TỔNG LIÊN ĐOÀN LAO ĐỘNG VIỆT NAM

**TRƯỜNG ĐẠI HỌC TÔN ĐỨC THẮNG**

**KHOA CÔNG NGHỆ THÔNG TIN**



**TRẦN TRIỀU HUY - 52000449**

**TRẦN THỊ NGỌC ÁNH - 52000008**

**TÌM HIỂU MÔ HÌNH**

**NHẬN DẠNG TIẾNG NÓI**

**CHO TIẾNG DÂN TỘC M’NÔNG VÀ XÂY DỰNG**

**ỨNG DỤNG THỬ NGHIỆM**

**DỰ ÁN CÔNG NGHỆ THÔNG TIN**

**KỸ THUẬT PHẦN MỀM**

**THÀNH PHỐ HỒ CHÍ MINH, NĂM 2024**

TỔNG LIÊN ĐOÀN LAO ĐỘNG VIỆT NAM

**TRƯỜNG ĐẠI HỌC TÔN ĐỨC THẮNG**

**KHOA CÔNG NGHỆ THÔNG TIN**



**TRẦN TRIỀU HUY - 52000449**

**TRẦN THỊ NGỌC ÁNH - 52000008**

**TÌM HIỂU MÔ HÌNH**

**NHẬN DẠNG TIẾNG NÓI**

**CHO TIẾNG DÂN TỘC M’NÔNG**

**VÀ XÂY DỰNG**

**ỨNG DỤNG THỬ NGHIỆM**

**DỰ ÁN CÔNG NGHỆ THÔNG TIN**

**KỸ THUẬT PHẦN MỀM**

Người hướng dẫn

**PGS. TS. Lê Anh Cường**

**THÀNH PHỐ HỒ CHÍ MINH, NĂM 2024**

**LỜI CẢM ƠN**

Lời đầu tiên, nhóm chúng tôi muốn bày tỏ lòng biết ơn sâu sắc đến người hướng dẫn trực tiếp của chúng tôi, Thầy Lê Anh Cường, vì sự giám sát, hướng dẫn và hỗ trợ của Thầy trong môn học Dự án Công nghệ Thông tin. Những kiến thức tiếp thu được từ Thầy không chỉ là tiền đề cho báo cáo này mà còn là hành trang quý báu cho chúng tôi trong tương lai. Tuy nhiên, kiến thức là vô hạn mà thời gian chỉ có hạn, nhóm không thể tránh khỏi những thiếu sót. Rất mong nhận được ý kiến đóng góp từ Thầy để chúng em hoàn thiện hơn kiến thức cho bản thân.

Chúng em xin chân thành cảm ơn!

*TP. Hồ Chí Minh, ngày 01 tháng 01 năm 2024.*

*Tác giả*

*(Ký tên và ghi rõ họ tên)*

*Trần Triều Huy*

*Trần Thị Ngọc Ánh*

**CÔNG TRÌNH ĐƯỢC HOÀN THÀNH**

**TẠI TRƯỜNG ĐẠI HỌC TÔN ĐỨC THẮNG**

Tôi xin cam đoan đây là công trình nghiên cứu của riêng tôi và được sự hướng dẫn khoa học của PGS. TS. Lê Anh Cường. Các nội dung nghiên cứu, kết quả trong đề tài này là trung thực và chưa công bố dưới bất kỳ hình thức nào trước đây. Những số liệu trong các bảng biểu phục vụ cho việc phân tích, nhận xét, đánh giá được chính tác giả thu thập từ các nguồn khác nhau có ghi rõ trong phần tài liệu tham khảo.

Ngoài ra, trong Dự án còn sử dụng một số nhận xét, đánh giá cũng như số liệu của các tác giả khác, cơ quan tổ chức khác đều có trích dẫn và chú thích nguồn gốc.

**Nếu phát hiện có bất kỳ sự gian lận nào tôi xin hoàn toàn chịu trách nhiệm về nội dung Dự án của mình**. Trường Đại học Tôn Đức Thắng không liên quan đến những vi phạm tác quyền, bản quyền do tôi gây ra trong quá trình thực hiện (nếu có).

*TP. Hồ Chí Minh, ngày 01 tháng 01 năm 2024*

*Tác giả*

*(Ký tên và ghi rõ họ tên)*

*Trần Triều Huy*

*Trần Thị Ngọc Ánh*

**TÌM HIỂU MÔ HÌNH NHẬN DẠNG TIẾNG NÓI CHO TIẾNG DÂN TỘC M’NÔNG VÀ XÂY DỰNG ỨNG DỤNG THỬ NGHIỆM**

**TÓM TẮT**

Người Việt Nam có rất nhiều nét đẹp văn hóa được lưu truyền từ bao đời nay. Tuy vậy, rất nhiều những nét đẹp đó chịu ảnh hưởng không tốt bởi rất nhiều yếu tố khác nhau như sự đổi mới, sự du nhập của văn hóa phương Tây trong thời điểm hiện tại. Hay xa hơn có thể đề cập đến quá trình đồng hóa của nền phong kiến phương Bắc. Nếu để những điều đó dần mai một hoặc biến mất sẽ là một sự đáng tiếc cho nền văn hóa nước nhà.

Những nét đẹp văn hóa trên có thể kể đến tiếng nói và chữ viết, cụ thể hơn là của người dân tộc thiểu số ở Việt Nam. Nước ta có 54 dân tộc, trong đó có nhiều dân tộc thiểu số có chữ viết và tiếng nói riêng. Nghị quyết Hội nghị lần thứ 5 của Ban chấp hành Trung ương Đảng (khoá VIII) về xây dựng và phát triển nền văn hoá Việt Nam tiên tiến, đậm đà bản sắc dân tộc, có nêu: “…Bảo tồn và phát triển ngôn ngữ, chữ viết của các dân tộc. Đi đôi với việc sử dụng tiếng nói, chữ viết phổ thông, khuyến khích thế hệ trẻ thuộc đồng bào các dân tộc thiểu số học tập, hiểu biết và sử dụng thành thạo tiếng nói, chữ viết của dân tộc mình…”. Chính phủ cũng ra ban hành rất nhiều chủ trương, quy định liên quan đến việc bảo tồn những nét văn hóa này, như giảng dạy tiếng nói và chữ viết cho các học sinh dân tộc thiểu số, đào tạo giáo viên và cấp chứng chỉ tại những cơ sở đào tạo, bồi dưỡng tiếng dân tộc ở địa phương,…

Song ngoài những công cụ cơ bản cho việc giảng dạy như sách giáo khoa, từ điển giấy, mọi thứ còn rất đơn sơ, khó tiếp cận so với tình hình hội nhập và phát triển khoa học công nghệ như hiện nay. Việc xuất hiện những công cụ mới mẻ hơn sẽ giúp công tác giảng dạy cũng như trong sinh hoạt hàng ngày của người dân tộc thiểu số trở nên thuận tiện, dễ dàng hơn, và nghiên cứu về phát hiện tiếng nói dân tộc sẽ đóng góp không hề nhỏ trong công cuộc bảo tồn những di sản ấy.

**MỤC LỤC**

[DANH MỤC HÌNH VẼ viii](#_Toc24033)

[DANH MỤC CÁC CHỮ VIẾT TẮT x](#_Toc1587)

[CHƯƠNG 1. GIỚI THIỆU 1](#_Toc31932)

[1.1 Bài toán 1](#_Toc13717)

[1.2 Các nghiên cứu liên quan 1](#_Toc13528)

[1.3 Mục tiêu nghiên cứu 2](#_Toc31766)

[1.4 Phương pháp nghiên cứu 2](#_Toc20473)

[1.5 Nội dung nghiên cứu 3](#_Toc11626)

[CHƯƠNG 2. KIẾN THỨC NỀN TẢNG 5](#_Toc18426)

[2.1 Lý thuyết về âm học và tiếng nói 5](#_Toc25873)

[2.1.1 Âm học 5](#_Toc30500)

[2.1.2 Tiếng nói 6](#_Toc13782)

[2.2 Hệ thống ngữ âm tiếng M’Nông 6](#_Toc18481)

[2.3 Khái quát về công nghệ nhận dạng tiếng nói 7](#_Toc24119)

[2.3.1 Khái niệm 7](#_Toc19347)

[2.3.2 Cách thức hoạt động 7](#_Toc28965)

[2.4 Mô hình Whisper 8](#_Toc943)

[2.4.1 Khái niệm 8](#_Toc7618)

[2.4.2 Cách thức hoạt động 8](#_Toc2089)

[2.4.3 Các thuật toán liên quan 9](#_Toc22017)

[2.4.4 Ý nghĩa của mô hình Whisper trong nhận diện tiếng nói 10](#_Toc24705)

[CHƯƠNG 3. PHƯƠNG PHÁP ĐỀ XUẤT 11](#_Toc21335)

[3.1 Xử lý dữ liệu 11](#_Toc14469)

[3.2 Mô hình 11](#_Toc12763)

[3.3 Multitask Format 12](#_Toc25959)

[3.4 Hiệu suất 13](#_Toc23947)

[3.5 Hướng tiếp cận 15](#_Toc10362)

[CHƯƠNG 4. THỰC NGHIỆM 16](#_Toc10893)

[4.1 Dữ liệu thực nghiệm 16](#_Toc17635)

[4.2 Cài đặt thực nghiệm 17](#_Toc30885)

[4.2.1 Load Dataset 17](#_Toc26559)

[4.2.2 Chuẩn bị Tokenizer, Feature Extractor và xủ lý data 18](#_Toc829)

[4.2.3 Load WhisperFeatureExtractor 18](#_Toc7824)

[4.2.4 Load WhisperTokenizer 19](#_Toc19109)

[4.2.5 Kết hợp để tạo ra WhisperProcessor 19](#_Toc13800)

[4.2.6 Chuẩn bị dữ liệu 20](#_Toc10372)

[4.2.7 Huấn luyện và đánh giá 21](#_Toc27317)

[4.2.8 Định nghĩa Data Collator 21](#_Toc28806)

[4.2.9 Số liệu đánh giá 23](#_Toc27285)

[4.2.10 Tải pre-trained checkpoint 24](#_Toc18383)

[4.2.11 Định nghĩa Traing Arguments 24](#_Toc17434)

[4.2.12 Huấn luyện mô hình 26](#_Toc28865)

[4.3 Đánh giá 27](#_Toc11754)

[4.4 Thực nghiệm và kết quả 28](#_Toc30214)

[CHƯƠNG 5. KẾT LUẬN 31](#_Toc1050)

[TÀI LIỆU THAM KHẢO 32](#_Toc14098)

# DANH MỤC HÌNH VẼ

[Hình 2.1 . Biểu diễn tín hiệu âm thanh 5](#_Toc30993)

[Hình 3.1 . WER trên LibriSpeech dev-clean (%) 13](#_Toc3896)

[Hình 3.2 . So sánh tính hiệu quả trên nhiều bộ dữ liệu 14](#_Toc11436)

[Hình 4.1 . Bộ dữ liệu tiếng M’Nông được đẩy lên Hugging Face 16](#_Toc21218)

[Hình 4.2 . Phân bố độ dài của các file âm thanh trong bộ dữ liệu 17](#_Toc31615)

[Hình 4.3 . Load dataset 17](#_Toc24053)

[Hình 4.4 . Kết quả của load dataset 18](#_Toc770)

[Hình 4.5 . Load WhisperFeatureExtractor 19](#_Toc7489)

[Hình 4.6 . Load WhisperTokenizer 19](#_Toc5935)

[Hình 4.7 . Tạo WhisperProcessor 20](#_Toc2632)

