



Rapport du projet

**Apprentissage de  
Modèles de Markov cachés  
et détection de mots-clés**

Chengwanli YANG

Master 2  
Reconnaissance de la parole

Sorbonne Université – Faculté des Lettres  
Janvier 2022

## Table des matières

<b>1.</b>	<b>Introduction.....</b>	<b>2</b>
1.1	Modèle de Markov caché et HTK .....	2
1.2	Description du corpus.....	2
1.3	Objectif du projet.....	2
<b>2.</b>	<b>Élaboration des ressources .....</b>	<b>3</b>
2.1	Transcription SAMPA .....	3
<b>3.</b>	<b>Segmentation et apprentissage des HMMs phonétiques.....</b>	<b>3</b>
3.1	Segmentation des phonèmes sur Praat .....	3
3.2	Préparation des données .....	4
3.3	Apprentissage des HMMs phonétiques.....	6
<b>4.</b>	<b>Décodage acoustico-phonétique et détection des mots-clés.....</b>	<b>7</b>
4.1	Décodage phonétiquement de corpus de test.....	7
4.2	Détection des mots-clés.....	9
4.2.1	Détection des caractères d'une syllabe.....	9
4.2.2	Détection des mots de 2 syllabes .....	12
4.2.3	Détection des mots de plus de 3 syllabes .....	13
<b>5.</b>	<b>Conclusion.....</b>	<b>14</b>

# **1. Introduction**

## **1.1 Modèle de Markov caché et HTK**

Le modèle de Markov caché (Hidden Markov Model ou HMM) est un modèle statistique. Il est une sorte de chaîne de Markov. Son état de l'exécution ne peut pas être observé directement, par contre, certaines variables affectées par l'état peuvent être connues.

Depuis les années 80, le modèle de Markov caché est utilisé à la reconnaissance de la parole avec un grand succès.

Pour développer notre système de reconnaissance, nous aurons besoin de logiciel HTK (Hidden Markov Model Tool Kit) qui est un ensemble d'outils pour la gestion des HMMs.

## **1.2 Description du corpus**

Notre corpus est un flux audio de 3 minutes en mandarin. Il y a deux locuteurs qui ont l'accent pékinois. Nous l'appelons Erhua. C'est-à-dire que le suffixe « -er » apparaît régulièrement à la fin des mots prononcés modifiant la voyelle de la syllabe précédente par une rétroflexion. En raison de cet accent, les locuteurs mangent souvent les phonèmes, surtout des consonnes fricatives, par exemple [sh] en pinyin. Ce qui rend plus difficile la segmentation et la reconnaissance des phonèmes dans notre travail.

## **1.3 Objectif du projet**

L'objectif de ce projet est de nous familiariser avec les outils HTK et la mise en œuvre de la détection des mots-clés à l'aide des modèles de Markov cachés phonétique.

Le projet se compose de 3 étapes suivantes. Premièrement, l'élaboration des ressources. Nous allons découper les 2 premières minutes de l'audio pour faire apprendre HMMs, et la dernière minute pour les tester. Nous phonétiserons les tours de parole avec le codage SAMPA (Speech Assessment Methods Phonetic Alphabet). Deuxièmement, la segmentation et l'apprentissage des HMMs phonétiques. À cette étape, nous allons utiliser les outils HTK afin d'entraîner les HMMs basés sur notre corpus. Troisièmement, le décodage acoustico-phonétiques et la détection des mots-clés. Cette partie expérimentale permet de décoder phonétiquement le corpus de test et de reconnaître des mots-clés.

## 2. Élaboration des ressources

### 2.1 Transcription SAMPA

Avant de faire la transcription SAMPA, nous avons découpé les tours de parole des 2 premières minutes en 25 fichiers étant moins de 9 secondes sur lesquels les HMMs auront été appris, et en sauvegardant dans le répertoire « wav ».

Le SAMPA (Speech Assessment Methods Phonetic Alphabet) est un jeu de caractères phonétiques utilisable sur ordinateur et basé sur l'alphabet phonétique international (API). Il a été originellement conçu pour six langues européennes. Quant au mandarin, il n'existe pas encore un tableau officiel complet des symboles SAMPA. Au début, nous avons utilisé la convention « SAMPA-C Labeling Convention for Standard Chinese »<sup>1</sup>. Cependant, dans l'étape suivante, l'apprentissage HMMs phonétiques, en raison de données de corpus, plus précisément, le nombre de segments des phonèmes insuffisant, il est impossible d'initialiser les HMMs. Par conséquent, nous avons choisi finalement la convention SAMPA+ créée par les étudiants chinois (Nishan TANG, Zhijie ZHOU et Can CUI) de Sorbonne en 2020. Les avantages de cette convention sont suivants : elle contient des phonèmes fréquents de mandarin, les formes de SAMPA+ ne sont pas changées beaucoup par rapport à pinyin et la création spécialement pour ce projet, elle est plus simplifiée que les autres conventions, par exemple, ils ont intégré créativement les phonèmes [ɿ] et [ɨ], spécialisés en chinois. Le plus grand intérêt est le fait que nous pouvons faire apprendre nos HMMs avec leurs corpus de l'apprentissage, car nous exploitons le même type de transcription SAMPA. Après avoir demandé à Can CUI, nous avons eu donc 8 minutes de corpus (2 min de notre corpus et 6 min de corpus des étudiants chinois de 2020) pour entraîner les HMMs. Le problème de l'initialisation a été donc résolu.

## 3. Segmentation et apprentissage des HMMs phonétiques

### 3.1 Segmentation des phonèmes sur Praat

Pour le mandarin, la segmentation des phonèmes est plus difficile que la langue européenne, telle que le français, l'espagnol, l'italien, etc. Grâce à Praat, il est possible de segmenter automatiquement les phonèmes en mandarin. Les étapes sont suivantes : l'ajout d'un tier de la transcription pinyin, « Interval → Align interval ».

---

<sup>1</sup> <http://www.d-ear.com/CCC/resources/SampaC.pdf>

Il a éventuellement besoin de les évaluer manuellement, par exemple les frontières des phonèmes, notamment les consonnes. Nous les avons corrigées à l'aide de spectrum, généralement, la courbe de la voyelle est plus lisible que celle de la consonne.

Une fois la segmentation faite, nous pouvons ajouter un tier pour annoter des phonèmes en SAMPA, y compris le silence. Et puis, nous avons le sauvegardé au format textgrid en supprimant le tier de la transcription pinyin.

