LỜI CẢM ƠN

Em xin được cảm ơn thầy, Thạc sĩ Nguyễn Công Phương, người đã tận tình hướng dẫn em trong suốt quá trình thực tập và làm đồ án tốt nghiệp.

Xin được gửi lời cảm ơn đến thầy Nguyễn Quốc Cường, anh Nguyễn Quang Vinh, toàn thể các thầy cô giáo, các anh chị, các bạn sinh viên tại trung tâm MICA và tất cả những người đã giúp đỡ và tạo điều kiện cho em hoàn thành đồ án này.

MỤC LỤC

MỞ ĐẦU

I. Nội dung đồ án

Nằm trong khuôn khổ của dự án VLSR…. Nhằm hoàn thiện một hệ thống phân tích, tổng hợp, nhận dạng và xây dựng giao diện tương tác Người-Máy bằng ngôn ngữ tiếng Việt, đề tài tốt nghiệp của em nhằm mục đích xây dựng mộ hệ thống nhận dạng 10 từ khoá âm thanh để điều khiển thiết bị, máy móc.

Những công việc cần làm khi thực hiện đề tài:

- Nghiên cứu thuật toán nhận dạng từ khoá độc lập: Tìm hiểu về nhận dạng tiếng nói, đặc biệt là nhận dạng từ khoá rời rạc. Trong đó cần tìm hiểu các nghiên cứu về các thuật toán phân tích đặc trưng âm thanh (Tần số, cường độ, các hệ số quan trọng: MFCC, LPC…) và các mô hình nhận dạng phổ biến (Mô hình Markov ẩn, mô hình Neuron..)
- Chạy thử, nhận xét kết quả để tìm mô hình và tham số tối ưu cho bài toán với 10 từ điều khiển : Tắt, Bật, Chạy, Dừng, Tiến, Lùi, Trái, Phải, Trên, Dưới.
- Cài đặt thuật toán trên chip điều khiển DSP (Texas Intrusments): Chuẩn thuật toán nhận dạng về dạng code C để nạp vào chip DSP (vi xử lý được dùng ở đây là TMS320C713).

Với nội dung như trên, đồ án được trình bày với kết cấu như sau:

…

Phạm vi ứng dụng của đề tài này rất rộng, bài toán nhận dạng tiếng nói tự động có thể ứng dụng để thiết kế hệ thống giao tiếp với máy tính bằng lời nói, các hệ thống điều khiển tự động, điều khiển robot, hỗ trợ người tàn tật, quay số điện thoại bằng lời nói, cửa đóng mở tự động,…

Do hạn chế về kiến thức và thời gian có hạn, đồ án này khó tránh khỏi những thiếu sót. Vì vậy em rất mong nhận được sự chỉ dẫn và góp ý của các thầy cô giáo đề đồ án được hoàn thiện hơn.

II. Trung tâm nghiên cứu quốc tế MICA

Trung tâm nghiên cứu quốc tế MICA được thành lập vào năm 2001 nhằm đáp ứng nhu cầu phát triển Công nghệ thông tin, truyền thông và đa phương tiện ở Việt Nam.

Các lĩnh vực hiện đang được nghiên cứu tại MICA :

- Xử lý các tín hiệu phức tạp (âm thanh, hình ảnh).
- Phát triển các ứng dụng đa phương tiện.
- Xây dựng các thiết bị đo ảo và phân tán.

Trung tâm MICA có ba nhiệm vụ chính sau đây:

- Tiến hành các hoạt động nghiên cứu chất lượng cao.
- Đào tạo cán bộ Việt Nam.
- Trở thành đối tác đắc lực của các ngành công nghiệp dựa vào các chuyên gia tư vấn của nhóm

Các nhóm nghiên cứu của MICA

- Nhóm SIA: nghiên cứu các hệ thống đo lường tiên tiến.
- Nhóm TIM: nghiên cứu về xử lí thông tin đa phương tiện.
- Nhóm API: nghiên cứu về giao thức công nghiệp và ứng dụng.
- Nhóm tư vấn MICA: tư vấn cho các nhà đầu tư nước ngoài có ý định đầu tư sản xuất tại Việt Nam.

Nhóm SIA

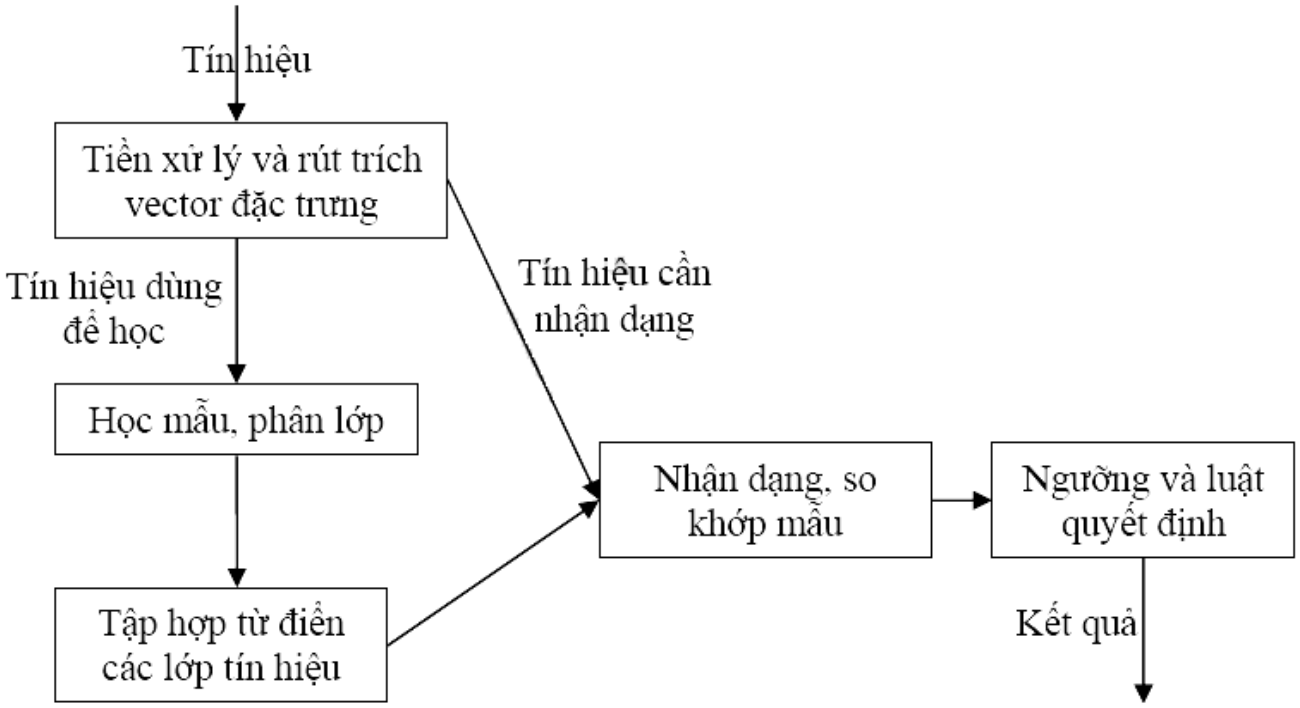
Nghiên cứu về các hệ thống đo lường tiên tiến, với các hướng nghiên cứu:

- Hệ thống đo lường nâng cao
- Dụng cụ đo ảo
- Xử lí nhúng
- Xử lí phân tán

Hướng nghiên cứu xử lí nhúng tập trung vào việc xử lí tín hiệu nâng cao trên các hệ thống chuyên biệt với chức năng thời gian thực như bộ xử lí DSP hay vi điều khiển. Đồng thời nghiên cứu và triển khai các kiến trúc nhúng và tự động hóa.

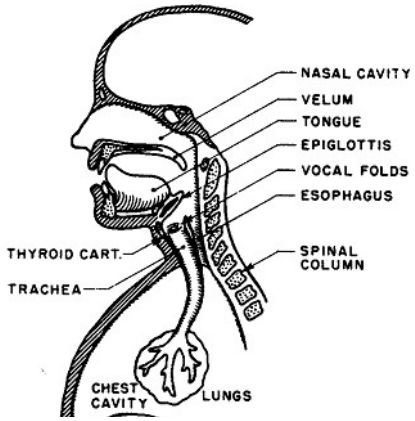
CHƯƠNG I: TỔNG QUAN VỀ NHẬN DẠNG TIẾNG NÓI

Những nghiên cứu đầu tiên về nhận dạng tiếng nói đã xuất hiện từ những năm 1950, với hệ thống nhận dạng các chữ số riêng biệt cho 1 người nói của Davis, Bidulph, và Balashek tại phòng thí nghiệm Bell. Và đến những năm 1980 thì các hệ thống nhận dạng tiếng nói đã được hoàn thiện với những thuật toán hiện đại. Những hệ thống với vốn từ vựng lớn, độ chính xác cao, nhận dạng tiếng nói liên tục, nhận dạng câu, cũng đã được xây dựng thành công. Và đến ngày nay, ngày càng nhiều các quốc gia thành công trong việc nghiên cứu các hệ thống tự động nhận dạng tiếng nói (ASR – Automatique Speech Recognition). Ở Việt Nam, một trong những trung tâm đi đầu trong việc nghiên cứu nhận dạng tiếng nói, là trung tâm MICA, ĐH Bách Khoa Hà Nội, với nhiều đề tài lớn về phân tích đặc trưng và nhận dạng tiếng Việt, và cũng đã có nhiều kết quả thành công.


Tín hiệu
Tiền xử lý và rút trích vector đặc trưng
Tín hiệu dùng để học
Học mẫu, phân lớp
Tập hợp từ điển các lớp tín hiệu
Tín hiệu cần nhận dạng
Nhận dạng, so khớp mẫu
Ngưỡng và luật quyết định
Kết quả

Hình 1Mô hình tổng quát của một hệ nhận dạng tiếng nói

I. Các khái niệm và đặc điểm âm học của tiếng nói



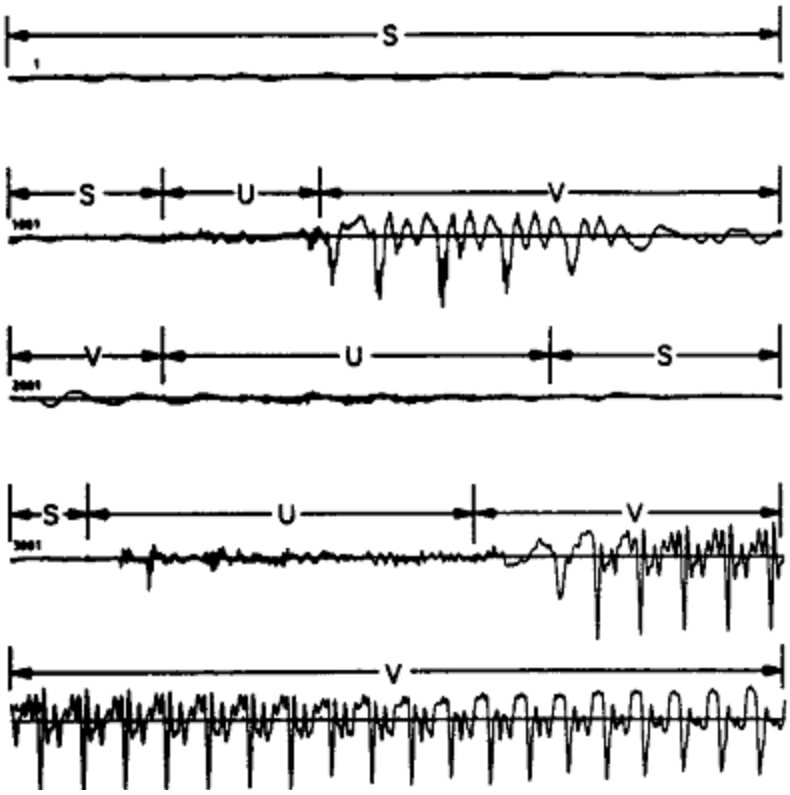
Hình II-2: Cấu tạo của hệ thống phát âm ở người

Quá trình tạo tiếng nói ở người

Mỗi người bình thường đều có một hệ thống phát ra âm thanh, hay tiếng nói.

Không khí được đưa vào phổi thông qua cơ chế hít thở thông thường, sau đó được đẩy từ phổi qua khí quản và làm rung các dây thanh quản. Các dòng khí được ngắt thành các dao động tuần hoàn khi đi qua khoang họng, khoang miệng, và cả khoang mũi. Tuỳ thuộc vào vị trí khác nhau của các bộ phận phát âm (hàm, lưỡi, môi, miệng..) mà các âm thanh khác nhau được phát ra.

Biểu diễn tiếng nói trong miền thời gian và miền tần số



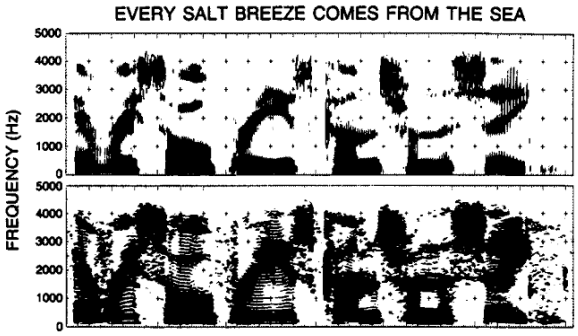
Hình II-3: Đồ thị theo thời gian tín hiệu ban đầu của câu "It's time"

Tín hiệu tiếng nói có thể coi là biến đổi chậm theo thời gian, khi ta phân tích trong những khoảng thời gian ngắn (từ 5 đến 100ms), các đặc điểm của nó khá ổn định. Tuy nhiên với những khoảng thời gian lớn hơn (trên 1/5s) các đặc điểm của tín hiệu có sự thay đổi phản ánh sự khác nhau của các từ được nói.

Các sự kiện chính khi một từ được phát ra có thể được phân loại (dán nhãn) theo một cách đơn giản là:

- Yên lặng (S – silence): khi không có âm nào được phát ra
- Không âm (U – unvoice): Khi dây thanh quản không rung
- Âm (V – voice): khi dây thanh quản rung và tạo các tín hiệu âm thanh giả tuần hoàn (tiếng nói)

Một cách biểu diễn khác của tín hiệu âm thanh là thông qua phổ tần số.



Hình II-3: Biểu đồ biểu diễn phổ tần số theo thời gian

Ngoài ra cũng có thể biểu diễn các đặc tính của âm thanh thông qua các tham số của phổ tần dựa trên một mô hình tạo tiếng nói. Ví dụ như các formant,...

II. Các hướng tiếp cận trong việc tự động nhận dạng tiếng nói

Một cách tổng quan, có 3 hướng tiếp cận một hệ thống nhận dạng tiếng nói :

1. Phương pháp: Ngữ âm-Âm học (Acoustic-Phonetic)
2. Phương pháp: Nhận dạng mẫu (Pattern-recognition)
3. Sử dụng: Trí tuệ nhân tạo (Artifical Intelligence)

- Phương pháp ngữ âm - âm học dựa trên những lý thuyết về âm học và ngữ âm cho rằng có hữu hạn các đơn vị âm học riêng biệt trong một ngôn ngữ và do vậy có thể được đặc trưng bởi một tập các thuộc tính biểu hiện trong tín hiệu tiếng nói hoặc biểu diễn phổ của nó. Bước đầu tiên của phương pháp này là phân đoạn tín hiệu tiếng nói thành các vùng có đặc tính âm học đặc trưng cho một đơn vị ngữ âm và gán cho mỗi vùng một nhãn ngữ âm. Bước thứ 2 là xác định một từ có nghĩa từ các chuỗi nhãn ngữ âm đó.

  Mặc dù vậy, phương pháp này gặp phải khá nhiều khó khăn khi triển khai thực tế như:
  - Sự đòi hỏi một vốn kiến thức khá lớn về các đặc diểm âm học của các đơn vị ngữ âm.
  - Sự lựa chọn các đặc trưng được thực hiện một cách không chắc chắn.
  - Không có một thuật toán hay thủ tục nào đủ mạnh để thực hiện dán nhãn các tín hiệu huấn luyện đủ mạnh cho nhiều khu vực khác nhau và được chấp nhận bởi đông đảo các nhà chuyên gia về ngôn ngữ học.
- Phương pháp nhận dạng mẫu sử dụng trực tiếp các mẫu thành phần của tiếng nói mà không quan tâm nhiều đến các đặc tính về mặt ngữ âm. Trong hầu hết các thuật toán nhận dạng mẫu, có 2 bước chính là huấn luyện và nhận dạng.

