TRƯỜNG ĐẠI HỌC BÁCH KHOA HÀ NỘI
VIỆN CÔNG NGHỆ THÔNG TIN VÀ TRUYỀN THÔNG

———————— * ————————

# ĐỒ ÁN
# TỐT NGHIỆP ĐẠI HỌC

## NGÀNH CÔNG NGHỆ THÔNG TIN

# XÂY DỰNG HỆ THỐNG DỊCH TIẾNG NÓI HAI CHIỀU ANH VIỆT TRÊN THIẾT BỊ DI ĐỘNG

Sinh viên thực hiện : **Nguyễn Văn Thịnh**
Lớp KSCLC HTTT&TT – K57
Giáo viên hướng dẫn: TS. **Đỗ Thị Ngọc Diệp**

HÀ NỘI 5-2017

# PHIẾU GIAO NHIỆM VỤ ĐỒ ÁN TỐT NGHIỆP

1. Thông tin về sinh viên
Họ và tên sinh viên: Nguyễn Văn Thịnh
Điện thoại liên lạc: 01626562138 Email: thinhnv1811@gmail.com
Lớp: KSCLC Hệ thống thông tin & TT K57 Hệ đào tạo: KSCLC-TN-TT
Đồ án tốt nghiệp được thực hiện tại: Viện nghiên cứu quốc tế MICA – Trường Đại Học Bách Khoa Hà Nội.
Thời gian làm ĐATN: Từ ngày 15/02/2017 đến 29/05/2017

2. Mục đích nội dung của ĐATN
Xây dựng hệ thống dịch tiếng nói tiếng nói hai chiều Anh-Việt trên thiết bị di động.

3. Các nhiệm vụ cụ thể của ĐATN
Các nhiệm vụ cụ thể của đồ án:
- Tìm hiểu về dịch tiếng nói, các thành phần trong hệ thống dịch tiếng nói.
- Đề xuất mô hình, thiết kế hệ thống dịch tiếng nói với cặp ngôn ngữ Anh - Việt trên thiết bị di động.
- Nghiên cứu xây dựng mô đun truyền nhận dữ liệu cho pha nhận dạng tiếng nói.
- Triển khai hệ thống hoàn thiện dựa trên mô hình ghép nối ba mô đun nhận dạng, dịch và tổng hợp cho cặp ngôn ngữ Anh-Việt.
- Thử nghiệm, đánh giá hệ thống.

4. Lời cam đoan của sinh viên:
Tôi – *Nguyễn Văn Thịnh* - cam kết ĐATN là công trình nghiên cứu của bản thân tôi dưới sự hướng dẫn của *TS. Đỗ Thị Ngọc Diệp*
Các kết quả nêu trong ĐATN là trung thực, không phải là sao chép toàn văn của bất kỳ công trình nào khác.

*Hà Nội, ngày 29 tháng 05 năm 2017*
Tác giả ĐATN

*Nguyễn Văn Thịnh*

5. Xác nhận của giáo viên hướng dẫn về mức độ hoàn thành của ĐATN và cho phép bảo vệ:

*Hà Nội, ngày tháng năm*
Giáo viên hướng dẫn

*TS. Đỗ Thị Ngọc Diệp*

# TÓM TẮT NỘI DUNG ĐỒ ÁN TỐT NGHIỆP

Ngày nay với sự phát triển của khoa học kỹ thuật, khoảng cách về địa lý giữa con người càng ngày càng được thu hẹp, con người ngày càng hội nhập và giao lưu với nhau nhiều hơn. Nhưng sự khác biệt về ngôn ngữ vẫn còn là một rào cản lớn ngăn cách sự hội nhập đặc biệt là với Việt Nam, một quốc gia còn đang phát triển, trình độ khoa học công nghệ và giáo dục còn ở mức thấp. Chính vì vậy lĩnh vực nghiên cứu Dịch Tiếng Nói áp dụng cho cặp ngôn ngữ Anh-Việt đang ngày càng trở nên cấp thiết. Nhiều công trình nghiên cứu, sản phẩm đã được phát triển trên thế giới như các sản phẩm của Google, Microsoft. Nhưng trong những dự án yêu cầu tính bảo mật thông tin không muốn chia sẻ ra bên ngoài ví dụ ứng dụng trong quốc phòng an ninh, hay trong việc triển khai xây dựng những sản phẩm mà đòi hỏi phải thay đổi hệ thống dịch theo cách riêng để phù hợp với dự án đặc thù, ta cần phải hiểu sâu về hệ thống và nắm vững được công nghệ. Trong công trình này chúng tôi đã triển khai xây dựng hệ thống dịch tiếng nói trên nền tảng di động, hệ thống được xây dựng dựa trên kiến trúc cơ bản bao gồm ba mô đun chính là: Nhận dạng tiếng nói, Dịch tự động và Tổng hợp tiếng nói.

Trong bài toán xây dựng hệ thống dịch tiếng nói có ba vấn đề chính cần phải giải quyết đó là: Xây dựng ba thành phần nhận dạng, dịch và tổng hợp, vấn đề thứ hai là xây dựng các bộ dữ liệu do từng mô đun, vấn đề cuối cùng là triển khai hệ thống kết nối cả ba thành phần theo mô hình phù hợp. Việc xây dựng một hệ thống bao gồm cả ba thành phần nêu trên cùng đặt trên một thiết bị di động sẽ khiến hệ thống hoạt động chậm và có thể bị treo do đó chúng tôi đề xuất xây dựng hệ thống trên cơ sở mô hình client-sever. Việc triển khai trên mô hình client-sever cũng gây ra một vấn đề là về thời gian truyền tin qua mạng Internet, nếu truyền trực tiếp tệp tin tiếng nói để nhận dạng có thể sẽ tốn nhiều thời gian, dung lượng truyền do đó chúng tôi lựa chọn giải pháp truyền các đặc trưng của tập tin tiếng nói, điều này làm giảm đáng kể thời gian truyền và nhận dữ liệu. Vấn đề cuối cùng là về thu thập dữ liệu huấn luyện cho từng mô đun: mô đun nhận dạng chúng tôi sử dụng bộ dữ liệu từ TED-LIUM với khoảng 100 giờ ghi âm để huấn luyện mô hình cho tiếng Anh và dữ liệu cho tiếng Việt sử dụng bộ dữ liệu VNSpeechCorpus (20 giờ) của Viện nghiên cứu MICA, dữ liệu cho mô đun dịch tự động sử dụng bộ dữ liệu 3,8 triệu cặp câu song ngữ Anh Việt, về tổng hợp tiếng nói chúng tôi sử dụng công cụ hỗ trợ của Google được tích hợp sẵn trên các thiết bị di động chạy hệ điều hành Android cho tổng hợp tiếng Anh và cho tổng hợp tiếng Việt sử dụng bộ công cụ tổng hợp tiếng nói được nghiên cứu và phát triển tại viện nghiên cứu quốc tế MICA.

Hệ thống đã được triển khai thử nghiệm trên các thiết bị di động sử dụng hệ điều hành Android. Kết quả đánh giá thử nghiệm trên một số bộ dữ liệu test cho từng mô đun của hệ thống: với Module nhận dạng cho tiếng anh có tỷ lệ từ lỗi WER là 28% và tiếng việt có tỷ lệ từ lỗi WER là 11,2%. Với mô đun dịch cho điểm số BLEU của chiều Anh-Việt là 46,7 và chiều Việt-Anh là 46,61.

# LỜI CẢM ƠN

Đầu tiên, tôi xin được gửi lời cảm ơn chân thành tới Viện nghiên cứu quốc tế MICA nơi đã tạo điều kiện cho tôi thực tập, nghiên cứ và làm đồ án tốt nghiệp. Tiếp đó, tôi xin cảm ơn TS. Đỗ Thị Ngọc Diệp, người đã tận tình hướng dẫn và giúp đỡ tôi trong suốt những năm tháng thực tập, nghiên cứu và hoàn thành đồ án này tại Viện nghiên cứu quốc tế MICA. Tôi cũng xin gửi lời cảm ơn chân thành đến TS. Mạc Đăng Khoa người đã tận tình giúp đỡ và hướng dẫn tôi trong suốt quãng thời gian thực hiện đồ án này.

Thêm vào đó tôi cũng muốn gửi lời cảm ơn đến anh Nguyễn Tiến Thành, chị Nguyễn Hằng Phương cùng toàn bộ các cán bộ tại viện nghiên cứu quốc tế MICA đã giúp đỡ tôi trong quá trình thực tập tại Viện nghiên cứu quốc tế MICA.

Cuối cùng tôi xin gửi lời cảm ơn chân thành tới người thân, gia đình và bạn bè những người đã ủng hộ, hỗ trợ tôi trong quá trình học tập.

Hà Nội, ngày 29 tháng 05 năm 2017

Nguyễn Văn Thịnh

# MỤC LỤC

# DANH MỤC HÌNH ẢNH

# DANH MỤC BẢNG

# DANH MỤC TỪ VIẾT TẮT VÀ THUẬT NGỮ

| Từ viết tắt | Từ đầy đủ | Ý nghĩa. |
|---|---|---|
| ASR | Automatic Speech Recognition | Nhận dang tiếng nói tự động. |
| MT | Machine Translation | Dịch máy. |
| SMT | Statistical Machine Translation | Dịch máy thống kê. |
| TTS | Text To Speech | Tổng hợp văn bản thành tiếng nói. |
| ITU | International Telecommunication Union | Tổ chức viễn thông quốc tế. |
| ATR | Advanced Telecommunications Research Institute International ATR | Viện nghiên cứu truyền thông nâng cao quốc tế. |
| MASTOR | Multilingual Automatic Speech To Speech Translator | Hệ thống dịch tiếng nói tự động đa ngôn ngữ. |
| A-STAR | Asian Speech Translation Advanced Research | Tên của tổ chức nghiên cứu dịch tiếng nói châu á. |
| U-STAR | Universal Speech Translation Advanced Research | Tên của tổ chức nghiên cứu dịch tiếng nói. |
| STML | Speech Translation Markup Language | Một giao thức truyền tin giữa các bộ phận trong hệ thống dịch tiếng nói trên cơ sở ba thành phần ASR,MT,TTS. |
| HMM | Hidden Markov Model | Mô hình Markov ẩn. |
| MFCCs | Mel-frequency cepstral coefficients | Các hệ số đặc trưng của tiếng nói. |
| MIT | Massachusetts Institute of Technology | Học viện công nghệ Massachusetts. |
| API | Application programming interface | Giao diện lập trình ứng dụng. |
| Alignment | | Dóng hàng, thể hiện sự tương quan vị trí giữa các từ trong câu ở ngôn ngữ nguồn và vị trí các từ là bản dịch của nó trong câu đích. |
| Noisy channel | | Phương pháp kênh nhiễu. |
| Formant | | Tần số cộng hưởng của tuyến âm. |
| Frame | | Khung tín hiệu. |
| Client-Sever | | Mô hình chủ khách. |

# MỞ ĐẦU

Sự khác biệt về ngôn ngữ là một trong những nguyên nhân cản trở sự phát triển của quá trình toàn cầu hóa, du lịch và kinh doanh trên toàn cầu. Công nghệ dịch tiếng nói được phát triển nhằm mục đích phá vỡ rào cản ngôn ngữ, đem lại thị trường kinh doanh, du lịch toàn cầu, thúc đẩy phát triển thương mại và giao lưu văn hóa. Công nghệ dịch tiếng nói đem lại khả năng tự động dịch lời nói của một người từ ngôn ngữ này sang ngôn ngữ khác, nó được bình chọn là một trong mười công nghệ sẽ đem lại sự thay đổi cho thế giới [1]. Công nghệ nhận dạng tiếng nói tự động bao gồm ba công nghệ riêng biệt là: nhận dạng tiếng nói tự động, dịch tự động và tổng hợp tiếng nói. Trước đây ba công nghệ này được phát triển một cách riêng biệt cho đến khi việc nghiên cứu dịch tiếng nói được đề xuất và bắt đầu phát triển trong những năm 1980. Hiện nay có một số hướng nghiên cứu khác phát triển theo hướng dịch trực tiếp từ dữ liệu âm thanh [2]. Thế nhưng công nghệ dịch tiếng nói dựa trên ba công nghệ nhận dạng tiếng nói, tổng hợp tiếng nói và dịch tiếng nói vẫn là hướng nghiên cứu phổ biến rộng rãi nhất.

Đề tài này tập trung phát triển hệ thống dịch tiếng nói dựa trên mô hình ghép nối ba mô đun nhận dạng tiếng nói, dịch tự động và tổng hợp tiếng nói. Nhưng nếu triển khai cả ba mô đun nói trên tại cùng một thiết bị sẽ gây tốn kém tài nguyên thiết bị do đó hệ thống sẽ được triển khai theo mô hình client-sever. Khi triển khai hệ thống theo mô hình client-sever nếu truyền toàn bộ dữ liệu âm thanh từ client tới sever cho pha nhận dạng thì sẽ gây tốn kém thời gian và dung lượng dữ liệu do đó việc truyền các đặc trưng của tiếng nói từ client tới sever được đề xuất.

Để giải quyết vấn đề nêu trên, tác giả đã đề ra ba nhiệm vụ chính cần hoàn thành trong đồ án này đó là:
- Xây dựng hệ thống dịch tiếng nói dựa trên mô hình client-sever.
- Tiến hành truyền các đặc trưng âm học của tệp tin tiếng nói từ client tới sever, thay cho việc truyền dữ liệu tiếng nói.
- Triển khai hệ thống hoàn thiện dựa trên mô hình ghép nối ba mô đun và thu thập dữ liệu tiếng việt.

Đồ án được thực hiện trong quá trình làm việc tại Phòng Giao Tiếp Tiếng Nói, Viện nghiên cứu quốc tế MICA, Đại Học Bách Khoa Hà Nội. Với môi trường nghiên cứu thuận lợi, nghiêm túc và với sự hướng dẫn tận tình của TS. Đỗ Thị Ngọc Diệp và TS. Mạc Đăng Khoa tôi đã thu được những kiến thức, kinh nghiệm quý báu để hoàn thành xây dựng hệ thống. Trong hệ thống này tôi cũng đã sử dụng các kết quả nghiên cứu đã được hoàn thành trước đó của Viện nghiên cứu MICA như bộ dữ liệu cho việc huấn luyện mô hình nhận dạng tiếng Việt VNSpeechCorpus và công cụ tổng hợp tiếng nói tiếng Việt trên hệ điều hành Android.

Bố cục của đồ án bao gồm cách phần sau:
**Chương 1: Giới thiệu tổng quan về dịch tiếng nói và vấn đề đặt ra với đồ án**. Chương này nói về tổng quan vấn đề dịch tiếng nói, các mô hình dịch tiếng nói, cơ sở lý thuyết về dịch tiếng nói, nhận dạng tiếng nói, dịch tự động, tổng hợp tiếng nói và đề ra các giải pháp với các vấn đề gặp phải khi xây dựng và hoàn thiện hệ thống.

**Chương 2: Xây dựng mô đun nhận dạng tiếng nói theo mô hình client-sever truyền nhận tham số đặc trưng**. Trong chương này nói về kiến trúc và cách xây dựng mô đun nhận dạng tiếng nói theo mô hình client-sever dựa trên việc truyền các đặc trưng.

**Chương 3: Xây dựng bộ cơ sở dữ liệu cho hệ thống dịch tự động tiếng nói**. Chương này nói về các chuẩn bị và xây dựng cơ sở dữ liệu cho các mô đun dịch tự động và nhận dạng tiếng nói.

**Chương 4: Triển khai xây dựng hệ thống dịch tiếng nói**. Chương này nói về cách thức triển khai xây dựng hệ thống dịch tiếng nói, cách ghép nối các mô đun.

**Chương 5: Thử nghiệm và đánh giá hoạt động của hệ thống.** Trình bày quá trình cài đặt và đánh giá hệ thống.

**Chương 6: Kết luận**. Tổng kết lại kết quả đã đạt được trong đồ án, những điểm còn hạn chế và nêu định hướng phát triển trong tương lai.

# CHƯƠNG 1: TỔNG QUAN VỀ DỊCH TIẾNG NÓI VÀ VẤN ĐỀ ĐẶT RA VỚI ĐỒ ÁN

## 1.1 Giới thiệu về dịch tiếng nói

### 1.1.1 Lịch sử

Dịch tiếng nói được giới thiệu lần đầu tiên tại triển lãm ITU Telecom World 1983, khi mà công ty NEC đã đưa ra một phiên bản thử nghiệm của một hệ thống dịch tiếng nói tự động. Đó là phiên bản thử nghiệm đầu tiên của một hệ thống dịch tiếng nói thời gian thực, giữa một cặp ngôn ngữ nhất định trong ba ngôn ngữ tiếng Anh, tiếng Nhật và tiếng Bồ Đào Nha [3]. Đến năm 1993 thì một cuộc thử nghiệm đã dẫn tới việc liên kết của ba nơi trên thế giới đang nghiên cứu về dịch tiếng nói đó là ATR, CMU (Carnegie Melon University) và Siemens. Sau khi bắt đầu dự án của ATR thì các dự án về dịch tiếng nói khác trên thế giới cũng được bắt đầu. Ở Đức có dự án Vermobil, ở Mỹ cũng bắt đầu với hai dự án TransTac và GALE, mục tiêu của dự án Gale là tạo ra một hệ thống có khả năng tự động thu thập những thông tin đa ngôn ngữ từ các bản tin ở các ngôn ngữ khác nhau, văn bản, các hình thức giao tiếp khác. Ngược lại mục tiêu của ATR là tạo ra hệ thống có thể dịch trực tiếp tiếng nói tiếng nói thời gian thực.

### 1.1.2 Tình hình phát triển

Hiện tại việc nghiên cứu và phát triển công nghệ dịch tiếng nói đang phát triển và mở rộng ở nhiều nơi, nhiều công ty và viện nghiên cứu trên thế giới. Như đã nói ở trên hiện tại hướng phát triển chính cho dịch tiếng nói vẫn là dựa trên ba công nghệ con bao gồm dịch tự động, nhận dạng tiếng nói và tổng hợp tiếng nói. Tại viện nghiên cứu công nghệ thông tin và truyền thông quốc tế NICT đang nghiên cứu và phát triển hệ thống có khả năng dịch tiếng nói đa ngôn ngữ cho điện thoại thông minh [4]. Tại Microsoft đang tập trung nghiên cứu và phát triển công nghệ mạng nơ ron trong các sản phẩm dịch tiếng nói của mình [5]. Nhóm nghiên cứu thuộc bộ phận kỹ thuật và khoa học máy tính trường đại học GOTHENBURG đang nghiên cứu phát triển hệ thống dịch tiếng nói sử dụng phương pháp học máy học sâu (deep learing) và dịch trực tiếp từ dữ liệu tiếng nói sang tiếng nói mà không qua biểu diễn dạng văn bản [6]. Hướng nghiên cứu dịch trực tiếp từ tiếng nói sang văn bản cũng đang được phát triển [7].

Hiện nay các công cụ nền tảng cho phát triển hệ thống dịch tự động tiếng nói đang ngày càng mạnh mẽ. Bộ công cụ cho mô đun nhận dạng có thể kể đến như các bộ công cụ mã nguồn mở của CMU SphinX hay Kaldi… , bộ công cụ CMU Sphinx có các công cụ hỗ trợ huấn luyện mô hình và các thư viện hỗ trợ lập trình. Bộ công cụ cho mô đun dịch tự động có Moses, Moses là một hệ thống dịch máy thống kê mã nguồn mở cho cho phép bạn huấn luyện mô hình dịch máy, sau đó sử dụng thuật toán tìm kiếm được tích hợp trong Moses để tìm bản dịch có xác xuất xảy ra cao nhất. Với mô đun tổng hợp tiếng nói có thể kể đến HTS, Marry TTS.

### 1.1.3 Các sản phẩm tiêu biểu trong ngoài nước

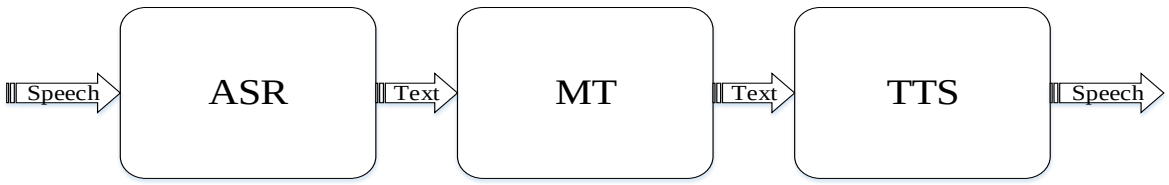
Có nhiều sản phẩm dịch tiếng nói đã được phát triển trên toàn thế giới theo nhiều phương pháp khác nhau như MASTOR của IBM, hệ thống Verbmobil, hệ thống VoiceTra của NICT hay các sản phẩm có sự tham gia của các nhóm nghiên cứu tới từ Việt Nam như các sản phẩm của tổ chức A-STAR và U-STAR, hay những hệ thống mà nhiều người biết đến như Google Translate. Google Translate (hệ thống này không chỉ có dịch tự động mà còn tích hợp cả nhận dạng tiếng nói, tổng hợp tiếng nói và chúng được ghép nối với nhau do đó có thể coi đây là hệ thống dịch tiếng nói) với hơn 100 ngôn ngữ.

Hệ thống MASTOR có khả năng dịch tiếng nói tự động đa ngôn ngữ trong thời gian thực trên những thiết bị có ít tài nguyên như thiết bị di động, cấu trúc của hệ thống này cũng gồm ba thanh phần cơ bản là ASR, MT, TTS và được triển khai theo phương pháp ghép nối tuần tự. VoiceTra là một trong những hệ thống dịch tiếng nói đa ngôn ngữ dựa trên cơ sở mạng đầu tiên cho điện thoại thông minh, nó tích hợp nhận dạng đa ngôn ngữ, dịch tự động đa ngôn ngữ, tổng hợp tiếng nói đa ngôn ngữ [4].

Những hệ thống có sự tham gia của các nhóm nghiên cứu tới từ Việt Nam như các sản phẩm của tổ chức A-STAR, phát triển hệ thống dịch tiếng nói dựa trên cơ sở mạng cho các ngôn ngữ Châu Á, việc giao tiếp giữa các mô đun trong hệ thống, được quản lý bởi một dịch vụ web được thiết kế bởi NIST gọi là STML servlet theo chuẩn STML. Tổ chức U-STAR hiện đang phát triển sản phẩm VoiceTra4U có khả năng dịch đa ngôn ngữ cho điện thoại thông minh trên nhiều nền tảng khác nhau.

## 1.2 Mô hình chung hệ thống dịch tự động tiếng nói

Như đã nêu trong phần trên, kiến trúc phổ biến nhất của hệ thống dịch tiếng nói hiện nay bao gồm ba thành phần chính: Nhận dạng tiếng nói, Dịch tự động, Tổng hợp tiếng nói. Hình 1 mô tả kiến trúc tổng quan của hệ thống dịch tiếng nói dựa trên ba mô đun vừa nêu.



