Université Paris-Sud (XI), Paris, France

Thèse no: 2803

Recherche des N Meilleures Solutions et Post-Traitements, en Reconnaissance de la Parole

Par

Mohamed Nabil Lokbani

Thèse préparée à : CNET-Lannion (France)

Soutenue le 9 septembre 1993 ; devant le jury:

M. M. Baudry (Président)

Mme R. Andre-Obrecht (Examinateurs externes ou Rapporteurs)
M. B. Merialdo

M. J.S. Lienard (Examinateurs)
M. J.L. Gauvain
M. B. Delyon
M. D. Jouvet

Résumé

Cette thèse porte sur l'étude d'une méthode de recherche des *N* meilleures solutions, puis son adaptation au système de reconnaissance automatique de la parole du CNET et sur le développement de post-traitements segmentaux ou syntaxiques pour retrouver la "meilleure" solution parmi les *N* proposées.
Le système réalisé comporte deux modules. Le module markovien et le module de post-traitement.

Le module markovien se charge de générer les *N* meilleures solutions pour un mot prononcé (ou une phrase). La méthode employée consiste à utiliser l'algorithme de Viterbi dans la phase aller (effectuée de manière synchrone) et l'algorithme A* dans la phase retour (effectuée de manière asynchrone). Cette recherche est introduite au niveau le plus bas de la modélisation, c'est-à-dire au niveau acoustique. Nous montrons en fin de compte que le temps de recherche des *N* meilleures solutions n'est qu'une fonction affine du nombre *N*, pour *N* petit.

Le module de post-traitement se charge de retrouver la solution "correcte" dans la liste proposée. Pour cela deux approches sont étudiées : le post-traitement syntaxique et le post-traitement segmental statistique.

Beaucoup de syntaxes sont trop complexes pour être traitées directement. Dans le cas des villes épelées l'information syntaxique est la liste des villes possibles. Cependant cette liste est difficile à intégrer dans l'algorithme de décodage. Une solution consiste à rechercher les *N* meilleures solutions pour chaque ville épelée et à utiliser par la suite un post-traitement syntaxique pour retrouver parmi les *N* solutions proposées la première qui se trouve dans la liste des villes. Les résultats obtenus pour la reconnaissance de villes épelées dans une liste de 30000 villes, ont permis de montrer que le taux de substitution était très faible 3% avec un taux de rejet de 13%.

Cependant dans le cas où il n'y a pas de contraintes syntaxiques (reconnaissance de mots isolés par exemple) le post-traitement syntaxique n'est plus adapté. Dans ce cas nous utilisons un post-traitement segmental. Le post-traitement segmental consiste à calculer pour chaque solution, un score segmental qui est combiné par la suite avec le score markovien. La solution fournie par le système est celle ayant le score de combinaison le plus élevé.

Alors que la méthode généralement utilisée pour calculer le score segmental d'une solution utilise des réseaux de neurones, nous proposons dans le cadre de cette thèse d'utiliser une approche statistique. Cette approche repose sur une nouvelle technique dans la représentation statistique des segments. Deux modèles sont associés à chaque segment : le premier représente les statistiques d'une segmentation correcte et le second représente les statistiques d'une segmentation incorrecte. Cette approche a été testée sur différents corpus de données et a permis une réduction du taux d'erreur de l'ordre de 15 à 25 % par rapport à l'utilisation du modèle de Markov seul.

1 Introduction

Un système de reconnaissance est un système capable de traduire ou de décoder un signal acoustique de parole émis par un locuteur, et reçu à travers un capteur, en un mot ou une séquence de mots. L'utilité d'un système de reconnaissance est montrée à travers les tâches qu'on veut réaliser sans pour cela utiliser nos mains ou nous déplacer. On peut citer parmi ces tâches : la réservation de billets de train ou d'avion, la commande vocale pour handicapés, etc.

