TRƯỜNG ĐẠI HỌC KHOA HỌC TỰ NHIÊN
KHOA CÔNG NGHỆ THÔNG TIN
BỘ MÔN CÔNG NGHỆ TRI THỨC

BÙI TRUNG HẢI – PHẠM NGỌC TUẤN

XÂY DỰNG ỨNG DỤNG TRỢ LÝ ẢO CHO MÁY TÍNH SỬ DỤNG GOOGLE SPEECH API

KHÓA LUẬN TỐT NGHIỆP CỬ NHÂN CNTT

TP. HCM, 2017

TRƯỜNG ĐẠI HỌC KHOA HỌC TỰ NHIÊN
KHOA CÔNG NGHỆ THÔNG TIN
BỘ MÔN CÔNG NGHỆ TRI THỨC

BÙI TRUNG HẢI – 1312165
PHẠM NGỌC TUẤN – 1312669

XÂY DỰNG ỨNG DỤNG TRỢ LÝ ẢO CHO MÁY TÍNH SỬ DỤNG GOOGLE SPEECH API

KHÓA LUẬN TỐT NGHIỆP CỬ NHÂN CNTT

GIÁO VIÊN HƯỚNG DẪN

TS. NGÔ MINH NHỰT

KHÓA 2013 - 2017

NHẬN XÉT CỦA GIÁO VIÊN HƯỚNG DẪN

....
....
....
....
....
....
....
....
....
....
....
....
....
....
....
....
....
....
....
....
....
....
....
....
....
....
....
....
....
....
....
....
....
....
....
....
....
....

TpHCM, ngày ….. tháng …… năm ……

Giáo viên hướng dẫn

[Ký tên và ghi rõ họ tên]

NHẬN XÉT CỦA GIÁO VIÊN PHẢN BIỆN

....
....
....
....
....
....
....
....
....
....
....
....
....
....
....
....
....
....
....
....
....
....
....
....
....
....
....
....
....
....
....
....
....
....
....
....
....

Khóa luận đáp ứng yêu cầu của Khóa luận cử nhân CNTT.

TpHCM, ngày ….. tháng …… năm ……

Giáo viên phản biện

[Ký tên và ghi rõ họ tên]

LỜI CẢM ƠN

Với lòng biết ơn sâu sắc, trước hết chúng em xin chân thành cảm ơn quý Thầy Cô khoa Công Nghệ Thông Tin - trường đại học Khoa Học Tự Nhiên, những người đã ân cần giảng dạy, xây dựng cho em một nền tảng kiến thức vững chắc để chúng em có thể thực hiện khóa luận này.

Đặc biệt, chúng em xin gửi lời tri ân sâu sắc đến Thầy Ngô Minh Nhựt. Thầy đã rất tận tâm, nhiệt tình hướng dẫn và chỉ bảo chúng em trong suốt quá trình thực hiện luận văn. Nếu không có sự giúp đỡ tận tình của thầy, chúng em chắc chắn không thể hoàn thành luận văn.

Cuối cùng, chúng con xin cảm ơn ba mẹ đã sinh thành, nuôi dưỡng, và dạy dỗ để chúng con có được thành quả như ngày hôm nay. Ba mẹ luôn là nguồn động viên, nguồn sức mạnh hết sức lớn lao mỗi khi chúng con gặp khó khăn trong cuộc sống.

Để hoàn thành luận văn này là tất cả những cố gắng, nỗ lực của chúng em. Tuy nhiên, sẽ không thể tránh khỏi những thiếu sót, kính mong nhận được sự cảm thông và giúp đỡ của quý Thầy Cô và các bạn.

TP. Hồ Chí Minh, 7/2017

Bùi Trung Hải

Phạm Ngọc Tuấn

ĐỀ CƯƠNG CHI TIẾT

Tên Đề Tài: Xây dựng ứng dụng trợ lý ảo cho máy tính sử dụng Google Speech API
Giáo viên hướng dẫn: ThS. Ngô Minh Nhựt
Thời gian thực hiện: từ ngày 15/12/2016 đến ngày 15/07/2017
Sinh viên thực hiện: Bùi Trung Hải – 1312165, Phạm Ngọc Tuấn - 1312669
Loại đề tài: Phát triển hệ thống, nghiên cứu thuật toán

Nội Dung Đề Tài: Tìm hiểu các phương pháp xử lý và tương tác với tín hiệu âm thanh, tiếng nói. Nghiên cứu hệ thống Natural Language Understanding đơn giản. Ứng dụng vào xây dựng ứng dụng trợ lý ảo cho máy tính.

Nội dung chi tiết của đề tài bao gồm:

- Xây dựng ứng dụng trợ lí ảo tương tác bằng giọng nói tiếng Anh cho máy tính
- Các vấn đề quan tâm:
    - Xử lý tín hiệu số (âm thanh, tiếng nói), hệ thống chuyển đổi tiếng nói thành văn bản.
    - Tương tác audio I/O, hệ thống chuyển đổi văn bản thành tiếng nói.
    - Giao thức truyền nhận dữ liệu REST API,…
    - Hệ thống Natural Language Understanding hiệu quả để xác định ý muốn của người dùng
    - Hệ thống chạy đa nhiệm
    - Tối ưu hóa hệ thống

- Các thành phần cơ bản của hệ thống:
    - Module thu âm từ microphone
    - Module nhận dạng từ khóa wake up
    - Module chuyển đổi tiếng nói thành văn bản
    - Module chuyển đổi văn bản thành hành động.
    - Module chuyển đổi văn bản thành tiếng nói
- Ứng dụng thử nghiệm sẽ hỗ trợ các tính năng:
    - Thông báo giờ hiện tại
    - Dự báo thời tiết trong ngày
    - Phát nhạc
    - Trả lời các câu hỏi Wh-question
    - Trả lời các thông tin cơ bản của ứng dụng: tên, tuổi,…

Kế Hoạch Thực Hiện:

- 15/12/2016 – 14/01/2017: Khảo sát, tìm hiểu về các thư viện python phục vụ cho việc tương tác và xử lý tín hiệu âm thanh.

- 15/01/2017 – 14/02/2017: Thiết kế các thành phần của hệ thống.

- 15/02/2017 – 14/03/2017: Cài đặt và thử nghiệm các thành phần: thu âm, nhận dạng từ khóa wake up, chuyển giọng nói thành văn bản, chuyển văn bản thành giọng nói.

- 15/03/2017 – 14/04/2017: Tìm hiểu và xây dựng hệ thống Natural Language Understanding.

- 15/04/2017 – 14/05/2017: Cài đặt các chức năng mà ứng dụng hỗ trợ.

- 15/05/2017 – 31/05/2017: Ráp nối tất cả các thành phần, tiến hành thử nghiện và hoàn thiện hệ thống.

- 01/06/2017 – 28/06/2017: Viết và hoàn thiện luận văn.

Xác nhận của GVHD	**Ngày 08 tháng 07 năm 2013** **SV Thực hiện**

MỤC LỤC

DANH MỤC HÌNH ẢNH

DANH MỤC BẢNG

TÓM TẮT KHÓA LUẬN

Trong xu hướng công nghệ hiện nay, vai trò của các trợ lý ảo ngày càng trở nên quan trọng. Các hãng công nghệ lớn thay nhau tung ra những trợ lý ảo của riêng mình tích hợp trên các thiết bị di động: Siri của Apple, Cortana của Microsoft, Google Assistant của Google, Alexa của Amazon,... Chức năng của các trợ lý ảo này ngày càng được mở rộng, từ những chức năng đơn giản như tra cứu, hỏi đáp, đến những chức năng cao hơn như quản lý lịch, gọi điện thoại, dẫn đường, điều khiển các thiết bị khác,... Khóa luận này có mục đích tạo ra một trợ lý ảo có khả năng chạy được trên nhiều nền tảng hệ điều hành khác nhau trên máy tính cá nhân.

Nhận diện giọng nói là một trong những thành phần quan trọng nhất của một trợ lý ảo. Nhiều công ty và nhóm nghiên cứu lớn nhỏ đã nghiên cứu và đưa ra các bộ toolkit cũng như API cho việc nhận diện giọng nói, trong đó một trong những API có chất lượng được đánh giá tốt nhất là Google Speech API của gã khổng lồ công nghệ Google. Do đó, chúng tôi muốn tận dụng chất lượng của Google Speech API để tạo nên một trợ lý ảo có độ chính xác cao về nhận diện giọng nói.

Kết quả sơ bộ mà khóa luận đạt được là tạo ra một trợ lý ảo có thể chạy trên các hệ điều hành phổ biến trên máy tính cá nhân như Windows, Linux, Mac. Trợ lý ảo có những chức năng cơ bản của một trợ lý ảo như hỏi đáp, tra cứu thông tin, trả lời các câu hỏi về thời gian, thời tiết, ngoài ra còn có thể phát nhạc theo yêu cầu và chào hỏi ở mức độ đơn giản.

Chương 1

Mở đầu

Nội dung của chương 1 giới thiệu tổng quan về đề tài, nêu ra mục tiêu của khóa luận, và cấu trúc nội dung của luận văn.

1.1 Tổng quan về đề tài

1.1.1 Giới thiệu về trợ lý ảo

Trợ lý ảo là một phần mềm trên máy tính hoặc thiết bị di động có khả năng hỗ trợ người dùng thực hiện nhiều loại công việc, nhận lệnh từ người dùng dưới dạng ngôn ngữ tự nhiên, thường là giọng nói. Nhờ khả năng nhận lệnh và phản hồi qua giọng nói, người dùng có thể ra lệnh cho trợ lý ảo mà không cần phải thao tác bằng tay trên thiết bị. Điều đó sẽ giúp tăng tính hiệu quả và tạo ra sự tự nhiên trong giao tiếp giữa người và máy, tạo ra những kênh tương tác mới khác với truyền thống, mang đến cho người dùng những trải nghiệm mới mẻ và thứ vị hơn khi sử dụng những thiết bị công nghệ.

Chức năng của các trợ lý ảo rất phong phú và đa dạng, từ những chức năng bình thường như hỏi đáp, tra cứu thông tin, bật nhạc, tìm kiếm trong danh bạ, quản lý lịch, đặt báo thức,... cho đến những chức năng đặc biệt như chơi game, trò chuyện, điều khiển các thiết bị trong gia đình, thậm chí là mua sắm, đặt chỗ nhà hàng, đặt vé máy bay,...

