**­­­­­­­­­­ĐẠI HỌC QUỐC GIA TP. HỒ CHÍ MINH**

**TRƯỜNG ĐẠI HỌC CÔNG NGHỆ THÔNG TIN**

**KHOA CÔNG NGHỆ PHẦN MỀM**

**HUỲNH THANH HUY**

**NGUYỄN HOÀNG VŨ**

**KHÓA LUẬN TỐT NGHIỆP**

**VOICE RECOGNITION**

**KỸ SƯ NGÀNH KỸ THUẬT PHẦN MỀM**

**TP. HỒ CHÍ MINH, 2015**

**ĐẠI HỌC QUỐC GIA TP. HỒ CHÍ MINH**

**TRƯỜNG ĐẠI HỌC CÔNG NGHỆ THÔNG TIN**

**KHOA CÔNG NGHỆ PHẦN MỀM**

**HUỲNH THANH HUY - 11520158**

**NGUYỄN HOÀNG VŨ – 11520489**

**KHÓA LUẬN TỐT NGHIỆP**

**VOICE RECOGNITION**

**KỸ SƯ NGÀNH KỸ THUẬT PHẦN MỀM**

**GIẢNG VIÊN HƯỚNG DẪN**

**HUỲNH TUẤN ANH**

**TP. HỒ CHÍ MINH, 2015**

DANH SÁCH HỘI ĐỒNG BẢO VỆ KHÓA LUẬN

Hội đồng chấm khóa luận tốt nghiệp, thành lập theo Quyết định số ……...........… ngày ………………….. của Hiệu trưởng Trường Đại học Công nghệ Thông tin.

* 1. …………………………………………. – Chủ tịch.
  2. …………………………………………. – Thư ký.
  3. …………………………………………. – Ủy viên.
  4. …………………………………………. – Ủy viên.

|  |  |
| --- | --- |
| ĐẠI HỌC QUỐC GIA  TP. HỒ CHÍ MINH  **TRƯỜNG ĐẠI HỌC**  **CÔNG NGHỆ THÔNG TIN** | **CỘNG HÒA XÃ HỘI CHỦ NGHĨA VIỆT NAM**  **Độc Lập - Tự Do - Hạnh Phúc** |
|  | *TP. HCM, ngày…..tháng…..năm 2015* |

**NHẬN XÉT KHÓA LUẬN TỐT NGHIỆP**

**(CỦA CÁN BỘ HƯỚNG DẪN)**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Tên khóa luận:** | | | | | | |
| **Voice Recognition** | | | | | | |
| **Nhóm SV thực hiện:** | | | | **Cán bộ hướng dẫn:** | | |
| Nguyễn Hoàng Vũ | |  | | | Huỳnh Tuấn Anh | |
| Huỳnh Thanh Huy | |  | | |  | |
| **Đánh giá Khóa luận**   1. Về cuốn báo cáo:   Số trang Số chương  Số bảng số liệu Số hình vẽ  Số tài liệu tham khảo Sản phẩm  Một số nhận xét về hình thức cuốn báo cáo:       1. Về nội dung nghiên cứu:        1. Về chương trình ứng dụng:        1. Về thái độ làm việc của sinh viên:       **Đánh giá chung:**      **Điểm từng sinh viên:**  Huỳnh Thanh Huy**: ………../10**  Nguyễn Hoàng Vũ**: ………../10** | | | | | | | | |
|  | | | | | **Người nhận xét**  (Ký tên và ghi rõ họ tên)  ………………………………… | | |
| ĐẠI HỌC QUỐC GIA  TP. HỒ CHÍ MINH  **TRƯỜNG ĐẠI HỌC**  **CÔNG NGHỆ THÔNG TIN** | | | **CỘNG HÒA XÃ HỘI CHỦ NGHĨA VIỆT NAM**  **Độc Lập - Tự Do - Hạnh Phúc** | | | | |
|  | | | *TP. HCM, ngày…..tháng…..năm 2015* | | | | |

**NHẬN XÉT KHÓA LUẬN TỐT NGHIỆP**

**(CỦA CÁN BỘ PHẢN BIỆN)**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Tên khóa luận:** | | | | | |
| **VOICE RECORGNITION** | | | | | |
| **Nhóm SV thực hiện:** | | **Cán bộ phản biện:** | | | |
| Nguyễn Hoàng Vũ |  | | | ....................................... | |
| Huỳnh Thanh Huy |  | | | ....................................... | |
| **Đánh giá Khóa luận**   1. Về cuốn báo cáo:   Số trang Số chương  Số bảng số liệu Số hình vẽ  Số tài liệu tham khảo Sản phẩm  Một số nhận xét về hình thức cuốn báo cáo:       1. Về nội dung nghiên cứu:        1. Về chương trình ứng dụng:        1. Về thái độ làm việc của sinh viên:       **Đánh giá chung:**      **Điểm từng sinh viên:**  Huỳnh Thanh Huy**: ………../10**  Nguyễn Hoàng Vũ**: ………../10** | | | | | |
|  | | | **Người nhận xét**  (Ký tên và ghi rõ họ tên)  ………………………………… | |

**LỜI CẢM ƠN**

Lời đầu tiên, cho phép nhóm em xin gửi lời cảm ơn sâu sắc nhất đến các thầy, cô giáo của trường Đại Học Công Nghệ Thông Tin Thành phố Hồ Chí Minh, những người đã tận tình giảng dạy, truyền đạt cho chúng em những kiến thức quý báu và cần thiết trong suốt những năm học tập tại trường.

Đặc biệt em xin gửi lời cảm ơn chân thành nhất tới thầy Huỳnh Tuấn Anh, người đã quan tâm, động viên, tận tình hướng dẫn và giúp đỡ chúng em trong suốt quá trình chúng em làm khóa luận tốt nghiệp, để chúng em có thể hoàn thanh tốt khóa luận của mình.

Và cuối cùng, cảm ơn gia đình đã tạo điều kiện cho chúng em hoàn thành việc của mình trong suốt khoảng thời gian học tập trên giảng đường và có điều kiện thuận lợi nhất để nhóm em hoàn thành tốt khóa luận này.

***Xin chân thành cảm ơn.***

Nhóm sinh viên thực hiện:

***Huỳnh Thanh Huy***

***Nguyễn Hoàng Vũ***

|  |  |
| --- | --- |
| ĐẠI HỌC QUỐC GIA  TP. HỒ CHÍ MINH  **TRƯỜNG ĐẠI HỌC**  **CÔNG NGHỆ THÔNG TIN** | **CỘNG HÒA XÃ HỘI CHỦ NGHĨA VIỆT NAM**  **Độc Lập - Tự Do - Hạnh Phúc** |

**ĐỀ CƯƠNG CHI TIẾT**

|  |  |
| --- | --- |
| **TÊN ĐỀ TÀI: “Voice Recognition”** | |
| **Cán bộ hướng dẫn: Huỳnh Tuấn Anh** | |
| **Thời gian thực hiện:** Từ ngày 3/10/2015 đến ngày 19/12/2015 | |
| **Sinh viên thực hiện:**   1. Huỳnh Thanh Huy – 11520158 2. Nguyễn Hoàng Vũ – 11520489. | |
| **Nội dung đề tài:***(Mô tả chi tiết mục tiêu, phạm vi, đối tượng, phương pháp thực hiện, kết quả mong đợi của đề tài)*   * Tìm hiểu về nhận dạng giọng nói, xây dựng ứng dụng hỗ trợ bé Bé học từ tiếng Việt . * **Kết quả mong đợi**: Góp phần khởi tạo thêm data về nhận dạng giọng nói. Ứng dụng có thể giúp trẻ em đặc biệt là trẻ em khiếm thính , giáo viên dạy trẻ khiếm thính tập nói cho các bé dễ dàng hơn. | |
| **Kế hoạch thực hiện:***(Mô tả kế hoạch làm việc và phân công công việc cho từng sinh viên tham gia)*   |  |  | | --- | --- | | **Mô tả công việc** | **Phân công** | | 1. Tìm hiểu các thư viện hỗ trợ nhận dạng giọng nói | Huỳnh Thanh Huy, Nguyễn Hoàng Vũ | | 1. Phân tích đánh giá các ứng dụng về nhận dạng giọng nói | Huỳnh Thanh Huy | | 1. Tìm hiểu, lên ý tưởng viết ứng dụng minh họa cho thư viện CMU Sphinx | Nguyễn Hoàng Vũ | | 1. Phân tích cấu trúc, thiết kế, tích hợp CMU Sphinx | Nguyễn Hoàng Vũ | | 1. Khảo sát nhu cầu nhận dạng tại trung tâm trẻ khiếm thính Thuận An | Huỳnh Thanh Huy | | 1. Thiết kế ứng dụng minh họa – Bé học từ | Huỳnh Thanh Huy | | 1. Tìm kiếm, thiết kế hình ảnh cho ứng dụng | Nguyễn Hoàng Vũ | | 1. Xây dựng dữ liệu từ vựng | Huỳnh Thanh Huy, Nguyễn Hoàng Vũ | | 1. Ghi âm dữ liệu âm thanh phục vụ cho ứng dụng | Huỳnh Thanh Huy, Nguyễn Hoàng Vũ | | 1. Huấn luyện để tích hợp nhận dạng | Nguyễn Hoàng Vũ | | 1. Thực hiện xây dựng, tích hợp, kiểm thử để hoàn tất ứng dụng demo nhận dạng Bé học từ | Nguyễn Hoàng Vũ, Huỳnh Thanh Huy | | 1. Tổng hợp tài liệu và viết báo cáo | Huỳnh Thanh Huy, Nguyễn Hoàng Vũ | | |
| **Xác nhận của CBHD**  (Ký tên và ghi rõ họ tên) | **TP. HCM, ngày 19 tháng 12 năm 2015**  **Sinh viên**  (Ký tên và ghi rõ họ tên) |

MỤC LỤC

[**LỜI MỞ ĐẦU 1**](#_Toc438916266)

[**TỔNG QUAN ĐỀ TÀI 4**](#_Toc438916267)

[**1. Lý do chọn đề tài 4**](#_Toc438916268)

[**2. Mục tiêu và phạm vi đề tài 4**](#_Toc438916269)

[**2.1. Mục tiêu 4**](#_Toc438916270)

[**2.2. Phạm vi đề tài 4**](#_Toc438916271)

[**CHƯƠNG 1 : CƠ SỞ LÝ THUYẾT VỀ NHẬN DẠNG GIỌNG NÓI 5**](#_Toc438916272)

[**1.1 HỆ NHẬN DẠNG GIỌNG NÓI 5**](#_Toc438916273)

[**1.1.1 Tổng quan 5**](#_Toc438916274)

[**1.1.2 Các hệ nhận dạng giọng nói 6**](#_Toc438916275)

[**1.1.3 Một số phương pháp nhận dạng giọng nói 6**](#_Toc438916276)

[**1.2 RÚT TRÍCH ĐẶC TRƯNG TÍN HIỆU GIỌNG NÓI 8**](#_Toc438916277)

[**1.2.1 Giới thiệu 8**](#_Toc438916278)

[**1.2.2 Tiền khuếch đại 8**](#_Toc438916279)

[**1.2.3 Tách từ 9**](#_Toc438916280)

[**1.2.4 Phân đoạn 9**](#_Toc438916281)

[**1.2.5 Lấy cửa sổ khung tín hiệu 10**](#_Toc438916282)

[**1.2.6 Rút trích đặc trưng 11**](#_Toc438916283)

[**1.2.7 Tìm hiểu về Formant 25**](#_Toc438916284)

[**1.3 GAUSSIAN MIXTURE MODEL 27**](#_Toc438916285)

[**1.4 MÔ HÌNH MARKOV ẨN 31**](#_Toc438916286)

[**1.4.1 Giới thiệu 31**](#_Toc438916287)

[**1.4.2 Ba bài toán cơ bản của mô hình Markov ẩn 33**](#_Toc438916288)

[**1.4.3 Thuật toán lan truyền xuôi 33**](#_Toc438916289)

[**1.4.4 Thuật toán lan truyền ngược 34**](#_Toc438916290)

[**1.4.5 Thuật toán lan truyền xuôi – ngược 35**](#_Toc438916291)

[**1.4.6 Thuật toán Viterbi 35**](#_Toc438916292)

[**1.4.7 Thuật toán Baum-Welch 36**](#_Toc438916293)

[**1.5 MÔ HÌNH HMM TRONG NHẬN DẠNG GIỌNG NÓI 37**](#_Toc438916294)

[**1.5.1 Giới thiệu 37**](#_Toc438916295)

[**1.5.2 HMM trong nhận dạng giọng nói 37**](#_Toc438916296)

[**CHƯƠNG 2:CÔNG CỤ HUẤN LUYỆN VÀ NHẬN DẠNG GIỌNG NÓI 43**](#_Toc438916297)

[**2.1 TỔNG QUAN 43**](#_Toc438916298)

[**2.2 HUẤN LUYỆN VÀ NHẬN DẠNG GIỌNG NÓI VỚI CMU SPHINX 43**](#_Toc438916299)

[**2.2.1 Giới thiệu 43**](#_Toc438916300)

[**2.2.2 Vận dụng mô hình HMM trong nhận dạng giọng nới với Sphinx4 44**](#_Toc438916301)

[**2.2.3 Các thành phần chính trong CMU Sphinx 44**](#_Toc438916302)

[**2.2.4 Các xử lý chính trong CMU Sphinx 50**](#_Toc438916303)

[**2.3 KẾT LUẬN 51**](#_Toc438916304)

[**2.3.1 Ưu điểm 51**](#_Toc438916305)

[**2.3.2 Nhược điểm 53**](#_Toc438916306)

[**2.4 SPHINX 4 VỚI NGÔN NGỮ TIẾNG VIỆT 54**](#_Toc438916307)

[**2.4.1 Tổng quan 54**](#_Toc438916308)

[**2.4.2 Corpus 54**](#_Toc438916309)

[**CHƯƠNG 3: XÂY DỰNG CHƯƠNG TRÌNH MINH HỌA TRÊN THIẾT BỊ DI ĐỘNG 56**](#_Toc438916310)

[**3.1 THIẾT LẬP MÔI TRƯỜNG CMU SPHINX 56**](#_Toc438916312)

[**3.1.1 Chuẩn bị hệ điều hành 56**](#_Toc438916313)

[**3.1.2 Chuẩn bị các gói cài đặt Sphinx 56**](#_Toc438916314)

[**3.2 XÂY DỰNG BỘ NGÔN NGỮ 56**](#_Toc438916315)

[**3.2.1 Giới thiệu 56**](#_Toc438916316)

[**3.2.2 Xây dựng bộ từ điển 57**](#_Toc438916317)

[**3.2.3 Xây dựng mô hình ngôn ngữ 58**](#_Toc438916318)

[**3.2.4 Xây dựng mô hình âm học 60**](#_Toc438916319)

[**3.3 CẤU HÌNH SPHINX 67**](#_Toc438916320)

[**3.3.1 Cấu hình thư mục huấn luyện 67**](#_Toc438916321)

[**3.3.2 Điều chỉnh tham số huấn luyện 68**](#_Toc438916322)

[**3.3.3 Thực thi huấn luyện 70**](#_Toc438916323)

[**3.4 SỬ DỤNG KẾT QUẢ HUẤN LUYỆN 71**](#_Toc438916324)

[**3.4.1 Giới thiệu ứng dụng 71**](#_Toc438916325)

[**3.4.2 Cấu trúc ứng dụng 72**](#_Toc438916326)

[**3.4.3 Hình ảnh demo 73**](#_Toc438916327)

[**3.5 THỬ NGHIỆM VÀ ĐÁNH GIÁ KẾT QUẢ 79**](#_Toc438916328)

[**3.6 KẾT LUẬN 79**](#_Toc438916329)

[**TÀI LIỆU THAM KHẢO 81**](#_Toc438916330)

[**PHỤ LỤC 85**](#_Toc438916331)

**CHÚ THÍCH HÌNH ẢNH**

[***Hình 1. 0.1Sơ đồ nhận dạng giọng nói*** 5](#_Toc438972878)

[***Hình 1. 0.2Quá trình rút trích đặc trưng*** 8](#_Toc438972879)

[***Hình 1. 0.3Phân đoạn tiếng nói thành các khung.*** 9](#_Toc438972880)

[***Hình 1. 0.4Đồ thì biểu diễn mối quan hệ giữa Mel và Hz.*** 12](#_Toc438972881)

[***Hình 1. 0.5Bộ lọc trên thang Mel.*** 13](#_Toc438972882)

[***Hình 1. 0.6 Bộ lọc trên tần số thật***. 13](#_Toc438972883)

[***Hình 1. 0.7 Minh họa các bước biến đổi MFCC*** 13](#_Toc438972884)

[***Hình 1. 0.8 Biểu đồ thang tần số Mel theo tần số thực.*** 15](#_Toc438972885)

[***Hình 1. 0.9 Băng lọc tần số Mel.*** 16](#_Toc438972886)

[***Hình 1. 0.10 Đưa tín hiệu vào băng lọc tần số Mel***. 16](#_Toc438972887)

[***Hình 1. 0.11 Mẫu tiếng nói dự báo tuyến tính*** 19](#_Toc438972888)

[***Hình 1. 0.12 Mẫu phân tích tiếng nói theo phương pháp LPC.*** 20](#_Toc438972889)

[***Hình 1. 0.13 Minh hoạ hiện tượng cộng hưởng*** 25](#_Toc438972890)

[***Hình 1. 0.14 Minh hoạ Formant*** 26](#_Toc438972891)

[***Hình 1. 0.15 Hàm mật độ Gauss.*** 27](#_Toc438972892)

[***Hình 1. 0.16 Mô hình GMM.*** 28](#_Toc438972893)

[***Hình 1. 0.17 Hàm mật độ của GMM có 3 phân phối Gauss*** 29](#_Toc438972894)

[***Hình 1. 0.18 Các ứng dụng của mô hình Markov ẩn.*** 31](#_Toc438972895)

[***Hình 1. 0.19 Chuỗi Markov với 5 trạng thái (S1 đến S5).*** 32](#_Toc438972896)

[***Hình 1. 0.20 Thuật toán lan truyền xuôi – ngược.*** 35](#_Toc438972897)

[***Hình 1. 0.21 Ước lượng Baum - Welch*** 37](#_Toc438972898)

[***Hình 1. 0.22 Biến đổi tín hiệu âm thanh sang tín hiệu số.*** 39](#_Toc438972899)

[***Hình 1. 0.23 (a) - Tín hiệu tương tự, (b) - Tín hiệu số.*** 39](#_Toc438972900)

[***Hình 1. 0.24 Rút trích đặc trưng giọng nói.*** 40](#_Toc438972901)

[***Hình 1. 0.25 Mô hình máy nhận dạng giọng nói với HMM.*** 41](#_Toc438972902)

[***Hình 1. 0.26 Nhận dạng tiếng nói với HTK*** 41](#_Toc438972903)

[***Hình 1. 0.1 Kiến trúc hệ thống Sphinx4*** 45](#_Toc438972904)

[***Hình 1. 0.2. Kiến trúc thành phần Frontend*** 46](#_Toc438972905)

[***Hình 1. 0.3 Chuỗi các xử lý của thành phần Frontend.*** 46](#_Toc438972906)

[***Hình 1. 0.4 Thành phần Search graph trong giải mã tiếng Việt*** 47](#_Toc438972907)

[***Hình 1. 0.5 Quá trình nhận dạng.*** 51](#_Toc438972908)

[***Hình 1. 0.1 Sơ đồ quá trình tạo mô hình ngôn ngữ.*** 58](#_Toc438972909)

[***Hình 1. 0.2 Thiết lập thông số Rate và Format.*** 66](#_Toc438972910)

[***Hình 1. 0.3 Thiết lập thông số về Channel*** 66](#_Toc438972911)

[***Hình 1. 0.4 Thiết lập định dạng lưu tập tin.*** 67](#_Toc438972912)

[***Hình 1. 0.5 Thông báo trạng thái trong quá trình huấn luyện*** 71](#_Toc438972913)

[***Hình 1. 0.6 Màn hình chính khi vào Game*** 73](#_Toc438972914)

[***Hình 1. 0.7 Màn hình hướng dẫn chơi*** 74](#_Toc438972915)

[***Hình 1. 0.8 Màn hình lưu điểm của bé*** 75](#_Toc438972916)

[***Hình 1. 0.9 Menu chọn game*** 76](#_Toc438972917)

[***Hình 1. 0.10 Màn hình game chính*** 77](#_Toc438972918)

[***Hình 1. 0.11 Màn hình khi bé đọc đúng từ*** 78](#_Toc438972919)

# LỜI MỞ ĐẦU

Giao tiếp là vấn đề quan trọng, và đặc biệt giao tiếp bằng giọng nói được sử dụng như một phương tiện đặt biệt quan trọng, trong vấn đề truyền tải thông tin hằng ngày. Với sự phát triển của Công Nghệ Thông Tin đòi hỏi vấn đề số hóa giọng nói trở nên rất quan trọng. Hiện nay, trên thế giới vấn đề nhận dạng giọng nói đặc biệt là tiếng Anh đã không còn quá mới mẻ và đạt được nhiều thành tựu có thể kể đến như Google Now, Cortana giúp cho con người có thể giao tiếp với máy tính một cách nhanh chóng và dễ dàng hơn, giúp thực hiện nhiều việc quan trọng có thể kể đến nhà thông minh, trợ lý ảo, giáo viên ngôn ngữ. Đối với việc nhận dạng Tiếng Việt còn một số khó khăn, do các API hiện tại ít hỗ trợ, chính vì thế nhóm chúng em muốn thực hiện khóa luận này với chủ để tìm hiểu nhận dạng giọng nói tiếng Việt và xây dựng ứng dụng demo quá trình nhận dạng , cùng với sự nỗ lực đó nhóm chúng em cũng muốn góp một phần nhỏ của mình vào việc tìm hiểu bộ nhận dạng giọng nói tiếng việt, góp một phần nhỏ sức lực của mình vào việc hoàn thành, nâng cấp nhận dạng giọng nói Tiếng Việt. Trong quá trình thực hiện, do có một số khó khăn chắc chắn không thể hoàn hảo trong quá trình nhận dạng, ứng dụng còn gặp nhiều lỗi nhận dạng không chính xác mong được thứ lỗi. Khóa Luận bao gồm 2 phần chính: Tìm hiểu nhận dạng thông qua CMU Sphinx và xây dựng ứng dụng demo.

**TÌNH HÌNH NHẬN DẠNG GIỌNG NÓI TRONG VÀ NGOÀI NƯỚC**

Nghiên cứu các khái niệm liên quan đến kỹ thuật nhận dạng tiếng nói đã được bắt đầu khá sớm từ 1936 tại phòng thí nghiệm Bell. Tuy nhiên không có sự tiến bộ nào đáng kể cho đến năm 1970 khi Lenny Baum của Đại học Princeton đã phát minh ra mô hình Markov ẩn (Hidden Markov Model), từ đó cung cấp một phương pháp thống kê để tạo ra văn bản tương ứng từ tiếng nói. Đây cũng chính là cơ sở lý thuyết để cho ra đời những công cu hỗ trợ huấn luyện và nhận dạng giọng nói như Hidden Markov Model Toolkit (HTK) hay công cụ CMU Sphinx của trường Đại học Carnegie Mellon.