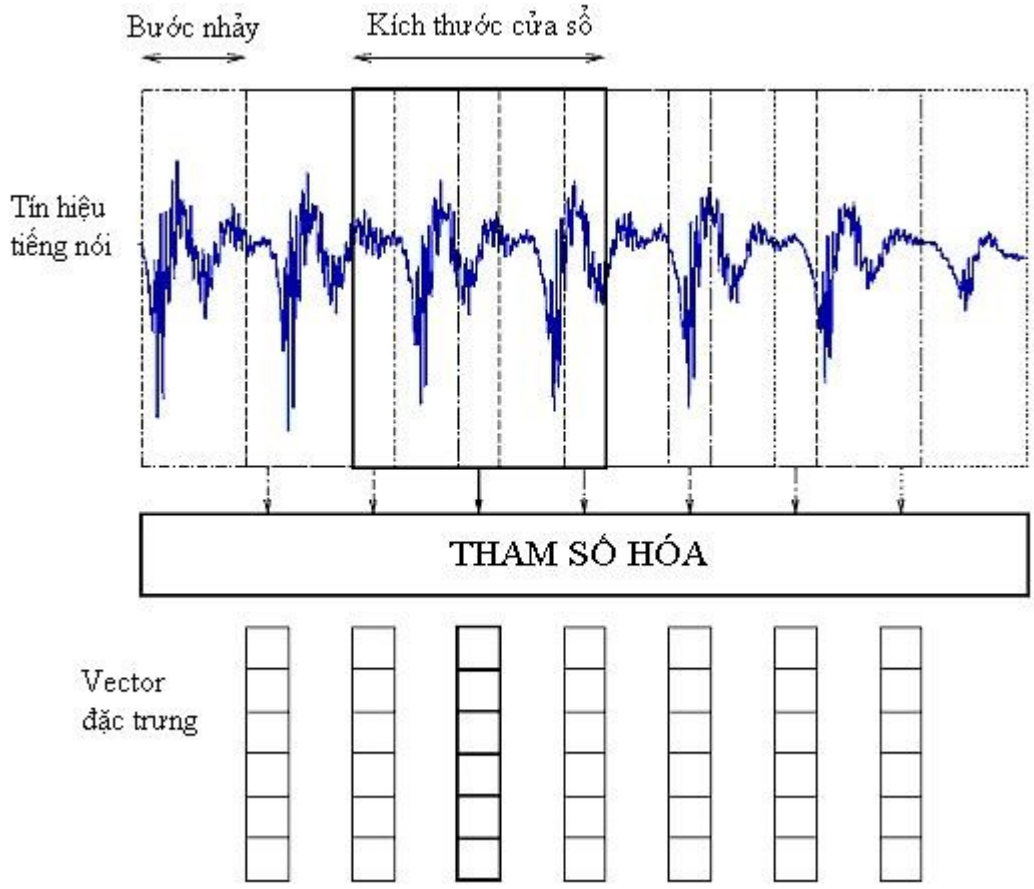
  Các "kiến thức" của hệ được xây dựng qua thủ tục huấn luyện. Hệ ghi nhận các đặc tính của các tín hiệu tiếng nói được "học" thông qua các tham số đặc trưng. Tín hiệu cần được nhận dạng sẽ được tính toán để đưa ra kết quả có xác suất gần nhất.

  Hiện nay, phương pháp nhận dạng mẫu đã trở nên phổ biến trong các nghiên cứu về nhận dạng tiếng nói, dựa trên các đặc điểm:
  - Lựa chọn đơn giản, phương pháp dễ hiểu.
  - Các thuật toán và quy luật trong phương pháp, cũng như các tập đặc tính đối với các bộ từ vựng và các người dùng khác nhau là bất biến và rất mạnh.

- Độ tin cậy và khả năng mở rộng cao.

Phương pháp sử dụng trí tuệ nhân tạo là sự kết hợp giữa 2 phương pháp Ngữ âm – Âm học và nhận dạng mẫu, sử dụng những khái niệm và ý tưởng của cả 2 phương pháp trên. Phương pháp này thực hiện các thủ tục nhận dạng dựa theo cách con người tưởng tượng, phân tích và đưa ra quyết định dựa vào các đặc tính âm học. Một hệ chuyên gia sẽ được sử dụng để phân đoạn và dán nhãn, thực hiện thuật toán học và thích ứng theo thời gian, có thể sử dụng một mạng neuron để học mối quan hệ giữa các nhân tố ngữ âm và các biến vào khác (âm học, từ vựng, cú pháp, ngữ nghĩa...) cũng như là phân biệt gữa các lớp âm gần giống nhau.

III. Các phương pháp phân tích đặc trưng của tín hiệu tiếng nói



Hình 2 Ttrích đặc trưng của tiếng nói

1. Mô hình LPC (Linear Predictive Coding model)

Mô hình LPC được sử dụng khá rông rãi trong các hệ thống nhận dạng tiếng nói là bởi các lý do sau:

- LPC cung cấp một mô hình tốt của tín hiệu tiếng nói. Đặc biệt đối với các trạng thái gần

ổn định của âm thanh, mô hình LPC cho ta một xấp xỉ khá tốt của phổ âm thanh. Tuy trong các vùng ngắn và không âm, mô hình LPC hoạt động kém hiệu quả hơn vùng có âm, nhưng nó vẫn cung cấp một mô hình có thể sử dụng tốt cho mục đích nhận dạng tiếng nói.

- Cách mà LPC được ứng dụng trong việc phân tích tín hiệu tiếng nói dẫn đến một sự phân tách hợp lý các âm nguồn âm thanh. Và như vậy, việc biểu diễn chi tiết các đặc điểm của các dải âm thanh là hoàn toàn có thể.
- Phương pháp tính toán của LPC chính xác về mặt toán học và đơn giản, trực tiếp trong việc cài đặt lên cả phần cứng hoặc phần mềm. Số lượng tính toán trong xử lý LPC cũng ít hơn trong phương pháp filters-bank
- Mô hình LPC hoạt động tốt trong các ứng dụng nhận dạng. Knh nghiệm cho thấy, các hệ thống nhận dạng sử dụng mô hình LPC cho kết quả tốt hơn so với các hệ sử dụng filter-bank.

1.1. Mô hình LPC

Ý tưởng cơ bản của mô hình LPC là một mẫu tiếng nói cho trước tại thời điểm n, s(n) có thể được xấp xỉ bởi một tổ hợp tuyến tính của p mẫu tín hiệu quá khứ, theo biểu thức sau:

$$s(n) \approx a_1 s(n-1) + a_2 s(n-2) + ... + a_p s(n-p) \quad (1)$$

Trong đó các hệ số a1,a2,…ap được coi như không đổi trong khung thời gian phân tích. Biến đổi công thức (1), thêm vào đại lượng Gu(n) ta có:

$$s(n) = \sum_{i=1}^{p} a_i s(n-i) + Gu(n) \quad (2)$$

trong đó u(n) là kích thích chuẩn hoá và G là hệ số của kích thích. Bằng biến đổi sang miền Z ta có quan hệ:

$$S(z) = \sum_{i=1}^{p} a_i z^{-i} S(z) + GU(z) \quad (3)$$

từ đó dẫn đến hàm truyền của mô hình:

$$H(z) = \frac{S(z)}{GU(z)} = \frac{1}{1 - \sum_{i=1}^{p} a_i z^{-i}} = \frac{1}{A(z)} \quad (4)$$

1.2. Các biểu thức của phân tích LPC

Dựa trên mô hình liên hệ chính xác giữa s(n) và u(n)

$$s(n) = \sum_{k=1}^{p} a_k s(n-k) + Gu(n)$$

ta coi tổ hợp tuyến tính của các tín hiệu quá khứ là một ước lượng của $\hat{s}(n)$

$$\tilde{s}(n) = \sum_{k=1}^{p} a_k s(n-k) \tag{5}$$

Sai số ước lượng e(n) được định nghĩa:

$$e(n) = s(n) - \tilde{s}(n) = s(n) - \sum_{k=1}^{p} a_k s(n-k) \tag{6}$$

Với hàm truyền sai số :

Vấn đề cơ bản của phân tích dự đoán tuyến tính là xác định tập các hệ số{ak} tiên đoán trực tiếp từ tín hiệu tiếng nói để các đặc tính phổ của bộ lọc trùng với tín hiệu sóng tiếng nói trong cửa sổ phân tích.

Do các đặc điểm phổ tần của tiếng nói t hay đổi theo thời gian, do vậy các hệ số tiên đoán tại một thời điểm n phải được ước lượng từ một phân đoạn ngắn của tín hiệu tiếng nói xảy ra gần n. Và như thế, hướng tiếp cận cơ bản là tìm một tập các hệ số tiên đoán có sai số dự đoán bình phương đạt cực tiểu trên một phân đoạn ngắn của tín hiệu sóng tiếng nói. Thông thường, tín hiệu tiếng nói được phân tích trên các khung liên tiếp với độ dài khoảng 10ms.

Bài toán này được giải dựa trên phương pháp tự tương quan, khi đó các hệ số ak ước lượng đc sẽ là nghiệm của phương trình:

$$\sum_{k=1}^{p} r_n(|i-k|)\hat{a}_k = r_n(i) \qquad \text{i} \le \text{i} \le \text{p}$$

Hay có thể biểu diễn dưới dạng ma trận như sau:

$$\begin{bmatrix} r_n(0) & r_n(1) & r_n(2) & \cdots & r_n(p-1) \\ r_n(1) & r_n(0) & r_n(1) & \cdots & r_n(p-2) \\ r_n(2) & r_n(1) & r_n(0) & \cdots & r_n(p-3) \\ \vdots & \vdots & \vdots & & \vdots \\ r_n(p-1) & r_n(p-2) & r_n(p-3) & \cdots & r_n(0) \end{bmatrix} \begin{bmatrix} \hat{a}_1 \\ \hat{a}_2 \\ \hat{a}_3 \\ \vdots \\ \hat{a}_p \end{bmatrix} = \begin{bmatrix} r_n(1) \\ r_n(2) \\ r_n(3) \\ \vdots \\ r_n(p) \end{bmatrix} \tag{7}$$

Với r(k) là hệ số tự tương quan của tín hiệu dời đi k mẫu

$$r(k) = \sum_{n=-\infty}^{\infty} x(n)x(n+k) \tag{8}$$

Hệ phương trình này được giải bằng thuật toán Levinson-Durbin.

Thuật toán Levinson-Dunbin:

Khởi tạo: p=1
Tính sai số bình phương trung bình bậc nhất:

$$E_1 = r(0)\left(1 - a_1^2(1)\right)$$

trong đó $a_1(1) = -\frac{r(1)}{r(0)}$

Đệ qui: Với p=2,3...,P

1) Tính hệ số Kp (hệ số PARCOR)

$$K_p = \frac{r(p) - \sum_{i=1}^{p-1} a_i r(p-i)}{E_{p-1}}$$

2) Tínhcác hệ số dự báo bậc p:

$$a_p(k) = a_{p-1}(k) - K_p a_{p-1}(p-k)$$ Với k=1,2,...,p-1

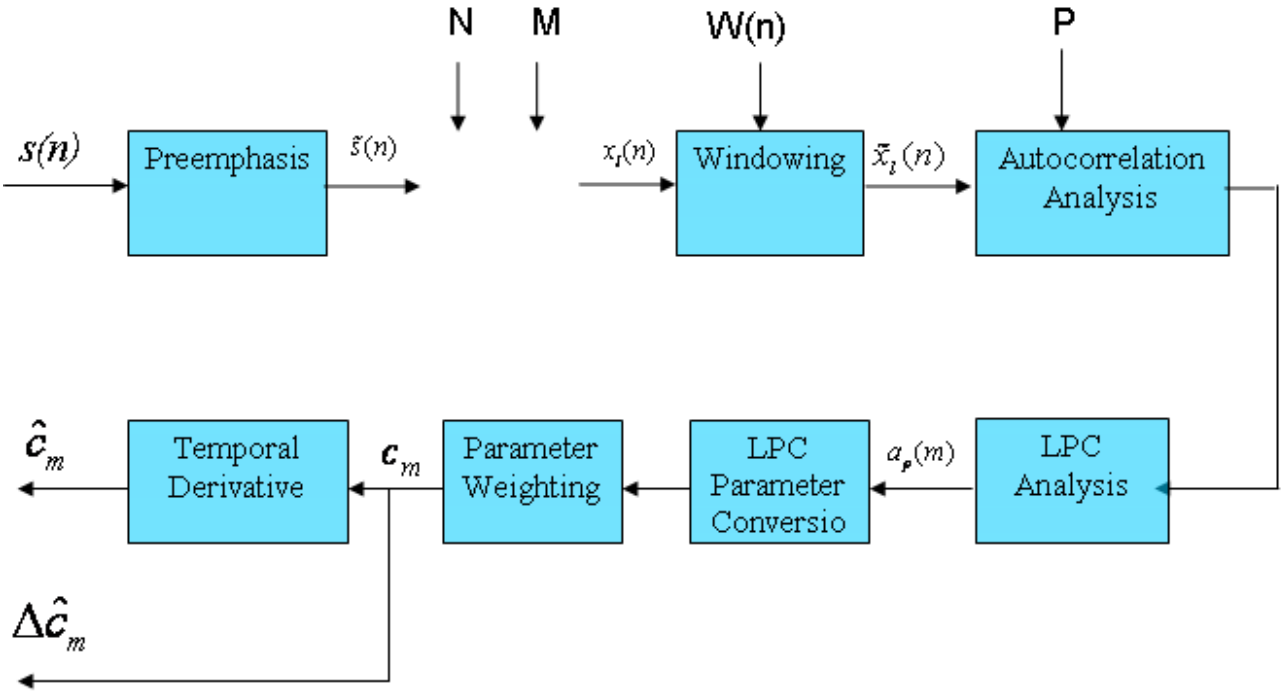
ap(p) = Kp

3) Tính sai số bình phương trung bình bậc p:

$$E_p = E_{p-1}\left(1 - K_p^2\right)$$

4) Quay lại bước 1, thay p bằng p+1 nếu p=< P

1.3. Các bước thực hiện thuật toán LPC để trích đặc trưng của tín hiệu



Hình 3Các bước thực hiện thuật toán LPC

Nguyễn Việt Hưng
KS CLC – THCN – K48

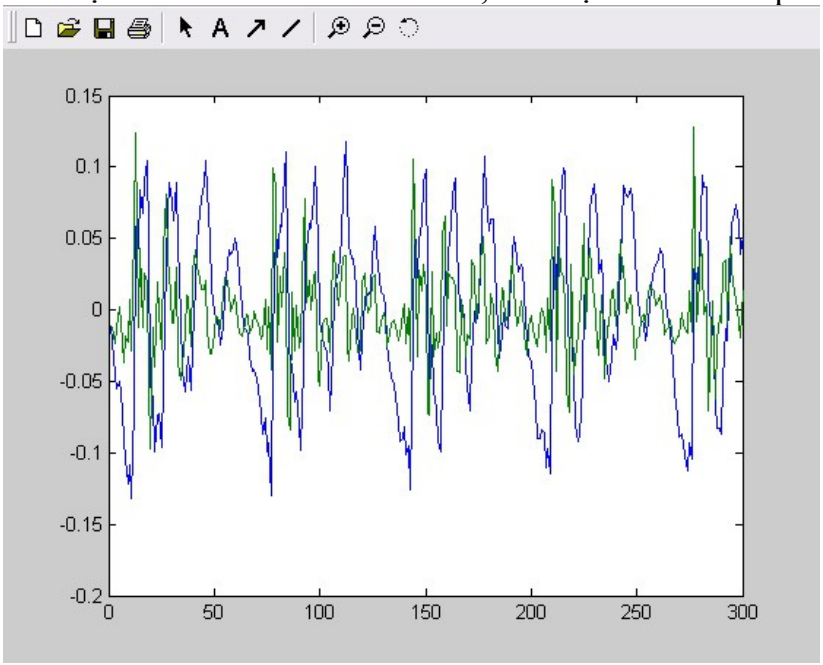
- Bước 1: Lọc nhiễu, sử dụng bộ lọc thông cao

  $H(s) = \frac{s}{s + w_c}$

  Với tần số cắt dưới 50-250 Hz để lọc nhiễu tần số thấp do microphone gây ra.
- Bước 2: Pre-emphasis để làm bằng phổ (spectrally flaten)
  Tín hiệu s(n) được cho qua một bộ lọc thông thấp:
  $H(z)=1-az^{-1}$

  $\tilde{x}(n) = x(n) - ax(n-1)$ với $0.9 \le a \le 1$

  Thường chọn a=0.9375
  Tín hiệu ban đầu màu xanh da trời, tín hiệu sau Pre-emphasis màu xanh lá



Hình II-5: Tín hiệu preemphasized

- Bước 3:Tín hiệu được phân đoạn thành các frame, mỗi frame N mẫu, độ chồng lấp M mẫu

  $M = \frac{1}{3}N$

  Chọn tần số lấy mẫu
  Chọn N và M

- Bước 4: cửa sổ hóa các frame, nhằm giảm sự gián đoạn của tín hiệu tại đầu và cuối mỗi frame. Hay nói cách khác là giảm dần tín hiệu về 0 tại các khoảng bắt đầu và kết thúc của mỗi khung.
  Cửa sổ thường được dùng là cửa sổ Hamming.

$$W(n) = \begin{cases} 0.54 - 0.46\cos\left(\frac{2n\pi}{M}\right) & 0 \le n \le M \\ 0 & n \notin [0, M] \end{cases}$$

➔ $x(n) = x_l(n)W(n) \qquad 0 \le n \le N-1$

- Bước 5: Xác định hệ số LPC dung thuật toán Levinson – Dubin cho mỗi frame

$\hat{x}(n) = -a(1)x(n-1) - a(3)x(n-2) - ... - a(p)x(n-p)$

$a_p(m)$ = LPC coefficient , $0 \le m \le p$

Ta có p + 1 hệ số a, với a(0) = 1.