1.1.2 Khảo sát thị trường trợ lý ảo

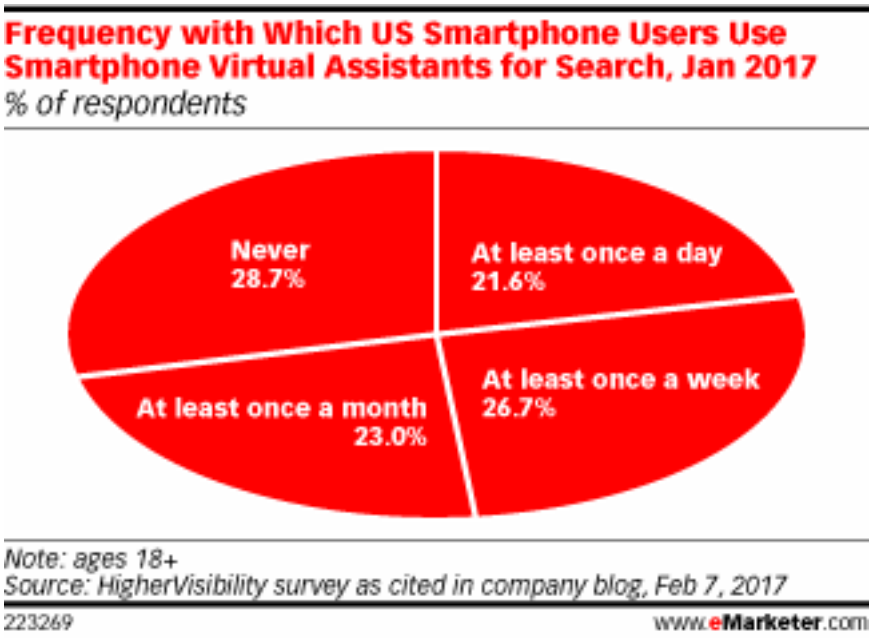
Tính đến hiện tại, gần như tất cả các hãng công nghệ lớn đều đã tung ra trợ lý ảo của riêng mình:

- Apple với Siri, hoạt động trên các thiết bị của Apple như iPhone, iPad, iPod Touch, Mac và Apple TV.
- Microsoft với Cortana, hoạt động trên các phiên bản mới của Windows như Windows 10, Windows 10 Mobile, Windows Phone 8.1, và các thiết bị khác như Microsoft Band, Xbox One.
- Amazon với Alexa, hoạt động trên loa thông minh Amazon Echo.
- Google với Google Assistant, hoạt động trên các thiết bị Android, loa thông minh Google Home và ứng dụng nhắn tin Allo.
- Gần đây, Samsung đã ra mắt trợ lý ảo của mình mang tên Bixby, chạy trên các dòng điện thoại Samsung Galaxy.
- Facebook cũng đã công bố trợ lý ảo của mình mang tên M, sẽ ra mắt trong năm 2017 trên các ứng dụng Facebook và Facebook Messenger.

Trong xu hướng đó, số lượng người dùng và tần suất sử dụng của các trợ lý ảo đang ngày càng gia tăng. Theo một khảo sát thực hiện vào tháng 01/2017 trên các người dùng smartphone tại Mỹ[3], có gần 27% người được hỏi nói rằng họ dùng trợ lý ảo ít nhất một lần mỗi tuần, và khoảng 22% người dùng sử dụng trợ lý ảo hàng ngày. Trong khi đó, có 28.7% số người được hỏi chưa bao giờ sử dụng trợ lý ảo.

Khi được hỏi về lý do sử dụng trợ lý ảo, 1/3 số người được hỏi nói rằng tìm kiếm bằng trợ lý ảo dễ hơn tìm kiếm bằng tay. Khoảng 1/4 số người được khảo sát nói rằng họ không thể gõ chữ trên smartphone, hoặc không thể nhìn rõ trên smartphone, hoặc vì tìm kiếm bằng trợ lý ảo nhanh hơn tìm kiếm bằng tay. Tuy nhiên, lý do lớn nhất mà nhiều người dùng sử dụng trợ lý ảo là lái xe. Có đến hơn một nửa số người được hỏi cho biết họ sử dụng trợ lý ảo trong khi lái xe.

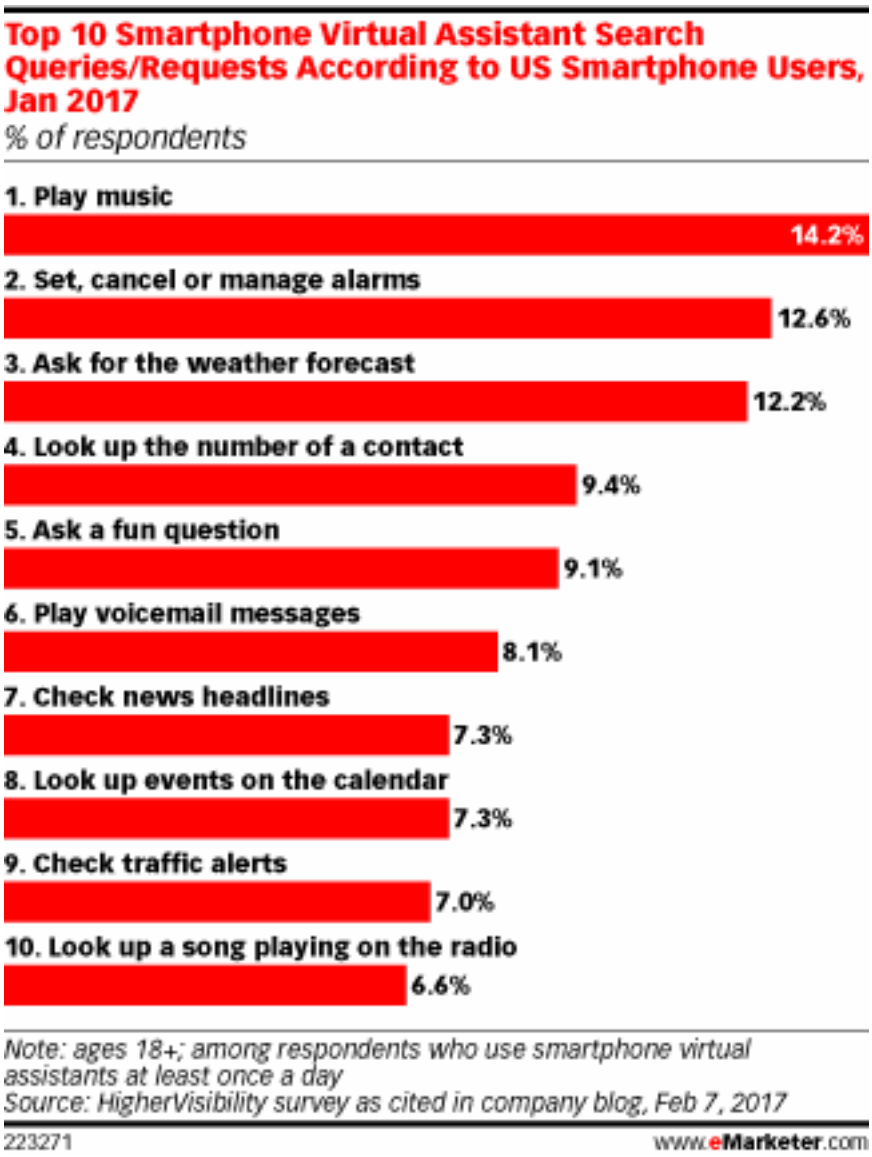
Nhìn vào hình 1.2 có thể thấy phần đông người dùng sử dụng trợ lý ảo với mục đích bật nhạc, quản lý báo thức, hỏi thông tin dự báo thời tiết. Ngoài ra, họ còn dùng



Hình 1.1: Tần suất người dùng smartphone ở Mỹ sử dụng trợ lý ảo, tháng 01/2017[3]

trợ lý ảo để tìm kiếm số điện thoại trong danh bạ, hỏi câu hỏi vui, bật các tin nhắn thoại hoặc xem tin tức.

Như vậy có thể nói các trợ lý ảo đang ngày càng góp một phần quan trọng trong cuộc sống của những người dùng công nghệ. Không chỉ tạo ra một trải nghiệm mới, các trợ lý ảo còn giúp tiết kiệm thời gian và công sức. Ngoài ra, các trợ lý ảo còn có thể giúp được những người khuyết tật hoặc người cao tuổi có thể tiếp cận với các thiết bị công nghệ một cách dễ dàng hơn.



Hình 1.2: Những chức năng được sử dụng nhiều nhất trên trợ lý ảo theo người dùng smartphone ở Mỹ, tháng 01/2017[3]

1.2 Mục tiêu của khóa luận

Nhận thấy các trợ lý ảo nêu trong phần trước đa phần chỉ hoạt động trên một số nền tảng nhất định của hãng làm ra chúng, và chủ yếu dành cho các thiết bị di động, chúng tôi thực hiện khóa luận này với mục đích chính là tạo ra một ứng dụng trợ lý ảo có khả năng chạy trên các nền tảng hệ điều hành phổ biến trên máy tính cả nhân như Windows, Linux, Mac, với các chứng năng cơ bản:

- Hỏi đáp, tra cứu thông tin
- Hỏi giờ, thời tiết
- Phát nhạc
- Trò chuyện đơn giản

Ứng dụng phải có khả năng tự kích hoạt khi người dùng gọi tên, nhận biết chính xác các câu nói của người dùng, phản hồi một cách chính xác bằng giọng nói.

Các mục tiêu nhỏ:

- Ứng dụng có khả năng tự kích hoạt khi người dùng gọi tên
- Nhận biết chính xác câu nói của người dùng
- Phản hồi một cách chính xác và tự nhiên
- Thời gian xử lý ngắn (tính từ lúc người dùng hoàn thành câu nói đến lúc ứng dụng bắt đầu phản hồi).

1.3 Nội dung luận văn

Nội dung của luận văn sẽ gồm những phần sau:

- Chương 1: *Mở đầu*: Giới thiệu tổng quan về đề tài, mục tiêu của khóa luận và cấu trúc nội dung của luận văn.

