MODELES DE MARKOV CACHES.

Louis SAUTER

1. INTRODUCTION.

La technique dite des Modèles de Markov Cachés (MMC ou HMM pour "Hidden Markov Models") a connu récemment une grande popularité dans le monde de la reconnaissance de la parole. En effet, de nombreuses équipes de recherche ont obtenu de bons résultats en reconnaissance de mots isolés ou enchaînés à l'aide de cette technique. Nous avons pensé qu'elle pourrait être utilisée de manière profitable en reconnaissance multilocuteur basée sur des demisyllabes.

Cette technique est fondée sur l'hypothèse que la parole est produite par une chaîne de Markov sous-jacente. Les paramètres observables sont des fonctions probabilistes de cette chaîne. Le mot "caché" provient du fait que la chaîne de Markov n'est pas observable directement, mais seulement par l'intermédiaire de la fonction probabiliste.

Nous allons maintenant donner quelques informations de fond concernant les chaînes de Markov, puis nous reviendrons sur le concept de Modèle de Markov Caché.

2. MODELES DE MARKOV CACHES.

2.1. Chaînes de Markov.

Soit $\mathbf{E}$ un ensemble fini non vide dont les éléments peuvent être interprétés comme les états possibles d'un système. Dans la suite, nous supposerons que $\mathbf{E} = \{1,2, \ldots ,N\}$, ce qui est possible sans perte de généralité en numérotant les états du système de 1 à N. On considère la suite $(\mathbf{E}_t)$, où t est un entier positif et où pour tout t, $\mathbf{E}_t = \mathbf{E}$. Un élément x_t de $\mathbf{E}_t$ est interprété comme un état possible du système à l'instant t.

Définition :

La suite aléatoire (X_t) est une chaîne (ou processus) de Markov sur $\mathbf{E}$ si pour tout entier t et pour tout élément

$$(x_0,x_1, \ldots ,x_{t+1}) \in \prod_{\tau=0}^{\tau=t} \mathbf{E}_\tau$$

on a

$$P(X_{t+1}=x_{t+1} \mid X_0=x_0,X_1=x_1, \ldots ,X_t=x_t) = P(X_{t+1}=x_{t+1} \mid X_t=x_t) \tag{1}$$

Cette égalité signifie qu'il est inutile de connaître les états du système antérieurs à t pour la prévision de son état à l'instant t+1 ; l'état est un résumé suffisant de l'histoire du système.

Posons

$$v_i^j(t) = P(X_{t+1} = j \mid X_t = i) \tag{2}$$

et

$$V(t) = [v_i^j(t)] \tag{3}$$

alors $V(t)$ est une matrice stochastique. On dit que $(V(t))$ est la suite de matrices stochastiques de la chaîne $(\mathbf{X}_t)$.

2.2. Modèles de Markov Cachés.

L'utilisation de MMC en reconnaissance de la parole est fondée sur l'hypothèse que la parole est produite par un processus de Markov sous-jacent. Les états eux-mêmes ne sont pas observables, d'où le terme "caché". La parole effectivement observée s_t est en fait une fonction probabiliste de la chaîne de Markov. A chaque transition, un symbole est émis selon la densité $f(s_t \mid X_{t-1})$. (Note : nous supposons ici que la probabilité d'émission dépend uniquement de l'état de départ et non de l'état d'arrivée.)

Soit S une suite finie d'observations correspondant à l'élocution d'un mot

$$S = s_1 s_2 \cdots s_T \tag{4}$$

La chaîne de Markov sous-jacente correspond à la suite d'états

$$X = x_0 x_1 \cdots x_T \tag{5}$$

Le Modèle de Markov Caché **M** est caractérisé par un nombre (fini) d'états N, la matrice de transition V et la densité de probabilité pour les observations, $f_i(s_t)$.

$$f_i(s_t) = f(s_t \mid X_{t-1} = i) \tag{6}$$

Note : pour définir le modèle de Markov, il faut également connaître des probabilités initiales pour chaque état. Nous supposerons dans la suite qu'à l'instant t=0, le système est dans l'état 1.

La reconnaissance de la parole utilisant les MMC comprend deux parties : l'apprentissage et la reconnaissance. Pendant l'apprentissage, nous devons calculer un modèle pour chaque mot (ou segment) dans le vocabulaire. Pour cela, on cherche la matrice de transition V et les densités $f_i(s)$ qui maximalisent la probabilité a posteriori de produire les élocutions du corpus d'apprentissage. La reconnaissance est effectuée en cherchant le mot dont le modèle produit le mot test avec le maximum de vraisemblance. Nous examinons maintenant les algorithmes utilisés pour l'apprentissage et la reconnaissance.

