Etude Comparative Entre les Librairies de Reconnaissance Vocale

Hamza Fribia, Halima Bahi

الملخّــص

إنّ توافر الأدوات اللاّزمة للتعرّف على الكلام -مع تكونها من وحدات، وتوفرها مجانيا، وانفتاح مصدرها- سمح للباحثين والمبرمجين بتنفيذ واختبار خوارزميّات جديدة. هذه الميزات مكّنت من تطوير أنظمة قوية للتعرف الآليّ على الكلام. عمليّا، تمّ استخدام مكتبات مختلفة لإنشاء أنظمة التّعرّف على الكلام في عدة لغات، منها : «أش تي كا»، «سفنكس»، «كالدي»، «ماتلاب» و«جافا سبيتش»، إلخ. لكن ما هي المكتبة الأفضل من بين تلك المكتبات ؟ للإجابة عن هذا السّؤال عقدنا مقارنة بين المكتبات الأكثر استخداما : «أش تي كا»، «سفنكس» و «ماتلاب»، وتطبيقها على التّعرف على الكلمات المعزولة والجمل المتواصلة.

الكلمات المفاتيح : التّعرّف الآلي على الكلام، المكتبات، أش تي كا، سفنكس، مصفوفة الالتباس.

Etude Comparative entre les Librairies de Reconnaissance Vocale

Hamza.Frihia[1,2], Halima.Bahi[1,2]

[1] Laboratoire Gestion Electronique de Document (LabGED),
Université Badji Mokhar (UBMA), Annaba, Algérie.
[2]Département d'informatique, Université Badji Mokhar, Annaba, Algérie.
frihiahamza@yahoo.fr, bahi@labged.net

Résumé

La disponibilité des outils de reconnaissance de la parole avec leur architecture modulaire et leur open source permet aux chercheurs et programmeurs d'implémenter et tester de nouveaux algorithmes. Ces caractéristiques facilitent le développement de puissants systèmes de reconnaissance automatique de la parole (RAP).

En pratique, différentes boite à outils ont été utilisées pour la création des systèmes de reconnaissance vocale sur n'importe quelle langage, on cite : HTK, sphinx, Kaldi, ASR de Matlab et java speech… Pour répondre à la question : quelle est la meilleure librairie, nous voulons effectuer une comparaison entre les librairies les plus utilisé HTK et sphinx et Matlab appliqués pour la reconnaissance des mots isolés et des phrases continues, tout en transmettant notre expérience.

Keywords: Reconnaissance automatique de la parole, librairies, HTK, Sphinx, Matrice de confusion.

1. Introduction

Le monde de Reconnaissance Automatique de la Parole (RAP) est devenu très flexible et pratique grâce au progrès des boites à outils et leurs disponibilités qui a permis aux chercheurs d'avancer de manière très rapide ces dernières années. Les premières applications de reconnaissance automatique de la parole (RAP) étaient basées sur la reconnaissance de mots isolés (1970) alors que les applications actuelles s'orientent de plus en plus vers la parole spontanée en passant par la parole continue. Néanmoins, des problèmes restent à résoudre pour accroître la robustesse de ces systèmes et pour étendre leurs capacités de dialogue. Les outils de modélisation les plus utilisés dans le domaine de la reconnaissance vocale, sont les Modèles de Markov Cachés (MMC ou HMM pour Hidden Markov Models).

Dans ce travail, nous nous intéressons à l'étude des différents outils de construction des systèmes RAP basés sur les HMMs pour voir leur capacité à gérer des systèmes de grand vocabulaire. Pour cela, nous avons testé les librairies les plus utilisé dans le domaine de la RAP tels que la plateforme open source HTK (Hidden Markov Model ToolKit), Sphinx 4 et la toolbox de Matlab. Nous avons utilisé le corpus TIMIT et réalisé deux autres cor-

pus avec l'aide des étudiants du département d'informatique de notre université.

