Algorithmes stochastiques d'apprentissage et de reconnaissance

Jean-François MARI
Bâtiment LORIA, Campus scientifique
B.P. 239
54506 Vandœuvre-lès-Nancy CEDEX

1 Définitions des chaînes de Markov

1.1 Processus stochastique

Soit un ensemble T, engendré par l'indice t, et une variable aléatoire s_t indicée par t, on appelle processus stochastique l'ensemble : $(s_t,t \in T)$. En général, T est dénombrable et représente le temps, le processus est alors dit *discret*.

1.2 Suite stochastique

Soit un système pouvant se trouver dans un ensemble dénombrable d'états noté $\{u_1,u_2 \dots\}$. Entre les instants t et $t+1$, le système passe aléatoirement de l'état u à l'état v. On appellera Ω l'ensemble des transitions notées (u,v) entre 2 états. Pour représenter ce processus aléatoire, on définit la variable s_t de la façon suivante : $s_t = u$ signifie que le système est dans l'état u au temps t. Par définition l'ensemble $(s_1,s_2,\dots s_t,\dots)$ est une suite stochastique à ensemble discret d'états.

1.3 Chaîne de Markov

Une chaîne de Markov est une suite stochastique à ensemble discret d'états telle que :

– Suite du premier ordre

$$\begin{aligned} Prob(s_t = v/s_{t-1} = u,s_{t-2} = \dots) \\ = Prob(s_t = v/s_{t-1} = u) \end{aligned} \quad (1)$$

– Suite du deuxième ordre

$$Prob(s_t = w/s_{t-1} = v, s_{t-2} = u, s_{t-3} = ...) \\ = Prob(s_t = w/s_{t-1} = v, s_{t-2} = u) \quad (2)$$

En d'autres termes, l'évolution du système entre 2 instants t et $t+1$ ne dépend que de l'état de ce système au temps $t-1$ (ordre un) ou aux instants précédents $t-1$ et $t-2$ (ordre deux).

Une chaîne est dite homogène si les probabilités de transition ne dépendent pas du temps. On peut alors poser : $a_{uv} = Prob(s_t = v/s_{t-1} = u)$ et $a_{uvw} = Prob(s_t = w/s_{t-1} = v, s_{t-2} = u)$.

Si l'ensemble des états $\mathbf{S}$ est fini et possède N états alors on dit que la chaîne est finie. Dans la suite, nous nous limiterons aux chaînes de cette sorte. Dans ce cas, on a les contraintes suivantes :

– Chaînes du premier ordre

$$\sum_{k=1}^{N} a_{jk} = 1 \quad , 1 \leq j \leq N$$

– Chaînes du second ordre

$$\sum_{k=1}^{N} a_{ijk} = 1 \quad avec \quad (i,j) \in \Omega$$

Dans un processus de Markov d'ordre 2, la probabilité de la suite d'états :

$$s = s_1, s_2, ..., s_T$$

est définie par :

$$Prob(s) = \Pi_{s_1} a_{s_1 s_2} \prod_{t=3}^{T} a_{s_{t-2} s_{t-1} s_t}$$

où Π_i est la probabilité de l'état u_i à $t = 1$ et a_{uv} est la probabilité de la transition $u \rightarrow v$ au temps $t = 2$. Dans la suite de cet exposé on représentera un état u_i par l'entier i quand il n'y aura pas d'ambiguïté.

2 Application à la reconnaissance de la parole

Les modèles de Markov utilisés en reconnaissance de la parole constituent un cas très particulier de modèles de Markov. Les ensembles T et S sont finis et on utilise deux processus stochastiques. L'un est qualifié de *visible*, il est constitué d'une suite de variables aléatoires O_t dont les réalisations sont les trames de parole. Ce processus stochastique est gouverné par un deuxième processus stochastique. En général, on suppose que c'est une suite de Markov d'ordre 1 (HMM1) ou 2 (HMM2). Les états de ce processus sont cachés et représentent les variables aléatoires O_t d'où sont issues les observations o_t.

L'exemple des urnes de Polya [8] illustre bien les deux composantes d'un HMM. Dans cet exemple, nous avons plusieurs urnes contenant des boules de couleurs différentes. Ces urnes sont cachées derrière un voile. L'observateur ne voit pas de quelle urne est issue la boule qu'on lui présente. Le processus de tirage choisit l'urne dans laquelle se fera le tirage selon un processus Markovien ; le choix de l'urne suivante se fait uniquement en fonction de l'urne précédemment utilisée. Ce processus de tirage correspond au processus caché ; une urne correspond à un état. La boule de couleur issue de l'urne est la seule observation que l'on puisse faire sur ce processus particulier de tirage.