2.3. Apprentissage.

Comme il a été dit ci-dessus, nous cherchons la matrice de transition et les densités de probabilité qui maximalisent la probabilité a posteriori de produire les élocutions du corpus d'apprentissage. Pour faire cela, nous allons utiliser l'algorithme dit "Forward-Backward". Nous ne donnerons pas de démonstration de la validité de ce calcul ; voir [6,7,8] pour une justification mathématique.

Autant que cela est possible, les notations utilisées ici correspondent aux programmes en FORTRAN. Les lettres grecques sont épelées dans les programmes.

Nous supposons que la matrice de transition est stationnaire, ainsi que les densités d'émission des symboles. Nous allons d'abord décrire ces algorithmes dans le cas où les symboles émis appartiennent à un alphabet fini. Dans ce cas, nous pouvons parler de probabilité d'émission d'un symbole à chaque transition, et définir

$$v_i^j = P(X_t=j \mid X_{t-1}=i) \tag{7}$$

$$f_i(s) = P(S_t=s \mid X_{t-1}=i) \tag{8}$$

Ces deux égalités étant vérifiées pour tout t strictement positif.

Chacune des L répétitions d'un mot (ou segment) du corpus d'apprentissage correspond à l'émission d'une suite $S^{(l)}$ de symboles de longueur $T^{(l)}$. Notons **S** l'ensemble de ces suites. Etant donné un Modèle de Markov Caché **M**, la probabilité a posteriori d'avoir produit le corpus d'apprentissage est

$$P(\mathbf{S} \mid \mathbf{M}) = P(S^{(1)},S^{(2)}, \ldots ,S^{(L)} \mid \mathbf{M}) = \prod_{l=1}^{L} P(S^{(l)} \mid \mathbf{M}) \tag{9}$$

où

$$P(S^{(l)} \mid \mathbf{M}) = \sum_{X^{(l)}} P(S^{(l)},X^{(l)} \mid \mathbf{M}) \tag{10}$$

$X^{(l)}$ désigne les séquences d'états ayant pu engendrer $S^{(l)}$.

En définissant une fonction auxiliare dont on calcule le Lagrangien, on démontre qu'il est possible de trouver un maximum local de (9) en recalculant itérativement le modèle **M**. Le calcul de la nouvelle estimation de **M** nécessite la définition de "forward probabilities"

$$\xi_t^{(l)}(i) = P(S_1^{(l)}=s_1, \ldots ,S_t^{(l)}=s_t,X_t^{(l)}=i) \tag{11}$$

et de "backward probabilities"

$$\eta_t^{(l)}(i) = P(S_{t+1}^{(l)}=s_{t+1}, \ldots ,S_T^{(l)}=s_T \mid X_t^{(l)}=i) \tag{12}$$

Ces probabilités peuvent être calculées itérativement ; les premières dans le sens des indices croissants, les deuxièmes dans le sens des indices décroissants (d'où les noms "forward" et "backward").

$$\xi_t^{(l)}(i) = \sum_{j=1}^{N} \xi_{t-1}^{(l)}(j)\ v_j^i\ f_j(s_t) \tag{13}$$

$$\eta_t^{(l)}(i) = \sum_{j=1}^{N} \eta_{t+1}^{(l)}(j)\ v_i^j\ f_j(s_{t+1}) \tag{14}$$

En remarquant que pour tout t

$$\xi_t^{(l)}(i)\eta_t^{(l)}(i) = P(S^{(l)},X_t^{(l)}=i \mid \mathbf{M}) \tag{15}$$

on déduit de (10)

$$P(S^{(l)} \mid \mathbf{M}) = \sum_{i=1}^{N} \xi_t^{(l)}(i)\eta_t^{(l)}(i) \tag{16}$$

Nous noterons cette valeur $\sigma^{(l)}$. En particulier, pour t=0, seul l'état initial 1 est possible

$$\sigma^{(l)} = \eta_0^{(l)}(1) \tag{17}$$

D'autre part, si à la fin du mot (ou du segment) on est obligatoirement dans l'état N (état final), alors pour $t=T$

$$\sigma^{(l)} = \xi_T^{(l)}(N) \tag{18}$$

Posons encore

$$\gamma_t^{(l)}(i,j) = \frac{P(S^{(l)},X_{t-1}^{(l)}=i,X_t^{(l)}=j)}{P(S^{(l)})} \tag{19}$$

γ peut être calculé selon l'équation

$$\gamma_t^{(l)}(i,j) = \frac{\xi_{t-1}^{(l)}(i)\ v_i^j\ f_i(s_t)\ \eta_t^{(l)}(j)}{\sigma^{(l)}} \tag{20}$$

