Reconnaissance de chiffres isolés embarquée dans un téléphone portable

Christophe Lévy[1,2]*, Georges Linarès*[1]*, Pascal Nocera*[1]*, Jean-François Bonastre*[1]

[1]Laboratoire Informatique d'Avignon, Avignon, France
[2]Stepmind SA, Le Cannet, France
{christophe.levy, georges.linares, pascal.nocera, jean-francois.bonastre}@lia.univ-avignon.fr

ABSTRACT

This paper is focused on cellular phone embedded speech recognition. We present several methods able to fit speech recognition system requirements to cellular phone resource. The proposed techniques are evaluated on a digit recognition task using both French and English corpora.
Several parameterization algorithms (LPCC, MFCC, PLP) are compared to the LPC included in the GSM norm. The MFCC and PLP parameterization algorithms perform significantly better than the other ones.
In order to achieve good performance with reasonable resource needs, we develop several methods to embed classical HMM-based speech recognition system in cellular phone. We first propose an automatic on-line building of phonetic lexicon which allows a minimal but unlimited lexicon. Then we reduce the HMM model complexity by decreasing the number of components per state.
Finally, we evaluate our propositions by comparing Dynamic Time Warping (DTW) with our HMM system. The experiments show that our HMM system outperforms DTW for speaker independent task.

1. INTRODUCTION

Le niveau de performance atteint par les systèmes de dictée vocale à grand vocabulaire est relativement élevé, mais ces systèmes nécessitent de grandes capacités de mémoire et de calcul. Les meilleurs systèmes utilisent notamment des modèles acoustiques composés de plusieurs millions de paramètres. Les temps de décodage deviennent relativement importants (plus de 10 fois le temps réel sur un ordinateur standard). L'intégration d'un système de reconnaissance automatique de la parole (RAP) dans un téléphone portable nécessite de limiter fortement la consommation de ressources (tant en terme de complexité qu'en terme de taille des modèles). Dans cet article, nous présentons plusieurs approches pour l'adaptation des systèmes de RAP aux ressources offertes par les téléphones portables.

De nombreux codec GSM sont basés sur l'algorithme LPC[1](notamment les codecs EFR[2] ou AMR[3]). L'utilisation de cet algorithme permet de limiter les ressources nécessaires (tout autre analyse impliquerait du développement supplémentaire). Nous comparons ce type de paramétrisation avec différentes solutions utilisées généralement en reconnaissance de la parole : les coefficients MFCC[4] et les coefficients PLP[5]. Nous étudions également l'influence de la taille du vecteur acoustique sur les systèmes de reconnaissance dans un contexte réaliste (mots isolés et petit vocabulaire).

Ensuite, nous présentons une solution pour réduire la complexité des HMM. Nous étudions notamment l'évolution du taux d'erreur par mot en fonction de l'occupation mémoire des modèles.

Enfin, nous comparons deux approches classiques de décodage pour la reconnaissance de mots isolés : la DTW (Dynamic Time Warping - [3]) et les HMM (Hidden Markov Model - [9]).

2. CONTEXTE D'EXPLOITATION DES TÉLÉPHONES PORTABLES

Si les téléphones portables de première génération ne proposaient que peu de services (tels que la mémorisation de quelques numéros de téléphones), les nouvelles générations fournissent, pour leur part, beaucoup plus de fonctionnalités. Elles permettent de télécharger des sonneries, de jouer, de gérer un agenda ... Ces nouvelles fonctionnalités engendrent souvent une interface complexe. Ces contraintes ergonomiques, amplifiées par la miniaturisation des téléphones, expliquent que le "name dialing"[6], la reconnaissance de chiffres ou le pilotage par la voix deviennent standards. Même si les dernières générations de téléphones offrent des performances (aussi bien en calcul qu'en espace mémoire) nettement supérieures à celles offertes par le passé, les ressources disponibles restent néanmoins bien inférieures à celles nécessaires pour les systèmes de reconnaissance de la parole classiques.

Actuellement, une puce de téléphone contient :
– quelques kO de mémoire[7],
– un processeur avec une fréquence proche de 50 MHz,
– un DSP (Digital Signal Processor), lui aussi, avec une fréquence avoisinant les 50 MHz.