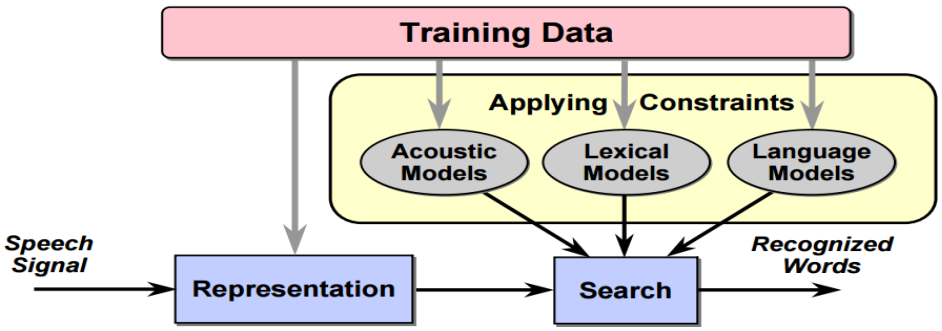

Hình 1: Kiến trúc cung của hệ thống dịch tiếng nói

Hoạt động của hệ thống dựa trên kiến trúc này có thể được mô tả một cách đơn giản như sau: dữ liệu tiếng nói trong ngôn ngữ nguồn được thu lại và đưa vào bộ ASR. Bộ ASR là bộ nhận dạng tiếng nói tự động, kết quả đầu ra của bộ này là chuỗi các từ được nói trong dữ liệu âm thanh đầu vào. Kết quả này được đưa vào bộ MT, MT là bộ dịch tự động chịu trách nhiệm dịch chuỗi các từ đầu vào trong một ngôn ngữ gọi là ngôn ngữ nguồn và trả về kết quả là bản dịch của các từ này trong ngôn ngữ khác gọi là ngôn ngữ đích. Kết quả dịch của bộ MT được đưa vào bộ tổng hợp tiếng nói TTS, bộ TTS có nhiệm vụ phát ra tiếng nói trong ngôn ngữ đích. Kết quả cuối cùng tiếng nói trong ngôn ngữ nguồn được dịch thành tiếng nói trong ngôn ngữ đích.

## 1.3 Nhận dạng tiếng nói

### 1.3.1 Tổng quan về nhận dạng tiếng nói

Nhận dạng tiếng nói là quá trình chuyển từ tiếng nói thành văn bản, việc nghiên cứu và phát triển các hệ thống nhận dạng tiếng nói tự động (ASR) hoạt động trên các loại máy đã được bắt đầu từ những năm 1930 - 1950 của thế kỷ trước. Đến nay thì nhận dạng tiếng nói đã trải qua năm thế hệ [8]. Trong đó thế hệ thứ năm từ năm 2000 đến nay phát triển các giải pháp xử lý song song để tăng độ tin cậy, kết hợp phương pháp âm học – âm vị và mô hình Markov ẩn để phát hiện và chuẩn hóa các thành phần bất quy tắc của ngôn ngữ.




Hình 2: Kiến trúc hệ thống nhận dạng tiếng nói tự động (ASR) [9].

Hệ thống nhận dạng tiếng nói tự động ASR được xây dựng để nhận ra được tiếng nói con người ở đầu vào và chuyển thành văn bản tương ứng ở đầu ra. Hình 2 mô tả kiến trúc hoạt động phổ biến của một hệ thống nhận dạng tiếng nói tự động ASR, hệ thống này gồm 2 thành phần Representation và Search. Representation là bộ phận chuyển đổi các tín hiệu tiếng nói thành các đặc trưng, biểu diễn đặc trưng tiếng nói phổ biến nhất thường được áp dụng trong nhận dạng tiếng nói là các véc tơ MFCCs. Search là bộ phận tìm ra chuỗi văn bản tương ứng với các đặc trưng biểu diễn tín hiệu tiếng nói được trích ra bởi bộ Representation. Bộ Search được xây dựng dựa trên ba mô hình là mô hình âm học (Acoustic Model), mô hình từ vựng (Lexical model) và mô hình ngôn ngữ (Language Model). Mô hình âm học cho từng âm dựa trên mô hình Markov sử dụng phân bố Gaussian nhiều chiều, trong khuôn khổ đề tài này tôi sẽ không trình bày chi tiết về mô hình này. Các mô hình Markov ẩn được sử dụng để xây dựng mô hình từ vựng và mô hình ngôn ngữ được lựa chọn là mô hình N-gram.

### 1.3.2 Mô hình ngôn ngữ N-gram

Mô hình ngôn ngữ N-gram là một mô hình xác suất có khả năng dự đoán sự xuất hiện của một từ dựa trên N-1 từ đã xuất hiện từ trước đó. Ta xét một chuỗi các từ w1,w2,..,wn hay viết gọn là wn và xác suất xuất hiện của chuỗi từ này là P(w1,w2,..,wn) hay P(wn). $w_1^n$ là chuỗi n từ xuất phát từ w1. Bài toán của ta là tính xác xuất P(wn) khi đã biết n-1 từ trước đó. Ta có:

$$P(X_1...X_n) = \prod_{k=1}^{n} P(X_k \mid X_1^{n-1}) \quad (1.3.1)$$

Áp dụng cho chuỗi các từ ta được:

$$P(\mathrm{w}_n) = P(\mathrm{w}_1^n) = \prod_{k=1}^{n} P(\mathrm{w}_k \mid \mathrm{w}_1^{k-1}) \quad (1.3.2)$$

Với mô hình N-gram ta tính xác suất xuất hiện của từ thứ k dựa trên N từ đã xuất hiện trước đó. Do đó:

$$P(\mathrm{w}_k \mid \mathrm{w}_1^{k-1}) \approx \mathrm{P}(\mathrm{w}_k \mid \mathrm{w}_{k-N+1}^{k-1}) \quad (1.3.3)$$

Ta ước lượng xác suất ở biểu thức (1.3.3) như sau:

$$P(\mathrm{w}_k \mid \mathrm{w}_{k-N+1}^{k-1}) = \frac{C(\mathrm{w}_{k-N+1}^{k-1}\mathrm{w}_k)}{C(\mathrm{w}_{k-N+1}^{k-1})} \quad (1.3.4)$$

Trong biểu thức trên có $C(\mathrm{w}_{k-N+1}^{k-1}\mathrm{w}_k)$ là số lần chuỗi các từ $\mathrm{w}_{k-N+1},,,\mathrm{w}_k$ xuất hiện và $C(\mathrm{w}_{k-N+1}^{k-1})$ là số lần chuỗi các từ $\mathrm{w}_{k-N+1},,,\mathrm{w}_{k-1}$ xuất hiện. ta đếm số lần xuất hiện này dựa trên dữ liệu được đưa vào để huấn luyện mô hình ngôn ngữ.

Cuối cùng thay kết quả của (1.3.4) vào (1.3.2) ta sẽ tính được xác suất xuất hiện của chuỗi các từ w1,,,wn.

### 1.3.3 Mô hình Markov ẩn

Mô hình Markov ẩn (HMM) là một mô hình máy trạng thái, mô hình này cho phép chúng ta xem xét đến hai thành phần là các sự kiện quan sát được (như các đặc trưng của tiếng nói) và các sự kiện ẩn (ví dụ như các từ).

Một mô hình markov ẩn được xác định bởi các thành phần sau [10]:

Q = $q_1, q_2,,,\mathrm{q}_N$ :Tập của N trạng thái

A = $a_{11}, a_{12},,,a_{nn}$ : ma trận chuyển trạng thái với aij là xác suất chuyển từ trạng thái I sang trạng thái j. có $\sum_{j=1}^{N} \mathrm{aij} = 1 \forall i$.

O = $o_1, o_2,,,o_T$ : là một chuỗi T các quan sát được tại các thời điểm t khác nhau. Mỗi oi thuộc tập V = {v1,v2,…vm} là tập tất cả các quan sát có thể được trong mỗi trạng thái.

B ={ $b_j(k)$ } : phân bố xác xuất quan sát được các quan sát o trong trạng thái $S_j$.

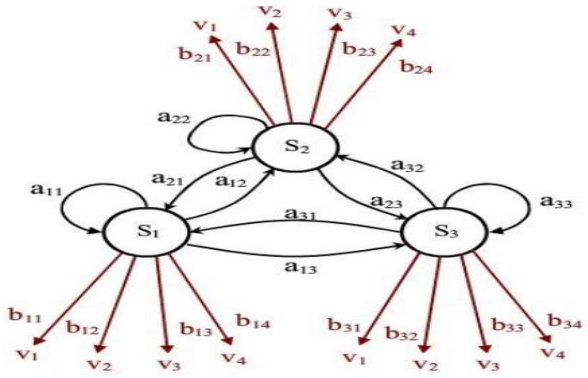
$b_j(k) = P(v_k = o_t, q_t = S_j)$ có bj(k) là xác xuất quan sát được sự kiện vk trong trại thái Sj.

Trong mô hình markov ẩn ta chấp nhận hai giả thiết sau: xác suất xuất hiện của một trạng thái chỉ phụ thuộc vào trạng thái trước đó. Xác suất thấy được quan sát o chỉ phụ thuộc vào trạng thái đã tạo ra quan sát đó.

Trong nhiều trường hợp chúng ta sẽ xem xét đến hai thành phần sau:

Π = {πi} :phân bố xác suất khởi tạo.

QA = {qx,qy…} ; QA (con của Q) là tập chấp nhận được.




Hình 3: Mô hình Markov ẩn với ba trạng thái

Hình 3 cho ta thấy mô hình markov ẩn với ba trạng thái. Ở mỗi trạng thái sự kiện có thể quan sát được thuộc V = (v1,v2,v3,v4). B = (b11,b12,…,b34) phân bố xác xuất quan sát được sự kiện và bj(k) là xác xuất quan sát được sự kiện Vk trong trạng thái Sj. Các trạng thái Sj được coi là ẩn so với việc quan sát. Đó là lý do tại sao ta gọi là mô hình markov ẩn.

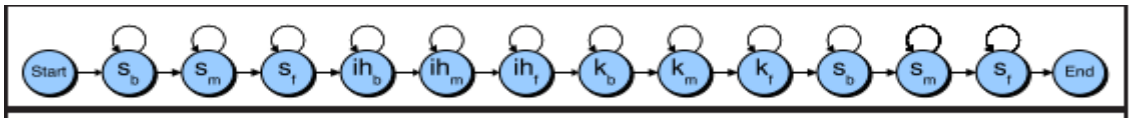
Với mô hình markov ẩn ta xem xét đến ba vấn đề chính sau:

- Vấn đề 1 (Computing likelihood): cho mô hình λ(A,B,π) và chuỗi quan sát được O xác định độ tương đồng (likelihood) P(O|λ).
- Vấn đề 2 (decoding): Cho một chuỗi quan sát O và mô hình HMM λ(A,B,π), tìm ra chuỗi Q tối ưu nhất đã phát sinh ra O.
- Vấn đề 3 (learning): cho một chuỗi quan sát O và tập các trạng thái của HMM, điều chỉnh các tham số λ = {A, B, π} của HMM để P(O| λ) lớn nhất. (Đây chính là bài toán huấn luyện mô hình. Bài toán này đem lại khả năng rất quan trọng của HMM đó là mô hình hóa đối tượng cụ thể trong thực tế với dữ liệu liên tục).

Lần lượt giải quyết ba vấn đề trên bằng ba thuật toán sau: Forward, Viterbi, Baum Welch.

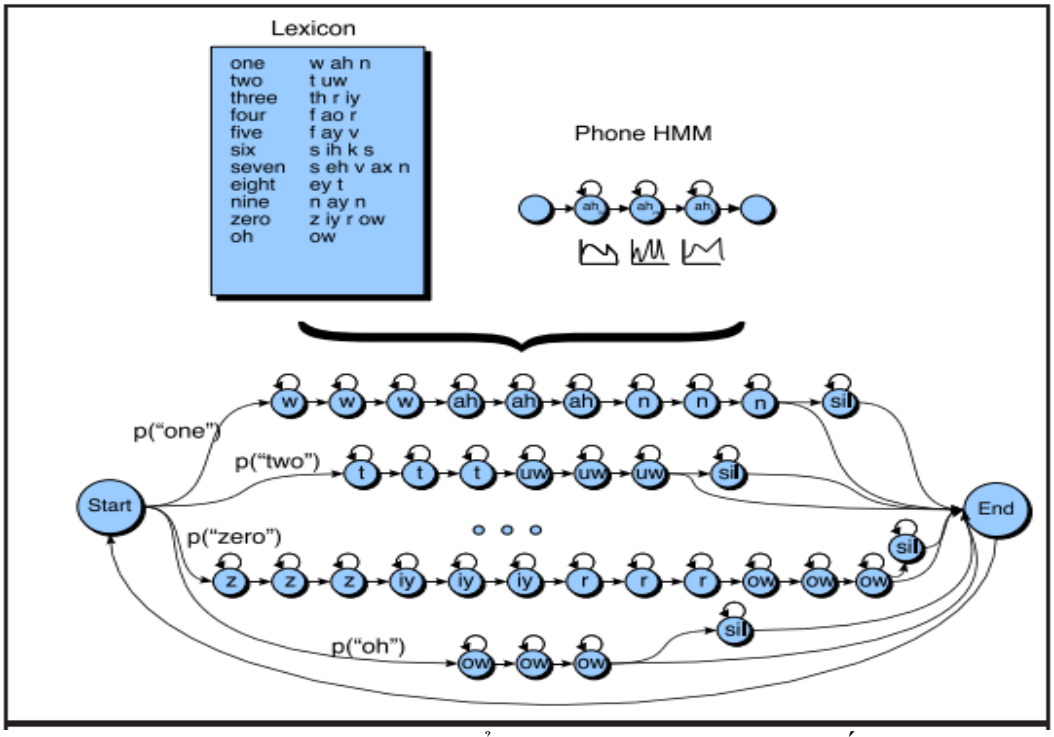
### 1.3.4 Áp dụng mô hình Markov ẩn cho nhận dạng tiếng nói

Trong mô hình chung của nhận dạng tiếng nói mô hình Markov được sử dụng để mô hình hóa từng âm. Có nhiều cách triển khai áp dụng mô hình Markov ẩn khác nhau, ở đây tôi xin trình bầy cách đơn giản nhất là mỗi âm vị được mô hình bởi một mô hình Markov ẩn ba trạng thái,mỗi trạng thái chỉ có thể chuyển sang trạng thái kế tiếp hoặc ở lại là chính nó [10]. Một từ được phiên âm là một chuỗi các âm vị. Do đó một từ sẽ được mô hình hóa bằng cách ghép nối các mô hình Markov ba trạng thái của các âm vị cấu thành nên từ đó.



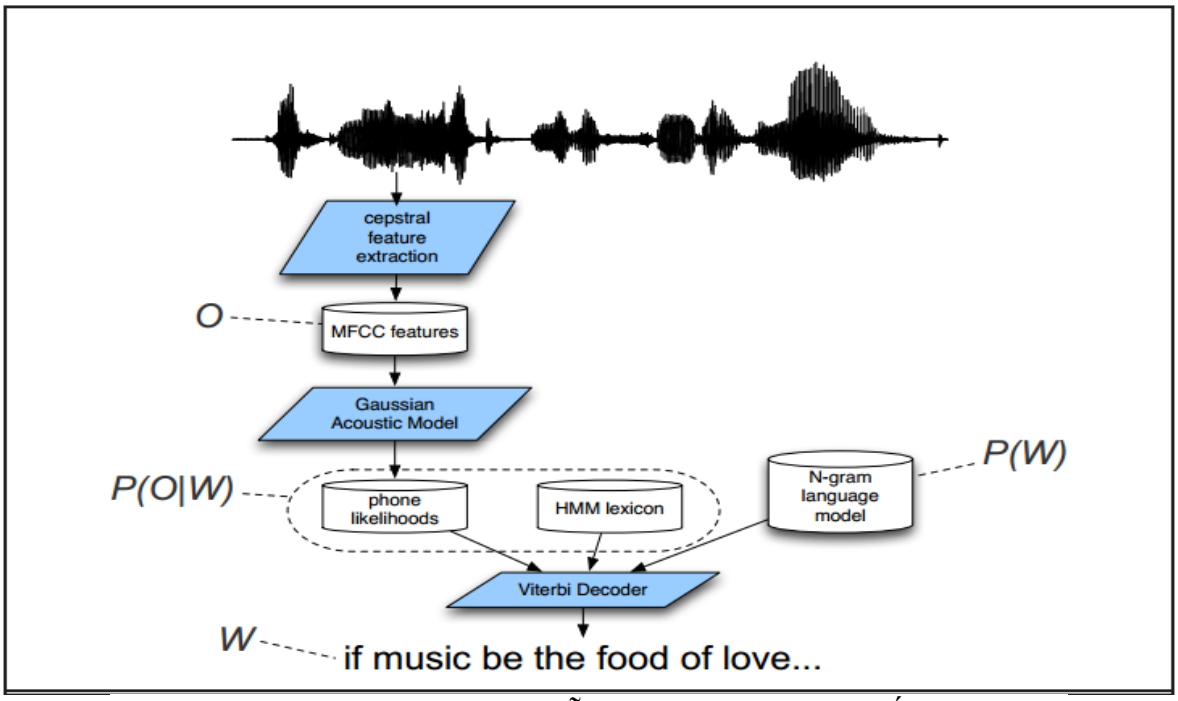

Hình 4: Mô hình HMM từ “six” [10]

Hình 4 mô tả mô hình Markov ẩn áp dụng cho từ six với phiên âm của từ “six” là /s ih k s/. Một âm vị được mô hình hóa bởi mô hình Markov ẩn ba trạng thái, ví dụ âm vị /s/ được mô hình hóa bởi mô hình Markov ẩn với ba trạng thái là $s_b, s_m, s_t$. Hình 5 mô tả một mô hình từ vựng được xây dựng trên mô hình Markov ẩn, với các chuỗi trạng thái tính từ trạng thái bắt đầu đến trạng thái kết thúc mô tả một từ trong từ điển.




Hình 5: Mô hình Markov ẩn cho bộ từ vựng là các số [10]

Sau khi đã mô hình hóa bộ từ vựng bằng mô hình Markov ẩn, mô hình ngôn ngữ bằng mô hình N-gram và mô hình âm học bằng mô hình Markov-Gaussian nhiều chiều ta có được một hệ thống có thể coi hoàn chỉnh của một bộ ASR. Hình 6 cho ta thấy qua trình để nhận ra một chuỗi văn bản từ tín hiệu tiếng nói đầu vào.




Hình 6: Quá trình nhận dạng chuỗi văn bản từ tín hiệu tiếng nói [10]

Tín hiệu tiếng nói được đưa vào một bộ trích chọn đặc trưng (cepstral feature extraction). Trong bộ trích chọn đặc trưng tín hiệu tiếng nói sẽ được chuyển thành một chuỗi các véc tơ đặc trưng, cụ thể ở đây là các véc tơ MFCCs. Các véc tơ đặc

trưng này cũng chính là các quan sát $o_i$ đầu vào của mô hình Markov ẩn (Mô hình hóa bộ từ vựng). Các véc tơ đặc trưng hay các quan sát được đưa vào bộ mô hình âm học (Gaussian Acoustic Model), nhiệm vụ của bộ mô hình âm học là tính ra xác suất ở mỗi trạng thái của mô hình Markov ẩn quan sát được véc tơ đặc trưng (hay chính là các quan sát đầu vào o), xác suất này chính là phân bố xác suất B. Sau khi có được chuỗi các quan sát $O = o_1, o_2, ..., o_n$ là các véc tơ đặc trưng và phân bố xác suất B thì mô hình Markov ẩn và mô hình ngôn ngữ được sử dụng để tìm ra chuỗi từ phù hợp.

Giả sử $\mathrm{W} = \mathrm{w}_1, \mathrm{w}_2, ..., \mathrm{w}_n$ là chuỗi các từ cần tìm. Ta có công thức để tìm W như sau:

$$\mathrm{W} = \arg\max_{\mathrm{W} \in L} P(\mathrm{W} \mid O) \quad (1.3.5)$$

Dựa vào công thức nhân xác suất ta có:

$$\mathrm{W} = \arg\max_{\mathrm{W} \in L} \frac{P(O \mid \mathrm{W})\,\mathrm{P}(\mathrm{W})}{P(O)} \quad (1.3.6)$$

Chúng ta có thể bỏ quan P(O), suy ra:

$$\mathrm{W} = \arg\max_{\mathrm{W} \in L} P(O \mid \mathrm{W}) P(\mathrm{W}) \quad (1.3.7)$$

Trong biểu thức trên ta tính P(W) dựa vào mô hình ngôn ngữ N-gram, P(O|W) được tính dựa vào mô hình Markov ẩn và phân bố xác suất B tính được từ mô hình âm học. Cuối cùng ta tìm được chuỗi các từ W có xác suất cao nhất.

## 1.4 Dịch tự động văn bản

### 1.4.1 Tổng quan về dịch tự động

Dịch tự động hay còn gọi là dịch máy, là công cụ để dịch chuỗi các từ từ ngôn ngữ này sang ngôn ngữ khác. Bắt đầu từ những năm 1950 dịch máy đã trở thành một lĩnh vực nghiên cứu và bắt đầu được mở rộng, năm 1952 hội nghị chuyên đề đầu tiên về dịch máy được tổ chức tại MIT dưới sự chủ chì của Yehoshua Bar-Hillel [11]. Đến năm 1954 Georgetown experiment một trong những dự án đầu tiên về dịch tự động đã thành công trong việc dịch hơn 60 câu tiếng Nga sang tiếng Anh [11]. Vào cuối những năm 1980 và đầu những năm 1990 những nhà khoa học của phòng nghiên cứu IBM đã thành công trong việc áp dụng hướng tiếp cận dịch máy thống kê vào dịch tự động, và từ đó dịch máy thống kê trở thành hướng nghiên cứu phổ biến nhất trong lĩnh vực dịch tự động [12].

Trong khuôn khổ đề tài này chúng ta chỉ tập trung nghiên cứu về dịch máy thống kê, đây là hướng nghiên cứu phổ biến nhất hiện nay. Hệ thống dịch máy thống kê được tạo nên trên cơ sở các mô hình thống kê có các tham số được huấn luyện từ các cặp câu song ngữ. Có ba hướng phát triển chính của dịch máy thống kê đó là:

- Dịch máy thống kê trên cơ sở từ: Ý tưởng cơ bản của phương pháp này dựa trên việc dịch các từ từ ngôn ngữ này sang ngôn ngữ khác và việc dóng hàng vị trí các từ trong câu.
- Dịch máy thống kê trên cơ sở cụm từ: Mục đích để giảm bớt hạn chế của việc dịch trên cơ sở từ bằng cách dịch trên cơ sở cụm từ.

- Dịch máy thống kê trên cơ sở cú pháp: phương pháp này dựa trên ý tưởng dịch các đơn vị cú pháp.

### 1.4.2 Mô hình cơ bản của dịch máy thống kê trên cơ sở từ

Để thống nhất trong toàn bộ chương về dịch tự động ta sẽ sử dụng một quy ước chung sau:

- Quá trình dịch sẽ dịch từ ngôn ngữ nguồn sang ngôn ngữ đích.
- Gọi F là câu gồm có các từ $f_1, f_2, ..., f_{l_f}$ trong ngôn ngữ nguồn và có độ dài $l_f$.
- Gọi E là câu gồm có các từ $e_1, e_2, ..., e_{l_e}$ trong ngôn ngữ đích có độ dài $l_e$.
- Việc dịch một câu từ ngôn ngữ nguồn sang ngôn ngữ đích chính là dịch từ F sang E, hay nói cách khác E chính là bản dịch của F mà ta muốn tìm.