La plupart des systèmes de reconnaissance utilisés aujourd'hui, emploient pour la phase de décodage une méthode statistique. Cette dernière est à base de modèles de Markov cachés. Ces modèles ont eu beaucoup de succès grâce aux taux de reconnaissance élevés obtenus aujourd'hui. Cependant ce taux de reconnaissance chute lorsque le système est mis en exploitation (locuteur peu coopératif, bruit ambiant, etc.), c'est pour cela que plusieurs voies de recherches sont étudiées pour réduire le taux d'erreur : la modélisation de l'application, la détection bruit/parole, la reconnaissance dans un environnement bruité, la recherche des N meilleures solutions, etc.

Nous nous intéressons dans le cadre de cette thèse à la recherche des N meilleures solutions. Cette recherche implique la mise en œuvre de techniques permettant au système de reconnaissance de développer non plus une solution unique (la meilleure au sens des probabilités) mais les N meilleures solutions possibles pour un mot prononcé (ou une phrase). Cette recherche n'a cependant d'intérêt que si un post-traitement est effectué pour retrouver la solution "correcte" dans la liste proposée. Nous proposons pour cela deux approches : le post-traitement syntaxique et le post-traitement segmental.

Le système réalisé (figure 1), est constitué de deux modules : le module markovien, qui se charge de générer les N meilleures solutions, et le module de post-traitement qui détermine la solution "correcte".

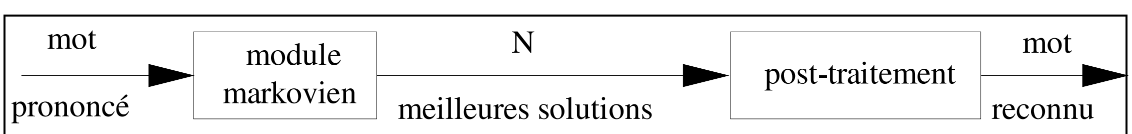

Figure 1: Schéma bloc du système réalisé

2 Module markovien

Ce module regroupe le système de reconnaissance développé par le CNET et une méthode de recherche des N meilleures solutions.

2.1 Système de reconnaissance et base de données

2.1.1 Système de reconnaissance

Le système de reconnaissance développé au CNET (PHIL90), permet la reconnaissance de vocabulaires dont la taille reste limitée à une centaine de mots environ, de manière indépendante du locuteur. Il autorise la reconnaissance de mots isolés ou de mots connectés.

Ce système de reconnaissance utilise une modélisation markovienne de l'application. A chaque mot du vocabulaire est associé un modèle de Markov caché constitué d'un ensemble d'états q_i, de transitions entre ces états, $t_{i,j}$ et de densité de probabilité d'émission dans l'espace acoustique $B_{i,j}$ liées aux transitions. Chaque fonction de densité de probabilité possède deux paramètres, le vecteur des moyennes $m_{i,j}$ et la matrice de covariance diagonale $\sum_{i,j}$.

L'analyse acoustique calcule toutes les 16 ms (fenêtre de Hanning de 32 ms avec un recouvrement de 50%) 8 coefficients cepstraux obtenus à partir des sorties d'un banc de filtres uniformément répartis sur une échelle Mel (MFCC), complétés par un paramètre d'énergie, plus les dérivées temporelles d'ordre un et deux de ces 9 coefficients.

Les phases d'apprentissage et de reconnaissance sont réalisées par l'algorithme de Viterbi. En phase de reconnaissance, le mot reconnu est identifié au modèle qui donne la plus grande probabilité d'émission du mot observé.

Dans le cadre de cette thèse, chaque mot du vocabulaire est décrit en une suite de phonèmes. Le traitement des phénomènes de coarticulation entre les phonèmes conduit à employer une modélisation qui prend en considération ces phénomènes contextuels. C'est l'approche par allophones qui consiste à modéliser chaque phonème par un modèle de Markov comportant plusieurs entrées et plusieurs sorties, choisies et validées en fonction du contexte dans lequel le phonème apparaît.

2.1.2 Corpus de données

Quatre corpus de données, enregistrés à travers le réseau téléphonique, ont été utilisés pour les différents tests. Ces corpus de données sont les chiffres isolés (0 à 9), les 36 mots isolés (ou expressions) du vocabulaire Trégor, les nombres à deux chiffres (00 à

99) et les épellations de noms de villes (120 villes de France). Le tableau suivant indique pour les quatre corpus utilisés, le nombre de données (locuteurs et enregistrements) ayant servi pour la phase d'apprentissage des modèles et le nombre de données ayant servi pour la phase de mesure des performances de reconnaissance (test).

