Optimisation d'un arbre de decision par un algorithme de recuit simule ; application a la reconnaissance des voyelles et semi-voyelles

Valérie Le Maire

INRIA

UNITÉ DE RECHERCHE
INRIA-SOPHIA ANTIPOLIS

Institut National
de Recherche
en Informatique
et en Automatique

Domaine de Voluceau
Rocquencourt
B.P.105
78153 Le Chesnay Cedex
France
Tél.:(1) 39 63 55 11

Rapports de Recherche

N° 854

OPTIMISATION D'UN ARBRE DE DECISION PAR UN ALGORITHME DE RECUIT SIMULE ; APPLICATION A LA RECONNAISSANCE DES VOYELLES ET SEMI-VOYELLES.

Valérie LE MAIRE

JUIN 1988

RR_0854

IRISA INSTITUT DE RECHERCHE EN INFORMATIQUE
ET SYSTÈMES ALÉATOIRES

Campus Universitaire de Beaulieu
35042 - RENNES CÉDEX
FRANCE
Téléphone : 99 36 20 00
Télex : UNIRISA 950 473 F
Télécopie : 99 38 38 32

Publication Interne n° 402
Avril 1988
34 Pages

Optimisation d'un arbre de décision par un algorithme de recuit simulé; Application à la reconnaissance des voyelles et semi-voyelles.

Valérie Le Maire.

Résumé.

Ce rapport présente une utilisation de l'algorithme de recuit simulé. Un site ou un état est un arbre de décision, leur ensemble est muni d'une structure de voisinages appropriée, et la fonction à minimiser est un critère lié à la vraisemblance de l'arbre.

Cette étude est appliquée au décodage acoustico-phonétique en reconnaissance de la parole. Le but est l'étiquetage des segments du signal qui correspondent à des voyelles et semi-voyelles. Les feuilles des arbres de décision représentent les classes des voyelles et semi-voyelles de la langue française.

Decision tree optimisation by an annealing algorithm; Application to vowel recognition.

Abstract.

In this paper we describe an annealing algorithm utilisation. A state is a decision tree, provided by an appropriate neighbourhood and the criterion to minimise is a function of the tree likelihood.

The application is the labelling of speech segments in an acoustic-phonetic decoding. The terminal vertices of the decision tree represent the classes of vowels and semi vowels, and the first experiments concern only the vocalic segments recognition.

Introduction.

Dans le cadre du décodage acoustico-phonétique en reconnaissance de la parole, une phrase prononcée est représentée par un signal sur lequel sont effectués les trois traitements suivants:

1. Segmentation automatique du signal,

   c'est-à-dire détection des changements et estimation de leur localisation.

2. Identification des unités et frontières détectées.

   Ce traitement s'effectue en trois étapes:

   - étiquetage des frontières,
   - classification en zones transitoires et zones homogènes,
   - identification des zones homogènes.

3. Décodage phonétique de la phrase prononcée.

Le travail réalisé prend place dans l'étape qui vise à identifier certains segments homogènes correspondant aux voyelles et semi-voyelles.

Nous considérons les arbres de décision dont les feuilles sont les classes phonétiques. A partir d'un arbre initialement construit en utilisant un critère de vraisemblance, nous voulons déterminer l'arbre de vraisemblance maximum. Pour cela, nous utilisons un algorithme stochastique appelé "recuit simulé" ("annealing algorithm" en anglais), qui permet d'atteindre un minimum global d'une fonction.

Ce document comprend quatre parties. Après une présentation de la partie statistique (arbre de décision, maximum de vraisemblance), nous étudierons l'algorithme de recuit simulé. La troisième partie sera consacrée à l'application de l'algorithme précédent à la recherche d'un meilleur arbre de décision possible. Enfin nous comparerons les efficacités en terme de reconnaissance de l'arbre initial et de l'arbre optimal obtenu.

Plan:

1. Partie statistique, [10].

1.1. Discrimination.

Soient Ω un ensemble de R^k et $\{ \Omega_l, l=1 \text{ à } s \}$ une partition de Ω.

Un individu i de Ω est caractérisé par un vecteur de k variables: $X_i \in R^k$.

Le problème est d'affecter une observation X_i à un des sous-groupes $\Omega_1, \ldots \Omega_S$. Une règle possible qui minimise le nombre de décisions erronnées par nombre de décisions prises consiste à affecter X_i à Ω_l si

$$P(\Omega_l / X_i) > P(\Omega_h / X_i), \quad \forall h, h = 1 \text{ à } s, h \neq l$$

Cette règle est la "règle de l'erreur minimale de Bayes".

En général, $P(\Omega_l/X_i)$ est inconnue. Mais $P(\Omega_l/X_i)$ est proportionnelle à $P(X_i/\Omega_l)\, P(\Omega_l)$, d'après le théorème de Bayes.

Les valeurs de $P(\Omega_l/X_i)$ peuvent être évaluées:

-soit par une méthode paramétrique (le plus souvent avec une hypothèse de normalité)

-soit par une méthode non paramétrique (à l'aide d'une discriminante).

Cette dernière méthode est plus efficace quand la distribution des données est inconnue ce qui sera notre cas.

De plus, afin d'éviter la considération de bornes (les seules valeurs possibles des probabilités sont comprises entre 0 et 1), une discriminante logistique est préférée à une discriminante linéaire.

1.2. Modèle logistique.

Avant de présenter la discriminante logistique, et pour alléger l'écriture dans la suite du document, nous définissons les notations suivantes:

X'_i représente le vecteur ($x_{i0}, x_{i1}, ..., x_{ik}$)

où ($x_{i1}, ..., x_{ik}$) = X_i,

et x_{i0} est la valeur d'une variable fictive toujours égale à 1.

$< , >$ représente le produit scalaire canonique de R^{k+1}.

Dans le cas de 2 classes (s=2), la surface entre Ω_1 et Ω_2 est modélisée à l'aide de $\beta \in R^{k+1}$ par

$$\log \frac{P[\Omega_1 / X_i]}{P[\Omega_2 / X_i]} = <\beta , X'_i>$$

Alors $P[\Omega_l / X_i]$ est calculable à l'aide de β:

$$P[\Omega_1 / X_i] = \frac{\exp <\beta, X'_i>}{1 + \exp <\beta, X'_i>} ,$$

$$P[\Omega_2 / X_i] = \frac{1}{1 + \exp <\beta, X'_i>} .$$

Cela se généralise au cas de s classes à l'aide de $\{ \beta'_l, \beta'_l \in R^{k+1}, l=1 \text{ à } s\}$:

$$P[\Omega_l / X_i] = \frac{\exp <\beta'_l, X'_i>}{1 + \sum_{h=1}^{s} \exp <\beta'_h, X'_i>} , \text{ pour h=1 à s.}$$

Mais comme $\sum_{l=1}^{s} P[\Omega_l / X_i] = 1$, cette formule est redondante.

En posant $\beta_l := \beta'_l - \beta'_s$, le modèle logistique devient:

$$P[\Omega_l / X_i] = \frac{\exp <\beta_l, X'_i>}{1 + \sum_{h=1}^{s-1} \exp <\beta_h, X'_i>} \quad \text{pour } l \neq s$$

$$\text{et} \quad P[\Omega_s / X_i] = \frac{1}{1 + \sum_{h=1}^{s-1} \exp <\beta_h, X'_i>}$$

$\beta := \{ \beta_l , l=1 \text{ à } s-1 \}$ est appelé "classificateur".

β est estimé par la méthode du maximum de vraisemblance, à partir d'un échantillon de n individus déjà classés, appelé ensemble d'entraînement.

Nous notons $z_{il} = 1$ si le $i^{ème}$ individu appartient à la $l^{ème}$ catégorie,

0 sinon.