Nous présenterons dans cet article après l'introduction, la définition des HMMs, leurs éléments constitutifs et leurs utilisations dans la section 2. La section 3 décrit les librairies les plus utilisées pour construire les systèmes RAP. Une comparaison entre ces outils sera présentée dans la section 4. La section 5 présentera les résultats expérimentaux. Nous achèverons notre article par une conclusion.

2. Les HMMs

En intelligence artificielle, les Modèles de Markov Cachés représentent un outil statistique qui permet de modéliser des phénomènes stochastiques. Ils sont basés sur le modèle de Markov, sauf qu'on ne peut pas observer directement la séquence d'états (les états sont cachés) [1]. L'utilisation de ces modèles se fait en rapport avec le temps, à chaque unité de temps, on opère une transition, ce qui génère finalement une séquence d'états.

2.1. Utilisation

Les HMM sont des modèles statistiques très puissants qui trouvent leurs applications dans de multiples domaines. Nous avons déjà évoqué la reconnaissance vocale: il s'agit de faire correspondre une séquence de phonèmes (la séquence d'observations) avec une séquence de mots (la séquence d'états cachés) [2]. D'autres applications existent telles que : l'analyse cryptographique, la traduction automatique, la recherche de virus polymorphe, la reconnaissance d'écriture manuscrite, la bio-informatique, l'analyse financière et bien d'autres... De façon générale, un HMM permet de retrouver des informations cachées que l'on sait liées à des informations observables.

2.2. Les éléments d'un HMM

Un HMM est caractérisé par le quintuplet $\{\pi_i\ ,\ S,\ X,\ A, B\ \}$ où :

- π_i est la distribution de probabilité de l'état initial.
- S est l'ensemble d'états $S = \{s_1,\ s_2, \dots\ s_N\}$.
- X est l'ensemble des observations ou les symboles à émettre par les états $X = \{x_1, x_2, \dots\ x_M\}$.
- A est la probabilité de déplacement d'un état a un autre $S_i \rightarrow S_j$ avec $A_{ij} = P(S_j|S_i)$.
- B est la fonction de distribution de probabilité des observations X_k pour l'état j à l'instant t avec $B_t(k) = P(X_k|S_j)$.

2.3. Reconnaissance par un HMM

Supposons le HMM qui génère la séquence d'observations $O = O_1 O_2 \dots O_T$, le processus de reconnaissance se déroule comme suit :

- choisir $s_1 = S_i$ l'état initial selon la distribution initiale d'états π_i.
- Mettre *t=1*.
- Choisir $O_t = x_k$ selon la distribution de probabilité d'observation $B_t(k)$ à l'état S_i.
- le transit vers un nouveau état $s_{t+1} = S_j$ selon la distribution de probabilité de transition A_{ij} pour l'état S_i.
- Si $t < T$ alors $t = t + 1$ et aller à l'étape 3 sinon terminer.

3. Librairies

Il existe plusieurs librairies pour le domaine de RAP, parmi ces librairies : HTK, sphinx, Kaldi, ASR de Matlab, java speech, ISIP. Dans ce travail, nous nous intéressons à trois (3) librairies qui sont les plus utilisées: HTK et Sphinx et le Toolbox de Matlab.

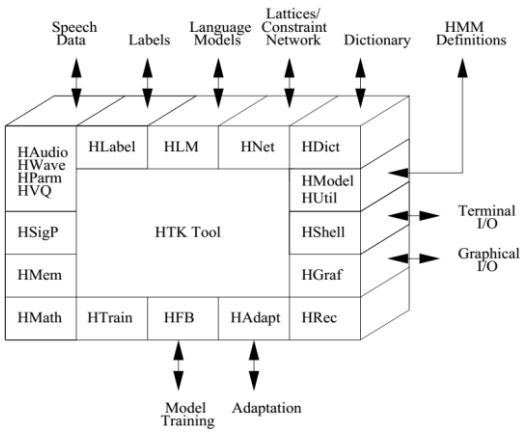

Figure 1: Architecture de HTK [3]