- Chương 2: *Tổng quan về tín hiệu âm thanh, tiếng nói. Thư viện PyAudio*: Giới thiệu tổng quan về tín hiệu âm thanh, tiếng nói, các thành phần của âm thanh, cách lưu trữ âm thanh trong máy tính, các thông số của file âm thanh; giới thiệu về thư viện PyAudio: chức năng, cách cài đặt, cách sử dụng.

- Chương 3: *Speech to Text*: Giới thiệu về bài toán Speech to Text, các ứng dụng, các vấn đề cần giải quyết trong Speech to Text; giới thiệu về các thư viện pocketsphinx và Google Speech to Text: chức năng, cách cài đặt, cách sử dụng, ưu và nhược điểm, vai trò của các thư viện đó trong hệ thống.

- Chương 4: *Text to Speech*: Giới thiệu về bài toán Text to Speech, các ứng dụng; giới thiệu về Google Text to Speech và iSpeech: chức năng, cách sử dụng, ưu và nhược điểm, vai trò trong hệ thống.

- Chương 5: *Intent Classification*: Giới thiệu về bài toán Intent Classification, các thuật toán để giải quyết bài toán này, các ứng dụng; giới thiệu về thư viện Rasa NLU: chức năng, cách cài đặt, cách sử dụng, vai trò trong hệ thống.

- Chương 6: *Ứng dụng Alexa*: Giới thiệu tổng quan về ứng dụng Alexa, các module trong chương trình, luồng hoạt động giữa các module, các chức năng của ứng dụng.

- Chương 7: *Kết luận và Hướng phát triển*: Nếu ra các kết quả đạt được của khóa luận, các hạn chế và hướng phát triển.

Chương 2

Tín hiệu âm thanh, tiếng nói. Thư viện PyAudio

Nội dung chương 2 sẽ giới thiệu tổng quan về âm thanh và tiếng nói, cách âm thanh được lưu trữ trên máy tính, các ứng dụng của âm thanh và tiếng nói. Chương 2 cũng sẽ giới thiệu về chức năng, cách cài đặt, cách sử dụng cũng như các ưu nhược điểm của thư viện PyAudio.

2.1 Tổng quan

2.1.1 Tổng quan về âm thanh

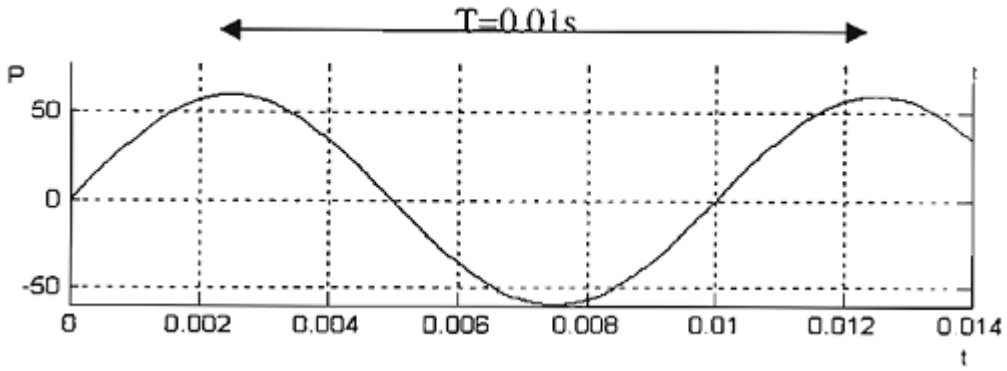
Trong vật lý, âm thanh được định nghĩa là các giao động cơ học lan truyền thông qua các phương tiện truyền dẫn như: không khí, nước, chất rắn,... Các giao động cơ học đó còn được gọi là sóng âm. Sóng âm được tạo ra do có sự biến đổi về áp suất theo thời gian. Vận tốc lan truyền trong không khí của sóng âm vào khoảng 343.2 m/s.

Đối với con người, âm thanh là những giao động có thể được cảm nhận thông qua thính giác. Các giao động khi lan truyền trong không khí sẽ va đập và làm rung màng nhĩ, qua đó não bộ sẽ thu được tín hiệu âm thanh. Con người có thể nghe được âm thanh có tần số từ 16 Hz đến 20 kHz. Âm thanh có tần số cao hơn 20 kHz gọi là siêu âm, âm thanh có tần số thấp hơn 16 Hz gọi là hạ âm.

Cách đơn giản nhất để biểu diễn âm thanh là dưới dạng sóng sin với trục x là thời gian, trục y là áp xuất:

$$P = A\sin(2\pi ft)$$

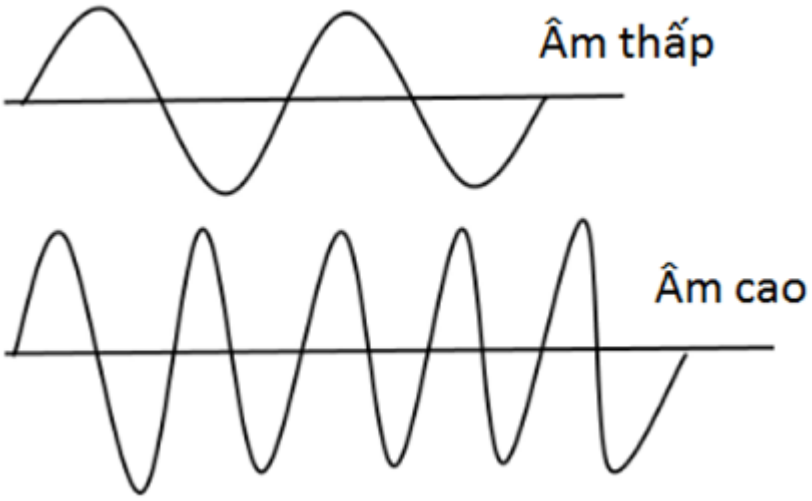
Trong đó: P là áp suất, đơn vị là decibels (dB) hoặc pascals
A là biên độ của sóng, đơn vị là decibels (dB)
t là thời gian, đơn vị là giây (s)
f là tần số, đơn vị là hertz (Hz)



Hình 2.1: Sóng sin có biên độ 60 dB, tần số 100 Hz

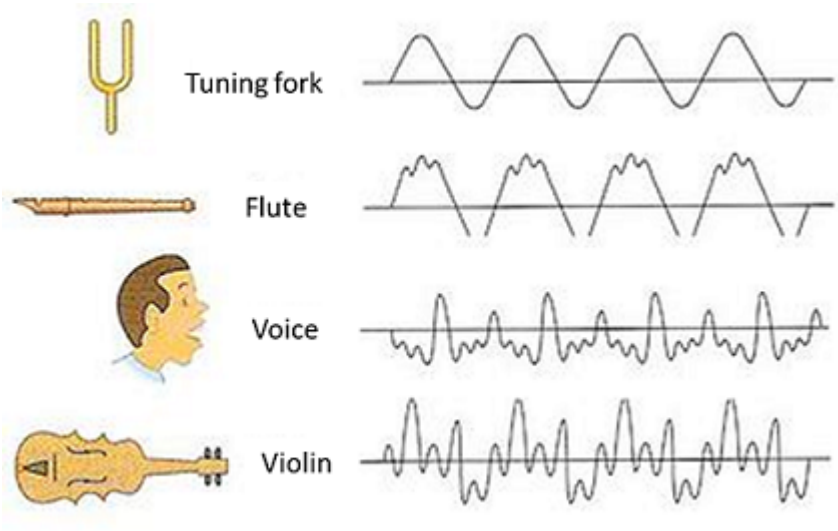
Âm thanh có một số đặc trưng cơ bản như: độ cao, độ mạnh, độ dài và âm sắc.

- **Độ cao** (Pitch): âm thanh luôn có một độ cao nhất định. Độ cao của âm thanh phụ thuộc vào tần số của sóng âm. Tần số càng lớn thì âm thanh càng cao, tần số càng bé thì âm thanh càng trầm.



Hình 2.2: Minh họa độ cao (Pitch) của âm

- **Độ mạnh** (Intensity): hay còn gọi là độ to của âm thanh. Độ mạnh của âm thanh phụ thuộc vào biên độ sóng âm. Biên độ càng lớn thi cường độ âm càng mạnh, biên độ càng bé thì cường độ âm càng yếu.
- **Độ dài** (Duration): là thời gian kéo dài của sóng âm.
- **Âm sắc** (Timbre): âm sắc là một đặc trưng sinh lý của âm, giúp phân biết âm thanh do các nguồn khác nhau phát ra. Âm sắc liên quan mật thiết với đồ thị giao động âm.



Hình 2.3: Minh họa âm sắc (Timbre) của âm

2.1.2 Tổng quan về tiếng nói

Trong tự nhiên, âm thanh bao gồm nhiều loại được tạo ra từ nhiều nguồn khác nhau:

- **Âm nhạc**: âm thanh được phát ra từ các nhạc cụ.
- **Tiếng kêu**: được phát ra từ các loại động vật. Ví dụ: cá heo (1-164 kHz).
- **Tiếng động**: âm thanh phát ra từ sự va chạm giữa hai vật.
- **Tiếng ồn**: là những âm thanh không mong muốn.
- **Tiếng nói**: là những âm thanh được phát ra từ miệng con người

Ta có thể phân loại tiếng nói dựa theo thanh:

- **Âm hữu thanh**: là âm khi phát ra có sự dao động của đôi dây thanh quản.
- **Âm vô thanh**: phát ra khi đôi dây thanh quản không dao động. Thí dụ phần cuối của phát âm English, chữ sh cho ra âm xát.

Hoặc theo âm:

- **Nguyên âm**: là âm phát ra có thể kéo dài. Tất cả nguyên âm đều là âm hữu thanh, nghĩa là tuần hoàn và khá ổn định trong một đoạn thời gian vài chục ms.
- **Phụ âm**: là âm chỉ phát ra một nhát, không kéo dài được. Có phụ âm hữu thanh và phụ âm vô thanh.

Tiếng nói đóng vai trò quan trọng trong hoạt động động giao tiếp giữa con người, nó là phương tiện giao tiếp nhanh, tiện lợi và phổ biến nhất.

