Reconnaissance de la parole par réseaux multi-couches.

Léon-Yves BOTTOU

Laboratoire d'Intelligence Artificielle de Paris 5, Ecole des Hautes Etudes en Informatique, Université de Paris 5, 45 Rue des Saints Pères, PARIS-FRANCE.

tel (1) 47 03 31 27.

Résumé.

Nous exposons ici diverses méthodes pour tenir compte des spécificités de la dimension temporelle du signal de parole à l'intérieur de réseaux multi-couches. Nous décrivons ensuite quelques simulations réalisées avec un petit ensemble de mots difficiles, fourni par le LIMSI (Orsay), utilisant l'une des structures exposées auparavant. Ces expérimentations concernent une reconnaissance de mots isolés multi-locuteurs. Les résultats sont du même ordre de grandeur que ceux obtenus par programmation dynamique sur les mêmes données.

Abstract.

We describe in this paper some methods for introducing the temporal dimension specific to speech data into a multi-layer perceptron structure. Some experiments have been carried out, with a small set of difficult words provided by LIMSI (Orsay - France). These simulations deal with isolated- speaker independent- word recognition. The network exhibit results similar to those achieved by dynamic time warping techniques on the same set.

Catégorie: applications

Mots clés: reconnaissance de la parole - multi-locuteurs - mots isolés
réseaux de neurones - TDNN - rétro-propagation

Reconnaissance de la parole par réseaux multi-couches.

Léon-Yves BOTTOU

Laboratoire d'Intelligence Artificielle de Paris 5, Ecole des Hautes Etudes en Informatique,
Université de Paris 5, 45 Rue des Saints Pères, PARIS

1 - Introduction

La reconnaissance de la parole est réputée être un problème difficile [13,9]. Dans les années 50, les pionniers, forts des théories nouvelles de traitement du signal, pensaient que ce problème se ramenait à quelques opérations de filtrage. Ils n'avaient pas alors pu constater que nous n'*entendons* pas une suite de phonèmes, mais que nous *reconstruisons* mentalement. Par exemple, le signal correspondant à une voyelle donnée dépend dans des proportions considérables de la nature des consonnes qui l'entourent. A leur tour, celles-ci dépendent d'un contexte un peu plus large encore. Ce phénomène est connu sous le nom d' *effet de coarticulation* . La variabilité du signal d'un locuteur à l'autre est tout aussi importante. Les caractéristiques dont nous connaissons l'importance, par exemple la notion de *formants* , peuvent d'un locuteur à l'autre, être quantitativement identiques, et représenter pourtant deux phonèmes différents. Notre cortex utilise probablement des informations de nature différente (nous voyons souvent la personne qui parle) pour accomplir cette tâche difficile.

L'approche la plus fiable actuellement, consiste à reconnaître des mots entiers. On évite ainsi d'aborder de front le redoutable problème de la coarticulation. Des techniques comme la programmation dynamique (DTW) [5,12] , ou les modèles de markov cachés (HMM) [4], présentent des performances appréciables, au prix toutefois de diverses contraintes. Ces contraintes s'expriment en quelques oppositions. Il faut faire un compromis entre :

- reconnaissance mono-locuteur ou multi-locuteurs,
- avec des mots isolés, ou bien enchaînés,
- sur des vocabulaires de taille petite ou grande,

- dans un contexte bruité ou non.

Or, il semble aujourd'hui que les réseaux connexionnistes présentent des propriétés intéressantes pour ce genre de problèmes [10]. Adaptativité et résistance au bruit font partie de leurs propriétés naturelles . En revanche, peu de modèles sont explicitement prévus pour traiter des données temporelles (cf §2). Si une approche connexionniste fait ses preuves sur ce dernier point, on peut espérer construire ainsi des systèmes plus performants de reconnaissance automatique de la parole[1,6,14,18] .

Dans la section 2, nous décrivons et commentons quelques méthodes pour traiter des données temporelles dans des réseaux multi-couches utilisant l'algorithme de rétro-propagation du gradient [3,7,15]. La section 3 est consacrée à la description de deux simulations, sur une base de données de petite taille. Des comparaisons avec les méthodes classiques sont ébauchées dans la section 4.

2 - Réseau multi-couches et problèmes temporels.

Usuellement, les entrées d'un réseau multi-couches n'ont aucune structure. Chaque cellule est connectée sans tenir compte de ses attributs, ou de sa signification réelle. On confie à l'algorithme d'apprentissage la tâche de découvrir quelle structure ont ces entrées, et comment l'exploiter.

Dans le cas précis de la reconnaissance automatique de la parole, toutes les données en entrée sont indicées par le temps. Ne pas en tenir compte, c'est imposer à l'algorithme de rétro-propagation une charge extrêmement lourde, longue, et d'autant plus inutile que l'on connaît fort bien les propriétés de ces entrées : invariance par translation et par distorsions, tout au moins dans certaines limites, importance du contexte. Toutes les méthodes classiques sont fondées sur la connaissance de ces deux propriétés. La programmation dynamique, comme les chaînes de markov cachées, sont essentiellement des méthodes d'utilisation du contexte pour réduire l'impact des distorsions temporelles.

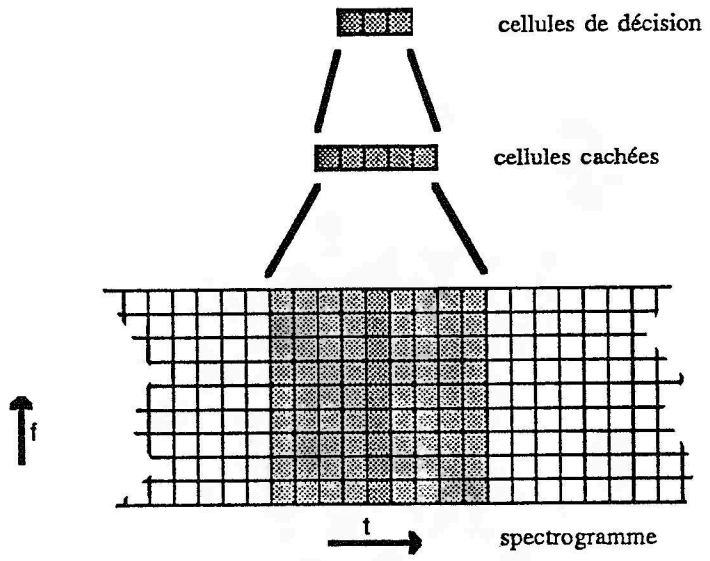
Sans information a priori, l'algorithme de rétro-propagation, appliqué à de telles données, est très lent. En outre, les performances du réseau entraîné se révèlent parfois décevantes, et

toujours difficiles à obtenir [2,8,11]. Afin de retrouver la rapidité et la fiabilité habituelles de cet algorithme, il est nécessaire d'introduire des informations dans la structure même du réseau.

2.1 - Introduire des informations sur le contexte en entrée du réseau.

La première idée consiste à introduire la notion de contexte. Il suffit pour cela de connecter les cellules de la première couche cachée, non seulement aux entrées à l'instant t, mais aussi aux données dans un intervalle de temps entourant ces entrées.

(fig 1)
Prise en compte du contexte



Cette structure ne contient en revanche aucune information sur la nature des distorsions auxquelles devrait résister le réseau. Cette information devra donc être introduite dans les données d'apprentissage. Par ailleurs, si l'on veut avoir une fenêtre de contexte assez large, (suffisante pour embrasser un mot entier, par exemple) , on perd toutes les propriétés de localité dans le temps des connexions. Ces deux inconvénients combinés réduisent beaucoup l'intérêt d'un tel modèle.