3.1. HTK

Hidden Markov Model Toolbox (HTK) a été développé par l'université de Cambridge. Cette boite à outils dédiée aux Modèles de Markov Cachés est principalement utilisée pour la reconnaissance de la parole. Elle se compose d'un ensemble de modules et d'outils disponibles gratuitement et téléchargeable à partir du site. HTK est implémenté en langage C et il s'exécute en ligne de commande, il est capable de mettre en œuvre un grand vocabulaire, indépendamment du locuteur et est applicable sur n'importe quelle langue. La documentation sur HTK est très riche avec des exemples pratiques (vers 300 pages) sur http://htk.eng.cam.ac.uk/.

L'outil HTK comprend trois phases principales que l'on peut appliquer sur un HMM. D'abord, la phase de préparation données qui vise à enregistrer, étiqueter et segmenter des données d'apprentissage en utilisant l'outil HSLab. Ensuite, l'extraction des vecteurs de caractéristiques après la configuration des paramètres avec l'outil HCopy. La phase d'apprentissage sert à créer les modèles acoustiques qui représentent les vecteurs de caractéristiques. Durant l'apprentissage, les vecteurs sont d'abord utilisés pour l'initialisation des paramètres des HMMs en utilisant l'outil HInit et HCompV, car les paramètres de l'HMM doivent être correctement initialisés. Ensuite, Hrest est l'outil qui permet de réestimer les paramètres du modèle HMM (c'est l'implémentation de l'algorithme Baum-Welch). Le processus de reconnaissance se fait avec l'outil HVite. Il existe d'autres outils dans HTK qui paraissent être intéressants comme l'outil de calcul du taux d'erreur et le test des performances du système. La figure ci-dessus montre l'architecture de l'outil HTK [3].

HTK est distingué pour l'utilisation des HMMs, il ne permet pas de faire des combinaisons ou d'hybridations des HMMs avec d'autres classifieurs. En dépit de la richesse de sa documentation, nous avons remarqué que quelques outils dans HTK ont moins d'information du point de vue pratique.

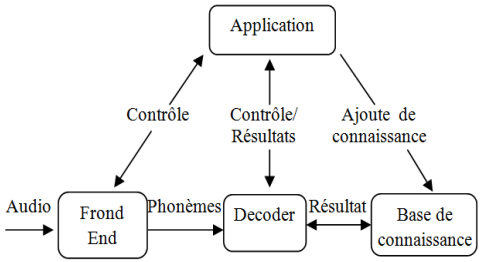

Figure 2: Architecture d'application du Sphinx4 (d'après [5])

3.2. Sphinx

Sphinx est une librairie de reconnaissance vocale gratuitement téléchargeable, avec la possibilité de modifier le code source, il a la capacité d'implémenter des systèmes avec un large vocabulaire, indépendant du locuteur [4]. Les premières versions de Sphinx (1, 2 et 3) sont écrites en langage C, mais la version récente du sphinx 4 est écrite en java. Sphinx 1, 2, 3 et 4 sont des décodeurs, l'outil d'apprentissage s'appelle SphinxTrain. Sphinx 4 est très souple dans sa configuration, avec une architecture modulaire qui permet aux programmeurs et chercheurs de tester de nouveaux algorithmes. La figure 2 représente l'architecture du Sphinx 4 qui s'articule autour de trois modules principaux [5]. L'outil de création des modèles acoustiques SphinxTrain est écrit en langage C et l'outil de reconnaissance Sphinx 4 est écrit en java. La documentation sur Sphinx n'est pas assez riche par rapport à HTK et il n'existe pas assez d'exemples sur l'outil de création des modèles acoustiques « SphinxTrain ».

Le tableau comparatif (Table .1) qui est issu d'une expérience de l'institut HPI (Hasso Plattner Institut) de l'Université de Potsdam [6] montre que les performances de Sphinx4 sont meilleures que HTK.