Nhận dạng giọng nói với các ngôn ngữ phổ biến như tiếng Anh, tiếng Pháp, tiếng Tây Ban Nha đã thu được các kết quả rất tốt, có nhiều ứng dụng thực tiễn như:

* Hệ thống giao tiếp hỏi đáp thông tin tự động qua điện thoại.
* Hệ thống truy vấn thông tin thoại.
* Hệ thống thông dịch tiếng nói xuyên ngữ tự động.
* Các ứng dụng tiếng nói trên thiết bị di động như trình duyệt web,…
* Các phần mềm điều kiển máy tính như Voice Finger, Speech Recognition,…

Ở Việt Nam, cũng có nhiều nhóm nghiên cứu về nhận dạng tiếng Việt. Có thể liệt kê các đề tài sau:

* Năm 2004, nhóm Nguyễn Tiến Dũng, Vũ Tất Thắng và Lương Chí Mai đã sử dụng HMM trong bài toán nhận dạng danh sách 64 thành phố và tỉnh thành ở Việt Nam.
* Năm 2008, nhóm Trần Việt Khải và Bồ Xuân Tú của Trường Đại học quốc tế thuộc Đại học quốc gia Tp. Hồ Chí Minh đã sử dụng Sphinx4 trong nhận dạng các chữ số tiếng Việt từ 0 đến 9 và ứng dụng kết quả nhận dạng để duyệt web.

Ngoài ra với việc sử dụng bộ thư viện của Microsoft Speech SDK để nhận dạng các từ tiếng Anh, sau đó thực hiện chuyển đổi từ nhận dạng được sang tiếng Việt, nhóm BK02 (sinh viên Trường Đại học Bách khoa TP Hồ Chí Minh) đã cho ra đời phần mềm Vspeech [1]. Các chức năng của phần mềm có thể kể đến là: duyệt web bằng giọng nói, mở một ứng dụng bằng giọng nói,...

# TỔNG QUAN ĐỀ TÀI

## Lý do chọn đề tài

Trong cuộc sống, giao tiếp là một phần thiết yếu. Đặc biệt là giao tiếp thông qua giọng nói là một vấn đề thật sự cần thiết và đáng lưu tâm. Với sự phát triển của Công Nghệ Thông Tin vấn đề số hóa giọng nói rất quan trọng, Mặc khác trên thế giới hiện tại chỉ đang phát triển chủ yếu cho việc nhận dạng Tiếng Anh, rất ít đề tài nghiên cứu và phát triển về nhận dạng going nói tiếng Việt được chia sẽ, bên cạnh đó các trò chơi dành cho trẻ em xuất hiện ngày càng nhiều, tuy nhiên các trò chơi này luôn chú tâm đến vấn đề nhận dạng hình ảnh, sự việc giúp cho trẻ hình dung thê giới xung quanh. Hay những ứng dụng giúp trẻ phát âm Tiếng Anh ngay từ khi các bé còn rất nhỏ. Chính vì thế chúng em chọn đề tài “Voice Recognition” với mong muốn thực hiện ứng dụng nhận dạng Tiếng Việt, cùng với đó là việc xây dựng ứng dụng giúp trẻ vui chơi cùng Tiếng Việt.

## Mục tiêu và phạm vi đề tài

### Mục tiêu

+ Xây dựng, tích hợp thành công hệ thống nhận dạng không cần bất kỳ

+ Xây dựng ứng dụng giúp trẻ tập nói Tiếng Việt,

+ Có thêm tài liệu về vấn đề nhận dạng Tiếng Việt giúp các bạn tìm hiểu sau có them tài liệu sử dụng và phát triển

### Phạm vi đề tài

+ Sử dụng Engine mã nguồn mở Sphinx để thực hiện quá trình nhận dạng

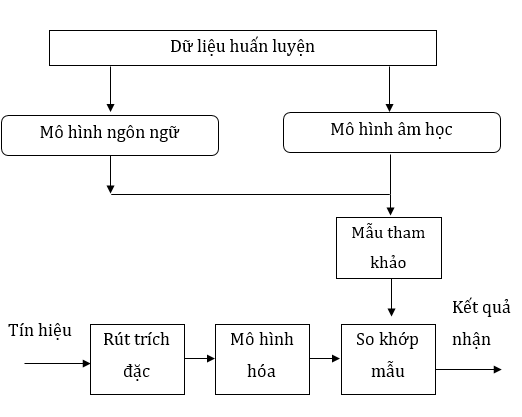
+ Dữ liệu nhận dạng là tất cả các từ đã được chọn trong ứng dụng

+ Thiết bị đầu cuối hướng đến là thiết bị Android.

# CHƯƠNG 1 : ****CƠ SỞ LÝ THUYẾT VỀ NHẬN DẠNG GIỌNG NÓI****

* 1. **HỆ NHẬN DẠNG GIỌNG NÓI**
     1. **Tổng quan**

Nhận dạng giọng nói là quá trình biến đổi các tín hiệu âm thanh thành một chuỗi các từ.



***Hình 1. 0.1Sơ đồ nhận dạng giọng nói***

Tín hiệu giọng nói sẽ được rút trích đặc trưng, kết quả thu được của quá trình này gọi là các vector đặc trưng.

Để thực hiện việc so sánh, hệ thống phải được huấn luyện và xây dựng các đặc trưng. Trong quá trình huấn luyện, hệ thống sử dụng các vector đặc trưng để ước lượng, tính toán các tham số cho các mẫu gọi là các mẫu tham khảo. Một mẫu tham khảo sẽ được dùng để so sánh và nhận dạng.

Trong quá trình nhận dạng, dãy các vector đặc trưng của tín hiệu cần nhận dạng được so sánh với các mẫu tham khảo được xây dựng trong quá trình huấn luyện. Hệ thống sẽ tính toán độ tương đồng của dãy các vector đặc trưng với mẫu tham khảo. Việc tính toán độ tương đồng được thực hiện bằng cách áp dụng các thuật toán như thuật toán Viterbi trong mô hình Markov ẩn. Mẫu có độ tương đồng cao nhất được cho là kết quả của quá trình nhận dạng.

* + 1. **Các hệ nhận dạng giọng nói**
       1. **Nhận dạng từ liên tục và tách biệt**

Một hệ nhận dạng giọng nói có thể là một trong hai dạng: nhận dạng liên tục và nhận dạng từng từ. Nhận dạng liên tục tức là nhận dạng giọng nói được phát liên tục trong một chuỗi tín hiệu chẳng hạn như một câu nói, một mệnh lệnh hoặc một đoạn văn. Các hệ thống dạng này rất phức tạp vì nếu người nói phát âm liên tục sẽ rất khó tách từ. Trong đó, kết quả tách từ ảnh hưởng rất lớn đến các quá trình sau.

Đối với mô hình nhận dạng từng từ, mỗi từ cần nhận dạng được phát âm một cách rời rạc. Các ứng dụng thực tiễn đối với mô hình dạng này có thể kể đến như hệ thống điều khiển bằng giọng nói, quay số bằng giọng nói, nhập văn bản bằng giọng nói, tìm kiếm bằng giọng nói,…

* + - 1. **Nhận dạng phụ thuộc người nói và độc lập với người nói**

Đối với nhận dạng phụ thuộc người nói thì mỗi hệ nhận dạng chỉ phục vụ được cho một người và nó sẽ không hiểu người khác nói gì nếu chưa được huấn luyện. Do đó, đối với những hệ thống dạng này không được áp dụng rộng rãi.

Hệ nhận dạng độc lập với người nói thì khắc phục được nhược điểm của hệ nhận dạng phụ thuộc nhưng hệ nhận dạng này lại gặp vấn đề về độ chính xác. Vì trong thực tế mỗi người có một giọng nói khác nhau, ở những vùng miền và độ tuổi khác nhau cũng ảnh hưởng đến quá trình nhận dạng. Dó đó, để khắc phục vấn đề này đòi hỏi hệ thống phải được huấn luyện với lượng dữ liệu lớn và phong phú.

* + 1. **Một số phương pháp nhận dạng giọng nói**
       1. **Phương pháp âm – ngữ học**

Phương pháp này dựa trên lý thuyết về âm – ngữ học. Trong lời nói tồn tại các đơn vị ngữ âm xác định, có tính phân biệt. Các bước nhận dạng của phương pháp này bao gồm:

*Bước 1: Phân đoạn và gán nhãn*

Chia tín hiệu tiếng nói thành các đoạn có đặc tính âm học đặc trưng cho một đơn vị ngữ âm, đồng thời gán cho mỗi đoạn âm thanh đó một hay nhiều nhãn ngữ âm phù hợp.

*Bước 2: Nhận dạng*

Dựa trên một số điều kiện ràng buộc về từ vựng để xác định một hay một chuỗi từ đúng trong các chuỗi nhãn ngữ âm đã được tạo ra ở bước 1.

* + - 1. **Phương pháp nhận dạng mẫu**

*Bước 1:* Sử dụng tập mẫu tiếng nói hay còn gọi là cơ sở dữ liệu tiếng nói để huấn luyện. Kết quả thu được của bước này là các mẫu tiếng nói đặc trưng.

*Bước 2:* So sánh mẫu tiếng nói cần nhận dạng với các mẫu tiếng nói đặc trưng được tạo ra ở bước 1 để nhận dạng.Mô hình Markov ẩn (HMM), mạng nơ-ron nhân tạo (ANN), so sánh thời gian động (DTW),… áp dụng phương pháp nhận dạng mẫu này.

* + - 1. **Phương pháp ứng dụng trí tuệ nhân tạo**

Phương pháp ứng dụng trí tuệ nhân tạo kết hợp các phương pháp trên nhằm tận dụng tối đa các ưu điểm của chúng, đồng thời bắt chước các khả năng của con người trong phân tích và cảm nhận các sự kiện bên ngoài để áp dụng vào nhận dạng tiếng nói.

Đặc điểm của phương pháp này là sử dụng hệ chuyên gia để phân đoạn, gán nhãn ngữ âm. Sử dụng mạng nơ-ron nhân tạo để học mối quan hệ giữa các ngữ âm.

Việc sử dụng hệ chuyên gia nhằm tận dụng kiến thức con người vào hệ nhận dạng:

- Kiến thức về âm học: Để phân tích phổ và xác định đặc tính âm học của các mẫu tiếng nói.

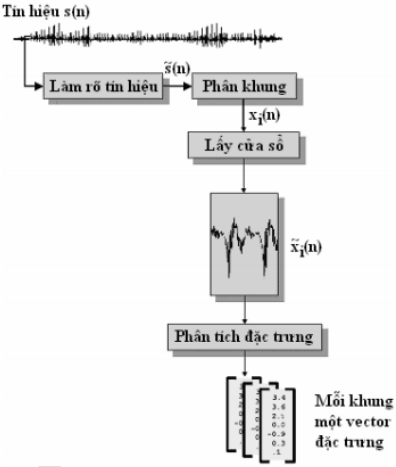
- Kiến thức về từ vựng: sử dụng để kết hợp các khối ngữ âm thành các từ cần nhận dạng.

- Kiến thức về cú pháp: nhằm kết hợp các từ thành các câu cần nhận dạng.

- Kiến thức về ngữ nghĩa: nhằm xác định tính logic của các câu đã được nhận dạng.

* 1. **RÚT TRÍCH ĐẶC TRƯNG TÍN HIỆU GIỌNG NÓI**
     1. **Giới thiệu**

Rút trích đặc trưng của tiếng nói là một trong những công đoạn quan trọng trong quá trình nhận dạng tiếng nói, điều này giúp giảm thiểu số lượng dữ liệu trong việc huấn luyện và nhận dạng. Mặt khác, việc rút trích đặc trưng còn làm rõ sự khác biệt của tiếng nói này với tiếng nói khác hay làm mờ đi sự khác biệt của hai lần phát âm cho cùng một từ. Kết quả của quá trình rút trích đặc trưng là một hoặc nhiều vector đặc trưng. Có nhiều phương pháp để thực hiện rút trích đặc trưng, 2 trong số đó là phương pháp MFCC và LPC.



***Hình 1. 0.2 Quá trình rút trích đặc trưng***

Tín hiệu âm thanh lưu trong máy tính là tín hiệu digital [13], mô hình hoá tín hiệu âm thanh trong máy tính dưới dạng toán học là một hàm s(n) với n chỉ thời gian (đơn vị thông thời là ms) và s(n) là biên độ âm.

* + 1. **Tiền khuếch đại**

Làm nổi rõ đặc trưng của tín hiệu và làm cho nó ít nhạy cảm hơn với các hiệu ứng do độ chính xác hữu hạn ở những bước sau.

Theo các nghiên cứu về âm học thì giọng nói có sự suy giảm 20dB/decade khi lên tần số cao do đặc điểm sinh lý của hệ thống phát âm con người. Để khắc phục sự suy giảm này, chúng ta sẽ phải tăng cường tín hiệu lên một giá trị gần 20dB/decade. Bên cạnh đó, hệ thống thính giác con người có xu hướng nhạy cảm hơn với vùng tần số cao. Dựa vào những đặc điểm trên, ta sẽ áp dụng bộ lọc thông cao để tiền xử lý các tín hiệu thu được nhằm làm rõ vùng tín hiệu mà tai người có thể nghe được. Bộ lọc áp dụng công thức sau:



Trong đó apre là hệ số nhấn mạnh, thường có giá trị là 0.9700002861.

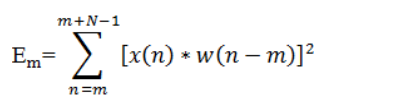
Bộ lọc này có chức năng tăng cường tín hiệu tại tần số cao (tần số trên 1KHz).Tín hiệu tiếng nói đã số hoá, s(n), được đưa qua một hệ số bậc thấp, để làm phẳng tín hiệu về phổ và làm nó ít bị ảnh hưởng bởi các hiệu ứng có độ chính xác hữu hạn sau này trong quá trình xử lý tín hiệu Trong trường hợp sử dụng bộ lọc áp theo công thức trên đầu ra của dãy tiền khuếch đại s’(n), liên quan đến đầu vào của dãy tín hiệu s(n), theo đẳng thức vi phân sau:



* + 1. **Tách từ**

Tín hiệu giọng nói s(n) sau khi được khuếch đại, sẽ được chuyển sang để tách từ. Đây là công đoạn chia toàn bộ tín hiệu thu được thành những đoạn tín hiệu mà trong đó chỉ chứa nội dung của một từ.

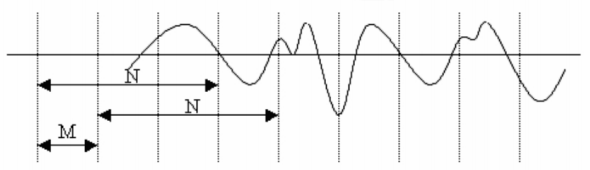
Có nhiều phương pháp để tách điểm đầu và điểm cuối của một ra khỏi toán bộ tín hiệu tiếng nói, trong đó phương pháp dùng hàm năng lượng thời gian ngắn là phương pháp được sử dụng phổ biến. Với một cửa sổ kết thức tại mẫu thứm, hàm năng lượng thời gian ngắn E(m) được xác định :



* + 1. **Phân đoạn**

Tín hiệu tiếng nói là một đại lượng biên thiên theo thời gian và không ổn định nên không thể xử lý trực tiếp trên đó được. Do đó, tín hiệu được chia ra thành các khung với chiều dài tương đối nhỏ để lấy được các đoạn tín hiệu tương đối ổn định và xử lý tiếp trong các bộ lọc tiếp theo. Theo các nghiên cứu đã có thì trong khoang thời gian 10-20ms, tín hiệu tiếng nói tương đối ổn định. Nên ở bước này, người ta thường phân tín hiệu thành các khung với kích thước 20-30ms. Nhưng để tránh mất mát và làm gián đoạn tín hiệu ban đầu, khi phân khung, người ta chồng lấp các khung lên nhau khoảng 10-15ms.

Trong bước này tín hiệu được tiền khuếch đại **,** được chia thành các khung, mỗi khung gồm N mẫu, khoảng cách giữa các khung là M mẫu.



***Hình 1. 0.3 Phân đoạn tiếng nói thành các khung.***

Hình 1.0.3 minh họa cách phân thành các khung trong trường hợp M = 1/3N.

Cụ thể khung thứ nhất gồm N mẫu tiếng nói đầu tiên. Khung thứ hai bắt đầu từ mẫu thứ M và kết thúc ở vị trí M+N-1. Tương tự khung thứ I bắt đầu từ mẫu thứ i\*M và kết thúc ở vị trí i\*M+N-1. Tiến trình này tiếp tục cho đến khi các mẫu tiếng nói đều đã thuộc về một hay nhiều khung. Như vậy, nếu MN thì các khung kề nhau sẽ có sự chồng lấp dẫn đến kết quả là các phép rút trích đặc trưng có tương quan với nhau từ khung này sang khung khác. Ngược lại, nếu M>N thì sẽ không có sự chồng lấp giữa các khung kề nhau.

* + 1. **Lấy cửa sổ khung tín hiệu**

Lấy cửa sổ khung tín hiệu cho mỗi khung để giảm thiểu sự gián đoạn ở đầu và cuối mỗi khung. Một dãy tín hiệu con được lấy ra từ một tín hiệu dài hơn hoặc dài vô hạn x(n) gọi là một cửa sổ tín hiệu. Việc quan sát tín hiệu x(n) bằng một đoạn XN(n) trong khoảng n0...(n0 + N – 1) tương đương với việc nhận x(n) với một hàm cửa sổ w(n-n0).

x(n) n0 nn0 + N - 1

0 (n<n0) v (n>n0 + N – 1)

(2.3)

XN(n) = x(n).w(n-n0) =

Trong xử lý tín hiệu số, các cửa sổ thường được biểu diễn thông qua cửa sổ Hamming tổng quát:



(2.4)

W(n) =

0

Tùy theo các giá trị khác nhau của  mà ta có các cửa sổ khác nhau:

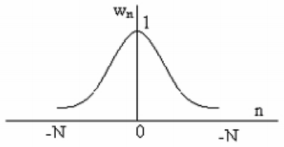
Với  = 0.54, ta có cửa sổ Hamming:



(2.5)

W(n) =

0



***Hình 1.2.1. Cửa sổ Hamming vớilà 0.54.***

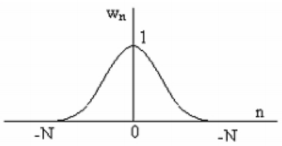
Với  = 0.5, ta có cửa sổ Hanning:



(2.6)

W(n) =

0



***Hình 1.2.2. Cửa sổ Hanninglà 0.5***

Với  = 1, ta có cửa sổ chữ nhật:

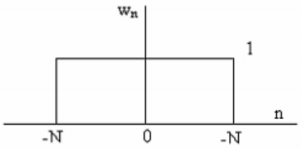


(2.7)

W(n) =

1

0



***Hình 1.2.3. Cửa sổ chữ nhật***

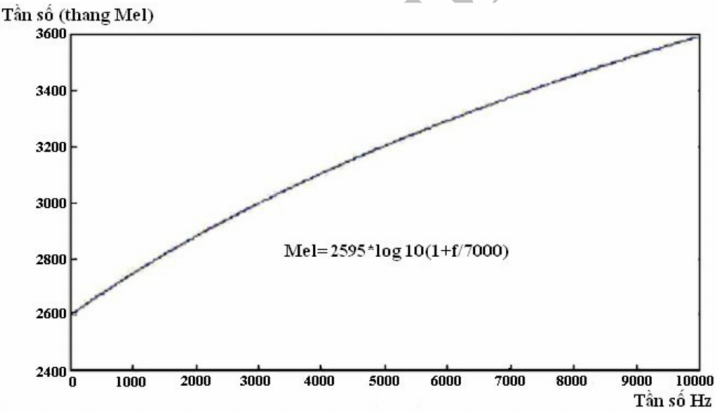
Ý nghĩa của việc áp cửa sổ : là nhằm mục đích có được dữ liệu theo miền tần số chuẩn để đưa vào phép biến đổi Fourier rời rạc

* + 1. **Rút trích đặc trưng**
       1. **Rút trích đặc trưng với MFCC**

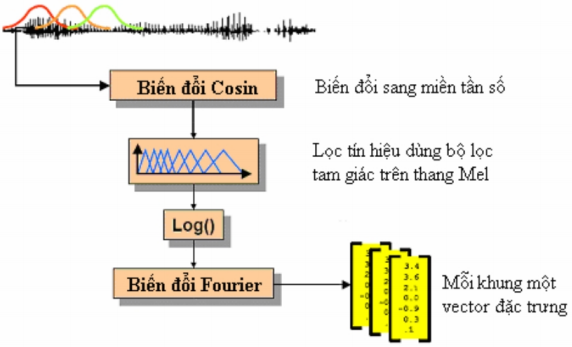
Các nghiên cứu cho thấy rằng hệ thống thính giác của con người thu nhận âm thanh với độ lớn các tần số âm thanh không theo thang tuyến tính. Do đó, các thang âm thanh đã ra đời cho phù hợp với sự tiếp nhận của thính giác con người.

Mel Scale Frequency Cepstral Coefficients (MFCC) là một phương pháp rút trích đặc trưng sử dụng dãy bộ lọc được Davis và Mermelstein đưa ra vào năm 1980 khi họ kết hợp các bộ lọc cách khoảng không đều với phép biến đổi Cosin rời rạc (Discrete Cosin Transform) thành một thuật toán hoàn chỉnh được ứng dụng trong lĩnh vực nhận dạng giọng nói liên tục. Đồng thời cũng định nghĩa khái niệm hệ số Cepstral và thang đo tần số Mel (Mel scale).

Các thang được xây dựng bằng thực nghiệm, cho nên người ta xây dựng các công thức để xấp xỉ sự chuyển đổi này. Trong các thang và công thức dạng đó thì đặc trưng MFCC sử dụng thang Mel. Thang Mel được thể hiện thông qua đồ thị sau:

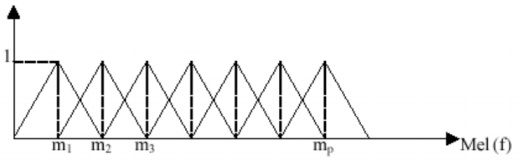


Hình *.* Đồ thì biểu diễn mối quan hệ giữa Mel và Hz.

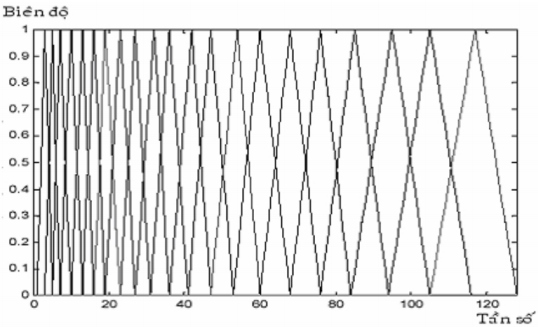


Hình Các bước rút trích đặc trưng MFCC.

Ta dùng phép biến đổi Fourier để chuyển đổi tín hiệu từ miền thời gian sang miền tần số sau đó ta dùng dãy bộ lọc tín hiệu đó là dãy bộ lọc tam giác có tần số giưa đều nhau trên thang Mel.

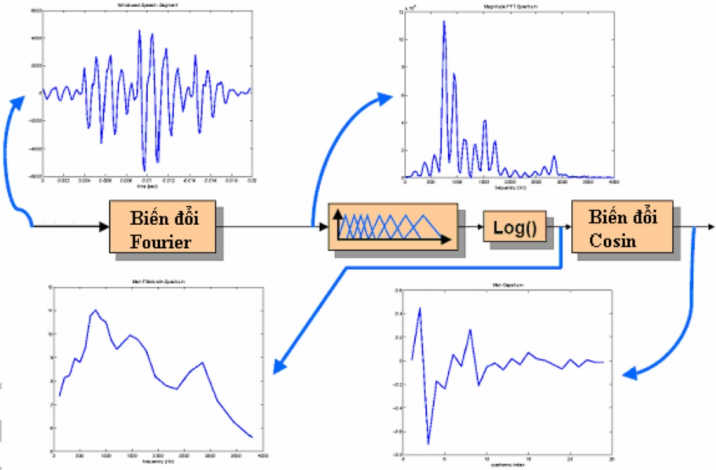


Hình *.* Bộ lọc trên thang Mel.



Hình Bộ lọc trên tần số thật*.*

Lấy log trên dãy kết quả từ dãy bộ lọc và thực hiện biến đổi cosin rời rạc ta thu được các hệ số đặc trưng MFCC.