La figure 1 présente les paramètres de l'alignement, les segments phonétiques et la transcription SAMPA sur Praat.

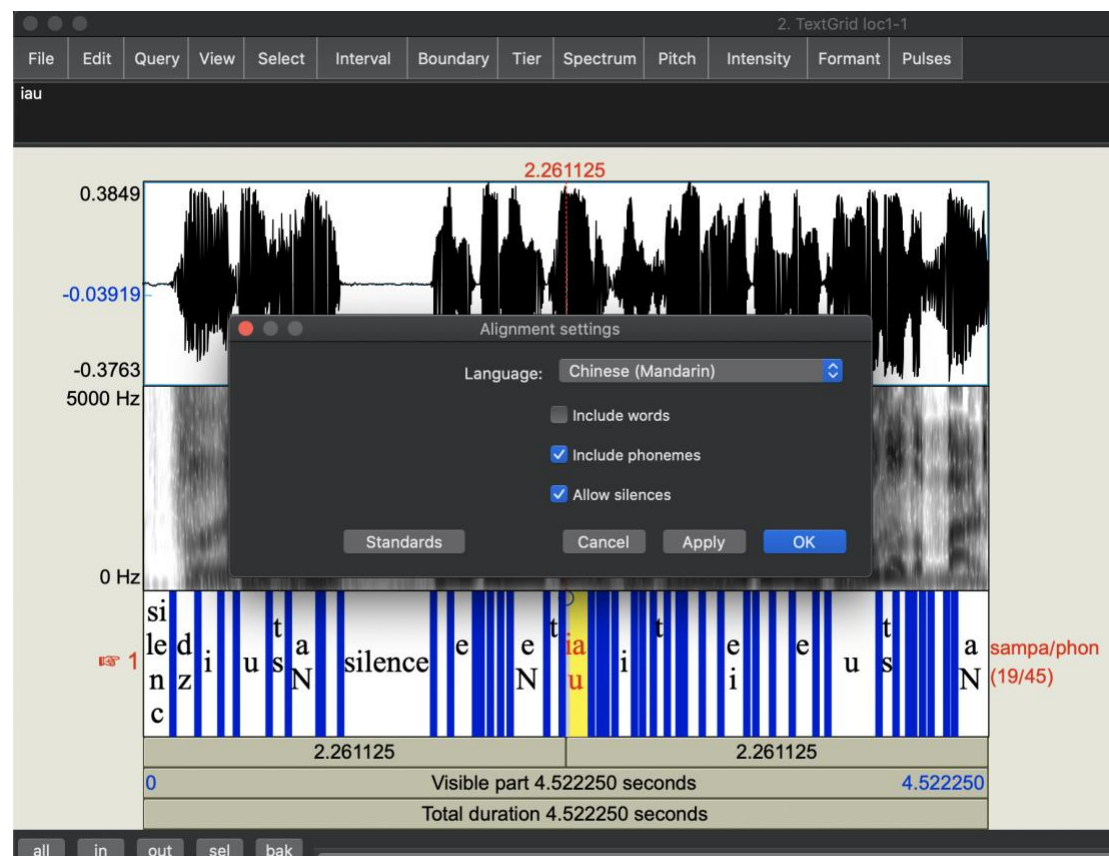
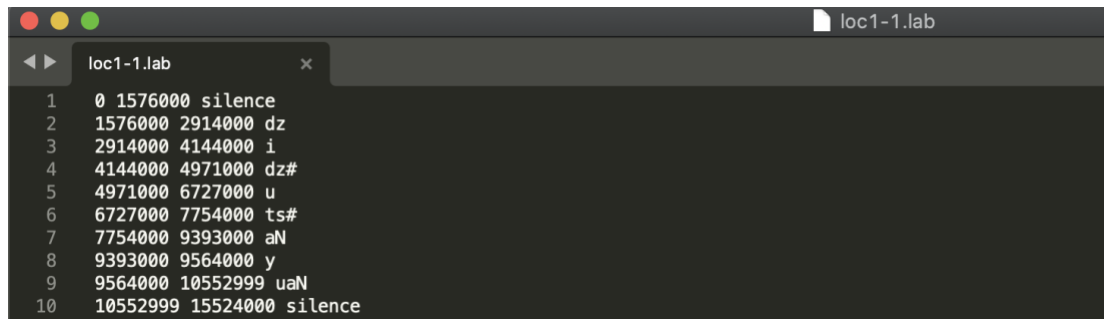


figure 1 Segmentation des phonèmes sur Praat et annotation SAMPA

### 3.2 Préparation des données

Avant le commencement de l'apprentissage HMMs, il est indispensable de préparer des données, comme des fichiers lab, le fichier « lex0000.txt » et des fichiers mfc.

Les fichiers lab comprennent les segmentations et les étiquetages phonétiques des signaux de parole d'apprentissage. En raison de grand nombre de segments phonétique, il est impossible de les créer manuellement. Grâce à CUI, à TANG et à ZHOU, ils ont partagé les codes python pour convertir des fichiers. Le script « textgrid\_to\_lab.ipynb » sert à convertir automatiquement le format textgrid en lab.

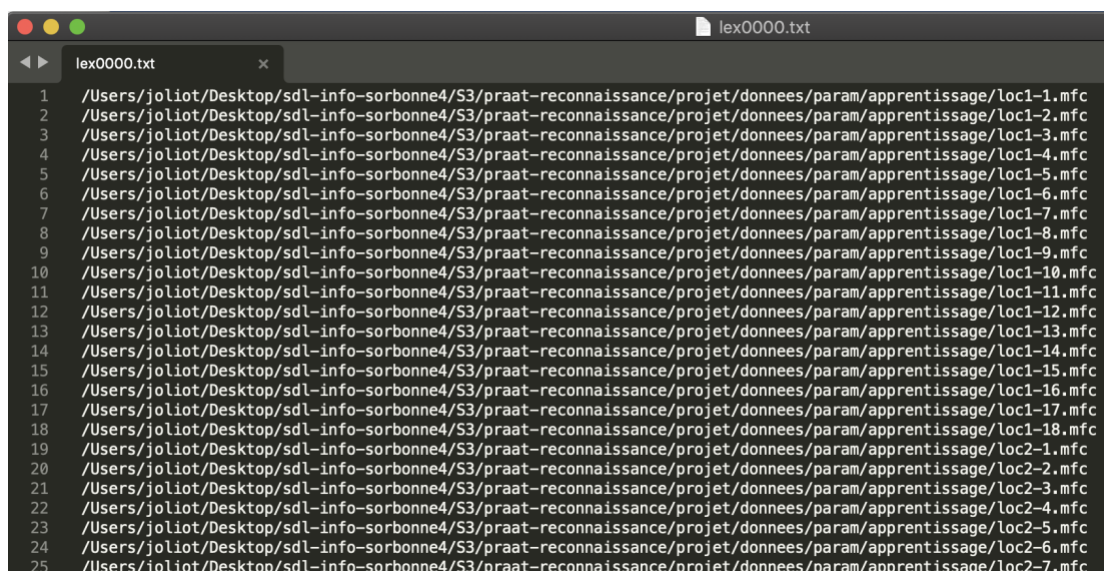


```
loc1-1.lab
1 0 1576000 silence
2 1576000 2914000 dz
3 2914000 4144000 i
4 4144000 4971000 dz#
5 4971000 6727000 u
6 6727000 7754000 ts#
7 7754000 9393000 aN
8 9393000 9564000 y
9 9564000 10552999 uaN
10 10552999 15524000 silence
```