Corpus	Locuteurs		Enregistrements	
	Test	Apprentissage	Test	Apprentissage
Chiffres (10 mots)	388	382	3622	3555
Trégor (36 mots)	384	381	12842	12719
Villes épelées	90	90	1425	1633
Nombres à 2 chiffres	397	392	7288	7304

Tableau 1 : Nombre de locuteurs et d'enregistrements par corpus de données

2.2 Recherche des N meilleures solutions

La méthode utilisée pour rechercher les N meilleures solutions est adoptée de celle proposée par [Soong, 91][1] et développée pour rechercher les N meilleures solutions au niveau syntaxique. Cependant, en vue d'introduire le post-traitement segmental, nous avons dû l'adapter pour la recherche des N meilleures solutions au niveau acoustique. Cela augmente la complexité dans la manière de gérer le temps de recherche et l'espace mémoire nécessaire au développement des N meilleures solutions.

2.2.1 Méthode de recherche

La méthode de recherche des N meilleures solutions se déroule en deux phases.

Pendant la phase aller, l'algorithme de Viterbi se charge de calculer et de mémoriser, pour chaque instant τ (τ varie entre 1 et T ; T nombre de trames dans le signal de parole) et pour chaque état q_i du réseau (i varie de 1 à NQ), la probabilité maximale d'observation des τ premières trames le long du meilleur des chemins atteignant l'état q_i à l'instant τ.

La phase retour est réalisée par l'algorithme A*. Il est utilisé pour remonter le chemin à partir de l'état final du réseau (T, NQ). Le coût global de la séquence de mots

[1][Soong, 91] F. K. Soong et E. Huang : "A Tree-trellis based fast search for finding the N best sentence hypotheses in continous speech recognition" ; Proc. ICASSP, Toronto, Mai 1991, pp. 705-708.

correspondant pour un chemin s en cours de développement, à un instant donné τ et pour un état intermédiaire du réseau acoustique q_i, est défini par :

$$f(s,\tau,q_i) = Log\ g(s,\tau,q_i) + Log\ h(\tau,q_i) \qquad \text{(Eq. 1)}$$

Les deux fonctions utilisées sont définies comme suit :

$Log\ g(s,\tau,q_i)$ est le coût (calculé par l'algorithme A*) entre l'état final q_{NQ} à l'instant T et l'état intermédiaire q_i à l'instant τ, correspondant à la séquence de mots identifiée par le chemin s en cours de développement. $Log\ h(\tau,q_j)$ est le coût (estimé au sens de l'application de l'algorithme A*) du meilleur chemin partant de l'état initial q_1 à l'instant 1 et aboutissant à l'état intermédiaire q_i à l'instant τ.
Dans l'approche utilisée, en deux passes, l'estimation de la portion du signal non encore traitée (i.e. $Log\ h(\tau,q_i)$) provient des valeurs mémorisées pendant la phase aller. Cette estimation (au sens de l'algorithme A*) est optimale et conduit alors à une recherche très efficace des N meilleures solutions.
* h est déterminée de manière récursive (algorithme de Viterbi), pendant la phase aller, par :

$$h(\tau,q_j) = \underset{q_i}{Max}[h(\tau-1,q_i).a_{i,j}.B_{i,j}(X[\tau])] \qquad \text{(Eq. 2)}$$

où $h(\tau,q_j)$ est la probabilité maximale d'observation des τ premières trames, le long du meilleur des chemins atteignant l'état q_j au temps τ, $a_{i,j}$ est la probabilité de transition pour passer de l'état q_i à l'état q_j et $B_{i,j}(X[\tau])$ est la valeur de la distribution associée à cette transition pour la trame $X[\tau]$.
* g est déterminée de manière récursive (algorithme A*), pendant la phase retour, par:

$$g(s,\tau,q_j) = \underset{q_k}{Max}[g(s,\tau+1,q_k).a_{j,k}.B_{j,k}(X[\tau+1])] \qquad \text{(Eq. 3)}$$