Cơ sở của việc "Dịch máy thống kê trên cơ sở từ" là việc dịch một từ trong ngôn ngữ nguồn sang ngôn ngữ đích, việc này được thực hiện dựa trên một hàm phân bố xác suất:

$$P_f : e \to P_f(e) \quad (1.4.1)$$

Trong đó f là từ trong ngôn ngữ nguồn, kết quả đầu ra của hàm phân bố xác suất là một lựa chọn từ e trong ngôn ngữ đích. Ta có hai tính chất của hàm phân bố xác suất trên là:

$$\begin{cases} \sum_e P_f(e) = 1 \\ \forall e : 0 \le P_f(e) \le 1 \end{cases} \quad (1.4.2)$$

Để hiểu cách ước lượng phân bố xác suất này ta xem xét ví dụ sau: tìm hàm phân bố xác suất của việc dịch từ haus trong tiếng Đức sang tiếng Anh. Để tìm được phân bố xác xuất này ta dựa trên việc đếm số lần xuất hiện của từ haus trong văn bản tiếng Đức và số lần xuất hiện các từ là bản dịch của từ haus trong các văn bản tiếng Anh, chú ý văn bản tiếng Anh này phải là bản dịch các câu trong văn bản Tiếng đức trước đó. Ta có: từ "haus" xuất hiện 10000 lần, từ "house" xuất hiện 8000 lần ,từ "building" xuất hiện 16000, từ "how" xuất hiện 200 lần, từ "house hold" xuất hiện 100 lần và từ "shell" xuất hiện 100 lần. Vậy ta có: $P_{haus}(house) = 0.8$ ; $P_{haus}(building) = 0.16$ Bằng cách này ta có thể tìm được phân bố xác suất:

$$P_{haus}(e) = \begin{cases} 0.8 \text{ nếu } e = house \\ 0.16 \text{ nếu } e = building \\ 0.02 \text{ nếu } e = how \\ 0.01 \text{ nếu } e = house\ hold \\ 0.01 \text{ nếu } e = shell \end{cases}$$

Khi áp dụng cho nhiều từ thì ta sẽ lưu các xác suất này vào một bảng được gọi là bảng xác suất dịch máy thống kê, bảng này gồm các từ trong ngôn ngữ nguồn và phân bố xác suất của các bản dịch của nó trong ngôn ngữ đích, các xác suất trong bản này ký hiệu là t(e|f) với f là ngôn ngữ nguồn còn e là ngôn ngữ đích và t(e|f) là xác suất từ f được dịch sang từ e. Từ phân bố xác suất dịch cho từng từ, ta có thể xây dựng được một mô hình dịch tự dạng đơn giản dựa trên phân bố xác suất sau:

$$P(E \mid F) = \prod_{j=1}^{l_e} t(e_j, f_j) \quad (1.4.3)$$

Vấn đề tiếp theo chúng ta sẽ đề cập đến sau đây là làm cách nào để áp dụng mô hình ngôn ngữ của ngôn ngữ đích trong dịch tự động để bản dịch chính xác hơn và mang nhiều ngữ nghĩa hơn. Để làm được điều này ta xem xét cách tiếp cận Noisy channel. Sử dụng công thức bayes ta có:

$$P(E \mid F) = P(F \mid E) * P(E) / P(F) = P(F \mid E) * P(E) \quad (1.4.4)$$

Suy ra:

$$E = \arg\max_E (P(F \mid E) * P(E)) \quad (1.4.5)$$

Vậy để tìm được bản dịch E của F thì ta cần tính P(E) và P(F|E) [13]. Trong đó P(E) được tính bởi mô hình ngôn ngữ trong ngôn ngữ đích, còn P(F|E) được tính bởi mô hình xác suất (1.4.3) và các mô hình xác suất trình bày ở chương sau. Có một điều dễ gây nhầm lẫn ở đây là trong công thức (1.4.3) ta tính P(E|F) nhưng ở đây ta lại cần tính P(F|E), thực ra đây chỉ là đảo lại nguồn và đích vì chúng là bản dịch của nhau. Vậy để tính toán P(F|E) ta chỉ cần đảo lại một chút ở trên, đó là thay vì ta lưu bảng xác suất là các xác suất t(e|f) thì bây giờ ta lưu t(f|e). Ở các mô hình sau cũng tương tự. F là câu trong ngôn ngữ nguồn , E là bản dịch của F trong ngôn ngữ đích nhưng để tìm E ta không tính trực tiếp P(E|F) mà ta tính P(F|E) rồi từ đó thông qua Noisy channel tính P(E|F) và tìm ra E.

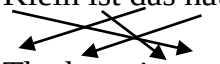
### 1.4.3 Mô hình dịch máy thống kê trên cơ sở từ

Trong chương này chúng ta sẽ tập trung vào các mô hình dịch máy thống kê trên cơ sở từ được xây dựng bởi IBM, ta sẽ xem xét đến hai mô hình đầu tiên và là cơ sở cho phương pháp tiếp cận này đó là mô hình IBM 1 và IBM 2.

Mô hình dịch tự động ta trình bày ở chương trước còn quá thô sơ và đơn giản, có một vấn đề ma ta nhìn thấy ngay đó là trong các ngôn ngữ khác nhau thì thứ tự các từ trong câu cũng khác nhau mà trong mô hình dịch tự động cơ bản vị trí các từ sau khi dịch không thay đổi dẫn đến bản dịch có tỉ lệ sai sót rất cao, do đó để giải quyết vấn đề này ta xem xét đến một khái niệm gọi là Alignment (Dóng hàng), thể hiện sự tương quan vị trí giữa các từ trong câu của ngôn ngữ nguồn và vị trí các từ là bản dịch của nó trong câu đích. Ta chuẩn hóa việc dóng hàng trong câu bằng một hàm: $a: j \to i$ với j là vị trí của từ f trong câu F và i là vị trí của e trong câu E, e là bản dịch của f.

Ví dụ:

Klein ist das haus



The hous is small

1 2 3 4

a : {1->4 , 2->3 , 3->1 , 4->2} hay cách viết khác: a(1) = 4, a(2)=3.

Trong đây câu "The hous is small" là trong ngôn ngữ nguồn và "Klein ist das haus" là câu trong ngôn ngữ đích, để tránh nhầm lẫn hãy xem kĩ lại về Noisy channel.

Chú ý: có thể có những từ trong ngôn ngữ đầu ra không có quan hệ với bất cứ từ nào trong ngôn ngữ nguồn thay vào đó chúng ta sẽ coi nó được dịch từ NULL trong ngôn ngữ nguồn.

Dựa trên khái niệm về Aligment và xác suất dịch cặp từ t(f|e) trong bảng xác suất dịch máy thống kê ta tiến hành xây dựng mô hình dịch máy thống kê dựa trên xác

suất thống kê. Ở đây tôi xin trình bày về hai mô hình xác suất cho dịch máy thống kê đó là IBM 1 và IBM 2.

Áp dụng công thức xác suất có điều kiện:

$$P(X \mid Y) = \mathrm{P(X, Y)} / \mathrm{P(Y)} \quad (1.4.6)$$

Suy ra:

$$P(X, Y \mid Z) = P(X, Y, Z) / \mathrm{P(Z)}$$

$$\mathrm{P(Y \mid X, Z) = P(X, Y, Z) / P(X, Z)}$$

$$\mathrm{P(X \mid Z) = P(X, Z) / P(Z)}$$

$$\Rightarrow P(Y \mid \mathrm{X, Z}) * \mathrm{P(X \mid Z) = (P(X, Y, Z) / P(X, Z)) * (P(X, Z) / P(Z)) = P(X, Y, Z) / P(Z)}$$

$$\Rightarrow P(Y \mid X, Z) * P(X \mid Z) = P(X, Y \mid Z) \quad (1.4.7)$$

Gọi a là hàm dóng hàng (hàm Alignment), với mỗi cặp $(f_j, e_i)$ thì ta có a(j)=i. Áp dụng biểu thức (1.4.7) với F,a,E ta được:

$$P(F \mid a, E) = P(a \mid E, F) * P(F \mid E) \quad (1.4.8)$$

$$\Rightarrow P(F \mid E) = P(F, a \mid E) / P(a \mid F, \mathrm{E}) = \sum_a P(F, a \mid E)$$

$$\Rightarrow P(F \mid E) = \sum_{a(1)=0}^{l_e} \ldots \sum_{a(l_f)=0}^{l_e} P(F, a \mid E) \quad (1.4.9)$$

Xét đến việc ký tự NULL có thể được thêm vào quá trình dịch do vậy ta lấy $(l_e + 1)^{lf}$ là tất cả những cách dóng hàng (alignment) có thể, từ đó ta tính P(F,a|E) như sau:

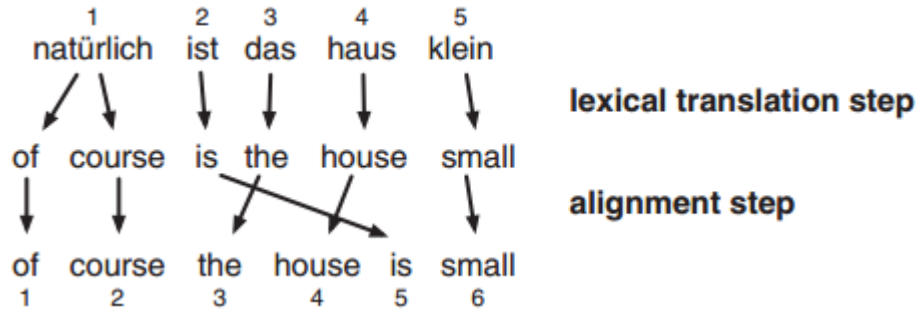
$$P(F, a \mid E) = \frac{\varepsilon}{(l_e + 1)^{l_f}} \prod_{j=1}^{l_f} t(f_j \mid e_{a(j)}) \quad (1.4.10)$$

Cuối cùng từ (1.4.9) và (1.4.10) ta tìm được P(F|E) như sau:

$$P(F \mid E) = \frac{\varepsilon}{(l_e + 1)^{l_f}} \prod_{j=1}^{l_f} \sum_{i=0}^{l_e} t(f_j \mid e_i) \quad (1.4.11)$$

Phân bố xác suất trong công thức (1.4.11) là phân bố xác suất của mô hình IBM 1, mô hình IBM 1 đã tính đến sự thay đổi vị trí sau khi dịch nhưng kết quả dịch vẫn còn rất kém và vẫn cần sắp xếp lại các từ trong bản dịch do đó trong mô hình IBM 2 sau đây ta đưa thêm một phân bố xác suất dóng hàng (Alignment) để sắp xếp vị trí từ j trong F với từ i trong E (hai từ j và từ i là bản dịch của nhau).

$$a(j \mid i, l_e, l_f)$$




Hình 7: Quá trình hoạt động mô hình IBM2

Hình 7 cho thấy mô hình IBM 2 hoạt động thông qua hai bước. Bước một lexical translation là bước dịch từng từ trong E sang F và bước hai là sắp xếp lại các từ

trong F. Xem lại cách tiếp cận Noisy channel ở phần trên để tránh nhầm lẫn và hiểu được tại sao lại là dịch từ E sang F.

Mô hình IBM 2 gồm hai thành phần [14]:
- t(f|e): xác suất dịch từ e sang f
- $a(j|i,l_e,l_f)$: là xác suất để với mỗi từ f được đặt ở vị trí j với điều kiện từ e được đặt ở vị trí i và độ dài hai câu là le và lf. Với le thuộc (1,…,Le), lf thuộc (1,…,Lf), i thuộc (1…le), j thuộc (1…lf). Le và Lf là độ dài lớn nhất có thể của các câu E và F.

Từ định nghĩa trên ta có biểu thức toán học hợp nhất hai bước lexical translation và alignment là :

$$P(F,a|\mathrm{E}) = \varepsilon \prod_{j=1}^{l_f} t(f_j|e_{a(j)}) \times a(a(j)|j,l_e,l_f) \quad (1.4.12)$$

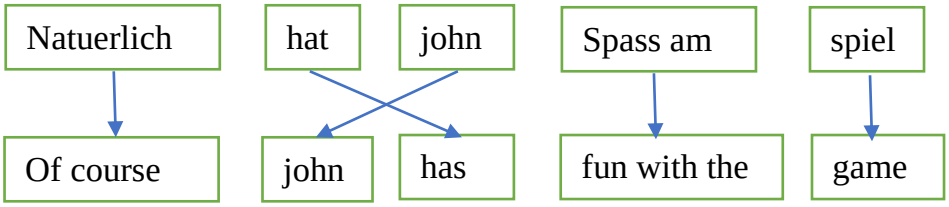
Từ (1.4.9) và (1.4.12) suy ra:

$$P(F|E) = \varepsilon \prod_{j=1}^{l_f} \sum_{i=1}^{l_e} t(f_j|e_{a(j)}) \times a(a(j)|i,l_e,l_f) \quad (1.4.13)$$

### 1.4.4 Dịch máy thống kê trên cơ sở cụm từ

Mô hình dịch máy thống kê trên cơ sở từ đã phát triển đến những mô hình rất mạnh, nhưng nó vẫn có những vấn đề chưa được giải quyết triệt để như là việc có những từ bị mất đi hoặc thêm vào sau khi dịch. Do đó mô hình dịch máy thống kê trên cơ sở cụm từ ra đời để cải thiện những thiếu sót của mô hình dịch máy thống kê trên cơ sở từ.

Cơ sở của việc dịch máy thống kê trên cơ sở cụm từ chính là việc chia các câu thành các cụm từ và sau đó dịch các cụm từ trong câu của ngôn ngữ nguồn sang cụm từ trong câu của ngôn ngữ đích. Ta xét ví dụ minh họa sau:




Trong ví dụ này câu trong ngôn ngữ nguồn được chia thành các cụm từ (phrase), các cụm này được dịch thành các cụm trong ngôn ngữ đích và ghép lại thành câu trong ngôn ngữ đích. Chú ý cụm từ (phrase) ở đây khác với cụm từ trong ngôn ngữ học ví dụ cụm "fun with the" trong ví dụ trên không có ý nghĩa trong ngôn ngữ học.

Mô hình xác suất cho dịch máy thống kê dựa trên cơ sở cụm từ như sau:

$$P(F|E) = P(f_i^{-I}|e_i^{-I}) = \prod \phi(\overline{f_i}|\overline{e_i}) \times d(start_i - end_{i-1} - 1) \quad (1.4.14)$$

Trong đó F được chia thành I cụm $\overline{f_i}$ và mỗi cụm $\overline{f_i}$ được dịch thành cụm $\overline{e_i}$ (vì theo cách tiếp cận noisy channel) . Xác suất $\phi(\overline{f_i}|\overline{e_i})$ là xác suất mà $\overline{e_i}$ được dịch

thành $\overline{f_i}$ Việc sắp xếp lại câu sau khi dịch được xử lý bằng mô hình distance_based reordering:

$$d(start_i - end_{i-1} - 1)$$

Với $start_i$ là vị trí đầu tiên của cụm từ mà dịch sang cụm i trong ngôn ngữ đích và $end_{i-1}$ là vị trí cuối cùng của cụm mà dịch sang cụm i-1 trong ngôn ngữ đích. Hàm distance_based reordering:

$$d = \alpha^{|x|} \text{ với } \alpha \in [0,1] \quad (1.4.15)$$

Dựa trên mô hình xác suất xây dựng dựa trên công thức (1.4.14) và tập dữ liệu huấn luyện gồm các cặp câu song ngữ ta có thể tiến hành ước lượng xác suất dịch các của cụm từ giống như làm với các từ. Trên toàn bộ tập dữ liệu ta tiến hành trích rút các cụm từ từ tất cả các cặp câu $E^{(k)}, F^{(k)}, k = 1...n$ , k chính là cặp câu thứ k trong tập dữ liệu các cặp câu song ngữ.sau khi trích rút ta lưu trữ các cặp $(\bar{e}, \overline{f})$ và đếm số lần xuất hiện của cặp đó thông qua biến count$(\bar{e}, \overline{f})$. Sau khi đếm xong ta có thể ước lượng được xác suất mới theo công thức sau:

$$\phi(\overline{f}, \bar{e}) = \frac{count(\bar{e}, \overline{f})}{\sum_{\overline{f_i}} count(\bar{e}, \overline{f_i})} \quad (1.4.16)$$

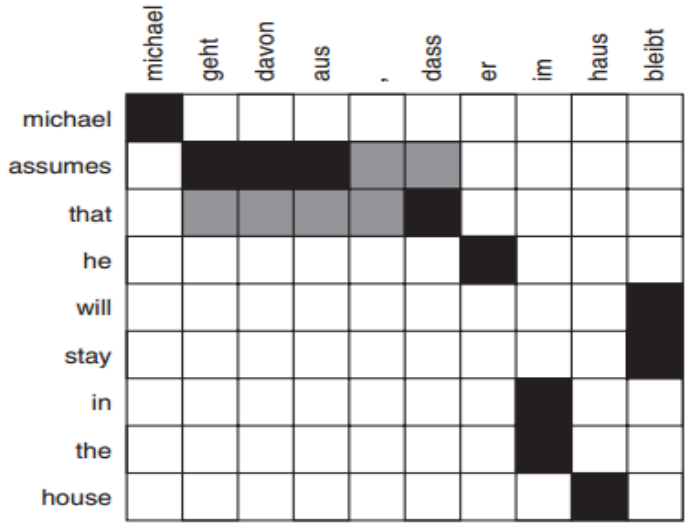
Sau khi tính được xác suất mới ta cật nhật lại bảng xác suất dịch cụm từ, bảng này lưu trữ xác suất dịch các cụm từ trong ngôn ngữ nguồn, thành các cụm từ trong ngôn ngữ đích .

Mô hình dịch máy thống kê trên cơ sở cụm từ mà ta trình bày trên đây đều phải dựa trên cơ sở là các cụm từ của câu, vấn đề còn lại bây giờ là làm thế nào ta trích rút được các cụm từ này từ câu. Từ dữ liệu đầu vào là các cặp câu $E^{(k)}, F^{(k)}, k = 1...n$ ta tiến hành trích rút các cặp câu $(\bar{e}, \overline{f})$ thông qua hai giai đoạn.

Giai đoạn một tìm dóng hàng (Alignment) thích hợp cho mỗi cặp câu, với mỗi $E \in E^{(k)}$ và $F \in F^{(k)}$ thì Alignment hợp lý nhất được tìm theo công thức (1.4.18) sau [15]:

$$P(F, a \mid E, l_f) = \mathrm{P(F, a \mid E)} = \varepsilon \prod_{j=1}^{l_f} t(f_j \mid e_{a(j)}) \times a(a(j) \mid j, l_e, l_f) \quad (1.4.17)$$

$$a^{(k)} = \arg\max_a P(F, a \mid E, l_f) \quad (1.4.18)$$




Hình 8: Ma trận Alignment [12]

Hình 8 mô tả một ma trận Alignment cho một cặp câu, trong đó các ô trong ma trận gọi là các điểm Alignment và các ô màu đen là các điểm Alignment của cặp câu được tính từ Alignment hợp lý nhất mà ta tìm được ở trên.

Giai đoạn hai: sau khi có được Alignment ta tiến hành trích rút các cặp phrase (cụm từ). Trước tiên xem xét khái niệm **consistency:** ta nói cặp $(\overline{f},\overline{e})$ được gọi là tương thích (consistent) với alignment A khi và chỉ khi tất các các từ $f_1, f_2,..., f_n$ trong $\overline{f}$ có các điểm alignment sinh ra từ nó đều thuộc A và tất cả các từ $e_1,e_2,...,e_n$ trong $\overline{e}$ có các điểm aligment sinh ra từ nó đều thuộc A. Xem trên hình 8 thì A là tập tất cả các điểm có màu đen, $(\overline{f},\overline{e})$ được gọi là tương thích với A khi và chỉ khi tất cả các điểm đen trên các cột $f_j$ và tất cả các điểm đen trên các hàng $e_i$ đều phải thuộc A.

Mô tả bằng toán học **consistency**: cặp $(\overline{f},\overline{e})$ được gọi là tương thích với alignment A khi và chỉ khi thoả mãn điều kiện sau:

$$\begin{cases} \forall e_i \in \overline{e} : (e_i, f_j) \in A \Rightarrow f_j \in \overline{f} \\ \forall f_j \in \overline{f} : (e_i, f_j) \in A \Rightarrow e_i \in \overline{e} \\ \exists e_i \in \overline{e}, f_j \in \overline{f} : (e_i, f_j) \in A \end{cases} \quad (1.4.19)$$

Từ khái niệm **consistency** ta áp dụng để tìm tất cả các cụm (phrase) từ một cặp $E^{(k)}, F^{(k)}, k = 1...n$. Ý tưởng là lặp lại trên tất cả các cụm $\overline{e}$ và tìm ra cụm ngắn nhất $\overline{f}$ ($\overline{e}, \overline{f}$ phải tương thích với alignment A) và ghép cặp chúng.

Giải thuật tìm tất cả các cụm của cặp câu $E^{(k)}, F^{(k)}, k = 1...n$.

Gọi BP là tập các cụm mà ta trích rút được từ cặp câu $E^{(k)}, F^{(k)}, k = 1...n$

Lặp với mỗi cụm $\overline{e}$ của $E^{(k)}$:

- Tìm cụm $\overline{f}$ ngắn nhất mà sao cho tất cả các từ trong $\overline{e}$ đều được dóng hàng với từ trong $\overline{f}$.
- Kiểm tra xem $\overline{f}$ có khác rỗng hay không, nếu có thì chuyển sang cụm $\overline{e}$ khác

- Kiểm tra xem cặp $(\bar{e}, \bar{f})$ có tương thích với alignment A hay không, nếu không thì chuyển sang cụm $\bar{e}$ khác.

Thêm cụm $(\bar{e}, \bar{f})$ )vào BP.

Cuối cùng sau khi trích được tất cả các cặp câu thì ta tiến hành tính xác suất dịch theo công thức (1.4.16) và lưu các giá trị này vào bảng xác suất dịch cụm từ.