Les systèmes de reconnaissance de RAP utilisent généralement un lexique composé de plusieurs dizaines de millier de mots. De tels lexiques ne peuvent pas être embarqués dans un téléphone portable. L'application visée étant le "name dialing", le système doit être capable de reconnaître un nom, un prénom, un nom de lieu, etc. La nature du vocabulaire (noms propres) ajoute une contrainte supplémentaire au lexique : en fonction des noms choisis

[1]Linear Predictive Codec - [12]
[2]Enhanced Full Rate - [2]
[3]Adaptive Multi-Rate - [1]
[4]Mel Frequency Cepstral Coefficients - [5]
[5]Perceptual Linear Predictive - [7]
[6]reconnaissance vocale de nom ou prénom
[7]la puce développée par Stepmind SA dispose de moins de 10 kO de ROM.

par l'utilisateur le lexique doit pouvoir être étendu.

Enfin, d'un point de vue ergonomique, la phase d'entrainement du système doit être la plus courte possible ; nous utilisons une seule répétition de chaque mot.

3. BASES DE DONNÉES ET PROTOCOLES EXPÉRIMENTAUX

Comme précisé dans dans la section 2, l'application majeure de la RAP embarquée est le "name dialing". Cependant, devant des contraintes de disponibilité de corpus[8] les expériences ont été réalisées sur une tâche de reconnaissance de chiffres isolés.

De plus, afin d'être cohérent avec la contrainte d'ergonomie (décrite dans le chapitre 2) nous avons choisi un mode dépendant du locuteur où les modèles sont appris avec une seule répétition de chaque chiffre. Nous utilisons deux corpus :

- le premier est le sous-ensemble du corpus français BDSONS [4] contenant les chiffres isolés. Dans un premier temps, nous avons subdivisé ce nouvel ensemble en un ensemble de test et un ensemble d'adaptation. Le premier est composé de 800 occurrences de chiffres prononcés par 16 locuteurs (5 répétitions de chacun des 10 chiffres par locuteur). Le second est composé de 1400 chiffres prononcés par 14 locuteurs, différents des précédents (10 répétitions de chaque chiffre par locuteur). Ce second ensemble est utilisé pour adapter le HMM indépendant du locuteur (section 5.1).
- le second est un corpus anglais : TI_DIGITS [8]. Il est composé d'enregistrements de 225 locuteurs, répartis en deux sous-ensembles : un ensemble d'apprentissage (112 locuteurs) et un de test (113 locuteurs). Chaque locuteur a prononcé deux fois chacun des onze chiffres (de 1 à 9, plus "oh" et "zero"). Le sous-ensemble d'apprentissage (respectivement de test) contient donc 2464 occurrences de chiffres (respectivement, 2486 occurrences de chiffres).

L'occurrence y (0-4 pour BDSONS ou 0-1 pour TI_DIGITS) du chiffre z (0-9 pour BDSONS ou 0a, 0b, 1-9 pour TI_DIGITS) prononcée par le locuteur x est notée $L_xU_yD_z$. Une référence, pour un chiffre donné, est construite en utilisant une seule répétition. Nous avons défini 3 protocoles expérimentaux :