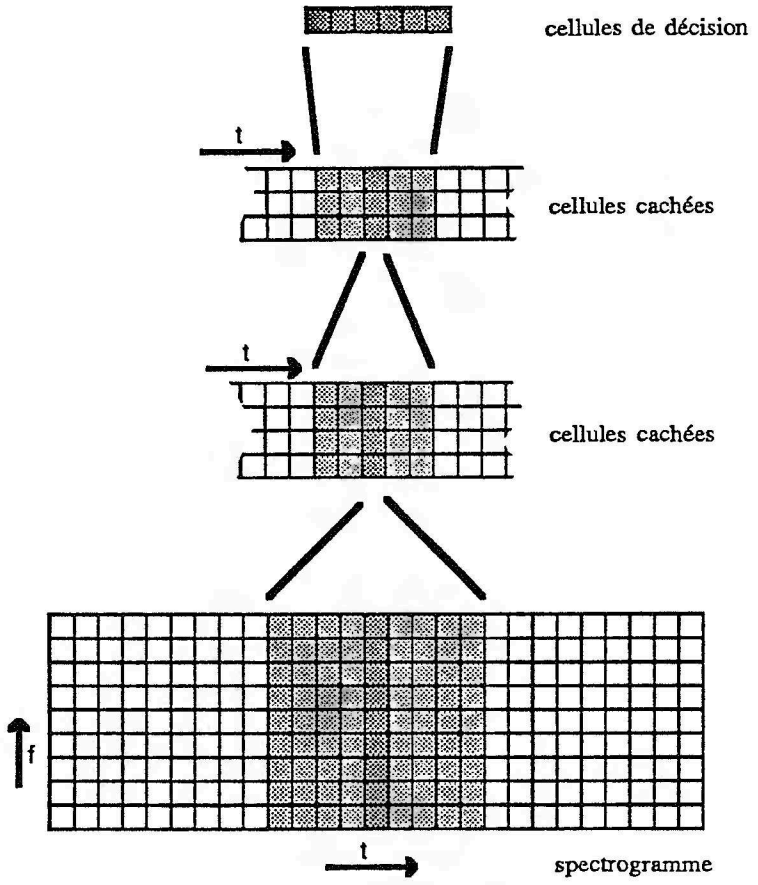
2.2 - T.D.N.N: Tenir compte du contexte dans les couches internes.

L'idée de prendre un contexte dans les informations d'entrée en suggère une autre, plus évoluée. Pourquoi ne pas prendre en compte le contexte à l'intérieur des couches cachées, sur les informations de plus haut niveau générées par le réseau ?

Ce type de réseau, pour la reconnaissance de la parole, a été initialement introduit dans [26,27], sous le nom de TDNN (Time Delay Neural Network) sous l'impulsion de G. Hinton .

(fig 2)

Prise en compte du contexte à plusieurs niveaux dans le réseau



Un léger aménagement de l'algorithme de rétro-propagation est nécessaire pour entraîner un tel réseau. Il suffit en fait d'effectuer un dépliage temporel [25]. L'algorithme de rétro-propagation s'applique directement sur le réseau déplié. Cependant, certaines connexions

du réseau déplié correspondent à la même connexion du réseau temporel. Elles doivent donc conserver un poids identique. Il suffit pour cela de leur donner la même valeur initiale, et de les ajuster avec la somme des gradients de toutes les connexions partageant le même poids. On peut implémenter cet algorithme de façon très efficace, en ne plaçant dans les connexions qu'un pointeur vers une valeur du poids unique pour toutes ces connexions.

Ces contextes hiérarchisés ont en fait une portée qui peut être grande, tout en conservant des propriétés locales. Plus deux instants sont proches, plus ils influent l'un sur l'autre dans des couches de faible niveau. L'apprentissage des distorsions temporelles en sera donc considérablement facilité.

Ce type de réseau possède également un avantage déterminant. L'ensemble des états internes possède la même structure temporelle que les données en entrée. Il est donc beaucoup plus facile d'appréhender le fonctionnement du réseau, et de comprendre quelles modifications sont nécessaire pour en améliorer les performances. Les travaux actuels exploitant cette structure sont fréquemment accompagnés d'interprétations des états internes générés par le réseau.

2.3 - Apport de la théorie de l'information.

Un signal de parole enregistré sur un canal haute fidelité représente plus de 300000 bits par seconde. Le même à travers un téléphone ne possède plus qu'une quantité d'information de 20000 bits par seconde, et l'on sait que l'on peut compresser ce signal à environ 7000 bits par seconde [9]. Les spectrogrammes que nous utilisons contiennent moins de 10000 bits par seconde.

On peut également définir une *information phonétique*, associée à la suite de symboles représentant les phonèmes prononcés dans le signal de parole. Cela représente environ 50 bits par seconde [9].

Dans un T.D.N.N, on espère fréquemment obtenir des états *quasi-phonétiques* dans la dernière couche cachée. Soit 50 bits/seconde d'information. Le nombre de variables d'état dans cette couche est le produit du nombre de cellules par le nombre de tranches temporelles considérées. Cela représente environ 1000 états par seconde. Il serait souhaitable que cette valeur soit de l'ordre de grandeur de la quantité d'information phonétique dans le signal, vingt fois plus faible.

Les états des cellules sont calculés avec la même fréquence, quel que soit leur niveau de représentation des données. Or on s'attendrait à ce que les descripteurs de haut niveau soient plus longtemps significatifs. Il n'est donc pas nécessaire de les calculer aussi souvent. (fig 3). On peut également réduire le nombre de cellules dans cette couche. La première solution apporte une erreur de discrétisation, alors que la seconde réduit le nombre de connections et donc la faculté du réseau d'apprendre des notions complexes.

Notre préférence va à la réduction de la fréquence de calcul des états internes. Outre le gain en temps de calcul, imposer aux cellules cachées des intervalles de constance plus longs leur permet de résister à de petites distorsions temporelles. Ces distorsions ne peuvent s'étendre bien sûr au delà des effets de bord introduits par la discrétisation de plus en plus grossière dans le temps des cellules cachées. Cela donne cependant au réseau de bonnes caractéristiques temporelles.

3 - Simulations.

Nous relatons dans cette section une expérience, dont le but est d'évaluer grossièrement les performances des systèmes connexionnistes en reconnaissance automatique de la parole. Il s'agit principalement de tester leur capacité à construire une méthode de reconnaissance indépendante du locuteur.

Nous avons choisi, dans un premier temps, de travailler sur des mots entiers, isolés. Cela nous permet d'utiliser des données provenant d'expériences sur des systèmes classiques, et d'établir éventuellement des comparaisons.

3.1 - Données fournies par le LIMSI.

Dans le cadre d'un projet CEE BRAIN, le LIMSI (Laboratoire d'Informatique pour la Mécanique et les Sciences de l'Ingenieur, Orsay), qui a une longue et solide expérience du traitement automatique de la parole, nous a procuré de nombreuses données. Pour une première simulation, six locuteurs, hommes, ont prononcé une fois chacun 5 mots, dont les 4 premiers forment deux paires minimales. Ces mots sont :

CINQ, SEPT, CABOT, CAPOT, CABOTIN

L'acquisition et les prétraitements ont été effectués au LIMSI, avec des techniques classiques. Le signal de parole subit tout d'abord un filtrage anti-repliement, puis est échantillonné à 10kHz. On effectue ensuite, toutes les 12.8 ms, une FFT sur 256 points. Les 256 lignes de ce spectrogramme sont moyennées en 16 canaux, répartis selon l'échelle de Bark (il s'agit d'une échelle linéaire dans les basses fréquences, et logarithmique dans les hautes fréquences). On code ensuite leur logarithme sur 8 bits.

En fin de traitement, les données sont représentées par un tableau de nombres codés sur huit bits, dont les lignes représentent 16 canaux de fréquences réparties entre 100 et 5000Hz, et les colonnes, des intervalles de 12.8 ms. En outre, les débuts et fins de chaque mot sont repérés avec une précision assez faible par l'appareillage d'acquisition.