où $g(s,\tau,q_j)$ est la probabilité maximale d'observation des $T-\tau$ dernières trames, le long du meilleur des chemins partant de l'état q_j au temps τ et aboutissant à l'état q_{NQ} au temps T, et correspondant à la séquence de mots identifiée par le chemin s en cours de développement.
L'algorithme A* permet de développer plusieurs séquences de mots et donc d'obtenir les N meilleures solutions. Les N meilleures solutions calculées sont distinguées par leur étiquette. Ainsi les solutions obtenues sont des solutions syntaxiquement différentes et non pas des alignements différents d'une même solution.

Les mémorisations successives réalisées dans la phase retour permettent d'obtenir en fin de procédure le score, l'alignement (chemin) et les étiquettes de chacun des segments phonétiques constituant chacune des N meilleures solutions développées.

2.2.2 Résultats

Dans le système de reconnaissance fournissant les *N* meilleures solutions, on comptabilise une erreur (taux résiduel) lorsque la solution correcte n'apparaît pas dans les *N* solutions proposées. On définit alors le taux d'erreur résiduel comme le rapport du nombre d'erreurs (au sens défini ci-dessus) sur le nombre total d'enregistrements. Le taux d'erreur résiduel est bien évidemment fonction du nombre de solutions proposées.

Sur la figure 2, l'axe des abscisses correspond au nombre de solutions développées et l'axe des ordonnées correspond au taux d'erreur résiduel pour le nombre de solutions considérées. Nous présentons uniquement les résultats obtenus sur le corpus de test pour les chiffres isolés, le Trégor et les nombres à deux chiffres.

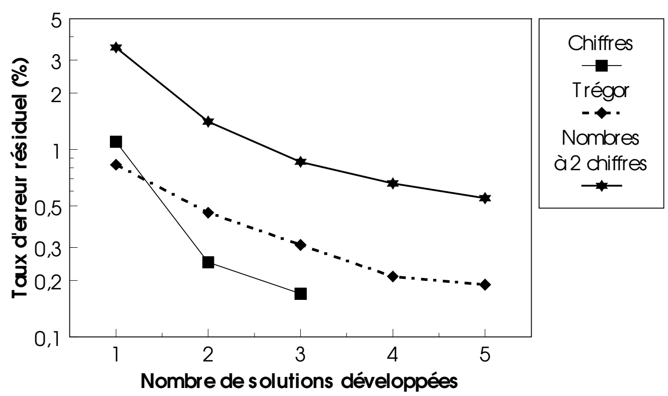

Figure 2 : Taux d'erreur résiduel sur 3 corpus de données

Les résultats obtenus montrent qu'un post-traitement efficace devrait permettre de réduire de manière substantielle les taux d'erreur (le choix de la bonne réponse parmi les 3 à 5 premières solutions permettrait de diviser par 4 les taux d'erreur).

2.2.3 Temps de recherche

Pour avoir un temps de recherche efficace, nous avons dû gérer d'une manière rigoureuse l'espace mémoire nécessaire pour le développement des N meilleures solutions. Nous avons décrit dans le document de thèse comment retrouver à chaque fois le prochain nœud acoustique à développer d'une part, et comment éliminer les chemins véhiculant la même séquence de mots reconnus de la pile pour gagner de la place mémoire d'autre part. La figure 3 présente, à titre d'exemple, la courbe de régression du temps de recherche des 10 meilleures solutions (on peut développer jusqu'à 100 solutions) sur le corpus des nombres pour 200 enregistrements.

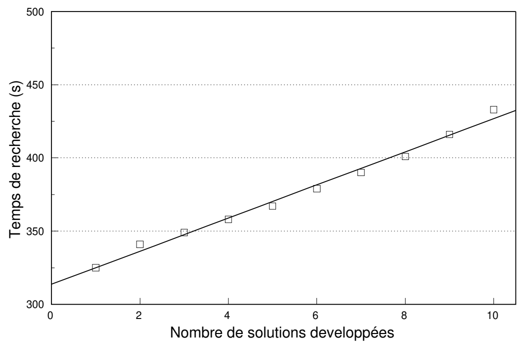

Figure 3 : Temps de calcul pour le traitement de 200 nombres.