En reconnaissance de la parole à l'aide de HMM, on suppose que l'élocution peut être décomposée en une suite de segments dans lesquels chaque trame est issue du même processus aléatoire, c'est à dire possédant une moyenne et une variance qui ont été estimées pendant l'apprentissage. Dans la suite de ce chapitre, nous nous intéresserons à l'estimation selon le maximum de vraisemblance. Cet estimateur est efficace dans le cas d'un corpus exhaustif ; ce qui n'est jamais le cas dans notre pratique et justifie la constitution de corpus de plus en plus importants. Sur un corpus exhaustif, les variances des densités des HMM sont minimales. Un segment est modélisé par un état du processus de Markov caché. À chaque instant on ne sait pas de quel segment provient une trame car les densités associées aux états se recouvrent.

Dans la figure 1 on distingue les segments modélisés par chaque état de la chaîne de Markov. On représente l'alignement le plus probable entre la suite de trames représentant le signal de parole et la suite cachée d'états produite par la chaîne de Markov d'ordre deux. On distingue nettement que les premiers états de chacun des modèles ont capturé la partie vocalique tandis que les derniers états capturaient les parties consonantiques de la prononciation des lettres épelées du nom *FOHR*.

En modélisant ainsi les segments composant le signal de parole, on sup-

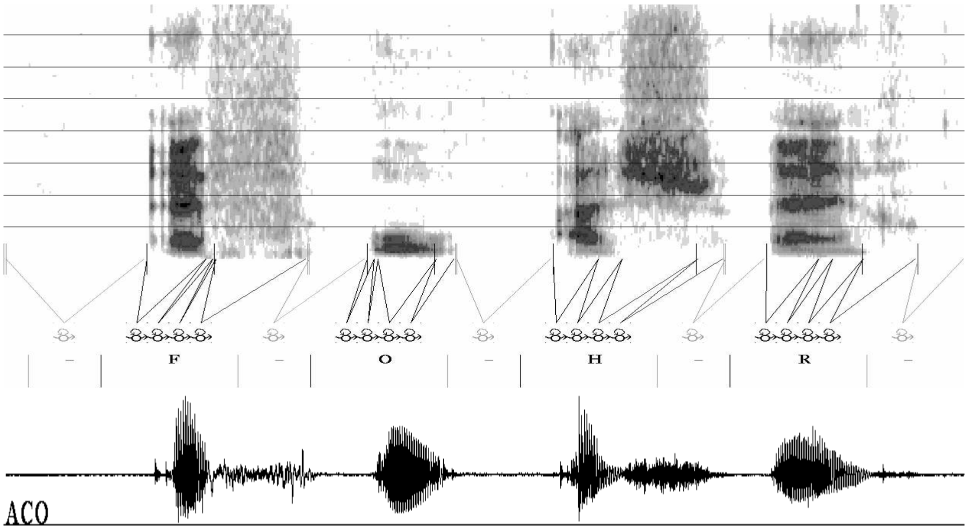


FIG. 1 – *Chaque état modélise un segment de parole*

pose que chaque trame est indépendante des trames voisines. Cette hypothèse ignore la réalité de l'articulation du conduit vocal mais a l'avantage de permettre la réalisation d'algorithmes d'apprentissage et de reconnaissance rapides.

Le problème de l'apprentissage sera d'estimer les paramètres des processus stochastiques à partir d'exemples d'élocutions. Le problème de la reconnaissance sera de reconstituer la suite de segments la plus probable connaissant la suite de trames observées.

2.1 Notations et définitions

Dans la suite nous appellerons :

- s_t la suite de Markov finie d'ordre 1 ou 2, homogène, appelée *suite cachée*. Ce processus stochastique est entièrement défini par la donnée des probabilités initiales et de la matrice de transitions entre états. Dans le cas d'un processus d'ordre 1, la matrice (a_{uv}) donne la probabilité de transition entre les états u et v. Dans le cas d'un processus d'ordre 2, la matrice (a_{uvw}) donne la probabilité de transition entre les états u,v,w.
- y_t le processus stochastique visible. y_t prend ses valeurs dans R^d. La densité de chaque y_t est définie par la donnée des densités $b_i()$ représentant chacun des états s_i du processus de Markov caché. On modé-

lise $b_i()$ par une somme pondérée de M lois normales $\mathcal{N}(\mu_{im},\Sigma_{im})$ de moyenne μ_{im} et de matrice de covariances Σ_{im}.

$$b_i(y_t) = \sum_{m=1}^{M} c_{im}\mathcal{N}(y_t;\mu_{im},\Sigma_{im}), \quad \text{avec} \quad \sum_{m=1}^{M} c_{im} = 1$$

Un modèle de Markov caché du deuxième ordre est défini par la donnée de :

- $\mathbf{S} = \{u_1,u_2,\ldots,u_N\}$, un ensemble fini de N états ;
- $\mathbf{A} = (a_{ijk})$ la matrice donnant les probabilités de transition entre états ;
- π le vecteur de probabilités initiales de la suite stochastique cachée ;
- $b_i(.)$ les lois des densités, souvent appelées abusivement *pdf*, associées aux états s_i du processus stochastique caché.

L'ensemble de ces paramètres $\theta = \{S,A,\pi,b_i(.)\}$ constitue le modèle de Markov caché θ du deuxième ordre.