Hình . Minh họa các bước biến đổi MFCC.

Như vậy, quá trình rút trích đặc trưng theo phương pháp MFCC sẽ được tiến hành như sau:

Tín hiệu sau khi qua tiền xử lý sẽ được chia thành các khung (frame) có khoảng thời gian ngắn. Dựa vào mỗi khung đó, sau khi áp dụng các bước biến đổi và lọc sẽ được một vecto tương ứng.

1. Biến đổi FFT (Fast Fourier Transform)

Biến đổi FFT thực chất là một biến đổi DFT (Discrete Fourier Transform ) nhưng được tối ưu bằng các thuật toán nhanh và gọn hơn để đáp ứng các yêu cầu xử lý theo thời gian thực trong các lĩnh vực như xử lý âm thanh, hình ảnh,...

Fast Fourier là một phép biến đổi thuận nghịch có đặc điểm bảo toàn tính tuyến tính bất biến, tuần hoàn và tính trễ. Dùng để biến đổi tín hiệu tương tự sang miền tần số, nó gồm các công thức như sau:

Công thức phép biến đổi thuận (dùng để phân tích tín hiệu):

X(k)=-j2/N k = 0,1,2,…,N-1

Công thức phép biến đổi nghịch (dùng để tổng hợp lại tín hiệu):

x(n) = 

n = 0,1,2,…N-1

Trong đó: x(n) = a(n) + b(n)

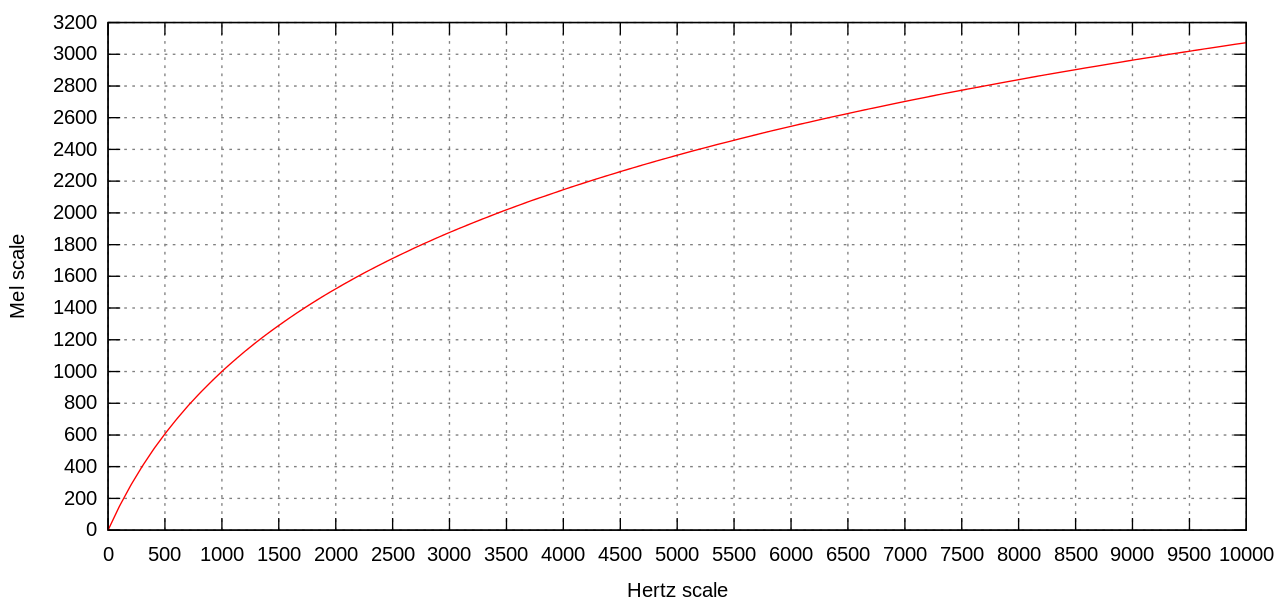
Kết quả chúng ta có được khi thực hiện FFT là dãy tín hiệu Xt(k) để đưa vào bộ lọc Mel-scale.

1. Lọc qua bộ lọc Mel-scale

Trong lĩnh vực nghiên cứu về nhận dạng tiếng nói, đòi hỏi chúng ta phải hiểu và mô phỏng chính xác khả năng cảm thụ tần số âm thanh của tai người. Chính vì thế các nhà nghiên cứu đã xây dựng một thang tần số - hay gọi là thang tần số Mel (Mel scale) dựa trên cơ sở thực nghiệm nhiều lần khả năng cảm nhận âm thanh của con người. Thang tần số Mel được định nghĩa trên tần số thực theo công thức:

m = 2595log10

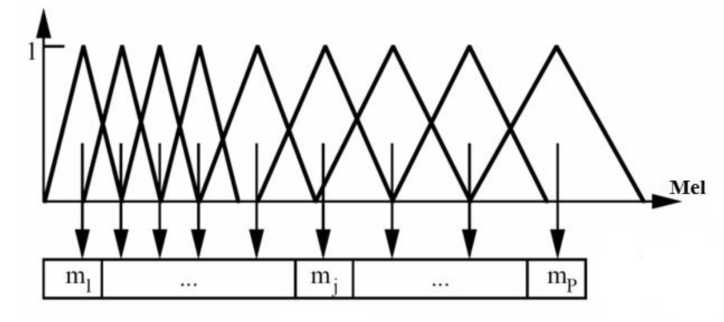
Trong đó: m là tần số trong thang Mel, đơn vị là Mel; f là tần số thực, đơn vị là Hz.



Hình Biểu đồ thang tần số Mel theo tần số thực.

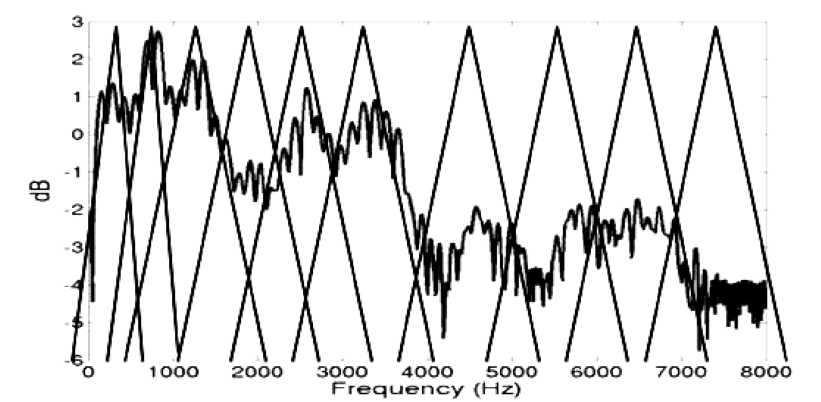
Theo biểu độ trên thì trong khoảng tần số thấp hơn 1kHz thì đồ thị trên gần như là tuyến tính, nghĩa là trong khoảng tần số dưới 1kHz, tần số Mel và tần số thực. Trong khoảng tần số trên 1kHz thì mối quan hệ này là quan hệ Logarit.

Dựa vào các thực nghiệm trên tai người, người ta đã xác định được các tần số thực mà tai người có thể nghe duoc va chứa đựng nhiều thông tin. Sau đó chuyển các tần số này sang tần số Mel va xây dựng một thang đo như sau:



Hình Băng lọc tần số Mel.

Ta dùng thang đo này để áp vào dãy sóng tín hiệu thu được sau khi thực hiện FFT.



Hình Đưa tín hiệu vào băng lọc tần số Mel.

Kết quả của bước này là chúng ta sẽ có được tập hợp các tần số Yt(m) là giao điểm của sóng tần số vời thang tần số Mel từ dãy tín hiệu Xt(k).

1. Logarit giá trị năng lượng (Logarit of filter energies)

Mục đích của bước này là nén các giá trị đã lọc được vào miền giá trị nhỏ hơn để xử lý nhanh hơn. Nên các giá trị thu được ở mỗi kênh lọc sẽ được lấy Logarit với công thức log{|Yt(m)|2}.

1. Biến đổi cosin rời rạc

Dựa vào phổ tín hiệu tiếng nói của con người trên miền tần số, ta có thể thấy rằng phổ tín hiệu khá trơn, nên khi lấy các giá trị năng lượng ra từ các bộ lọc, các giá trị này có sự tương quan khá gần nhau, dẫn đến các đặc trưng ta rút được sẽ không rõ ràng. Chính vì thế, ta thực hiên biến đổi DCT (Discrete Cosin Transform) để làm rời rạc các giá trị này ra cho nó ít tương quan với nhau, làm tăng tính đặc trưng của các tham số. Giá trị thu được sau bước này ta gọi là hệ số Cepstral.

Ci = 

N là số kênh lọc

Trong đó:

Mj là giá trị logarit năng lượng của mạch lọc thứ j.

i là bậc của hệ số cepstral

Thông thường người ta lấy i trong đoạn [1,12] là số lượng đặc trưng trong mỗi vecto đặc trưng. Trong các hệ nhận dạng, số lượng đặc trưng nằm trong khoảng (10,15) là đủ để cho kết quả nhận dạng tương đối mà dữ liệu xử lý lại không quá lớn.

Sau khi thực hiện biến đổi DCT, theo công thức trên ta thấy các hệ số thu được sẽ tăng tuyến tính theo số bậc của nó. Hệ số Cepstral có số bậc cao sẽ có giá trị rất cao, ngược lại các hệ số với số bậc thấp sẽ có giá trị rất thấp. Sự chênh lệch này sẽ gây khó khăn cho chúng ta trong qua trình mô hình hóa dữ liệu và xử lý sau này. Vì khi có sự chênh lệch cao, ta phải dùng miền giá trị lớn để biểu diễn dữ liệu, và gặp khó khăn khi đưa vào các mô hình xử lý xác suất,..Nên để có các hệ số tối ưu cho các qua trình sau, ta sẽ thực hiện việc điều chỉnh các hệ số này để giảm sự chênh lệch. Việc này thực hiện bằng công thức:

c'n = exp(n \* fc) . cn

Cuối cùng chúng ta sẽ thu được các giá hệ số Cepstral đã được tinh chế. Các hệ số này là đặc trưng MFCC mà chúng ta sẽ sử dụng để huấn luyện và nhận dạng.

* + - 1. **Rút trích đặc trưng với LPC**

1. *Giới thiệu*

LPC là chữ viết tắt của cụm từ: Linear Predictive Coding (mã hóa dự báo tuyến tính). Đây được xem là một trong những phương pháp được sử dụng rộng rãi trong việc rút trích đặt trưng của tín hiệu âm thanh (hay còn được gọi là tham số hóa tính hiệu âm thanh). Đóng vai trò quan trọng trong các kỹ thuật phân tích tiếng nói. Đây còn được xem là một phương pháp hiệu quả cho việc nén (mã hóa với chất lượng tốt) dữ liệu tiếng nói ở mức bit rate thấp.

1. *Cơ sở âm học của phương pháp LPC*

Tiếng nói hay còn được xem là âm thanh do con người phát ra từ miệng bắt nguồn từ từ sự rung động của dây thanh âm (do sự thay đổi áp suất không khí từ phổi đưa lên), sự rung động này mang 2 đặc tích là cường độ (intensity) và tần số (frequency). Âm thanh này sau đó được truyền qua cuống họng đến khoang miệng và khoan mũi. Tại đây dựa vào cấu tạo vòng miệng khi nói, cách đặt lưỡi, chuyễn động của lưỡi và cơ miệng... sẽ góp phần gây ra sự cộng hưởng của âm thanh (hay còn được gọi là các Formant), kết quả chính là tiếng nói mà ta nghe được. LPC phân tích những tín hiệu tiếng nói bằng cách ước tính các formant, loại bỏ đi những thành phần ảnh hưởng của nó (những thứ không mang giá trị tiếng nói trong âm phát ra), và ước lượng các đặc điểm về cường độ, tần số của phần âm thanh còn lại. Qúa trình loại bỏ ở trên còn được gọi là quá trình lọc nghịch đảo (inverse filtering) và phần âm thanh còn lại gọi là“cặn” (residue) mang những yếu tố và đặc trưng cốt lõi của âm thanh. Kết quả còn lại sau quá trình LPC là những con số, mà mô tả những đặc điểm quan trọng của các formant cũng như phần âm thanh còn lại. các con số này có thể được dùng đại diện như tín hiệu tiếng nói ban đầu, hiệu quả hơn trong việc lưu trữ, phân tích nội dung, truyền tải tiếng nói... LPC còn được dùng trong quá trình tổng hợp lại tiếng nói từ các con số đặc trưng trên.

1. Nội dung phương pháp rút trích đặc trưng LPC

Ý tưởng của LPC là một mẫu tiếng nói cho trước ở thời điểm n, s(n), có thể được xấp xỉ bằng một tổ hợp tuyến tính của p mẫu tiếng nói trước đó:

s(n)  a­­1(n-i) + a2s(n-2) + ... + aps(n-p)

Trong đó a1, a2, ..., an coi là các hằng trên toàn khuông phân tích tiếng nói. Ta chuyển đẳng thức (1) trên tương đương bằng cách thêm giới hạn kích thích, Gu(n), có:

s(n) = 

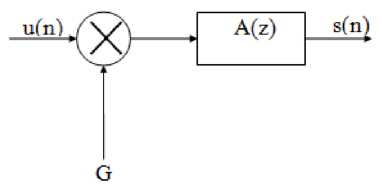
Trong đó u(n) là kích thích đã chuẩn hoá, G là độ khuếch đại của kích thích. Biểu diễn đẳng thức (2) trong miền z ta có quan hệ:

s(n) = 

dẫn đến hàm chuyển

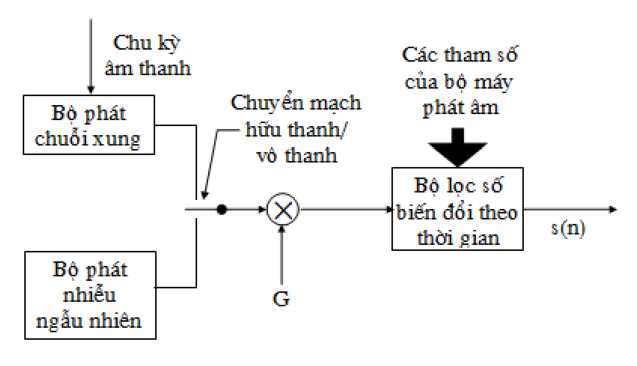
H(z) = 

Hình sau diễn tả đẳng thức bên trên, mô tả nguồn kích thích đã chuẩn hóa u(n), sau đó được nâng mức với độ khuếch đại G, và đóng vai trò đầu vào của hệ toàn cực, H(z) = , tạo ra tín hiệu tiếng nói s(n).



Hình Mẫu tiếng nói dự báo tuyến tính.

Chức năng kích thích thực tế đối với tiếng nói về bản chất là chuỗi liên tục các xung gần như điều hoà (với các âm hữu thanh) và là một nguồn nhiễu ngẫu nhiên (với các âm vô thanh), mẫu hoà âm thích hợp đối với tiếng nói, ứng với phân tích LPC được mô tả trong phía dưới. Tại đây, nguồn kích thích đã chuẩn hoá được chọn nhờ một chuyển mạch mà vị trí của nó được điều khiển bởi đặc tính hữu thanh/vô thanh của tiếng nói, chọn cả chuỗi xung gần điều hoà làm kích thích đối với các âm hữu thanh, chọn cả chuỗi nhiễu ngẫu nhiên cho các âm vô thanh. Độ khuếch đại G của nguồn được ước tính từ tín hiệu tiếng nói, và nguồn được nâng mức được dùng làm đầu vào của bộ lọc số (H(z)), điều khiển bởi đặc tính của các tham số dải phát âm của tiếng nói đang được tạo. Như vậy các tham số của mẫu này là phân loại hữu thanh/vô thanh, chu kỳ cao độ của âm hữu thanh, tham số khuếch đại, và các hệ số của bộ lọc số {ak}, các tham số này đều biến đổi rất chậm theo thời gian.



Hình Mẫu phân tích tiếng nói theo phương pháp LPC.

Dựa vào mẫu trên hình, quan hệ chính xác giữa s(n) và u(n) là



Ta coi tổ hợp tuyến tính của các mẫu tiếng nói trước đó là ước lượng (n), được định nghĩa là:



Bây giờ ta thiết lập lỗi dự báo, e(n), được định nghĩa là:



Với hàm chuyển đổi lỗi:



(2.21)

Rõ ràng là khi s(n) được tạo thực sự bởi một hệ tuyến tính kiểu thì lỗi dự báo e(n) sẽ bằng Gu(n), kích thích được khuếch đại.

Vấn đề cơ bản của phân tích dự báo tuyến tính là xác định tập các hệ số dự báo, {ak}, trực tiếp từ tín hiệu tiếng nói. Vì các đặc tính phổ của tiếng nói biến đổi theo thời gian nên các hệ số dự báo tại thời điểm cho trước n, phải được ước lượng từ một đoạn tín hiệu tiếng nói ngắn xuất hiện quanh thời điểm n. Như vậy cách cơ bản là tìm một tập các hệ số dự báo giảm thiểu lỗi dự báo trung bình bậc hai trong một đoạn dạng sóng tiếng nói. (Thường thì kiểu phân tích phổ thời gian ngắn nàyđược thực hiện trên các khuông tiếng nói liên tiếp, có dãn cách khuông khoảng 10ms).

Nội dung tiếp theo trình bày các bước chính trong công đoạn phân tích đặc trưng theo phương pháp LPC.

1. Phân tích tự tương quan

Sau khi tiến hành công đoạn nhận với hàm cửa sổ. Mỗi khung của tín hiệu được chia cửa sổ là tự tương quan với khung tiếp theo để cho:

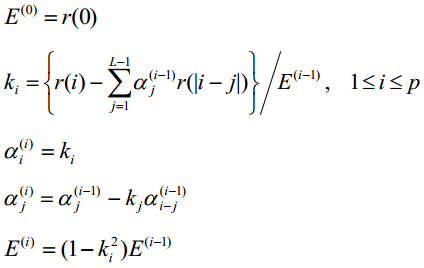


re(m) =

Trong đó giá trị tự tương quan cao nhất, p, là bậc của phép phân tích LPC. Các giá trị p thường dùng là từ 8 đến 16, với p=8 được sử dụng nhiều trong các hệ phân tích LPC.

1. Phân tích LPC

Bước xử lý tiếp theo là phân tích LPC, chuyển từng khuông của p+1 mối tự tương quan thành một "tập tham số LPC", đó là các hệ số LPC. Phương pháp chính thức để chuyển từ các hệ số tự tương quan sang tập tham số LPC (để dùng cho phương pháp tự tương quan LPC) được gọi là phương pháp Durbin và có thể cho một cách hình thức như thuật toán (để thuận tiện, ta sẽ bỏ qua  nhỏ ở dưới ):



Trong đó tổng trong đẳng thức được bỏ qua đối với i=1. Tập các đẳng thức trên được giải đệ qui với i=1,2,...,p, và lời giải cuối cùng được cho là

am = các hệ số LPC =a ,1 ≤ m ≤ p

km = các hệ số PARCOR

gm = các hệ số truyền miền logarit = log()

Lúc này ta có thể dùng các hệ số LPC làm vector đặc trưng cho từng khung. Tuy nhiên có một phép biến đổi tạo ra dạng hệ số khác có độ tập trung cao hơn từ các hệ số LPC, đó là phép phân tích Ceptral.

1. Phân tích cepstral

Một tập tham số LPC rất quan trọng, có thể suy trực tiếp từ tập hệ số LPC, là các hệ số Cepstral LPC, c(m). Dạng đệ qui là:

Cm = am + 

Cm = 

Trong đó ơ2 là độ khuếch đại trong mẫu LPC. Các hệ số cepstral, là các hệ số của biểu diễn chuyển đổi Fourier của logarit phổ cường độ, được mô tả là một tập đặc tính mạnh, đáng tin cậy hơn các hệ số LPC. trong đó Q.

1. Đặt trọng số cho các hệ số cepstral

Do tính nhạy cảm của các hệ số cepstral bậc thấp đối với sườn phổ tổng thể và do tính nhạy cảm cua hệ số cepstral bậc cao đối với nhiễu (và các dạng biến đổi giống nhiễu khác), nó trở thành kỹ thuật chuẩn để định trọng các hệ số cepstral nhờ một cửa sổ được làm hẹp sao cho giảm thiểu những nhạy cảm này. Chuyển sang ceptral có trọng số:

c’m = wm.cm với 1 ≤ m ≤ Q

(2.31)

(2.30)

Hàm trọng số thích hợp là bộ lộc thông dài (trong miền cepstral):

Wm = 1 + với 1 ≤ m ≤ Q

1. Tính đạo hàm cepstral



Với  là hằng số chuẩn và (2K+1) là số lượng Frame cần tính. K=3 là giá trị thích hợp để tính đạo hàm cấp.Vector đặc trưng của tín hiệu gồm Q hệ số cepstral va Q hệ số đạo hàm cepstral.

1. Nhận xét phương pháp LPC

Tóm lại, trong mô hình phân tích LPC trên, chúng ta cần phải đặ tả các tham số bao gồm:

* N: số mẫu trong mỗi Frame phân tích
* M: số mẫu cách nhau giữa 2 Frame kề nhau
* P: cấp phân tích LPC
* Q: số chiều của các vector cepstral dẫn xuất từ các hệ số LPC
* K: số frame được dùng tính các đạo hàm cepstral.

Tuy mỗi tham số đều có thể thay đổi trên dải rộng, bảng sau cho các giá trị đặc trưng đối với các hệ phân tích ở 3 tần số lấy mẫu khác nhau (6.67kHz, 8kHz và 10kHz).

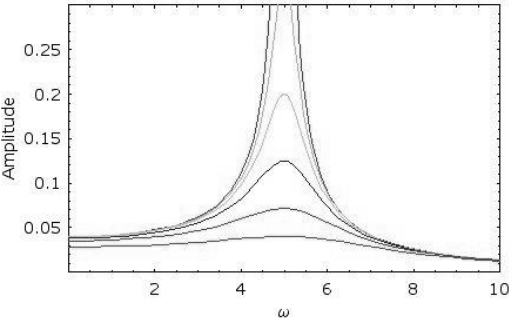
Bảng Các giá trị đặc trưng cho các tham số phân tích LPC.

|  |  |  |  |
| --- | --- | --- | --- |
| Tham số | F1=6.67kHz | F1=8kHz | F1=10kHz |
| N | 300 (45ms) | 240 (30ms) | 300 (30ms) |
| M | 100 (15ms) | 80 (10ms) | 100 (10ms) |
| P | 8 | 10 | 10 |
| Q | 12 | 12 | 12 |
| K | 3 | 3 | 3 |

Mô hình LPC là mô hình thích hợp cho việc xử lý tín hiệu tiếng nói. Với miền tiếng nói hữu thanh có trạng thái gần ổn định, mô hình tất cả các điểm cực đại của LPC cho ta một xấp xỉ tốt đối với đường bao phổ cơ quan phát âm. Với miền tiếng nói vô thanh, mô hình LPC tỏ ra ít hữu hiệu hơn so với miền hữu thanh, nhưng nó vẫn là mô hình hữu ích cho các mục đích nhận dạng tiếng nói. Mô hình LPC đơn giản và dễ cài đặt trên phần cứng và phần mềm.