On a constaté sur les 3 corpus de données testés (chiffres, Trégor et nombres) que le temps de recherche était une fonction affine du nombre N de solutions développées, pour N petit.

3 Module de post-traitement

Dans le cadre de la thèse, nous disposons de 3 types de corpus. Les corpus des mots isolés (chiffres et Trégor), le corpus des nombres à deux chiffres et le corpus des villes épelées.

La reconnaissance des villes épelées nécessite l'introduction d'une syntaxe complexe dans la modélisation pour décrire toutes les villes possibles. Cette syntaxe qui se résume à la liste des villes possibles est difficile à intégrer dans l'algorithme de décodage. Par contre, son utilisation dans le cadre d'un post-traitement syntaxique est simple. Il suffit de vérifier si chaque solution appartient ou non à la liste.

Cependant dans le cas où il n'y a pas de contraintes syntaxiques complexes (chiffres, Trégor et nombres à deux chiffres) le post-traitement syntaxique n'est plus adapté. Dans ce cas nous avons étudié l'apport du post-traitement segmental dans la réduction du taux d'erreur.

Nous avons donc développé les deux types de post-traitement : syntaxique et segmental.

3.1 Post-traitement syntaxique

Pour une ville épelée, on va rechercher les N meilleures solutions. Chaque solution, est comparée avec la liste de villes constituant le dictionnaire. Si la ville correspondant à la solution développée se trouve dans le dictionnaire, on parle alors de solution syntaxiquement correcte. Cependant si cette solution ne correspond pas à la ville prononcée, on parle alors de solution syntaxiquement correcte et incorrectement reconnue. C'est une erreur de reconnaissance (erreur de substitution). Dans le cas où elle correspond à la ville prononcée, cette solution est déclarée syntaxiquement correcte et correctement reconnue. C'est une réponse correcte. Une autre possibilité est qu'aucune des solutions développées n'appartienne au dictionnaire. Dans ce cas la ville prononcée est rejetée (rejet). Cette procédure de post-traitement syntaxique s'arrête dès qu'on trouve une solution syntaxiquement correcte dans la liste proposée, ou bien dès qu'on atteint le nombre maximal de solutions.

Le tableau suivant résume les performances du post-traitement syntaxique pour la reconnaissance des épellations de noms de villes, obtenues en développant 100 solutions sur le corpus de test et en utilisant 3 dictionnaires de tailles différentes (118 noms, 5000 noms et 30000 noms).

Taille du dictionnaire	Épellation des noms de villes N=100		
	Taux de rejet	Taux de substitution	Taux d'erreur global
118	14.9%	0.2%	15.1%
5000	14.4%	0.8%	15.2%
30000	13.6%	2.9%	16.5%

Tableau 2 : Performances du post-traitement syntaxique
sur les épellations des noms de villes pour N=100

Sans traitement spécifique, les résultats obtenus sur la reconnaissance des villes épelées ont montré que le taux d'erreur sur le corpus de test avoisine les 60 %. Avec un post-traitement syntaxique et pour un nombre de solutions développées égal à 100, les résultats obtenus montrent que : pour une taille de dictionnaire de 30000 noms de villes, une solution syntaxiquement correcte est incorrectement reconnue dans 3% des cas uniquement. Cependant le taux d'erreur global reste élevé, autour de 16.5%. Pour réduire ce taux d'erreur, une solution consiste à rechercher l'épellation la plus probable en fonction des lettres reconnues (modèle de Markov discret).

Il faut noter cependant que, en raison d'un taux de substitution très faible, on peut envisager de réaliser des applications utilisant l'épellation des villes s'il est possible de renvoyer la communication vers un opérateur en cas de rejet.

3.2 Post-traitement segmental statistique

3.2.1 Introduction

Le principe du post-traitement segmental repose sur le calcul d'un nouveau score pour chaque solution proposée. Le choix de la bonne réponse s'effectue alors en tenant compte de ce nouveau score et du score obtenu par le module markovien. La figure 2 illustre les différents modules du post-traitement segmental.