- le protocole "user" : seules les occurrences de test prononcées par le locuteur d'apprentissage ont été utilisées. Ce protocole correspond au fonctionnement classique d'un téléphone portable (le propriétaire et l'utilisateur ne sont qu'une seule et même personne). Avec le corpus BDSONS, nous avons simulé 80 pseudo-locuteurs en apprenant une référence à l'aide d'une seule occurrence et en utilisant les 4 autres occurrences disponibles pour faire les tests. En répétant ce principe pour chacune des 5 occurrences, cette méthode permet de réaliser 3200 tests. En résumé, pour l'apprentissage, nous avons utilisé les fichiers $L_xU_{y1}D_{0-9}$ et les fichiers $L_xU_{y2}D_{0-9}$ (avec $y2$ différent de $y1$) pour les tests. Ce protocole n'a pas été utilisé avec le corpus TI_DIGITS.
- le protocole "other" : nous avons utilisé pour le test uniquement les occurrences de chiffres prononcées par un locuteur différent du locuteur d'apprentissage. Ce protocole simule la situation dans laquelle l'utilisateur du portable n'est pas la personne qui a participé à la phase de construction du lexique. Nous avons simulé 80 pseudo-locuteurs en utilisant la même approche que précédemment, pour le protocole BDSONS. Toutes les occurrences différentes de celles utilisées pour l'apprentissage sont utilisées pour les tests. Ce qui donne 60000 tests (80 pseudo-locuteurs * 15 autres locuteurs * 5 occurrences * 10 chiffres). Pour TI_DIGITS, nous avons 224 pseudo-locuteurs (112 véritables locuteurs et 2 répétitions de chacun des 11 chiffres). Ce qui donne 556864 tests de reconnaissance en utilisant les 2486 occurrences du sous-ensemble de test.
- le protocole "all" : toutes les occurrences différentes de celles utilisées pour l'apprentissage sont utilisées pour le test qu'elles proviennent du locuteur d'apprentissage ou non. Ce qui donne 63200 tests pour BDSONS. Ce protocole n'a pu être utilisé pour TI_DIGITS étant donné que les locuteurs du sous ensemble de test ne sont pas les mêmes que ceux du sous ensemble d'apprentissage.

L'efficacité de chacune des méthodes est estimée en utilisant le taux d'erreur par mot (Word Error Rate - WER) tel que défini par [10].

4. PARAMÉTRISATION

En téléphonie numérique fixe, le codage PCM nécessite un taux de transfert de 56/64kO (8KHz x 7/8 bits). Pour la téléphonie mobile, ce taux s'avère être trop élevé ; de nouveaux codecs ont donc été développés. Parmi ceux basés sur les LPC, le débit varie entre 12,2kO/s (EFR) et 4,75kO/s (AMR plus bas débit). L'utilisation des LPC pour la RAP évite l'implementation d'une nouvelle analyse et donc nécessite moins de ressources mémoire[9].

Pour évaluer la viabilité de cette solution, nous avons comparé différents algorithmes de paramétrisation (LPC, LPCC[10], MFCC et PLP) dans le cadre spécifique de la reconnaissance embarquée dans le téléphone. Les 12 premiers coefficients auxquels on ajoute l'energie (soit 13 coefficients) ont été utilisés pour le décodage (sans les delta ni les delta-delta). Nous présentons également une version plus économe, basée sur les PLP, ne contenant pas 13 mais seulement 6 coefficients (5 + l'énergie), appelée PLP6.

Les LPCC sont calculés à partir des coefficients LPC. Ceci permet de minimiser l'accroissement du coût de calcul, car ils sont obtenus par une simple recursion (*cf.* équation 1).

$$Eq.1 : LPCC_i = -LPC_i + \frac{1}{i}\sum_{k=1}^{i-1}(i-k)LPC_k LPCC_{i-1}$$

où LPC_i (respectivement LPC_k) est le $i^{ème}$ coefficient (respectivement $k^{ème}$ coefficient) issu des coefficients LPC et où $LPCC_{i-1}$ représente le $(i-1)^{ème}$ coefficient cepstral.

Toutes les expériences ont été réalisées avec un système de reconnaissance basé sur la DTW. La distance utilisée pour ces tests est la distance euclidienne. Cette dernière a été choisie pour sa simplicité bien que [11] ai montré qu'elle n'est pas la plus performante.

4.1. Résultats

Les résultats (*cf.* tableau 1) montrent que les coefficients LPCC (les coefficients cepstraux issus d'un codage LPC) sont beaucoup plus performants que les coefficients LPC classiques. Le WER (taux d'erreur par mot) passe de 11%

[8] nous ne disposons pas d'un corpus de noms propres adapté

[9] les méthodes classiques de paramétrisation pour la reconnaissance de la parole ne sont pas incluses sur les téléphones portable

[10] Linear Predictive Cepstrum Coefficient

à 4,8% pour le protocole "user", appliqué au corpus BDSONS (sans MSR). Le gain est similaire pour le corpus TI_DIGITS ainsi que pour les expériences avec la normalisation des paramètres (avec MSR).