[Hình 4.8 . In data đầu tiên 20](#_Toc12335)

[Hình 4.9 . Điều chỉnh tốc độ lấy mẫu 20](#_Toc9726)

[Hình 4.10 . Resample mẫu âm thanh về tốc độ chuẩn 20](#_Toc23346)

[Hình 4.11 . Hàm chuẩn bị dữ liệu 21](#_Toc4301)

[Hình 4.12 . Áp dụng hàm chuẩn bị dữ liệu 21](#_Toc12959)

[Hình 4.13 . Định nghĩa và khởi tạo Data Collator 23](#_Toc25564)

[Hình 4.14 . Xác định WER 23](#_Toc8557)

[Hình 4.15 . Hàm trả về WER 24](#_Toc12177)

[Hình 4.16 . Tải một pre-trained checkpoint 24](#_Toc30160)

[Hình 4.17 . Cấu hình lại mô hình 24](#_Toc17237)

[Hình 4.18 . Ý nghĩa các tham số liên quan 25](#_Toc1817)

[Hình 4.19 . Chuyển các đối số huấn luyện cho Trainer 26](#_Toc8855)

[Hình 4.20 . Huấn luyện mô hình 26](#_Toc29399)

[Hình 4.21 . Kết quả huấn luyện 27](#_Toc19219)

[Hình 4.22 . Đưa kết quả lên Hugging Face Hub 27](#_Toc13635)

[Hình 4.23 . Công thức tính WER 28](#_Toc12372)

[Hình 4.24 . Ví dụ đơn giản sử dụng mô hình đã qua huấn luyện 29](#_Toc8411)

[Hình 4.25 . Kết quả khi thực hiện kiểm tra trên bộ dữ liệu sử dụng để kiểm định 29](#_Toc4215)

[Hình 4.26 . Kết quả khi thực hiện kiểm tra trên bộ dữ liệu sử dụng để kiểm định 30](#_Toc31052)

# DANH MỤC CÁC CHỮ VIẾT TẮT

ASR Automatic Speech Reconigtion

ML Machine Learning

WER Word Error Rate

# GIỚI THIỆU

## Bài toán

Đề tài của nhóm chúng tôi mang tên “Hệ thống nhận diện tiếng nói dân tộc M’Nông”. Để thực hiện được việc nhận dạng tiếng nói dân tộc, trước tiên phải xây dựng một hệ thống có khả năng nhận dạng tiếng nói tự động (Automatic Speech Recognition - ASR), là một hệ thống có khả năng biến đổi âm thanh thành từ ngữ. Việc xây dựng hệ thống ASR đòi hỏi phải có lý thuyết, kỹ thuật từ nhiều phương diện khác nhau như: học máy, trí tuệ nhân tạo, lý thuyết xác suất thống kê, âm học - vật lý,… Ngày nay trên thế giới đã có nhiều nhóm nghiên cứu phát triển thành công các mô hình nhận dạng tiếng nói cho các ngôn ngữ phổ biến như: tiếng Anh, tiếng Tây Ban Nha, tiếng Trung Quốc,… nhưng giải pháp về nhận dạng cho tiếng Việt và đặc biệt là tiếng nói dân tộc thiểu số tại Việt Nam vẫn còn rất nhiều hạn chế.

## Các nghiên cứu liên quan

Lịch sử phát triển của công nghệ nhận dạng tiếng nói đã có từ rất lâu đời và là một trong những lĩnh vực thu hút nhiều cá nhân, tổ chức tiến hành nghiên cứu bởi ứng dụng to lớn của nó trong đời sống xã hội. Nhờ vào sự phát triển của trí tuệ nhân tạo mà việc con người có thể “trò chuyện” với máy tính đã không còn là bất khả thi.

Lần đầu tiên công nghệ này trình làng với thế giới vào năm 1877 khi nhà khoa học Thomas Edison phát minh ra máy ghi âm - thiết bị có khả năng ghi nhớ và tạo lại âm thanh. Sau đó, ngày một nhiều nghiên cứu về nhận dạng tiếng nói được công bố như: Audrey - bộ nhận dạng chữ số tự động - được phát triển bởi nhóm các kỹ sư tại Bell Labs (1952), Shoe Box (IBM, 1962), chương trình DARPA SUR (Bộ Quốc phòng Mỹ), IVR (SpeechWorks và Nuance), Hidden Markov Model (Đại học Cambridge),… hay gần hơn và được áp dụng rộng rãi có thể kể đến Siri của Apple cho phép người dùng tương tác với trợ lý ảo bằng giọng nói hoặc Google Assistant - trợ lý ảo được phát triển bởi Google cho điện thoại di động và nhà thông minh.

Ở Việt Nam đến nay đã có nhiều nghiên cứu về lĩnh vực này và đạt được một số thành tựu nhất định như: Vspeech (Đại học Bách Khoa Tp.HCM), Vietvoice hay AILab của phòng thí nghiệm AI thuộc Đại học Khoa học Tự nhiên. Có thể truy cập phần mềm miễn phí tại địa chỉ <http://www.ailab.hcmus.edu.vn/>.

## Mục tiêu nghiên cứu

Có thể thấy, lịch sử phát triển của các hệ thống nhận dạng tiếng nói từ trong đến ngoài nước là vô cùng phong phú và lâu đời. Tuy nhiên, Việt Nam là một quốc gia với 54 dân tộc anh em cùng sinh sống, bên cạnh tiếng Kinh phổ biến nhất trong cộng đồng người Việt Nam, chúng ta còn rất nhiều tiếng nói khác.

Dân tộc M’Nông là dân tộc thiểu số sinh sống tại Việt Nam, với dân số khoảng 127.334 người (theo số liệu cuộc Tổng điều tra dân số và nhà ở năm 2019), cư trú tại 51 trên tổng số 63 tỉnh, thành phố. Tiếng M’Nông là một ngôn ngữ thuộc ngữ tộc Môn-Khmer của ngữ hệ Nam Á. Tiếng M’Nông được nghiên cứu lần đầu bởi nhà ngôn ngữ học Richard Phillips vào đầu những năm 1970 và đây là một ngôn ngữ có một hệ thống ngữ âm phức tạp, với nhiều thanh điệu và phụ âm.

Hệ thống nhận dạng tiếng nói dân tộc M’Nông là một hệ thống có khả năng nhận dạng tiếng nói của người M’Nông thông qua ứng dụng công nghệ thông tin.

Mục tiêu khi chúng tôi thực hiện đề tài là xây dựng một hệ thống nhận dạng tiếng nói dân tộc M’Nông có độ chính xác cao, có thể áp dụng vào thực tế và đáp ứng được nhiều yêu cầu của nhiều mục đích sử dụng khác nhau.

## Phương pháp nghiên cứu

Nghiên cứu về hệ thống nhận dạng tiếng nói dân tộc M’Nông đòi hỏi cần một mô hình kỹ lưỡng và toàn diện. Chính vì thế, cần nghiên cứu và phân tích các tài liệu về lý thuyết nhận dạng tiếng nói, đặc điểm ngữ âm của tiếng M’Nông, cũng như các nghiên cứu trước đây về hệ thống nhận dạng tiếng nói dân tộc thiểu số. Mục đích nhằm nắm rõ các nguyên lý, phương pháp và thực tiễn của nhận dạng tiếng nói, cũng như các đặc điểm của tiếng M’Nông.

Đối tượng được chúng tôi chọn để nghiên cứu là tiếng M’Nông của những người nói giọng huyện Lắk thuộc tỉnh Đắk Lắk. Tất cả các nội dung nghiên cứu trong dự án chỉ giới hạn trong phạm vi điều kiện người nói trong trạng thái bình thường, các vấn đề người nói cố tình cải trang hay giả giọng nói đều nằm ngoài phạm vi nghiên cứu của dự án.

Chúng tôi sử dụng bộ dữ liệu bao gồm các mẫu âm thanh tiếng M’Nông do nhóm thu thập. Bao gồm các câu, từ thường dùng trong đời sống hằng ngày và được thu âm bởi chính người dân địa phương nói tiếng M’Nông. Dữ liệu âm thanh được thu từ ba người nói và với các giọng nói khác nhau (nam, nữ) để hệ thống có thể nhận diện được nhiều giọng nói.

Dữ liệu thu thập cần được xử lý để chuẩn hóa trước khi tiến hành huấn luyện. Trong quá trình xử lý dữ liệu, chúng tôi phân đoạn âm thanh thành các đoạn âm thanh nhỏ hơn, mỗi đoạn sẽ tương ứng với một câu hoặc một từ trong tiếng M’Nông.

Mô hình nhận dạng tiếng nói là một thuật toán trong học máy và được sử dụng để phân loại các mẫu âm thanh. Trong đề tài này, chúng tôi sử dụng mô hình Whisper của OpenAI vì đây là một hệ thống nhận dạng giọng nói tự động mã nguồn mở, thiết kế dễ sử dụng và cho độ chính xác cao.

Trong quá trình thực hiện huấn luyện mô hình, chúng tôi cung cấp cho mô hình các mẫu âm thanh và nhãn của các mẫu âm thanh đó. Mô hình sẽ học cách phân loại các mẫu âm thanh dựa trên các nhãn đã cung cấp và cho ra sản phẩm là văn bản.

Sau khi huấn luyện, mô hình được thử nghiệm trên dữ liệu mới để đánh giá độ chính xác và có các điều chỉnh phù hợp.

## Nội dung nghiên cứu

Chương 1: Mô tả tổng quan: giới thiệu về đề tài, bài toán được đề cập, các nghiên cứu có liên quan đến đề tài nhận dạng tiếng nói trong và ngoài nước, các mục tiêu nghiên cứu đề ra cũng như phương pháp để tiến hành nghiên cứu.

Chương 2: Lý thuyết về kiến thức nền tảng: về âm học, ngữ âm học, đặc điểm nổi bật trong tiếng M’Nông, các kiến thức để xây dựng và sử dụng một hệ thống nhận dạng tiếng nói. Các cơ sở lý thuyết về mô hình Whisper, bao gồm các thuật toán có liên quan, ý nghĩa của Whisper trong một hệ thống nhận dạng tiếng nói.

Chương 3: Trình bày phương pháp đề xuất: xây dựng kiến trúc, giải pháp của hệ thống.

Chương 4: Trình bày chi tiết thực nghiệm, thống kê về dữ liệu, các mô hình thực nghiệm là gì, xây dựng mô hình huấn luyện, tiến hành quá trình huấn luyện và giải thích kết quả huấn luyện.

Chương 5: Kết luận và chỉ ra những kết quả đạt được, những điểm còn hạn chế cũng như kinh nghiệm tích lũy sau quá trình thực hiện dự án, từ đó đưa ra hướng cải thiện và phát triển dự án.

# KIẾN THỨC NỀN TẢNG

## Lý thuyết về âm học và tiếng nói

### Âm học

* Khái niệm

Âm thanh là sự dao động cơ của các vật chất trong môi trường lan truyền và khi nó đến tai chúng ta sẽ cảm nhận được âm thanh. Trong môi trường chân không sẽ không có âm thanh bởi vì không có dao động cơ được tạo ra. Âm thanh được phát dưới dạng sóng âm và sóng âm có thể lan truyền trong chất rắn, lỏng và khí. Khi bắt đầu nghiên cứu về âm thanh, ta sẽ chú ý đến 2 đặc trưng chính: đặc trưng vật lý và đặc trưng sinh học.