Nous avons choisi d'utiliser 4 locuteurs pour l'apprentissage, et 2 pour effectuer les tests en généralisation. Cela ne permet d'avoir que 20 mots pour l'apprentissage et 10 pour le test. Dans ces conditions, la précision des mesures de performance est relativement faible. Afin de donner plus d'indications, nous indiquons les bornes que nous avons relevées au cours de nos simulations, ainsi qu'une valeur typique de ces mesures.

Cet ensemble de données reste cependant extrêmement réduit, pour appliquer sereinement l'algorithme de rétro-propagation. Le réseau devra donc contenir une quantité maximale d'information a priori dans sa structure.

3.2 - Structure du réseau.

Nous avons implémenté un TDNN. Il contient :

-une couche d'entrée de 16 cellules (une par canal de fréquence). Comme la longueur des mots est normalisée à 65 (longueur du mot le plus long), les entrées forment donc une matrice de 16 lignes et 65 colonnes.

-une couche cachée de 3 cellules, connectées à un contexte de 3 unités de temps de la couche d'entrée. Leurs états sont calculés toutes les 2 unités de temps. On obtient donc une matrice de 3 lignes et 32 colonnes.

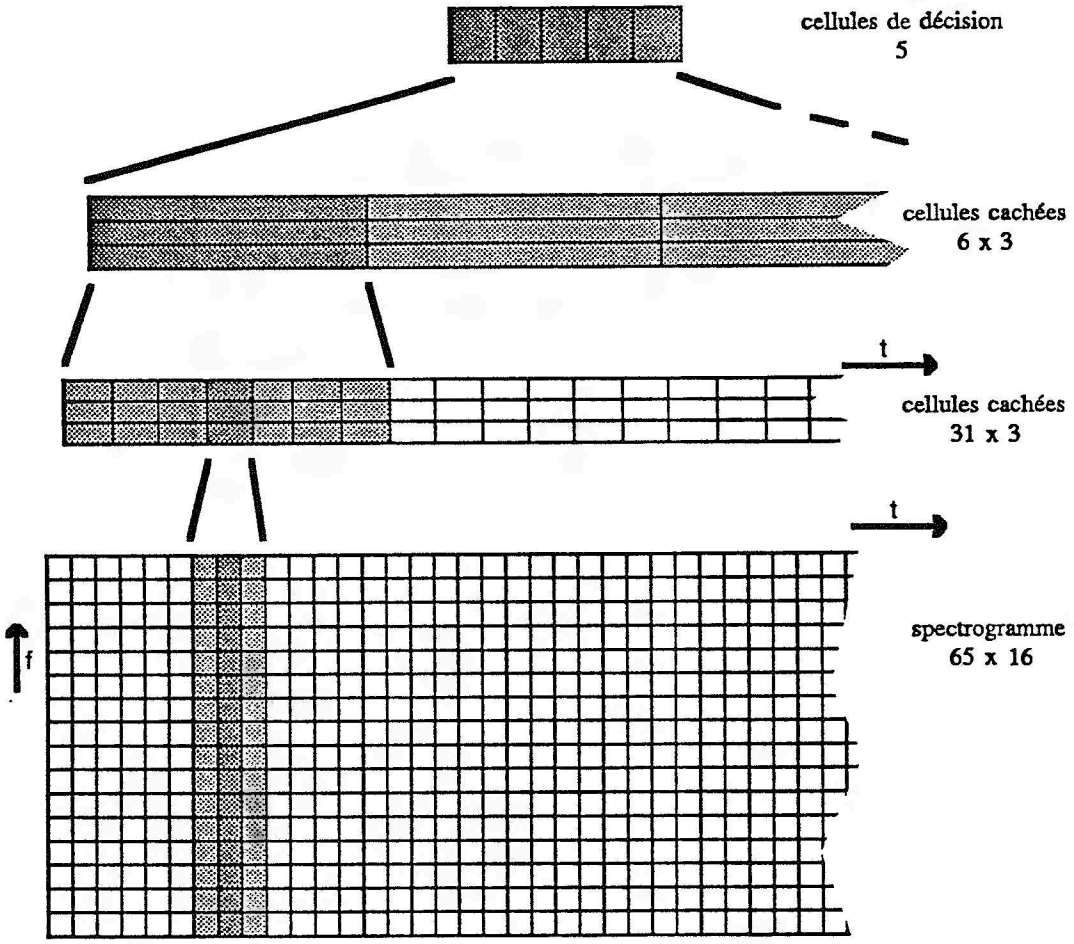
- une seconde couche cachée de 3 cellules, connectées à un contexte de 7 valeurs des états de la seconde couche, et calculées 5 fois moins souvent. Cela représente 3 lignes et 6 colonnes.

- une couche de sortie de 5 cellules.

(fig 3)

Réseau pour le problème

cinq, sept, capot, cabot, cabotin



Si l'on exprime cette structure en termes de réseaux itératifs, la couche d'entrée contient 1040 cellules, la première couche cachée, 96, la seconde, 18, et la couche de sortie 5 seulement. Elles sont connectées par des masques 16x3 puis 7x3, et totalement dans la dernière couche de poids. Cela totalise 4788 connections, mais seulement 383 paramètres, en comptant les seuils.

Avant de présenter les données au réseau, elles sont ramenées dans l'intervalle [-1,1]. Pour éviter que la reconnaissance ne porte que sur la longueur des mots, on normalise leur longueur à 65, longeur du mot le plus long. Cette opération nous permet en outre de ne pas nous soucier, pendant l'apprentissage, des périodes de silence.

Il est important de souligner que les mots ne sont pas parfaitement alignés. La normalisation de la longueur des mots consiste en un étirement réalisé entre les marques de début et fin de mot déterminées par l'appareil d'acquisition. A titre comparatif, un alignement manuel précis des mots dans le système classique sur cette base de données ramène le taux d'erreur de 20% à 5%. Le problème de l'alignement est sans doute l'un des plus crucial dans tous les systèmes de reconnaissance de mots.

Pour apliquer l'algorithme de rétro-propagation du gradient, nous avons utilisé la fonction de transfert :

$$f(x) = 1.63 \text{ th } (0.67 x) + 0.05 x$$

Cette fonction est une sigmoïde "penchée", les coefficients ont été choisis de telle sorte que f(1)=1, pour simplifier l'interprétation des variables du réseau. Les poids ont été initialisés avec des valeurs aléatoires uniformes, de sorte que l'écart type des sommes pondérées corresponde aux points de courbure maximum de la sigmoïde. i.e. dans

$$\left[\frac{2.38}{\sqrt{n}}, \frac{2.38}{\sqrt{n}} \right]$$

où **n** est le nombre de connections vers la cellule en aval.

Les poids sont simplement ajustés selon la formule

$$W_{ij}^{t+1} = W_{ij}^{t} + \varepsilon \sum_{(k,l) \in H(i,j)} X_k Y_l$$

avec X_i : état de la cellule amont

Y_j: gradient sur la cellule aval.

H(i,j) : ensemble des connexions partagées avec (i,j)

La vitesse d'apprentissage ε vaut 0.01 dans la première couche cachée, 0.02 dans la seconde et 0.03 en sortie.

3.3 - Apprentissage sur les spectogrammes non déformés.

Dans cette première expérience, nous avons utilisé les 20 mots prononcés par les 4 premiers locuteurs. Ces mots sont seulement étirés régulièrement à la même longueur.

On obtient 100% sur l'ensemble d'apprentissage en 400 itérations au plus. En revanche, sur l'ensemble de test, les résultats sont extrêmement dépendants de l'initialisation des poids. Cette dépendance était très prévisible, car l'ensemble d'apprentissage était trop petit. On constate usuellement de 2 à 4 (typiquement 3) erreurs sur 10 mots. Ce résultat, quoique prévisible, est manifestement mauvais. (cf fig 5 à 7)

Certes les TDNN contiennent a priori des informations sur la nature des distorsions temporelles, mais cette information a priori n'est pas complète. Ces informations sont altérées par des effets de discrétisation de plus en plus importants dans les couches supérieures. Or nous devons apporter l'information manquante par notre ensemble d'exemples. Il est ici bien trop réduit.