Les paramétrisations basées sur une analyse en banc de filtres du signal (telles que MFCC et PLP) offrent de bien meilleures performances. Le WER est toujours inférieur à 0,3% avec le protocole "user", ce qui est largement en deçà des 11% et 4,8% obtenus avec les paramétrisations basées sur le LPC. Néanmoins, ces méthodes de paramétrisation sont complexes et nécessitent plus de ressources. La normalisation des paramètres (MSR) améliore significativement les performances avec les protocoles "all" et "other", notamment lors de l'utilisation d'une paramétrisation MFCC (une diminution du WER de 41% à 27% avec le corpus TI_DIGITS et de 36% à 15,6% avec le corpus BDSONS). Par contre, on constate une légère dégradation avec le protocole "user".

Pour finir, il est intéressant de noter que le WER obtenu en utilisant la paramétrisation compacte (PLP6) est semblable à celui obtenu avec les 13 coefficients.

TAB. 1: TAUX D'ERREUR PAR MOT (WER) D'UN RECONNAISSEUR BASÉ SUR UNE DTW EN UTILISANT DIFFÉRENTES MÉTHODES DE PARAMÉTRISATION DE SIGNAL : LPC, LPC avec soustraction de la moyenne et réduction de la variance (LPC MSR), LPCC, LPCC MSR, MFCC, MFCC MSR, PLP, PLP MSR, compacte PLP (PLP6) et compacte PLP avec MSR (PLP6 MSR)

	BD		TI
	"user"	"all"	"other"
LPC	11.00%	53.84%	65.81%
LPC MSR	14.56%	58.90%	67.51%
LPCC	4.88%	35.29%	42.54%
LPCC MSR	5.19%	25.27%	34.35%
MFCC	0.19%	36.03%	41.66%
MFCC MSR	0.31%	15.60%	27.24%
PLP	0.12%	26.08%	36.29%
PLP MSR	0.28%	23.09%	29.34%
PLP6	0.69%	23.29%	28.09%
PLP6 MSR	0.91%	17.40%	24.83%

5. RÉDUCTION DES RESSOURCES NÉCESSAIRES POUR LES HMM

Dans la littérature, on trouve deux algorithmes principaux pour la reconnaissance de chiffres isolés : le premier suit le principe d'un calcul de coût de distorsion entre deux formes (la DTW), alors que le second calcule la vraisemblance pour qu'une suite d'observations soit produite par un modèle (HMM). La DTW est réputée pour son très bon rapport performance/ressources. Les systèmes à base de HMM sont, eux, reconnus pour leurs performances et leurs facultés d'adaptation. Malheureusement, ils sont très souvent composés de modèles acoustiques comlexes.

Avant de comparer les deux approches dans le cas très précis de la reconnaissance embarquée dans un téléphone mobile, deux techniques de minimisation de la taille des modèles HMM sont présentées.

5.1. Conditions expérimentales

Dans cette section, les protocoles présentés auparavant (*cf.* section 3) sont utilisés avec le corpus BDSONS. Le signal est paramétrisé avec 12 coefficients MFCC plus l'énergie pour chaque trame (20 ms), ensuite une normalisation MSR (soustraction de la moyenne et réduction de la variance) est appliquée. La stratégie de décodage est basée sur l'algorithme de Viterbi.

Les HMM sont classiques : HMM gauche-droit, indépendant du contexte et avec 3 états émetteurs. Les modèles de phonème (non contextuels) sont appris en 3 étapes :
- Premièrement, les modèles sont appris grâce au corpus Français BREF120. Ce corpus contient environ 40 heures de parole prononcées par 120 locuteurs. Cette phase d'entrainement se déroule en utilisant l'algorithme EM et en optimisant le critère du Maximum de Vraissemblance.
- Ensuite, les modèles sont adaptés au corpus de test et à la tâche (BDSONS et reconnaissance de chiffre) en utilisant un ensemble de 1400 chiffres (*cf.* 3). Cette adaptation est faite en utilisant MAP (Maximum A Posterior [6]). Cette étape produit des modèles indépendants du locuteur.
- Enfin, une deuxième adaptation (toujours avec MAP) est effectuée pour obtenir les modèles dépendants du locuteur. Cette seconde adaptation est faite avec une seule occurrence de chacun des 10 chiffres.