figure 2 Exemple d'un fichier lab

Sur la figure 2, nous constatons que l'intervalle de temps et le phonème sont alignés, les secondes sont étendues dix millions fois.

Le fichier « lex0000.txt » est une liste phonétisée du vocabulaire, autrement dit les chemins d'accès des fichiers mfc de l'apprentissage, comme la figure 3. Le script « lex0\_phones.ipynb » permet de le générer à partir des fichiers textgrid et de créer les fichiers « phoneHTK » (liste des HMMs phonétiques) et « SampaToHTK ».



```
lex0000.txt
1 /Users/joliot/Desktop/sdl-info-sorbonne4/S3/paat-reconnaissance/projet/donnees/param/apprentissage/loc1-1.mfc
2 /Users/joliot/Desktop/sdl-info-sorbonne4/S3/paat-reconnaissance/projet/donnees/param/apprentissage/loc1-2.mfc
3 /Users/joliot/Desktop/sdl-info-sorbonne4/S3/paat-reconnaissance/projet/donnees/param/apprentissage/loc1-3.mfc
4 /Users/joliot/Desktop/sdl-info-sorbonne4/S3/paat-reconnaissance/projet/donnees/param/apprentissage/loc1-4.mfc
5 /Users/joliot/Desktop/sdl-info-sorbonne4/S3/paat-reconnaissance/projet/donnees/param/apprentissage/loc1-5.mfc
6 /Users/joliot/Desktop/sdl-info-sorbonne4/S3/paat-reconnaissance/projet/donnees/param/apprentissage/loc1-6.mfc
7 /Users/joliot/Desktop/sdl-info-sorbonne4/S3/paat-reconnaissance/projet/donnees/param/apprentissage/loc1-7.mfc
8 /Users/joliot/Desktop/sdl-info-sorbonne4/S3/paat-reconnaissance/projet/donnees/param/apprentissage/loc1-8.mfc
9 /Users/joliot/Desktop/sdl-info-sorbonne4/S3/paat-reconnaissance/projet/donnees/param/apprentissage/loc1-9.mfc
10 /Users/joliot/Desktop/sdl-info-sorbonne4/S3/paat-reconnaissance/projet/donnees/param/apprentissage/loc1-10.mfc
11 /Users/joliot/Desktop/sdl-info-sorbonne4/S3/paat-reconnaissance/projet/donnees/param/apprentissage/loc1-11.mfc
12 /Users/joliot/Desktop/sdl-info-sorbonne4/S3/paat-reconnaissance/projet/donnees/param/apprentissage/loc1-12.mfc
13 /Users/joliot/Desktop/sdl-info-sorbonne4/S3/paat-reconnaissance/projet/donnees/param/apprentissage/loc1-13.mfc
14 /Users/joliot/Desktop/sdl-info-sorbonne4/S3/paat-reconnaissance/projet/donnees/param/apprentissage/loc1-14.mfc
15 /Users/joliot/Desktop/sdl-info-sorbonne4/S3/paat-reconnaissance/projet/donnees/param/apprentissage/loc1-15.mfc
16 /Users/joliot/Desktop/sdl-info-sorbonne4/S3/paat-reconnaissance/projet/donnees/param/apprentissage/loc1-16.mfc
17 /Users/joliot/Desktop/sdl-info-sorbonne4/S3/paat-reconnaissance/projet/donnees/param/apprentissage/loc1-17.mfc
18 /Users/joliot/Desktop/sdl-info-sorbonne4/S3/paat-reconnaissance/projet/donnees/param/apprentissage/loc1-18.mfc
19 /Users/joliot/Desktop/sdl-info-sorbonne4/S3/paat-reconnaissance/projet/donnees/param/apprentissage/loc2-1.mfc
20 /Users/joliot/Desktop/sdl-info-sorbonne4/S3/paat-reconnaissance/projet/donnees/param/apprentissage/loc2-2.mfc
21 /Users/joliot/Desktop/sdl-info-sorbonne4/S3/paat-reconnaissance/projet/donnees/param/apprentissage/loc2-3.mfc
22 /Users/joliot/Desktop/sdl-info-sorbonne4/S3/paat-reconnaissance/projet/donnees/param/apprentissage/loc2-4.mfc
23 /Users/joliot/Desktop/sdl-info-sorbonne4/S3/paat-reconnaissance/projet/donnees/param/apprentissage/loc2-5.mfc
24 /Users/joliot/Desktop/sdl-info-sorbonne4/S3/paat-reconnaissance/projet/donnees/param/apprentissage/loc2-6.mfc
25 /Users/joliot/Desktop/sdl-info-sorbonne4/S3/paat-reconnaissance/projet/donnees/param/apprentissage/loc2-7.mfc
```

figure 3 Exemple du fichier lex0000.txt

Les fichiers mfc sont les paramètres d'analyse (MFCC) du corpus, ils sont représentés par des vecteurs. Le script « hcopy.bash » permet d'extraire les MFCC de tous les fichiers audio de l'apprentissage, et de les sauvegarder dans le répertoire « param ».

