## Tóm tắt nội dung

Tiếng nói là công cụ giao tiếp vô cùng hiệu quả và không thể thiếu của con người. Ngày nay, với ý tưởng mở rộng việc giao tiếp người với máy móc qua tiếng nói thay vì những thiết bị đầu vào phức tạp và không dễ nhớ, rất nhiều nhà nghiên cứu khoa học đã đầu tư công sức vào việc xây dựng những hệ thống nhận dạng tiếng nói tự động cho nhiều kiểu giọng nói và nhiều ngôn ngữ. Đặc điểm chung của các hệ nhận dạng này là đều bắt đầu bằng quá trình tìm hiểu và mô phỏng các đặc điểm của tiếng nói, hay còn gọi là quá trình "trích chọn đặc trưng". Công việc này đặt nền tảng quan trọng cho việc áp dụng các phương pháp nhận dạng và quyết định tới tính chính xác của toàn hệ thống.

Tiếp tục những nghiên cứu trên, khóa luận này tìm hiểu những đặc điểm của tiếng nói nói chung và tiếng nói tiếng Việt nói riêng với mục đích kết xuất được các đặc trưng tiếng nói tiếng Việt dưới dạng số thực cho quá trình nhận dạng. Đồng thời áp dụng mô hình thống kê HMM để nhận dạng sử dụng phương pháp phân biệt thanh điệu để có kết quả kiểm chứng mức độ chính xác của quá trình trích chọn đặc trưng và hướng tới ứng dụng.

## Lời cảm ơn

Đầu tiên, tôi xin chân thành cảm ơn tiến sĩ Lê Anh Cường, đồng cảm ơn tiến sĩ Lê Sỹ Vinh hiện cùng đang công tác tại bộ môn Khoa Học Máy Tính - khoa Công nghệ Thông Tin - trường Đại Học Công Nghệ - Đại Học Quốc Gia Hà Nội, hai thầy hướng dẫn trực tiếp và cùng hướng dẫn tôi hoàn thành khóa luận này. Nhờ sự động viên giúp đỡ nhiệt tình cùng những lời khuyên bổ ích, những ý tưởng sáng tạo của hai thầy trong quá trình hướng dẫn đã giúp tôi hoàn thành khóa luận này một cách tốt nhất.

Tiếp theo tôi xin dành lời cảm ơn tới PGS.TS Lương Chi Mai và anh Vũ Tất Thắng hiện đang công tác ở Viện Khoa Học và Công Nghệ Việt Nam, hai người đã giúp đỡ và cho tôi rất nhiều lời khuyên cũng như kinh nghiệm hữu ích khi gặp những khó khăn, bế tắc trong quá trình hoàn thành khóa luận.

Ngoài ra, xin gửi lời cảm ơn tới người bạn cùng nhóm nghiên cứu: Đàm Tiến Dũng, người đã cùng sát cánh, giúp đỡ và động viên tôi rất nhiều từ khi bắt đầu tới khi hoàn thành. Đồng cảm ơn tới các bạn cùng lớp và các anh chị học khóa trên với những chia sẻ và kinh nghiệm bổ ích.

Cuối cùng tôi xin gửi lời cảm ơn tới gia đình tôi, ba và mẹ là những người luôn ủng hộ và là chỗ dựa tinh thần vững chắc cho những năm học đại học nói chung và việc hoàn thành khóa luận cuối khóa nói riêng.

## Mục lục

# Danh mục hình minh họa

# *Chương 1.* MỞ ĐẦU

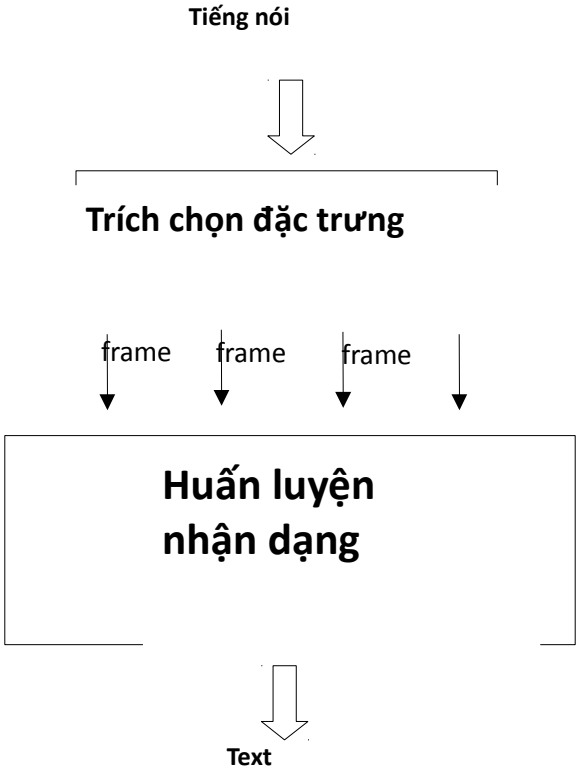
Chương đầu tiên dành để đặt vấn đề về đề tài nhận dạng tiếng nói nói chung và nhận dạng tiếng nói tiếng Việt nói riêng qua đó xác định tầm quan trọng của việc tìm hiểu đặc trưng của tiếng nói đối với hệ thống nhận dạng. Đồng thời, chương này chỉ ra những nghiên cứu hiện tại và hướng nghiên cứu sử dụng cũng như mục tiêu của khóa luận đối với đề tài này.

## 1.1. **Đặt vấn đề**

Một trong những mục đích và xu hướng quan trọng nhất của khoa học ngày nay là hướng tới việc tự động hóa các công việc chân tay, và thực tế máy móc với hiệu suất và tốc độ vượt trội đã thực sự thay thế sức lao động của con người trong rất nhiều lĩnh vực như điều khiển máy, chế tạo linh kiện, vật liệu… Một cách tự nhiên, điều này nảy sinh ra nhu cầu giao tiếp giữa con người với máy móc, khi việc giao tiếp bình thường thông qua các thiết bị đầu vào dần trở nên phức tạp như chính những cỗ máy đó. Trong nhiều năm qua, việc nghiên cứu và xây dựng hệ thống nhận dạng tiếng nói phục vụ giao tiếp người – máy đã được rất nhiều nhà nghiên cứu khoa học trên thế giới đầu tư thời gian công sức và đạt được nhiều kết quả khả quan. (VD: Framework nhận dạng tiếng nói Sphinx4, xây dựng bởi công ty Sun, đã nhận dạng được những câu nằm trong tập từ điển khoảng 65.000 từ) *[8]*

Đặc điểm chung của các hệ thống nhận dạng tiếng nói, dù sử dụng phương pháp nào, đó là trước hết phải số hóa tín hiệu tiếng nói để máy tính có thể hiểu được, qua đó tìm được những đặc trưng riêng của tiếng nói so với những đặc trưng của các âm thanh khác như nhạc cụ, tiếng ồn… Chính vì vậy việc trích chọn đặc trưng có thể nói là vấn đề quan trọng hàng đầu cho một hệ thống nhận dạng tiếng nói, trích chọn được các đặc trưng càng chính xác, độ chính xác trong việc nhận dạng của hệ thống càng cao, điều này hoàn toàn đúng với tất cả tiếng nói của mọi ngôn ngữ, trong đó bao gồm cả tiếng nói tiếng Việt.

Hình sau đây chỉ ra vị trí của quá trình trích chọn đặc trưng trong hệ thống nhận dạng tiếng nói bất kì:




*Hình 1 : Vị trí của việc trích chọn đặc trưng trong hệ thống nhận dạng tiếng nói*

## 1.2. Hướng nghiên cứu và phương pháp sử dụng

Có ba phương pháp chính, đều khá hiệu quả để tìm hiểu được đặc trưng của tiếng nói, thứ nhất đó là mô phỏng lại quá trình phát ra tiếng nói của bộ máy phát âm (bao gồm các bộ phận bên trong miệng, mũi), thứ hai là mô phỏng quá trình thu nhận âm thanh của bộ máy cảm nhận âm thanh (các bộ phận bên trong tai) và phương pháp phân tích phổ - tổng hợp của hai phương pháp trên. Hiện nay phương pháp chọn đặc trưng mô phỏng quá trình phát âm và thu nhận âm là phương pháp "*mã hóa dự đoán tuyến tính*" – LPC (Linear predictive coding) *[6][9]* và phương pháp lấy "*hàm biên độ trung bình*" – AMDF *[12]* (Average magnitude different function) mô tả sự cảm nhận cao độ âm thanh của tai, tuy

nhiên hạn chế của hai phương pháp này thể hiện ở việc kết quả nhận dạng còn chưa thật cao. *[12]*

Trong khóa luận này, ta sẽ đề cập tới kĩ thuật trích chọn đặc trưng MFCC *[2]* kết hợp của 2 phương pháp trên để trích chọn đặc trưng tiếng nói tiếng Việt, và sử dụng lại phương pháp AMDF *[9]* để trích chọn đặc trưng đặc thù của tiếng Việt là thanh điệu. Trích chọn đặc trưng MFCC được xem là một phương pháp rất hiệu quả và được áp dụng trong nhiều hệ nhận dạng nổi tiếng như Sphinx của công ty Sun.

Sử dụng kết quả trích chọn đặc trưng, ta áp dụng một phương pháp nhận dạng rất hiệu quả là dùng mô hình HMM *[5]* để huấn luyện và nhận dạng tiếng nói, sử dụng đặc trưng thanh điệu để phân biệt thanh điệu tiếng Việt cho mỗi tín hiệu âm thanh ban đầu.

## 1.3. **Giới hạn và mục tiêu của đề tài**

Mục tiêu của việc tìm hiểu đặc trưng trong tiếng nói hướng tới việc xây dựng một hệ thống nhận dạng tiếng nói tiếng Việt với độ chính xác cao, tuy nhiên trong phạm vi thời gian và khuôn khổ của một khóa luận cử nhân Công Nghệ Thông tin, tôi giới hạn nội dung nghiên cứu trong những vấn đề dưới đây:

Thứ nhất, tuy việc trích chọn đặc trưng MFCC và AMDF được áp dụng cho cả hệ nhận dạng tiếng nói liên tục (tiếng nói được nói theo câu) và hệ nhận dạng rời rạc (nói từng từ riêng biệt), nhưng hệ thống mà khóa luận này xây dựng là hệ nhận dạng rời rạc, với bộ từ điển là bộ chữ số đếm tiếng Việt (KHONG, MOT, HAI, BA, BON, NAM, SAU, BAY, TAM, CHIN), và phân biệt thanh điệu không phụ thuộc từ điển.

Thứ hai, hệ nhận dạng của chúng tôi (bao gồm cả người cùng nhóm nghiên cứu) xây dựng là "Phụ thuộc người nói", do chưa có điều kiện thu âm để huấn luyện và kiểm thử với nhiều kiểu giọng nói nên không thể coi hệ thống xây dựng là "Không phụ thuộc người nói" được. Hệ thống xây dựng được sẽ chỉ huấn luyện và nhận dạng với giọng nói của một người.