5.2. Réduction du coût des HMM

Embarquer un système de reconnaissance à base de HMM dans un téléphone portable nécessite de drastiques réductions tant en terme de mémoire qu'en terme de puissance de calcul. Nous proposons deux techniques pour réduire ces coûts :
- Généralement le dictionnaire utilisé dans un système de reconnaissance de parole contient l'ensemble des mots connus, soit plusieurs dizaines de milliers de mots. Pour réduire la taille du dictionnaire sans limiter les applications, nous proposons l'utilisation d'un dictionnaire dynamique. Ceci permet à l'utilisateur de choisir ses propres mots (comme des noms propres). Le dictionnaire est automatiquement construit grâce à un décodage acoustico-phonétique qui fourni la transcription phonétique d'un mot.
- Afin de respecter les contraintes des téléphones, nous avons réduit la complexité des modèles. Nous avons étudié la corrélation entre le nombre de gaussiennes par état du HMM et le taux d'erreur du système (de 1 à 128 gaussiennes par état).

Résultats du lexique dynamique : Le tableau 2 présente les résultats obtenus en utilisant le lexique dynamique. Les performances sont comparées, en terme de WER, à celles obtenues avec un système plus classique, à lexique statique. Le lexique dynamique permet un gain significatif d'un point de vue de l'utilisation de l'espace mémoire et d'un point de vue des fonctionnalités envisageables. En effet, l'utilisation d'un lexique dynamique, pour la reconnaissance de noms/prénoms s'avère quasi indispensable tant le nombre de noms/prénoms est important.

TAB. 2: TAUX D'ERREUR AVEC LE CORPUS BDSONS EN UTILISANT UN LEXIQUE STATIQUE ET UN LEXIQUE DYNAMIQUE : 16 gaussiennes par état, 3200 tests pour "user", 60000 pour "other" et 63200 pour "all"

	user	other	all
Lexique statique	1.38%	5.38%	5.18%
Lexique dynamique	1.88%	6.71%	6.46%

Résultats de la diminution de la taille des modèles : Afin d'étudier l'influence de la taille des modèles sur le taux d'erreurs, 5 configurations (de 128 gaussiennes par état à 1 gaussienne par état) ont été testées. Les résultats (*cf.* tableau 3) montrent, comme on pouvait s'y attendre,

que la diminution du nombre de gaussiennes entraîne une augmentation du taux d'erreur. Il faut tout de même noter que l'augmentation du taux reste faible (moins de 0,2% avec le protocole "user") entre les modèles très complexes (128 gaussiennes) et les modèles à complexité moyenne (16 gaussiennes). Ceci est du au peu de données d'adaptation dont on dispose (1400 chiffres pour la première adaptation et seulement une occurrence de chacun des 10 chiffres pour la seconde).

Le HMM avec 16 gaussiennes par état représente le compromis le plus intéressant pour l'application visée.

TAB. 3: TAUX D'ERREUR D'UN SYSTÈME À BASE DE HMM AVEC DIFFÉRENTS NOMBRES DE GAUSSIENNES PAR ÉTAT : tous les tests sont réalisés avec un lexique dynamique.

	user	other	all	taille du modèle (en kO)
128g. / etat	1,88%	5,88%	5,68%	360
64g. / etat	1,66%	6,84%	6,58%	180
16g. / etat	1,88%	6,71%	6,46%	45
4g. / etat	12,41%	35,60%	34,43%	11
1g. / etat	21,09%	74,56%	71,85%	3

5.3. DTW vs. HMM

En situation mono-locuteur (protocole "user"), les systèmes basés sur la DTW obtiennent de meilleurs résultats (0,31% de taux d'erreur avec une paramétrisation MFCC MSR - *cf.* Table 1) que le système HMM (entre 1,66 % et 21,09% suivant la taille du modèle - *cf.* Table 3).