Jusqu'à maintenant, nous disposons des données nécessaires pour l'entraînement des HMMs phonétiques. Pour enrichir les modèles, nous avons fusionné les données (fichiers lab, mfc et wav) des étudiants chinois de 2020. Nous avons également créé le répertoire racine « projet » qui contient le répertoire « donnees », le répertoire « config » dans lequel sont les fichiers de configuration des modules logiciels HTK ; le

répertoire « lists » pour les fichiers de configuration de l'application ; le répertoire « tmp » pour les dictionnaires ainsi que les scripts d'application utilisés en Perl (« runApprentissage.pl », « runDAP.pl », « runDetections1.pl », etc.). Dans l'intérieur de « donnees », il y a les dossiers « lab », « param » (les mfc d'apprentissage et de test), « wav » (les audios d'apprentissage et de test), « lists » (les fichiers des vecteurs d'analyse) et « hmms » (les modèles de Markov cachés).

Étant donné que nous ferons 3 expérimentations pour la détection des mots-clés dans la section 4, nous avons créé les dossiers pour regrouper les fichiers répétitifs, par exemple, le dossier « label » qui est à l'intérieur du répertoire racine contient les fichiers « lable.lst » des 3 détections. Le même cas dans le répertoire « lists », les dossiers « test\_1syllabe », « test\_2syllabe » et « test\_4syllabe » regroupent les fichiers utilisés de chaque expérience (« keywords.dic », « keywords.net », « keywordsplus.dic », etc.).

### **3.3 Apprentissage des HMMs phonétiques**

Lorsque toutes les préparations sont achevées, nous pouvons faire apprendre des HMMs selon les 6 minutes de l'apprentissage.

Tout d'abord, il faut créer 4 sous-répertoires dans « hmms », « hmm.0 », « hmm.1 », « hmm.2 » et « hmm.3 ». Le rôle de ces 4 sous-répertoires est de sauvegarder les HMMs de chaque phonème et les modèles établis à partir de ces HMMs.

Ensuite, il faut juste exécuter le script « runApprentissage.pl » dans le répertoire racine « projet » en modifiant le chemin défini, les HMMs seront générés. Maintenant, nous entrons dans le détail. Le script « runApprentissage.pl » se compose de 4 fonctions, « HInit » pour l'initialisation des HMMs, « HRest » pour l'apprentissage en mode isolé des HMMs, « HERest » pour l'apprentissage en mode connecté des HMMs et « HHed » pour la duplication des gaussiennes des HMMs. Voici un extrait des HMMs globaux entraînés pour tous les phonèmes sur la figure 4.

```

HMMmacro
1 ~o
2 <STREAMINFO> 1 12
3 <VECSIZE> 12<NULLD><MFCC><DIAGC>
4 ~h "@"
5 <BEGINHMM>
6 <NUMSTATES> 5
7 <STATE> 2
8 <NUMMIXES> 3
9 <MIXTURE> 1 2.343219e-01
10 <MEAN> 12
11 -2.619823e+00 -4.910905e+00 9.459058e+00 -8.886514e+00 -8.731084e+00 -2.389992e+00 -1.693627e+01 4.821197e+00 -1.588907e-01
12 5.627428e-01 -8.892193e-01 -9.402205e-01
13 <VARIANCE> 12
14 1.894117e+01 4.129163e+01 7.635931e+01 7.930250e+01 8.379511e+01 9.102001e+01 9.775419e+01 9.404732e+01 6.508639e+01 8.886760e+01
15 5.132529e+01 2.966661e+01
16 <CONST> 7.148202e+01
17 <MIXTURE> 2 4.601618e-01
18 <MEAN> 12
19 -2.436619e+00 -1.118475e+01 3.026020e+00 -1.490852e+01 -8.651515e-01 5.980329e+00 -1.265806e+01 4.082367e+00 1.335442e+01
20 -3.390848e+00 -9.081961e+00 -7.966055e+00
21 <VARIANCE> 12
22 5.322351e+00 1.211598e+01 1.199009e+01 1.775002e+01 5.789770e+01 4.272863e+01 5.880890e+01 3.134310e+01 7.192172e+01 5.375011e+01
23 3.150400e+01 1.477155e+01
24 <CONST> 6.131701e+01
25 <MIXTURE> 3 3.055170e-01

```

figure 4 Extrait de HMMs globaux entraînés (hmm.3)

## 4. Décodage acoustico-phonétique et détection des mots-clés

Dans cette section, nous allons évaluer la performance de nos HMMs générés. Dans un premier temps, nous ferons la reconnaissance de la dernière minute du corpus, et calculerons le taux d'erreur. Dans un second temps, nous allons faire 3 expérimentations à l'aide du réseau de décodage construit : la détection des caractères d'une syllabe, la détection des mots de 2 syllabes et la détection des mots plus de 3 syllabes.

### 4.1 Décodage phonétiquement de corpus de test

Nous avons d'abord extrait la dernière minute du corpus « test.wav » dans le répertoire « wav », et généré le fichier mfc dans « param/test ».