La vraisemblance du classificateur β pour le $i^{ème}$ individu est

$$L_i(\beta) = \prod_{l=1}^{s} \left(P_\beta [\Omega_l / X_i] \right)^{z_{il}}$$

Sous l'hypothèse d'indépendance entre les individus, la vraisemblance de β est

$$L(\beta) = \prod_{i=1}^{n} \prod_{l=1}^{s} \left(P_\beta [\Omega_l / X_i] \right)^{z_{il}}$$

Soit en passant au logarithme:

$$l(\beta) = \sum_{i=1}^{n} \sum_{l=1}^{s} z_{il} \log P_\beta [\Omega_l / X_i]$$

Il faut donc trouver β tel que cette fonction soit maximum. Comme il n'y a pas de solution exacte, nous utilisons l'algorithme de Newton-Raphson: β est calculé itérativement par la formule:

$$\beta_{n+1} = \beta_n - H^{-1}(\beta_n)\ G(\beta_n)$$

où H est la matrice Hessienne,

et G est le vecteur Gradient de la fonction à maximiser.

1.3. Arbre de décision binaire.

Un arbre de décision est une suite de partitions emboitées par finesse décroissante. A chaque noeud de l'arbre est associée la fonction de décision (ou classificateur) relative à une fusion de classes.

L'avantage majeur d'un arbre de décision par rapport à une discrimination classique comme celle décrite dans le paragraphe précédent, est la décomposition d'une décision globale en une suite de décisions simples et locales.

Mais le choix de la structure de l'arbre de décision sera prépondérant dans l'évaluation de ses qualités discriminantes. D'où l'importance de l'algorithme de construction d'un arbre de décision.

1.4. Construction d'un arbre de décision binaire.

La meilleure méthode est celle qui consiste à essayer tous les arbres binaires possibles. Le nombre d'arbres binaires à s feuilles est $B(s) = \frac{(2s - 3)!}{2^{s-2}(s - 2)!}$ pour $s \geq 2$.

Une recherche exhaustive est donc impossible (pour s = 7, la valeur atteinte est déjà 10935).

Une solution non optimale est donnée par l'algorithme suivant:

1. calculer tous les modèles logistiques (β) correspondant aux derniers noeuds non terminaux de l'arbre,
2. choisir le noeud pour lequel $l(\beta)$ est maximum,
3. fusionner les classes correspondant au noeud choisi,
4. retourner en 1 si la racine de l'arbre n'est pas atteinte.

Mais rien ne permet d'affirmer que l'arbre de décision ainsi obtenu est le meilleur arbre possible. Nous avons essayé d'améliorer cet arbre à l'aide d'un algorithme stochastique: l'algorithme de recuit simulé.

2. Algorithme de recuit simulé.

2.1. Présentation.

L'algorithme de recuit simulé permet d'atteindre le minimum d'une fonction sur un ensemble, sans faire une recherche exhaustive. La méthode utilisée adopte une idée issue de la mécanique statistique: simuler le refroidissement lent d'un système physique, pour atteindre les états d'énergie minimum. Pour cela, on applique de petites perturbations à une solution courante, et alors qu'on accepte toujours que la fonction décroisse, on accepte qu'elle ne croisse qu'avec une probabilité non nulle, contrôlée par un paramètre: la "température" T. En appliquant successivement cet algorithme, dit de "métropolis" ou de "relaxation stochastique", pour chacune des températures d'une suite de valeurs initialement grandes et lentement décroissantes vers zéro, on simule le processus de refroidissement lent appelé "recuit".

2.2. Description de l'algorithme, [3],[4],[7].

Nous devons minimiser une fonction U sur un ensemble fini d'états: S, muni d'une structure de voisinage: $\{N_s, s\in S\}$, où N_s est inclu dans S, et représente le voisinage de s. On suppose qu'il existe une matrice de transition R sur S telle que:

$$R(s,s')>0 \quad \Rightarrow \quad s'\in N_s.$$

Un tel système est noté: (S,U,R). Soit $(T_1,T_2,...)$ une suite de nombres strictement positifs telle que:

$$T_1\geq T_2\geq ... \qquad \text{et} \qquad \lim_{k\to+\infty} T_k = 0$$

Une suite d'états $X_0,X_1,..$, est construite à partir de l'algorithme séquentiel suivant:

- un état initial X_0 est choisi.
- connaissant l'état à l'instant k: $X_k=s$,

on choisit un candidat voisin Y_k dans N_s,

avec la probabilité $P(Y_k=s' \mid X_k=s)=R(s,s')$

Si $p_k=\exp(-\Delta U^+/T_k)$ où $\Delta U=U(Y_k)-U(X_k)$,

alors $X_{k+1}:=\begin{cases} Y_k \text{ avec la probabilité } p_k, \\ X_k \text{ sinon.} \end{cases}$

Cela spécifie la manière dont la suite $X_1,X_2,..$est choisie.

Les propriétés de cet algorithme sont décrites dans le paragraphe suivant. En

particulier, elles donnent une condition suffisante de convergence de l'algorithme vers un des états en lesquels U est minimal.

2.3. Description théorique, [4].

Le processus aléatoire $X = (X_k, k>0)$ produit par cet algorithme, est une chaîne de Markov à temps discret non homogène qui a pour probabilité de transition:

$$P(s,s') = P(X_{k+1} = s' / X_k = s)$$

$$= \begin{cases} R(s,s') \exp\left(- \dfrac{(U(s')-U(s))^+}{T_k}\right) & \text{si } s \neq s' \\ R(s,s) + \sum_t R(s,t) \left(1 - \exp\left(- \dfrac{(U(t)-U(s))^+}{T_k}\right)\right) = 1 - \sum_{t \neq s} P(s,t) & \text{sinon.} \end{cases}$$

Le but de ce paragraphe est de montrer à quelles conditions:

$$\lim_{k \to +\infty} P(X_k \in S^*) = 1$$

où S* représente le sous-ensemble des états de S en lesquels U atteint sa valeur minimale.

Definitions:

Un chemin de i à j est une suite d'états $i_0, i_1, \ldots i_{p-1}, i_p$, telle que

- $i_0 = i$, $i_p = j$,
- $R(i_k, i_{k+1}) > 0$, $\forall k: 0 \leq k < p$

Un chemin d'altitude inférieure à un réel E de i à j, est un chemin de i à j tel que pour tout état s sur ce chemin, on aît $U(s) < E$.

Propriétés:

(S,U,R) est fortement irréductible

si pour tout couple d'états (i,j) , il existe un chemin de i à j.

Cette propriété est la connexité du graphe (S,R).

(S,U,R) est faiblement réversible

si pour un réel E, et deux états i et j, il existe un chemin de i à j d'altitude inférieure à E, alors il en existe aussi un dans l'autre sens.

Dans le cas particulier de la mise en oeuvre de l'algorithme avec une température constante T (c'est-à-dire le cas de l'algorithme de relaxation stochastique), la chaîne de Markov obtenue est homogène, stationnaire et possède (si $0<T\leq\infty$) une unique mesure à l'equilibre Π_T: l'unique mesure invariante par la matrice des probabilités de transition, et on a:

$$\Pi_T(s) = \frac{\Pi_\infty(s)\exp(-\frac{U(s)}{T})}{Z_T} \qquad \text{pour } 0<T<+\infty$$

$$\text{où } Z_T = \sum_s \Pi_\infty(s)\exp(-\frac{U(s)}{T})$$

L'hypothèse de forte irréductibilité de R permet, avec le fait que P est apériodique si $S\neq S^*$, de conclure par le théorème ergodique de Markov (la chaîne est récurrente irréductible et apériodique) que

$$\lim_{k\to\infty} P(X_k\in S^*) = \sum_{s\in S^*}\Pi_T(s)$$

Il en découle que

$$\lim_{k\to\infty} P(X_k\in S^*) = \frac{\sum_{s\in S^*}\Pi(s)\exp(-\frac{U(s)}{T})}{\sum_{s\in S}\Pi(s)\exp(-\frac{U(s)}{T})}$$

peut être aussi proche de 1 que l'on veut en choisissant T suffisament petit, c'est-à-dire:

$$\lim_{T\to 0}\ (\lim_{\substack{k\to\infty \\ T_k=T}} P(X_k\in S^*)) = 1$$

L'idée du recuit simulé est d'essayer d'obtenir

$$\lim_{k\to+\infty} P(X_k\in S^*) = 1$$

en laissant T_k tendre vers 0 quand $k\to\infty$.