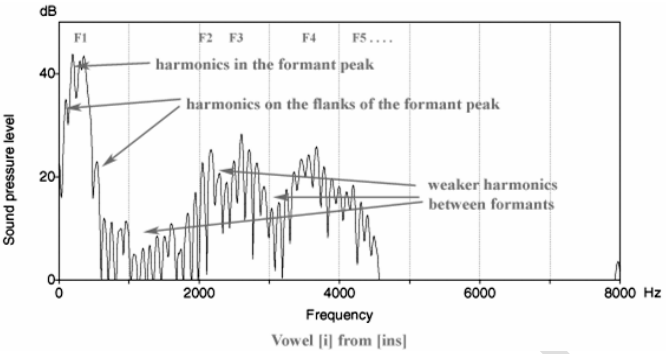
* + 1. **Tìm hiểu về Formant**

Để hiểu rõ hơn về khái niệm Formant, trước tiên chúng ta cần hiểu khái niệm cộng hưởng là gì? Đó là một hiện tượng xảy ra trong dao động cưỡng bức khi một vật dao động được kích thích bởi một ngoại lực tuần hoàn có cùng tần số với giao động riêng của nó.



Hình. Minh hoạ hiện tượng cộng hưởng.

Formant là dải tần số được tăng cường do hiện tượng cộng hưởng trong ống dẫn thanh, đặc trưng cho âm sắc của mỗi nguyên âm. Trong mỗi dải tần như thế có một tần số được tăng cường hơn cả gọi là đỉnh formant (formant peak).



Hình 2*.* Minh hoạ Formant.

Formant là trong các thành phần quan trọng hình thành nên âm học, tạo ra sự đặc trưng cho từng âm thanh. Một nguyên âm do một người phát ra có nhiều formant:

F1: ứng với cộng hưởng vùng yết hầu.

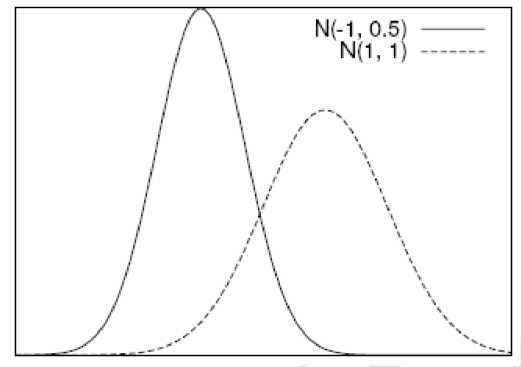
F2: ứng với cộng hưởng khoang miệng.

Khi ta nói, sẽ có các sự cộng hưởng khác ở khoang mũi tạo nên các formant khác (F3, F4, F5..) chính các formant này sẽ quy định các đặc trưng giọng nói riêng cho từng người vì khoang mũi của mỗi người là cố định, có cấu tạo tùy thuộc vào thể trạng của từng người. Ngoài ra còn có 2 nơi cộng hưởng khác là khoang miệng và yết hầu. Nhưng hộp cộng hưởng ở 2 nơi này có khả năng thay đổi linh động phụ thuộc vị trí của các cơ quan khác như môi, lưỡi, hàm. Khi 3 cơ quan này thay đổi vị trí sẽ tạo ra các hộp cộng hưởng khác nhau về thể tích, hình dáng, đường lưu thông của khí trong thanh quản,..Nên sẽ làm biến đổi các âm sắc, chính vì điều đó mà chúng ta thấy ở các lần nói khác nhau của một từ thì chưa chắc chúng được phát âm ra giống nhau. Nên trong lĩnh vực âm học, người ta tập trung nghiên cứu formant tạo ra ở 2 vùng này là F1, F2.

Formant được xác định dựa trên sự tập trung năng lượng âm học xung quanh một tần số đặc biệc trong dải sóng. Trong 1 dải sóng có thể xác định vài formant, mỗi formant có thể có cùng cường độ âm nhưng có tần số khác nhau.

* 1. **GAUSSIAN MIXTURE MODEL**

Mô hình Gauss (Gaussian Mixture Model - GMM) là một dạng mô hình thống kê được xây dựng từ việc huấn luyện các tham số thông qua dữ liệu học. Mô hình GMM còn có một số tên gọi khác như Weighted Normal Distribution Sums hay Radial Basis Function Approximations...

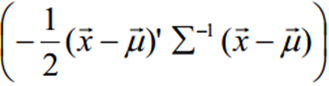


Hình Hàm mật độ Gauss.

Về cơ bản, mô hình GMM xấp xỉ một hàm mật độ xác suất bằng hợp các hàm mật độ Gauss. Hình phía dưới minh họa hai hàm mật độ Gauss với các tham số khác nhau. Một cách hình thức, hàm mật độ xác suất của phân phối Gauss fN(x,2) được cho bởi công thức:

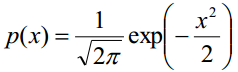
p(x) = 

(2.37)

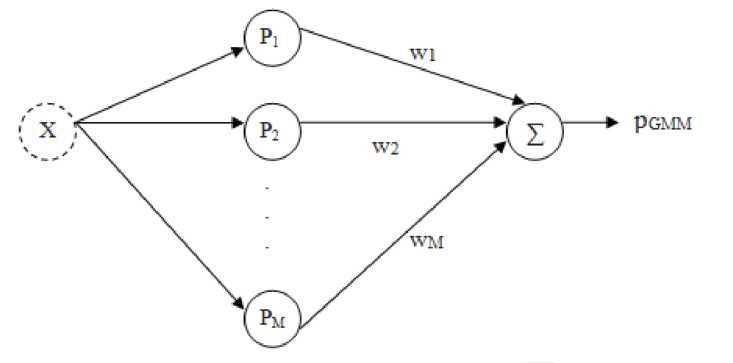
Trong đó, là giá trị trung bình,là độ lệch chuẩn. Trong trường hợp x là vector gồm D thành phần, hàm mật độ xác suất của phân phối Gauss fN (x, ,  được cho bởi công thức:

p() = exp

Khi đó,  là vector trung bình, là ma trận hiệp phương sai. Nếu chọn =0 và  =1, công thức (2.35) sẽ trở thành hàm mật độ chuẩn Gauss tiêu chuẩn:



Từ “Gauss” được đặt theo tên của nhà toán học người Đức Carl Friedrich Gauss. Ông đã định nghĩa hàm mật độ Gauss và áp dụng trong phân tích dữ liệu thiên văn.

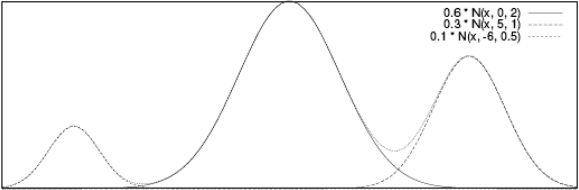


Hình Mô hình GMM.

Cho trước M phân phối Gauss có các hàm mật độ p1, p2, ..., pM, hàm mật độ xác suất của mô hình GMM được minh họa trong hình 2.20 chính là tổng trọng của M phân phối Gauss theo công thức:

pGMM() = ()

trong đó, wi là trọng số của phân phối Gauss thứ i, thỏa ràng buộc 0< wi <1 và . Các trọng số này thể hiện mức độ ảnh hưởng của mỗi phân phối Gauss đối với mô hình GMM. Như vậy, phân phối Gauss có phương sai và trọng số lớn bao nhiêu thì có mức độ ảnh hưởng lớn bấy nhiêu đối với kết xuất của mô hình. Hình 2.21 cho thấy mức độ ảnh hưởng của từng phân phối Gauss lên GMM.



Hình Hàm mật độ của GMM có 3 phân phối Gauss.

Như vậy, một mô hình GMM có M phân phối Gauss sẽ được đại diện bởi bộ tham số  = { wi, i, }, i . Tùy thuộc vào cách tổ chức của ma trận hiệp phương sai (covariance matrix), GMM có thể có một số biến thể khác nhau:

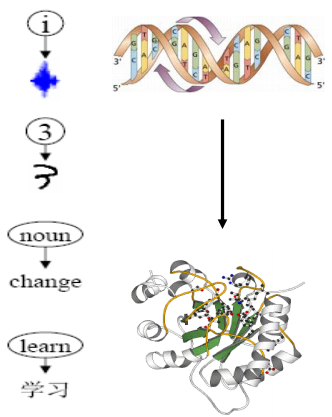
* Nodal covariance matrices GMM: mỗi phân phối Gauss trong GMM có một ma trận hiệp phương sai riêng.
* Grand covariance matrix GMM: mọi phân phối Gauss trong một GMM dùng chung một ma trận hiệp phương sai.
* Global covariance matrix GMM: mọi phân phối Gauss trong tất cả các GMM dùng chung một ma trận hiệp phương sai.

Ngoài ra, xét về dạng thức, ma trận hiệp phương sai gồm hai loại: full (dạng đầy đủ) và diagonal (dạng ma trận đường chéo). Thông thường, dạng nodal-diagonal covariance matrices GMM được sử dụng phổ biến nhất

* 1. **MÔ HÌNH MARKOV ẨN**
     1. **Giới thiệu**
        1. **Định nghĩa**

Mô hình Markov ẩn (tiếng Anh *Hidden Markov Model* – HMM) 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. Các tham số của mô hình được rút ra sau đó có thể sử dụng để thực hiện các phân tích kế tiếp.

Các ứng dụng của mô hình Markov ẩn như nhận dạng giọng nói, nhận dạng chữ viết tay, xử lý ngôn ngữ thống kê, dịch máy, tin sinh học (khớp xấp xỉ nhiều chuỗi, tìm Motif, tìm kiếm tương tự).



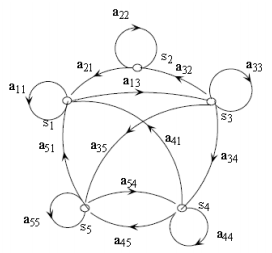
***Hình Các ứng dụng của mô hình Markov ẩn.***

* + - 1. **Thuộc tính Markov**

Một dãy trạng thái ngẫu nhiên gọi là có thuộc tính Markov nếu như xác suất chuyển sang trạng thái tiếp theo chỉ phụ thuộc vào trạng thái hiện tại và quá khứ.

Dãy chuyển trạng quan sát được gọi là Xích Markov, dãy chuyển trạng không quan sát được gọi là mô hình Markov ẩn.

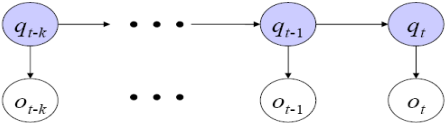
Chuỗi (Xích) Markov



***Hình Chuỗi Markov với 5 trạng thái (S1 đến S5).***

Trong đó N là số trạng thái trong chuỗi Markov (ở đây N = 5), mỗi trạng thái được đánh số từ (1…N). St là trạng thái của hệ thống ở thời điểm t. Aij là xác suất chuyển trạng thái với các tính chất sau: tổng tất cả a = 1, aij >=0. Aij = P[qt = j | qt-1=i], với P[qt = j | qt-1 = i, qt-2 = k,…] = P[qt = j | qt-1=i]

* + - 1. **Mô hình Markov ẩn**

qt là trạng thái ở thời điểm t.

S = {1,2,…,N} là tập các trạng thái ẩn.

Ot= (ký hiệu) quan sát tại thời điểm t.

V = {1,2,…M} là tập tất cả các ký hiệu quan sát được.

A = [aij] là xác suất chuyển trạng thái.

(1.39)



B = [bij] là xác suất nhả ký hiệu.

(1.41)

(1.40)



 = [i] là xác suất khởi trạng



Tiến trình thực hiện của HMM

t = 1;

Bắt đầu với si với xác suất i  (X = i).

Lặp lại quá trình sau:

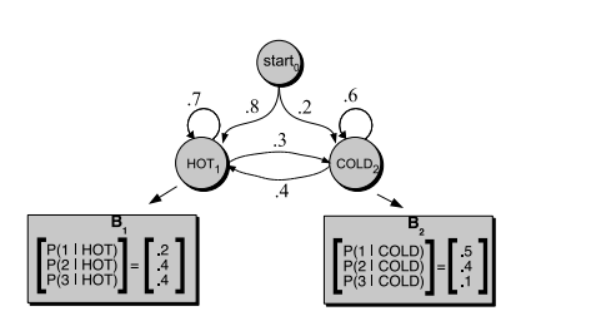
Chuyển từ trạng thái si sang trạng thái sj với xác suất aij (Xt+1=j)

Xuất ra ký hiệu quan sát ot = k với xác suất bijk.

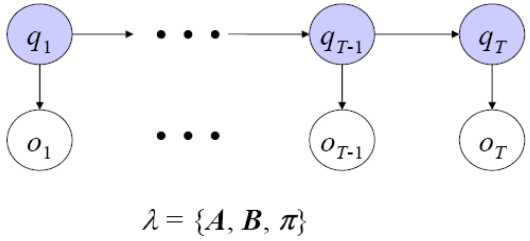
t = t + 1;

Kết thúc.

+ Ví dụ minh họa HMM đơn giản về mối liên hệ giữa số lượng que kem với thời tiết. Số lượng que kem được ăn mỗi ngày là các quan sát, ví dụ: O = {1,2,3}. Thời tiết nóng hay lạnh tương ứng với hai trạng thái. Giả sử chúng ta biết được số lượng que kem được ăn, không biết thời tiết, liệu chúng ta có dự đoán được thời tiết hôm ấy thế nào không. Thời tiết (trạng thái) được xem là “ẩn” so với số que kem được ăn (kết quả quan sát được). Chính vì vậy, mô hình này được gọi là mô hình Markov ẩn (hidden) – Hidden Markov Model (HMM).



* + 1. **Ba bài toán cơ bản của mô hình Markov ẩn**



Bài toán 1 (Bài toán ước lượng): Cho trước chuỗi tín hiệu quan sát O = O1 O2 … OT và mô hình HMM được đại diện bởi bộ tham số λ = (π, A, B). Làm sao để tính toán một cách hiệu quả P(O|λ) – xác suất phát sinh O từ mô hình λ?

Bài toán 2 (Bài toán giải mã – nhận dạng): Cho trước chuỗi tín hiệu quan sát O = O1 O2 … OT và mô hình HMM đại diện bởi bộ tham số λ = (π, A, B). Cần tìm ra chuỗi trạng thái tối ưu nhất Q = q1 q2 … qT đã phát sinh ra O?

Bài toán 3 (Bài toán học): Cho trước chuỗi tín hiệu quan sát O = O1 O2 … OT. Làm thế nào để xác định các tham số mô hình λ = (π, A, B) sao cho cực đại hóa xác suất P(O|λ)? Đây chính là bài toán học hay còn gọi là huấn luyện mô hình. Bài toán này đem lại một khả năng rất quan trọng của HMM: khả năng mô hình hóa một đối tượng cụ thể trong thực tế, mô hình hóa dữ liệu học.

**1.4.3 Bài toán 1 – Computing Likelihood**

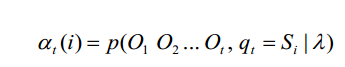
Mục tiêu của bài toán thứ nhất là tính p(O|λ) – xác suất phát sinh O từ mô

hình λ. Một giải pháp khả thi để tính p(O|λ) là thông qua thuật toán forwardbackward. Trước tiên, ta định nghĩa biến forward α t(j) là xác suất ở trạng thái

j tại thời điểm t và đã quan sát được đoạn O1, O

2, ..., Ot với mô hình λ cho

trước:



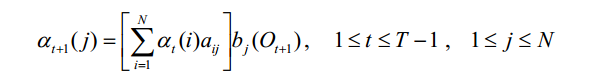
Các biến α t(i) có thể được tính theo qui nạp từng bước (hay thuật toán qui

hoạch động) như sau :

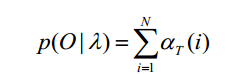
1. Khởi tạo:



1. Qui nạp:



1. Đầu ra:



Độ phức tạp tính toán của thuật toán forward là O(N2T).

Tương tự như trong thủ tục forward, thủ tục backward trước hết định nghĩa

biến backward βt(i) là xác suất quan sát được đoạn Ot+1, O

t+2, ..., O

T cho trước

trạng thái i ở thời điểm t và mô hình λ:



Các biến βt(i) cũng được tính theo qui nạp từng bước như sau:

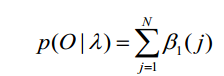
1. Khởi tạo:



2) Quy nạp :



3) Đầu ra:



Cũng giống như thuật toán forward, độ phức tạp của thuật toán backward

là O(N2T). Như vậy, thủ tục forward-backward là khả thi với chi phí tính toán

hoàn toàn có thể chấp nhận được.

Đối với việc tìm lời giải cho bài toán 1, ta chỉ cần đến phần forward trong

thủ tục forward-backward. Tuy nhiên, phần backward giúp tìm lời giải cho

bài toán 3.

* + 1. **Bài toán 2 – decoding (Thuật toán Virterbi)**

Mục tiêu của bài toán 2 là tìm ra chuỗi trạng thái “tối ưu” nhất Q = q1 q2 … qT đã phát sinh ra O. Một điều đáng lưu ý là có rất nhiều các tiêu chí “tối ưu” khác nhau cho việc xác định Q, nên lời giải cho bài toán này phụ thuộc vào tiêu chí ‘tối ưu” được chọn.

Thuật toán viterbi định nghĩa biến δt(i) là xác suất cao nhất của đoạn chuỗi trạng thái dẫn đến Si ở thời điểm t và đã quan sát được đoạn O1, O2, …, Ot cho trước mô hình λ:

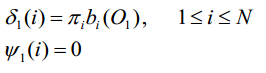


δt(i) có thể được tính theo qui nạp:

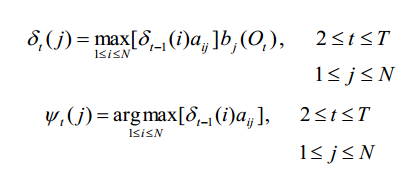


Tuy nhiên thuật toán viterbi còn cần đến mảng ψt(j) để lưu lại các tham số i làm cực đại hóa biểu thức trên, có thể hiểu tại thời điểm t trước j là i, nhằm mục đích truy vết (back trace). Chi tiết các bước của thuật toán viterbi như sau:

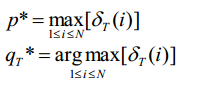
1. Khởi tạo:



1. Lặp qui nạp:



1. Kết thúc



1. quay lui – backtracking:



Kết thúc thuật toán, chuỗi Q = q1 \*q2\*..qT chính là lời giải thỏa đáng của bài toán 2.

* + 1. **Bài toán 3 – learning (Forward – Backward)**

Mục tiêu của bài toán thứ 3, cũng là bài toán phức tạp nhất trong ba bài

toán, là tìm cách cập nhật lại các tham số của mô hình λ = (π, A, B) sao cho

cực đại hóa xác suất p(O|λ) – xác suất quan sát được chuỗi tín hiệu O từ mô

hình.

Với một chuỗi tín hiệu quan sát hữu hạn bất kỳ O làm dữ liệu huấn

luyện, chưa có một phương pháp tối ưu nào cho việc ước lượng các tham số λ = (π, A, B) của mô hình theo hướng cực đại toàn cục. Tuy nhiên, bộ tham số λ có thể được chọn sao cho xác suất p(O|λ) đạt cực đại cục bộ bằng thuật toán Forward-Backward hoặc thuật toán Baum-Welch, hoặc thuật toán

Expectation-Maximization (EM).

Trước tiên, ta định nghĩa ξt(i,j) là xác suất ở trạng thái Si tại thời điểm t

và rơi vào trạng thái Sj ở thời điểm t+1 cho trước mô hình λ và chuỗi tín hiệu

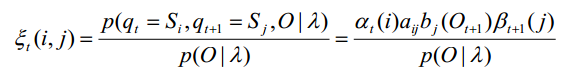
quan sát O:

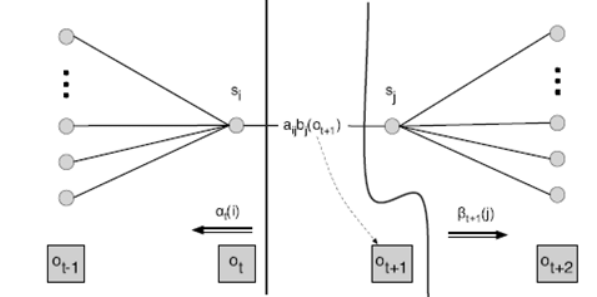


Áp dụng xác suất Bayes:



Suy ra:





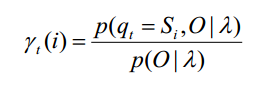


Gọi γt(i) là xác suất ở trạng thái Si vào thời điểm t cho trước chuỗi tín

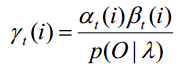
hiệu quan sát O và mô hình λ:

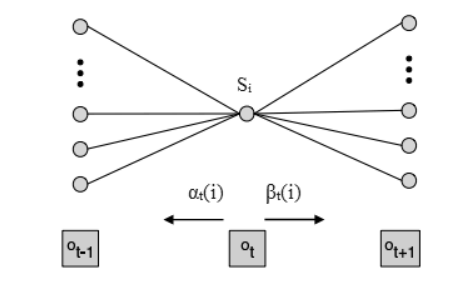


Áp dụng xác suất Bayes, suy ra:



Áp dụng xác suất forward và backward

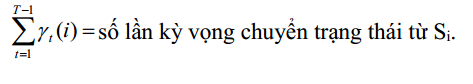


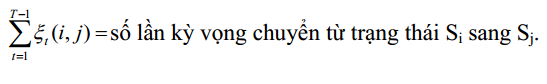




Nếu ta lấy tổng γt(i) theo t thuộc [1, T-1], kết quả nhận được là số lần kỳ

vọng chuyển trạng thái từ Si. Tương tự, lấy tổng ξt(i,j) theo t thuộc[1, T-1], ta sẽ có số lần kỳ vọng chuyển từ trạng thái Si sang Sj:

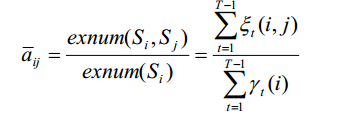


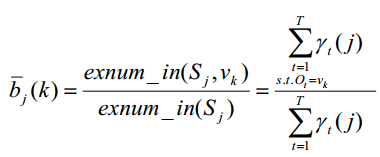


Với các đại lượng này, ta có các biểu thức cập nhật tham số của HMM

như sau:







exnum(Si,Sj): số lần kỳ vọng chuyển từ trạng thái Si sang trạng thái Sj

exnum(Si): số lần kỳ vọng chuyển trạng thái từ Si.

exnum\_in(Sj,vk): số lần kỳ vọng ở trạng thái Sj và quan sát được tín hiệu vk.

exnum\_in(Sj): số lần kỳ vọng ở trạng thái Sj.

Từ mô hình ban đầu λ = (A, B, π) và chuỗi tín hiệu quan sát O, ta tính được

vế phải của các biểu thức trên, kết quả nhận được chính là các tham

số mới của mô hình



Như vậy, thuật toán Baum-Welch sẽ được áp dụng qua nhiều bước lặp để ước lượng và cập nhật các tham số mô hình cho đến khi hội tụ. Tuy nhiên, kết quả cuối cùng chỉ đạt được tối ưu cục bộ mà thôi. Thông thường, nếu các tham số được khởi tạo với các giá trị thích hợp, thì có một phần khả năng nào đó có thể giúp mô hình đạt được tối ưu toàn cục khi huấn luyện.

* 1. **MÔ HÌNH HMM TRONG NHẬN DẠNG GIỌNG NÓI**
     1. **Giới thiệu**

HMM được bắt đầu xây dựng và công bố từ những năm 1960, đây là mô hình toán học về thống kê. Nhưng đến 1980, mô hình này mới bắt đầu được nghiên cứu để ứng dụng trong lĩnh vực nhận dạng, đặc biệt trong lĩnh vực nhận dạng giọng nói. Từ năm 1980, các phương pháp thống kê bắt đầu được sử dụng mạnh mẽ trong kỹ thuật nhận dạng, đặc biệt là mô hình Markov ẩn do có độ chính xác cao.