Il sera utile de définir également les fonctions suivantes

$$\zeta(i,j,s) = \sum_{l=1}^{L}\sum_{t=1}^{T^{(l)}} \delta(S_t^{(l)},s)\gamma_t^{(l)}(i,j) \tag{21}$$

où δ désigne le delta de Kronecker.

$$\zeta(i,j) = \sum_{s} \zeta(i,j,s) \tag{22}$$

$$\zeta(i) = \sum_{j} \zeta(i,j) \tag{23}$$

Alors, le réestimation de V se fait selon l'équation

$$v_i^j = \frac{\sum_{l=1}^{L} \sum_{t=1}^{T^{(l)}} \gamma_t^{(l)}(i,j)}{\sum_{l=1}^{L} \sum_{t=1}^{T^{(l)}} \gamma_t^{(l)}(i)} = \frac{\zeta(i,j)}{\zeta(i)} \tag{24}$$

D'autre part, $f_i(s)$ est réestimé par l'équation

$$f_i(s) = \frac{\sum_{j=1}^{N} \zeta(i,j,s)}{\zeta(i)} \tag{25}$$

2.4. Reconnaissance.

Nous décrivons maintenant l'algorithme de Viterbi.

Il s'agit du calcul de la probablité d'émettre la séquence S en suivant la suite d'états X la plus probable. On cherche donc

$$P_v(S) = \max_X P(S,X) \tag{26}$$

Cette valeur peut être trouvée à l'aide des techniques de programmation dynamique. (Note : il ne s'agit pas de l'algorithme de *comparaison dynamique* habituellement utilisé en reconnaissance de la parole.)

$$P(S,X) = \prod_{i=1}^{T} v_{x_{i-1}}^{x_i} f_{x_{i-1}}(s_i) \tag{27}$$

Posons

$$P_t(i) = \max P(x_t=i, S_1=s_1, \ldots, S_t=s_t) \tag{28}$$

alors

$$P_t(i) = \max_j P_{t-1}(j)\ v_j^i\ f_j(s_t) \tag{29}$$

et finalement,

$$P_v(S) = P_T(N) \tag{30}$$

où N est l'état final du système après l'émission d'un mot.

3. UTILISATION DE MMC POUR DES SEGMENTS.

3.1. Segments en Reconnaissance de la parole.

Un des problèmes majeurs en reconnaissance de la parole est la coarticulation. La manière dont le contexte influe sur différents phonèmes n'est pas bien maîtrisée, et peut difficilement être traitée automatiquement. Une manière de traiter ce problème est d'utiliser des segments englobant plusieurs phonèmes, tels que des diphones, des syllabes ou des demi-syllabes [4-5].

Des références globales pour des mots isolés peuvent être reconstruites en concatenant des références de segments. Cette méthode nécessite une comparaison dynamique pour chaque mot (ou référence) dans le lexique. Nous avons présenté par ailleurs comment cette méthode peut être utilisée en reconnaissance multi-locuteur.

Une approche différente est fondée sur des algorithmes de reconnaissance de mots connectés (on obtient ainsi un algorithme de segments connectés). La recherche est guidée par le lexique d'une manière semblable à l'utilisation de la syntaxe en reconnaissance de mots connectés.

3.2. Expérience 1.

Un premier essai a été effectué dans lequel des MMC sont calculés pour chaque segment présent dans le corpus. Un MMC est ensuite calculé pour chaque mot du vocabulaire en concaténant les modèles des segments composant le mot. Ces modèles sont ensuite utilisés pour la reconnaissance. Cette méthode est très voisine de ce qui avait été fait avec la programmation dynamique. Une différence importante est qu'ici, il n'y qu'une référence par mot au lieu de cinq. Dans cette expérience, nous avons utilisé la quantification vectorielle. Ainsi, les observables appartiennent à un ensemble fini, donc les densités $f_i(s)$ sont discrètes. Le nombre d'éléments dans le dictionnaire est de 64, ce qui correspond au nombre de classes donnant les meilleurs résultats avec la programmation dynamique. Nous avons utilisé l'algorithme des nuées dynamiques sur l'ensemble des segments de référence qui avait été calculé dans le cadre de notre étude précédente.

