la date de récupération de leurs pailles.



**Figure 4-7 l’envoi de l’emploi du temps aux étudiants et aux enseignants.**

#### L’interface de l’enseignant :

L’enseignant Il doit d’abord écrit son pseudo et son mot de passe pour qu’il puisse se connecter et accéder à leur interface principale qui est visualisé comme suit :



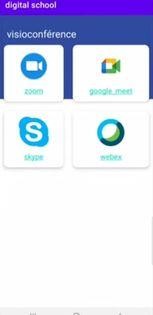
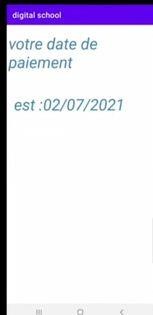
###### Figure 4-8 l’interface principale de l’enseignant.

Si l’enseignant clique sur classes, la liste de ces classes sera affichéeet il peut choisir une de ces dernières pour qu’il puisse envoyer à leurs élèves de cette classe choisie des devoirs à faire, des cours PDF ou vidéo, recevoir

des questions par les étudiants, il peut aussi saisir et envoyer leurs notes par la reconnaissance vocale. Ces taches sont présentées visuellement par la figure ci-dessous :



###### Figure 4-9 la tache de la saisie vocale des notes.



**Figure 4-10 la suite des autres taches de l’enseignant.**

#### L’interface de l’étudiant :

L’étudiant est presque comme l’enseignant, il doit remplie son pseudo et le mot de passe pour qu’il puisse connecter et accéder à leur interface principale qui est présenté dans la figure suivante.



###### Figure 4-11 l’interface de l’étudiant.

De même l’étudiant peut recevoir des notifications ajoutées par l’administrateur, voire la date de paiement et il peut donner une question de n’importe quel cours. Ceci est montré visuellement comme suit.



**Figure 4-12 visualisation de l’interface de l’étudiant (suite).**

### Conclusion :

Au long de ce chapitre, nous allons décrit la phase de réalisation et d’implémentation de notre application en présentant tout d’abord l’environnement de développement ainsi que l’implémentation de notre système de reconnaissance vocale avec l’utilisation de la bibliothèque Pocketsphinx. Enfin nous avons clôturé ce chapitre par une visualisation et explication générale de toute l’application que nous avons réalisée.

# Conclusion générale

##### . Au long de cette étude que nous avons fait, notre travail consiste à concevoir et à développer une application Android pour la gestion d’un établissement scolaire, doté d’un système de reconnaissance de la parole, pour l’optimisation de la saisie de notes par les enseignants. Cette application permet de garantir une interaction permanente entre trios acteurs principaux qui sont: l’administration, les enseignants et les étudiants par exemple: les enseignants peuvent voir les notifications ajoutés par l’administrateur, répondre aux questions des élèves, et les élèves auront accès à tout moment à ces informations (devoirs à faire, bulletins de notes etc.).

Dans le but de réaliser notre projet, nous avons présenté d’une manière générale le domaine de la reconnaissance vocale qui est basé sur les modèles de Markov cachés (HMM). En détaillant ces trois principaux modules : l’extraction de paramètre, la modélisation acoustique et le module d’identification.

##### Nous avons présenté par la suite, la conception de notre système de reconnaissance automatique de la parole à base de la plateforme CMU Sphinx. Nous avons commencé tout d’abord par la préparation des données acoustiques, ensuite nous avons préparé l’ensemble de données linguistique comme la liste des phonèmes, la transcription textuelle, le dictionnaire phonétique, etc. Une le système de reconnaissance est conçu, nous effectuons une série d’expériences afin de sélectionner les meilleurs résultats qui s’adaptent avec notre application.

.Enfin nous allons procéder à la phase de développement et d’implémentation de notre application, ou nous avons présenté l’environnement de développement, basé sur les langages JAVA, XML et Android. Pour finir nous avons clôturé notre étude par le déroulement et la présentation visuelle de quelque interface de notre application « digital school ».

Dans les perspectives, nous projetons d’étendre l’application de reconnaissance vocale pour un large vocabulaire.

**BIBLIOGRAPHIE**

**[Amrous-2015]** Anissa Imen AMROUS, «*Coopération de Connaissances dans les systèmes de Reconnaissance Automatique de la Parole*. Thèse de doctorat, 2015 Universitédes Sciences et de la Technologie Houari Boumediene, Algérie.

