**TRƯỜNG ĐẠI HỌC BÁCH KHOA HÀ NỘI**

**HÀ NỘI, 5/2025**

**BÁO CÁO   
TẠO SINH ÂM THANH**

**ASR – Nhận diện giọng nói**

|  |  |  |
| --- | --- | --- |
| **Họ và tên** | **MSSV** | **Email** |
| **Dương Tiến Hoàng** | **20240888e** | **Hoang.dt240888e@gmail.com** |
| **Nguyễn Tuấn Anh** | **20241171e** | **Anh.nt241171e@gmail.com** |
| **Nguyễn Duy Khánh Linh** | **20240821e** | **Linh.ndk240821e@gmail.com** |
| **Trần Quốc Hùng** | **20240486e** | **Hung.tq240486@gmail.com** |

**Ngành Khoa học máy tính**

|  |  |
| --- | --- |
| **Giảng viên hướng dẫn:** | TS. Nguyễn Thị Thu Trang  Chữ ký của GVHD |
| **Bộ môn:** | Khoa học máy tính |
| **Trường:** | Công nghệ Thông tin và Truyền thông |

**LỜI CẢM ƠN**

Lời đầu tiên, chúng em xin gửi lời cảm ơn chân thành và sâu sắc nhất tới cô Nguyễn Thị Thu Trang, giảng viên phụ trách học phần Tạo sinh âm thanh (IT5428). Trong suốt quá trình học tập và thực hiện đề tài, cô đã luôn đồng hành cùng chúng em với sự tận tình chỉ dẫn, tinh thần trách nhiệm cao và tâm huyết trong từng buổi giảng. Sự hỗ trợ của cô không chỉ giúp chúng em hiểu sâu sắc về các kiến thức lý thuyết, mà còn tạo điều kiện để chúng em vận dụng hiệu quả trong bài tập lớn lần này.

Với những kiến thức nền tảng mà cô truyền đạt, đặc biệt là trong lĩnh vực nhận diện giọng nói (ASR), chúng em đã có cơ hội được tiếp cận với những mô hình tiên tiến như Wav2Vec2, cũng như làm quen với công cụ và thư viện lập trình như Hugging Face và các kỹ thuật xử lý âm thanh thực tiễn. Không những thế, cô còn luôn khuyến khích và truyền cảm hứng để chúng em phát huy khả năng nghiên cứu độc lập, tinh thần làm việc nhóm và ý thức trách nhiệm đối với đề tài.

Bản báo cáo này là kết quả của sự nỗ lực chung và cũng là minh chứng cho tinh thần học tập nghiêm túc mà cô Trang đã truyền cảm hứng cho chúng em. Dù còn những thiếu sót không thể tránh khỏi, chúng em tin rằng những gì đạt được là nền tảng vững chắc để tiếp tục phát triển hơn nữa trong các môn học và dự án sau này.

Một lần nữa, chúng em xin kính chúc cô Nguyễn Thị Thu Trang luôn mạnh khỏe, hạnh phúc và thành công hơn nữa trên con đường giảng dạy, đồng hành cùng nhiều thế hệ sinh viên đạt được những thành tựu trong học tập và nghiên cứu.

**Mục lục**

[1. Đặt vấn đề 4](#_Toc198452425)

[2. Mục tiêu 4](#_Toc198452426)

[3. Bộ dữ liệu 5](#_Toc198452427)

[3.1. Thu thập dữ liệu 5](#_Toc198452428)

[3.2. Phân tích dữ liệu 5](#_Toc198452429)

[3.2.1. Phương pháp thống kê 5](#_Toc198452430)

[3.2.2. Phương pháp truyền thống 6](#_Toc198452431)

[3.3. Tiền xử lý tín hiệu 7](#_Toc198452432)

[3.3.1. Xử lý tín hiệu âm thanh 7](#_Toc198452433)

[3.3.2. Xử lý văn bản lời thoại 7](#_Toc198452434)

[3.3.3. Xây dựng bộ ký tự (vocab) 8](#_Toc198452435)

[3.4. Trích xuất đặc trưng âm thanh 8](#_Toc198452436)

[3.5. Chiến lược xử lý dữ liệu 9](#_Toc198452437)

[4. Tổng quan về phương pháp huấn luyện 11](#_Toc198452438)

[4.1. Mô hình Wav2Vec2 và kiến trúc tổng thể 11](#_Toc198452439)

[4.2. Xử lý và chuẩn bị dữ liệu âm thanh đầu vào cho mô hình 13](#_Toc198452440)

[4.3. Kỹ thuật padding dữ liệu và xử lý batch 13](#_Toc198452441)

[4.4. Cấu hình và huấn luyện mô hình 14](#_Toc198452442)

[5. Đánh giá mô hình 19](#_Toc198452443)