2.2 Probabilité d'un alignement temporel

Un alignement temporel entre une suite d'états et une suite de trames tel qu'il est montré figure 1 est une correspondance (état, trame). Sa probabilité est définie par :

$$Prob(s,y) = \prod_t a_{s_{t-2}s_{t-1}s_t} b_{s_t}(y_t) \tag{3}$$

dans lequel le terme $a_{s_{t-2}s_{t-1}s_t}$ représente la probabilité de la contribution de la chaîne de Markov entre les instants t-2 et t (cas d'une chaîne d'ordre 2) qui définit la segmentation. Quant au terme $b_{s_t}(y_t)$ il représente la vraisemblance de la trame y_t conditionnée à la densité de l'état s_t.

Dans le cas des modèles de Markov gauche-droite décrits dans la figure 1, un alignement entre états et trames définit une segmentation.

Un grand intérêt des modèles de Markov cachés est qu'ils permettent lors de la reconnaissance une recherche de tous les alignements afin de retrouver la segmentation possédant la probabilité définie par l'équation 3 maximale. L'algorithme utilisé est *l'algorithme de Viterbi* qui est un algorithme de programmation dynamique [4].

3 L'algorithme d'apprentissage EM

L'algorithme EM permet l'apprentissage par maximum de vraisemblance de modèles à une composante cachée.

Soit θ, l'ensemble des paramètres définissant un modèle. On appelle Y_t la variable aléatoire dont les réalisations sont les observations y_t faites au temps t. $Y = y$ signifie $Y_1 = y_1, Y_2 = y_2, \dots Y_t = y_t$.

On cherche à maximiser la vraisemblance du modèle sur le corpus d'observations – $P(observations/modele)$ – que nous noterons $f(y/\theta) = P(Y = y/\theta)$.

Dans un HMM. la vraisemblance est la somme des probabilités de tous les alignements entre les états $s_1, s_2, \dots s_t$ et les observations $y_1, y_2, \dots y_t$.

$$f(y/\theta) = \sum_{s \in S^n} g(Y = y, S = s/\theta)$$

que l'on peut noter plus simplement

$$f(y/\theta) = \sum_{s \in S^n} g(y, s/\theta)$$

avec la probabilité d'un alignement défini comme suit

$$g(y, s/\theta) = \prod_t a_{s_t s_{t+1}} b_{s_{t+1}}(y_{t+1}).$$

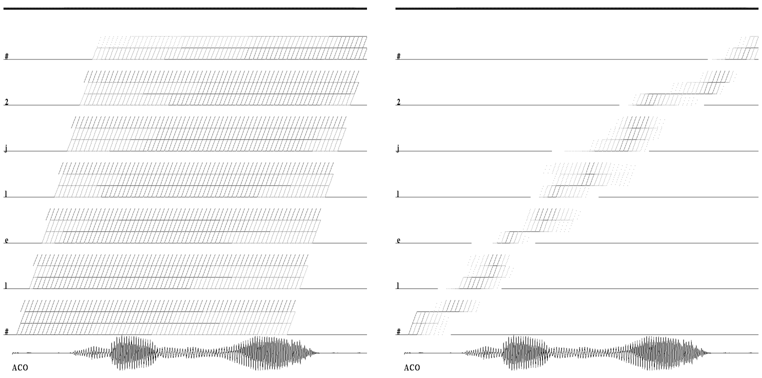
La log vraisemblance est définie par

$$L(y/\theta) = \log f(y/\theta).$$

Chercher à maximiser la log vraisemblance n'est pas pratique par les méthodes usuelles de descente de gradient. L'algorithme EM propose une alternative intéressante.

L'algorithme EM [3] construit une suite $\theta_0, \theta_1, \dots \theta_m \dots$ telle que $L(y/\theta_0) \leq L(y/\theta_1) \dots \leq L(y/\theta_m)$ qui converge vers un point stationnaire. Sous certaines conditions de régularité (1983), on obtient un maximum local ou global θ_{MLE}.

Puisque la suite des états $s_1, s_2, \dots s_t$ est inconnue – il y en a plusieurs de probabilités différentes – l'algorithme va remplacer $\log g(Y, S/\theta)$ par son espérance conditionnelle calculée sur une distribution d'alignements. La figure 2 montre deux distributions d'alignements. Sur chaque figure sont représentées en niveaux de gris les probabilités $Prob(S_t = s_t, S_{t+1} = s_{t+1}/\theta, y)$. Dans la



(a) $\Pi(s/y,\theta_0)$ (b) $\Pi(s/y,\theta_1)$

FIG. 2 – *distributions des alignements initiaux et finaux de la phrase « Les lieux »*

sous figure 2(a), on a utilisé un HMM – représenté par le paramètre θ_0 – où toutes les probabilités de transitions et d'observations sont équiprobables. Dans la sous figure 2(b), le vecteur de paramètres θ_1 rend les transitions voisines de la diagonale plus probables.