Từ việc xác định mục tiêu rõ ràng của mình, chúng tôi định hướng trong tương lai sẽ nghiên cứu sâu hơn về các kĩ thuật trích chọn đặc trưng cũng như kĩ thuật nhận dạng để mở rộng bộ từ vựng nhận dạng, hướng vào các ứng dụng giao tiếp người máy, điều khiển máy bằng giọng nói và các ứng dụng khác trong giao tiếp truyền thông…

# *Chương 2.* KỸ THUẬT TRÍCH CHỌN ĐẶC TRƯNG MFCC TRONG NHẬN DẠNG TIẾNG NÓI [1]

## 2.1. XỬ LÝ TÍN HIỆU ÂM THANH VÀ TRÍCH CHỌN ĐẶC TRƯNG

Tín hiệu âm thanh ngoài đời thực là tín hiệu liên tục, hay tín hiệu tương tự. Trước khi thực hiện bất cứ bước xử lý nào, tín hiệu âm thanh cần được số hóa. Việc này được thực hiện tự động bởi các thiết bị thu âm, bằng cách lấy mẫu tín hiệu đầu vào *[1]*. Như vậy, một tín hiệu âm thanh bất kỳ khi đã được đưa vào máy tính, là một tập các mẫu liên tiếp nhau, mỗi mẫu là giá trị biên độ của tín hiệu tại một thời điểm nhất định. Một tham số quan trọng trong việc lấy mẫu tín hiệu âm thanh là tần số lấy mẫu, *Fs*, tức là số mẫu được lấy trong một giây. Để có thể đo lường chính xác, cần phải lấy ít nhất 2 mẫu trong một chu kỳ của tín hiệu tương tự đầu vào. Như vậy, tần số lấy mẫu phải lớn hơn 2 lần tần số cao nhất của tín hiệu âm thanh đầu vào. Tuy nhiên, trên thực tế tai người chỉ có thể nhận biết được các âm thanh có tần số nhỏ hơn 10.000Hz *[12][3]*, do đó tần số lấy mẫu là 20.000Hz là đủ cho việc nhận dạng với độ chính xác rất cao. Trong lĩnh vực nhận dạng tiếng nói qua điện thoại, tần số lẫy mẫu chỉ cần là 8.000Hz vì chỉ có các tín hiệu có tần số nhỏ hơn 4.000Hz được truyền đi bởi điện thoại *[10]*. Các thiết bị thu âm thì thường có tần số lấy mẫu là 16.000Hz *[3]* .

Trích chọn đặc trưng đối với nhận dạng tiếng nói là việc tham số hóa chuỗi tín hiệu âm thanh dạng sóng đầu vào, biến đổi tín hiệu âm thanh thành một chuỗi các vector đặc trưng n chiều, mỗi chiều là một giá trị thực. Hiện nay, có rất nhiều phương pháp trích chọn đặc trưng như: LPC(Linear predictive coding – Dự đoán tuyến tính *[6][9]*), AMDF(Average magnitude different function – hàm biên độ trung bình), MFCC(Mel-frequency cepstral coefficients), hoặc kết hợp của các phương pháp trên *[12]*. Phần tiếp theo sẽ giới thiệu cụ thể về phương pháp trích chọn đặc trưng MFCC.

Trong bài toán nhận dạng tiếng nói đang xét, với tần số lấy mẫu mặc định 16.000Hz, một đoạn mẫu với một số lượng nhất định tạo thành một frame, như vậy tín hiệu tiếng nói

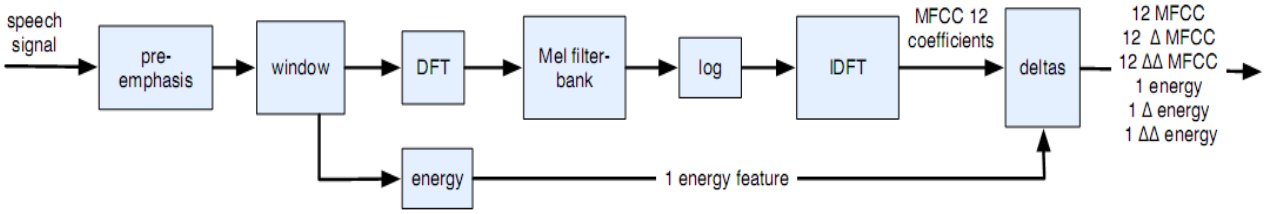
1 Đồng nghiên cứu và có phần nội dung chung với khóa luận *"Các kĩ thuật nhận dạng tiếng nói", 2010* của sinh viên Đàm Tiến Dũng – Đại học công nghệ.

là tập các frame liên tiếp nhau, trích chọn đặc trưng MFCC cho ta tập đặc trưng cho mỗi frame tiếng nói này. Tại sao phải chia thành các frame và các frame cụ thể chúng có đặc trưng thế nào, ta sẽ đề cập tới ở ngay phần sau đây.

## 2.2. TRÍCH CHỌN ĐẶC TRƯNG MFCC [2]

Trong nhận dạng tiếng nói, kỹ thuật trích chọn đặc trưng MFCC là phương pháp phổ biến nhất. MFCC là viết tắt của Mel-frequency cepstral coefficients. Kỹ thuật này dựa trên việc thực hiện biến đổi để chuyển dữ liệu âm thanh đầu vào (đã được biến đổi Fourier cho phổ) về thang đo tần số Mel, một thang đo diễn tả tốt hơn sự nhạy cảm của tai người đối với âm thanh. Kỹ thuật trích chọn đặc trưng này gồm các bước biến đổi liên tiếp, trong đó đầu ra của bước biến đổi trước sẽ là đầu vào của bước biến đổi sau. Đầu vào của quá trình trích chọn đặc trưng này sẽ là một đoạn tín hiệu tiếng nói. Vì tín hiệu âm thanh sau khi được đưa vào máy tính đã được rời rạc hóa nên đoạn tín hiệu tiếng nói này bao gồm các mẫu liên tiếp nhau, mỗi mẫu là một giá trị thực, thể hiện giá trị biên độ của âm thanh tại 1 thời điểm.

Trích chọn đặc trưng MFCC gồm sáu bước như trong hình vẽ sau, kết quả là một tập gồm 39 giá trị đặc trưng cho mỗi một frame tiếng nói.



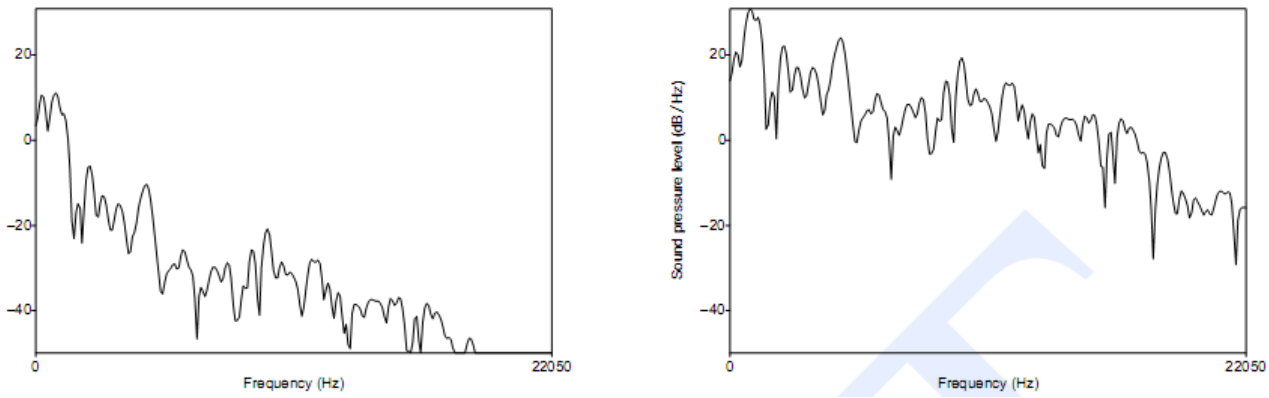

*Hình 2 (nguồn [2]): Sơ đồ quá trình trích chọn đặc trưng MFCC*

### 2.2.1. Pre-emphasis

Tín hiệu âm thanh thường được thu ở môi trường đời thường, tiếng nói bình thường của một người cũng không được to, trừ khi nói to có chủ định, do đó nhiễu của môi trường (tần số thấp) nhiều khi có cường độ lớn bằng một phần đáng kể (nghe có thể dễ dàng nhận ra) của tiếng nói khi thu âm, bước đầu tiên của quá trình trích chọn đặc trưng

2 Nội dung tham khảo từ cuốn *Speech and Language Processing, 2007, chapter 9.* Tác giả Daniel Jurafsky & Jame H.Martin.

MFCC sẽ xử lý vấn đề này, bằng việc thực hiện tăng cường độ của những tần số cao lên nhằm làm tăng năng lượng ở vùng có tần số cao – vùng tần số của tiếng nói, một cách dễ hiểu là làm tiếng nói lớn hơn lên để ảnh hưởng của các âm thanh môi trường và nhiễu trở thành không đáng kể. Việc tăng cường độ của vùng tần số cao lên đồng thời làm cho thông tin rõ ràng hơn đối với mẫu tiếng nói. Hình sau mô tả trước và sau quá trình Pre-emphasis của một đoạn tín hiệu âm thanh:



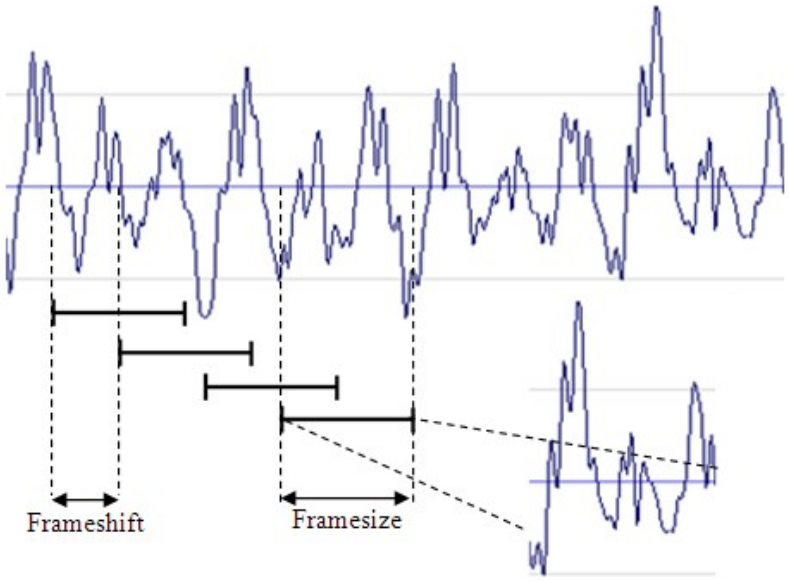

*Hình 3 (nguồn [2]): Một đoạn tần âm thanh trước và sau Pre-Emphasis*

### 2.2.2. Windowing

Trong hệ thống nhận dạng tiếng nói được trình bày ở khóa luận này, với mục đích nâng cao độ chính xác của việc nhận dạng tiếng, thay vì nhận dạng từng từ riêng biệt, mỗi một từ trong đoạn hội thoại sẽ được phân tích thành các âm vị (subphone) *[7]*, và hệ thống sẽ nhận dạng từng âm vị. Âm vị ở đây là đơn vị phát âm của một từ, các âm vị cấu thành tiếng nói, trong tiếng Anh, nó là đơn vị cấu thành phiên âm của từ (chẳng hạn *ONE*: *w-ah-n*, âm vị ở đây là *w*, *ah* và *n*), trong cách phát âm của tiếng Việt, cách viết của từ chính là hình thức văn bản của âm vị (chẳng hạn "*MOT*" = "*m*-oo-*t*", âm vị là *m, oo* và *t*).

Vì lý do đó, các đặc trưng cần phải được trích chọn trên từng âm vị, thay vì cả từ hay cả đoạn tiếng nói dài. Windowing là việc cắt đoạn tín hiệu âm thanh đầu vào ra thành các mẩu tín hiệu có thời lượng nhỏ, gọi là các frame. Mỗi frame này sau đó sẽ được nhận dạng nó thuộc âm vị nào. Nói cách khác, một frame sẽ là một tập gồm một số mẫu của tín hiệu ban đầu ta đã đề cập ở phần 2.1.

Một lý do khác cho thấy sự cần thiết của việc windowing là vì tín hiệu âm thanh thay đổi rất nhanh, do đó các thuộc tính như biên độ, chu kỳ sẽ không ổn định. Khi tín hiệu âm thanh được cắt ra thành những đoạn nhỏ thì ở mỗi đoạn, có thể coi tín hiệu đó là ổn định, các đặc trưng của tín hiệu là không đổi theo thời gian. Hình vẽ sau mô tả quá trình *Windowing:*




*Hình 4 (nguồn [2]): Minh họa quá trình Windowing*

Để thực hiện việc này, chúng ta sử dụng một cửa sổ (window) chạy dọc tín hiện âm thanh và cắt ra các đoạn tín hiệu nằm trong cửa sổ đó. Một cửa sổ được định nghĩa bằng các thông số:

- Frame size: độ rộng của cửa sổ, cũng là độ lớn của frame tín hiệu sẽ được cắt ra.
- Frame shift: bước nhảy của cửa sổ, là độ dài đoạn mà cửa sổ sẽ trượt để cắt ra frame tiếp theo.