3.4 - Utilisation de fortes déformations des spectrogrammes.

Nous savons donc quelle information manque à notre réseau. Il semble donc intéressant de déformer algorithmiquement nos exemples, afin que leur nouvel ensemble reflète les propriétés que nous voudrions apporter à notre réseau.

A partir des 20 mots de la base d'exemples, nous avons générés 400 motifs extrêmement distordus dans le temps. Ces distorsions étirent le mot jusqu'à une durée totale de 65 échantillons. Elles sont assez fortes pour ramener 80% d'un mot dans une moitié seulement du spectrogramme, ou au contraire ramener la moitié d'un mot dans 80% du spectrogramme. Ces déformations sont appliquées sans tenir compte de la structure phonétique du mot. Si elles n'altèrent pas trop les voyelles, elles peuvent rendre, dans les cas extrêmes, les consonnes inaudibles.

L'algorithme de déformation, pour un mot de longueur n , consiste à calculer récursivement la suite de fonctions f_i, telles que

$\forall i = 6,5,4,...,2$

f_i linéaire par morceaux,

$f_i(0) = 0,\ f_i(64) = n,$

$\forall\ k=1,..,2^{6-i} -1,\quad f_i(k2^i) = f_{i+1}(k2^i)$

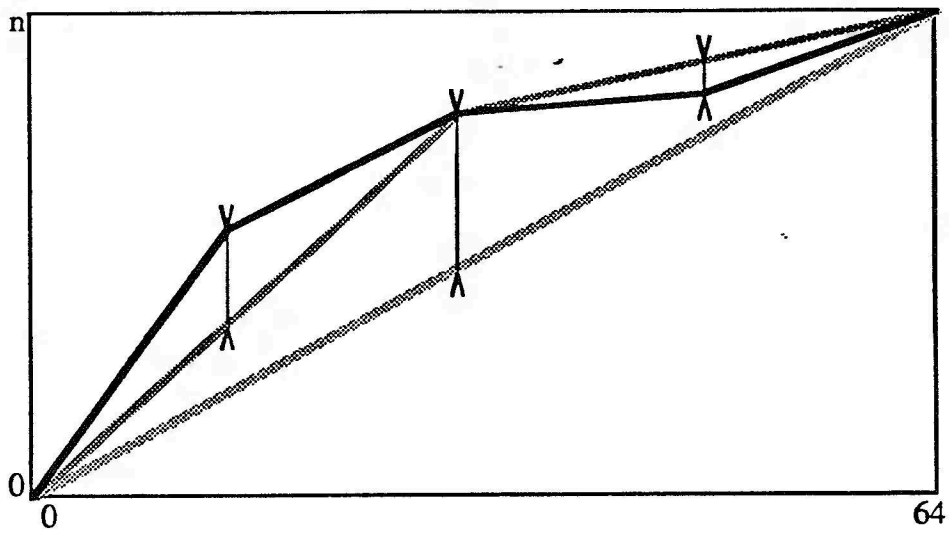
$\forall\ k=0,..,2^{6-i} -1,\quad f_i(k2^i+2^{i-1}) = (f_i(k2^i)+f_i((k+1)2^i))/2 + \omega_{ik}$

avec

ω_{ik} = nombre aléatoire uniforme dans [-0.8 L_{ik}, 0.8 L_{ik}] ,
avec $L_{ik} = f_i((k+1)2^i) - f_i(k2^i)$.

(fig 4)

génération de la fonction de déformation



Les diverses courbes représentent des exemples de fonctions f_6, f_5, f_4

On déduit le spectre déformé S' par

$$S'(F, T) = S(F, f_2(T))$$
pour F (canal) $\in$ [0,15], T (temps) $\in$ [0,64]

On effectue donc l'apprentissage avec les données ainsi déformées (cf fig 8 à 12). Les tests de reconnaissance concernent 3 jeux de mots:

- (1) 50 des 400 mots déformés pris au hasard,
- (2) les 20 mots non déformés des locuteurs de l'ensemble d'apprentissage,
- (T) les 10 mots non déformés des locuteurs de l'ensemble de test.

500 à 2000 itérations, soit 8 à 30 minutes d'apprentissage sur une station Apollo, sont nécessaires pour que les mots de (2) soient tous correctement reconnus. Il faut souligner que l'apprentissage ne portait pas sur ces mots, mais sur des déformations de ces mots. En

revanche, la performance sur l'ensemble d'apprentissage (1), ne dépasse jamais 94%. Les 400 mots que nous avons très fortement déformés forment probablement un ensemble partiellement incohérent.

Le réseau effectue de 0 à 3 (typiquement 1) erreurs sur l'ensemble (T) de 10 mots, qui représente la performance en reconnaissance multi-locuteurs. On peut parfois réduire ces erreurs en effectuant quelques passes d'apprentissage supplémentaires avec les mots de l'ensemble (2). Cette technique est cependant difficile à appliquer, et peu fiable, elle ne semble donc pas intéressante dans ce problème. En moyenne, le taux de reconnaissance est de 90%.

Les meilleurs réseaux (i.e. pour certaines configurations initiales des poids) reconnaissaient à 100% les mots des ensembles (2) et (T).

4 - Critiques et comparaisons.

4.1 - Comparaison avec les techniques classiques

Sur la même base de données, la technique de programmation dynamique mise au point au LIMSI [5,12], donne 21% d'erreurs, dans les mêmes conditions de découpage des mots, et avec une référence par mot. Le réseau donne usuellement 10% d'erreurs, parfois même aucune erreur, mais utilise quatre locuteurs pour l'apprentissage. En revanche, la quantité de mémoire utilisée pour stocker les références est à peu près égale à celle utilisée pour stocker les poids.

Contrairement à la programmation dynamique, le système connexionniste requiert une phase d'apprentissage assez coûteuse. Cependant, il est beaucoup plus rapide pendant les opérations de reconnaissance.

On le voit, il n'est pas facile de trouver un terrain de comparaison objectif. Il est remarquable que les réseaux accomplissent une performance très comparable. En outre, ces résultats ont été obtenus avec un très faible ensemble d'apprentissage, conditions très éloignées de l'utilisation optimale de la rétro-propagation de gradient.

4.2 - Séparation ou Reconnaissance ?

Ces tests mesurent tous les capacités de séparation du réseau. Pour reconnaître les mots, il faut aussi répondre à la question 'Ce signal est-il un des mots à reconnaître'. Afin de nous rendre compte du comportement du réseau de ce point de vue, nous avons présenté du bruit blanc en entrée d'un réseau entraîné. Les sorties sont alors toujours négatives, ou très faiblement positives (cf fig 12). Aucune n'est nettement active et supérieure à toutes les autres. Bien qu'il ne constitue pas une garantie, ce test apporte une indication favorable.

5 - Conclusion.

Ces petites expériences doivent bien sûr être confirmées sur une base de données plus importante. Elle est néanmoins très satisfaisante sur plusieurs points. Les problèmes que nous avons rencontrés sont en grande partie dûs à la taille trop petite de notre ensemble d'apprentissage. En outre, la nette amélioration que nous avons constatée en introduisant des données déformées confirme le rôle prépondérant joué par leurs propriétés temporelles.

Nous travaillons actuellement, toujours en étroite collaboration avec le LIMSI, sur un ensemble de 135 mots et 25 locuteurs.

6 - Remerciements.

Nous tenons à remercier le LIMSI (Laboratoire d'Informatique pour la Mécanique et les Sciences de l'Ingenieur , à Orsay), et en particulier Jean Sylvain LIENARD (directeur du LIMSI) et Pascal BLANCHET , pour leurs données, leur soutien permanent, et leurs conseils avisés. Ce travail eût été absolument impossible sans leur concours.

7 - Bibliographie

[1] J.S. BRIDLE, R.K. MOORE: Boltzmann Machines for speech pattern processing. Proc. Inst. Acoust. Autumn Meeting, Nov. 1984.

