Mục lục

[Lời mở đầu 5](#_Toc357089316)

[Chương 1: TỔNG QUAN VỀ HỆ THỐNG NHẬN DẠNG TIẾNG NÓI 6](#_Toc357089317)

[1.1.Giới thiệu 6](#_Toc357089318)

[1.2.Sơ đồ khối của hệ thống nhận dạng tiếng nói. 7](#_Toc357089319)

[1.2.1.Mô đun thu và xử lý tín hiệu nói đầu vào. 9](#_Toc357089320)

[1.2.2.Mô đun trích chọn đặc trưng. 14](#_Toc357089321)

[1.2.3. Mô đun nhận dạng 15](#_Toc357089322)

[1.3. Những khó khăn gặp phải trong nhận dạng tiếng nói 16](#_Toc357089323)

[1.4. Tác dụng của thuật toán phát hiện tiếng nói 17](#_Toc357089324)

[Chương 2: CÁC THUẬT TOÁN PHÁT HIỆN TIẾNG NÓI 20](#_Toc357089325)

[2.1.Thuật toán qua điểm không ( Zero crossing ) 20](#_Toc357089326)

[2.2.Thuật toán năng lượng trong thời gian ngắn“Short time energy” 22](#_Toc357089327)

[2.3.Thuật toán LTSD ( Long-term spectral divergence ) 23](#_Toc357089328)

[2.4.Nhận xét, so sánh giữa các thuật toán 25](#_Toc357089329)

[Chương 3: TRIỂN KHAI THUẬT TOÁN PHÁT HIỆN TIẾNG NÓI LTSD 28](#_Toc357089330)

[3.1.Các bước triển khai thuật toán 28](#_Toc357089331)

[3.2.Platform 29](#_Toc357089332)

[3.3.Công việc đã thực hiện 29](#_Toc357089333)

[3.3.1.VAD offline trên máy tính PC 29](#_Toc357089334)

[3.3.2.VAD online trên máy tính PC 31](#_Toc357089335)

[3.3.3.Áp dụng phương pháp VAD cho thiết bị Android 33](#_Toc357089336)

[3.3.4.VAD trên server nhận dạng 34](#_Toc357089337)

[3.4.Đánh giá độ chính xác của thuật toán 35](#_Toc357089338)

[3.4.1.Các bước thực hiện 36](#_Toc357089339)

[3.4.2.Bộ thông số đầu vào của thuật toán VAD\_LTSD 38](#_Toc357089340)

[3.4.3.Sử dụng phương pháp ROC tính toán độ chính xác 38](#_Toc357089341)

[3.4.4.Kết quả đánh giá 39](#_Toc357089342)

[Chương 4: KẾT NỐI VỚI MÔ ĐUN NHẬN DẠNG 40](#_Toc357089343)

[4.1.Sơ đồ chung 40](#_Toc357089344)

[4.2.Cách thức hoạt động 40](#_Toc357089345)

[4.3.Các vấn đề gặp phải và giải pháp 41](#_Toc357089346)

[4.4.Đánh giá tốc độ gửi nhận client-server 42](#_Toc357089347)

[Chương 5: KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN 44](#_Toc357089348)

[5.1.Kết luận 44](#_Toc357089349)

[5.2.Hướng phát triển 45](#_Toc357089350)

[Tài liệu tham khảo 46](#_Toc357089351)

**Danh mục hình vẽ**

Hình 1-1: Sơ đồ khối hệ thống nhận dạng tiếng nói

Hình 1-2: Rời rạc hóa tín hiệu tiếng nói

Hình 1-3 : Phân khung tín hiệu thành các frame nhỏ

Hình 1-4: Cửa sổ Hamming 128 điểm

Hình 1-5: Ảnh hưởng của cửa sổ Hamming lên tín hiệu

Hình 1-6: Biên độ phổ sau khi FFT

Hình 1-7: Sơ đồ khối phương pháp LPC

Hình 1-8: Mô hình thuật toán MFCC

Hình 2-1 : Zero crossing

Hình 2-2: Phân bố LTSD theo cấp N

Hình 2-3: Sơ đồ thuật toán VAD\_LTSD

Hình 3-1: Sơ đồ thực hiện VAD offline cho máy tính PC

Hình 3-2: Quá trình thực hiện VAD online trên máy tính PC

Hình 3-3: Quá trình áp dụng VAD cho thiết bị Android

Hình 3-4: Gán nhãn cho file thu âm nói liên tục

Hình 3-5: Gán nhãn cho file thu âm nói ngắt quãng trong môi trường yên tĩnh

Hình 3-6: File txt có được sau khi gán nhãn

Hình 3-7:File .txt sau khi gán nhãn file thu âm nói ngắt quãng

Hình 4-1: Sơ đồ kết nối client-server của hệ thống nhận dạng

**Danh mục bảng**

Bảng 3-1: Kết quả đánh giá độ chính xác VAD\_LTSD

Bảng 4-1: Tốc độ gửi nhận client-server

# Lời mở đầu

Sau 6 tuần thực tập trên trung tâm MICA và 15 tuần làm đồ án với sự giúp đỡ rất nhiệt tình của các thầy cô giáo em đã hoàn thành được đề tài “Ứng dụng thuật toán phát hiện tiếng nói trong hệ thống nhận dạng tiếng nói”. Đặc biệt em xin gửi tới TS Nguyễn Quốc Cường lòng biết ơn sâu sắc, người đã trực tiếp hướng dẫn và giúp đỡ em tìm hiểu, tiếp cận với lĩnh vực xử lí tín hiệu và nhận dạng tiếng nói. Xin cảm ơn anh Nguyễn Đăng Khoa đã giúp đỡ, góp ý nhiệt tình trong quá trình em hoàn thành đồ án.

Em xin cảm ơn các thầy cô giáo trong bộ môn Kỹ thuật đo và Tin học công nghiệp và toàn thể các thầy cô giáo trong chương trình đào tạo kĩ sư chất lượng cao tại Đại Học Bách Khoa Hà Nội đã cho chúng em kiến thức, kinh nghiệm cùng những bài học quý giúp chúng em trưởng thành hơn.

Nội dung trình bày đồ án:

* Chương 1: Giới thiệu chung hệ thống nhận dạng tiếng nói, sơ đồ khối của hệ thống, những khó khăn gặp phải và tác dụng của thuật toán phát hiện tiếng nói trong một hệ thống nhận dạng tiếng nói.
* Chương 2: Các phương pháp, thuật toán phát hiện tiếng nói và lý do chọn thuật toán phát hiện tiếng nói “Long term spectral divergence” ( VAD\_LTSD)
* Chương 3: Miêu tả chi tiết về thuật toán VAD\_LTSD và các bước triển khai thuật toán trong các ứng dụng cùng kết quả đánh giá độ chính xác của thuật toán.
* Chương 4: Cách thức kết nối mô hình client-server với mô đun nhận dạng sử dụng Sphinx 4 và cách ghép nối mô phát hiện tiếng nói với mô đun nhận dạng trên server.
* Chương 5: Kết luận và hướng phát triển

Hà Nội, ngày 23 tháng 05 năm 2013

# Chương 1: TỔNG QUAN VỀ HỆ THỐNG NHẬN DẠNG TIẾNG NÓI

## 1.1.Giới thiệu

Nhận dạng tiếng nói tự động ( Automatic Speech Recognition – ASR ) đã từng là chủ đề được nghiên cứu nhiều nhất trong lĩnh vực xử lý âm thanh suốt nhiều thập kỉ qua. Nói chung đây được hiểu là vấn đề nhận dạng các từ riêng rẽ trong một từ điển tạo ra bởi người nói cho trước, quá trình nhận dạng chỉ dựa vào thông tin chứa đựng trong tín hiệu tiếng nói và vào các hiểu biết trước đó về phạm vi của vấn đề được nói đến. Trong thập niên 50, khi bắt đầu có những nghiên cứu về nhận dạng tiếng nói, nhiều nghiên cứu cho rằng sự phát triển nhanh chóng của ngành khoa học máy tính sẽ làm cho việc nhận dạng trở nên dễ dàng hơn. Nhưng đáng tiếc là một vài thập kỉ sau đó, người ta đã nhận ra rằng đó là một phỏng đoán sai. Nhận dạng tiếng nói được nhận định là một vấn đề rất phức tạp và ngày nay nhiều câu hỏi khó và mang tính mở được đặt ra xa hơn những gì đã giải quyết được. Những khó khăn kéo theo là sự tăng kích thước từ điển ( trên 50000 từ ), việc nhận dạng tiếng nói liên tục, số lượng và đặc trưng phát âm của người nói trong nhận dạng không phụ thuộc người nói ( trước kia là hệ thống phụ thuộc một người nói, quá trình huấn luyện và nhận dạng đều lấy mẫu tín hiệu từ chính người đó ), việc nhận dạng tiếng nói tự phát ( a, ô, ư, … ), sự tăng ảnh hưởng của các điều kiện môi trường xung quanh ( nhiễu, méo tín hiệu trên kênh, … ) và nhiều vấn đề khác nữa. Hầu hết những vấn đề như trên phát sinh ra khi chuyển từ điều kiện phòng thí nghiệm sang điều kiện thực tế.

Nhận dạng tiếng nói có nhiều ứng dụng trong thực tế như: tạo lập từ điển âm thanh, tự động sinh văn bản từ tín hiệu tiếng nói, truy cập cơ sở dữ liệu bằng tiếng nói, giao tiếp người-máy, dịch vụ truy cập từ xa thông qua đường điện thoại, ứng dụng cả trong điều khiển tự động, … và còn rất nhiều lĩnh vực khác.

Vấn đề nhận dạng tiếng nói có thể được xem như là vấn đề phân loại mang tính thống kê ( các lớp khác nhau ), giống như hình thức nhận dạng mẫu cổ điển. Một lớp được định nghĩa là một tập W các từ cho phép trong một điển đóng, một tham số cho tín hiệu tiếng nói được chọn ( như chuỗi các vector đặc trưng âm học X ) và một xác suất điều kiện MAP ( Maximum a Posteriori ). Vấn đề phân loại có thể được bắt đầu bằng việc tìm ra chuỗi thỏa P(W | X ) đạt cực đại. Theo luật Bayes:

 (1.1)

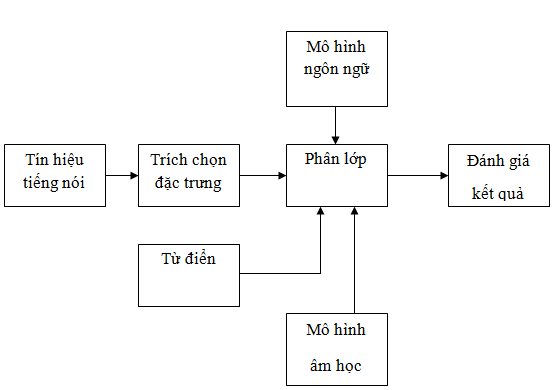
Trong đó P(W | X) là xác suất chuỗi vector X được phân loại vào lớp W, P(X | W) là xác suất xuất hiện của chuỗi vector X khi sử dụng mô hình của lớp W, P(W) là xác suất xuất hiện mô hình lớp W, P(X) là xác suất xuất hiện chuỗi vector X.

