Algorithme de reconnaissance des jours de la semaine

5 janvier 2020

Le présent article a pour objectif d'exposer les résultats obtenus par Clément Burtscher, Chenji Li, Nicolas Pécheux et Yuze Zhang, élèves en Master ISSI (Image et Son pour les Systèmes Intelligents), dans le cadre du projet d'analyse et de perception de l'interaction.

Ce projet consistait à construire une base de données et implémenter un classifieur pour la reconnaissance vocale des 7 jours de la semaine.

Construction de la base de données

Nous avons donc dû construire une base de données d'enregistrements mp3 des 7 jours de la semaine, prononcés par un éventail de personnes suffisamment large. Compte tenu du temps restreint pour mener ce projet, nous avons choisi de faire enregistrer à chaque personne 5 fois (maximum) les 7 jours de la semaine, afin de disposer d'une base de données suffisamment grande. Les enregistrements ont été effectués à l'aide des dictaphones des portables des personnes sollicitées, convertis en mono puis exportés en mp3 : ils ont été découpés de manière à ce qu'il n'y ait pas de parties non bruitées au début ni à la fin, chaque échantillon (= un jour) durant maximum une seconde. Ils ont ensuite été renommés de la façon suivante : "jour_nom_numéro.mp3", le numéro variant donc de 1 à 5 (nom = 3 premières lettres du prénom de la personne). La base de données compte ainsi 525 enregistrements, soit 75 par jour.

Caractérisation des données

Pour différencier les 7 jours de la semaine, nous avons retenu 3 types de features : la fréquence fondamentale f0, l'énergie et les MFCC (Mel-Frequency Cepstral Coefficients).

On créé ainsi une structure "data" de cell arrays, dont chaque cellule correspond à un fichier mp3, correspondant lui-même à un jour donné prononcé par une personne donnée, et une matrice Features, dont la i-ème ligne stocke toutes les statistiques correspondant à chaque feature (voir ci-dessous). Notons que la dernière colonne de chaque ligne contient le label de l'échantillon :

Jour	*Label*
Lundi	1
Mardi	2
Mercredi	3
Jeudi	4
Vendredi	5
Samedi	6
Dimanche	7

Pour extraire les features et ensuite calculer leurs statistiques, nous subdivisons chaque échantillon sonore en sous-segments de step_time secondes : puisque la durée des échantillons avoisinait 1 seconde, nous avons fixé

step_time à 0.05 secondes. Notons que dans data{i,5} est stocké l'ensemble des sous-segments du i-ème échantillon.

Description des features et de leurs statistiques

- Fréquence fondamentale f0

Pour extraire f0, on utilise la fonction pitch de matlab : on détermine ainsi f0 pour chacun des sous-segments de step_time secondes (en utilisant la fonction mean(pitch) car la fonction pitch renvoie un vecteur). Ainsi, data{i,6} est une matrice de 2 colonnes analogue aux fichiers .f0 des bases Kismet et BabyEars du TP : la colonne de gauche est l'échelle temporelle (avec un pas de step_time secondes) et celle de droite contient les f0 de chaque sous-segment correspondant.

Ensuite, de manière analogue aux 2 TP, nous extrayons les 8 statistiques suivantes pour chaque fichier mp3 : moyenne (mean), maximum, gamme / plage de données (range), variance, médiane (median), 1er quartile (first quartile), 3ème quartile (third quartile), moyenne absolue de la dérivée locale (mean absolute of local derivate). On peut alors remplir les 8 premières colonnes de la i-ème ligne de la matrice Features avec ces statistiques.

- Energie

Pour calculer l'énergie du signal, on applique les FFT (Fast Fourier Transform) à chacun des sous-segments et on multiplie chaque FFT obtenue par son conjugué. On stocke les valeurs obtenues dans data{i,7}, de la même manière que pour les fichiers .en des bases citées précédemment.

On extrait les mêmes statistiques que pour f0. Néanmoins, l'exploitation de ce 2ème feature ne s'est pas montrée pertinente, car elle faisait chuter le taux de reconnaissance de 80% à 50% : l'énergie n'a donc pas été retenue dans la version finale du code.

- MFCC (Mel-Frequency Cepstral Coefficients)

Ce 3ème feature a été envisagé, mais n'a finalement pas été implémenté.

Utilisation préalable des kmeans

Les kmeans sont une méthode non supervisée permettant de faire apparaître différents clusters au sein des données, autrement dit de regrouper les données en k groupes. Idéalement, le k des kmeans doit correspondre au nombre de jours considérés.