Mỗi frame sau đó sẽ được nhân với một hệ số, giá trị của hệ số này tùy thuộc vào từng loại cửa sổ.

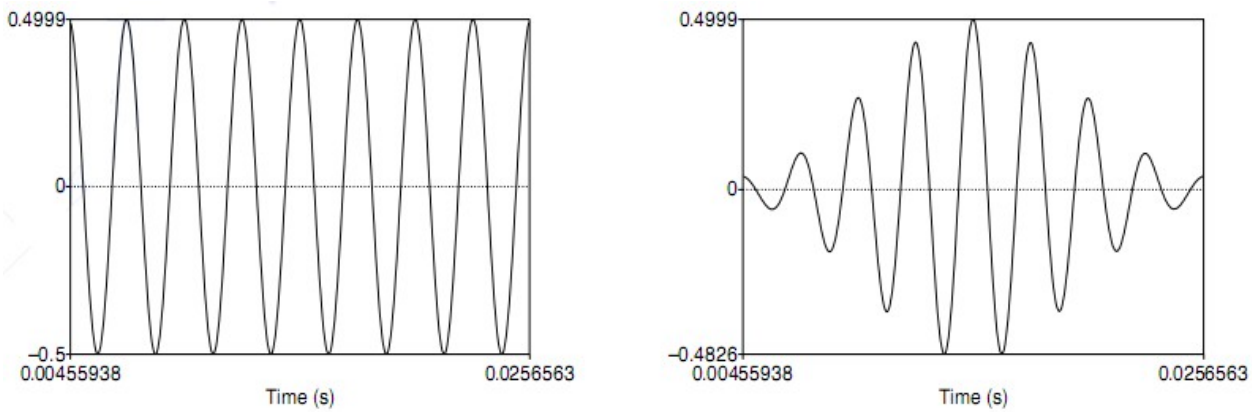
Trong đó $x[n]$ là giá trị của mẫu thứ n, $y[n]$ là giá trị của mẫu thứ n sau khi nhân với hệ số, $w[n]$ là hệ số cho mẫu thứ n trong frame đó.

Loại cửa sổ đơn giản nhất là cửa sổ **Rectangular**, giá trị của các hệ số $w[n]$ được cho bởi công thức sau:

Nói cách khác, của sổ Rectangular với bước nhảy là frame shift, ta lấy frame size giá trị liên tiếp của tín hiệu làm một frame.

Một loại cửa sổ khác thông dụng hơn trong trích chọn đặc trưng MFCC là cửa sổ **Hamming**. Trong loại cửa sổ này, giá trị của tín hiệu sẽ giảm dần về 0 khi tiến dần ra hai biên của frame. Nói cách khác, nếu sử dụng cửa sổ Hamming để lấy ra các frame, năng lượng của mỗi frame sẽ tập trung ở giữa frame, một ưu điểm nữa là các giá trị biên của cửa sổ Hamming tiến dần về 0 sẽ làm bước biến đổi Fourier ngay sau trở nên dễ dàng hơn (với cửa sổ Rectangular các giá trị giữ nguyên so với mẫu tiếng nói, bên ngoài cửa sổ nhận giá trị 0, các giá trị sẽ bị tăng đột ngột ở hai biên). Hệ thống nhận dạng trong khóa luận này trình bày sẽ sử dụng cửa sổ Hamming. Biểu thức hệ số của cửa sổ này là:

So sánh hai loại cửa sổ Rectangular và Hamming



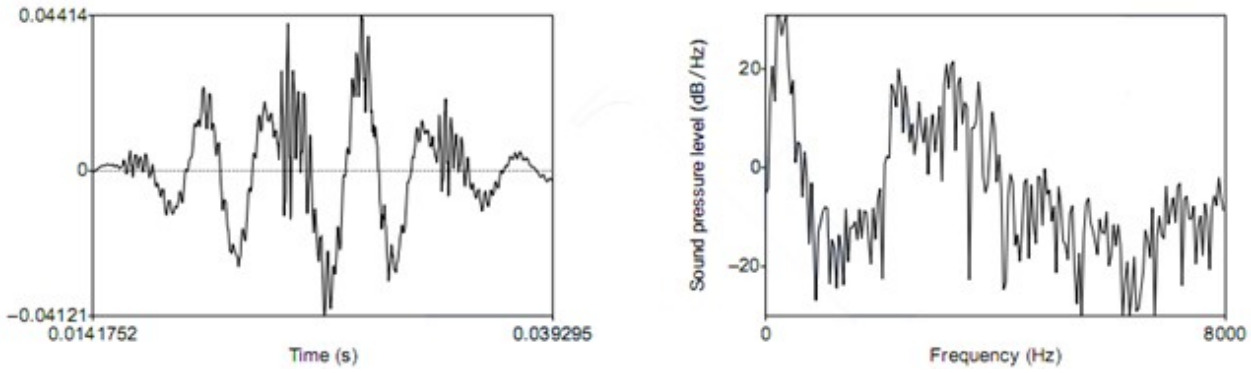

*Hình 5 (nguồn [2]): So sánh Rectangular (trái) và Hamming window (phải)*

### 2.2.3. DFT (Discrete fourier transform)

Bước biến đổi tiếp theo là thực hiện biến đổi Fourier rời rạc đối với từng mẫu tín hiệu đã được cắt ra. Qua phép biến đổi này, tín hiệu sẽ được đưa về không gian tần số.

Công thức của biến đổi Fourier:

Trong đó x[n] là giá trị của mẫu thứ n trong frame, X[k] là một số phức biểu diễn cường độ và pha của một thành phần tần số trong tín hiệu gốc, N là số mẫu trong một frame. Thông thường người ta sử dụng biến đổi FFT (Fast fourier transform) thay vì DFT. Biến đổi FFT nhanh hơn nhiều so với biến đổi DFT, tuy nhiên thuật toán này đòi hỏi giá trị N phải là một lũy thừa của 2. Hình sau mô tả trước và sau khi biến đổi DFT của một cửa sổ:



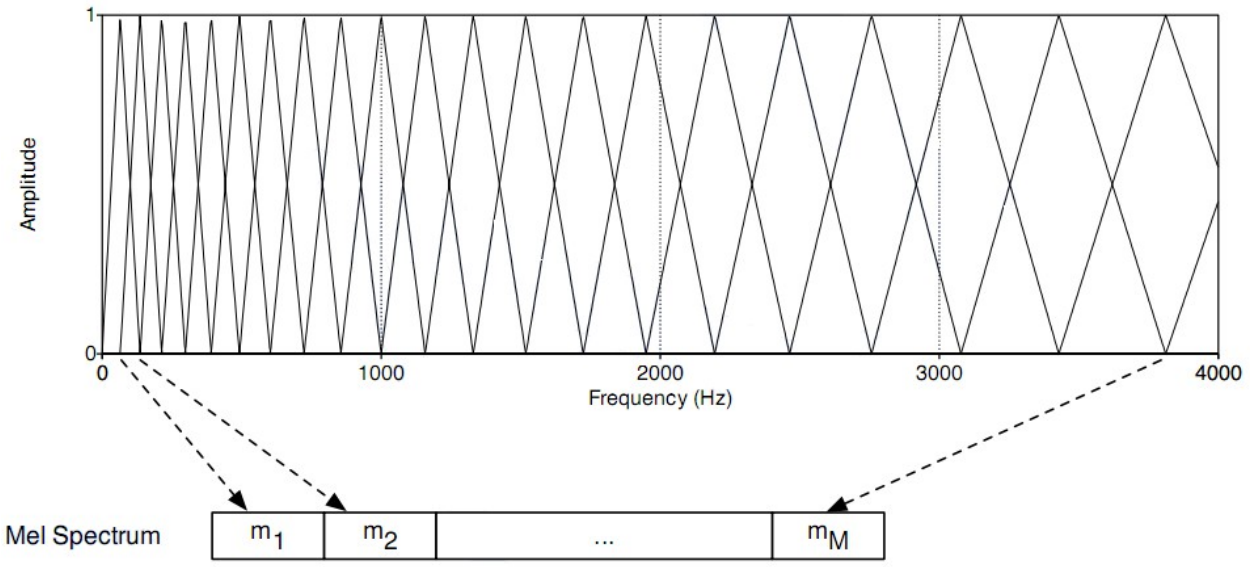

*Hình 6 (nguồn [2]): Biến đổi DFT cho một Hamming window*

### 2.2.4. Mel filter-bank and log

Kết quả của quá trình biến đổi Fourier thể hiện năng lượng của tín hiệu ở những dải tần số khác nhau. Tuy nhiên, tai của người lại không có sự nhạy cảm như nhau đối với mọi dải tần số. Do đó việc mô hình hóa tính chất này của tai người trong quá trình trích chọn đặc trưng làm tăng khả năng nhận dạng của hệ thống. Trong mô hình trích chọn đặc trưng MFCC, tần số sẽ được chuyển sang thang đo tần số mel theo công thức:

Trong đó $f$ là tần số ở thang đo thường, $f_{\text{mel}}$ là tần số ở thang đo mel. Người ta sử dụng các băng lọc để tính các hệ số mel. Sử dụng bao nhiêu băng lọc thì sẽ cho ra bấy nhiêu hệ số mel, và các hệ số mel này sẽ là đầu vào cho quá trình tiếp theo của trích chọn đặc trưng MFCC.

Hình vẽ sau biểu diễn mô hình các băng lọc trong thang đo tần số bình thường và thang đo mel:




*Hình 7 (nguồn [2]): Mô hình các băng lọc trong thang đo tần số bình thường và thang đo mel*

Cuối cùng của giai đoạn này, ta lấy logarit cơ số tự nhiên của phổ tính theo thang đo Mel, thao tác này có 2 nguyên nhân, một là do tai người nhạy cảm với âm thanh cường độ thấp hơn, hai là làm các giá trị đặc trưng nhỏ đi, tiện cho việc tính toán.

### 2.2.5. Discrete consinse transform

Bước tiếp theo của việc trích chọn đặc trưng MFCC là biến đổi fourier ngược với đầu vào là các hệ số phổ mel của bước trước, đầu ra sẽ là các hệ số cepstrum (MFCC – Mel Frequency Cepstrum Coefficients).

Sau khi thực hiện biến đổi Fourier thì dãy tín hiệu theo thời gian đã được chuyển thành phổ tần số, và việc áp dụng các băng lọc tần số mel giúp cô đọng phổ tần số về một số hệ số nhất định (bằng với số băng lọc). Các hệ số này thể hiện các đặc trưng của nguồn âm thanh như tần số cơ bản, xung âm thanh… Tuy nhiên, các đặc trưng này không quan trọng đối với việc phân biệt các âm khác nhau. Thay vào đó, các đặc trưng về bộ máy phát âm (khoang miệng, khoang mũi, thanh quản, hầu) rất cần thiết cho việc nhận dạng các âm. Việc thực hiện biến đổi fourier ngược sẽ giúp tách biệt các đặc trưng về nguồn

âm và bộ máy phát âm từ các hệ số (các đặc trưng về bộ máy phát âm là các hệ số đầu tiên).

### 2.2.6. Feature extraction

Từ các hệ số mel thu được từ quá trình trước, thông thường chúng ta chỉ lấy ra 12 hệ số đầu tiên để chọn làm đặc trưng. 12 hệ số này chỉ đặc trưng cho các bộ phận của bộ máy phát âm. Như vậy chúng ta đã có 12 đặc trưng đầu tiên.