Với một chuỗi quan sát âm học X cho trước, việc cố gắng tìm cực đại P(W | X) có thể được chuyển thành việc tìm lớp để cực đại vế phải của biểu thức (1.1). Giá trị P(W) được tham chiếu như một mô hình ngôn ngữ ( language model ) phụ thuộc vào các ràng buộc ở mức cao và các hiểu biết ngôn ngữ ở khía cạnh chấp nhận từ đó trong tình huống cụ thể đó không. Giá trị P(X|W) được hiểu như một mô hình âm học. Nó mô tả chuỗi thống kê các tham số quan sát âm học trong không gian đặc trưng cho bởi từ tương ứng. Mô hình Markov ẩn ( Hidden Markov Model – HMM ) là một mô hình tiêu biểu ở mức âm học. Mô hình Markov ẩn mặc dù là một tiếp cận tốt của vấn đề mô hình âm học trong nhận dạng tiếng nói, cho phép hiệu năng nhận dạng cao trong nhiều hoàn cảnh song nó vẫn mang nhiều hạn chế.

## 1.2.Sơ đồ khối của hệ thống nhận dạng tiếng nói.

Một hệ thống nhận dạng nói chung thường bao gồm 2 phần, phần huấn luyện ( training phase ) và phần nhận dạng ( recognition phase ). Huấn luyện là quá trình học những mẫu chuẩn được cung cấp bởi những tiếng khác nhau ( từ hoặc âm vị ( phone )), để từ đó hình thành bộ từ vựng của hệ thống. “Nhận dạng” hay còn gọi là decoding là quá trình quyết định xem từ nào đã được đọc căn cứ vào bộ từ vựng đã được huấn luyện.

Tín hiệu tiếng nói sau khi được lấy mẫu được đưa qua khối xử lý tín hiệu để trích chọn đặc trưng phục vụ cho quá trình nhận dạng. Các phương pháp trích chọn đặc trưng thường được sử dụng trong nhận dạng tiếng nói là MFCC ( Mel – scale Frequency Ceptral Coeficients) và PLP (Peceptual Linear Prediction). Sau khi trích chọn đặc trưng, các vector đặc trưng được đưa vào khối nhận dạng, kết hợp với các hiểu biết về đặc trưng ngôn ngữ học và đặc trưng về âm học của tiếng nói, khối nhận dạng sẽ đưa ra kết quả nhận dạng hợp lý nhất. Bộ tham số của mô hình nhận dạng thu được trong quá trình luyện mô hình với một tập dữ liệu train có trước. Nếu tập train mà càng lớn thì khả năng nhận dạng của mô hình sẽ càng cao.



Hình 1-1: Sơ đồ khối hệ thống nhận dạng tiếng nói

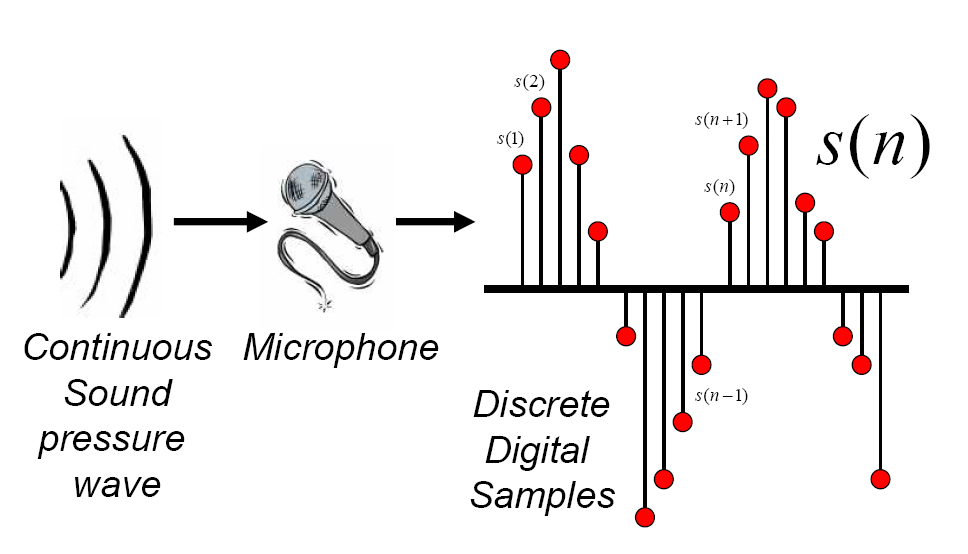
Để thuận tiện cho việc kiểm tra và đánh giá kết quả từ sơ đồ trên ta sẽ chia chương trình nhận dạng tiếng nói thành 3 mô-đun riêng biệt:

* Mô-đun 1: Tiến hành thu tín hiệu tiếng nói từ MIC. Sau đó tiến hành lấy mẫu tín hiệu tiếng nói.
* Mô-đun 2: Trích chọn đặc trưng tín hiệu tiếng nói đã thu ở mô-đun 1 bằng phương pháp MFCC, đồng thời thực hiện ước lượng vector các vector đặc trưng này.
* Mô-đun 3: Sau khi trích chọn đặc trưng, các vector đặc trưng được đưa vào khối nhận dạng bao gồm bộ decoder, mô hình âm hình và mô hình ngôn ngữ.

### 1.2.1.Mô đun thu và xử lý tín hiệu nói đầu vào.

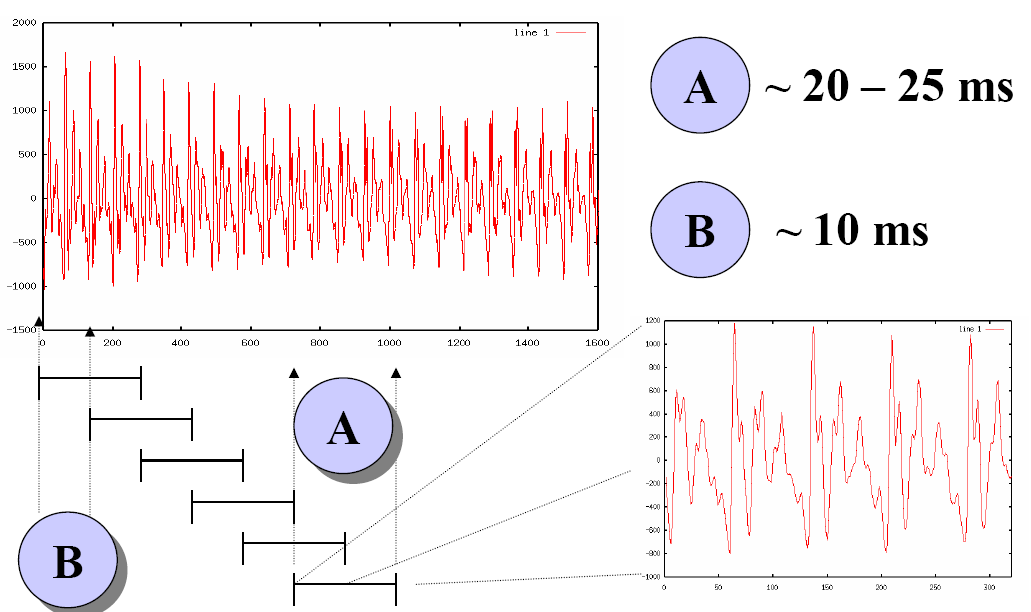
Mô đun này tiến hành thu tín hiệu tiếng nói đầu vào từ đường MIC để phát hiện phần tín hiệu tiếng nói và phần tín hiệu nhiễu. Từ đó ta có thể tách tín hiệu tiếng nói ra khỏi nền nhiễu. Hiện nay có nhiều phương pháp tách tiếng nói khác nhau, hiện nay người ta hay dùng nhất là dựa vào mức năng lượng. Phương pháp này dựa vào tính chất của năng lượng tiếng nói thường lớn hơn năng lượng của tín hiệu nhiễu. Quá trình cơ bản của thuật toán như sau: một mẫu nhỏ của nền nhiễu được lấy trong suốt khoảng lặng (silence) cho đến trước điểm bắt đầu của tín hiệu tiếng nói. Từ đây ngưỡng tiếng nói được xác định dựa trên năng lượng khoảng lặng và năng lượng đỉnh. Ban đầu những điểm kết thúc được xác định ở những nơi năng lượng tín hiệu vượt qua ngưỡng này, sau đó ta tính khoảng cách giữa hai điểm xem có thỏa mãn độ dài của một từ hay không.

Nhiệm vụ đầu tiên của mô đun này là thu âm tín hiệu âm thanh đầu vào. Quá trình thu âm lời nói thực chất là quá trình biến đổi một tín hiệu liên tục thành một tín hiệu rời rạc (qua microphone). Tần số lấy mẫu của Microphone thường chọn 16 KHz, còn tần số lấy mẫu của Telephone ( điện thoại) thường là 8 KHz. Lí do là bởi tai người có thể nghe được các âm thanh có tần số từ 20 Hz đến 10 KHz ( tính nhất trong khoảng từ 100Hz đến 5 KHz). Theo định lý lấy mẫu Shannon, tần số lấy mẫu phải lớn hơn 2 lần tần số lớn nhất đo được. Vì vậy, việc chọn tần số lấy mẫu của Microphone 16 KHz là hợp lí.



Hình 1-2: Rời rạc hóa tín hiệu tiếng nói

Để có thể tiền xử lí tín hiệu đầu vào tốt, một kĩ thuật hay được sử dụng là phân khung và cửa sổ hóa tín hiệu. Đặc điểm của tín hiệu tiếng nói là tín hiệu chậm biến đổi theo thời gian, hay còn được gọi là “quasi – stationary”. Khi xem xét một khoảng thời gian đủ ngắn, khoảng từ 5 – 100ms, tính chất của nó gần như là ổn định. Tuy nhiên, trong một khoảng thời gian dài, tính chất của nó bị thay đổi, phản ánh những âm thanh khác nhau được nói. Vì vậy phân tích khoảng thời gian ngắn (short time spectral analysis) thường được sử dụng trong phân tích tính chất của tín hiệu tiếng nói. Trong khi một từ được phát ra có thể dài vài giây, nên cần thiết phải chia các tín hiệu thu được thành các frame nhỏ, các frame này có độ dài tương ứng 20 – 25 ms, hai frame liên tiếp cách nhau khoảng từ 10 đến 24 ms.