Le MMC pour chaque segment a deux états. La matrice de transition est une matrice 2×2 triangulaire supérieure de la forme

$$\begin{bmatrix} \nu & 1-\nu \\ 0 & 1 \end{bmatrix} \tag{31}$$

Le modèle pour chaque segment est obtenu avec l'algorithme dit "Forward-Backward". Le modèle pour chaque mot est obtenu en concatenant les modèles des segments. La deuxième ligne des matrices de transition des segments est remplacée par le vecteur ligne

$$\begin{bmatrix} \alpha & 1-\alpha \end{bmatrix} \tag{32}$$

où α a été fixé expérimentalement à 0.5. On obtient donc une matrice de transition de la forme

$$\begin{bmatrix} \nu_1 & 1-\nu_1 & 0 & 0 & 0 & 0 & \cdots \\ \alpha & 1-\alpha & 0 & 0 & 0 & 0 & \cdots \\ 0 & 0 & \nu_2 & 1-\nu_2 & 0 & 0 & \cdots \\ 0 & 0 & \alpha & 1-\alpha & 0 & 0 & \cdots \\ 0 & 0 & 0 & 0 & \nu_3 & 1-\nu_3 & \cdots \\ 0 & 0 & 0 & 0 & \alpha & 1-\alpha & \cdots \\ \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots \end{bmatrix} \tag{33}$$

Finalement, la reconnaissance est faite à l'aide de l'algorithme de Viterbi.

3.3. Expérience 2.

Dans cette expérience, nous n'avons pas utilisé la quantification vectorielle. Les densités de probabilité $f_i(s)$ ont été supposées gaussiennes, avec des matrices de covariance diagonales. Ceci n'est qu'une expérience préliminaire, car il est connu que des mélanges gaussiens donnent de meilleurs résultats [9]. Le programme d'apprentissage a été largement modifié car il faut calculer à chaque itération une nouvelle estimation des moyennes et de la covariance. Voir [8] pour plus d'informations sur ce calcul.

3.4. Résultats.

Le nombre d'erreurs avec la programmation dynamique, sur un corpus de 180 élocutions (vocabulaire test très difficile), était de 16. Avec la quantification vectorielle, ce nombre était de 21.

Le nombre d'erreurs dans la première expérience était de 27, soit approximativement 25 % de plus qu'avec la programmation dynamique.

Le nombre d'erreurs pour la deuxième expérience était de 30. Ce résultat plutôt mauvais est peut-être dû aux valeurs initiales utilisées pour démarrer l'algorithme Forward-Backward.

4. REFERENCES.

[1] L. Sauter, "Segment-based Speaker Independent Isolated Word Recognition," *Proc. Congrès AFCET Informatique*, Paris, Jan 1985.

[2] L. C. Sauter, "Isolated Word Recognition using a Segmental Approach,", *Proc. ICASSP 85*, pp. 850-853, Mar 1985.

[3] L. R. Rabiner and J. G. Wilpon, "Speaker Independent Isolated Word Recognition for a Moderate Size (54 Word) Vocabulary," *IEEE Trans. Acoust. Speech and Sign. Proc.*, Vol. ASSP-27, No. 6, pp. 583-587, Dec. 1979.

[4] O. Fujimura, "Syllable as a Unit of Speech Recognition," *IEEE Trans. Acoust. Speech and Sign. Proc.*, Vol. ASSP-23, No. 1, pp. 79-82, Feb. 1975.

[5] Th. Schotola, "On the use of demisyllables in automatic word recognition," *Speech Communication*, Vol. 3, No. 1, pp. 63-86, April 1984.

[6] L. E. Baum, T. Petrie, G. Soules and N. Weiss, "A Maximization Technique Occuring in the Statistical Analysis of Probabilistic Functions of Markov Chains," *The Ann. of Math. Stat.*, Vol. 41, No. 1, pp. 164-171, 1970.

[7] F. Jelinek, "Continuous Speech Recognition by Statistical Methods," *Proc. of the IEEE*, vol. 64, no. 4, pp. 532-556, April 1976

[8] L. A. Liporace, "Maximum Likelihood Estimation for Multivariate Observations of Markov Sources," *IEEE Trans. on Inform. Theory*, Vol. IT-28, No. 5, pp. 729-734, Sep. 1982.

[9] L. R. Rabiner, B.-H Juang, S. E. Levinson and M. M. Sondhi, "Recognition of Isolated Digits Using Hidden Markov Models With Continuous Mixture Densities," *AT&T Tech. Journ.*, VOl. 64, No. 6, Part 1, pp. 1211-1234, July-Aug. 1985.

[10] L. R. Rabiner and S. E. Levinson, "A Speaker-Independent, Syntax-Directed, Connected Word Recognition Sytem Based on Hidden Markov Models and Level Building," *IEEE Trans. on Acoust., Speech and Sign. Proc.*, Vol. ASSP-33, No. 3, pp. 561-573, June 1985.