Dans la suite, nous noterons $\Pi(S,Y/\theta)$ cette distribution définie par

$$\Pi(S,Y/\theta) = \frac{g(Y,S/\theta)}{f(Y/\theta)} \tag{4}$$

Quand il n'y a qu'un alignement possible, cette distribution est représentée dans la figure 2 par un chemin joignant le coin inférieur gauche au coin supérieur droit et matérialisant l'alignement.

On définit ensuite une fonction de θ qui dépende d'un paramètre θ'. Nous notons :

$$Q(\theta/\theta') = E(\log g(Y,S/\theta)/y,\theta')$$

$Q(\theta/\theta')$ représente l'espérance de $\log g(Y,S/\theta)$ à l'aide de la répartition calculée avec y,θ'.

On imagine maintenant que l'algorithme cherche à construire itérativement une suite de distributions $\Pi_0,\Pi_1,\ldots$ dans lesquelles les dynamiques des probabilités d'alignements seront croissantes. On peut voir la sous-figure 2(b) comme une version moins floue de la sous-figure 2(a). Il est même possible d'appliquer à EM des techniques de restauration d'images.

On définit par $H(\theta/\theta')$ l'espérance de la log probabilité d'un alignement du HMM paramétré par θ à l'aide de la distribution des alignements calculés à l'aide du HMM représenté par θ' ; cette espérance est une fonction de θ paramétrée par θ'. On a

$$H(\theta/\theta') = \sum_{s \in S^n} \log \Pi(S/Y,\theta)\Pi(S/Y,\theta')$$

Comme $\sum_{s \in S^n} \Pi(S/Y,\theta') = 1$, on obtient

$$L(Y/\theta) = \sum_{s \in S^n} \log f(Y/\theta)\Pi(S/Y,\theta') = E(\log f(Y/\theta)/y,\theta')$$

En appliquant la définition de Π donnée dans l'équation 4 on obtient

$$E(\log f(Y/\theta)/y,\theta') = E(\log \frac{g(Y,S/\theta)}{\Pi(s/Y,\theta)}/y,\theta')$$

et donc

$$L(Y/\theta) = Q(\theta/\theta') - H(\theta/\theta').$$

Parmi toutes les valeurs de θ, choississons celle qui donne le maximum de la fonction $Q(\theta/\theta')$; on a donc $Q(\theta/\theta') \geq Q(\theta'/\theta')$. Il nous reste à appliquer l'inégalité de Jansen qui stipule

$$H(\theta'/\theta') \geq H(\theta/\theta') \quad \forall\theta.$$

pour voir que ce choix de θ donne

$$L(y/\theta) \geq L(y/\theta').$$

Cette propriété de monotonie permet de démontrer la convergence de l'algorithme [7]. A partir de la donnée d'un modèle initial θ_0, l'algorithme EM construit une suite de modèles $\theta_1, \ldots \theta_m$ de la façon suivante :

étape E : comme espérance ; on calcule $Q(\theta/\theta_m)$

étape M : comme maximisation ; on cherche la valeur de θ qui maximise $Q(\theta/\theta_m)$.

3.1 Application aux HMM d'ordre 1

On a

$$g(y,s) = \prod_t a_{s_t s_{t+1}} b_{s_{t+1}} y(t+1)$$

On appelle $1_{\{s_t=u,s_{t+1}=v\}}$ la fonction indicatrice qui vaut 1 si et seulement $s_t = u$ et $s_{t+1} = v$. On peut regrouper tous les $a_{s_t s_{t+1}}$ identiques et exprimer $g(y,s)$ à l'aide de ces indicatrices.

$$g(y,s) = \prod_{(u,v)\in S^2} a_{uv}^{\sum_t 1_{\{s_t=u,s_{t+1}=v\}}} \prod_t b_{s_{t+1}} y(t+1)$$

Intéressons nous au premier produit de cette expression et voyons sa contribution dans le calcul de $Q(\theta/\theta_m)$.

$$\begin{aligned} Q(\theta/\theta_m) &= E(\log g(Y,S/\theta)/y,\theta_m) \\ &= \sum_{(u,v)\in S^2} E(\sum_t 1_{\{s_t=u,s_{t+1}=v\}}/y,\theta_m) \log a_{uv} + \cdots \end{aligned} \tag{5}$$

L'espérance conditionnelle relativement à θ_m et y de la fonction indicatrice $1_{\{s_t=u,s_{t+1}=v\}}$ vaut

$$E(1_{\{s_t=u,s_{t+1}=v\}}/y,\theta_m) = Prob(s_t = u, s_{t+1} = v/y,\theta_m)$$

Ce qui donne

$$Q(\theta/\theta_m) = \sum_{(u,v)\in S^2} \sum_t Prob(s_t = u, s_{t+1} = v/y,\theta_m) \log a_{uv} + \cdots$$