Le script « runDAP.pl » permet d'obtenir le résultat de décodage, comme la figure 5.

```

(base) maie:projet joliot$ perl runDAP.pl
Read 59 physical / 59 logical HMMs
Read lattice with 62 nodes / 178 arcs
Created network with 123 nodes / 239 links
File: /Users/joliot/Desktop/sdl-info-sorbonne4/S3/praat-reconnaissance/projet/donnees/param/test/test.mfc
eN b uaN k @ l iau ueN dz# u dz @ silence k @ @ #s iE au u aN y ei #dz iaN l dz i #s k l x au @ s# y d k u u
i #s x g f ts# uo dz# s# g iaN ts# au d t ou l r u #dz a #dz s# ai i n i iE i n n ai r dz# iaN d i r aN aN y
i b uei iaN s# r ai eN silence ts# s# l oNN l aN d r s# n n i iu #dz k eN k ai ai #s d ou #s oNN g au aN b
k s# s# #i #dz r eN silence b d uaN aN s# l l uaN yE dz dz l u y uaN au oNN u x aN t #dz y oNN au uaN #ts y
uaN y s x ou u iN eN ts# oNN u f ua uaN au silence x ua #dz uaN l y @ u y iE silence s dz ui iE i silence k
uo m ei ts# ua l t ai x aN silence silence b uaN l @ y l uaN dz n uaN i dz# r i i iE #dz i l a uaN ai i dz#
l s# l k a eN ia @ ts# uo @ #s ei #dz uo n l i @ iou a uaN l dz d au #dz ua i #s iE uaN n t a silence #ts y
ts uo l s# k x ou l ei d a eN #dz i eN k ai eN dz k a silence b ua aN silence silence t y m yE n u uaN ei #t
s iaN dz# #dz ui s# r k eN ts# u dz ua aN dz# u dz i s y r aN iE ai eN k x l l x g x eN ou au uaN i #ts ui k
aN r ts# uo ui #dz iN m iNN uaN silence silence l r i uo y ts# r n y d ai iNN n i k x aN dz# eNN eN au ueN
n uo iNN #ts y l x ou @ iE ui m n iNN silence k eNN aNN o @ #s k f uaN ou x a eN @ dz# uaN ua @ i @ #i #s iN
N x au silence eN uaN r eN i y iaN b oNN r ai i s# dz ai n u l n i d #i #dz i #s iE @ i #s i p ei iu x u x u
a silence t #ts s# s# s# g x x au d a #s y uaN n @ n @ iNN ou b x o l l au uaN ts# eN dz# aN p silence eN yE
@ dz l ui m uaN uaN dz# u dz @ i uo i #ts y iaN silence b x u #dz iNN ts# u dz i s# #i dz dz l #ts x f k ai
dz# x silence ts# aNN n ai #dz d d oNN k au eN aN l s# y uo x k eN uo #s x #ts x @ au ou n @ n ts# @ au l d
z# u dz ua b u dz# dz# u dz @ silence #ts y s# r yE iau ui #s dz# r silence t silence @ i ts# g u dz l k k @
x uo @ uo i #s iE yn dz# r #dz ei ts# au aNN #s #dz iN x uo uaN r uaN m uaN dz# ei s# s# p m uaN l uaN dz#
u dz aN a silence s @ ui ou @ #dz @ eN aN dz# ui #s ei #s ia iau p aN iou f uo l i #s y @ k == [5998 frame
s] -40.2006 [Ac=-241123.5 LM=0.0] (Act=121.0)

```

figure 5 Décodage acoustico-phonétique de test.wav



Pour évaluer la qualité de la reconnaissance de « test.wav », il faut marquer manuellement les phonèmes reconnus corrects (en vert).

eN b uaN k @ l iau ueN dz# u dz @ silence k @ @ #s iE au u aN y ei #dz iaN l dz i #s k l x au @ s# y d k u ui  
 #s x g f ts# uo dz# s# g iaN ts# au d t ou l r u #dz a #dz s# ai n i iE i n n ai r dz# iaN d i r aN aN y i b uei iaN  
 s# r ai eN silence ts# s# l oNN l aN d r s# n n i u #dz k eN k ai ai #s d ou #s oNN g au aN b k s# s# i #dz r  
 eN silence b d uaN aN s# l l uaN yE dz dz l u y uaN au oNN u x aN t #dz y oNN au uaN #ts y uaN y s x ou u  
 iN eN ts# oNN u f ua uaN au silence x ua #dz uaN l y @ u y iE silence s dz ui iE i silence k uo m ei ts# ua l t ai  
 x aN silence silence b uaN l @ y l uaN dz n uaN i dz# r i i iE #dz i l a uaN ai i dz# l s# l k a eN ia @ ts# uo @  
 #s ei #dz uo n l i @ iou a uaN l dz d au #dz u a i #s iE uaN n t a silence #ts y ts uo l s# k x ou l ei d a eN #dz i  
 eN k ai eN dz k a silence b ua aN silence silence t y m yE n u uaN ei #ts iaN dz# #dz ui s# r k eN ts# u dz ua  
 aN dz# u dz i s y r aN iE ai eN k x l l x g x eN ou au uaN i #ts ui k aN r ts# uo ui #dz iN m iNN uaN silence  
 silence l r i uo y ts# r n y d ai iNN n i k x aN dz# eNN eN au ueN n uo iNN #ts y l x ou @ iE ui m iNN silence  
 k eNN aNN o @ #s k f uaN ou x a eN @ dz# uaN ua @ i @ #i #s iNN x au silence eN uaN r eN i y iaN b oNN r  
 ai i s# dz ai n u l n i d #i #dz i #s iE @ i #s i p ei iu x u x ua silence t #ts s# s# s# g x x au d a #s y uaN n @ n  
 @ iNN ou b x o l l au uaN ts# eN dz# aN p silence eN yE @ dz l u i m uaN uaN dz# u dz @ i uo i #ts y iaN  
 silence b x u #dz iNN ts# u dz i s# #i dz dz l #ts x f k ai dz# x silence ts# aNN n ai #dz d d oNN k au eN aN l s#  
 y uo x k eN uo #s x #ts x @ au ou n @ n ts# @ au l dz# u dz ua b u dz# dz# u dz @ silence #ts y s# r yE iau ui  
 #s dz# r silence t silence @ i ts# g u dz l k k @ x uo @ uo i #s iE yn dz# r #dz ei ts# au aNN #s #dz iN x uo  
 uaN r uaN m uaN dz# ei s# s# p m uaN l uaN dz# u dz aN a silence s @ ui ou @ #dz @ eN aN dz# ui s# ei #s  
 ia iau p aN iou f uo l i #s y @ k

figure 6 Marquages sur les phonèmes reconnus corrects

Sur la figure 6, nous voyons qu’il n’y a pas beaucoup de phonèmes reconnus corrects, 82 phonèmes (y compris le silence) parmi 660 phonèmes ont été correctement reconnus, soit le taux d’erreur est de :

$$(660-82)/660*100\% = 87,58\%$$

Il semble que la performance de notre système de reconnaissance est faible, par contre, il y a quand même des remarques importantes. Tout d’abord, la reconnaissance correcte des voyelles est mieux que celle de consonne. Cela n’est pas surpris, en raison du moyen de prononciation des locuteurs de notre corpus, des consonnes ont été souvent submergées. De plus, il n’y a pas mal d’erreurs dans les consonnes [dz#], [ts#] et [s#]. Ce sont les consonnes fricatives, comme [j] et [ʒ] en français. En outre, vu que les HMMs ont été enchaînés avec les corpus fusionnés, concernant l’annotation SAMPA, il manque un accord entre les annotateurs. C’est-à-dire que nous, surtout moi, n’avons pas l’accord pour les phonèmes proches. Par exemple, la consonne [y] n’existe pas dans la convention SAMPA+, nous l’avons fusionné avec la consonne [j], [#dz] en SAMPA. Cependant, nous ne savons pas comment les autres annotateurs ont fait dans ce cas. Ainsi, c’est peut-être une raison de provoquer le problème de la reconnaissance.