[5.1. Word Error Rate (WER) 19](#_Toc198452444)

[5.2. Tập dữ liệu kiểm thử 19](#_Toc198452445)

[5.3. Kết quả và phân tích 19](#_Toc198452446)

[6. Kết luận và hướng phát triển 20](#_Toc198452447)

[6.1. Kết luận 20](#_Toc198452448)

[6.2. Khó khăn và thách thức 20](#_Toc198452449)

[6.3. Hướng phát triển 20](#_Toc198452450)

[7. Phân công công việc 21](#_Toc198452451)

# 1. Đặt vấn đề

Trong những năm gần đây, công nghệ nhận dạng tiếng nói tự động (ASR – Automatic Speech Recognition) đã trở thành một trong những lĩnh vực then chốt trong trí tuệ nhân tạo, với nhiều ứng dụng thực tiễn như trợ lý ảo, chuyển lời nói thành văn bản, điều khiển thiết bị bằng giọng nói và hỗ trợ người khuyết tật. Đặc biệt, ASR góp phần quan trọng trong việc nâng cao khả năng tiếp cận công nghệ cho người khiếm thị nhờ vào tương tác bằng giọng nói thay cho thao tác trực quan.

Tuy nhiên, phần lớn các hệ thống ASR hiện nay vẫn tập trung chủ yếu vào các ngôn ngữ phổ biến như tiếng Anh hoặc tiếng Trung, trong khi những ngôn ngữ ít tài nguyên như tiếng Việt còn gặp nhiều hạn chế về dữ liệu huấn luyện, mô hình tối ưu và độ chính xác khi nhận dạng. Thêm vào đó, tiếng Việt là ngôn ngữ đơn âm có dấu, phụ thuộc nhiều vào ngữ điệu, và có đặc điểm âm vị học khác biệt, khiến việc áp dụng trực tiếp các mô hình ASR phổ biến trở nên không hiệu quả.

Trong bối cảnh đó, mô hình Wav2Vec 2.0 do Facebook AI phát triển đã mở ra hướng tiếp cận mới với khả năng học đặc trưng âm thanh theo phương pháp tự giám sát (self-supervised learning), cho phép huấn luyện hiệu quả ngay cả khi chỉ có lượng nhỏ dữ liệu gán nhãn. Đây là một cơ hội lớn cho việc xây dựng hệ thống ASR tiếng Việt chất lượng cao trong điều kiện hạn chế về tài nguyên.

Do vậy, việc tập trung nghiên cứu và đánh giá hiệu quả của mô hình Wav2Vec 2.0 trong bài toán nhận dạng tiếng nói tiếng Việt là cần thiết, nhằm góp phần cải thiện độ chính xác của ASR cho ngôn ngữ này, đồng thời khám phá tiềm năng áp dụng của các mô hình học tự giám sát trong bối cảnh đa ngôn ngữ và ít dữ liệu.

# 2. Mục tiêu

Mục tiêu của đề tài là xây dựng một hệ thống nhận dạng tiếng nói tiếng Việt hiệu quả dựa trên mô hình Wav2Vec 2.0, thông qua quy trình xử lý và đánh giá dữ liệu bài bản. Trước tiên, nghiên cứu tập trung vào việc làm sạch dữ liệu đầu vào, bao gồm các bản ghi âm tiếng Việt và văn bản chuyển lời tương ứng, nhằm loại bỏ các mẫu bị nhiễu, không đồng bộ hoặc transcript không đầy đủ, vốn có thể ảnh hưởng tiêu cực đến quá trình huấn luyện mô hình. Bên cạnh đó, nghiên cứu sẽ đánh giá mức độ đa dạng và quy mô của tập dữ liệu để xác định khả năng sử dụng trong bài toán huấn luyện ASR, đồng thời chuyển đổi dữ liệu sang định dạng phù hợp với mô hình Wav2Vec 2.0. Tiếp theo, đề tài hướng đến xây dựng chiến lược huấn luyện mô hình hiệu quả trên các nền tảng miễn phí như Google Colab và Kaggle, thông qua các kỹ thuật tối ưu nhằm giảm thiểu tài nguyên tính toán mà vẫn đạt được kết quả chính xác. Cuối cùng, nghiên cứu sẽ tiến hành kiểm thử mô hình trên các tập dữ liệu tiếng Việt công khai, sử dụng các chỉ số như Word Error Rate để đánh giá hiệu suất, từ đó phân tích lỗi và đề xuất các hướng cải tiến mô hình trong tương lai.

# 3. Bộ dữ liệu

## 3.1. Thu thập dữ liệu

Dữ liệu huấn luyện và đánh giá được cung cấp bởinhóm nghiên cứu Vbee. Bộ dữ liệu gồm các 56427 audio tiếng việt và bản ghi chép tương ứng với tổng cộng hơn 100 giờ tiếng nói.