## 1.5 Tổng hợp tiếng nói

### 1.5.1 Tổng quan về tổng hợp tiếng nói

Tổng hợp tiếng nói là phương pháp tạo tiếng nói nhân tạo. Một hệ thống tổng hợp tiếng nói thường là một hệ thống tạo ra tiếng nói từ văn bản. Nghiên cứu về tổng hợp tiếng nói đã bắt đầu từ rất lâu, năm 1779 nhà khoa học người Đan Mạch Christian Kratzenstein đã xây dựng mô phỏng đơn giản hệ thống cấu âm của con người, mô hình này đã có thể phát ra được âm thanh của một số nguyên âm dài. Đến tận thế kỷ 19 các nghiên cứu tổng hợp tiếng nói vẫn còn ở mức đơn giản, phải sang đến thế kỷ 20 khi mà có sự lớn mạnh của hệ thống điện, điện tử thì mới thực sự xuất hiện những hệ thống tổng hợp tiếng nói chất lượng. Hiện nay có nhiều phương pháp tổng hợp tiếng nói, trong đó có một số phương pháp phổ biến [16]:

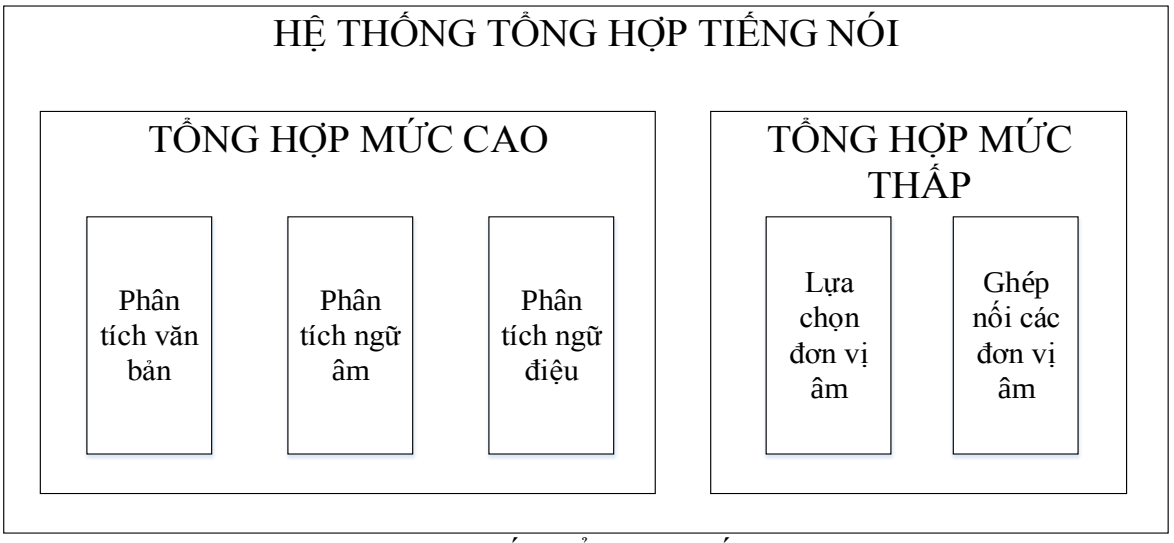
- Phương pháp mô phỏng hệ thống phát âm của con người.
- Phương pháp tổng hợp Formant.
- Phương pháp tổng hợp ghép nối.

Thừa hưởng những kết quả nghiên cứu của viện nghiên cứu quốc tế MICA, nội dung đồ án này sẽ tập trung tìm hiểu phương pháp tổng hợp ghép nối.

### 1.5.2 Phương pháp tổng hợp ghép nối

Tổng hợp ghép nối là phương pháp ghép các đoạn tiếng nói đã được thu âm từ trước. Do trong phương pháp này chúng ta sử dụng các đoạn tiếng nói hoàn toàn tự nhiên nên có thể hy vọng rằng tiếng nói được tổng hợp trong kỹ thuật này cũng hoàn toàn tự nhiên. Nhưng trong phương pháp này có một vấn đề lớn đó là làm sao ghép nối các đoạn tiếng nói thì giảm thiểu các hiện tượng không liên tục về phổ và ngữ điệu. Sự không liên tục về phổ xuất hiện khi các formant tại điểm kết nối không ăn khớp và sự không liên tục về ngữ điệu xuất hiện khi cao độ tại điểm kết nối không ăn khớp.

Quá trình tổng hợp tiếng nói được chia thành hai mức xử lý là tổng hợp mức cao và tổng hợp mức thấp. Hình 9 mô tả mô hình một hệ thống có hai mức nên trên.




Hình 9: Mô hình hệ thống tổng hợp tiếng nói từ văn bản

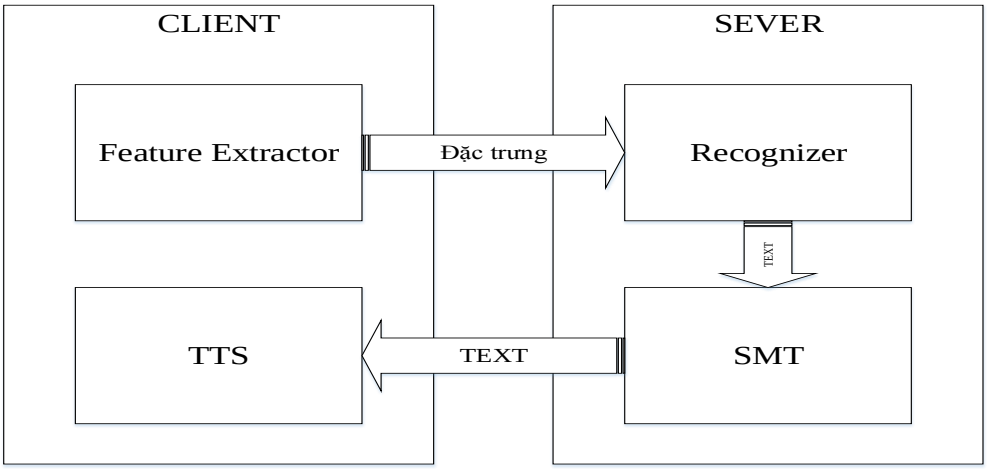
Tổng hợp mức cao là giai đoạn đầu của quá trình tổng hợp. Tại giai đoạn này văn bản đầu vào được chuẩn hóa, từ đó phát sinh thông tin về ngữ âm, ngữ điệu.

Tổng hợp mức thấp là giai đoạn sau của quá trình tổng hợp. Sau khi văn bản đã được chuẩn hóa và phát sinh ngữ điệu, bộ tổng hợp mức thấp sẽ tiến hành lựa chọn đơn vị âm từ một tập các đơn vị âm ứng viên sao cho phù hợp nhất. Sau khi đơn vị âm được lựa chọn ta sẽ ghép chúng với nhau để tạo thành câu hoàn chỉnh. Ở bước ghép nối các đơn vị âm để không gây ra sự chênh lệch về phổ và ngữ điệu, ta sẽ sử dụng thuật toán ghép nối để làm giảm sự không liên tục xuống mức thấp nhất. Hiện nay một phương pháp phổ biến được sử dụng để làm điều này là phương pháp PSOLA, phương pháp này cho phép kết nối trơn các đoạn tín hiệu tiếng nói, giúp điều khiển tốt cao độ cũng như trường độ của tiếng nói. Phương pháp PSOLA gồm ba bước cơ bản [17]:
- phân tích tín hiệu thành các sóng cơ bản
- Tính toán các điểm đánh dấu cao độ
- Tổng hợp lại các đoạn tín hiệu đã được biến đổi ở bước 2.

## 1.6 Vấn đề đặt ra cho đồ án

Trong phần 1.2 ta đã lựa chọn xây dựng mô hình dịch tiếng nói dựa trên ba mô đun là nhận dạng tiếng nói ASR, dịch tự động MT. Nhưng khi triển khai hệ thống gồm ba mô đun này ta gặp phải những vấn đề sau:
- Vấn đề thứ nhất là làm sao để ghép nối ba mô đun.
- Vấn đề thứ hai sau khi ghép nối ba mô đun nếu đặt toàn bộ ba mô đun này trên một thiết bị sẽ gây tốn kém, lãng phí bộ nhớ.
- Vấn đề thứ ba sinh ra khi ta xây dựng hệ thống theo mô hình client-sever để giải quyết vấn đề thứ hai, thì lại nảy sinh một vấn đề mới đó là nếu ở client ta ghi âm tiếng nói và truyền lên sever thì sẽ gây chậm trễ thời gian và tốn kém dung lượng truyền qua mạng.
- Vấn đề cuối cùng là thu thập bộ dữ liệu huấn luyện cho nhận dạng tiếng nói và dịch tự động.

Nhiệm vụ của đồ án này là xây dựng được hệ thống dịch tiếng nói giải quyết toàn bộ bốn vấn đề trên. Hình 10 mô tả kiến trúc chung của hệ thống được thiết kế để giải quyết các vấn đề 1,2,3.




Hình 10: Kiến trúc hệ thống theo mô hình Client sever

Dựa theo kiến trúc hình 10, để giải quyết vấn đề thứ nhất thì chúng ta sẽ ghép nối các mô đun theo hướng nối tiếp, kết quả của mô đun nhận dạng sẽ là đầu vào của mô đun dịch và kết quả của mô đun dịch sẽ là đầu vào của mô đun tổng hợp tiếng. Để giải quyết vấn đề thứ hai và ba hệ thống sẽ được triển khai theo mô hình client sever trong đó:

- Mô đun nhận dạng ASR: Mô đun nhận dạng tiếng nói sẽ được tách thành hai phần, phần Feature Extractor để trích ra các đặc trưng, mô đun này đặt ở client. Phần Recognizer để nhận các đặc trưng và tiến hành quá trình nhận dạng, mô đun này đặt ở sever. Dữ liệu truyền qua mạng là các đặc trưng tiếng nói do đó đã giải quyết được vấn đề thứ 2. Bài toán truyền các đặc trưng của tệp tin tiếng nói cũng đã được đề cập đến trong đề tài "Nghiên cứu và phát triển mô đun trích chọn đặc trưng trên thiết bị smartphone, ứng dụng trong nhận dạng tiếng nói" của Đỗ Đức Anh [18]. Nhưng cách xây dựng mô đun và lập trình chưa tối ưu. Do đó tôi sẽ xây dựng mô đun theo một thiết kế khác để có tốc độ xử lý tốt hơn, có thể được sử dụng lại và mở rộng bởi nhà phát triển khác. Trong mô đun này chúng tôi lựa chọn sử dụng bộ công cụ hỗ trợ CMU Sphinx [19] cho xây dựng chương trình và huấn luyện dữ liệu.
- Mô đun dịch tự động MT:Mô đun MT sử dụng phương pháp dịch máy thống kê và được đặt hoàn toàn trên sever. Trong mô đun này chúng tôi lựa chọn sử dụng bộ công cụ hỗ trợ Moses [20] cho việc xây dựng chương trình và huấn luyện dữ liệu.
- Mô đun Tổng hợp tiếng nói TTS: Trong mô đun TTS chúng tôi lựa chọn sử dụng bộ công cụ tổng hợp tiếng nói được Google tích hợp sẵn trên hệ điều hành Android cho tiếng Anh và cho tiếng Việt sử dụng bộ công cụ tổng hợp tiếng nói của viện nghiên cứu quốc tế MICA.

Vấn đề cuối cùng là thu thập bộ cơ sở dữ liệu cho hai mô đun nhận dạng và dịch:

- Cơ sở dữ liệu cho mô đun nhận dạng gồm hai phần nhận dạng tiếng Anh và nhận dạng tiếng Việt. Dữ liệu cho nhận dạng tiếng Anh được huấn luyện từ bộ dữ liệu TED-LIUM [21] với khoảng hơn 100 giờ ghi âm và khoảng 33000 từ vựng trong từ điểm âm vị. Dữ liệu cho nhận dạng tiếng Việt được huấn luyện dựa trên bộ dữ liệu

được trích ra từ dữ liệu VNSpeechCorpus của viện nghiên cứu quốc tế MICA với khoảng hơn 5h ghi âm và 3000 từ vựng.
- Cơ sở dữ liệu cho mô đun dịch được huấn luyện từ bộ dữ liệu gồm 3,8 triệu cặp câu song ngữ Anh Việt [22](được thu thập từ trang http://opus.lingfil.uu.se/) tương ứng là bản dịch của nhau trong từng ngôn ngữ. Bộ dữ liệu này được sử dụng để huấn luyện cho cả hai chiều Anh Việt và Việt Anh.

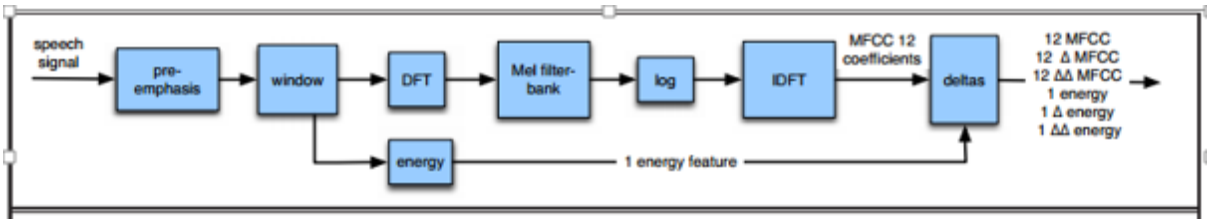
Trong các phần tiếp theo chúng ta sẽ tìm cách giải quyết vấn đề thứ hai và ba ở chương hai bằng cách xây dựng mô đun nhận dạng tiếng nói theo mô hình client-sever dựa trên việc truyền các đặc trưng. Ở chương ba chúng ta sẽ xây dựng cơ sở dữ liệu cần thiết cho hệ thống. Cuối cùng ta sẽ tìm cách hoàn thiện các thành phần hệ thống và ghép nối các mô đun ở chương bốn.

# CHƯƠNG 2: XÂY DỰNG MÔ ĐUN NHẬN DẠNG TIẾNG NÓI THEO MÔ HÌNH CLIENT-SERVER TRUYỀN NHẬN THAM SỐ ĐẶC TRƯNG

## 2.1 Phương pháp trích chọn đặc trưng MFCC

Trong phần trước chúng ta đã đề cập đến việc truyền các tham số đặc trưng trong mô hình dịch tiếng nói client-sever thay vì truyền trực tiếp dữ liệu tiếng nói, để đảm bảo thời gian và dung lượng truyền. Các tham số đặc trưng mà ta lựa chọn để truyền là các véc tơ đặc trưng MFCC của tiếng nói, đây là cách được sử dụng phổ biến để biểu diễn đặc trưng tiếng nói. Trong chương này ta sẽ tập trung vào cách để trích chọn các véc tơ đặc trưng MFCC từ dữ liệu tiếng nói đầu vào.

Phương pháp trích chọn đặc trưng MFCC dựa trên việc thực hiện chuyển đổi từ dữ liệu âm thanh đầu vào trong miền phổ về thang đo tần số Mel, một thang đo diễn tả tốt hơn sự nhạy cảm của tai người đối với tiếng nói [10]. Hình 5: mô tả các bước để tiến hành trích chọn đặc trưng MFCC. Thứ tự các bước lần lượt là Pre-emphasis, window, DFT, Mel filter bank, log, IDFT, deltas. Đầu ra của bước này là đầu vào của bước kế tiếp, đầu ra của quá trình là một véc tơ đặc trưng có 39 giá trị.

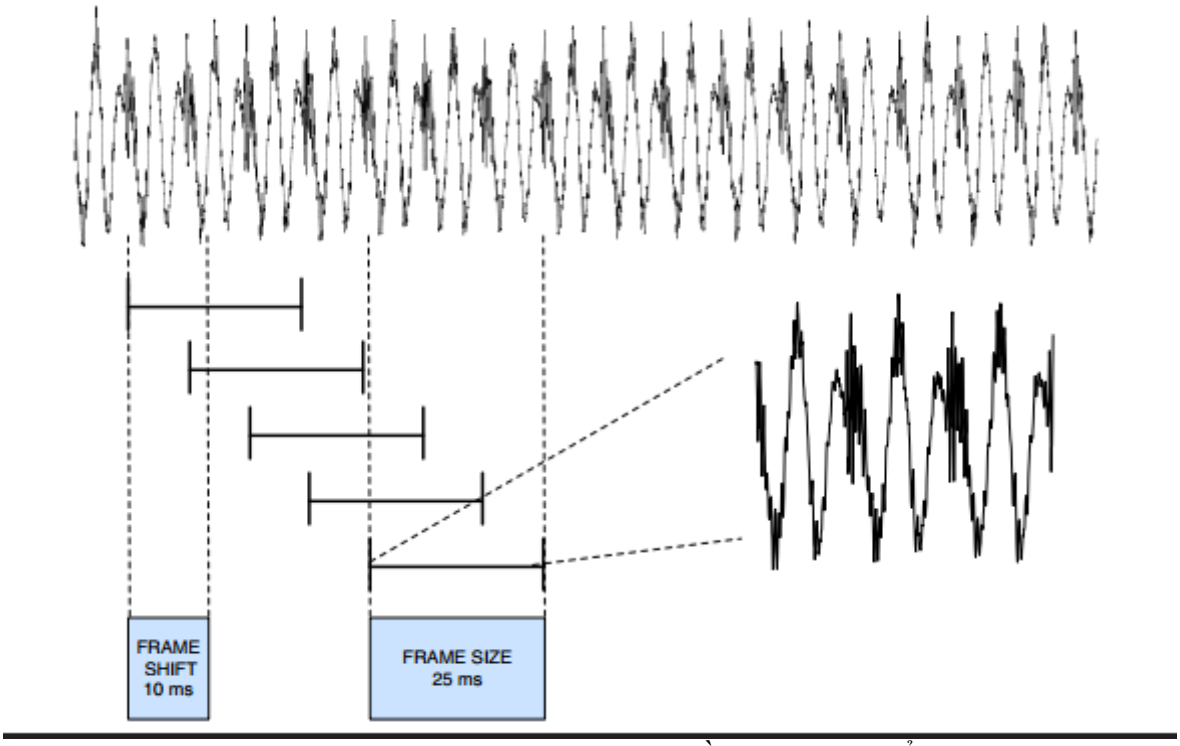



Hình 5: Quá trình trích chọn đặc trưng MFCC [10]

Pre-emphasis: do cấu trúc đặc biệt của môi trường thanh quản nên mức năng lượng ở tần số cao suy giảm hơn so với tần số thấp mà ở tần số cao thông tin về formant có nhiều giá trị cho mô hình âm học (mô hình âm học là một mô hình trong bộ ASR biểu diễn mối quan hệ giữa tín hiệu tiếng nói và các âm vị) do đó cần làm tăng năng lượng của tín hiệu ở tần số cao, quá trình này được gọi là pre-emphasis. Pre-emphasis cho tín hiệu tiếng nói đầu vào qua một bộ lọc: $H(z) = 1 - \alpha^{z-1}$ trong đó $0.9 < \alpha < 1$ .

Cửa sổ hóa (window): Tín hiệu tiếng nói biến đổi liên tục do đó các thông tin thu được cũng không phải là tinh trên toàn bộ dải tín hiệu do đó để có được những đặc trưng thống kê coi như không đổi ta trích tín hiệu trong một khoảng thời gian ngắn, ta sử dụng một cửa sổ nhỏ để làm việc này và tín hiệu được trích ra thành các frame, ta trượt cửa sổ trên toàn bộ tín hiệu tiếng nói ta được một loạt các frame. Thông thường ta sử dụng cửa sổ hamming:

$$w[n] = \begin{cases} 0.54 - 0.46\cos(\frac{2\pi n}{N}); 0 \le n \le N-1 \\ 0 \end{cases}$$




Hình 11: Trích các frame tính hiệu bằng hàm cửa sổ [10]

Hình 11 cho thấy các frame có độ rộng khoảng 25ms và mỗi cạnh trái của frame cách nhau 10ms, chúng sẽ được trượt cho đến khi hết tín hiệu.

Biến đổi DFT: Tính biến đổi fourier rời rạc của tín hiệu dựa vào công thức:

$$X[k] = \sum_{n=0}^{N-1} x[n] e^{-j2\pi kn/N}$$

Mel filter bank log: Tai nghe của con người cảm nhận sự thay đổi của âm thanh không tuyến tính (kém ở tần số cao đặc biệt trên 1000hz). Do đó để quan sát ta không sử dụng tần số thông thường mà dùng thang đo mel với tần số

Fmel = 2595log(1+Fhz/700)

Tai nghe con người thu nhận âm thanh như những bộ lọc và chỉ tập trung vào những tần số nhất định. Dựa vào đặc điểm này hệ mạch lọc tam giác được sử dụng để thu thập năng lượng trên mỗi frame. Trong miền tần số các tần số trung tâm của bộ lọc phân bố không tuyến tính, các thành phần <1000hz tập trung nhiều bộ lọc hơn vì nó chứa nhiều thông tin hơn. Ta sử dụng 20 đến 40 bộ lọc tam giác để thu thập năng lượng trên mỗi frame trong đó tần số trung tâm của mỗi bộ lọc này trong thang đo mel là tuyến tính cách đều do đó ta dễ dang xác định được vị trí của các bộ lọc, và độ rộng của bộ lọc tam giác kéo dài từ tần số trung tâm của bộ lọc phía trước tới tần số trung tâm của bộ lọc phía sau. Lấy tín hiệu cho đi qua các bộ lọc sau đó tính tổng các giá trị còn lại ta được một giá trị mi, với tất cả các bộ lọc ta được một chuỗi giá trị M = (m1,m2,..,mk).

Log: lấy log( $|m_i|^2$ ) tất cả các giá trị trong chuỗi giá trị vừa thu được.

IDFT: tính các hệ số cepstral C(i)

$$C(i) = \sum_{j=1}^{K} \log(|m_{i|}{}^2)\cos(i(j-\frac{1}{2})\frac{\pi}{K}$$

Ta lấy 12 hệ số C(i) đầu tiên ta được 12 hệ số MFCC, tinh $\Delta C(i)$ :

$$\Delta C(t) = \frac{\sum_{n=1}^{N} n(C_{t+n} - C_{t-n})}{2\sum_{n=1}^{N} n\text{^}2}$$

Thường chọn N= 2. Tiếp tục tinh $\Delta\Delta C$ bằng cách thay Ct bởi $\Delta C(t)$ ta được tổng cộng 36 hệ số đầu tiên của véc tơ đặc trưng MFCC. Tiếp theo ta tinh năng lượng

$$E = \sum_{t=t1}^{t2} x[t]^2$$

Tiếp theo tính $\Delta E$ và $\Delta\Delta E$ từ E qua đó ta đã thu thập đủ 39 giá trị của véc tơ đặc trưng MFCC.

## 2.2 Xây dựng mô đun truyền các tham số đặc trưng MFCC

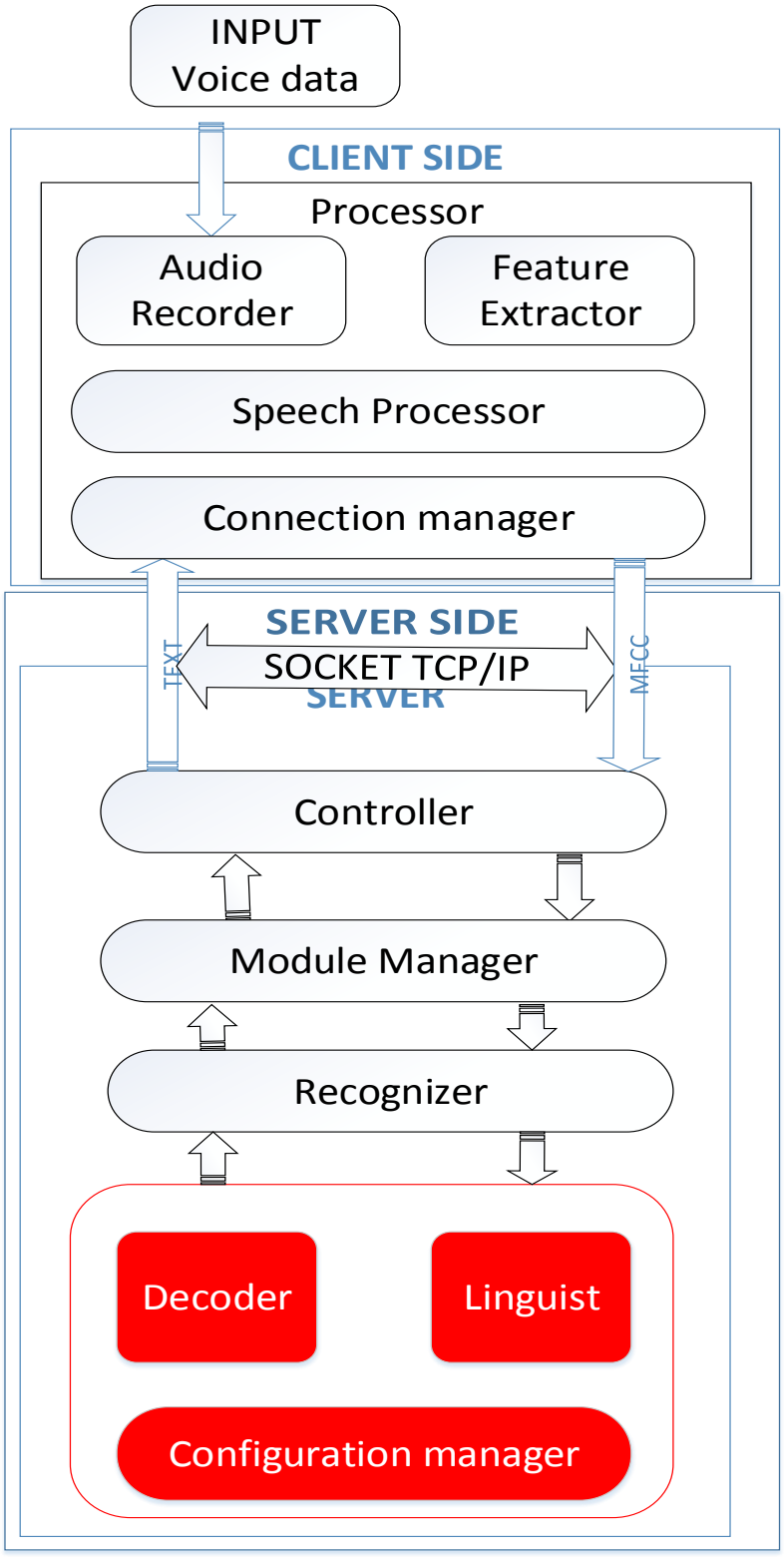
### 2.2.1 Kiến trúc của mô đun truyền nhận các tham số đặc trưng

Mô đun truyền nhận các tham số đặc trưng được xây dựng dựa trên mô hình client sever. Hình 12 mô tả kiến trúc các khối của mô đun truyền nhận các tham số đặc trưng MFCC.