* + 1. **HMM trong nhận dạng giọng nói**

Đây là mô hình thống kê, thành phần của mô hình bao gồm tập N trang thái {Si}. Các trạng thái dịch chuyển qua lại với nhau với một xác suất nhất định, tập xác suất này được gọi là ma trận dịch chuyển trạng thái A = [aij]. Mô hình hoạt động với chuỗi dữ liệu đầu vào là O = {O1O2…OT} gọi là chuỗi quan sát. Đây là dữ liệu rút trích từ giọng nói cần nhận dạng.

Mỗi quan sát Ot có một xác suất xuất hiện trên mỗi trạng thái Si. Tập xác suất này gọi là phân phối xác suất của quan sát B.

(1.62)

B = {bj(Ot)}

Ngoài ra còn có tập = {} là xác suất quan sát đầu tiên o1 tại trạng thái i. Tập λ = (π, A, B, S) là các tham số của HMM. Khi có chuỗi quan sát được đưa vào mô hình, từ đầu ra ở các trạng thái sẽ rút ra được các tham số ẩn trong chuỗi quan sát.

* + - 1. **Hai vấn đề chính của HMM**

Có 2 vấn đề chính cần phải giải quyết để nó có thể ứng dụng trong hệ thống nhận dạng [18]:

Vấn đề 1 (Nhận dạng): Cho chuỗi quan sát O = {O1O2…OT} và mô hình λ. Tính xác suất P(O| λ) của chuỗi O trên mô hình đó.

Vấn đề 2 (Huấn luyện): Làm thế nào điều chỉnh các tham số của mô hình λ để P(O| λ) cực đại.

Giải quyết vấn đề 1: Để xác định xác suất của chuỗi quan sát O trên mô hình có sẵn λ, chúng ta sử dụng thuật toán lan truyền xuôi đã được trình bày ở mục

Giải quyết vấn đề 2: Nội dung vấn đề 2 là thực hiện quá trình huấn luyện hệ thống để điều chỉnh mô hình λ sao cho đạt được các thông số tối ưu. Để thực hiện điều này, chúng ta sẽ sử dụng thuật toán Baum-Welch.

 = 

(1.64)

(1.63)

 = 

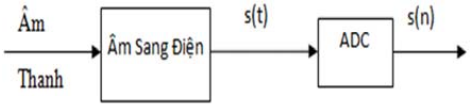
 = 

(1.65)

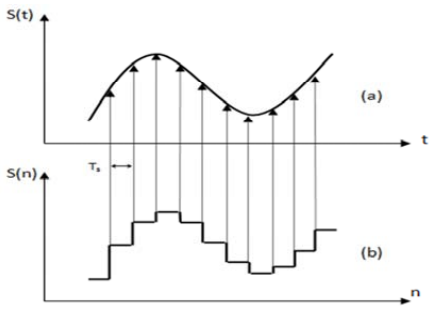
Với R là số mẫu giọng nói để huấn luyện của mỗi từ.

* + - 1. **Giọng nói và đặc trưng**

Giọng nói là một dạng sóng âm thanh dùng để giao tiếp của con người. Để có thể lưu trữ, xử lý, phân tích và nhận dạng với sự hỗ trợ của máy tính, tín hiệu giọng nói cần phải được chuyển thành tín hiệu điện tương tự và thông qua bộ biến đổi để chuyển tín hiệu điện tương tự thành tín hiệu số.



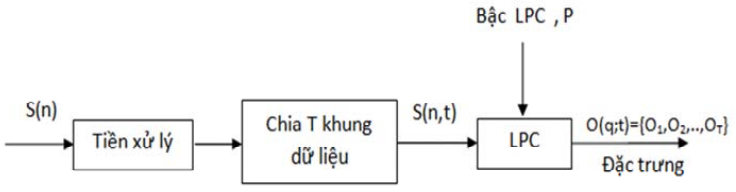
***Hình Biến đổi tín hiệu âm thanh sang tín hiệu số.***



***Hình 1. 0.23 (a) - Tín hiệu tương tự, (b) - Tín hiệu số.***

* + - 1. **Rút trích đặc trưng giọng nói**

Giọng nói hay âm thanh trước khi được phân tích hay nhận dạng cần phải được rút trích các đặc trưng của nó. Bởi vì dữ liệu giọng nói có nhiều thông tin nên chỉ cần rút trích những thông tin cần thiết cho quá trình nhận dạng. Có nhiều phương pháp để rút trích, một trong số đó là phương pháp LPC được đánh giá rất hiệu quả trong nhận dạng giọng nói.

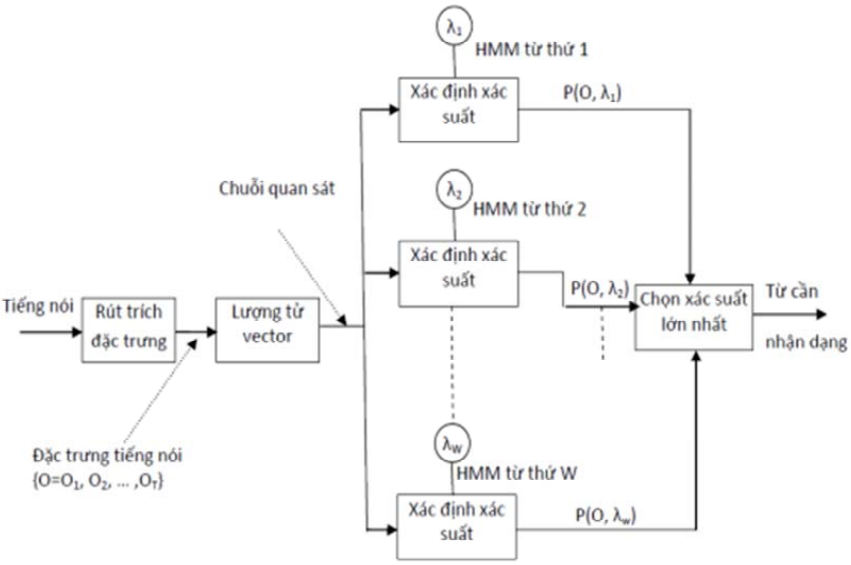


***Hình Rút trích đặc trưng giọng nói.***

Mẫu giọng nói dưới dạng số S(n) trước khi rút trích được đưa qua khối tiền xử lý để loại bỏ các nhiễu hoặc các tín hiệu tần số cao (vượt qua tần số âm thanh) bằng các mạch lọc số, sau đó sẽ chia thành T khung dữ liệu. Cuối cùng tín hiệu qua khối LPC với bậc P để đạt được chuỗi đặc trưng O = {O1O2…OT} của mẫu giọng nói, với mỗi thành phần Oi là một vector có P phần tử dưới dạng số thực.

* + - 1. **Mô hình nhận dạng giọng nói dựa trên HMM**

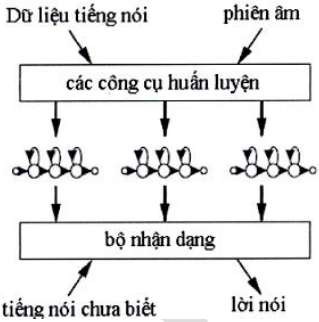
Vấn đề đặt ra là cần xây dựng một máy nhận dạng có thể nhận biết W từ đơn [20]. Để làm được điều này, chúng ta phải xây dựng mô hình cho mỗi từ thông qua quá trình huấn luyện (vấn đề 2). Giọng nói cần nhận dạng sẽ được rút trích đặc trưng và đưa vào khối lượng tử để rời rạc đặc trưng giọng nói thành các chuỗi quan sát [9]. Sau đó, chuổi quan sát sẽ được tính xác suất trên mô hình HMM của mỗi từ (vấn đề 1). Từ nhận dạng được sẽ là từ có xác suất cao nhất.



***Hình Mô hình máy nhận dạng giọng nói với HMM.***

* + - 1. **Nhận dạng giọng nói với HTK và Sphinx**

HTK (Hidden Markov Model Toolkit) là một bộ công cụ để xây dựng mô hình Markov ẩn (HMM).



***Hình Nhận dạng tiếng nói với HTK***

Trong hình mô tả một hệ thống nhận dạng tiếng nói sử dụng HTK bao gồm 2 giai đoạn chính:

1. Các công cụ huấn luyện HTK được sử dụng để ước lượng các tham số của một tập HMM sử dụng các câu nói và kết hợp các phát âm của chúng để huấn luận.
2. Sau đó, một câu nói chưa biết thông qua công cụ nhận dạng HTK sẽ cho kết quả là các phát âm của những tiếng nói này.

Tương tự như HTK, Sphinx cũng là một hệ thống nhận dạng giọng nói. Tuy nhiên khác với các hệ nhận dạng khác, mô hình HMM trong Sphinx có cấu trúc cố định trong bộ nhớ và đơn thuần chỉ là một đồ thị có hướng của các đối tượng. Trong đồ thị này, mỗi nút tương ứng với một trạng thái HMM và mỗi cung biểu diễn xác suất biến đổi từ trạng thái này sang trạng thái khác trong HMM.

Mỗi trạng thái HMM có khả năng phát sinh một đánh giá từ một đặc trưng quan sát. Quy tắc để tính toán điểm số được thực hiện bởi chính trạng thái HMM, do đó che dấu các thực thi của nó với phần còn lại của hệ thống.

* **So sánh hệ thống HTK và Sphinx :**

Sau quá trình thử nghiệm trên cả hai hệ thống HTK và Sphinx, chúng ta dễ dạng nhận thấy việc triển khai huấn luyện trên HTK thường xuyên gặp nhiều khó khăn do quy trình thực hiện bao gồm nhiều công đoạn phức tạp hơn Sphinx. Ngoài ra, tài liệu tham khảo HTK tuy phong phú nhưng khá nặng về mặt kỹ thuật vì thế đối người mới bắt đầu sẽ gặp nhiều khó khăn. Trong khi đó, Sphinx cung cấp một trang chủ [10] khá đầy đủ thông tin kèm một cộng đồng trên diễn đàn thường xuyên được cập nhật và giải đáp thắc mắc. Vì thế, đối với người muốn phát triển nhanh một hệ nhận dạng giọng nói cho một ngôn ngữ như tiếng Việt thì Sphinx là sự lựa chọn ưu tiên. Đó cũng chính là lý do mà đề tài này chọn Sphinx làm công cụ huấn luyện giọng nói thay vì chọn HTK.

# 1.6 Các khái niệm và thuật ngữ

**Acoustic model:** mô hình âm học – là bản lưu các âm kèm với kí tự text.

VD: chữ A -> đọc là ah, được lưu theo định dạng từ điển lexicon.

**Cepstra / Cepstral/ Cepstrum:** Là một mô hình vec-tơ đặc trưng, được dùng riêng cho kỹ thuật nhận dạng giọng nói. Hệ thống sẽ chuyển giọng

của chúng ta thành mô hình này để tiện cho việc phân tích, có thể hiểu như hệ

thống sẽ chuyển đổi lời nói của chúng ta thành một dãy các biến số. Khi một

biến số thay đổi, nghĩa là chúng ta nói một từ khác.

**Corpus (Corpora):**  là một tập hợp các file giọng nói được thu

sẵn để giúp máy học tốt hơn. Corpus này có thể là 1 tập tin dài có tiếng nói

được thu âm sẵn (vd: các file của đài tiếng nói việt nam VOV, dài nhiều giờ

đồng hồ). Máy sẽ cố gắng nhận dạng các tập tin này, và lưu lại các kết quả có

thích phù hợp cao, và biến đó thành các âm chuẩn làm mẫu. Lấy ví dụ: trong

cơ sở dữ liệu ta lưu từ “một” với tốc độ đọc vừa phải làm chuẩn, khi tìm thấy

1 từ “ một” có tốc độ đọc nhanh hơn một tí, với cao độ và âm sắc khác đi một

tí, máy sẽ cho điểm tương thích cho từ một này, đạt đến mức nhất định, hệ

thống sẽ chấp nhận và lưu lại. Sau này khi ta đọc từ “một” nhanh hơn một tí,

máy sẽ lập tức nhận ra ngay.

**HMM: Hidden Markov Models** - mô hình markov ẩn – là một mô hình

dùng để phân tích các âm theo đặc trưng và giúp hệ thống xác định sự khác

biệt giữa các âm thanh.

**Language model:** mô hình ngôn ngữ - đây như là 1 bộ từ điển sự kết

hợp giữa các từ có thể có, để giúp hệ thống tiên đoán chính xác hơn. VD: từ

“trái” sẽ có các từ đi sau như “cam”, “mận” … Nhờ có mô hình ngôn ngữ mà

hệ thống nhận dạng chính xác hơn.

**Linguist:** Cơ sở dữ liệu của Sphinx, lưu trữ các âm thanh chuẩn làm

mẫu để so sánh , phân tích và nhận dạng. Gồm 2 phần là: accoustic model và

language model.

**Lattice:** lưới – là một đồ thị có hướng diễn tả các biến trong quá trình

nhận dạng. Thường là chọn ra các tổ hợp không có trong thực tế nhất ; Trong

trường hợp đó đo, lưới (lattice ) đóng vai trò trung gian để trình bày kết quả.

N-best list: danh sách các biến tốt nhất giống như lattice – lưới, nhưng

các tổ hợp kết quả thưa hơn lattice.

**Speech database:** Cơ sở dữ liệu lời nói – là một tập hợp các đoạn ghi

âm điển hình lấy từ các cơ sở dữ liệu nhiệm vụ. Nếu chúng ta phát triển hệ

thống đối thoại, nó có thể đối thoại với các đoạn ghi âm của người dùng . Đối

với hệ thống chính tả, nó có thể đọc các đoạn ghi âm. CSDL lời nói thường

được dùng để huấn luyện, đồng điệu và kiểm tra hệ thống giải mã.

**Text databases :** các CSDL văn bản là các mẫu văn bản thu gom để

dùng cho việc huấn luyện mô hình ngôn ngữ và nhiều ứng dụng khác. Thông

thường, CSDL văn bản được thu gom trong một mẫu văn bản. Vấn đề là

chuyển các tài liệu hiện tại (PDF , trang web , bản scan ) thành các mẫu văn

bản nói. Nghĩa là, bạn cần phải loại bỏ các thẻ và tiêu đề, mở rộng số hình

thức nói của họ, và mở rộng các chữ viết tắt.

**Word confusion networks :** mạng các từ nghi vấn là các lưới lattice

mà trật tự chặt chẽ của các nốt được lấy trong các cạnh của lưới lattice.

# CHƯƠNG 2: ****CÔNG CỤ HUẤN LUYỆN VÀ NHẬN DẠNG GIỌNG NÓI****

* 1. **TỔNG QUAN**

Huấn luyện và nhận dạng là hai công đoạn quan trọng trong một hệ thống nhận dạng giọng nói. Hiện nay có hai công cụ hỗ trợ cho việc này là HTK và CMU Sphinx.

Trích chọn   
đặc trưng

(Feature   
Extraction)

Mô hình huấn luyện

(Training Acoustic Model)

Bộ dữ liệu huấn luyện

(Training Data)

HMM

Bộ dữ liệu kiểm tra

(Testing Data)

So khớp

(Viterbi Decoder)

Trích chọn   
đặc trưng

(Feature   
Extraction)

Nhận dạng

***Hình 2.2.1. Mô hình tổng quát quá hình huấn luyện và nhận dạng giọng nói***

* 1. **HUẤN LUYỆN VÀ NHẬN DẠNG GIỌNG NÓI VỚI CMU SPHINX**
     1. **Giới thiệu**

Sphinx4 là một hệ thống được viết bằng ngôn ngữ lập trình Java. Nó được phát minh bởi nhóm nhà khoa học thuộc Trường Đại học Carnegie Mellon, Sun Microsystems Laboratories, Mitsubishi Electric Research Labs (MERL) và Hewlett Packages (HP) cùng với sự đóng góp của Đại học California Santa Cruz (UCSC) và Viện Công nghệ Massachusetts (MIT).

Sphinx4 là hệ thống nhận dạng giọng nói được xây dựng với kỹ thuật cốt lõi là HMM. Sphinx4 cũng có thể được sử dụng để nhận dạng nhiều dạng ngôn ngữ khác nhau trên thế giới bằng cách thiết lập lại một số thông số của Sphinx4.

Sphinx4 được xem là phiên bản tiên tiến nhất của Sphinx. Nó vẫn giữ tất cả chức năng của phiên bản trước đó và phát triển thêm những tính năng mới.

Sphinx4 là một hệ nhận dạng giọng nói dựa trên mô hình HMM [19] được xem là phổ biến nhất bởi vì khả năng nhận dạng với độ chính xác cao ngay cả trong trường hợp nhận dạng với từ vựng lớn. Hơn nữa, Sphinx4 cũng còn có những điểm mạnh khác mà hiếm khi chúng ta tìm thấy trong những hệ nhận dạng khác. Từng bước, chúng ta sẽ phân tích lý do tại sao chúng ta nên sử dụng nó trong phần giới thiệu các thành phần chính của Sphinx4.

* + 1. **Vận dụng mô hình HMM trong nhận dạng giọng nới với Sphinx4**

Mô hình Markov ẩn (HMM) là một mô hình thống kê, trong đó hệ thống đang được mô hình giả định là một quá trình Markov với các tham số chưa biết và nhiệm vụ là xác định các tham số ẩn từ các tham số quan sát được [9].

Trong quá trình nhận dạng giọng nói, sau khi giọng nói được thu nó sẽ được chia thành các khung mà chúng ta cần xử lý để tạo ra các câu văn bản. Mỗi khung tượng trưng một trạng thái, nhóm các trạng thái tượng trưng cho âm vị và nhóm của nhiều âm vị được biểu diễn như một từ mà chúng ta cần nhận dạng.

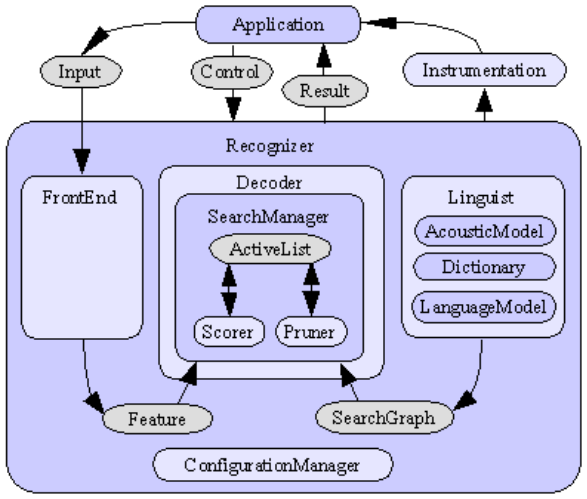
Trong cơ sở dữ liệu gọi là mô hình ngôn ngữ, chúng ta lưu trữ các giá trị tham chiếu của trạng thái, âm vị và từ để so sánh với các dữ liệu quan sát được (giọng nói).

Bằng cách sử dụng HMM, chúng ta sử dụng một mô hình thống kê trên mỗi âm vị. Và mục tiêu của hệ nhận dạng giọng nói là tìm ra dãy các trạng thái mà có xác suất cao nhất.

* + 1. **Các thành phần chính trong CMU Sphinx**

Có ba thành phần chính trong hệ thống nhận dạng Sphinx4. Đó là Frontend, Decoder và Linguist. Mỗi thành phần sẽ được mô tả sau. Trong luận văn này, chỉ tập trung giải thích thành phần Linguist bởi vì đây là thành phần cấp cao mà chúng ta có thể truy cập trực tiếp để thay đổi dữ liệu cần thiết cho việc xây dựng một ứng dụng mới.

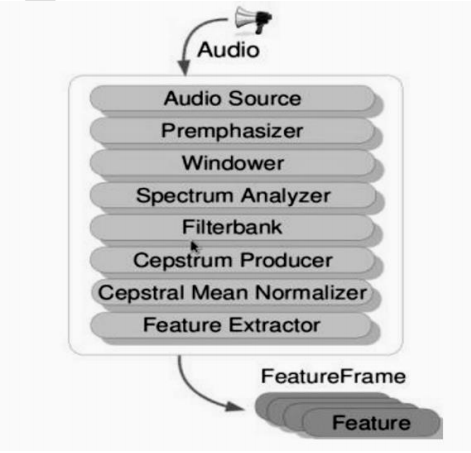
Hai thành phần xử lý cốt lõi của Sphinx4 có thể kể đến là quá trình huấn luyện (training) và quá trình giải mã (decoding). Quá trình huấn luyện là một trong những bước cần thiết để xây dựng mô hình ngôn ngữ học (bao gồm mô hình ngôn ngữ và mô hình âm học). Ở bước này, chúng ta cần thu âm giọng nói của nhiều người làm cơ sở dữ liệu huấn luyện. Sau khi hệ thống được huấn luyện, nó có thể được sử dụng để nhận dạng giọng nói bằng cách giải mã giọng nói thành văn bản tương ứng. Quá trình nhận dạng được mô tả rất chi tiết trong biểu đồ dưới đây.



***Hình 1. 0.1 Kiến trúc hệ thống Sphinx4***

* + - 1. **Thành phần Frontend**

Chức năng của thành phần này là nhận vào các tín hiệu từ bên ngoài, thực hiện qua một số bộ lọc và xử lý dữ liệu để cho kết quả là một tập các vector đặc trưng.



***Hình 1. 0.2. Kiến trúc thành phần Frontend***

Bên trong của thành phần Frontend này là một chuỗi các mô đun con có khả năng xử lý tín hiệu giao tiếp được nối ghép với nhau. Các chuỗi này có thể hoạt động song song gọi là DataProcessor.



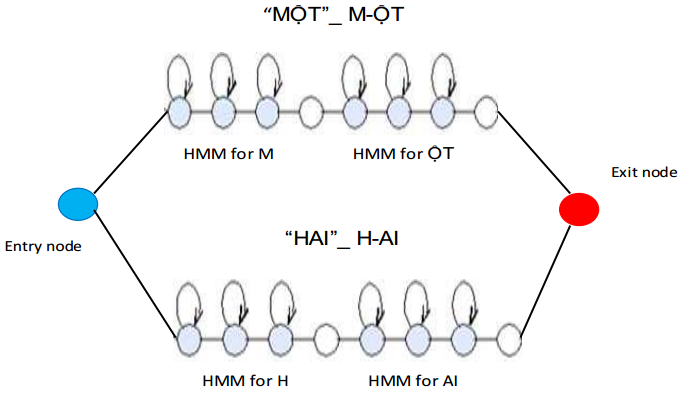
***Hình 1. 0.3 Chuỗi các xử lý của thành phần Frontend.***

Việc xử lý bằng các chuỗi DataProcessor nối ghép với nhau giúp chúng ta linh động trong việc rút trích đặc trưng của âm thanh. Với một tín hiệu, ta có thể rút trích đặc trưng bằng nhiều phương pháp khác nhau sau đó so sánh và tìm ra các đặc trưng tối ưu hoặc áp dụng liên tiếp các phương pháp với nhau để tìm ra đặc trưng tốt nhất. Đồng thời cho phép các chuỗi này thực hiện song song với nhau làm cho hệ thống nhận dạng nhanh chóng trong thời gian thực với lượng tín hiệu đầu vào lớn.

* + - 1. **Thành phần Search graph**

Đây là kết quả mà bộ ngôn ngữ phát sinh được cuối cùng để đưa vào sử dụng trong bộ giải mã. Cấu tạo của thành phần Search graph này là một đồ thị có hướng, trong đó mỗi nút được gọi là một trạng thái tìm kiếm dùng để biểu diễn một trong hai trạng thái là phát (emitting state) hoặc không phát (non-emitting state). Mỗi cung của đồ thị biểu diễn sự biến đổi của các trạng thái. Trên các cung này, có các giá trị xác suất được tính toán từ mô hình âm học nhằm biểu diễn khả năng chuyển từ trạng thái này đến trạng thái khác. Một trạng thái có thể có nhiều cung hướng đến trạng thái khác.

Hãy xem xét thành phần Search graph trong việc giải mã các từ "một" và "hai".