Tập dữ liệu được xây dựng từ các cặp tệp âm thanh định dạng .wav và các tệp văn bản lời thoại tương ứng định dạng .txt. Dữ liệu được tổ chức trong các thư mục phân cấp, phản ánh cấu trúc ngữ liệu thu thập từ nhiều nguồn khác nhau. Trong quá trình xây dựng tập huấn luyện, một bước kiểm tra được thực hiện nhằm đảm bảo mỗi tệp âm thanh đều có một bản transcript tương ứng. Các cặp thiếu một trong hai thành phần, hoặc không tuân thủ định dạng đặt tên đồng nhất, sẽ bị loại bỏ để đảm bảo tính toàn vẹn dữ liệu.

Sau quá trình lọc sơ bộ, tập dữ liệu được biểu diễn dưới dạng cấu trúc có tổ chức gồm hai thành phần chính: đường dẫn âm thanh, nội dung lời thoại, và các thuộc tính liên quan khác phục vụ quá trình xử lý tín hiệu và huấn luyện mô hình. Cấu trúc này tạo điều kiện thuận lợi cho việc mở rộng, kiểm thử, và tích hợp với các pipeline học máy hiện đại.

## 3.2. Phân tích dữ liệu

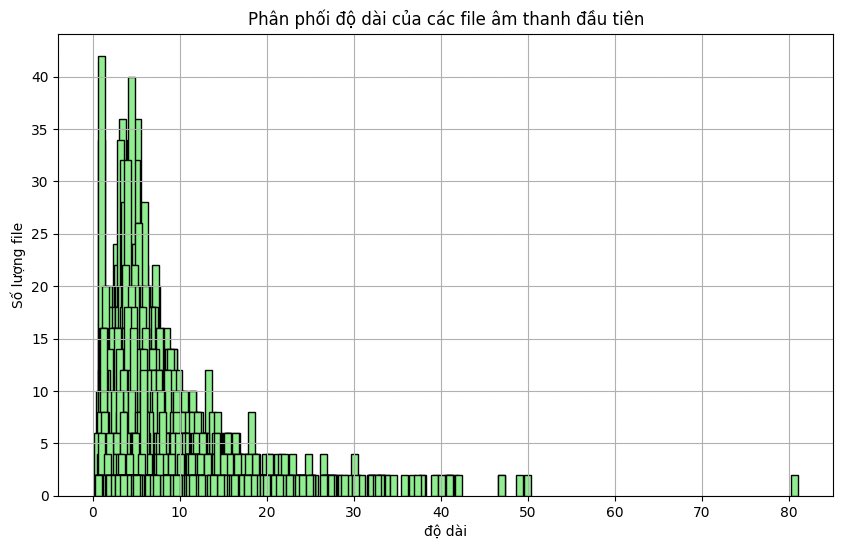
Chất lượng và đặc điểm của tập dữ liệu đóng vai trò then chốt trong hiệu quả huấn luyện và khả năng tổng quát hóa của bất kỳ hệ thống nhận diện giọng nói tự động (ASR) nào. Do đó, trước khi tiến hành huấn luyện mô hình, việc phân tích dữ liệu một cách toàn diện là cần thiết để đánh giá các khía cạnh phân bố, tính đa dạng, cũng như độ sạch và sự đồng nhất của ngữ liệu. Trong nghiên cứu này, dữ liệu được phân tích theo hai hướng tiếp cận chính: (a) phương pháp thống kê – cung cấp góc nhìn định lượng và khách quan thông qua các chỉ số đo lường; và (b) phương pháp truyền thống – cho phép kiểm tra định tính thông qua đánh giá trực tiếp trên mẫu dữ liệu cụ thể..

### 3.2.1. Phương pháp thống kê

Phân tích thống kê được tiến hành trên toàn bộ tập dữ liệu huấn luyện nhằm xác định các đặc điểm cơ bản của dữ liệu âm thanh và văn bản đi kèm, bao gồm độ dài, tần suất lấy mẫu và sự phân bố từ vựng. Các kết quả phân tích này không chỉ giúp nhận diện các bất thường có thể ảnh hưởng đến hiệu quả huấn luyện, mà còn định hướng các bước xử lý tiền huấn luyện như lọc, chuẩn hóa, tăng cường dữ liệu và lựa chọn kiến trúc mô hình phù hợp.

**\* Phân tích phân phối độ dài đoạn ghi âm**

Thời lượng của các đoạn ghi âm được thống kê và trực quan hóa dưới dạng biểu đồ tần suất. Kết quả cho thấy phân phối độ dài khá rộng, với phần lớn các đoạn rơi vào khoảng 2–20 giây. Tuy nhiên, tồn tại một tỷ lệ nhỏ các đoạn quá ngắn (<1 giây) hoặc quá dài (>40 giây). Điều này cho thấy sự cần thiết của việc áp dụng các ngưỡng lọc hoặc chia nhỏ các đoạn dài nhằm tối ưu hóa cho mô hình học sâu, vốn nhạy cảm với độ dài input không đồng nhất. Việc xác định ngưỡng phù hợp giúp tăng tính ổn định và hiệu quả huấn luyện.