[2] J.L. ELMAN, D. ZIPSER: learning the hidden structure of speech. Tech. Report, Univ. of California, San Diego, feb. 1987.

[3] F. FOGELMAN SOULIE, P. GALLINARI, Y. LE CUN, S. THIRIA: Automata networks and Artificial Intelligence. In " Automata networks in computer Science", F. Fogelman Soulié, Y. Robert, M. Tchuente Eds, Manchester Univ. Press, (1987), 133-186.

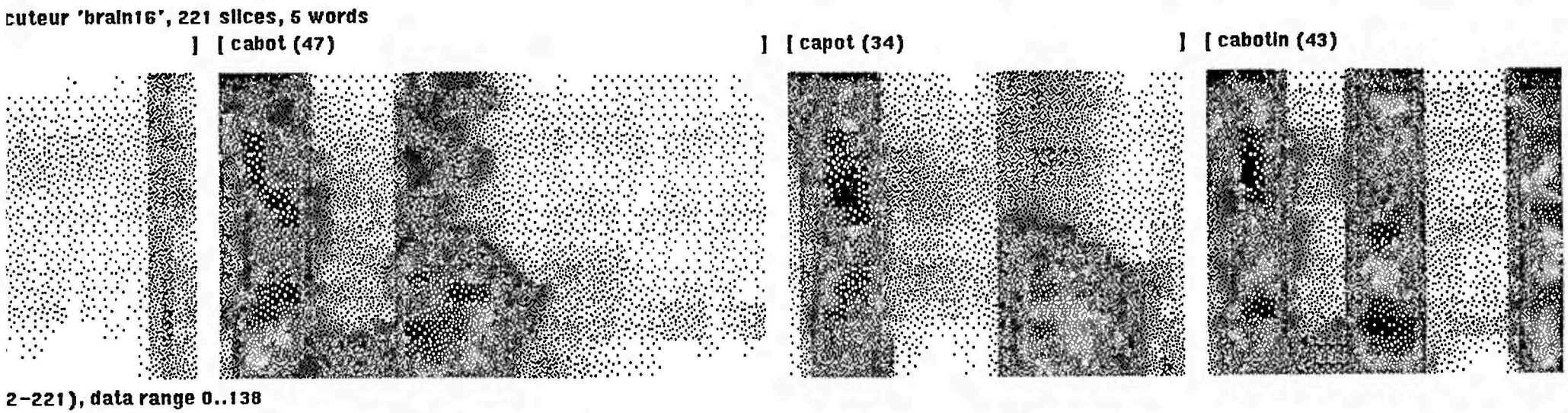
[4] F.JELINEK, L.R. BAHL, R.L.MERCER Continuous Speech Recognition: statistical methods. Handbook of statistics II, H.P.Krishnaiad Ed. , North Holland 1982.

[5] J.L. GAUVAIN, J. MARIANI, J.S. LIENARD: On the use of time compression for word-based speech recognition - IEEE - ICASSP - Boston 83.

[6] T. KOHONEN: The "Neural" phonetic typewriter. IEEE Computer, 11-22, March 1988.

[7] Y. LE CUN: Modèles connexionnistes de l'apprentissage, Thèse, Paris, (1987).

Exemples de spectrogrammes.
Ils permettent de se rendre compte de l'influence des prétraitements, et des problèmes d'alignement, surtout en fin de mot. Comparez par exemple les deux mots CABOT prononcés par deux locuteurs différents.



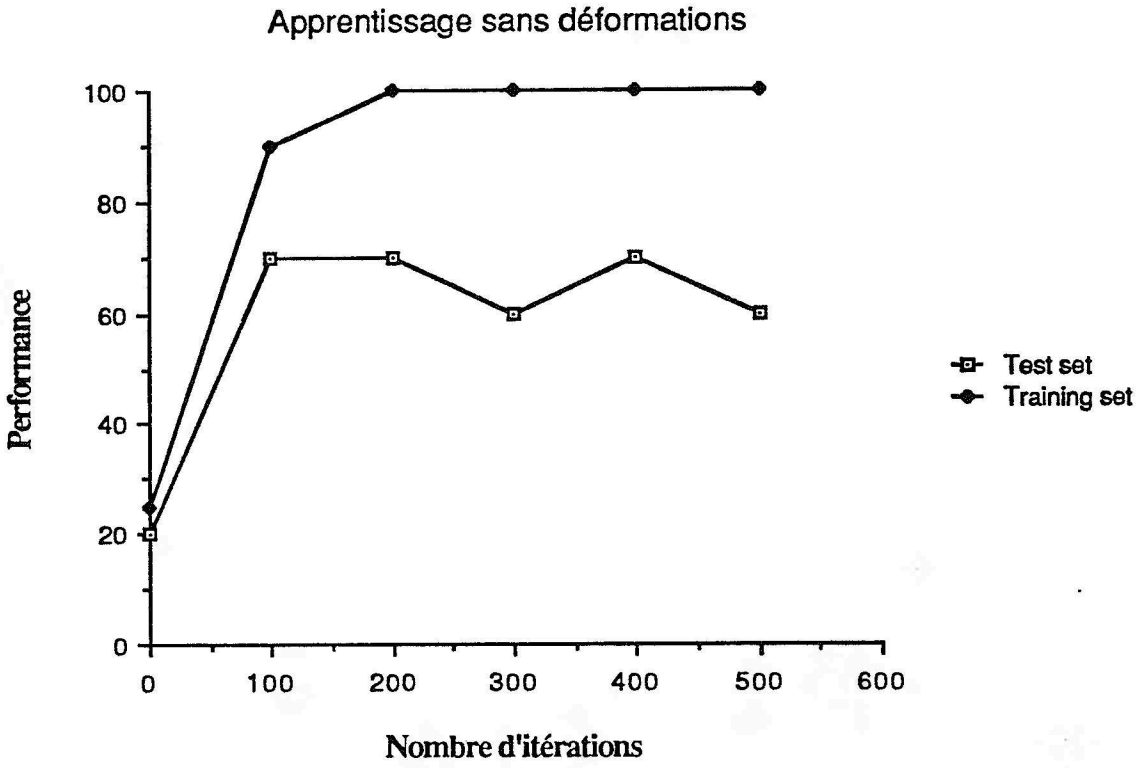


fig 5 - Courbe de performances, données non déformées
En abscisse, le nombre d'itérations, en ordonnée, les taux de performance. On peut observer l'allure erratique du taux de généralisation. Il est souvent faible et peu reproductible.