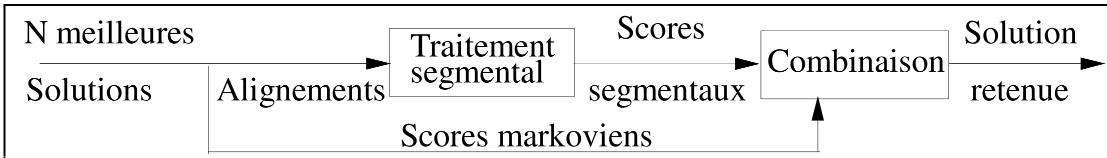

Figure 4 : Post-traitement segmental

Le score associé au post-traitement doit être établi à partir d'informations le moins redondantes possibles avec celles traitées dans le module markovien, de manière à garder une certaine complémentarité dans les caractéristiques discriminantes extraites.

Le module markovien étant un système de reconnaissance synchrone à la trame d'analyse, la complémentarité souhaitée est assurée par le choix d'un traitement segmental de chaque solution.

Chaque solution est divisée en un nombre déterminé de segments. Un segment peut représenter un phonème, une syllabe ... ou une partie de ceux-ci. Sur chaque segment est calculé un score de post-traitement. Ce score prend en compte des informations non accessibles à un système de reconnaissance trame par trame telles que la durée d'un segment, son spectre moyen, son évolution spectrale, etc. Le score global d'un mot est alors obtenu par une recombinaison des scores de chacun des segments le constituant.

Alors que la méthode généralement utilisée pour calculer le score segmental d'une solution utilise des réseaux de neurones, nous proposons dans le cadre de cette thèse d'utiliser une approche statistique. Cette approche repose sur une nouvelle technique dans la représentation statistique des segments.

3.2.2 Description de l'observation

Pour une observation inconnue *X*, on développe les N meilleures solutions. L'alignement obtenu pour chaque solution, et fourni par le module markovien, définit les différents segments utilisés pour décrire la solution. La figure 5 donne un exemple où les 3 meilleures solutions sont proposées pour le mot prononcé "5". L'observation correspondant à la k^{ème} meilleure solution est notée X_{i_k}.

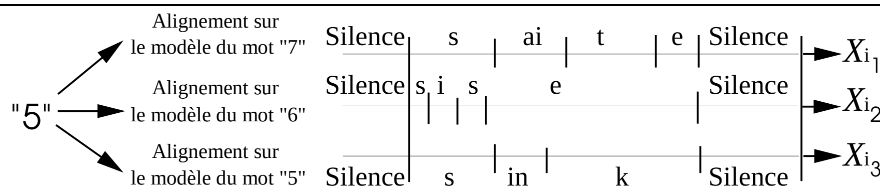

Figure 5 : Exemple d'alignements acoustiques obtenus en prononçant le mot "5".

Ainsi, on remarque qu'on n'a plus une observation unique mais N observations correspondant aux N meilleures solutions fournies par le premier module. Afin de ne pas privilégier une segmentation plus qu'une autre, on définit l'observation globale *X* comme étant la concaténation des observations associées à chacune des solutions proposées :

$$X = \{X_{i_1}, X_{i_2}, X_{i_3}, \ldots, X_{i_N}\} \quad \text{(Eq.4)}$$

où N est le nombre de solutions développées.

L'indice i sert à distinguer l'ordre dans lequel les solutions sont développées.

Les segments d'une solution sont déclarés corrects, si la solution obtenue correspond au mot prononcé (X_{i_3} sur la figure 5). Dans le cas contraire, ces segments sont déclarés incorrects. Ainsi sont définis deux types d'étiquettes : correcte et incorrecte. A chacune d'elle est associé un modèle statistique. Le modèle des corrects correspond à une segmentation correcte et le modèle des incorrects correspond à une segmentation incorrecte.

