

SI Project

Walid Tolba Manel Abdelaziz

January 12, 2023

Chapter 1

Introduction

De nos jours, le progres de la civilisation est base en grande partie sur la science de l'informatique. L'intelligence artificielle est une discipline d'une grande ampleur, qui implique l'introduction des machines programmees avec un system intelligent afin d'arriver a reproduire certains aspects d'intelligence humaine en suivant des methodes developpees et bien definies tel que L'apprentissage automatique, qui est le fait de donner une capacite d'apprentissage a la machine sans la programmer d'une maniere explicite.

1.1 L'apprentissage Automatique

L'apprentissage automatique se different selon le type des donnees d'entree a traiter par la machine , on distingue plusieurs applications de l'apprentissage automatique telles que, la reconnaissance des images, la reconnaissance du texte, moteur de recherche et langage naturel ...etc

1.2 La Reconnaissance De La Parole

La Reconnaissance de la parole est une form d'apprentissage automatique que se base sur l'application des techniques et methodes d'informatique, elle permet a la machine de traiter et de comprendre des donnees fournies oralement par un utilisateur humain dans le but d'interagir de facon naturelle et efficace avec l'Homme.

1.3 Domaine d'applications

Les avantages de la reconnaissance de la parole sont multiples, nous représentons quelques domaines d'application:

1.3.1 Le domaine medical

La Reconnaissance De parole est utilisée dans le domaine médical grâce à ses fonctionnalités qui servent à aider les personnes handicapées.

1.3.2 La commande vocale

- assistance mobile
- commande vocale en robotique.
- appareillage pour handicapés moteurs.

1.3.3 La Saisie De Données Par la voie

pour remplissage de formulaires, passage de commandes commerciales ou autre utilisation.

1.4 Exemples d'assistants vocaux

les applications les plus connues qui se basent sur la reconnaissance de la parole:

- Duolingo
- Memrise
- Alexa
- Babbel
- Elsa

Chapter 2

Consideratoin pratique pour la realisation du projet

2.1 Les outils necessaire

- Pocketsphinx: bibliotheque legere de reconnaissance ecrite en C. - CMU-clmtk: outils pour les modeles de langage - Sphinxtrain: outil d'entrainement de modeles acoustiques.

2.2 Preparation d'installation

Pour la construction du modele acoustique, nous aurons besoin d'effectuer quelques instructions que sont:

- Mettre a jour les gestionnaire des paques avec la commande: `sudo apt-get update`.
- Installation des outiles necessaire avec la commande: `sudo apt-get install build-essential cmake python3`
- Telechargement des outiles:
 - Pocketsphinx
 - Sphinxtrain
 - CmuLMTK

- Installation de Pocketsphinx et Sphinxtrain avec l'utilisation de cmake.
- Installation de CMULMTK. - verification d'instalation. remarque: nous utilisons une distrubition linux base sur debian.

2.3 Collection de la base de donnees audio

Pour la collection de la base de donnee audio, nous avons utilise une application sous android, que s'appelle "Enregistreur vocal", sous quelques conditions:

- des echantillons devraient etre pris dans un endroit calme sans bruit.
- le format convenable des enregistrements c'est bien "wav" avec frequence d'echantillonnage specifique 16 kHz mono.

Nous avons consacre 30 memos vocaux pour chaque mot et chaque phrase, ainsi nous avons essaye d'assurer une prononciation qui se distinct d'une personne a une autre.

Apres, on ajoute les nouveaux mots dans le fichier an4.dic, la transcription dans an4-train.transcription et les noms des fichiers dans an4-train.fileids.

Apres, on choisit quelques mot pour faire le test, on les rajoute dans les deux fichiers an4-test.fileids et an4-test.transcription.

2.4 Les etapes pour la creation de modele acoustique arabe

Les etaps sont:

- creation de modele de langage avec un script.
- creation de dossier an4 contient deux sous dossiers: wav pour les memos vocaux et etc pour les fichiers nessecaire.
- les fichiers nessecaire sont: an4-train.transcription, an4-train.fileids, an4-test.transcription, an4-test.fileids, an4.dic, an4.fillter, an4.phone et le modele de langage an4.lm.DMP.
- ouvrir un terminal dans le dossier an4.
- execution la commande sphinxtrain -t an4 setup, pout le creation d'un fichier de configuratoin.
- modifier le fichier de configuration, apres execute la commande: sphinxtrain run.
- teste avec: sphinxtrain -s decode run.

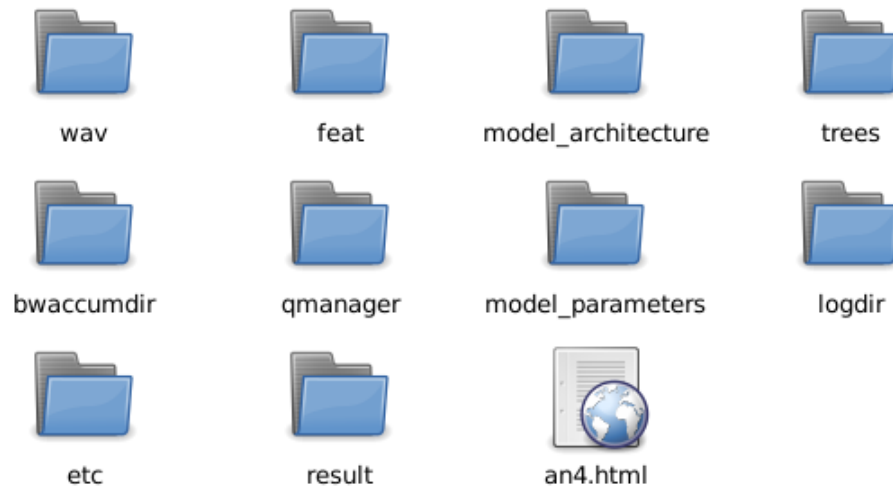


Figure 2.1: Les fichiers necessaire

```
tolba@debian: ~/Desktop/SI_Project/Workspace/wmdb/an4
File Edit View Search Terminal Help
tolba@debian:~/Desktop/SI_Project/Workspace/wmdb/an4$ shpinxtrain -t an4 setup
bash: shpinxtrain: command not found
tolba@debian:~/Desktop/SI_Project/Workspace/wmdb/an4$ sphinxtrain -t an4 setup
Sphinxtrain path: /usr/local/share/sphinxtrain
Sphinxtrain binaries path: /usr/local/libexec/sphinxtrain
Setting up the database an4
tolba@debian:~/Desktop/SI_Project/Workspace/wmdb/an4$ ~
```

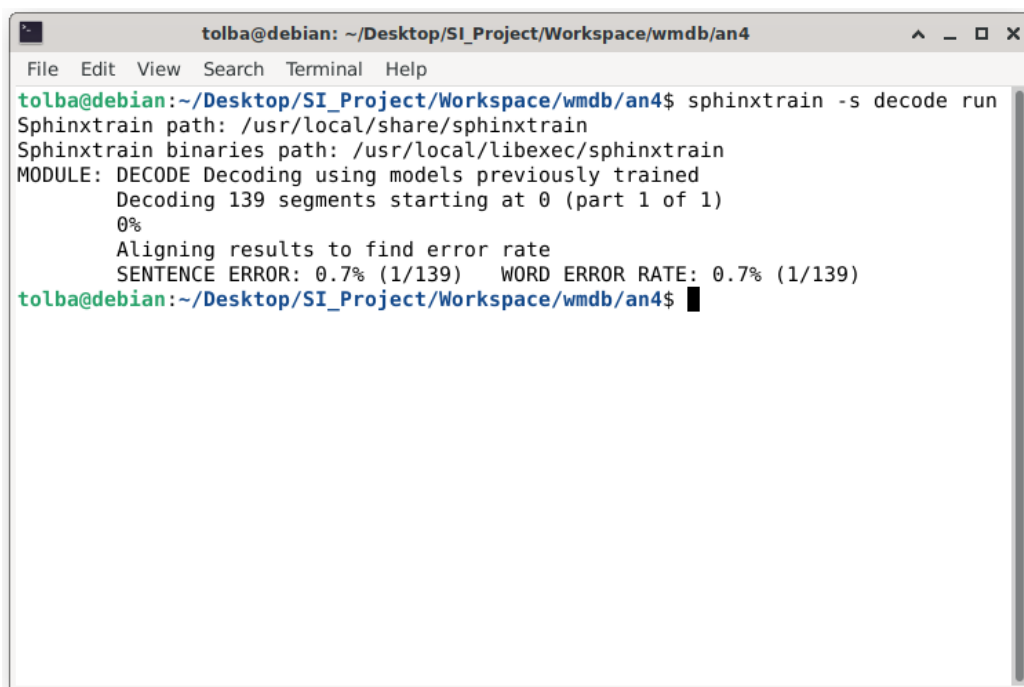
Figure 2.2: La preparation de la base de donnee

```
tolba@debian: ~/Desktop/SI_Project/Workspace/wmdb/an4
File Edit View Search Terminal Help
tolba@debian:~/Desktop/SI_Project/Workspace/wmdb/an4$ shpinxtrain -t an4 setup
bash: shpinxtrain: command not found
tolba@debian:~/Desktop/SI_Project/Workspace/wmdb/an4$ sphinxtrain -t an4 setup
Sphinxtrain path: /usr/local/share/sphinxtrain
Sphinxtrain binaries path: /usr/local/libexec/sphinxtrain
Setting up the database an4
tolba@debian:~/Desktop/SI_Project/Workspace/wmdb/an4$ sphinxrun
bash: sphinxrun: command not found
tolba@debian:~/Desktop/SI_Project/Workspace/wmdb/an4$ sphinxrun
bash: sphinxrun: command not found
tolba@debian:~/Desktop/SI_Project/Workspace/wmdb/an4$ sphinxtrain run
Sphinxtrain path: /usr/local/share/sphinxtrain
Sphinxtrain binaries path: /usr/local/libexec/sphinxtrain
Running the training
MODULE: 000 Computing feature from audio files
Extracting features from segments starting at (part 1 of 1)
```

Figure 2.3: Le debut de l'entrainement

```
tolba@debian: ~/Desktop/SI_Project/Workspace/wmdb/an4
File Edit View Search Terminal Help
Skipped for continuous models
MODULE: 10 Training Context Independent models for forced alignment and VTLN
Skipped: $ST::CFG_FORCEDALIGN set to 'no' in sphinx_train.cfg
Skipped: $ST::CFG_VTLN set to 'no' in sphinx_train.cfg
MODULE: 11 Force-aligning transcripts
Skipped: $ST::CFG_FORCEDALIGN set to 'no' in sphinx_train.cfg
MODULE: 12 Force-aligning data for VTLN
Skipped: $ST::CFG_VTLN set to 'no' in sphinx_train.cfg
MODULE: 20 Training Context Independent models
Phase 1: Cleaning up directories:
    accumulator...logs...qmanager...models...
Phase 2: Flat initialize
Phase 3: Forward-Backward
    Baum welch starting for 1 Gaussian(s), iteration: 1 (1 of 1)
    0% 10% 20% 30% 40% 50% 60% 70% 80% 90% 100%
    Normalization for iteration: 1
    Current Overall Likelihood Per Frame = -146.545372478358
    Baum welch starting for 1 Gaussian(s), iteration: 2 (1 of 1)
    0% 10% 20% 30% 40% 50% 60% 70% 80% 90% 100%
    Normalization for iteration: 2
    Current Overall Likelihood Per Frame = -143.605611362271
    Convergence Ratio = 2.93976111608669
    Baum welch starting for 1 Gaussian(s), iteration: 3 (1 of 1)
    0% 10% 20% 30%
```

Figure 2.4: L⁶entrainement

A terminal window titled 'tolba@debian: ~/Desktop/SI_Project/Workspace/wmdb/an4'. The window contains the following text: 'tolba@debian:~/Desktop/SI_Project/Workspace/wmdb/an4\$ sphinxtrain -s decode run', 'Sphinxtrain path: /usr/local/share/sphinxtrain', 'Sphinxtrain binaries path: /usr/local/libexec/sphinxtrain', 'MODULE: DECODE Decoding using models previously trained', 'Decoding 139 segments starting at 0 (part 1 of 1)', '0%', 'Aligning results to find error rate', 'SENTENCE ERROR: 0.7% (1/139) WORD ERROR RATE: 0.7% (1/139)', and 'tolba@debian:~/Desktop/SI_Project/Workspace/wmdb/an4\$'.

```
tolba@debian: ~/Desktop/SI_Project/Workspace/wmdb/an4
File Edit View Search Terminal Help
tolba@debian:~/Desktop/SI_Project/Workspace/wmdb/an4$ sphinxtrain -s decode run
Sphinxtrain path: /usr/local/share/sphinxtrain
Sphinxtrain binaries path: /usr/local/libexec/sphinxtrain
MODULE: DECODE Decoding using models previously trained
Decoding 139 segments starting at 0 (part 1 of 1)
0%
Aligning results to find error rate
SENTENCE ERROR: 0.7% (1/139) WORD ERROR RATE: 0.7% (1/139)
tolba@debian:~/Desktop/SI_Project/Workspace/wmdb/an4$
```

Figure 2.5: La fin de L'entrainement

Chapter 3

Implementation

les outiles que nous utilisons sont:

- Python: un langage de programmation haut niveau.
- SpeechRecognition: un bibliotheque de reconnaissance de la parole.
- PocketSphinx: un bibliotheque de reconnaissance de la parole.
- Tkinter: un bibliotheque de GUI.
- CustomTkinter: un bibliotheque modern de GUI base sur Tkinter.
- SMTPLib: un bibliotheque de messagerie.
- VLC-python: un bibliotheque pour le multimedia.
- Debian: system d'exploitation.

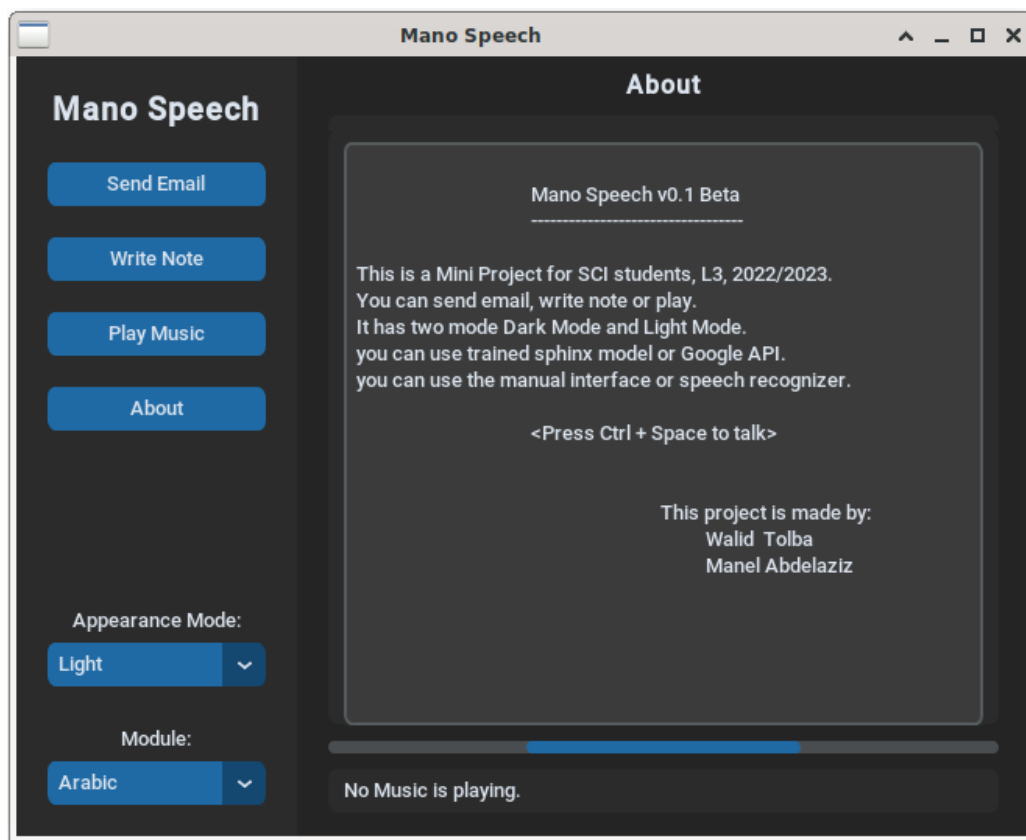


Figure 3.1: en Dark mode

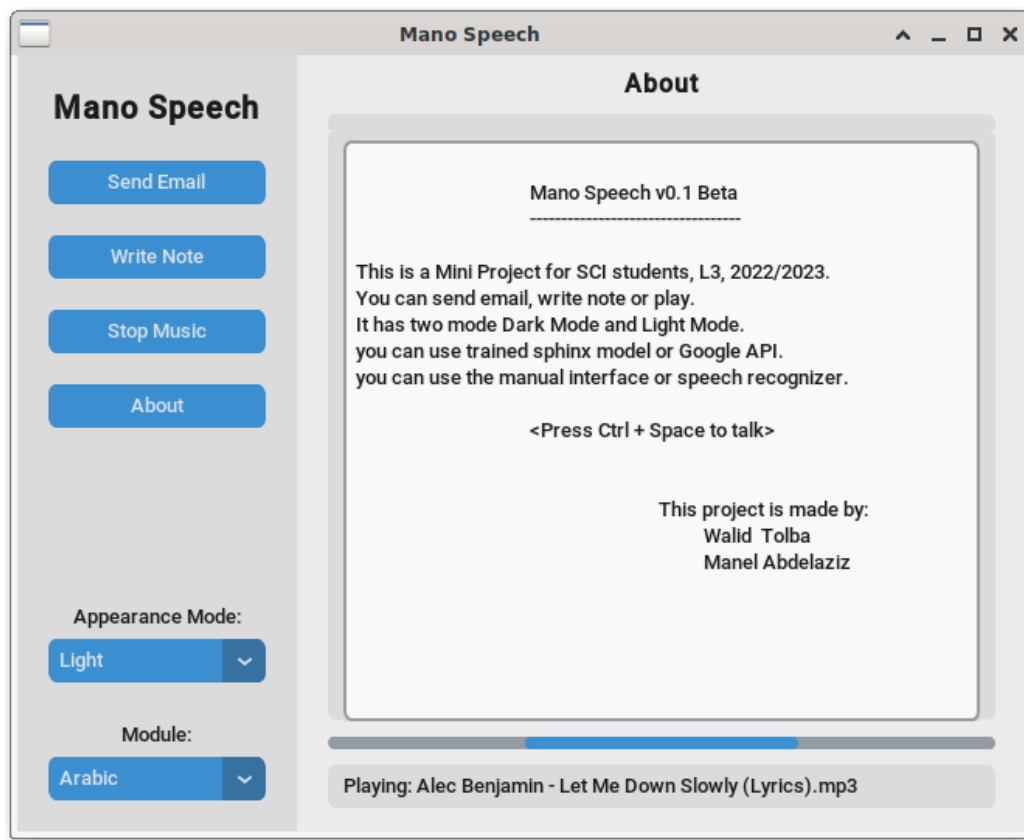


Figure 3.2: en Light mode

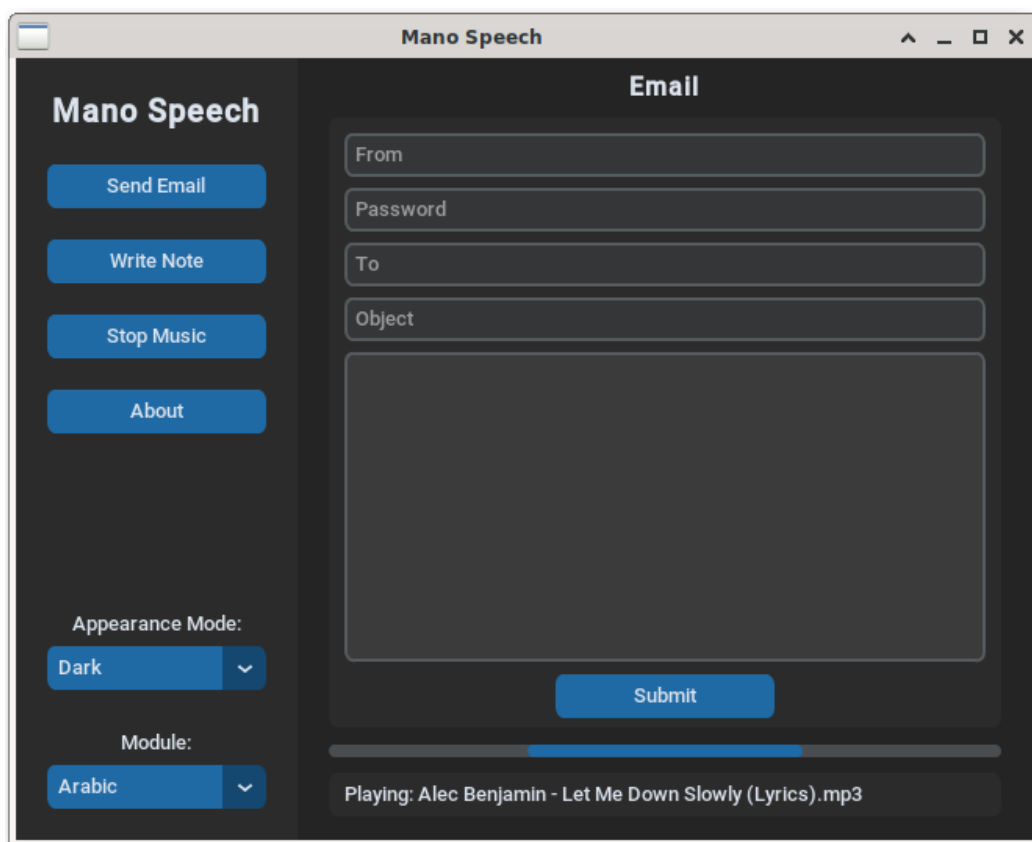


Figure 3.3: send Email

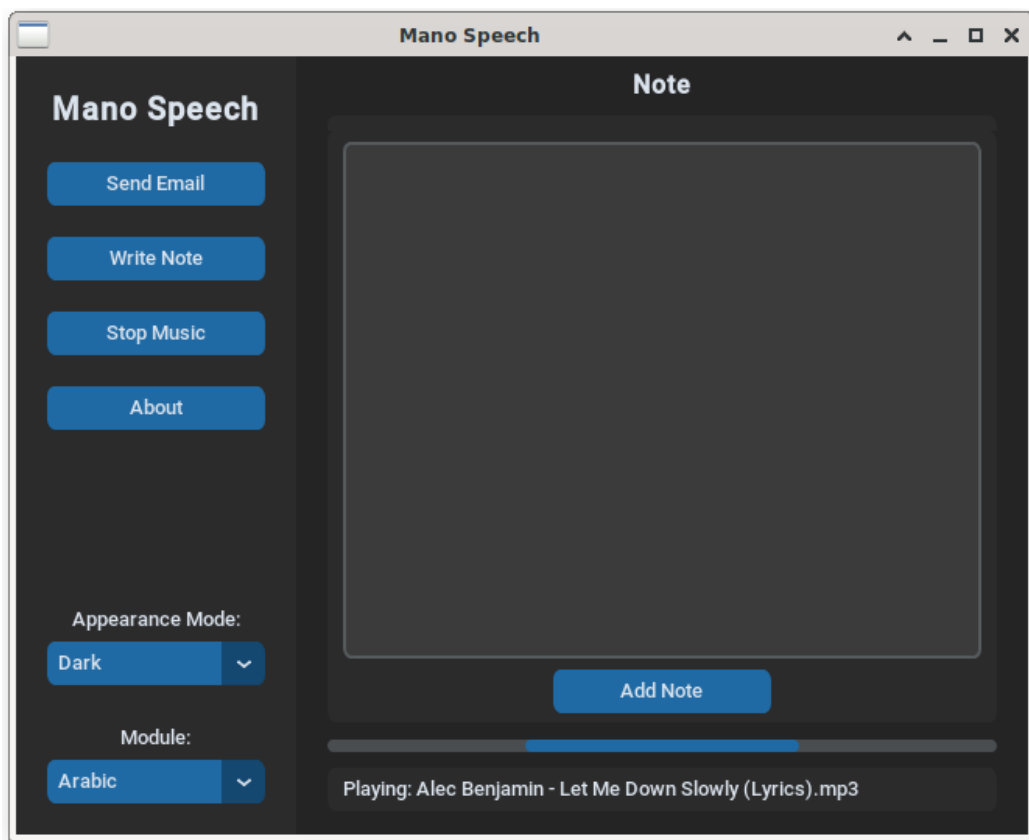


Figure 3.4: write note

Chapter 4

Performance

4.1 Mesurer les performances en temp de VP, VN, FP, FN et SA

1. Vrai Positif VP: represente les mots qui ont ete bien prononees et acceptes.
2. Vrai Negatif VN: represente les mots qui ont ete mal prononees et acceptes.
3. Faux Positif FP: represente les mots qui ont ete mal prononees et rejetes.
4. Faux Negatif FN: represente les mots qui ont ete bien prononees et rejetes.
5. Le score de precision: represente la precision et il se calcule avec cette formule:

$$SA = \frac{VP + VN}{VP + VN + FP + FN} * 100 \quad (4.1)$$

Performance	Percentage
VP	90
VN	40
FP	10
FN	60
SA	75