On maximise $Q(\theta/\theta_m)$ en maximisant chacun des termes. Voyons comment maximiser le premier terme sous la contrainte

$$\sum_{v\in S} a_{uv} = 1, \quad \forall u \in S$$

On introduit la quantité (nulle) $\lambda(1 - \sum_v a_{uv})$ appelée, multiplicateur de Lagrange.

$$\begin{aligned} Q(\theta/\theta_m) = &\sum_{u\in S}\sum_{v\in S}\sum_t Prob(s_t = u, s_{t+1} = v/y,\theta_m) \times \log a_{uv} \\ &+ \sum_{u\in S} \lambda(1 - \sum_v a_{uv}) + \cdots \end{aligned} \tag{6}$$

Dérivons partiellement suivant a_{uv}, il vient

$$\frac{\partial Q(\theta/\theta_m)}{\partial a_{uv}} = \frac{\sum_t Prob(s_t = u, s_{t+1} = v/y,\theta_m)}{a_{uv}} - \lambda = 0$$

On tire la valeur de a_{uv} en fonction de λ et on somme sur tous les v, sachant que l'on a $\sum_v a_{uv} = 1$ on obtient

$$\lambda = \frac{1}{\sum_{v \in S} \sum_t Prob(s_t = u, s_{t+1} = v / y, \theta_m)}$$

d'où la valeur de a_{uv} qui maximise $Q(\theta/\theta_m)$

$$a_{uv} = \frac{\sum_t Prob(s_t = u, s_{t+1} = v / y, \theta_m)}{\sum_v \sum_t Prob(s_t = u, s_{t+1} = v / y, \theta_m)}$$

On maximise de la même façon les autres termes de $Q(\theta/\theta_m)$.

Il reste à calculer $Prob(s_t = u, s_{t+1} = v / y, \theta_m)$. Cette quantité se calcule par récurrence sur u, v et t. à l'aide des fonctions *avant* et *arrière* appelées respectivement α et β et calculées sur le modèle θ_m. On définit $\alpha(t,u)$ par la probabilité d'un alignement partiel jusqu'au temps t.

$$\alpha(t,u) = Prob(s_t = u, y1, \ldots y_t)$$

On a la relation de récurrence

$$\alpha(t,v) = \sum_u \alpha(t-1,u) a_{uv} b_v(y_{t+1}) \tag{7}$$

On peut calculer $Prob(y_1, \ldots y_T)$ à l'aide de $\alpha(T,v)$.

$$Prob(y_1, \ldots y_T) = \sum_{v \in S} \alpha(T,v) \tag{8}$$

De même, on définit la fonction $\beta(t,u)$ par

$$\beta(t,u) = Prob(t_{t+1}, \ldots y_T / s_t = u)$$

qui se calcule par récurrence

$$\beta(t,u) = \sum_{v \in S} a_{uv} b_v(y_{t+1} \beta(t+1,v) \tag{9}$$

Par des applications répétées de la formule de Bayes [5], on montre que

$$\begin{aligned} Prob(s_t = u, s_{t+1} = v / y) &= \frac{Prob(s_t = u, s_{t+1} = v, y)}{Prob(y_1, \ldots y_T)} \\ &= \frac{\alpha(t,u) a_{uv} b_v(y_t) \beta(t+1,v)}{\sum_{v \in S} \alpha(T,v)} \end{aligned} \tag{10}$$

3.2 Complexité et implémentation

On appellera dans la suite N_{inp} le facteur de branchement d'entrée moyen du réseau formé de N états définissant la topologie d'un HMM. Typiquement dans le cas d'un réseau gauche-droite, où tous les états sont visités [1], cette valeur vaut deux. Le calcul de α et β demande $N \times N_{inp} \times T$ opérations.

Afin de réduire le nombre d'opérations dans le calcul de α et β nous avons créé un ensemble de listes et de tables pour accélérer les accès aux éléments non nuls qui interviennent dans les calculs. La description de ces accès va illustrer ce paragraphe sur la complexité.

Calcul de α

Pour le calcul de α dans l'équation 7, il faut retrouver rapidement à partir du couple v l'ensemble des transitions (u,v) telle que a_{uv} soit non nul. On utilise une table qui à chaque état, associe l'ensemble des transitions entrantes. Cette table possède N entrées. Chaque entrée pointe sur une liste de N_{inp} éléments comme illustré ci-dessous :

$$v \to (u_0,v,a_{u_0v}),(u_1,v,a_{u_1v}),\cdots$$

Notons que le calcul de $Prob(s_t = u,s_{t+1} = v/y)$ dans l'équation 10 utilise les mêmes accès.

Calcul de β

Le calcul de β dans l'équation 9 est accéléré quand on accède facilement aux transitions (u,v) à partir de l'état u donnée :

$$u \to (u,v_0,a_{uv_0}),(u,v_1,a_{uv_1}),\cdots$$

Cette table possède N entrées et chaque entrée pointe sur une liste de N_{inp} éléments.