Chọn p và bỏ a(0) . Ta có vecto đặc trưng có độ dài = p cho mỗi frame.

- Bước 6 : Chuyển các hệ số dự báo tuyến tính thành các hệ số ceptral.

$c_0 = \ln \sigma^2$ (σ^2 là hệ số G của mô hình LPC)

$c_m = a_m + \sum_{k=1}^{m-1}\left(\frac{k}{m}\right)c_k a_{m-k}$ Với $1 \le m \le p$

$c_m = \sum_{k=Q-p}^{m-1}\left(\frac{k}{m}\right)c_k a_{m-k}$ Với p< m <Q (thường lấy Q=3/2p)

Hệ số ceptral là các hệ số của biến đổi Fourier cho log cường độ phổ. Các hệ số này được cho là đáng tin cậy hơn các hệ số LPC

- Bước 7: Tính toán các hệ số ceptral có trọng số

$\hat{c}_m = w_m c_m$ Với $1 \le m \le Q$

Trong đó :

$$w_m = \left[1 + \frac{Q}{2}\sin\left(\frac{\pi m}{Q}\right)\right] \qquad \text{Với } 1 \le m \le Q$$

Việc này nhằm giảm sự ảnh hưởng của overall spectral slope tới các hệ số ceptral bậc thấp và nhiễu tới các hệ số ceptral bậc cao. Thực chất là ta dùng một cửa sổ ceptral giảm dần ở hai đầu. Hàm w_m de-emphasize c_m quanh m = 1 và m = Q

- Bước 8 : Tính đạo hàm của các hệ số ceptral

$$\frac{\partial c_m(t)}{\partial t} = \Delta c_m(t) \approx \mu \sum_{k=-K}^{K} k c_m(t+k)$$

Trong đó μ là hằng số chuẩn hoá (thường lấy 0.375)và (2K+1) là số frame được tính.

- Kết thúc:

Vector đặc trưng là vector có 2Q thành phần gồm Q hệ số ceptral có trọng số và Q đạo hàm của hệ số ceptral.

Một số tham số thường dùng [1]

Tham số	F_s = 6.67kHz	F_s = 8 kHz	F_s = 10 kHz
N	300 (45 msec)	240 (30 msec)	300 (30 msec)
M	100 (15 msec)	80 (10 msec)	100 (10 msec)
P	8	10	10
Q	12	12	12
K	3	3	3

Bảng II-1Bảng tham số LPC

2. Phương pháp MFCC (Mel-Frequency Ceptrum Coefficients)

Bên cạnh LPC thì MFCC cũng là 1 phương pháp phổ biến. MFCC dựa trên những nghiên cứu về những dải thông quan trọng (critical) của tai người đối với tần số. Và để thu được những đặc trưng ngữ âm quan trọng người ta sử dụng các bộ lọc tuyến tính với dải tần thấp và các bộ lọc có đặc tính loga với dải tần số cao. Trong phương pháp này, ta sử dụng Mel-scale tuyến tính với các tần số dưới 1000Hz và tỉ lệ loga với các tần số trên 1000Hz.

2.1. Mel-frequency scale

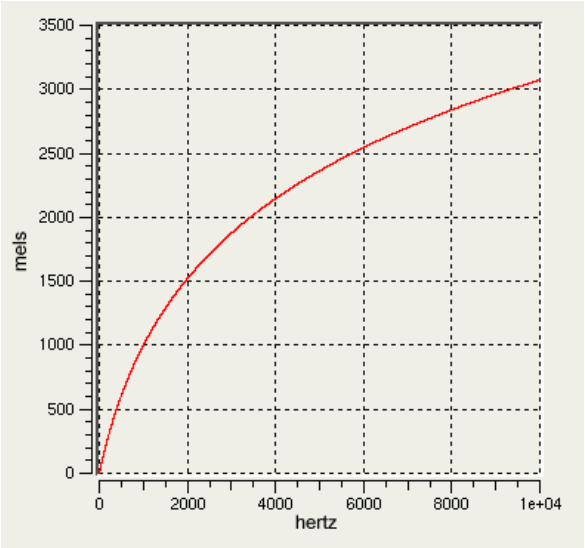
Các nghiên cứu tâm sinh lý đã chỉ ra rằng nhận thức của con người đối với tần số của âm thanh của các tín hiệu tiếng nói không theo một tỉ lệ tuyến tính. Vì vậy người ta sử dụng một cách đo dựa trên tỉ lệ "Mel".

Để chuyển từ thang tần số sang mel scale ta sử dụng công thức

$$m = 1127.01048 \ln(1 + \frac{f}{700}) \quad \text{(hay Mel(f) = m = } 2595\log_{10}(1+f/700)\text{)} \quad \text{(Mel)}$$

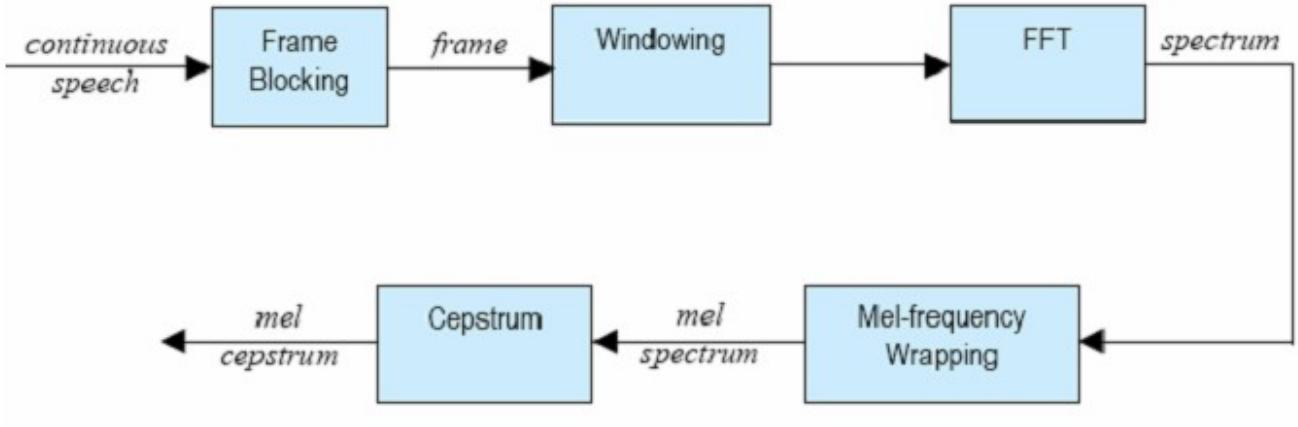
Và công thức biến đổi ngược :

$$f = 700(e^{m/1127.01048} - 1)$$



Hình II-6 Tấn số Mel

2.2. Thực hiện trích đặc trưng bằng phương pháp MFCC



Hình 4 Qui trình trích đặc trưng MFCC

a) Frame Blocking

Tín hiệu được chặt thành từng frame N mẫu với độ chồng lấp M mẫu.
Thường lấy M = 1/3N
(Ta lấy N=512 – để dễ cho việc tính FFT và M=100)

b) Cửa sổ hoá

Cửa sổ hóa các frame, nhằm giảm sự gián đoạn của tín hiệu tại đầu và cuối mỗi frame. Hay nói cách khác là giảm dần tín hiệu về 0 tại các khoảng bắt đầu và kết thúc của mỗi khung.
Cửa sổ thường được dùng là cửa sổ Hamming.

$$W(n) = \begin{cases} 0.54 - 0.46\cos\left(\frac{2n\pi}{M}\right) & 0 \le n \le M \\ 0 & n \notin [0, M] \end{cases}$$

➔ $x_l(n) = x_l(n)W(n) \qquad 0 \le n \le N-1$

c) Biến đổi Fourier nhanh (FFT)

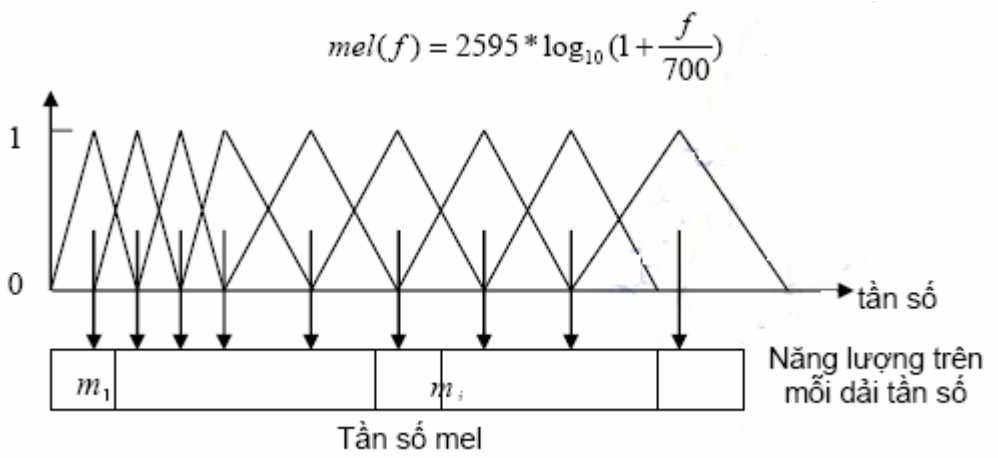
Biến đổi các khung tín hiệu từ miền thời gian về miền tần số.

$$X_n = \sum_{k=0}^{N-1} x_k e^{-2\pi jkn/N}$$ Với n = 0,1,2….,N-1

d) Chuyển đổi Mel-Frequency

Thực hiện chuyển đổi theo công thức (Mel).

$B(f) = 1127.01048\ln(1+\frac{f}{700})$ {hoặc có thể làm tròn $B(f) = 1125\ln(1+\frac{f}{700})$}



Hình 5 Các bộ lọc tam giác để tính năng lượng trên mỗi dải tần số

e) Wrapping và biến đổi DCT

Để tính được M hệ số MFCC, thang Mel được chia thành M dải, mỗi dải có độ rộng $B_{max}(f)/M$. Dựa vào các dải này ta xây dựng M bộ lọc tam giác H_m. Từ đó tính ra M giá trị năng lượng:

$$S_m = \log\left[\sum_{k=1}^{N} X^2(k) H_m(k)\right]$$ Với $m \in \{1, 2, ..., M\}$

Sau đó thực hiện phép biến đổi cosin rời rạc DCT (Discrete Cosine Transformation) ta sẽ thu được các hệ số MFCC:

MFCC = DCT (S_m)

Biến đổi cosin rời rạc:

$$C(u) = \alpha(u) \sum_{x=0}^{N-1} f(x) \cos\left[u\left(x + \frac{1}{2} \right) \frac{\pi}{N} \right] \qquad \text{với u = 0,1,2,…,N-1}$$

Biến đổi ngược:

$$f(x) = \sum_{u=0}^{N-1} \alpha(u)\, C(u) \cos\left[u\left(x + \frac{1}{2} \right) \frac{\pi}{N} \right] \qquad \text{với x = 0,1,2…,N-1}$$

$$\text{Với } \alpha(u) = \begin{cases} \sqrt{\frac{1}{N}} & \text{Với u = 0} \\ \sqrt{\frac{2}{N}} & \text{Với u} \neq 0 \end{cases}$$

Iç. Một số vấn đề khác

1. Vấn đề xác định điểm đầu và điểm cuối của tín hiệu (speech detection)

Mục đích của việc xác định tín hiệu là để tách biệt các đoạn tín hiệu tiếng nói cần quan tâm với các phần khác của tín hiệu (môi trường, nhiễu …). Điều này là rất cần thiết trong nhiều lĩnh vực. Đối với việc tự động nhận dạng tiếng nói, speech detection là cần thiết để tách riêng đoạn tín hiệu là tiếng nói từ đó tạo ra các mẫu (pattern) phục vụ cho việc nhận dạng.

Câu hỏi đặt ra ở đây là làm sao để xác định chính xác tín hiệu tiếng nói, từ đó cung cấp mẫu "tốt nhất" cho việc nhận dạng. Trong trường hợp tín hiệu được thu trong điều kiện môi trường gần lí tưởng (gần như không có nhiễu) thì việc xác định chính xác tiếng nói là vấn đề không khó. Tuy nhiên, thong thường trong thực tế, một vài vấn đề nãy sinh sẽ gây khó khăn cho việc xác định chính xác. Một trong những vấn đề điển hình nhất là cách phát âm của người nói. Ví dụ, khi phát âm, người nói thường tạo ra các âm thanh nhân tạo như tiếng chép môi, hơi thở hoặc là tiếng lách tách trong miệng.

Yếu tố thứ 2 làm cho việc xác định tiếng nói trở nên khó khăn là điều kiện môi trường mà tiếng nói được tạo ra. Một môi trường lí tưởng với nhiễu và tạp âm gần như không có là không thực tế, do vậy bắt buộc phải xem xét việc phát ra tiếng nói trong môi trường có nhiễu (như tiếng máy móc, quạt, tiếng xì xào của những người xung quanh), thậm chí còn trong cả trường hợp môi trường xung quanh không ổn định (tiếng sập cửa, tiếng xe cộ...)

Yếu tố cuối cùng trong việc làm giảm chất lượng tín hiệu là sự mất mát trong hệ thống truyền tín hiệu, như là chất lượng của kênh thông tin, hay mất mất do sự module hoá (lượng tử hoá, số hoá)

Speech detection thực sự quan trọng đối với phương pháp nhận dạng dựa trên so sánh mẫu (pattern comparison), và cũng nâng cao chất lượng của mẫu đối với phương pháp HMM hay mạng Neuron. Tuy nhiên trong nội dung đồ án do chỉ tập trung vào HMM và mạng Neuron nên không đi sâu vào việc xác định tín hiệu, tín hiệu tiếng nói được xác định ở ngưỡng 5%

2. Lượng tử hoá Vector

2.1. Khái niệm

Việc phân tích đặc trưng của tín hiệu tiếng nói bằng phương pháp LPC hay MFCC đều cho

ta các vector đặc trưng k chiều kí hiệu là $v^{m,}$ m = 1,2,…M

Ý tưởng đặt ra ở đây là sử dụng duy nhất một vector để biểu diễn cho một đơn vị các vector đặc trưng. Giảm số lượng các vector đặc trưng biểu diễn tín hiệu về một số lượng hữu hạn cá vector "duy nhất" mà mỗi cái đại diện cho một đơn vị cơ bản của tín hiệu. Và từ đó đưa ra khái niệm về một tập các vector lượng tử (codevector) hay còn gọi là codebook. Phương pháp này gọi là lượng tử hoá vector (vector quantization) VQ

Những điểm mạnh của lượng tử hoá vector :

- Giảm dung lượng lưu trữ thông tin phân tích phổ.
- Giảm lượng tính toán.
- Đưa ra một biểu diễn rời rạc cho tín hiệu âm tiếng nói. Bằng việc đưa ra một codebook "tốt", ta có thể gán cho mỗi codevector một nhãn ngữ âm ứng với một frame tín hiệu, và cho ta một hệ nhận dạng tiếng nói khá hiệu quả.

Tuy nhiên, nó vẫn có một số hạn chế như:

- Mất mát thông tin trong việc phục hồi vector gốc.
- Dung lượng lưu trữ không phải lúc nào cũng là nhỏ.

2.2. Một số yếu tố cần quan tâm

1. Một tập đủ lớn cá vector phân tích, còn gọi là tập huấn luyện.
2. Một chuẩn để đo sự tương đồng, hay khoảng cách giữa các cặp vector
3. Thủ tục tính vector nhân.
4. Thủ tục gán một vector bất kì với một codevector.

2.3. Qui trình thực hiện lượng tử hoá vector

Thực hiện dựa trên thuật toán K-means cluster.

1. Khởi tạo: Chọn M vector khởi tạo của codebook
2. Tìm vector gần nhất: với mỗi vector phân tích, tìm codevector trong codebook hiện tại

gần nhất và gán nó cho vector đó

3. Cập nhật nhân: Cập nhật các codevector sử dụng các vector huấn luyện đc gán cho nó.
4. Lặp: lặp lại bước 2 và bước 3 đến khi đạt được khoảng cách trung bình mong muốn.