En conclusion, bien que le résultat de cette expérimentation soit insatisfaisant, les remarques que nous avons obtenues sont appréciables pour l’amélioration des HMMs.

## 4.2 Détection des mots-clés

La détection des mots-clés joue un rôle important dans la reconnaissance de parole. Notamment dans le commercial, la détection des mots-clés permet de connaître l'émotion (triste, content, colère) de client dans la conversation téléphonique par exemple.

Il existe 3 méthodes pour la détection, le filtrage d'un treillis des hypothèses d'un décodage acoustico-phonétique, le filtrage d'un treillis des hypothèses d'une dictée vocale et la méthode que nous utiliserons, la mise en concurrence d'une boucle des mots-clés et d'un modèle du monde à boucle de phonèmes.

Les moyens de la préparation des données des 3 détections sont pareils, nous la détaillerons donc dans la première détection. Pour la deuxième et la troisième détection, nous n'analyserons que leurs résultats.

### 4.2.1 Détection des caractères d'une syllabe

En mandarin, un caractère est une syllabe qui dispose d'une consonne et d'une voyelle. Il nous faut préparer quelques fichiers avant la détection. Tout d'abord, nous avons choisi les 5 caractères (*dang, guang, yang, cheng, li*) à détecter en les sauvegardant dans le fichier « label.lst ». Le fichier « test.lab » indique l'intervalle de temps des occurrences des caractères, comme la figure 7. Ensuite, dans le dossier « lists/test\_1syllabe » du répertoire racine « projet », les fichiers suivants sont indispensables : les fichiers « keywords.dic » et « keywordsplus.dic » sont les phonétisations du vocabulaire; « keywords.net » et « keywordsplus.net » sont les réseaux de décodage, le modèle isolé et le modèle du monde, comme la figure 8. « lex1.net » est la grammaire pour la reconnaissance de mots isolés, et « lex1.txt » pour l'énumération de tous les 5 caractères à détecter.

```

1 111245999 113018000 dang
2 124510000 126296000 dang
3 267307000 268488000 dang
4 96486000 98519000 guan
5 277235000 278746000 guan
6 569629000 571538000 yang
7 374110000 375772000 yang
8 439631000 441117000 cheng
9 49000 671000 li
10 67473000 68778000 li
11 451824000 452528999 li

```

figure 7 Fichier test.lab pour détecter les 5 caractères

```

keywords_1.net
1 keyword = dang | guan | yang | cheng | li
2 ( silence < $keyword silence > )

keywordsplus_1.net
1 $keyword = dang | guan | yang | cheng | li ;
2 $monde = < eN | g | uaN | y | dz# | ei | @ | x |
  ua | t | i | n | iE | iNN | d | a | #dz | ia | #s
  | dz | ai | uo | m | uei | iN | oNN | p | #i | yn
  | f | ts# | aNN | au | s# | iaN | l | k | aN | r
  | uaNN | ui | iau | yE | #ts | u | iu | s | iou |
  eNN | b | ts | uai | iaNN | er | ou | ioNN | ueN
  | o | silence >;
3 ( < $monde | $keyword > )
4

```

figure 8 Exemples de keywords.net et keywordsplus.net

Pour la première détection, il faut exécuter le script « runDetections1.pl » pour obtenir le fichier « net.2 » dans le répertoire « tmp ». Nous l’avons copié dans le dossier « lists » en renommant « keywordsplus.net2 ». Et puis, nous pouvons exécuter les scripts « runDetections2.pl » et « runDetections3.pl », bien sûr qu’il faut les paramétrer d’abord.

Après avoir exécuté les 3 scripts de détection, nous pouvons calculer la Figures of Merit (FOM) à l’aide de la commande « HResults -w -f label.lst EN.rec ». Voici le FOM obtenu :

```

----- Figures of Merit -----

```

KeyWord:	#Hits	#FAs	#Actual	FOM
dang:	0	0	3	0.00
guan:	0	0	2	0.00
yang:	0	0	2	0.00
cheng:	0	0	1	0.00
li:	0	0	3	0.00
Overall:	0	0	11	0.00

figure 9 Détection des 5 caractères sans récompense

Sur la figure 9, il n’y a aucun de caractère étant reconnu. Pour améliorer le résultat de FOM, nous pouvons ajouter la récompense aux arcs.

La figure 10 montre un extrait du fichier « keywordsplus.net2 », les lignes 3, 5,6,7,8 présentent les caractères (*li, cheng, yang, guang, dang*). Sur la partie gauche de la figure 10, le chiffre de la colonne I est pareil que le chiffre de la colonne E sur la partie droite. Nous pouvons donc confirmer que le chiffre représente le même caractère dans les 2 colonnes. Afin d'ajouter la récompense, la syntaxe est « I= ». Nous avons modifié plusieurs fois la récompense, le résultat n'était pas toujours très satisfaisant.

Line	I	W	J	S	E	l
1	VERSION=1.0					
2	N=68	L=258				
3	I=0	W=li				
4	I=1	W=NULL				
5	I=2	W=cheng				
6	I=3	W=yang				
7	I=4	W=guan				
8	I=5	W=dang				
9	I=6	W=silence				
10	I=7	W=NULL				
11	I=8	W=o				
12	I=9	W=ueN				
13	I=10	W=ioNN				
14	I=11	W=ou				
15	I=12	W=er				
16	I=13	W=iaNN				
17	I=14	W=uai				
18	I=15	W=ts				
19	I=16	W=b				
20	I=17	W=eNN				
21	I=18	W=iou				
22	I=19	W=s				
23	I=20	W=iu				
24	I=21	W=u				
25	I=22	W=#ts				
69	I=66	W=NULL				
70	I=67	W=NULL				
71	J=0	S=1	E=0		l=50	
72	J=1	S=7	E=0		l=20	
73	J=2	S=67	E=0		l=20	
74	J=3	S=0	E=1			
75	J=4	S=2	E=1			
76	J=5	S=3	E=1			
77	J=6	S=4	E=1			
78	J=7	S=5	E=1			
79	J=8	S=1	E=2		l=50	
80	J=9	S=7	E=2		l=20	
81	J=10	S=67	E=2		l=20	
82	J=11	S=1	E=3		l=95	
83	J=12	S=7	E=3		l=15	
84	J=13	S=67	E=3		l=15	
85	J=14	S=1	E=4		l=65	
86	J=15	S=7	E=4		l=25	
87	J=16	S=67	E=4		l=25	
88	J=17	S=1	E=5		l=95	
89	J=18	S=7	E=5		l=5	
90	J=19	S=67	E=5		l=5	
91	J=20	S=1	E=6			
92	J=21	S=7	E=6			
93	J=22	S=67	E=6			

figure 10 Fichier « keywordsplus.net2 »

