

TRƯỜNG ĐẠI HỌC KHOA HỌC TỰ NHIÊN

KHOA CÔNG NGHỆ THÔNG TIN

BỘ MÔN CÔNG NGHỆ PHẦN MỀM

TRẦN NGỌC QUANG – NGUYỄN HOÀNG QUYÊN

**XÂY DỰNG MÔ HÌNH
NHẬN DẠNG ÂM THANH TIẾNG VIỆT**

KHÓA LUẬN TỐT NGHIỆP CỬ NHÂN CNTT

TP.HCM, 2021

TRƯỜNG ĐẠI HỌC KHOA HỌC TỰ NHIÊN

KHOA CÔNG NGHỆ THÔNG TIN

BỘ MÔN CÔNG NGHỆ PHẦN MỀM

TRẦN NGỌC QUANG - 1712706

NGUYỄN HOÀNG QUYÊN - 1712712

XÂY DỰNG MÔ HÌNH

NHẬN DẠNG ÂM THANH TIẾNG VIỆT

KHÓA LUẬN TỐT NGHIỆP CỬ NHÂN CNTT

GIÁO VIÊN HƯỚNG DẪN

TS. NGÔ HUY BIÊN

KHOÁ 2017 - 2021

NHẬN XÉT CỦA GIÁO VIÊN HƯỚNG DẪN

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

TpHCM, ngày . . . tháng . . . năm 2021

Giáo viên hướng dẫn

[Kí tên và ghi rõ họ tên]

NHẬN XÉT CỦA GIÁO VIÊN PHẢN BIỆN

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

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

TpHCM, ngày . . . tháng . . . năm 2021

Giáo viên phản biện

[Kí tên và ghi rõ họ tên]

LỜI CẢM ƠN

Chúng em xin chân thành cảm ơn đến quý Thầy Cô trong Khoa Công Nghệ Thông Tin, trường Đại Học Khoa Học Tự Nhiên đã quan tâm, tận tình giúp đỡ và giảng dạy cho chúng em trong suốt quá trình học tập tại trường, qua đó tạo điều kiện thuận lợi cho chúng em học tập, áp dụng các kiến thức đã học để hoàn thành đề tài tốt nghiệp này.

Đặc biệt, chúng em xin gửi lời cảm ơn sâu sắc đến Thầy – Tiến sĩ Ngô Huy Biên - người đã luôn tận tình hướng dẫn, đưa ra những định hướng cho hướng đi của luận văn, nhiệt tình chỉ bảo, góp ý cho chúng em trong các vấn đề về nội dung, kiến thức suốt quá trình thực hiện khóa luận. Chúng em xin gửi đến thầy lời tri ân sâu sắc nhất.

Đồng thời chúng em cũng không quên gửi những lời cảm ơn chân thành đến những người thân trong gia đình và bạn bè đã luôn động viên, ủng hộ vật chất lẫn tinh thần cho chúng em trong thời gian thực hiện khóa luận.

Mặc dù đã cố gắng hoàn thành khóa luận, song do thời gian thực hiện khóa luận cùng với khả năng nghiên cứu và kinh nghiệm của bản thân có hạn nên chúng em khó tránh khỏi những hạn chế và thiếu sót. Vì vậy, chúng em rất mong nhận được sự góp ý và chỉ bảo của quý Thầy Cô.

Chúng em xin chân trọng cảm ơn!

TpHCM, ngày . . . tháng . . . năm 2019

Sinh viên

ĐỀ CƯƠNG CHI TIẾT

Tên đề tài: Xây dựng mô hình nhận dạng âm thanh tiếng Việt		
Giáo viên hướng dẫn: TS. Ngô Huy Biên		
Thời gian thực hiện: 14/09/2020 đến 19/03/2021		
Sinh viên thực hiện: Trần Ngọc Quang – 1712706, Nguyễn Hoàng Quyên - 1712712		
Loại đề tài: Nghiên cứu		
Nội dung đề tài: Trình bày lý do xây dựng mô hình nhận dạng âm thanh tiếng Việt. Trình bày lý thuyết nền tảng và giải pháp để xử lý việc nhận một tập tin âm thanh tiếng Việt và xuất ra nội dung văn bản ở dạng Tiếng Việt. Xây dựng, thu thập dữ liệu, và đào tạo mô hình để nhận một tập tin âm thanh tiếng Việt và xuất ra nội dung ở dạng văn bản Xây dựng dịch vụ web (API) để nhận một tập tin âm thanh tiếng Việt và xuất ra nội dung ở dạng văn bản. Website mẫu việc sử dụng API của mô hình dịch máy từ tiếng Anh sang tiếng Việt đã xây dựng. Cải tiến độ chính xác của mô hình với mục tiêu là 75%. Đề xuất giải pháp cải thiện độ chính xác của mô hình và định hướng mở rộng cho ứng dụng của mô hình.		
Kế hoạch thực hiện:		
Thời gian thực hiện	Công việc thực hiện	Người thực hiện
7/9/2020-13/9/2020	- Nhận đề tài - Xây dựng bản kế hoạch sơ bộ cho các công việc cần thực hiện	Quang, Quyên

14/9/2020-20/9/2020	<ul style="list-style-type: none"> - Tìm hiểu và phân tích các yêu cầu về kiến trúc nền cho đề tài. - Khảo sát, dùng thử các hệ thống cung cấp dịch vụ mẫu có sẵn trên thị trường: fpt.ai, vais.vn,... - Tạo Trello 	Quang, Uyên
21/9/2020-27/9/2020	<ul style="list-style-type: none"> - Thống nhất nội dung chính của ứng dụng demo việc sử dụng API. - Biên soạn đề cương cho luận văn (dạng slide). 	Quang, Uyên
28/9/2020-11/10/2020	<ul style="list-style-type: none"> - Tìm hiểu lý thuyết nền tảng trong máy học. - Biên soạn đề cương cho luận văn (dạng word). - Tìm hiểu lý thuyết nền tảng trong việc nhận dạng giọng nói. 	Quang, Uyên
14/10/2020-1/11/2020	<ul style="list-style-type: none"> - Tìm hiểu về các thư viện Scikit-Learn, Tensorflow, Keras. - Viết chương 1 luận văn. - Tìm hiểu các model và kiến trúc, chạy thử các ví dụ để đánh giá. - Chỉnh sửa chương 1 luận văn. 	Quang, Uyên
2/11/2020-8/11/2020	<ul style="list-style-type: none"> - Chạy thử mô hình nhận dạng âm thanh Tiếng Việt sang văn bản. 	Quang, Uyên
9/11/2020-22/11/2020	<ul style="list-style-type: none"> - Thu thập dữ liệu âm thanh. - Chỉnh sửa dữ liệu âm thanh. 	Quang, Uyên
23/11/2020-29/11/2020	<ul style="list-style-type: none"> - Tìm hiểu và xây dựng mô hình nhận dạng âm thanh Tiếng Việt. 	Quang, Uyên

30/11/2020-13/12/2020	- Tiếp tục xây dựng mô hình nhận dạng âm thanh Tiếng Việt.	Quang, Uyên
14/12/2020-20/12/2020	- Huấn luyện mô hình. - Viết chương 2 luận văn.	Quang, Uyên
21/12/2020-31/12/2020	- Cải tiến mô hình. - Chỉnh sửa chương 2 luận văn.	Quang, Uyên
4/1/2020-17/1/2020	- Viết chương 3 luận văn. - Chỉnh sửa chương 3 luận văn.	Quang, Uyên
18/1/2020-31/1/2020	- Xây dựng và triển khai hệ thống cung cấp dịch vụ web (API). - Viết chương 4 luận văn.	Quang, Uyên
1/2/2020-7/2/2020	- Xây dựng ứng dụng demo việc sử dụng API trên nền tảng web. - Chỉnh sửa chương 4 luận văn.	Quang, Uyên
8/2/2020-14/2/2020	- Viết chương 5 luận văn. - Chỉnh sửa chương 5 luận văn.	Quang, Uyên
15/2/2020-21/2/2020	- Hoàn thành luận văn. - Chỉnh sửa và cải thiện hiệu năng ứng dụng demo. - Nâng cấp mô hình hoàn thiện hơn. - Cải thiện hiệu năng hệ thống cung cấp dịch vụ web (API).	Quang, Uyên
22/2/2020-28/2/2020	- Hoàn chỉnh cuốn luận văn.	Quang, Uyên
1/3/2020-7/3/2020	- Hoàn chỉnh slide trình bày. - Hoàn chỉnh sản phẩm khoá luận.	Quang, Uyên

Xác nhận của giáo viên hướng dẫn

Ngày . . . tháng . . . năm 2020

	Sinh viên thực hiện
--	----------------------------

MỤC LỤC

DANH SÁCH HÌNH	6
DANH SÁCH BẢNG	9
CHƯƠNG 1. GIỚI THIỆU LUẬN VĂN.....	10
1.1. GIỚI THIỆU ĐỀ TÀI.....	10
1.2. LÝ DO LỰA CHỌN ĐỀ TÀI.....	11
1.3. HƯỚNG PHÁT TRIỂN CỦA LUẬN VĂN	12
1.4. MỤC TIÊU CỦA LUẬN VĂN.....	12
1.5. PHẠM VI ĐỀ TÀI.....	13
CHƯƠNG 2. VẤN ĐỀ THỰC TẾ VÀ ĐỘNG LỰC PHÁT TRIỂN.....	15
2.1. PHỤC VỤ NHU CẦU TRONG CUỘC SỐNG HÀNG NGÀY	15
2.2. ÁP DỤNG VÀO NHIỀU LĨNH VỰC TRONG CUỘC SỐNG.....	15
2.3. TIỀM NĂNG KINH TẾ CAO.....	16
2.4. PHỤC VỤ NHU CẦU SỬ DỤNG CÔNG NGHỆ CHO ĐA DẠNG NGƯỜI TRONG XÃ HỘI.....	17
2.5. PHỤC VỤ TRONG CÁC CUỘC HỌP, CUỘC THẢO LUẬN.....	18
CHƯƠNG 3. CÁC CÔNG TRÌNH NGHIÊN CỨU SẴN CÓ.....	19
3.1. CÁC KẾT QUẢ NGHIÊN CỨU SẴN CÓ.....	19
3.2. CÁC MÔ HÌNH, PHƯƠNG PHÁP, THUẬT TOÁN	19
3.3. NHẬN DẠNG GIỌNG NÓI TỰ ĐỘNG TỪ ĐẦU ĐẾN CUỐI (END – TO – END).....	21
3.3.1. Mô hình nhận dạng tiếng nói đầu cuối (end -to-end) dựa trên kỹ thuật phân loại thời gian kết nối (CTC)	22
3.3.1.1. Giới thiệu.....	22

3.3.1.2. Kỹ thuật phân loại thời gian kết nối (CTC)	22
3.3.2. Mô hình nhận dạng tiếng nói với bộ mã hóa và bộ giải mã dựa trên sự chú ý	24
3.3.2.1. Đặt vấn đề	24
3.3.2.2. Bộ mã hóa, bộ giải mã dựa trên sự chú ý.....	25
3.3.2.3. Kiến trúc mô hình.....	27
3.4. TỔNG KẾT.....	30
CHƯƠNG 4. PHƯƠNG PHÁP NGHIÊN CỨU CỦA ĐỀ TÀI.....	32
4.1. TỔNG QUAN KIẾN TRÚC HỆ THỐNG	32
4.2. PHƯƠNG PHÁP TIỀN XỬ LÝ ÂM THANH.....	34
4.2.1. Tổng quan giải pháp.....	34
4.2.2. Chi tiết giải pháp	35
4.3. PHƯƠNG PHÁP XÂY DỰNG MÔ HÌNH ÂM THANH	39
4.3.1. Tổng quan giải pháp.....	39
4.3.2. Mô hình mạng nơ-ron hồi quy RNN.....	39
4.3.3. Chuẩn hóa hàng loạt (Batch normalization).....	44
4.3.4. SortaGrad.....	46
4.3.5. Biến đổi tần số (Frequency Convolution)	46
4.3.6. Bước sóng (Striding).....	47
4.3.7. Bộ giải mã	47
4.3.7.1. Bộ giải mã tham lam	47
4.3.7.2. Bộ giải mã tìm kiếm chùm với mô hình ngôn ngữ	49
4.4. PHƯƠNG PHÁP HẬU XỬ LÝ VĂN BẢN KẾT QUẢ	53
4.4.1. Mô hình dấu câu	53
4.4.1.1. Vấn đề thực tế	53

4.4.1.2. Giải pháp	53
4.4.2. Nhận dạng từ Tiếng Anh thông dụng	54
4.4.2.1. Đặt vấn đề	54
4.4.2.2. Giải pháp	54
4.5. GIẢI PHÁP XÂY DỰNG MÁY CHỦ.....	55
4.6. GIẢI PHÁP XÂY DỰNG ỨNG DỤNG MẪU.....	56
4.7. TỔNG KẾT.....	58
CHƯƠNG 5. THỬ NGHIỆM VÀ ĐÁNH GIÁ	59
5.1. THU THẬP DỮ LIỆU HUẤN LUYỆN	59
5.1.1. Thu thập dữ liệu cho mô hình âm thanh.....	59
5.1.1.1. FPT, VIVOS.....	60
5.1.1.2. VINBIGDATA.....	60
5.1.2. Thu thập dữ liệu cho mô hình ngôn ngữ	60
5.2. HUẤN LUYỆN MÔ HÌNH.....	61
5.2.1. Tiền xử lý dữ liệu	61
5.2.2. Sửa chữa, cải tiến mã nguồn, dữ liệu	64
5.2.2.1. Dữ liệu huấn luyện	64
5.2.2.2. Mã nguồn	65
5.2.3. Cấu hình	67
5.2.4. Xây dựng mô hình ngôn ngữ	67
5.2.5. Hậu xử lý văn bản kết quả.....	68
5.3. CẢI TIẾN ĐỘ CHÍNH XÁC MÔ HÌNH.....	69
5.3.1. Thay đổi tham số huấn luyện	69
5.3.1.1. Điều chỉnh epoch.....	70

5.3.1.2. Điều chỉnh tham số của tìm kiếm tham lam.....	72
5.3.1.3. Điều chỉnh tham số batch_size.....	74
5.3.1.4. Điều chỉnh tham số rnn_type	76
5.3.1.5. Điều chỉnh tham số hidden layer.....	78
5.3.1.6. Điều chỉnh tham số hidden size	79
5.3.2. Thu thập thêm dữ liệu huấn luyện.....	82
5.4. CÁC MÔ HÌNH SO SÁNH	82
5.5. KẾT QUẢ THỰC NGHIỆM.....	83
5.6. ĐÓNG GÓI MÔ HÌNH.....	84
5.7. CÔNG CỤ XỬ LÝ	85
5.7.1.1. Công cụ phục vụ xây dựng dữ liệu mô hình ngôn ngữ	85
5.7.1.2. Công cụ phục vụ tiền xử lí âm thanh	87
5.8. LỖI HUẤN LUYỆN MÔ HÌNH	88
5.9. XÂY DỰNG MÁY CHỦ	90
5.10. CÀI ĐẶT CHỨC NĂNG GHI ÂM TRÊN WEBSITE	91
5.11. CÀI ĐẶT CHỨC NĂNG TẢI LÊN TẬP TIN ÂM THANH TRÊN ỨNG DỤNG	93
5.12. TỔNG KẾT.....	94
CHƯƠNG 6. TỔNG KẾT VÀ ĐÁNH GIÁ	95
6.1. KIẾN THỨC ĐẠT ĐƯỢC	95
6.2. KẾT QUẢ MÔ HÌNH HUẤN LUYỆN	96
6.3. KẾT QUẢ HỆ THỐNG.....	96
6.3.1. Môi trường phát triển	96
6.3.2. Môi trường triển khai	97
6.3.3. Chức năng đã cài đặt.....	97

6.4.	KẾT QUẢ ỨNG DỤNG.....	97
6.4.1.	Môi trường phát triển	97
6.4.2.	Môi trường triển khai	97
6.4.3.	Chức năng đã cài đặt.....	97
6.5.	SO SÁNH CÁC KẾT QUẢ THU ĐƯỢC VỚI MỤC TIÊU BAN ĐẦU	98
6.6.	ĐỊNH HƯỚNG PHÁT TRIỂN VÀ NGHIÊN CỨU TRONG TƯƠNG	LAI 99
6.7.	LỜI KẾT	100
	TÀI LIỆU THAM KHẢO	102

DANH SÁCH HÌNH

Hình 3.1. Hai bước của trình tạo chuỗi lặp lại dựa trên sự chú ý	28
Hình 4.1 Kiến trúc tổng quan hệ thống	32
Hình 4.2. Minh họa kiến trúc mô hình	33
Hình 4.3 Tín hiệu thô trong miền thời gian của câu nói “anh có thể gọi tôi không”	36
Hình 4.4 Minh họa ma trận số thực phổ của câu nói “anh có thể gọi tôi không” ...	37
Hình 4.5. Hình ảnh phổ của câu nói “anh có thể gọi tôi không”	38
Hình 4.6. Ma trận sau khi tiền xử lí âm thanh của câu nói “anh có thể gọi tôi không”	38
Hình 4.7 Minh họa kiến trúc hệ thống DeepSpeech 2	40
Hình 4.8 Cấu trúc của mô hình với 2 lớp tích chập thể hiện ở mã nguồn	43
Hình 4.9 Cấu trúc của mô hình với 5 lớp hồi quy thể hiện ở mã nguồn.....	43
Hình 4.10 Cấu trúc mô hình với 1 lớp kết nối đầy đủ được thể hiện ở mã nguồn...	44
Hình 4.11 Ma trận thể hiện phân phối xác suất từng kí tự.....	48
Hình 4.12 Kết quả ma trận với bộ giải mã tham lam của câu nói minh họa.....	48
Hình 4.13 Văn bản đầu ra “anh có thể gọi tôi không” với bộ giải mã tham lam.	49
Hình 4.14 Ma trận phân phối xác suất với bộ giải mã tìm kiếm chùm.	50
Hình 4.15 Minh họa một số kết quả với bộ giải mã tìm kiếm chùm.	51
Hình 4.16. Minh họa một phần văn bản phục vụ cho mô hình ngôn ngữ.....	55
Hình 4.17 Kiến trúc chung xây dựng máy chủ	56
Hình 4.18 Bản mẫu xây dựng ứng dụng minh họa	57
Hình 5.1 Minh họa một số tập tin văn bản phục vụ xây dựng mô hình ngôn ngữ...	61

Hình 5.2 Minh hoạ cấu trúc thư mục chứa bộ dữ liệu âm thanh.....	62
Hình 5.3 Minh hoạ cấu trúc thư mục chứa bộ dữ liệu văn bản dịch.	63
Hình 5.4 Minh hoạ nội dung một tập văn bản dịch.....	63
Hình 5.5 Minh hoạ tập tin csv chứa dữ liệu huấn luyện.	64
Hình 5.6 Đoạn mã nguồn gây lỗi khi tập tin đầu vào là đoạn âm thanh rỗng.....	65
Hình 5.7 Minh hoạ bảng kí tự trong ngôn ngữ Tiếng Việt.	66
Hình 5.8 Đoạn mã nguồn chỉnh sửa khi đọc tập tin âm thanh đầu vào.	66
Hình 5.9 Minh hoạ tập tin mô hình ngôn ngữ ở hai định dạng.	68
Hình 5.10 Kết quả đầu ra khi chưa tích hợp mô hình dấu câu.....	68
Hình 5.11 Kết quả đầu ra khi đã tích hợp mô hình dấu câu.....	68
Hình 5.12 Kết quả nhận dạng câu nói có chứa từ Tiếng Anh trước khi cải tiến.....	69
Hình 5.13 Kết quả nhận dạng từ Tiếng Anh ssau khi cải tiến mô hình ngôn ngữ ...	69
Hình 5.14 Minh hoạ một số từ trong bộ từ điển Tiếng Việt nhóm tự xây dựng	86
Hình 5.15 Lỗi hệ thống khi huấn luyện mô hình (1).....	88
Hình 5.16 Lỗi hệ thống khi huấn luyện mô hình (2).....	88
Hình 5.17 Lỗi nội dung tập tin csv.....	89
Hình 5.18 Xây dựng máy chủ với terminal.....	90
Hình 5.19 Minh hoạ kết quả khởi chạy, thông tin truy cập ngrok	91
Hình 5.20 Giao diện ứng dụng mẫu với chức năng ghi âm.	91
Hình 5.21 Cho phép truy cập bảo mật khi sử dụng ứng mẫu trên trình duyệt.....	92
Hình 5.22 Minh hoạ kết quả đoạn ghi âm trên ứng dụng mẫu.....	92
Hình 5.23 Giao diện ứng dụng mẫu với chức năng tải lên tập tin âm thanh.	93

DANH SÁCH BẢNG

Bảng 5.1 Điều chỉnh tham số epoch = 75	71
Bảng 5.2 Điều chỉnh tham số epoch = 70	71
Bảng 5.3 Điều chỉnh tham số epoch = 50	72
Bảng 5.4 Điều chỉnh tham số $\alpha=2$, $\beta=-0.2$, beam_width =1024	73
Bảng 5.5 Điều chỉnh tham số $\alpha=2$, $\beta=1$, beam_width =1024	73
Bảng 5.6 Điều chỉnh tham số $\alpha=2$, $\beta=-0.2$, beam_width = 2048	74
Bảng 5.7 Điều chỉnh tham số batch_size = 16	75
Bảng 5.8 Điều chỉnh tham số batch_size = 32	75
Bảng 5.9 Điều chỉnh tham số batch_size = 25	76
Bảng 5.10 Điều chỉnh tham số rnn_type = lstm	77
Bảng 5.11 Điều chỉnh tham số rnn_type = gru	77
Bảng 5.12 Điều chỉnh tham số rnn_type = rnn	78
Bảng 5.15 Điều chỉnh tham số hidden size = 512	80
Bảng 5.16 Điều chỉnh tham số hidden size = 1024	81
Bảng 5.17 Điều chỉnh tham số hidden size = 1600	81
Bảng 5.18 So sánh kết quả tỉ lệ lỗi từ trên tập dữ liệu FPT, VIVOS của mô hình dựa trên sự chú ý và mô hình Deepspeech 0.4.	82
Bảng 5.19 Kết quả tỉ lệ lỗi từ các mô hình nhận dạng giọng nói Tiếng.	83
Bảng 6.1 Kết quả huấn luyện mô hình về tỉ lệ lỗi từ có và không có sử dụng mô hình ngôn ngữ.	96
Bảng 6.2 Bảng so sánh kết quả thực hiện đề tài với mục tiêu ban đầu	99

CHƯƠNG 1. GIỚI THIỆU LUẬN VĂN

1.1. GIỚI THIỆU ĐỀ TÀI

Cuộc sống ngày càng hiện đại, nhu cầu áp dụng những công nghệ tiên tiến để phục vụ các công việc trong cuộc sống của mỗi người càng cao. Hơn một thập kỷ gần đây, với sự phát triển mạnh mẽ của công nghệ thông tin, công nghệ xử lý ngôn ngữ tự nhiên như mã hóa, nhận dạng tiếng nói, giả lập giọng nói... đã trở thành xu hướng nghiên cứu mới được nhiều nhà khoa học quan tâm ở các lĩnh vực khác nhau như tin học, toán học, điều khiển, điện tử,...

Với xu hướng mới đó, vấn đề giao tiếp giữa con người và máy tính đang đòi hỏi những sự cải tiến vượt bậc. Công nghệ đã làm tăng lên tốc độ xử lý dữ liệu của các thiết bị hiện đại trong cuộc sống hàng ngày của chúng ta. Cùng với sự tăng lên của tốc độ xử lý dữ liệu, thời gian mà máy móc đợi con người nhập liệu cũng tăng lên. Nhìn chung, hầu hết các phương pháp nhập liệu truyền thống, có thể kể đến như nhập liệu bằng tay, nhập liệu thông qua các thiết bị điều khiển, ... có tốc độ và thời gian thao tác nhập dữ liệu tương đối chậm. Để giải quyết vấn đề này, giọng nói là giải pháp hợp lý nhất tính đến thời điểm hiện tại. Với sự trợ giúp của công nghệ, người dùng có thể dễ dàng điều khiển thiết bị và nhập tài liệu bằng giọng nói. Đó là chưa kể đến lợi ích to lớn mà những phần mềm dựa trên nhận dạng giọng nói có thể mang lại cho những người khiếm thị. Họ có thể tận hưởng những tiến bộ công nghệ tương tự như những gì mà một người bình thường có thể làm, không còn khoảng cách xuất hiện do những khiếm khuyết về giác quan. Nhận dạng giọng nói cho phép tạo tài liệu nhanh hơn vì phần mềm nói chung tạo ra các từ thông qua giọng nói thường nhanh hơn nhiều so với một người có thể nhập. Có thể nói, giọng nói đang ngày càng trở nên "quyền lực" hơn khi có thể điều khiển mọi thiết bị công nghệ hiện đại.

Và thực tế, nhận dạng giọng nói tự động đang dần bùng nổ, đem lại nhiều lợi ích và thuận lợi hơn cho con người. Nhận thấy được tiềm năng phát triển to lớn như vậy, hàng loạt các dịch vụ nhận dạng âm thanh tiếng nói cũng như là ứng dụng sử

dụng dịch vụ này đã ra đời. Từ các tổ chức đến cá nhân, công nghệ này được sử dụng rộng rãi vì những lợi thế khác nhau mà nó mang lại.

1.2. LÝ DO LỰA CHỌN ĐỀ TÀI

Trong những năm gần đây, các nhà nghiên cứu đang tập trung vào công nghệ nhận dạng giọng nói và đã có một số thành công đối với việc nhận dạng tiếng Anh và một số ngôn ngữ khác. Triển khai, xây dựng mô hình và ứng dụng các mô hình này vào thực tế ứng dụng vấn đề này là một việc làm hết sức có ý nghĩa đặc biệt trong giai đoạn công nghiệp hoá hiện đại hoá hiện nay.

Đối với Tiếng Việt, Tiếng Việt là một trong ngôn ngữ có nhiều thanh điệu nhất thế giới, cùng với những đặc trưng riêng về vùng miền, mặc dù đã có một số công trình nghiên cứu về lĩnh vực nhận dạng giọng nói Tiếng Việt nhưng lĩnh vực này nói chung vẫn còn khá mới ở nước ta.

Áp dụng các kiến thức đã học của bản thân trong quá trình học tập cũng như quá trình tìm hiểu, nghiên cứu vào việc xây dựng mô hình có ý nghĩa lớn, có tiềm năng cao trong tương lai, nhóm sinh viên lựa chọn đề tài “Xây dựng mô hình nhận dạng âm thanh Tiếng Việt”. Mục tiêu cơ bản của đề tài này, nhóm sinh viên muốn nghiên cứu, thu thập dữ liệu và xây dựng mô hình nhận dạng âm thanh Tiếng Việt hiệu quả, đạt độ chính xác cao so với các ứng dụng hiện có trên thị trường.

Ngoài ra, việc chọn đề tài này giúp nhóm sinh viên tiếp cận với lĩnh vực học máy, nghiên cứu tìm hiểu thông tin từ các nguồn tài liệu quý giá. Hơn thế nữa, sau khi thực hiện đề tài, nhóm sinh viên sẽ có thêm kinh nghiệm và hiểu biết về quy trình làm ra một dự án thực tế, không những thế nhóm sinh viên còn được học hỏi các kiến thức chuyên môn liên quan đến học sâu, huấn luyện mô hình, và áp dụng vào xây dựng ứng dụng minh họa. Chính những điều đó sẽ là nền tảng quý báu hỗ trợ đắc lực cho nhóm sinh viên trên con đường học vấn và việc làm trong tương lai.

1.3. HƯỚNG PHÁT TRIỂN CỦA LUẬN VĂN

Tiếng Việt được coi là một ngôn ngữ khó học với người nước ngoài bởi ngữ pháp, thanh điệu và đặc trưng vùng miền. Máy tính cũng giống như người nước ngoài - để nó nghe hiểu và diễn giải được giọng nói tiếng Việt thành dạng văn bản không phải là việc dễ dàng. Nhận dạng tiếng nói đóng vai trò quan trọng trong giao tiếp giữa người và máy. Nó giúp máy móc hiểu và thực hiện các hiệu lệnh của con người. Hiện nay trên thế giới, lĩnh vực nhận dạng tiếng nói đã đạt được nhiều tiến bộ vượt bậc. Đối với ngôn ngữ tiếng Anh, việc nhận dạng có thể đạt độ chính xác tới 99%. Ở Việt Nam, lĩnh vực nhận dạng giọng nói còn khá mới và hiện độ chính xác nhìn chung chưa cao. Luận văn này, nhóm sinh viên sẽ xây dựng, huấn luyện mô hình nhận dạng âm thanh Tiếng Việt và xây dựng ứng dụng demo với mục tiêu cơ bản là:

- ❖ Mô hình được xây dựng đạt độ chính xác tối thiểu 75%.
- ❖ Ứng dụng mẫu sẽ được phát triển trên nền tảng web với chức năng chính nhận một tập tin âm thanh Tiếng Việt và chuyển sang văn bản Tiếng Việt, bên cạnh đó cho phép ghi âm trực tiếp âm thanh và tải lên tập tin âm thanh, văn bản sau khi chuyển đổi hỗ trợ tải về.