Définitions:

On appelle coupe, toute composante connexe de la restriction du graphe (S,R) aux sites de potentiels inférieurs à un réel E.

On appelle profondeur d'une coupe C, le réel qui vérifie:

$$d = \min\{U(s), s\notin C / (\exists\, s'\in C, R(s',s)>0)\} - \min\{U(s), s\in C\}$$

Théorème:

sous les hypothèses de

. forte irréductibilité

. réversibilité faible

. décroissance vers 0 de la suite des températures $(T_k, k>0)$,

on a:

(a) Pour tout état s de S différent d'un minimum local de U,

$$\lim_{k\to\infty} P(X_k = s) = 0$$

(b) Si l'ensemble B (appelé base de C) des états d'énergie minimale d'une coupe C de profondeur d ne contient que des minima locaux de U,

alors $$\lim_{k\to\infty} P(X_k \in B) = 0 \quad \Leftrightarrow \quad \sum_{k=1}^{\infty} \exp(-\frac{d}{T_k}) = +\infty$$

(c) (conséquence de (a) et (b)) Si d* représente le maximum des profondeurs des coupes dont les bases ne contiennent que des minima locaux et non globaux, alors

$$\lim_{k\to\infty} P(X_k \in S^*) = 1 \quad \Leftrightarrow \quad \sum_{k=1}^{\infty} \exp(-\frac{d^*}{T_k}) = +\infty$$

Remarques :

A température élevée, T grand, tous (ou presque) les candidats voisins sont admis. Si T=+∞, alors l'algorithme est celui de promenade aléatoire sur S.

A température faible, T proche de zéro, presque seuls les candidats qui font décroître la fonction, sont admis. Si T=0, l'algorithme est celui de descente systématique de U sur S, et on obtient alors un minimum local de U.

Notons N_s^+ le sous ensemble du voisinage de s dans lequel la valeur de U est supérieure à U(s). Si y est un élément de N_s^+, alors plus l'accroissement ΔU de U (ΔU=U(y)-U(s)) est grand, moins y aura de chance d'être admis, sauf quand l'accroissement de U est suffisamment compensé par T (puisque la probabilité que y soit admis est $P_y = \exp\left(-\frac{(U(y) - U(s))^+}{T}\right)$).

Toute la difficulté de la mise en oeuvre d'un tel algorithme réside dans le choix de T_n. Une conséquence de la théorie est la règle:

$$T_n = \frac{a}{\text{Log}(n+1)}$$

où a est une constante liée à la fonction à minimiser. En effet, il suffit d'avoir $a \geq d^*$ pour pouvoir conclure que $\lim_{k \to +\infty} P(X_k \in S^*) = 1$. Mais la convergence est trop lente pour la pratique. Il semble qu'il faille prendre une valeur du paramètre T qui respecte l'ordre de grandeur des ΔU.

3. Application de cette méthode à la reconnaissance des voyelles.

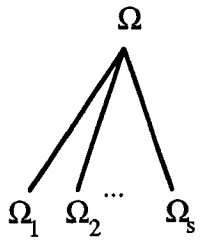
L'algorithme de recuit simulé est appliqué au cas d'un graphe dont les sommets sont des arbres de décision.

3.1. Voisinage d'un arbre, [1],[9].

D'après la partie 1, pour un ensemble Ω de données, un noeud permet de définir s catégories qui respectent la classification voulue.

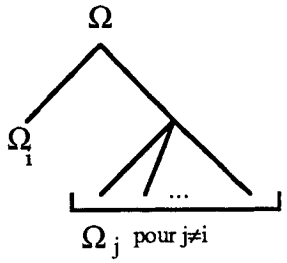
Définition: Nous dirons que deux arbres sont **voisins** si

l'un d'eux n'utilise qu'un seul noeud pour séparer les données de l' ensemble Ω en s catégories, $\Omega_1,..\Omega_S$,

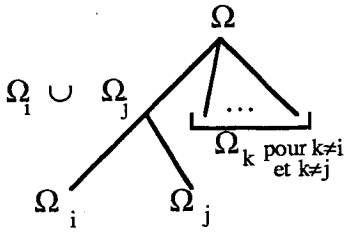


alors que l'autre en utilise deux, de l'une des manières suivantes:

1er cas: le premier des 2 noeuds est binaire et sépare une des catégories Ω_i, de la réunion de toutes les autres,
et le deuxième sépare les catégories Ω_j pour $j \neq i$:



2ème cas: le deuxième noeud est binaire et sépare 2 des catégories Ω_i et Ω_j
et le premier sépare les autres classes et celle formée de $\Omega_i \cup \Omega_j$



(En pratique, pour des questions de temps de calcul, on se bornera aux arbres au plus quaternaires, donc à $s \leq 4$.)

3.2. Fonction à minimiser: Potentiel.

Reprenons les notations du paragraphe 1. Les observations appartiennent à R^7; les variables utilisées sont l'énergie et les 6 premiers coefficients `spectraux`.

Définition:

La vraisemblance d'un arbre de décision est égale au produit des vraisemblances des classificateurs de chacun des noeuds.

Nous notons β_ν, le classificateur correspondant au noeud ν,

et s_ν, le nombre de classes fusionnées en ce noeud.

Nous écrivons donc que:

$$L(\{\beta_\nu, \nu: \text{noeud de l'arbre}\}) = \prod_{i \in \Omega} \prod_{\nu, \text{noeud de l'arbre}} \prod_{l=1}^{s_\nu} (P_\beta(\Omega_l / X_i))^{z_{il}}$$

soit, en passant au Log:

$$l(\{\beta_\nu, \nu: \text{noeud de l'arbre}\}) = \sum_{i \in \Omega} \sum_{\nu, \text{noeud}} \sum_{l=1}^{s_\nu} z_{il} \log P_\beta(\Omega_l^\nu / X_i)$$

$$= \sum_{\nu, \text{noeud}} \sum_{l=1}^{s_\nu} \sum_{i \in \Omega_l} \log P_\beta(\Omega_l^\nu / X_i)$$

L'arbre pour lequel ce critère est maximum est le meilleur possible, et ce critère est bien représentatif du choix des étapes. On le choisit donc comme la fonction à maximiser.

On définit le critère d'un arbre X par:

$$c(\{\beta\}) = \sum_\nu c_\beta(\nu) \quad \text{avec} \quad c_\beta(\nu) = \sum_{l=1}^{s_\nu} \sum_{i \in \Omega_l} \text{Log } P_\beta(\Omega_l / X_i)$$

Le potentiel U est défini comme l'opposé du critère. Comme seuls les noeuds modifiés voient leur participation au critère modifiée (dernière formule du §1.3.), la différence de potentiel ΔU est donnée par la formule:

$$\Delta U = \sum_{v(Y)\neq v(X)} (-c(v(Y))) - \sum_{v(X)\neq v(Y)} (-c(v(X)))$$

où X note l'arbre courant , Y l'arbre candidat,

v(X) un noeud de X, et v(Y) un noeud de Y.

Cette expression contiendra au plus 3 termes d'après la relation de voisinage décrite précédemment.

3.3. Choix de la décroissance de la température.

La règle $T_n = {}^{a}/_{\log(n+1)}$ n'étant pas pratiquable, nous cherchons une décroissance de la température plus rapide, tout en restant efficace. Nous imposons notamment que T_{n+1} respecte l'ordre de grandeur de la partie positive de l'accroissement positif du potentiel observé, de sorte que le processus puisse s'échapper des minima locaux.