Đặc trưng thứ 13 là năng lượng của âm. Năng lượng của mỗi khung tín hiệu được tính ngay từ sau bước windowing:

Với 13 đặc trưng đó, chúng ta thêm vào 13 đặc trưng delta thể hiện tốc độ thay đổi của của âm giữa các khung tín hiệu, được tính bằng công thức:

trong đó d(t) là đặc trưng delta của khung t, c(t+1) và c(t-1) là các đặc trưng phổ của khung ngay sau và trước khung t; và 13 đặc trưng double delta thể hiện gia tốc thay đổi của âm giữa các khung tín hiệu. Công thức tính các đặc trưng double delta giống với công thức tính các đặc trưng delta, khi coi c(t) là giá trị của các đặc trưng delta.

### 2.2.7. Tổng kết

Trích chọn đặc trưng MFCC sẽ thu được các đặc trưng sau đây:

- 12 giá trị đặc trưng phổ Mel được biến đổi Fourier ngược
- 12 giá trị delta phổ
- 12 giá trị double delta phổ
- 1 giá trị mức năng lượng
- 1 giá trị delta mức năng lượng
- 1 giá trị double delta mức năng lượng

Tổng cộng: 39 đặc trưng cho mỗi frame tiếng nói.

# *Chương 3.* ĐẶC TRƯNG VỀ THANH ĐIỆU CỦA TIẾNG VIỆT

Ở chương trước ta đã trình bày về việc chọn đặc trưng cho tiếng nói tiếng Việt thông qua mô phỏng bộ máy phát âm, đây cũng là đặc trưng chung cho các ngôn ngữ khác. Tuy nhiên, tiếng nói tiếng Việt còn có những đặc điểm riêng, đặc thù và độc đáo, việc tìm hiểu những đặc trưng này và đưa chúng vào phục vụ nhận dạng sẽ làm tăng độ chính xác toàn cục của hệ thống nhận dạng tiếng nói.

## 3.1. **Khái niệm ngôn điệu, ngữ điệu và thanh điệu** [3]

Nói một cách nôm na, trong ngôn ngữ nói, "ngôn điệu" là cái mang lại âm sắc cho tiếng nói, âm sắc là biểu hiện tự nhiên của giọng nói, mang ý nghĩa nhấn mạnh hoặc thể hiện sắc thái tình cảm, lời nói không có ngôn điệu giống như lời nói của robot, không giống tiếng nói tự nhiên. Các nhà ngôn ngữ học cho rằng bản chất ngôn điệu là sự phủ lên âm tiết các yếu tố trọng âm, thanh điệu, ngữ điệu và trường độ. Vai trò của ngôn điệu rất quan trọng trong tổng hợp tiếng nói, nếu không xử lý được vấn đề ngôn điệu thì không thể tổng hợp được tiếng nói tự nhiên của con người được. Đặc trưng quan trọng nhất của ngôn điệu là độ cao, độ dài, độ to, tương ứng là các đại lượng tần số cơ bản F0, thời gian của âm tiết, âm vị D, và cường độ I.

Ngôn điệu của lời nói liên kết chặt chẽ với khái niệm "ngữ điệu". Có thể nói ngữ điệu là sự nâng cao hạ thấp của lời nói trong câu, khi xét là một âm tiết (trong tiếng Việt gọi là một tiếng) ngữ điệu lúc này trở thành thanh điệu của âm tiết đó. Đặc trưng chính cho tính chất này là tần số cơ bản của giọng nói: F0. Việc lấy các giá trị F0 theo thời gian tạo thành đường nét F0. Trong lời nói liên tục, đường nét F0 cho mỗi thanh điệu có các đặc trưng khác nhau, tín hiệu thô ban đầu là dạng thô của đường nét F0, ở chương này, ta đi nghiên cứu cách làm mịn đường nét F0 cho mỗi âm tiết riêng biệt, theo đúng giới hạn ban đầu của bài toán.

## 3.2. **Tìm đường nét F0 và nghiên cứu đặc điểm của từng thanh điệu trong tiếng Việt**

Trong tiếng Việt, có 6 thanh điệu được sử dụng: thanh **ngang**, **huyền**, **sắc**, **hỏi**, **nặng** và **ngã**. Trong văn học xưa từng xuất hiện luật bằng trắc: thanh bằng chỉ âm tiết có đường nét có chiều hướng đi ngang hoặc đi xuống (là thanh ngang, huyền) thanh trắc chỉ âm tiết có đường nét đi lên (thanh sắc, nặng, ngã), tuy nhiên phân loại như vậy là chưa chặt chẽ và đầy đủ. Sau đây, ta sẽ đưa ra một cách làm mịn đường nét F0 thể hiện thanh điệu tiếng nói và nghiên cứu đặc điểm của từng thanh điệu.

---

3 Nội dung tham khảo trong tài liệu: *"Mô hình Fujisaki và áp dụng trong phân tích thanh điệu tiếng Việt"* của *Bạch Hưng Nguyên, Nguyễn Tiến Dũng.*

### 3.2.1. Tính đường nét thanh điệu [4]

#### 3.2.1.1. *Hàm biên độ trung bình (AMDF – Average Magnitude Difference Fucntion)*

Hàm hiệu biên độ trung bình của một tín hiệu là hiệu biên độ của chính nó rời đi p mẫu, được tính bởi công thức:

$$d(p) = |$$

Ở đây $x(n)$ là giá trị biên độ thứ $n$ của tín hiệu, $N$ là số giá trị biên độ (thường là số giá trị được lấy ra trong 1 khoảng thời gian cố định, với tần số lấy mẫu là $Fs$)

Nếu $x(n)$ là tín hiệu tuần hoàn với chu kì $T$ thì khi $p$ tiến dần tới giá trị $T$, hàm $d(p)$ sẽ đạt giá trị nhỏ nhất.

Do tín hiệu là rời rạc nên sẽ tồn tại giá trị nguyên $p_0$ sao cho $d(p_0)$ là nhỏ nhất, khi đó giá trị $f_0 = Fs/p_0$ được coi là tần số cơ bản của đoạn tín hiệu đó, nói cách khác nó đặc trưng cho thanh điệu của đoạn tín hiệu đó, $f_0$ là một giá trị trong đường nét $F0$ – đặc trưng cho thanh điệu của toàn bộ tín hiệu giọng nói ban đầu.

Giọng nói của người bình thường có tần số cơ bản là khoảng 90Hz với giọng nam và 200Hz với giọng nữ, ta sẽ lấy $p_0$ sẽ nằm trong khoảng rộng hơn từ $F_s/250$ đến $F_s/80$.

Cứ mỗi đoạn tín hiệu kéo dài từ *10-25ms* ta lại lấy một giá trị $f_0$ như vậy, tập $f_0$ theo thời gian thu được chính là đường nét $F0$.

#### 3.2.1.2. *Thực hiện tìm đường nét F0*

- Cắt xén tín hiệu làm nổi rõ chu kì cơ bản

$$y[n] =$$

Trong đó $C$ được chọn vào khoảng 1/3 giá trị biên độ cực đại trên toàn tín hiệu

- Tính hàm biên độ trung bình: tín hiệu sau khi được cắt xén được đưa vào hàm lấy biên độ trung bình như trong mục 3.2.1.1 với $N$ là độ dài của một khung (gồm các giá trị được lấy trong 1 khoảng thời gian nhất định, ở đây lấy số giá trị trong 1 frame (khoảng 10-25ms) như ở mục I đã trình bày).

4 Nội dung tham khảo trong tài liệu *"Nhận dạng tiếng Việt dùng mạng Neuron kết hợp trích đặc trưng dùng LPC và AMDF", 2005*, tác giả *Hoàng Đình Chiến.*

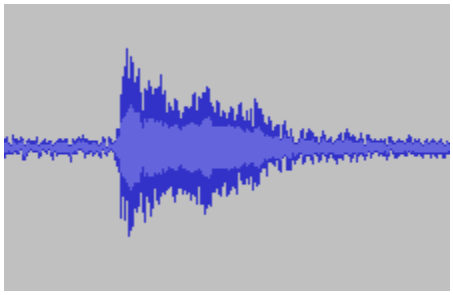
- Làm mịn: với các $d(p_0) > 0.7*d_{max}(p)$ ta coi đó là khung vô thanh, tính giá trị đặc trưng $f_0 = 0$. Sau khi được tập $\{ f_0 \}$ tiếp tục làm mịn đường nét F0 bằng cách: nếu các khung vô thanh ở đầu hoặc cuối âm tiết thì sẽ được thay thế bởi giá trị $f_0$ kế cận, nếu khung vô thanh ở giữa âm tiết thì thay bằng trung bình của 2 giá trị $f_0$ ngay cạnh. Cuối cùng làm trơn đường nét F0 bằng bộ lọc với đáp ứng xung $h$ = [0.1, 0.2, 0.4, 0.2, 0.1]
- Lấy đặc trưng: Tùy vào nhu cầu sử dụng bao nhiêu đặc trưng mà lấy các giá trị từ đường nét *F0*, có thể lấy các giá trị trên đường nét, hoặc có thể biến đổi rời rạc đường nét về một số giá trị đặc trưng nhất định.

### 3.2.2. Đặc điểm của từng thanh điệu dựa vào đường nét [5]

#### 3.2.2.1. *Thanh ngang*

Đường nét của thanh ngang thường có xu hướng giảm nhẹ, điều này dễ hiểu bởi khi phát âm, mức năng lượng gần như không đổi và giảm dần về cuối âm tiết, thanh ngang dễ bị nhầm lẫn với thanh huyền vì đường nét của chúng tương tự nhau (xu hướng không đổi hoặc giảm nhẹ)

Hình mô tả đường nét thô của thanh ngang.

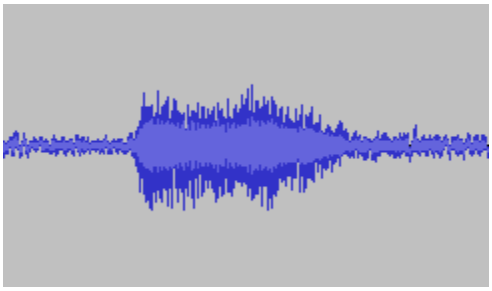


*Hình 8: Đường nét thô của thanh ngang, âm vị "a"*

#### 3.2.2.2. *Thanh huyền*

5 Nội dung tham khảo từ *"Mô hình Fujisaki và áp dụng trong phân tích thanh điệu tiếng Việt"*, tác giả Bạch Hưng Nguyên, Nguyễn Tiến Dũng

Đường nét thanh huyền khi phát âm chuẩn có xu hướng không tăng, không giảm, gần giống với thanh ngang, điều này ta vừa đề cập tới, nó gây khó khăn trong việc phân biệt riêng hai thanh điệu này. Hình sau là phổ biên độ thô của thanh huyền:

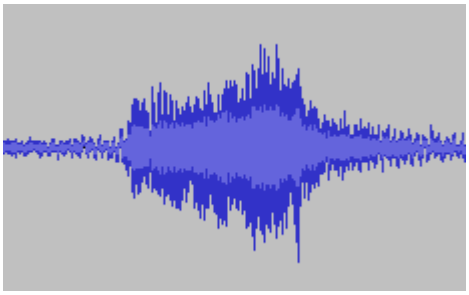


*Hình 9: Đường nét thô của thanh huyền, âm vị "à"*

3.2.2.3. ***Thanh sắc***

Thanh sắc có đường nét đi lên, khá giống với thanh ngã và thanh nặng, thanh sắc có âm vực bắt đầu cao hơn 2 thanh còn lại, có báo cáo thí nghiệm kết luận rằng: cho đường nét của thanh sắc và thanh ngã giống hệt nhau, khi tổng hợp lại người nghe vẫn phân biệt được 2 thanh này. Tuy nhiên, thanh ngã và thanh nặng cũng còn những đặc điểm quan trọng khác để phân biệt với các thanh còn lại.