Mô đun này được chia thành hai phần chính, phần client và phần sever. Phần client gồm có các khối: Audio Recorder, Feature Extractor, Speech Processor, Connection Manager. Phần sever gồm có khối Controller. Trong đó khối Audio Recorder chịu trách nhiệm ghi dữ liệu âm thanh, khối Feature Extrator chịu trách nhiệm. nhận dữ liệu tiếng nói được ghi lại bởi Audio Recorder và tiến hành trích chọn các véc tơ đặc trưng MFCC từ dữ liệu tiếng nói này. Khối Speech Processor là khối xử lý chính chịu trách nhiệm quản lý, xâu chuỗi và liên kết các hoạt động của các khối Audio Recorder, Feature Extractor và Connection Manager. Khối Connection Manager nhận các đặc trưng từ khối Speech Processor (khối Speech Processor sâu chuỗi và ghép nối các kết quả của các khối còn lại) gửi lên sever và nhận lại kết quả trả về cho Speech Processor. Phía sever khối Controller có nhiệm vụ gửi và nhận dữ liệu từ client,ngoài ra khối này còn là cổng giao tiếp với Module Manager để xử lý các đặc trưng. Bộ Module Manager chịu trách nhiệm quản lý các mô đun trong trường hợp này mô đun cần quản lý chỉ là bộ nhận dạng (trong hệ thống dịch bộ này còn quản lý cả bộ dịch). Bộ nhận dạng nhận các véc tơ đặc trưng được phân phối từ Module Manager tiến hành nhận dạng và trả lại kết quả.

Mô tả hoạt động của hệ thống: Dữ liệu tiếng nói được ghi âm bởi bộ Audio Recorder sẽ được ghi vào một hàng đợi, Bộ Feature Extractor lấy dữ liệu từ hàng đợi ra và tiến hành trích chọn các đặc trưng. Toàn bộ đặc trưng được trích chọn bởi bộ Feature Extractor sẽ được truyền cho bộ Connection Manager. Bộ Connection Manager gửi đặc trưng lên sever và nhận kết quả trả về. Toàn bộ quá trình này được thiết lập và quản lý bởi bộ Speech Processor. Ở phía sever sau khi nhận các đặc trưng từ client bộ Controller chuyển các đặc trưng cho Module Manager, Module

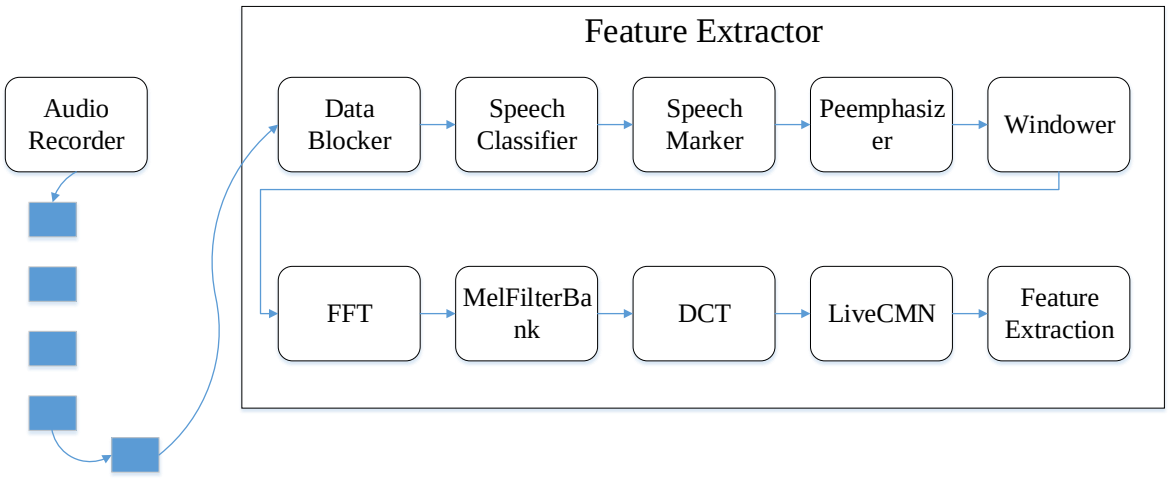
Manager lại chuyển đặc trưng cho bộ nhận dạng Recognizer. Recognizer nhận dạng tiếng nói sau đó trả kết quả cho Module Manager sau đó được chuyển về client bởi Controller. Điểm khác biệt trong hoạt động của hệ thống này so với hệ thống được xây dựng bởi Đỗ Đức Anh [18] là: trong hệ thống của Đỗ Đức Anh thì việc ghi âm và trích chọn đặc trưng được tiến hành tuần tự, còn trong hệ thống này thì hai công việc trên được tiến hành song song qua đó giảm thời gian tính toán.




Hình 12: Kiến trúc mô đun nhận dạng theo mô hình client sever

### 2.2.2 Khối ghi âm tiếng nói và trích chọn đặc trưng

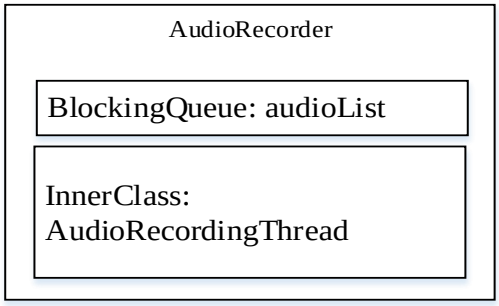
Khối ghi âm dữ liệu và khối trích chọn đặc trưng là Audio Recorder và Feature Extractor. Trong khối Feature Extractor có các đối tượng con được mô tả trong hình 13, trong đó các khối sẽ sâu chuỗi với nhau để lần lượt thực hiện các bước trích chọn đặc trưng. Lớp Data Blocker để đóng gói dữ liệu, Speech Classifier để phân lớp âm thanh thành hai phần tiếng nói và không phải tiếng nói. Speech Marker để đánh dấu điểm đầu cuối của âm thanh và điểm đầu cuối của tiếng nói. Các bộ Preemphasizer, Windower, FFt, MelFilterBank, DCT, LiveCMN, Feature Extraction lần lượt thực hiện các bước của phương pháp trích chọn đặc trưng MFCC. Các véc tơ đặc trưng đầu ra sẽ được lưu vào một mảng số thực hai chiều.




Hình 13: Quá trình trích chọn đặc trưng từ tiếng nói

Các lớp trong bộ Feature Processor và lớp AudioRecorder trong bộ Audio Recorder đều kế thừa từ một lớp trừu tượng là BaseDataProcessor. Trong lớp BaseDataProcessor có chứa một đối tượng của chính lớp này có tên là predecessor, trong mỗi lớp của bộ Feature Extractor thì predecessor chính là đối tượng của lớp phía trước nó (hay nói cách khác là khối phía trước). Ví dụ Speech classifier thì predecessor là DataBlocker. Chính đối tượng predecessor này giúp ta có thể ghép nối các lớp trong bộ FeatureExtractor với lớp AudioRecorder thành một chuỗi liên tục, trong đó các đối tượng của lớp phía sau chứa các đối tượng của lớp phía trước, điều này giúp cho việc thực hiện tuần tự việc trích chọn đặc trưng mà vẫn đảm bảo các khối có thể rút ra cắm vào một cách đơn giản.

Lớp AudioRecorder được mô tả trên hình 14, lớp này có chứa hai thành phần chính đó là một hàng đợi có tên audioList và một lớp con AudioRecordingThread.




Hình 14: Lớp AudioRecorder

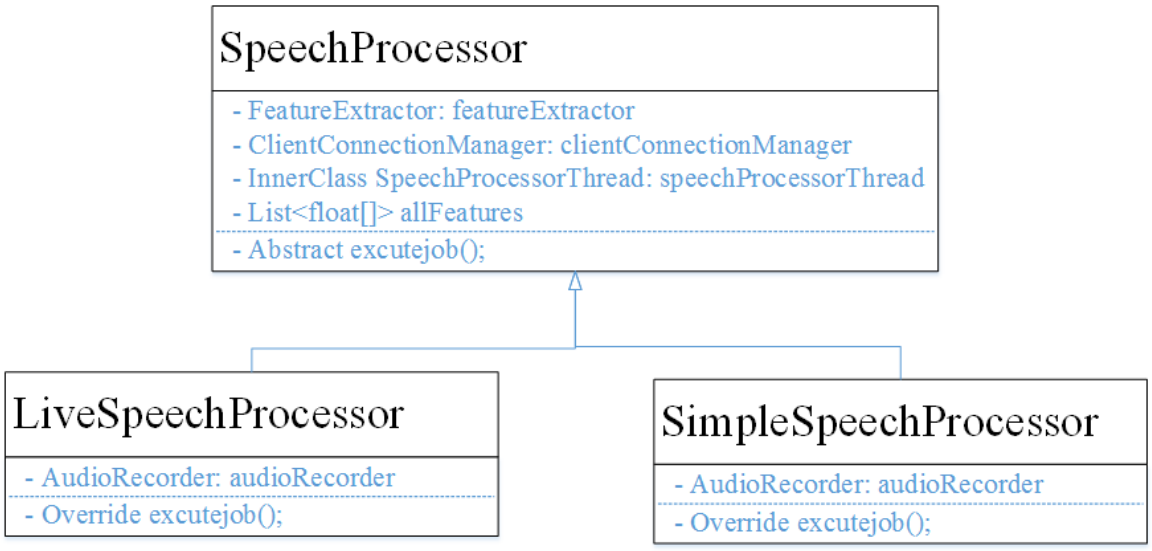
Lớp con AudioRecordingThread là một lớp để tạo ra một luồng mới phục vụ cho việc ghi âm, khi luồng này được khởi chạy dữ liệu tiếng nói sẽ được ghi lại vào hàng đợi audioList. Dữ liệu âm thanh ở hàng đợi được lấy ra bởi bộ DataBlocker trong FeatureExtractor, bộ DataBlocker lấy được dữ liệu từ AudioRecorder là vị AudioRecorder cũng tham gia vào chuỗi liên tiếp nêu trên và đối tượng predecessor trong DataBlocker chính là đối tượng của lớp AudioRecorder. Ngoài ra lớp AudioRecorder còn có hai phương thức startRecording() và stopRecording () để điều khiển việc bắt đầu và kết thúc ghi âm.

Do dữ liệu âm thanh được ghi lại trong một luồn riêng được tạo bởi AudioRecorder nên việc ghi âm dữ liệu và trích chọn đặc trưng được thực hiện song song.

### 2.2.3 Khối xử lý chính

Khối xử lý chính có biểu đồ lớp được mô tả trong hình 15. Trong đó lớp cha SpeechProcessor có chứa các đối tượng FeatureExtractor, ClientConnectionManager của khối ConnectionManager, một đối tượng của lớp SpeechProcessorThread và một đối tượng allFeatures chứa tất cả các véc tơ đặc trưng. Trong lớp SpeechProcessor có một phương thức trừu tượng excutejob(), phương thức này sẽ được các lớp con kế thừa lại và thực hiện theo cách riêng của nó.

Hai lớp con của khối xử lý chính (SpeechProcessor) là LiveSpeechProcessor và SimpleSpeechProcessor, hai lớp này chỉ khác nhau đôi chút là SimpleSpeechProcessor không có khả năng xác định các điểm bắt đầu và kết thúc của lời nói, còn LiveSpeechProcessor có khả năng này. Hai lớp con này khá giống nhau nên tôi sẽ chỉ nói về LiveSpeechProcessor.




Hình 15: Biểu đồ lớp của khối xử lý chính

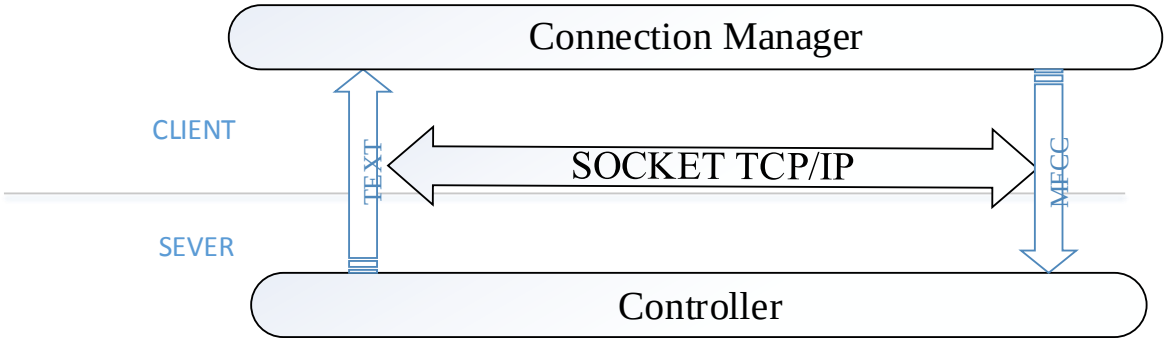
Hệ thống thực tế sẽ hoạt động dựa trên các đối tượng của lớp LiveSpeechProcessor hay SimpleSpeechProcessor. Phương thức excutejob() được kế thừa và ghi đè từ lớp cha, sẽ thực hiện việc gọi hàm start() trong AudioRecorder để bắt đầu ghi âm, và gọi hàm trích chọn đặc trưng để tính các đặc trưng và ghi vào đối tượng allFeatures.

Toàn bộ quá trình thực hiện bắt đầu từ ghi âm, trích chọn đặc trưng, gửi dữ liệu lên sever và nhận kết quả trả về sẽ được thực hiện trong một luồng riêng biệt được tạo ra bởi đối tượng speechProcessorThread của lớp SpeechProcessorThread. Trong lớp SpeechProcessorThread có phương thức run() được gọi mỗi khi luồng được tạo bởi lớp này bắt đầu hoạt động. Khi phương thức run() được gọi thì đầu tiên sẽ gọi hàm excutejob để ghi âm và tính các véc tơ đặc trưng, các véc tơ đặc trưng này được lưu trong đối tượng allFreatures, sau đó các véc tơ đặc trưng trong allFeatures sẽ được truyền vào đối tượng clientConnectionManager và những đặc trưng này sẽ được gửi lên sever. Kết quả trả về từ clientConnectionManager được lưu vào một biến mang tên result.

Khối này được thiết kế để có thể mở rộng, nếu muốn xây dựng một lớp xử lý mới thì có thể kế thừa lớp SpeechProcessor và ghi đè lại phương thức excutejob(); để thực hiện với mục đích riêng. Ở đây đối tượng audioRecorder được khai báo trong các lớp con là vì khi mở rộng chúng ta có thể thay nó bằng một lớp ghi âm khác hay theo một cách khác.

### 2.2.4 Khối truyền nhận dữ liệu

Khối truyền nhận dữ liệu gồm hai phần Connection Manager đặt ở client và Controller đặt ở sever. Connection Manager chịu trách nhiệm gửi toàn bộ các véc tơ đặc trưng lên sever và đợi sever gửi kết quả trả về, Controller chịu trách nhiệm đợi client gửi các đăc trưng lên, sau đó ghép nối với các mô đun khác tiến hành xử lý và gửi trả lại  kết quả cho client.




Hình 16: Sơ đồ khối bộ truyền nhận dữ liệu

Hình 16 mô tả sơ đồ khối của bộ truyền nhận gồm hai bộ phận như đã nêu ở trên. Dữ liệu được truyền từ Connection Manager đến Controller và ngược lại thông qua giao thức TCP/IP. Giải thuật client-sever để truyền nhận dữ liệu giữa hai khối này như sau:

Giải thuật cho sever:

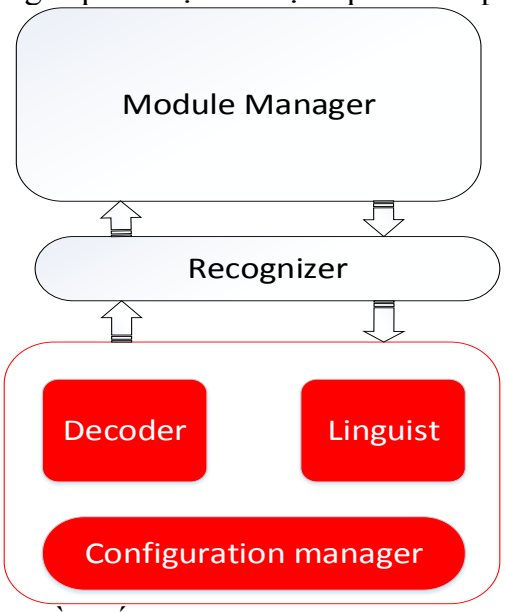
- Bộ Connection Manager tạo sever socket và đăng ký với hệ thống.
- Đặt socket vừa tạo ở chế độ chờ và lắng nghe kết nối.
- Khi có yêu cầu kết nối từ client, chấp nhận kết nối và tạo một luồng (Thread) con để xử lý. Quay lại trạng thái chờ lắng nghe kết nối mới.
- Công việc của luồng mới được tạo bao gồm: Nhận thông tin kết nối của client, nhận các véc tơ đặc trưng từ client và gọi các mô đun xử lý, trả kết quả về lại cho client, đóng kết nối và kết thúc luồng.

Giải thuật cho client:

- Xác định địa chỉ sever.
- Tạo socket.
- Gửi các véc tơ đặc trưng và chờ nhận kết quả trả về.
- Đóng socket.

### 2.2.5 Khối nhận dạng

Khối nhận dạng gồm hai thành phần: thành phần thứ nhất là bộ Recognizer và thành phần thứ hai là các bộ Decoder, Linguist, Configuration Manager. Các thành phần trong bộ thứ hai được cung cấp bởi bộ thư viện Sphinx 4-5prealpha.




Hình 17: Sơ đồ khối bộ quản lý mô đun và bộ nhận dạng

Bộ Recognizer có chức năng nhận dạng từ các véc tơ đặc trưng, chuyển hóa các véc tơ đặc trưng này thành dữ liệu dạng nhị phân (các véc tơ số thực) và giao tiếp với bộ thứ hai thông qua một giao diện lập trình được cung cấp bởi thư viện Sphinx 4. Khi nhận được kết quả nhận dạng ở bộ thứ hai thì sẽ trả về cho Module Manager.

Thành phần thứ hai là các bộ Decoder, Linguist, Congiuration Manager được cung cấp bởi bộ thư viện Sphinx 4-5prealpha.

## 2.3 Đánh giá so sánh hiệu năng của hệ thống giữa việc truyền các đặc trưng và truyền dữ liệu âm thanh

Để kiểm nghiệm giải pháp truyền các véc tơ đặc trưng MFCC từ client đến sever trong mô đun nhận dạng ASR của hệ thống dịch tiếng nói và so sánh với việc truyền trực tiếp dữ liệu tiếng nói, tôi đã xây dựng một chương trình thử nghiệm cho hệ thống truyền trực tiếp tiếng nói và một chương trình cho hệ thống truyền các véc tơ đặc trưng MFCC, cả hai chương trình này đều dựa trên mô hình client sever. Triển khai chương trình thử nghiệm với sever đặt trên một chiếc laptop với Ram 4gb và CPU core i5. Hai ứng dụng client cho việc truyền dữ liệu tiếng nói và truyền dữ liệu là các véc tơ đặc trưng được cài đặt trên thiết bị Android với Ram 1gb và CPU 4 core 1,3ghz.

### 2.3.1 Đánh giá thời gian nhận dạng

Phương pháp thử nghiệm: đọc 20 câu tiếng việt để thử nghiệm chương trình nhận dạng tiếng việt trên hai hệ thống nêu trên. Thời gian được tính từ khi người sử dụng dừng nói cho đến khi họ nhận được kết quả trả về.

Kết quả thử nghiệm bảng 1 cho thấy rằng việc truyền các đặc trưng MFCC cho tốc độ nhanh hơn:

| | Hệ thống truyền trực tiếp dữ liệu tiếng nói (ms) | Hệ thống truyền đặc trưng MFCC (ms) |
|---|---|---|
| Thời gian chạy trung bình | 2363 | 2212 |

Bảng 1: Kết quả so sánh hệ thống dịch tiếng nói dựa vào việc truyền nhận trực tiếp dữ liệu tiếng nói với việc truyền đăc trưng MFCC.

### 2.3.2 Đánh giá dung lượng dữ liệu truyền

Đánh giá dung lượng dữ liệu phải truyền từ client tới sever qua mạng của cả hai hệ thống. Bảng 2 cho thấy kết quả đo thử nghiệm dung lượng cần truyền của hai hệ thống và cho ta thấy được rằng việc truyền các đặc trưng tốn ít dung lượng hơn hẳn.

| Câu | Hệ thống truyền trực tiếp dữ liệu tiếng nói (byte) | Hệ thống truyền đặc trưng MFCC (byte) |
|---|---|---|
| Xin chào mọi người | 56360 | 25584 |
| Anh có khỏe không | 53800 | 24024 |
| Tối nay thế nào | 60200 | 27144 |
| Thời tiết hà nội ra sao | 64040 | 29016 |
| Mình đi chơi nhé | 43560 | 19344 |
| Trời đẹp thật | 35880 | 15600 |

Bảng 2: Kết quả đo dữ liệu cần truyền

Từ các kết quả thử nghiệm trên ta thấy rằng việc xây dựng hệ thống trên cơ sở truyền nhận các véc tơ đặc trưng MFCC sẽ giúp làm giảm thời gian truyền nhận (không nhiều lắm) và giảm tương đối dung lượng phải truyền.