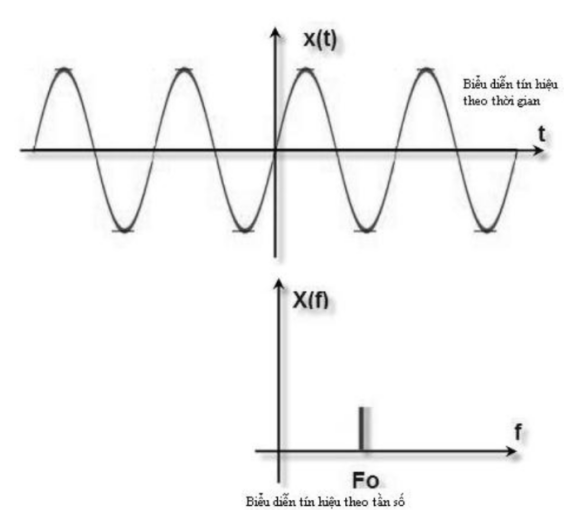
* Biểu diễn tín hiệu của âm thanh theo thời gian và tần số

Ta có thể dùng hàm toán học x(t) để biểu diễn âm thanh theo thời gian. Trong đó:

- t: thời gian

- x: biên độ biến thiên (ly độ)

Biểu diễn x(t) bằng đồ thị thời gian. Đặt x(t) = A.sin οt = A.sin 2πFοt:



Hình 2.1. Biểu diễn tín hiệu âm thanh

Phổ tính hiệu là cách biểu diễn x(t) theo tần số. Với tín hiệu sine đề cập ở trên, đồ thị phổ chính là một vạch có cao độ A tại một điểm có tần số Fο. Khi đó ta nói đó là phổ vạch. Thực tế, với x(t) bất kỳ, biến thiên và không tuần hoàn, ta sẽ dùng Fourier để tính toán phổ tín hiệu. Khi đó ta có được phổ liên tục X(Ω).

* Các đại lượng đặc trưng

Tần số (f) là số lần dao động của phần tử trong một giây. Đơn vị: Hz.

Cường độ âm thanh là năng lượng được sóng âm truyền trong một đơn vị thời gian qua mặt tiết diện đặt vuông góc với phương truyền âm.

Chu kỳ âm thanh (T) là thời gian âm thanh thực hiện một dao động tuần hoàn.

Tốc độ truyền âm là tốc độ truyền năng lượng âm từ nguồn đến nơi thu. Đơn vị: m/s.

Thanh áp là lực tác dụng vào tai người nghe. Đơn vị: 1pa=1 N/m2.

Đơn vị đo âm thanh: Bel = 10 lg P2/P1; Deccibel = 20 lg I2/I1.

### Tiếng nói

Các cơ quan phát ra âm thanh của con người bao gồm phổi, thanh quản, khí quản, mũi và miệng. Dây thanh âm ở thanh quản sẽ rung lên khi có luồng không khí đi qua. Miệng và mũi đóng vai trò là một ống âm.

Khi nói, không khí được đẩy từ phổi qua miệng và khoang mũi. Luồng không khí này bị cản trở và thay đổi do hoạt động của lưỡi và môi. Điều này tạo ra các co giãn của không khí, sóng âm, âm thanh. Những âm thanh tạo thành, tương ứng với nguyên âm và phụ âm, thường được gọi là âm vị. Các âm vị được kết hợp với nhau tạo thành từ. Mỗi âm vị được hình thành trong quá trình nói phụ thuộc vào ngữ cảnh của nó, tức là phụ thuộc vào âm vị đứng ngay trước và âm vị đứng ngay sau nó. Tuy nhiên, tiếng nói không chỉ là chuỗi các âm vị tạo thành từ và câu. Có nhiều thành phần của tiếng nói mang thông tin, ví dụ: phong cách nói thể hiện qua âm điệu, cao độ, âm lượng, ngữ điệu. Thông tin này đôi khi được gọi là thông tin phi từ vựng (paralinguistic) của tiếng nói và cũng được sử dụng trong nhiều phương pháp nhằm nâng cao độ chính xác của hệ thống nhận diện.

## Hệ thống ngữ âm tiếng M’Nông

Tiếng M’Nông là một ngôn ngữ thuộc nhóm ngôn ngữ Môn-Khmer và được sử dụng bởi dân tộc M’Nông ở Việt Nam, Campuchia và Lào. Tiếng M’Nông có hệ thống ngữ âm tương đối phức tạp, với 6 thanh điệu, 44 âm vị, bao gồm 26 phụ âm và 18 nguyên âm được phân loại dựa trên cách phát âm.

Ngoài ra, tiếng M’Nông còn có các phụ âm và nguyên âm biến thể, được tạo thành bằng cách kết hợp các âm vị cơ bản.

Tính chất ngữ âm của tiếng M’Nông: Tiếng M’Nông là một ngôn ngữ có trật tự âm tiết mở, nghĩa là âm tiết bắt buộc phải có nguyên âm. Tiếng M’Nông cũng là một ngôn ngữ có phân chia âm tiết thành âm tiết chính và âm tiết phụ. Âm tiết chính là âm tiết chứa thanh điệu và âm tiết phụ là âm tiết không chứa thanh điệu.

Thanh điệu tiếng M’Nông có thể được phân loại thành 2 nhóm chính: thanh điệu đơn và thanh điệu kép. Thanh điệu đơn là thanh điệu chỉ có một độ cao nhất định. Thanh điệu kép là thanh điệu có hai độ cao.

## Khái quát về công nghệ nhận dạng tiếng nói

### Khái niệm

Automatic Speech Recognition (ASR) là một nhánh trong Machine Learning. Đây là công nghệ cho phép máy tính có thể nhận dạng và chuyển đổi từ ngôn ngữ nói thành văn bản viết. Hiện nay có khá nhiều ứng dụng cho hệ thống ASR, chẳng hạn như phần mềm đọc chính tả hay trợ lý ảo. Thay vì phải lập trình các quy tắc để chuyển đổi dữ liệu đầu vào từ giọng nói thành đầu ra là văn bản thì mô hình Học máy được đào tạo bằng cách đưa các tập dữ liệu lớn vào một thuật toán. Trải qua quá trình huấn luyện, mô hình ngày càng cho khả năng suy luận tốt hơn, và có khả năng nhận dạng được tiếng nói của con người (2021, VinBigData).

### Cách thức hoạt động

Để chuyển đổi từ giọng nói sang văn bản, hệ thống phải thực hiện một quá trình gồm nhiều bước liên tục. Khi nói, ta sẽ tạo ra những rung động trong không khí. Bộ Analog-to-Digital Converter (ADC) chuyển các sóng analog này thành dữ liệu mà máy tính có thể hiểu được.

Để làm được điều này, hệ thống cần thu thập các mẫu âm thanh bằng cách đo chính xác sóng âm ở các khoảng thời gian gần với nhau, sau đó lọc âm thanh đã được số hoá để loại bỏ các tiếng ồn không cần thiết, đôi khi sẽ tách chúng thành các dải tần số khác nhau. Nó cũng sẽ tinh chỉnh âm thanh đến một mức âm lượng không thay đổi hoặc sẽ có sắp xếp theo thời gian. Và không phải lúc nào con người cũng nói với tốc độ giống nhau nên âm thanh phải được điều chỉnh để phù hợp với tốc độ mà âm thanh mẫu được ghi nhận trong bộ nhớ của máy tính.

Tiếp theo, tín hiệu được chia thành nhiều phần nhỏ hơn (khoảng vài phần trăm giây, thậm chí là phần ngàn giây trong trường hợp có phụ âm cuối khó phân biệt). Chương trình sau đó đặt những phần âm thanh này vào các âm vị có sẵn trong ngôn ngữ thích hợp.

## Mô hình Whisper

### Khái niệm

Mô hình Whisper là một mô hình nhận dạng tiếng nói đa ngôn ngữ và đa nhiệm, được phát triển bởi OpenAI. Mô hình này được huấn luyện trên một tập dữ liệu khổng lồ gồm 680.000 giờ âm thanh tiếng nói, bao gồm 96 ngôn ngữ khác nhau.

Mô hình Whisper hoạt động theo hướng end-to-end, sử dụng kiến trúc mạng noron nhân tạo Transformer.

Mô hình Whisper đã đạt được kết quả ấn tượng trong các thử nghiệm. Độ chính xác mà mô hình mang lại trong các tác vụ nhận dạng giọng nói và dịch giọng nói tương đương với các mô hình nhận dạng tiếng nói chuyên biệt cho từng ngôn ngữ.

### Cách thức hoạt động

Whisper là một công cụ được tạo ra bởi OpenAI có khả năng hiểu và chuyển đổi ngôn ngữ nói thành văn bản, tương tự như cách Siri hoặc Alexa hoạt động. Cách mà Whisper hoạt động giống như một thông dịch viên. Nó sử dụng một kiến trúc mã hóa-giải mã Transformer. Để đơn giản hóa, hãy tưởng tượng ai đó dịch từ tiếng Anh sang tiếng Việt. Phần ‘mã hóa’ sẽ hiểu câu tiếng Anh, và phần ‘giải mã’ sau đó sẽ tạo ra câu tiếng Việt. Trong Whisper, phần ‘mã hóa’ hiểu ngôn ngữ nói (âm thanh), và phần ‘giải mã’ tạo ra văn bản.

Để học cách hiểu và chuyển đổi ngôn ngữ nói, Whisper đã được huấn luyện bằng cách sử dụng một lượng lớn dữ liệu từ internet - tương đương với việc liên tục lắng nghe suốt hơn 77 năm. Dữ liệu này bao gồm nhiều ngôn ngữ khác nhau và đa nhiệm (bảo gồm nhiều loại nhiệm vụ khác nhau, không chỉ là chuyển đổi). Dữ liệu âm thanh mà Whisper học được xử lý theo một cách cụ thể để làm cho hệ thống dễ hiểu hơn. Điều này giống như khi bạn điều chỉnh cài đặt trên TV để làm cho hình ảnh rõ ràng hơn. Âm thanh được ‘tái tạo’ thành chất lượng chuẩn (16.000 Hz), sau đó nó được biến đổi thành biểu đồ thị trực quan mà hệ thống có thể học. Quá trình biến đổi này được thực hiện trong các phần nhỏ chồng lên nhau để đảm bảo không có phần nào của âm thanh bị bỏ lỡ.

### Các thuật toán liên quan

Mô hình Whisper sử dụng các thuật toán sau:

- Kiến trúc mạng nơ-ron nhân tạo Transformer: Kiến trúc Transformer là một kiến trúc mạng nơ-ron nhân tạo được sử dụng phổ biến trong các tác vụ xử lý ngôn ngữ tự nhiên như dịch máy, tóm tắt đoạn văn bản, và nhận dạng tiếng nói. Kiến trúc Transformer có khả năng xử lý thông tin theo hướng end-to-end, giúp mô hình có thể học hỏi các mối quan hệ phức tạp giữa các từ và các đoạn văn bản.

- Kỹ thuật chuyển đổi attention: Kỹ thuật chuyển đổi attention là một kỹ thuật được sử dụng trong kiến trúc Transformer để giúp mô hình tập trung vào các thông tin quan trọng trong một đoạn văn bản. Kỹ thuật này hoạt động bằng cách tính toán một trọng số cho mỗi từ trong đoạn văn bản, dựa trên mức độ liên quan của từ đó với từ đang được xử lý.

- Kỹ thuật học có giám sát: Kỹ thuật học có giám sát là một kỹ thuật học máy trong đó mô hình được huấn luyện trên một tập dữ liệu có nhãn. Trong trường hợp của mô hình Whisper, tập dữ liệu đào tạo bao gồm các mẫu âm thanh tiếng nói và nhãn của các mẫu âm thanh đó. Mô hình sẽ học cách phân loại các mẫu âm thanh dựa trên các nhãn đã cung cấp.