Hình 1-3 : Phân khung tín hiệu thành các frame nhỏ

Để tránh sự thay đổi đột ngột giữa các frame, sẽ có sự lặp lại giữa các frame liên tiếp. Ví dụ, mỗi frame đầu tiên có N mẫu; frame kế tiếp cũng sẽ có N mẫu, nhưng chỉ có M ( M < N ) mẫu là “mới”, còn lại N – M mẫu đầu tiên là N - M mẫu cuối cùng của frame đầu tiên, quá trình như vậy tiếp diễn cho những frame sau.

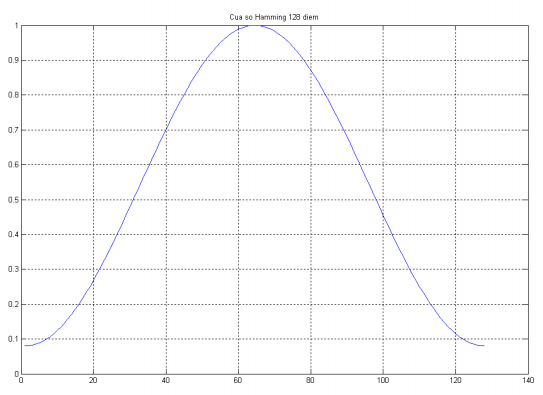
Bước tiếp theo là cửa sổ hóa mỗi frame để giảm sự không liên tục giữa điểm đầu và điểm cuối của mỗi frame. Nguyên tắc của việc cửa sổ hóa là giảm thiểu méo dạng phổ bằng cách làm hẹp dần ở phía đầu và cuối của mỗi frame.

Gọi cửa sổ là w(n), 0 ≤ n ≤ N – 1, với N là số phần tử trong mỗi frame. Tín hiệu sau khi cửa số hóa như sau:

y1(n) = x1(n) \* w(n) 0 ≤ n ≤ N – 1.

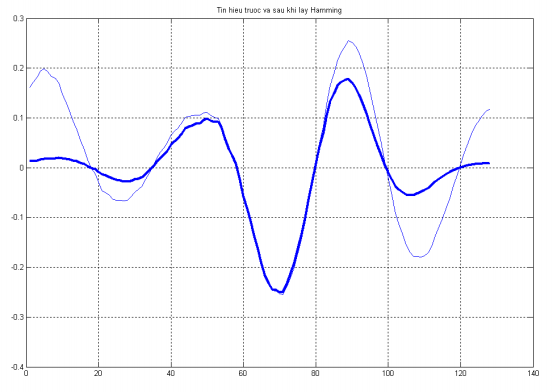
Thông thường cửa sổ Hamming được dùng, có dạng như sau:

w(n) = (1.2)



Hình 1-4: Cửa sổ Hamming 128 điểm

Tín hiệu sau khi Hamming hóa sẽ bị “thu nhỏ” lại ở hai đầu :

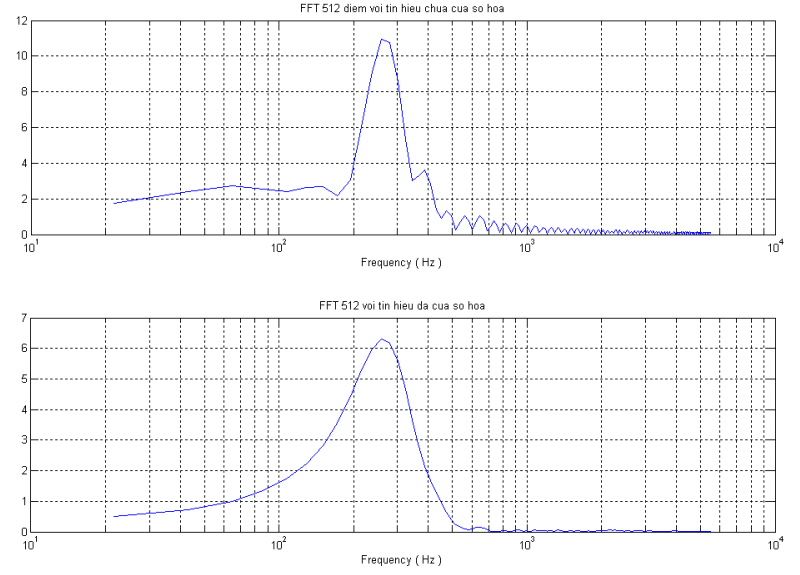


Hình 1-5: Ảnh hưởng của cửa sổ Hamming lên tín hiệu

Trong tiền xử lí tín hiệu, việc chuyển đổi tín hiệu sang miền tần số mang lại nhiều thông tin hơn và từ đó có thể áp dụng các thuật toán trong mô đun tiền xử lí. Với tín hiệu số thì DFT là 2 phương pháp biến đổi từ miền thời gian rời rạc sang miền tần số rời rạc. Mục đích của quá trình này là chuyển đổi tín hiệu thời gian thành tín hiệu tần số. Đầu vào là tín hiệu sau khi nhân với hàm cửa sổ x[n], x[m]. Đầu ra : Số phức X[k] đại diện cho thành phần tần số của tín hiệu ban đầu. Thông thường, ta hay sử dụng thuật toán FFT ( Fast Fourier Transform ) để thực hiện nhanh DFT. Công thức tính:

X[k] = (k = 0,1,..., N-1) (1.3)

Thông thường, chọn N = 512 hoặc 1024.



Hình 1-6: Biên độ phổ sau khi FFT.

Hình trên: Tín hiệu không có cửa sổ hóa.

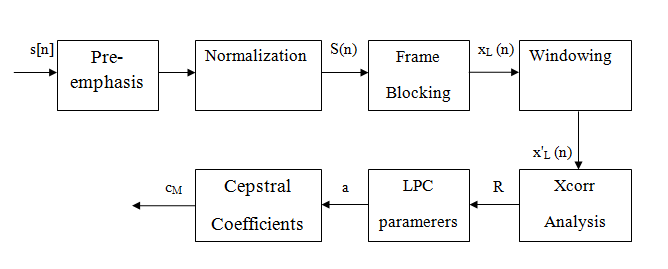
Hình dưới: Tín hiệu có bị cửa sổ hóa bằng Hamming.

### 1.2.2.Mô đun trích chọn đặc trưng.

Trích chọn đặc trưng của mẫu là một phần quan trọng của bất cứ hệ thống nhận dạng nào. Một cách lý tưởng, một đối tượng khác nhau sẽ có một hoặc nhiều đặc trưng. Các đặc trưng càng khác nhau giữa các đối tượng thì việc nhận dạng càng chính xác.

Việc nhận dạng sẽ dựa trên các đặc trưng này, có thể sử dụng 1 đặc trưng hoặc kết hợp nhiều đặc trưng lại với nhau. Với các hệ thống ASR hiện nay, thường chỉ sử dụng một đặc trưng của tín hiệu âm thanh. Cho tới thời điểm hiện nay, các phương pháp chủ yếu để tách đặc trưng có thể kể đến như: Linear Prediction Coding (LPC), Mel – Frequency Cepstrum Coefficients ( MFCC ), Principle Components Analysis (PCA) và các phương pháp khác...

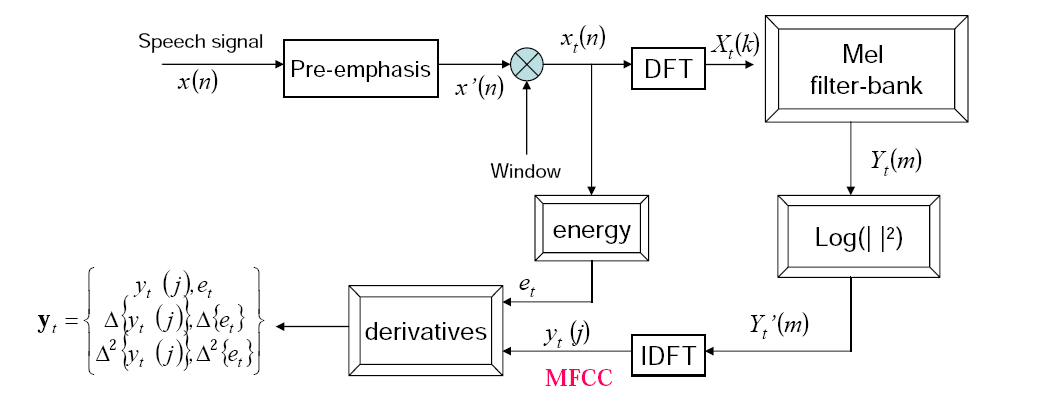
* Linear Prediction Coding



Hình 1-7: Sơ đồ khối phương pháp LPC

Toàn bộ một hệ thống ASR dùng phương pháp LPC được miêu tả khá đầy đủ tại địa chỉ: <http://www.clear.rice.edu/elec532/PROJECTS98/speech/cepstrum/cepstrum.html>

* Mel – Frequency Cepstrum Coefficients.



Hình 1-8: Mô hình thuật toán MFCC

Toàn bộ hệ thống ASR dùng phương pháp MFCC được miêu tả chi tiết tại :

LSA 352 Speech Recognition and Synthesis - Dan Jurafsky

### 1.2.3. Mô đun nhận dạng

Sau khi tín hiệu tiếng nói được trích chọn đặc trưng thì mỗi từ được đặc trưng bởi một ma trận hệ số thực. Lúc này ta đã có một cơ sở dữ liệu các vector đặc trưng ứng với từng từ. Mô đun nhận dạng bao gồm 3 phần: bộ decoder, mô hình âm học, mô hình ngôn ngữ. Bộ decoder sẽ kết hợp với mô hình âm học và mô hình ngôn ngữ để phát sinh ra chuỗi từ có xác suất hậu nghiệm lớn nhất đối với các vector đặc trưng đầu vào.

Phần này sẽ trình bày một số khái niệm về mô hình. Mô hình ở đây là mô tả các đối tượng toán học-những cái mà tập trung những thuộc tính thông thường nhất của từ được nói. Tùy theo cấu trúc tiếng nói 2 mô hình thường được sử dụng trong hệ thống nhận dạng là:

* Mô hình âm học: Chứa những đặc tính âm học cho mỗi senone. Đó là những mô hình không phụ thuộc vào ngữ cảnh gồm những đặc tính ( những vector đặc trưng tiêu biểu nhất cho mỗi phone ) và mô hình phụ thuộc vào ngữ cảnh ( xây dựng từ những senone cùng với ngữ cảnh ).
* Mô hình ngôn ngữ: Được sử dụng để hạn chế không gian tìm kiếm các từ. Nó định nghĩa những từ có thể theo sau, trước đó những từ được nhận dạng và giúp hạn chế đáng kể quá trình ánh xạ bằng việc loại bỏ những từ ít xảy ra nhất. Thông thường mô hình ngôn ngữ được sử dụng là mô hình n-gram, nó chứa những thống kê của chuỗi từ và thu hẹp trạng thái của những mô hình ngôn ngữ. Để đạt được độ chính xác tốt thì trong mô hình ngôn ngữ phải khá tốt trong việc thu hẹp không gian tìm kiếm, nghĩa là nó phải tốt trong việc tiên đoán được xác suất xuất hiện của từ tiếp theo. Mô hình ngôn ngữ có thể chứa những đoạn dữ liệu nhỏ hơn như từ con (subwords) hoặc các phone.
* Từ điển ngữ âm: Nó chứa sự ánh xạ từ các từ thành các phone.