Nous nous proposons de calculer récursivement ces accroissements ΔU_m^+ de potentiel observés à chaque instant par la formule, [2]:

$$\Delta U_m^+(n+1) = \Delta U_m^+(n) + \gamma\,(\Delta U^+(n) - \Delta U_m^+(n))$$

où $\Delta U^+(n)$ est la partie positive de la dernière différence de potentiel observée, et γ est une constante d'oubli.

Ainsi $\Delta U^+(n)$ est positif lorsque le site courant de l'algorithme est un minimum local du potentiel (ΔU^+ est d'autant plus grand que le minimum est encaissé, c'est-à-dire que la différence de potentiel moyenne sur son voisinage est grande).

T_{n+1} est proportionnelle à la valeur obtenue:

$$T_{n+1} = T\,.\,\Delta U_m^+(n+1) \quad \text{où T est une constante.}$$

On espére ainsi permettre à la température d'augmenter suffisamment pour échapper aux minima locaux. Les paramètres à ajuster (γ, T) doivent être tels que le site courant de l'algorithme s'échappe au mieux des minima locaux.

3.4. Choix du critère d'arrêt.

Il est évident qu'une règle heuristique est nécessaire pour définir la convergence (cf. paragraphe 2.3.). L'algorithme est arrêté lorsque les candidats voisins sont refusés depuis au moins F itérations, F étant fixé a priori.

Théoriquement, F doit être suffisamment grand pour être sûr d'avoir parcouru tous les sites voisins. On espère ne pas en être trop loin en prenant F = 500, car le temps requis pour une étude plus approfondie serait trop long.

3.5. Résultats.

Les expérimentations sont réalisées à partir d'un corpus de voyelles et le cadre est mono-locuteur. Sont considérées les 11 voyelles suivantes:

/ a , i , j , o , an , w , u , y , e , ∂ , on /

L'ensemble d'apprentissage est construit à partir de trois listes de phrases phonétiquement équilibrées; l'ensemble test est extrait de deux listes de phrases phonétiquement équilibrées.

Les figures sont réunies dans l'annexe (A.4.).

3.5.1. Recherche de la température initiale.

Afin de déterminer une température pour laquelle tous les candidats voisins ou presque sont admis, nous avons utilisé l'algorithme à température constante pour différentes valeurs de T pendant 500 itérations, et à partir du même arbre initial. Les résultats obtenus nous ont conduit à un compromis entre la promenade aléatoire dans l'ensemble des états, et des temps de calcul supportables. T_0=50 ou 25.

3.5.2. Algorithme de recuit simulé.

1ère expérience.:

Nous avons étudié les décroissances du potentiel pour une suite de températures constante par morceaux:

- l'algorithme de relaxation stochastique est appliqué à n processus initialisés à un même site s, pendant p itérations avec la température T_k.
- à l'issue de ces p itérations, sont extraits le potentiel moyen $U_m(p)$ atteint et le site d'énergie minimale parmi ces n*p sites visités, soit s_*, qui servira de site initial pour l'algorithme à la température T_{k+1}.

L'arbre initial est X_0 (figure 1): l'arbre obtenu par le "calcul direct" (procédure décrite dans le paragraphe 1.4), n = 10, p = 100 et la suite décroissante des températures est: (50,45,40,35,30,25,22,19,16,13,10,8,6,4,3,2,...)

Le meilleur arbre X_2 (figure 2) est déjà atteint pour T_k=13, soit après avoir visité (10*n*p =) 10000 sites.
En examinant les meilleurs arbres obtenus pour chaqué température, on observe qu'il existe un arbre binaire X_3 (figure 3) meilleur que l'arbre initial au sens du critère considéré.

2ème expérience:.

Dans cette expérience, la température a été choisie décroissante en fonction de ΔU^+ (cf paragraphe 3.3.).

$\gamma = 0.01$; $T = 0,2$; $T_0 = 25$ et $\Delta U_m^+(0) = T_0/T = 125$.

Une premiere tentative a été faite avec comme arbre initial X_0 (figure1). Son potentiel est donc assez petit: $U_0 = 477,16$. L'arbre atteint est
X_4 (figure 4) dont le potentiel est $U_4 = 314,59$.
Afin d'accentuer le rôle de l' algorithme de recuit simulé dans ce cadre, une deuxième tentative a été faite, avec comme arbre initial X_5 (figure 5), choisi "très mauvais": il est impossible de déterminer les classificateurs correspondant à certains noeuds. Dans ce cas, la valeur -10^8 est affectée au critère. Le potentiel de l'arbre est donc très grand: U_5 est de l'ordre de 10^8. Plusieurs arbres ont été atteints:
X_7 (figure 7) dont le potentiel est $U_7 = 292,43$,
X_8 (figure 8) dont le potentiel est $U_8 = 306,19$.

Différentes tentatives avec les mêmes initialisations, ayant abouti à des arbres de potentiels différents, nous ne pouvons affirmer avoir exhibé le meilleur arbre de décision. Une expérimentation a montré qu'un choix mieux adapté de F (paragraphe 3.4) permettrait certainement de remédier à cela.

Toutefois, nous avons obtenu plusieurs arbres de potentiels plus faibles que l'arbre initial, les meilleurs étant ceux des figures 2 et 3 (ce dernier est un arbre binaire) Afin de vérifier que ces arbres constituent effectivement de meilleurs classificateurs, nous allons comparer leurs efficacités en terme de reconnaissance de la parole dans le paragraphe suivant.

4. Evaluation de la Reconnaissance à l'aide des arbres.

4.1. Méthode.

Le but est de reconnaître les phonèmes d'un ensemble à l'aide de l'arbre de décision afin d'évaluer l'efficacité de ce dernier en terme de reconnaissance de la parole. La démarche utilisée est la suivante:

à chaque noeud en partant de la racine, la $i^{ème}$ donnée de l'ensemble est affectée à la branche de plus forte probabilité.
Or, en reprenant les notations du paragraphe 1, la probabilité que la $i^{ème}$ donnée appartienne à la $l^{ème}$ feuille Ω_l est estimée par:

$$P(\Omega_l / X_i) = \prod_{\Omega_{l_h}} P(\Omega_{l_h} / X_i)$$

où (Ω_{l_h}) est la suite induite par l'arbre, des ensembles qui contiennent Ω_l.
Les résultats fournis sont les matrices de confusion et taux de reconnaissance.

4.2. Résultats.

De manière à comparer les résultats avec ceux obtenus par une méthode de quantification vectorielle [11], nous avons réuni les éléments des classes des "e" et des "ε".

En cours d'études, nous avons relevé différents arbres dont nous avons aussi évalué les efficacités en reconnaissance.

Une première expérimentation a consisté à prendre comme données les éléments de l'ensemble d'apprentissage lui-même, ceci afin plutôt de se persuader de la bonne représentativité de l'ensemble d'apprentissage dans l'ensemble des phonèmes, que de fournir une évaluation de l'efficacité des arbres.

Les résultats sont résumés par leurs taux de reconnaissance dans le tableau suivant, et restitués de manière plus précise en annexe (A.3.).

		1ère tentative				2ème tentative				autre
	QV	arbre initial X0	arbres atteints			arbre initial X5	arbres atteints			X9
			X2	X3	X4		X7	X8	F=2000 X6	
U		477	288	309	314	$\approx 2.10^8$	292	306	294	443
ensemble d'apprentissage	93	83	86	85	85		87	84	88	86
ensemble test	73	70	73	72	72		74	74	76	75

Comparaison des taux de reconnaissance (en pourcentage).

4.3. Conclusion.

Les arbres de décision obtenus par l'algorithme de recuit simulé (X_2, X_3, X_4, X_6 , X_7 et X_8) améliorent les taux de reconnaissance des arbre initiaux (X_0, X_5). Les taux de reconnaissance obtenus en utilisant des arbres de décision sont plus faibles qu'en utilisant la quantification vectorielle dans le cas de l'ensemble d'apprentissage, alors qu'ils en sont proches dans le cas de l'ensemble test. Il semble donc que les arbres de décision fournissent une classification moins dépendante de l' ensemble d'entraînement que la quantification vectorielle.