# CHƯƠNG 3: THU THẬP BỘ CƠ SỞ DỮ LIỆU CHO HỆ THỐNG DỊCH TỰ ĐỘNG TIẾNG NÓI

## 3.1 Thu thập bộ dữ liệu cho mô đun nhận dạng tiếng nói

### 3.1.1 Chuẩn bị dữ liệu cho nhận dạng tiếng nói

Dữ liệu cần chuẩn bị cho việc huấn luyện mô hình nhận dạng tiếng nói bao gồm các thành phần sau: Các tệp tin ghi âm tiếng nói, tệp văn bản chứa nội dung các tệp tin ghi âm (nội dung của từng tệp ghi âm phải được móc nối đến chính xác tệp tin tiếng nói), tập tin văn bản để huấn luyện mô hình ngôn ngữ, bộ từ điển âm vị.

Trong đề tài này tôi đã chuẩn bị dữ liệu để xây dựng mô hình nhận dạng cho cả tiếng Anh và tiếng Việt. Dữ liệu chuẩn bị để huấn luyện mô hình nhận dạng tiếng Anh bao gồm: 1495 tệp tin âm thanh định dạng sph, 1495 tập tin nội dung định dạng stm, tệp tin văn bản để huấn luyện mô hình ngôn ngữ được lấy từ các tập tin nội dung sau khi dữ liệu âm thanh và các tập tin nội dung đã được tiền xử lý, từ điển âm vị được lấy từ bộ từ điển âm vị của TED-LIUM với 159848. Do các tệp tin âm thanh là các tệp lớn, thời gian gi âm dài và tệp tin stm là tệp chứa nội dung tệp ghi âm nhưng không phải toàn bộ mà là những mẩu nhỏ được đánh dấu thời gian bắt đầu và kết thúc cụ thể, do đó quá trình tiền xử lý ta phải cắt các tập tin âm thanh thành các tập nhỏ hơn tại các thời điểm bắt đầu và kết thúc của tập nhỏ hơn này được lấy từ tập tin stm, sau đó tập hợp các câu nội dung trong các tập stm thành một tập tin nội dung cụ thể, cuối cùng móc nối tập tin âm thanh với phần nội dung của nó trong tập tin nội dung. Toàn bộ 1495 tệp tin âm thanh, 1495 tệp tin nội dung dạng stm và từ điển âm vị được lấy từ bộ dữ liệu TED-LIUM [21]. Dữ liệu tiếng Việt được lấy từ bộ dữ liệu được trích ra từ bộ VNSpeechCorpus của viện nghiên cứu quốc tế MICA với khoảng hơn 5h ghi âm và 3000 từ vựng cho bộ từ điển âm vị.

### 3.1.2 Huấn luyện mô hình nhận dạng sử dụng công cụ sphinx

Việc huấn luyện mô hình nhận dạng này là giống nhau với các mô hình tiếng Anh và tiếng Việt do đó ở đây tôi chỉ trình bày cách huấn luyện chun cho hả hai mô hình này.

Sử dụng các công cụ có sẵn như IRSTLM, SRLIM,CMUCLMK để huấn luyện mô hình ngôn ngữ từ tập tin văn bản. Kết quả ta sẽ được mô hình ngôn ngữ định dạng arpa.

Sau khi có mô hình ngôn ngữ ta tiến hành huấn luyện mô hình âm học. Công cụ được lựa chọn ở đây là CMU Sphinx [19], tiến hành cài đặt bộ công cụ này theo hướng dẫn trên trang chủ của nó. Chạy lệnh sphinxtrain –t digit setup (digit là tên mô hình mà mình đặt). Khi chạy lệnh trên hai thư mục wav và etc sẽ được tạo Chuẩn bị các tập tin âm thanh để vào thư mục wav. Chuẩn bị các tệp sau để đưa vào thư mục etc:

- Tệp .dic là từ điển âm vị.
- Tệp tin .phone chứa các âm vị trong từ điển.
- Tệp tin mô hình ngôn ngữ.

- Tệp tin .filler chứa các từ và phiên âm của các từ cần lọc (có thể là nhiễu và âm im lặng).
- Tệp tin train.transcription là tập tin chứa nội dung các tệp ghi âm, cấu trúc các dòng trong tệp này như sau:
<s> khoong khoong moojt nghifn baary trawm boosn muwowi chisn trawm tasm hai nghifn nawm trawm sasu ba </s> (file1)
Phần nằm giữa <s> và </s> là nội dung tệp âm thanh có tên là "file1".
- Tệp tin train.fileids là tệp tin văn văn bản trong đó mỗi dòng của nó như sau:
"thinh/file1" là đường dẫn tương đối đến file1 lưu ý là mỗi dòng trong tên này phải chứa đường dẫn đến tệp âm thanh phải tương ứng với mỗi dòng trong tệp train.trainscription chứa nội dung tệp âm thanh.
- Tệp test.transcription và test.fileid cũng tương tự như train.transcription và train.fileids có điều các tệp này tham chiếu đến dữ liệu để thử nghiệm mô hình sau khi huấn luyện.

Sau khi tự tạo thư mục etc thì tệp sphinx-train.cfg sẽ được tự động tạo ra, ta cấu hình các tham số về quá trình huấn luyện, tham số mô hình, tham số dữ liệu và tham số giải mã trong tệp tin này. Tham khảo hướng dẫn trên trang chủ của CMU Sphinx để biết cách cầu hình cụ thể các tham số này.

Chạy lệnh "sphinxtrain run" để tiến hành huấn luyện mô hình. Kết quả mô hình được huấn luyện sẽ nằm trong các thư mục con của thư mục model-parameter. Sau khi huấn luyện ta đã có được đầy đủ dữ liệu cho chương trình bao gồm tệp tin từ điển âm vị, tệp tin mô hình ngôn ngữ và các tệp tin mô hình âm học được huấn luyện bởi CMU Sphinx.

## 3.2 Xây dựng bộ dữ liệu cho mô đun dịch tự động

Dữ liệu cần chuẩn bị bao gồm dữ liệu các cặp câu song ngữ và dữ liệu cho mô hình ngôn ngữ. Dữ liệu các cặp câu song ngữ bao gồm 3,8 triệu cặp câu song ngữ Anh Việt được thu thập từ trang http://opus.lingfil.uu.se/ [22]. Dữ liệu cho mô hình ngôn ngữ được lấy từ chính dữ liệu các cặp câu song ngữ.

Sau khi đã có đầy đủ dữ liệu ta tiến hành huấn luyện các mô hình dịch, trước tiên ta tiến hành huấn luyện mô hình ngôn ngữ bằng các công cụ SRILM, IRSTLM đã nêu ở trên. Tiếp theo ta tiến hành huấn luyện mô hình dịch sử dụng công cụ Moses [20].

Để huấn luyện mô hình dịch, đầu tiên ta tiến hành cài đặt Moses theo hướng dẫn trên trang chủ của bộ công cụ này. Sau đó tiến hành huấn luyện mô hình dịch, một kịch bản cho các bước huấn luyện mô hình dịch bằng Moses được nêu trong phần phụ lục A. Sau khi huấn luyện mô hình dịch ta sẽ được một tệp tin cấu hình tên là moses.ini và các tệp dữ liệu khác của mô hình, tệp moses.ini này chứa các cấu hình về mô hình và các đường dẫn đến các tệp tin dữ liệu của mô hình. Tệp moses.ini sẽ được sử dụng để khởi chạy hệ thống dịch tự động.

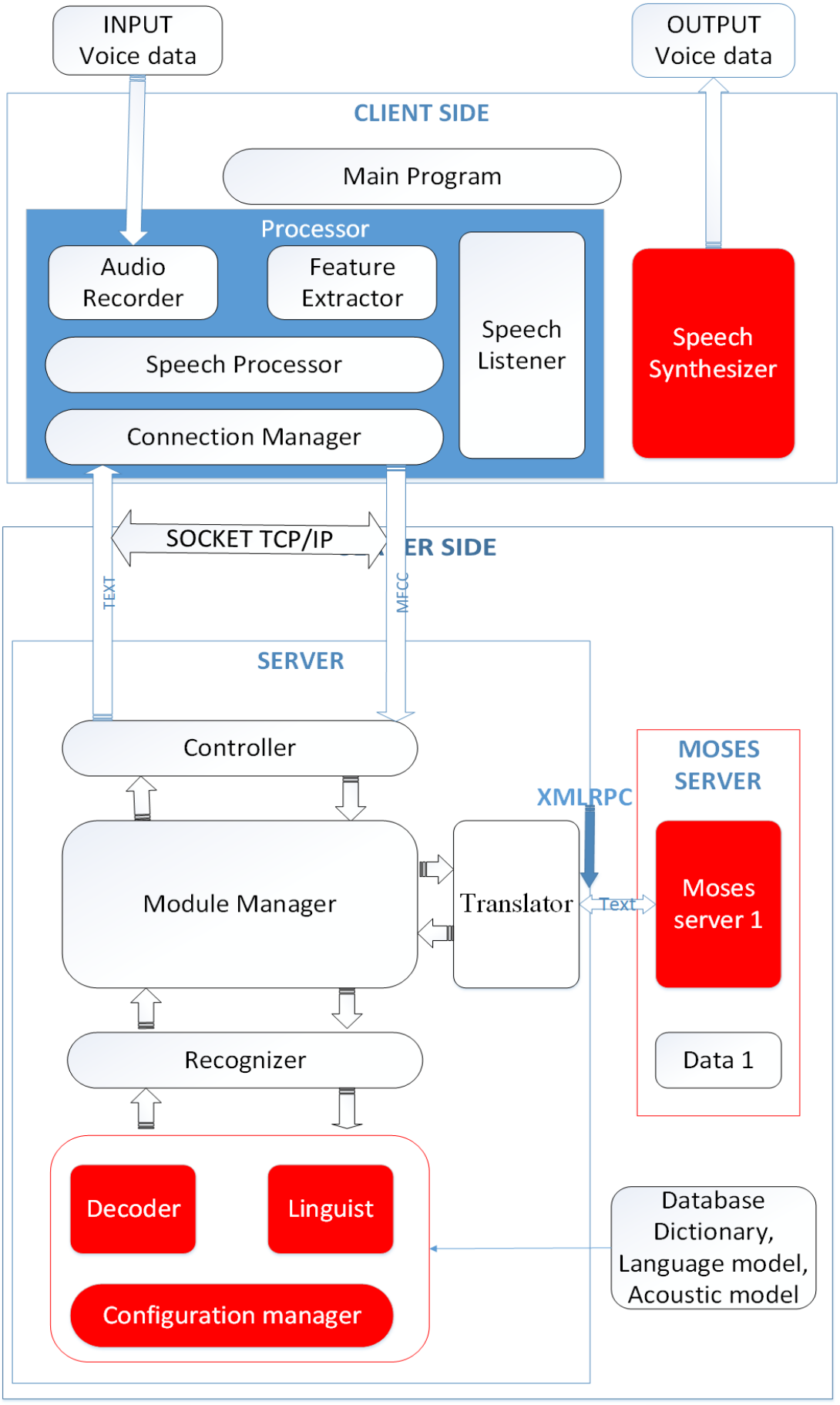
# CHƯƠNG 4: TRIỂN KHAI XÂY DỰNG HỆ THỐNG DỊCH TIẾNG NÓI

## 4.1 Kiến trúc tổng quan của hệ thống

Kiến trúc phần nhân chính của hệ thống dịch tiếng nói được mô tả trên hình 18. Hệ thống này bao gồm ba mô đun chính: Mô đun Nhận dạng được xây dựng theo mô hình client sever truyền các tham số đặc trưng, mô đun dịch tự động và mô đun tổng hợp tiếng nói. Mô đun nhận dạng đã được trình bày chi tiết trong chương 2, nó gồm các thành phần trong khối Processor phía client và phía sever thì là các bộ Controller, Module Manager, Recognizer, Decoder, Linguist, Configuration Manager. Khác với kiến trúc của chương trình trong chương hai, bộ Module Manager lần này không chỉ quản lý mô đun nhận dạng (Recognizer) mà còn quản lý cả mô đun dịch (Translator) và điều phối dữ liệu giữa các mô đun này. Mô đun dịch gồm các thành phần là Translator và Moses sever. Mô đun Speech Synthesizer được đặt ở client chịu trách nhiệm tổng hợp tiếng nói. Biểu đồ các lớp của hệ thống được trình bày trong phần phụ lục D và phụ lục E.

Mô tả hoạt động của hệ thống: Dữ liệu tiếng nói được ghi lại bởi bộ Audio Recorder, bộ Audio Recorder vào một hàng đợi. Bộ Feature Extractor lấy dữ liệu âm thanh trong hàng đợi và tiến hành trích chọn đặc trưng sau đó gửi các đặc trưng cho Speech Processor. Speech Processor gửi các đặc trưng lên sever thông qua Connection Manager. Ở sever, bộ Controller nhận dữ liệu là các véc tơ đặc trưng chuyển cho Module Manager. Module Manager chuyển các đặc trưng cho Recognizer sau đó bộ recognizer tiến hành nhận dạng tiếng nói từ các đặc trưng, dưới sự hỗ trợ của bộ thư viện sphinx4-5prealpha. Kết quả nhận dạng được trả lại cho Module Manager, bộ này sau đó chuyển kết quả nhận dạng cho Translator để tiến hành dịch văn bản. kết quả dịch được trả về cho Module Manager và Controller. Controller gửi kết quả dịch dạng văn bản về cho client. Bộ Speech Processor nhận kết quả dịch dạng văn bản và tạo ra một sự kiện để truyền kết quả này thông qua sự kiện đó. Các lớp thực thi nằm trong bộ Main Program của Speech Listener sẽ ghi đè lại các hàm xử lý sự kiện này và nhận kết quả trả về thông qua nó. Các lớp thực thi nhận được kết quả dịch dạng văn bản và gọi chức năng tổng hợp tiếng nói (Text to Speech) của bộ Speech Synthesizer để tạo ra tiếng nói trong ngôn ngữ đích.

Toàn bộ các thành phần ở client trong kiến trúc nêu trên đều được viết thành một thư viện có tên là SpeechTranslationLib. Thư viện này cung cấp một giao diện lập trình đơn giản và dễ triển khai, lớp triển khai chỉ cần khởi tạo hai đối tượng, một đối tượng của lớp LiveSpeechProcessor (đã nói ở chương 2, ta cũng có thể khởi tạo các đối tượng khác kế thừa từ lớp SpeechProcessor tùy theo chương trình), và đối tượng thứ hai là đối tượng của lớp SimpleTextToSpeech của bộ SpeechSynthesizer sau đó gọi phương thức startListenning() của LiveSpeechProcessor. Để nhận kết quả trả về lớp thực thi cần kế thừa SpeechListener và triển khai các phương thức thực thi tương ứng với các sự kiện được sinh ra, trong đó quan trọng nhất là triển khai phương thức onResult, kết quả dịch dạng văn bản sẽ được trả về trong phương thức này. Nhận kết quả trả về và gọi chức năng tạo tiếng nói từ văn bản của đối tượng thực thi cho lớp SimpleTextToSpeech.

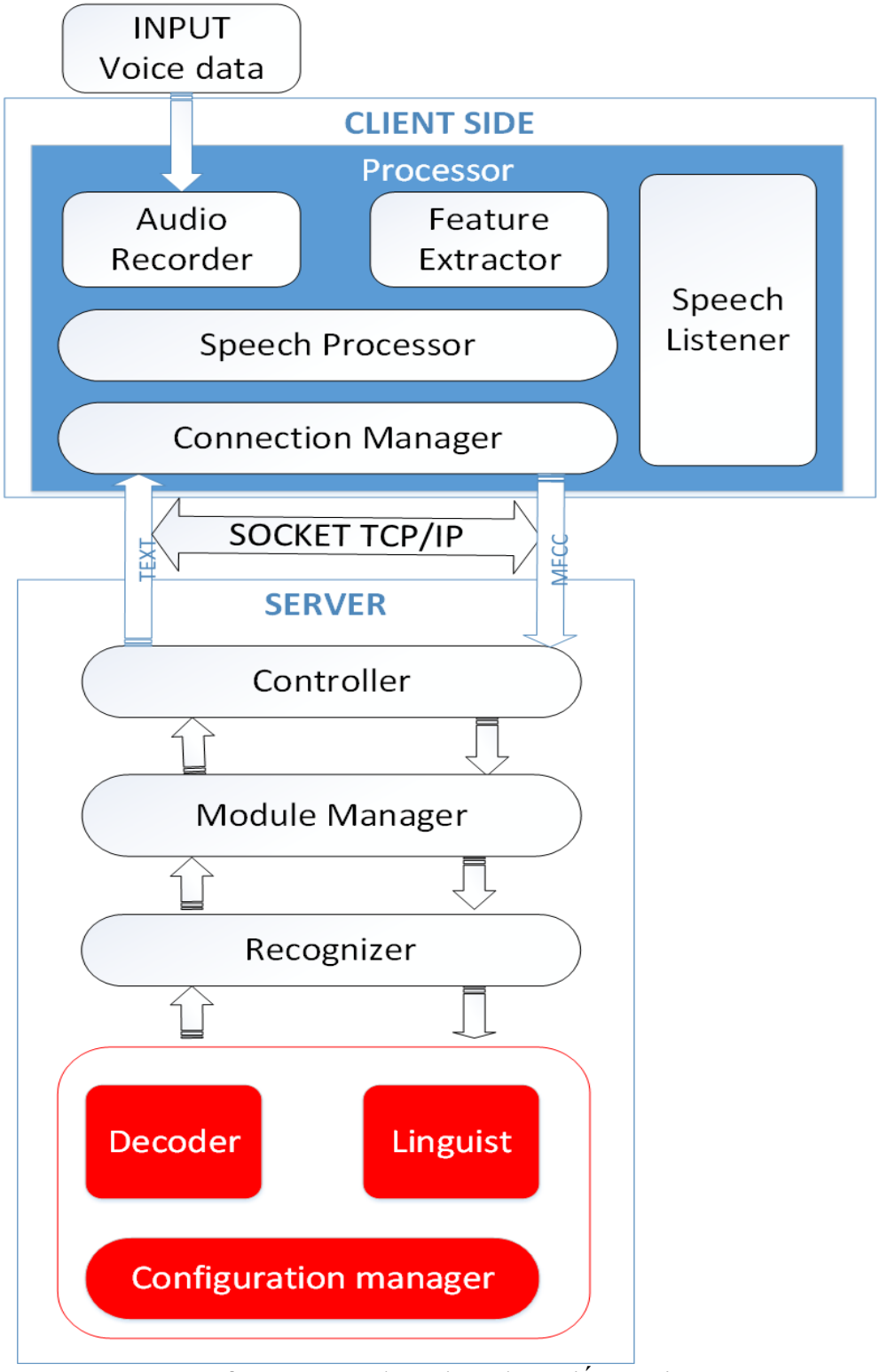



Hình 18: Kiến trúc hệ thống dịch tiếng nói

## 4.2 Xây dựng và ghép nối các mô đun

### 4.2.1 Mô đun nhận dạng tiếng nói

Mô đun nhận dạng tiếng nói trong hệ thống ta đang xây dựng được kế thừa từ mô đun nhận dạng tiếng nói theo mô hình client-sever truyền nhận tham số đặc trưng mà ta xây dựng trong chương 2, hình 19 mô tả kiến trúc mô đun này.



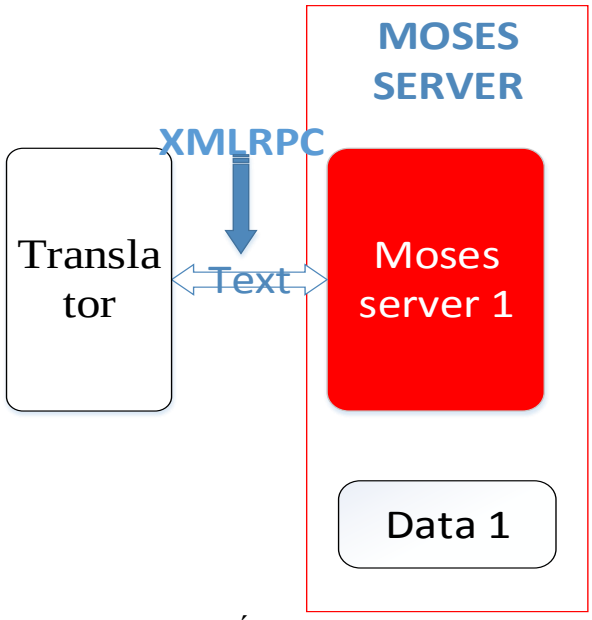

Hình 19: Mô đun nhận dạng tiếng nói

Trong phần này có thêm một số phần mở rộng và thay đổi so với trước đó, ở phía client có thêm một bộ là Speech Listener, bộ Speech Listener này được xây dựng để các lớp triển khai nó (hay kế thừa) có thể lắng nghe các sự kiện trong quá trình dịch tiếng nói như là: BeginOfSpeech bắt đầu nói, EndOfSpeech, kết thúc nói, Process các trạng thái của hệ thống trong quá trình xử lý, Result xảy ra khi có kết quả trả về.

Ngoài ra một thay đổi nữa là bộ Module Manager không còn quản lý duy nhất Bộ nhận dạng (Recognizer) mà còn quản lý thêm bộ dịch (Translator). Bộ Module Manager này chính là điểm ghép nối mô đun nhận dạng và mô đun dịch.

### 4.2.2 Mô đun dịch văn bản

Mô đun dịch văn bản được tích hợp vào hệ thống bao gồm hai phần Translator và Moses sever. Moses sever có thể triển khai trên một thiết bị độc lập so với sever hiện tại và giao tiếp với bộ Translator thông qua giao thức XMLRPC. Hình 20 thể hiện kiến trúc mô đun dịch.




Hình 20: Kiến trúc mô đun dịch

Moses là một hệ thống dịch tự động mã nguồn mở cho phép tự huấn luyện mô hình dịch cho bất cứ cặp ngôn ngữ nào [20]. Moses cũng được tích hợp bộ giải mã, bộ này có thể dịch một đoạn văn bản dựa trên mô hình dịch mà được huấn luyện dựa trên chính hệ thống Moses này. Để xây dựng Moses sever trước tiên cần cài đặt moses, sau khi cài đặt thì tiếp tục chuẩn bị dữ liệu và huấn luyện mô hình dịch (điều này được trình bày kỹ trong chương trước). Sau khi cài moses và có đầy đủ mô hình dịch cho cặp ngôn ngữ Anh Việt, ta tiến hành khởi chạy Moses như một sever. Khi khởi tạo sever chúng ta có thể cấu hình hai thông số --sever để sử dụng Moses như một sever và –sever-port để xác định cổng mà tại đó moses sever lắng nghe các yêu cầu được gửi đến (trong phần phụ lục B có trình bày một kịch bản mẫu để khởi chạy moses sever).

Để truy cập đến Moses sever, thì một yêu cầu theo giao thức XMLRPC cần được gửi tới http://host:port/RPC2 trong đó host là địa chỉ của thiết bị cài đặt Moses.