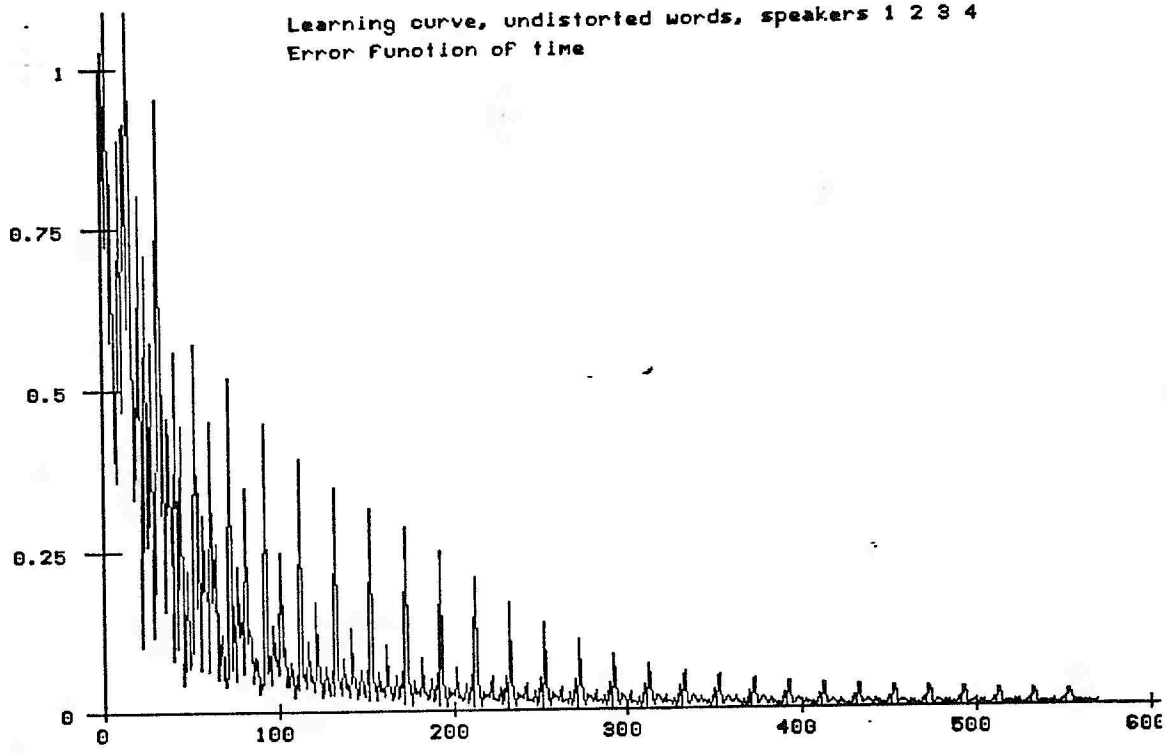


fig 6 - Courbe d'apprentissage , données non déformées
En abscisse, le nombre d'itérations, en ordonnée, l'erreur quadratique moyenne sur chaque mot présenté. Cet apprentissage est très rapide, mais les performances en généralisation sont faibles.

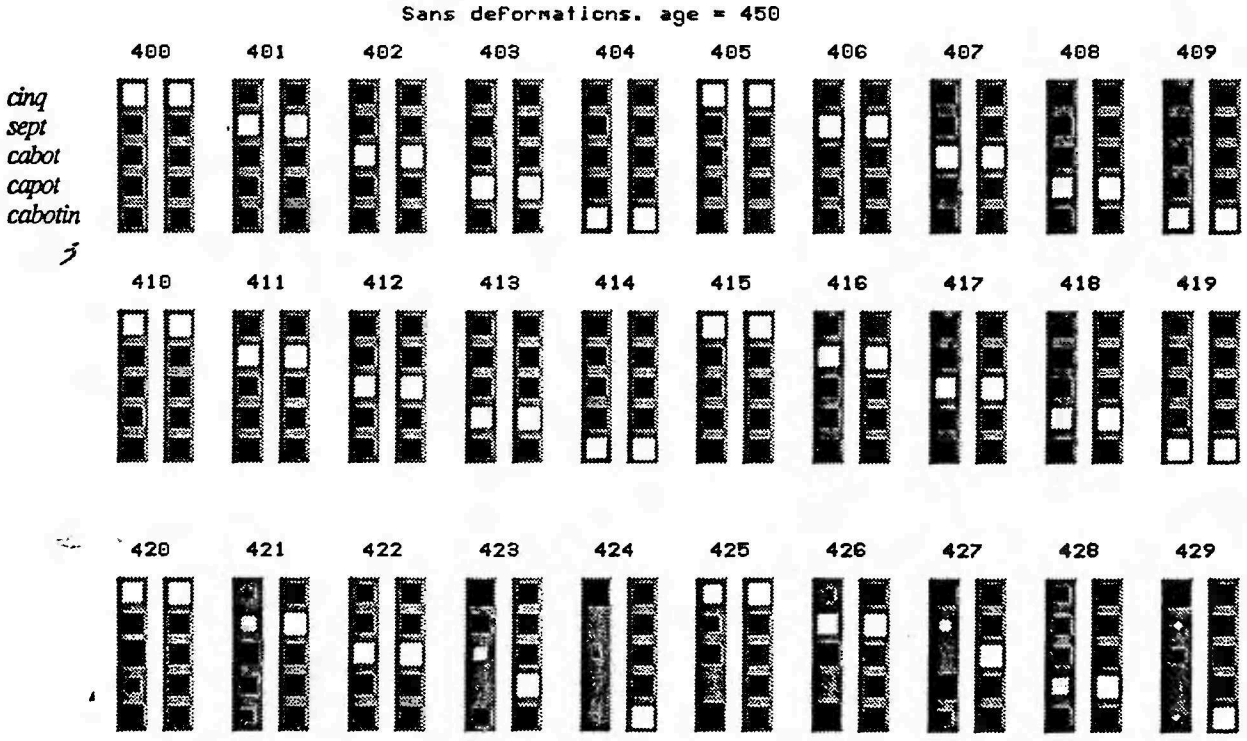


fig 7 - Saturation des sorties, données non déformées
L'ensemble (2) est constitué des mots 400 à 419, l'ensemble (T), des mots 420 à

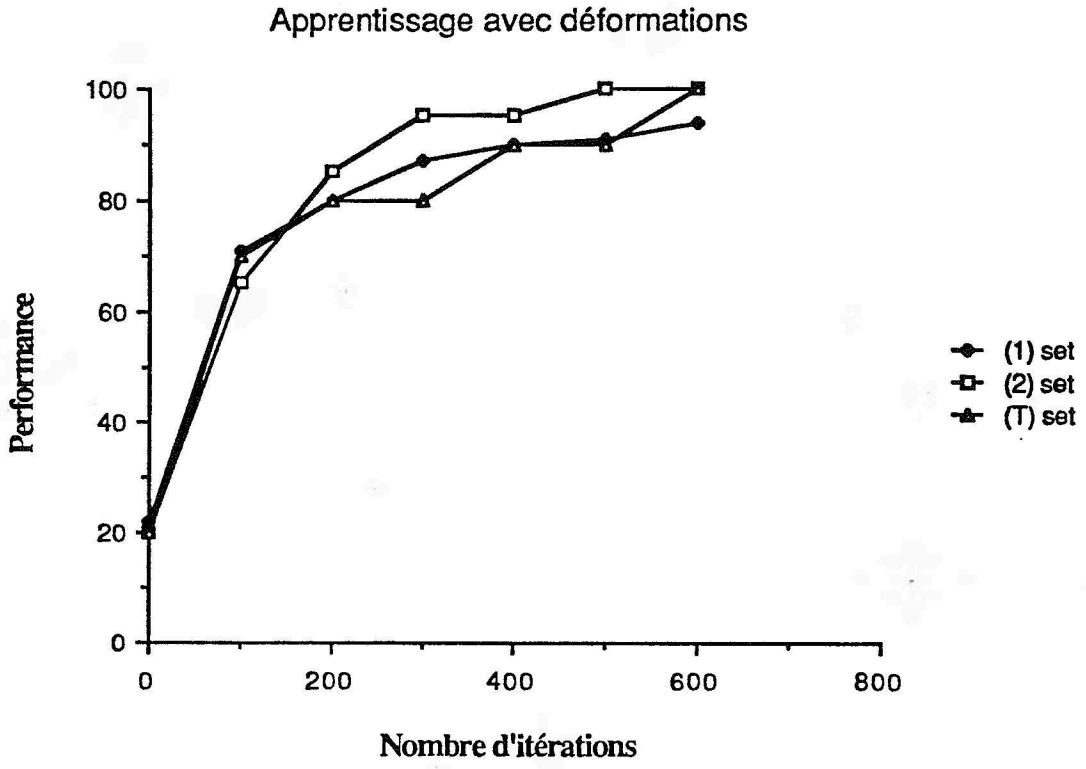


fig 8- Courbe de performances, données déformées
En abscisse, le nombre d'itérations, en ordonnée, les taux de performance. Le taux sur l'ensemble (1) ne dépasse jamais 94%, alors qu'il atteint 100% sur (2) et (T).