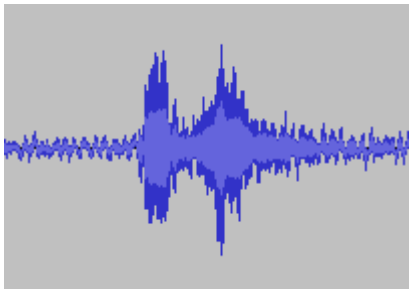
Quan sát đường nét (ở dạng phổ) thô của thanh sắc:



*Hình 10: Đường nét thô của thanh sắc, âm vị "á"*

### 3.2.2.4. *Thanh ngã*

Đường nét thanh ngã bị gãy ở giữa, không chỉ gãy ở F0 mà thanh ngã còn bị gãy ở phổ, đó chính là khác biệt lớn nhất giữa thanh ngã với các thanh còn lại. Hình sau mô tả điều này

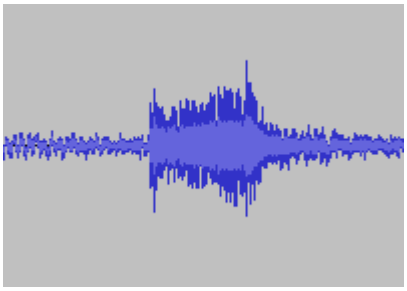


*Hình 11: Đường nét thô của thanh ngã, âm vị "ã"*

### 3.2.2.5. *Thanh nặng*

Thanh nặng có đặc trưng bị gẫy, đứt và đi xuống đột ngột ở cuối âm, thanh nặng cũng gặp khó khăn khi phân biệt với thanh sắc, nếu cho thanh nặng đường nét F0 của thanh sắc thì người nghe vẫn phân biệt được đó là thanh nặng, có điều phần cuối âm tiết cảm giác bị nhấn lên, nếu âm tiết được phát âm rõ ràng, chuẩn để chủ động hạ giọng cuối âm tiết có thanh nặng, khả năng phân biệt 2 thanh này sẽ cao hơn.

Sau đây là hình mô tả đường nét thô dạng phổ của thanh nặng:

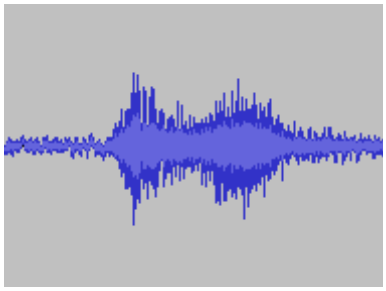


*Hình 12: Đường nét thô của thanh nặng, âm vị "ạ"*

### 3.2.2.6. *Thanh hỏi*

Đường nét của thanh hỏi có đặc trưng là được nâng cao ở hai đầu và cao độ thấp ở giữa âm tiết, tuy nhiên trong tiếng nói tự nhiên, đặc trưng này không được thể hiện rõ ràng do những yếu tố như tốc độ nói, kiểu nói của mỗi người và tùy ngữ cảnh thanh điệu này được nhấn như thế nào, thanh hỏi trong giọng nói tự nhiên, không ngữ cảnh hay bị nhầm lẫn với thanh huyền và thanh ngang.

Trường hợp phát âm lý tưởng cho âm tiết có thanh hỏi:



*Hình 13: Đường nét thô của thanh hỏi, âm vị "ả"*

## *Chương 4.*

# SỬ DỤNG ĐẶC TRƯNG TIẾNG NÓI NÓI CHUNG VÀ TIẾNG VIỆT NÓI RIÊNG CHO MÔ HÌNH NHẬN DẠNG TIẾNG NÓI TIẾNG VIỆT

Như ta đã trình bày ở chương 1, trích chọn đặc trưng MFCC mô phỏng quá trình phát ra tiếng nói của bộ máy phát âm thông qua 39 đặc trưng cho mỗi frame tín hiệu, như vậy mỗi frame sẽ được coi như 1 vector 39 chiều giá trị thực và một tín hiệu tiếng nói là một tập các frame. Mục tiêu bài toán trở thành: với tiếng nói đầu vào bất kì, ta gán nhãn cho các frame (sau khi trích chọn đặc trưng) sao cho phù hợp nhất với "mô hình âm học"

của hệ thống ta xây dựng (khái niệm mô hình âm học sẽ được nhắc lại trong chính chương này) . Bằng việc áp dụng "mô hình Markov ẩn - HMM" để gán nhãn frame, tiếng nói sẽ được nhận dạng về hình thức văn bản (text). Bên cạnh đó, ta cũng thực hiện phân đường nét F0 (đường đặc trưng cho thanh điệu) để minh họa việc phân biệt thanh điệu cho các âm tiết phát âm giống nhau.

## 4.1. **Mô hình Markov ẩn (Hidden Markov Model)**

Ở phần này ta sẽ giới thiệu mô hình thống kê HMM để áp dụng mô hình này vào bài toán nhận dạng tiếng nói.

### 4.1.1. Xích Markov, quá trình Markov

Xích Markov (đặt theo tên nhà toán học người Nga Andrei Andreyevich Markov) là một dãy $X_1$, $X_2$, $X_3$, ... gồm các biến ngẫu nhiên. Tập tất cả các giá trị có thể có của các biến này được gọi là *không gian trạng thái S*, giá trị của $X_n$ là trạng thái của quá trình (hệ) tại thời điểm *n*.

Nếu việc xác định (dự đoán) phân bố xác suất có điều kiện của $X_{n+1}$ khi cho biết các trạng thái quá khứ là một hàm chỉ phụ thuộc $X_n$ thì:

$$P(X_{n+1} = x \mid X_0, X_1, \ldots, X_n) = P(X_{n+1} = x \mid X_n)$$

trong đó *x* là một trạng thái nào đó của quá trình (*x* thuộc *không gian trạng thái S*) . Đó là thuộc tính Markov.

Một cách đơn giản để hình dung một kiểu chuỗi Markov cụ thể là qua một ôtômat hữu hạn (*finite state machine*). Nếu hệ ở trạng thái *y* tại thời điểm *n* thì xác suất mà hệ sẽ chuyển tới trạng thái *x* tại thời điểm *n+1* không phụ thuộc vào giá trị của thời điểm *n* mà chỉ phụ thuộc vào trạng thái hiện tại *y*. Do đó, tại thời điểm *n* bất kỳ, một xích Markov hữu hạn có thể được biểu diễn bằng một ma trận xác suất, trong đó phần tử *x*, *y* có giá trị bằng $P(X_{n+1} = x \mid X_n = y)$ và độc lập với chỉ số thời gian *n* (nghĩa là để xác định trạng thái kế tiếp, ta không cần biết đang ở thời điểm nào mà chỉ cần biết trạng thái ở thời điểm đó là gì).

Một quá trình mang tính ngẫu nhiên có đặc tính giống như xích Markov ta gọi là quá trình Markov bậc 1. Quá trình Markov bậc n là dãy biến ngẫu nhiên mà dự đoán phân bố xác suất có điều kiện $X_{n+1}$ là một hàm phụ thuộc $X_1$, $X_2$, …, $X_n$. Tuy nhiên ở đây, áp dụng cho bài toán nhận dạng giọng nói, ta chỉ xét tới quá trình Markov bậc 1 (hay xích Markov). Để tiện cho việc trình

bày, trong khóa luận này, nếu không có ghi chú thêm, ta hiểu quá trình Markov chính quá trình Markov bậc 1.

### 4.1.2. Mô hình Markov ẩn (Hidden Markov Model - HMM)

**Mô hình Markov ẩn** là mô hình thống kê trong đó hệ thống được mô hình hóa được cho là một quá trình Markov với các tham số không biết trước và nhiệm vụ là xác định các tham số ẩn từ các tham số quan sát được, dựa trên sự thừa nhận này. Thông thường, các tham số biết trước là xác suất chuyển trạng thái trong xích Markov bằng việc quan sát các chuỗi đã biết, các tham số chưa biết có thể là phân phối xác suất của trạng thái đối với quan sát… Chính vì vậy có 3 vấn đề mà HMM cần giải quyết:

- Cung cấp cho mô hình các tham số, tính xác suất của dãy đầu ra cụ thể. Giải bằng thuật toán tiến trước.
- Cung cấp cho mô hình các tham số, tìm dãy các trạng thái (ẩn) có khả năng lớn nhất mà có thể sinh ra dãy đầu ra đã cung cấp. Giải bằng thuật toán *Viterbi*.
- Cung cấp dãy đầu ra, tìm tập hợp có khả năng nhất của chuyển tiếp trạng thái và các xác suất đầu ra. Giải bằng thuật toán *Baum-Welch*.

Trong bài toán nhận dạng tiếng nói, giải pháp cho vấn đề thứ ba sẽ áp dụng cho phần training và giải pháp cho vấn đề hai là phần decode để gán nhãn thích hợp nhất cho tín hiệu, hai thuật toán Viterbi và Baum-Welch sẽ được trình bày vào các phần ngay sau đây.

## 4.2. Áp dụng mô hình HMM cho bài toán nhận dạng tiếng nói, sử dụng trích chọn đặc trưng MFCC [6]

### 4.2.1. Mô hình hóa nhận dạng tiếng nói bằng HMM

6 Nội dung tham khảo trong cuốn "*Speech and Language Processing", 2007, chapter 9,* tác giả *Daniel Jurafsky & Jame H.Martin.*

#### 4.2.1.1. *Mô hình hóa*

Bây giờ, ta sẽ mô hình hóa bài toán nhận dạng tiếng nói một cách khái quát bằng HMM (các công thức và kí hiệu từ mục này sẽ sử dụng đồng nhất):

Tập trạng thái $Q$: là trạng thái các thành phần trong quá trình Markov có thể có:

$$Q = q_1 q_2 \ldots q_n$$

Ở đây, trạng thái chính là các âm vị khác nhau của toàn bộ từ vựng, bởi ta cắt tín hiệu tiếng nói thành các frame, các frame này mang các đặc trưng phát âm và cảm nhận âm của một âm vị (khái niệm âm vị đã được nhắc tới từ Chương 2).

Ta coi mỗi một từ khi nói là một chuỗi trạng thái theo thời gian:

$$O = o_1 o_2 \ldots o_t$$

Trong đó quan sát thứ i: $\boldsymbol{o_i}$ chính là 1 frame tiếng nói sau khi áp dụng kĩ thuật trích chọn MFCC, đặc trưng bởi 1 vector 39 chiều số thực. Tiếng nói là tập hợp các frame tiếng nói liên tiếp như vậy.

Tập từ vựng: $V = v_1 v_2 \ldots v_v$

Chẳng hạn trong bài toán nhận dạng chữ số, tập từ vựng sẽ là “khong”, “mot”, “hai”, “ba”, …, “chin” đây cũng chính là tập từ vựng của hệ thống ta đang xây dựng.

Ma trận chuyển trạng thái

$$A = a_{01} a_{02} \ldots a_{n1} \ldots a_{nn}$$

Trong đó $a_{ij}$ là xác suất chuyển từ trạng thái $i$ sang trạng thái $j$ đối với một quan sát bất kì.

Ma trận likelihood cho dãy quan sát $O$, ở đây $b_j(o_t)$ là xác suất trạng thái $j$ nhìn quan sát thứ $t$ (thời điểm $t$), cũng có thể hiểu đây là xác suất để quan sát $o_t$ nhận trạng thái $j$.

$$B = b_j(o_t)$$

Ngoài ra, ta bổ sung 2 trạng thái $q_0$, $q_{end}$ độc lập với các trạng thái $q_1 \ldots q_n$ với ý nghĩa là trạng thái bắt đầu và kết thúc cho mọi dãy quan sát.

Với mỗi tín hiệu được đặc trưng bởi chuỗi quan sát $O$, ta cần tính được các xác suất $A$ và $B$ để phục vụ cho quá trình nhận dạng, trong một số bài toán ứng dụng HMM bình thường, các dãy quan sát trong tập training ta thường biết trước được trạng thái của mỗi quan sát nhờ việc chủ động đưa vào các dãy quan sát theo ý muốn.

Tuy nhiên trong bài toán nhận dạng tiếng nói, $O$ là tín hiệu liên tục, được rời rạc hóa và trích chọn thành các frame quan sát $o_i$, với một tín hiệu tiếng nói ta biết trước đó là từ gì, từ đó được cấu thành bởi bao nhiêu subphone, nhưng không hề biết bao frame nào thuộc subphone nào.