Ngoài các thuật toán trên, mô hình Whisper còn sử dụng một số kỹ thuật khác để cải thiện hiệu quả của mô hình:

- Kỹ thuật giảm kích thước: Kỹ thuật giảm kích thước được sử dụng để giảm kích thước của các mảng dữ liệu, giúp mô hình có thể chạy nhanh hơn và hiệu quả hơn.

- Kỹ thuật tăng tốc: Kỹ thuật tăng tốc được sử dụng để tăng tốc độ của mô hình, chẳng hạn như sử dụng các kỹ thuật tính toán song song hoặc sử dụng các phần cứng chuyên dụng.

### Ý nghĩa của mô hình Whisper trong nhận diện tiếng nói

Với 680,000 giờ âm thanh được gán nhãn, bộ dữ liệu của Whisper là một trong những bộ dữ liệu lớn nhất từng được tạo ra trong lĩnh vực nhận diện tiếng nói có giám sát.

Mô hình Whisper trong hệ thống nhận diện giọng nói giúp cải thiện độ chính xác của việc nhận dạng giọng nói trong các môi trường có tiếng ồn hoặc khi người nói nói nhỏ. Điều này có thể được ứng dụng trong nhiều lĩnh vực như: trợ lý ảo, tạo phụ đề, dịch ngôn ngữ, bảo mật,…

# PHƯƠNG PHÁP ĐỀ XUẤT

## Xử lý dữ liệu

Theo xu hướng của các nghiên cứu gần đây, chúng tôi áp dụng một phương pháp tối giản để xử lý dữ liệu. Khác với các hệ thống ASR khác, mô hình Whisper có thể dự đoán bản thô của audio mà không cần bất kỳ tiêu chuẩn hay ràng buộc nào. Chúng tôi phân đoạn âm thanh thành các đoạn âm thanh nhỏ hơn, mỗi đoạn tương ứng với một câu hoặc một từ. Tất cả các đoạn này được huấn luyện và sử dụng làm dữ liệu huấn luyện.

Nhiều bản ghi trên internet thực tế không phải do con người tạo ra mà là kết quả của các hệ thống ASR. Các nghiên cứu hiện nay đã chỉ ra rằng việc huấn luyện trên các bộ dữ liệu chứa cả dữ liệu được tạo ra bởi con người và máy có thể ảnh hưởng đáng kể đến hiệu suất của các hệ thống dịch. Để tránh việc học "ngôn ngữ bản ghi" không tự nhiên, chúng tôi xây dựng bộ dữ liệu âm thanh từ chính người dân địa phương nói tiếng M’Nông tại huyện Lắk của tỉnh Đắk Lắk, điều này làm tăng cường độ tin cậy cho bộ dữ liệu và chất lượng âm thanh tốt cũng giúp mô hình hoạt động mạnh mẽ hơn.

## Mô hình

Chúng tôi sử dụng một kiến trúc có sẵn là Transformer mã hóa-giải mã vì kiến trúc này đã được xác minh có thể mở rộng và đáng tin cậy. Tất cả âm thanh được lấy mẫu thành 16.000 Hz, và biểu diễn quang phổ Mel với cường độ log 80 channels được tính toán trên cửa sổ 25 mili giây với bước nhảy là 10 mili giây.

Để chuẩn hóa các đặc trưng, đầu vào được tỷ lệ từ -1 đến 1 với giá trị trung bình xấp xỉ 0 trên toàn bộ dữ liệu tiền đào tạo. Bộ mã hóa xử lý biểu diễn đầu vào này với một tầng stem nhỏ bao gồm hai tầng tích chập với chiều rộng bộ lọc là 3 và hàm kích hoạt GELU (Hendrycks & Gimpel, 2016), trong đó tầng tích chập thứ hai có bước nhảy là 2. Phần nhúng có vị trí hình sin sau đó được thêm vào đầu ra của stem, tiếp đó là áp dụng các khối Transformer của bộ mã hóa. Transformer sử dụng các khối pre-activation và áp dụng một lớp chuẩn hóa cuối cho đầu ra của bộ mã hóa. Bộ giải mã sử dụng phần nhúng được học và biểu diễn thông tin đầu vào-đầu ra được ràng buộc. Bộ mã hóa và bộ giải mã có cùng chiều rộng và số lượng khối Transformer.

Năm 2021, Radford và cộng sự đã nhận thấy trong khi tinh chỉnh một mô hình thị giác máy tính trên bộ dữ liệu ImageNet đã tăng 9.2% độ chính xác khi phân loại đối tượng mà không có bất kỳ cải thiện nào về độ chính xác trung bình khi phân loại các đối tượng giống nhau trên 7 bộ dữ liệu khác. Một mô hình có thể đạt hiệu suất vô cùng tốt khi đào tạo trên một bộ dữ liệu vẫn có thể mắc nhiều lỗi cơ bản khi được đánh giá trên bộ dữ liệu khác. Mô hình Whisper nhóm chúng tôi chọn được đánh giá trên speech-to-text thông thường (không phải trên tiếng M’Nông) nhưng nhóm vẫn chọn vì độ chính xác ổn, có thể chấp nhận và thiết kế dễ tiếp cận.

## Multitask Format

Việc dự đoán từ ngữ nào được nói trong một đoạn âm thanh là vấn đề cơ bản của nhận dạng tiếng nói nhưng đó không phải là vấn đề duy nhất. Một hệ thống nhận dạng giọng nói đầy đủ tính năng bao gồm nhiều thành phần bổ sung như phát hiện giọng hoạt động, phân loại người nói và chuẩn hóa văn bản. Những thành phần này thường được xử lý riêng lẻ dẫn đến một hệ thống khá phức tạp. Để giảm độ phức tạp này, mô hình Whisper có thể thực hiện toàn bộ quy trình xử lý giọng nói mà không chỉ là phần nhận diện cơ bản.

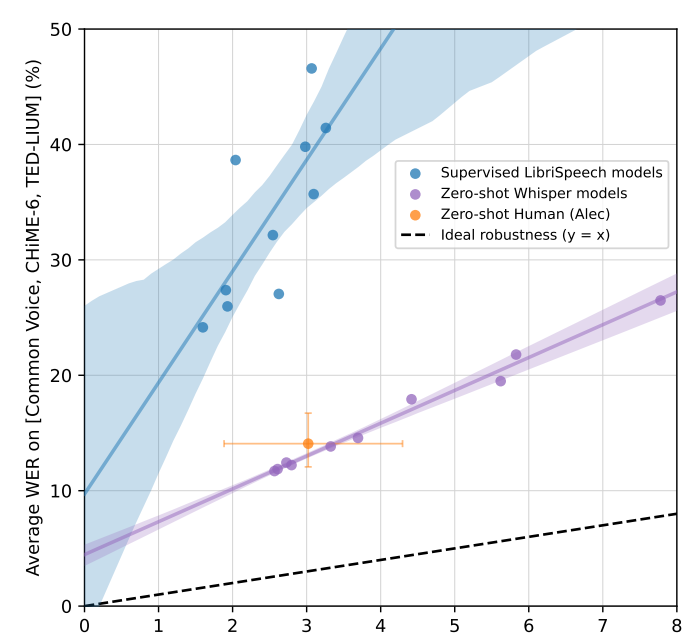
Để làm được điều này, Whisper sử dụng định dạng đơn giản để chỉ định tất cả các nhiệm vụ và thông tin điều kiện dưới dạng một chuỗi các token đầu vào cho bộ giải mã. Whisper chỉ định điểm bắt đầu trước khi dự đoán là một token <start>.

Trước hết, Whisper dự đoán ngôn ngữ đang được nói và biểu diễn bằng một token. Trong trường hợp không có giọng nói được phát hiện, mô hình được huấn luyện để dự đoán token <silence>. Token tiếp theo chỉ định nhiệm vụ “transcription” hoặc “translation” qua token <transcription> hoặc <translation>. Sau đó, mô hình dự đoán liệu có timestamp hay không bằng cách thêm một token <timestamp>. Tại đây, mọi thứ đã được chỉ định đầy đủ và bắt đầu cho ra output.

## Hiệu suất

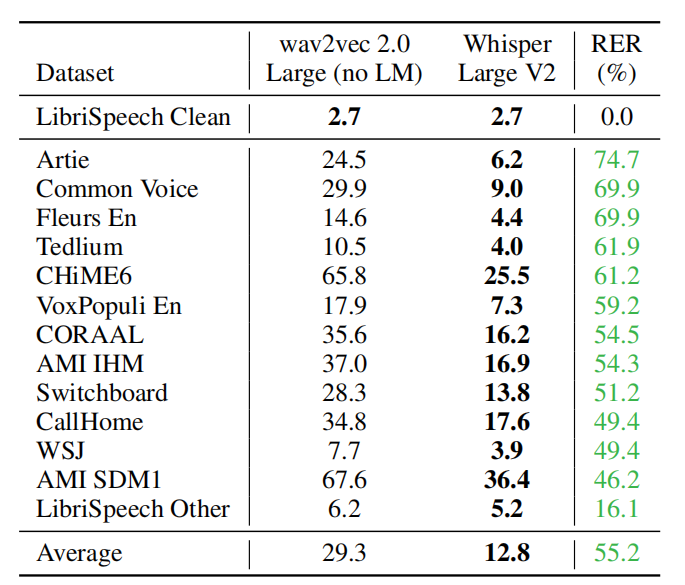
Các mô hình Whisper được huấn luyện trên dữ liệu âm thanh rộng và đa dạng và được đánh giá trong cài đặt zero-shot có thể cho hiệu suất tốt hơn các hệ thống đang tồn tại hiện nay. Có thể so sánh các mô hình Whisper với hiệu suất con người và các mô hình học máy tiêu biểu hiện nay và kiểm tra hiệu suất của chúng.

Các mô hình Whisper zero-shot mạnh mẽ gần với con người. Dù có thể sánh ngang hoặc vượt qua hiệu suất của con người trên tập dev-clean của LibriSpeech, các mô hình LibriSpeech lại mắc khoảng gấp đôi số lỗi so với con người trên các bộ dữ liệu khác. Tuy nhiên trong trường hợp này, độ ổn định ước lượng của các mô hình Whisper zero-shot bao gồm khoảng 95% khoảng tin cậy.



Hình 3.1. WER trên LibriSpeech dev-clean (%)

Chúng ta xem xét tổng thể hiệu suất trung bình trên nhiều bộ dữ liệu và tính hiệu quả để đo lường sự khác biệt trong hiệu suất giữa một tập dữ liệu tham chiếu và một hoặc nhiều tập dữ liệu bên ngoài. Nhóm nghiên cứu Whisper sử dụng LibriSpeech làm tập dữ liệu tham chiếu do vai trò quan trọng của nó trong nghiên cứu nhận diện giọng nói ngày nay và có nhiều mô hình được phát hành dựa trên LibriSpeech.



Hình 3.2. So sánh tính hiệu quả trên nhiều bộ dữ liệu

Cả hai mô hình đều cho hiệu suất trong khoảng 0.1% trên LibriSpeech nhưng mô hình Whisper thực hiện tốt hơn nhiều trên các bộ dữ liệu khác và gây ra ít lỗi hơn trung bình 55.2%. Kết quả được đánh giá dưới dạng tỉ lệ lỗi từ (WER) cho cả hai mô hình.