1.4. MỤC TIÊU CỦA LUẬN VĂN

Để hoàn thành tốt đề tài luận văn, bản luận văn và sản phẩm cuối cùng của nhóm sinh viên sẽ đảm bảo các mục tiêu sau đây:

- ❖ Bản luận văn trình bày lý thuyết nền tảng và giải pháp để xử lý việc nhận 1 tập tin âm thanh tiếng Việt và xuất ra nội dung văn bản ở dạng tiếng Việt.
- ❖ Xây dựng, thu thập dữ liệu, và đào tạo mô hình để nhận 1 file âm thanh tiếng Việt và xuất ra nội dung ở dạng văn bản.
- ❖ Cải tiến độ chính xác của mô hình với mục tiêu là 75%.
- ❖ Xây dựng ứng dụng web chuyển đổi từ giọng nói sang văn bản Tiếng Việt. Ứng dụng mẫu áp dụng mô hình được xây dựng là ứng dụng web dựa trên nền tảng Flask. Ứng dụng cho phép định dạng, chỉnh sửa văn bản trực tuyến (căn lề, kích thước chữ, phông chữ, định dạng chữ, ...), bên cạnh đó ứng dụng cho

phép người dùng tải lên tập tin âm thanh ít nhất 3 định dạng (.mp3, .webm, .wav). và cho phép người dùng tải xuống tập tin văn bản (.txt). Ứng dụng cũng hỗ trợ ghi âm, nghe lại đoạn âm thanh đã ghi âm trước khi thực hiện chuyển đổi.

1.5. PHẠM VI ĐỀ TÀI

Sản phẩm của đề tài “Xây dựng mô hình nhận dạng âm thanh tiếng Việt” là (1) một mô hình cho phép nhận vào một tập tin âm thanh Tiếng Việt và trả về một đoạn văn bản Tiếng Việt, (2) xây dựng một hệ thống hướng người dùng. Nhóm sinh viên sẽ tập trung vào việc phát triển một cách đầy đủ mô hình này dựa trên engine đã có sẵn trên thị trường đồng thời cải tiến mô hình và đưa ra kết quả tốt nhất. Khả năng nhận dạng ngôn ngữ Tiếng Việt của mô hình được đào tạo từ bộ dữ liệu âm thanh Tiếng Việt do nhóm tìm hiểu, thu thập từ các nguồn dữ liệu mở. Sản phẩm luận văn được thực hiện một cách toàn diện nên sẽ được áp dụng, mở rộng vào các dịch vụ web (API). Nhóm sẽ xây dựng một API sử dụng mô hình trên và một trang web mẫu sử dụng API này để biểu diễn mô hình.

Sau đây nhóm sinh viên sẽ trình bày tổng quan về cấu trúc sẽ được trình bày, ngoài phần mở đầu và kết luận, khoá luận được tổ chức thành 6 chương như sau:

- ❖ **Chương 1: Giới thiệu luận văn** giới thiệu khái quát về đề tài, trình bày một số lý do, hướng phát triển, và mục tiêu mà khóa luận này hướng tới.
- ❖ **Chương 2: Vấn đề thực tế và động lực phát triển** trình bày những ứng dụng thực tế mà kết quả của đề tài mang lại.
- ❖ **Chương 3: Các công trình nghiên cứu sẵn có** liên quan giới thiệu chi tiết về những nghiên cứu, mô hình, kết quả đã có liên quan đến nhận dạng âm thanh nói chung và nhận dạng với Tiếng Việt nói riêng, đồng thời cũng giới thiệu sơ về mô hình của nhóm.
- ❖ **Chương 4: Phương pháp nghiên cứu của đề tài** phân tích mô hình mà nhóm áp dụng, và trình bày những cải tiến áp dụng vào mô hình này trong nhận dạng âm thanh Tiếng Việt.

- ❖ **Chương 5: Thử nghiệm và đánh giá** trình bày quá trình thử nghiệm của khoá luận và đưa ra một số đánh giá, nhận xét các kết quả mà mô hình của nhóm đạt được so với các nghiên cứu trước đó.
- ❖ **Chương 6: Tổng kết và đánh giá** đưa ra kết luận về khoá luận và những hướng nghiên cứu sắp tới.

CHƯƠNG 2. VẤN ĐỀ THỰC TẾ VÀ ĐỘNG LỰC PHÁT TRIỂN

Bài toán nhận dạng giọng nói nói chung và xây dựng mô hình nhận dạng âm thanh Tiếng Việt nói riêng đã được phát hiện từ rất lâu và đang thu hút nhiều sự quan tâm. Sau đây nhóm sinh viên sẽ nêu qua một số lĩnh vực mà công nghệ này mang lại.

2.1. PHỤC VỤ NHU CẦU TRONG CUỘC SỐNG HÀNG NGÀY

Trong cuộc sống hằng ngày, việc sử dụng công nghệ nhận dạng giọng nói giúp giảm tiết kiệm nhiều thời gian, sức lực cho con người. Có thể kể đến như : điều khiển các thiết bị trong nhà thông qua giọng nói; tìm kiếm thông tin, tìm kiếm kênh trên tivi, máy tính,... mà không cần nhập tay; nhắn tin với bạn bè khi đang bận một công việc khác, ghi chú lại các thông tin trong cuộc hội thảo, cuộc nói chuyện mà không cần tốn thời gian bằng việc ghi lại bằng tay,... Các trợ lý ảo trên điện thoại hoặc các mạch nhận dạng giọng nói trên các thiết bị đang thực hiện tốt các nhiệm vụ trên giúp con người có trải nghiệm tốt nhất trong cuộc sống thường nhật.

2.2. ÁP DỤNG VÀO NHIỀU LĨNH VỰC TRONG CUỘC SỐNG

Công nghệ nhận dạng giọng nói và việc sử dụng trợ lý ảo đã nhanh chóng được áp dụng vào nhiều lĩnh vực khác nhau trong cuộc sống, chẳng hạn như văn phòng, kinh doanh, y tế, vạn vật kết nối, giáo dục, ...

❖ Trong lĩnh vực văn phòng

Công nghệ nhận dạng giọng nói ở lĩnh vực văn phòng đã phát triển mạnh, có vai trò kết hợp các nhiệm vụ đơn giản để tăng hiệu quả, giảm các công việc truyền thống do con người thực hiện để tăng năng suất.

Các công việc mà trợ lý ảo có thể thực hiện được như: tìm kiếm tài liệu, văn bản, báo cáo,... trên thiết bị; nhận dạng thông tin muốn đưa vào máy tính; in, xuất tài liệu; thực hiện các công việc trong cuộc hội thảo; tạo các biểu đồ, bảng báo cáo bằng dữ liệu, ... Tất cả các yêu cầu trên đều có thể giao tiếp với trợ lý ảo bằng giọng nói.

❖ Trong lĩnh vực kinh doanh

Nhận dạng giọng nói giúp tăng cường nhu cầu tìm kiếm bằng giọng nói, từ đó mở ra một khía cạnh mới mà nhà tiếp thị có thể tiếp cận người tiêu dùng. Dữ liệu giọng nói của người dùng có thể giải thích độ tuổi, tầng lớp và các thông tin liên quan đến sinh trắc nhân khẩu như văn hóa của họ. Các nhà tiếp thị có thể tối ưu hóa những thông tin này để đón đầu các xu hướng tiêu dùng.

❖ Trong lĩnh vực y tế

Đặc điểm của công nghệ nhận dạng giọng nói là giảm thiểu thời gian giao tiếp với thiết bị công nghệ. Do đó, việc áp dụng công nghệ này vào các bệnh viện sẽ tiết kiệm thời gian quý giá bởi việc truy cập thông tin rảnh tay, mang lại ích lợi vô cùng tích cực đến sự an toàn của bệnh nhân, nâng cao hiệu quả y tế. Một số thuận lợi cụ thể của nhận dạng giọng nói trong lĩnh vực y tế như: tìm các hồ sơ y tế nhanh chóng; truy vấn thông tin về quản lý như số giường trống, số bệnh nhân hiện có, ... cải thiện quy trình làm việc; rút gọn thời gian nhập liệu; trong tương lai có thể có sự tương tác giữa bệnh nhân và trợ lý ảo tại nhà để tiết kiệm thời gian đợi ở bệnh viện,...

❖ Trong lĩnh vực vạn vật kết nối

Các trợ lý ảo trên điện thoại thông minh cho phép người sử dụng tương tác, điều khiển các thiết bị trong mạng lưới vạn vật kết nối (Internet of Things - IoT), đặc biệt là công nghệ nhà thông minh (Smart Home). Trong hiện tương lai, công nghệ nhận dạng có thể giúp người dùng trực tiếp ‘ra lệnh’ cho các thiết bị này mà không cần thiết phải chạm trực tiếp vào điện thoại. Điều này tạo cảm giác thoải mái, tiện lợi cho người dùng.

2.3. TIỀM NĂNG KINH TẾ CAO

Nhận dạng giọng nói cho phép chuyển đổi giọng nói thành văn bản, giúp việc tạo và sử dụng thông tin trở nên dễ dàng hơn bởi vì văn bản dễ dàng hơn để lưu trữ, xử lý và sử dụng, cho cả máy tính và con người. Các ứng dụng được xây dựng để phục vụ nhu cầu nhận dạng và tích hợp các dịch vụ nhận dạng đang ngày càng phát triển và là lĩnh vực hấp dẫn đối với thị trường công nghệ. Sẽ rất thuận tiện nếu chúng

ta có thể sử dụng phần mềm dựa trên nhận dạng giọng nói để mua hàng, kiểm tra thời tiết, gửi email, tìm kiếm thông tin trên internet theo một cách mới để tương tác với máy móc.

Các ứng dụng chụp ảnh giải trí trên các thiết bị di động không những mang lại giá trị cho bản thân người dùng mà còn đem đến những giá trị về kinh tế cho phía nhà phát triển. Bắt kịp xu thế và nhu cầu của người dùng, các doanh nghiệp, cùng với những cải tiến ấn tượng về khả năng hiểu ngôn ngữ tự nhiên và tỷ lệ chính xác của nhận dạng giọng nói, đã khiến các công ty ngày càng muốn tiếp tục xây dựng tính năng hỗ trợ giọng nói trải nghiệm thậm chí vượt ra ngoài phạm vi riêng tư, các công ty, doanh nghiệp chủ động hơn trong cách tiếp cận với nhận dạng giọng nói để giới thiệu hoặc kết hợp công nghệ nhận dạng giọng nói vào các sản phẩm của họ.

Hơn thế nữa, lĩnh vực nhận dạng giọng nói và tổng quan hơn là trí tuệ nhân tạo sẽ ngày càng trở nên phức tạp hơn trong tương lai. Ngành công nghiệp này đã và đang thu hút với hàng trăm công ty trên thế giới thử nghiệm và tích hợp sản phẩm và dịch vụ của họ với công nghệ nhận dạng giọng nói. Theo lời của Brian Roemmele, người sáng lập và Tổng biên tập của tạp chí Multiplex “60 năm qua, con người đã thích nghi với máy tính. 60 năm tới, máy tính sẽ thích ứng với chúng ta. Tiếng nói của chúng ta sẽ là nền tảng cho điều đó, nó sẽ là một cuộc cách mạng và nó sẽ thay đổi mọi thứ”.

Trên cơ sở đó, việc xây dựng mô hình nhận dạng giọng nói cũng như các ứng dụng các mô hình này vào các phần mềm hiện có trong nhiều lĩnh vực của cuộc sống đem lại một tiềm năng kinh tế rất cao.

2.4. PHỤC VỤ NHU CẦU SỬ DỤNG CÔNG NGHỆ CHO ĐA DẠNG NGƯỜI TRONG XÃ HỘI

Công nghệ ngày càng hiện đại, một số tầng lớp của xã hội gần như bị ‘lạc hậu’ đi vì khó khăn trong việc sử dụng các thiết bị điện tử thông minh. Người lớn tuổi, người cận thị, người tiếp cận với thiết bị điện tử hiện đại muộn khó khăn trong việc nhập thông tin từ bàn phím, màn hình điện thoại di động với phím nhỏ; người khiếm

thì khó khăn trong việc sử dụng các thiết bị thông minh,... Công nghệ nhận dạng giọng nói giúp giải quyết các khó khăn trên.

2.5. PHỤC VỤ TRONG CÁC CUỘC HỌP, CUỘC THẢO LUẬN

Việc áp dụng nhận dạng giọng nói trong các cuộc họp cho phép các đại biểu cuộc họp có thể nắm bắt nội dung một cách tức thời bằng văn bản và có thể lưu lại toàn bộ nội dung của phiên họp, nâng cao chất lượng của cuộc họp. Đồng thời, nhờ việc có thể lưu lại toàn bộ nội dung phiên họp nhanh chóng, chúng ta có thể bàn giao cho các cuộc họp khác tiết kiệm thời gian hơn.

Từ những lợi ích mà công nghệ nhận dạng giọng nói có thể mang lại, nhóm mong muốn xây dựng một mô hình nhận dạng có độ chính xác cao để phục vụ cho xã hội. Trong chương 3, nhóm trình bày một số mô hình hiện đại và cách tiếp cận trong nhận dạng giọng nói.

CHƯƠNG 3. CÁC CÔNG TRÌNH NGHIÊN CỨU SẴN CÓ

3.1. CÁC KẾT QUẢ NGHIÊN CỨU SẴN CÓ

Qua quá trình tìm kiếm, sưu tập, nhóm có được một số mã nguồn, kết quả mô hình, có thể phục vụ cho nghiên cứu đề tài:

- ❖ Mô hình nhận dạng giọng nói từ đầu đến cuối dựa mạng hồi quy kết hợp với phân loại theo thời gian kết nối (CTC) với kết quả là mã nguồn được xây dựng cho mô hình nhận dạng giọng nói trên ngôn ngữ Tiếng Anh từ cộng đồng mã nguồn mở của tác giả Sean Naren dựa trên ý tưởng của bài báo DeepSpeech 2, một nghiên cứu của Baidu được công bố vào ngày 08/12/2015 tại Silicon Valley AI Lab.
- ❖ Mô hình nhận dạng tiếng nói từ đầu đến cuối (end-to-end) của nhóm sinh viên thực hiện Luận văn với đề tài “Xây dựng hệ thống cung cấp dịch vụ nhận dạng âm thanh Tiếng Việt” năm 2019 dựa trên DeepSpeech 0.4 với kết quả đạt tỉ lệ lỗi từ 25.84 %.
- ❖ Mô hình nhận dạng giọng nói dựa trên sự chú ý với mã nguồn mở được xây dựng cho ngôn ngữ Tiếng Anh, của tác giả Alexander-H-Liu dựa trên ý tưởng bài báo Attention-Based Models for Speech Recognition được công bố vào tháng 6 năm 2015.

Bộ dữ liệu công khai sẵn có nhóm kế thừa được để nhóm sử dụng huấn luyện mô hình:

- ❖ Bộ dữ liệu FPT: với khoảng 5GB dữ liệu tương đương khoảng 35 giờ dữ liệu từ dự án mở của FPT.
- ❖ VIVOS Corpus: khoảng 15 giờ dữ liệu từ phòng thí nghiệm AILab của Trường Đại học Khoa Học Tự Nhiên.

3.2. CÁC MÔ HÌNH, PHƯƠNG PHÁP, THUẬT TOÁN

Nhận dạng giọng nói là một lĩnh vực nghiên cứu thu hút từ rất lâu, các mô hình, phương pháp, thuật toán cũng được ra đời để phục vụ cho nhiệm vụ này.

❖ Mô hình Markov ẩn

Mô hình này là mô hình thống kê xuất ra một chuỗi các ký hiệu hoặc đại lượng, được sử dụng trong nhận dạng giọng nói vì tín hiệu giọng nói có thể được xem như một tín hiệu tĩnh hoặc một tín hiệu tĩnh trong thời gian ngắn. Ưu điểm của mô hình này là có thể đào tạo tự động và sử dụng đơn giản, khả thi về mặt tính toán.

❖ Nhận dạng giọng nói dựa trên độ cong thời gian động

Đây là một thuật toán để đo độ giống nhau giữa hai chuỗi có thể khác nhau về thời gian hoặc tốc độ. Hiện nay thuật toán này phần lớn đã bị thay thế bởi cách tiếp cận dựa trên mô hình markov ẩn thành công hơn.

❖ Mạng thần kinh

Mạng thần kinh tạo ra ít giả định hơn về các thuộc tính thống kê tính năng so với mô hình markov ẩn và có một số phẩm chất khiến chúng trở thành mô hình nhận dạng hấp dẫn để nhận dạng giọng nói. Khi được sử dụng để ước tính xác suất của một phân đoạn tính năng giọng nói, mạng nơ-ron cho phép đào tạo phân biệt một cách tự nhiên và hiệu quả. Thành công của mô hình này là đạt hiệu quả cao trong việc phân loại các đơn vị thời gian ngắn như âm vị riêng lẻ và các từ biệt lập. Song, các mạng thần kinh ban đầu hiếm khi thành công đối với các nhiệm vụ nhận dạng liên tục vì khả năng mô hình hóa các phụ thuộc thời gian còn hạn chế.

Các mạng thần kinh truyền tải trực tiếp và lặp lại sâu (Deep Learning) đây là một mạng thần kinh nhân tạo với nhiều lớp đơn vị ẩn giữa các lớp đầu vào và đầu ra. Một trong những hiệu quả vượt trội của mạng này là mang lại khả năng học tập lớn, do đó có tiềm năng mô hình hóa các mẫu dữ liệu giọng nói phức tạp.

❖ Nhận dạng giọng nói tự động đầu cuối

Năm 2014 được xem như là sự bùng nổ của hệ thống đầu cuối, các mô hình nhận dạng đầu cuối sẽ tổng hợp tất cả các thành phần của trình nhận dạng giọng nói truyền thống. Điều này đem lại hiệu quả rất cao vì nó đơn giản hóa quy trình đào tạo và quy trình triển khai.

Phần tiếp theo, nhóm sinh viên sẽ trình bày chi tiết hơn về hệ thống nhận dạng tiếng nói từ đầu đến cuối và hai hướng tiếp cận chính trong hệ thống này.

3.3. NHẬN DẠNG GIỌNG NÓI TỰ ĐỘNG TỪ ĐẦU ĐẾN CUỐI (END – TO – END)

Các mô hình nhận dạng tiếng nói hiện đại thường được chia thành hai thành phần chính: mô hình âm thanh và mô hình ngôn ngữ. Tùy theo bài toán tiếp cận mà giải pháp mỗi mô hình nhận dạng tiếng nói sẽ được tùy chỉnh để cài đặt phù hợp. Mạng nơ-ron là mô hình học mạnh mẽ, đã đạt được những kết quả tiên tiến trong nhiều nhiệm vụ học máy có giám sát và không giám sát. Những tiến bộ gần đây về thuật toán và phần cứng máy tính đã giúp chúng ta huấn luyện mạng thần kinh đầu cuối (end-to-end) hỗ trợ cho các nhiệm vụ mà trước đây đòi hỏi trình độ chuyên môn đáng kể từ con người. So với cách tiếp cận truyền thống, mạng thần kinh đầu cuối đòi hỏi ít sự nỗ lực hơn từ con người và mang lại kết quả cao hơn. Nó cho phép dịch trực tiếp dữ liệu âm thanh ra chuỗi chữ mà không cần đại diện ngữ âm trung gian. Những kết quả khả quan này có thể đạt được là nhờ sự phong phú đa dạng của tập dữ liệu huấn luyện.

Công việc gần đây về nhận dạng giọng nói đầu cuối có thể được phân loại thành hai cách tiếp cận chính: (1) Phân loại theo thời gian kết nối (CTC) và (2) bộ mã hóa-giải mã dựa trên sự chú ý. Cả hai phương pháp đều giải quyết vấn đề nhận dạng giọng nói và đạt được những thành tựu đáng kể.

Dưới đây, nhóm sinh viên sẽ trình bày chi tiết về hai giải pháp hệ thống nhận dạng tiếng nói hiện đại: (1) Mô hình bao gồm các mạng thần kinh theo hướng đầu cuối (end- to- end) dựa trên kỹ thuật phân loại thời gian kết nối và (2) mô hình bao gồm các mạng thần kinh tuần hoàn dựa trên sự chú ý. Đồng thời đưa ra lý giải tại sao hệ thống nhận dạng tiếng nói hiện đại dựa trên phân loại thời gian kết nối được nhóm sinh viên lựa chọn để huấn luyện.

3.3.1. Mô hình nhận dạng tiếng nói đầu cuối (end -to-end) dựa trên kỹ thuật phân loại thời gian kết nối (CTC)

3.3.1.1. Giới thiệu

Mô hình mạng thần kinh được huấn luyện với chức năng mất phân loại theo thời gian kết nối (CTC) để dự đoán phiên âm giọng nói từ âm thanh được giới thiệu bởi Alex Graves của Google DeepMind và Navdeep Jaitly của Đại học Toronto vào năm 2014. Mô hình bao gồm các mạng nơ-ron tuần hoàn và một lớp CTC. Mô hình mạng bao gồm nhiều lớp kết nối lặp lại, bộ tích chập và phi tuyến tính, cũng như tác động của một phiên bản cụ thể của chuẩn hóa hàng loạt (BatchNorm) được áp dụng cho Mạng thần kinh hồi quy (RNN). Trái ngược với các phương pháp tiếp cận đào tạo quy mô lớn trước đây sử dụng máy chủ tham số và cập nhật không đồng bộ, mô hình này sử dụng thuật toán tối ưu Stochastic gradient descent (SGD), dễ gỡ lỗi hơn trong khi thử nghiệm các ý tưởng mới và cũng hội tụ nhanh hơn với cùng mức độ song song dữ liệu [19].

Mô hình mạng thần kinh hồi quy kết hợp với phân loại thời gian kết nối(CTC-RNN) đã được nghiên cứu và hoạt động tốt trong nhận dạng giọng nói đầu cuối với đầu ra ở các thử nghiệm trước đó [22]. Mô hình CTC-RNN cũng đã được chứng minh là hoạt động tốt trong việc dự đoán âm vị, mặc dù vẫn cần một từ vựng trong trường hợp này. Song, mô hình này được cho là hiệu quả hơn vì đào tạo các mạng CTC-RNN từ đầu mà không cần căn chỉnh theo chiều khung để đào tạo trước.

3.3.1.2. Kỹ thuật phân loại thời gian kết nối (CTC)

Nhiệm vụ nhận dạng giọng nói sẽ bắt đầu với một tập dữ liệu về các đoạn âm thanh và bản ghi tương ứng. Tuy nhiên, sẽ rất khó khăn để biết cách ánh xạ các ký tự trong bản ghi âm phù hợp với âm thanh. Đây chính là vấn đề lớn làm cho việc đào tạo một công cụ nhận dạng giọng nói khó hơn lúc đầu. Phân loại theo thời gian kết nối (CTC) ra đời như là một cách để tránh việc không biết sự liên kết giữa đầu vào và đầu ra. Và mô hình nhận dạng tiếng nói đầu cuối kết hợp với phân loại theo thời gian kết nối đặc biệt phù hợp với các ứng dụng như nhận dạng giọng nói.

Kiến trúc CTC trong nhận dạng giọng nói sử dụng khung đầu vào có độ dài T , với khung đầu vào là $x = \{x_1, \dots, x_T\}$ và các ký tự đầu ra $y = \{y_1, \dots, y_U\}$ có độ dài U , trong đó $y_u \in \{1, \dots, K\}$ với K là số nhãn phân biệt, hay K là số các ký tự trong bảng chữ cái.

Theo [4], mạng thần kinh thường được huấn luyện dựa trên phân loại mức khung (frame-level) trong nhận dạng giọng nói. Điều này yêu cầu một mục tiêu huấn luyện riêng cho mỗi khung, sự căn chỉnh giữa âm thanh và chuỗi văn bản kết quả. Tuy nhiên việc căn chỉnh chỉ đáng tin cậy khi bộ phân loại được đào tạo, dẫn tới sự phụ thuộc vòng giữa phân đoạn và nhận dạng. Bên cạnh đó, việc căn chỉnh là không liên quan đến phần lớn các nhiệm vụ nhận dạng giọng nói. Phân loại thời gian kết nối là một hàm mục tiêu cho phép một RNN được huấn luyện cho các nhiệm vụ dịch chuỗi mà không yêu cầu bất cứ sự căn chỉnh nào giữa chuỗi đầu vào và chuỗi đích.

Ý tưởng chính của CTC là sử dụng biểu diễn nhãn trung gian $\pi = (\pi_1, \dots, \pi_T)$, cho phép lặp lại các nhãn và sự xuất hiện của nhãn trống ($-$), với mục đích đại diện cho sự ánh xạ đặc biệt khi không có nhãn, tức là, $\pi_t \in \{1, \dots, K\} \cup \{-\}$. CTC huấn luyện mô hình để $P(y | x)$ đạt giá trị lớn nhất, với $P(y|x)$ là phân phối xác suất trên tất cả các chuỗi nhãn có thể có $\Phi(y')$:

$$P(y|x) = \sum_{\pi \in \Phi(y')} P(\pi|x)$$

trong đó:

- y' là một chuỗi đã sửa đổi của y , được thực hiện bằng cách chèn các ký hiệu trống ($-$) giữa các nhãn để cho phép khoảng trống trong đầu ra, tức là $y_u \in \{1, \dots, K\} \cup \{-\}$.
- π là biểu diễn nhãn trung gian cho phép lặp lại các nhãn và sự xuất hiện của nhãn trống ($-$).
- T là độ dài của khung âm thanh đầu vào.

CTC được áp dụng trên mạng thần kinh hồi quy (RNN), được hiểu là phân phối nhân bao gồm cả khoảng trống. Xác suất của chuỗi nhãn $P(\pi | x)$ được tính gần đúng bằng tích xác suất của mỗi nhãn dựa trên giả thiết độc lập có điều kiện:

$$P(\pi|x) \approx \prod_{t=1}^T P(\pi_t|x) = \prod_{t=1}^T q_t(\pi_t)$$

trong đó:

- $q_t(\pi_t)$ biểu thị kích hoạt softmax của nhãn π_t trong lớp đầu ra RNN q tại thời điểm t .

Sự mất mát CTC (CTC loss) được định nghĩa là khả năng xảy ra trong log âm của chuỗi ký tự chân trị cơ bản y^* :

$$\mathcal{L}_{CTC} \triangleq -\ln P(y^*|x)$$

Vì CTC không mô hình hóa rõ ràng các phụ thuộc liên nhãn dựa trên giả định độc lập có điều kiện nên mô hình được giới hạn trong thông tin ngôn ngữ ở cấp ký tự. Do đó, các mô hình từ vựng hoặc mô hình ngôn ngữ thường được tích hợp thêm, giống như khung kết hợp. Chi tiết về kiến trúc mô hình này sẽ được trình bày ở chương 4, cũng là mô hình mà nhóm sinh viên lựa chọn cho đề tài này.

3.3.2. Mô hình nhận dạng tiếng nói với bộ mã hóa và bộ giải mã dựa trên sự chú ý

3.3.2.1. Đặt vấn đề

Phần trước đó vừa trình bày về phần hệ thống nhận dạng tiếng nói đầu cuối với kỹ thuật phân loại thời gian kết nối mà nhóm sinh viên sẽ sử dụng trong luận văn này, ở mục này sẽ trình bày tóm tắt thông tin về hệ thống nhận dạng tiếng nói với kỹ thuật dựa trên sự chú ý.

Gần đây, mạng lặp lại dựa trên sự chú ý đã được áp dụng thành công cho nhiều tác vụ khác nhau, chẳng hạn như tổng hợp chữ viết tay, dịch máy, tạo chú thích hình ảnh và phân loại đối tượng trực quan. Các mô hình này hoạt động dựa trên xử lý lặp đi lặp lại đầu vào bằng cách chọn nội dung có liên quan ở mỗi bước. Đồng thời, mô hình dựa trên sự chú ý này cũng được mở rộng vào lĩnh vực nhận dạng giọng nói.