***Hình 1. 0.4 Thành phần Search graph trong giải mã tiếng Việt***

* + - 1. **Thành phần Decoder**

Thành phần này có nhiệm vụ chính là sử dụng các đặc trưng từ thành phần Fontend kết hợp với thành phần Search graph để tiến hành giải mã và áp dụng các thuật toán để cho ra kết quả nhận dạng.

Thành phần Decoder có cấu tạo đơn giản nhưng phưc tạp về mặt thuật toán cài đặt vì ảnh hưởng tới tốc độ tìm kiếm, bao gồm một thành phần quản lý tìm kiếm (Search Manager) có khả năng tháo lắp và các mã hỗ trợ khác để đơn giản hóa quá trình giải mã cho một ứng dụng.

Nhiệm vụ của thành phần quản lý tìm kiếm là nhận dạng các tập vector đặc trưng để tìm ra ánh xạ tương ứng của nó trong đồ thị tìm kiếm (Search graph). Để cho kết quả chính xác, Sphinx cung cấp các tiện ích có khả năng phát sinh lưới và các đánh giá độ tin cậy từ kết quả. Ngoài ra, Sphinx còn cho phép thay đổi thiết lập để tăng hiệu suất tìm kiếm.

* + - 1. **Thành phần Linguist**

Bằng các công cụ và phương pháp ngôn ngữ, thành phần này sẽ đọc vào các tập tin cấu trúc của một ngôn ngữ rồi mô hình hóa chúng vào thành phần Search graph để sử dụng trong việc tìm kiếm nhận dạng.

Thành phần này có cấu tạo rất phức tạp vì nó quy định hầu như toàn bộ phạm vi ngôn ngữ mà chúng ta cần nhận dạng, bao gồm các thành phần con sau :

1. *Thành phần mô hình ngôn ngữ*

Thành phần này có chức năng đọc vào tập tin cấu trúc ngôn ngữ ở cấp độ các từ (word level). Đây cũng là thành phần có vai trò quan trọng để xác định những nội dung cần nhận dạng. Cấu trúc ngôn ngữ được mô hình hóa ở thành phần này bao gồm hai dạng : Graph-driven grammar và Stochastic N-Gram.

Mô hình Graph-driven grammar biểu diễn một đồ thị có hướng trong đó mỗi nút biểu diễn một từ đơn và mỗi cung biểu diễn xác suất dịch chuyển sang một từ khác.

Mô hình Stochastic N-Gram cung cấp các xác suất cho các từ được cho là dựa vào việc quan sát n-1 từ trước đó.

Nhiều dạng ngôn ngữ được Sphinx hỗ trợ như :

* SimpleWordListGrammar : Định nghĩa một từ dựa trên danh sách các từ. Một tham số tùy chọn chỉ ra ngữ pháp có lặp hay không. Trường hợp không lặp, ngữ pháp sẽ được dùng cho nhận dạng từ tách biệt.
* JSGFGrammar : Hỗ trợ JavaTM Speech API Grammar Format (JSGF), định nghĩa một biểu diễn theo BNF, độc lập với các nền tảng.
* LMGrammar : Định nghĩa một ngữ pháp dựa trên mô hình ngôn ngữ thống kê và được đánh giá làm việc tốt với các mô hình ngôn ngữ xấp xỉ 1000 từ.
* FSTGrammar : Hỗ trợ một bộ chuyển đổi trạng thái giới hạn trong định dạng ngữ pháp ARPA FST.
* SimpleNGramModel : Được cung cấp để hỗ trợ cho các mô hình ASCII N-Gram trong định dạng ARPA. SimpleNGramModel phù hợp với các mô hình ngôn ngữ nhỏ.
* LargeTrigramModel: Hỗ trợ các mô hình N-Gram được phát sinh bởi CMU Cambridge Statictical Language Modeling Toolkit. Large Trigram Model tối ưu việc lưu trữ bộ nhớ nên phù hợp với các mô hình ngôn ngữ lớn (trên 100MB).

1. *Thành phần từ điển*

Thành phần này có nhiệm vụ cung cấp cách phát âm cho các từ mà chúng ta đã xây dựng trong mô hình ngôn ngữ và hỗ trợ phân lớp các từ thành các lớp khác nhau để hỗ trợ cho việc tìm kiếm.

1. *Thành phần mô hình âm học*

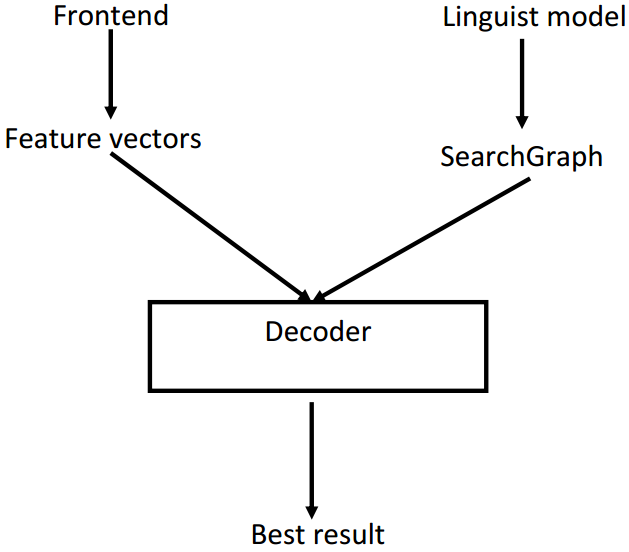
Thành phần mô hình âm học cung cấp một ánh xạ giữa một đơn vị tiếng nói và HMM có thể được đánh giá dựa vào các đặc trưng được cung cấp bởi thành phần Fontend. Các ánh xạ có thể đưa thông tin vị trí của các từ và ngữ cảnh từ thành phần mô hình ngôn ngữ. Định nghĩa ngữ cảnh này được xây dựng từ cấu trúc ngữ pháp của mô hình ngôn ngữ.

* + 1. **Các xử lý chính trong CMU Sphinx**
       1. **Huấn luyện**

Quá trình huấn luyện được sử dụng để xây dựng dựng mô hình ngôn ngữ học (bao gồm mô hình ngôn ngữ và mô hình âm học) được coi là thành phần quan trọng nhất của một công cụ nhận dạng giọng nói xây dựng trên Sphinx4. Bước ban đầu của quá trình huấn luyện là định nghĩa một danh sách âm vị và từ điển cho mô hình. Sau đó, chúng ta sẽ thu âm giọng nói để làm dữ liệu huấn luyện (training data). Trong quá trình huấn luyện, giọng nói sẽ được chuyển thành dạng cepstral (những vector đặc trưng). Dựa trên những vector đặc trưng này, HMMs được xây dựng ch mỗi âm vị trong danh sách âm vị. Sau khi huấn luyện, chúng ta sẽ có một mô hình ngôn ngữ học được xây dựng trên HMM mà sẽ được sử dụng để nhận dạng giọng nói.

* + - 1. **Nhận dạng**

Nhiệm vụ chính của quá trình nhận dạng là tìm ra từ tốt nhất có thể mà sẽ phù hợp với giọng nói tương ứng. Giai đoạn đầu tiên của quá trình nhận dạng là giống với quá trình huấn luyện, giọng nói sẽ được thu âm và chuyển thành các vector đặc trưng bởi thành phần frontend. Mô hình ngôn ngữ học sẽ được sử dụng để định nghĩa từ cần giải mã. Tại bộ giãi mã, dựa vào thành phần Search graph và vector đặc trưng, nó sẽ áp dụng thuật toán tìm kiếm để tìm ra kết quả tốt nhất.



***Hình 1. 0.5 Quá trình nhận dạng.***

* 1. **KẾT LUẬN**

Sphinx4 được coi là một framework tốt cho các sinh viên, các giáo sư sử dụng trong các nghiên cứu của họ. Sphinx4 cung cấp rất nhiều tính năng mà chúng ta không thể tìm thấy ở các framework khác.

* + 1. **Ưu điểm**
       1. **Sphinx4 là một hệ thống nhận dạng tiếng nói liên tục**

Sphinx4 được phát triển bằng cách kết hợp các mô hình tiên tiến trước đó cũng như những yêu cầu mới trong nhiều lĩnh vực. Nó tăng cường và mở rộng những khả năng của các phiên bản trước như Sphinx2, Sphinx3 [10].

Hầu hết các công nghệ nhận dạng giọng nói sử dụng để phát triển Sphinx là các công nghệ tiên tiến được cải thiện thường xuyên. Ví dụ, chất lượng thu âm giọng nói được tăng lên bằng cách áp dụng rất tốt kỹ thuật lọc tiếng ồn.

* + - 1. **Sphinx4 là một hệ thống có khả năng tùy chỉnh linh động**

Kiến trúc mô đun của Sphinx4 cho phép chúng ta phân chia hệ thống nhận dạng thành nhiều phần nhỏ và giúp chúng ta có thể nghiên cứu từng thành phần một cách riêng biệt. Các nhà nghiên cứu thậm chí có thể phát triển thêm những tính năng mới để cải thiện hiệu suất nhận dạng.

Kiến trúc của thành phần Frontend bao gồm việc cài đặt của nhiều mô đun để xử lý các công đoạn như : tiền khuếch đại, cửa sổ Hamming, tần số Mel, ngân hàng bộ lọc tần số, biến đổi cosin rời rạc và trích xuất đặc trưng cepstral,…

Kiến trúc mô hình ngôn ngữ bao gồm nhiều mô hình ngôn ngữ riêng lẻ hỗ trợ ASCII và các phiên bản nhị phân như unigram, bigram, trigram, Java Speech API Grammar format (JSGF), and ARPA format FST grammars.

Xây dựng sẵn các thuật toán tìm kiếm tối ưu như thuật toán breath first và thuật toán word pruning giúp chúng ta dễ dàng tùy chỉnh cho phù hợp với nhu cầu nhận dạng.

* + - 1. **Sphinx4 có thể thực hiện nhiều tác vụ nhận dạng**

Hệ thống Sphinx4 có thể làm việc với các hình thức đầu vào (input) khác nhau để cho ra kết quả tương tự. Người sử dụng có thể nói trực tiếp với hệ thống thông qua microphone, kết quả nhận dạng gần như ngay lập tức được chuyển thành dạng văn bản (text). Bên cạnh đó, hệ thống còn cho phép chúng ta nhận dạng giọng nói thông qua các tập tin âm thanh đã được thu âm sẵn. Nội dung của những tập tin này cũng được hệ thống nhận dạng như cách người dùng nói trực tiếp vào hệ thống.

Sphinx4 cung cấp cho chúng ta khả năng làm việc với các dạng từ vừng có kích thước khác nhau. Chúng ta có thể nhận dạng chỉ với những con số rời rạc, những con số liên tục, số lượng từ nhỏ hoặc kết hợp chúng lại để tạo thành lượng từ vựng lớn. Trong hệ thống Sphinx4, những nhà phát triển Sphinx đã xây dựng lượng từ vựng lên đến 64.000 từ cho tiếng Anh. Đó là điều khá tốt cho chúng ta sử dụng Sphinx4 trong việc nhận dạng lượng từ vựng tiếng Việt lớn.

* + - 1. **Sphinx4 có thể làm việc độc lập với người nói**

Sau khi hệ thống được nhận dạng với Sphinx4, nó có thể nhận dạng giọng nói của bất kỳ ai. Điều này có thể được giải thích bằng cách xem xét HMM mà được sử dụng như là cốt lõi của Sphinx4. Bởi vì kết quả được tao ra dựa trên những tham số quan sát được, vì vậy giọng nói của người được nhận dạng không cần thiết phải chính xác như giọng được huấn luyện.

Chúng ta hãy xem xét tiếng Việt với giọng nói của ba vùng miền như miền Bắc, miền Trung và miền Nam. Nếu chúng ta sử dụng sphinxtrain cho cả ba vùng, hệ thống có thể nhận dạng tất cả giọng nói tiếng Việt với độ chính xác rất cao.

* + - 1. **Sphinx4 là hệ thống mã nguồn mở**

Chúng ta có thể chạy hệ thống Sphinx4 trên hầu hết các hệ điều hành và môi trường có hỗ trợ nền tảng Java như windows, linux, solaris,…Nó rất thuận tiện cho tất cả mọi người làm việc quen thuộc với các nền tảng khác nhau có thể hợp tác để cùng làm việc với Sphinx4.

Bên cạnh đó, Sphinx4 là miễn phí để tải về và sử dụng, vì thế nó rất phù hợp cho sinh viên, giáo sư, các nhà nghiên cứu… những người quan tâm đến hệ thống nhận dạng giọng nói tạo ra những chương trình nhận dạng giọng nói của riêng mình phục vụ cho mục đích nghiên cứu của họ.

* + 1. **Nhược điểm**

Sphinx4 là một hệ thống mã nguồn mở vì vậy nó không phải là hệ thống chuẩn mà có thể dễ dang đươc thực thi, nó bao gồm một loạt các gói mà người sử dụng cần phải cấu hình để chạy hệ thống.

Bên cạnh đó, Sphinx4 vẫn tồn tại một số lỗi mà chúng ta cần phải khắc phục nếu chúng ta muốn phát triển một công cụ nhận dạng giọng nói dựa trên Sphinx4. Bởi vì không có nhiều tài liệu hướng dẫn chi tiết về kiến trúc của Sphinx4 và cách mà hệ thống làm việc. Vì vậy chúng ta vẫn còn hiểu biết mơ hồ về nó và chính vì thế chúng ta có thể gặp phải những lỗi bất ngờ trong quá trình sử dụng [10].

* 1. **SPHINX 4 VỚI NGÔN NGỮ TIẾNG VIỆT**
     1. **Tổng quan**

Các hệ thống nhận dạng giọng nói yêu cầu hai loại tập tin tương ứng với hai mô hình để nhận dạng giọng nói. Thứ nhất là mô hình âm học (Acoustic Model), được tạo ra bằng cách ghi âm giọng nói kết hợp với phiên âm của chúng và sau đó biên dịch (compiling) để tạo thành một đại diện thống kê của âm thanh tạo nên mỗi từ. Thứ hai là mô hình ngôn ngữ (Language Model) hoặc tập tin Grammar. Một mô hình ngôn ngữ là một tập tin chứa các xác suất của chuỗi các từ. Một Grammar là một tập tin nhỏ hơn chứa một kết hợp của các từ đã được xác định trước.

Trong phần này, chúng tôi sẽ cung cấp một phương pháp xây dựng mô hình âm học cho tiếng Việt được sử dụng cho thành phần fontend của SphinxTrain.

* + 1. **Corpus**
       1. **Giới thiệu**

Corpus là tập hợp các đơn vị âm thanh định nghĩa một từ cụ thể trong từ vựng. Đây là sự kết hợp giữa sự phát âm của âm thanh và từ điển.

Giọng nói

(Speech)

Danh sách âm vị

(Phone list)

Phát âm

(Audio utterances)

Thu âm (Recording)

Định nghĩa từ (Word definition)

Từ vựng

(Vocabulary)

Định nghĩa âm thanh

(Sound definition)

**Corpus**

Mô hình âm vị

(Acoustic model)

Huấn luyện (Training)

***Hình 2.2.2. Tổng quan về Corpus***

* + - 1. **Tầm quan trọng của Corpus trong hệ nhận dạng giọng nói**

Corpus đóng một vai trò rất quan trọng trong hệ thống nhận dạng giọng nói. So với HMM, nó được coi là xương sống của hệ thống mà chúng ta có thể sửa đổi để cải thiện độ chính xác của hệ thống.

Trong Sphinx4, những người phát triển hệ thống nhận dạng cần phải quan tâm nhiều đến vai trò của corpus. Độ chính xác của nhận dạng thông thường tỷ lệ thuận với khả năng của corpus [10].

Để tạo một corpus, chúng ta có thể thu thập cách phát âm từ nhiều nguồn. Nó có thể từ tivi, tập tin âm thanh, hoặc ghi âm trực tiếp. Tuy nhiên, điều đó không có nghĩa rằng tất cả cách phát âm có thể được sử dụng để tạo ra các corpus, chỉ có những phát âm tốt sau khi được lọc có thể được sử dụng.

# CHƯƠNG 3: ****XÂY DỰNG CHƯƠNG TRÌNH MINH HỌA TRÊN THIẾT BỊ DI ĐỘNG****

1. 1. **THIẾT LẬP MÔI TRƯỜNG CMU SPHINX**

**3.1.1 Chuẩn bị hệ điều hành**

Linux là môi trường hệ điều hành thích hợp nhất để cài đặt Sphinx và thực hiện huấn luyện. Trong các hệ điều hành Linux thì Ubuntu được xem như là hệ điều hành phổ biến. Sphinx dễ dàng được cài đặt trong hệ điều hành Ubuntu.

Tuy nhiên trong luận văn này nhóm sử dụng hệ điều hành Windows để cài đặt và thực thi việc huấn luyện file.

**3.1.2 Chuẩn bị các gói cài đặt Sphinx**

Các gói bao gồm:

* Pocketsphinx —một thư viện nhận dạng viết bằng ngôn ngữ C.
* Sphinxbase — gói thư viện nền, hỗ trợ các thư viện cần thiết cho các
* gói khác
* Sphinx4 — gói hỗ trợ nhận dạng viết bằng java
* CMUclmtk — bộ công cụ xây dựng mô hình ngôn ngữ
* Sphinxtrain — bô công cụ huấn luyện mô hình ngữ âm
* Các gói cài đặt có thể được tải trực tiếp từ trang chủ của CMU Sphinx [27]
  1. **XÂY DỰNG BỘ NGÔN NGỮ**
     1. **Giới thiệu**

Bộ ngôn ngữ cho chương trình nhận dạng giọng nói bao gồm 3 thành phần chính là bộ từ điển, mô hình ngôn ngữ và mô hình âm học. Phần này sẽ mô tả quá trình xây dựng các thành phần đó bằng các công cụ của Sphinx.

* + 1. **Xây dựng bộ từ điển**

Bộ từ điển bao gồm các ký tự hoặc các từ mong muốn chương trình nhận dạng. Do các công cụ huấn luyện của Sphinx chưa hỗ trợ tốt cho unicode nên các ký tự không thuộc bảng mã ASCII sẽ sử dụng phương pháp:

* Các ký tự không thuộc bảng mã ASCII sẽ được thay thế bằng kiểu gõ telex.
* Xây dựng bảng phiên âm tiếng Việt mức âm vị dưới dạng ASCII. (Tham khảo phụ lục).

Bảng phiên âm tiếng tiếng Việt mức âm vị được xây dựng dựa trên các tiêu chí:

* Biểu diễn được hết các âm vị có thể có trong tiếng Việt dưới dạng mã ASCII.
* Mọi âm vị đều được tổ hợp từ các ký hiệu sẵn có trên bàn phím để tiện lợi cho việc nhập liệu.
* Các thanh điệu được ký hiệu bằng các ký tự S, F, R, X, J và khoảng trắng.
* Thanh điệu được đặt cho các nguyên âm

Bộ từ điển được tổ chức như sau [7]:

cas k as

ly l y

now n ow

noofi ng oof i

vis v is

xerng x er ngz

xoo x oo

duf d uf

ghees g ees

bi b i

sasch s eas kc

gajch g aj kc

xe x e

mefo m ef o

chuoojt ch uoj t

bosng b os ngz

quaafn k aaf n

bof b of

buwowsm b was m

caso k as o

cof k of

dieefu d ief u

eesch ees kc

gaf g af

hoor h oor

nai n ai

ong o ngz

rufa r uf a

saau s aa u

sosc s os c

ve v e

vejt v ej tc

voi v o i

quaf k uof

nguwja ng waj

hoa h o a

bow b ow

maajn m aaj nz

me m e

quyst k is t

taso t as o

bieern b i er nz

boj b oj

caay k aa iz

chai ch a iz

rawsn r aws n

roofng r oof ngz

ddaaju dd aaj u

desp d es pc

gajo g aj o

guoosc g uos c

goox g oox

keso k es o

las l as

lee l ee

luwju l uwj u

thufng th uf ng

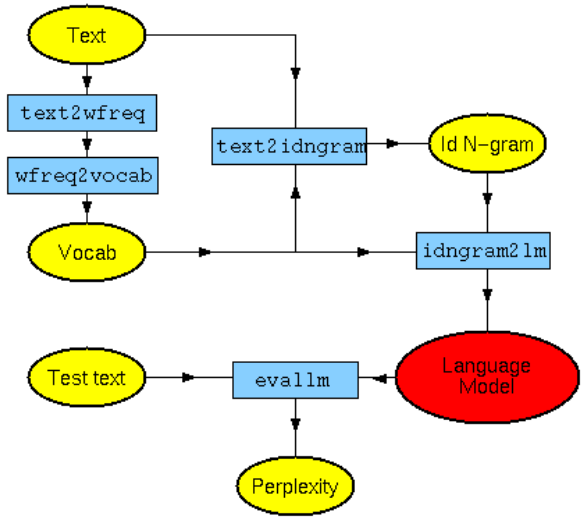
troosng tr oos ngz

tusi t us iz

mux m ux

* + 1. **Xây dựng mô hình ngôn ngữ**

Quá trình tạo mô hình ngôn ngữ được mô tả như hình sau :



***Hình 1. 0.1 Sơ đồ quá trình tạo mô hình ngôn ngữ.***

* + - 1. **Chuẩn bị tập tin văn bản**

Chuẩn bị tập tin văn bản chứa nội dung cần huấn luyện và đặt trong cặp thẻ <s></s>. Ví dụ tạo tập tin tuvung.txt có nội dung như sau:

<s> cá </s>

<s> ly </s>

<s> mũ </s>

<s> nơ </s>

<s> nồi </s>

<s> ví </s>

<s> xẻng </s>

<s> xô </s>

<s> dù </s>

<s> ghees </s>

<s> bi </s>

<s> sasch </s>

<s> gajch </s>

<s> xe </s>

<s> mefo </s>

<s> chuoojt </s>

<s> bosng </s>

<s> quaafn </s>

<s> bof </s>

<s> buwowsm </s>

<s> caso </s>

<s> cof </s>

<s> dieefu </s>

<s> eesch </s>

<s> gaf </s>

<s> hoor </s>

<s> nai </s>

<s> ong </s>

<s> rufa </s>

<s> saau </s>

<s> sosc </s>

<s> ve </s>

<s> vejt </s>

<s> voi </s>

<s> quaf </s>

<s> nguwja </s>

<s> hoa </s>

<s> bow </s>

<s> maajn </s>

<s> me </s>

<s> quyst </s>

<s> taso </s>

<s> bieern </s>

<s> boj </s>

<s> caay </s>

<s> chai </s>

<s> rawsn </s>

<s> roofng </s>

<s> ddaaju </s>

<s> desp </s>

<s> gajo </s>

<s> guoosc </s>

<s> goox </s>

<s> keso </s>

<s> las </s>

<s> lee </s>

<s> luwju </s>

<s> thufng </s>

<s> troosng </s>

<s> tusi </s>

Vì Sphinx chưa hỗ trợ unicode, nên các ký tự không thuộc ASCII sẽ được chuyển thành kiểu gõ telex:

<s> cas </s>

<s> ly </s>

<s> mux </s>

<s> now </s>

<s> noofi </s>

<s> vis </s>

<s> xerng </s>

<s> xoo </s>

<s> duf </s>

<s> ghees </s>

<s> bi </s>

<s> sasch </s>

<s> gajch </s>

<s> xe </s>

<s> mefo </s>

<s> chuoojt </s>

<s> bosng </s>

<s> quaafn </s>

<s> bof </s>

<s> buwowsm </s>

<s> caso </s>

<s> cof </s>

<s> dieefu </s>

<s> eesch </s>

<s> gaf </s>

<s> hoor </s>

<s> nai </s>

<s> ong </s>

<s> rufa </s>

<s> saau </s>

<s> sosc </s>

<s> ve </s>

<s> vejt </s>

<s> voi </s>

<s> quaf </s>

<s> nguwja </s>

<s> hoa </s>

<s> bow </s>

<s> maajn </s>

<s> me </s>

<s> quyst </s>

<s> taso </s>

<s> bieern </s>

<s> boj </s>

<s> caay </s>

<s> chai </s>

<s> rawsn </s>

<s> roofng </s>

<s> ddaaju </s>

<s> desp </s>

<s> gajo </s>

<s> guoosc </s>

<s> goox </s>

<s> keso </s>

<s> las </s>

<s> lee </s>

<s> luwju </s>

<s> thufng </s>

<s> troosng </s>

<s> tusi </s>…

* + - 1. **Xây dựng bộ từ vựng**

Bộ từ vựng là một tập tin .vocab, chứa tất cả các từ hoặc tiếng trong tập tin văn bản. Nó được tạo ra bởi CMUclmk và sẽ được dùng để tạo mô hình ngôn ngữ. Tạo bộ từ vựng này bằng cách nhập các lệnh sau vào terminal:

**text2wfreq < tuvung.txt | wfreq2vocab > tuvung.vocab**

**text2idngram -vocab tuvung.vocab -idngram tuvung.idngram < tuvung.txt**

Tập tin từ vựng tuvung.vocab chứa tất cả các từ trong văn bản được sắp xếp theo thứ tự alphabet.