2.2 Lưu trữ âm thanh trong máy tính

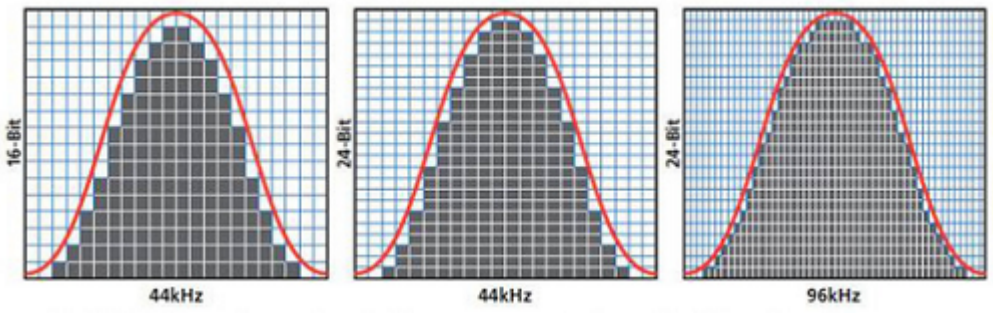
Tất cả âm thanh mà chúng ta nghe được trong tự nhiên đều tồn tại dưới dạng sóng âm, là các sóng cơ học tuần hoàn liên tục analog. Trong khi đó, máy tính xử lý và lưu trữ thông tin dưới dạng các xung điện tử rời rạc digital. Vì vậy, để có thể lưu trữ và xử lý tín hiệu âm thanh trên máy tính, ta phải mô phỏng sóng âm bằng những mẫu rời rạc. Việc mô phỏng được đặc trưng bởi các thông số:

- **Mẫu (Sample) là gì**: là đơn vị âm thanh nhỏ nhất được lưu trong máy tính. Để có các xung điện tử rời rạc, ta cần lấy mẫu nhiều lần từ tín hiệu analog. Mỗi mẫu là giá trị biên độ của sóng âm tại thời điểm lấy mẫu.
- **Tần số lấy mẫu (Sample Rate)**: là số lần lấy mẫu trên một giây, đơn vị là Hz. Tần số lấy mẫu càng cao, tín hiệu số thu được càng chính xác.
- **Độ dày bit (BitDepth)**: để lưu lại dưới dạng số, mỗi mẫu được biểu diễn bằng một lượng bit dữ liệu nhất định gọi là BitDepth. BitDepth càng lớn âm thanh cáng sắc nét, trung thực.

- **Kênh (Chanel)**: Bằng thuật toán, tín hiệu số có thể được chia thành nhiều kênh để khi nghe bằng hệ thống âm thanh vòm sẽ tạo ra cảm giác thật nhất.

Một trong những phương pháp chuyển đổi tín hiệu analog sang digital phổ biến nhất hiện nay là Pulse-Code Modulation (PCM). Kỹ thuật PCM bao gồm 3 bước:

- **Lấy mẫu (Sampling)**: quá trình rời rạc hoá tín hiệu analog đầu vào theo tần số lấy mẫu f. Ví dụ f = 44100 Hz, ta sẽ lấy mẫu 44100 lần trong một giây.



Hình 2.4: Minh họa tín hiệu analog được lấy mẫu theo nhiều tần số khác nhau

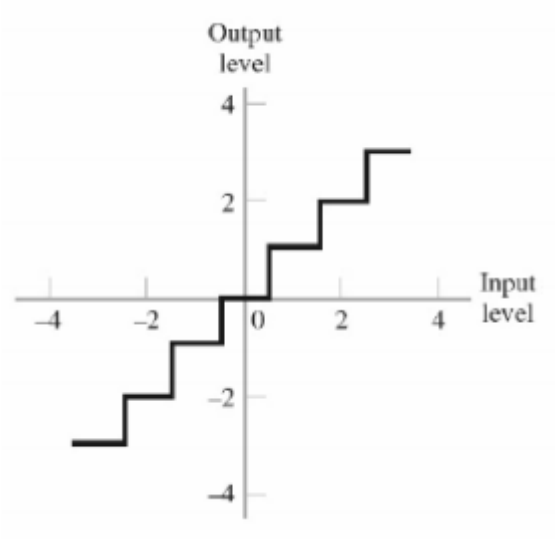
- **Lượng tử hóa (Quantization)**: tín hiệu analog sau khi được lấy mẫu thì mỗi mẫu có thể có vô số các giá trị. Thay vì sử dụng giá trị mẫu chính xác, giá trị mẫu được thay bằng giá trị gần nhất trong M giá trị cho phép. Các giá trị lượng tử được gọi là các mức lượng tử, nếu các mức lượng tử cách đều nhau gọi là lượng tử đều, ngược lại gọi là lượng tử không đều. Số mức lượng tử M phụ thuộc vào số khả năng mà BitDepth có thể biểu diễn.

- **Mã hóa (Encoding)**: Là quá trình biến đổi giá trị các mẫu sau khi lượng tử thành các từ mã dài n bit (n là BitDepth).

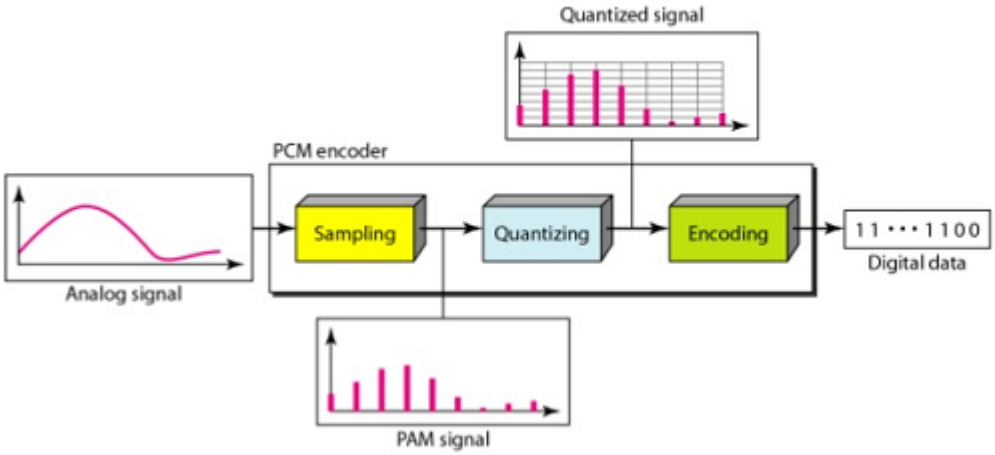
PCM cùng với những biến thể của nó là nền tảng cho âm thanh số. PCM sẽ sẽ hình thành một dạng sóng, sóng này ít nhiều có thể được chạy ngay bởi một bộ xử lý tín hiệu số. Trong khi hầu hết các định dạng khác khi thao tác với âm thanh thì cần thông qua các thuật toán điều khiển nên phải giải mã chúng khi sử dụng.

2.2.1 Lưu trữ không nén (uncompressed)

Các tín hiệu số thu được thông qua PCM sẽ được lưu trữ nguyên bản, không qua bất kỳ phương pháp nén hay sửa đổi.



Hình 2.5: Minh họa quá trình lượng tử hóa



Hình 2.6: Sơ đồ khối mô phỏng phương pháp PCM

- **Ưu điểm**: khi lưu trữ dưới dạng không nén, tín hiệu âm thanh có thể được chạy ngay bởi một bộ xử lý tín hiệu số mà không cần thông qua các thuật toán giải mã.
- **Khuyết điểm**: kích thước tập tin âm thanh sẽ rất lớn làm hao phí không gian lưu trữ và băng thông đường truyền.

Các định dạng âm thanh tiêu biểu cho phương pháp lưu trữ không nén là: WAV, AIFF,...

2.2.2 Lưu trữ nén

Các tín hiệu số thu được thông qua PCM sẽ được nén thông qua các thuật toán khác nhau.

- **Ưu điểm**: kích thước tập tin âm thanh nhỏ. Tiết kiệm không gian lưu trữ và băng thông đường truyền.
- **Khuyết điểm**: chất lượng âm thanh có thế bị giảm sút tùy vào thuật toán nén. Tập tin âm thanh muốn thao tác được phải thông qua quá trình giải mã

Nén không mất mát (Lossless compression)

Các tín hiệu âm thanh gốc sẽ được nén mà không mất dữ liệu và đảm bảo chất lượng ban đầu sau khi giải nén.

Để làm được điều này, cần nhờ đến nhiều thuật toán nén khác nhau. Tuy nhiên, ý tưởng chung của các thuật toán đều là tìm ra quy luật lặp của dữ liệu, sau đó tìm 1 cách hiển thị khác tối ưu hơn, tốn ít dữ liệu hơn. Ví dụ thay vì lưu chuỗi "aaaa bbb cccccc" ta sẽ chuyển thành chuỗi "a4 b3 c6" ít tốn dữ liệu hơn.

Tập tin âm thanh được nén bằng phương pháp này sẽ có tỉ lệ nén không cao, vào khoảng 1/2 đến 1/3 dung lượng âm thanh gốc.

Tiêu biểu cho phương pháp nén không mất mát là các định dạng âm thanh: FLAC, ALAC, APE,...

Nén có mất mát (Lossy compression)

Các tín hiệu âm thanh khi được nén sẽ bị loại bỏ đi các thành phần không quan trọng nhằm giảm tối đa kích thước lưu trữ. Mỗi thuật toán nén sẽ có những tiêu chí khác nhau để chọn lựa các thành phần âm thanh sẽ bị bỏ.Ví dụ, ngưỡng nghe của tai người là những âm thanh có tần số từ 14 Hz đến 20 kHz, như vậy thuật toán sẽ bỏ bớt đi các âm thanh có tần số nằm ngoài khoảng đó.

Bên cạnh đó, khi giải mã tập tin âm thanh bị nén, các thuật toán sẽ tạo ra các âm thanh giả nhằm lấp vào các phần đã bị bỏ bớt đi. Hệ quả của việc này là bạn thường nghe các âm thanh méo mó. Các tập tin nhạc được nén với tỉ lệ càng cao thì sự méo

tiếng càng nhiều. Bạn sẽ rất dễ dàng nhận ra sự khác biệt khi nghe hai tập tin nhạc gốc và nhạc nén bị mất mát dữ liệu.