En outre il est peut-être possible d'améliorer encore les arbres obtenus, en s'inspirant des "sous-dictionnaires" utilisés en quantification vectorielle, c'est-à-dire en divisant les classes phonétiques (= feuilles des arbres considérés ici) en sous-classes plus petites, dont le nombre pourrait être fixé par l'analyse des données [8] (il ne faut pas tomber dans l'excès d'une classe par donnée!). La classe phonétique est la réunion des sous-classes, donc des feuilles correspondant au même phonème. On obtiendrait ainsi des classificateurs plus fins, mais aussi plus liés à l'ensemble d'entraînement.

D'autre part, le tableau précédent montre que l'efficacité d'un arbre de décision n'est pas croissante avec sa vraisemblance ($U_2 < U_9$ mais X_9 est plus efficace que X_2). Un potentiel tenant compte non seulement de la vraisemblance des arbres de décision, mais aussi de fonctions plus compliquées (où les paramètres seraient la vraisemblance et le taux de reconnaissance par exemple) serait peut-être mieux adapté.

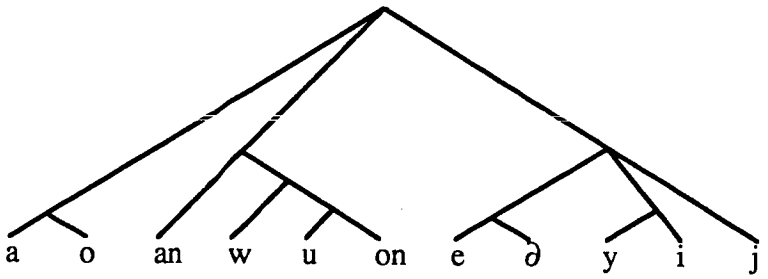
Conclusion.

Nous avons augmenté la vraisemblance d'un arbre de décision dans le but de la reconnaissance de la parole des voyelles et semi-voyelles. La méthode utilisée est l'algorithme de recuit simulé. La fonction à minimiser est un "potentiel" lié à la vraisemblance des fonctions de décision de chacun des noeuds de l'arbre.

Les inconvénients de cette méthode sont les suivants:

- D'une part la règle théorique qui assurerait la convergence de l'algorithme utilisé vers l'arbre de potentiel le plus faible, n'est pas praticable. La règle heuristique appliquée n'est qu'une règle pour laquelle on espère obtenir cette convergence.
- D'autre part, les calculs effectués pour l'évaluation des potentiels de chaque arbre sont très longs.

Un des avantages de ce procédé réside dans le fait que plusieurs arbres de potentiel plus faibles que celui de l'arbre initial ont été obtenus, et même des arbres binaires. De plus, le meilleur arbre observé,



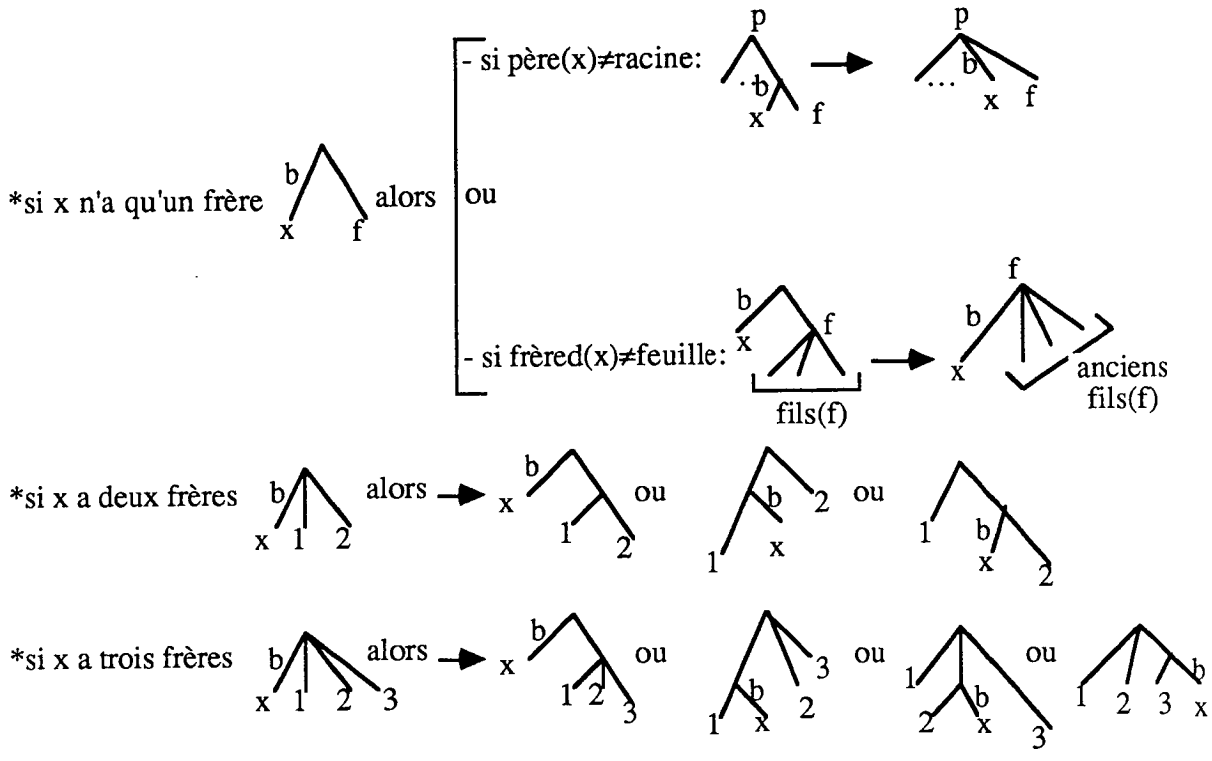
correspond à une classification bien connue des phonéticiens: le triangle acoustique des voyelles françaises (les trois branches principales concordent avec les trois sommets du triangle). Comme nous avons utilisé une paramétrisation différente de celles qui interviennent dans la construction du triangle (les formants), nous espérons avec une analyse des fonctions de décision trouvées, améliorer nos connaissances acoustiques du signal.

Il reste à étendre cette approche à la reconnaissance des phonèmes du français. L'algorithme de recuit est couteux, mais il est utilisé en phase d'apprentissage, et seuls les résultats de reconnaissance pourront confirmer son utilité ou non.

ANNEXES.

A.1. Candidat voisin.

Un candidat voisin est un arbre tiré au hasard dans le voisinage de l'arbre courant, X . On simule ce tirage au sort en déplaçant une branche tirée au hasard parmi celles de X, vers une des positions (aussi tirée au hasard) les plus proches possibles, mais distinctes de sa position précédente. Plus précisément, un noeud x est choisi selon la loi uniforme parmi tous les noeuds de X, différents de la racine. Il désigne ainsi la branche, b, qui l'atteint. La nouvelle position de b est tirée parmi les différentes possibilités décrites ci-dessous:



On se borne aux arbres au plus quaternaires pour des questions de temps de calcul. Si, lorsque x n'a qu'un frère et l'un des noeuds obtenus a plus de quatre fils, on recommence le tirage au sort d'un arbre.

A.2. Calcul du potentiel d'un noeud: Type "dictionnaire".

L'application de l'algorithme de recuit simulé au cas particulier considéré implique souvent, et même de plus en plus souvent à mesure que l'algorithme tourne et que la température décroit, les mêmes calculs de c, participation au critère d'un noeud. Ces calculs sont très longs; il faut éviter de les refaire.