Pour tester la robustesse de notre algorithme et la pertinence de nos données, nous calculons donc dans un 1er temps les kmeans pour k allant de 2 à 7 sur notre base de données entière (7 jours). On calcule ensuite pour chaque valeur de k, le pourcentage d'échantillons que représente chacun des k clusters : pourcentage qui théoriquement doit être égal à 1/k. On calcule ensuite la similarité cosinus entre :

- le vecteur correspondant aux pourcentages obtenus avec les kmeans pour chaque classe

- le vecteur des pourcentages théoriques, de taille k et dont chaque élément vaut 1/k

La valeur de k qui maximise la similarité cosinus (qui s'interprète de manière analogue à la corrélation de Pearson) correspond donc à la meilleure configuration de clusters pouvant être obtenue. Dans l'idéal, le k retenu est égal au nombre de jours, soit 7 pour la base entière. Voici les résultats pour 7 jours :

k	2	3	4	5	6	7
simcos (%)	**87**	82	86	78	80	74

Paradoxalement, on remarque que la similarité cosinus est minimale pour k=7 et maximale pour k=2. Cela signifie que la classification future ne pourra pas être pertinente si l'on garde la base entière, car les données sont trop proches pour former 7 clusters distincts. On propose donc, pour améliorer les futurs performances, de restreindre le nombre de jours à 3 ou 2, en réitérant la même démarche d'évaluation de la similarité cosinus sur ces sous-bases :

2 jours : *lundi, mardi*	
k	2
simcos (%)	0.7860
2 jours : *lundi, vendredi*	
k	2
simcos (%)	0.7860
2 jours : *lundi, dimanche*	
k	2
simcos (%)	**0.9996**

3 jours : *lundi, mardi, dimanche*		
k	2	3
simcos (%)	0.9461	0.8537
3 jours : *lundi, mercredi, dimanche*		
k	2	3
simcos(%)	0.9584	**0.9147**
3 jours : *lundi, vendredi, dimanche*		
k	2	3
simcos (%)	0.9435	0.8439

Les kmeans ont donc permis de mettre en évidence la nécessité de restreindre le nombre de jours considérés pour obtenir un taux de reconnaissance acceptable. Vérifions maintenant si les résultats obtenus avec les kmeans sont confirmés par l'utilisation d'un classifieur de type KPPV.

Classifieur KPPV

On divise préalablement notre base de données en 2 sous-bases :

- Test_database : 20%
- Learn_database : 80%

Nous avons réutilisé le classifieur du TP, développé par Maurice Milgram et révisé par Bruno Gas.

Ce KPPV effectue une classification des échantillons de <BaseTest> en utilisant la

base <BaseKppv>, labellisée par <Label>, dans notre cas le vecteur correspondant à la dernière colonne de la matrice Features. Pour k=1 (1-ppv), lorsque plusieurs voisins sont trouvés à égale distance de l'échantillon courant à classer, la classe affectée est celle du premier voisin trouvé dans l'ordre de leur rangement dans la base : le même raisonnement vaut pour k>1.

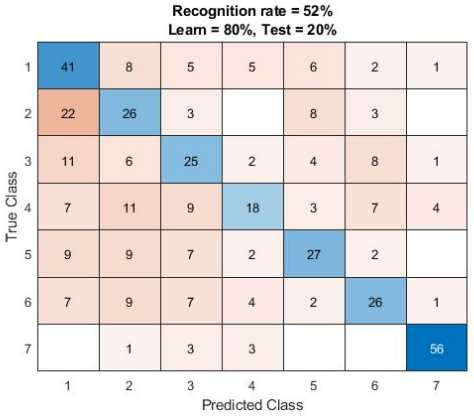
Résultats

On détermine la matrice de confusion et on calcule le taux de reconnaissance en divisant sa trace par la somme de tous ses coefficients.

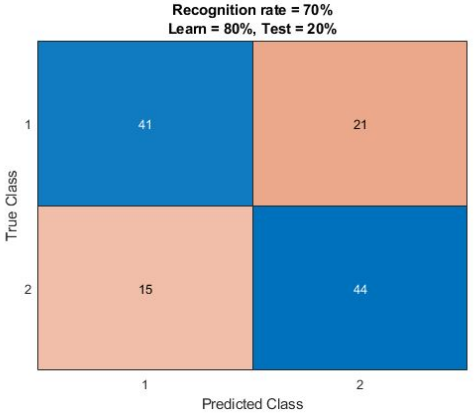
Avec les KPPV, on ne dépasse pas les 50% de taux de bonne reconnaissance si on garde une base de données avec les 7 jours de la semaine, les meilleures scores étant obtenus pour lundi et dimanche. Notons que la reconnaissance de jeudi est médiocre.