Représentation des matrices α et β

La matrice α a une taille maximale de N réels. Elle est calculée pendant la phase aller (*Forward* en anglais). Pendant la phase retour, elle ne sert qu'aux calculs des comptes représentés par $Prob(s_t = u,s_{t+1} = v/y)$.

La matrice β est moins grosse puisque le calcul de β au temps t se fait en utilisant les seules valeurs de β au temps $t + 1$.

1. Des essais effectués sur des modèles où les états pouvaient être sautés nous ont montré que cette topologie dégradait les performances de reconnaissance. À l'heure actuelle, cette topologie a disparu des systèmes de presque toutes les équipes.

La mise en œuvre de ces deux algorithmes nécessite la résolution de plusieurs problèmes : la représentation des probabilités par des nombres réels infinitésimaux (de l'ordre de 10^{-100}) et la réduction du temps de calcul par des techniques de recherche en faisceaux (élagage en français québécois [6]). Ces problèmes sont maintenant bien documentés. La thèse de Brown [1] décrit comment implanter un seuillage dans les calculs de α et β. Ce seuillage accélère grandement les calculs et permet des expériences qui n'étaient pas possibles auparavant.

Représentation des réels

Le calcul de α dans l'équation 7 provoque rapidement un débordement dans la représentation des probabilités. À chaque étape t, nous normalisons $\alpha(t,u)$ en ramenant la plus grande valeur à 1. Le maximum est sauvegardé dans un tableau $scale[t]$. Appelons $\alpha'(t,u) = \frac{\alpha(t,u)}{scale[t]}$ et $\beta'(t,v) = \frac{\beta(t,v)}{scale[t]}$. Alors que α' ne peut jamais déborder, $\beta'(t,v)$ est toujours sujet à des problèmes de représentation ; mais ces erreurs sont informatives sur l'inadéquation du modèle aux données. L'influence des facteurs d'échelle $scale[t]$ s'annulent au numérateur et dénominateur dans l'équation 10 que l'on calculera de la façon suivante.

$$Prob(s_t = u, s_{t+1} = v/y) = \frac{\alpha'(t,u)a_{uv}b_v(y_t)\beta'(t+1,v)}{\sum_{v \in S} \alpha'(T,v)}$$

4 L'algorithme de Viterbi pour le premier ordre

Décrivons cet algorithme sur un exemple simple de modèle discret du premier ordre. Il est tiré de la thèse de A. Kriouile [5] qui fut mon premier étudiant studieux dans ce domaine. Dans cet exemple, nous considérons un HMM1 “préhistorique” dans lequel chaque densité $b_i()$ est discrète, c'est à dire susceptible d'engendrer un nombre fini de symboles. Il y a 10 ans, tous les HMM utilisaient des densités discrètes afin d'assurer une vitesse raisonnable, sur les machines de l'époque, aux algorithmes d'apprentissage et de reconnaissance.

Soit le modèle de Markov du premier ordre de la figure 3, défini par la matrice de transitions entre états A telle que :

$$A = \begin{pmatrix} 0.3 & 0.5 & 0.2 \\ 0 & 0.3 & 0.7 \\ 0 & 0 & 1 \end{pmatrix}$$

et par la matrice B fixant les probabilités des 2 observations – notées a et b – sur chacun des trois états telle que :

$$B = \begin{pmatrix} 1 & 0 \\ 0.5 & 0.5 \\ 0 & 1 \end{pmatrix}$$

sans oublier le vecteur π donnant la distribution des probabilités initiales sur les trois états :

$$\pi = \begin{pmatrix} 0.6 \\ 0.4 \\ 0 \end{pmatrix}$$

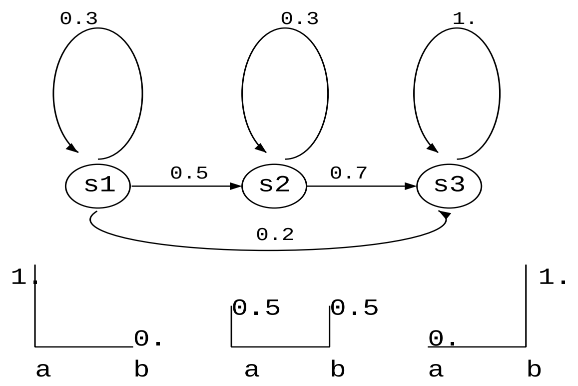


FIG. 3 – *Mon premier modèle de Markov*

Pour trouver le meilleur chemin de T états : $S = (s_1,s_2,\ldots,s_T)$ associé à une suite de T observations : $y = (y_1,y_2,\ldots,y_T)$, on définit $\delta_t(v)$ qui est la probabilité du meilleur alignement état – observation entre les instants 1 et t et s'achevant sur v.

$$\delta_t(v) = \max_{s_t}[Prob(s_1,\ldots s_{t-2},s_{t-1},s_t = v,y_1,\ldots,y_t/\theta)], \quad \begin{matrix} 1 & \le v & \le N \\ 1 & \le t & \le T \end{matrix}$$

Une méthode récursive de calcul est donnée par la formule suivante :

$$\delta_t(v) = \max_{1 \le u \le N}[\delta_{t-1}(u) \cdot a_{uv}] \cdot b_v(y_t), \quad \begin{matrix} 1 & \le v & \le N \\ 1 & \le t & \le T \end{matrix} \tag{11}$$

avec l'initialisation suivante :

$$\delta_0(i) \quad = \quad \Pi_i, \qquad 1 \leq i \leq N$$

Le déroulement de l'algorithme est illustré par la figure 4.