Chẳng hạn từ *"MOT"* được trích chọn và rời rạc bởi 30 frames quan sát. Các âm vị cấu thành từ "*mot*" là "*m – oo – t*" nhưng ta không hề biết là bao nhiêu frame đầu tiên có trạng thái "*m*", bao nhiêu frame tiếp theo có trạng thái "*oo*" và bao nhiêu frame cuối có trạng thái "*t*".

Việc gán nhãn thủ công - "handed labeling" cho tập training là thực hiện được nhưng rất tốn công sức (để gán nhãn cho 1 giờ tiếng nói ta sẽ phải tốn 400 giờ gán nhãn bằng tay), bù lại việc gán nhãn thủ công sẽ làm việc tính các xác suất $A$, $B$ trở nên đơn giản và chính xác hơn, tuy nhiên, trong bài toán mở là tập từ vựng tăng lên rất lớn, tập training còn lớn hơn nhiều, chính vì vậy một phương pháp gán nhãn tự động mang tên "Embeded training" được đưa ra nhằm tăng tốc quá trình gán nhãn cho quá trình training. Phương pháp này sẽ được giới thiệu ở phần sau.

#### 4.2.1.2. ***Huấn luyện (Training)*** [7]

Mục này sẽ đề cập đến vấn đề huấn luyện HMM, được coi là phần khó nhất trong ba vấn đề được nêu ra ở phần này. Nhiệm vụ của việc huấn luyện HMM là điều chỉnh các tham số mô hình ($A$, $B$, $\pi$) để đạt được một mô hình tối ưu nhất cho các mẫu huấn luyện.

Mục tiêu cuối cùng là đưa ra được các tham số $A$, $B$ (với ý nghĩa nêu ở trên) cho các mẫu huận luyện. Có khá nhiều kỹ thuật đã được đưa ra cho vấn đề này, tuy nhiên trong mục này tôi sẽ chỉ giới thiệu một kỹ thuật huấn luyện khá thông dụng, đó là kỹ thuật huấn luyện sử dụng thuật toán Baum-Welch, hay còn gọi là thuật toán Forward-Backward, một

7 Nội dung nghiên cứu chung với khóa luận *"Các kĩ thuật nhận dạng tiếng nói", 2010,* sinh viên Đàm Tiến Dũng, Đại Học Công Nghệ.

trường hợp riêng của thuật toán tối ưu hóa kỳ vọng (Expectation Maximization Algorithm).

Thuật toán này dựa trên phương pháp lặp để đạt được cực đạt địa phương của hàm xác suất *P(O|λ)* - xác suất của quan sát *O* với mô hình λ hiện tại. Trong mỗi vòng lặp, các tham số của mô hình sẽ được điều chỉnh lại, và mô hình mới sẽ tốt hơn mô hình cũ, như đã được chứng minh bởi Baum và nhiều người khác. Thuật toán sẽ dừng lại khi gặp điều kiện hội tụ, tức là khi xác suất mô hình *P(O|λ)* không tăng nữa hoặc tăng rất ít, hoặc khi gặp phải điều kiện tới hạn của tính toán. Mô hình luôn luôn hội tụ, tuy nhiên chỉ có thể đảm bảo giá trị đạt được của *P(O|λ)* là một cực đại địa phương.

Trước khi đi vào thuật toán cụ thể, cần định nghĩa hai xác suất: $\gamma_t(i)$ và $\xi_t(i,j)$. Đầu tiên là xác suất $\gamma_t(i)$, có ý nghĩa là xác suất ở trạng thái *i* tại quan sát thứ *t*, với một dãy quan sát và mô hình cho trước:

Vì và nên ta có:

Xác suất thứ hai là $\xi_t(i,j)$, có ý nghĩa là xác suất ở trạng thái i tại quan sát thứ t và trạng thái j tại quan sát thứ t+1, với mô hình λ và dãy quan sát *O* cho trước:

Theo công thức Bayes ta có:

Sử dụng các thuật toán Forward và Backward, ta tính được:

Do vậy ta có:

Nếu ta cộng $\gamma_t(i)$ đối với tất cả các giá trị của *t* (ngoại trừ *t=T*), ta sẽ thu được một kết quả là giá trị kỳ vọng về số lần mà trạng thái *i* đoán nhận một quan sát trên tất cả các

quan sát. Mặt khác, nếu ta cộng $\xi_t(i,j)$ đối với tất cả các giá trị của $t$ (ngoại trừ $t=T$), ta sẽ thu được một giá trị kỳ vọng về số lần trạng thái $i$ chuyển sang trạng thái $j$. Từ những đánh giá trên, các tham số của mô hình sẽ được tính toán lại như sau:

Ở đây, dấu ^ thể hiện các tham số mới sau khi điều chỉnh lại. Sau khi cập nhật lại các tham số như trên, chúng ta sẽ thu được một mô hình mới phù hợp hơn mô hình cũ $\lambda$, đối với dãy quan sát O:

Cụ thể thuật toán Baum-Welch được mô tả như sau:

1. Khởi tạo $A$, $B$ và $\pi$.
2. Lặp:

   Bước kỳ vọng: tính các giá trị $\gamma_t(i)$ và $\xi_t(i,j)$.

   Bước tối ưu hóa: tính lại các tham số $A$, $B$ và $\pi$.

#### 4.2.1.3. ***Huấn luyện nhúng (Embedded training)***

Cụ thể về thuật toán Baum-Welch trong bài toán nhận dạng tiếng nói sẽ được mô tả trong phần này.

Trong thực tế, để gán nhãn cho một tập dữ liệu dài một tiếng, có thể cần thời gian lên đến 400 tiếng. Do đó phương pháp huấn luyện hand-labeled word training là không khả thi trong các bài toán có bộ từ vựng lớn. Một kỹ thuật huấn luyện khác, không đòi hỏi dữ liệu gán nhãn sẵn, đã được xây dựng dựa trên thuật toán Baum-Welch (mục **4.2.1.2**), đó là phương pháp huấn luyện nhúng (Embedded training). Phương pháp này gồm hai bước như sau:

a. Xây dựng mô hình Markov ẩn $\lambda=(A,B,\mu)$ cho từ cần huấn luyện. Các tham số $A$, $B$ và $\mu$ được khởi tạo như sau:

   $\pi_i$ sẽ có giá trị bằng 1 nếu mô hình bắt đầu với trạng thái $i$, ngược lại $\pi_i = 0$.

   $a_{ij}$ sẽ có giá trị bằng 0.5 nếu $i = j$ hoặc bước chuyển từ $i$ sang $j$ là một bước chuyển tồn tại trong mô hình, ngược lại $a_{ij} = 0$. Riêng $a_{nn}$ sẽ bằng 1.

Khởi tạo các giá trị kỳ vọng  và phương sai , trong đó giá trị kỳ vọng và phương sai cho mỗi trạng thái sẽ là kỳ vọng và phương sai của tất cả các vector đầu vào. Sau đó tính $b_j(o_t)$ dựa vào  và .

$$b_j o_t = exp( - )$$

Ta sẽ giải thích kĩ hơn công thức này ở phần sau.

b. Chạy thuật toán Baum-Welch cho mô hình λ.

Bảng sau mô tả HMM cho từ *MOT* và các giá trị khởi tạo của A và $\pi$:

| | m | o | t |
|---|---|---|---|
| $\pi$ | 1 | 0 | 0 |

| | m | o | t |
|---|---|---|---|
| m | 0.5 | 0.5 | 0 |
| o | 0 | 0.5 | 0.5 |
| t | 0 | 0 | 1 |

. . . . . .

Như vậy, mỗi mẫu tiếng nói training sẽ có một ma trận chuyển trạng thái *A* (transition matrix) và một ma trận phân phối trạng thái *B* (observation likelihood) cho mỗi mẫu. Bây giờ, với một mẫu tiếng nói *O* nằm ngoài tập training cần nhận dạng, ta cần tính lại ma trận *A* và phân phối các trạng thái với mẫu này đối với tập training đã xây dựng. Cách tính 2 ma trận này như sau:

Gọi $A_k(i,j)$, $B_k(j, t)$, $S_k$ ($k = 1..M$ với $M$ là số mẫu training) lần lượt là ma trận chuyển trạng thái, phân phối các trạng thái và số lượng quan sát của mẫu thứ k trong tập training.

Ma trận chuyển trạng thái A của toàn tập training được tính theo công thức:

$$A_{ij} =$$

Tính ma trận phân phối các trạng thái cho dãy quan sát *O* khó khăn hơn một chút.

Ta đã có các ma trận $B_k(j, t)$ với ý nghĩa tại mẫu thứ *k*, xác suất để quan sát *t* có trạng thái *j* là $B_k(j, t)$. Như vậy, nếu coi mỗi trạng thái là một biến ngẫu nhiên nhận một giá trị $o_t$ nào đó, thì mỗi trạng thái sẽ có một phân phối nhất định. Ở đây ta gán phân phối này là phân phối chuẩn với giá trị kì vọng và phương sai .

Phân phối của ta như sau:

$$f(x \mid \ \mid = \ exp( - \ )$$

với x ở đây tạm coi là giá trị trong không gian vector của biến quan sát $o_t$. Như vậy:

$$b_j o_t = \ exp( - \ )$$

cụ thể ở đây $o_t$ là một vector 39 chiều thực, do đó $b_j o_t$ được tính lại theo công thức phân phối đối với vector đa chiều

$$b_j o_t =$$

Ở đây  và  lần lượt là kì vọng và phân phối của thành phần thứ d đối với trạng thái thứ *j*, hai giá trị này được tính theo công thức kì vọng và phương sai đối trong không gian một chiều:

$$=$$

$$=$$

Ở đây $X_{jd}$ chính là giá trị chiều thứ *d* của tất cả các biến $o_t$ của toàn bộ các mẫu trong tập huấn luyện và xác suất  tương ứng với $o_t$ của mẫu thứ k chính bằng $B_k(j, t)$. Như vậy, các xác suất trên đều có thể tính toán được, do đó ta hoàn toàn có được ma trận xác suất *A, B* như yêu cầu ban đầu.

#### 4.2.1.4. ***Nhận dạng (Decoding)***

*Thuật toán Viterbi*:

Phần "Decoding" nhận đầu vào là một dãy quan sát $O = o_1\ o_2\ ...\ o_T$ (đặc trưng cho tín hiệu tiếng nói) và cho ra một dãy được gán trạng thái có xác suất lớn nhất đối với tập training. Để làm được điều này, ta sử dụng giải thuật Viterbi.

Giả sử $A = a_{ij}$ và $B = b_j(o_t)$ tương ứng là ma trận chuyển trạng thái của tập training và phân phối các trạng thái đối với dãy quan sát $O$.

Gọi $F = f(j, t)$ là xác suất lớn nhất để quan sát thứ $t$ có trạng thái $j$, với giả thiết các $o_k$ $(k<t)$ đều đã có trạng thái. Khi đó $f(j, t)$ được tính như sau:

$$F(j, t) = max\ [\ F(i, t-1).\ a_{ij}.b_jo_t\ ]\ (với\ i = 1..n)$$

Vậy một $f(j_T, T)$ nào đó chính là xác suất mang trạng thái lớn nhất của quan sát $O$ ban đầu, và giá trị $f(j_T, T)$ này được tính từ một giá trị $f(j_{T-1}, T-1)$ nào đó, nếu thực hiện việc tìm kiếm như vậy, ta sẽ được dãy trạng thái có khả năng nhất của quan sát O đã cho:

$$O = j_1\ j_2\ ...\ j_T$$

Chú ý rằng các trạng thái ở đây chính là số thứ tự của các âm vị tạo thành tiếng nói, chẳng hạn từ "MOT" sẽ có dãy âm vị "m – oo – t" tương ứng với dãy trạng thái "1, 2, 3" (nếu coi thứ tự của các âm vị "m", "oo", "t" trong cả tập âm vị là 1, 2, 3).

Do đó, với từ "MOT" là tiếng nói đầu vào, thuật toán Viterbi sẽ đưa ra cho ta dãy trạng thái có khả năng nhất dạng tương tự như sau: 1 1 1 1 2 2 2 2 2 3 3 3 3 3, tương đương với m m m m oo oo oo oo oo t t t t t, như vậy ta có thể dễ dàng tổng hợp dãy trạng thái trên để biết từ vừa nói là "MOT", nói cách khác dãy trạng thái (âm vị) đó chính là kết quả của toàn hệ thống HMM ta vừa xây dựng.