* + - 1. **Xây dựng mô hình ngôn ngữ**

Mô hình ngôn ngữ có định dạng .arpa. Để tạo ra mô hình này, ta sử dụng lệnh sau:

**idngram2lm -vocab\_type 0 -idngram tuvung.idngram -vocab tuvung.vocab -arpa tuvung.arpa**

Định dạng ARPA (hay Doug Paul) cho mô hình N-gram có cấu trúc như sau:

Tập tin này có phần mở đầu với từ khóa \data\, liệt kê số lượng N-gram. Sau đó các N-gram được liệt kê mỗi dòng, được nhóm lại thành từng phần theo chiều dài. Mỗi phần bắt đầu với từ khóa \N-gram; trong đó N là chiều dài 1, 2, …. Mỗi dòng N-gram bắt đầu với logarit (cơ số 10) của điều kiện xác suất p của Ngram đó, theo sau bởi các từ w1, w2, … wN tạo nên N-gram đó. Từ khóa \end\ kết thúc biểu diễn mô hình.

All probs and back-off weights (bo\_wt) are given in log10 form.

Data formats:

Beginning of data mark: \data\

ngram 1=nr # number of 1-grams

ngram 2=nr # number of 2-grams

ngram 3=nr # number of 3-grams

\1-grams:

p\_1 wd\_1 bo\_wt\_1

\2-grams:

p\_2 wd\_1 wd\_2 bo\_wt\_2

\3-grams:

p\_3 wd\_1 wd\_2 wd\_3

end of data mark: \end\

Để Sphinx có thể sử dụng được, phải chuyển tập tin này sang dạng nhị phân bằng công cụ sphinxbase. Lệnh chuyển đổi:

**sphinx\_lm\_convert -i tuvung.arpa -o tuvung.lm.DMP**

* + 1. **Xây dựng mô hình âm học**
       1. **Giới thiệu**

Mô hình âm học bao gồm một biểu diễn thống kê các âm thanh riêng biệt tạo nên mỗi từ trong mô hình ngôn ngữ hay bộ ngữ pháp. Mỗi âm thanh riêng biệt tương ứng với một âm vị. Quá trình huấn luyện mô hình âm học được sử dụng bằng công cụ sphinxtrain.

* + - 1. **Chuẩn bị dữ liệu**

Tất cả quá trình này được thực hiện trên môi trường Windows.

Tạo thư mục dùng để huấn luyện mang tên tuvung, trong đó có 2 thư mục con là etc và wav.

Thư mục etc chứa các tập tin như sau:

tuvung.dic - bộ tự điển âm vị, âm tiết

tuvung. phone - tập tin chứa danh sách các âm vị

tuvung.lm.DMP - Mô hình ngôn ngữ

tuvung.filler - Danh sách các khoảng lặng

tuvung \_train.fileids - Danh sách các tập tin huấn luyện

tuvung \_train.transcription - Dữ liệu dạng văn bản của tập tin huấn luyện

tuvung \_test.fileids - Danh sách các tập tin test

tuvung \_test.transcription – Dữ liệu dạng văn bản của tập tin test

Tạo 2 thư mục con là train và test trong thư mục wav, trong đó thư mục train chứa dữ liệu dùng để huấn luyện, thư mục test chứa dữ liệu dùng để test.

wav

| train

| từ\_huấn luyện\_1.wav – tập tin thu âm lần 1 của người huấn luyện dành cho từ thứ nhất.

| từ\_huấn luyện\_2.wav – tập tin thu âm lần 2 của người huấn luyện dành cho từ thứ nhất.

| …

| từ\_huấn\_luyện\_1.wav – tập tin thu âm lần 1 của người huấn luyện dành cho từ thứ hai.

| từ\_huấn\_luyện\_2.wav – tập tin thu âm lần 2 của người huấn luyện dành cho từ thứ hai.

| …

| từ\_huấn\_luyện\_1.wav – tập tin thu âm lần 1 của người huấn luyện dành cho từ thứ ba.

| từ\_huấn\_luyện\_2.wav – tập tin thu âm lần 2 của người huấn luyện dành cho từ thứ ba.

| …

| test

| từ\_nhận\_dạng\_1.wav – tập tin thu âm lần 1 của từ cần nhận dạng dành cho từ thứ nhất.

| từ\_nhận\_dạng\_2.wav – tập tin thu âm lần 2 của từ cần nhận dạng dành cho từ thứ nhất.

| …

| từ\_nhận\_dạng\_1.wav – tập tin thu âm lần 1 của từ cần nhận dạng dành cho từ thứ hai.

| từ\_nhận\_dạng\_2.wav – tập tin thu âm lần 2 của từ cần nhận dạng dành cho từ thứ hai.

| …

| từ\_nhận\_dạng\_1.wav – tập tin thu âm lần 1 của từ cần nhận dạng dành cho từ thứ ba.

| từ\_nhận\_dạng\_2.wav – tập tin thu âm lần 2 của từ cần nhận dạng dành cho từ thứ ba.

| …

**Tập tin tuvung.dic**

Tập tin này là tập tin từ điển đã chuẩn bị từ đầu. Nó chứa nội dung về cách phát âm của một từ trong bộ huấn luyện. Mỗi một dòng trong tập tin là định nghĩa cách đọc của một từ.

Trong tiếng Việt, cách đọc và cách viết một từ gần như gắng liền với nhau. Không cần có hướng dẫn cách đọc khi học tiếng Việt, trong tiếng Anh cách đọc và cách viết không phụ thuộc nhau, ví dụ “lead” (dẫn đầu) & “head” (cái đầu).

Để xây dựng tập tin này cho tiếng Việt, ta có thể định nghĩa các từ bằng nhiều cách như sau [3]:

BAN B A N

Với cách trên, ta xem từ “BAN” là một âm tiết với sự kết hợp của 3 âm vị là B, A, N.

BAN B AN

Với cách này, ta xem từ “BAN” là một âm tiết với sự kết hợp của 2 âm vị là B, AN.

Sphinx không hỗ trợ định nghĩa ở dạng word-base, nghĩa là cách đọc của một từ không được chính là từ đó. Ví dụ: BAN BAN là không được cho phép. Tuy nhiên có thể làm một phương pháp tương đương thay thế nếu muốn xây dựng theo kiểu word-base. Khi đó phải định nghĩa từ theo kiểu một từ có nhiều cách đọc, ví dụ: BAN BAN BANG

Ý nghĩa của dòng định nghĩa trên là từ “ban” có thể đọc theo hai cách là “ban” (cách đọc đúng chuẩn) hoặc đọc là “bang” (cách đọc người miền Nam).

Lưu ý, chỉ được dùng các ký hiệu a-z, A-Z, 0-9 để đảm bảo không gây lỗi cho tập tin này.

**Vấn đề thanh điệu**

Chúng ta sẽ xem các âm vị đi chung với thanh điệu sẽ là một âm vị độc lập. Khi đó thay vì xem thanh điệu như một âm vị khác theo cách định nghĩa sau (định nghĩa cho từ “bản”):

BARN B A R N

Ta sẽ xem âm ả là một âm vị khác, độc lập với âm a khi đó ta định nghĩa như sau:

BARN B AR N

**Tập tin tuvung.phone**

Tập tin này chứa tất các các âm vị (phiên âm) sử dụng trong tập tin trên, mỗi một dòng là một âm vị, nên sắp xếp các âm vị đó theo thứ tự để Sphinx dễ quản lý. Lưu ý thêm một âm vị đặc biệt vào tập tin này đó là SIL. Đây là âm vị đại diện cho khoảng lặng.

**Tập tin tuvung.lm.DMP**

Tập tin này là mô hình ngôn ngữ thống kê được xây dựng từ trước bằng công cụ CMUclmk, định dạng ARPA hoặc DMP.

**Tập tin tuvung.filler**

Tập tin này chứa các âm tiết dùng để “làm đầy”, thông thường là các khoảng lặng, được định nghĩa như sau:

<s> SIL

</s> SIL

<sil> SIL

**Tập tin tuvung\_train.fileids**

Tập tin này là tập tin liệt kê đường dẫn đến các tập tin ghi âm trên mỗi dòng nằm trong thư mục train thuộc thư mục wav, có trong sơ đồ thư mục trình bày phía trên. Lưu ý, không chỉ định phần mở rộng .wav và mỗi dòng là một tập tin.

train/a11

train/a12

train/a13

train/a14

train/a15

train/a16

train/a17

train/a18

train/b11

train/b12

train/b13

train/b14

train/b15

train/b16

train/b17

train/b18

…

**Tập tin tuvung\_train.transcription**

Đây là phần nội dung mà tập tin âm thanh (.wav) đã thu âm được. Để huấn luyện cho Sphinx hiểu những gì chúng ta nói, cần cung cấp một tập tin dạng văn bản để giúp cho Sphinx hiểu và học từ đó. Cấu trúc một tập tin .transcript gồm nhiều dòng, mỗi một dòng là nội dung của một tập tin .wav kèm theo tên tập tin .wav đó.

<s> a </s> (a1)

<s> a </s> (a2)

<s> a </s> (a3)

<s> a </s> (a4)

<s> a </s> (a5)

<s> b </s> (b1)

<s> b </s> (b2)

<s> B </s> (b12)

<s> B </s> (b13)

<s> B </s> (b14)

<s> B </s> (b15)

<s> B </s> (b16)

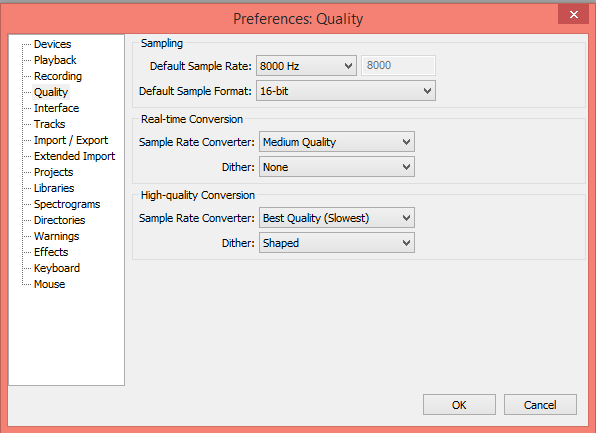
…

* + - 1. **Thiết lập định dạng âm thanh huấn luyện**

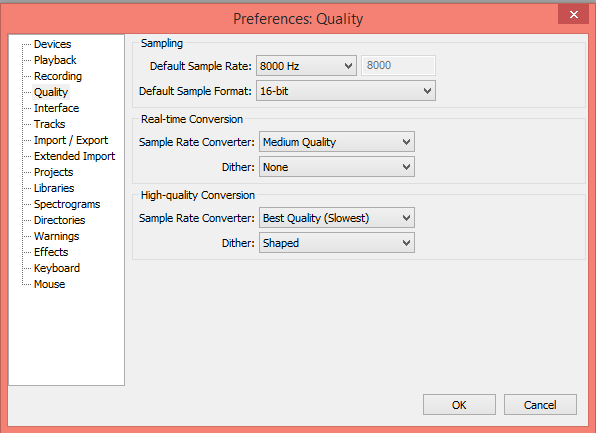
Dùng các chương trình ghi âm để ghi âm các từ cần huấn luyện. Âm thanh được ghi vào với các thông số sau [22]:

* Default Sample Rate Format: 8000Hz
* Default Sample Format: 16-bit
* Channels: 1(Mono)
* File Format: wav, raw hoặc sph

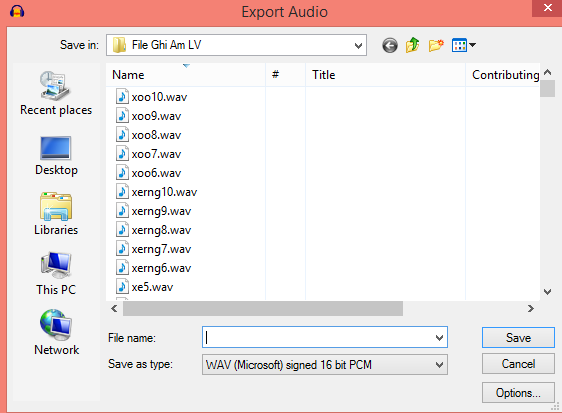
Luận văn sử dụng chương trình Audacity phiên bản 2.0.5



***Hình 1. 0.2 Thiết lập thông số Rate và Format.***



***Hình 1. 0.3 Thiết lập thông số về Channel***



***Hình 1. 0.4 Thiết lập định dạng lưu tập tin.***

* 1. **CẤU HÌNH SPHINX**
     1. **Cấu hình thư mục huấn luyện**

Sau khi đã cài đặt các gói cần thiết trong Windows, chúng ta chép thư mục **tuvung** đã tạo ở bước trên vào cùng thư mục chứa thư mục **sphinx** đã tạo trước đó [10].

Trước khi bắt đầu quá trình huấn luyện, chúng ta sử dụng các lệnh của sphinxtrain và pocketsphinx để cấu hình thư mục huấn luyện:

sphinxtrain -t tuvung setup

Trong đó **tuvung** là tên của thư mục huấn luyện. Lệnh trên sẽ sao chép hai tập tin cấu hình cần thiết vào thư mục etc của thư mục huấn luyện:

**Tập tin feat.params**

-nfilt \_\_CFG\_NUM\_FILT\_\_

-lowerf \_\_CFG\_LO\_FILT\_\_

-upperf \_\_CFG\_HI\_FILT\_\_

-feat \_\_CFG\_FEATURE\_\_

-svspec \_\_CFG\_SVSPEC\_\_

-agc \_\_CFG\_AGC\_\_

-cmn \_\_CFG\_CMN\_\_

-varnorm \_\_CFG\_VARNORM\_\_

**Tập tin sphinx\_train.cfg**

# Configuration script for sphinx trainer

-\*-mode:Perl-\*-

$CFG\_VERBOSE = 1;

# Determines how much goes to the screen.

# These are filled in at configuration time

$CFG\_DB\_NAME = "tuvung";

# Experiment name, will be used to name model files and log files

$CFG\_EXPTNAME = "$CFG\_DB\_NAME";

# Directory containing SphinxTrain binaries

$CFG\_BASE\_DIR = "/home/user/tuvung";

$CFG\_SPHINXTRAIN\_DIR = "/usr/local/lib/sphinxtrain";

$CFG\_BIN\_DIR = "/usr/local/libexec/sphinxtrain";

$CFG\_SCRIPT\_DIR = "/usr/local/lib/sphinxtrain/scripts";

…

# Audio waveform and feature file information

$CFG\_WAVFILES\_DIR = "$CFG\_BASE\_DIR/wav";

$CFG\_WAVFILE\_EXTENSION = 'wav';

$CFG\_WAVFILE\_TYPE = 'mswav'; # one of nist, mswav, raw

$CFG\_FEATFILES\_DIR = "$CFG\_BASE\_DIR/feat";

$CFG\_FEATFILE\_EXTENSION = 'mfc';

$CFG\_VECTOR\_LENGTH = 13;

* + 1. **Điều chỉnh tham số huấn luyện**

Thông tin cấu hình nằm trong tập tin **sphinx\_train.cfg**. Một số cấu hình quan trọng:

Cấu hình định dạng của tập tin âm thanh dùng trong huấn luyện

$CFG\_WAVFILES\_DIR = "$CFG\_BASE\_DIR/wav";

$CFG\_WAVFILE\_EXTENSION = 'wav';

$CFG\_WAVFILE\_TYPE = 'mswav'; # one of nist, mswav, raw

Điều chỉnh loại mô hình (huấn luyện HMM liên tục, bán liên tục), bỏ dấu # trước mô hình cần huấn luyện:

$CFG\_HMM\_TYPE = '.cont.'; # Sphinx 4, PocketSphinx

#$CFG\_HMM\_TYPE = '.semi.'; # PocketSphinx

#$CFG\_HMM\_TYPE = '.ptm.'; # PocketSphinx (larger data sets)

Cấu hình tham số mật độ CFG có thể nhận các giá trị 4, 8, 16, 32, 64 tùy theo độ lớn của dữ liệu:

elsif ($CFG\_HMM\_TYPE eq '.cont.') {

$CFG\_DIRLABEL = 'cont';

# Single stream features - Sphinx 3

$CFG\_FEATURE = "1s\_c\_d\_dd";

$CFG\_NUM\_STREAMS = 1;

$CFG\_INITIAL\_NUM\_DENSITIES = 1;

$CFG\_FINAL\_NUM\_DENSITIES = 8;

die "The initial has to be less than the final number of densities"

if ($CFG\_INITIAL\_NUM\_DENSITIES > $CFG\_FINAL\_NUM\_DENSITIES);

}

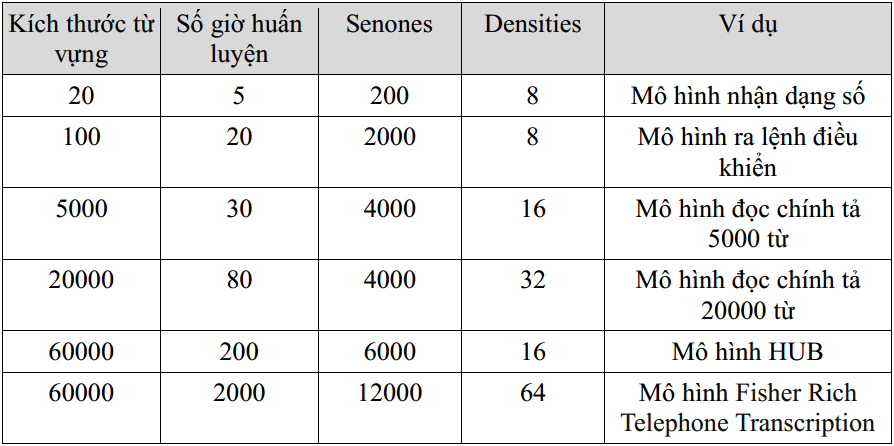
Cấu hình số lượng các senone để huấn luyện trong một mô hình. Số lượng senone càng lớn, sphinx phân biệt các âm càng chính xác. Nhưng mặt khác, nếu chúng ta có quá nhiều senone, mô hình sẽ không đủ tổng quát để nhận dạng các tiếng nói vô hình. Nghĩa là số từ lỗi sẽ tăng cao trên dữ liệu chưa huấn luyện. Đó là lý do quan trọng để không nên huấn luyện quá mức các mô hình. Trong trường hợp có quá nhiều senone mô hình sẽ phát sinh cảnh báo lỗi.

# Number of tied states (senones) to create in decision-tree clustering

$CFG\_N\_TIED\_STATES = 200;

Theo nghiên cứu của nhóm CMUSphinx thì cấu hình dựa theo bảng sau:

Bảng Thông số cấu hình.



* + 1. **Thực thi huấn luyện**

Để bắt đầu quá trình huấn luận, chúng ta thực hiện chuyển đến thư mục huấn luyện (thư mục tuvung)

Hình 2.1. Thực hiện chuyển đến thư mục huấn luyện tại Terminal.

Gõ lệnh huấn luyện: **sphinxtrain run**

Lệnh trên sẽ duyệt qua các phần yêu cầu, phát sinh các thư mục feat, logdir, model\_parameters, model\_architecture, result, trees và qmanager bên trong thư mục tuvung.

Trong các giai đoạn, giai đoạn quan trọng nhất là kiểm tra tất cả mọi thứ được cấu hình đúng và dữ liệu đầu vào của chúng ta là phù hợp. Đừng bỏ qua các thông báo lỗi trong quá trình huấn luyện. Quá trình huấn luyện sẽ xuất ra các thông báo dạng:

Baum welch starting for 2 Gaussian(s), iteration: 3 (1 of 1)

0% 10% 20% 30% 40% 50% 60% 70% 80% 90% 100%

Normalization for iteration: 3

Current Overall Likelihood Per Frame = 30.6558644286942

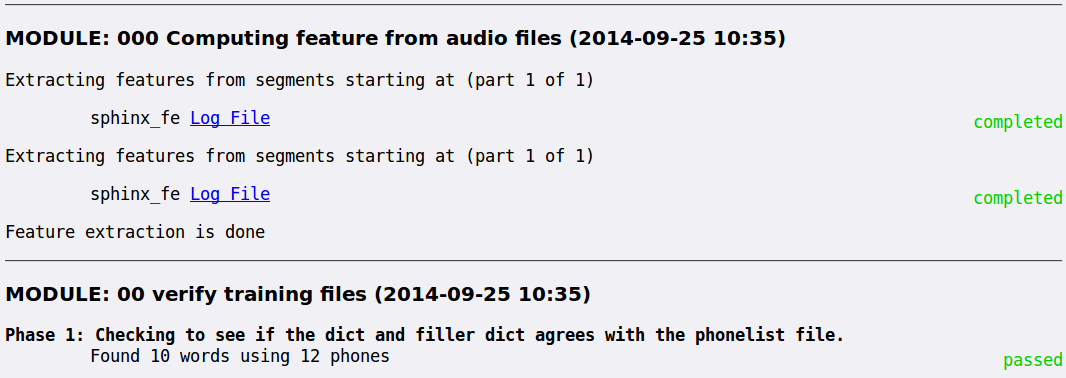
Convergence Ratio = 0.633864444461992

Baum welch starting for 2 Gaussian(s), iteration: 4 (1 of 1)

0% 10% 20% 30% 40% 50% 60% 70% 80% 90% 100%

Normalization for iteration: 4

Kết thúc quá trình huấn luyện, chúng ta thu được tập tin tuvung.html là tập tin ghi nhận quá trình huấn luyện.



***Hình 1. 0.5 Thông báo trạng thái trong quá trình huấn luyện***

Kết thúc quá trình huấn luyện, mô hình âm học (acoustic model) được lưu trong thư mục **model\_parameters/tuvung.cd\_cont\_200**, bao gồm các tập tin sau:

mdef

feat.params

mixture\_weights

means

noisedict

transition\_matrices

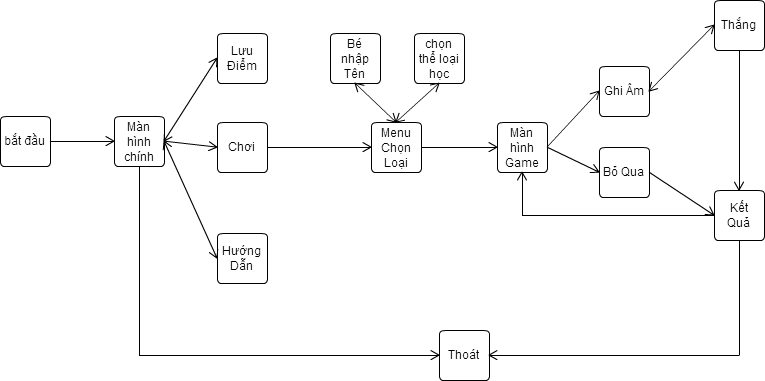
variances

* 1. **SỬ DỤNG KẾT QUẢ HUẤN LUYỆN**

### 3.4.1 Giới thiệu ứng dụng

Hiện tại trên thi trường có rất nhiều dòng game dành trẻ em, chủ yếu là dòng game xác nhận hình ảnh, giúp trẻ nhận biết, thông qua việc nhìn hình ảnh và âm thanh để xác nhận đối tượng. Ứng dụng demo mà chúng em thực hiện nhằm mang đến một hành động khác của trẻ em là giọng nói, giúp các em chủ động nói và nhận dạng vật thông qua phát âm từ đó. Ứng dụng chia ra nhiều cấp đi từ việc những con số đơn giản, cho đến những từ phức tạp. Giúp các em bập bẹ từng tiếng bước đầu trong quá trình tập nói của mình, Từ khóa nhận dạng đuộc ghi âm trực tiếp từ giọng nói của chúng em, nên việc có lỗi trong quá trình nhận dạng là không thể tránh khỏi.