Tập tin âm thanh được nén bằng phương pháp này sẽ có tỉ lệ nén rất cao, lên đến 1/10 dung lượng âm thanh gốc.

Tiêu biểu cho phương pháp nén không mất mát là các định dạng âm thanh: MP3, AAC, WMA,...

2.3 Ứng dụng

Hiện nay âm thanh, tiếng nói đã được nghiên cứu và ứng dụng rộng rãi trong nhiều lĩnh vực của cuộc sống như: truyền thông, âm nhạc, chế tạo sonar,...

Trong luận văn này, tiếng nói giữ vai trò quan trọng là phương tiện tương tác chính giữa người dùng và ứng dụng. Người dùng sử dụng tiếng nói để ra lệnh cho ứng dụng, và ứng dụng sử dụng tiếng nói để phản hồi kết quả cho người dùng.

2.4 Thư viện PyAudio

2.4.1 Tổng quan

PyAudio là thư viện mã nguồn mở hỗ trợ người dùng thao tác với âm thanh trên máy tính một cách dễ dàng. PyAudio được viết bằng Python dựa trên thư viện PortAudio. PyAudio có khả năng hoạt động và cài đặt dễ dàng trên đa nền tảng: Windows, Mac OS, và Linux . Tính tới tháng 06/2017 phiên bản mới nhất của PyAudio là 0.2.11

2.4.2 Chức năng

PyAudio cũng cấp tính năng giúp người dùng dễ dàng thu và phát âm thanh từ dữ liệu thô dưới dạng từng sample. PyAudio có thể hoạt động ở chế độ đồng bộ (Synchronous) hoặc không đồng bộ (Asynchronous).

2.4.3 Cài đặt

- **Windows**: python -m pip install pyaudio

- **Mac OS**: brew install portaudio pip install pyaudio
- **Linux**: pip install pyaudio
- **Build từ source**: https://people.csail.mit.edu/hubert/git/pyaudio.git

2.4.4 Cách sử dụng

```
"""PyAudio Example: Play a wave file."""
import pyaudio
import wave
import sys
CHUNK = 1024

if len(sys.argv) < 2:
   print("Plays a wave file.\n\nUsage: %s filename.wav" % sys.argv[0])
   sys.exit(-1)

wf = wave.open(sys.argv[1], 'rb')

# instantiate PyAudio (1)
p = pyaudio.PyAudio()

# open stream (2)
stream = p.open(format=p.get_format_from_width(wf.getsampwidth()),
                channels=wf.getnchannels(),
                rate=wf.getframerate(),
                output=True)

# read data
data = wf.readframes(CHUNK)

# play stream (3)
while len(data) > 0:
   stream.write(data)
   data = wf.readframes(CHUNK)
```

```
# stop stream (4)
stream.stop_stream()
stream.close()

# close PyAudio (5)
p.terminate()
```

Đoạn code trên minh họa cách sử dụng thư viện PyAudio để phát âm thanh từ một file WAV. Chúng tôi sẽ dùng nó để hướng dẫn cách sử dụng đơn giản thư viện PyAudio

- Để sử dụng thư viện PyAudio, trước hết cần khởi tạo PyAudio bằng cách sử dụng câu lệnh textbfpyaudio.PyAudio() (1). Câu lệnh này sẽ khởi tạo thư viện PortAudio bên dưới.
- Để có thể thu âm hoặc phát audio, ta mở các stream tương ứng bằng câu lệnh **pyaudio.PyAudio.open()** (2). Input stream để thu âm, output stream để phát audio. Một số tham số cần chú ý trong câu lệnh này:
  - **format**: là định dạng được sử dụng khi encoding. Định dạng này phụ thuộc vào số lượng bit dùng để mã hóa một mẫu (BitDepth).
  - **chanels**: số lượng kênh của tập tin âm thanh cần phát hoặc số lượng kênh của âm thanh cần thu âm.
  - **rate**: là tần số lấy mẫu (Sample Rate)
  - **output**: giá trị này bằng **true** thì đây là stream output dùng để phát âm thanh.
  - **input**: giá trị này bằng **true** thì đây là stream input dùng để thu âm thanh.
- Khi cần phát âm thanh, ta ghi dữ liệu âm thanh lên output stream bằng hàm **pyaudio.Stream.write()**. Khi cần thu âm thanh, ta đọc dữ liệu âm thanh thu được từ input stream thông qua hàm **pyaudio.Stream.read()** (3). Đoạn code minh họa đang chạy ở chế độ blocking, nên các hàm **pyaudio.Stream.write()** và **pyaudio.Stream.read()** sẽ block chương trình đến khi chúng thực hiện việc đọc/ghi xong.

- Để tạm dừng đọc/ghi âm thanh ta dùng hàm **pyaudio.Stream.stop_stream()**. Để kết thúc stream ta dùng hàm **pyaudio.Stream.close()** (4).
- Cuối cùng để giải phóng PortAudio ta dùng hàm **pyaudio.PyAudio.terminate()** (5).

2.4.5 Các ưu, khuyết điểm

- **Ưu điểm**: thư viện cài đặt đơn giản, hỗ trợ chạy trên nhiều hệ điều hành.
- **Nhược điểm**: thư viện hỗ trợ ít tính năng. Không hỗ trợ thu âm đồng thời từ nhiều micro.

2.4.6 Ứng dụng

Trong luận văn này, thư viên PyAudio đảm nhận vai trò thu âm giọng nói của người dùng. Cung cấp thông tin cho các thành phần khác của hệ thống xử lý.

Chương 3

Speech to Text

Nội dung chương 3 sẽ giới thiệu tổng quan về bài toán Speech to Text, mô hình hoạt động, các ứng dụng của Speech to Text, các vấn đề cần giải quyết của module Speech to Text trong một hệ thống trợ lý ảo và cách giải quyết các vấn đề đó. Chương 3 cũng sẽ giới thiệu về chức năng, cách cài đặt, cách sử dụng cũng như các ưu nhược điểm của các thư viện pocketsphinx và Google Speech to Text.

3.1 Tổng quan

Speech to Text, hay Speech Recognition là một lĩnh vực trong khoa học máy tính, trong đó nghiên cứu và phát triển các phương pháp và công nghệ để máy tính có thể nhận biết và chuyển đổi ngôn ngữ nói sang dạng văn bản. Speech recognition là một bài toán khó dành cho các nhà khoa học, vì tiếng nói luôn thay đổi theo thời gian và có sự khác biệt giữa tiếng nói của những người khác nhau, tốc độ nói, ngữ cảnh và môi trường khác nhau.

Những hệ thống speech recognition đầu tiên trên thế giới có khả năng nhận biết rất hạn chế: số lượng từ vựng mà chúng có thể nhận biết chỉ ở mức vài chục, và chỉ có thể nhận biết chính xác nếu người dùng nói một cách rất rõ ràng. Ngày nay, với sự phát triển bùng nổ của deep learning và big data, các công nghệ speech recognition cũng đã phát triển rất nhanh về số lượng từ vựng và độ chính xác. Các hệ thống speech recognition hiện đại nhất có thể nhận biết được hàng chục nghìn, thậm chí là hàng trăm nghìn từ khác nhau, và độ lỗi khi nhận biết đã giảm đến gần mức nhận biết của con người. Ngày càng nhiều công ty công nghệ đã tham gia vào cuộc đua về speech

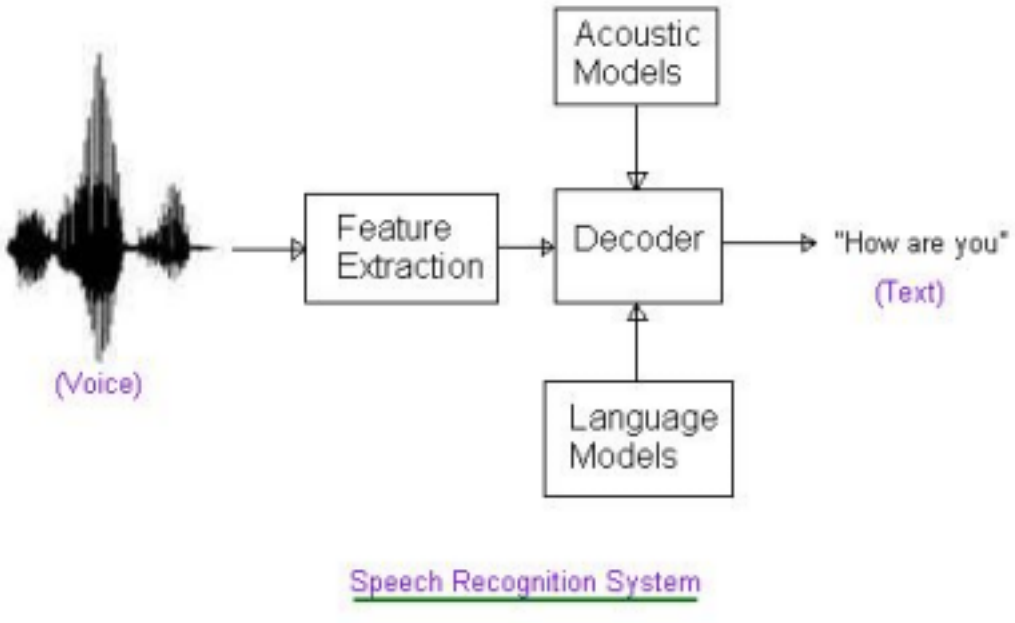
recognition, có thể kể đến Google, Microsoft, IBM, Baidu, Apple, Amazon,...

3.2 Mô hình hoạt động

Hiện nay có rất nhiều kỹ thuật khác nhau để phát triển speech recognition. Tuy nhiên có thể nhận thấy một điểm chung của đa phần các kỹ thuật này là các mô hình thống kê, trong đó nổi bật là Hidden Markov Model (HMM).

Khi huấn luyện, các câu nói trong tập dữ liệu sẽ thường được chia thành các thành phần âm (phonemes), tín hiệu âm thành đầu vào cũng sẽ được chia thành những đoạn rất ngắn, và sử dụng HMM để tìm sự liên hệ giữa các phonemes và tín hiệu âm thanh. Khi đó, khi một đoạn âm thanh mới được đưa vào, HMM sẽ trả về những chuỗi âm có thể tương ứng với đoạn âm thanh đó. Mô hình này được gọi là acoustic model.