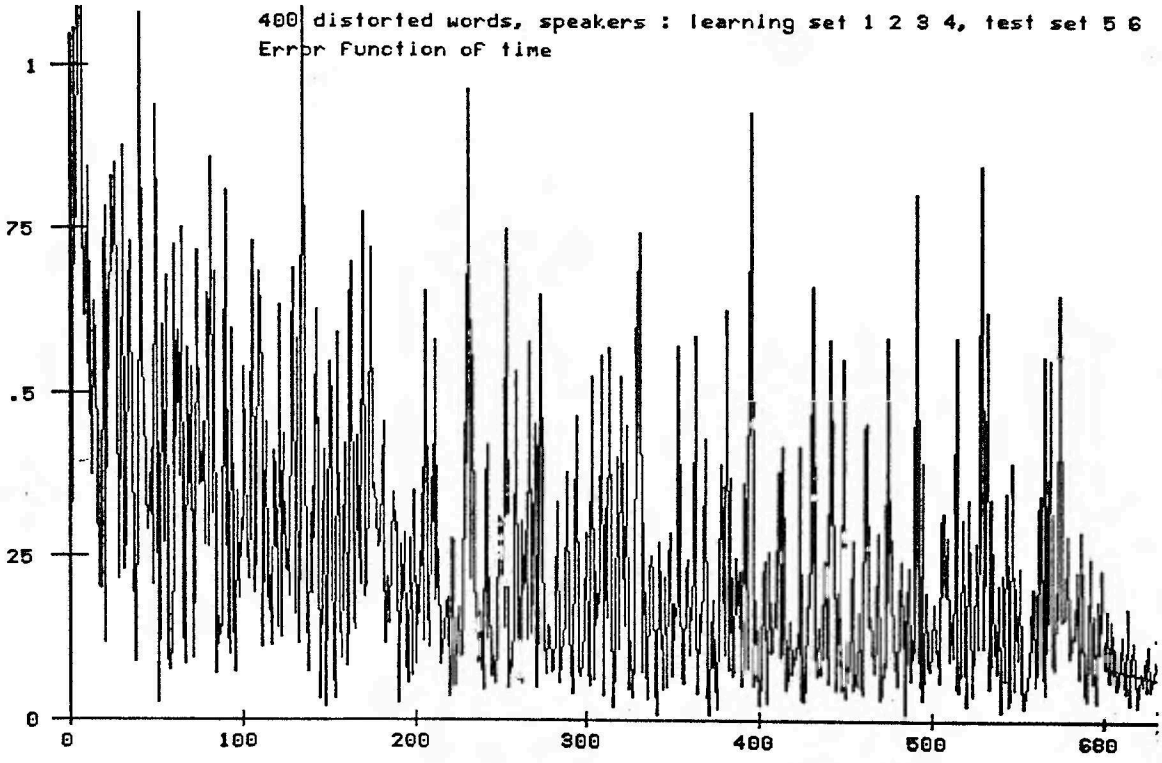


fig 9 - Courbe d'apprentissage , données déformées
En abscisse, le nombre d'itérations, en ordonnée, l'erreur quadratique moyenne sur chaque mot présenté. On n'apprend jamais totalement ces exemples, mais les performances sur les mots non deformés sont satisfaisantes.

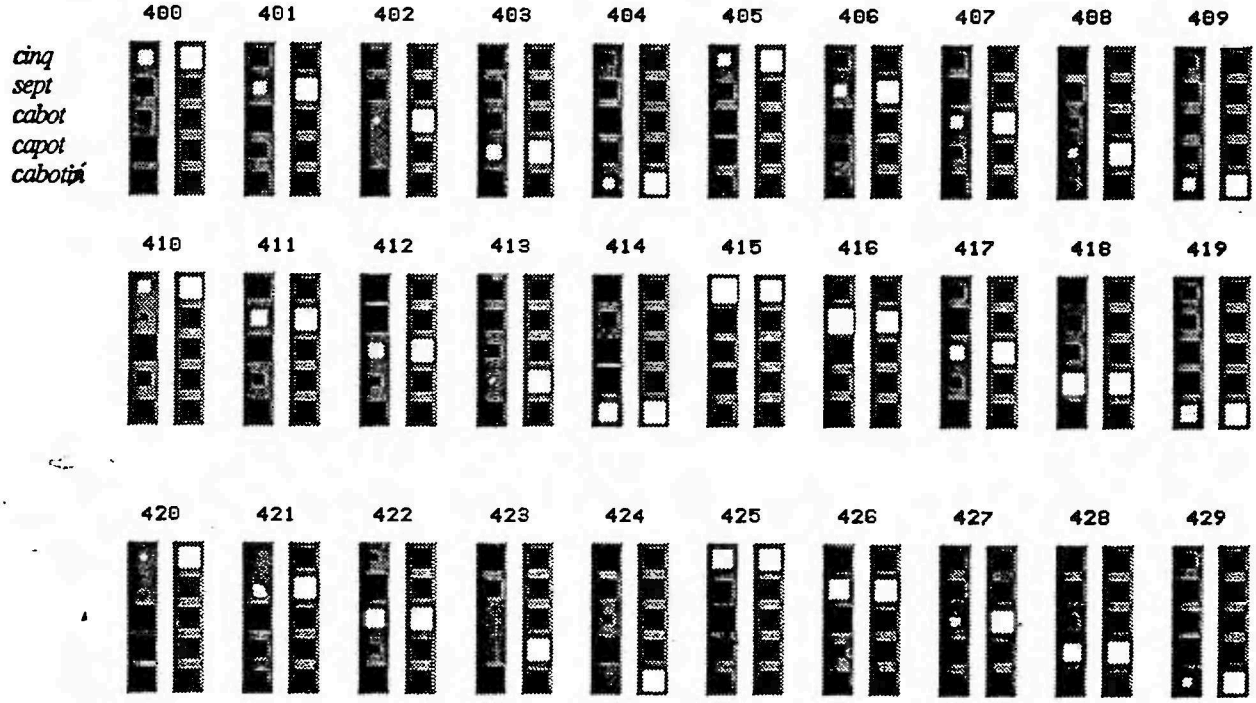


fig 10 - Saturation des sorties, données déformées
L'ensemble (2) est constitué des mots 400 à 419, l'ensemble (T), des mots 420 à 429. Les colonnes de gauche représentent les sorties obtenues, celles de droite, les

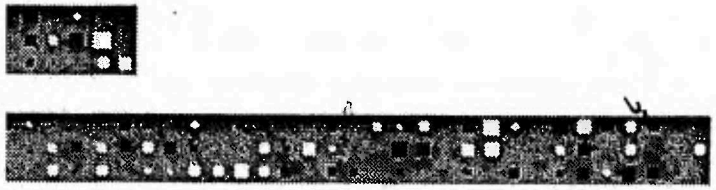
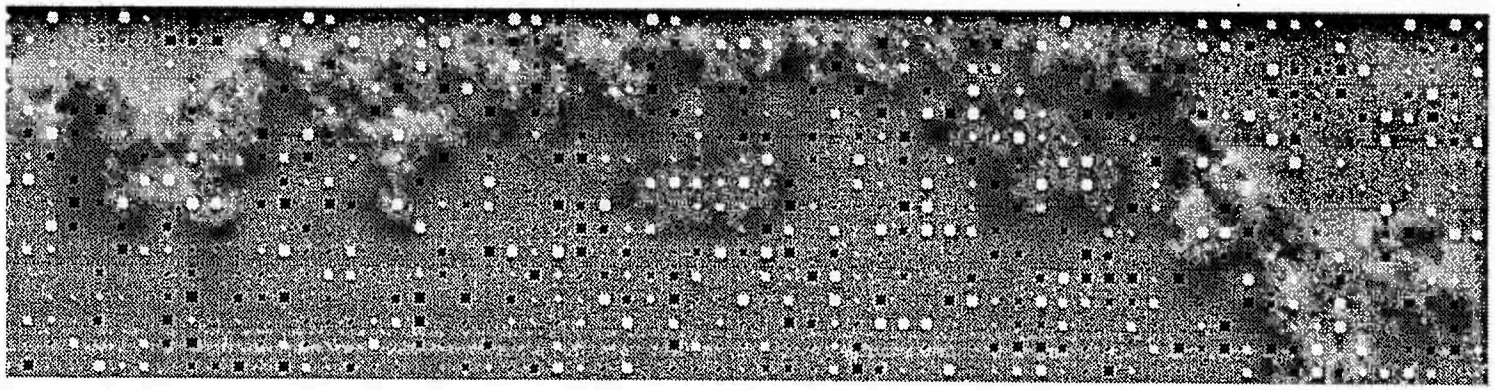


fig 11bis - On présente ici un bruit blanc en entrée. Aucune cellule de sortie n'est nettement plus active que les autres. De bas en haut: la couche d'entrée, les deux couches cachées, la couche de sortie. La taille des carrés montre la valeur des états des cellules, leur couleur, le signe de ces cellules.