Trong thực tế, khi thực hiện ta sử dụng thuật toán LGB:

1. Cho một bộ T gồm M vector đặc trưng và một số $\varepsilon > 0$ đủ "nhỏ"

$x_m \in \{x_1, x_2, ...x_M\}$ Với x^m là vector k chiều

N là tập các codevector. $c_n \in \{c_1, c_2, ...c_N\}$

2. N = 1

$$c_1^* = \frac{1}{M}\sum_{m=1}^{M} x_m$$

Tính $D_{tb}^* = \frac{1}{Mk}\sum_{m=1}^{M}\left\|x_m - c_1^*\right\|^2$

3. Chia đôi.

Với i = 1,2…,N

$$c_i^{(0)} = (1+\varepsilon)\, c_i^*$$

$$c_{N+i}^{(0)} = (1-\varepsilon)\, c_i^*$$

Đặt N = 2N

4. Lặp

Đặt $D_{tb}^{(o)} = D_{tb}^*$ Khởi tạo chỉ số lặp i = 0

i. Với mỗi m = 1,2,…,M Tìm min $\|x_m - c_n\|^2$ với n = 1,2,…,N

Giả sử n^* là chỉ số thoả mãn, gán $Q(x_m) = c_{n^*}^{(i)}$

ii. Với n = 1,2,…,N cập nhật các codevector

Nguyễn Việt Hưng
KS CLC – THCN – K48

$$c_n^{(i+1)} = \frac{\sum_{(Q(x_m)=c_n^i)} x_m}{\sum_{(Q(x_m)=c_n^i)} 1}$$

iii. Đặt i = i + 1

iv. Tính $D_{tb}^{(i)} = \frac{1}{Mk}\sum_{m=1}^{M}\left\|x_m - Q(x_m)\right\|^2$

v. Nếu $\frac{\left(D_{tb}^{(i-1)} - D_{tb}^{(i)}\right)}{D_{tb}^{(i-1)}} > \varepsilon$ thì quay lại bước (i)

vi. Đặt $D_{tb}^{*} = D_{tb}^{(i)}$

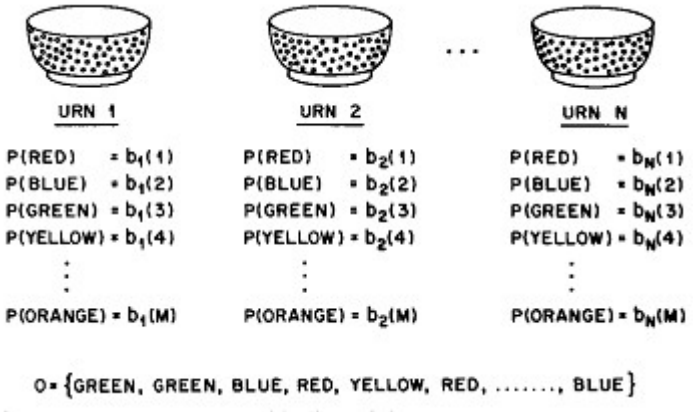
Và $c_n^{*} = c_n^{(i)}$ Với n = 1,2,…,N là tập codevector đạt được

5. Lặp lại bước 3 và 4 cho đến khi đạt được số lượng codevector mong muốn, ta có codebook các vector lượng tử.

CHƯƠNG II: CÁC THUẬT TOÁN VÀ MÔ HÌNH NHẬN DẠNG TIẾNG NÓI

1. Mô hình Markov ẩn (HMM)

Ví dụ điển hình về mô hình markov mô hình bình và bóng màu:



Hình 6 Mô hình bóng màu

Có N bình, mỗi bình có M quả bóng màu khác nhau

Xác suất lấy được bóng màu trong mỗi bình được xác định P(màu) = $b^i(j)$; i = 1,2,…,N; j = 1,…,N

Đồng thời cho trước ma trận xác xuất chuyển trạng thái giữa các bình

Mỗi trạng thái quan sát ứng với 1 bình được chọn và màu bóng được lấy ra.

2. Các thành phần của HMM

1. N: số trạng thái
   Tập các trạng thái S = (S_1, S_2, …, S_N)
   Trạng thái quan sát được tại thời điểm t là q_t
2. M: Số hiện tượng quan sát được trong từng trạng thái và cũng là output của hệ
   V = {V_1, V_2, …,V_M}
3. Xác suất chuyển trạng thái A = {a_{ij}}
   $a_{ij} = P\left[q_{t+1} = S_j \,|\, q_t = S_i\right] \qquad 1 \le i, j \le N$

   - $a_{ij} > 0 \forall i, j$ S_j có thể đến được từ mọi S_i
   - $a_{ij} = 0$ với một số (i,j) một số trạng thái không thể liền sau nhau.

4. Xác suất quan sát được hiện tượng tại trạng thái j
   B = {$b_j(k)$}
   $bj(k) = P\left[v_k - t \,|\, q_t = S_i\right]$

Nguyễn Việt Hưng
KS CLC – THCN – K48

$1 \le j \le N \quad 1 \le k \le M$

5. Trạng thái khởi tạo $\Pi = \{\pi_i\}$

$\pi_t = P[q_1 = S_i]$ với $1 \le i \le N$

6. Chuỗi kết quả quan sát $O = O_1O_2\ldots O_T$

O_t : 1 hiện tượng của V

T : Số trạng thái quan sát.

Một mô hình HMM được kí hiệu $\lambda = (A, B, \pi)$

3. 3 vấn đề của HMM

1. Cho chuỗi quan sát $O = O^1O^2\ldots O^T$, và mô hình $\lambda = (A,B,\pi)$, làm sao ta có thể tính một cách hiệu quả xác suất $P(O|\lambda)$?
2. Cho chuỗi quan sát $O = O^1O^2\ldots O^T$, và mô hình λ , làm sao để chọn được chuỗi $Q = q^1q^2\ldots q^T$ tối ưu theo nghĩa xác suất.
3. Làm sao để thay đổi các tham số của $\lambda = (A,B,\pi)$ để xác suất $P(O|\lambda)$ đạt cực đại ?

4. Áp dụng vào bài toán nhận dạng các từ riêng biệt

Với mỗi từ, thiết kế một hệ N-trạng thái HMM riêng rẽ. Tín hiệu của từ được biểu diễn theo vecto phổ tần.

Giả sử việc mã hoá được thực hiện với việc sử dụng một codebook với M vecto phổ duy nhất. Do đó mỗi quan sát sẽ là chỉ số của vecto phổ gần nhất với tín hiệu tiếng nói ban đầu.

Bởi vậy, với mỗi từ phải huấn luyện, tức là sử dụng các mẫu khác nhau của từ đó (do một hoặc nhiều người phát âm) để tối ưu các tham số.

Có 3 việc chính cần làm trong việc xây dựng một bộ nhận dạng dựa trên HMM

1. Xây dựng một mô hình cho các từ riêng rẽ, sử dụng vấn đề 3 để tối ưu hoá các tham số cho mỗi mô hình tương ứng với từ đó.
2. Dùng vấn đề 2 để phân đoạn cá kết quả huấn luyện vào các trạng thái, và từ đó nghiên

cứu đặc tính của các vector phổ sinh ra các quan sát xuất hiện trong mỗi trạng thái. Mục đích ở đây là huấn luyện mô hình (thêm trạng thái, thay đổi kich thước codebook…) từ đó tăng khả năng mô hình hoá của chuỗi từ được nói.

3. Cuối cùng, việc nhận dạng từ được biểu diễn bằng cách sử dụng vấn đề 1. Từ được chọn làm kết quả nhận dạng là từ có kết quả tính được cao nhất.

5. Giải quyết 3 vấn đề của HMM

Như đã trình bày ở trên, giải quyết 3 vấn đề của HMM cũng chính là tìm ra thuật toán đề thực hiện một hệ nhận dạng dựa trên HMM.

Vấn đề 1:

Các giải pháp được đưa ra.

- Tính trực tiếp P(O|λ)

  $$P(O|\lambda) = \sum_{allQ} P(O|Q,\lambda).P(Q|\lambda)$$

  Nhưng theo cách này khối lượng tính toán ($2TN^T$ phép tính) là quá lớn nên hoàn toàn không khả thi.

- Dùng thủ tục Forward – Backward

Sử dụng biến forward $\alpha^t(i)$ và thuật toán qui nạp

$$\alpha_t(i) = P(O_1O_2...O_t, q_t = S_i | \lambda)$$

là xác suất của một đoạn chuỗi quan sát $O^1O^2…O^t$, và trạng thái S^i tại thời điểm t, với mô hình λ cho trước.

Giải theo thuật toán qui nạp:

i. Khởi tạo : $\alpha^1(i) = \pi^i b^i(O^1)$ $\quad 1 \le i \le N$

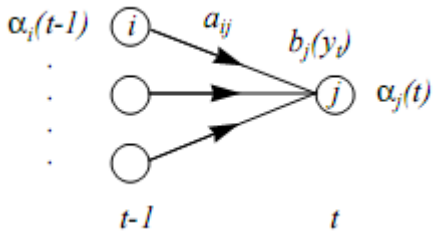
ii. Lặp:

$$\alpha_{t+1}(j) = \left[\sum_{j=1}^{N} \alpha_t(i)\, a_{ij}\right] b_j(O_{t+1}) \qquad 1 \le t \le T-1$$

$$1 \le j \le N$$

iii. Kết thúc:

$$P(O \mid \lambda) = \sum_{i=1}^{N} \alpha_T(i)$$



Hình 7hàm forward

Số lượng phép tính sử dụng là N^2T phép tính

- Tương tự ta có biến backward

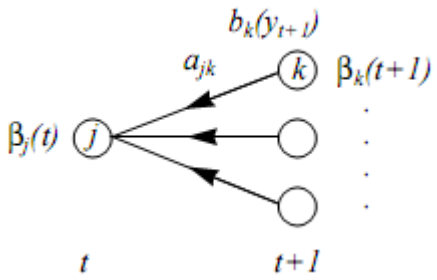
$\beta^t(i) = P(O^{t+1}O^{t+2}\ldots O^T | q^t = S^i, \lambda)$

i. Khởi tạo : $\beta^T(i) = 1$ $\quad$ với $1 \le i \le N$

ii. Qui nạp

$$\beta_t(i) = \sum_{j=1}^{N} a_{ij} b_i(O_{t+1}) \beta_{t+1}(j)$$

với t = T – 1, T – 2, …, 1, 1 ≤ i ≤ N



Hình 8 hàm backward

Số lượng phép tính sử dụng là $N^2 T$

Vấn đề 2:

Một kĩ thuật phổ biến được sử dụng để tìm ra chuỗi tốt nhất gần với chuỗi quan sát, dựa trên phương thức lập trình động, được gọi là thuật toán Viterbi.

Thuật toán Viterbi: Tìm ra chuỗi duy nhất Q = $\{q^1 q^2 \ldots q^T\}$ gần nhất với chuỗi quan sát

O = $\{O^1 O^2 \ldots O^T\}$

Ta định nghĩa $\delta_t(i) = \max_{q_1, q_2 \ldots, q_{t-1}} P[q_1 q_2 \ldots q_t = i, O_1 O_2 \ldots O_t \mid \lambda]$

Đệ qui:

$$\delta_{t+1}(j) = \left[\max_i \delta_t(i)\, a_{ij}\right] . b_j(O_{t+1})$$

Khởi tạo:

$$\delta_1(i) = \pi_i b_i(O_1) \qquad 1 \le i \le N$$

$$\psi^1(i) = 0$$

Hồi qui:

$$\delta_t(j) = \max_{1 \le i \le N} \left[\delta_{t-1}(i) . a_{ij}\right] . b_j(O_t) \qquad 2 \le t \le T$$

$1 \le j \le N$

$$\psi_t = \arg\max_{1 \le i \le N} \left[\delta_{t-1}(i)\, a_{ij} \right] \qquad 2 \le t \le T$$

$1 \le j \le N$

ψ^i là track argument

Kết thúc:

$$P^* = \max_{1 \le i \le N} \left[\delta_T(i) \right]$$

$$q_t^* = \arg\max_{1 \le i \le N} \left[\delta_T(i) \right]$$

Path (Chuỗi trạng thái) back tracking:

$$q_t^* = \psi_{t+1}\left(q_{t+1}^* \right) \qquad t = T-1,\ T-2,\ldots,1$$

Vấn đề 3:

Vấn đề thứ 3 và cũng là vấn đề khó nhất trong phương pháp HMM là tìm ra một phương thức để thay đổi các tham số (A,B,π) để xác suất của chuỗi quan sát ứng với mô hình cho trước là cực đại. Ở đây ta chỉ sử dụng phương pháp lặp Baum-Welch (hay tương đương với phương pháp EM-expectation modification), hoặc là sử dụng đạo hàm riêng để tìm cực trị địa phương.

Một số biến cần định nghĩa :

- Là xác suất của việc ứng với chuỗi quan sát O, và mô hình λ, trạng thái ở thời điểm t là S^i

$$\gamma_t(i) = P(q_t = S_i \mid O, \lambda)$$

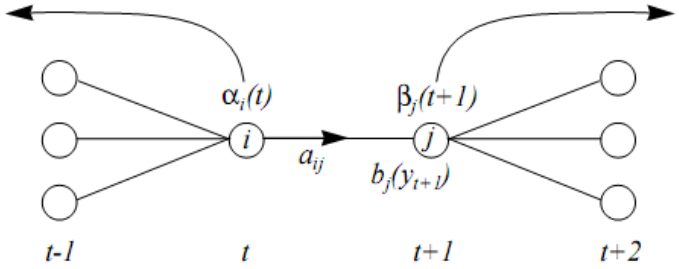
$$\gamma_t(i) = \frac{\alpha_t(i)\, \beta_t(i)}{\sum_{i=1}^{N} \alpha_t(i)\, \beta_t(i)}$$

điều kiện xác suất $\sum_{i=1}^{N}\gamma_t(i)=1$

- Xác suất của sự kiến ứng với chuỗi quan sát và mô hình cho trước, thì tại thời điểm t là ở trạng thái S^i và ở thời điểm t+1 là trạng thái S^j.

$$\xi_t(i,j)=P\left(q_t=S_i,q_{t+1}=S_j\mid O,\lambda\right)$$

$$\xi_t(i,j)=\frac{\alpha_t(i)\,a_{ij}b_j(O_{t+1})\,\beta_{t+1}(j)}{P(O\mid\lambda)}=\frac{\alpha_t(i)\,a_{ij}b_j(O_{t+1})\,\beta_{t+1}(j)}{\sum_{i=1}^{N}\sum_{j=1}^{N}\alpha_t a_{ij}b_j(O_{t+1})\,\beta_{t+1}(j)}$$



Hình 9Biến Forward-backward

- Liên hệ giữa 2 biến: $\gamma_t(i)=\sum_{j=1}^{N}\xi_t(j,j)$

$\sum_{t=1}^{T-1}\gamma_t(i)=$ Số lần chuyển vị từ S^i (số lần trạng thái S^i được "viếng thăm")

$\sum_{t=1}^{T-1}\xi_t(i,j)=$ Số lần chuyển vị từ S^i tới S^j

Dựa vào các công thức trên ta đưa ra phương pháp để ước lượng các tham số của một mô hình HMM.

$\bar{\pi}_i=$ xác suất ở trạng thái S^i tại thời điểm (t=1) $=\gamma_1(i)$

$$\bar{a}_{ij}=\frac{\sum_{t=1}^{T-1}\xi(i,j)}{\sum_{i=1}^{T-1}\gamma_t(i)}$$

$$\bar{b}_j(k) = \frac{\text{expected number of time in state j and observing symbol } v_k}{\text{expected number of times in state j}}$$

$$= \frac{\sum_{t=1}^{T} \gamma_t(i)}{\sum_{t=1}^{T} \gamma_t(j)}_{O_t = v_k}$$

Ta định nghĩa mô hình hiện tại λ = (A,B,λ) , và mô hình ước lượng lại $\bar{\lambda} = (\bar{A}, \bar{B}, \bar{\pi})$

Theo Baum thì mô hình $\bar{\lambda}$ "tốt" hơn mô hình λ theo nghĩa P(O|$\bar{\lambda}$) > P(O|λ) hay nói cá khác ta xác định một mô hình mới mà xác suất tạo ra chuỗi quan sát được cao hơn.

Dựa vào các thủ tục trên, và kết hợp với vòng lặp, ta nâng cao được xác suất của chuỗi O quan sát được từ mô hình cho đến khi đạt đến điểm giới hạn. Ước lượng cuối cùng được gọi là ước lượng xác suất cực đại của HMM.

Cũng cần nói rằng, thuật toán forward-bachward chỉ cho ta các cực trị địa phương.

Một số chú ý:

Các ràng buộc xác suất:

$$\sum_{i=1}^{N} \bar{\pi}_i = 1$$

$$\sum_{j=1}^{N} \bar{a}_{ij} = 1 \qquad 1 \le i \le N$$

$$\sum_{k=1}^{M} \bar{b}_j(k) = 1 \qquad 1 \le j \le N$$

Các ràng buộc này đều tự động được thoả mãn trong mỗi vòng lặp.

