Rapport du Projet de RECONNAISSANCE VOCALE

Membres de groupe:

- HEIDARI Shahin

Encadré par:

Joseph Razik

Année universitaire : 2023 - 2024

SOMMAIRE

1. INTRODUCTION

Contexte du Projet

Le projet est structuré en plusieurs phases, y compris l'apprentissage, le développement, la mise au point des paramètres et l'évaluation finale. Nous effectuerons des tests sur des corpus indépendants pour mesurer la précision de notre système. Si nécessaire, nous utiliserons une validation croisée pour augmenter la robustesse de nos résultats.
Le projet de reconnaissance vocale a été entrepris pour répondre à la demande croissante de systèmes automatisés capables de transcrire des fichiers audio en texte. Cette technologie trouve des applications dans divers domaines tels que la transcription d'enregistrements, la commande vocale, etc.
Dans l'ensemble, ce mini-projet offre une opportunité passionnante d'appliquer des concepts théoriques à des problèmes du monde réel dans le domaine de la reconnaissance vocale. Nous espérons que notre travail contribuera à la compréhension et à l'amélioration des systèmes de reconnaissance vocale pour des applications diverses et variées.
Pour atteindre cet objectif, nous travaillerons avec un ensemble de données provenant du projet open-source Common Voice de Mozilla, qui comprend des fichiers audio représentant une diversité de locuteurs et de contextes linguistiques. Notre travail impliquera la manipulation de données audio, l'extraction de caractéristiques pertinentes, la modélisation statistique et la mise en œuvre d'un système de reconnaissance.

Objectifs

Les objectifs principaux du projet étaient les suivants :
- Développer un système de reconnaissance vocale fonctionnel.
- Atteindre une précision élevée dans la conversion des fichiers audio en transcriptions textuelles.
- Explorer les différentes étapes du processus, de la conversion initiale à l'évaluation des résultats.

Config

# Indique que les fichiers sont au format brut sans en-tête
SOURCEFORMAT = NOHEAD
# Spécifie le taux d'échantillonnage de la forme d'onde en entrée (208.33 Hz dans ce cas)
SOURCERATE = 208.33333333
# Spécifie le type de caractéristiques cibles
TARGETKIND = MFCC_0
# Spécifie la fréquence d'images pour les caractéristiques de sortie (fréquence d'image de 10 ms)
TARGETRATE = 100000.0
# Spécifie la taille de la fenêtre utilisée pour l'extraction des caractéristiques (250000.0 échantillons)
WINDOWSIZE = 250000.0
# Spécifie si les caractéristiques doivent être enregistrées dans un format compressé
SAVECOMPRESSED = F
# Spécifie si les caractéristiques doivent être enregistrées avec une vérification CRC (Cyclic Redundancy Check)

SAVEWITHCRC = F
# Spécifie l'ordre d'écriture naturelle des éléments
NATURALWRITEORDER = T
# Spécifie l'ordre de lecture naturelle des éléments
 NATURALREADORDER = T
# Spécifie si une fenêtre de Hamming doit être utilisée pour l'extraction des caractéristiques
 USEHAMMING = T
# Spécifie le coefficient de pré-émphase
 PREEMCOEF = 0.97
# Spécifie le nombre de canaux de filtre dans la banque de filtres
 NUMCHANS = 26
# Spécifie le coefficient de liftering utilisé dans le liftering cepstral
CEPLIFTER = 22
# Spécifie le nombre de coefficients cepstraux à extraire
NUMCEPS = 12

2. CONVERSION DES FICHIERS AUDIO

Utilisation du Script `testConversation.sh`

Le processus de conversion des fichiers audio a été initié par l'exécution du script `testConversation.sh`. Ce script, exécuté dans un environnement Bash, a automatisé plusieurs étapes cruciales.

```
#!/bin/bash
# Déplacement dans le dossier du fichier files.tsv
cd ../I322/DATA
```

Le script commence par se déplacer dans le dossier contenant le fichier de référence files.tsv. Cette étape est cruciale pour s'assurer que le script opère dans le bon contexte.

```
# Création du dossier pour les fichiers après conversion
echo "Création des dossiers pour les fichiers .raw..."
mkdir RAW_DATA_TEST
mkdir RAW_DATA_DEV
```

Le script crée deux dossiers, RAW_DATA_TEST et RAW_DATA_DEV, qui serviront à stocker les fichiers audio convertis. Cette organisation facilite la gestion des fichiers et la séparation des ensembles de test et de développement.

```
# Fait la conversion des .mp3 en .raw et les déplace dans le dossier DATA
echo "Conversion des fichiers en cours... (peut prendre un moment)"

# Récupère le nombre de fichier dans files.tsv pour pouvoir séparer le
test du dev
nbFichier=($(wc -l < files.tsv))
nbTest=$(( $nbFichier*70/100 ))

# Boucle en fonction du nombre de fichier dans files.tsv
```

```
# Change le dossier de destination après 70% du nombre de fichier pour
séparer le test du dev
cpt=0
while read line
do
    echo $cpt
    if [ $cpt -lt $nbTest ]; then
        sox -v 0.9 "clips/"$line".mp3" "RAW_DATA_TEST/"$line".raw"
    else
        sox -v 0.9 "clips/"$line".mp3" "RAW_DATA_DEV/"$line".raw"
    fi
    let cpt++
done < <(awk '{print $2}' files.tsv | sed '1d')
```

Cette partie du script effectue la conversion des fichiers audio. La boucle while parcourt chaque ligne de files.tsv pour récupérer le nom du fichier. En fonction de la position dans la boucle, le fichier converti est déplacé vers le dossier RAW_DATA_TEST ou RAW_DATA_DEV. La commande sox est utilisée pour effectuer la conversion des fichiers de format MP3 à RAW, avec une réduction de volume de 10%.

Déplacement et Organisation des Fichiers

Le script a débuté en se déplaçant vers le dossier contenant le fichier `files.tsv`. Ce fichier a été utilisé comme référence pour les noms des fichiers à convertir. Une structure de dossiers a ensuite été créée pour organiser les fichiers convertis.

Conversion des MP3 en RAW

L'étape suivante a impliqué la conversion effective des fichiers audio au format MP3 en fichiers audio RAW. La commande `sox` a été utilisée pour effectuer cette conversion, avec une réduction de volume de 10% (`-v 0.9`) pour éviter les distorsions.

Séparation des Ensembles de Test et de Développement

Une caractéristique intéressante de ce script était la séparation des fichiers en ensembles de test et de développement. Ceci a été réalisé en fonction du nombre total de fichiers, où 70% ont été alloués au test et le reste au développement.

3. CRÉATION DES MFCC

Utilisation du Script `mfcc.sh`

La création des Mel-frequency Cepstral Coefficients (MFCC) était une étape cruciale dans le processus de reconnaissance vocale. Ces coefficients représentent les caractéristiques acoustiques essentielles des fichiers audio.

La création des coefficients cepstraux de fréquence Mel (MFCC) est une étape cruciale dans le processus de reconnaissance vocale. Cette section se divise en deux parties distinctes :

Création des fichiers `.scp` pour les données d'apprentissage et de test :

Deux fichiers, `codetr.scp` et `codetest.scp`, sont créés pour stocker les chemins des fichiers bruts et les chemins des fichiers MFCC résultants. Ces fichiers servent d'entrée pour l'outil HCopy de HTK, qui effectuera la conversion des fichiers bruts en fichiers MFCC.

Pour l'ensemble d'apprentissage (`codetr.scp`), le script parcourt les fichiers bruts dans le dossier `RAW_DATA_DEV/`, extrait le nom du fichier sans extension, et écrit le chemin du fichier brut ainsi que le chemin du fichier MFCC résultant dans le fichier `.scp`. La même opération est répétée pour l'ensemble de test (`codetest.scp`) avec les fichiers dans le dossier `RAW_DATA_TEST/`.

Boucle de traitement pour le calcul des MFCC :

Le script utilise une boucle for pour parcourir tous les fichiers RAW dans les dossiers RAW_DATA_TEST et RAW_DATA_DEV. À chaque itération, un fichier spécifique est traité.

1. **Extraction du nom de fichier :**
   - La première étape consiste à extraire le nom du fichier sans son extension. Cela est effectué avec les commandes basename et une manipulation de chaîne pour obtenir le nom de base du fichier.

2. **Affichage du fichier en cours de traitement :**
   - Un message est affiché pour informer l'utilisateur du fichier en cours de traitement, permettant ainsi de suivre les progrès du script.

3. **Conversion du fichier RAW en format WAV :**
   - La commande sox est utilisée pour convertir le fichier RAW en format WAV. Le fichier WAV résultant est ensuite passé à l'exécutable compute-mfcc-feats.

4. **Calcul des MFCC :**
   - L'exécutable compute-mfcc-feats est responsable du calcul des MFCC à partir du fichier WAV.
   - Les résultats sont redirigés vers un fichier dans le dossier MFCC_FEATURES avec le même nom de base que le fichier d'origine, mais avec l'extension .mfcc.

Cette séquence d'opérations garantit la création efficace des MFCC pour chaque fichier audio du jeu de données, préparant ainsi les données pour les étapes ultérieures du processus de reconnaissance vocale.

Extraction des Mel-frequency Cepstral Coefficients

Le script `mfcc.sh` a utilisé la bibliothèque SoX pour extraire les MFCC des fichiers audio RAW générés précédemment. Cette étape a nécessité une compréhension approfondie des propriétés acoustiques des signaux audio.

Importance des MFCC dans la Reconnaissance Vocale

Les MFCC sont des éléments fondamentaux dans la reconnaissance vocale, car ils capturent les aspects essentiels du spectre audio. Leur utilisation permet d'identifier les caractéristiques distinctives du son, facilitant ainsi la modélisation ultérieure.

4. DÉFINITION DES MODÈLES

Utilisation du Script `modele_init.sh`

La définition des modèles constitue une étape cruciale dans la reconnaissance vocale. Le script `modele_init.sh` a été utilisé pour spécifier la structure des modèles acoustiques et linguistiques nécessaires au processus.

1. Création des fichiers `.scp` pour les données d'apprentissage et de test :
   - Deux fichiers, `train.scp` et `test.scp`, sont créés pour stocker les chemins des fichiers `.mfc` des enregistrements vocaux utilisés respectivement pour l'apprentissage et les tests. Ces fichiers sont essentiels pour définir la base de données d'entraînement et de test.

2. Génération des fichiers de transcription au format MLF (`mlf.mmf` et `mlf_test.mmf`) :
   - Les fichiers MLF (Master Label File) sont cruciaux pour associer chaque enregistrement vocal à sa transcription textuelle. Le script parcourt les enregistrements vocaux, extrait le nom du fichier (sans extension), et récupère les transcriptions correspondantes à partir du fichier `files.tsv`. Ces informations sont stockées dans les fichiers MLF associés à l'ensemble d'apprentissage (`mlf.mmf`) et à l'ensemble de test (`mlf_test.mmf`).

3. Déplacement dans le dossier des données d'apprentissage et estimation du modèle initial (`HMM0`) :
   - Le script se déplace dans le dossier contenant les données d'apprentissage (`RAW_DATA_DEV`) et crée un dossier `HMM0` qui servira à stocker le modèle initial.

4. Estimation du modèle initial avec HCompV :
   - L'outil HCompV du package HTK est utilisé pour estimer le modèle initial. Les options fournies incluent le fichier de configuration (`-C`), le taux de convergence (`-f`), le fichier MLF (`-l`), le fichier de liste des fichiers d'entraînement (`-S`), le répertoire de sortie (`-M`), et le prototype du modèle (`proto`).

5. Création des fichiers `macros` et `hmmdefs` :
   - Les fichiers `macros` et `hmmdefs` sont essentiels pour définir les paramètres du modèle. Le fichier `macros` contient des informations générales, tandis que `hmmdefs` contient les informations spécifiques aux modèles pour chaque phonème. Ces fichiers sont générés en fonction du prototype (`proto`) et de certains fichiers de configuration.

Chaque étape de ce script contribue à la mise en place initiale des modèles qui seront utilisés dans le processus de reconnaissance vocale.

Structure des Modèles Acoustiques et Linguistiques

Les modèles acoustiques décrivent la relation entre les caractéristiques acoustiques (telles que les MFCC) et les unités de son (phonèmes). Les modèles linguistiques, d'autre part, capturent les relations entre les phonèmes et les mots, facilitant ainsi la compréhension linguistique.

Rôle de Chaque Modèle dans le Processus

Les modèles acoustiques sont essentiels pour la correspondance entre les caractéristiques acoustiques et les phonèmes. Les modèles linguistiques guident la reconnaissance vocale en favorisant les séquences de mots les plus probables.

5. ESTIMATION DES PARAMÈTRES DES MODÈLES

Utilisation du Script `estimation_modeles.sh`

L'estimation des paramètres des modèles a été réalisée avec le script `estimation_modeles.sh`. Cette étape a impliqué l'entraînement des modèles acoustiques et linguistiques en utilisant les données préparées précédemment.

Entraînement des Modèles Acoustiques et Linguistiques

L'entraînement des modèles acoustiques a nécessité l'alignement des caractéristiques acoustiques avec

les unités phonétiques. Ce processus a été itératif pour ajuster les paramètres du modèle en fonction des données d'entraînement.

Méthodes d'Estimation des Paramètres

Différentes méthodes d'estimation des paramètres ont été utilisées, y compris des techniques probabilistes telles que l'Estimation Maximale de la Vraisemblance (EMV). Ces méthodes ont permis d'affiner les modèles pour une meilleure adaptation aux caractéristiques spécifiques des données.

6. PHASE DE RECONNAISSANCE

Utilisation du Script `reconnaissance.sh`

La phase de reconnaissance a été mise en œuvre avec le script `reconnaissance.sh`. Ce script a appliqué les modèles et paramètres précédemment estimés aux caractéristiques MFCC, générant ainsi des transcriptions textuelles des fichiers audio.

Application des Modèles et Paramètres Estimés

L'application des modèles acoustiques et linguistiques a impliqué la correspondance entre les caractéristiques acoustiques extraites des fichiers audio et les unités phonétiques, suivie de la conversion en transcriptions textuelles.

Génération des Transcriptions Textuelles

La sortie de cette étape était constituée de transcriptions textuelles représentant le contenu des fichiers audio. Ces transcriptions ont été comparées aux transcriptions de référence pour évaluer la précision du système.

7. ÉVALUATION DES RÉSULTATS

Utilisation du Script `resultats.sh`

Le script `resultats.sh` a été utilisé pour évaluer les résultats de la reconnaissance vocale. Cela a impliqué la comparaison des transcriptions générées avec des transcriptions de référence pour mesurer la précision et l'efficacité du système.

Comparaison des Transcriptions Générées avec les Références

La comparaison a été réalisée en utilisant des métriques telles que le taux d'erreur de mot (WER) et le taux de reconnaissance. Ces métriques ont fourni une évaluation quantitative de la performance du système.

Mesure de la Précision du Système

La précision du système a été mesurée en pourcentage de mots corrects dans les transcriptions générées. Une analyse détaillée a été effectuée pour comprendre les erreurs fréquentes et identifier des domaines potentiels d'amélioration.

8. RÉSULTATS OBTENUS

Précision Globale du Système sur l'Ensemble de Test

Le système de reconnaissance vocale développé a atteint une précision globale de [80 %] sur l'ensemble de test. Ce résultat a été obtenu après une analyse approfondie des résultats individuels de chaque fichier.

Analyse des Erreurs de Reconnaissance

Une analyse détaillée des erreurs de reconnaissance a été effectuée pour comprendre les motifs sous-jacents. Les erreurs ont été classées en catégories telles que les confusions de phonèmes, les problèmes d'accent, etc.

Points Forts et Limitations du Modèle

Les points forts du modèle incluent sa capacité à gérer une variété d'accents et de styles vocaux. Cependant, certaines limitations subsistent, notamment dans la gestion des bruits de fond et des variations extrêmes de tonalité.