Quá trình xây dựng mô hình âm học ( Acoustic model ) còn được gọi là quá trình học những đơn vị âm thanh trong cơ sở dữ liệu. Hiện nay có nhiều phương pháp xây dựng mô hình âm học ví dụ như: phương pháp kết hợp mô hình Markov và mạng Nơron, phương pháp mạng Nowrron nhiều lớp ( MLP ), thuật toán gói thời gian động (DTW), phương pháp sử dụng mô hình thống kê như mô hình Markov ẩn… Trong đó phương pháp xây dựng trên dựa trên mô hình HMM được đánh giá là có tính hiện thực cao, do mô hình còn đề cập đến yếu tố thay đổi thời gian của tiếng nói. Để phù hợp với yêu cầu tính toán của thiết bị nhúng thì phương pháp xây dựng dựa trên mô hình HMM là tối ưu nhất.

## 1.3. Những khó khăn gặp phải trong nhận dạng tiếng nói

Điều kiện lý tưởng cho việc thực hiện nhận dạng tiếng nói nói chung và âm thanh nói riêng là tiếng nói sẽ ổn định kể cả trong lúc huấn luyện và lúc nhận dạng. Tiếng nói của mỗi người là duy nhất, không trùng lẫn với những người khác. Do đó, cho đến thời điểm hiện tại, việc nhận dạng âm thanh, tiếng nói là một công việc rất khó khăn.

Việc ứng dụng hệ thống ASR cho đến thời điểm hiện nay vẫn còn nhiều hạn chế là vì một số khó khăn do bản thân đối tượng mà nó nhắm đến, trong trường hợp này là tiếng nói hay âm thanh, là một đối tượng không ổn định. Các khó khăn có thể kể đến như:

* Tiếng nói thay đổi theo thời gian, theo độ tuổi
* Tình trạng sức khỏe. Một người khi khỏe mạnh sẽ phát âm khác hẳn so với khi ốm đau, ví dụ như cảm cúm chẳng hạn
* Tốc độ nói
* Với một người, trong một khoảng thời gian ngắn, việc phát âm một từ nhiều lần khác nhau có thể khác nhau.

Bên cạnh những khó khăn trên, một hệ thống nhận dạng còn bị ảnh hưởng bởi ngoại cảnh:

* Nhiễu nền, tiếng ồn hoặc tiếng động khác từ môi trường xung quanh. Ví dụ một người nói trong môi trường yên tĩnh sẽ dễ nghe hơn là ở ngoài đường phố.
* Khoảng cách từ miệng người nói đến Microphone: Khoảng cách này càng xa thì hiệu ứng gây ra tiếng vọng càng lớn làm cho kết quả nhận dạng càng kém chính xác
* Microphone để thu âm có thể khác nhau trong những tình huống khác nhau.

## 1.4. Tác dụng của thuật toán phát hiện tiếng nói

Như đã đề cập ở phần 1.3 ngoại cảnh ảnh hưởng trực tiếp tới kết quả của hệ thống nhận dạng, thông thường tín hiệu tiếng nói khi vào mô đun nhận dạng được tiền xử lí để nâng cao chất lượng tiếng nói. Tiền xử lí có thể bao gồm các nhiệm vụ sau:

* Loại bỏ nhiễu nền môi trường ( lọc nhiễu nền ), loại bỏ nhiễu khác ( tiếng nói của người khác, tiếng động, … )
* Loại bỏ tiếng vọng
* Loại bỏ khoảng lặng, loại bỏ đi những xung ngắn ( tiếng đập bàn, đập ghế ) xen vào tín hiệu tiếng nói trong thời gian ngắn

Trong đồ án này tôi xin tập trung trình bày thuật toán phát hiện tiếng nói ( Voice Activity Detection ) để thực hiện nhiệm vụ thứ 3 trong mô đun tiền xử lí là: loại bỏ khoảng lặng, loại bỏ tiếng động trong thời gian ngắn.

Sở dĩ những khoảng lặng và những tiếng động này làm cho kết quả nhận dạng kém chính xác hơn vì nếu có chúng thì hệ thống có thể vẫn nhận dạng ra một từ hoặc một cụm từ nào đó có trong từ điển của hệ thống. Hơn nữa việc loại bỏ này sẽ tăng thời gian nhận dạng, tăng thời gian truyền nhận ( trong trường hợp kết nối thu âm ở client – nhận dạng ở server ) vì tín hiệu đầu vào đã được bỏ bớt đi những phần không cần thiết.

# Chương 2: CÁC THUẬT TOÁN PHÁT HIỆN TIẾNG NÓI

Hiện nay có nhiều thuật toán phát hiện tiếng nói trong tín hiệu âm thanh đã được triển khai, chúng dựa vào các đặc tính miền thời gian, miền tần số và đặc tính năng lượng của tín hiệu âm thanh. Sau đây tôi xin trình bày một số thuật toán hay được sử dụng

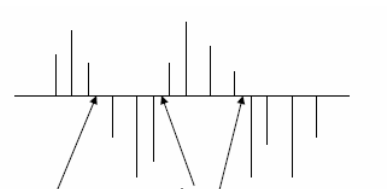
## 2.1.Thuật toán qua điểm không ( Zero crossing )

Thuật toán này xác định điểm bắt đầu, điểm kết thúc của tín hiệu tiếng nói dựa trên hai đại lượng tĩnh của tín hiệu tiếng nói là: hàm giả năng lượng E ( Pseudo-Energy ) và tỷ lệ vượt quá điểm không ZCR ( Zero Crossing Rate ).

Trong một dãy giá trị tín hiệu tiếng nói được rời rạc hóa, điểm không là điểm tại đó diễn ra sự đổi dấu cường độ tín hiệu và được mô tả bởi:

sgn[x(n+1)]  sgn[x(n)] (2.1)

Trong đó sgn(.) là hàm dấu.



Zero crossing

Hình 2-1 : Zero crossing

Năng lượng là đại lượng được dùng để xác định vùng chứa âm hữu thanh, vô thanh. Nhưng hàm năng lượng thường rất nhạy cảm với nhiễu. Do vậy, người ta thường sử dụng hàm giả năng lượng trong tính toán. Hàm giả năng lượng được xác định bởi :

 (2.2)

Trong đó

*  là hàm giả năng lượng
* N là kích thước khung cửa sổ

Tỷ lệ vượt quá điểm không ZCR. Ta thấy khung có năng lượng càng cao thì tỷ lệ vượt quá điểm không càng thấp và ngược lại. Như vậy tỷ lệ vượt quá điểm không là đại lượng đặc trưng cho tần số tín hiệu tiếng nói. Ở đây, chúng ta cần xác định tham số ngưỡng cho hàm giả năng lượng với hai ngưỡng trên và dưới và một ngưỡng tỷ lệ vượt quá điểm không.

Kí hiệu:

*  : ngưỡng năng lượng trên ( cao )
*  : ngưỡng năng lượng dưới ( thấp )
* ZCR\_T : ngưỡng tỷ lệ vượt quá điểm không.

Thuật toán này được mô tả như sau:

* Chia chuỗi tín hiệu thành các khung. Tính giá trị hàm giả năng lượng và tỷ lệ vượt quá điểm không ZCR tương ứng trên mỗi khung.
* Xét từ khung đầu tiên. Đánh dấu khung thứ i là khung bắt đầu nếu tại khung thứ i tỷ lệ vượt quá điểm không của ZCR vượt ngưỡng ( ZCR > ZCR\_T ) và giá trị hàm giả năng lượng vượt ngưỡng dưới (>) theo hướng tăng của hàm giả năng lượng.
* Xét các khung kế tiếp. Đánh dấu khung kế tiếp thuộc từ. Nếu hàm giả năng lượng vượt ngưỡng trên ( > ) theo hướng tăng của năng lượng.
* Điểm bắt đầu của từ được xác định khi hàm giả năng lượng trên khung đó nhỏ hơn ngưỡng dưới (<) và đồng thời tỷ lệ vượt quá điểm không trên khung lớn hơn ngưỡng (ZCR > ZCR\_T ).
* Điểm kết thúc từ được xác định nếu tại đó, tỷ lệ vượt quá điểm không nhỏ hơn ngưỡng (ZCR < ZCR\_T ) và hàm năng lượng tương ứng nhỏ hơn ngưỡng dưới (<) theo xu hướng đi xuống của hàm giả năng lượng.

## 2.2.Thuật toán năng lượng trong thời gian ngắn“Short time energy”

Thuật toán này có tên gọi là phát hiện tiếng nói dựa trên hàm năng lượng thời gian ngắn. Hàm năng lượng thời gian ngắn của tín hiệu tiếng nói được tính bằng cách chia tín hiệu tiếng nói thành các khung, mỗi khung dài N mẫu. Mỗi khung được nhân với một hàm cửa sổ W(n). Nếu hàm cửa sổ bắt đầu xét ở mẫu thứ m thì hàm năng lượng thời gian ngắn  được xác định như sau:

 (2.3)

Trong đó :

* n là biến rời rạc
* m là số mẫu thử thứ m
* N là tổng số mẫu tiếng nói

Hàm cửa sổ W(n) thường dùng là hàm cửa sổ hanning được xác định như sau :

 (2.4)

Phương pháp xác định điểm đầu và điểm cuối tiếng nói như sau :

* Với mỗi khung của tín hiệu, xác định hàm năng lượng thời gian ngắn  . Nếu  (giá trị ngưỡn năng lượng cho trước ) thì đánh dấu là điểm bắt đầu khung ( kí hiệu là khung B ). Ngược lại, xét tiếp khung kế tiếp cho đến khi xác định được khung B. Nếu không xác định được khung B thì kết luận : đó không là tín hiệu tiếng nói.
* Tính  của khung kế tiếp khung B cho đến khi  thì dừng và đánh dấu khung đó là điểm kết thúc của một từ ( kí hiệu khung E ). Sau khi xác định điểm bắt đầu và kết thúc, dựa vào độ dài thời gian đoạn âm thanh đó để thêm bước kiểm tra : tín hiệu đó có chắc là tiếng nói không ? ( một từ tiếng Việt rõ ràng thường dài hơn 200 ms )

## 2.3.Thuật toán LTSD ( Long-term spectral divergence )

Phương pháp này giả thiết rằng thông tin chủ yếu để phát hiện tiếng nói trong môi trường nhiễu là sự biến đổi theo thời gian của phổ biên độ tín hiệu. Phương pháp này phân tích trên khung tín hiệu dài để phát hiện ranh giới tín hiệu tiếng nói dựa trên phép ước lượng có tên là Long-term spectral envelope.