Des expériences montrent que la qualité des modèles acoustiques créés par Sphinx est meilleure que celle de HTK [7], si on ne prend en considération que la base TIMIT.

Critère	Sphinx 4	HTK
Taux de reconnaissance (60% de score total)	6.5	6
Indépendance (15%)	8	8
Coût (5%)	10	7
Modularité (15%)	10	0
Actualité (5%)	7	6
Score total	7.45	6.35

Table 1. Comparaison entre les logiciels de RAP [6]

3.3. Matlab

Matlab inclut une boite à outil (toolbox) incluant des algorithmes d'apprentissage artificiel basé sur les modèles de Markov cachés et des algorithmes de détection des séquences temporelles hors ligne et en ligne (http://Pudn.com).

4. Comparaison

Nous avons effectué une étude comparative entre les librairies HTK, Sphinx et Matlab. Le tableau suivant montre les différences entre ces librairies (Table 2.).

Critères	HTK	Sphinx	Matlab
Structure	un système complexe	bien structuré (modulaire)	Assez Simple
MAJ	Mise à jour régulière	Mise à jour régulière	Mise à jour régulière
Documentation	Il est bien documenté à la fois théorique et pratique.	a documentation de ces systèmes est relativement pauvre	Documentation à la fois théorique et pratique (Pudn.com)
Langage	Langage C	Java	Langage C
Traitement du signal	Banc de filtres, MFCC, LPC, PLP, LPreflexC, ClpC, IREFC et MELSPEC.	MFCC, PLP, spectre.	MFCC, Utilise HTK pour extraire des caractéristiques (MLF) [8]
Formats d'entrée	WAV, TIMIT, NIST, OIG, AIFI	WAV, MP3.	WAV, FLV
Apprentissage	HRest / HERest	SphinxTrain	ForwardBackwardAnalyses[8], Vit-reestim
Décodeur	HVite	Sphinx 4	Viterbi_log

Table 2. Tableau comparatif entre HTK, Sphinx et Matlab

5. Expérimentation

Pour mener nos expériences, sous avons demandé à un groupe d'étudiants (8 étudiants) de prononcer cinq mots. Les données obtenues sont 40 échantillons de format wav. Ces données sont utilisées pour tester les trois librairies HTK, Sphinx et Matlab. Donc, nous avons réalisé trois systèmes de RAP. Pour le premier système nous avons utilisé la librairie HTK avec un corpus contenant des fichiers son qui sont des mots anglais prononcé par des étudiants Algériens. Pour le deuxième système nous avons utilisé le toolbox Matlab avec la base précédente.

Le troisième système a été créé avec Sphinx4 en utilisant les modèles acoustiques créé sur la base TIMIT (contient des mots prononcés par des personnes anglaises).

Pour la préparation des données nous avons extrait les coefficients MFCC et leurs dérivées du signal .wav (39 coefficients). L'apprentissage se fait à l'aide de l'algorithme Boum-Welch, le nombre de gaussiennes est 8, avec tied-state.

	Children	Good morning	School	Teacher	Welcome
Children	**7**	0	0	0	1
Good morning	0	**8**	0	0	0
School	1	0	**6**	0	1
Teacher	0	0	0	**7**	1
Welcome	1	0	0	0	**7**

Table 3. Table de confusion du système RAP construit par Matlab

	Children	Good morning	School	Teacher	Welcome
Children	**8**	0	0	0	0
Good morning	0	**8**	0	0	0
School	0	0	**8**	0	0
Teacher	0	0	0	**8**	0
Welcome	0	0	0	0	**8**

Table 4. Table de confusion du système construit par HTK

	Children	Good morning	School	Teacher	Welcome
Children	**6**	0	0	2	0
Good morning	1	**7**	0	0	0
School	0	0	**5**	3	0
Teacher	3	0	0	**5**	0
Welcome	0	0	0	2	**6**

Table 5. Table de confusion du système construit par Sphinx 4

Enfin la phase de reconnaissance est réalisée avec l'algorithme de Viterbi.