Mặc dù mô hình Whisper cho một tỉ lệ lỗi WER trên LibriSpeech clean-test khá bình thường là 2.5, tương đương với hiệu suất của một mô hình cơ bản vào giữa năm 2019, nhưng mô hình Whisper có các đặc tính mạnh mẽ khác biệt so với các mô hình LibriSpeech được giám sát và vượt qua tất cả các mô hình LibriSpeech trên các bộ dữ liệu khác. Ngay cả mô hình Whisper nhỏ nhất, có chỉ 39 triệu tham số và tỷ lệ lỗi WER trên LibriSpeech test-clean là 6.7, cũng vượt trội hơn so với mô hình LibriSpeech tốt nhất khi được đánh giá trên các bộ dữ liệu khác. Khi so sánh với một con người, những mô hình Whisper cho thấy độ chính xác và tính mạnh mẽ của chúng khá tương đồng. Mặc dù hiệu suất rất gần nhau nhưng mô hình Whisper đạt được độ giảm lỗi trung bình là 55.2% khi được đánh giá trên các bộ dữ liệu nhận diện giọng nói khác.

## Hướng tiếp cận

Bài toán được đặt ra là biên dịch tiếng M’Nông ở dạng âm thanh thành văn bản tiếng Việt. Trong ngôn ngữ nói hàng ngày của người dân tộc thiểu số nói chung và dân tộc M’Nông nói riêng, họ sử dụng rất nhiều từ mượn của tiếng nói phổ thông. Nên ta sẽ muốn trong quá trình phiên dịch, mô hình phải vừa phát hiện được tiếng M’Nông lẫn tiếng Việt thông thường. Tiếng Việt và tiếng M’Nông có cùng cấu trúc ngữ pháp giờ đây trở thành một đặc điểm ta có thể tận dụng được để tìm phương pháp xử lý bài toán: tinh chỉnh (fine-tune) lại một mô hình phát hiện tiếng Việt phổ thông.

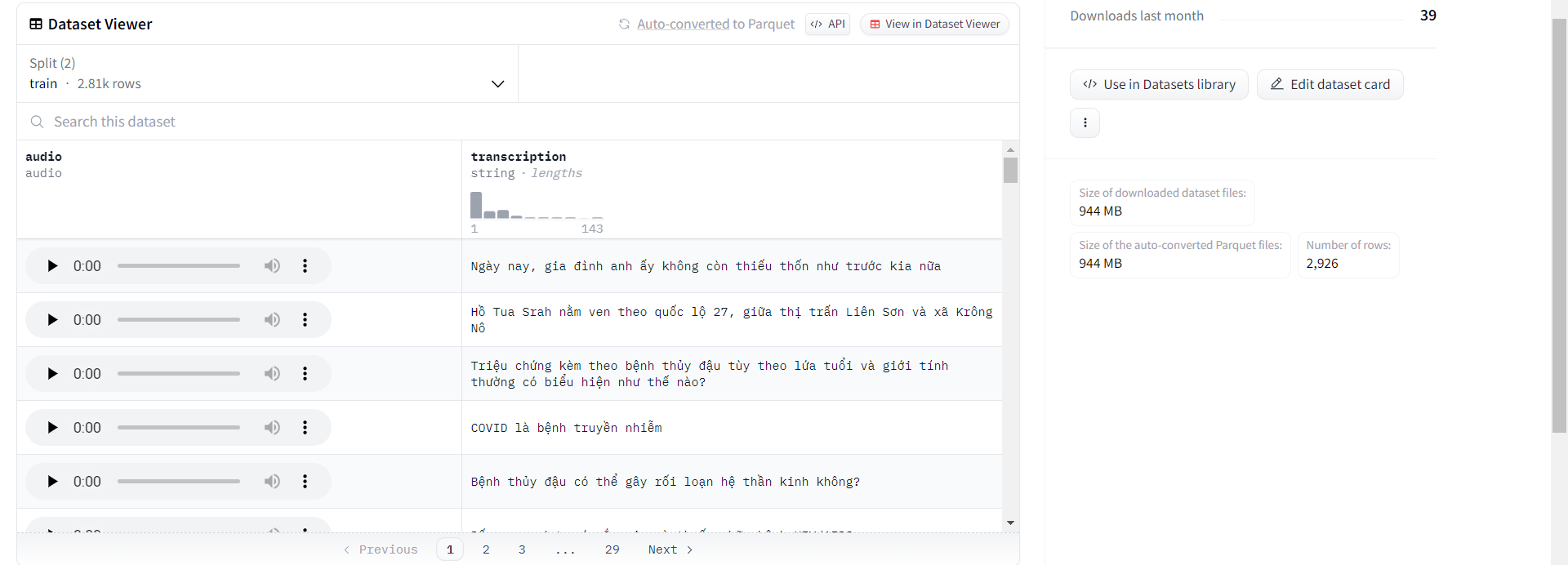
Nếu ta coi việc biên dịch tiếng M’Nông thành tiếng Việt là đang phiên âm tiếng Việt với các cách phát âm khác nhau của cùng một từ ngữ, hướng đi hiệu quả nhất cho bài toán sẽ là tinh chỉnh lại mô hình phát hiện tiếng nói tự động dựa trên bộ dữ liệu tiếng M’Nông đã thu được. Khi tinh chỉnh, các từ tiếng Việt xuất hiện trong câu sẽ được dịch y nguyên, giúp ta không cần phải huấn luyện lại phần từ vựng tiếng Việt cho mô hình. Với mô hình Whisper trên - đã được huấn luyện cho rất nhiều ngôn ngữ trong đó có tiếng Việt, việc huấn luyện sẽ thuận lợi đáng kể.

# THỰC NGHIỆM

## Dữ liệu thực nghiệm

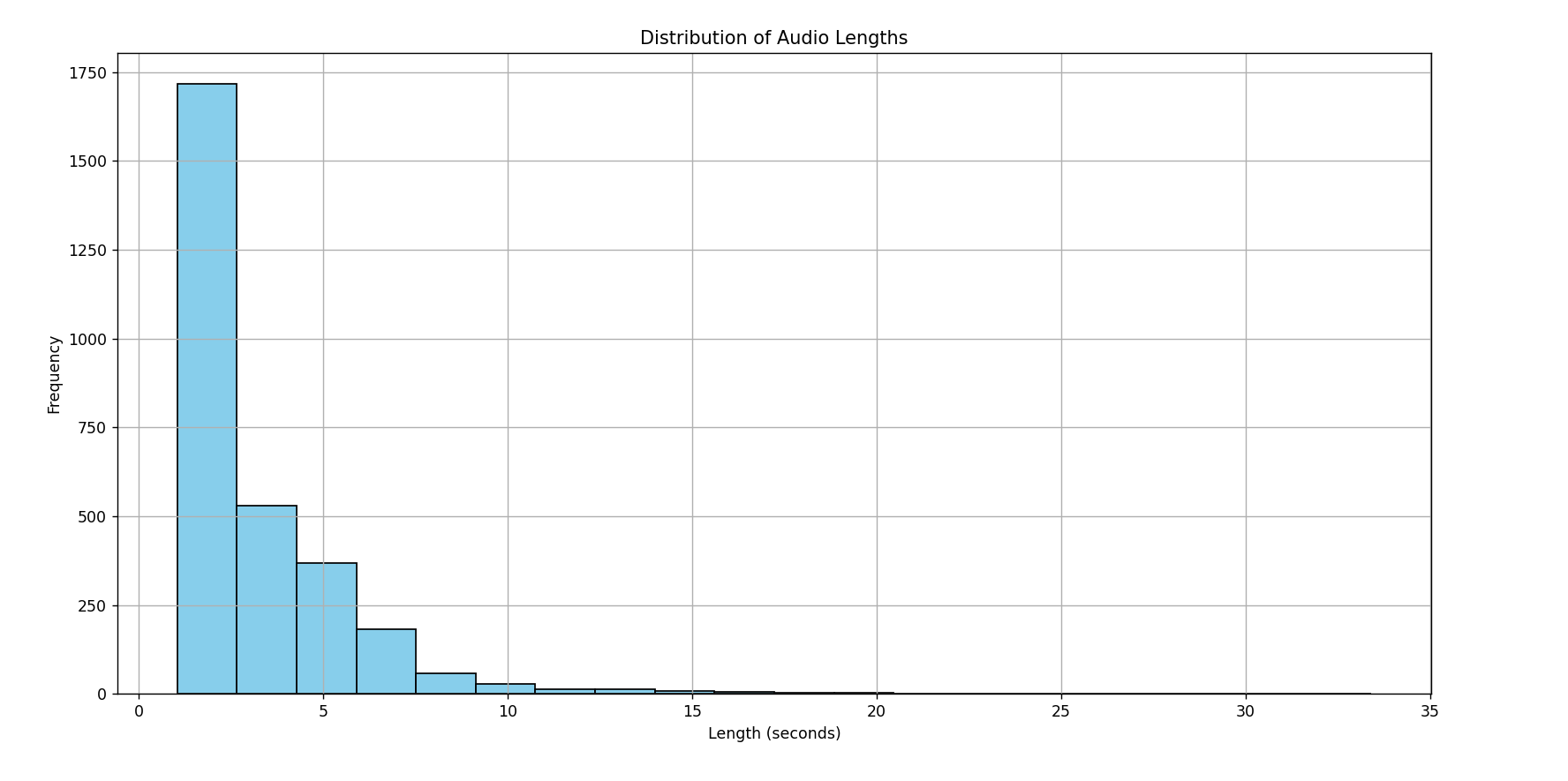
Thu thập dữ liệu: Nhóm liên hệ trực tiếp đến địa phương, cụ thể: Trung tâm Giáo dục nghề nghiệp - Giáo dục thường xuyên huyện Lắk, thầy Tuyn Bing - công tác tại Ban nghiên cứu giáo dục học sinh dân tộc tỉnh Đắk Lắk, Đài Truyền thanh - Truyền hình huyện Lắk.

Dữ liệu được lưu trữ trên nền tảng lưu trữ của Hugging Face. Dữ liệu được chia làm 2 phần: cột “audio” chứa file audio tiếng M’Nông và cột “transcription” chứa bản dịch tiếng Việt của audio tương ứng. Bộ dữ liệu sẽ được chia thành các bộ nhỏ hơn, mang vai trò tương ứng như: “train”, “test”.



Hình 4.1. Bộ dữ liệu tiếng M’Nông được đẩy lên Hugging Face

Chi tiết bộ dữ liệu: 2926 file ghi âm, bao gồm 1338 file thu âm câu, 1588 file thu âm từ với tổng thời lượng là 2 giờ 34 phút. Bộ dữ liệu này mới chỉ gồm có 3 giọng đọc, thời lượng còn khá thấp, nhưng cũng vừa đủ để ta thử nghiệm phương án được đưa ra.



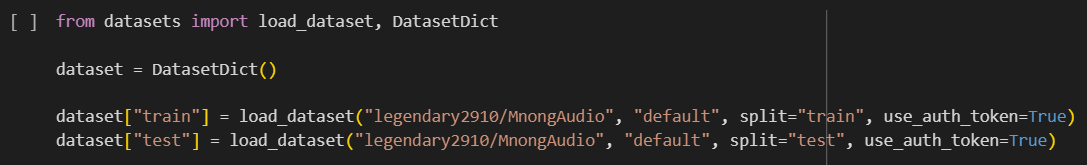
Hình 4.2. Phân bố độ dài của các file âm thanh trong bộ dữ liệu

## Cài đặt thực nghiệm

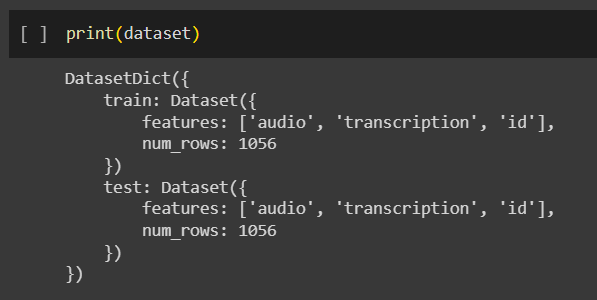
Những bước tiến hành cài đặt thực nghiệm sau đây dựa trên “Step-by-step Guide” được cung cấp bởi chính nhóm phát triển mô hình Whisper.

### Load Dataset

Đầu tiên, ta tải bộ dữ liệu đã được lưu trên Hugging Face Hub. Sau đó lấy phần “train” và “test” của bộ dữ liệu.



Hình 4.3. Load dataset



Hình 4.4. Kết quả của load dataset

### Chuẩn bị Tokenizer, Feature Extractor và xủ lý data

Quy trình ASR có thể được phân rã thành ba thành phần:

1. Một Feature Extractor có tác dụng tiền xử lý dữ liệu thô.
2. Một mô hình thực hiện nhiệm vụ sequence-to-sequence mapping.
3. Một Tokenizer thực hiện xử lý kết quả trả về thành văn bản.

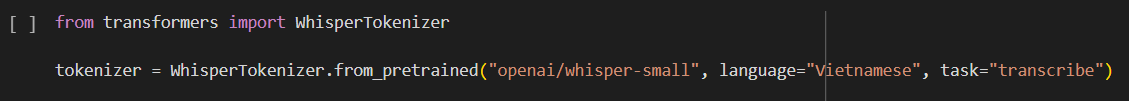
Trong Transformer, mô hình Whisper đi kèm với một bộ feature extractor và một bộ tokenizer lần lượt là WhisperFeatureExtractor và WhisperTokenizer.

### Load WhisperFeatureExtractor

Tốc độ của nguồn âm thanh đầu vào phải phù hợp với tốc độ lấy mẫu mà mô hình của chúng ta lựa chọn. Nếu lấy một mẫu âm thanh với tốc độ lấy mẫu là 16kHz và nghe nó với tốc độ lấy mẫu là 8kHz sẽ khiến âm thanh nghe như đang chạy ở nửa tốc độ. Tương tự, truyền âm thanh với tốc độ lấy mẫu không đúng có thể làm sai lệch mô hình ASR. Bộ feature extractor của Whisper cần đầu vào âm thanh với tốc độ lấy mẫu là 16kHz, vì vậy chúng ta cần tinh chỉnh đầu vào với dữ liệu theo giá trị này.

Bộ feature extractor của Whisper thực hiện hai hoạt động. Trước hết, nó chia độ dài của các mẫu âm thanh sao cho tất cả các mẫu có chiều dài đầu vào là 30 giây. Các mẫu ngắn hơn 30 giây được thêm độ dài bằng cách thêm các số không vào cuối chuỗi (số không trong tín hiệu âm thanh tương ứng với không có tín hiệu hoặc im lặng). Các mẫu dài hơn 30 giây được cắt bớt thành 30 giây.

Hoạt động thứ hai mà bộ feature extractor của Whisper thực hiện là chuyển đổi các mảng âm thanh đã được chia độ dài thành các biểu đồ tần số log-Mel.



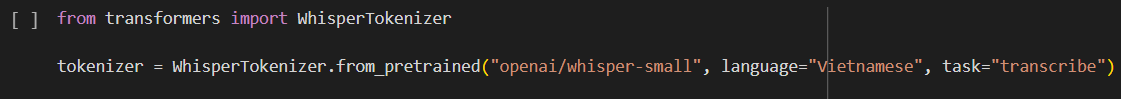
Hình 4.5. Load WhisperFeatureExtractor

### Load WhisperTokenizer

Mô hình Whisper xuất ra các token văn bản chỉ định chỉ số của văn bản dự đoán trong từ điển các mục từ vựng. Tokenizer ánh xạ một chuỗi các token văn bản thành chuỗi văn bản thực tế (ví dụ: [1169, 3797, 3332] -> "the cat sat").

Trong trước đây, khi sử dụng các mô hình chỉ có trình mã hóa cho ASR, chúng ta giải mã bằng cách sử dụng Connectionist Temporal Classification (CTC). Ở đây, chúng ta cần đào tạo một tokenizer CTC cho mỗi bộ dữ liệu chúng ta sử dụng. Một trong những ưu điểm của việc sử dụng kiến trúc mã hóa-giải mã là chúng ta có thể trực tiếp tận dụng tokenizer từ mô hình được đào tạo trước.

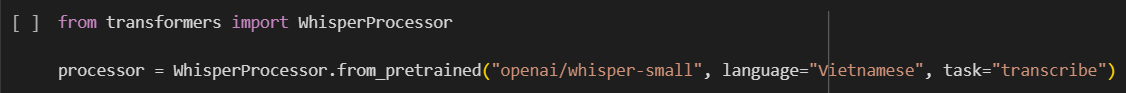
Tokenizer của Whisper được đào tạo trên các bản ghi chép cho 96 ngôn ngữ được đào tạo trước. Do đó, nó có một byte-pair mở rộ rất phù hợp cho hầu hết các ứng dụng ASR đa ngôn ngữ.



Hình 4.6. Load WhisperTokenizer

### Kết hợp để tạo ra WhisperProcessor

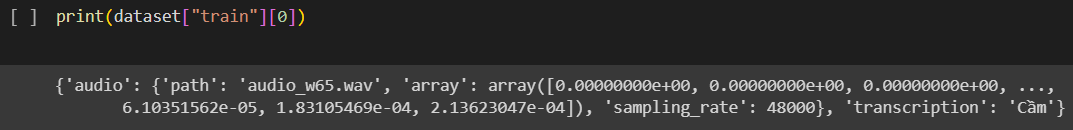
Để đơn giản hóa việc sử dụng bộ feature extractor và tokenizer, chúng ta có thể kết hợp cả hai vào một lớp duy nhất là WhisperProcessor. Đối tượng này kế thừa từ WhisperFeatureExtractor và WhisperProcessor và có thể được sử dụng trên đầu vào âm thanh và dự đoán của mô hình khi cần thiết.



Hình 4.7. Tạo WhisperProcessor

### Chuẩn bị dữ liệu

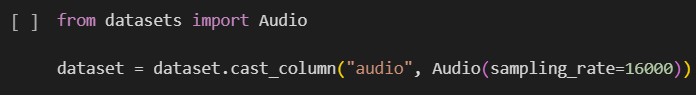
Bây giờ hãy in ra ví dụ đầu tiên trong bộ dữ liệu để xem dữ liệu ở dạng như thế nào.



Hình 4.8. In data đầu tiên

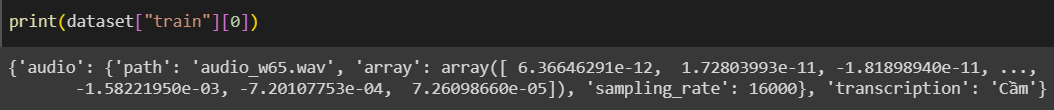
Có thể thấy rằng chúng ta có một mảng âm thanh đầu vào 1 chiều và bản dịch tương ứng. Chúng ta đã nói về tầm quan trọng của tốc độ lấy mẫu và thực tế là chúng ta cần phải điều chỉnh tốc độ lấy mẫu của âm thanh với tốc độ của mô hình Whisper (16kHz). Vì âm thanh đầu vào của chúng ta được lấy mẫu ở 48kHz, cần giảm mẫu xuống 16kHz trước khi truyền vào bộ feature extractor của Whisper.

Chúng ta sẽ đặt tốc độ lấy mẫu của đầu vào âm thanh đúng bằng cách sử dụng phương thức cast\_column của bộ dữ liệu. Thao tác này không thay đổi âm thanh ngay tại chỗ, mà thay vào đó báo hiệu cho bộ dữ liệu để resample các mẫu âm thanh trực tiếp lần đầu tiên chúng được tải.



Hình 4.9. Điều chỉnh tốc độ lấy mẫu

Việc tải lại mẫu âm thanh sẽ resample nó về tốc độ chúng ta mong muốn.

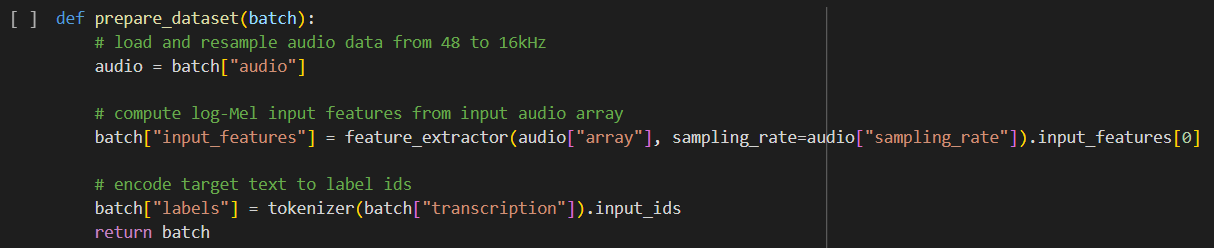


Hình 4.10. Resample mẫu âm thanh về tốc độ chuẩn

Có thể thấy rằng tốc độ lấy mẫu đã được giảm xuống còn 16kHz, giá trị mảng cũng khác nhau.

Bây giờ chúng ta viết một hàm để chuẩn bị dữ liệu sẵn sàng cho mô hình:

1. Tải và resample dữ liệu âm thanh bằng cách gọi batch[“audio”]
2. Sử dụng bộ feature extractor để tính các đặc trưng đầu vào log-Mel spectrogram từ mảng âm thanh 1 chiều.
3. Mã hóa transcriptions thành các nhãn thông qua tokenizer.



Hình 4.11. Hàm chuẩn bị dữ liệu

Áp dụng hàm chuẩn bị dữ liệu cho tất cả các ví dụ đào tạo bằng cách sử dụng phương thức .map của bộ dữ liệu.



Hình 4.12. Áp dụng hàm chuẩn bị dữ liệu

Đến đây chúng ta đã chuẩn bị đầy đủ dữ liệu cho quá trình huấn luyện.

### Huấn luyện và đánh giá

Để Trainer thực hiện các công việc liên quan đến huấn luyện, chúng ta cần:

1. Xác định data collator: Lấy dữ liệu đã được xử lý trước và chuẩn bị các tensor PyTorch.
2. Đo độ chính xác: Cần một hàm compute\_metrics để xử lý tính toán các chỉ số lỗi từ vựng (WER) để đánh giá mô hình.
3. Tải một pre-trained chechkpoint và cấu hình nó cho quá trình huấn luyện.
4. Xác định đối số được sử dụng bởi Trainer trong khi lên lịch huấn luyện.

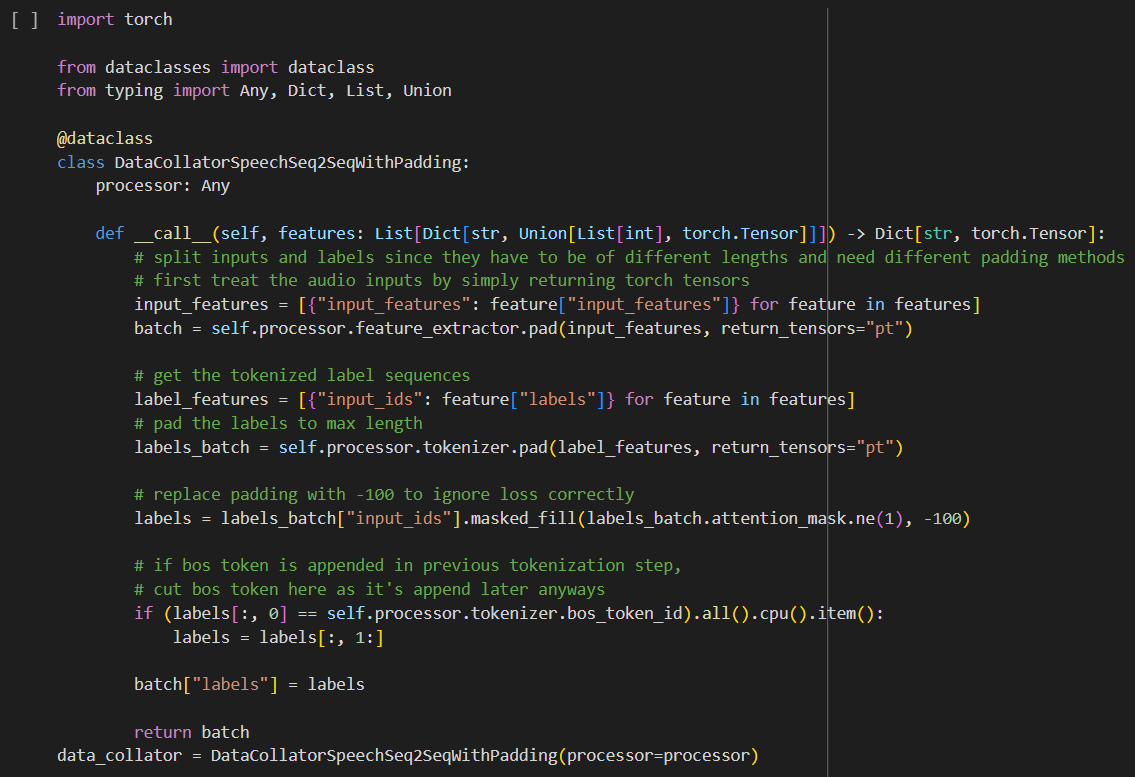
### Định nghĩa Data Collator

Bộ thu thập dữ liệu cho một mô hình nói chung sequence-to-sequence xử lý input\_features và nhãn một cách độc lập: input\_features phải được xử lý bởi bộ feature extractor và gán nhãn bởi tokenizer.

Các input\_features đã được chia độ dài thành 30 giây và chuyển đổi thành log-Mel spectrogram có kích thước cố định, vì thế những gì chúng ta cần phải làm là chuyển đổi chúng thành các tensor PyTorch. Chúng ta thực hiện điều này bằng cách sử dụng phương thức .pad của bộ feature extractor với return\_tensors=pt.

Ngược lại, các nhãn không được chia độ dài. Đầu tiên, chúng ta chia độ dài các chuỗi đến độ dài tối đa trong lô bằng cách sử dụng phương thức .pad của tokenizer. Các token sau đó được thay thế bằng -100 để những token này không được tính vào khi tính toán mất mát. Sau đó, chúng ta cắt bỏ token bắt đầu của bản trích từ đầu của chuỗi nhãn vì chúng ta sẽ thêm nó vào sau này trong quá trình huấn luyện.

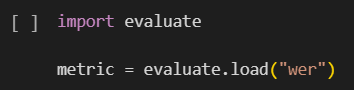
Chúng ta có thể tận dụng WhisperProcessor đã định nghĩa trước đó để thực hiện cả hai thao tác feature extractor và tokenizer. Sau đó khởi tạo bộ thu thập dữ liệu vừa định nghĩa.



Hình 4.13. Định nghĩa và khởi tạo Data Collator

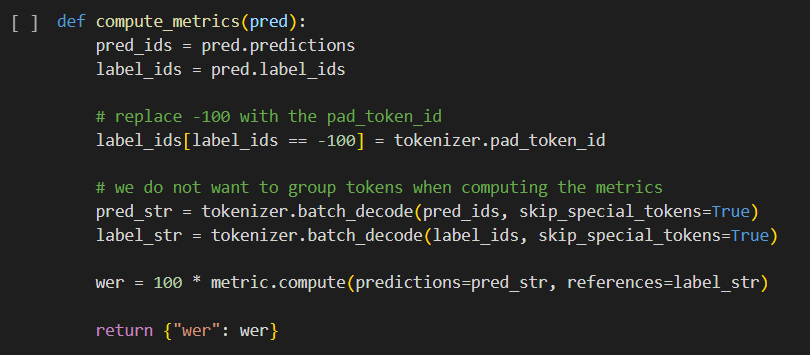
### Số liệu đánh giá

Tiếp theo, cần xác định chỉ số đánh giá mà chúng ta sẽ sử dụng trên tập đánh giá của mình. Ta sẽ sử dụng chỉ số Word Error Rate (WER), chỉ số 'de-facto' để đánh giá các hệ thống ASR.



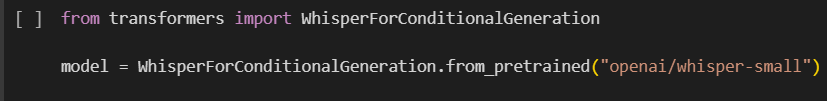
Hình 4.14. Xác định WER

Sau đó, chúng ta chỉ cần định nghĩa một hàm lấy dự đoán của mô hình và trả về chỉ số WER. Hàm này được gọi là compute\_metrics, đầu tiên thay thế -100 bằng pad\_token\_id trong label\_ids (hoàn ngược bước chúng ta áp dụng trong bộ thu thập dữ liệu để bỏ qua đúng các token). Sau đó, nó giải mã predicted và label ids thành chuỗi. Cuối cùng, nó tính toán chỉ số WER giữa các dự đoán và nhãn tham chiếu.



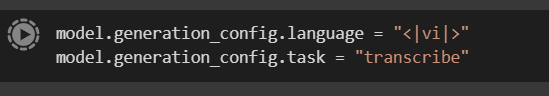
Hình 4.15. Hàm trả về WER

### Tải pre-trained checkpoint



Hình 4.16. Tải một pre-trained checkpoint

Mô hình Whisper có khả năng dự đoán những ngôn ngữ khác nhau xuất hiện trong câu được huấn luyện. Điều này khiến cho việc huấn luyện và dự đoán gặp khó khăn khi những từ mới của tiếng M’Nông xuất hiện có thể được mô hình dự đoán thành một ngôn ngữ khác. Nên ra ép mô hình phải sử dụng tiếng Việt và tác vụ phiên âm bằng việc cấu hình lại mô hình.



Hình 4.17. Cấu hình lại mô hình

### Định nghĩa Traing Arguments

Trong bước cuối cùng, chúng ta định nghĩa tất cả các tham số liên quan đến quá trình huấn luyện. Một phần của các tham số được giải thích dưới đây:

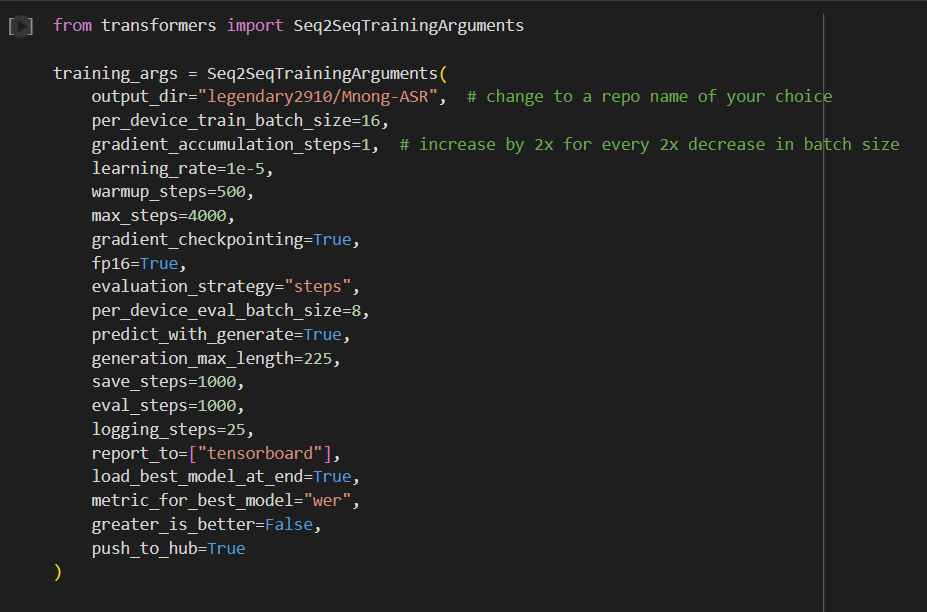
- `output\_dir`: thư mục cục bộ để lưu trữ trọng số của mô hình. Đây cũng sẽ là tên kho chứa trên Hugging Face Hub.

- `generation\_max\_length`: số lượng tối đa các token để tạo ra theo cách phát sinh tự động trong quá trình đánh giá.

- `save\_steps`: trong quá trình huấn luyện, các điểm kiểm tra trung gian sẽ được lưu và tải lên bất đồng bộ lên Hub mỗi save\_steps.

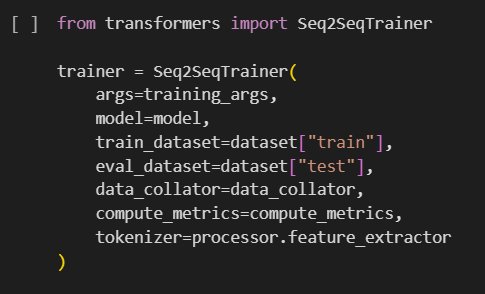
- `eval\_steps`: trong quá trình huấn luyện, đánh giá các điểm kiểm tra trung gian sẽ được thực hiện mỗi eval\_steps.

- `report\_to`: nơi lưu trữ nhật ký huấn luyện. Các nền tảng được hỗ trợ là "azure\_ml", "comet\_ml", "mlflow", "neptune", "tensorboard" và "wandb". Chọn nền tảng ưa thích của bạn hoặc để là "tensorboard" để đăng nhập vào Hub.



Hình 4.18. Ý nghĩa các tham số liên quan

Chúng ta có thể chuyển các đối số huấn luyện cho Trainer cùng với mô hình, bộ dữ liệu, bộ thu thập dữ liệu và hàm compute\_metrics.



Hình 4.19. Chuyển các đối số huấn luyện cho Trainer

Và với thao tác vừa rồi, mô hình đã sẵn sàng để huấn luyện.

### Huấn luyện mô hình

Với tổng gần 3000 dữ liệu thu được, nhóm sử dụng dữ liệu trên để huấn luyện và trích ra 8% tổng dữ liệu (khoảng 230 dữ liệu) cho việc đánh giá thông qua phương pháp WER (sẽ đề cập trong phần tiếp theo).

Việc huấn luyện được thực hiện trên Google Colab trong tổng thời gian 3 tiếng, với các thông số kỹ thuật như sau:

- CPU: Intel(R) Xeon(R) CPU @ 2.00GHz.

- GPU: Tesla V100-SXM2-16GB.

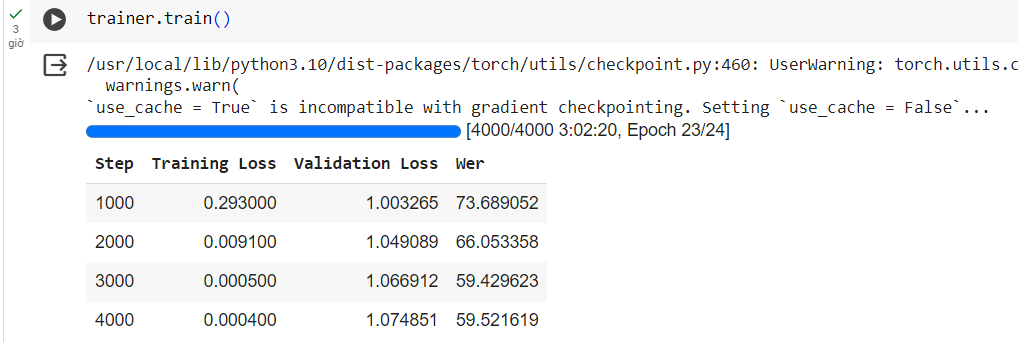
- RAM: 52GB.

Sau khi huấn luyện xong, mô hình đã được huấn luyện sẽ được đẩy lên Huggingface Hub cho việc sử dụng sau này dưới tên “legendary2910/MNong-ASR”.

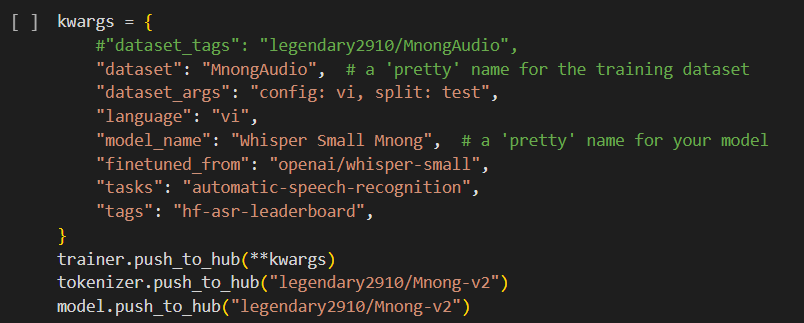


Hình 4.20. Huấn luyện mô hình

Output:



Hình 4.21. Kết quả huấn luyện





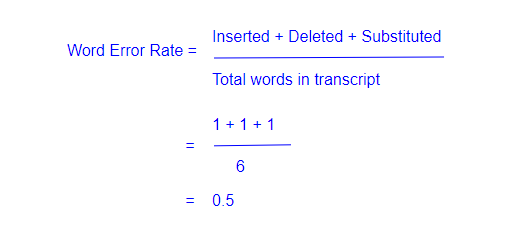
Hình 4.22. Đưa kết quả lên Hugging Face Hub

## Đánh giá

Trong hệ thống ASR thường sử dụng metric là Word Error Rate (WER) cho bài toán này. WER so sánh đầu ra dự đoán với transcript mục tiêu, từng từ hoặc từng ký tự với ký tự để phát hiện sự khác biệt giữa hai câu.

Sự khác biệt này có thể bao gồm các từ có trong transcript nhưng không được dự đoán (Deletion), một từ không có trong transcript nhưng lại được thêm vào dự đoán (Insertion), hoặc một từ được thay thế giữa dự đoán và transcript (Substitution).

Công thức tính metric đo tỉ lệ của số từ khác biệt giữa transcript và dự đoán.

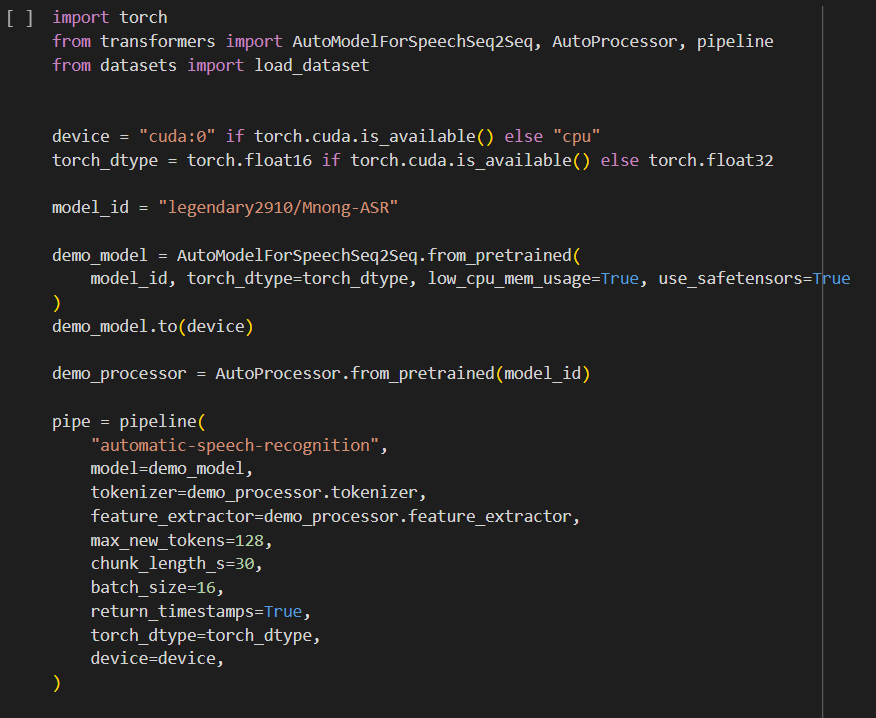


Hình 4.23. Công thức tính WER

Hiện tại có thể thấy ở Hình 4.21, kết quả huấn luyện mô hình còn chưa tốt với WER khá cao. Nguyên nhân có thể đến từ việc dữ liệu dù đa dạng, nhưng không có quá nhiều giọng đọc cho cùng một câu, và tất cả các dữ liệu đều khá khác nhau.

## Thực nghiệm và kết quả

Vì mô hình đã được lưu lại trên Huggingface Hub, ta có thể tạo ra một ví dụ đơn giản bằng Pipeline cho việc sử dụng mô hình trong tương lai. Bằng cách này, ta có thể sử dụng mô hình để thực hiện phiên dịch những file âm thanh trong bộ dữ liệu sử dụng để đánh giá. Qua đó có thể quan sát được kết quả ra sao trong quá trình huấn luyện.



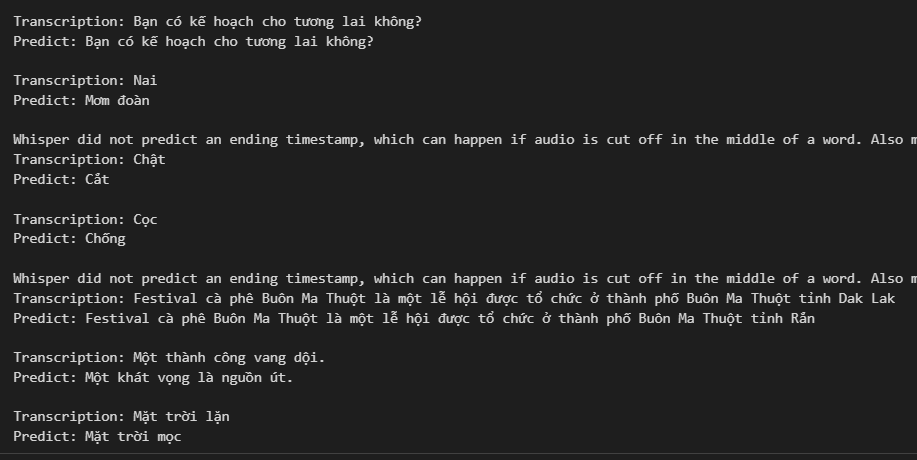
Hình 4.24. Ví dụ đơn giản sử dụng mô hình đã qua huấn luyện



Hình 4.25. Kết quả khi thực hiện kiểm tra trên bộ dữ liệu sử dụng để kiểm định

Ở phần chụp lại kết quả trên, phần “Transcription” hiển thị cho phần dịch đúng tương ứng với file âm thanh, phần “Predict” hiển thị kết quả mà mô hình đã dự đoán ý nghĩa của câu nói có trong file âm thanh. Ta có thể thấy mô hình dự đoán những từ ta đã sử dụng trong quá trình huấn luyện với độ chính xác cao.

Vì sự hạn chế của bộ dữ liệu, vẫn có những từ ngữ mà mô hình chưa được học xuất hiện trong bộ dữ liệu đánh giá, khiến cho kết quả dự đoán bị sai đi hoặc sai hoàn toàn.



Hình 4.26. Kết quả khi thực hiện kiểm tra trên bộ dữ liệu sử dụng để kiểm định

# KẾT LUẬN

Đây là một đề tài mới lạ đối với nhóm, yêu cầu nhiều sự tâm huyết cũng như thời gian dành cho đề tài. Việc phiên dịch tiếng nói của những dân tộc ít người có thể giúp cho công tác giáo dục tiếng nói và trong đời sống sinh hoạt hàng ngày của người dân tộc M’Nông nói riêng và người dân tộc thiểu số nói chung được thuận tiện hơn. Việc chưa hề có bất kì dự án nào tương tự cũng gây nên không ít khó khăn cho nhóm khi thực hiện đề tài.

Qua việc huấn luyện trên, ta có thể thấy được bất cập của bộ dữ liệu ảnh hưởng ít nhiều đến chất lượng của kết quả ra sao. Bộ dữ liệu chỉ có sự tham gia của 3 giọng đọc, số lượng dữ liệu dạng từ vựng lại chiếm quá nhiều trong bộ dữ liệu; những câu dài được sử dụng để phiên dịch sang tiếng M’Nông cho quá trình thu âm lại quá phức tạp cho một ngôn ngữ thô sơ như tiếng M’Nông, làm cho việc phiên dịch trở nên khó khăn. Những lý do trên đã khiến cho kết quả của mô hình không được như mong đợi.

Ngoài ra, về mặt kỹ thuật cũng xảy ra rất nhiều vấn đề trong quá trình huấn luyện. Từ xử lý việc mô hình tự động phát hiện đa ngôn ngữ, cho đến cấu hình lại mô hình, lưu trữ mô hình sau khi huấn luyện,…

Tuy có nhiều khó khăn như thế, xong kết quả có được cũng có một vài dấu hiệu khả quan, như việc mô hình đã có thể dịch được khá sát những từ vựng đã được sử dụng trong quá trình huấn luyện. Đây sẽ là nền tảng để nhóm tiếp tục khắc phục những nhược điểm trên nhằm cải thiện kết quả, cũng như hy vọng vào việc đề tài này có thể đem lại nhiều những giá trị thực tiễn hơn để phục vụ cộng đồng, phục vụ đất nước.

# TÀI LIỆU THAM KHẢO

Tiếng Việt

…

Tiếng Anh

[1] Alcorn, M. A., Li, Q., Gong, Z., Wang, C., Mai, L., Ku, W.S., and Nguyen, A. Strike (with) a pose: Neural networks are easily fooled by strange poses of familiar objects. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp.4845–4854, 2019.

[2] Child, R., Gray, S., Radford, A., and Sutskever, I. Generating long sequences with sparse transformers. arXiv preprint arXiv:1904.10509, 2019.

[3] Seide, F., Li, G., Chen, X., and Yu, D. Feature engineering in context dependent deep neural networks for conversational speech transcription. In 2011 IEEE Workshop on Automatic Speech Recognition & Understanding, pp.24–29. IEEE, 2011.

[4] Towards Datascience “Foundations of NLP Explained BLEU Score and WER Metrics”.

[5] Ketan Doshi, “Audio Deep Learning Made Simple: Automatic Speech Recognition (ASR), How it Works”.

[6] Scott Duda, “Urban Environmental Audio Classification Using Mel Spectrograms”.