Luật quyết định việc phân loại tiếng nói hay không phải tiếng nói dựa vào độ phân kỳ của phổ tín hiệu (Long-term spectral divergence – LTSD).

Về mặt toán học LTSE và LTSD được định nghĩa như sau:

* Gọi x(n) là tín hiệu tiếng nói trong môi trường nhiễu, x(n) sẽ được phân khung trong quá trình xử lý.
* Gọi X(k,l) là phổ biên độ tại băng k của khung thứ l. Khi đó LTSE cấp N (chiều dài khung tín hiệu) tại băng k của khung tín hiệu l được định nghĩa:

  (2.5)

* LTSD cấp N của khung tín hiệu thứ l được định nghĩa:

 (2.6)

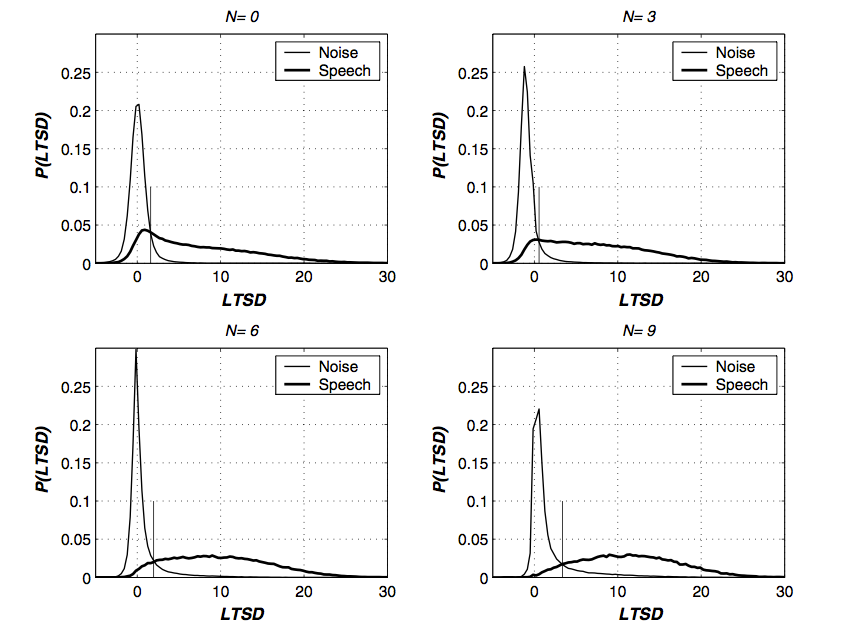
Trong đó N(k,l) là trung bình phổ năng lượng của nhiễu được ước lượng liên tục sau mỗi khung tín hiệu theo công thức:

 (2.7)

(1): if speech pause is detected

(2): otherwise

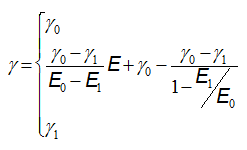
* LTSD được dùng làm đơn vị đo để phân biệt tiếng nói và nhiễu. Để làm rõ vai trò của LTSD nhóm tác giả đã khảo sát hàm phân bố LTSD của tín hiệu tiếng nói và nhiễu trên nhiều môi trường nhiễu khác nhau (trên ô tô, ngã tư đường, ...).



Hình 2-2: Phân bố LTSD theo cấp N

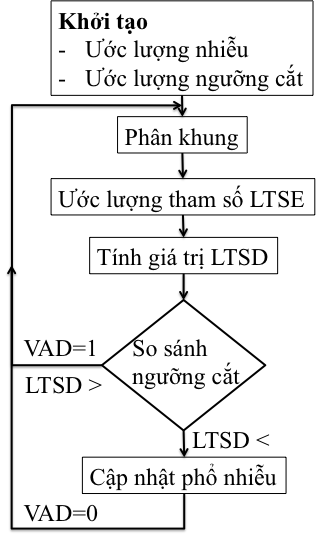
Qua kết quả trên có thể thấy rằng độ phân biệt giữa phân bố LTSD của tiếng nói và nhiễu càng cao khi N càng lớn.

Để phân loại được phần tín hiệu tiếng nói và nhiễu nền, giá trị LTSD được tính tại mỗi khung tín hiệu và so sánh với một ngưỡng xác định trước  . Giá trị được tính dựa trên ngưỡng  cho môi trường nhiễu nền có năng lượng thấp nhất và ngưỡng  cho môi trường nhiễu nền có năng lượng cao nhất . Các giá trị  ,  tìm được nhờ khảo sát trên số liệu thực tế trong hai môi trường với điều kiện đã nêu trên. Công thức toán học ước lượng như sau:

 (2.8)

Trong đó E là năng lượng của nhiễu nền tại thời điểm ước lượng.

Sơ đồ giải thuật chi tiết của thuật toán:



Hình 2-3: Sơ đồ thuật toán VAD\_LTSD

## 2.4.Nhận xét, so sánh giữa các thuật toán

Hai thuật toán đầu tiên dựa vào đặc tính năng lượng trong thời gian ngắn của tín hiệu và đặc tính trong miền thời gian. Đây là 2 thuật toán đơn giản, dễ dàng triển khai trên phần cứng và cho kết quả tốt với tín hiệu tiếng nói sạch ( ít nhiễu nền ) hoặc trong môi trường có tỷ lệ giữa năng lượng tiếng nói so với năng lượng nhiễu nền cao ( high SNR ( Signal\_Noise\_Ratio )). Tuy nhiên trong môi trường mà tỷ lệ năng lượng này thấp hơn ( < 10dB ) thì khả năng phát hiện chính xác tiếng nói hay không tiếng nói giảm rõ rệt.

Thuật toán thứ 3: LTSD sử dụng đặc tính phổ biên độ tín hiệu biến đổi theo thời gian trên khung tín hiệu dài ( long term features ). Để đánh giá tiếng nói hay không tiếng nói tại một khung cần sử dụng nhiều đặc tính của nhiều khung xung quanh trước và sau khung đang được đánh giá. Điều đó khiến kết quả chính xác hơn. Hơn nữa thuật toán này cho kết quả tốt trong môi trường có nhiều kiểu nhiễu khác nhau ( tỷ lệ SNR có thể cao hoặc thấp ) vì thuật toán cập nhật nhiễu nền thường xuyên theo từng khung.

Vì những lý do trên trong đồ án, tôi đã quyết định chọn thuật toán VAD\_LTSD để triển khai và ứng dụng vào trong một hệ thống nhận dạng lớn. Đây là một trong những thuật toán phát hiện tiếng nói tốt nhất hiện nay, nó đòi hỏi tốc độ xử lý cũng như bộ nhớ lớn của phần cứng. Sự phát triển hệ thống phần cứng của máy tính PC cùng điện thoại di động Android và máy tính bảng Android như hiện nay hoàn toàn cho phép triển khai thuật toán một cách dễ dàng.

# Chương 3: TRIỂN KHAI THUẬT TOÁN PHÁT HIỆN TIẾNG NÓI LTSD

## 3.1.Các bước triển khai thuật toán

Nội dung của thuật toán đã được trình bày chi tiết trong chương 2. Sau đây là các bước triển khai thuật toán.

Bước 1: Thu âm tín hiệu tiếng nói: Sử dụng microphone của máy tính hoặc điện thoại thu âm tín hiệu âm thanh với định dạng: tần số lấy mẫu là 16000 KHz, số bit lượng tử hóa: 16 bit, kiểu lượng tử hóa: có dấu ( signed ), số kênh : mono chanel

Bước 2: Phân khung tín hiệu:

* Sau khi nhận được tín hiệu âm thanh đã được lấy mẫu từ micro, tiến hành phân khung tín hiệu, 1 khung gồm 400 mẫu ( tương đương 25 ms tín hiệu thu âm ).
* Sử dụng phân khung overlap với 16 mẫu trùng nhau ở mỗi khung ( lặp 1ms sau mỗi khung, khoảng cách giữa 2 khung liên tiếp là 384 mẫu tương đương với 24ms )

Bước 3: Tính LTSE, LTSD, ngưỡng cắt và cập nhật năng lượng nhiễu

* Với mỗi một khung nhận được thì biến đổi FFT 512 điểm để tính 400 giá trị LTSE cho khung đó.
* Không áp dụng thuật toán cho 2 khung đầu tiên, coi 2 khung đó là không có tiếng nói. Sử dụng 500 mẫu đầu tiên để tính toán năng lượng môi trường ban đầu cho thuật toán. Từ đó tính toán được ngưỡng cắt và LTSD cho những khung ban đầu, làm cơ sở để tính toán cho những khung tiếp theo.
* Liên tục cập nhật năng lượng nhiễu nền theo công thức sau:

 (3.1)

(1): Nếu khung đó được phát hiện là tiếng nói thì năng lượng nhiễu nền bằng năng lượng nhiễu nền của khung liền trước

(2): Nếu khung đó được phát hiện là khoảng lặng thì năng lượng nhiễu nền lúc này chính là năng lượng trung bình của khung đó.

Công thức này đảm bảo năng lượng nhiễu nền luôn được cập nhật khi môi trường thay đổi ( ví dụ người nói di chuyển từ môi trường yên lặng ra môi trường ồn ào ( từ phòng họp ra đường phố ))

* Xử lí kết quả đầu ra: Sau khi tính toán được LTSD và ngưỡng cắt cho từng khung, có thể đưa ra vector đánh dấu theo từng khung ( 1 là tiếng nói, 0 là khoảng lặng ). Sử dụng vector đánh dấu đó để bỏ đi những khoảng lặng từ tín hiệu ban đầu và đưa vào khối nhận dạng, để tự ngắt thu âm và gửi lên server, để đánh giá độ chính xác của thuật toán,…

## 3.2.Platform

Sau 4 tháng thực tập và thực hiện đồ án tốt nghiệp tại viện nghiên cứu quốc tế Mica, tôi đã thử với nhiều thiết bị phần cứng và những nền tảng phần mềm. Dưới đây là những nền tảng mà thuật toán có thể triển khai:

* Phần cứng: Máy tính PC ( destop, laptop ), thiết bị Android ( điện thoại, tab )
* Phần mềm: hệ điều hành: Windows ( XP, 7, 8 ), Linux (Ubuntu), Android ( version 2.3 trở lên )

Để triển khai thuật toán trên những platform khác nhau ở trên, tôi đã sử dụng ngôn ngữ lập trình Java cho máy tính PC vì đây là một ngôn ngữ hướng đối tượng, độc lập nền. Vì vậy thuật toán sau khi đóng gói đều có thể chạy trên Windows và Linux. Với thiết bị Android tôi đã sử dụng ngôn ngữ lập trình mà hệ điều hành Android hỗ trợ với cốt lõi dựa trên ngôn ngữ Java.

## 3.3.Công việc đã thực hiện

### 3.3.1.VAD offline trên máy tính PC

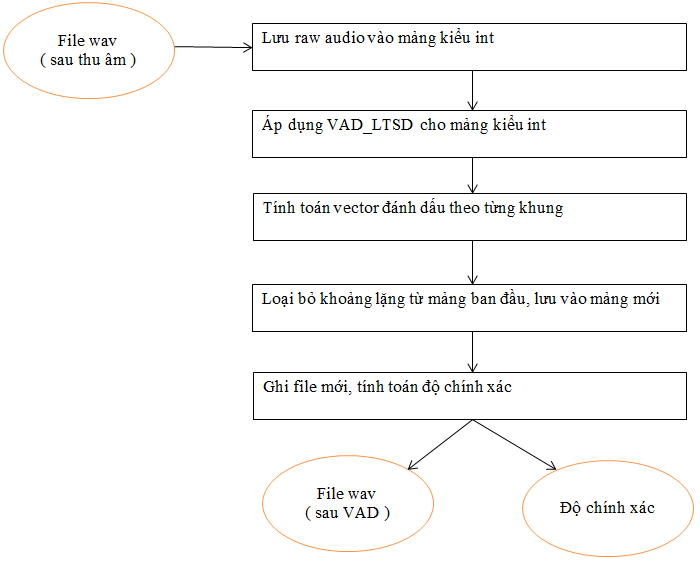
Ứng dụng của chương trình này là nhận đầu vào 1 file âm thanh có những đoạn là tiếng nói và không tiếng nói, sau khi áp dụng thuật toán VAD sẽ cho ra file âm thanh mới chỉ còn những đoạn tiếng nói đồng thời tính toán mảng đánh dấu và độ chính xác của thuật toán. Ứng dụng được miêu tả chi tiết như dưới đấy:

Đầu vào: file wav lưu giữ âm thanh đã được thu với định dạng : tần số lấy mẫu là 16 KHz, số bit lượng tử là 16 bits signed, số kênh là mono chanel

Đầu ra:

* File wav đã bỏ đi khoảng lặng
* Vector đánh dấu theo từng khung là tiếng nói hay không

Quá trình thực hiện:



Hình 3-1: Sơ đồ thực hiện VAD offline cho máy tính PC

* Lấy tín hiệu âm thanh ( raw audio) ( đã được lấy mẫu ) từ file wav ban đầu và lưu vào một mảng kiểu int
* Áp dụng thuật toán VAD trên mảng raw audio trên ( chi tiết việc áp dụng VAD đã được trình bày trong phần 2.3 “Triển khai thuật toán LTSD” )
* Tính toán vector đánh dấu tiếng nói hay không tiếng nói theo từng khung
* Từ vector đánh dấu và mảng raw audio ban đầu, loại bỏ đi những khung là khoảng lặng và giữ lại những khung là tiếng nói vào một mảng kiểu int khác
* Ghi lại file sau khi VAD từ mảng kiểu int trên và tính toán độ chính xác của thuật toán ( được trình bày chi tiết trong phần 3.4.4 )

### 3.3.2.VAD online trên máy tính PC

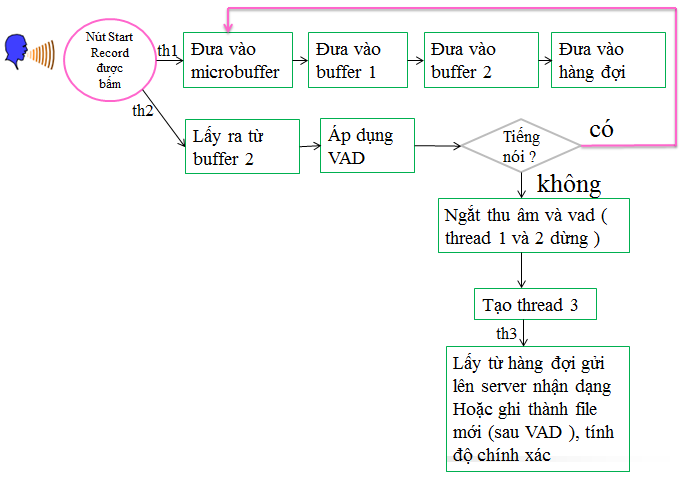
Ứng dụng của chương trình này là vừa thu âm bằng microphone của máy tính PC, vừa áp dụng thuật toán VAD với luồng âm thanh vào, khi người sử dụng không nói ( tạo ra khoảng lặng ) quá một khoảng thời gian đặt trước xác định ( từ 1,5 đến 2s ) chương trình sẽ tự động ngắt thu âm. Với luồng âm thanh nhận được cho đến lúc ngắt, chương trình sẽ thực hiện VAD rồi đưa vào hàng đợi để ghi thành file hoặc có thể gửi lên server nhận dạng. Ứng dụng được miêu tả trực tiếp như dưới đây:

Đầu vào: âm thanh thu từ microphone của máy tính PC

Đầu ra:

* File wav đã bỏ đi khoảng lặng hoặc một mảng tiếng nói đã bỏ đi khoảng lặng để gửi lên server nhận dạng
* Mảng đánh dấu theo từng khung là tiếng nói hay không.

Quá trình thực hiện



Hình 3-2: Quá trình thực hiện VAD online trên máy tính PC

* Khi bấm nút thu âm trên giao diện, chương trình tạo 2 thread:
  + Thread 1
    - Thu âm, lưu mẫu từ vùng đệm của micro vào vùng đệm chương trình
    - Lưu số mẫu vào một hàng đợi ( phục vụ cho việc gửi lên server sau này )
  + Thread 2
    - Lấy ra từ vùng đệm chương trình, sử dụng phương pháp tính của VAD để tự ngắt thu âm
* Khi ngắt thu âm ( tự ngắt tùy thuộc vào kết quả của thread 2 ) chương trình tạo thread 3:
  + VAD số mẫu từ hàng đợi rồi ghi thành file
  + Hoặc gửi lên server nhận dạng

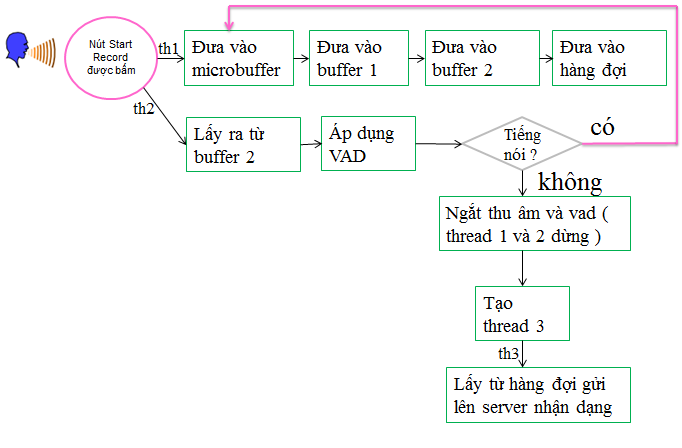
### 3.3.3.Áp dụng phương pháp VAD cho thiết bị Android

Ứng dụng của chương trình này là vừa thu âm bằng microphone của thiết bị Android ( điện thoại, tab,…), vừa áp dụng thuật toán VAD với luồng âm thanh vào, khi người sử dụng không nói ( tạo ra khoảng lặng ) quá một khoảng thời gian đặt trước xác định ( từ 1,5 đến 2s ) chương trình sẽ tự động ngắt thu âm. Với luồng âm thanh nhận được cho đến lúc ngắt, chương trình sẽ gửi lên server nhận dạng và nhận kết quả là dòng text từ server nhận dạng, sau đó hiện thị lên màn hình. Người sử dụng tiếp tục bấm nút thu âm để nói câu tiếp theo và chờ đợi kết quả nhận dạng hiển thị lên trên màn hình. Ứng dụng được miêu tả trực tiếp như dưới đây:

Đầu vào: âm thanh thu từ micro của điện thoại hoặc tab

Đầu ra: mảng tiếng nói để gửi lên server

Quá trình thực hiện:

Hình 3-3: Quá trình áp dụng VAD cho thiết bị Android

* Khi bấm nút thu âm trên màn hình, chương trình tạo 2 thread chạy song song
  + Thread 1
    - Thu âm, đưa tín hiệu âm thanh vào vùng đệm của micro ( microbuffer ),
    - Đưa vào vùng đệm chương trình ( buffer 1 ),
    - Đưa vào buffer 2 để thread 2 có thể lấy ra và áp dụng VAD ( đây là vùng đệm sử dụng cấu trúc tránh tương tranh được java hỗ trợ ( sử dụng cấu trúc Array Blocking Queue với kích thước là 8000, chỉ đưa vào buffer 2 nếu buffer 2 chưa đầy và chỉ lấy ra được khi buffer không trống )
    - Đưa vào hàng đợi để khi ngắt thu âm sẽ gửi lên server nhận dạng
  + Thread 2
    - Lấy ra từ buffer 2 mỗi lần 8000 mẫu
    - Áp dụng thuật toán VAD để tính toán xem ngắt hay không ngắt thu âm
* Khi ngắt thu âm ( tự ngắt tùy thuộc vào kết quả của thread 2 ) chương trình sẽ lấy số mẫu từ hàng đợi lưu sang một mảng kiểu short và gửi lên server nhận dạng sau đó đợi kết quả từ server nhận dạng.
* Khi nhận được kết quả từ server nhận dạng chương trình sẽ hiển thị kết quả trên giao diện của main thread UI
* Sau khi hiển thị kết quả, nút ghi âm lại được enable. Người dùng tiếp tục bấm nút ghi âm để thực hiện lần tiếp theo: nói và chờ đợi kết quả nhận dạng

### 3.3.4.VAD trên server nhận dạng

Ứng dụng của chương trình này là đưa vào trước khối nhận dạng trên server một mô đun độc lập để thực hiện VAD giúp bỏ đi khoảng lặng và những xung âm thanh không phải là tiếng nói để kết quả nhận dạng chính xác hơn và tốc độ nhận dạng nhanh hơn.. Việc ghép nối sẽ được miêu tả chi tiết trong chương 4. Dưới đây là những đặc trưng của ứng dụng này:

* Mô đun này có thể có hoặc không tùy thuộc vào yêu cầu hay người lập trình
* Độc lập với Sphinx 4 trên server
* Đứng trước module nhận dạng Sphinx 4
* Đầu vào, đầu ra, quá trình thực hiện tương tự như phần 3.3.1 ( VAD offline trên PC )
  + Đầu vào : file wav âm thanh ban đầu
  + Đầu ra: file wav chỉ còn những đoạn tiếng nói được nối liền lại với nhau
  + Thực hiện: Lấy số mẫu từ file wav ban đầu và đưa vào một mảng, áp dụng thuật toán VAD với mảng đó và ghi lại thành file wav mới.

## 3.4.Đánh giá độ chính xác của thuật toán

Như đã đề cập ở phần 3.3 thuật toán VAD\_LTSD có nhiều ứng dụng khác nhau. Một vấn đề đặt ra là độ chính xác của thuật toán được đánh giá như thế nào ? Làm thế nào để biết được khả năng phát hiện tiếng nói, không tiếng nói, sự sai sót, nhầm lẫn giữa những khoảng tiếng nói và không tiếng nói sau khi áp dụng thuật toán so với tín hiệu âm thanh ban đầu.