Le résultat de l'étape de reconnaissance est les matrices de confusions et les précisions qui sont montrés ci-dessus (Tables 3, 4 et 5).

Le tableau 6 présente le taux de reconnaissance pour chaque système.

Le tableau ci-dessus (Table .6) montre que les performances d'un système de reconnaissance de la parole sont fortement liées à l'accent des locuteurs (la langue maternelle) en phase d'apprentissage et également à la qualité des modèles construits.

	Taux de reconnaissance des systèmes créés par :		
	Sphinx	**HTK**	**Matlab**
Corpus test prononcé par des locuteurs anglais	100 %	20 %	0 %
Corpus test prononcé par des locuteurs arabes	40 %	100 %	80 %

Table 6. Les taux de reconnaissance pour chaque système par rapport au corpus

Le taux de reconnaissance de Sphinx4 est imbattable dans le cas où le corpus de test comprend des mots prononcés par des Anglais, ceci peut s'expliquer par les modèles qui sont construit à partir de la base TIMIT, qui est mondialement connue. HTK est meilleur si le corpus est celui des Algériens car les modèles construit sont basés sur des prononciations Algériennes. Pour la Toolbox Matlab le taux de reconnaissance est 80% sur le corpus des étudiants Algériens, par contre pour la base des Anglais le taux de reconnaissance est 0 % parce que les modèles construites sous Matlab sur un corpus prononcé par des locuteurs arabes et testé par le corpus des anglais.
Nous remarquons que la précision du système de HTK est meilleure de celle de sphinx et Matlab. Ceci peut s'expliquer par une meilleure modélisation et donc un apprentissage plus précis (Table .7).

	Précision
HTK	1
Matlab	0,875
Sphinx	0,725

Table 7. Les précisions des systèmes.

6. Conclusions

Les recherches dans le domaine de la reconnaissance de la parole sont devenues moins difficiles grâce à la disponibilité des librairies gratuites et open source. Dans ce papier, nous avons vu les différences entre les librairies les plus utilisées dans le RAP. Nous avons réalisé trois applications avec HTK, Sphinx et Matlab pour voir les performances de reconnaissance pour chaque librairie.

7. References

[1] Rabiner, L., "A tutorial on hidden Markov models and selected applications in speech recognition", Proceedings of the IEEE, 77, 1989.

[2] Institut Electronique et Informatique Gaspard-Monge (IGM) http://www-igm.univ-mlv.fr/.

[3] Young, S., Evermann, G., Kershaw, D., Moore, D., Odell, J., Ollason, D., Valtchev, V. and Woodland, P., "The HTK Book", Cambridge University Engineering Dept., 2001.

[4] CMUSphinx, Open Source Toolkit For Speech Recognition, Project By CMU, "Sphinx-4 Application Programmer's Guide", Online: http://cmusphinx.sourceforge.net/wiki/tutorialsphinx4

[5] Lamere, P., Wok, K., Walker, W., Gouvea, E., Singh, R., Raj, B. and Wolf, P., "Design of the CMU Sphinx-4 decoder", Proceedings of the 8th European Conference on Speech Communication and Technology, Geneve, Switzerland, pp. 1181–1184, Sept. 2003.

[6] Yang, H.J., Oehlke, C. and Meinel, C., "German speech recognition: A solution for the analysis and processing of lecture recordings", HPI. In Proc. Of 10th IEEE/ACIS, Sanya, Heinan Island, China, 2011.

[7] Samu, K. and Barot, M., "A comparison of public domain software tools for speech recognition", Workshop on Spoken Language Processing. In ISCA-Supported Event., Mumbai, India, 2003.

[8] Jang, J-S. R., "ASR (Automatic Speech Recognition) Toolbox", available from the link at the author's homepage at "http://mirlab.org/jang".