### 3.4.2 Cấu trúc ứng dụng



Tổng quang các phần của màn hình:

+ Màn hình chính: bao gầm việc bắt đầu chơi, hướng dẫn lưu thông tin điểm

+ Màn hình chọn loại game: tại đây các bé thực hiện nhập để lưu lại thông tin, chọn loại chơi bao gồm có 3 cấp độ: học số, học từ, học ký tự

+ Màn hình chơi game: đầu tiên các bé chọn ghi âm, sau khi các bé đọc xong từ được hiển thị trên màn hình nhấn dừng và xác nhận từ bé dọc có chính xác không, nếu chính xác se có một bảng thông báo hiện ra chúc mừng bé đã hoàn thành từ một âm thanh cỗ vũ được phát lên, và sang từ mới. Khi bé đọc 3 lần không chính xác hệ thống sẽ tự động chuyển sang từ mới để không tạo cảm giác khó chịu cho bé. Ngoài ra bé cũng có thể chủ động nhấn nút bỏ qu để chuyển sang từ mới, thú vị hơn.

Trước khi chuyển qua từ mới, hệ thống sẽ có thông báo và đọc từ đúng cho bé nghe và có thang điểm được đánh dấu theo sao để tạo cảm giác hứng thú cho bé .

### 3.4.3 Hình ảnh demo

Sau đây hình ảnh demo ứng dụng



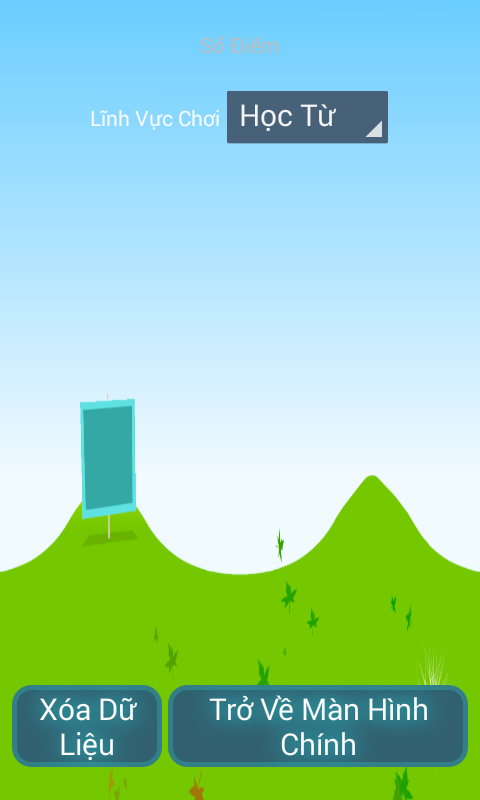
***Hình 1. 0.6 Màn hình chính khi vào Game***



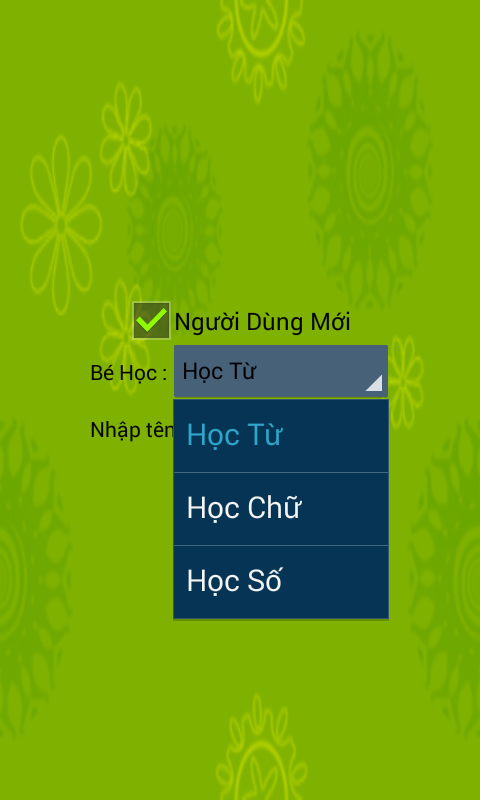
***Hình 1. 0.7 Màn hình hướng dẫn chơi***

Các nút để thực hiện hướng dẫn hoặc nhấn back

Để quay về màn hình chính và trải nghiệm



***Hình 1. 0.8 Màn hình lưu điểm của bé***



***Hình 1. 0.9 Menu chọn game***

tại đây các bé cần phải nhập tên của mình

Sau đó chọn loại mà bé muốn chơi

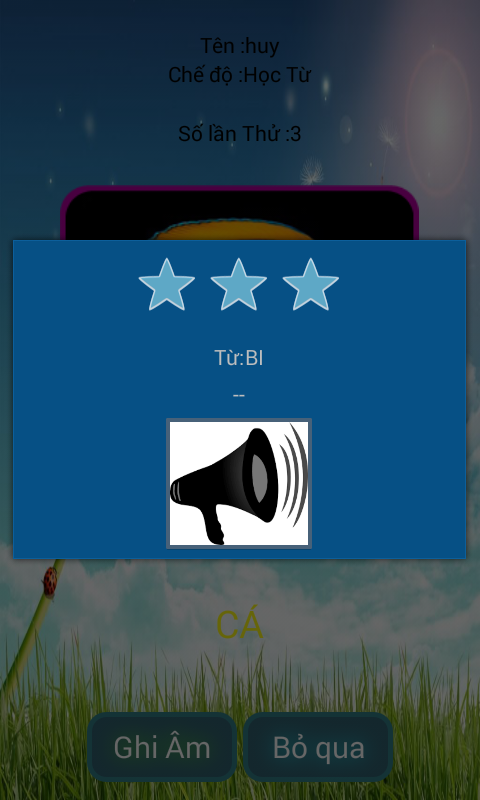
Bao gồm: học số, học chữ cái, học từ



***Hình 1. 0.10 Màn hình game chính***

bé chọn Ghi Âm sau đó phát âm

Từ được hiển thị và nhấn dừng



***Hình 1. 0.11 Màn hình khi bé đọc đúng từ***

khi bé phát âm đúng từ yêu cầu:

Hệ thống sẽ mở một âm thanh khuyến khích bé.

Sau đó hiện ra thông báo chấm điểm thông qua các sao

Bé có thể chạm vào loa để nghe lại giọng mình, nếu bé chọn bỏ qua

Thì khi nhấp vào hình loa sẽ là âm thanh đúng cần phát âm

* 1. **THỬ NGHIỆM VÀ ĐÁNH GIÁ KẾT QUẢ**

Đánh Giá Tỷ Lệ Chính Xác Của Nhận Dạng

|  |  |  |  |
| --- | --- | --- | --- |
| Người Nói | Nội Dung | Tỷ Lệ Chính Xác | Ghi Chú |
| Huỳnh Thanh Huy | Cá | 2/10 | Không Thực Hiện Nâng Cấp Khả Năng Nhận Dạng |
| Ly | 2/10 |
| Mũ | 1/10 |
| Nơ | 2/10 |
| Nồi | 1/10 |
| Ví | 2/10 |
| Xẻng | 1/10 |
| Xô | 1/10 |
| Dù | 1/10 |
| Ghế | 2/10 |
| Bi | 1/10 |
| Sách | 1/10 |
| Gạch | 1/10 |
| Xe | 1/10 |
| Mèo | 1/10 |
| Chuột | 1/10 |
| Bóng | 1/10 |
| Quần | 1/10 |
| Bò | 2/10 |
| Bướm | 1/10 |
| Cá | 2/10 |
| Cáo | 1/10 |
| Cò | 2/10 |
| Diều | 1/10 |
| Ếch | 1/10 |
| Gà | 1/10 |
| Hỗ | 1/10 |
| Nai | 1/10 |
| Ong | 1/10 |
| Rùa | 1/10 |
| Sâu | 1/10 |
| Sóc | 1/10 |
| Ve | 2/10 |
| Vẹt | 1/10 |
| Voi | 1/10 |
| Quà | 1/10 |
| Ngựa | 1/10 |
| Hoa | 2/10 |
| Bơ | 1/10 |
| Mận | 1/10 |
| Me | 2/10 |
| Quýt | 1/10 |
| Táo | 1/10 |
| Biển | 1/10 |
| Bọ | 1/10 |
| Cây | 1/10 |
| Chai | 2/10 |
| Rắn | 1/10 |
| Rồng | 1/10 |
| Đậu | 1/10 |
| Dép | 1/10 |
| Gạo | 1/10 |
| Guốc | 1/10 |
| Gỗ | 1/10 |
| Kéo | 1/10 |
| Lá | 1/10 |
| Lê | 1/10 |
| Lựu | 1/10 |
| Thùng | 2/10 |
| Trống | 1/10 |
| Túi | 1/10 |
| A | 2/10 |
| Ă | 1/10 |
| Â | 1/10 |
| B | 1/10 |
| C | 2/10 |
| D | 2/10 |
| Đ | 1/10 |
| E | 1/10 |
| Ê | 2/10 |
| G | 1/10 |
| H | 1/10 |
| I | 1/10 |
| K | 1/10 |
| L | 1/10 |
| M | 2/10 |
| N | 1/10 |
| O | 1/10 |
| Ô | 1/10 |
| Ơ | 1/10 |
| R | 1/10 |
| S | 1/10 |
| T | 1/10 |
| U | 2/10 |
| Ư | 1/10 |
| V | 1/10 |
| X | 1/10 |
| Y | 1/10 |
| Một | 8/10 | Tập Trung Nâng Cấp Khả Năng Nhận Dạng Các Từ Này |
| Hai | 9/10 |
| Ba | 9/10 |
| Bốn | 7/10 |
| Năm | 8/10 |
| Sáu | 8/10 |
| Bảy | 8/10 |
| Tám | 8/10 |
| Chín | 7/10 |
| Không | 8/10 |
|  |  |

Phân tích đánh giá ứng dụng demo bé học chữ số:

**Ưu điểm:**

+ Đã thực hiện và tích hợp thành công nhận dạng giọng nói tiếng việt không cần bất kỳ kết nối internet nào

+ Tích hợp thành công hệ thống nhận dạng lên thiết bị điện thoại

+ Ứng dụng tương đối ổn định về mặt nhận dạng

+ Tạo ra điểm nhấn mới cho dòng game dành cho trẻ em với cách thức học mới hơn

+ Xây dựng thành công ứng dụng tích hợp nhận dạng giọng nói.

**Nhược Điểm:**

+ Nhận dạng chưa thật chính xác, không thể thu thập quá nhiều giọng nói của các bé.

* 1. **KẾT LUẬN**

Sau khi tiến hành quá trình thu âm, huấn luyện, thử nghiệm và đánh giá kết quả, chúng em đã đúc kết được các kinh nghiệm sau:

Quá trình thu âm để tạo bộ dữ liệu huấn luyện ảnh hưởng rất lớn đến kết quả nhận dạng. Dữ liệu huấn luyện nên thu âm ở nhiều trạng thái như: phát âm to, phát âm nhỏ, phát âm nhanh và phát âm chậm. Ví dụ, khi tiến hành thu âm ký tự “A”, chúng em đã thu âm chữ “A” lúc nói nhỏ, lúc nói to, lúc nói nhanh và lúc nói chậm trên cùng một ký tự.

Đối với hệ thống nhận dạng phụ thuộc vào người huấn luyện thì số lần thu âm tối thiểu cần 10 lần/từ.

**3.6.1 Kết quả đạt được**

Qua quá trình nghiên cứu về nhận dạng giọng nói tiếng Việt và ứng dụng thử nghiệm , luận văn đã làm được một số công việc sau:

* Tìm hiểu về tiếng nói, các phương pháp xử lý tiếng nói, rút trích đặc trưng.
* Tìm hiểu và thực hiện huấn luyện mô hình âm học theo âm vị, áp dụng cho tiếng Việt.
* Tìm hiểu kiến trúc một hệ thống nhận dạng tiếng nói qua các công cụ của CMUSphinx.
* Xây dựng thành công chương trình thử nghiệm bằng giọng nói.

Do chưa có nhiều kiến thức về xử lý tiếng nói nên luận văn không tránh khỏi nhiều thiếu sót. Tuy nhiên, với một số kết quả đã đạt được hy vọng luận văn sẽ góp một phần nhỏ vào việc nghiên cứu nhận dạng tiếng nói tiếng Việt và ứng dụng nó trong việc phát triển nhận dạng giọng nói trên thiết bị di động.

**3.6.2 Hướng phát triển**

Do việc thu âm xử lý dữ liệu chưa được phong phú nên kết quả nhận dạng chưa được tốt. Việc này có thể được khắc phục bằng cách thu nhiều mẫu hơn, huy động thêm những người tình nguyện để thu âm, có thể thu âm theo từng vùng miền (Bắc, Trung, Nam), thu âm theo từng độ tuổi, thu âm theo giới tính,..

# TÀI LIỆU THAM KHẢO

**Tiếng Việt:**

[1]. Vspeech: <http://bk02.sourceforge.net/vspeech/features.htm>.

[2]. Huỳnh Thanh Giàu (2012), *Nghiên cứu về nhận dạng tiếng nói tiếng Việt và ứng dụng thử nghiệm trong điều khiển máy tính,* Luận văn Thạc sĩ, Đại học Lạc Hồng.

[3]. Võ Văn Hoà, Tôn Thanh Hùng (2013), *Nghiên cứu về nhận dạng giọng nói tiếng Việt và ứng dụng trong điều khiển*, Đồ án Đại học, Đại học Công nghệ Thông tin TP.HCM.

[4]. Trần Việt Khải, Bồ Xuần Tú (2008), *Sphinx4 trong nhận dạng các chữ số tiếng Việt từ 0 đến 9*, Đại học quốc gia TP.Hồ Chí Minh.

[5]. Nguyễn Hồng Quang (2004), *Nhận dạng tiếng nói tiếng Việt – Tìm hiểu và ứng dụng*, Đồ án Đại học, Đại học KHTN TP.HCM.

[6]. Nguyễn Cao Quí (2013), *Ứng dụng mô hình Markov ẩn để nhận dạng tiếng nói trên FPGA*, Tạp chí khoa học, Đại học Cần Thơ.

**Tiếng Anh:**

[7]. Installing Apache Ant: <https://ant.apache.org/manual/install.html#windows>.

[8]. B.H. Juang and L.R. Rabiner (1991), *Hidden Markov Models for Speech Recognition*, Speech Research Department, AT&T Bell Laboratories.

[9]. Sphinx-4 Documentation: <http://cmusphinx.sourceforge.net/wiki/tutorialsphinx4/>.

**[**10]. Keyboard control :<http://www.codeproject.com/Articles/28064/Global-Mouse-and-Keyboard-Library>.

[11]. Oracle documentation: <http://docs.oracle.com/javase/7/docs/api/java/awt/Robot.html>.

[12]. Digital audio: <<http://en.wikipedia.org/wiki/Digital_audio>>.

[13]. Vietnamese alphabet: <<http://en.wikipedia.org/wiki/Vietnamese_alphabet>>.

[14]. Using sphinx train models: <http://www.exocomm.com/solutions/software/open-source/audio/sphinx/sphinx4/doc/UsingSphinxTrainModels.html>.

[15]. Keyboard virtual: <http://help.adobe.com/en\_US/AS2LCR/Flash\_10.0/help.html?content=00000520.html>.

[16]. Sample Code:<http://www.java-tips.org/java-se-tips/java.awt/how-to-use-robot-class-in-java.html>.

[17]. JeffBilmes (2002), *What HMMs Can Do*, Dept of EE , University of Washington.

[18]. Lawrence Rabiner, Biing Hwang Juang (1993), *Fundamental of Speech Recognition*, ISBN 0-13-285826-6.

[19]. Lawrence R. Rabiner (1989), *A Tutorial on Hidden Markov Models and Selected Application in Speech Recognition*, Proceedings of the IEEE, Vol.77, No.2.

[20]. Speech recognizer: <https://puneetk.com/speech-recognizer-in-java-tutorial**>.**

[21]. Record your Speech with Audacity: <http://www.voxforge.org/home/submitspeech/windows/step-2>.

[22] Download news version CMUSphinx.

http://cmusphinx.sourceforge.net/wiki/download/

[23] Training Acoustic Model For CMUSphinx

http://cmusphinx.sourceforge.net/wiki/tutorialam

[24] A speech recognizer written entirely in the JavaTM programming language

http://cmusphinx.sourceforge.net/sphinx4/

[25] Sphinx Knowledge Base Tool -- VERSION 3

http://www.speech.cs.cmu.edu/tools/lmtool-new.html

[26] Download Voxforge

http://www.voxforge.org/home/downloads

[27] Sphinx-4 Application Programmer's Guide

http://cmusphinx.sourceforge.net/wiki/tutorialsphinx4

[28] Configuration Management for Sphinx-4

http://cmusphinx.sourceforge.net/sphinx4/javadoc/edu/cmu/sphinx/util/

props/doc-files/ConfigurationManagement.html

[29] How to use SphinxTrain

http://ronaldramdhan.wordpress.com/2010/03/11/sphinxtrain/

[30] Creating a text corpus from Wikipedia

http://trulymadlywordly.blogspot.ru/2011/03/creating-text-corpusfrom-wikipedia.html

[31] The CMU Statistical Language Modeling (SLM) Toolkit

http://www.speech.cs.cmu.edu/SLM\_info.html

[32] SPEECH and LANGUAGE PROCESSING

http://www.cs.colorado.edu/~martin/slp.html

[33] Sphinx Knowledge Base Tool

http://www.speech.cs.cmu.edu/tools/lmtool-adv.html

[34] The CMU Audio Databases

http://www.speech.cs.cmu.edu/databases/

[35] festvox

http://www.festvox.org/index.html

[37] Constant Field Values

http://cmusphinx.sourceforge.net/sphinx4/javadoc/constantvalues.html#edu.cmu.sphinx.jsgf.JSGFGrammar.PROP\_BASE\_GRA

[38] LOGIOS Lexicon Tool

http://www.speech.cs.cmu.edu/tools/lextool.html

[39] Adapting the default acoustic model

http://cmusphinx.sourceforge.net/wiki/tutorialadapt

[40] The CMU Pronouncing Dictionary

<http://www.speech.cs.cmu.edu/cgi-bin/cmudict>

## PHỤ LỤC

**BẢNG PHIÊN ÂM TIẾNG VIỆT DƯỚI DẠNG MÃ ASCII**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| STT | Âm vị | | Chữ | Ví dụ | | Mô tả | |
| IPA | ASCII |  |  | |  | |
| **Âm đầu**  Âm đâu | | | | | | | |
| 1 | b | b | b | | ba | | phụ âm tắc, hai môi, hữu thanh, không bật hơi, chỉ xuất hiện trong âm tiết không có âm đệm. |
| 2 | d | dd | đ | | đẩy | | phụ âm tắc, đầu lưỡi lợi, hữu thanh, không bật hơi |
| 3 | t | t | t | | tùng | | phụ âm tắc, đầu lưỡi răng, vô thanh, không bật hơi. |
| 4 | t’ | th | th | | thích | | phụ âm tắc, vô thanh, bật hơi, đầu lưỡi răng. |
| 5 | ~~J~~ | tr | tr | | trăng | | phụ âm tắc, đầu lưỡi vòm miệng, vô thanh, không bật hơi. |
| 6 | c | ch | ch | | chỉ | | phụ âm tắc, vô thanh, mặt lưỡi, không bật hơi. |
| 7 | k | k | k (trước i, e, ê) | | keo | | phụ âm tắc, vô thanh, gốc lưỡi, không bật hơi. |
| c (trước u, ư, a, o,...) | | cảnh | |
| q (trước u) | | quậy | |
| 8 | m | m | m | | mềm | | phụ âm vang mũi, hai môi, xuất hiện trong âm tiết không có âm đệm. |
| 9 | n | n | n | | nóng | | phụ âm vang mũi, đầu lưỡi lợi. |
| 10 |  | nh | nh | | nhà | | phụ âm vang mũi, mặt lưỡi. |
| 11 |  | ng | ng (trước u, ư, o, ô, ơ, a, ă, â) | | ngủ | | phụ âm vang mũi, gốc lưỡi. |
| ngh (trước i, e, ê) | | nghỉ | |
| 12 | f | f | ph | | phê | | phụ âm xát, vô thanh, môi răng, xuất hiện trong âm tiết không có âm đệm. |
| 13 | v | v | v | | vội | | phụ âm xá, hữu thanh, môi răng, xuất hiện trong âm tiết không có âm đệm. |
| 14 | s | x | x | | xa | | phụ âm xát, vô thanh, đầu lưỡi lợi. |
| 15 | z | d | d | | dễ | | phụ âm xát, hữu thanh, đầu lưỡi lợi. |
| gi | | giỏi | |
| g (trước i) | | gì | |
| 16 | l | l | l | | lắm | | phụ âm vang bên, đầu lưỡi răng. |
| 17 | § | s | s | | sơn | | phụ âm xát, vô thanh, dầu lưỡi vòm miệng, uốn lưỡi |
| 18 |  | r | r | | rằm | | phụ âm xát, hữu thanh, đầu lưỡi vòng miệng, uốn lưỡi. |
| 19 | X | kh | kh | | khá | | phụ âm xát, vô thanh, gốc lưỡi. |
| 20 | Y | g | g (trước u, ư, o, ô, ơ, a, ă, â) | | găm | | phụ âm xát cuối lưỡi, hữu thanh. |
| gh (trước i, e, ê) | | ghế | |
| 21 | h | h | h | | hòa | | Phụ âm xát, vô thanh, họng. |
| 22 | p | p | p | | pi | | phụ âm tắc, hai môi, ... |
| 23 | w | w | o (trước nguyên âm rộng a, ă, e) | | hoa | | có cấu tạo giống nguyên âm chính /u/, có độ mở hẹp, phát âm cực trầm, tròn môi, thuộc hàng sau. |
| u (còn lại) | | hủy | |
| **Âm chính**  Am chính | | | | | | | |
| 24 | i | i | y (đứng sau u) | suy | | nguyên âm đơn dài, hàng trước, hẹp, không tròn môi, có tính bổng, trước /k,/ bị rút ngắn. | |
| i (còn lại) | tính | |
| 25 | e | ee | ê | chê | | nguyên âm đơn, dài, hàng trước, hơi hẹp, không trong môi, có tính chất bổng, trước /k, / bị rút ngắn. | |
| 26 |  | e | e | chè | | nguyên âm đơn, dài, hàng trước, hơi rộng, không tròn môi, có tính chất bổng. | |
| 27 |  | ea | a (trước ch, nh) | sách | | nguyên âm đơn, ngắn. Gần như là thể ngắn của . | |
| 28 | u | u | u | sung | | nguyên âm đơn, dài, hàng sau, hẹp, tròn môi, có âm sắc trầm. Đứng trước /k, / bị rút ngắn | |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Tên thanh điệu | Sắc | Huyền | Hỏi | Ngã | Nặng |
| Ký hiệu | S | F | R | X | J |