3.3.2.2. Bộ mã hóa, bộ giải mã dựa trên sự chú ý

Không giống như cách tiếp cận CTC, mô hình dựa trên chú ý dự đoán trực tiếp từng kết quả đích mà không yêu cầu đại diện trung gian hoặc bất kỳ giả định nào, cải thiện tỉ lệ lỗi ký tự (CER) so với CTC khi không sử dụng mô hình ngôn ngữ bên ngoài. Mô hình phát ra mỗi phân phối nhân trên các nhãn trước đó theo phương trình đệ quy sau:

$$P(y|x) = \prod_u P(y_u|x, y_{1:u-1})$$

trong đó:

- x là chuỗi đầu vào.
- y_u là ký tự thứ u trong chuỗi đầu ra cần nhận dạng.
- $y_{1:u-1}$ tất cả các ký tự đã được biết trước đó.

Bộ khung bao gồm hai mạng hồi quy thần kinh với một bộ mã hóa và một bộ giải mã, với cấu trúc như vậy giúp cho mô hình có thể được huấn luyện để học hai độ dài khác nhau của chuỗi. Bộ mã hóa biến đổi đầu vào x thành biểu diễn mức cao $h = \{h_1, \dots, h_L\}$:

$$h = \text{Encoder}(x)$$

sau đó bộ giải mã tạo ra phân phối xác suất trên các ký tự y_u dựa vào h và tất cả các ký tự đã thấy trước đó $y_{1:u-1}$:

$$y_u \sim \text{AttentionDecoder}(h, y_{1:u-1})$$

Ở đây, mã thông báo bắt đầu câu < sos > và mã thông báo cuối câu < eos > đặc biệt được thêm vào tập đích để bộ giải mã kết thúc và tạo câu hoàn chỉnh khi bắt gặp kí tự đặc biệt < eos >. Hàm mất mát của mô hình chú ý được tính toán:

$$\mathcal{L}_{Attention} \triangleq -\ln P(y^*|x) = -\sum_U \ln P(y_u^*|x, y_{1:u-1}^*)$$

trong đó

- $y_{1:u-1}^*$ là gốc của các ký tự trước đó.

Cơ chế chú ý hỗ trợ trong thủ tục giải mã bằng cách tích hợp tất cả các đầu vào h thành c_u dựa trên các vector trọng số chú ý của chúng trên đầu vào L xác định vị trí cần tập trung ở bước đầu ra u. Các phương trình sau đại diện cho cách tính a_u và c_u :

$$e_{u,l} = \begin{cases} \text{content - based:} \\ w^T \tanh(Ws_{u-1} + Vh_1 + b) \\ \text{location - based:} \\ f_u = F * \alpha_{u-1} \\ w^T \tanh(Ws_{u-1} + Vh_1 + Uf_{u,1} + b) \end{cases}$$

$$a_{u,l} = \frac{\exp(\gamma e_{u,l})}{\sum_l \exp(\gamma e_{u,l})}$$

$$c_u = \sum_l a_{u,l} h_l$$

trong đó:

- w, W, V, F, U, b là các tham số có thể huấn luyện.
- s_{u-1} là trạng thái bộ giải mã.
- γ là hệ số làm sắc nét.
- $*$ biểu thị tích chập.
- L là số lượng khung đầu vào bị bỏ qua và $L < T$.

Với c_u, s_{u-1} và y_{u-1} , bộ giải mã tạo ra nhãn tiếp theo y_u và cập nhật trạng thái theo công thức sau:

$$y_u \sim \text{Generate}(c_u, s_{u-1})$$

$$s_u = \text{Recurrency}(s_{u-1}, c_u, y_u)$$

trong đó các hàm Generate và Recurrency dùng để biểu thị tương ứng một mạng chuyển tiếp (feed forward network) và một mạng lặp lại.

3.3.2.3. Kiến trúc mô hình

Tại mỗi bước thời gian trong việc tạo chuỗi đầu ra (chuỗi âm vị), cơ chế chú ý sẽ chọn hoặc đo lường các tín hiệu được tạo ra bởi cơ chế trích xuất đặc trưng đã huấn luyện ở tất cả các bước thời gian trong chuỗi đầu vào. Sau đó, vector đặc trưng đã đánh trọng số giúp tạo điều kiện tạo ra phần tử tiếp theo của chuỗi đầu ra.

Trình tạo chuỗi lặp lại dựa trên sự chú ý là một mạng thần kinh lặp lại ngẫu nhiên tạo chuỗi đầu ra $y = \{y_1, \dots, y_T\}$ từ đầu vào x . Trong thực tế, x thường được xử lý bởi một bộ mã hóa để xuất ra biểu diễn đầu vào tuần tự $h = \{h_1, \dots, h_L\}$ phù hợp hơn cho cơ chế chú ý.

Cụ thể hơn, y là một chuỗi các âm vị $y = \{y_1, \dots, y_T\}$ và đầu vào $x = \{x_1, \dots, x_{L'}\}$ là một chuỗi các vector đặc trưng. Mỗi vector đặc trưng được trích xuất từ một cửa sổ nhỏ chồng chéo của khung âm thanh. Bộ mã hóa (encoder) được triển khai dưới dạng mạng lặp lại hai chiều sâu (BiRNN), để tạo thành biểu diễn tuần tự h có độ dài $L = L'$.

Ở bước thứ i , trình tạo chuỗi lặp lại dựa trên sự chú ý tạo ra đầu ra y_i bằng cách tập trung vào các yếu tố liên quan của h :

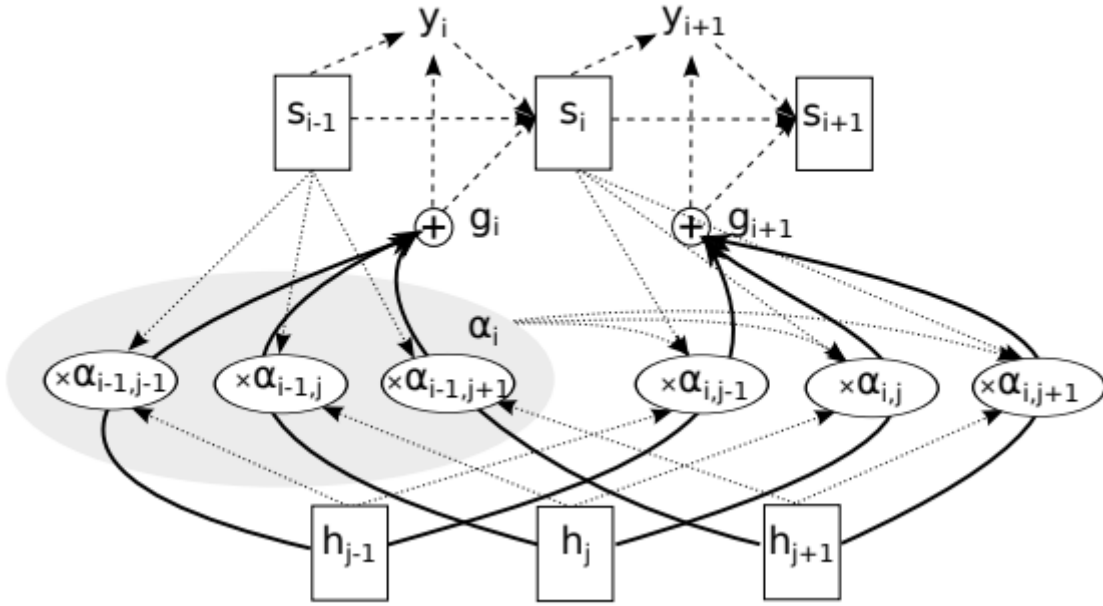
$$\alpha_i = \text{Attend}(s_{i-1}, \alpha_{i-1}, h) \quad (1)$$

$$g_i = \sum_{j=1}^L \alpha_{i,j} h_j \quad (2)$$

$$y_i \sim \text{Generate}(s_{i-1}, g_i) \quad (3)$$

trong đó

- s_{i-1} là trạng thái (i - 1) của mạng thần kinh tuần hoàn mà chúng ta gọi là bộ tạo (generator).
- $\alpha_i \in \mathbb{R}^L$ là vector của trọng số chú ý, cũng thường được gọi là sự liên kết.



Hình 3.1. Hai bước của trình tạo chuỗi lặp lại dựa trên sự chú ý

Hình 3.1 thể hiện cơ chế chú ý kết hợp (tính toán α), dựa trên cả nội dung (h) và thông tin vị trí (α trước đó). Các đường chấm tương ứng với công thức (1), đường liền nét dày cho công thức (2) và các đường đứt nét đối với công thức (3).

Trạng thái của trình tạo mới được tính:

$$s_i = \text{Recurrency}(s_{i-1}, g_i, y_i)$$

Theo [23], cơ chế chú ý được chia thành: cơ chế chú ý dựa trên vị trí, cơ chế chú ý dựa trên nội dung và cơ chế kết hợp. Cơ chế chú ý trong công thức (1) là cơ chế chú ý kết hợp.

❖ Cơ chế chú ý dựa trên nội dung

Nếu α_{i-1} bị loại khỏi đối số Attend, tức là,

$$\alpha_i = \text{Attend}(s_{i-1}, h)$$

Kết quả này được gọi là cơ chế chú ý dựa trên nội dung. Trong trường hợp này, Attend thường được thực hiện bằng cách cho điểm từng yếu tố trong h một cách riêng biệt và thông qua chuẩn hóa điểm số:

$$e_{i,j} = \text{Score}(s_{i-1}, h_j)$$
$$\alpha_{i,j} = \exp(e_{i,j}) / \sum_{j=1}^L \exp(e_{i,j})$$

Hạn chế chính của cơ chế này là các phần tử giống hệt nhau hoặc rất giống nhau của h được tính điểm như nhau bất kể vị trí của chúng trong dãy đầu vào. Đây là vấn đề về khi đầu vào là các đoạn giọng nói tương tự.

❖ Cơ chế chú ý dựa trên vị trí

Bên cạnh cơ chế dựa trên nội dung, cơ chế chú ý dựa trên vị trí hoạt động dựa trên tính toán sự liên kết từ trạng thái máy tạo và sự liên kết trước đó:

$$\alpha_i = \text{Attend}(s_{i-1}, \alpha_{i-1})$$

Những mặc hạn chế của cơ chế chú ý này trong nhiệm vụ nhận dạng giọng nói là cơ chế chú ý dựa trên vị trí này sẽ phải dự đoán khoảng cách giữa các âm vị do trong cơ chế chỉ sử dụng s_{i-1} , điều này dường như là rất khó do sự khác biệt của đại lượng này.

Đối với những hạn chế liên quan đến cả cơ chế dựa trên nội dung và vị trí, cơ chế chú ý kết hợp là một lựa chọn cho nhiệm vụ nhận dạng giọng nói.

Kiến trúc chung về mô hình nhận dạng giọng nói với cơ chế chú ý đã được giới thiệu, tuy nhiên, song vì mô hình chú ý không sử dụng bất kỳ giả định độc lập có điều kiện nào, nên nó thường được chứng minh là cải thiện Tỷ lệ lỗi ký tự (CER) so với CTC khi không sử dụng mô hình ngôn ngữ bên ngoài. Tuy nhiên, trong các nhiệm vụ nhận dạng giọng nói trong môi trường thực, mô hình cho kết quả kém vì sự liên kết

được ước tính trong cơ chế chú ý dễ bị hỏng do nhiễu. Một vấn đề khác là mô hình khó có thể học lại từ đầu do sự sai lệch trên các chuỗi đầu vào dài hơn, và do đó kỹ thuật cửa sổ thường được sử dụng để giới hạn khu vực được khám phá bởi cơ chế chú ý.

3.4. TỔNG KẾT

Một cách tiếp cận là khung bộ mã hóa dựa trên sự chú ý học cách ánh xạ giữa các chuỗi đầu vào và đầu ra có độ dài thay đổi trong một bước bằng cách sử dụng phương pháp thuần túy theo hướng dữ liệu. Tuy nhiên, mô hình nhận dạng giọng nói dựa trên sự chú ý gặp phải một số nhược điểm so với mô hình nhận dạng giọng nói từ đầu đến cuối kết hợp phân loại thời gian kết nối theo hai cách. Đầu tiên, mô hình dựa trên sự chú ý trong nhiệm vụ nhận dạng giọng nói cho kết quả kém (nhóm sinh viên đã huấn luyện và đánh giá mô hình), đặc biệt là trong điều kiện ồn ào và khó được đào tạo trong giai đoạn đào tạo ban đầu với các chuỗi đầu vào dài, so với CTC. Điều này là do mô hình chú ý quá linh hoạt để dự đoán sự liên kết thích hợp trong những trường hợp như vậy do thiếu các ràng buộc từ trái sang phải như được sử dụng trong CTC. Thứ hai, so với dịch máy, nhận dạng giọng nói khác hơn về mặt các chuỗi đầu vào là dài hơn nhiều (hàng nghìn khung hình phổ thay vì hàng chục từ), điều này dẫn đến một thách thức trong việc phân biệt các đoạn giọng nói tương tự trong một câu nói duy nhất. Nó cũng khác với tổng hợp chữ viết tay, vì đầu vào ồn ào hơn nhiều và không có cấu trúc rõ ràng. Vì những lý do này, nhận dạng giọng nói với cơ chế chú ý là một thách thức khi phát triển các kiến trúc dựa trên sự chú ý mới có khả năng xử lý các đầu vào dài và ồn ào.

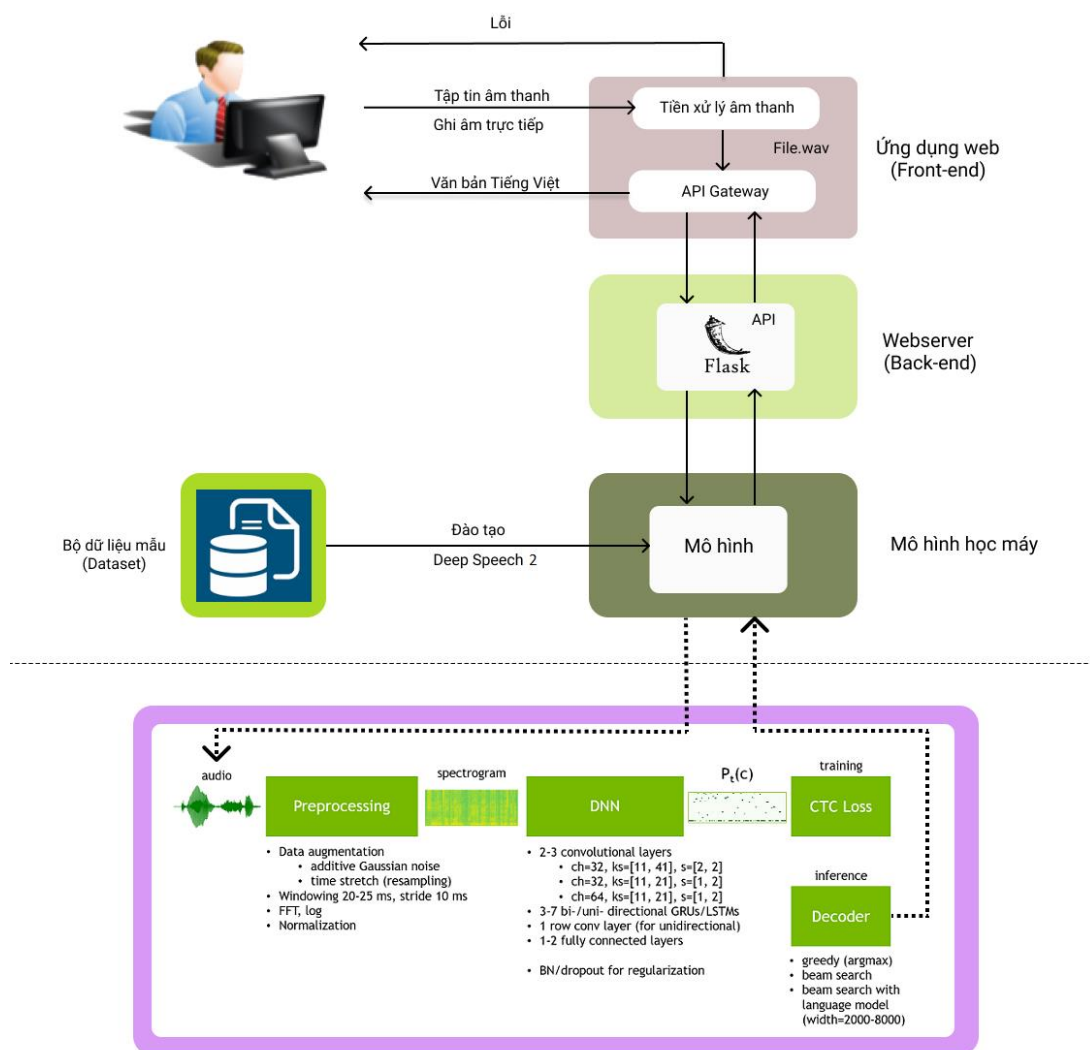
Thông qua chương 3, nhóm sinh viên đã trình bày các kiến thức cơ bản về mô hình và hai cách tiếp cận trong nhận dạng giọng nói hiện đại đó là nhận dạng giọng nói từ đầu đến cuối kết hợp với phân loại thời gian kết nối và nhận dạng giọng nói dựa trên sự chú ý, đồng thời nhóm sinh viên cũng đã dẫn chứng một số lý do mô hình nhận dạng giọng nói từ đầu đến cuối kết hợp với phân loại thời gian kết nối đạt được độ chính xác cao và ổn định hơn trong nhiệm vụ nhận dạng âm thanh Tiếng Việt.

Chương tiếp theo, nhóm sinh viên sẽ làm rõ phương pháp nghiên cứu của đề tài cho từng thành phần trong hệ thống nhận dạng âm thanh tiếng Việt với mô hình từ đầu đến cuối kết hợp với phân loại thời gian kết nối, đồng thời nhóm sinh viên cũng trình bày rõ các cải tiến được tích hợp thêm vào mô hình sẵn có, hướng xây dựng máy chủ và cả ứng dụng website mẫu.

CHƯƠNG 4. PHƯƠNG PHÁP NGHIÊN CỨU CỦA ĐỀ TÀI

4.1. TỔNG QUAN KIẾN TRÚC HỆ THỐNG

Nhóm sinh viên dự kiến xây dựng sản phẩm là trang web với kiến trúc và mô hình được minh họa trong hình 4.1.

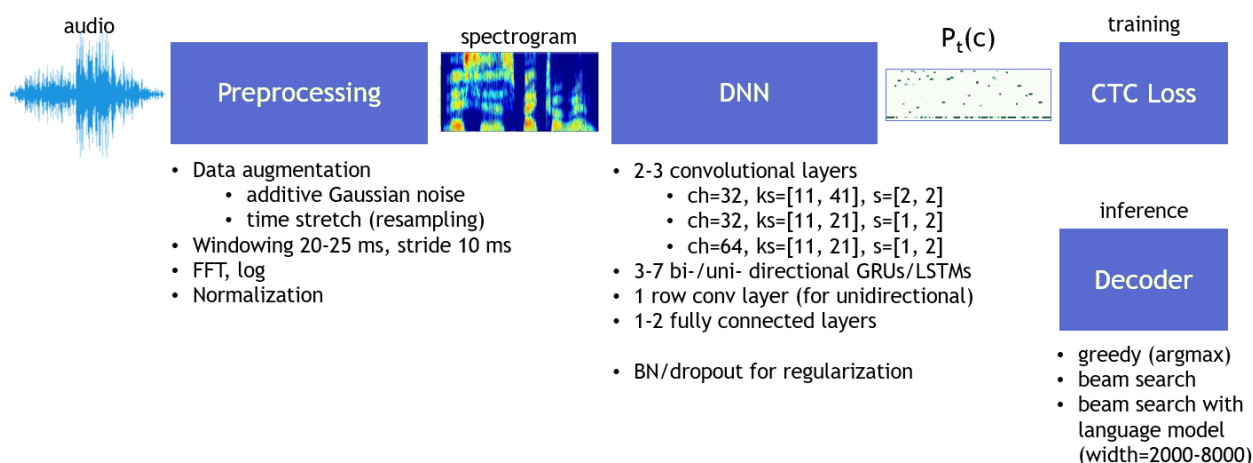


Hình 4.1 Kiến trúc tổng quan hệ thống

Đề tài sử dụng kiến trúc mô hình nhận dạng đầu cuối dựa trên kỹ thuật phân loại thời gian kết nối được xây dựng kết hợp cùng ý tưởng của bài báo DeepSpeech 2, một nghiên cứu của Baidu được công bố vào ngày 08/12/2015 tại Silicon Valley AI

Lab. Nội dung của bài báo trình bày về nhận dạng giọng nói được thực hiện trên ngôn ngữ Tiếng Anh (English) và tiếng Quan Thoại (Mandarin).

Deep Speech 2 sử dụng mô hình thần kinh hồi quy (RNN) và sử dụng phân loại thời gian kết nối để dự đoán đầu ra:



Hình 4.2. Minh họa kiến trúc mô hình

Đầu vào của hệ thống là đoạn âm thanh thô chưa được xử lý. Phần tiền xử lý (Preprocessing) lấy một tín hiệu dạng sóng âm thanh thô và chuyển nó thành một biểu đồ phổ có kích thước ($N_{\text{timesteps}}$, $N_{\text{frequency_features}}$). $N_{\text{timesteps}}$ phụ thuộc vào thời lượng của tệp âm thanh gốc. Tiếp theo học sâu (DNN) tạo ra phân phối xác suất $P_t(c)$ trên các ký tự từ vựng c cho mỗi bước thời gian t . Deep Speech 2 được đào tạo với nhiều thử nghiệm với mạng thần kinh được huấn luyện với chức năng suy giảm phân loại theo thời gian kết nối để dự đoán phiên âm giọng nói từ âm thanh. Nhóm sinh viên sử dụng tỷ lệ lỗi từ (WER) làm phương pháp được sử dụng đánh giá độ chính xác của mô hình.

Tiếp theo để đưa các từ ra khỏi mô hình được đào tạo, mô hình cần sử dụng một bộ giải mã (Decoder). Bộ giải mã chuyển đổi phân phối xác suất trên các ký tự được cho ra bởi mạng học sâu thành văn bản. Có hai loại bộ giải mã được sử dụng với các mô hình dựa trên phân loại thời gian kết nối là: (1) bộ giải mã tham lam (Greedy decoder) và (2) bộ giải mã tìm kiếm chùm (Beam search decoder) với mô hình ngôn ngữ. Một bộ giải mã tham lam xuất ra ký tự có thể xảy ra nhất ở mỗi bước thời gian.

Nó có tốc độ xử lý nhanh và có thể tạo ra các câu rất chính xác, nhưng có thể mắc nhiều lỗi chính tả nhỏ. Tuy nhiên, do bản chất của chỉ số WER, một lỗi ký tự cũng làm cho một từ không chính xác. Nhóm sinh viên quyết định chọn bộ giải mã tìm kiếm chùm với mô hình ngôn ngữ cho đánh giá độ chính xác của mô hình. Một bộ giải mã tìm kiếm chùm có chức năng ghi lại mô hình ngôn ngữ cho phép kiểm tra nhiều giải mã bằng cách chỉ định điểm cao hơn cho nhiều N-grams tùy vào mô hình ngôn ngữ nhất định đồng thời mô hình ngôn ngữ cũng giúp sửa lỗi chính tả cho văn bản đầu ra. Song bộ giải mã này cũng có những hạn chế là nó chậm hơn đáng kể so với một bộ giải mã tham lam.

Cuối cùng, đầu ra của hệ thống là một đoạn văn bản Tiếng Việt hoàn chỉnh.

Bên cạnh kết hợp với ý tưởng của bài báo DeepSpeech 2, nhóm sinh viên tiến hành cải tiến mã nguồn, dữ liệu để phù hợp với đặc trưng của ngôn ngữ Tiếng Việt, đồng thời bổ sung một số thành phần để nâng cao kết quả cho mô hình, cụ thể, nhóm sinh viên đã tích hợp mô hình dấu câu để nâng cao độ tự nhiên cho kết quả văn bản đầu ra và chọn lọc mô hình ngôn ngữ đáp ứng nhận dạng các từ Tiếng Anh thông dụng.

Phần tiếp theo, nhóm sinh viên trình bày lý thuyết cơ bản, giải pháp về kiến trúc mô hình dựa trên ý tưởng bài báo DeepSpeech 2 cũng như các chỉnh sửa giúp nâng cao sự thích hợp trên ngôn ngữ Tiếng Việt.

4.2. PHƯƠNG PHÁP TIỀN XỬ LÝ ÂM THANH

4.2.1. Tổng quan giải pháp

Để xử lý dữ liệu, âm thanh dạng sóng chuyển đổi thành chương trình quang phổ và cấp cho mạng nơ-ron để tạo ra đầu ra. Cách truyền thống để thực hiện tăng dữ liệu thường được áp dụng cho dạng sóng. Bên cạnh đó còn một cách tiếp cận khác đó là thao tác trên phổ.

Nhóm sinh viên đề xuất sử dụng kỹ thuật Short Time Fourier Transform (STFT) với mục đích tiếp cận âm thanh của tiếng nói tiếng Việt dưới dạng phổ, cụ thể

hơn là một chuỗi các vector n chiều (mỗi chiều là một giá trị thực) để phục vụ cho quá trình huấn luyện.

Biến đổi Fourier thời gian ngắn (STFT) là một chuỗi các biến đổi Fourier của tín hiệu cửa sổ. STFT cung cấp thông tin tần số được chuẩn hóa theo thời gian cho các tình huống trong đó các thành phần tần số của tín hiệu thay đổi theo thời gian, trong khi biến đổi Fourier tiêu chuẩn cung cấp thông tin tần số được tính trung bình trong toàn bộ khoảng thời gian của tín hiệu.

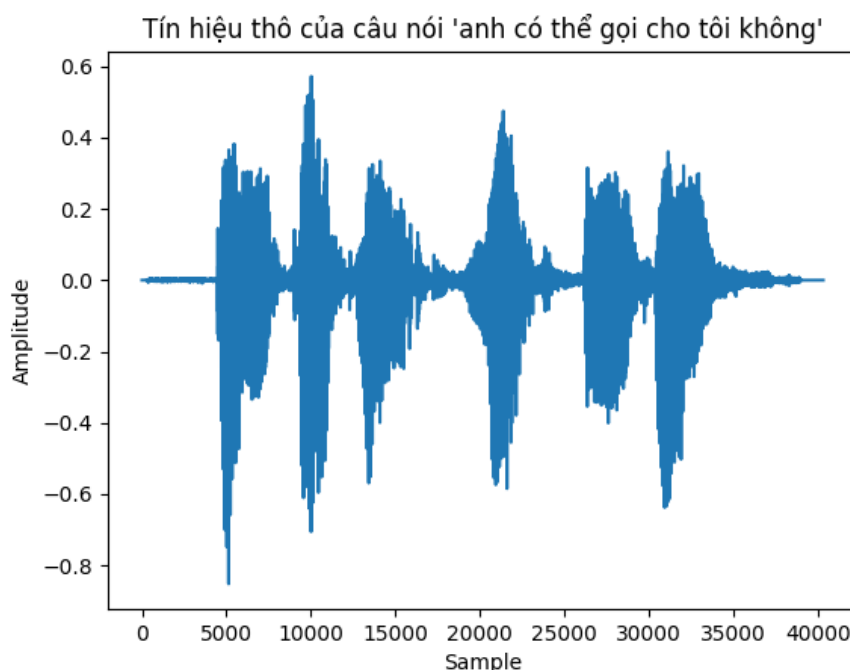
Về cơ bản phép biến đổi Fourier là quá trình phân hủy âm thanh tuần hoàn thành tổng các sóng sin mà tất cả các sóng dao động đều dao động ở các tần số khác nhau. Từ đó có thể mô tả một âm thanh rất phức tạp miễn là nó có tính chu kỳ là một tổng như sự chồng chất của một loạt các sóng sin khác nhau ở các tần số khác nhau.

4.2.2. Chi tiết giải pháp

Nhóm sinh viên sử dụng đầu vào là tập tin WAV PCM 16 bit, với tần số lấy mẫu 16000 Hz, thời lượng không vượt quá 15 giây. Tập tin WAV là tín hiệu giọng nói Tiếng Việt rõ ràng (hoặc bị nhiễu).

Mình họa ví dụ dưới đây nhóm sinh viên sử dụng câu nói: “anh có thể gọi tôi không”.

Tín hiệu thô sẽ được biểu diễn với âm độ như sau trong miền thời gian:



Hình 4.3 Tín hiệu thô trong miền thời gian của câu nói “anh có thể gọi tôi không”

Bước đầu tiên là áp dụng phân tích tín hiệu thô thành phổ sử dụng STFT. Phép biến đổi Fourier cổ điển được thiết kế để chuyển từ biểu diễn thời gian của tín hiệu sang biểu diễn tần số và ngược lại. Tuy nhiên, tín hiệu lời nói thường có đặc tính là các đặc tính tần số của chúng thay đổi tương đối chậm theo thời gian. Do đó, sẽ rất hữu ích nếu sử dụng một phép biến đổi hiển thị nội dung phổ của tín hiệu giọng nói như một hàm của thời gian. Biến đổi Fourier trong thời gian ngắn là một biến đổi như vậy. STFT là một phép biến đổi từ miền thời gian sang miền tần số thời gian. Nó tạo ra một biểu diễn "quang phổ" hai chiều. Thực tế là biểu diễn này có chứa trục tần số cũng như trục thời gian, một mặt mang lại sự hiểu biết còn thiếu trong trường hợp biểu diễn tần số hoặc thời gian thuần túy, mặt khác nó tạo ra một số vấn đề phụ.

Biến đổi Fourier $F(\omega)$ của một tín hiệu thực trên một đơn vị thời gian thường sẽ bao gồm các số phức. Bộ lọc tiền nhấn mạnh có thể được áp dụng vào tín hiệu x với công thức sau:

$$X(\omega, m) = STFT(x(n)) := DTFF(x(n - m)\omega(n))$$

trong đó:

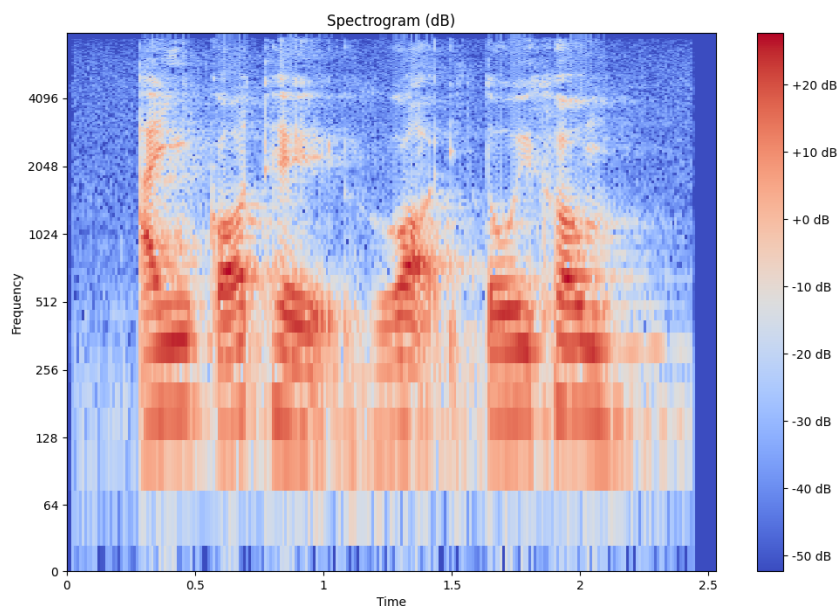
- $w(n)$ là hàm cửa sổ có độ dài RR .
- STFT của tín hiệu $x(n)$ là một hàm của hai biến: thời gian và tần số.

Bước tiếp theo, tách một phổ có ma trận số phức D thành các thành phần độ lớn (S) và pha (P) của nó, khi đó phổ là một ma trận số thực như sau:

```
array([[3.78624858e-04, 7.47191339e-05, 4.01736241e-02, ...,  
       1.32672536e-04, 1.00108684e-04, 2.23067264e-04],  
       [2.88458262e-04, 1.71806586e-03, 4.32713631e-02, ...,  
       7.15882935e-05, 1.10202267e-04, 1.57766659e-05],  
       [5.11781871e-05, 3.22661789e-03, 4.80496473e-02, ...,  
       1.08016502e-04, 1.09839411e-04, 2.04812433e-04],  
       ...,  
       [4.33522572e-05, 8.89793468e-05, 1.32498183e-04, ...,  
       6.39611412e-05, 1.16795612e-04, 6.90769694e-05],  
       [1.51771301e-05, 5.58131280e-05, 1.44885858e-04, ...,  
       2.40029752e-05, 1.05789890e-04, 3.81983746e-05],  
       [1.12807566e-05, 1.15962726e-04, 6.16297529e-05, ...,  
       2.00541682e-05, 1.08255876e-04, 2.73341147e-05]])
```

Hình 4.4 Minh họa ma trận số thực phổ của câu nói “anh có thể gọi tôi không”

Mô tả phổ của ma trận số thực đó như sau:



Hình 4.5. Hình ảnh phổ của câu nói “anh có thể gọi tôi không”

Bước cuối cùng phổ đầu vào được chuẩn hóa hàng loạt (Batch Normalization), sẽ được nhóm trình bày trong phần sau của chương, ta có được ma trận phổ đầu vào cuối cùng là một tensor các số thực.

Kết quả tiền xử lí âm thanh vào được biểu diễn ở hình :

```
tensor([[ -0.3778, -0.3787, -0.2624, ..., -0.3785, -0.3786, -0.3783],
        [ -0.3781, -0.3739, -0.2536, ..., -0.3787, -0.3786, -0.3789],
        [ -0.3788, -0.3694, -0.2401, ..., -0.3786, -0.3786, -0.3783],
        ...,
        [ -0.3788, -0.3787, -0.3786, ..., -0.3788, -0.3786, -0.3787],
        [ -0.3789, -0.3788, -0.3785, ..., -0.3789, -0.3786, -0.3788],
        [ -0.3789, -0.3786, -0.3788, ..., -0.3789, -0.3786, -0.3789]])
```

Hình 4.6. Ma trận sau khi tiền xử lí âm thanh của câu nói “anh có thể gọi tôi không”

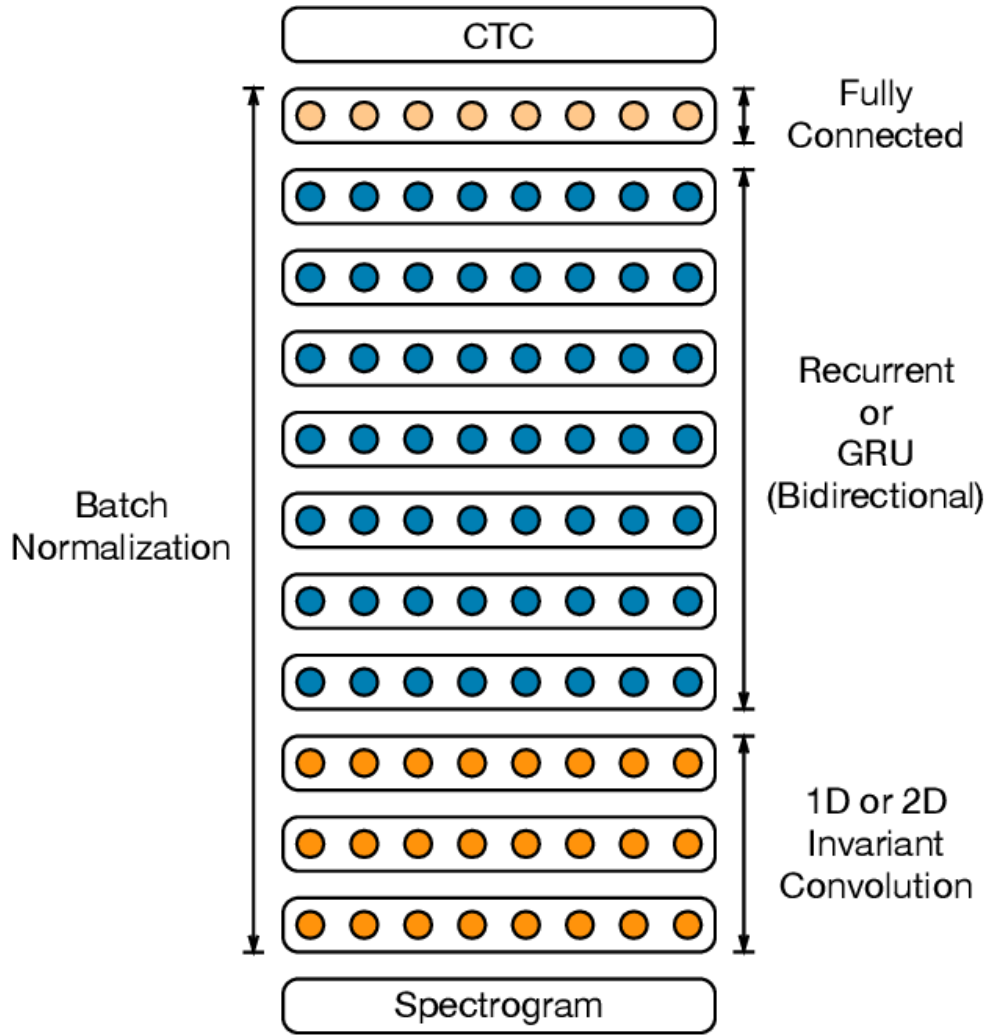
4.3. PHƯƠNG PHÁP XÂY DỰNG MÔ HÌNH ÂM THANH

4.3.1. Tổng quan giải pháp

Một mô hình nhiều lớp đơn giản với một lớp hồi quy không thể hiệu quả với tập dữ liệu lớn lời nói đã được gắn nhãn. Để mô hình học tập được tập dữ liệu lớn, cần phải tăng dung lượng mô hình thông qua chiều sâu. Nhóm sinh viên dự kiến sử dụng kiến trúc mô hình có nhiều lớp bao gồm nhiều lớp hồi quy hai chiều (bidirectional recurrent) và lớp tích tụ (convolutional). Đồng thời, để tối ưu hóa mô hình, nhóm sử dụng chuẩn hóa hàng loạt cho RNN (Batch Normalization) và một chương trình tối ưu hóa mới gọi là SortaGrad. Đồng thời nhóm cũng khai thác những bước tiến dài (long strides) giữa các đầu vào RNN để giảm việc tính toán.

4.3.2. Mô hình mạng nơ-ron hồi quy RNN

Một mạng nơ tron hồi quy được đào tạo để nhận các âm phổ đầu vào và cho ra các ma trận thể hiện xác suất xuất hiện của một kí tự trong bảng kí tự Tiếng Việt, sau đó sử dụng bộ giải mã để đạt được văn bản Tiếng Việt cuối cùng. Kiến trúc mô hình RNN được minh họa trong hình.



Hình 4.7 Minh họa kiến trúc hệ thống DeepSpeech 2

Hình 4.7 minh họa hệ thống Deepspeech 2 gồm 1 đến 3 lớp tích chập và 1 đến 7 lớp RNN hoặc GRU.

Cụ thể, trong một tập huấn luyện $X = \{(x^i, y^i), i \in N^*\}$ với x là phát âm đầu vào và y là nhãn văn bản Tiếng Việt tương ứng với đầu ra. Mỗi phát âm x^i là một chuỗi thời gian có độ dài là T^i . Trong đó, mỗi lát cắt thời gian nhất định của T^i là một vector của các đặc trưng âm thanh $x_t^i, t = 1, 2, \dots, T^i$.

Nhóm sinh viên tiếp tục sử dụng âm phổ của các đoạn âm thanh được chuẩn hóa bằng STFT được trình bày ở phần trước đó làm các đặc trưng cho hệ thống, do đó $x_{t,p}^i$ biểu thị cho độ lớn của khoảng tần số thứ p trong khung âm thanh tại thời điểm t .

Mục tiêu của RNN là chuyển đổi đầu vào x^i thành chuỗi ký tự cho nhãn y^i tương ứng. Với mỗi khoảng thời gian t , RNN cho ra một xác suất dự đoán trên tập hợp các ký tự: $p(l_t|x)$ với l_t là một ký tự trong bảng chữ cái $l_t \in \{a, \text{ă}, \text{â}, b, c, \text{á}, \text{à}, \text{ã}, \dots\}$ hoặc khoảng trắng,...

Mô hình RNN mà nhóm dự kiến xây dựng bao gồm một số lớp đơn vị ẩn (hidden unit), một hoặc nhiều lớp phức hợp (convolution layer), tiếp đó là một hoặc nhiều lớp hồi quy (recurrent layer), cuối cùng là một hoặc nhiều lớp kết nối đầy đủ (connected layer).

Phần đầu tiên của mạng là một hoặc nhiều vòng tích chập theo thời gian của đầu vào. Đoạn âm thanh sau khi được tiền xử lí sẽ được đưa vào lớp này. Biểu diễn ẩn ở lớp l được cho bởi h^l với quy ước rằng h^0 biểu diễn của đầu vào x . Cửa sổ ngữ cảnh (Context window) là số lượng từ được dự đoán có thể xuất hiện trong phạm vi của từ đã cho. Đối với cửa sổ ngữ cảnh (context window) có kích thước c , nút kích hoạt thứ i ở bước thời gian t của lớp tích chập được đưa ra bởi công thức :

$$h_{t,i}^l = f(w_i^{l \circ} h_{t-c:t+c}^{l-1}).$$

Trong đó:

- Ký hiệu \circ biểu thị yếu tố học tập giữa bộ lọc thứ i và của sổ ngữ cảnh của các lớp kích hoạt trước đó và hàm f là một hàm phi tuyến tính bậc nhất.
- c là kích thước của cửa sổ ngữ cảnh.

Hàm chỉnh lưu tuyến tính (ReLU) được rút gọn thành $\sigma(x) = \min\{\max\{x, 0\}, 2\}$ làm độ phi tuyến tính cho mô hình. Lấy mẫu phụ là một kỹ thuật đã được phát minh ra để giảm sự phụ thuộc vào định vị chính xác trong các danh sách các ảnh xạ đặc trưng được tạo ra bởi các lớp phức hợp trong CNN. Mẫu phụ là một trong bộ nhỏ (R_r) được chọn bằng cách lấy ngẫu nhiên một cách đơn giản từ mẫu được sử dụng để bắt đầu lặp lại trên mẫu đó (N_r). Nhóm sinh viên dự kiến lấy mẫu phụ bằng cách tách một số khung tính chập trong một số lớp, thường là lớp đầu tiên. Mục đích của việc này là rút ngắn số bước thời gian cho các lớp hồi quy ở trên.

Theo sau lớp tích tụ là một hoặc nhiều lớp hồi quy hai chiều. Ở mỗi lớp hồi quy hai chiều đều có tham số chuẩn hóa, trừ lớp đầu tiên. Các lớp hồi quy kích hoạt tiến theo thời gian ($\overrightarrow{h^l}$) và lùi theo thời gian ($h^{l\leftarrow}$) được tính bằng công thức:

$$\overrightarrow{h_t^l} = g(h_t^{l-1}, \overrightarrow{h_{t-1}^l})$$

$$h_t^{l\leftarrow} = g(h_t^{l-1}, h_{t+1}^{l\leftarrow})$$

Hai tập hợp các nút kích hoạt được cộng lại để tạo thành các đầu ra cho lớp h^l ,
 $h^l = \overrightarrow{h^l} + h^{l\leftarrow}$.

Hàm $g(.)$ có thể rút gọn bằng phép hồi quy:

$$\overrightarrow{h_t^l} = f(W^l \cdot h_t^{l-1} + \overrightarrow{U^l} \cdot \overrightarrow{h_{t-1}^l} + b^l) \quad (1)$$

Trong đó

- W^l : trọng số tại lớp đơn vị ẩn thứ l tương ứng của đầu vào.
- $\overrightarrow{U^l}$: trọng số của ma trận hồi quy tại lớp ẩn thứ l .
- b^l : độ sai lệch (bias) của lớp ẩn thứ l .

Mô hình được áp dụng một hoặc nhiều lớp kết nối đầy đủ tiếp theo sau các lớp hồi quy hai chiều, với:

$$h_t^l = f(W^l \cdot h_t^{l-1} + b^l).$$

Lớp đầu ra L là một phép tính softmax phân bố xác suất trên các ký tự trong bộ ký tự của ngôn ngữ Tiếng Việt $l = \{a, \hat{a}, \hat{a}, b, c, d, \hat{d}, \dots\}$ được cho bởi công thức:

$$p(l_t = k|x) = \frac{\exp(w_k^L \cdot h_t^{L-1})}{\sum_j \exp(w_j^L \cdot h_t^{L-1})}$$

Minh họa các tham số của lớp kết nối đầy đủ, với mô hình RNN có 2 lớp tích tụ:

```

model.conv
MaskConv(
  (seq_module): Sequential(
    (0): Conv2d(1, 32, kernel_size=(41, 11), stride=(2, 2), padding=(20, 5))
    (1): BatchNorm2d(32, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
    (2): Hardtanh(min_val=0, max_val=20, inplace=True)
    (3): Conv2d(32, 32, kernel_size=(21, 11), stride=(2, 1), padding=(10, 5))
    (4): BatchNorm2d(32, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
    (5): Hardtanh(min_val=0, max_val=20, inplace=True)
  )
)

```

Hình 4.8 Cấu trúc của mô hình với 2 lớp tích chập thể hiện ở mã nguồn

Tiếp theo lớp tích tụ là 5 lớp hồi quy hai chiều với kích thước mỗi lớp ẩn là 1600, tần số lấy mẫu (sample rate) của âm thanh đầu vào là 16000Hz, kích thước cửa sổ (windows size) là 0.02.

```

model.rnn
Sequential(
  (0): BatchRNN(
    (rnn): LSTM(1312, 1600, bidirectional=True)
  )
  (1): BatchRNN(
    (batch_norm): SequenceWise (
      BatchNorm1d(1600, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True))
    (rnn): LSTM(1600, 1600, bidirectional=True)
  )
  (2): BatchRNN(
    (batch_norm): SequenceWise (
      BatchNorm1d(1600, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True))
    (rnn): LSTM(1600, 1600, bidirectional=True)
  )
  (3): BatchRNN(
    (batch_norm): SequenceWise (
      BatchNorm1d(1600, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True))
    (rnn): LSTM(1600, 1600, bidirectional=True)
  )
  (4): BatchRNN(
    (batch_norm): SequenceWise (
      BatchNorm1d(1600, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True))
    (rnn): LSTM(1600, 1600, bidirectional=True)
  )
)

```

Hình 4.9 Cấu trúc của mô hình với 5 lớp hồi quy thể hiện ở mã nguồn

Cuối cùng là các lớp kết nối đầy đủ (fully connected layer) với kích thước mỗi lớp hồi quy hai chiều là 1600 và số lượng các kí tự của Tiếng Việt là 93:

```
model.fc
Sequential(
  (0): SequenceWise (
    Sequential(
      (0): BatchNorm1d(1600, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
      (1): Linear(in_features=1600, out_features=93, bias=False)
    )
  )
)
```

Hình 4.10 Cấu trúc mô hình với 1 lớp kết nối đầy đủ được thể hiện ở mã nguồn

Mô hình được huấn luyện bằng cách sử dụng hàm mất mát CTC $L(x, y; \theta)$, với (x, y) là cặp giá trị đầu vào và đầu ra, θ là các tham số hiện tại của mạng.

Đạo hàm L theo các tham số của mạng $\nabla_{\theta} L(x, y; \theta)$ được sử dụng để cập nhật thông số của mạng thông qua sự lan truyền ngược (backpropagation) bằng thuật toán thời gian.

4.3.3. Chuẩn hóa hàng loạt (Batch normalization)

Để mở rộng mô hình khi tăng thêm tập dữ liệu huấn luyện, nhóm sinh viên dự kiến tăng độ sâu của mạng bằng cách thêm nhiều lớp đơn vị ẩn hơn thay vì làm cho mỗi lớp lớn hơn. Chuẩn hóa hàng loạt (BatchNorm) là một kỹ thuật đào tạo các mạng nơ-ron theo chiều sâu, chuẩn hóa các đầu vào thành một lớp cho từng lô nhỏ. Mục đích chính của chuẩn hóa hàng loạt là giúp ổn định quá trình đào tạo và giảm số lượng epoch cần thiết để đào tạo mô hình.

Trong một lớp chuyển tiếp (feed- forward), theo sau bởi một phép biến đổi phi tuyến tính $f(\cdot)$, mô hình được thêm một phép biến đổi BatchNorm bằng cách áp dụng $f(B(Wh))$ thay vì $f(Wh + b)$, trong đó B được tính với công thức:

$$B(x) = \gamma \left(\frac{x - E[x]}{\sqrt{Var[x] + \epsilon}} \right) + \beta$$

Trong đó:

- E : giá trị trung bình thực hiện trên một lô nhỏ.

- Var: phương sai thực nghiệm trên một lô nhỏ.
- γ, β : các tham số cho phép lớp chia tỉ lệ và dịch chuyển từng đơn vị ẩn như mong muốn, đây là các tham số có thể học được.
- ϵ : hằng số rất nhỏ dùng để ổn định số.

Trong các lớp tích chập, giá trị trung bình và phương sai được ước tính trên các đơn vị đầu ra tạm thời cho một bộ lọc tích chập nhất định trên lô nhỏ.

Dựa theo ý tưởng bài báo, nhóm sinh viên cũng nghiên cứu hai phương pháp mở rộng BatchNorm cho RNN hai chiều:

Phương pháp đầu tiên được thực hiện bằng cách thêm một phép biến đổi BatchNorm ngay trước mọi trường hợp phi tuyến tính, phương trình $g(\cdot)$ sẽ trở thành:

$$\vec{h}_t^l = f(B(W^l \cdot h_t^{l-1} + \vec{U}^l \cdot \vec{h}_{t-1}^l)).$$

Trong phương pháp này, trung bình và phương sai được tích lũy qua một bước thời gian duy nhất của một lô nhỏ. Sự phụ thuộc tuần tự giữa các bước thời gian ngăn cản việc tính trung bình trên tất cả các bước thời gian. Kỹ thuật này không dẫn đến cải tiến trong việc tối ưu hóa. Điều này cũng tỏ ra không hiệu quả và rất phức tạp trong việc lan truyền ngược (back propagation) [19].

Phương pháp thứ hai dựa trên chuẩn hóa theo trình tự theo bài báo trình bày có thể khắc phục được những vấn đề này. Tính toán được cho theo công thức:

$$\vec{h}_t^l = f(B(W^l \cdot h_t^{l-1}) + \vec{U}^l \cdot \vec{h}_{t-1}^l).$$

Trong phương pháp này, qua mỗi đơn vị ẩn, trung bình và phương sai được tính toán trên tất cả các lô nhỏ theo chiều dài của chuỗi. Phương pháp BatchNorm hoạt động tốt quá trình huấn luyện, nhưng sẽ khó triển khai trên hệ thống nhận dạng giọng nói khi tự đoán, vì hệ thống thường sẽ dự đoán một câu nói duy nhất trong triển khai hơn là một lô trong huấn luyện. Việc chuẩn hóa mỗi tế bào thần kinh về giá trị trung bình và phương sai của chính nó so với khi huấn luyện là tính toán trên cả trình tự sẽ làm giảm hiệu suất. Cách tối ưu được nhóm sinh viên sử dụng là lưu trữ giá trị trung

bình được thu thập trong quá trình huấn luyện và sử dụng chúng để đánh giá trong quá trình triển khai lên hệ thống. Sử dụng kỹ thuật này, nhóm có thể đánh giá một lời nói duy nhất tại một thời điểm với kết quả tốt hơn so với đánh giá một loạt lớn đầu vào.

4.3.4. SortaGrad

Khi huấn luyện các âm thanh mẫu có độ dài khác nhau đặt ra một số thách thức về thuật toán. Một giải pháp khả thi là cắt bớt sự lan truyền ngược theo thời gian để tất cả mẫu có độ dài và trình tự giống nhau trong quá trình đào tạo. Tuy nhiên, điều này hạn chế khả năng học tập dài hạn cho mô hình.

Hàm chi phí CTC (CTC Cost Function) mà nhóm sẽ sử dụng ngầm định phụ thuộc vào độ dài câu nói:

$$\mathcal{L}(x, y; \theta) = - \log \sum_{l \in \text{Align}(x, y)} \prod_t^T p_{ctc}(l_t | x; \theta)$$

Trong đó;

- ❖ Align (x, y) là tập hợp tất cả các căn chỉnh có thể có của các ký tự của phiên mã y với các âm thanh đầu vào x dưới toán tử CTC. Ví dụ: từ “học” có các căn chỉnh h-ọc, họ-c, -học,...

Trong phương trình trên, số hạng bên trong là tích theo các bước thời gian của dãy, số hạng này thu hẹp lại theo độ dài của dãy vì $p_{ctc}(l_t | x; \theta) < 1$, hậu quả dẫn đến $\prod_t^T p_{ctc}(l_t | x; \theta) < 1$.

4.3.5. Biến đổi tần số (Frequency Convolution)

Tích chập tạm thời (Temporal convolution) thường được sử dụng trong nhận dạng giọng nói để mô hình hóa hiệu quả theo thời gian cho các phát âm có độ dài thay đổi. Nhiều mô hình giọng nói có lớp đầu tiên xử lý các khung đầu vào với một cửa sổ ngữ cảnh. Ngoài ra, lấy mẫu phụ là điều cần thiết để mạng nơ-ron hồi quy có thể kiểm soát tính toán với âm thanh có tốc độ mẫu cao (high sample-rate).

Các biến đổi trong miền tần số và thời gian khi áp dụng cho các đặc trưng đầu vào trước bất kỳ quá trình xử lý nào khác có thể cải thiện hiệu suất nhận dạng giọng nói. Sự biến đổi tần số cố gắng mô hình hóa phương sai do dự biến đổi của loa ngắn hơn so với các mạng kết nối đầy đủ lớn. Vì thứ tự âm phổ của các đặc trưng bị loại bỏ bởi các lớp kết nối đầy đủ và lớp hồi quy, nên biến đổi tần số hoạt động tốt hơn trên các lớp đầu tiên của mạng.

4.3.6. Bước sóng (Striding)

Trong các lớp tích chập, áp dụng bước sóng dài hơn và bối cảnh (context) rộng hơn để tăng tốc quá trình đào tạo vì chỉ cần ít bước thời gian hơn để mô hình hóa một đầu vào. Việc lấy âm thanh β mẫu đầu vào thông qua FFT và bước sóng chập làm giảm số lượng bước thời gian và tính toán cần thiết trong các lớp, nhưng làm giảm hiệu suất.

4.3.7. Bộ giải mã

Để đưa các từ ra khỏi một mô hình được đào tạo, người ta cần sử dụng bộ giải mã. Bộ giải mã chuyển đổi phân phối xác suất của ma trận kết quả trên các ký tự thành văn bản Tiếng Việt cuối cùng. Có hai loại bộ giải mã thường được sử dụng với các mô hình dựa trên CTC: bộ giải mã tham lam và bộ giải mã tìm kiếm chùm với tính điểm lại mô hình ngôn ngữ.

4.3.7.1. Bộ giải mã tham lam

Một bộ giải mã tham lam xuất ra ký tự có thể xảy ra nhất ở mỗi bước thời gian. Nó rất nhanh và nó có thể tạo ra các bản ghi rất gần với cách phát âm gốc. Nhưng nó có thể mắc nhiều lỗi chính tả nhỏ.

Nhóm sinh viên sử dụng câu nói: “anh có thể gọi tôi không” để dự đoán văn bản đầu ra của mô hình với bộ giải mã tham lam. Mô hình được huấn luyện có 5 lớp hồi quy hai chiều, mỗi lớp có 1600 lớp đơn vị ẩn và được huấn luyện với LSTM. Một đầu ra của mô hình sau khi qua mạng nơ – ron được một ma trận. Một ma trận 3 chiều thể hiện phân phối xác suất của từng ký tự trên mỗi bước thời gian.


```

out
tensor([[[[1.0000e+00, 0.0000e+00, 0.0000e+00, ..., 0.0000e+00,
          0.0000e+00, 0.0000e+00],
          [1.0000e+00, 0.0000e+00, 0.0000e+00, ..., 0.0000e+00,
          0.0000e+00, 0.0000e+00],
          [1.0000e+00, 0.0000e+00, 0.0000e+00, ..., 0.0000e+00,
          0.0000e+00, 0.0000e+00],
          ...,
          [1.0000e+00, 0.0000e+00, 0.0000e+00, ..., 0.0000e+00,
          0.0000e+00, 0.0000e+00],
          [1.0000e+00, 0.0000e+00, 0.0000e+00, ..., 0.0000e+00,
          0.0000e+00, 0.0000e+00],
          [1.0000e+00, 0.0000e+00, 5.9605e-08, ..., 0.0000e+00,
          0.0000e+00, 0.0000e+00]]], device='cuda:0', dtype=torch.float16,
        grad_fn=<SoftmaxBackward>)]
out.size()
torch.Size([1, 127, 93])

```

Hình 4.11 Ma trận thể hiện phân phối xác suất từng kí tự

Bộ giải mã tham sẽ chọn những kí tự có xác suất cao nhất trên mỗi bước thời gian và trả về ma trận là thứ tự các kí tự dự đoán trong bảng chữ cái Tiếng Việt của văn bản đầu ra:

```

max_probs
tensor([[ 0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0, 42,  2, 58, 58, 92, 92,
          92, 92, 92, 92, 92, 92, 92, 59,  0,  5, 92, 92, 92, 92, 92, 92, 92, 87, 87,
          58, 58, 47, 47, 92, 92, 92, 92, 92, 92, 92, 92, 92, 92, 92, 92, 92, 92, 92,
          92, 92, 92, 92, 92, 37, 37,  0, 56, 49, 49, 92, 92, 92, 92, 92, 92, 92, 92,
          92, 92, 92, 92, 92, 92, 87,  0, 27, 49, 92, 92, 92, 92, 92, 92, 92, 12,
          12, 58, 58, 27,  2, 37, 37,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,
          0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,
          0]], device='cuda:0')
max_probs.size()
torch.Size([1, 127])

```

Hình 4.12 Kết quả ma trận với bộ giải mã tham lam của câu nói minh họa.

Sau khi có được ma trận thứ tự các kí tự, nhóm dựa vào bảng chữ cái Tiếng Việt để ánh xạ thành văn bản hoàn chỉnh.

```
string_offsets
tensor([12, 13, 14, 16, 25, 27, 28, 34, 36, 38, 40, 59, 62, 63, 65, 78, 80, 81,
        82, 89, 91, 93, 94, 95], dtype=torch.int32)
string
'anh có thể gọi tôi không'
```

Hình 4.13 Văn bản đầu ra “anh có thể gọi tôi không” với bộ giải mã tham lam.

4.3.7.2. Bộ giải mã tìm kiếm chùm với mô hình ngôn ngữ

Bộ giải mã tham lam đem lại tốc độ nhanh cho quá trình đào tạo nhưng nó có thể mắc nhiều lỗi chính tả nhỏ. Hơn thế nữa, do bản chất của chỉ số WER, ngay cả một lỗi ký tự cũng làm cho cả một từ không chính xác. Một bộ giải mã tìm kiếm chùm với tính điểm lại mô hình ngôn ngữ cho phép kiểm tra nhiều giải mã có thể cùng một lúc với việc ấn định điểm cao hơn cho nhiều N-gram có thể xảy ra hơn theo một mô hình ngôn ngữ nhất định. Việc tích hợp chung với mô hình ngôn ngữ giúp sửa lỗi chính tả. Tuy nhiên, nhược điểm là nó chậm hơn đáng kể so với một bộ giải mã tham lam.

Nhóm dự kiến sử dụng thuật toán tìm kiếm chùm (BeamSearch) để tìm phiên âm tối ưu. Ý tưởng của BeamSearch là tìm kiếm từ có xác suất xuất hiện cao nhất, và từ này sẽ là vector đầu vào để dự đoán từ tiếp theo.

Thuật toán BeamSearch có tham số gọi là beam width, tại mỗi bước dự đoán, thay vì chọn từ xác suất lớn nhất, thuật toán chọn beam width kết quả có xác suất cao nhất. Tiếp tục tính toán các bộ xác suất cho đến khi kết thúc, ta sẽ thu được một cây xác suất. Nhân các xác suất của cả chuỗi có điều kiện, câu kết quả là câu có xác suất lớn nhất.

Sử dụng ma trận kết quả của ví dụ ở bộ giải mã tham lam, ma trận phân phối xác suất này sẽ được áp dụng thuật toán tìm kiếm chùm với beam width là 1024 và cho ra kết quả:

```

out
tensor([[[ 42, 2, 58, ..., -2096100000,
          -1089857892, 0],
         [ 42, 2, 58, ..., 1005404160,
          1362262600, 1058044060],
         [ 42, 2, 58, ..., 0,
          1004933632, 2069648372],
         ...,
         [ 42, 2, 58, ..., 1064607442,
          -1597555776, 1064642830],
         [ 42, 2, 58, ..., 874244856,
          -1082910581, 2061943196],
         [ 42, 2, 58, ..., -1081720921,
          -1928527872, 1060994559]]], dtype=torch.int32)
out.size()
torch.Size([1, 1024, 127])

```

Hình 4.14 Ma trận phân phối xác suất với bộ giải mã tìm kiếm chùm.

Sau khi có được ma trận thứ tự các kí tự trên từng bước thời gian, thuật toán tìm kiếm chùm dựa vào bảng chữ cái Tiếng Việt để ánh xạ thành văn bản hoàn chỉnh cho từng kết quả trong beam width.

```

0000: 'anh có thể gọi tôi không'
0001: 'anh có thể gọi tôi không '
0002: 'anh có thể gọi tôi không'
0003: 'anh có thể gọi tôi không'
0004: 'anh có thể gọi tôi không'
0005: 'anh có thể gọi tôi không'
0006: 'anh có thể gọi tôi không'
0007: 'anh có thể gọi tôi không'
0008: 'anh có thể gọi tôi không'
0009: 'anh có thể gọi a tôi không'
0010: 'anh có thể gọi tôi không'
0011: 'anh có thể gọi tôi không'
0012: 'anh có thể gọi tôi không'
0013: 'anh có thể gọi t tôi không'
0014: 'anh có thể gọi tôi không r'
0015: 'anh có thể gọi tôi không l'
0016: 'anh có thể gọi tôi không g'
0017: 'anh có thể gọi h tôi không'
0018: 'anh có thể gọi tôi không i'
0019: 'anh có thể gọi tôi không b'
0020: 'anh có thể gọi tôi không h'
0021: 'anh có thể gọi tôi không u'
0022: 'anh có thể gọi tôi không a'
0023: 'anh có thể gọi tôi không d'
0024: 'anh có thể gọi tôi k không'

```

Hình 4.15 Minh họa một số kết quả với bộ giải mã tìm kiếm chùm.

Cuối cùng nhóm chọn kết quả trả về đầu tiên trong tất cả kết quả làm kết quả cuối cùng của mô hình khi sử dụng thuật toán tìm kiếm chùm.

Mô hình ngôn ngữ (Language model)

Mô hình RNN được đào tạo qua bộ dữ liệu lớn. Để mạng có thể đạt độ chính xác cao nhất đồng thời cải thiện tỉ lệ lỗi từ, nhóm dự kiến sử dụng mô hình ngôn ngữ n-gram (cụ thể là 5-gram) vì chúng mở rộng quy mô phù hợp với lượng lớn văn bản không được đánh nhãn.

Mô hình 5-grams của câu nói “bạn cho tôi mượn được không”

\1-grams:

-1.20412 <unk> 0
 0 <s> -0.30103
 -0.87312675 </s> 0
 -0.87312675 bạn -0.30103
 -0.87312675 cho -0.30103
 -0.87312675 tôi -0.30103
 -0.87312675 mượn -0.30103
 -0.87312675 được -0.30103
 -0.87312675 không -0.30103

\2-grams:

-0.24644431 không </s> 0
 -0.24644431 <s> bạn -0.30103
 -0.24644431 bạn cho -0.30103
 -0.24644431 cho tôi -0.30103
 -0.24644431 tôi mượn -0.30103
 -0.24644431 mượn được -0.30103
 -0.24644431 được không -0.30103

\3-grams:

-0.105970904 được không </s> 0
 -0.105970904 <s> bạn cho -0.30103
 -0.105970904 bạn cho tôi -0.30103
 -0.105970904 cho tôi mượn -0.30103
 -0.105970904 tôi mượn được -0.30103
 -0.105970904 mượn được không -0.30103

\4-grams:

-0.049761247 mượn được không </s> 0
 -0.049761247 <s> bạn cho tôi -0.30103
 -0.049761247 bạn cho tôi mượn -0.30103
 -0.049761247 cho tôi mượn được -0.30103
 -0.049761247 tôi mượn được không -0.30103

\5-grams:

```

-0.024168313    tôi mượn được không </s>
-0.024168313    <s> bạn cho tôi mượn
-0.024168313    bạn cho tôi mượn được
-0.024168313    cho tôi mượn được không

\end\

```

Trong quá trình tìm hiểu, phiên âm y tối đa hóa $Q(y)$:

$$Q(y) = \log \log (p_{ctc}(x)) + \alpha \log \log (p_{lm}(y)) + \beta word_count(y) \quad (3)$$

Trọng số α kiểm soát các đóng góp tương đối của mô hình ngôn ngữ và mạng CTC.

Trọng số β khuyến khích nhiều từ hơn trong phiên âm. Các tham số này được điều chỉnh nhiều hơn trong một tập phát triển.

4.4. PHƯƠNG PHÁP HẬU XỬ LÝ VĂN BẢN KẾT QUẢ

Phần này tổng hợp các bổ sung vào mô hình mà nhóm sinh viên thực hiện nhằm nâng cao độ tự nhiên cho văn bản kết quả thông qua tích hợp mô hình dấu câu và nhận dạng từ Tiếng Anh, kết quả này cũng là các cải tiến so với kết quả của luận văn trước đây.

4.4.1. Mô hình dấu câu

4.4.1.1. Vấn đề thực tế

Kết quả dự đoán của mô hình là một đoạn văn bản Tiếng Việt thô (chỉ chứa các từ Tiếng Việt, không có dấu câu). Trong thực tế, văn bản tiếng Việt cần được cải thiện tính tự nhiên hơn bằng việc được bổ sung thêm các thành phần dấu câu. Để thực hiện điều đó, nhóm sinh viên tiến hành đưa văn bản dự đoán của mô hình qua một mô hình dấu câu.

4.4.1.2. Giải pháp

Nhóm sinh viên tiến hành thu thập và xây dựng một mô hình dấu câu. Mô hình này được nhóm sinh viên huấn luyện lại dựa trên ý tưởng của tài liệu “Vietnamese

Punctuation Prediction Using Deep Neural Networks” do nhóm tác giả Thuy Pham, Nhu Nguyen, Quang Pham, Han Cao, Binh Nguyen biên soạn. Kiến trúc mô hình được thực hiện dựa trên học sâu kết hợp với BiLSTM và mô hình chú ý với mất tiêu cự.

Nhóm sinh viên tiến hành huấn luyện trên tập dữ liệu của trang báo điện tử Báo mới với gần 40,000 bài báo, quá trình cài đặt và huấn luyện 3-4 ngày. Kết quả huấn luyện của mô hình được đóng gói trong thư mục điểm dừng Checkpoint, mô hình nhận dạng âm thanh sau khi cho kết quả văn bản sẽ được đưa vào mô hình dấu câu để xử lý cho ra kết quả cuối cùng.

Bên cạnh đó, nhóm đã cải tiến thêm mã nguồn, nhóm tự cài đặt mã nguồn cho phép nhận vào một tập tin văn bản và cho ra văn bản tích hợp với dấu câu. Điều này phục vụ cho việc tích hợp với mô hình nhận dạng âm thanh trước đó.

4.4.2. Nhận dạng từ Tiếng Anh thông dụng

4.4.2.1. Đặt vấn đề

Bên cạnh tăng sự tự nhiên của văn bản đầu ra thông qua việc tích hợp dấu câu, nhóm sinh viên đồng thời xây dựng hệ thống giúp nhận dạng các từ mượn thông dụng của Tiếng Anh trong văn bản Tiếng Việt, một số minh họa cho các từ mượn Tiếng Anh thông dụng: “*website* thương mại điện tử”, “*click* chuột”, “*email* điện tử”,...

4.4.2.2. Giải pháp

Nhóm sinh viên đã cải tiến mô hình nhận dạng âm thanh Tiếng Việt này thông qua việc cải tiến mô hình ngôn ngữ, khi sử dụng với bộ giải mã tìm kiếm chùm. Nhóm đã tiến hành thu thập thêm dữ liệu cho mô hình ngôn ngữ bao gồm các từ Tiếng Anh thông dụng được sử dụng trong ngôn ngữ Tiếng Việt, và chọn lọc các câu nói chính xác để đạt được độ chính xác cao nhất. Nhóm sinh viên thực hiện dựa trên việc thu thập, sưu tầm dữ liệu từ các trang báo điện tử mà một số nguồn khác với đa dạng từ Tiếng Anh thông dụng trong nhiều lĩnh vực, cụ thể nhóm thu thập được 2.2GB dữ liệu

văn bản từ trang báo điện tử Dân trí, cùng với khoảng 800MB dữ liệu văn bản Tiếng Việt từ các trang truyện ngắn, báo nói,.. được nhóm thu thập và chuẩn hóa.



nếu anh đặt phòng online thông qua các website đặt phòng trực tuyến thì giá phòng sẽ tiết kiệm được hơn rất nhiều
nếu bán hàng online thì đăng ký trên website của bộ công thương công bố các giấy phép trên trang bán hàng của mình để khách hàng
yên tâm
nếu bạn không có thời gian để theo dõi giá phòng hãy đăng ký nhận thông tin qua các website đặt phòng khách sạn
nếu bạn không thể tới tận croatia để thăm bảo tàng này bạn cũng có thể ghé qua website của bảo tàng và chia sẻ ấn danh những câu
chuyện của mình
nếu bất cứ thông số nào không đạt viện sẽ công bố rộng rãi trên website của viện
nếu cán bộ của đảng quản lý thì đưa lên website của đảng
nếu chưa đủ chỉ tiêu trường sẽ có thông báo cụ thể trên website
nếu có một nhân viên trung thành làm việc hai mươi bốn tháng bảy không đòi tăng lương không cần nghỉ phép không chia doanh số lu
ôn luôn tận tụy chăm sóc khách hàng mọi lúc mọi nơi bất kể ngày đêm thì đó đích thị là website của bạn
nếu có nhu cầu đặt hàng người tiêu dùng sẽ ăn trực tiếp vào sản phẩm trên website cho vào giỏ hàng
nếu đăng tuyển trên website hay các trang tìm việc thì rất nhiều người ứng tuyển nhưng ở phiên giao dịch việc làm trực tiếp chún
g tôi có thể tìm kiếm lựa chọn và lôi kéo được những ứng viên đã có kinh nghiệm tôi kỳ vọng là vậy đại diện phụ trách nhân sự ng
ân hàng chia sẻ
nếu đặt hàng tại các website nước ngoài mà phổ biến nhất là amazon ebay người dùng cần đăng ký tài khoản và thẻ thanh toán quốc
tế
nếu đồng ý tham gia và nộp tiền nhà đầu tư sẽ được tạo id cung cấp mật khẩu để truy cập quản lý tài khoản cá nhân tại website
nếu em quan tâm có thể tham khảo thông tin trên website công ty

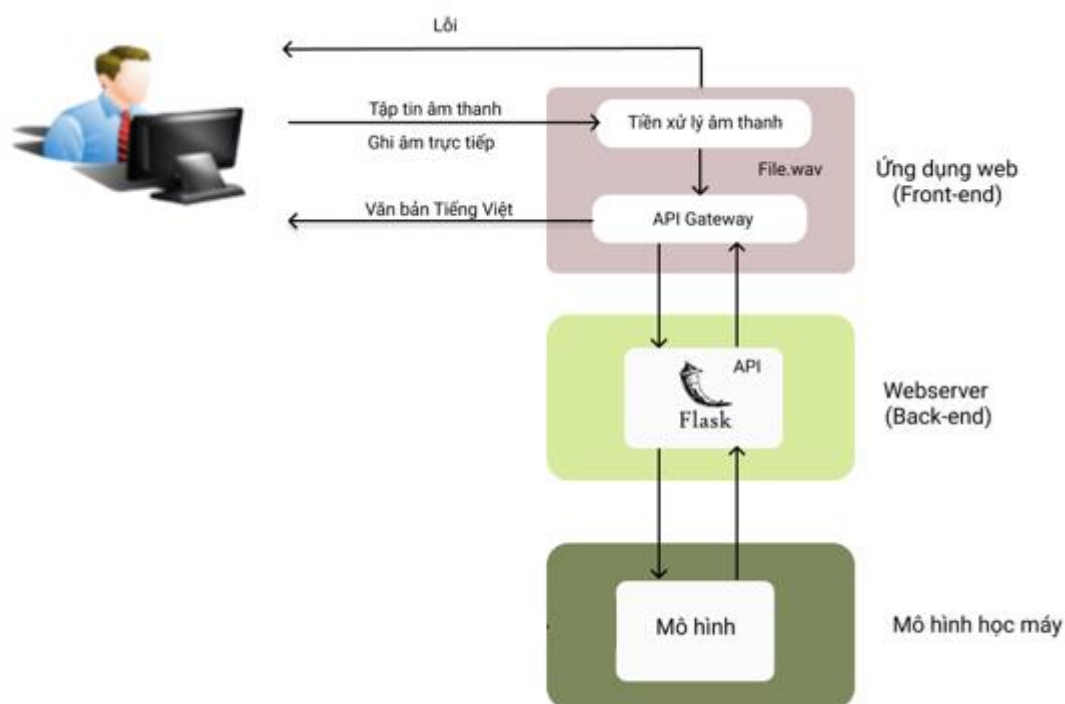
Hình 4.16. Minh họa một phần văn bản phục vụ cho mô hình ngôn ngữ.

Sau khi chuẩn hóa và chọn lọc các từ phù hợp với ngôn ngữ Tiếng Việt và Tiếng Anh, dữ liệu văn bản thu thập được khoảng 1,5 GB dữ liệu.

Bên cạnh đó, bộ dữ liệu VINBIGDATA nhóm sinh viên thu thập được có chứa một số mẫu âm thanh với các từ mượn Tiếng Anh thường dùng, điều này cũng một phần giúp nâng cao tỉ lệ nhận dạng các từ Tiếng Anh của mô hình.

4.5. GIẢI PHÁP XÂY DỰNG MÁY CHỦ

Nhóm sinh viên dự kiến xây dựng máy chủ với mục đích tạo ra giao diện lập trình ứng dụng (API), làm trung gian giữa ứng dụng mẫu và mô hình âm thanh đã được huấn luyện.



Hình 4.17 Kiến trúc chung xây dựng máy chủ

Máy chủ cho phép ứng dụng mẫu gửi đến một đoạn âm thanh dạng wav của tiếng Việt, sử dụng mô hình học máy xử lý đoạn âm thanh và trả về đoạn văn bản tiếng Việt cho ứng dụng mẫu.

Nhóm lựa chọn sử dụng Framework Flask của python để xây dựng máy chủ. Vì Flask dễ cài đặt và triển khai, phù hợp với ứng dụng web quy mô vừa và nhỏ, giúp tập trung phát triển mục tiêu của nhóm là xây dựng giao diện lập trình ứng dụng, dễ dàng triển khai trên các dịch vụ máy chủ đám mây.

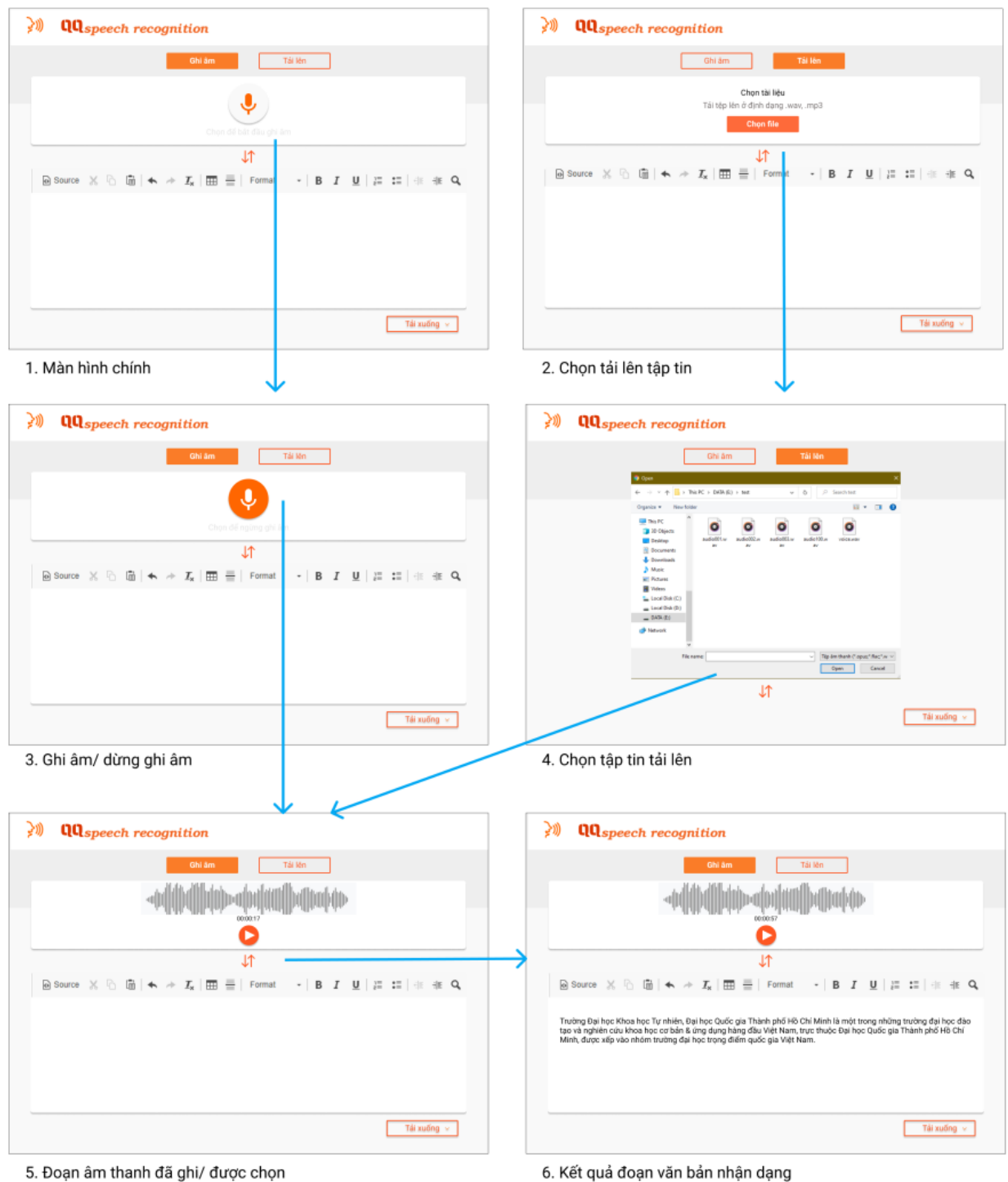
Sau khi xây dựng máy chủ, nhóm dự kiến triển khai máy chủ lên nền tảng đám mây Heroku.

4.6. GIẢI PHÁP XÂY DỰNG ỨNG DỤNG MẪU

Theo giải pháp về hệ thống, nhóm sinh viên xây dựng một ứng dụng mẫu với nhiệm vụ cho phép người dùng sử dụng dịch vụ chuyển đổi âm thanh tiếng Việt thành

văn bản tiếng Việt. Ứng dụng mẫu sử dụng giao diện lập trình ứng dụng (API) của máy chủ vừa được tạo ở trên.

Bản mẫu giao diện ứng dụng:



Hình 4.18 Bản mẫu xây dựng ứng dụng minh họa

4.7. TỔNG KẾT

Thông qua chương 4, nhóm sinh viên đã trình bày một hệ thống nhận dạng âm thanh tiếng Việt dựa trên mô hình nhận dạng từ đầu đến cuối sử dụng kỹ thuật phân loại thời gian kết nối CTC, hướng xây dựng máy chủ và cả ứng dụng trên nền tảng web. Chương kế tiếp sẽ trình bày về cài đặt và triển khai của mô hình và những cách khắc phục khó khăn cụ thể nếu có cho các giải pháp đã trình bày ở chương này.

CHƯƠNG 5. THỬ NGHIỆM VÀ ĐÁNH GIÁ

5.1. THU THẬP DỮ LIỆU HUẤN LUYỆN

5.1.1. Thu thập dữ liệu cho mô hình âm thanh

Để mô hình có độ chính xác cao, dữ liệu huấn luyện cần phải đủ nhiều và đủ lớn.

Qua quá trình thu thập dữ liệu, nhóm sinh viên đã sưu tầm được một lượng lớn tập dữ liệu và tất cả các bộ dữ liệu đều đã được đánh nhãn. Bên cạnh 50 giờ dữ liệu thô (bao gồm khoảng 15 giờ dữ liệu từ phòng thí nghiệm AILab của Trường Đại học Khoa Học Tự Nhiên và khoảng 35 giờ dữ liệu từ dự án mở của FPT) được kế thừa từ khóa luận trước đó, nhóm sinh viên cũng tiến hành thu thập thêm dữ liệu huấn luyện.

Bộ dữ liệu InfoRe Technology 1 do nhóm tự thu thập, với 25 giờ dữ liệu chứa khoảng 14000 tập tin từ một dự án mã nguồn mở trên diễn đàn github. Sau khi xử lý và chọn lọc, nhóm sinh viên nhận thấy bộ dữ liệu này bao gồm các tập tin âm thanh chưa được chọn lọc thật sự kĩ, phần lớn các tập tin âm thanh được cắt mất đoạn âm thanh cuối, tạo cảm giác bị hụt hẫng khi nghe ở phần cuối nội dung câu nói. Điều này ảnh hưởng đến độ tự nhiên của quá trình huấn luyện.

Bộ dữ liệu miễn phí VinBigdata-VLSP2020-100h được nhóm sinh viên tự tìm kiếm từ dự án của VinBigData với khoảng 100 giờ dữ liệu giọng nói. Đây là tập dữ liệu chứa các câu nói tự nhiên, có các câu nói ồn ào, với các tập tin có độ dài khác nhau, nhóm sinh viên chọn những tập tin với độ dài dưới 15 giây để phục vụ huấn luyện.

Qua quá trình tiền xử lý để sàng lọc dữ liệu với các công cụ chọn lọc do nhóm tự cài đặt, chi tiết các công cụ được nêu ở phần 5.8.2.2, nhóm đã chọn ra hai bộ dữ liệu có chất lượng và số lượng đủ tốt cho mô hình là (1) bộ dữ liệu FPT, VIVOS, (2) bộ dữ liệu VINBIGDATA.

5.1.1.1. FPT, VIVOS

Việc chọn lọc dữ liệu huấn luyện này một phần sẽ giảm bớt tình trạng gây nhiễu cho mô hình trong quá trình huấn luyện. Điều này dẫn đến việc chỉ còn khoảng hơn 32.95 giờ dữ liệu âm thanh tương ứng với 30757 mẫu để huấn luyện. Trong đó, dữ liệu được chia nhỏ thành 3 bộ train, dev, test với kích thước như sau:

- ❖ Bộ train: chứa 26701 mẫu ước tính tương đương khoảng 27.88 giờ dữ liệu (nhóm đã lược bỏ 210 mẫu không phù hợp so với dữ liệu gốc vì là các tập tin rỗng).
- ❖ Bộ dev: chứa khoảng 1697 mẫu ước tính tương đương khoảng 1.6 giờ dữ liệu (nhóm đã lược bỏ 2 mẫu không phù hợp).
- ❖ Bộ test: chứa 2359 mẫu ước tính tương đương khoảng 3.47 giờ dữ liệu (nhóm đã lược bỏ 2 mẫu không phù hợp).

5.1.1.2. VINBIGDATA

Bộ dữ liệu này chứa một số tập tin âm thanh với độ dài lên đến 1- 2 phút, qua quá trình lược bỏ chỉ còn khoảng 84.85 giờ dữ liệu tương đương với 53563 mẫu để huấn luyện. Trong đó, bộ dữ liệu được nhóm chia nhỏ thành 3 bộ train, dev, test (với tỉ lệ 8;1:1) với kích thước như sau:

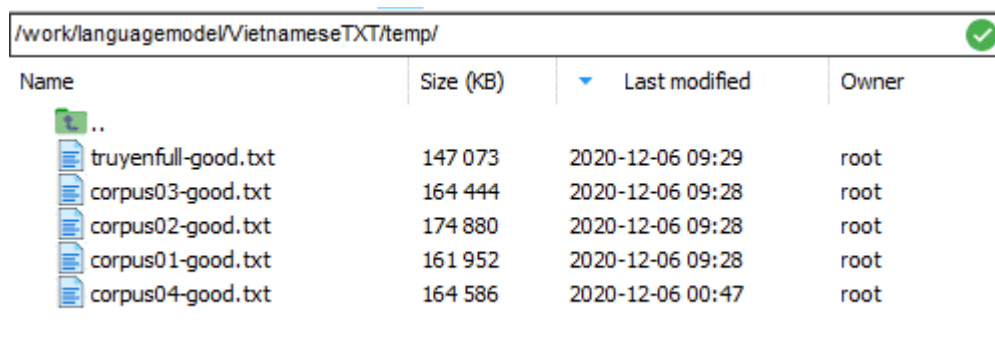
- ❖ Bộ train: chứa 42850 mẫu ước tính tương đương khoảng 67.79 giờ dữ liệu.
- ❖ Bộ dev: chứa khoảng 5356 mẫu ước tính tương đương khoảng 8.5 giờ dữ liệu.
- ❖ Bộ test: chứa 5357 mẫu ước tính tương đương khoảng 8.56 giờ dữ liệu.

Nhóm sinh viên sẽ sử dụng tập dữ liệu kết hợp của hai tập này để phục vụ cho việc huấn luyện và cải tiến mô hình.

5.1.2. Thu thập dữ liệu cho mô hình ngôn ngữ

Để tăng độ chính xác cho văn bản Tiếng Việt đầu ra về mặt ngữ nghĩa và chính tả, bộ dữ liệu cho mô hình ngôn ngữ. Nhóm sinh viên tự xây dựng mô hình ngôn ngữ phục vụ riêng cho mô hình nhận dạng. Dữ liệu huấn luyện mô hình ngôn ngữ mà nhóm sử dụng được nhóm thực hiện crawl từ các trang truyện Tiếng Việt như

gacsach.com, truyenfull.vn,.. và bộ dữ liệu văn bản Tiếng Việt của Corpus. Các bộ truyện, tác phẩm văn học ưu tiên có nội dung hiện đại, thuần Việt, không có teencode, hạn chế thấp nhất những từ viết tắt, từ mượn, ...



Name	Size (KB)	Last modified	Owner
..			
truyenfull-good.txt	147 073	2020-12-06 09:29	root
corpus03-good.txt	164 444	2020-12-06 09:28	root
corpus02-good.txt	174 880	2020-12-06 09:28	root
corpus01-good.txt	161 952	2020-12-06 09:28	root
corpus04-good.txt	164 586	2020-12-06 00:47	root

Hình 5.1 Minh họa một số tập tin văn bản phục vụ xây dựng mô hình ngôn ngữ

Sau khi thu thập, sưu tầm dữ liệu, nhóm viết công cụ xây dựng một bộ từ điển Tiếng Việt và một bộ lọc từ để chuẩn hóa văn bản, chuyển số thành chữ, loại bỏ các câu không hợp lệ (chứa từ viết tắt, không có trong từ điển Tiếng Việt,...).

Công cụ thu thập dữ liệu từ các trang báo, cũng như công cụ chuẩn hóa dữ liệu cho mô hình ngôn ngữ được nhóm trình bày chi tiết ở phần 5.8.2.1.

5.2. HUẤN LUYỆN MÔ HÌNH

Mã nguồn xây dựng mô hình huấn luyện của đề tài được phát triển dựa trên Python, sử dụng thư viện Pytorch để cài đặt.

5.2.1. Tiền xử lý dữ liệu

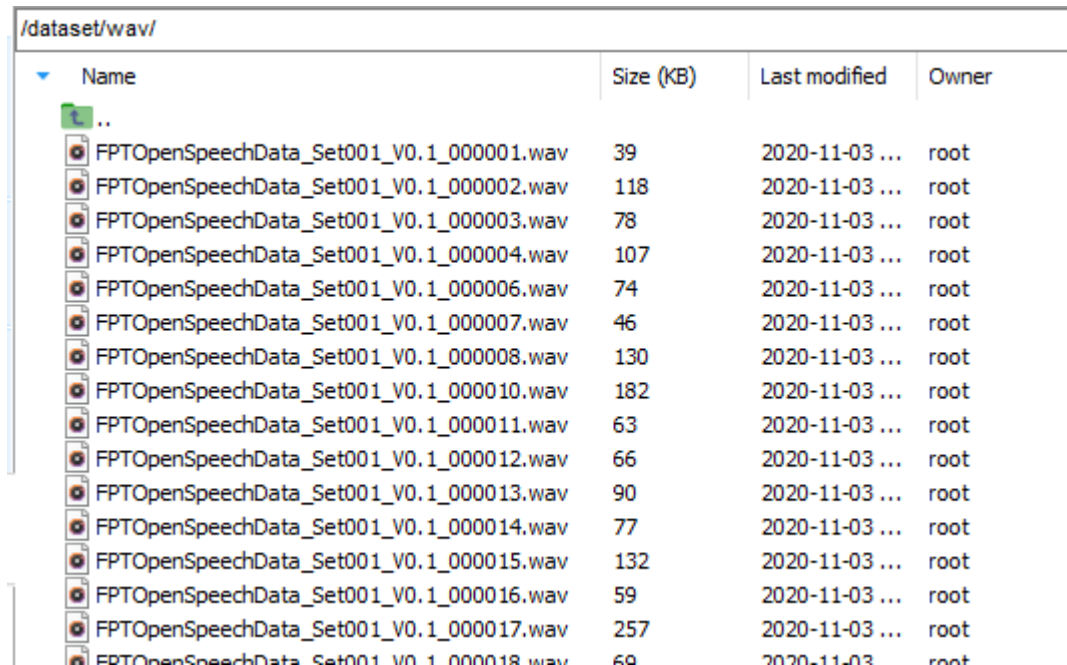
Để huấn luyện một mô hình nhận dạng âm thanh tiếng Việt dựa trên DeepSpeech2 thì bộ dữ liệu phải có ba thành phần chính bao gồm:

❖ Âm thanh

Để huấn luyện một hệ thống nhận dạng âm thanh tiếng Việt đủ tốt thì lượng dữ liệu âm thanh dùng để huấn luyện cũng phải đủ nhiều và đủ tốt. Ngoài ra dữ liệu âm thanh dùng để huấn luyện cũng cần thỏa mãn các điều kiện:

- Độ dài tối đa của mỗi tập tin âm thanh là 15 giây.

- Tập tin âm thanh cần chuyển về định dạng *.wav nhằm đồng nhất định dạng đầu vào và tăng chất lượng âm thanh.
- Giá trị số lần lấy mẫu trên 1 giây (sample rate) là 16000 Hz.



Name	Size (KB)	Last modified	Owner
..			
FPTOpenSpeechData_Set001_V0.1_000001.wav	39	2020-11-03 ...	root
FPTOpenSpeechData_Set001_V0.1_000002.wav	118	2020-11-03 ...	root
FPTOpenSpeechData_Set001_V0.1_000003.wav	78	2020-11-03 ...	root
FPTOpenSpeechData_Set001_V0.1_000004.wav	107	2020-11-03 ...	root
FPTOpenSpeechData_Set001_V0.1_000006.wav	74	2020-11-03 ...	root
FPTOpenSpeechData_Set001_V0.1_000007.wav	46	2020-11-03 ...	root
FPTOpenSpeechData_Set001_V0.1_000008.wav	130	2020-11-03 ...	root
FPTOpenSpeechData_Set001_V0.1_000010.wav	182	2020-11-03 ...	root
FPTOpenSpeechData_Set001_V0.1_000011.wav	63	2020-11-03 ...	root
FPTOpenSpeechData_Set001_V0.1_000012.wav	66	2020-11-03 ...	root
FPTOpenSpeechData_Set001_V0.1_000013.wav	90	2020-11-03 ...	root
FPTOpenSpeechData_Set001_V0.1_000014.wav	77	2020-11-03 ...	root
FPTOpenSpeechData_Set001_V0.1_000015.wav	132	2020-11-03 ...	root
FPTOpenSpeechData_Set001_V0.1_000016.wav	59	2020-11-03 ...	root
FPTOpenSpeechData_Set001_V0.1_000017.wav	257	2020-11-03 ...	root
FPTOpenSpeechData_Set001_V0.1_000018.wav	68	2020-11-03 ...	root

Hình 5.2 Minh hoạ cấu trúc thư mục chứa bộ dữ liệu âm thanh

Dữ liệu âm thanh sau khi được chọn lọc sẽ được tổng hợp trong 1 thư mục là wav.

❖ Văn bản

Là bản dịch (transcript) tương ứng với nội dung lời nói trong các tập tin âm thanh.

Dữ liệu âm thanh và văn bản sau khi đã được kiểm chứng và sãn lọc sẽ được tổng hợp trong một thư mục txt, thư mục này chứa các văn bản Tiếng Việt của từng âm thanh mẫu.

/dataset/txt/		
Name	Size (KB)	Last modified
..		
FPTOpenSpeechData_Set001_V0.1_000001.txt	1	2020-11-03
FPTOpenSpeechData_Set001_V0.1_000002.txt	1	2020-11-03
FPTOpenSpeechData_Set001_V0.1_000003.txt	1	2020-11-03
FPTOpenSpeechData_Set001_V0.1_000004.txt	1	2020-11-03
FPTOpenSpeechData_Set001_V0.1_000006.txt	1	2020-11-03
FPTOpenSpeechData_Set001_V0.1_000007.txt	1	2020-11-03
FPTOpenSpeechData_Set001_V0.1_000008.txt	1	2020-11-03
FPTOpenSpeechData_Set001_V0.1_000010.txt	1	2020-11-03
FPTOpenSpeechData_Set001_V0.1_000011.txt	1	2020-11-03
FPTOpenSpeechData_Set001_V0.1_000012.txt	1	2020-11-03
FPTOpenSpeechData_Set001_V0.1_000013.txt	1	2020-11-03
FPTOpenSpeechData_Set001_V0.1_000014.txt	1	2020-11-03
FPTOpenSpeechData_Set001_V0.1_000015.txt	1	2020-11-03
FPTOpenSpeechData_Set001_V0.1_000016.txt	1	2020-11-03
FPTOpenSpeechData_Set001_V0.1_000017.txt	1	2020-11-03

Hình 5.3 Minh họa cấu trúc thư mục chứa bộ dữ liệu văn bản dịch.

Nội dung của một tập tin văn bản mẫu:

```

VIVOSSPK46_291.txt X
C: > Users > QUANG > DOCUME~1 > MobaXterm > slash > RemoteFiles > 131350_10_7 > VIVOSSPK46_291.txt
1  hình ảnh anh cùng những tháng ngày hạnh phúc ủa về

```

Hình 5.4 Minh họa nội dung một tập văn bản dịch.

❖ Tập tin có định dạng *.csv

Tập tin này tổng hợp đường dẫn tập tin âm thanh mẫu và tập tin văn bản (bản dịch) tương ứng với nội dung được chia thành 3 cột theo thứ tự sau:

- Cột thứ nhất: Nội dung cột này chứa đường dẫn trực tiếp đến một tập tin âm thanh.
- Cột thứ hai: Nội dung cột này chứa đường dẫn trực tiếp đến một tập tin văn bản tương ứng của tập tin âm thanh.

1	/dataset/wav/FPTOpenSpeechData_Set001_V0.1_000001.wav	/dataset/txt/FPTOpenSpeechData_Set001_V0.1_000001.txt
2	/dataset/wav/FPTOpenSpeechData_Set001_V0.1_000002.wav	/dataset/txt/FPTOpenSpeechData_Set001_V0.1_000002.txt
3	/dataset/wav/FPTOpenSpeechData_Set001_V0.1_000003.wav	/dataset/txt/FPTOpenSpeechData_Set001_V0.1_000003.txt
4	/dataset/wav/FPTOpenSpeechData_Set001_V0.1_000004.wav	/dataset/txt/FPTOpenSpeechData_Set001_V0.1_000004.txt
5	/dataset/wav/FPTOpenSpeechData_Set001_V0.1_000006.wav	/dataset/txt/FPTOpenSpeechData_Set001_V0.1_000006.txt
6	/dataset/wav/FPTOpenSpeechData_Set001_V0.1_000007.wav	/dataset/txt/FPTOpenSpeechData_Set001_V0.1_000007.txt
7	/dataset/wav/FPTOpenSpeechData_Set001_V0.1_000008.wav	/dataset/txt/FPTOpenSpeechData_Set001_V0.1_000008.txt
8	/dataset/wav/FPTOpenSpeechData_Set001_V0.1_000010.wav	/dataset/txt/FPTOpenSpeechData_Set001_V0.1_000010.txt
9	/dataset/wav/FPTOpenSpeechData_Set001_V0.1_000011.wav	/dataset/txt/FPTOpenSpeechData_Set001_V0.1_000011.txt

Hình 5.5 Minh họa tập tin csv chứa dữ liệu huấn luyện.

Hình 5.5 minh họa tập tin chứa đường dẫn dữ liệu bản dịch văn bản tương ứng với mỗi đường dẫn tập tin âm thanh.

5.2.2. Sửa chữa, cải tiến mã nguồn, dữ liệu

5.2.2.1. Dữ liệu huấn luyện

Bộ dữ liệu huấn luyện FPT do nhóm thu thập có tồn tại một số tập tin âm thanh rỗng, các tập tin âm thanh rỗng này được biến đổi tần số và đưa vào chuẩn hóa. Độ lệch chuẩn của các vector âm thanh này là 0, dẫn đến kết quả chuẩn hóa bị gặp lỗi khi chia cho giá trị độ lệch chuẩn này. Nhóm sinh viên nhận biết được lỗi này và tiến hành chọn lọc lại dữ liệu đầu vào, nhóm sinh viên đã cài đặt công cụ để kiểm tra và loại bỏ các tập tin đầu vào không chứa nội dung (tập tin rỗng) để đảm bảo cho quá trình huấn luyện.

Hình bên dưới minh họa đoạn mã nguồn chuẩn hóa dữ liệu đầu vào, và tính độ lệch chuẩn của phổ âm thanh:

```

# STFT
D = librosa.stft(y, n_fft=n_fft, hop_length=hop_length,
                 win_length=win_length, window=self.window)#window='hamming'
spect, phase = librosa.magphase(D)
# S = log(S+1)
spect = np.log1p(spect)
spect = torch.FloatTensor(spect)
if self.normalize:
    mean = spect.mean()
    std = spect.std()
    spect.add_(-mean)
    spect.div_(std)

```

Hình 5.6 Đoạn mã nguồn gây lỗi khi tập tin đầu vào là đoạn âm thanh rỗng.

5.2.2.2. Mã nguồn

Mã nguồn mà nhóm tham khảo thực hiện nhiệm vụ nhận dạng giọng nói trên ngôn ngữ Tiếng Anh, nhóm sinh viên đã tiến hành cài đặt lại mã nguồn này đảm bảo nhiệm vụ nhận dạng trên ngôn ngữ Tiếng Việt.

❖ Tập tin labels.json

Bảng kí tự của ngôn ngữ Tiếng Việt khác với Tiếng Anh, nhóm sinh viên đã thu thập tổng hợp các kí tự trong bảng chữ cái Tiếng Việt để thay thế cho tập tin labels.json gốc phục vụ cho huấn luyện mô hình.

{...} labels.json > ...	19	"ế",
1	20	"ã",
2	21	"à",
3	22	"r",
4	23	"y",
5	24	"ũ",
6	25	"s",
7	26	"ó",
8	27	"ộ",
9	28	"ỹ",
10	29	"ô",
11	30	"ò",
12	31	"í",
13	32	"è",
14	33	"ĩ",
15	34	"ê",
16	35	"v",
17	36	"ạ",
18	37	"ã",

Hình 5.7 Minh họa bảng kí tự trong ngôn ngữ Tiếng Việt.

❖ **Chỉnh sửa nội dung tập tin cấu hình (train_config, inference_config)**

Các tham số phục vụ huấn luyện và đánh giá mô hình khác nhau giữa nhiệm vụ nhận dạng giọng nói trên ngôn ngữ Tiếng Việt và ngôn ngữ gốc, nên nhóm sinh viên đã sửa chữa lại mã nguồn để đem lại sự phù hợp hơn cho mô hình nhận dạng.

❖ **Chỉnh sửa đọc nội dung văn bản Tiếng Việt**

Để đọc nội dung văn bản Tiếng Việt, nhóm chỉnh sửa mã nguồn tại tập tin data_loader.py, thêm tham số encoding= “utf8” khi mở tập tin văn bản đầu vào.

```
def parse_transcript(self, transcript_path):
    with open(transcript_path, 'r', encoding='utf8') as transcript_file:
        transcript = transcript_file.read().replace('\n', '')
        transcript = list(filter(None, [self.labels_map.get(x) for x in list(transcript)]))
    return transcript
```

Hình 5.8 Đoạn mã nguồn chỉnh sửa khi đọc tập tin âm thanh đầu vào.

5.2.3. Cấu hình

Chi tiết kiến trúc mô hình đã được trình bày trong Chương 4, các tham số chi tiết của mô hình được nhóm trình bày trong phần này. Mô hình thiết lập số lượng đơn vị trạng thái ẩn (hidden sizes) của cả lớp hồi quy hai chiều là 1024. Với đầu vào trên bộ dữ liệu FPT, VIVOS cũng như VINBIGDATA, nhóm sinh viên tính toán và đặt độ dài tối đa cho đầu vào là 15 giây âm thanh Tiếng Việt.

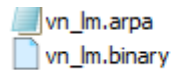
Ngoài ra, nhóm sinh viên cũng thiết lập số lượng các lớp RNN là 7, tỷ lệ học tập (learning rate) ban đầu là 0.0003 được lấy từ tham số mặc định của mô hình. Ngoài ra, chuẩn hóa hàng loạt cho RNN (Batch Normalization) được áp dụng trong quá trình huấn luyện để tối ưu hóa việc huấn luyện. Mô hình được xáo trộn dữ liệu huấn luyện ngẫu nhiên ở mỗi epoch để tăng tốc độ huấn luyện.

Cuối cùng, nhóm sinh viên sử dụng máy tính có cấu hình gồm 1 GPU Nvidia GTX RTX 2800 chạy trên hệ điều hành Ubuntu 16.04 để tiến hành huấn luyện và thử nghiệm mô hình. Việc huấn luyện mô hình trên bộ dữ liệu FPT VINBIGDATA diễn ra trong vòng 3-4 ngày với các thông số của mô hình được đề cập bên trên.

Sau quá trình huấn luyện, nhóm sinh viên sử dụng bộ giải mã tìm kiếm chùm kết hợp mô hình ngôn ngữ với alpha là 1, beta là 2 được sử dụng trong giai đoạn giải mã khi tạo văn bản kết quả để thu được kết quả mô hình tốt nhất. Chi tiết về cấu hình mô hình ngôn ngữ được nhóm sinh viên trình bày ở phần tiếp theo.

5.2.4. Xây dựng mô hình ngôn ngữ

Sau khi thu thập được dữ liệu văn bản cho mô hình ngôn ngữ và chuẩn hóa để phù hợp với bộ từ điển Tiếng Việt, nhóm sinh viên sử dụng Bộ công cụ mô hình ngôn ngữ KenLM để chuyển đổi về tập tin có định dạng *.arpa để phù hợp cho huấn luyện mô hình. Sau quá trình nghiên cứu và đánh giá, nhóm sinh viên nhận thấy tập tin ở định dạng *.arpa yêu cầu thời gian tải khá chậm nên nhóm sử dụng bộ công cụ mô hình ngôn ngữ KenLM để chuyển đổi tập tin về định dạng *.binary để giảm độ lớn của mô hình ngôn ngữ cũng như nâng cao tốc độ dự đoán kết quả cũng như đánh giá mô hình.



12 810 052
6 297 974

Hình 5.9 Minh họa tập tin mô hình ngôn ngữ ở hai định dạng.

5.2.5. Hậu xử lí văn bản kết quả

❖ Mô hình dấu câu

Như giải giáp của việc tích hợp mô hình dấu câu ở mục 4.4.1, dưới đây nhóm minh họa kết quả của quá trình thử nghiệm tích hợp mô hình dấu câu và mô hình nhận dạng giọng nói để cho kết quả văn bản tự nhiên hơn.

Minh họa một tập tin âm thanh với câu nói : “anh có thể gọi tôi không”, kết quả của mô hình DeepSpeech2 cho ra văn bản Tiếng Việt như sau:

```
Loading model cost : 4.437466859817505
Audio transcribe cost : 1.4598979949951172

Output transcript :  anh có thể gọi tôi không
```

Hình 5.10 Kết quả đầu ra khi chưa tích hợp mô hình dấu câu.

Kết quả sau khi tích hợp mô hình dấu câu:

```
Load models...
Comma transcribe cost : 0.27153444290161133

Output transcript :  Anh có thể gọi tôi không?
```

Hình 5.11 Kết quả đầu ra khi đã tích hợp mô hình dấu câu.

❖ Nhận dạng từ Tiếng Anh thông dụng

Mục 4.4.2 đã giới thiệu về giải pháp giúp mô hình có thể nhận dạng được từ Tiếng Anh thông dụng nâng cao độ tự nhiên cho mô hình, trong phần này sẽ minh họa kết quả qua quá trình nhóm sinh viên nghiên cứu và thử nghiệm.

Kết quả mô hình âm thanh nhận dạng trước đó với câu nói: “Tuổi trẻ on lai tiếp tục mời bạn đọc theo dõi các ý kiến sau”:

```
Load models...
Comma transcribe cost : 0.21509075164794922

Output transcript : Tuổi trẻ on lai tiếp tục mời bạn đọc theo dõi các ý kiến sau:
```

Hình 5.12 Kết quả nhận dạng câu nói có chứa từ Tiếng Anh trước khi cải tiến

Minh họa một tập tin âm thanh với câu nói: “Tuổi trẻ on lai tiếp tục mời bạn đọc theo dõi các ý kiến sau”, kết quả của mô hình nhận dạng sau khi cải tiến mô hình ngôn ngữ:

```
Load models...
Comma transcribe cost : 0.25981593132019043

Output transcript : Tuổi trẻ online tiếp tục mời bạn đọc theo dõi các ý kiến sau:
```

Hình 5.13 Kết quả nhận dạng từ Tiếng Anh ssau khi cải tiến mô hình ngôn ngữ

5.3. CẢI TIẾN ĐỘ CHÍNH XÁC MÔ HÌNH

Để tìm ra những giá trị tối ưu nhất có ảnh hưởng nhiều đến độ chính xác của mô hình và đạt được độ chính xác cao nhất cho mô hình, nhóm sinh viên tiến hành cải tiến công việc huấn luyện mô hình dựa vào quá trình điều chỉnh các siêu tham số (hyperparameters), thu thập thêm dữ liệu huấn luyện.

5.3.1. Thay đổi tham số huấn luyện

Quá trình huấn luyện được nhóm triển khai trên máy tính gồm 1 GPU NVidia GTX RTX 2800 chạy trên hệ điều hành Ubuntu 16.04. Sau một khoảng thời gian huấn luyện và ghi nhận kết quả, nhóm sinh viên đã thống kê được những siêu tham số có ảnh hưởng khá nhiều đến kết quả dự đoán của mô hình bao gồm các siêu tham số sau:

- epochs: một epoch là một lần duyệt qua hết tất cả số lượng các mẫu trong tập huấn luyện.

- `batch_size`: số lượng mẫu được sử dụng cho một lần cập nhật trọng số khi huấn luyện và khi kiểm tra mô hình.
- `hidden_size`: độ rộng của một lớp ẩn được dùng khi khởi tạo các lớp trong mô hình.
- `hidden_layers`: số lớp ẩn được dùng khi khởi tạo mạng hồi quy RNN
- `rnn_type`: loại mạng được sử dụng, mô hình nhóm sử dụng có hai loại là lstm và gru.

Trong quá trình sử dụng mô hình ngôn ngữ với bộ giải mã tìm kiếm chùm, các siêu tham số sau cũng ảnh hưởng đến kết quả dự đoán của mô hình:

- `alpha, beta`: các tham số của bộ giải mã tìm kiếm chùm
- `beam_width`: độ rộng của bộ giải mã tìm kiếm chùm.

5.3.1.1. Điều chỉnh epoch

❖ Chọn epoch = 75

Tên tham số	Giá trị	Ghi chú
epochs	75	
batch_size	32	Giá trị mặc định
hidden_size	1024	Giá trị mặc định
hidden_layers	7	Giá trị mặc định
rnn_type	lstm	Giá trị mặc định
Alpha	2	Giá trị mặc định
Beta	-0.2	Giá trị mặc định
Beam_width	2048	Giá trị mặc định

WER	39.592
------------	---------------

Bảng 5.1 Điều chỉnh tham số epoch = 75

❖ Chọn epoch = 70

Tên tham số	Giá trị	Ghi chú
epochs	70	
batch_size	32	Giá trị mặc định
hidden_size	1024	Giá trị mặc định
hidden_layers	7	Giá trị mặc định
rnn_type	lstm	Giá trị mặc định
Alpha	2	Giá trị mặc định
Beta	-0.2	Giá trị mặc định
Beam_width	2048	Giá trị mặc định
WER	38.277	

Bảng 5.2 Điều chỉnh tham số epoch = 70

❖ Chọn epoch = 50

Tên tham số	Giá trị	Ghi chú
epochs	50	
batch_size	32	Giá trị mặc định
hidden_size	1024	Giá trị mặc định

hidden_layers	7	Giá trị mặc định
rnn_type	lstm	Giá trị mặc định
Alpha	2	Giá trị mặc định
Beta	-0.2	Giá trị mặc định
Beam_width	2048	Giá trị mặc định
WER	36.639	

Bảng 5.3 Điều chỉnh tham số epoch = 50

Kết luận: chọn epoch = 50 cho các lần huấn luyện sau.

5.3.1.2. Điều chỉnh tham số của tìm kiếm tham lam

❖ Chọn alpha=2, beta=-0.2, beam_width =1024

Tên tham số	Giá trị	Ghi chú
epochs	50	Giá trị được chọn
batch_size	32	Giá trị mặc định
hidden_size	1024	Giá trị mặc định
hidden_layers	7	Giá trị mặc định
rnn_type	lstm	Giá trị mặc định
Bộ dữ liệu	FPT, VIVOS	Giá trị mặc định
Alpha	2	
Beta	1	

Beam_width	1024	
WER	33.918	

Bảng 5.4 Điều chỉnh tham số $\alpha=2$, $\beta=-0.2$, beam_width =1024

❖ Chọn $\alpha=2$, $\beta=1$, beam_width =1024

Tên tham số	Giá trị	Ghi chú
epochs	50	Giá trị được chọn
batch_size	32	Giá trị mặc định
hidden_size	1024	Giá trị mặc định
hidden_layers	7	Giá trị mặc định
rnn_type	lstm	Giá trị mặc định
Alpha	2	
Beta	1	
Beam_width	1024	
WER	32.604	

Bảng 5.5 Điều chỉnh tham số $\alpha=2$, $\beta=1$, beam_width =1024

❖ Chọn $\alpha=2$, $\beta=-0.2$, beam_width = 2048

Tên tham số	Giá trị	Ghi chú
epochs	50	Giá trị được chọn
batch_size	32	Giá trị mặc định

hidden_size	1024	Giá trị mặc định
hidden_layers	7	Giá trị mặc định
rnn_type	lstm	Giá trị mặc định
Alpha	2	
Beta	-0.2	
Beam_width	2048	
WER	36.639	

Bảng 5.6 Điều chỉnh tham số alpha=2, beta=-0.2, beam_width = 2048

Kết luận: chọn alpha=2, beta=1, beam_width = 1024 cho các lần huấn luyện sau.

5.3.1.3. Điều chỉnh tham số batch_size

❖ Chọn batch_size = 16

Tên tham số	Giá trị	Ghi chú
epochs	50	Giá trị được chọn
batch_size	16	
hidden_size	1024	Giá trị mặc định
hidden_layers	7	Giá trị mặc định
rnn_type	lstm	Giá trị mặc định
Alpha	2	Giá trị được chọn
Beta	1	Giá trị được chọn

Beam_width	1024	Giá trị được chọn
WER	35.938	

Bảng 5.7 Điều chỉnh tham số batch_size = 16

❖ Chọn batch_size = 32

Tên tham số	Giá trị	Ghi chú
epochs	50	Giá trị được chọn
batch_size	32	
hidden_size	1024	Giá trị mặc định
hidden_layers	7	Giá trị mặc định
rnn_type	lstm	Giá trị mặc định
Alpha	2	Giá trị được chọn
Beta	1	Giá trị được chọn
Beam_width	1024	Giá trị được chọn
WER	32.604	

Bảng 5.8 Điều chỉnh tham số batch_size = 32

❖ Chọn batch_size = 25

Tên tham số	Giá trị	Ghi chú
epochs	50	Giá trị được chọn
batch_size	25	

hidden_size	1024	Giá trị mặc định
hidden_layers	7	Giá trị mặc định
rnn_type	lstm	Giá trị mặc định
Alpha	2	Giá trị được chọn
Beta	1	Giá trị được chọn
Beam_width	1024	Giá trị được chọn
WER	34.208	

Bảng 5.9 Điều chỉnh tham số batch_size = 25

Kết luận: chọn batch_size=32 cho các lần huấn luyện sau.

5.3.1.4. Điều chỉnh tham số rnn_type

❖ Chọn rnn_type = lstm

Tên tham số	Giá trị	Ghi chú
epochs	50	Giá trị được chọn
batch_size	32	Giá trị được chọn
hidden_size	1024	Giá trị mặc định
hidden_layers	7	Giá trị mặc định
rnn_type	lstm	
Alpha	2	Giá trị được chọn
Beta	1	Giá trị được chọn

Beam_width	1024	Giá trị được chọn
WER	32.604	

Bảng 5.10 Điều chỉnh tham số rnn_type = lstm

❖ Chọn rnn_type = gru

Tên tham số	Giá trị	Ghi chú
epochs	50	Giá trị được chọn
batch_size	32	Giá trị được chọn
hidden_size	1024	Giá trị mặc định
hidden_layers	7	Giá trị mặc định
rnn_type	gru	
Alpha	2	Giá trị được chọn
Beta	1	Giá trị được chọn
Beam_width	1024	Giá trị được chọn
WER	27.344	

Bảng 5.11 Điều chỉnh tham số rnn_type = gru

❖ Chọn rnn_type = rnn

Tên tham số	Giá trị	Ghi chú
epochs	50	Giá trị được chọn
batch_size	32	Giá trị được chọn

hidden_size	1024	Giá trị mặc định
hidden_layers	7	Giá trị mặc định
rnn_type	rnn	
Alpha	2	Giá trị được chọn
Beta	1	Giá trị được chọn
Beam_width	1024	Giá trị được chọn
WER	33.440	

Bảng 5.12 Điều chỉnh tham số rnn_type = rnn

Kết luận: chọn rnn_type = gru cho các lần huấn luyện sau.

5.3.1.5. Điều chỉnh tham số hidden layer

❖ Chọn hidden layer = 5

Tên tham số	Giá trị	Ghi chú
epochs	50	Giá trị được chọn
batch_size	32	Giá trị được chọn
hidden_size	1024	Giá trị mặc định
hidden_layers	7	
rnn_type	lstm	Giá trị được chọn
Alpha	2	Giá trị được chọn
Beta	1	Giá trị được chọn

Beam_width	1024	Giá trị được chọn
WER	27.787	

Bảng 5.13 Điều chỉnh tham số hidden layer = 5

❖ Chọn hidden layer = 7

Tên tham số	Giá trị	Ghi chú
epochs	50	Giá trị được chọn
batch_size	32	Giá trị được chọn
hidden_size	1024	Giá trị mặc định
hidden_layers	5	
rnn_type	lstm	Giá trị được chọn
Alpha	2	Giá trị được chọn
Beta	1	Giá trị được chọn
Beam_width	1024	Giá trị được chọn
WER	25.347	

Bảng 5.14 Điều chỉnh tham số hidden layer = 7

Kết luận: chọn hidden layer = 7 cho các lần huấn luyện sau.

5.3.1.6. Điều chỉnh tham số hidden size

❖ Chọn hidden size = 512

Tên tham số	Giá trị	Ghi chú
-------------	---------	---------

epochs	50	Giá trị được chọn
batch_size	32	Giá trị được chọn
hidden_size	512	
hidden_layers	7	Giá trị được chọn
rnn_type	lstm	Giá trị được chọn
Alpha	2	Giá trị được chọn
Beta	1	Giá trị được chọn
Beam_width	1024	Giá trị được chọn
WER	26.350	

Bảng 5.15 Điều chỉnh tham số hidden size = 512

❖ Chọn hidden size = 1024

Tên tham số	Giá trị	Ghi chú
epochs	50	Giá trị được chọn
batch_size	32	Giá trị được chọn
hidden_size	1024	
hidden_layers	7	Giá trị được chọn
rnn_type	lstm	Giá trị được chọn
Alpha	2	Giá trị được chọn
Beta	1	Giá trị được chọn

Beam_width	1024	Giá trị được chọn
WER	25.787	

Bảng 5.16 Điều chỉnh tham số hidden size = 1024

❖ Chọn hidden size = 1600

Tên tham số	Giá trị	Ghi chú
epochs	50	Giá trị được chọn
batch_size	32	Giá trị được chọn
hidden_size	1600	
hidden_layers	7	Giá trị được chọn
rnn_type	lstm	Giá trị được chọn
Alpha	2	Giá trị được chọn
Beta	1	Giá trị được chọn
Beam_width	1024	Giá trị được chọn
WER	22.732	

Bảng 5.17 Điều chỉnh tham số hidden size = 1600

Kết luận: chọn hidden_size = 1600 cho các lần huấn luyện sau.

Tổng kết: các tham số epoch=50, alpha=2, beta=1, beam=1024, batch_size=32, rnn_type=gru, hidden_layer=7, hidden_size=1600 cho kết quả tốt nhất cho huấn luyện và đánh giá.

5.3.2. Thu thập thêm dữ liệu huấn luyện

Sau khi triển khai website mẫu, nhóm sử dụng chính trang website mẫu tích hợp với API sử dụng mô hình nhận dạng giọng nói để thu thập dữ liệu mà người sử dụng website đóng góp. Nhóm đăng ký tài khoản của API của FPT để hỗ trợ việc thu thập, giúp tăng tính đúng đắn cho văn bản được nhận dạng. Chính những dữ liệu thu thập này được tiến hành để tiếp tục huấn luyện cho mô hình. Tuy nhiên dữ liệu này đòi hỏi quá trình chọn lọc và yêu cầu nhiều về thời gian.

5.4. CÁC MÔ HÌNH SO SÁNH

Việc huấn luyện và thử nghiệm trên các mô hình cơ sở tạo tiền đề cho việc đánh giá các mô hình của nhóm sinh viên. Từ đó triển khai các ý tưởng cải tiến và nâng cấp sẽ tiết kiệm thời gian. Nhóm sinh viên đã tiến hành triển khai mô hình so sánh là: (1) mô hình nhận dạng với cơ chế chú ý, (2) Mô hình nhận dạng đầu cuối DeepSpeech 0.4 của khóa luận 2019. Kết quả thử nghiệm của các mô hình trên tập dữ liệu FPT, VIVOS được thể hiện dựa trên tỉ lệ lỗi từ:

Mô hình dựa trên sự chú ý	Mô hình DeepSpeech 0.4
27.953	25.84

Bảng 5.18 So sánh kết quả tỉ lệ lỗi từ trên tập dữ liệu FPT, VIVOS của mô hình dựa trên sự chú ý và mô hình DeepSpeech 0.4.

Mô hình (1) là một mô hình bao gồm bộ mã hoá và bộ giải mã kết hợp với cơ chế chú ý. Việc áp dụng thêm cơ chế chú ý được nêu ở chương 3 là điểm nhấn của mô hình. Cơ chế chú ý là một kỹ thuật phổ biến với việc cải thiện điểm số của các mô hình. Sau quá trình nghiên cứu và kiểm nghiệm, nhóm sinh viên nhận thấy một số điểm hạn chế của mô hình (1) như: đầu vào là đoạn âm thanh lặp lại, đầu vào quá dài.

Mặc dù mô hình cơ sở (2) có thể giải quyết các vấn đề trên song vẫn tồn đọng một số điểm hạn chế như văn bản kết quả sai ngữ pháp, chưa nhận dạng được các từ Tiếng Anh thông dụng, chưa nhận dạng dấu câu. Vì vậy, nhóm sinh viên đã tìm hiểu,

nghiên cứu để cải tiến, nâng cấp mô hình của mình bằng cách áp dụng phương pháp chuẩn hoá mô hình ngôn ngữ, tích hợp mô hình dấu câu vào mô hình của nhóm. Mặc dù số điểm không chênh lệch bao nhiêu, nhưng kết quả thu được có tính dễ đọc và tính đúng đắn về mặt ngữ pháp trên cùng tập kiểm tra (test dataset).

5.5. KẾT QUẢ THỰC NGHIỆM

Sau quá trình huấn luyện và điều chỉnh tham số mô hình, nhóm sinh viên đưa ra mô hình cuối cùng với kết quả được trình bày trong bảng. Nhóm sinh viên so sánh kết quả trên tập dữ liệu FPT, VIVOS cho cả 3 mô hình so sánh và tiến hành chỉnh tham số trên mô hình của nhóm với bộ dữ liệu tổng hợp FPT, VIVOS, VINBIGDATA. Đối với tập dữ liệu FPT, VIVO, nhóm sinh viên nhận thấy sự vượt trội của mô hình của nhóm sinh viên so với các mô hình so sánh trên 2360 tập dữ liệu kiểm tra (test dataset).

Bộ dữ liệu	Mô hình nhóm sinh viên	Mô hình dựa trên sự chú ý	Mô hình Deepspeech 0.4 (Kết quả khóa luận 2019)
FPT và VIVOS	25.081	27.593	25.84

Bảng 5.19 Kết quả tỉ lệ lỗi từ các mô hình nhận dạng giọng nói Tiếng.

Qua quá trình huấn luyện và đánh giá kết quả, nhóm sinh viên nhận thấy mô hình mạng thần kinh dựa trên kỹ thuật phân loại thời gian kết nối cho kết quả cao và ổn định hơn mô hình dựa trên sự chú ý. Mô hình dựa trên sự chú ý thường gặp thách thức về độ chính xác đối với âm thanh đầu vào dài và ồn ào.

Về mô hình của nhóm sinh viên, khi thực hiện huấn luyện và đánh giá trên tập dữ liệu FPT, VIVOS kết hợp với VINBIGDATA nhóm sinh viên nhận thấy kết quả huấn luyện được cải thiện đáng kể khi sử dụng mạng GRU, cụ thể tỉ lệ lỗi từ đạt được

của cả mô hình là 22.732 so với mô hình sử dụng mạng RNN. Các mạng GRU hoạt động tốt hơn các mạng RNN tuy nhiên, khi mở rộng kích thước mô hình, các mạng RNN đơn giản hoạt động tốt hơn một chút.

Mô hình của nhóm hoạt động tốt nhất có 10 lớp với 2 lớp tích chập hai chiều, 7 lớp GRU, một lớp đầu ra được kết nối đầy đủ cùng với chuẩn hóa hàng loạt. Đồng thời việc đánh giá mô hình kết hợp với mô hình ngôn ngữ giúp cải thiện đáng kể độ chính xác của mô hình, bảng 6.1.

Đặc biệt hơn, việc tích hợp xử lý văn bản kết quả, tích hợp vào mô hình dấu câu, cải tiến mô hình ngôn ngữ giúp nhận dạng các từ mượn Tiếng Anh thông dụng là một trong những điểm khác biệt so với nghiên cứu trong khóa luận năm 2019, đồng thời giúp văn bản đầu ra tự nhiên hơn.

Trên đây là phân tích dựa trên sự chủ quan của nhóm qua quá trình nghiên cứu và thử nghiệm về mặt mô hình mà nhóm sinh viên mang lại. Ngoài việc giải quyết được vấn đề nhận dạng giọng nói Tiếng Việt và được đánh giá cao hơn so với các mô hình so sánh, mô hình của nhóm sinh viên còn đem lại kết quả tự nhiên hơn thông qua nhận dạng từ Tiếng Anh và việc chuẩn hóa dấu câu của văn bản kết quả. Tuy nhiên, mô hình của nhóm sinh viên vẫn còn những hạn chế như hiệu suất kém khi nhận dạng âm thanh trong môi trường ồn ào, vẫn còn một số lỗi ngữ pháp. Do đó, để có thêm thời gian nghiên cứu và phát triển, mô hình của nhóm sinh viên có thể mang lại một kết quả tốt hơn.

5.6. ĐÓNG GÓI MÔ HÌNH

Để xuất mô hình sau khi hoàn tất quá trình huấn luyện, ta cần gán đường dẫn sẽ lưu mô hình vào cờ “*best_val_model_name*” khi chạy DeepSpeech2. DeepSpeech2 sẽ xuất mô hình đã được huấn luyện dưới dạng một tập tin có định dạng *.pth. Tập tin này có thể hiểu là mô hình được đóng gói từ đồ thị (graph) của pytorch với các tham số được chọn lọc trong quá trình huấn luyện.

5.7. CÔNG CỤ XỬ LÝ

Trong quá trình thực hiện luận văn, nhóm đã xây dựng một số công cụ đặc thù để giải quyết các vấn đề về dữ liệu văn bản, âm thanh. Mã nguồn các công cụ được nhóm sinh viên cài đặt và chi tiết hướng dẫn được lưu trữ tại nguồn github của nhóm.

5.7.1.1. Công cụ phục vụ xây dựng dữ liệu mô hình ngôn ngữ

Nhằm tăng độ chính xác cho mô hình được sử dụng, nhóm sinh viên tự xây dựng mô hình ngôn ngữ phục vụ riêng cho mô hình nhận dạng. Dữ liệu cho mô hình ngôn ngữ được nhóm sinh viên chọn lọc và lấy từ các trang báo điện tử, các trang truyện ngắn và chọn lọc lại các từ thích hợp trong từ điển do nhóm tự xây dựng trước đó. Đây là một số công cụ mà nhóm cài đặt để phục vụ các chức năng trên:

❖ Công cụ thu thập dữ liệu

Một số nguồn dữ liệu nhóm tham khảo được như gacsach.com, dantri.vn, truyenfull.vn với nội dung văn bản là các bài báo tiếng Việt, các tiểu thuyết, truyện tiếng Việt,... Nội dung đa dạng, ngữ pháp tương đối chuẩn thích hợp cho việc huấn luyện. Các công cụ nhóm xây dựng có chức năng thu thập văn bản ở các trang trên, tiền xử lý một số vấn đề đơn giản như xuống dòng, dấu cách,... sau đó lưu lại dưới dạng tập tin văn bản.

- Thu thập dữ liệu truyenfull.vn: Công cụ là một tập tin python với đầu vào cho công cụ là đường dẫn một thể loại, ở đây nhóm chỉ thu thập văn bản của thể loại Truyện Việt Nam. Kết quả thu được của nhóm là nội dung của hơn 70 quyển truyện, mỗi quyển có nhiều chương, mỗi chương thu được một tập tin văn bản. Thời gian thu thập là khoảng 1-2 ngày.
- Thu thập dữ liệu gacsach.com: nhóm cải tiến công cụ thu thập truyenfull.vn. Kết quả thu được là nội dung văn bản của hơn 150 quyển truyện với thời gian thu thập khoảng 3-4 ngày.
- Thu thập dữ liệu dantri.vn: Công cụ là một tập tin python với đầu vào là quãng thời gian cần thu thập dữ liệu, ở đây nhóm tiến hành thu thập từ ngày 01 tháng

01 năm 2014 đến ngày 24 tháng 01 năm 2021. Kết quả thu được là hơn 330 nghìn bài báo với dung lượng văn bản là 2.2GB với thời gian thu thập là 6-7 ngày.

❖ Công cụ chuẩn hóa nội dung dữ liệu

Từ nội dung văn bản thuần được lấy từ các trang báo, truyện,... công cụ này có chức năng chuẩn hóa, lọc dấu câu, phân tách dòng, chuyển số thành chữ, định dạng ngày, tháng, năm. Kết quả thu được là một tập tin văn bản với nội dung mang tính chính xác hơn.

❖ Công cụ chọn lọc từ điển

Từ dữ liệu văn bản đã được chuẩn hóa bằng công cụ chuẩn hóa bên trên, công cụ này có nhiệm vụ lọc ra chỉ những dòng nội dung có toàn bộ các từ nằm trong bộ từ điển mà nhóm cung cấp. Nội dung của tập tin văn bản thu được có tính đúng đắn cao nhất.

Bộ từ điển Tiếng Việt do nhóm tự sưu tầm và sànlọc, gồm 7747 từ và kí tự hợp lệ trong Tiếng Việt:

```
final_dictionary_02.txt X
C: > Users > PC > DOCUMEN~1 > MobaXt
82  bực
83  bặc
84  bấc
85  bậc
86  bạc
87  bách
88  bạch
89  bai
90  báí
91  bài
92  bấí
93  bảí
94  bại
95  bam
96  bám
97  bàm
98  bẵm
99  bẳm
100 bẳm
101 bẳm
102 bặm
103 bẩm
104 bầm
105 bẳm
106 bẩm
107 bậm
108 bặm
109 ban
```

Hình 5.14 Minh họa một số từ trong bộ từ điển Tiếng Việt nhóm tự xây dựng

Với mục tiêu ban đầu là nhận dạng âm thanh Tiếng Việt nên dữ liệu cho mô hình ngôn ngữ được lọc bỏ các câu có chứa từ Tiếng Anh thông dụng đi kèm và chọn lọc những câu đúng với bộ từ điển Tiếng Việt mà nhóm sinh viên đã xây dựng. Sau khi đạt được độ chính xác ổn định khi nhận dạng âm thanh Tiếng Việt, nhóm sinh viên tiến hành cải tiến mô hình ngôn ngữ giúp mô hình có thể nhận dạng được các câu với từ mượn thông dụng bằng một bộ từ điển Tiếng Anh có hơn 10 nghìn từ.

Dữ liệu cho mô hình ngôn ngữ cải tiến này được nhóm sinh viên thu thập từ trang báo điện tử Dân trí với đa dạng các lĩnh vực, và nội dung. Minh họa một số câu trong bộ dữ liệu của mô hình ngôn ngữ: “Với việc ra mắt bản xem trước *Android 12* dành cho nhà phát triển trong tuần này”, “Nghe nhạc, xem phim, chơi *game*”, “Va chạm giữa các xe *container*”,...

5.7.1.2. Công cụ phục vụ tiền xử lí âm thanh

❖ Công cụ chuẩn hóa tần số lấy mẫu 16000Hz (`sample_rate`)

Trong quá trình thu thập dữ liệu âm thanh, một số mẫu âm thanh có tần số mẫu lớn, cho kết quả không chính xác khi huấn luyện. Do đó, nhóm cài đặt một công cụ python để lọc và chuẩn hóa các tập tin âm thanh có tần số lấy mẫu lớn hơn 16000Hz trở thành 16000Hz.

❖ Công cụ kiểm tra các tập tin đầu vào rỗng

Một số tập tin âm thanh có độ lớn nhưng không có nội dung làm quá trình huấn luyện không thành công, đồng thời khiến mô hình cho kết quả sai. Một công cụ python được nhóm cài đặt giúp lọc ra các tập tin có nội dung, loại bỏ các tập tin rỗng đã giải quyết vấn đề này.

❖ Công cụ chuyển đổi tập tin âm thanh stereo thành mono

Mô hình huấn luyện của nhóm sinh viên yêu cầu các tập tin âm thanh đầu vào thuộc loại mono, những âm thanh stereo sẽ được mô hình nhận dạng và cho ra kết quả bị lỗi. Hiểu được vấn đề đó, nhóm sinh viên cài đặt công cụ này để chọn lọc âm thanh giúp đồng bộ cho quá trình huấn luyện.

5.8. LỖI HUẤN LUYỆN MÔ HÌNH

Một số lỗi trong quá trình huấn luyện mô hình được nhóm sinh viên tổng hợp lại.

❖ Lỗi Unicode

```
root@3884699c7340:/content/deepspeech.pytorch# python train.py data.train_manifest=/dataset/vi_train.csv data.val_manifest=/dataset/vi_test.csv data.batch_size=32 training.epochs=70 checkpointing.checkpoint=true checkpointing.save_n_recent_models=3 checkpointing.load_auto_checkpoint=true data.num_workers=0
/usr/local/lib/python3.6/dist-packages/omegaconf/omegaconf.py:645: UserWarning: update() merge flag is is not specified, defaulting to False.
For more details, see https://github.com/omry/omegaconf/issues/367
  stacklevel=1,
Traceback (most recent call last):
  File "train.py", line 24, in <module>
    hydra_main()
  File "/usr/local/lib/python3.6/dist-packages/hydra/main.py", line 37, in decorated_main
    strict=strict,
  File "/usr/local/lib/python3.6/dist-packages/hydra/_internal/utils.py", line 253, in run_hydra
    lambda: hydra.run(
  File "/usr/local/lib/python3.6/dist-packages/hydra/_internal/utils.py", line 185, in run_and_report
    func()
  File "/usr/local/lib/python3.6/dist-packages/hydra/_internal/utils.py", line 256, in <lambda>
    overrides=args.overrides,
  File "/usr/local/lib/python3.6/dist-packages/hydra/_internal/hydra.py", line 114, in run
    job_subdir_key=None,
  File "/usr/local/lib/python3.6/dist-packages/hydra/core/utils.py", line 107, in run_job
    ret.return_value = task_function(task_cfg)
  File "train.py", line 20, in hydra_main
    train(cfg=cfg)
  File "/content/deepspeech.pytorch/deepspeech_pytorch/training.py", line 105, in train
    labels = json.load(label_file)
  File "/usr/lib/python3.6/json/__init__.py", line 296, in load
    return loads(fp.read(),
  File "/usr/lib/python3.6/encodings/ascii.py", line 26, in decode
    return codecs.ascii_decode(input, self.errors)[0]
UnicodeDecodeError: 'ascii' codec can't decode byte 0xe1 in position 12: ordinal not in range(128)
root@3884699c7340:/content/deepspeech.pytorch#
```

Hình 5.15 Lỗi hệ thống khi huấn luyện mô hình (1)

Nguyên nhân của lỗi này là do hệ điều hành của máy trường không nhận diện được nội dung Unicode khi hiện thị lên màn hình.

Để khắc phục lỗi này nhóm đã tiến hành cấu hình lại máy trường cho phép sử dụng bộ mã hóa UTF-8, với các dòng lệnh thực hiện như sau:

```
export LC_ALL="en_US.UTF-8"
export LC_CTYPE="en_US.UTF-8"
sudo dpkg-reconfigure locales
```

❖ Cảnh báo không tương thích giữa Geforce RTX và Pytorch

```
The current PyTorch install supports CUDA capabilities sm_37 sm_50 sm_60 sm_70.
If you want to use the GeForce RTX 2080 Ti GPU with PyTorch, please check the instructions at https://pytorch.org/get-started/locally/

warnings.warn(incompatible_device_warn.format(device_name, capability, " ".join(arch_list), device_name))
{"output": [{"transcription": "AND HE YHUNGN EN GIT EP CUNIN BORNE'S AR OM HIS IS AN SONTYOUF FACKOULYX CYTDIN DUM THORT OR SHE TANTHOUGHT"}], "meta": {"acoust
model": {"path": "/content/deepspeech.pytorch/models/deepspeech_checkpoint_epoch_10.pth"}, "language_model": {"path": ""}, "decoder": {"alpha": 0.0, "beta": 0.0
type": "greedy"}}}
/content/lib/python3.6/site-packages/omegaconf/basecontainer.py:244: UserWarning: cfg.pretty() is deprecated and will be removed in a future version.
Use OmegaConf.to_yaml(cfg)
```

Hình 5.16 Lỗi hệ thống khi huấn luyện mô hình (2)

Nguyên nhân là do máy không tương thích giữa phiên bản Cuda và phiên bản Pytorch.

Để khắc phục lỗi này nhóm đã tiến hành cài đặt phiên bản tương thích cho Cuda, với các dòng lệnh:

```
pip install torch==1.7.0+cu101 torchvision==0.8.1+cu101 torchaudio==0.7.0 -f .
```

❖ Lỗi List index out of range

```
Epoch: [1][802/5017]    Time 0.956 (1.164)    Data 0.150 (0.180)    Loss 114.9492 (169.5042)
Traceback (most recent call last):
  File "train.py", line 25, in <module>
    hydra_main()
  File "/usr/local/lib/python3.6/dist-packages/hydra/main.py", line 37, in decorated_main
    strict=strict,
  File "/usr/local/lib/python3.6/dist-packages/hydra/_internal/utils.py", line 253, in run_hydra
    lambda: hydra.run(
  File "/usr/local/lib/python3.6/dist-packages/hydra/_internal/utils.py", line 185, in run_and_report
    func()
  File "/usr/local/lib/python3.6/dist-packages/hydra/_internal/utils.py", line 256, in <lambda>
    overrides=args.overrides,
  File "/usr/local/lib/python3.6/dist-packages/hydra/_internal/hydra.py", line 114, in run
    job_subdir_key=None,
  File "/usr/local/lib/python3.6/dist-packages/hydra/core/utils.py", line 107, in run_job
    ret.return_value = task_function(task_cfg)
  File "train.py", line 20, in hydra_main
    train(cfg=cfg)
  File "/work/Source/deepspeech.pytorch/deepspeech_pytorch/training.py", line 209, in train
    for i, (data) in enumerate(train_loader, start=state.training_step):
  File "/usr/local/lib/python3.6/dist-packages/torch/utils/data/dataloader.py", line 435, in __next__
    data = self._next_data()
  File "/usr/local/lib/python3.6/dist-packages/torch/utils/data/dataloader.py", line 475, in _next_data
    data = self._dataset_fetcher.fetch(index) # may raise StopIteration
  File "/usr/local/lib/python3.6/dist-packages/torch/utils/data/_utils/fetch.py", line 44, in fetch
    data = [self.dataset[idx] for idx in possibly_batched_index]
  File "/usr/local/lib/python3.6/dist-packages/torch/utils/data/_utils/fetch.py", line 44, in <listcomp>
    data = [self.dataset[idx] for idx in possibly_batched_index]
  File "/work/Source/deepspeech.pytorch/deepspeech_pytorch/loader/data_loader.py", line 232, in __getitem__
    audio_path, transcript_path = sample[0], sample[1]
IndexError: list index out of range
root@2267c61c746e:/work/Source/deepspeech.pytorch#
```

Hình 5.17 Lỗi nội dung tập tin csv

Nguyên nhân của lỗi này là do khi đọc nội dung trong tập tin *.csv, nội dung từng dòng của tập tin *.csv sẽ được lưu vào biến sample với sample[0] là nội dung cột thứ nhất của dòng đó, sample[1] là nội dung cột thứ hai của dòng, thông báo lỗi List index out of range là do trong tập tin *.csv có một dòng không đúng định dạng(dòng bị không có nội dung).

Nhóm sinh viên đã tìm giải pháp cho lỗi này là tìm và sửa lại, hoặc xóa các dòng bị lỗi trong tập tin *.csv.

5.9. XÂY DỰNG MÁY CHỦ

Flask Framework nền tảng được nhóm sinh viên chọn để xây dựng hệ thống máy chủ API làm cầu nối để người dùng sử dụng mô hình nhận dạng âm thanh. Với các yếu tố như dễ triển khai, tốc độ triển khai nhanh gọn, sự tiện ích và tính thông dụng nên việc chọn nền tảng này để xây dựng máy chủ là quyết định phù hợp với nhu cầu đặt ra của nhóm sinh viên.

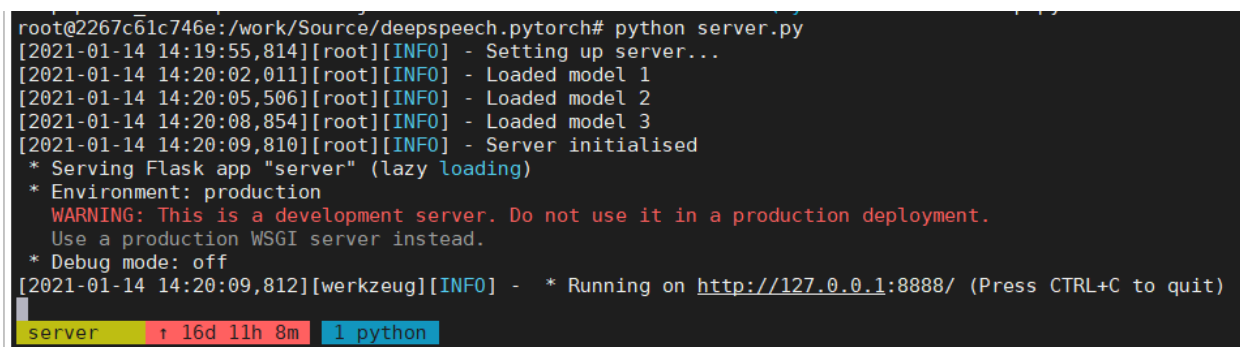
Hệ thống máy chủ trong giới hạn luận văn này sẽ cung cấp ra bên ngoài duy nhất một giao diện lập trình ứng dụng (Application Programming Interface-API) để chuyển đổi từ âm thanh tiếng nói nhận được và trả về dữ liệu văn bản tiếng Việt tương ứng. Tuy nhiên khi tiến hành xây dựng API nhóm sinh viên gặp hai vấn đề cần giải quyết:

- ❖ Đảm bảo dữ liệu được bảo mật.
- ❖ Kích thước dữ liệu âm thanh gửi đi có thể lớn.

Để giải quyết được cả hai vấn đề trên, nhóm sinh viên quyết định dùng phương thức đăng (POST) cho API nhóm xây dựng.

Để đưa API từ máy chủ cục bộ ra bên ngoài môi trường Internet, nhóm sinh viên sử dụng công cụ đường hầm ngrok để chạy máy chủ trên terminal và "bộ ghép kênh" Terminal Multiplexer (Tmux) để giữ trạng thái hoạt động của terminal đó.