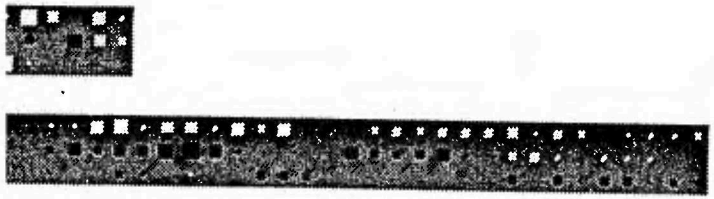
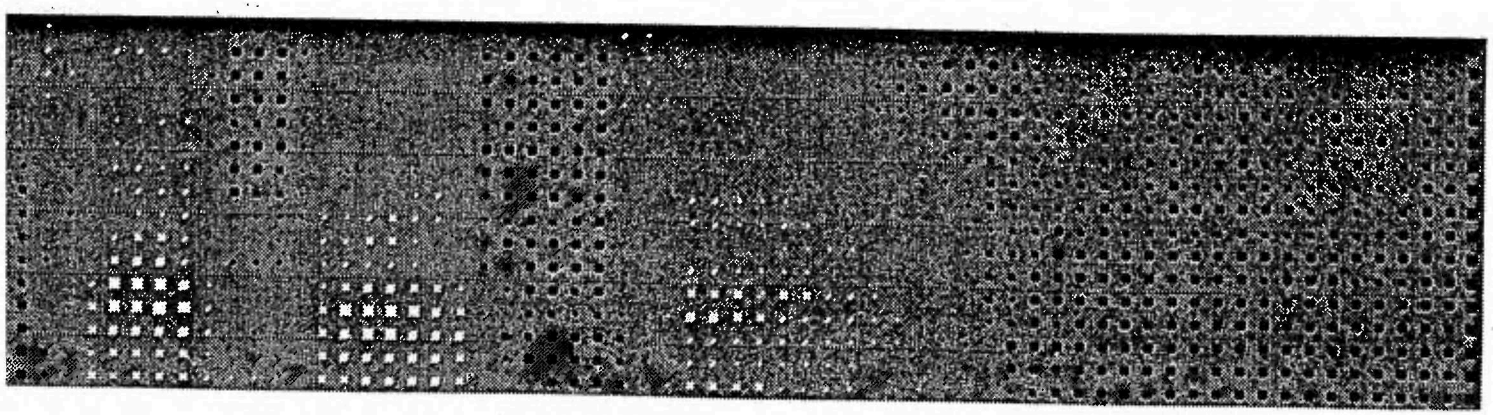


fig 11 - Le mot présenté en entrée est ici CABOTIN. On peut remarquer le mauvais alignement en fin de mot. De bas en haut: La couche d'entrée, les deux couches cachées, la couche de sortie. La taille des carrés montre la valeur des états des cellules, leur couleur, le signe de ces cellules.



[8] H.C. LEUNG, V.W. ZUE: Some phonetic recogntion experiments using artificial neural nets. Proceedings ICASSP 88, S-Vol.1, 422-425, 1988.

[9] J.S. LIENARD: Les processus de la communication parlée. Masson 1977.

[10] R.P. LIPPMANN, BEN GOLD: Neural-net classifiers useful for speech recognition. IEEE First Int. Conf. on Neural Networks, San Diego, IV-417-426, 1987.

[11] D. LUBENSKY:Learning spectral-temporal dependencies using connectionist networks. Proceedings ICASSP 88, S-Vol.1, 418-421, 1988.

[12] J. MARIANI, B. PROUTS, J.L. GAUVAIN, J.T. GANGOLF : Man Machine Speech Communication Systemes, including word-based recognition and text to speech synthesis. IFIP World Computer Congress, Paris 83.

[13] D. O'SHAUGHNESSY: Speech Communication, human and machine. Addison Wesley. 1987.

[14] R.W. PRAGER, T.D. HARRISON, F. FALLSIDE: Boltzmann machines for speech recognition. Computer, Speech and Language, 1, 3-27, 1986.

[15] D.E. RUMELHART, G.E. HINTON, R.J. WILLIAMS: Learning internal representations by error propagation. In "Parallel distributed processing", D.E. Rumelhart, J.L.McClelland eds, MIT Press, vol 1, (1986), 318-362.

[16] A. WAIBEL, T. HANAZAWA, G. HINTON, K. SHIKANO, K. LANG: Phoneme recognition: neural networks vs. Hidden Markov Models. Proceedings ICASSP 88, S-Vol.1, 107-110, 1988.

[17] A. WAIBEL, T. HANAZAWA, G. HINTON, K. SHIKANO, K. LANG: Phoneme recogntion using Time-Delay Neural Networks. Preprint ATR Interpreting Telephony research Laboratories, oct 1987.

[18] R.L. WATROUS, L. SHASTRI : Learning phonetic features using connectionist networks. IEE First Int. Conf. on Neural Networks, San Diego, IV-381-388, 1987.

[8] H.C. LEUNG, V.W. ZUE: Some phonetic recogntion experiments using artificial neural nets. Proceedings ICASSP 88, S-Vol.1, 422-425, 1988.

[9] J.S. LIENARD: Les processus de la communication parlée. Masson 1977.

[10] R.P. LIPPMANN, BEN GOLD: Neural-net classifiers useful for speech recognition. IEEE First Int. Conf. on Neural Networks, San Diego, IV-417-426, 1987.

[11] D. LUBENSKY:Learning spectral-temporal dependencies using connectionist networks. Proceedings ICASSP 88, S-Vol.1, 418-421, 1988.

[12] J. MARIANI, B. PROUTS, J.L. GAUVAIN, J.T. GANGOLF : Man Machine Speech Communication Systemes, including word-based recognition and text to speech synthesis. IFIP World Computer Congress, Paris 83.

[13] D. O'SHAUGHNESSY: Speech Communication, human and machine. Addison Wesley. 1987.

[14] R.W. PRAGER, T.D. HARRISON, F. FALLSIDE: Boltzmann machines for speech recognition. Computer, Speech and Language, 1, 3-27, 1986.

[15] D.E. RUMELHART, G.E. HINTON, R.J. WILLIAMS: Learning internal representations by error propagation. In "Parallel distributed processing", D.E. Rumelhart, J.L.McClelland eds, MIT Press, vol 1, (1986), 318-362.

[16] A. WAIBEL, T. HANAZAWA, G. HINTON, K. SHIKANO, K. LANG: Phoneme recognition: neural networks vs. Hidden Markov Models. Proceedings ICASSP 88, S-Vol.1, 107-110, 1988.

[17] A. WAIBEL, T. HANAZAWA, G. HINTON, K. SHIKANO, K. LANG: Phoneme recogntion using Time-Delay Neural Networks. Preprint ATR Interpreting Telephony research Laboratories, oct 1987.

[18] R.L. WATROUS, L. SHASTRI : Learning phonetic features using connectionist networks. IEE First Int. Conf. on Neural Networks, San Diego, IV-381-388, 1987.