Bên cạnh acoustic model, người ta cũng sẽ tạo ra các language model. Language model sẽ chứa một danh sách các từ (gọi là từ điển), cấu trúc âm của các từ, và có thể có thêm cấu trúc tổng quát của các câu. Language model này sẽ nhận các chuỗi âm output từ acoustic model và tìm cách ghép lại thành các từ và câu.



[1]

Hình 3.1: Cấu trúc một hệ thống speech recognition đơn giản

[1] http://www.rfwireless-world.com/Terminology/automatic-speech-recognition-system.html

Ngoài HMM, nhiều mô hình khác cũng đã được các nhà phát triển speech recognition sử dụng để thay thế hoặc hỗ trợ cho HMM như Gaussian Mixture Model, Deep Neural Networks, Recurrent Neural Networks, Finite-State Transducers...

3.3 Ứng dụng

- Chuyển hướng cuộc gọi tự động, quay số bằng giọng nói, tìm kiếm bằng giọng nói.
- Dùng trong việc ra lệnh bằng giọng nói trên các máy bay quân sự.
- Dùng trong các ứng dụng dạy học ngoại ngữ.
- Phụ đề tự động trong các video.
- Điều khiển robot bằng giọng nói.
- Hỗ trợ người khuyết tật.
- Phiên dịch tự động.
- ...

3.4 Các vấn đề cần giải quyết

Speech to Text luôn đóng vai trò là một trong những phần quan trọng nhất trong một ứng dụng trợ lý ảo. Trong một hệ thống trợ lý ảo điển hình, sẽ có hai vấn đề lớn về speech recognition: dò tìm keyword và chuyển đổi lệnh của người dùng thành văn bản.

3.4.1 Dò tìm keyword

Mỗi hệ thống trợ lý ảo sẽ được xác định trước một keyword, keyword này sẽ được dùng để kích hoạt trợ lý ảo. Mỗi khi người dùng gọi keyword này, ứng dụng sẽ chuyển sang trạng thái thu âm để ghi nhận lệnh từ phía người dùng. Keyword này thường là

một từ hoặc cụm từ ngắn, dễ phát âm và dễ nhận biết, và thường chứa tên của ứng dụng, ví dụ như của Google Assistant là "OK Google", của Siri là "Hey Siri",...

Yêu cầu

Thứ nhất, chức năng dò tìm keyword của hệ thống phải hoạt động liên tục trong suốt quá trình ứng dụng chạy, do hệ thống phải có phản ứng ngay lập tức khi người dùng gọi keyword. Nếu chức năng này bị gián đoạn, hệ thống sẽ dễ bỏ lỡ mất keyword. Do đó, phần nhận biết tiếng nói trong chức năng này nên hoạt động offline, để tránh những gián đoạn trên đường mạng.

Thứ hai, tốc độ nhận biết keyword phải rất nhanh. Hệ thống sẽ liên tục thu âm từ microphone thành các frame âm thanh để xử lý liên tục trong thời gian thực, và phải liên tục kiểm tra các đoạn âm thanh liên tiếp nhau xem có chứa keyword hay không. Khi phát hiện ra một đoạn âm thanh thu được có chứa keyword, hệ thống phải ngay lập tức phản hồi lại cho người dùng và chuyển sang trạng thái thu âm lệnh của người dùng. Nếu tốc độ nhận biết keyword chậm, việc chuyển trạng thái sang thu âm sẽ bị trễ, và lệnh của người dùng khi thu vào có thể sẽ không đầy đủ, dẫn tới phản hồi sai.

Thứ ba, độ chính xác của việc nhận biết keyword phải đạt mức tương đối cao. Do từ khóa được chọn là một từ dễ đọc và dễ nhận biết nên độ chính xác của chức năng này không cần phải quá cao. Ngoài ra việc đặt ngưỡng nhận biết ở mức không quá cao sẽ làm hạn chế tối đa số lượng false negatives (trường hợp người dùng gọi keyword nhưng ứng dụng không nhận ra). Tuy nhiên điều đó sẽ làm gia tăng số lượng false positives (trường hợp người dùng không gọi keyword nhưng ứng dụng lại nhận ra), do đó cần phải tìm giải pháp để giảm số lượng false positives này.

Giải pháp

Trong hệ thống này, chúng tôi sẽ sử dụng thư viện pocketsphinx để cài đặt chức năng phát hiện keyword. Đây là một thư viện speech recognition có khả năng hoạt động offline, tốc độ xử lý nhanh, tuy nhiên độ lỗi của thư viện này là lớn hơn so với một vài thư viện khác.

Ngoài ra, ngay khi pocketsphinx nhận ra được keyword trong một đoạn âm thanh, hệ thống sẽ gửi đoạn âm thanh đó vào thư viện Google Speech to Text để kiểm tra lại

một lần nữa. Hai bước kiểm tra qua hai model khác nhau sẽ giúp hệ thống hạn chế được số lượng false positives.

3.4.2 Chuyển đổi lệnh người dùng thành văn bản

Sau khi phát hiện ra keyword, hệ thống sẽ được kích hoạt và bắt đầu thu âm lệnh của người dùng. Sau khi người dùng nói xong, hệ thống sẽ phải chuyển lệnh của người dùng sang dạng văn bản để có thể xử lý và hiểu được lệnh đó.

Yêu cầu

Phần chức năng chuyển đổi câu lệnh của người dùng thành văn bản chỉ bắt đầu hoạt động khi người dùng muốn ra lệnh, tức là sau khi người dùng gọi keyword, còn những khoảng thời gian khác, chức năng này sẽ nằm ở trạng thái chờ. Việc thu âm lệnh người dùng chỉ ngừng lại khi người dùng ngừng ra lệnh. Nếu hệ thống ngừng thu âm quá sớm, lệnh của người dùng thu vào sẽ bị thiếu và không chính xác. Nếu hệ thống ngừng thu âm quá muộn sẽ tạo ra độ trễ lớn từ lúc người dùng ra lệnh đến lúc ứng dụng phản hồi, và có thể tạp âm sau khi người dùng nói xong sẽ lọt vào phần speech recognition khiến kết quả nhận biết bị sai lệch.

Ngoài ra, độ chính xác của việc chuyển lệnh của người dùng về văn bản phải đạt mức rất cao, để đảm bảo hệ thống có thể hiểu đúng được ý định của người dùng.

Giải pháp

Trong hệ thống trợ lý ảo này, module chuyển đổi lệnh người dùng thành văn bản được cài đặt sử dụng thư viện Google Speech to Text. Đây là công cụ nhận biết tiếng nói của Google được đánh giá rất cao về độ chính xác, nhờ đó hệ thống sẽ đảm bảo hiểu đúng được hầu hết các lệnh của người dùng.

Để việc thu âm có thể ngừng đúng lúc, trong hệ thống sẽ có một giá trị thời gian T và một ngưỡng cường độ D. Khi âm thanh thu được từ microphone có cường độ nằm dưới mức D trong một khoảng thời gian T liên tục thì hệ thống sẽ xem như người dùng đã kết thúc việc ra lệnh và dừng trạng thái thu âm để bắt đầu xử lý đoạn âm thanh đã thu được.

3.5 Thư viện pocketsphinx

3.5.1 Tổng quan

Pocketsphinx là một thư viện speech recognition mã nguồn mở được phát triển bởi Viện Công nghệ Ngôn Ngữ, thuộc trường Đại học Carnegie Mellon (Mỹ). Pocketsphinx là phiên bản tối ưu của CMU Sphinx-II, cũng là một hệ thống nhận dạng tiếng nói khá nổi tiếng của ĐH Carnegie Mellon. Nhóm nghiên cứu này đã tối ưu hóa pocketsphinx rất nhiều về bộ nhớ và thuật toán, nhằm có thể tạo ra một thư viện nhận dạng tiếng nói liên tục có khả năng đưa vào các thiết bị cầm tay.

Hệ thống của pocketsphinx chủ yếu dựa trên những mô hình phổ biến của speech recognition như Gaussian Mixture Model và thuật toán Viterbi trên Hidden Markov Model, kết hợp với nhiều thuật toán hỗ trợ khác[2].

Pocketsphinx được đánh giá cao nhờ vào khả năng hoạt động offline và tốc độ xử lý khá nhanh. Ngoài ra, pocketsphinx có khả năng xử lý real-time liên tục trên stream âm thanh (thay vì phải hoàn tất việc thu âm và xử lý trên file âm thanh thu được).

3.5.2 Cách cài đặt

3.5.3 Cách sử dụng

các thông số của thư viện

3.5.4 Các ưu, nhược điểm

- **Ưu điểm:** Hỗ trợ xử lý offline, xử lý real-time liên tục, hỗ trợ chức năng dò tìm keyword.
- **Nhược điểm:** Độ chính xác khá kém. Độ lỗi word error rate khi thử nghiệm của pocketsphinx lên đến 13.95%[2].

3.5.5 Ứng dụng

Trong hệ thống trợ lý ảo của khóa luận này, pocketsphinx sẽ được sử dụng trong module WakeUp. Pocketsphinx sẽ giúp phát hiện ra khi người dùng gọi wake-up word,

qua đó kích hoạt hệ thống.

3.6 Thư viện Google Speech To Text

3.6.1 Tổng quan

Google Speech Recognition là hệ thống nhận biết tiếng nói do Google nghiên cứu và phát triển. Hệ thống này đã được Google tích hợp vào trợ lý ảo Google Assistant trên các thiết bị Android và loa thông minh Google Home, và cũng đã được đưa vào ứng dụng tìm kiếm Google Search. Google cũng đã đưa ra Google Speech API cho phép các lập trình viên sử dụng hệ thống speech recognition này vào các ứng dụng của họ. Google Speech to Text là một thư viện trên Python có khả năng nhận biết tiếng nói nhờ vào việc gọi đến Google Speech API.

Trong quá trình phát triển, các nhà nghiên cứu của Google đã sử dụng nhiều kỹ thuật khác nhau cho Google Speech Recognition, từ những kỹ thuật cổ điển như Gaussian Mixture Model, đến những kỹ thuật hiện đại hơn như Deep Neural Networks hay Long Short-term Memory Recurrent Neural Networks (LSTM RNNs)[1].

Nhờ sự thay đổi và tiến bộ liên tục, Google Speech Recognition được đánh giá rất cao về độ chính xác của mình. Tại Google I/O 2017, Google đã công bố độ chính xác của công nghệ nhận biết tiếng nói này đã đạt độ lỗi word error rate là 4.9%[4], tức là độ chính xác lên đến hơn 95%.

3.6.2 Cách cài đặt

3.6.3 Cách sử dụng

Các thông số của thư viện

3.6.4 Các ưu, nhược điểm

- **Ưu điểm:** Độ chính xác rất cao.
- **Nhược điểm:** Yêu cầu internet để hoạt động, không xử lý real-time trên stream âm thanh được mà phải gửi toàn bộ file âm thanh.

3.6.5 Ứng dụng

Trong hệ thống này, thư viện Google Speech to Text được dùng trong module SpeechToText. Module này sẽ đóng vai trò chuyển đổi những câu lệnh của người dùng từ dạng âm thanh sang dạng văn bản.

Chương 4

Text To Speech

Nội dung chương 4 sẽ giới thiệu tổng quan về bài toán Text to Speech, mô hình hoạt động, các ứng dụng của Text to Speech, các vấn đề cần giải quyết của module Text to Speech trong một hệ thống trợ lý ảo và cách giải quyết các vấn đề đó. Chương 4 cũng sẽ giới thiệu về chức năng, cách cài đặt, cách sử dụng cũng như các ưu nhược điểm của các thư viện iSpeech và Google Text to Speech.

4.1 Tổng quan

Text to Speech (TTS), hay còn gọi là hệ thống tổng hợp giọng nói, là một lĩnh vực trong khoa học máy tính. Text to Speech nghiên cứu phương pháp tạo ra giọng nói nhân tạo từ văn bản. Giọng nói nhân tạo này được đánh giá dựa trên hai tiêu chí: mức độ tự nhiên và mức độ dễ nghe. Mức độ tự nhiên chỉ sự tương đồng về ngữ điệu của giọng nói tổng hợp với giọng nói con người. Mức độ dễ nghe đánh giá khả năng phát âm rõ ràng, và khả năng nghe hiểu của con người với giọng nói tổng hợp. Việc có quá nhiều ngôn ngữ trên thế giới, cộng thêm việc mỗi ngôn ngữ lại có nhiều ngữ điệu khác nhau tùy vùng miền đã đặt ra những thách thức không hề đơn giản cho các nhà khoa học.

Text to Speech đã được bắt đầu phát triển từ rất lâu trước đây và đã trải qua một quá trình cải tiến lâu dài. Có thể nói khởi nguồn của nó là mô hình bắt chước giọng nói người với năm nguyên âm (u, e, o , a, i), được phát triển vào năm 1779 bởi nhà khoa học người Đan Mạch Christian Kratzenstein tại viện hàn lâm khoa học Nga. Từ đó đến nay, sau nhiều năm phát triển cải tiến, hệ thống tổng hợp giọng nói đã có nhiều

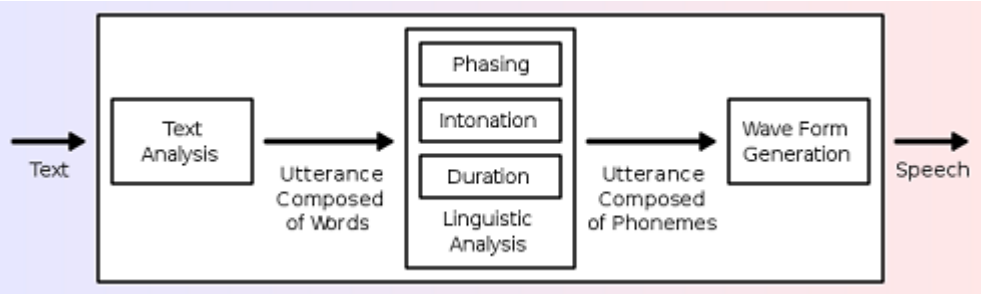
bước phát triển vượt bật, giọng nói tạo ra ngày càng giống với ngữ điệu người và hỗ trợ nhiều loại ngôn ngữ trên thế giới. Hiện nay, có rất nhiều công ty tham gia vào phát triển hệ thống Text to Speech, trong đó nổi bật là các công ty: Google, Microsoft, iSpeech, Amazon,...

4.2 Mô hình hoạt động

Một hệ thống Text to Speech thông thường bao gồm hai thành phần chính: front-end và back-end.

- Front-end, hay bộ phận tiền xử lý, thực hiện hai nhiệm vụ chính. Thứ nhất, chuyển đổi các ký tự số, ký tự đặc biệt, viết tắt trong văn bản về dạng viết đầy đủ, quá trình này gọi là quá trình chuẩn hóa (normalization). Thứ hai, phân phia và đánh dấu văn bản thành từng từ, nhóm từ, mệnh đề, câu văn và đoạn văn, quá trình này gọi là Tokenization. Sau đó gán mã phát âm tương ứng với từng từ của văn bản. Như vậy input của bộ phận front-end sẽ là văn bản, và output là một dạng mã hóa cách phát âm của văn bản đó.

- Back-end, còn gọi là bộ phận tổng hợp. Nhiệm vụ của phần này là tổng hợp các thông tin từ front-end thành giọng nói ở dạng sóng âm thanh.



Hình 4.1: Minh họa mô hình hoạt động của hệ thống Text to Speech

Có nhiều kỹ thuật dùng trong tổng hợp âm thanh. Tùy thuộc vào đặc tả của hệ thống, thiên về mức độ dễ nghe, thiên về mức độ tự nhiên, hay gia tăng đồng thời cả hai tính chất trên mà sẽ lựa chọn kỹ thuật thích hợp. Có hai kỹ thuật chính thường được dùng là tổng hợp ghép nối và tổng hợp cộng hưởng tần số, ngoài ra còn có một số kỹ thuật khác.

- **Tổng hợp ghép nối** Tổng hợp ghép nối dựa trên việc nối vào nhau các đoạn của một giọng nói đã được ghi âm. Thông thường, tổng hợp ghép nối tạo ra giọng nói tương đối tự nhiên. Tuy nhiên, giọng nói tự nhiên được ghi âm có sự thay đổi từ lần phát âm này sang lần phát âm khác, và công nghệ tự động hóa việc ghép nối các đoạn của sóng âm thỉnh thoảng tạo ra những tiếng cọ xát không tự nhiên ở phần ghép nối. Có ba kiểu tổng hợp ghép nối.

  – **Tổng hợp chọn đơn vị** Tổng hợp chọn đơn vị dùng một cơ sở dữ liệu lớn các giọng nói ghi âm (thông thường dài hơn 1 giờ đồng hồ ghi âm). Trong lúc ghi âm, mỗi câu phát biểu được tách ra thành các đơn vị khác như: các âm tỏ lời đơn lẻ, âm tiết, hình vị, từ, nhóm từ, và câu văn. Thông thường, việc tách ra như vậy cần một máy nhận dạng tiếng nói được đặt ở chế độ khớp với văn bản viết tương ứng với đoạn ghi âm, và dùng đến hiển thị sóng âm và phổ âm thanh. Một bảng tra các đơn vị được lập ra dựa trên các phần đã tách và các thông số âm học như tần số cơ bản, thời lượng, vị trí của âm tiết, và âm tỏ lời gần đó. Khi chạy, các câu phát biểu được tạo ra bằng cách xác định chuỗi đơn vị phù hợp nhất từ cơ sở dữ liệu. Quá trình này được gọi là chọn đơn vị, và thường cần dùng đến cây quyết định để thực hiện.

    Kỹ thuật chọn đơn vị tạo ra độ tự nhiên cao do không áp dụng các kỹ thuật xử lý tín hiệu số lên các đoạn giọng nói đã ghi âm, tuy rằng một số hệ thống có thể áp dụng xử lý tín hiệu tại các đoạn nối giữa các đơn vị để làm liền mạch kết quả sau khi ghép nối. Thực tế, các hệ thống chọn đơn vị có thể tạo ra giọng nói không thể phân biệt được với người thật. Tuy nhiên, để đạt độ tự nhiên cao, thường cần một cơ sở dữ liệu lớn chứa các đơn vị để lựa chọn; có thể lên tới vài gigabyte, tương đương với hàng chục giờ ghi âm.

  – **Tổng hợp âm kép** Tổng hợp âm kép dùng một cơ sở dữ liệu giọng nói nhỏ chứa tất cả các âm kép (chuyển tiếp âm thanh) xuất hiện trong ngôn ngữ đang xét. Số lượng âm kép phụ thuộc vào đặc tính ghép âm học của ngôn ngữ: tiếng Tây Ban Nha có 800 âm kép, tiếng Đức có 2500. Trong tổng hợp âm kép, chỉ có một ví dụ của âm kép được chứa trong cơ sở dữ liệu.

Khi chạy, lời văn được chồng lên các đơn vị này bằng kỹ thuật xử lý tín hiệu số như mã tiên đoán tuyến tính, PSOLA hay MBROLA.

Chất lượng của âm thanh tổng hợp theo cách này thường không cao bằng phương pháp chọn đơn vị nhưng tự nhiên hơn tổng hợp cộng hưởng tần số. Tổng hợp âm kép tạo ra các tiếng cọ xát ở phần ghép nối và đôi khi giọng nói kiểu robot do các kỹ thuật xử lý tín hiệu số gây ra. Lợi thế của phương pháp này là kích thước cơ sở dữ liệu nhỏ. Các ứng dụng thương mại của phương pháp này đang ít dần, tuy nhiên có nhiều hệ thống như thế này được phân phát tự do, và phục vụ cho nghiên cứu.

- **Tổng hợp chuyên ngành** Tổng hợp chuyên biệt ghép nối các từ và đoạn văn đã được ghi âm để tạo ra lời phát biểu. Nó được dùng trong các ứng dụng có các văn bản chuyên biệt cho một chuyên ngành, sử dụng lượng từ vựng hạn chế, như các thông báo chuyến bay hay dự báo thời tiết.

  Công nghệ này rất đơn giản, và đã được thương mại hóa từ lâu, đã đi vào các đồ vật như đồng hồ biết nói hay máy tính bỏ túi biết nói. Mức độ tự nhiên của các hệ thống này có thể rất cao vì số lượng các câu nói không nhiều và khớp với lời văn và âm điệu của giọng nói ghi âm. Tuy nhiên các hệ thống này bị hạn chế bởi cơ sở dữ liệu chuyên ngành, không phục vụ mọi mục đích mà chỉ hoạt động với các câu nói mà chúng đã được lập trình sẵn.

- **Tổng hợp cộng hưởng tần số** Tổng hợp cộng hưởng tần số không sử dụng bất cứ mẫu giọng thật nào khi chạy. Thay vào đó, tín hiệu âm thanh cho ra dựa trên một mô hình âm thanh. Các thông số như tần số cơ bản, sự phát âm, và mức độ tiếng ồn được thay đổi theo thời gian để tạo ra dạng sóng cho giọng nói nhân tạo. Phương pháp này đôi khi còn được gọi là tổng hợp dựa trên quy tắc, dù cho nhiều hệ thống ghép nối mẫu âm thanh thật cũng có dùng các thành phần dựa trên quy tắc.

  Nhiều hệ thống dựa trên tổng hợp cộng hưởng tần số tạo ra giọng nói nhân tạo, như giọng rôbốt, không tự nhiên, và phân biệt rõ ràng với giọng người thật. Tuy nhiên độ tự nhiên cao không phải lúc nào cũng là mục đích của hệ thống và hệ thống này cũng có các ưu điểm riêng của nó.

Hệ thống này nói khá dễ nghe, ngay cả ở tốc độ cao, không có tiếng cọ xát do ghép âm tạo ra. các hệ thống này hoạt động ở tốc độ cao, có thể hướng dẫn người khiếm thị nhanh chóng dò dẫm trên máy tính, bằng cách đọc to những gì hiện ra trên màn hình. Các hệ thống này cũng nhỏ gọn hơn các hệ thống ghép nối âm, vì không phải chứa cơ sở dữ liệu mẫu âm thanh lớn. Nó có thể dùng trong các hệ thống nhúng khi bộ nhớ và tốc độ xử lý có hạn. Hệ thống này cũng có khả năng điều khiển mọi khía cạnh của tín hiệu âm thanh đi ra, no cho ra một dải rộng các lời văn và ngữ điệu, và không chỉ thể hiện được câu nói thường hay câu hỏi, mà cả các trạng thái tình cảm thông qua âm điệu của giọng nói.

Các ví dụ về các hệ thống cho ra ngữ điệu chính xác (nhưng không cho ra ngay lập tức sau khi nhận đầu vào) là các công trình cuối những năm 1970 của đồ chơi Speak & Spell của Texas Instruments, và các trò chơi video của SEGA đầu những năm 1980 như: Astro Blaster, Zektor, Space Fury, và Star Trek. Hiện vẫn chưa có hệ thống cho ra intonation chính xác ngay sau khi nhận văn bản đầu vào.

- **Tổng hợp mô phỏng phát âm** Tổng hợp mô phỏng phát âm là các kỹ thuật tổng hợp giọng nói dựa trên mô hình máy tính của cơ quan phát âm của người và quá trình phát âm xảy ra tại đó. Hệ thống tổng hợp mô phỏng phát âm đầu tiên là ASY, thường được dùng cho các thí nghiệm trong nghiên cứu, được phát triển ở phòng thí nghiệm Haskins vào giữa những năm 1970 bởi Philip Rubin, Tom Baer, và Paul Mermelstein. ASY dựa trên mô hình cơ quan phát âm đã được tạo ra bởi phòng thí nghiệm Bell vào những năm 1960 và 1970 bởi Paul Mermelstein, Cecil Coker, và các đồng nghiệp khác. Tổng hợp mô phỏng phát âm đã từng chỉ là hệ thống dành cho nghiên cứu khoa học cho mãi đến những năm gần đây. Lý do là rất ít mô hình tạo ra âm thanh chất lượng đủ cao hoặc có thể chạy hiệu quả trên các ứng dụng thương mại. Một ngoại lệ là hệ thống dựa trên NeXT; vốn được phát triển và thương mại hóa bởi Trillium Sound Research Inc, ở Calgary, Alberta, Canada. Đây là một công ty tách ra từ Đại học Calgary nơi các nghiên cứu ban đầu đã được thực hiện. Theo sau các vụ chuyển nhượng các từng phần của NeXT (bắt đầu từ Steve Jobs vào cuối những năm 1980 và việc hợp nhất với Apple năm 1997), phần mềm của Trillium được phân phát

với giấy phéo tự do GPL. Dự án gnuspeech, một dự án của GNU, tiếp tục phát triển phần mềm này. Phần mềm gốc NeXT và các chuyển đổi sang cho Mac OS/X và GNUstep trong GNU/Linux có thể tìm thấy tại trang GNU savannah; chúng đều kèm theo tài liệu hướng dẫn trực tuyến và các bài viết liên quan đến lý thuyết nền tảng của công trình. Hệ thống, vốn được thương mại hóa lần đầu vào năm 1994, tạo ra một máy tổng hợp giọng nói dựa trên mô phỏng phát âm hoàn chỉnh, dựa trên mô hình ống dẫn sóng tương đương với cơ quan phát âm của người. Nó được điều khiển bởi Mô hình Phần Riêng biệt của Carré; bản thân mô hình này lại dựa trên công trình của Gunnar Fant và các người khác ở Phòng thí nghiệm Công nghệ Giọng nói Stockholm thuộc Viện Cộng nghệ Hoàng gia Thụy Điển về tổng hợp giọng nói cộng hưởng tần số. Công trình này cho thấy các cộng hưởng tần số trong ống cộng hưởng có thể được điều khiển bằng cách thay đổi tám tham số tương đồng với các cách phát âm tự nhiên của cơ quan phát âm của người. Hệ thống bao gồm một từ điển phát âm cùng với các quy tắc phát âm tùy thuộc ngữ cảnh để giúp ghép nối âm điệu và tạo ra các tham số phát âm; mô phỏng theo nhịp điệu và ngữ điệu thu được từ các kết quả nghiên cứu ngữ âm học.

- **Tổng hợp lai** Các hệ thống tổng hợp lai kết hợp các yếu tố của tổng hợp cộng hưởng tần số với tổng hợp ghép nối để giảm thiểu các tiếng cọ xát khi ghép nối các đoạn âm thanh.

  Một ví dụ là RecSimCat, phát triển bởi Shakti Singh Parmar có thể tạo ra giọng dễ nghe và tự nhiên.[cần dẫn nguồn]

- **Tổng hợp dựa trên HMM** Tổng hợp dựa trên HMM là một phương pháp dựa vào mô hình Markov ẩn (HMM, viết tắt cho thuật ngữ tiếng Anh Hidden Markov model). Trong hệ thống này, phổ tần số của giọng nói, tần số cơ bản, và thời lượng đều được mô phỏng cùng lúc bởi HMM. Dạng sóng của giọng nói được tạo từ mô hình Markov ẩn dựa trên tiêu chí khả thực cực đại.

4.3 Ứng dụng

Các hệ thống này có nhiều ứng dụng. Ví dụ như hệ thống này có thể giúp người có thị lực kém (hoặc khiếm thị) nghe được máy đọc ra văn bản; đặc biệt là các văn bản có thể xử lý trên máy tính. Hệ thống như vậy có thể lắp đặt trong phần mềm xử lý văn bản hay trình duyệt mạng.

4.4 Google Text To Speech

4.4.1 Tổng quan

4.4.2 Chức năng

4.4.3 Cách cài đặt

4.4.4 Cách sử dụng

4.4.5 Ưu, nhược điểm

4.5 iSpeech

4.5.1 Tổng quan

4.5.2 Chức năng

4.5.3 Cách cài đặt

4.5.4 Cách sử dụng

4.5.5 Ưu, nhược điểm

Chương 5

Intent Classification

5.1 Tổng quan

5.2 Mô hình hoạt động

5.3 Ứng dụng

5.4 Thư viện Rasa NLU

5.4.1 Tổng quan

5.4.2 Chức năng

5.4.3 Mô hình hoạt động

5.4.4 Cách cài đặt

5.4.5 Cách sử dụng

5.4.6 Chuẩn bị dữ liệu

5.4.7 Đánh giá model

5.4.8 Ứng dụng

Chương 6

Ứng dụng Alexa

6.1 Tổng quan

6.2 Mô hình hoạt động

6.2.1 Các module chính

Microphone

Chức năng: Các vấn đề và cách giải quyết:

Recorder

Wakeup

Text To Speech

Speech to Text

Intent Classification

Intent Processor

6.2.2 Luồng hoạt động giữa các module

6.3 Các chức năng chính

Thông báo giờ

Chức năng chi tiết: Cách thức hoạt động:

Thông báo thời tiết

Phát nhạc

Giao tiếp cơ bản

Trả lời câu hỏi Wh-question

Chương 7

Kết Luận và Hướng Phát Triển

7.1 Kết quả đạt được

7.1.1 Về mặt lý thuyết

7.1.2 Về mặt thực nghiệm

7.2 Hướng phát triển

TÀI LIỆU THAM KHẢO

[1] F. Beaufays, "Google research blog: The neural networks behind google voice transcription," https://research.googleblog.com/2015/08/the-neural-networks-behind-google-voice.html, Aug. 2015. 24

[2] D. Huggins-Daines, M. Kumar, A. Chan, A. W. Black, M. Ravishankar, and A. I. Rudnicky, "Pocketsphinx: A free, real-time continuous speech recognition system for hand-held devices," in *Acoustics, Speech and Signal Processing, 2006. ICASSP 2006 Proceedings. 2006 IEEE International Conference on*, vol. 1. IEEE, 2006, pp. I–I. 23

[3] A. McCarthy, "How do people use virtual assistants on their smartphones?" https://www.emarketer.com/Article/How-Do-People-Use-Virtual-Assistants-on-Their-Smartphones/1015251, Feb. 2017. 2, 3, 4

[4] E. Protalinski, "Google's speech recognition technology now has a 4.9% word error rate," https://venturebeat.com/2017/05/17/googles-speech-recognition-technology-now-has-a-4-9-word-error-rate/, May 2017. 24