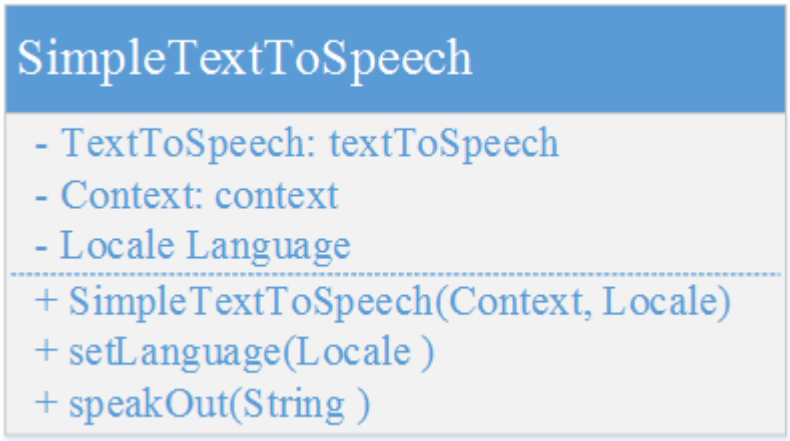
Tham số truyền khi gửi yêu cầu đến Moses sever cần có hai thành phần, thành phần thứ nhất chính là văn bản mà ta muốn dịch trong ngôn ngữ nguồn, và thành phần thứ hai là một tùy chọn (nếu là true thì sẽ được xem xét gửi thông tin về dóng hàng trong kết quả). Kết quả tra về cùng gồm hai thành phần: thành phần thứ nhất là kết quả dịch trong ngôn ngữ đích và thành phần thứ hai nếu có thể sẽ là thông tin về dóng hàng.

Bộ Translator chính là một thực hiện của bên gửi yêu cầu đến Moses sever theo giao thức XMLRPC. Trong Translator chứa các thuộc tính về địa chỉ Moses sever và cổng mà Moses sever đang lắng nghe. Và phần quan trọng nhất trong Translator là phương thức translate(). Phương thức này nhận đầu vào là đoạn văn bản cần dịch, khởi tạo các tham số, gửi yêu cầu dịch đến Moses sever, nhận kết quả trả về và trả lại kết quả dịch.

### 4.2.3 Mô đun tổng hợp tiếng nói

Mô đun tổng hợp tiếng nói sử dụng bộ công cụ tổng hợp tiếng nói của Google tích hợp sẵn cho hệ điều hành Android cho tổng hợp tiếng Anh và bộ công cụ tổng hợp tiếng Việt được phát triển bởi viện nghiên cứu quốc tế MICA.

Hệ thống dịch tiếng nói giao tiếp với các công cụ tích hợp sẵn này thông qua một API (giao diện lập trình ứng dụng). Trong mô đun tổng hợp tiếng nói của hệ thống này chúng tôi có xây dựng một lớp SimpleTextToSpeech dựa trên API được cung cấp. Hình 21 biểu diễn lớp SimpleTextToSpeech




Hình 21: Lớp SimpleTextToSpeech

Trong lớp SimpleTextToSpeech thì TextToSpeech là một lớp được cung cấp sẵn để giao tiếp với các công cụ tổng hợp tiếng nói được cài đặt sẵn, Context là lớp môi trường, đối tượng language của lớp Locale để xác định ngôn ngữ sẽ nói, trong trường hợp này ta sử dụng các thẻ ngôn ngữ “vi’ cho tiếng Việt và “en” cho tiếng Anh.

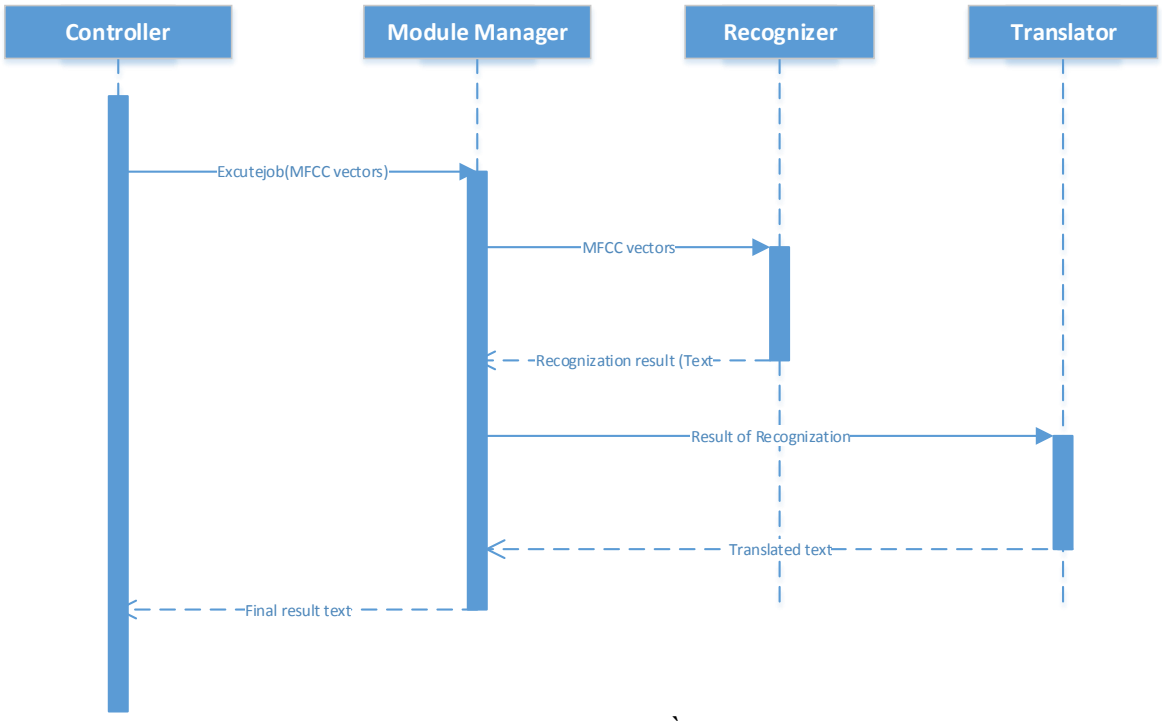
Phương thức khởi tạo của lớp trên cần có hai đối số truyền vào là môi trường (Context) và ngôn ngữ sẽ nói (Locale). Phương thức speakOut(String text) là phương thức quan trọng nhất để nói đoạn văn bản truyền vào. Ngoài ra ta có thể chỉ định một ngôn ngữ cụ thể thông qua hàm Locale.

### 4.2.4 Ghép nối các mô đun

Việc ghép nối ba mô đun chính lại với nhau chia làm hai phần: Phần thứ nhất ghép nối mô đun dịch và nhận dạng, điểm ghép nối đặt trên sever. Phần thứ hai là ghép nối mô đun tổng hợp tiếng nói với hai mô đun nhận dạng và dịch đã ghép nối trước đó, điểm ghép nối đặt tại client.

Dựa trên kiến trúc hệ thống hình 18 ta thấy rằng điểm kết nối hai mô đun dịch tự động và nhận dạng là bộ Module Manager. Module Manager không còn quản lý duy nhất Bộ nhận dạng (Recognizer) mà còn quản lý thêm bộ dịch (Translator). Trong lớp ModuleManager (thuộc bộ Module Manager) có xây dựng một phương thức là excutejob() phương thức này nhận đầu vào là tập các véc tơ đặc trưng sau đó chuyển các véc tơ đặc trưng này cho bộ Recognizer, sau khi có kết quả nhận dạng nó chuyển kết quả này cho bộ Translator để dịch kết quả trả về cuối cùng của phương thức này là kết quả dịch dạng văn bản. phương thức excutejob() là phương thức chính điều phối việc dịch và nhận dạng.

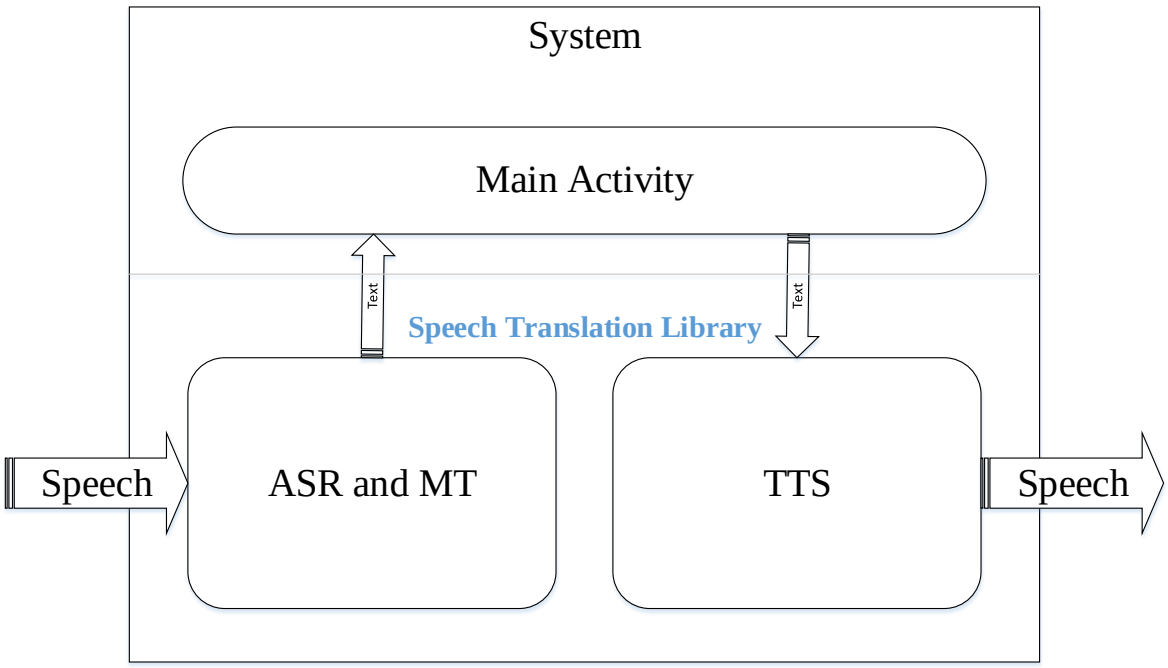
Hình 22 mô tả quá trình dịch tiếng nói trên sever từ khi nhận dữ liệu đầu vào là các véc tơ đặc trưng của tiếng nói đến khi có được kết quả cuối cùng.




Hình 22: Quá trình nhận dạng và dịch từ đầu vào là các véc tơ đặc trưng

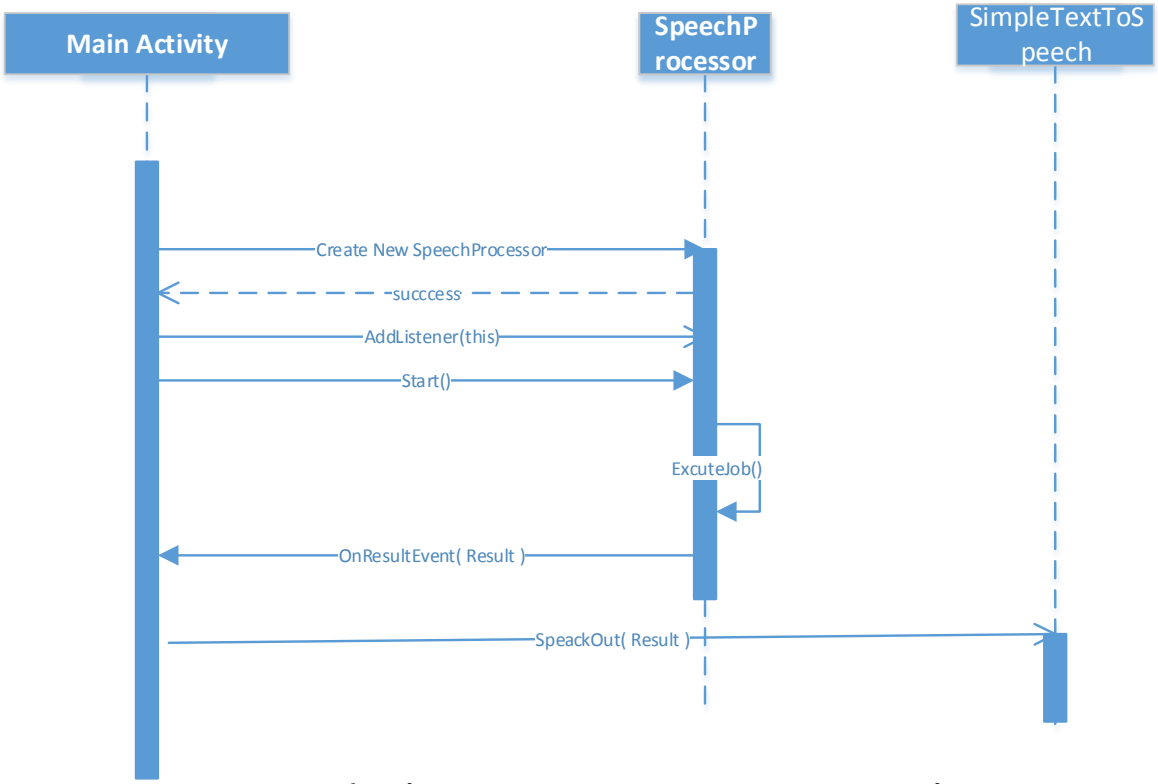
Phần tiếp theo chúng ta sẽ nói đến việc ghép nối mô đun tổng hợp tiếng nói với hai mô đun còn lại. Như ta đã nói, toàn bộ các bộ phận phía client trong kiến trúc hình 18 đều được tích hợp lại dưới dạng một thư viện lập trình và trong đó mô đun tổng hợp tiếng nói vẫn là tách rời và độc lập với hai mô đun còn lại, do đó việc ghép nối mô đun này được diễn ra khi tích hợp thư viện vào chương trình và xây dựng lô gic chương trình. Hình 23 mô tả việc kết nối mô đun tổng hợp với phần còn lại của hệ

thống. Trong đó Main Activity là lớp thực thi trong bộ Main Program (xem kiến trúc hình 18).




Hình 23: Hệ thống tích hợp ba mô đun

Quá trình hoạt động: đầu tiên dữ liệu tiếng nói sẽ được nhận dạng và dịch thông qua hai bộ ASR và MT, kết quả dịch này sẽ được gửi cho Main Activity thông qua sự kiện onResult(). Chi tiết quá trình hoạt động được trình bày trong biểu đồ hình 24:




Hình 24: Biểu đồ quá trình hoạt động chung của hệ thống

Lớp MainActivity của bộ Main Program phải kế thừa Interface SpeechListnener để có thể lắng nghe và xử lý các sự kiện. Sau khi khởi tạo SpeechProcessor và thêm bộ

lắng nghe sự kiện là chính đối tượng của lớp MainActiviy thì vào trạng thái chờ đợi kết quả. Bên phía SpeechProcessor sau khi trích chọn đặc trưng gửi lên sever và nhận được kết quả trả về sẽ tạo ra một sự kiện ResultEvent, khi đó MainActivity sẽ bắt sự kiện này và xử lý. Kết quả dịch và nhận dạng được gửi về từ sever sẽ được truyền như là một tham số của sự kiện này. Sau khi có được kết quả thì MainActivity sẽ gọi phương thức speakOut(String ) của đối tượng simpleTextToSpeech (đây là đối tượng của lớp SimpleTextToSpeech) để tổng hợp tiếng nói từ kết quả dịch nhận được.

# CHƯƠNG 5: THỬ NGHIỆM VÀ ĐÁNH GIÁ HOẠT ĐỘNG CỦA HỆ THỐNG

## 5.1 Cài đặt thử nghiệm

Cài đặt chương trình gồm hai phần: phần một cài đặt và khởi động sever, phần hai là cài đặt chương trình client trên thiết bị di động sử dụng hệ điều hành Android.

Cài đặt chương trình sever:

- Sao chép chương trình đã được biên dịch vào một thư mục trong sever
- chuyển vào thư mục nơi sao chép chương trình khởi động sever bằng dòng lệnh:

java SpeechTranslationSever

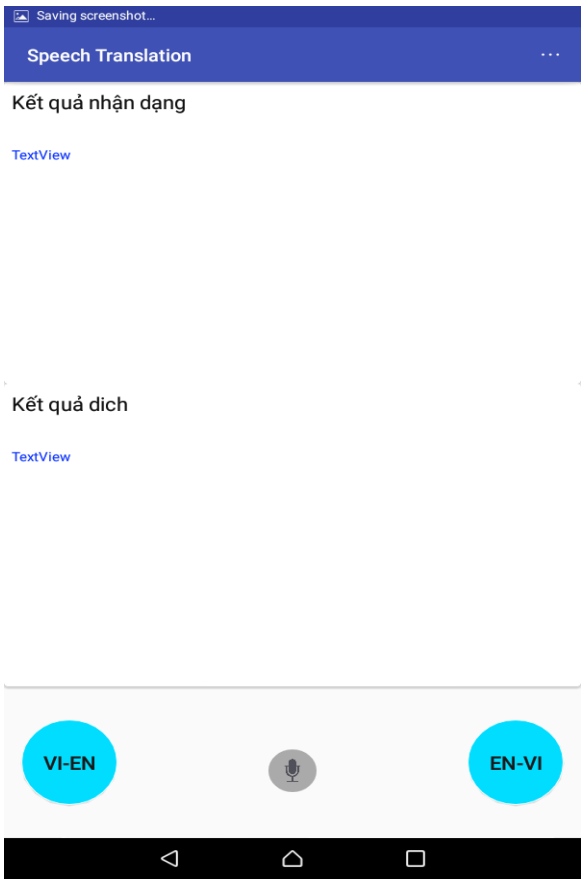
- Với dòng lệnh trên sẽ khởi động mô hình mặc định được cài đặt sẵn, để sử dụng các mô hình dịch và nhận dạng khác mà mình tự huấn luyện ta có thể thêm các tùy chọn sau vào dòng lệnh:
    + -m : đường dẫn đến mô hình âm học được huấn luyện bằng Sphinx.
    + -d : đường dẫn đến tệp tin từ điển âm vị.
    + -l : đường dẫn đến tệp tin mô hình ngôn ngữ.
    + --port : cổng mà tại đó sever lắng nghe yêu cầu từ client.
    + --mosesport : cổng mà tại đó Moses sever lắng nghe yêu cầu.
    + --mosessever : địa chỉ của Moses sever.

Một kịch bản mẫu cho việc khởi chạy sever được viết trong phụ lục C.

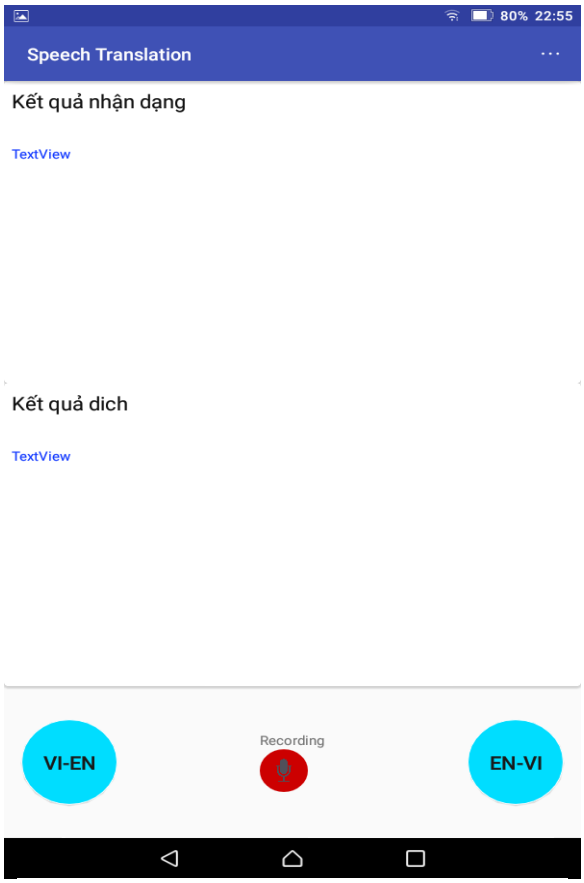
Cài đặt chương trình client:

- Sao chép tệp tin apk (tệp tin này được biên dịch và đóng gói từ chương trình ở client) vào bộ nhớ thiết bị.
- Chọn tệp tin apk và tiến hành cài đặt theo các bước của hệ thống.
- Ở lần đầu sử dụng, mở chương trình cài đặt lên vào phần Settings để cài đặt địa chỉ ip và cổng mà sever lắng nghe.

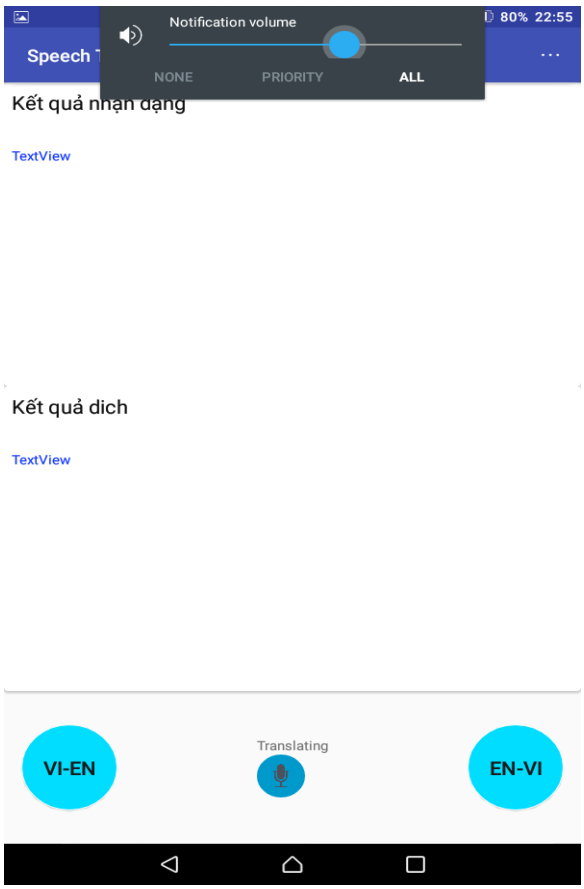
Sau đây là một số hình ảnh của chương trình khi cài đặt thực tế:



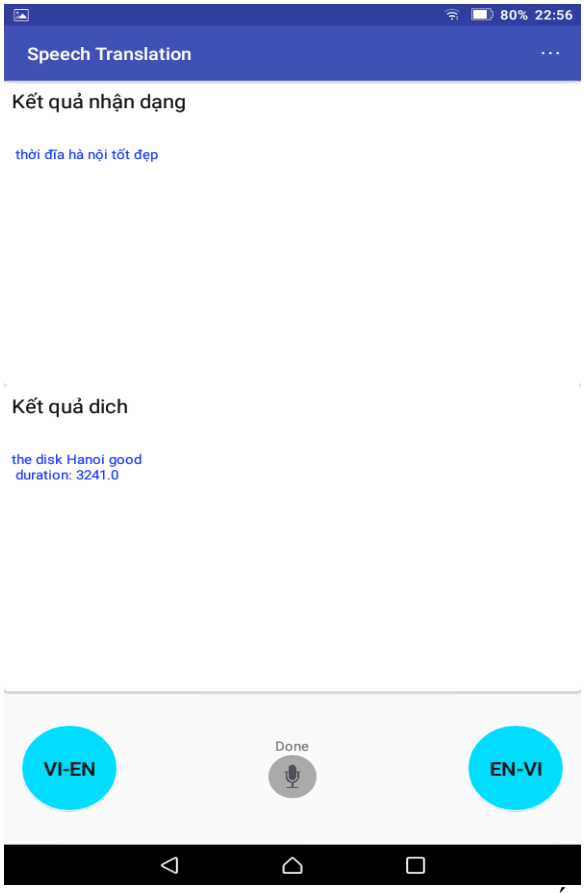

Hình 25: Chương trình sau khi khởi động




Hình 26: Chương trình khi đang ghi âm




Hình 27: Chương trình khi đang chờ sever dịch




Hình 28: Chương trình sau khi nhận được kết quả

## 5.2 Đánh giá chất lượng

Chất lượng nhận dạng được đánh giá bằng cách sử dụng một tập dự liệu gồm các tệp tin ghi âm và tệp tin chứa nội dung tệp ghi âm đó (giống như khi huấn luyện), sử dụng tệp tin ghi âm để đưa vào nhận dạng và so sánh với tệp tin nội dung sau đó tính tỉ lệ từ lỗi bằng công thức sau:

$$WER = \frac{100(Ins + Del + Sub)}{N_{word}}$$

Trong đó Nword: là tổng số từ trong văn bản, Ins là sô từ bị thêm vào, Del là số từ bị mất đi khi nhận dạng, Sub là số từ nhận dạng sai.