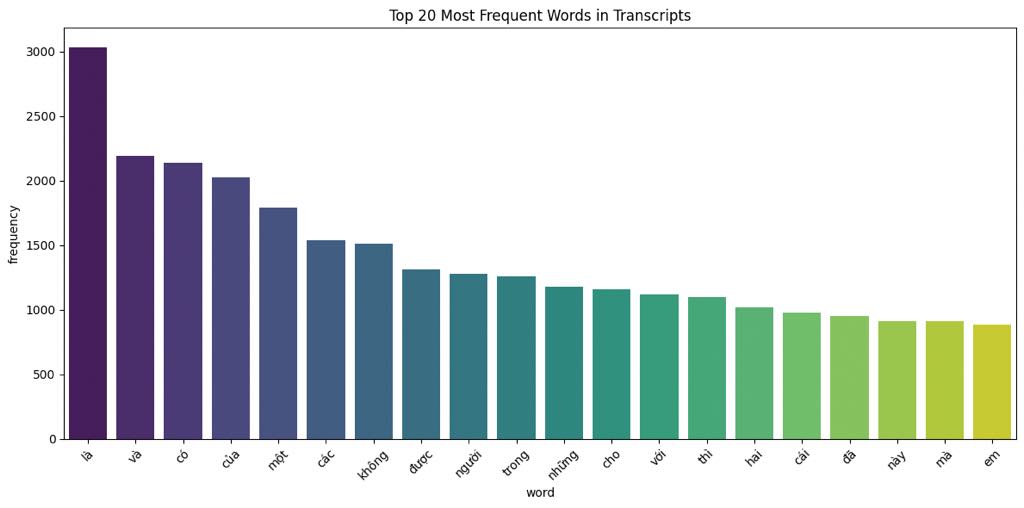


**\* Phân tích phân phối tần suất lấy mẫu ban đầu của đoạn ghi âm**

Sampling rate là một yếu tố kỹ thuật quan trọng ảnh hưởng đến chất lượng tín hiệu âm thanh và khả năng xử lý của hệ thống. Kết quả phân tích cho thấy phần lớn các đoạn ghi âm được thu ở tần suất chuẩn là 16 kHz, đây cũng là mức sampling rate phổ biến và tương thích với hầu hết các mô hình ASR hiện đại. Tuy nhiên, vẫn tồn tại một tỉ lệ nhỏ các đoạn sử dụng tần suất không chuẩn như 8 kHz, 22.05 kHz hoặc 44.1 kHz. Sự không đồng nhất này có thể gây lỗi hoặc làm suy giảm hiệu quả tiền xử lý, đặc biệt trong các bước trích xuất đặc trưng âm thanh như MFCC hoặc log-Mel spectrogram. Để đảm bảo tính nhất quán và giảm thiểu sai số huấn luyện, toàn bộ dữ liệu cần được chuẩn hóa sampling rate về cùng một giá trị, đồng thời áp dụng các thuật toán nội suy (resampling) phù hợp nhằm hạn chế mất mát thông tin.

**\* Phân tích phân phối từ vựng**

Tập transcript đi kèm được tổng hợp và phân tích từ vựng để đánh giá độ bao phủ ngôn ngữ, tính đa dạng và sự cân bằng của tập dữ liệu. Kết quả cho thấy phân phối tần suất từ vựng tuân theo định luật Zipf: một số lượng nhỏ từ rất phổ biến (như “và”, “là”, “của”, “có”,...) xuất hiện với tần suất cao, trong khi phần lớn các từ còn lại chỉ xuất hiện một vài lần. Sự mất cân bằng này dễ dẫn đến hiện tượng mô hình thiên lệch về các từ phổ biến, khiến khả năng nhận diện từ hiếm kém hiệu quả hơn. Các biện pháp được đề xuất bao gồm: (1) tăng cường các mẫu chứa từ hiếm bằng cách tạo thêm dữ liệu tổng hợp; (2) sử dụng phương pháp token hóa theo đơn vị con từ (subword units) như BPE hoặc unigram để giảm kích thước từ điển và cải thiện khả năng tổng quát hóa; và (3) điều chỉnh hàm mất mát trong huấn luyện bằng trọng số từ vựng để cân bằng ảnh hưởng của các từ có tần suất thấp.



### 3.2.2. Phương pháp truyền thống

Bên cạnh các phân tích định lượng, dữ liệu cũng được đánh giá trực quan thông qua việc chọn mