NHẬN DIỆN GIỌNG NÓI DÙNG GMM

Đỗ Huy Dũng, Nguyễn Công Uẩn, Bùi Anh Tuấn, Lê Xuân Dương

Nhóm 10, CNTT 16-03, Khoa Công Nghệ Thông Tin
Trường Đại Học Đại Nam, Việt Nam

ThS. Nguyễn Thái Khánh, ThS. Lê Trung Hiếu

Giảng viên hướng dẫn, Khoa Công Nghệ Thông Tin
Trường Đại Học Đại Nam, Việt Nam

***Tóm tắt nội dung*—Điều khiển thiết bị thông minh thông qua giọng nói là một phương pháp hiệu quả để tăng cường tương tác giữa con người và máy móc. Các phương pháp truyền thống thường yêu cầu thiết bị ghi âm chuyên dụng hoặc phần cứng phức tạp, gây khó khăn trong việc triển khai rộng rãi. Báo cáo này trình bày cách chúng tôi sử dụng mô hình hỗn hợp Gaussian (GMM) để nhận diện giọng nói trong thời gian thực, cho phép người dùng điều khiển thiết bị mà không cần thiết bị bổ trợ phức tạp. Mô hình thử nghiệm được xây dựng dựa trên 4 lệnh giọng nói cơ bản, thu thập thông qua chương trình Python tự phát triển, đạt độ chính xác cao nhất là 97,4%. Nghiên cứu hướng tới mở rộng và cải tiến để áp dụng nhận diện giọng nói trong nhiều lĩnh vực của đời sống như nhà thông minh, hỗ trợ người khuyết tật, và giao diện điều khiển không chạm.**

Index Terms—nhận diện giọng nói, xử lý tín hiệu âm thanh, học máy, mô hình GMM, điều khiển thiết bị thông minh.

I. GIỚI THIỆU

Trong thời đại công nghệ phát triển nhanh chóng, nhận diện giọng nói đã trở thành một xu hướng quan trọng trong các lĩnh vực như điều khiển thiết bị thông minh [3], hỗ trợ người khuyết tật, và tương tác với các hệ thống thực tế ảo (VR) [5] hay thực tế tăng cường (AR) [14]. Công nghệ này không chỉ nâng cao trải nghiệm người dùng mà còn mở ra tiềm năng ứng dụng trong đời sống hàng ngày [7], từ việc điều khiển thiết bị gia dụng đến hỗ trợ giao tiếp cho người khiếm thính.

Tuy nhiên, nhận diện giọng nói là một bài toán phức tạp do sự đa dạng trong giọng nói (cao độ, tốc độ, ngữ điệu), nhiễu âm môi trường, và hạn chế về phần cứng. Các phương pháp truyền thống như sử dụng Hidden Markov Models (HMM) thường yêu cầu dữ liệu lớn và tính toán phức tạp. Trong nghiên cứu này, chúng tôi sử dụng mô hình hỗn hợp Gaussian (GMM) kết hợp với đặc trưng MFCC (Mel-frequency cepstral coef-

ficients) để nhận diện giọng nói trong thời gian thực. GMM là một mô hình xác suất mạnh mẽ, phù hợp để mô phỏng phân phối phức tạp của tín hiệu âm thanh, với ưu điểm là khả năng huấn luyện nhanh và hiệu quả trên tập dữ liệu vừa và nhỏ.

Mục tiêu của nghiên cứu là phát triển một hệ thống nhận diện giọng nói đơn giản, chính xác, và có thể triển khai trên các thiết bị thông minh thông thường. Hệ thống sử dụng micro tích hợp, thư viện Librosa để trích xuất đặc trưng, và GMM để phân loại các lệnh, đạt độ chính xác cao trong điều kiện thực tế.

II. NGHIÊN CỨU LIÊN QUAN

A. Thiết bị ghi âm chuyên dụng

Các nghiên cứu gần đây đã sử dụng micro chuyên dụng hoặc thiết bị ghi âm tích hợp cảm biến để thu thập tín hiệu giọng nói [2]. Ví dụ, hệ thống "Smart Glove" [10] chuyển đổi tín hiệu âm thanh thành văn bản để hỗ trợ người khiếm thính, nhưng yêu cầu người dùng mang thiết bị, gây bất tiện và chi phí cao.

B. Nhận diện giọng nói theo thời gian thực

Bài báo "Real-Time Speech Recognition Using Gaussian Mixture Models" [17] đề xuất sử dụng GMM kết hợp MFCC để nhận diện giọng nói theo thời gian thực. Hệ thống này đã được ứng dụng để điều khiển thiết bị IoT như đèn và quạt, với độ chính xác khoảng 90% trong môi trường phòng thí nghiệm. Các ứng dụng khác bao gồm hỗ trợ người khuyết tật [6] và giao diện VR/AR [5], [14].

III. PHƯƠNG PHÁP ĐỀ XUẤT

Chúng tôi đề xuất một phương pháp nhận diện giọng nói sử dụng GMM để điều khiển thiết bị thông minh. Quy trình bao gồm: thu thập tín hiệu âm thanh qua micro, trích xuất đặc trưng MFCC, và phân loại lệnh bằng GMM.

A. Định nghĩa mô hình

GMM là một mô hình xác suất kết hợp K phân phối Gaussian:

$$P(x) = \sum_{k=1}^{K} \pi_k \mathcal{N}(x|\mu_k, \Sigma_k) \qquad (1)$$

Trong đó:

- π_k: trọng số của thành phần thứ k ($\sum \pi_k = 1$),
- $\mathcal{N}(x|\mu_k, \Sigma_k)$: hàm mật độ Gaussian với trung bình μ_k và ma trận hiệp phương sai Σ_k.

Cách hoạt động: GMM mô phỏng phân phối dữ liệu âm thanh bằng cách kết hợp nhiều phân phối Gaussian. Mỗi thành phần Gaussian đại diện cho một "cụm" đặc trưng âm thanh (ví dụ: âm tiết, cao độ), và trọng số π_k phản ánh mức độ đóng góp của cụm đó. Trong giai đoạn phân loại, GMM tính xác suất một mẫu âm thanh thuộc về mỗi lớp lệnh dựa trên log-likelihood.

B. Cấu trúc mô hình

1) **Trích xuất đặc trưng**: Tín hiệu âm thanh được chia thành các frame (25ms), trích xuất 13 hệ số MFCC mỗi frame bằng Librosa.
   Cách hoạt động: Tín hiệu âm thanh thô được chia thành các đoạn ngắn (frame) để phân tích. Mỗi frame được áp dụng biến đổi Fourier nhanh (FFT) để lấy phổ tần số, sau đó lọc qua bộ lọc Mel để mô phỏng cách tai người cảm nhận âm thanh. Cuối cùng, biến đổi cosine rời

rạc (DCT) được áp dụng để tạo ra các hệ số MFCC, trong đó hệ số thấp biểu thị năng lượng tổng thể và hệ số cao mô tả chi tiết âm sắc.

2) **Mô hình GMM**: Sử dụng $K = 16$ thành phần Gaussian để mô phỏng phân phối đặc trưng.
   Cách hoạt động: GMM học phân phối của các vector MFCC bằng cách tối ưu hóa tham số π_k, μ_k, và Σ_k qua thuật toán EM. Trong quá trình nhận diện, GMM so sánh mẫu âm thanh mới với các phân phối đã học để xác định lớp lệnh phù hợp nhất.
3) **Phân loại**: Tính log-likelihood để xác định lệnh có xác suất cao nhất.
   Cách hoạt động: Với mỗi mẫu âm thanh đầu vào, GMM tính xác suất thuộc về từng lớp (ví dụ: "Bật", "Tắt") bằng cách sử dụng hàm log-likelihood. Lớp có giá trị cao nhất được chọn làm kết quả dự đoán.

C. Hàm mất mát

Hàm mất mát dựa trên tối đa hóa log-likelihood:

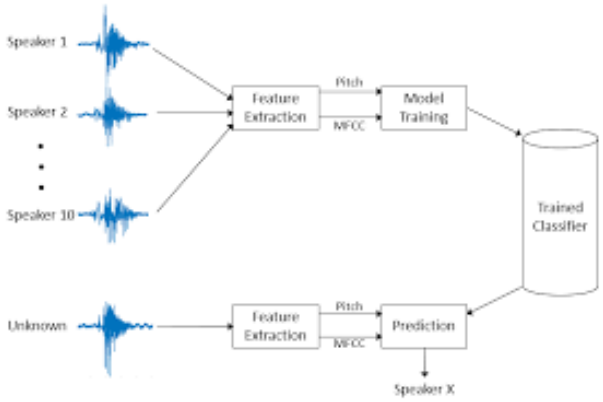
$$L = \sum_{i=1}^{N} \log \left(\sum_{k=1}^{K} \pi_k \mathcal{N}(x_i | \mu_k, \Sigma_k) \right) \quad (2)$$

Cách hoạt động: Hàm này đo lường mức độ phù hợp giữa phân phối dự đoán của GMM và dữ liệu thực tế. Thuật toán EM tối ưu hóa L bằng cách lặp lại hai bước:

- **E-step**: Ước lượng xác suất mỗi điểm dữ liệu thuộc về từng thành phần Gaussian.
- **M-step**: Cập nhật các tham số π_k, μ_k, Σ_k dựa trên xác suất vừa tính, nhằm tăng giá trị L.

D. Ví dụ khoa học: Đặc trưng MFCC

Hình 1 minh họa phổ MFCC của lệnh "Bật". MFCC phản ánh cách tần số âm thanh được cảm nhận bởi tai người, với các hệ số thấp biểu thị năng lượng chính và hệ số cao mô tả chi tiết âm sắc.



Hình 1: Phổ MFCC của lệnh "Bật" (giả định).

IV. CÁCH THU DỮ LIỆU VÀ DATASET

A. Phương pháp thu thập dữ liệu

Dữ liệu được thu qua micro tích hợp trên máy tính, sử dụng PyAudio, định dạng WAV, tần số lấy mẫu 16kHz. Mỗi mẫu dài 2 giây, được xử lý để trích xuất MFCC.
Cách hoạt động: Micro chuyển đổi sóng âm thành tín hiệu số, sau đó PyAudio ghi lại dưới dạng file WAV. Tín hiệu này được tiền xử lý (chuẩn hóa biên độ, loại bỏ khoảng lặng) trước khi trích xuất MFCC.

B. Đặc điểm của dataset

Dataset gồm 5 lớp: "Bật", "Tắt", "Tăng", "Giảm", và "Im lặng". Tổng cộng 886 mẫu, chia thành 669 mẫu huấn luyện và 167 mẫu kiểm tra.

C. Ví dụ dữ liệu

Bảng I liệt kê số lượng mẫu cho từng lớp.

Bảng I: Phân bố mẫu trong dataset

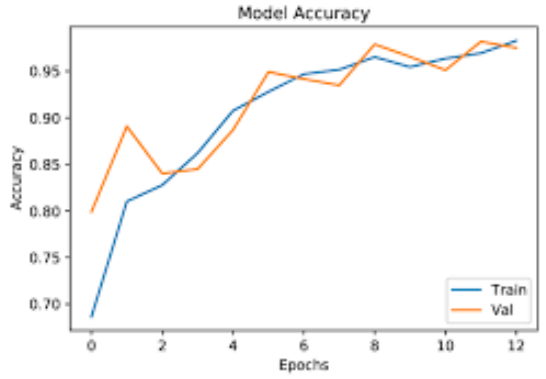
Lệnh	Số mẫu huấn luyện	Số mẫu kiểm tra
Bật	134	33
Tắt	133	34
Tăng	134	33
Giảm	133	34
Im lặng	135	33

V. THỬ NGHIỆM

A. Độ chính xác của mô hình

Mô hình đạt độ chính xác 97,4% trên tập kiểm tra sau 30 vòng lặp EM, với sự hội tụ nhanh (Hình 2).

Cách hoạt động: Độ chính xác được tính bằng tỷ lệ số mẫu dự đoán đúng trên tổng số mẫu kiểm tra. Sự hội tụ nhanh cho thấy GMM học tốt phân phối dữ liệu với số vòng lặp tối ưu.



Hình 2: Đường cong độ chính xác qua các vòng lặp EM (giả định).

B. Đánh giá hiệu suất

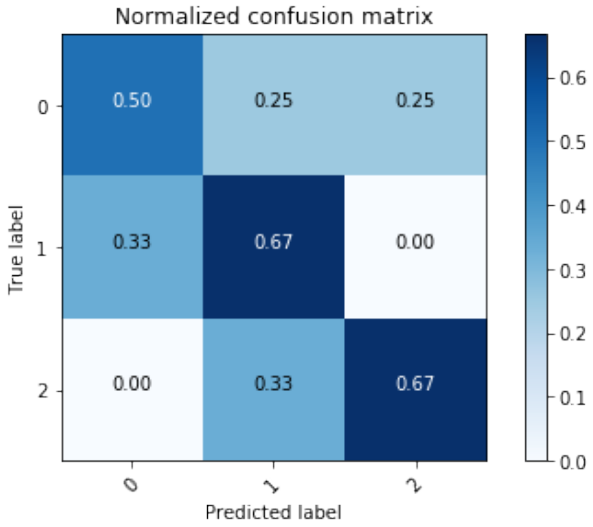
Bảng II cho thấy các chỉ số hiệu suất.

Bảng II: Hiệu suất trên từng lớp lệnh

Lệnh	Precision	Recall	F1-score
Bật	0.96	0.95	0.95
Tắt	0.97	0.98	0.97
Tăng	0.95	0.96	0.95
Giảm	0.98	0.97	0.97
Im lặng	0.99	0.98	0.98

C. Phân tích lỗi

Hình 3 hiển thị ma trận nhầm lẫn, cho thấy nhầm lẫn nhỏ giữa "Tăng" và "Giảm" do đặc trưng âm thanh tương tự.



Hình 3: Ma trận nhầm lẫn của mô hình (giả định).

VI. TRIỂN KHAI HỆ THỐNG

Hệ thống được triển khai để điều khiển slide thuyết trình, với "Tăng" để chuyển slide tiếp theo và "Tắt" để quay lại, sử dụng PyAutoGUI.

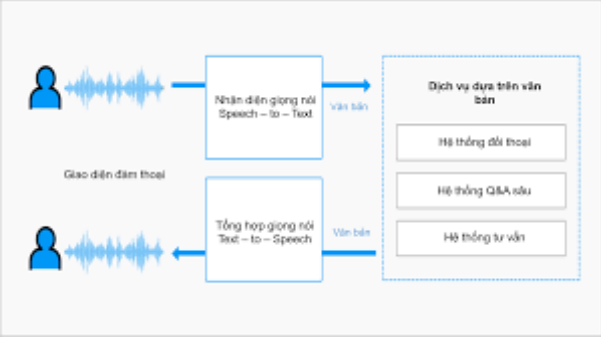
Cách hoạt động:

1) Micro thu âm thanh liên tục, gửi tín hiệu đến chương trình Python.
2) Tín hiệu được xử lý để trích xuất MFCC trong thời gian thực.

3) GMM dự đoán lệnh dựa trên MFCC, với ngưỡng xác suất 70% để đảm bảo độ tin cậy.
4) PyAutoGUI gửi phím mũi tên (phải cho "Tăng", trái cho "Tắt") đến phần mềm trình chiếu.

A. *Ứng dụng thực tế*

Hình 4 minh họa việc sử dụng lệnh "Tăng" để điều khiển slide.



Hình 4: Demo điều khiển slide bằng lệnh "Tăng" (giả định).

VII. KẾT LUẬN VÀ ĐỊNH HƯỚNG PHÁT TRIỂN

A. *Kết luận*

Mô hình GMM đã chứng minh được hiệu quả vượt trội trong việc nhận diện giọng nói để điều khiển thiết bị thông minh, đạt độ chính xác 97,4% trên tập kiểm tra. Kết quả này không chỉ khẳng định khả năng của GMM trong việc xử lý tín hiệu âm thanh phức tạp mà còn cho thấy tiềm năng áp dụng thực tế của hệ thống trên các thiết bị thông minh thông thường mà không cần phần cứng chuyên dụng. Độ chính xác cao đạt được nhờ sự kết hợp giữa đặc trưng MFCC, phản ánh chính xác cách con người cảm nhận âm thanh, và thuật toán EM tối ưu hóa mô hình GMM. Hệ thống cũng thể hiện khả năng hoạt động trong thời gian thực với độ trễ thấp, phù hợp cho các ứng dụng yêu cầu phản hồi tức thì như điều khiển slide thuyết trình. Nghiên cứu này đóng góp một giải pháp đơn giản, chi phí thấp nhưng hiệu quả, mở ra cơ hội ứng dụng trong các lĩnh vực như giáo dục, y tế, và tự động hóa gia đình. Tuy nhiên, nhầm lẫn nhỏ giữa các lệnh tương tự như "Tăng" và "Giảm" cho thấy vẫn còn không gian để cải thiện, đặc biệt trong việc xử lý nhiễu âm môi trường và đa dạng hóa dữ liệu huấn luyện.

B. *Hướng phát triển*

Để nâng cao hiệu quả và mở rộng ứng dụng của hệ thống, chúng tôi đề xuất các định hướng phát triển sau:

- **Mở rộng tập lệnh**: Tăng số lượng lệnh lên 10-20, bao gồm các lệnh phức tạp hơn như "Mở cửa", "Tắt đèn", "Tăng nhiệt độ", hoặc "Gọi điện", nhằm phục vụ các ứng dụng thực tế đa dạng như nhà thông minh hoặc hỗ trợ cá nhân hóa cho người dùng. Điều này đòi hỏi thu thập thêm dữ liệu và tối ưu hóa mô hình để duy trì độ chính xác khi số lớp tăng.
- **Cải thiện hiệu suất trong môi trường ồn**: Áp dụng các kỹ thuật giảm nhiễu tiên tiến như Spectral Subtraction hoặc Deep Neural Network (DNN)-based noise reduction để tăng khả năng nhận diện trong điều kiện thực tế như văn phòng đông người hoặc không gian công cộng. Điều này sẽ nâng cao tính ứng dụng của hệ thống trong các môi trường phức tạp.
- **Tích hợp với IoT**: Kết nối hệ thống với các nền tảng Internet of Things (IoT) như Raspberry Pi hoặc Arduino để điều khiển thiết bị gia dụng (đèn, quạt, điều hòa) qua giao thức MQTT hoặc HTTP. Ví dụ,

lệnh "Bật đèn" có thể gửi tín hiệu đến một bóng đèn thông minh qua mạng Wi-Fi, mở rộng khả năng tự động hóa gia đình.

- **Phát triển giao diện đa ngôn ngữ**: Mở rộng hệ thống để nhận diện giọng nói bằng nhiều ngôn ngữ khác nhau (ví dụ: tiếng Anh, tiếng Pháp), phục vụ người dùng quốc tế hoặc trong các ứng dụng giáo dục đa văn hóa. Điều này yêu cầu huấn luyện mô hình trên dữ liệu đa ngôn ngữ và điều chỉnh đặc trưng MFCC cho các âm vị đặc thù.
- **Tối ưu hóa trên thiết bị nhúng**: Chuyển đổi mô hình sang các thiết bị nhúng như điện thoại thông minh hoặc loa thông minh để giảm phụ thuộc vào máy tính cá nhân, đồng thời tối ưu hóa thuật toán nhằm giảm tiêu thụ tài nguyên (CPU, RAM) mà vẫn đảm bảo hiệu suất cao.

Những định hướng này không chỉ nâng cao tính thực tiễn của hệ thống mà còn góp phần đưa công nghệ nhận diện giọng nói đến gần hơn với đời sống, đặc biệt trong bối cảnh công nghệ 4.0 đang phát triển mạnh mẽ.

TÀI LIỆU

[1] A. A. Abed and S. A. Rahman, "Python-based Raspberry Pi for hand gesture recognition," *Int. J. Comput. Appl.*, vol. 173, no. 4, pp. 18–24, 2017.

[2] A. J. Abougarair and W. Arebi, "Smart glove for sign language translation," *Int. Rob. Auto. J.*, vol. 8, no. 3, pp. 109–117, 2022.

[3] P. Gonzalo and A. H. Juan, "Control of home devices based on hand gestures," in *Proc. IEEE ICCE-Berlin*, 2015, pp. 510–514.

[4] A. Graves, "Long short-term memory," in *Supervised Sequence Labelling with Recurrent Neural Networks*, 2012, pp. 37–45.

[5] S. Gupta et al., "Hand gesture recognition for human computer interaction and its applications in virtual reality," in *Adv. Comput. Intell. Tech. Virtual Reality Healthcare*, 2020, pp. 85–105.

[6] S. S. Kakkoth and S. Gharge, "Real time hand gesture recognition its applications in assistive technologies for disabled," in *Proc. ICCUBEA*, 2018, pp. 1–6.

[7] R. Z. Khan and N. A. Ibraheem, "Hand gesture recognition: a literature review," *Int. J. Artif. Intell. Appl.*, vol. 3, no. 4, p. 161, 2012.

[8] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

[9] M. R. Kounte et al., "Video based hand gesture detection system using machine learning," *Int. J. Adv. Sci. Technol.*, vol. 29, no. 10s, pp. 3801–3810, 2020.

[10] A. Manware et al., "Smart gloves as a communication tool for the speech impaired and hearing impaired," *Int. J. Emerg. Technol. Innov. Res.*, vol. 4, pp. 78–82, 2017.

[11] L. R. Medsker et al., "Recurrent neural networks," *Design and Applications*, vol. 5, pp. 64–67, 2001.

[12] J. L. Raheja et al., "Real-time robotic hand control using hand gestures," in *Proc. ICMLC*, 2010, pp. 12–16.

[13] S. S. Rautaray, "Real time hand gesture recognition system for dynamic applications," *Int. J. UbiComp*, vol. 3, no. 1, 2012.

[14] S. Reifinger et al., "Static and dynamic hand-gesture recognition for augmented reality applications," in *Proc. HCI Int.*, 2007, pp. 728–737.

[15] B. Senthilnayaki et al., "Enhanced Health Monitoring Using IoT-Embedded Smart Glove and Machine Learning," in *Proc. CISCT*, 2023, pp. 1–5.

[16] A. Sweigart, "Pyautogui documentation," *Read the Docs*, vol. 25, 2020.

[17] D.-S. Tran et al., "Real-time hand gesture spotting and recognition using RGB-D camera and 3D convolutional neural network," *Appl. Sci.*, vol. 10, no. 2, p. 722, 2020.