De plus, toutes ces valeurs sont trop nombreuses (de l'ordre de 5^{12}) pour être toutes calculées une fois pour toutes (c'est justement pour éviter un calcul exhaustif qu'on utilise l'algorithme de recuit simulé), ou même pour réserver une place entière en mémoire pour chacune d'elle. D'où l'utilisation d'une table, appelée dictionnaire.

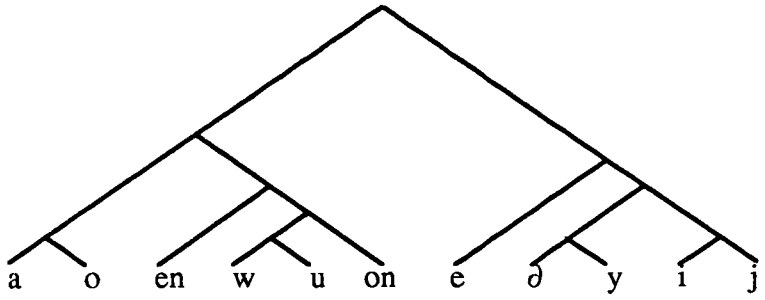
Une table est un tableau de dimension supportable, dont chaque composant est la tête d'une liste. C'est un compromis d'accès direct et d'accès par parcours de liste, et il faut, afin de prendre les avantages de l'un et de l'autre, choisir ce tableau de façon à ce qu'il soit à peu près régulièrement rempli par des listes de longueurs similaires.

Le dictionnaire choisi ici est une matrice 5×5×5×5 indéxée par des éléments de [0,1,..s] (où s est le nombre maximum autorisé de branches issues d'un noeud), correspondant respectivement aux phonèmes a,i,e et u. Les composants de cette matrice sont les listes de toutes valeurs des participations au critère, couplées des noms des phonèmes séparés à chaque noeud correspondant. Ce choix est fait parce que

- d'une part, ce tableau contient 625 composants, ce qui est supportable,
- d'autre part, on pense que l'arbre initial n'est pas trop éloigné du meilleur arbre possible et que les données des classes phonétiques des a,i,e,et u, sont séparées le plus tôt possible dans cet arbre (triangle phonétique des voyelles). On espère ainsi que la table sera à peu près régulièrement remplie.

A.3. Résultats de la reconnaissance

A.3.1. Arbre initial



évaluation à partir de l'ensemble d'entrainement:

matrice de confusion:

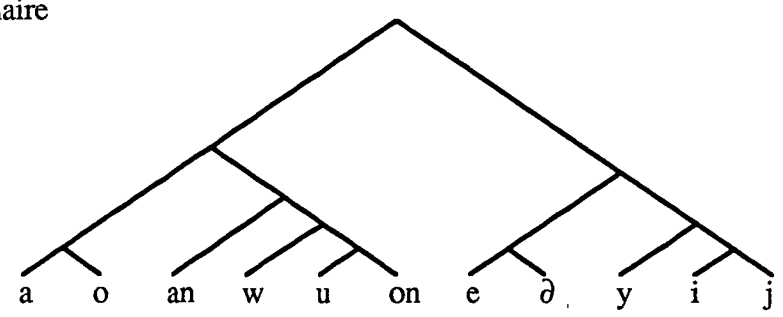
	a	o	en	w	u	on	e	ə	y	i	j	
a	137	1	0	0	0	0	12	0	0	0	0	150
o	1	35	1	2	1	1	0	6	0	0	0	47
en	2	0	56	1	0	1	0	0	0	0	0	60
w	2	5	4	11	2	1	0	1	0	0	0	26
u	0	0	0	2	42	1	0	1	4	0	0	50
on	0	1	1	3	1	34	0	0	0	0	0	40
e	4	0	0	0	0	0	145	0	4	4	0	157
ə	1	8	0	1	0	0	1	41	2	0	0	54
y	0	2	0	0	0	0	4	0	17	6	0	31
i	0	0	0	0	0	0	3	1	4	52	0	61
j	0	0	0	0	0	1	2	2	1	4	4	14
taux de reconnaissance par classe phonétique:	91	74	93	42	84	85	92	76	55	85	29	83

évaluation à partir de l'ensemble test:

matrice de confusion:

	a	o	en	w	u	on	e	ə	y	i	j	
a	115	6	3	0	0	0	5	0	0	0	0	129
o	5	18	3	8	8	5	0	1	0	0	0	48
en	2	0	32	8	0	0	0	0	0	0	0	42
w	1	1	0	6	4	3	0	0	0	0	0	15
u	0	0	0	6	23	2	0	0	5	0	0	36
on	0	0	3	12	7	20	0	0	0	0	0	42
e	5	0	0	0	0	0	103	5	1	1	0	115
ə	5	11	0	5	0	1	2	41	3	0	0	68
y	2	0	0	0	0	0	0	1	35	0	0	38
i	0	0	0	0	0	0	8	1	1	38	3	51
j	0	0	0	0	0	0	5	0	0	11	5	21
taux de reconnaissance par classe phonétique :	89	37	76	40	64	48	90	60	92	75	24	70

A.3.2. Meilleur arbre binaire



évaluation à partir de l'ensemble d'entrainement:

matrice de confusion:

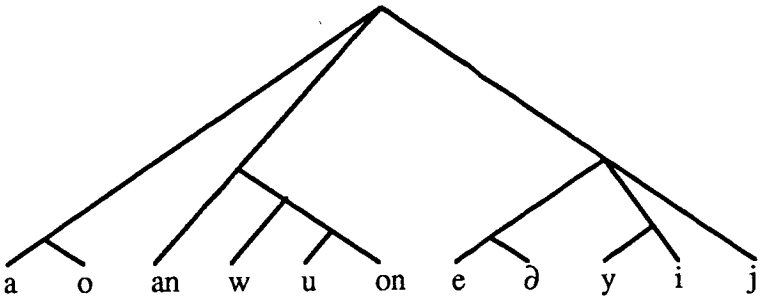
	a	o	en	w	u	on	e	∂	y	i	j	
a	137	1	0	0	0	0	10	2	0	0	0	150
o	1	35	1	2	2	0	0	6	0	0	0	47
en	2	0	56	1	0	1	0	0	0	0	0	60
w	2	5	4	10	3	1	0	0	1	0	0	26
u	0	0	0	2	45	0	0	0	3	0	0	50
on	0	1	1	1	0	37	0	0	0	0	0	40
e	4	0	0	0	0	0	148	0	0	5	0	157
∂	1	7	0	1	0	0	0	43	2	0	0	54
y	0	2	0	0	0	0	2	4	22	1	0	31
i	0	0	0	0	0	0	3	1	0	56	1	61
j	0	0	0	0	0	1	3	1	1	4	4	14
taux de reconnaissance par classe phonétique:	91	74	93	38	92	92	94	80	71	92	29	86

évaluation à partir de l'ensemble test:

matrice de confusion:

	a	o	en	w	u	on	e	∂	y	i	j	
a	115	6	3	0	0	0	5	0	0	0	0	129
o	5	19	3	9	7	4	0	1	0	0	0	48
en	2	0	32	7	0	1	0	0	0	0	0	42
w	1	1	0	8	3	2	0	0	0	0	0	15
u	0	0	0	4	23	3	0	0	6	0	0	36
on	0	0	3	16	7	16	0	0	0	0	0	42
e	5	1	0	0	0	0	106	2	1	0	0	115
∂	5	12	0	4	0	0	3	42	2	0	0	68
y	2	0	0	0	0	0	0	7	22	7	0	38
i	0	0	0	0	0	0	10	0	2	36	3	51
j	0	0	0	0	0	0	7	0	2	9	3	21
taux de reconnaissance par classe phonétique:	89	40	76	53	64	38	92	62	58	71	14	72

A.3.3. Meilleur arbre



évaluation à partir de l'ensemble d'entrainement:

matrice de confusion:

	a	o	en	w	u	on	e	ə	y	i	j	
a	139	2	2	1	0	0	6	0	0	0	0	150
o	1	33	1	2	2	1	0	7	0	0	0	47
en	1	0	57	1	0	1	0	0	0	0	0	60
w	2	5	4	10	3	1	0	0	1	0	0	26
u	0	0	0	2	46	0	0	0	2	0	0	50
on	0	0	1	2	0	37	0	0	0	0	0	40
e	6	0	0	0	0	0	145	0	0	6	0	157
ə	1	8	0	0	0	0	0	43	2	0	0	54
y	0	2	0	0	0	0	1	3	24	0	1	31
i	0	0	0	0	0	0	3	1	1	56	0	61
j	0	0	0	0	0	1	3	1	1	5	3	14
taux de reconnaissance par classe phonétique:	93	70	95	38	92	92	92	80	77	92	21	86

évaluation de l'ensemble test:

matrice de confusion:

	a	o	en	w	u	on	e	ə	y	i	j	
a	115	6	3	0	0	0	5	0	0	0	0	129
o	5	18	3	8	8	5	0	1	0	0	0	48
en	2	0	32	8	0	0	0	0	0	0	0	42
w	1	1	0	6	4	3	0	0	0	0	0	15
u	0	0	0	6	24	1	0	0	5	0	0	36
on	0	0	3	12	7	20	0	0	0	0	0	42
e	2	0	0	0	0	0	106	5	1	1	0	115
ə	4	7	0	2	0	0	2	45	8	0	0	68
y	2	0	0	0	2	0	0	1	33	0	0	38
i	0	0	0	0	0	0	8	1	0	42	0	51
j	0	0	0	0	0	1	5	0	0	13	2	21
taux de reconnaissance par classe phonétique:	89	37	76	40	67	46	92	66	87	82	10	73

A.3.4. Résultats obtenus par une méthode de quantification vectorielle [11]

évaluation à partir de l'ensemble d'entrainement:

matrice de confusion:

	a	o	en	w	u	on	e	∂	y	i	j	
a	140	1	0	0	0	0	3	2	0	0	0	146
o	0	44	0	1	1	0	0	1	0	0	0	47
en	2	0	55	2	0	0	0	1	0	0	0	60
w	1	0	2	21	0	1	0	0	0	0	0	25
u	0	1	0	3	45	0	0	0	0	0	0	49
on	0	0	0	1	0	36	0	3	0	0	0	40
e	8	0	0	0	0	0	144	1	1	1	3	158
∂	1	2	0	0	0	0	0	74	1	0	0	78
y	0	0	0	0	0	0	0	2	29	0	0	31
i	0	0	0	0	0	0	1	0	1	56	1	59
j	0	0	0	0	0	0	1	0	0	3	10	14
taux de reconnaissance par classe phonétique:	96	94	92	84	92	90	91	95	94	95	71	93

évaluation à partir de l'ensemble test:

matrice de confusion:

	a	o	en	w	u	on	e	∂	y	i	j	
a	111	3	3	4	0	0	3	1	0	0	0	125
o	1	20	0	12	3	5	0	7	0	0	0	48
en	1	0	30	6	0	4	0	1	0	0	0	42
w	1	1	0	9	3	1	0	0	0	0	0	15
u	0	1	0	3	29	2	0	0	0	0	0	35
on	0	2	8	3	0	28	0	1	0	0	0	42
e	2	0	0	1	0	0	105	3	1	0	1	113
∂	0	18	0	0	0	0	3	43	1	0	3	68
y	1	0	0	0	0	0	0	9	27	0	0	37
i	0	0	0	0	0	0	8	0	1	30	11	50
j	0	0	0	0	0	0	6	0	0	10	4	20
taux de reconnaissance par classe phonétique:	89	42	71	60	83	67	93	63	73	60	20	73

A.4. Figures

figure1: X_0

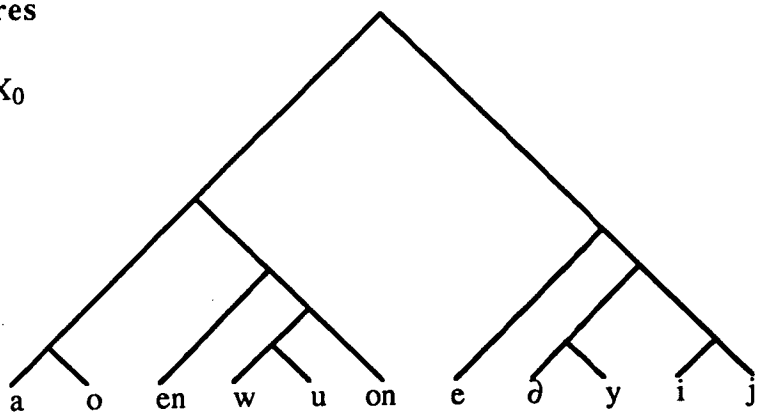


figure2: X_2

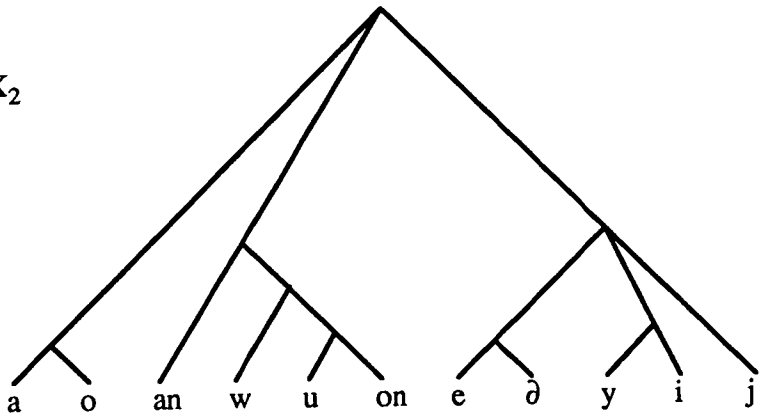


figure3: X_3

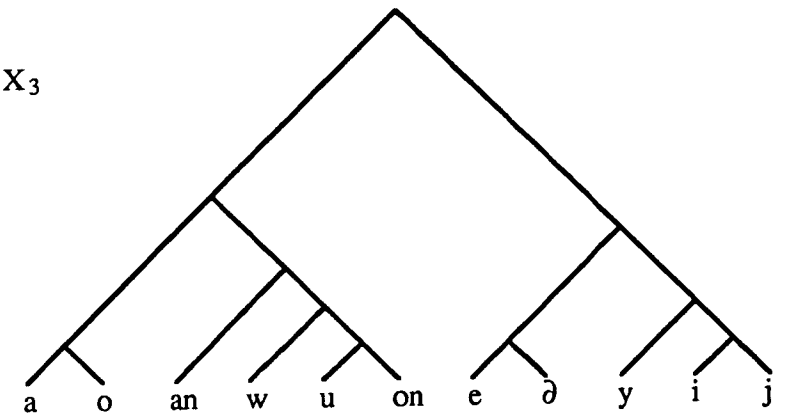


figure4: X_4

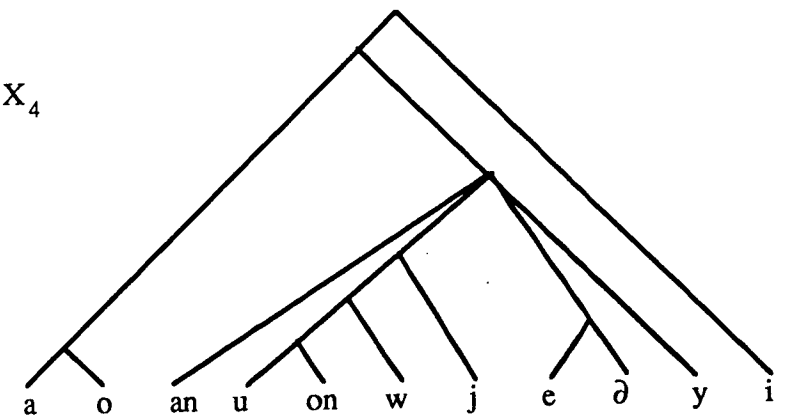


figure5: X_5

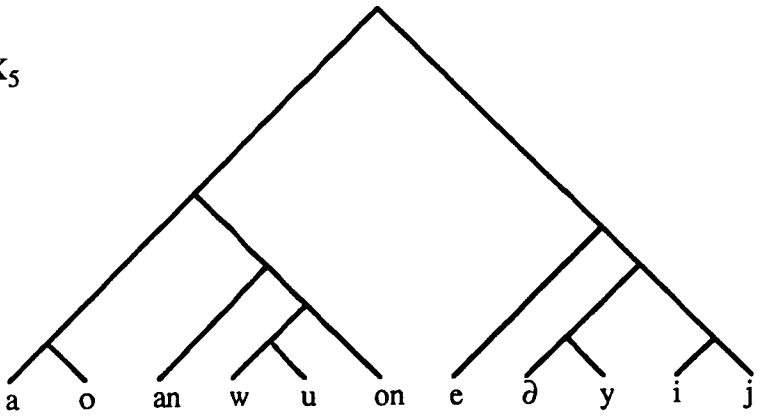


figure6: X_6

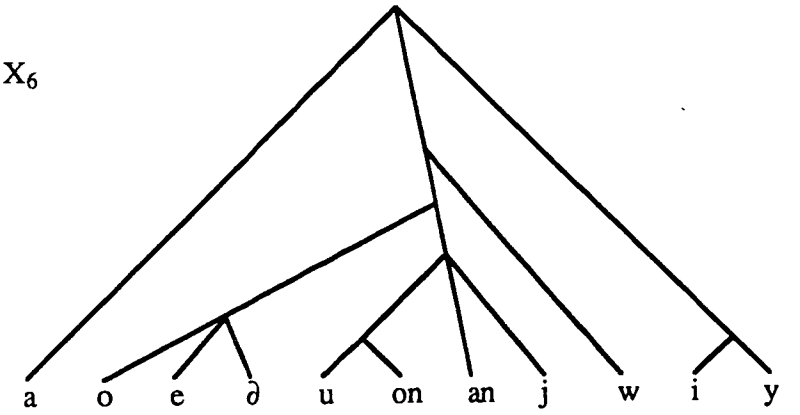


figure7: X_7

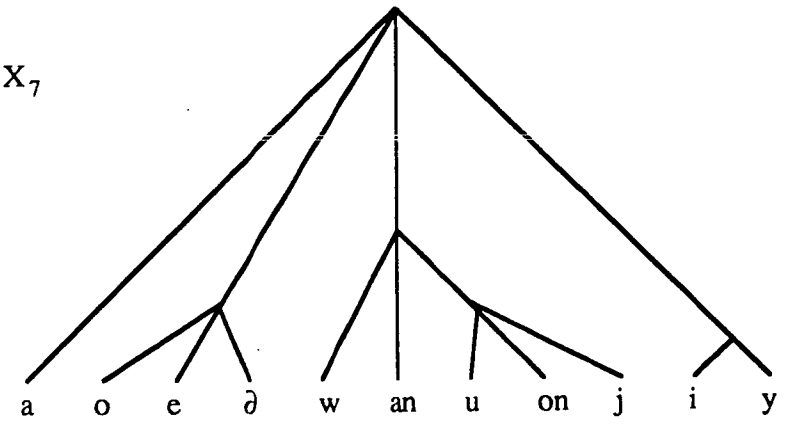


figure8: X_8

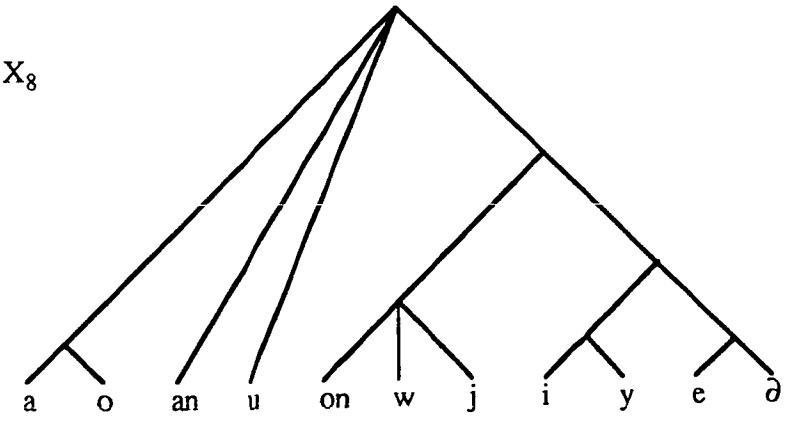
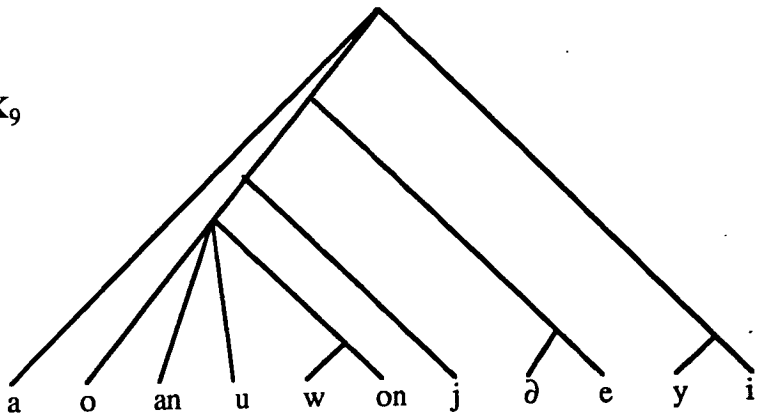


figure9: X_9



Remerciements.

Je tiens à remercier très sincèrement Régine ANDRE-OBRECHT, Annie MORIN, et Albert BENVENISTE, grâce aux quels j'ai pu réaliser cette étude.

Bibliographie.

[1] E.N.Adams, "N-trees as Nesting: Complexity, Simularity, and Consensus", Journal of classification, vol.3, pp.299-317.

[2] B.Delyon, "Utilisation de l'algorithme de recuit simulé pour des potentiels correspondants à des sources de Markov cachées", Rapport de convention CNET, 1987.

[3] S.Geman and D.Geman, "Stochastic Relaxation, Gibbs Distribution, and the Bayesian Restoration of Images", Transactions on Pattern Analysis and Machine Intelligence, vol.6, pp.721-741, November 1984.

[4] B.Hajek, "A tutorial survey of theory and applications of simulated annealing", Proceeding of 24th conference on Decision and Control, December 1985.

[5] D.J.Hand, "Discrimination and classification", John Wiley & sons, 1981.

[6] R.H.Jones, "Probability estimation Using a multinomial logistic function", J.Statis.Comput.Simul., vol.3, pp.315-329, 1975.

[7] R.Kindermann, J.Laurie Snell, "Markov Random Fields and their Applications", American mathematical society, vol.1.

[8] I.C.Lerman,"Evaluation et logiciel d'un indice de similarité entre objets d'un type quelconque. Application au problème du consensus en classification.", Publication interne IRISA-Rennes n° 262, Juillet 1985.

[9] M.Lundy, "Applications of annealing algorithm to some combinatorial problems in Statistics", Rapport interne, Statistical Laboratory, Cambridge University.

[10] A.Morin, "Reconnaissance des voyelles de la parole continue",Préparation d'une thèse d'état, Juin 1988.

[11] H.Y.Su, "Reconnaissance Acoustico-Phonétique en Parole Continue par Quantification Vectorielle; Adaptation du Dictionnaire au Locuteur", thèse d'Université de Rennes 1, novembre 1987.

Imprimé en France
par
l'Institut National de Recherche en Informatique et en Automatique