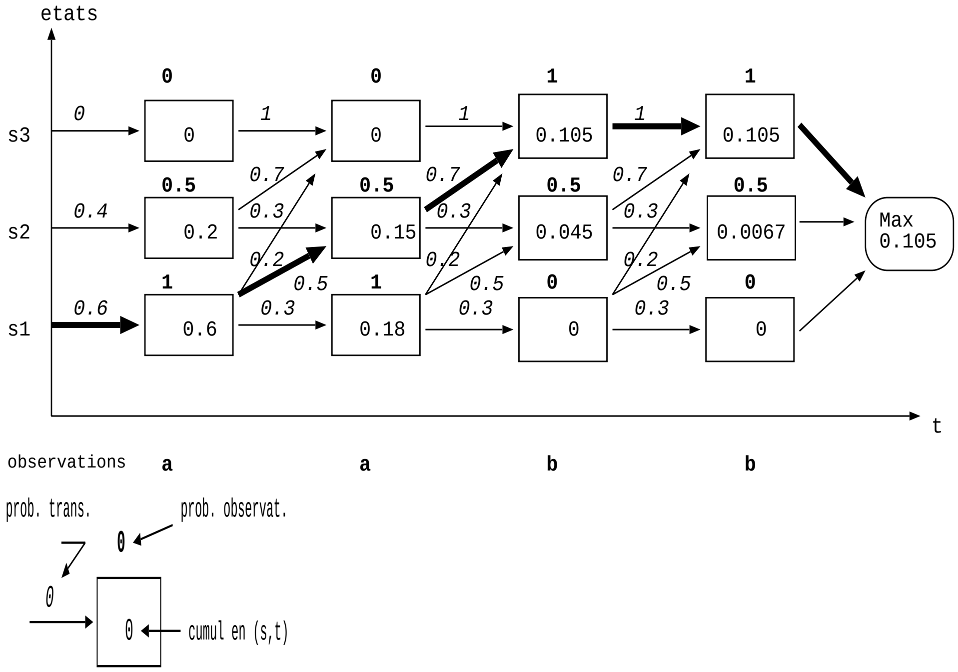


FIG. 4 – *Calcul de δ pour la suite d'observations* aabb

4.1 L'algorithme de Viterbi pour les suites de modèles

En reconnaissance de la parole continue, les frontières temporelles des unités à reconnaître ne sont généralement pas connues. On peut connecter les différents HMM dans un réseau plus vaste modélisant la suite d'unités à reconnaître. Le résultat de l'interconnexion reste un HMM sur lequel l'algorithme de Viterbi peut être utilisé pour retrouver la suite des états cachés correspondant à une élocution. La segmentation en unités est alors un produit secondaire de la reconnaissance. Un cas réel est présenté dans la figure 5. Dans cet exemple, on a représenté chaque phonème du français par un HMM2. La phrase prononcée est : « C'est en forgeant qu'on devient forgeron ». Une phonétisation possible de la phrase est donnée verticalement. La correspondance entre les phonèmes du français et leur représentation dans le programme d'alignement est donnée dans la table 1(d'après Calliope [2]).

				Consonnes				
[p]	paie	cl p	[t]	taie	cl t	[k]	quai	cl k
[b]	baie	vcl b	[d]	dais	vcl d	[g]	gai	vcl g
[m]	mais	m	[n]	nez	n	[ɲ]	gagner	
[f]	fait	f	[s]	sait	s	[ʃ]	chez	S
[v]	vais	v	[z]	zéro	z	[ʒ]	geai	Z
[ω]	ouais	w	[j]	yéyé	j	[l]	lait	l
[ʀ]	raie	R						
				Voyelles				
[i]	lit	i	[y]	lu	y	[u]	loup	u
[e]	les	e	[ø]	leu	2	[o]	lot	o
[ε]	lait	E	[ɔ]	lotte	O	[a]	là	a
[ə]	le	@	[ε̃]	lin	in	[ã]	lent	an
[õ]	long	on						

TAB. 1 – *Les phonèmes du français et leur représentations dans les alignements*

En niveaux de gris sont représentées les vraisemblances des trames calculées à l'aide des densités définissant les états. Comme ces vraisemblances sont normalisées pour que leur somme sur l'ensemble des modèles soit constante à chaque instant, on peut les interpréter comme des probabilités *a posteriori* des états connaissant les trames. On suppose alors les états *a priori*équiprobables.