Trong khuôn khổ đồ án tôi đã chọn đánh giá thuật toán với trường hợp offline cho máy tính PC vì những điều kiện thuận lợi về phần cứng, phần mềm và đặc biệt là việc lập trình java trên PC cho phép dễ dàng kiểm tra kết quả.

Để có thể đánh giá được độ chính xác của thuật toán, yêu cầu cần có những điều kiện sau

* Có file thu âm ban đầu, có phương pháp để quan sát và gán nhãn tín hiệu đó để làm cơ sở so sánh.
* Tạo ra được mảng đánh dấu ban đầu
* Tạo ra được mảng đánh dấu của file sau khi áp dụng thuật toán
* Có phương pháp để định lượng được độ chính xác

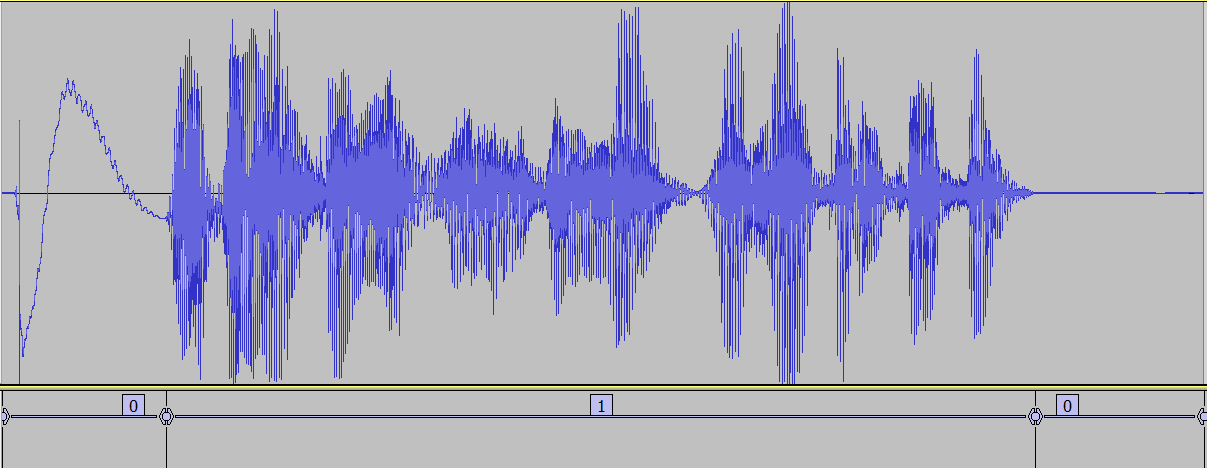
Những vấn đề và yêu cầu trên được trình bày chi tiết trong những phần dưới đây

### 3.4.1.Các bước thực hiện

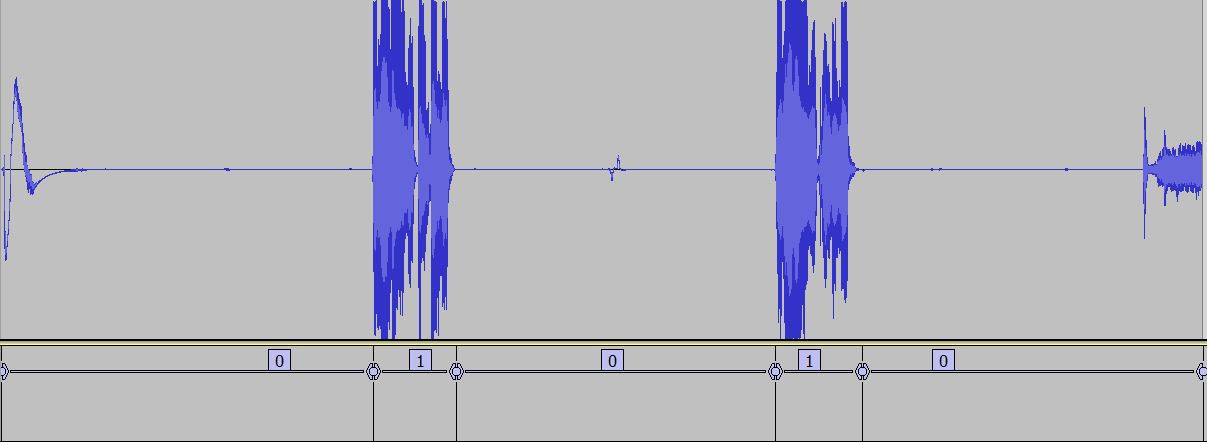
Thu âm bằng một chương trình viết bằng java trên PC

Quan sát phổ tín hiệu ban đầu và gán nhãn bằng phần mềm Audacity

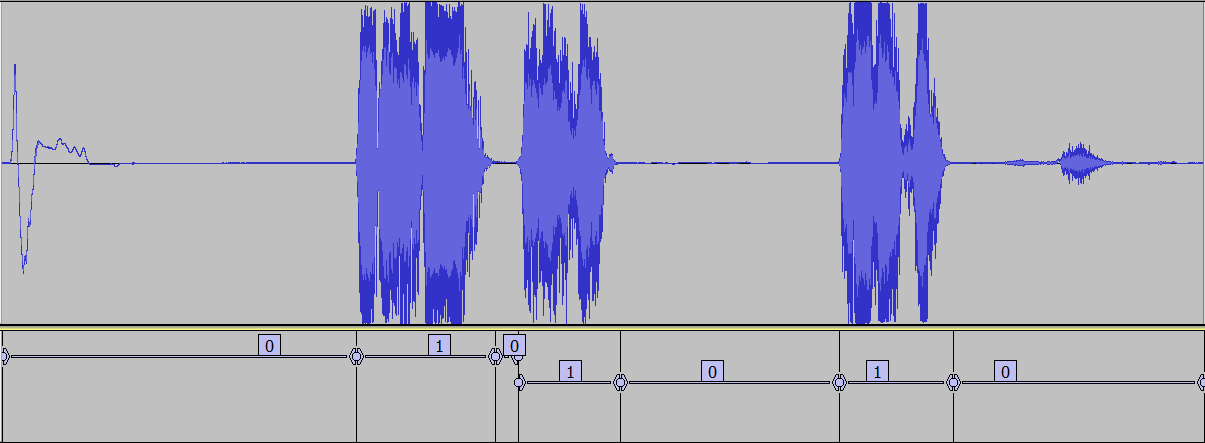
* 1: tiếng nói
* 0: không có tiếng nói



Hình 3-4: Gán nhãn cho file thu âm nói liên tục



Hình 3-5:Gán nhãn cho file thu âm nói ngắt quãng

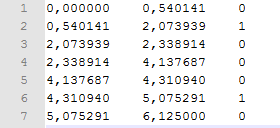


Hình 3-5: Gán nhãn cho file thu âm nói ngắt quãng trong môi trường yên tĩnh

Sau gán nhãn sẽ có file txt là đầu vào một chương trình java khác để tạo một mảng đánh dấu ban đầu



Hình 3-6: File txt có được sau khi gán nhãn



Hình 3-7:File .txt sau khi gán nhãn file thu âm nói ngắt quãng

Áp dụng thuật toán VAD ban đầu sẽ tạo ra mảng đánh dấu lúc sau

Từ 2 mảng trên tính toán được các thông số đo độ chính xác bằng một chương trình khác trên java

Sau VAD thì ghi lại file wav mới ( bỏ đi khoảng lặng ) để thuận tiện cho việc kiểm tra và đưa ra bảng kết quả

### 3.4.2.Bộ thông số đầu vào của thuật toán VAD\_LTSD

Bộ thông số đầu vào được tính toán trước dựa vào những điều kiện thực tế. Bộ thông số này bao gồm:

* E0: Năng lượng nhiễu nền trong môi trường ít nhiễu nhất
* E1: Năng lượng nhiễu nền trong môi trường nhiều nhiễu nhất
* Threshold\_0: ngưỡng cắt tương ứng với môi trường ít nhiễu nhất
* Threshold\_1: ngưỡng cắt tương ứng với môi trường nhiều nhiễu nhất
* Số mấu trên một khung sau khi phân khung
* Overlap: khoảng cách giữu 2 khung liên tiếp
* Anpha: hệ số cập nhật nhiễu môi trường
* Số khung liền kể sử dụng

Sau đây là giá trị của các thông số trên

* E0: (tính theo 10 \* log10 ( năng lượng môi trường ít nhiễu nhất )) 6.02
* E1: (tính theo 10 \* log10 ( năng lượng môi trường nhiều nhiễu nhất )) 85
* Threshold\_0: 35
* Threshold\_1: 20
* Phân khung: số mẫu trên 1 khung: 400
* Overlap: khoảng cách giữa 2 khung liên tiếp: 384 mẫu
* Anpha: 0,7
* Số khung liền kề sử dụng: 9

### 3.4.3.Sử dụng phương pháp ROC tính toán độ chính xác

Sử dụng phương pháp ROC ( Receiver operating characteristic ) để đánh giá độ chính xác của thuật toán.

Tính toán các thông số cho độ chính xác của thuật toán ( tính bằng 1 chương trình viết trên java )

* Tpr: true positive rate
* Fpr : false positive rate
* Acc: accuracy ( độ chính xác )

### 3.4.4.Kết quả đánh giá

Với 14 file wav thu âm được qua micro của laptop, kết quả đánh giá của thuật toán VAD như sau:

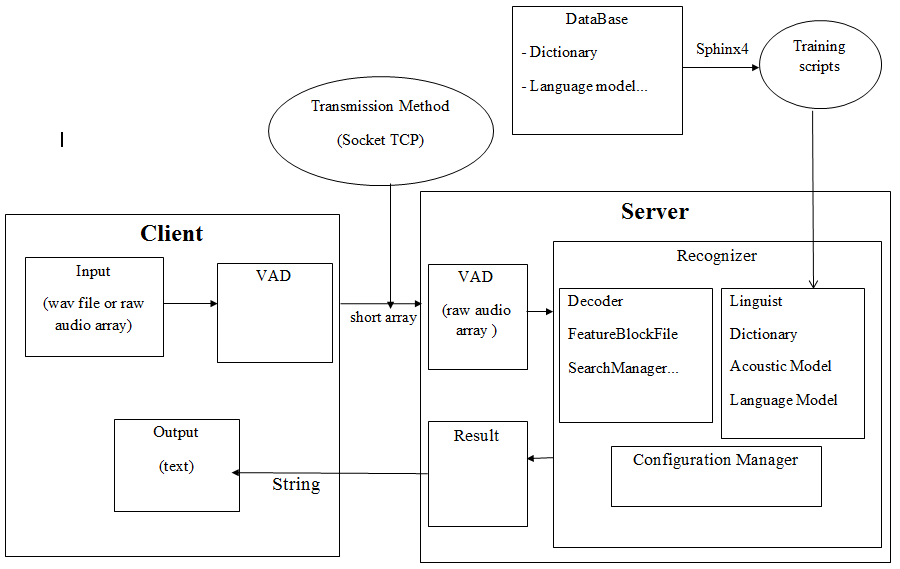
|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| STT | Tên file (.wav) | Thời lượng (s ) | TPR | FPR | ACC ( % ) |
| 1 | hien1 | 12 | 0.99 | 0.10 | 92 |
| 2 | hien2\_ko tieng noi | 17 | NaN | 0.09 | 90 |
| 3 | hien3\_noi lien tuc | 4 | 0.83 | 0.30 | 79 |
| 4 | hien4 | 19 | 0.94 | 0.04 | 95 |
| 5 | hien5\_tieng noi ngat quang | 22 | 1.00 | 0.06 | 94 |
| 6 | hien6\_noi lien tuc | 3 | 0.89 | 0.27 | 84 |
| 7 |  |  |  |  |  |
| 8 | hien8\_moi truong yen tinh | 5 | NaN | 0.09 | 90 |
| 9 | hien9\_tieng noi trong moi truong nhieu oto | 20 | 0.90 | 0.21 | 81 |
| 10 | hien10 | 21 | 0.95 | 0.08 | 91 |
| 11 | hien11 | 12 | 0.81 | 0.13 | 84 |
| 12 | hien12 | 14 | 0.88 | 0.06 | 91 |
| 13 | hien13 | 5 | 0.96 | 0.14 | 89 |
| 14 | hien14\_noi ngat quang | 7 | 0.96 | 0.06 | 94 |

\*:NaN: “ không xác định” vì chia cho 0 ( tướng ứng với trường hợp có tiếng nói: P = 0 )

Bảng 3-1: Kết quả đánh giá độ chính xác VAD\_LTSD

# Chương 4: KẾT NỐI VỚI MÔ ĐUN NHẬN DẠNG

## 4.1.Sơ đồ chung



Hình 4-1: Sơ đồ kết nối client-server của hệ thống nhận dạng

## 4.2.Cách thức hoạt động

Môi trường truyền là mạng: Internet ( mạng dây, 3G ), LAN ( mạng dây, wifi )

Kiểu kết nối: Multi Client – Server

* Client gửi cho server một mảng kiểu short là tín hiệu âm thanh đã được lấy mẫu
* Server nhận dạng và trả lại kết quả dạng text

Phương thức truyền : Socket TCP

Cách thức hoạt động của cả hệ thống Multi client – server nhận dạng tiếng nói được mô tả như sau :

Sau khi thu âm ở client, client sẽ tự động ngắt thu âm và mở kết nối socket với server, số mẫu tiếng nói ban đầu được ép kiểu từ int thành short và được lưu vào một mảng short và được gửi toàn bộ lên server để tăng tốc độ gửi.

Server khi chạy luôn mở socket để lắng nghe và kết nối khi có một client muốn kết nối. Với mỗi một client đến server tạo một thread riêng để đáp ứng với client đó :

* Nhận mảng short clien gửi lên
* Chuyển mảng short thành mảng int
* VAD mảng int để bỏ đi khoảng lặng và những xung âm thanh ngắn
* Ghi thành file âm thanh định dạng wav làm đầu vào cho khối nhận dạng
* Nhận dạng file âm thanh đó và trả ra kết quả là một string
* Gửi lại string xuống client
* Đóng kết nối với client tương ứng và kết thúc thread

Vì mỗi thread trên server được đặc trưng bởi 1 socket riêng tương ứng với client nên server luôn đảm bảo việc nhận đúng và trả đúng client

Server khi khởi động sẽ tạo song song 3 bộ regconizer tương ứng là 3 bộ decoder, trong mỗi bộ regconizer đó đều có mô đun VAD và mô đun nhận dạng riếng sử dụng Sphinx 4. Ba yêu cầu đầu tiên sẽ lần lượt được phân vào 3 bộ recognizer theo thứ tự 1, 2, 3. Mỗi khi một regconizer nhận yêu cầu từ một client nó sẽ bật cờ báo bận, sau khi nhận dạng xong sẽ bật cờ báo rỗi, với một recognizer sẽ có một biến thời gian đi kèm để đo thời gian rảnh rỗi. Kể từ yêu cầu thứ 4 đến từ client, dữ liệu âm thanh nhận được từ client sẽ được phân vào bộ regconizer đang rảnh rỗi và có thời gian rảnh rỗi lớn nhất.

## 4.3.Các vấn đề gặp phải và giải pháp

Tiết kiệm tài nguyên sử dụng của server : 1 bộ recognize cho tất cả các client → thời gian chờ đợi của client lớn.

Giảm thời gian chờ đợi của client : tạo ra nhiều bộ reconize hoạt động song song, chung dữ liệu → có thể xảy ra tình trạng lãng phí tài nguyên (có những bộ nhận dạng khởi tạo nhưng không cần đến)→ Cần cân bằng giữa 2 tiêu chí.

Tính toán sơ bộ :

* Trong 10s có khoảng 100 client kết nối → có 100 file âm thanh chuyển đến.
* Thời gian nhận dạng 1 file ≈ 0,5s → tổng thời gian nhận dạng ≈ 50s.
* Để đáp ứng kịp trong 10s cần số bộ nhận dạng = 50 : 10 = 5 bộ

## 4.4.Đánh giá tốc độ gửi nhận client-server

Tốc độ gửi nhận client – server phụ thuộc vào

* Môi trường truyền : Lan, 3g, Internet
* Dung lượng dữ liệu: dữ liệu càng lớn thì càng mất nhiều thời gian để gửi

Dưới đây là bảng đánh giá kết quả

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | Lan(mica) | Lan(Connectify) | 3G | Internet |
| Không nói(2s thu âm) | ~ 800 ms | ~ 600 ms |  | ~ 600 ms |
| 10 từ ( 5s thu âm ) | ~ 1200 ms | ~ 1100 ms |  | ~ 1200 ms |
| 22 từ ( 9s thu âm ) | ~ 2000 ms | ~ 1800 ms |  | ~ 3000 ms |

Bảng 4-1: Tốc độ gửi nhận client-server

# Chương 5: KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN

## 5.1.Kết luận

Với 6 tuần thực tập tại viện nghiên cứu Mica và 15 tuần làm đồ án dưới sự hướng dẫn của tiến sĩ Nguyễn Quốc Cường cùng sự giúp đỡ nhiệt tình của các thầy cô, anh chị, bạn bè tại viện nghiên cứu Mica, đồ án đã hoàn thành những nhiệm vụ sau:

* Tìm hiểu chung về hệ thống nhận dạng tiếng nói và đưa ra những khó khăn gặp phải trong một hệ thống nhận dạng tiếng nói
* Tìm hiểu được những thuật toán phát hiện tiếng nói tiêu biểu từ đơn giản đến phức tạp
* Áp dụng một trong những thuật toán tốt nhất hiện nay là VAD\_LTSD để loại bỏ khoảng lặng, loại bỏ những xung ngắn không là tiếng nói trên máy tính PC và thiết bị Android giúp kết quả nhận dạng chính xác hơn và tăng tốc độ truyền nhận, tốc độ nhận dạng.
* Đưa ra được phương pháp cập nhật nhiễu nền môi trường giúp thuật toán thích nghi tốt hơn với sự thay đổi môi trường.
* Đánh giá độ chính xác của thuật toán, trung bình là 90%, đây là kết quả tốt, chấp nhận được đối với mô đun phát hiện tiếng nói của hệ thống nhận dạng.
* Kết nối được 2 mô đun độc lập: tiền xử lí tại client và nhận dạng trên server với cách thức multiclients-server. Từ đó cho phép phát triển và nâng cao chất lượng của từng mô đun mà vẫn đảm bảo tính kết nối của cả hệ thống.

Tuy nhiên với khoảng thời gian eo hẹp cùng với vốn kiến thức có hạn đồ án vẫn tồn tại những hạn chế sau:

* Đôi khi vẫn phát hiện nhâm giữa tiếng nói và không tiếng nói
* Chưa có cơ chế kiểm tra việc truyền nhận đầy đủ và chính xác từ client đến server
* Việc truyền nhận client-server đôi khi còn chậm ( phụ thuộc nhiều vào mạng )

Sở dĩ tồn tại những hạn chế trên, chủ yếu là do những nguyên nhân sau:

* Bộ thông số đầu vào của thuật toán phát hiện tiếng nói ( gồm có: năng lượng nhiễu nền của môi trường ít nhiễu nhất: , năng lượng nhiễu nền của môi trường nhiều nhiễu nhất: , ngưỡng cắt trong môi trường ít nhiễu nhất là Threshold\_0 và ngưỡng cắt trong môi trường nhiều nhiễu nhất là Threshold\_1 ) chưa tối ưu.
* Bản chất của thuât toán VAD\_LTSD.
* Cách truyền nhận chưa tối ưu, chưa nén được dữ liệu.

## 5.2.Hướng phát triển

Với những hạn chế và nguyên nhân được chỉ ra rõ ràng trong phần 5.1, trong tương lai đồ án sẽ được bổ sung và hoàn thiện. Điều đó có thể đạt được nếu như những nhiệm vụ sau được hoàn thành:

* Tìm hiểu và triển khai một số thuật toán tốt hơn
* Tính toán chính xác bộ thông số đầu vào để có được bộ thông số tối ưu. Điều này đòi hỏi phải xác định đúng và tìm đến môi trượng ít nhiễu nhất và nhiều nhiễu nhất để tiến hành thu âm.
* Tìm hiểu một số cách truyền nhận khác
  + Ghi thành file ở client và truyền cả file lên server
  + Tìm cách nén dữ liệu để gửi và không làm mất mát thông tin
* Ghép thêm khối loại bỏ tiếng vọng, khối lọc nhiễu để hoàn thiện mô đun tiền xử lí tại client giúp hệ thống nhận dạng chính xác hơn.

# Tài liệu tham khảo

[1] Separation of Voiced and Unvoiced using Zero crossing rate and Energy of the Speech Signal - Bachu R.G., Kopparthi S., Adapa B., Barkana B.D. - Electrical Engineering Department School of Engineering, University of Bridgeport

[2] Introduction Basics of signal processing-Pr. Eric Castelli-January 2013

[3] Analysis of Speech signal – Pr.Eric Castelli-February, 2013

[4] Noise Robust Voice Activity Detection-Pham Chau Khoa-School of Computer Engineering – Nanyang Technological University

[5] A New Adaptive Long-Term Spectral Estimation Voice Activity Detector-Javier Ram´ırez, Jos´e C. Segura, Carmen Ben´ıtez, Angel de la Torre, Antonio Rubio-Department of Electronics and Computer Technology-University of Granada, Spain 2003