**[Rabiner-1993]** L.R. RABINER and B.H. Juan, « Fundamentals of speech recognition». Englewood Cliffs, N.J., USA: Prentice Hall, 1993.

**[FOHR-2006]** D. FOHR, C. CERISARA, J.P. HATON, Y. LAPRIE, and K. SMAILI, « Reconnaissance

automatique de la parole: du signal à son interprétation », Paris: Dunod Edition, 2006.

**[Harris-1978]** F.J. HARRIS, « On the use of windows for harmonic analysis with the discrete Fourier transform », in Proc. IEEE, vol. 66 (1), Jan 1978, pp. 51-83

**[Hermansky-90]** H. HERMANSKY, « *Perceptual Linear Predictive (PLP) analysis of speech* » Acoustic Society Am, vol. 87, no. 4, pp. 1738-1752, April 1990.

**[Debyeche-07]** M. DEBYECHE, « *Reconnaissance automatique de la parole appliqué à la langue arabe* », Thèse de doctorat d’etat, 2007, Université des Sciences et dela Technologie Houari Boumediene (USTHB), Alger, Algérie.

**[Davis-80]** Steven B DAVIS and Paul MERMELSTEIN « *IEEE Transactions on Acoustics, Speech, and Signal Processing* », volume 28, chapter Comparison of parametric representations forMonosyllabic word recognition in continuously spoken sentences, page 357–366. Août 1980.

**[Jankovski-95]** C.R. JANKOVSKI, H.D. Vo, and R.P. Lipmann, «*A comparison of signal processing frontends for automatic word recognition* », IEEE trans. on Speech and Audio Processing, vol. 3, no. 4, pp. 286-293, July 1995.

**[Vapnik- 1998] :**Statistical Learning Theory, Wiley, ohn Wiley & Sons Inc (Verlag), ISBN 978-0-471- 03003-4.

**[Rabiner-1989]:** A tutorial on hidden markov models and selected applications in speech recognition, Proceedings of the IEEE, vol. 77, no 2, p. 257–286.

**[Jelinek-1976]:** Continuous speech recognition by statistical methods, Proceedings of the IEEE, vol. 64, no 4, p. 532–556.

**[Baker-1975 ] :**« The DRAGON system - An overview », Acoustics, Speech and Signal Processing, IEEE Transactions on Acoustics, vol. 23, no 1, p. 24–29.

**[Barrlaut-08]** L.BARRLAUT : « *Diagnostic pour la combinaison de systèmes de reconnaissance automatique de la parole »*. Thèse de doctorat, 2008, Université d’Avignon,France. tel-00424699.

**[Varela-2003]** A. Varela, H. Cuayáhuitl and J.A. Nolazco-Flores ( Creating a Mexican Spanish Version of the CMU Sphinx-III Speech Recognition System ) Springer , Vol. 2905, 2003.

## Webographie

**[Site-filtrage]**https://[www.flegrand.fr/scidoc/docmml/sciphys/tpelectro/filtrebruit/filtrebruit.html.](http://www.flegrand.fr/scidoc/docmml/sciphys/tpelectro/filtrebruit/filtrebruit.html) **[Site-0]**http ://[www.activestate.com.](http://www.activestate.com/)

**[Site-1]** https://developer.android.com/studio/intro/index.html.

**[Site-2]**https://docs.genymotion.com.

**[Site-3]** https://fr.wikipedia.org/wiki/Java\_(langage).

**[Site-4]** https://[www.java.com/fr/about/.](http://www.java.com/fr/about/)

**[Site-5]**https://fr.wikipedia.org/wiki/Extensible\_Markup\_Language. **[Site-6]**https://[www.w3schools.com/xml/.](http://www.w3schools.com/xml/)

**[Site-7]**[http://www.speech.cs.cmu.edu/sphinxman/.](http://www.speech.cs.cmu.edu/sphinxman/)

**Liste des abréviations**

**AVD :** Android Virtual Device **CMU :** Carnegie Mellon University **DCT :** Discret Cosinus Transform

**HMM :** Hidden Markov Model Toolkit

**JSGF:** JSpeech Grammar Format

**MFCC:** Mel Frequency Cepstral Coefficients **RAP:** Reconnaissance Automatique de la Parole **XML :** Software Development Kit

**SDK :** eXtensible Markup Language