6. So sánh 2 mô hình HMM

Hai mô hình HMM: $\lambda^1 = (A^1, B^1, \pi^1)$ và $\lambda^2 = (A^2, B^2, \pi^2)$

Khái niệm tương đương giữa 2 mô hình:

$E[O^t = v^k|\lambda^1] = E[O^t = v^k|\lambda^2]$

Trong thực tế, 2 mô hình có thể trông rất khác nhau nhưng lại tương đương nhau. Bởi vậy cần có khái niệm khoảng cách của 2 mô hình $D(\lambda^1,\lambda^2)$

$$D(\lambda_1,\lambda_2) = \frac{1}{T}\left[\log P\left(O^{(2)} \mid \lambda_1\right) - \log P\left(O^{(2)} \mid \lambda_2\right)\right]$$

Trong đó $O^{(2)}$ là chuỗi quan sát tạo bởi mô hình λ^2.

Công thức trên có ý nghĩa đo xem mô hình λ^1 "match" với các quan sát tạo bởi λ^2 như thế nào.

Tương tự ta có $D(\lambda^2,\lambda^1)$

Và khoảng cách đối xứng giữa 2 mô hình sẽ là: $D_s(\lambda_1,\lambda_2) = \frac{D(\lambda_1,\lambda_2) + D(\lambda_2,\lambda_1)}{2}$

7. Các cấu trúc mô hình HMM và lựa chọn mô hình cho bài toán

Trong khuôn khổ đồ án, mô hình HMM được sử dụng là DHMM (mô hình HMM rời rạc).

Có 2 loại cấu trúc chính của HMM như :

- Mô hình ergodic: Mọi trạng thái của mô hình đều có thể đến được chỉ bằng một bước. (Hoặc rộng hơn: là mọi trạng thái của mô hình đều đến được trong vòng một số hữu hạn bước). Do vậy đặc điểm của mô hình này là mọi phần tử của ma trận chuyển vị trạng thái a^{ij} đều là số dương.
- Trong các ứng dụng thực tế, người ta thường sử dụng mô hình trái-phải hay là mô hình Bakis. Trong đó các trạng thái chỉ xuất hiện theo chiều từ trái sang phải.

  Đặc điểm cơ bản của mô hình này :

  $a^{ij} = 0, \quad j < i$

  và $\pi_i = \begin{cases} 0, & i \neq 1 \\ 1, & i = 1 \end{cases}$ Do vậy chuỗi trạng thái bắt buộc phải bắt đầu từ trạng thái 1 và kết thúc ở trạng thái N

Mở rộng hơn ta có thể giới hạn $a^{ij} = 0, \quad j > i + \Delta$

$$\text{Và } a^{NN} = 1$$

$$A^{Ni} = 0, \ i < N$$

Ví dụ:

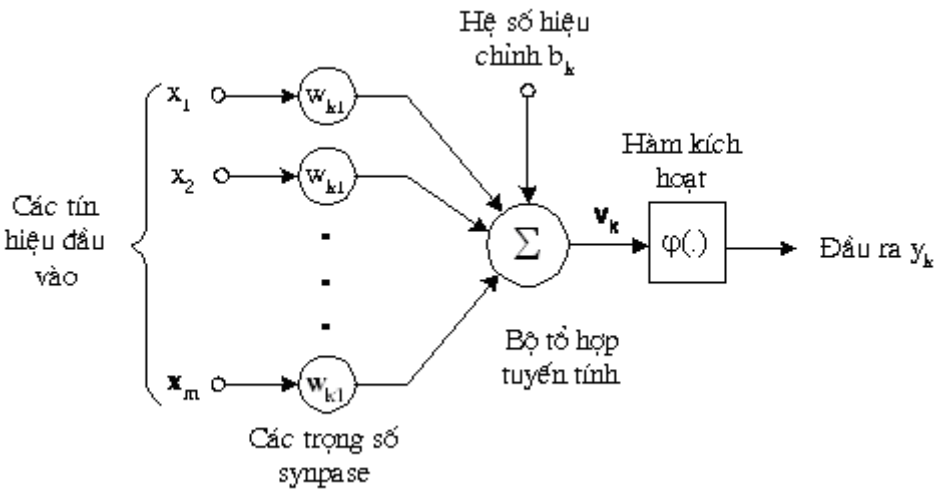
$$A = \begin{bmatrix} a_{11} & a_{12} & a_{13} & 0 \\ 0 & a_{22} & a_{23} & a_{24} \\ 0 & 0 & a_{33} & a_{34} \\ 0 & 0 & 0 & a_{44} \end{bmatrix}$$

Các đặc điểm này rất thích hợp cho việc mô hình hoá tín hiệu thay đổi theo thời gian như tín hiệu âm thanh. Do vậy, ta sẽ chọn mô hình này cho việc giải quyết bài toán.

8. Mô hình sử dụng mạng Neuron

1. Một số khái niệm cơ bản về mạng Neuron

Một mạng Neuron là một tập hợp liên kết lẫn nhau của các phần tử tính toán phí tuyến đơn giản.



Hình 10 Mô hình phi tuyến của một mạng Neuron

Các thành phần chính của một mạng neuron:

- Các kết nối, hay các trọng số của mỗi đầu vào w_i
- Bộ cộng: Tính tổng các tín hiệu đầu vào của neuron đã được nhân trọng số tương ứng. Từ đó tạo nên một bộ tổ hợp tuyến tính.
- Hàm kích hoạt (active function) để giới hạn biên độ đầu ra của Neuron.
- Ngoài ra còn có thể có các hệ số hiệu chỉnh b_k có tác dụng tăng giảm đầu vào thực của hàm kích hoạt

Với N đầu vào $x_1,x_2,\ldots, x_N$ được gán các trọng số $w_1,w_2,\ldots,w_N$. Kết quả output y sẽ được tính:

$$y = f\left(\sum_{i=1}^{N} w_i x_i - \phi\right)$$ Trong đó ϕ là ngưỡng nội hay offset

Một số hàm kích hoạt hay được dùng

1) Hàm ngưỡng:

$$f(x) = \begin{cases} +1, & x \geq 0 \\ -1, & x < 0 \end{cases}$$

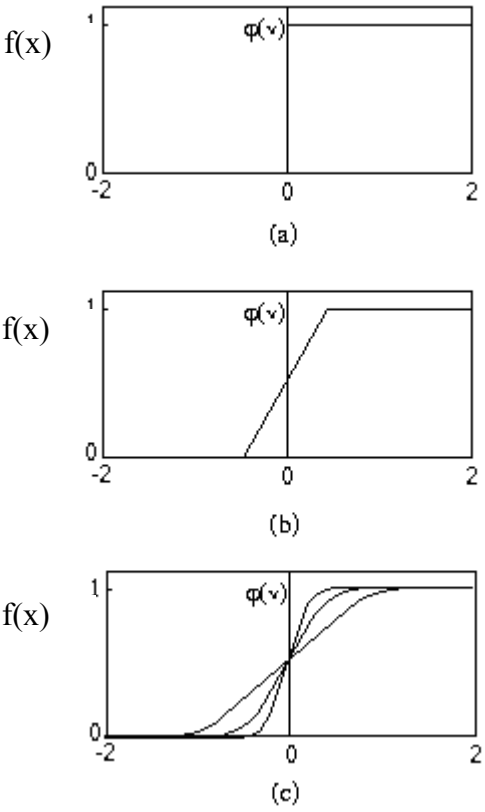
2) Hàm tuyến tính vùng

$$f(x) = \begin{cases} 1, & x \geq +\frac{1}{2} \\ x, & +\frac{1}{2} \geq x > -\frac{1}{2} \\ 0, & x \leq -\frac{1}{2} \end{cases}$$

3) Hàm signmoid:

$$f(x) = \tanh(\beta x), \quad \beta > 0$$

hay $$f(x) = \frac{1}{1+e^{-\beta x}} \quad \beta > 0$$



Hình 11a) Hàm ngưỡng b) Hàm tuyến tính c)Hàm sigmoid

2. Kiến trúc mạng Neuron

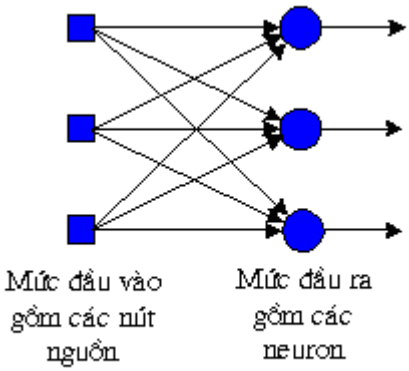
Có 3 kiểu mạng Neuron phổ biến là:

- Mạng tiến (Feed forward) đơn hoặc đa lớp
- Mạng hồi qui Hopfield
- Mạng tự tổ chức (self-orgnizing) hay mạng Kohonen

Mạng feed forward

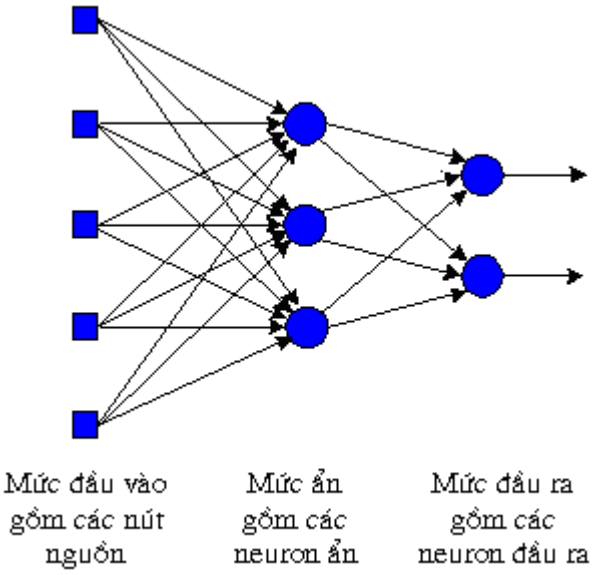
- Một lớp (single perceptron):

Là dạng đơn giản nhất, có một lớp đầu vào và các nút nguồn đều dẫn trực tiếp đến lớp đầu ra.



Hình 12 Cấu trúc mạng Neuron 1 mức

- Đa lớp (MLP: multilayer perceptrons):
  Dạng thường được sử dụng nhất là mạng 3 lớp gồm
  - Lớp đầu vào (Input layer)
  - Lớp đầu ra (Output layer)
  - Lớp ẩn (Hidden Layer)

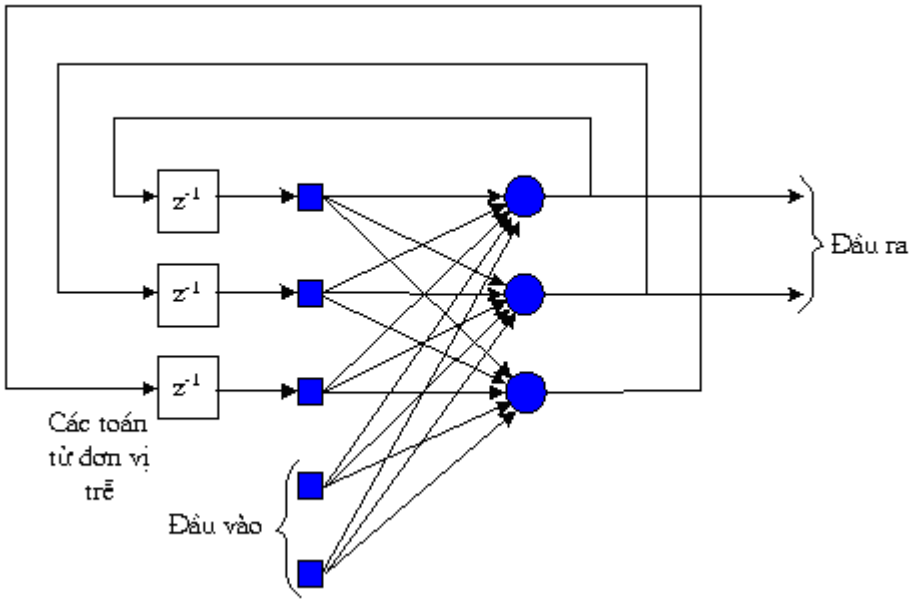


Hình 13 Mạng Neuron đa lớp

Định lý Komogorov: Chỉ cần tối đa 2 lớp ẩn để xây dựng mạng MLP xấp xỉ hàm phi tuyến bất kì với độ chính xác cho trước. [6]

Mạng hồi qui Hopfield

Là mạng có ít nhất một vòng lặp phản hồi. Như vậy đầu vào cho mỗi phần tử tính toán sẽ gồm cả các biến vào và biến ra.



Hình 14 Mạng hồi qui Hopfield

Công thức tính đầu ra cho vòng lặp thứ i sẽ là:

$$y_i(t) = f[x_i(t) + \sum_{j} w_{ij} y_j(t-1) - \phi]$$

Mạng Self-organizing Kohonen

Là kiến trúc ít được sử dụng hơn 2 kiến trúc trên, kiến trúc Kohonen là một qui trình tập hợp nhóm để đưa ra một codebook của các mẫu ổn định trong không gian biến vào để đưa ra đặc điểm của một vector đầu vào. Việc này liên quan nhiều đến qui trình lượng tử Vector (VQ) đã được trình bày ở phần trước.

3. Những điểm mạnh của kiến trúc mạng Neuron

- Có thể thực hiện một khối lượng tính toán song song lớn. Nhờ cấu trúc của các neuron là đơn giản, thuận lợi cho tính toán và hoàn toàn độc lập.
- Các nút trong cùng 1 lớp không ảnh hưởng lẫn nhau, nên mạng ít bị ảnh hưởng bởi sự tác động không tốt của các phần tử riêng lẻ trong mạng.
- Khả năng học, các hệ số đều có thể cải thiện thông qua quá trình học, tạo cho hệ một khả năng thích nghi tốt.
- Một mạng đủ lớn có thể xấp xỉ rất tốt bất kì một hệ động phi tuyến.

4. Qui trình học cho mạng tiến MLP 1 lớp ẩn:

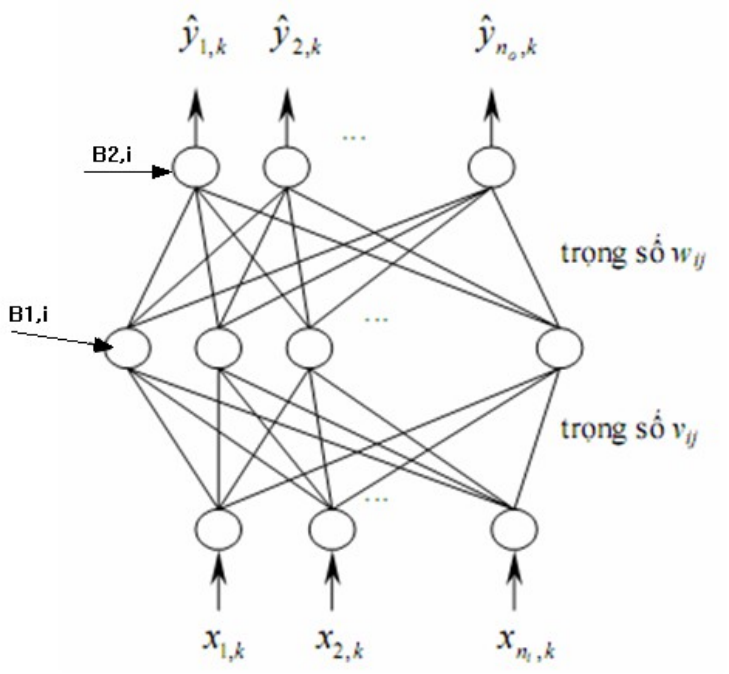
Trong phạm vi đồ án sẽ chỉ nghiên cứu và áp dụng mạng tiến 1 lớp ẩn, do tính đơn giản và khả năng tính toán cũng như áp dụng cao.

Các nút đầu vào x_i (i=1…N)

Các nút ẩn $\hat{h}_j$ (j=1…M)

Các nút ra $\hat{y}_l$ (l=1…K)

Gọi tập dữ liệu mẫu dùng để huấn luyện là (x_k, y_k), x_k là đặc tính phổ của mỗi khung tín hiệu, y_k là đáp ứng đầu ra mong muốn. Tập các trọng số ứng với các nút ẩn là v_{ij} , với các nút đầu ra là w_{ij} .



Hình III-0-1: Quá trình học mạng đa mức

Với mỗi vector đầu vào x_k các nút ẩn được tính toán bằng:

$$\hat{h}_{j,k} = f(\bar{h}_j) = f(\sum_{l=1}^{N} v_{jl}.x_{l,k} + B1_i)$$

Trong đó $x_0 = 0$ và hàm f(x) là hàm phi tuyến sigmoid:

$$f(x) = \frac{1}{1+e^{-\beta x}} \qquad \beta > 0$$

Đáp ứng đầu ra được tính theo công thức:

$$\hat{y}_{j,k} = \varphi(\bar{y}_{j,k}) = \varphi(\sum_{j=1}^{M} w_{ij}\hat{h}_{j,k} + B2_i)$$

Ở đây ta sử dụng : $\varphi(x) = x$

Sai số cho mỗi đầu ra: $\varepsilon_{i,k} = y_{i,k} - \hat{y}_{i,k}$

Việc huấn luyện mạng được thực hiện bằng phương pháp giảm gradient, xác định các trọng số sao cho sai số E đạt tới cực tiểu. Hàm lỗi sử dụng là hàm bình phương:

$$E_k = \frac{1}{2}\sum_{i=1}^{K}(y_{i,k} - \hat{y}_{i,k})^2$$

Thủ tục huấn luyện truyền ngược sai số (Error Back Propagation) được sử dụng để huấn luyện mạng MLP, các trọng số sẽ được cập nhật theo công thức sau:

$$w_{p,k} = w_{p,k-1} + \alpha\varepsilon^h_{p,k}x_k$$

$$v_{p,k} = v_{p,k-1} + \alpha\varepsilon_{p,k}\hat{h}_k$$

Trong dó α là hệ số học

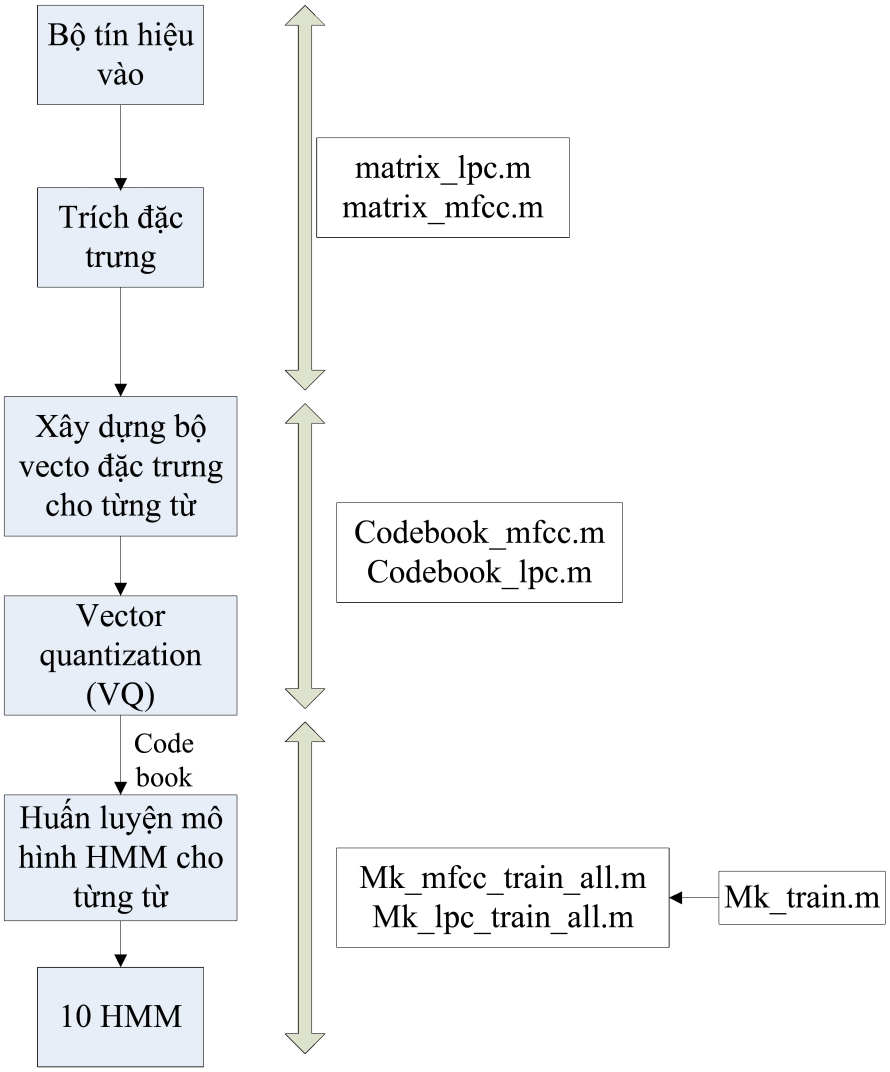
CHƯƠNG III : THỰC HIỆN BÀI TOÁN NHẬN DẠNG

I. Sử dụng mô hình HMM

Mô hình được lựa chọn là mô hình HMM trái phải và sự dụng cả 2 phương pháp trích đặc trưng LPC và MFCC để chạy thử nghiệm.

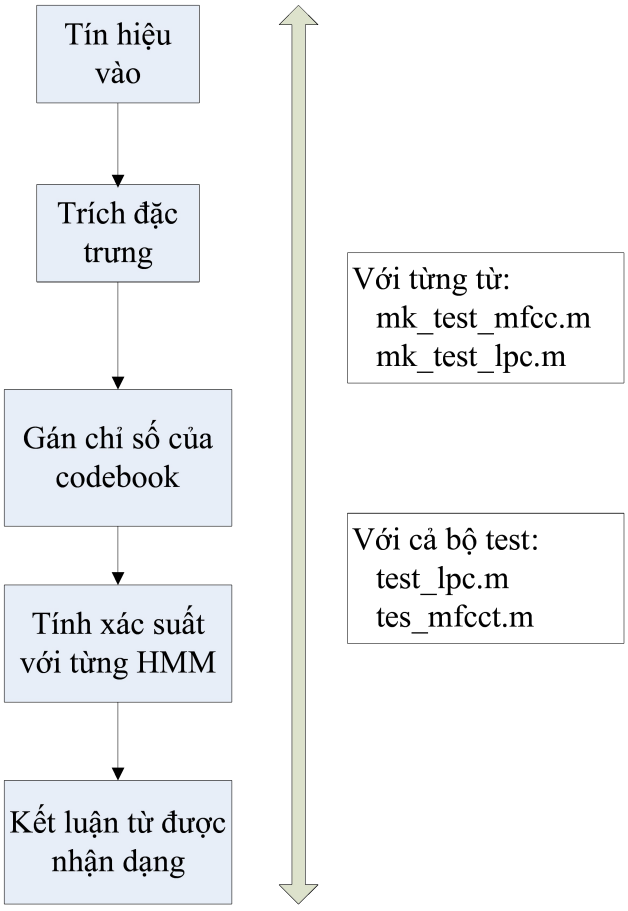
1. Xây dựng thuật toán trên nền công cụ Matlab.

1.1. Quá trình học :



Hình 15 Quá trình họcHMM

1.2. Quá trình kiểm tra :



Hình 16 Quá trình kiểm tra HMM

Với phương pháp MFCC: tín hiệu được chia thành các frame có độ dài N = 512 mẫu với độ chồng lấp M = 100.

Với phương pháp LPC: Các frame có kích thước N = 400; M = 100.

Các hàm chính:

- *codebook_lpc.m* và *codebook_mfcc.m* : Xây dựng code book cho tất cả các vector đặc trưng của tín hiệu được trích bằng phương pháp LPC và MFCC tương ứng. Trong đó có sử dụng các hàm con là *matrix_lpc.m và matrix_mfcc.m* để tạo các ma trận vector đặc trưng cho từng mẫu tín hiệu. Và hàm *vqsplit.m* để tiến hành lượng tử hoá vector
- *mk_lpc_train_all.m* và *mk_mfcc_train_all.m* : Huấn luyện mô hình HMM theo các dữ liệu đã được lượng tử hoá bằng codebook đã tính được từ bước trên, và tạo ra 10 mô hình HMM riêng biệt cho từng từ khoá nhận dạng. Các mô hình này được lưu lại dưới dạng file HMM_(từ khoá).mat
- *mk_test_lpc.m và mk_test_mfcc.m* để kiểm tra mẫu nhận dạng với thủ tục forward để tính xác suất của chuỗi quan sát với mô hình HMM cho trước. Điều kiện ‘viter’ có 2

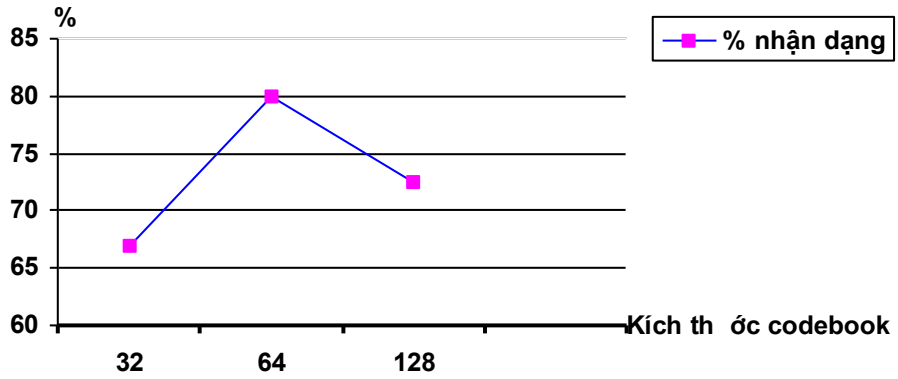
mức 1 và 0 tương ứng với việc hiện kết quả của chuỗi trạng thái tốt nhất ứng với chuỗi quan sát.

2. Chạy thử và kiểm tra kết quả

Các tham số được thay đổi và chạy thử với bộ dữ liệu gồm 20 người: 13 nam và 7 nữ. Các dữ liệu được thu âm bằng micro và máy tính cá nhân với mức nhiễu khá cao.
Mỗi người có 5 mẫu: 3 mẫu cho vào bộ huấn luyện, 2 mẫu cho vào bộ kiểm tra.
Các vector đặc trưng được trích từ MFCC gồm 13 mfcc và 12 delta.
Các vector đặc trưng được trích từ LPC có kích thước tuỳ theo bậc LPC.

⇨ Công việc cần làm là chạy thử kiểm tra để tìm rat ham số tối ưu cho mô hình

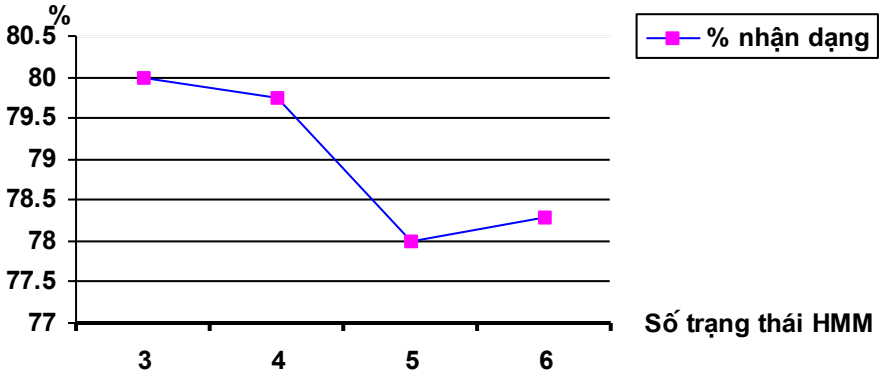
Các kết quả thu được ứng với các tham số:



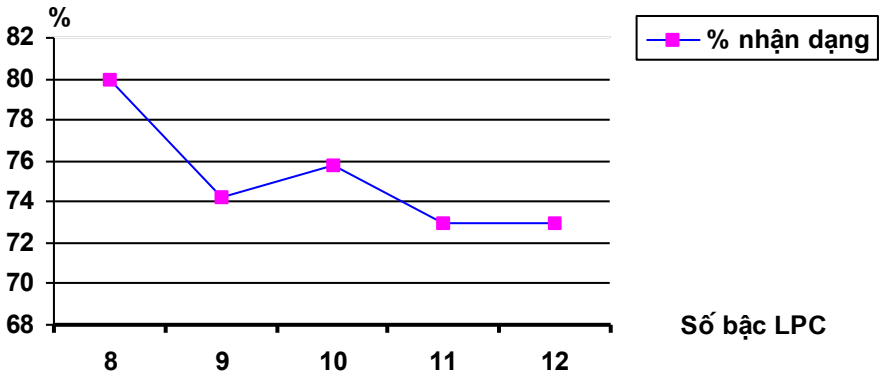
Hình 17 Kết quả theo kích thước codebook

⇨ ***Chọn kích thước codebook là 64***

2.1. Trích đặc trưng theo phương pháp LPC



Hình 18 Kết quả theo số trạng thái HMM

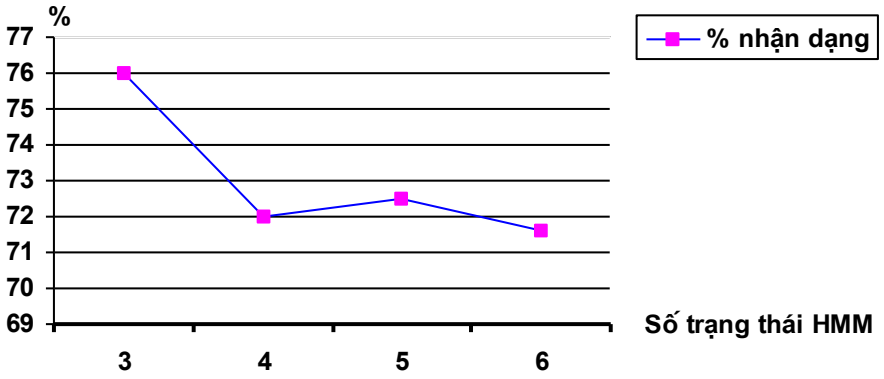


Hình 19 Kết quả theo số bậc LPC

Như vậy kết quả tối ưu nhận được là sử dụng LPC 8 bậc để trích đặc trưng và huấn luyện bằng mô hình HMM 3 trạng thái.

Kết quả nhận dạng đạt 80%

2.2. Trích đặc trưng theo phương pháp MFCC



Hình 20 Kết quả theo số trạng thái HMM

Kết quả nhận dạng đạt 76%

2.3. Nhận xét kết quả

Kết quả tốt nhất đạt được với phương pháp LPC bậc 8, mô hình HMM 3 trạng thái và kích thước codebook 64. Với khả năng nhận dạng trung bình là 80%.

Có thể nói đây là một kết quả chưa tốt bởi bên cạnh một số từ đã nhận dạng khá tốt (80-

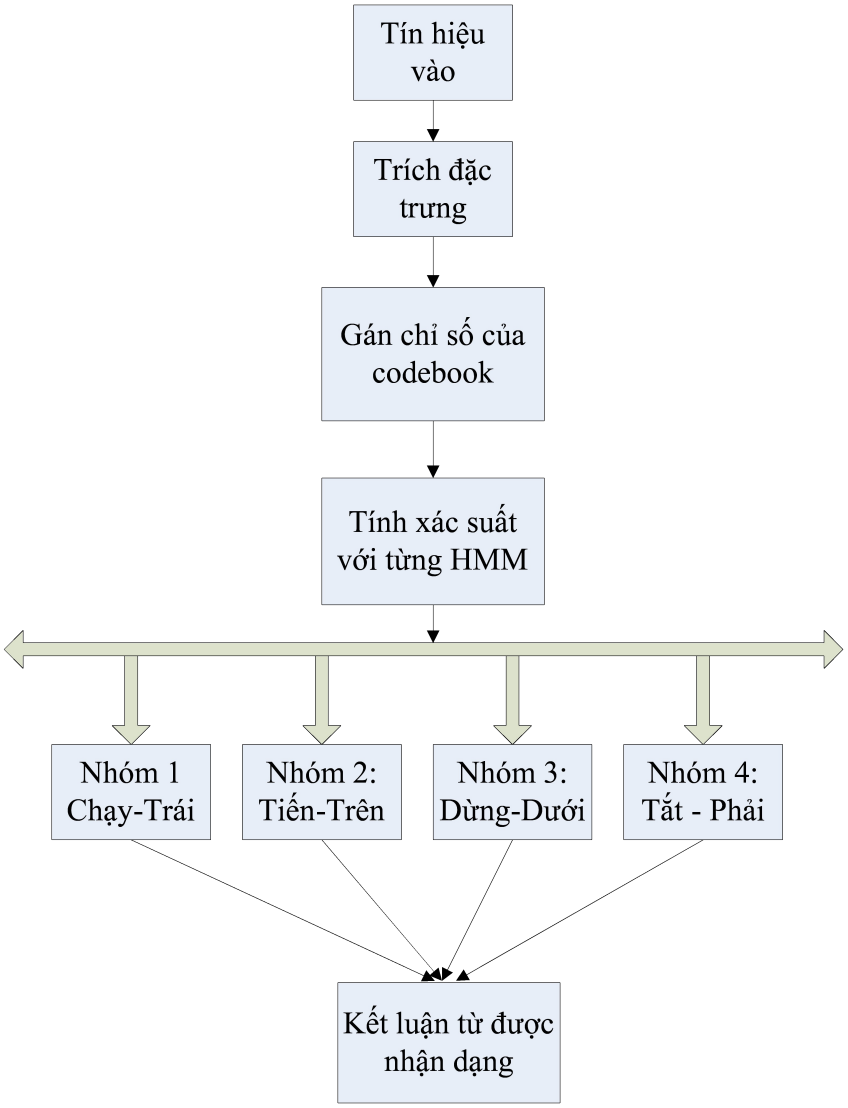
90%) thì những từ khác mô hình lại cho kết quả không cao. Hay nói cách khác là khả năng nhận dạng các từ không đồng đều.