----- Figures of Merit -----

KeyWord:	#Hits	#FAs	#Actual	FOM
dang:	0	364	3	0.00
guan:	1	134	2	0.00
yang:	2	297	2	0.00
cheng:	0	4	1	0.00
li:	2	77	3	0.00
Overall:	5	876	11	0.00

figure 11 Détection des 5 caractères avec récompense

La figure 11 présente le meilleur résultat que nous avons trouvé. Au total, 5 occurrences sont détectées parmi 11 occurrences. Le caractère « yang » a été bien trouvé. Et le système n'a pas reconnu « dang » et « cheng ».

### 4.2.2 Détection des mots de 2 syllabes

Nous avons choisi 3 mots ayant 2 syllabes, *zhuzi*, *yuanzi* et *yuanyin*. Ils sont fréquemment apparus au corpus de test, totalement 16 occurrences.

Après avoir la récompense (figure 12), nous avons obtenu le résultat ci-dessous (figure 13) :

Line	Parameter	Value
1	VERSION	1.0
2	N	66
3	L	250
4	I=0	W=yuanyin
5	I=1	W=NULL
6	I=2	W=yuanzi
7	I=3	W=zhuzi
8	I=4	W=silence
9	I=5	W=NULL
10	I=6	W=o
11	I=7	W=ueN
12	I=8	W=ioNN
13	I=9	W=ou
14	I=10	W=er
15	I=11	W=iaNN
16	I=12	W=uai
69	J=0	S=1 E=0
70	J=1	S=5 E=0
71	J=2	S=65 E=0
72	J=3	S=0 E=1
73	J=4	S=2 E=1
74	J=5	S=3 E=1
75	J=6	S=1 E=2 l=200
76	J=7	S=5 E=2 l=200
77	J=8	S=65 E=2 l=200
78	J=9	S=1 E=3 l=220
79	J=10	S=5 E=3 l=220
80	J=11	S=65 E=3 l=220
81	J=12	S=1 E=4
82	J=13	S=5 E=4
83	J=14	S=65 E=4

figure 12 Récompense pour les mots de 2 syllabes

----- Figures of Merit -----

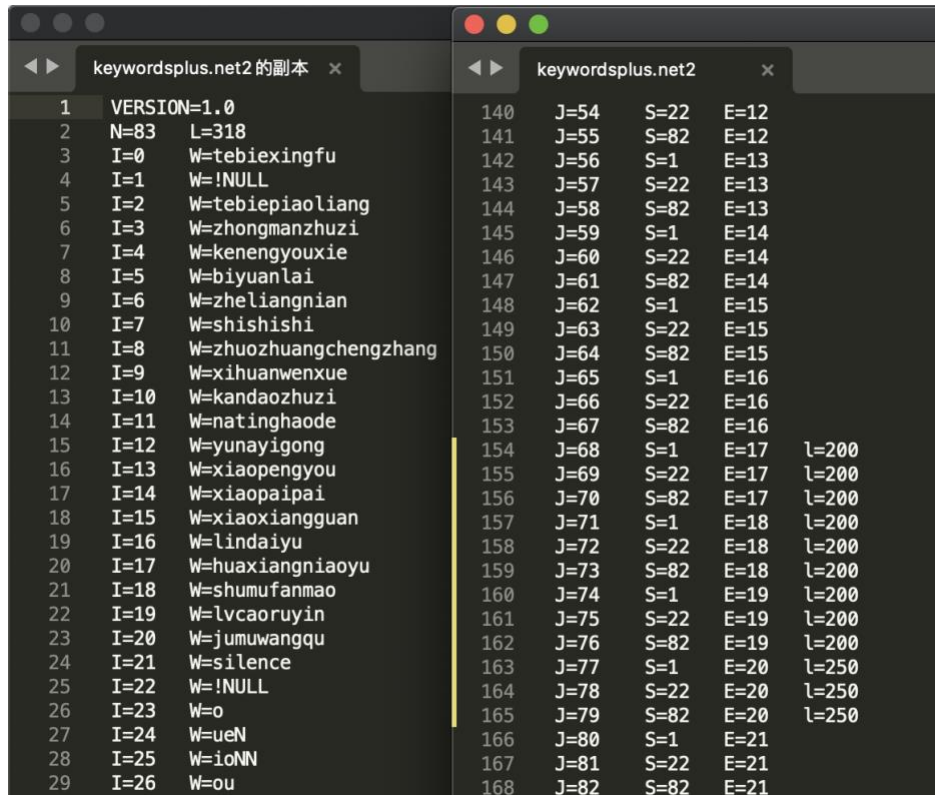
KeyWord:	#Hits	#FAs	#Actual	FOM
zhuzi:	10	435	10	0.00
yuanzi:	1	31	4	25.00
Overall:	11	466	14	7.14

figure 13 Détection des mots de 2 syllabes avec récompense

Nous pouvons constater d'abord que le résultat est bizarre, il n'existe pas le mot *yuanyin*. Nous avons bien vérifié tous les fichiers (« keywordsplus.net », « keywordsplus.dic » et « keywordsplus.net2 »), le mot *yuanyin* a été bien présenté. Nous supposons qu'il n'a pas été indiqué dans le résultat, c'est parce qu'il n'y a que 2 occurrences, il n'a pas donc été reconnu. Par contre, les autres mots sont bien détectés. Le système a trouvé 9 occurrences sur 10 pour le mot *zhuzi* et 2 sur 4 pour *yuanzi*. En conclusion, cette expérience est partiellement satisfaisante (sauf le mot *yuanyin*), car nous avons choisi les mots très fréquents du corpus, par conséquent, le système de reconnaissance a fait son travail.