Chất lượng dịch được đánh giá bằng cách sử dụng hai tệp tin văn bản chứa các cặp câu song ngữ. Từ hai tệp tin này tiến hành cho vào dịch và tính điểm số BLEU của mô hình. Điểm BLEU được tính bằng công thức sau:

$$\begin{cases} score = \exp\left\{\sum_{i=1}^{N} w_i \log(p_i) - \max\left(\frac{L_{ref}}{L_{tra}} - 1, 0\right)\right\} \\ P_i = \frac{\sum_j NR_j}{\sum_j NT_j} \end{cases}$$

Trong đó:

- $NR_j$: là số lượng các n-grams trong phân đoạn *j* của bản dịch dùng để tham khảo.
- $NT_j$: là số lượng các n-grams trong phân đoạn *j* của bản dịch bằng máy.
- $w_i = N^{-1}$
- $L_{ref}$: là số lượng các từ trong bản dịch tham khảo, độ dài của nó thường là gần bằng độ dài của bản dịch bằng máy.
- $L_{tra}$: là số lượng các từ trong bản dịch bằng máy.

Trong chương trình cúng tôi xây dựng :
- Để đánh giá tỉ lệ từ lỗi WER cho mô hình nhận dạng tiếng Anh chúng tôi sử dụng tập dữ liệu gồm có 1155 tệp tin ghi âm, và tệp tin chứa nội dung các tệp ghi âm này.
- Để đánh giá tỉ lệ từ lỗi WER cho mô hình nhận dạng tiếng Việt chúng tôi sử dụng tệp dữ liệu gồm có 444 tệp tin ghi âm, và tệp tin chứa nội dung các tệp ghi âm này.
- Để đánh giá điểm BLEU của mô hình dịch Anh Việt và Việt Anh chúng tôi sử dụng tệp dữ liệu gồm có 150000 cặp câu song ngữ để đánh giá.

Kết quả đánh giá các mô đun nhận dạng và dịch được trình bày trong bảng 3

| | **WER**<br>(Cho nhận dạng) | **BLEU**<br>(Cho dịch văn bản) |
|---|---|---|
| Dịch tiếng nói Anh Việt | 28% | 46,7 |
| Dịch tiếng nói Việt Anh | 11,2% | 46,61 |

Bảng 3: Kết quả đánh giá chất lượng các mô đun trong hệ thống

Kết quả trên đây cho thấy độ chính xác của hệ thống còn hạn chế, so với những sản phẩm đã được phát triển nổi tiếng trên thế giới thì kém hơn khá nhiều. Ví dụ tỷ lệ từ lỗi WER cho nhận dạng tiếng Anh hiện nay của Google là 4,9 % [23], độ chính xác

của hệ thống phụ thuộc nhiều vào độ lớn của dữ liệu : Google dùng hàng triệu câu để huấn luyện mô hình dịch, để huấn luyện mô hình nhận dạng họ dùng tới hơn 500 giờ ghi âm. Kết quả nhận dạng tiếng Anh kém hơn tiếng Việt là vì số lượng từ vựng của mô hình nhận dạng tiếng Anh lớn hơn số lượng từ vựng của mô hình nhận dạng tiếng Việt rất nhiều, và tính động lập người nói cao hơn.

## 5.3 Đánh giá hiệu năng

Đánh giá hiện năng của hệ thống sẽ được chia làm hai phần chính là đánh giá thời gian phản hồi và đánh giá tài nguyên chiếm dụng. Về môi trường thực hiện đánh giá, Chương trình sever được cài đặt trên một máy tính hiệu năng cao với CPU core i5, RAM 32GB. Chương trình client được cài đặt trên thiết bị Android Ram 1gb và CPU 4 core 1,3ghz.

Đánh giá thời gian dịch tiếng nói: Đánh giá dựa trên một tập dữ liệu gồm 20 câu có độ dài ngắn khác nhau cho chiều dịch Việt Anh. Bàng 4 thế hiện kết quả thử nghiệm này.

| Độ dài câu (từ) | Thời gian dịch trung bình (ms) |
|---|---|
| 2 | 935 |
| 4 | 3596 |
| 6 | 3700 |
| 8 | 4313 |
| 18 | 5391 |
| Trung bình | 3716 |

Bảng 4: Kết quả thử nghiệm giá thời gian dịch chiều Việt Anh

Thời gian phản hồi của hệ thống được tính từ lúc người đọc ngừng nói cho đến khi có kết quả trả về.

| Thời điểm | CPU (%) | Bộ nhớ trong (MB) |
|---|---|---|
| Thời điểm hệ thống đang ở trạng thái bình thường | 0 | 5.47 |
| Trong khi ghi âm | 11.2 | 5.80 |
| Chờ kết quả dịch từ sever | 0 | 6.07 |
| Phát ra tiếng nói | 0.43 | 6.08 |

Bảng 5: Kết quả đo chiếm dụng tài nguyên hệ thống của chương trình

# CHƯƠNG 6: KẾT LUẬN

## 6.1 Kết quả đạt được

Trong nghiên cứu này tác giả đã hoàn thành mục tiêu của đề tài là làm chủ được công nghệ, thiết kế và xây dựng được hệ thống dịch tiếng nói hai chiều trên nền tảng di động, dựa trên mô hình ba mô đun, đưa vào một số cải tiến như triển khai xây dựng hệ thống trên mô hình client sever dựa trên việc truyền nhận đặc trưng của dữ liệu tiếng nói. Hệ thống đã được triển khai thực tiễn và ổn định bước đầu cho thấy hiệu quả về thời gian chạy tốt hơn thông qua việc truyền các đặc trưng so với truyền trực tiếp dữ liệu tiếng nói. Chương trình xây dựng phía sever có thể dễ dàng cài đặt và khởi chạy trên các thiết bị khác nhau chỉ bằng dòng lệnh. Chương trình phía client có phần lõi chính được lập trình ở dạng thư viện có thể triển khai mở rộng ở nhiều chương trình khách nhau. Tỷ lệ lỗi của mô đun nhận dạng và độ chính xác của mô đun dịch đạt ở mức chấp nhận được, tỷ lệ lỗi này phụ thuộc rất nhiều vào dữ liệu sử dụng.

## 6.2 Những điểm còn hạn chế

Thử nghiệm ban đầu cho thấy kết quả dịch vẫn còn nhiều sai sót và dịch chưa chuẩn nguyên nhân của điều này có thể là do dữ liệu chưa đủ lớn để đảm bảo tính chính xác và thêm nữa việc sử dụng ghép nối các mô đun rời với mỗi mô đun là một mô hình riêng dựa trên xác suất thống kê điều này gây nên việc sai số sau mỗi mô đun tăng lên khá nhiều.

## 6.3 Hướng phát triển

Để cải thiện những hạn chế và tăng chất lượng dịch của hệ thống tôi đề xuất những hướng phát triển cho hệ thống như sau:

- Lựa chọn dữ liệu thuộc những lĩnh vực cụ thể cho cả mô đun nhận dạng, mô đun dịch và mô đun tổng hợp.
- Tăng lượng dữ liệu huấn luyện.
- Tối ưu hóa các khâu ghép nối và tối ưu hóa mã nguồn.

# TÀI LIỆU THAM KHẢO


1. Nakamura S. (2009). Overcoming the language barrier with speech translation technology. *Sci Technol Trends-Q Rev*, (**31**).
2. Dureja M. and Gautam S. (2015). Speech-to-Speech Translation: A Review. *Int J Comput Appl*, **129**(**13**), 28–30.
3. Kurematsu A. and Morimoto T. (1996), *Automatic speech translation: fundamental technology for future cross-language communications*, Gordon and Breach, Amsterdam.
4. Institute of Electrical and Electronics Engineers and Computer Society, eds. (2013), *2013 IEEE 14th International Conference on Mobile Data Management (MDM 2013): Milan, Italy, 3 - 6 June 2013 ; [proceedings ; including workshops]*, IEEE, Piscataway, NJ.
5. 29 M.T.M.T.M.} M. and Profile 234 Points 14 5 3 Recent Achievements Thread Mover I. Forums Answerer III Vanity Award View Microsoft Translator launching Neural Network based translations for all its speech languages. Translator, <https://blogs.msdn.microsoft.com/translation/2016/11/15/microsoft-translator-launching-neural-network-based-translations-for-all-its-speech-languages/>, accessed: 05/12/2017.
6. Bredmar F. (2017). Speech-to-speech translation using deep learning. .
7. Duong L., Anastasopoulos A., Chiang D., et al. (2016). An attentional model for speech translation without transcription. *Proceedings of NAACL-HLT*, 949–959, 949–959.
8. Benesty J., Sondhi M.M., and Huang Y., eds. (2008), *Springer handbook of speech processing*, Springer, Berlin ; London.
9. Glass J. and Zue V. Automatic Speech Recognition. MIT OpenCourseWare, <https://ocw.mit.edu/courses/electrical-engineering-and-computer-science/6-345-automatic-speech-recognition-spring-2003/>, accessed: 02/15/2017.
10. Jurafsky D. and Martin J.H. (2014), *Speech and language processing*, Pearson.
11. Chéragui M.A. (2012). Theoretical overview of machine translation. *Proc ICWIT*, 160.
12. Koehn P. (2010), *Statistical machine translation*, Cambridge University Press, Cambridge; New York.
13. Brown P.F., Pietra V.J.D., Pietra S.A.D., et al. (1993). The mathematics of statistical machine translation: Parameter estimation. *Comput Linguist*, **19**(**2**), 263–311.
14. Collins M. (2011). Statistical machine translation: IBM models 1 and 2. *Columbia Columbia Univ*.
15. Collins M. (2013). Phrase-Based Translation Models. *Columbia Columbia Univ*.
16. Nguyễn Mạnh H., Nguyễn Văn K., and Phan Tất H.T. *Phát Triển engine tổng hợp tiếng sử dụng đơn vị âm không đồng nhất trên nền tảng Android*, Đồ án tốt nghiệp, Đại Học Bách Khoa Hà Nội, Hà Nội.
17. Lại Hoàng N. and Quách Đại Q. (2009), *Tổng hợp tiếng nói trên DSP*, Đồ án tốt nghiệp, Đại Học Bách Khoa Hà Nội, Hà Nội.
18. (2014), *Nghiên cứu và phát triển mô đun trích chọn đặc trưng trên thiết bị smartphone ứng dụng trong nhận dạng tiếng nói*, Đồ án tốt nghiệp, Đại Học Bách Khoa Hà Nội, Hà Nội.

19.Shmyrev N. CMUSphinx Open Source Speech Recognition. CMUSphinx Open Source Speech Recognition, <http://cmusphinx.github.io/>, accessed: 05/16/2017.
20.Moses - Main/HomePage. <http://www.statmt.org/moses/>, accessed: 05/16/2017.
21.A. Rousseau, P. Deléglise, and Y. Estève, "Enhancing the TED-LIUM Corpus with Selected Data for Language Modeling and More TED Talks", in Proceedings of the Ninth International Conference on Language Resources and Evaluation (LREC'14), May 2014. .
22.Tiedemann J. (23-25). Parallel Data, Tools and Interfaces in OPUS. *Proceedings of the Eight International Conference on Language Resources and Evaluation (LREC'12)*, Istanbul, Turkey, European Language Resources Association (ELRA).
23.(2017). Google's speech recognition technology now has a 4.9% word error rate. VentureBeat, <https://venturebeat.com/2017/05/17/googles-speech-recognition-technology-now-has-a-4-9-word-error-rate/>, accessed: 05/31/2017.

# PHỤ LỤC

## Phụ lục A: Kịch bản huấn luyện mô hình dịch tự động bằng Moses

```
#!/bin/bash
#Hello I'm Thinh.
#english corpus in file data.en and vietnamese corpus in file data.vi
sourceLM=/home/students/thinhnv/data/corpus/corpus.txt
outLM=/home/students/thinhnv/model/st-lm.arpa
moses=/home/students/thinhnv/mosesdecoder
#Data folder contain parallel data and tuning data
#I build a en-vi sysmtem and parallel data for training this system
include data.vi and data.en
# parallel data for tune this system include tune.vi and tune.en
data=/home/students/thinhnv/data/moses
working=/home/students/thinhnv/working
irstlm=/home/students/thinhnv/data/moses/smtirstlm.blm.vi

export LANGUAGE=en_US.UTF-8
export LANG=en_US.UTF-8
export LC_ALL=en_US.UTF-8

#CORPUS PREPARATION
#tokenisation
 "$moses/scripts/tokenizer/tokenizer.perl" -l vi \
    < "$data/data.vi"    \
    > "$data/data.tok.vi"
 "$moses/scripts/tokenizer/tokenizer.perl" -l en \
    < "$data/data.en"    \
    > "$data/data.tok.en"

#truecaser
 "$moses/scripts/recaser/train-truecaser.perl" \
     --model "$data/truecase-model.vi" --corpus     \
     "$data/data.tok.vi"
 "$moses/scripts/recaser/train-truecaser.perl" \
     --model "$data/truecase-model.en" --corpus     \
     "$data/data.tok.en"
#Truecasing uses another script from the Moses distribution
"$moses/scripts/recaser/truecase.perl" \
   --model "$data/truecase-model.vi"         \
   < "$data/data.tok.vi" \
   > "$data/data.true.vi"
 "$moses/scripts/recaser/truecase.perl" \
   --model "$data/truecase-model.en"         \
   < "$data/data.tok.en" \
   > "$data/data.true.en"

#finally we clean, limiting sentence length to 80
 "$moses/scripts/training/clean-corpus-n.perl" \
    "$data/data.true" en vi \
    "$data/data.clean" 1 80
#Notice that the last command processes both sides at once

#BUILD LANGUAGE MODEL
# I use KENLM
#/home/students/thinhnv/mosesdecoder/bin/lmplz -o 3 <$sourceLM > $outLM

#TRAINING THE TRANSLATION SYSTEM
```

```
#current folder will be working folder
#cd $working
nohup nice "$moses/scripts/training/train-model.perl" -root-dir train \
 -corpus "$data/data.clean" \
 -f en -e vi -alignment grow-diag-final-and -reordering msd-
bidirectional-fe \
 -lm "0:3:$irstlm:8" \
 -cores 8 \
 -external-bin-dir "$moses/tools" >& training.out &

#TUNING
#prepare data for tuning
#tokenisation
 "$moses/scripts/tokenizer/tokenizer.perl" -l vi \
    < "$data/tune.vi"    \
    > "$data/tune.tok.vi"
 "$moses/scripts/tokenizer/tokenizer.perl" -l en \
    < "$data/tune.en"    \
    > "$data/tune.tok.en"

#Truecasing uses another script from the Moses distribution
"$moses/scripts/recaser/truecase.perl" \
   --model "$data/truecase-model.vi" \
   < "$data/tune.tok.vi" \
   > "$data/tune.true.vi"
 "$moses/scripts/recaser/truecase.perl" \
   --model "$data/truecase-model.en" \
   < "$data/tune.tok.en" \
   > "$data/tune.true.en"
#run tuning
 nohup nice "$moses/scripts/training/mert-moses.pl" \
  "$data/tune.true.en" "$data/tune.true.vi" \
  "$moses/bin/moses" "$working/train/model/moses.ini" --mertdir
"$moses/bin/" \
  --decoder-flags="-threads 8" \
  &> mert.out &
```

Trong kịch bản trên:
- sourceLM là đường dẫn đến tệp văn bản dung để huấn luyện mô hình ngôn ngữ.
- outLM là đường dẫn đến tệp mô hình ngôn ngữ sau khi huấn luyện.
- moses: là đường dẫn đến thư mục cài đặt Moses.
- data: là đường dẫn đến thư mục chứa dữ liệu cặp câu song ngữ.
- working: là thư mục làm việc nơi các tệp tin tạm.

## Phụ lục B: Kịch bản khởi động moses sever

```
#!/bin/bash
#Created by Nguyen Van Thinh 03/04/2017
#this script to run machine translation sever

#this folder contain moses toolkit
moses=/home/students/thinhnv/mosesdecoder
#this folder contain machine translation model
working=/home/students/thinhnv/working
workingvien=/home/students/thinhnv/working/vien
mosesmodel=/home/students/thinhnv/SpeechTranslation/moses/envi
export "PATH=$moses/opt/bin:/home/thinhnv/mysoft/bin:$PATH:$mysoft/bin"
export "LD_LIBRARY_PATH=$moses/opt/lib:$LD_LIBRARY_PATH:$mysoft/lib"
export
"PKG_CONFIG_PATH=$moses/opt/lib/pkgconfig:$PKG_CONFIG_PATH:$mysoft/lib/pk
gconfig"
```

```
export LANGUAGE=en_US.UTF-8
export LANG=en_US.UTF-8
export LC_ALL=en_US.UTF-8

#run moses as a sever
#English-Vietnamese
#"$moses/bin/mosesserver" --server-port 8095 -f "$working/binarised-
model/moses.ini" --server

#Vietnamese-English
"$moses/bin/mosesserver" --server-port 8095 -f "$mosesmodel/binarised-
model/moses.ini" --server
```

Trong kịch bản trên:
- moses: là đường dẫn đến thư mục cài Moses.
- mosesmodel: là đường dẫn đến thư mục chứa mô hình dịch được huấn luyện bởi Moses.

## Phụ lục C: Kịch bản khởi động sever

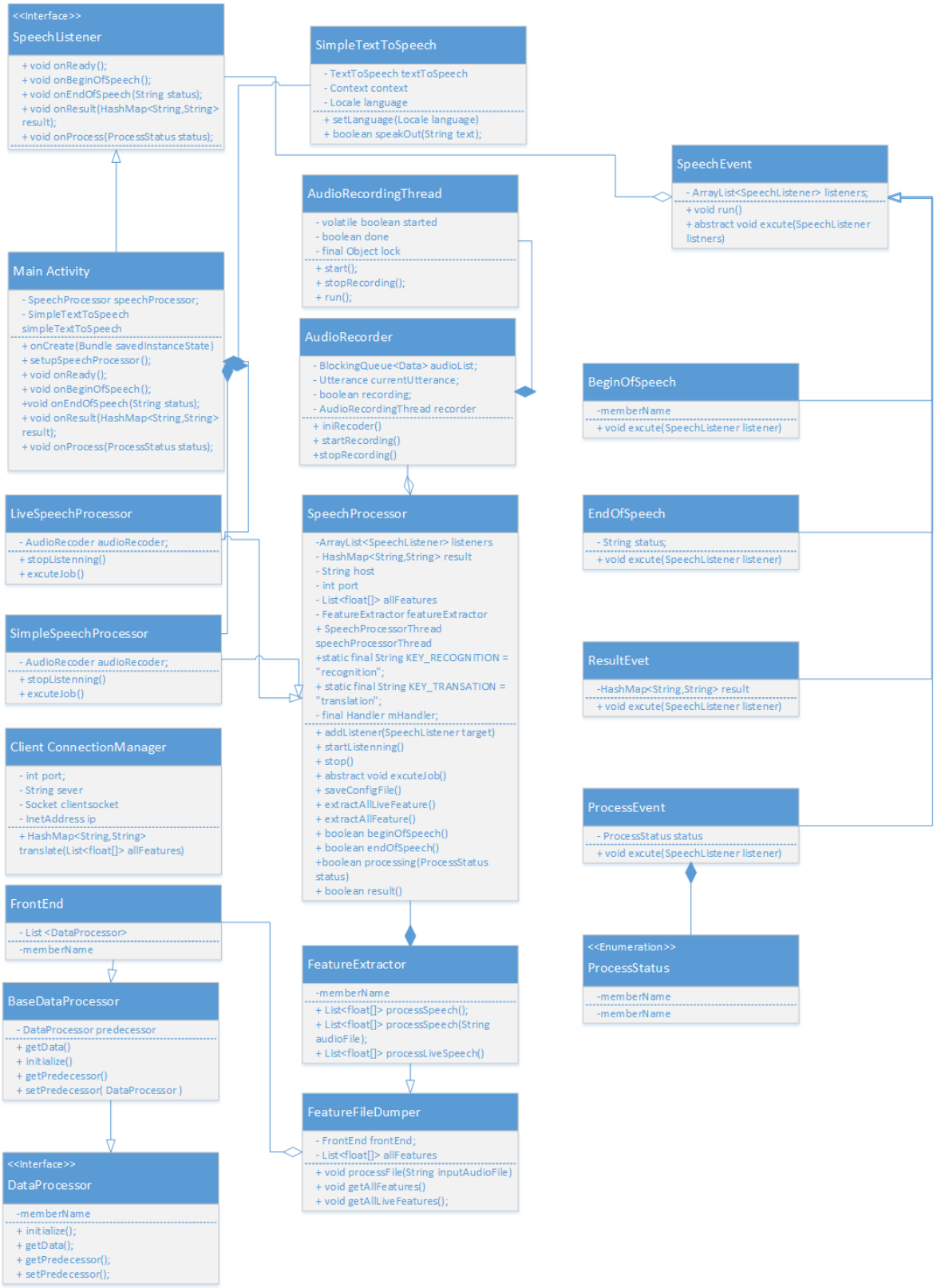
```
#!/bin/bash
#Created by Nguyen Van Thinh 03/04/2017
#this script to run speech translation sever

export LANGUAGE=en_US.UTF-8
export LANG=en_US.UTF-8
export LC_ALL=en_US.UTF-8
#this folder contain speech translation program
reconigzer=/home/students/thinhnv/SpeechTranslation/st-1.01
#this configuration is vietnamese english translation
acousticmodel=/home/students/thinhnv/SpeechTranslation/asrmodel/vien/st.c
d_cont_1000
dicionary=/home/students/thinhnv/SpeechTranslation/asrmodel/vien/st.dic
languagemodel=/home/students/thinhnv/SpeechTranslation/asrmodel/vien/st.l
m
port=9876
mosesport=8096
log=/home/students/thinhnv/SpeechTranslation/sever.log

#run speech translation sever
cd $reconigzer
java SpeechTranslationSever -m $acousticmodel -d $dicionary -l
$languagemodel --port $port --mosesport $mosesport
```

Trong kịch bản trên:
- reconigzer: là đường dẫn đến thư mục cài đặt chương trình sever.
- acousticmodel: đường dẫn đến mô hình âm học được huấn luyện.
- languagemodel: đường dẫn đến mô hình ngôn ngữ
- dichtionanry: đường dẫn đến bộ từ điển âm vị
- port: cổng mà sever lắng nghe
- mosesport: cổng mà Moses sever lắng nghe

# Phụ lục E: Biểu đồ lớp ở client



# Phụ lục F: Biểu đồ lớp ở sever