Avec le protocole "all", on s'aperçoit que les systèmes basés sur les HMM deviennent beaucoup plus performants (avec des modèles de taille raisonnable - jusqu'à 16 gaussiennes par état) que ceux à base de DTW. En effet, pour les premiers systèmes le taux d'erreur se trouve aux alentours de 6,5%, alors qu'il dépasse les 15% pour les seconds.

En terme de temps de calcul nécessaire à chacun des systèmes (*cf.* tableau 4), les deux approches semblent relativement proches lorsque la taille des modèles est petite. Pour des modèles de taille inférieure ou égale à 16 gaussiennes par état il faut moins de 0,4 fois le temps réel pour décoder (contre 0,3 pour la DTW).

TAB. 4: TEMPS NÉCESSAIRE POUR DÉCODER 1 SECONDE DE SIGNAL : les résultats sont donnés pour les systèmes HMM (entre 1 et 128 gaussiennes par état) et pour un système DTW

	HMM sys.					DTW
	1g	4g	16g	64g	128g	sys.
temps (s.)	0.30	0.31	0.40	0.79	1.33	0.31

6. CONCLUSION

Dans ce papier, nous nous sommes concentrés sur la reconnaissance de la parole embarquée dans un téléphone portable. Dans ce contexte, le système de reconnaissance doit respecter certaines contraintes telles que la place mémoire et les ressources de calcul.

Nous avons montré que les coefficients cepstraux LPC (LPCC) issus des LPC (par récursion) permettent un bon rapport performance/coût (calcul et mémoire). Néanmoins, les paramétrisations basées sur une analyse en banc de filtres (MFCC, PLP) restent plus performantes.

Nous avons aussi proposé deux approches pour réduire les ressources nécessaires aux systèmes à base de HMM. Pour cela, nous avons présenté un dictionnaire dynamique. Celui-ci permet de réduire considérablement la taille du lexique, d'apporter plus de souplesse et cela sans provoquer d'augmentation significative du taux d'erreur. Enfin, nous avons montré que le nombre de gaussiennes par état d'un HMM peut-être diminué de manière importante (jusqu'à 16 gaussiennes) tout en conservant un niveau de performance acceptable.

Pour finir, nous avons présenté un modèle très compact (moins de 45kO) et ne nécessitant pas plus de ressources d'un point de vue calcul qu'une DTW. Ce système très compact préserve un bon niveau de performance et autorise un grand champ d'application dans le domaine des interfaces homme/téléphone portable.

RÉFÉRENCES

[1] Adaptive multi-rate (amr) speech transcoding (gsm 06.90 version 7.2.1). 2000.

[2] Enhanced full rate (efr) speech transcoding (gsm 06.60 version 8.0.1). 2000.

[3] R. Bellman. *Dynamic Programming*. Princeton University Press, 1957.

[4] R. Carr, R. Descout, M. Esknazi, J. Mariani, and M. Rossi. The french language database : defining, planning and recording a large database. In *proceeding of ICASSP*, San Diego, 1984.

[5] S.B. Davis and P. Mermelstein. Comparison of parametric representations for monosyllabic word recognition in continuously spoken sentences. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 28(4) :357–366, August 1980.

[6] J.L. Gauvain and C.H. Lee. Maximum a posteriori estimation for multi-variante gaussian mixture observations of markov chains. *IEEE transactions on speech and audio processing*, 2 :291–298, 1994.

[7] H. Hermansky. Perceptual linear predictive (plp) analysis of speech. *Journal of Acoustical Society of America*, 87(4) :1738–1752, April 1990.

[8] R.G. Leonard. A database for speaker-independent digit recognition. In *proceedings of ICASSP*, volume 3, San Diego, 1984.

[9] L.R. Rabiner. A tutorial on hidden markov models and selected applications in speech recognition. *IEEE transactions Speech Audio Processing*, 77(2) :257–285, february 1989.

[10] C. Helmer Strik. Comparing the recognition performance of csrs : in search of an adequate metric and statistical significance test.

[11] Y. Stylianou and A.K. Syrdal. Perceptual and objective detection of discontinuities in concatenative speech synthesis. In *proceedings of ICASSP*, Salt Lake City, 2001.

[12] T.E. Tremain. The government standard linear predictive coding algorithm : Lpc10. *Speech Technology*, 1(2) :40–49, april 1982.