### 4.2.3 Détection des mots de plus de 3 syllabes

Nous avons choisi totalement 20 mots ayant plus de 3 syllabes. Pour simplifier la tâche de récompense, nous avons décidé de détecter les 4 proverbes chinois (*jumuwangqu*, *lvcaoruyin*, *shumufanmao* et *huaxiangniaoyu*). Chacun des proverbes a 4 caractères, soit 4 syllabes. Voici la récompense (figure 14) et le résultat (figure 15) :



keywordsplus.net2 的副本		keywordsplus.net2	
1	VERSION=1.0	140	J=54 S=22 E=12
2	N=83 L=318	141	J=55 S=82 E=12
3	I=0 W=tebiexingfu	142	J=56 S=1 E=13
4	I=1 W=!NULL	143	J=57 S=22 E=13
5	I=2 W=tebiepiaoliang	144	J=58 S=82 E=13
6	I=3 W=zhongmanzhuzi	145	J=59 S=1 E=14
7	I=4 W=kenengyouxie	146	J=60 S=22 E=14
8	I=5 W=biyuanlai	147	J=61 S=82 E=14
9	I=6 W=zheliangnian	148	J=62 S=1 E=15
10	I=7 W=shishishi	149	J=63 S=22 E=15
11	I=8 W=zhuzhuangchengzhang	150	J=64 S=82 E=15
12	I=9 W=xihuanwenxue	151	J=65 S=1 E=16
13	I=10 W=kandaozhuzi	152	J=66 S=22 E=16
14	I=11 W=natinghaode	153	J=67 S=82 E=16
15	I=12 W=yunayigong	154	J=68 S=1 E=17 l=200
16	I=13 W=xiaopengyou	155	J=69 S=22 E=17 l=200
17	I=14 W=xiaopaipai	156	J=70 S=82 E=17 l=200
18	I=15 W=xiaoxiangguan	157	J=71 S=1 E=18 l=200
19	I=16 W=lindaiyu	158	J=72 S=22 E=18 l=200
20	I=17 W=huaxiangniaoyu	159	J=73 S=82 E=18 l=200
21	I=18 W=shumufanmao	160	J=74 S=1 E=19 l=200
22	I=19 W=lvcaoruyin	161	J=75 S=22 E=19 l=200
23	I=20 W=jumuwangqu	162	J=76 S=82 E=19 l=200
24	I=21 W=silence	163	J=77 S=1 E=20 l=250
25	I=22 W=!NULL	164	J=78 S=22 E=20 l=250
26	I=23 W=o	165	J=79 S=82 E=20 l=250
27	I=24 W=ueN	166	J=80 S=1 E=21
28	I=25 W=ioNN	167	J=81 S=22 E=21
29	I=26 W=ou	168	J=82 S=82 E=21

figure 14 Récompense pour les 4 proverbes



----- Figures of Merit -----					
KeyWord:	#Hits	#FAs	#Actual	FOM	
jumuwangqu:	1	17	1	0.00	
lvcaoruyin:	1	182	1	0.00	
shumufanmao:	1	6	1	100.00	
huaxiangniaoyu:	1	0	1	100.00	
lindaiyu:	0	0	1	0.00	
xiaoxiangguan:	0	0	1	0.00	
xiaopaipai:	0	0	2	0.00	
xiaopengyou:	0	0	1	0.00	
yunayigong:	0	0	0	0.00	
natinghaode:	0	0	1	0.00	
kandaozhuzi:	0	0	1	0.00	
xihuanwenxue:	0	0	1	0.00	
zhuozhuangchengzhang:		0	0	1	0.00
shishishi:	0	0	1	0.00	
zheliangnian:	0	0	1	0.00	
biyuanlai:	0	0	1	0.00	
kenengyouxie:	0	0	1	0.00	
zhongmanzhuzi:	0	0	1	0.00	
tebiepiaoliang:	0	0	0	0.00	
tebiexingfu:	0	0	1	0.00	
Overall:	4	205	19	10.53	

figure 15 Détection des 4 proverbes avec récompense

Nous voyons que les proverbes *shumufanmao* et *huaxiangmiaoyu* ont été correctement reconnus, le taux de reconnaissance est 100%.

D'après les 3 expériences, nous pouvons conclure que le choix des mots-clés et l'occurrence des mots dans le corpus influencent le résultat de la détection. Néanmoins, dans l'emploi réel, afin d'obtenir le bon résultat, il faudrait bien entraîner des HMMs. Cependant, nous ne comprenons pas la règle ou la loi de l'ajout de récompense. Dans ces 3 expériences, nous avons d'abord ajouté la valeur de récompense par hasard, ensuite, modifié petit à petit la valeur. Ce processus était fatigant, nous n'avons pas trouvé le raccourci.

## 5. Conclusion

Dans ce projet, nous avons bien appris le principe de modèle de Markov caché, et les outils HTK. Nous avons d'abord élaboré des ressources, découpé le corpus en 2 minutes d'apprentissage et une minute de test. Il est tout à fait normal de rencontrer des problèmes pendant le travail. Par exemple les données de corpus ne sont pas suffisantes, ce qui a causé la difficulté de l'initialisation des HMMs. Ainsi, nous avons utilisé la convention SAMPA + pour fusionner le corpus avec les autres étudiants.

Ensuite, dans l'étape de l'apprentissage HMMs, grâce aux outils HTK, nous avons

généralisé les HMMs pour notre corpus.

Et puis, pour l'évaluation de notre système de reconnaissance, nous avons fait le décodage acoustico-phonétique du corpus de test, en calculant le taux d'erreur. La proportion 87,58% est insuffisante, c'est peut-être parce que, d'une part, bien que la convention SAMPA+ soit générée pour ce projet, elle manque encore quelques phonèmes fréquents en mandarin, par exemple la consomme [y] n'existe pas dans la convention, mais, elle apparaît souvent dans notre corpus. D'autre part, même si nous avons 8 minutes de l'apprentissage, le volume de corpus serait faible.

Finalement, les expériences de détection des mots-clés sont aussi appréciables. Nous avons détecté 3 types de mots-clés, les caractères d'une syllabe, les mots de 2 syllabes et les proverbes de 4 syllabes. Nous sommes contents de voir que les résultats se sont améliorés avec l'ajout de récompense.

En conclusion, notre système de reconnaissance a encore des espaces à s'améliorer. D'un côté, il faudrait compléter les données de corpus, le nombre de corpus, les minutes de l'audio et sa qualité. D'autre côté, l'amélioration de la convention SAMPA+, afin que plus de phonèmes présentent dans HMMs.

Pour finir ce rapport, nous tenons à remercier Nishan TANG, Zhijie ZHOU et surtout Can CUI. Ils ont partagé leurs scripts python, leurs corpus. CUI a répondu à nos questions avec sa gentillesse.