### 4.3. **Sử dụng đường nét F0 phân biệt thanh điệu tiếng Việt**

Như ta đã trình bày khá rõ ràng về đặc điểm thanh điệu của các đường nét F0 cho từng thanh điệu tiếng Việt. Từ các giá trị nằm trên đường nét và dựa vào đặc điểm ta phân biệt từng thanh điệu một theo các tiêu chí đưa ra như sau, các tiêu chí hoàn toàn có thể trở

nên linh động trong quá trình thực hiện lập trình, tuy nhiên để phân biệt ta sẽ sử dụng một công cụ phân loại có sẵn.

### 4.3.1. Thanh ngã

Thanh ngã có đường nét và phổ gãy đột ngột ở giữa, từ cách tính đường nét F0 bằng AMDF có thể thấy đoạn gãy có giá trị bằng 0. Ta quan niệm đoạn gãy ở giữa sẽ nằm trong khoảng thứ tự từ N/3 tới 2N/3 (N là số giá trị của đường nét F0).

Minh họa đường nét F0 của thanh ngã, âm vị "ã" có đoạn gãy:

*. . . 60 76 105 152 210 238* ***0 0*** *233 200 136 106 96 . . .*

Ở đây các số thực được làm tròn thành số nguyên gần nhất để tiện quan sát.

### 4.3.2. Thanh ngang

Thanh ngang có đoạn giữa "bằng phẳng", đoạn giữa ở đây cũng được tính theo khái niệm đoạn giữa đối với thanh ngã. Bằng việc giá trị trung bình của đoạn giữa và tính độ lệch trung bình của đoạn giữa với giá trị trung bình đó, ta có thể đánh giá được đoạn đó là "bằng phẳng" hay không. Độ lệch trung bình chấp nhận ở đây ta chọn từ 3-5 tùy vào giá trị trung bình. Như trường hợp sau là giọng nam (độ cao thấp) ta chọn độ lệch trung bình là  cho phép là 3.

Minh họa đoạn giữa của thanh ngang, âm vị "a":

*. . . 81 83 83 83 83 82 82 83 84 84 83 83 83 83 83 83 83 83 84 84 . . .*

### 4.3.3. Thanh huyền

Đoạn giữa thanh huyền có độ cao tăng nhẹ nhưng ổn định và giảm về cuối, thông thường 1/3 đoạn cuối giữa (1/3 đoạn cuối của đoạn giữa) sẽ giảm đáng kể, vì vậy ta sẽ tính giá trị trung bình cho 2/3 đoạn đầu giữa và 1/3 đoạn cuối giữa, tính độ lệnh trung bình cho 2 đoạn này để đánh giá. Độ lệch trung bình cho phép của 2 đoạn này chọn trong khoảng (3-10).

Đoạn giữa của thanh huyền, âm vị "à" (đoạn giảm được in đậm):

*. . . 80 83 90 102 105 100 91 84 81 80 81 78 71* ***55 39 31 35 49*** *67 . . .*

Tuy nhiên, ở trên là phát âm lý tưởng trong môi trường thu âm lý tưởng, trong giao tiếp bình thường, do thói quen của người Việt, thanh huyền được phát âm y hệt thanh ngang, chỉ khác là cường độ được giảm đi làm người nghe có cảm tưởng âm đó mang thanh huyền nhưng thực chất đó gần như là thanh ngang được hạ giọng, người quen nghe giọng cao mà khi nghe người giọng trầm nói thanh ngang sẽ rất dễ nhầm đó là thanh huyền, nếu không nói tới ngữ cảnh. Vì lý do đó, trong phần thực nghiệm, ta sẽ không phân biệt thanh ngang và thanh huyền mà coi cả hai là thanh ngang.

### 4.3.4. Thanh sắc

Về phần cuối thanh sắc, đường nét đi xuống rồi lại đi lên rất nhanh. Hai phần này chiếm khoảng 1/4 độ dài toàn âm tiết và kéo dài gần như nhau, cắt phần cuối này thành 2 đoạn, nếu một phần là các giá trị nhỏ và tăng đột ngột ở phần sau thì ta xác định đó là thanh sắc. Đoạn cuối thanh sắc, âm vị "á" có hai phần in đậm là giảm và tăng nhanh:

*. . . 105 106 96 78* ***56 38 27 28 43*** *68* ***94 102 92 87 103 127 148*** *96*

### 4.3.5. Thanh nặng

Thanh nặng khá dễ nhận biết bởi có xu hướng đi lên, gần cuối tăng mạnh rồi giảm đột ngột (trái với thanh sắc). Dùng các kĩ thuật tương tự như trên ta có thể phân biệt được thanh nặng. Dãy số dưới minh họa đường nét thanh nặng, âm vị "ạ", phần in đậm là việc lên giọng cao và giảm đột ngột:

*. . . 83 84 83* ***83 95 109 114*** *114* ***106 95 83 80 71***

### 4.3.6. Thanh hỏi

Đường nét F0 của thanh hỏi thường có các giá trị lớn, điều này dễ hiểu bởi việc phát âm thanh hỏi tốn nhiều năng lượng hơn. Nửa cuối của thanh hỏi có dạng lõm giữa (tức là giảm rồi lại tăng), các giá trị hai bên của đoạn cuối thường ở mức gần kịch trần đối với tần số cơ bản của người nói (150Hz với giọng nam và 250Hz với giọng nữ). Đường nét sau minh họa một giọng nói nam, thanh hỏi, âm vị "ả":

*. . .* ***167 158 150 139 125*** *113* ***98 98 89 88 88 91 93 93 94 93*** *97* ***109 127 164 136*** *. . .*

Phần in đậm thể hiện giá trị cao và một đoạn "lõm giữa".

### 4.3.7. Phân lớp thanh điệu

Từ những tìm hiểu trên, ta có thể thấy đường nét F0 là một vector với các giá trị thực, số chiều của hai vector đường nét khác nhau không nhất thiết phải giống nhau kể cả khi hai đường nét đó đặc trưng cho cùng thanh điệu, chính vì vậy ta sử dụng phương pháp phân lớp bằng máy vector hỗ trợ (Support Vector Machine – SVM), đây là phương pháp phân loại vector rất tốt. Cụ thể về SVM là cả một cơ sở lý thuyết phân lớp khác, tôi không trình bày cụ thể ở đây mà chỉ sử dụng như một công cụ được chứng minh là tốt, điểm tôi muốn nhấn mạnh đó đặc điểm các vector đường nét như trên chính là cơ sở để áp dụng công cụ này.

# *Chương 5.* KẾT QUẢ THỰC NGHIỆM

Bằng các phương pháp và kĩ thuật như đã nêu ở các chương trước, sau khi lập trình được hệ thống nhận dạng và phân biệt thanh điệu với các bộ dữ liệu âm thanh tự xây dựng, ta thu được kết quả thực nghiệm được trình bày rõ ở các phần dưới đây.

## 5.1. Kết quả của nhận dạng tiếng nói sử dụng đặc trưng MFCC

***Bộ từ vựng:*** Là bộ tiếng nói các chữ số đếm

| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
|---|---|---|---|---|---|---|---|---|---|
| KHONG | MOT | HAI | BA | BON | NAM | SAU | BAY | TAM | CHIN |

***Tập âm vị*** (ở đây sil là silent, chỉ trạng thái vô thanh): bao gồm thứ tự và tên

| 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| a | a(y) | a(u) | i | y | o | u | b | h | kh | m | (a)m | n | (i)n | ng | s | t | (o)t | tr | sil |

***Từ điển đầy đủ*** có dạng từ vựng, đi kèm là thứ phát âm gồm thứ tự các âm vị nối tiếp nhau.

**Khong**: 20 10 6 15 20 (tức Khong = sil – kh – o – ng – sil)

**Mot**: 20 11 6 18 20

**Hai**: 20 9 1 5 20

**Ba**: 20 8 1 20

**Bon**: 20 8 6 14 20

**Nam**: 20 13 3 12 20

**Sau**: 20 16 3 7 20

**Bay**: 20 8 2 5 20

**Tam**: 20 17 1 12 20

**Chin**: 20 19 4 14 20

Thực hiện huấn luyện và nhận dạng trên 3 bộ huấn luyện và test, kết quả thể hiện ở bảng sau:

| Bộ | Số dữ liệu training | Số dữ liệu test | Nhận dạng đúng | Tỉ lệ đúng |
|---|---|---|---|---|
| 1 | 150 | 60 | **57** | **95.00%** |
| 2 | 150 | 50 | **46** | **92.00%** |
| 3 | 150 | 50 | **50** | **100.00%** |
| **Tổng** | 450 | 160 | **153** | **95.63%** |

Tỉ lệ nhận dạng chính xác trung bình là **95.63%**.

## 5.2. Kết quả phân biệt thanh điệu

### 5.2.1. Mô tả bộ dữ liệu

Bộ dữ liệu thanh điệu dựa trên bộ dữ liệu số đếm, đó là các từ "KHONG, MOT, ..., CHIN" được nói với 5 thanh điệu khác nhau: thanh ngang, sắc, hỏi, nặng, ngã. (Như đã nói ở mục *4.3.3* ta coi thanh huyền và thanh ngang là một do thói quen nói và điều kiện thu âm bộ test làm cho 2 thanh này quá giống nhau).

Tổng cộng có 150 dữ liệu mẫu (training) cho 10 chữ số đếm, mỗi từ là 15 mẫu, chia đều cho 5 thanh điệu, riêng từ "MOT" chỉ có thanh sắc và thanh nặng ("MỐT", "MỘT") gồm 8 thanh sắc và 7 thanh nặng. Chi tiết xem trong bảng dữ liệu training sau:

| Dữ liệu training | Thanh ngang | Thanh sắc | Thanh hỏi | Thanh nặng | Thanh Ngã | Tổng |
|---|---|---|---|---|---|---|
| **KHONG** | 3 | 3 | 3 | 3 | 3 | 15 |
| **MOT** | 0 | 8 | 0 | 7 | 0 | 15 |
| **HAI** | 3 | 3 | 3 | 3 | 3 | 15 |

| | | | | | | |
|---|---|---|---|---|---|---|
| … | … | … | … | … | … | … |
| **CHIN** | 3 | 3 | 3 | 3 | 3 | 15 |
| **Tổng** | 27 | 35 | 27 | 34 | 27 | 150 |

Dữ liệu test gồm 100 mẫu, mỗi từ là 10 mẫu chia đều cho 5 thanh điệu, riêng từ "MOT" gồm 5 thanh sắc và 5 thanh nặng. Xem thêm ở bảng dữ liệu test sau:

| **Dữ liệu test** | **Thanh ngang** | **Thanh sắc** | **Thanh hỏi** | **Thanh nặng** | **Thanh Ngã** | **Tổng** |
|---|---|---|---|---|---|---|
| **KHONG** | 2 | 2 | 2 | 2 | 2 | 10 |
| **MOT** | 0 | 5 | 0 | 5 | 0 | 10 |
| **HAI** | 2 | 2 | 2 | 2 | 2 | 10 |
| … | … | … | … | … | … | … |
| **CHIN** | 2 | 2 | 2 | 2 | 2 | 10 |
| **Tổng** | 18 | 23 | 18 | 23 | 18 | 100 |

### 5.2.2. Kết quả

Sử dụng máy vector hỗ trợ với hàm nhân là hàm đa thức. Ta thu được kết quả như bảng kết quả sau:

| **Tên dữ liệu** | **Số lượng test** | **Phân lớp chính xác** | **Tỉ lệ chính xác** |
|---|---|---|---|
| **Thanh ngang** | 18 | **17** | **94.44%** |
| **Thanh sắc** | 23 | **21** | **91.30%** |
| **Thanh hỏi** | 18 | **15** | **83.33%** |
| **Thanh nặng** | 23 | **20** | **86.96%** |

| | | | |
|---|---|---|---|
| **Thanh ngã** | 18 | **14** | **77.78%** |
| **Tổng** | 100 | **87** | **87.00%** |

Độ chính xác trung bình: **87.00%**

## 5.3. **Nhận xét**

Đối với độ chính xác của hệ nhận dạng tiếng nói khi dùng trích chọn đặc trưng MFCC (95.63%) là chưa thật cao (so với hệ thống Sphinx4 của Sun với bộ chữ số đếm tiếng Anh là xấp xỉ 100% *[8]*). Điều này do bộ training do chúng tôi tự xây dựng không được thu trong môi trường lý tưởng, còn nhiễu trong mẫu training, thêm nữa là khuôn khổ thời gian chưa đủ nhiều để có thể nghiên cứu sâu và rộng hơn về các kĩ thuật trích chọn đặc trưng và huấn luyện. Tuy nhiên theo chúng tôi độ chính xác của hệ thống được xem là chấp nhận được, đối với mục đích nghiên cứu. Trong tương lai, với các thiết bị thu âm chuẩn xác và lọc ồn tốt hơn, về khách quan trích chọn đặc trưng MFCC sẽ có xu hướng mô phỏng tốt hơn giọng nói, về chủ quan, các nghiên cứu sâu hơn về các đặc tính vật lý, sinh học của giọng nói chắc chắn cũng nâng cao độ chính xác của hệ nhận dạng.

Còn với việc phân biệt thanh điệu bằng phân loại đường nét F0 với SVM, các nghiên cứu hiện tại đều chưa ở mức sâu đối với tiếng Việt nên nhìn chung độ chính xác chưa cao (87.00%). Rất tiếc do thời gian tìm hiểu có hạn, tôi chưa tìm được báo cáo kết quả nào về việc phân biệt thanh điệu tiếng Việt để có thể so sánh với kết quả thực nghiệm của mình.

Độ chính xác còn thấp như nói ở trên chỉ có thể khắc phục bằng cách có những nghiên cứu sâu hơn về các đặc trưng của tiếng Việt. Nếu thành công đây sẽ là một ưu điểm rất lớn bởi thanh điệu trước hết sẽ giúp cho quá trình phân cụm, làm giảm kích thước bộ từ vựng trước khi đưa vào nhận dạng, thêm nữa về hướng ứng dụng, thanh điệu giúp việc tổng hợp tiếng nói có ngữ điệu trở nên tốt hơn, ứng dụng trước mắt là tổng hợp tiếng nói dành cho robot.

# *Chương 6.* KẾT LUẬN CHUNG VÀ ĐỊNH HƯỚNG NGHIÊN CỨU PHÁT TRIỂN TRONG TƯƠNG LAI

Chương cuối này của khóa luận tổng kết lại các vấn đề đã nghiên cứu, những vấn đề nào tác giả đã giải quyết và hoàn thành, các vấn đề còn tồn tại và định hướng nghiên cứu trong tương lai.

## 6.1. Các vấn đề đã nghiên cứu và hoàn thành

Trong khóa luận này, chúng tôi đã nghiên cứu và thực hiện thành công việc trích chọn đặc trưng MFCC và tính toán làm mịn đường nét F0 đặc trưng cho thanh điệu tiếng Việt, qua đó áp dụng mô hình Markov ẩn (Hidden Markov Model – HMM) để nhận dạng

cho tiếng nói có đặc trưng MFCC và máy vector hỗ trợ (Support Vector Machine – SVM) để phân lớp đường nét thanh điệu.

Kết quả thu được chúng tôi đánh giá là chấp nhận được đối với một khóa luận cử nhân Công Nghệ Thông Tin, từ nền tảng này chắc chắn chúng tôi sẽ nghiên cứu các vấn đề trên một một cách kĩ lưỡng hơn để nâng tầm sản phẩm nghiên cứu của mình trở thành sản phẩm ứng dụng.

## 6.2. **Các vấn đề tồn tại**

Đối với việc trích chọn đặc trưng nói chung:

Trích chọn MFCC còn vấn đề về tín hiệu đầu vào, tín hiệu càng ít ồn, nhiễu thì đặc trưng MFCC càng thể hiện đúng đặc trưng của tiếng nói, trong khi với các thiết bị thu âm bình thường, việc có ồn và nhiễu là không thể tránh khỏi, chúng tôi buộc phải chấp nhận tồn tại này.

Đối với việc tìm và làm mịn đường nét thanh điệu F0, việc tìm đường nét vẫn là các biến đổi chưa thật sự phức tạp đối với đường nét thô (chính là phổ biên độ ban đầu của tín hiệu), vì vậy kết quả phân lớp đường nét F0 còn ở mức chưa cao. Một tồn tại nữa của việc làm mịn đường nét F0 bằng phương pháp AMDF là phân biệt thanh điệu kém hiệu quả với tiếng nói liên tục.

Về cả hệ thống nhận dạng tiếng nói nói chung, vấn đề tồn tại thực chất gắn với giới hạn nghiên cứu chúng tôi đặt ra ban đầu, trong khuôn khổ thời gian của một khóa luận tốt nghiệp, hệ thống chúng tôi xây dựng được là "phụ thuộc người nói" và "nhận dạng tiếng nói rời rạc". Trong khi một hệ thống nhận dạng tiếng nói hoàn hảo theo đúng nghĩa cần phải nhận dạng tiếng nói của nhiều người khác nhau, và tiếng nói ấy là tiếng nói trong giao tiếp bình thường: liên tục và có ngữ cảnh.

## 6.3. **Định hướng nghiên cứu phát triển trong tương lai**

Rất dễ hình dung về tính ứng dụng của việc nhận dạng giọng nói trong cả hiện tại và tương lai: giao tiếp người – máy, tổng hợp giọng nói cho robot, điều khiển máy bằng lệnh giọng nói... Những nhu cầu ứng dụng đó dẫn xuất cuối cùng tới việc phải độ chính xác

của hệ nhận dạng tiếng nói các ngôn ngữ nói chung và tiếng nói tiếng Việt nói riêng. Đây sẽ luôn là định hướng chung của tất cả các nhà nghiên cứu, chứ không chỉ riêng chúng tôi.

Những định hướng nghiên cứu phục vụ tăng độ chính xác của việc nhận dạng tiếng nói không chỉ tập trung vào quá trình Feature Extraction mà còn tập trung vào quá trình training, mở rộng bộ từ vựng hay xây dựng mô hình thống kê n-gram tốt phục vụ nhận dạng giọng nói liên tục... Chắc chắn việc tìm hiểu nghiên cứu Feature Extraction nói riêng và nhận dạng tiếng nói nói chung sẽ vẫn là đề tài mở và hấp dẫn với cộng đồng nghiên cứu khoa học.

* * * * * * * * * *

Trong quá trình hoàn thành khóa luận này, chắc chắn không tránh khỏi những khuyết điểm, thiếu sót, tôi rất mong nhận được những lời khuyên cũng như góp ý để có thể tự khắc phục và hoàn thiện tốt hơn trong tương lai, xin chân thành cảm ơn.

# PHỤ LỤC

**Source code xây dựng được**:

***FullFeatureExtractor.java:***

Phương thức *FeatureVector [] extract(String filePath)*: Lấy ra dãy Vector đặc trưng của file tiếng nói.

Các class đi kèm:

- *AudioDataReader*

- *AudioFrame*
- *WindowingProcessor*
- *DFTProcessor*
- *MelFilterBankProcessor*
- *DCTProcessor*

Các class này thực hiện các bước biến đổi để thu được đặc trưng MFCC. Hai class *DFTProcessor* và *DCTProcessor* có sử dụng thư viện biến đổi cosin và FFT của apache (apache.org)

***VietnamsesToneFeature.java:***

Phương thức lấy đặc trưng:

*ArrayList<Double> getFeatures(String fileName)*: Lấy đường nét của âm thanh trong file đầu vào.

Các phương thức khác:

*ArrayList<Double> FeatureWithTime(**double** time)*: lấy ra đường nét của tín hiệu với điều kiện sau một khoảng thời gian *time* lại tính một giá trị của đường nét

*ArrayList<Double> SmoothF0(ArrayList<Double> f0)*: làm mịn đường nét bằng các phép tính trung bình và qua bộ lọc có đáp ứng xung (0.1, 0.2, 0.4, 0.2, 0.1)

## Source code sử dụng lại để training, nhận dạng:

***HMMDecoder.java*:**

Được code và sử dụng lại các class training đi kèm để phù hợp cho việc chạy bộ dữ liệu thực nghiệm. Class này có sử dụng class *FullFeatureExtractor.java* để lấy các đặc trưng MFCC cho việc tính toán trong quá trình Training.

Nguồn: http://www.run.montefiore.ulg.ac.be

***svm_train.java*** và ***svm_predict.java*****:**

Hai class này nhận đầu vào là file tự định dạng chuẩn các vector đường nét F0, được lấy từ class ***VietnameseToneFeature.java***, chúng sử dụng bằng cách biên dịch và chạy trên máy ảo java, các tham số ngoài là các file dạng chuẩn được quy định trước.

Một ví dụ đơn giản: chạy svm_train bằng command line:

- *javac svm_train.java* (biên dịch svm_train)
- *java svm_train –t 1 Train* (huấn luyện file Train, hàm nhân – kernel fuction là hàm đa thức, biểu diễn bởi tham số -t 1, kết quả sẽ sinh ra file Train.model)
- *javac svm_predict.java* (biên dịch svm_predict)
- *java svm_predict Test Train.model output* (phân lớp các vector trong file Test, dựa vào mô hình huấn luyện Train.model, kết quả gi ra file output)

Các file Test, Train ở đây được ghi theo định dạng quy định của svm_train và svm_predict.

**Tài liệu tham khảo:**


[1] A. Stoclke, E. Shriberg, L. Ferrer, S. Kajarekar, K. Sonmez, G. Tur. *"Speech Recognition As Feature Extraction For Speaker Recognition", 2007.*

[2] Daniel Jurafsky & Jame H.Martin. *"Speech and Language Processing", 2007, chapter 9*

[3] Ilyas Potamitis, Nikos Fakotakis, George Kokkinakis. *"Speech Recognition Based On Feature Extraction With Variable Rate Frequency Sampling"*. Nguồn:

*http://www.springerlink.com/content/mcwpdr59pwkn88yv/*

[4] Joseph Keshet & Samy Bengio. *"Automatic Speech and Speaker Recognition", 2009.*

[5] Mark Gales & Steve Young. *"The Application of Hidden Markov Models in Speech Recognition", 2007.*

[6] Qifeng Zhu, Abeer Alwan. *"Non-linear feature extraction for robust speech recognition in stationary and non-stationary noise", 2003.*

[7] Shrikanth, Panayiotis Georgiou, Abhinav Sethy. *"Implicit Pronunciation Modeling for Speech Recognition Using Syllable-Centric Models"*. Nguồn:

imsc.usc.edu/research/project/pronun/pronun_nsf8.pdf

[8] Willie Walker, Paul Lamere, Philip Kwok, Bhiksha Raj, Rita Singh, Evandro Gouvea, Peter Wolf, Joe Woelfel. *"Sphinx4 white paper – A Flexible Open Source Framework for Speech Recognition", 2004.*

[9] Hoàng Đình Chiến. *"Nhận dạng tiếng Việt dùng mạng Neuron kết hợp trích đặc trưng dùng LPC và AMDF", 2005.*

[10] Nguyễn Văn Giáp, Trần Việt Hồng. *"Kĩ thuật nhận dạng tiếng nói và ứng dụng trong điều khiển", 2008.*

[11] Bạch Hưng Nguyên, Nguyễn Tiến Dũng. *"Mô hình Fujisaki và áp dụng trong phân tích thanh điệu tiếng Việt"*. Nguồn:
www.cs.cmu.edu/~nbach/papers/Fujisaki_ThaiNguyen082003.pdf

[12] "*Nhận dạng tiếng nói bằng mạng nơ ron, mô phỏng Matlab", 2009.* Nguồn: http://www.ebook.edu.vn/?page=1.39&view=4861