Có các nhóm từ hay bị nhận dạng nhầm với nhau : {dừng, dưới}; {trái, chạy}; (tiến trên); (tắt, phải).

Nguyên nhân :

- Chất lượng của các mẫu dữ liệu không cao (độ nhiễu lớn, và thu từ các môi trường nhiễu khác nhau)
- Các tham số chọn lựa chưa tối ưu
- Một số từ có cách phát âm gần giống nhau.

⇨ Thực hiện test 2 lần để nâng cao kết quả nhận dạng.



Hình 21 Test 2 lần để nâng cao kết quả

Kết quả thu được:

- Với phương pháp MFCC, tỉ lệ nhận dạng được tăng lên 80%

II. Sử dụng mạng Neuron

1. Xây dựng thuật toán trên công cụ Matlab

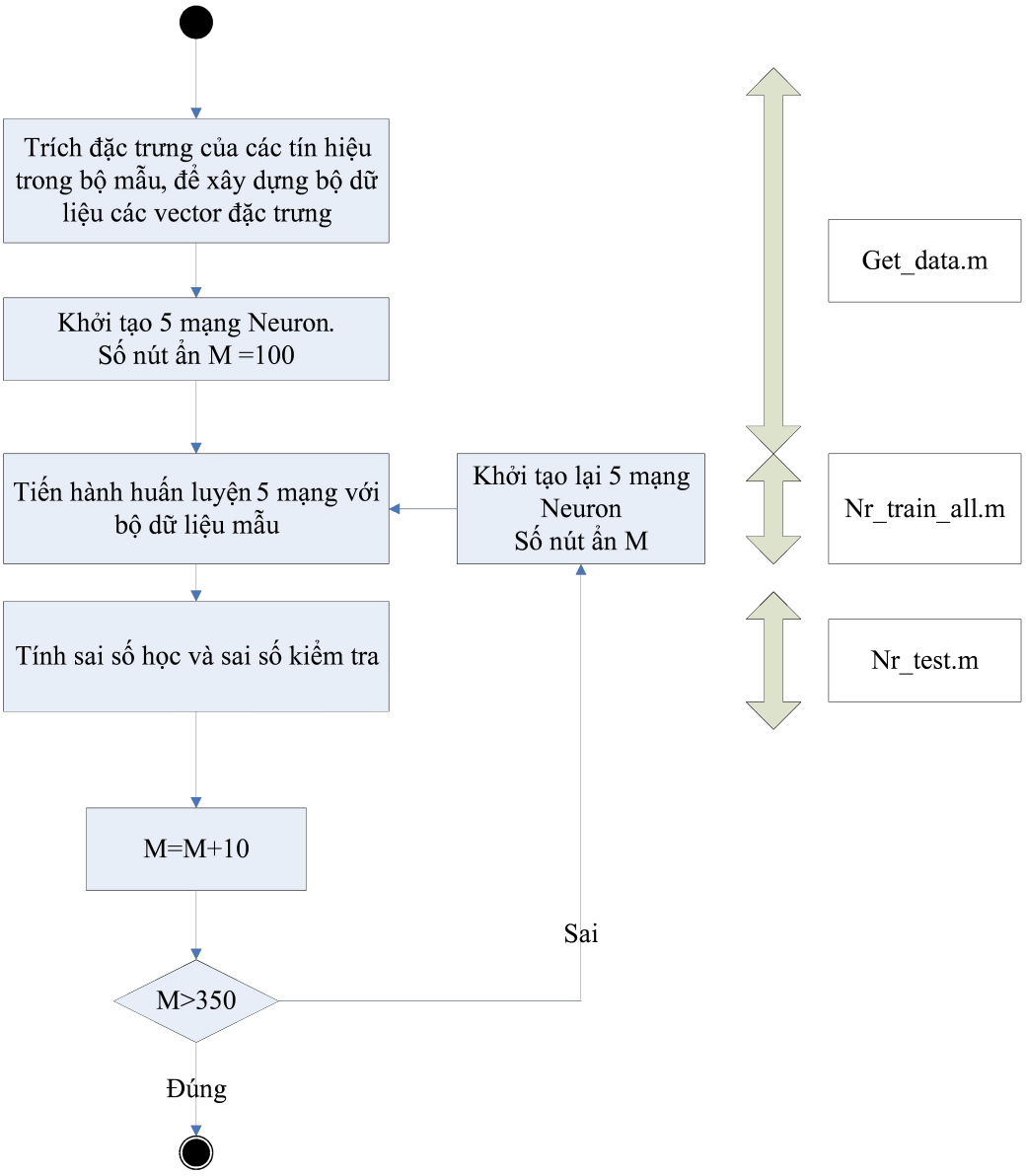
Mạng Neuron được xây dựng với 1 lớp ẩn và hai hàm ẩn tương ứng là hàm logsig và hàm pureline.

Mạng có 10 đầu ra ứng với 10 từ điều khiển.

Các tham số được lựa chọn :

+ Số bậc LPC : 8.
+ Các hệ số MFCC : 13

Việc cần làm, lựa chọn số nút ẩn tối ưu cho mô hình mạng Neuron



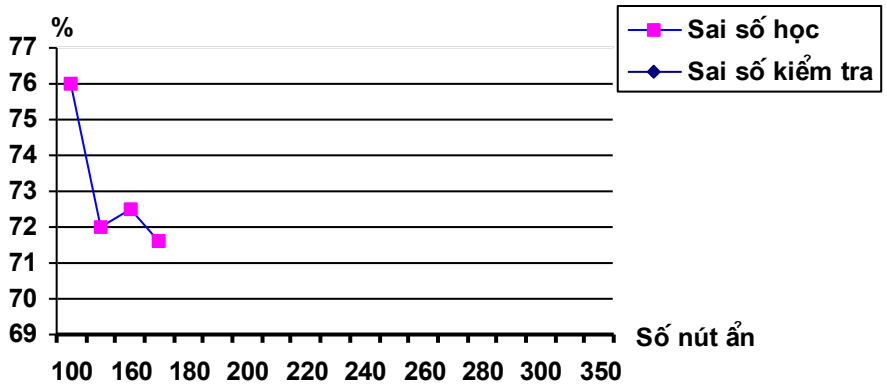
Hình 22 Quá trình tìm tham số tối ưu cho mạng Neuron

Các hàm chính :

- Hàm *get_data.m* gọi các hàm con : data_lpc (hoặc data_mfcc) và hàm matrix_lpc (hoặc matrix_mfcc) để thực hiện trích đặc trưng của tất cả các mẫu dữ liệu trong bộ học và xếp thành ma trận các vector đặc trưng lần lượt tương ứng với các từ điều khiển.
- Hàm *nr_train_all.m* : sẽ đưa dữ liệu vào và huấn luyện cho mạng Neuron. Mạng sau khi huấn luyện sẽ được lưu lại dạng file .mat
- Hàm *nr_test.m* : Sẽ load mạng Neuron từ file .mat rồi lần lượt đưa dữ liệu qua mạng Neuron đã được huấn luyện để kiểm tra.
-

2. Các kết quả thu được ứng với từng phương pháp trích đặc trưng

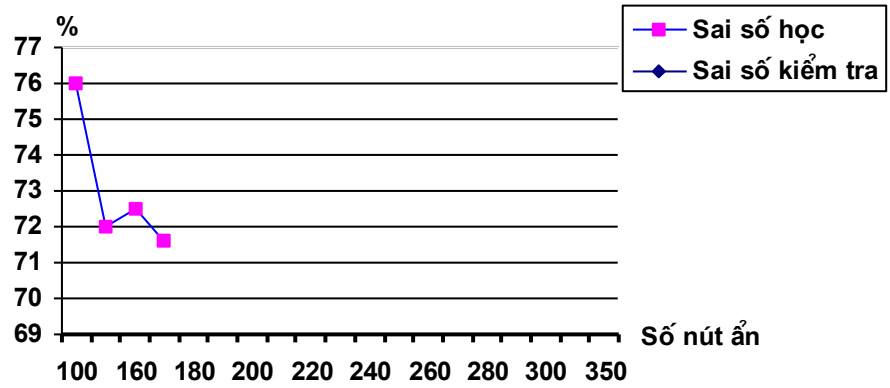
2.1. Trích đặc trưng bằng hương pháp LPC :



Hình 23Mạng Neuron với đặc trưng LPC

Kết quả nhận dạng đạt : 91%

2.2. Trích đặc trưng bằng phương pháp MFCC



Hình 24Mạng Neuron với đặc trưng MFCC

Kết quả nhận dạng đạt : 94%

III. Nhận xét kết quả :

Từ những kết quả thu được có thể thấy phương pháp nhận dạng bằng mạng Neuron cho những kết quả khả quan hơn so với sử dụng mô hình Markov ẩn. (Kết quả nhận dạng 94% so với 80 %)

Như vậy, qua quá trình thử nghiệm các mô hình cũng như các phương pháp trích đặc trưng. Quyết định cuối cũng được đưa ra là lựa chọn mô hình Neuron với 250 nút ẩn, các hàm truyền là hàm logsigmoid và hàm purelin. Cùng với phương pháp trích đặc trưng là MFCC 13 hệ số.

CHƯƠNG IV : CÀI ĐẶT THUẬT TOÁN NHẬN DẠNG TRÊN VI XỬ LÍ DSP

I. Giới thiệu về DSP C6713

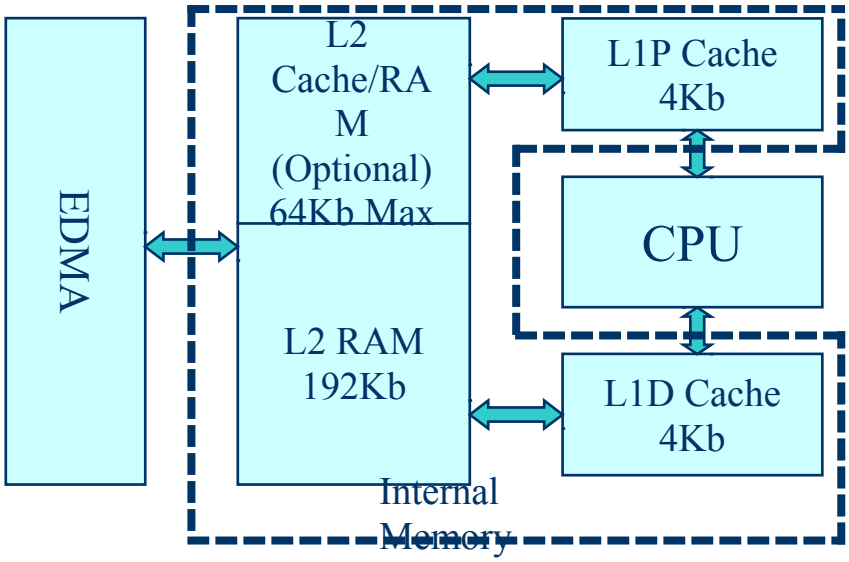
Vi xử lý TMS320C6713 nằm trong series chip DSP TMS320C67x là dòng chip DSP dấu phẩy động (floating-point) dựa trên nền TNS320C6000. C6713 dựa trên kiến trúc very-long-intruction-word (VLIW) được phát triển bởi Texas Intrusment (TI), điều này làm nó trở thành một sự lựa chọn hoàn hảo cho các ứng dụng đa kênh và đa chức năng.

II. Một số đặc điểm kĩ thuật của DSP C6713

Hoạt động trên tần số 225MHz, C6713 có thể thực hiện 1350 triệu phép toán dấu phẩy động trong một giây (MFLOPS), 1800 triệu câu lệnh trên giây (MIPS).

C6713 sử dụng kiến trúc bộ nhớ cache 2 cấp:

- Cấp1: Bộ nhớ chương trình (L1P) là bộ nhớ địa chỉ trực tiếp 4K-Byte, và bộ nhớ dữ liệu chiều 4K-Byte.
- Cấp2: là một không gian bộ nhớ 256K-Byte được sử dụng chung giữa bộ nhớ chương trình và bộ nhớ dữ liệu. Trong đó 64K có thể cấu hình thành memory hoặc cache hoặc là kết hợp của cả 2. 192K còn lại được định vị làm SRAM.



Hình : Cấu trúc bộ nhớ của DSP C6713

Bên trong vi xử lý C6713 có 8 khối hàm bao gồm 6 khối tính toán logic (ALU) và 2 khối nhân, một cơ chế bus địa chỉ 32bit cho phép đánh địa chỉ 4G, và 2 tập các thanh ghi general-perpose 32-bit.

C6713 có một bộ ngoại vi khá phong phú như 2 cổng Serial Audio đa kênh (McASPs), 2 cổng Serial Buffered đa kênh (McBSPs), 2 mạch bus tích hợp, 1 mô dun Input/Output General-Purpose (GPIO), 2 timer, …

Mô đun giao diện của hai kênh McASP hỗ trợ một dải xung nhịp đồng hồ cho việc truyền và một dải cho việc nhận. Mỗi kênh có 8 chân dữ liệu nối tiếp có thể được phân phối với bất kì

dải xung nhịp đồng hồ nào ở trên. Cổng nối tiếp hỗ trợ phân chia thời gian đa thành phần (time-division multiplexing) trên mỗi chân từ 2 đến 32 time slot. Nó có dải thông đủ rộng để hỗ trợ cả 16 chân dữ liệu nối tiếp truyền một tín hiệu 192kHz stereo. Tín hiệu có thể đc truyền và nhận trên các chân nối tiếp và đc định dạng một cách da dạng dựa trên định dạng âm thanh của Philips Inter-IC. Thêm vào đó, bộ truyền nhận của McASP có thể đc lập trình để đưa ra dữ liệu đc mã hoá theo các chuẩn S/PDIF, IEC60958, AES-3, CP-430, với 1 bộ nhớ RAM để chứa các dữ liệu của người dùng và trạng thái của kênh.

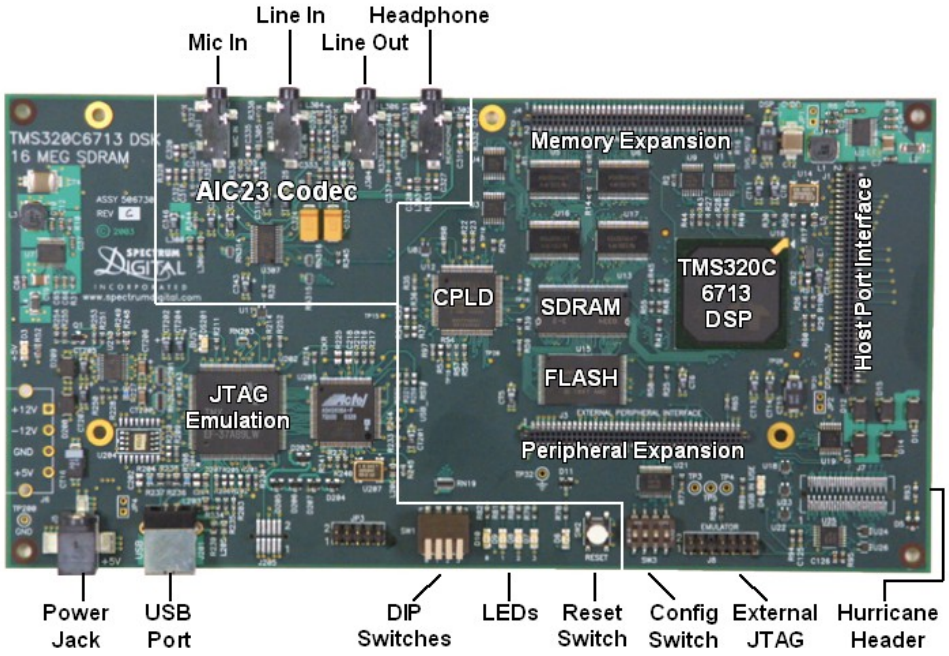
McASP cũng cung cấp các chức năng kiểm tra lỗi và phục hồi, như: mạch phát hiện lỗi xung đồng hồ để xác nhận xung master nằm trong dải tần số đã được lập trình.

2 cổng I2C trên TMS320C6713 cho phép DSP có thể dễ dàng điều khiển các thiết bị ngoại vi và có thể giao tiếp với chíp chính. Thêm vào đó, cổng McBSP có thể được sử dụng để giao tiếp với các thiết bị ngoại vi sử dụng chuẩn giao tiếp SPI (serial peripheral interface).