- ❖ Chạy máy chủ bằng terminal



```
root@2267c61c746e:/work/Source/deepspeech.pytorch# python server.py
[2021-01-14 14:19:55,814][root][INFO] - Setting up server...
[2021-01-14 14:20:02,011][root][INFO] - Loaded model 1
[2021-01-14 14:20:05,506][root][INFO] - Loaded model 2
[2021-01-14 14:20:08,854][root][INFO] - Loaded model 3
[2021-01-14 14:20:09,810][root][INFO] - Server initialised
* Serving Flask app "server" (lazy loading)
* Environment: production
  WARNING: This is a development server. Do not use it in a production deployment.
  Use a production WSGI server instead.
* Debug mode: off
[2021-01-14 14:20:09,812][werkzeug][INFO] - * Running on http://127.0.0.1:8888/ (Press CTRL+C to quit)
```

Hình 5.18 Xây dựng máy chủ với terminal

- ❖ Sử dụng ngrok cho phép truy cập đến cục bộ từ Internet

```
ngrok by @inconshreveable

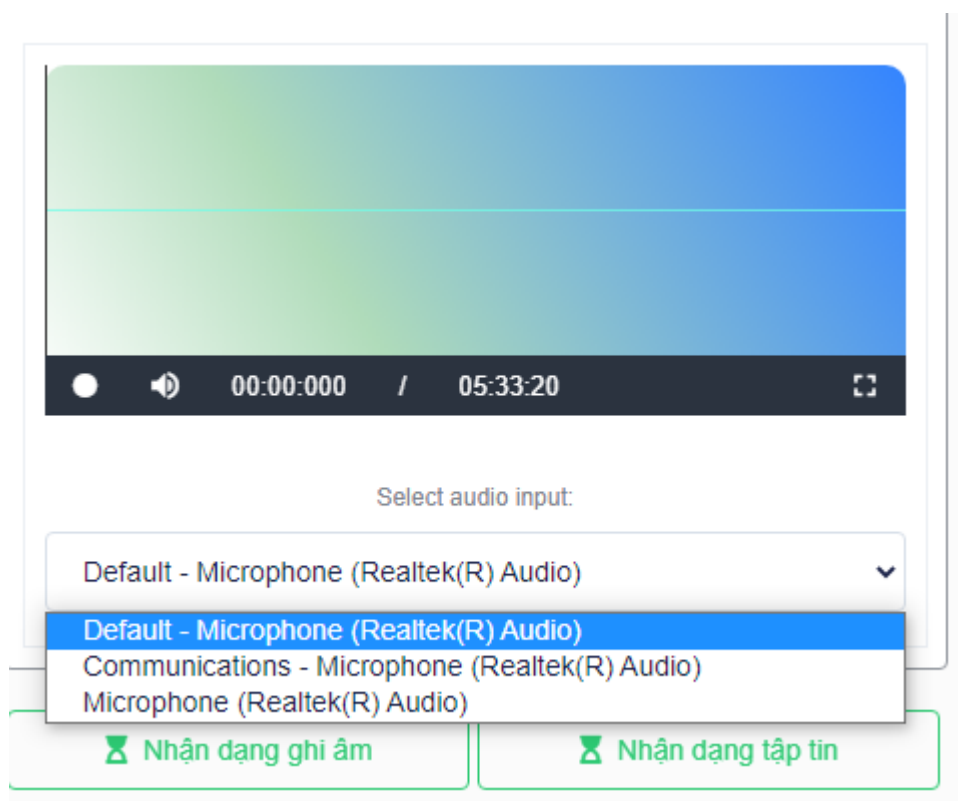
Session Status      online
Account             hocmai6@gmail.com (Plan: Free)
Version             2.3.35
Region              United States (us)
Web Interface        http://127.0.0.1:4040
Forwarding           http://d995bb2bd6f3.ngrok.io -> http://localhost:8888
Forwarding           https://d995bb2bd6f3.ngrok.io -> http://localhost:8888

Connections          ttl      opn      rt1      rt5      p50      p90
                    0        0        0.00     0.00     0.00     0.00
```

Hình 5.19 Minh họa kết quả khởi chạy, thông tin truy cập ngrok

5.10. CÀI ĐẶT CHỨC NĂNG GHI ÂM TRÊN WEBSITE

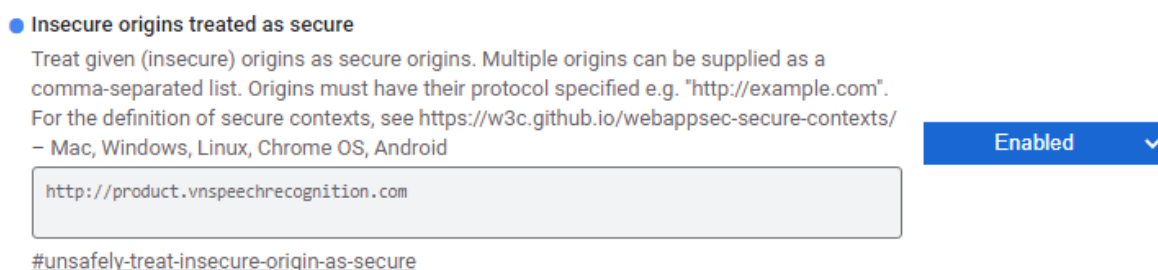
Một trong những vấn đề quan trọng cho ứng dụng website mẫu là khả năng sử dụng microphone ghi âm giọng nói. Để giải quyết vấn đề này, nhóm sử dụng thư viện RecordRTC.js để cho phép người dùng sử dụng WebRTC để ghi âm lại âm thanh giọng nói bằng các cổng giao tiếp có sẵn trên thiết bị.



Hình 5.20 Giao diện ứng dụng mẫu với chức năng ghi âm.

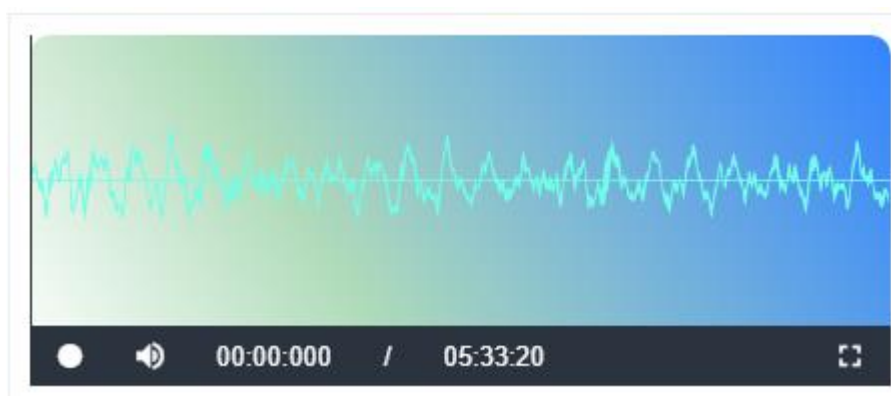
Khi sử dụng thư viện RecordRTC để ghi âm trên website, nhóm rút ra được một số lưu ý sau:

- ❖ Để có thể sử dụng micro của thiết bị và tạo bản ghi âm, người sử dụng phải cấp phép quyền truy cập đến micro cho trang web.
- ❖ Nếu trang web không được cấp chứng chỉ SSL, trang web sẽ không được phép truy cập quyền sử dụng microphone. Do đó, người sử dụng cần phải cấu hình trình duyệt để có thể sử dụng WebRTC ở `chrome://flags/#unsafely-treat-insecure-origin-as-secure`.



Hình 5.21 Cho phép truy cập bảo mật khi sử dụng ứng mẫu trên trình duyệt.

- ❖ Đầu ra sau khi kết thúc việc ghi âm là một url blob đại diện cho dữ liệu có phần mở rộng .webm.

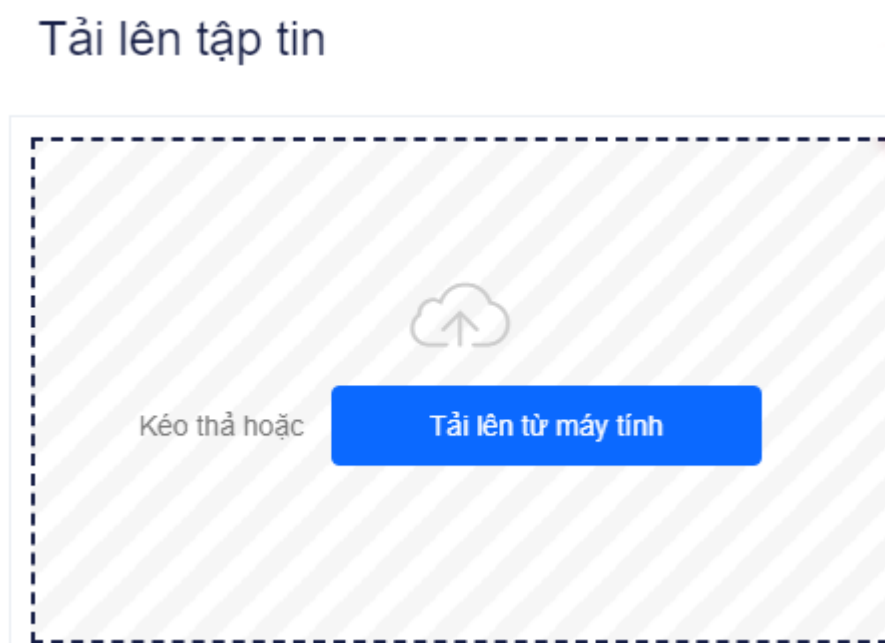


Hình 5.22 Minh họa kết quả đoạn ghi âm trên ứng dụng mẫu

Kết hợp với việc sử dụng RecordRTC để sử dụng microphone để ghi âm, nhóm sử dụng thư viện Wavesurfer để trực quan hóa quá trình ghi âm bằng biểu đồ dạng sóng.

5.11. CÀI ĐẶT CHỨC NĂNG TẢI LÊN TẬP TIN ÂM THANH TRÊN ỨNG DỤNG

Ngoài việc ghi âm giọng nói trực tiếp trên web bằng microphone, nhóm xây dựng chức năng cho phép người dùng tải lên một tập tin âm thanh từ thiết bị. Để cài đặt chức năng này, nhóm sử dụng thư viện Dropify.js.



Hình 5.23 Giao diện ứng dụng mẫu với chức năng tải lên tập tin âm thanh.

Cấu hình các phần mở rộng được phép tải lên: `allowedFileExtensions: ["wav", "mp3", "webm"]`.

5.12. TỔNG KẾT

Trong chương 5, nhóm sinh viên đã trình bày về cách thức cài đặt và triển khai cho các thành phần bao gồm mô hình nhận dạng âm thanh, hệ thống máy chủ api và các chức năng chính của website mẫu. Nội dung chi tiết cho một số phần cài đặt được nhóm sinh viên trình bày chi tiết ở phần tài liệu đính kèm, trong chương 6 tiếp theo, sẽ là tổng kết và đánh giá kết quả của toàn bộ quá trình nhóm sinh viên thực hiện luận văn.

CHƯƠNG 6. TỔNG KẾT VÀ ĐÁNH GIÁ

6.1. KIẾN THỨC ĐẠT ĐƯỢC

Trong thời gian thực hiện khóa luận, nhóm sinh viên đã tích lũy được nhiều điều, kiến thức mới:

- ❖ Được học và áp dụng phương pháp làm việc của một số quy trình phát triển như Kanban, thác nước (waterfall), ...
- ❖ Tiếp thu được kiến thức tổng quan về nhận dạng giọng nói và các kỹ thuật trong mạng nơ-ron để nhận dạng tiếng nói.
- ❖ Học hỏi được các kiến thức về khoa học máy tính, học sâu (deep learning), quá trình huấn luyện một mô hình hoàn chỉnh.
- ❖ Tiếp thu các kiến thức về nhận dạng giọng nói theo hướng đầu cuối với hai kỹ thuật phân loại thời gian kết nối và dựa trên sự chú ý.
- ❖ Lĩnh hội được kiến thức và kinh nghiệm thiết kế, xây dựng, triển khai và kiểm thử khi phát triển một ứng dụng web của hệ thống nhận dạng âm thanh Tiếng Việt.
- ❖ Tiếp thu được kiến trúc và luồng xử lý của hệ nhận dạng giọng nói tiên tiến hướng đầu cuối nói chung và dự án DeepSpeech2 nói riêng, tạo nền tảng cho việc tìm hiểu và phát triển mô hình nhận dạng giọng nói tiếng Việt.
- ❖ Học hỏi được cách hệ thống hóa kiến thức để áp dụng vào việc viết luận văn.
- ❖ Khả năng tìm kiếm và đọc hiểu tài liệu, sách báo được nâng cao. Khả năng tổng hợp các kiến thức từ các tài liệu cũng được cải thiện đáng kể.
- ❖ Lĩnh hội được khả năng chọn lọc và tích hợp các mô hình để đạt được độ chính xác cao nhất.
- ❖ Nâng cao được khả năng lên kế hoạch và đảm bảo hoàn thành trong khoảng thời gian cho trước.
- ❖ Hình thành kỹ năng tự chủ, tự tập và tinh thần trách nhiệm công việc.

- ❖ Học hỏi được các kiến thức về chọn lọc dữ liệu đầu vào và khả năng tìm kiếm, thu thập dữ liệu từ các nguồn khác nhau.
- ❖ Kinh nghiệm đọc hiểu, hiệu chỉnh mã nguồn được phát triển. Kinh nghiệm chỉnh sửa và khả năng khắc phục khi gặp lỗi dần được cải thiện.

6.2. KẾT QUẢ MÔ HÌNH HUẤN LUYỆN

Dữ liệu huấn luyện là tổng hợp của cả 2 tập dữ liệu đã được nhóm sinh viên chọn lọc và trình bày trong phần 5.1.

- ❖ Dữ liệu huấn luyện: 69551 mẫu (~96 giờ)
- ❖ Dữ liệu kiểm tra: 7053 mẫu (~12 giờ)
- ❖ Mô hình ngôn ngữ: N-gram (N = 5)

Kiến trúc	Mô hình ngôn ngữ	Không có mô hình ngôn ngữ
10 lớp - 7 lớp GRU	22.732	49.611

Bảng 6.1 Kết quả huấn luyện mô hình về tỉ lệ lỗi từ có và không có sử dụng mô hình ngôn ngữ.

Mô hình được huấn luyện và đánh giá cho kết quả tỉ lệ sai từ (WER) cuối cùng là 22.732%.

6.3. KẾT QUẢ HỆ THỐNG

6.3.1. Môi trường phát triển

- ❖ Hệ điều hành, Ubuntu 16.04 LTS.
- ❖ Công cụ xây dựng hệ thống: Visual Code, MobaXTerm
- ❖ Công cụ kiểm thử API: Postman
- ❖ Thư viện và nền tảng (framework) sử dụng:
- ❖ Python Flask (API)

- ❖ NodeJS Express (Website Client)

6.3.2. Môi trường triển khai

- ❖ Nền tảng đám mây Heroku.
- ❖ Công cụ tạo tunnel ngrok cho phép truy cập API từ máy chủ cục bộ.

6.3.3. Chức năng đã cài đặt

- ❖ Mô hình nhận dạng giọng nói cho phép nhận vào một đoạn âm thanh, chuyển đổi thành nội dung văn bản Tiếng Việt.
- ❖ Ứng dụng web cho phép người dùng sử dụng API trên giao diện web.

6.4. KẾT QUẢ ỨNG DỤNG

6.4.1. Môi trường phát triển

- ❖ Hệ điều hành: Window 10, Ubuntu 16.0 LTS
- ❖ Hệ quản trị cơ sở dữ liệu: MySQL
- ❖ Công cụ xây dựng ứng dụng: Visual Studio Code
- ❖ Thư viện và nền tảng (framework) sử dụng:
- ❖ Express NodeJS: Xây dựng xây dựng web theo mô hình MVC dựa trên mã nguồn NodeJS.
- ❖ SocketIO: Hỗ trợ tạo bộ đếm realtime khi người dùng sử dụng.
- ❖ RecordRTC: cho phép sử dụng microphone để ghi âm trên nền tảng web

6.4.2. Môi trường triển khai

- ❖ Máy chủ đám mây Heroku

6.4.3. Chức năng đã cài đặt

- ❖ Sử dụng microphone của thiết bị để ghi âm.
- ❖ Tải lên tập tin âm thanh với các định dạng (wav, mp3, mpeg,...).
- ❖ Chuyển đổi tập tin âm thanh thành văn bản Tiếng Việt.
- ❖ So sánh, đối chiếu văn bản đầu ra với văn bản cho trước.

6.5. SO SÁNH CÁC KẾT QUẢ THU ĐƯỢC VỚI MỤC TIÊU BAN ĐẦU

Mục tiêu ban đầu	Nhận xét mức độ hoàn thành
Trình bày động lực xây dựng mô hình nhận dạng âm thanh Tiếng Việt	Đã trình bày động lực ở chương 2 của luận văn.
Bản luận văn trình bày lý thuyết nền tảng và giải pháp để xử lý việc nhận 1 tập tin âm thanh tiếng Việt và xuất ra nội dung văn bản ở dạng tiếng Việt.	Đã trình bày ở chương 3 của luận văn.
Xây dựng, thu thập dữ liệu, và đào tạo mô hình để nhận 1 file âm thanh tiếng Việt và xuất ra nội dung ở dạng văn bản.	Đã được trình bày ở chương 4 và chương 5.
Cải tiến độ chính xác của mô hình với mục tiêu là 75%.	Đã được trình bày ở chương 5.
Xây dựng ứng dụng web chuyển đổi từ giọng nói sang văn bản Tiếng Việt. Ứng dụng mẫu áp dụng mô hình được xây dựng là ứng dụng web dựa trên nền tảng Flask. Ứng dụng cho phép định dạng, chỉnh sửa văn bản trực tuyến (căn lề, kích thước chữ, phông chữ, định dạng chữ, ...), bên cạnh đó ứng dụng cho phép người dùng tải lên tập tin âm thanh ít nhất 3 định dạng (.mp3, .acc, .wav).	Đã được trình bày ở chương 4 và chương 5.

Ứng dụng cũng hỗ trợ ghi âm, nghe lại đoạn âm thanh đã ghi âm trước khi thực hiện chuyển đổi và cho phép tải xuống tập tin văn bản.	
Viết 120 trang luận văn theo đúng chuẩn yêu cầu và trích dẫn các tài liệu tham khảo đầy đủ	Luận văn được hoàn thành chi tiết và chính xác.

Bảng 6.2 Bảng so sánh kết quả thực hiện đề tài với mục tiêu ban đầu

6.6. ĐỊNH HƯỚNG PHÁT TRIỂN VÀ NGHIÊN CỨU TRONG TƯƠNG LAI

Về chất lượng mô hình, trong tương lai, nhóm định hướng thu thập thêm dữ liệu giọng đọc và văn bản, cải thiện độ chính xác cho mô hình âm thanh, mô hình ngôn ngữ, mô hình dấu câu. Phân tích cụ thể hơn dữ liệu âm thanh có tiếng ồn, qua đó nhận dạng âm thanh chính xác hơn. Huấn luyện và tích hợp thêm mô hình nhận dạng tên riêng để có thể nhận dạng một số tên riêng, nâng cao độ tự nhiên và chính xác cho văn bản đầu ra. Song, nhóm cũng sẽ tiếp tục cải thiện độ tự nhiên của mô hình thông qua tích hợp nhận dạng các từ viết tắt, nâng cao độ chính xác cho văn bản đầu ra. Thời gian dự đoán một đoạn âm thanh đầu vào dài và cho ra kết quả cũng được nhóm sinh viên xem xét cải thiện, việc tích hợp mô hình dấu câu và mô hình ngôn ngữ làm tăng độ lớn cho mô hình nên tốc độ nhận dạng có thể bị ảnh hưởng.

Bên cạnh đó, việc thực hiện nghiên cứu hệ thống nhận dạng tiếng nói phát âm liên tục, tức thời để nâng cao tính ứng dụng trong lĩnh vực điều khiển thiết bị bằng giọng nói, nhận dạng giọng nói theo thời gian thực, phương pháp chống nhiễu môi trường là các hướng triển khai cần xem xét trong tương lai.

Về công nghệ, để ổn định hệ thống, nhóm dự kiến triển khai các hệ thống từ máy chủ cục bộ, máy chủ cloud Heroku sang Google Cloud VPS để có chất lượng tốt hơn.

Về ứng dụng mô hình, nhóm dự kiến cải thiện website mẫu, thêm các chức năng mới để có thể trở thành website cung cấp dịch vụ nhận dạng giọng nói. Song, nhóm sinh viên sẽ tiếp tục cải thiện tốc độ xử lý các tác vụ của ứng dụng, giúp ứng dụng chạy mượt mà và tạo trải nghiệm tốt hơn cho người dùng.

Ngoài ra, nhóm sinh viên sẽ tiến hành trình bày, chỉnh sửa mã nguồn theo khuôn mẫu để dễ dàng bảo trì và chỉnh sửa trong tương lai.

6.7. LỜI KẾT

Kết quả của quá trình học tập, nghiên cứu, làm việc nghiêm túc của nhóm sinh viên được kết tinh trong luận văn xây dựng mô hình nhận dạng âm thanh Tiếng Việt này. Quá trình học hỏi, nghiên cứu và thực hiện luận văn giúp nhóm sinh viên đã tích lũy cho mình nhiều kinh nghiệm lẫn kiến thức vô cùng quý báu trong việc xây dựng một mô hình lẫn việc thực hiện một đề tài nghiên cứu. Bên cạnh đó, cả một quá trình thực hiện luận văn cũng phần nào tôi luyện cho bản thân nhóm sinh viên rất nhiều kiến thức nền tảng, kinh nghiệm, khả năng giải quyết vấn đề và trải nghiệm quá trình hoàn thành một sản phẩm lâu dài một cách có bài bản.

Mặc dù kết quả của mô hình còn một vài điểm hạn chế và ứng dụng mẫu khá đơn giản, song sản phẩm hệ thống nhận dạng tiếng Việt đã mở ra cho bản thân nhóm sinh viên nhiều phương hướng phát triển và mở rộng trong tương lai. Ở nước ta, nhận dạng tiếng nói vẫn là một lĩnh vực khá mới mẻ và đang được đầu tư phát triển vì những trải nghiệm tuyệt vời mà chúng đem lại cho người dùng cùng với sự nổi lên của hệ thống học sâu nói chung và nhận dạng tiếng nói đặc biệt là tiếng Việt nói riêng hứa hẹn một sự phát triển vượt bậc trong lĩnh vực nhận dạng giọng nói. Thông qua một quá trình học tập và nghiên cứu lâu dài cho luận văn, nhóm sinh viên sẽ tiếp tục xây dựng, cải tiến mô hình và phát triển hoàn thiện hệ thống để phục vụ nhiều đối tượng người dùng với mong muốn nhận được nhiều đóng góp và phản hồi để tốt hơn. Vốn kiến thức dồi dào đã được đúc kết trong suốt quá trình nhóm sinh viên học tập và nghiên cứu cùng với quá trình tích lũy sau này chắc chắn sẽ là một hành trang vững chắc tiếp bước cho nhóm sinh viên trong con đường sự nghiệp trong tương lai.

TÀI LIỆU THAM KHẢO

- [1] V. H. Nguyen, “*An end-to-end model for vietnamese speech recognition,*” 2019.
- [2] A.Graves and N. Jaitly, “*Towards end-to-end speech recognition with recurrent neural networks. proceedings of the 31st international conference on international conference on machine learning,*” vol. 32, 2014.
- [3] Dong Wang , Xiaodong Wang , Shaohe Lv, “*An Overview of End-to-End Automatic Speech Recognition*”, 2019.
- [4] Suyoun Kim, Takaaki Hori, Shinji Watanabe, “*Joint ctc-attention based end-to-end speech recognition*”, 2016.
- [5] A.Graves, S. Fernandez, F. Gomez, J. Schmidhuber. “*Connectionist temporal classification: Labelling unsegmented sequence data with recurrent neural networks*”. In ICML, pp. 369-376. ACM, 2006.
- [6] Do Van Hai, “*Acoustic modeling for speech recognition under limited training data conditions*” PhD Thesis, Nanyang Technological University, 2015.
- [7] Alex Graves and Navdeep Jaitly, “*Towards end-to-end speech recognition with recurrent neural networks,*” in Proceedings of the 31st International Conference on Machine Learning (ICML-14), 2014.
- [8] Awni Hannun, Carl Case, Jared Casper, Bryan Catanzaro, Greg Diamos, Erich Elsen, Ryan Prenger, Sanjeev Satheesh, Shubho Sengupta, Adam Coates, et al., “*Deep speech: Scaling up endto-end speech recognition*”, 2014.
- [9] Jan Chorowski, Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio, “*End-to-end continuous speech recognition using attention-based recurrent nn: First results*”, 2014.
- [10] Jan K Chorowski, Dzmitry Bahdanau, Dmitriy Serdyuk, Kyunghyun Cho, and Yoshua Bengio, “*Attention-based models for speech recognition in Advances in Neural Information Processing Systems*”, 2015, pp. 577–585.

- [11] A.Graves, A.-r. Mohamed, and G. Hinton, “*Speech recognition with deep recurrent neural networks*”, 2013.
- [12] William Chan and Ian Lane, “*On online attention-based speech recognition and joint mandarin character-pinyin training*” Interspeech 2016, pp. 3404–3408, 2016.
- [13] H. H. Sak, A. Senior, and F. Beaufays, “*Long short-term memory recurrent neural network architectures for large scale acoustic modeling. In Interspeech*”, 2014.
- [14] Geoffrey Hinton, Li Deng, Dong Yu, George E Dahl, Abdelrahman Mohamed, Navdeep Jaitly, Andrew Senior, Vincent Vanhoucke, Patrick Nguyen, Tara N Sainath, et al., “*Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups*” Signal Processing Magazine, IEEE, vol. 29, no. 6, pp. 82–97, 2012.
- [15] J. Chorowski, D. Bahdanau, K. Cho, and Y. Bengio, “*End-to-end Continuous Speech Recognition using Attention-based Recurrent NN: First Results*”, 2015.
- [16] Sepp Hochreiter and Jurgen Schmidhuber, “Long short-term memory,” Neural computation, vol. 9, no. 8, pp. 1735–1780, 1997.
- [17] Chung-Cheng Chiu, Wei Han, Yu Zhang, Ruoming Pang, Sergey Kishchenko, Patrick Nguyen, Arun Narayanan, Hank Liao, Shuyuan Zhang, Anjuli Kannan, Rohit Prabhavalkar, Zhifeng Chen, Tara Sainath, Yonghui Wu, “*A Comparison Of End-To-End Models For Long-Form Speech Recognition*”, 2019.
- [18] Jan Chorowski, Dzmitry Bahdanau, Dmitriy Serdyuk, Kyunghyun Cho, Yoshua Bengio, “*Attention-Based Models for Speech Recognition*”, 2015.
- [19] Baidu Research – Silicon Valley AI Lab, Dario Amodei, Rishita Anubhai, Eric Battenberg, Carl Case, Jared Casper, Bryan Catanzaro, Jingdong Chen, Mike Chrzanowski, Adam Coates, Greg Diamos, Erich Elsen, Jesse Engel, Linxi Fan, Christopher Fougner, Tony Han, Awni Hannun, Billy Jun, Patrick

- LeGresley, Libby Lin, Sharan Narang, Andrew Ng, Sherjil Ozair, Ryan Prenger, Jonathan Raiman, Sanjeev Satheesh, David Seetapun, Shubho Sengupta, Yi Wang, Zhiqian Wang, Chong Wang, Bo Xiao, Dani Yogatama, Jun Zhan, Zhenyao Zhu, “*Deep Speech 2: End-to-End Speech Recognition in English and Mandarin*”, 2015.
- [20] Bo Li, Yanzhang He, Shuo-Yiin Chang. “*Towards End-to-End Speech Recognition*” ISCSLP Tutorial 4, 2018.
- [21] Takenori Yoshimura, Tomoki Hayashi, Kazuya Takeda, Shinji Watanabe, “*End-to-End Automatic Speech Recognition Integrated With CTC-Based Voice Activity Detection*”, 2020.
- [22] A.Hannun, C. Case, J. Casper, B. Catanzaro, G. Diamos, E. Elsen, R. Prenger, S. Satheesh, S. Sengupta, A. Coates, and A. Y. Ng, “*Deep speech: Scaling up end-to-end speech recognition*”, 2014.
- [23] Alex Graves, Greg Wayne, and Ivo Danihelka, “*Neural turing machines*”, 2014.
- [24] William Chan, Navdeep Jaitly, Quoc V. Le, Oriol Vinyals, “*Listen, Attend and Spell*”, 2015.