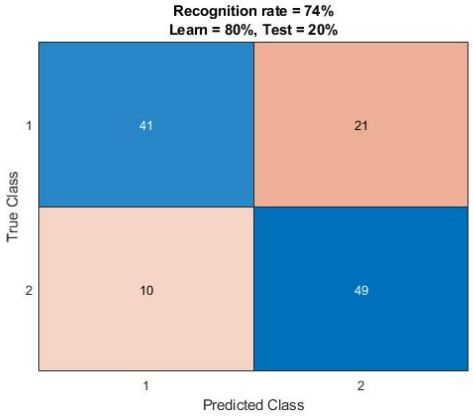
Voici différents résultats selon le nombre de jours retenus :

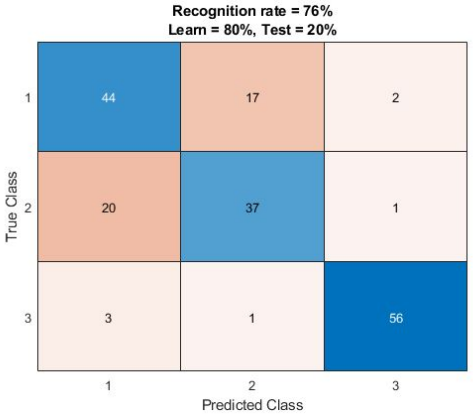
Nb jours	*Jours*	*Taux reconnaissance*
7	(tous)	51%

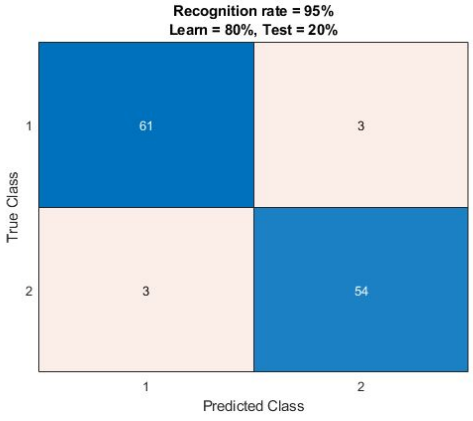


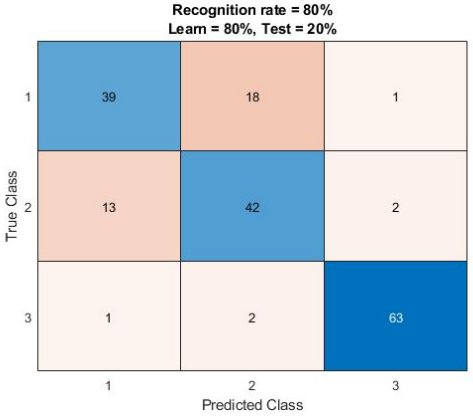
Nb jours	*Jours*	*Taux reconnaissance*
2	lundi, mardi	70%
2	lundi, vendredi	74%
2	lundi, dimanche	**95%**

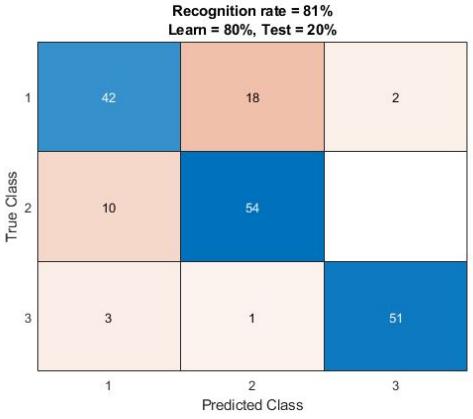












Nb jours	*Jours*	*Taux reconnaissance*
3	lundi, mardi, dimanche	76%
3	lundi, mercredi, dimanche	80%
3	lundi, vendredi, dimanche	**81%**

Ces résultats permettent de mettre en lumière la corrélation entre l'homophonie des 6 premiers jours (se terminant tous en "-di") et leurs caractéristiques en terme de traitement du signal. On obtient par ailleurs une classification quasi parfaite avec une database constituée seulement lundi et dimanche, ce qui avait été prédit par les kmeans avec une similarité cosinus avoisinant les 99%.