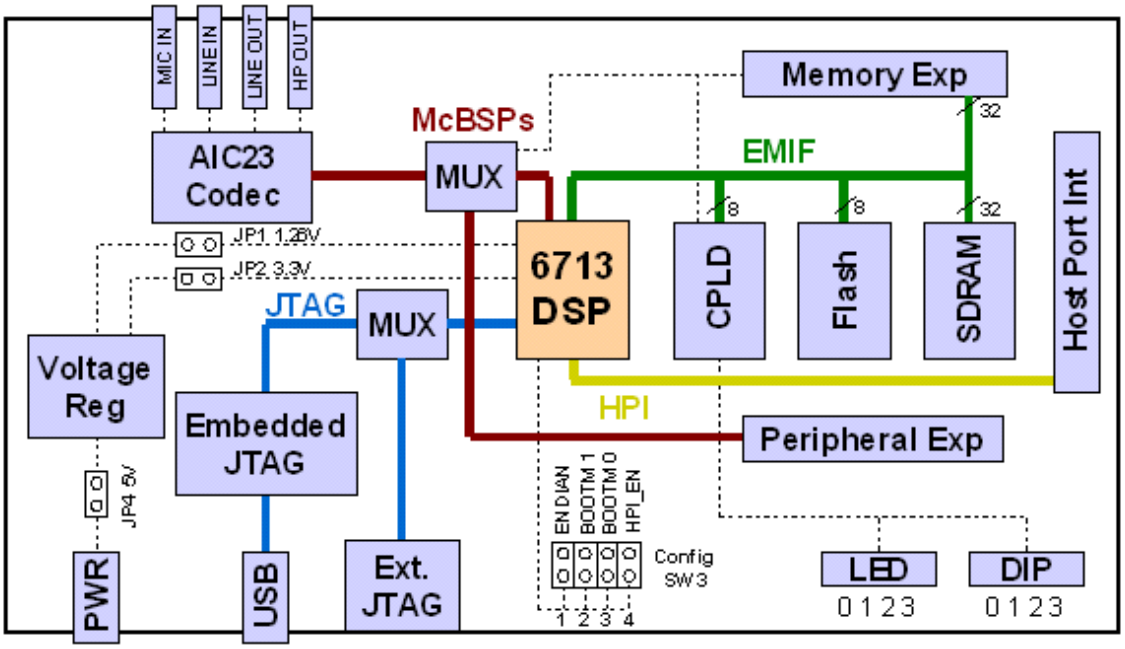
III. Bộ DSK 6713

Bộ kit phát triển C6713 là một bộ kit tương đối rẻ (395$) của hãng Texas Instrument bao gồm cả phần cứng và các phần mềm đi kèm dễ sử dụng. Các đặc trưng chính của bộ kit phát triển này là:

- Gồm một Vi xử lí TMS320C6713 DSP với tần số hoạt động lên đến 225Mhz và rất phù hợp với các ứng dụng liên qua đến tính toán đến dấu phẩy động.
- Một bộ codec AIC23 phù hợp với các ứng dụng về âm thanh
- Gồm 4 phím nhấn và 4 đèn LED dễ dàng cho việc kiểm tra
- Bộ nhớ flash và bộ nhớ SDRAM lên tới 16Mb.



Hình 25Bảng mạch DSK 6713

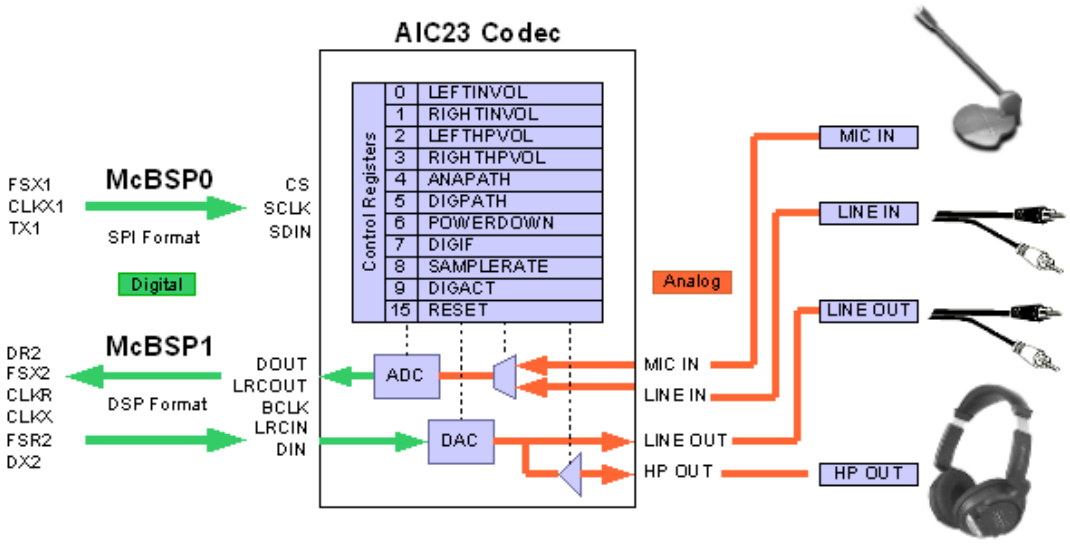


Hình 26 Cấu trúc bộ kit DSK 6713

Đi kèm với bộ kit phát triển này còn có phần mềm Code Composer Studio hỗ trợ cho việc viết và gỡ rối các chương trình hay thuật toán tính toán.

Iç. Bộ Codec AIC23

Mô hình bộ AIC23



Hình 27 Mô hình bộ CodecAIC23

DSK 6713 dùng bộ codec AIC23 của hãng Texas Instrumnet cho tín hiệu vào ra âm thanh. Bộ codec lấy mẫu tín hiệu tương tự từ đường mic in hay đường line in và chuyển chúng thành dạng số thông qua ADC sau đó được xử lí tiếp theo. Sau khi DSP kết thúc việc xử lí, dữ liệu

được chuyển qua một bộ DAC để thành dạng tín hiệu tương tự và đưa ra ở đầu ra của bộ codec là line out hay head phone.

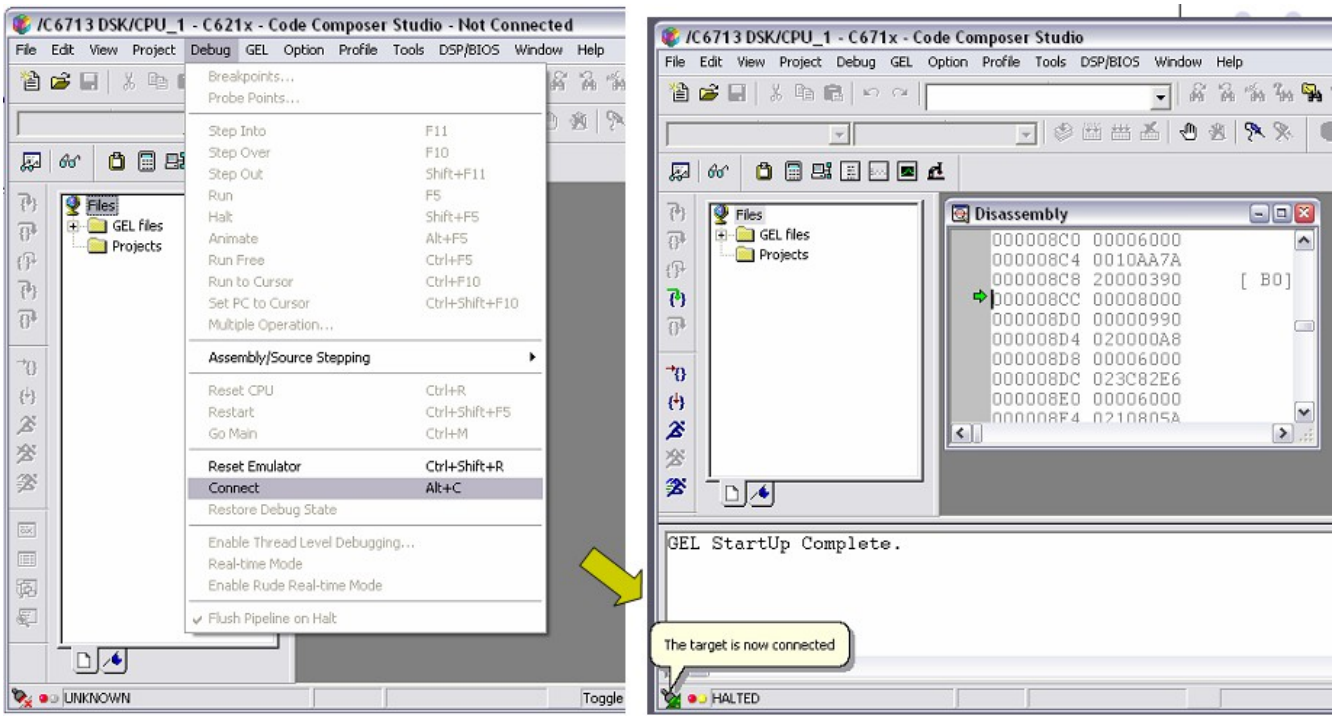
Giao tiếp với bộ codec thông qua 2 kênh nối tiếp, một dung để điều khiển bộ codec thông qua việc cấu hình cho các thanh ghi điều khiển bên trong và một cái còn lại dùng cho việc truyền nhận dữ liệu dạng số. AIC23 hỗ trợ nhiều cách câu hình mà có thể thay đổi định dạng của dữ liệu của kênh điều khiển và kênh dữ liệu.

Các thông số điều khiển được chứa trong các thanh ghi điều khiển. Các giá trị này được truyền thông qua kênh McBSP0 16 bit. Đây là đường truyền chỉ có một chiều duy nhất từ DSP đến bộ codec. Các thanh ghi có độ rộng là 9 bit. Một từ 16 bit được truyền đến bộ codec 16 bit gồm 6 bit đầu tiên để định ra địa chỉ của thanh ghi, 9 bit còn lại chứa dữ liệu được truyền đến thanh ghi.

ç. Code Compose Studio (CCS)

CCS cung cấp một Intergrated Development Environment (IDE). CCS có các tool cho việc soạn code, như 1 trình dịch C, một bộ hợp ngữ, và một linker. Nó có các chức năng đồ hoạ và hỗ trợ gỡ rối thời gian thực. Đây là một công cụ phần mềm dễ sử dụng để xây dựng và debug một chương trình.

Trình dịch C dịch 1 chương trình mã nguồn C (file .c)thành mã nguồn hợp ngữ (file .asm). Chương trình dịch hợp ngữ dịch file .asm này về ngôn ngữ máy (file .obj). Bộ linker sẽ kết hợp object file với các thư viện object để tạo thành một file chạy (.out) File chạy này có thể được nạp xuống và chạy trực tiếp trên vi xử lý C6713.



Hình 28Kết nối CCS với DSK 6713

I. Cài đặt thuật toán nhận dạng trên DSP 6713

çI. Thu tín hiệu âm thanh trên DSK 6713

Hai hàm chính *input_sample* và *output_sample*

Có 2 phương pháp lập trình thu âm:

- Thu âm sử dụng ngắt. (Tín hiệu vào từ line in)
- Thu âm dùng vòng quét. (Tín hiệu vào từ MIC)

Thu âm sử dụng ngắt: Sau khi khởi tạo và cho phép ngắt, chương trình đợi trong một vòng lặp vô hạn đến khi sự kiện ngắt xuất hiện. Ngắt được thực hiện trong mỗi chu kì lấy mẫu (Ví dụ với tần số lấy mẫu là 8kHz thì chu kì là 0.125ms), tại mỗi thời điểm đó, giá trị của tín hiệu vào được đọc và đưa đến ADC của bộ mã hoá và gửi ra đến bộ DAC.
Hoàn toàn có thể đặt thêm các hệ số khuyếch đại cho việc thu và phát tín hiệu.

Thu âm sử dụng vòng quét: ADC lấy tín hiệu vào từ đường MIC IN. Sử dụng kĩ thuật vòng quét như một thủ tục liên tục để kiểm tra xem khi nào dữ liệu đã sẵn sàng. Đây là kĩ thuật đơn giản hơn kĩ thuật ngắt nhưng nó kém hiệu quả hơn do dữ liệu cần được kiểm tra liên tục để xác định xem khi nào nó sẵn sàng để được nhận hay để truyền đi.

Các hàm chính như *input_sample, output_sample, comm_intr* hay *comm_poll* được định nghĩa sẵn trong file *C6713dskinit.c* . Điều này giúp cho các chương trình nguồn giảm đi kích thước rất nhiều.

çII. Cài đặt thuật toán trích đặc trưng MFCC và mạng Neuron lên chip DSP

Toàn bộ chương trình xử lý được chuyển từ code Matlab thành dạng code C và nằm trong file mfcc_neuron.c.

Tín hiệu sau khi được thu sẽ được sử lý để trích đặc trưng MFCC, sẽ cho ra một vectơ hệ số đặc trưng Ceptrals gồm 130 phần tử.

Mạng Neuron sau khi được huấn luyện bằng chương trình Matlab được lưu lại dưới dạng 4 ma trận W, L, B1, B2 trong các file text tương ứng : W.txt, L.txt, B1.txt, B1.txt và sẽ được chương trình C đọc vào.

Chương trình sẽ tính toán vector đặc trưng với mạng Neuron và đưa ra kết quả cuối cùng là từ nhận được.

KẾT LUẬN

Nhận xét kết quả chung của đồ án

Đồ án đã thực hiện được việc xây dựng các mô hình nhận dạng tiếng nói, cụ thể là nhận dạng các từ điều khiển rời rạc: Tắt, Bật, Chạy, Dừng, Tiền, Lùi, Trái, Phải, Trên, Dưới. Và đã tiền hành chạy thử nghiệm dựa trên các phương pháp phân tích đặc trưng của tín hiệu là LPC và MFCC.

Dựa trên cơ sở dữ liệu thu thập được đã đưa ra được một mô hình nhận dạng thích hợp nhất.

Tuy nhiên vẫn không tránh khỏi một số hạn chế:

- Số lượng mẫu còn ít, nên chưa khẳng định được sự hội tụ của thuật toán.
- Chất lượng mẫu không cao và không đồng nhất do tiến hành thu bằng máy tính cá nhân và ở các địa điểm khác nhau.
- Còn một số hạn chế trong phương pháp làm việc do thiếu kinh nghiệm.
- Việc chuyển đổi thuật toán từ Matlab sang C để cài đặt cho DSP vẫn gặp phải một số sai số trong tính toán. Dẫn đến kết quả nhận dạng trong thực tế không được cao như khi chạy thử nghiệm trên máy tính.

Ngoài ra còn một số khó khăn khách quan: như đặc trưng của các từ tiếng Việt khác biệt so với từ tiếng Anh, một số từ điều khiển có nhiều đặc điểm giống nhau dẫn đến nhận dạng sai (như trái - phải; dừng - dưới)…

Trong quá trình thực tập làm đồ án, sinh viên đã cố gắng hết sức nghiên cứ và làm việc nghiêm túc để hoàn thành yêu cầu của đề tài. Từ đó thu được những kiến thức và những kinh nghiệm rất bổ ích. Tuy nhiên, trong quá trình làm việc, cũng như trong bản thân đồ án này vẫn không tránh khỏi những thiếu sót, do vậy sự chỉ bảo, góp ý của các thấy cô giáo sẽ là sự giúp đỡ vô cùng quí báu để đồ án được hoàn thiện hơn.

TÀI LIỆU THAM KHÁO

[1] Fundamentals of speech recognition – Lawrence Rabiner – Prentice Hall 1993

[2] A turial on Hidden Markov Model and selected applications in speech recognition – Lawrence Rabiner. IEEE - 1989

[3] Chapter 9: Automatic Speech Recognition – Speech and Language Processing: An Introduction to natural processing, computational linguistics and speech recognition – Daniel Jurafsky & James H. Martin. 2007.

[4] Nhận dạng tiếng Việt dùng mạng Neuron và trích đặc trưng dùng LPC và AMDF – Hoàng Đình Chiến

[5] Speech Recognition using Neural Network – Joe Tebelskis 1995

[6] Bài giảng môn nhận dạng của thầy Trần Hoài Linh, ĐHBK Hà Nội, 2007

[7] Digital Signal Processing and Applications with the C6713 and C6416 DSK – Rulph Chassaing – Wiley 2004

[8] C Algorithms For Real-Time DSP – Paul M. Embree – Prentice Hall 1995

[9] Lập trình Matlab và ứng dụng – Nguyễn Hoàng Hải , Nguyễn Việt Anh – NXB Khoa học kĩ thuật HN 2005.

[10] Mel frequency ceptral coefficients – Wikipedia.org và các link tham chiếu

[11] HMM toolbox for matlab -http://www.cs.ubc.ca/~murphyk/Software/HMM/hmm.html

[12] Auditory toolbox for matlab - http://www.slaney.org/malcolm/pubs.html

[13] ECE4703 Real-Time DSP Orientation Lab - D. Richard Brown III 2004

Và một số tài liệu khác.