Nous avons montré que le calcul du score de chaque solution, en fonction des modèles corrects et des modèles incorrects, se réduit à :

$$R_{i_j} = Log\frac{\mathrm{P}(X_{i_j} / M_{E_{correct\ w_j}})}{\mathrm{P}(X_{i_j} / M_{E_{incorrect\ w_j}})} \quad \text{(Eq.5)}$$

R_{i_j} représente le rapport de probabilité d'émission de l'observation X_{i_j} par le modèle correct sur la probabilité d'émission de X_{i_j} par le modèle incorrect.

Dans le même contexte, deux modèles statistiques sont donc associés à chaque segment : le premier représente les statistiques des segments étiquetés corrects et le second modèle représente les statistiques des segments étiquetés incorrects. Le score d'un segment est obtenu en tenant compte des deux modèles statistiques (corrects et incorrects). Le score segmental d'une solution est une combinaison des scores des segments la constituant.

Finalement le score segmental et le score markovien d'une même solution sont combinés entre eux. La réponse fournie par le système est alors celle ayant le score final le plus élevé.

A partir des différentes expériences menées sur les corpus de données des chiffres isolés, des nombres à deux chiffres et du Trégor, nous avons constaté que l'utilisation du module de post-traitement seul n'était pas suffisant pour réduire le taux d'erreur obtenu par le modèle de Markov seul. De ce fait la prise en compte simultanée des scores du module markovien et du module de post-traitement, obtenus par chaque solution, était nécessaire. La coopération des deux systèmes (modèle de Markov et post-traitement) a alors permis une amélioration de l'ordre de 15 à 25 % par rapport à l'utilisation du modèle de Markov seul.

4 Conclusion et perspectives

Nous nous sommes intéressés dans cette thèse à l'étude d'une méthode de recherche des *N* meilleures solutions, son adaptation au système de reconnaissance du CNET et au développement de post-traitements segmentaux ou syntaxiques pour retrouver la "meilleure" solution parmi les N proposées.

Pour le recherche des N meilleures solutions, la méthode employée consiste à utiliser l'algorithme de Viterbi dans la phase aller (effectuée de manière synchrone) et l'algorithme A* dans la phase retour (effectuée de manière asynchrone). Cette recherche est introduite au niveau le plus bas de la modélisation, c'est-à-dire au niveau acoustique. Nous avons montré en fin de compte que le temps de recherche des *N* meilleures solutions n'est qu'une fonction affine du nombre *N*, pour *N* petit.

Le module de post-traitement se charge de retrouver la solution "correcte" dans la liste proposée. Pour cela deux approches ont été étudiées : le post-traitement syntaxique et le post-traitement segmental statistique.

Beaucoup de syntaxes sont trop complexes pour être traitées directement. Dans le cas des épellations de noms de villes, l'information syntaxique est la liste des villes possibles. Cependant cette liste est difficile à intégrer dans l'algorithme de décodage. Une solution consiste à rechercher les *N* meilleures solutions pour chaque ville épelée. Par la suite un post-traitement syntaxique est utilisé, pour retrouver parmi les *N* solutions proposées la première qui se trouve dans la liste des villes. Les résultats obtenus pour la reconnaissance de villes épelées dans une liste de 30000 villes, ont permis de montrer que le taux de substitution était très faible de l'ordre de 3% avec un taux de rejet de 13%.

Cependant dans le cas où il n'y a pas de contraintes syntaxiques (reconnaissance de mots isolés par exemple) le post-traitement syntaxique n'est plus adapté. Dans ce cas nous utilisons un post-traitement segmental. Le post-traitement segmental consiste à calculer pour chaque solution, un score segmental qui est combiné par la suite avec le score markovien. La solution fournie par le système est celle ayant le score de combinaison le plus élevé.

Nous avons proposé dans le cadre de cette thèse d'utiliser une approche statistique. Cette approche repose sur une nouvelle technique dans la représentation statistique des segments. Deux modèles sont associés à chaque segment : le premier représente les statistiques d'une segmentation correcte et le second représente les statistiques d'une segmentation incorrecte. Cette approche a permis une amélioration de l'ordre de 15 à 25 % par rapport à l'utilisation du modèle de Markov seul, sur les différents corpus de données testés.

Bien que l'amélioration obtenue avec le système proposé dans cette thèse soit significative, des améliorations sont encore possibles et des travaux dans cette direction sont à poursuivre.