Références

[1] Brown (P.F.). – *The Acoustic-Modeling Problem in Automatic Speech Recognition.* – Thèse de PhD, C.S.D., Carnegie Mellon University, 1987.

[2] Calliope. – *La parole et son traitement automatique.* – Masson, 1989.

[3] Dempster (A.P.), Laird (N.M.) et Rubin (D.B.). – Maximum-Likelihood From Incomplete Data Via The EM Algorithm. *Journal of Royal Statistic Society, Ser. B (methodological)*, vol. 39, 1977, pp. 1 – 38.

[4] Forney (G.D.). – The Viterbi Algorithm. *IEEE Transactions*, vol. 61, March 1973, pp. 268–278.

[5] Kriouile (A.). – *La reconnaissance automatique de la parole et les modèles de Markov cachés : modèles du second ordre et distatnce de Viterbi à optimalité locale.* – Thèse de PhD, Université de NANCY 1, 1990.

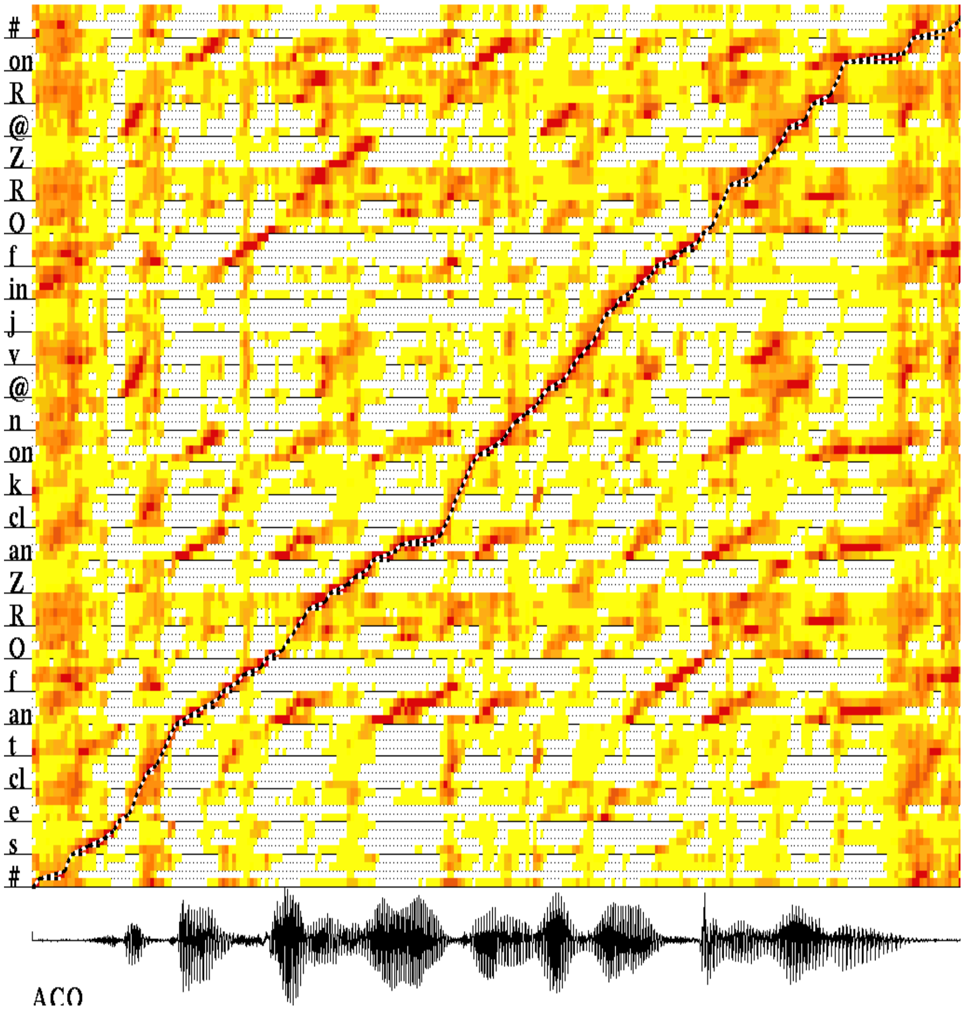


FIG. 5 – *Le meilleur chemin construit par l'algorithme de Viterbi détermine une segmentation en phonèmes dans la phrase :* « C'est en forgeant qu'on devient forgeron »

[6] Lacouture (R.). – *Algorithmes de recherche lexicale.* – Thèse de PhD, Université de Mac Hill, 1995.

[7] Mury (F.). – *Comparaison d'algorithmes d'identification de chaînes de Markov cachées et application à la détection de régions homogènes dans les séquences d'ADN.* – Thèse de doctorat, Université René Descartes, Paris V, 1997.

[8] Poritz (A.B.). – Hidden Markov Models: A Guided Tour. *In: Proceedings of IEEE-International Conference On Acoustics, Speech, and Signal Processing*, pp. 7 – 13. – New York, 1988.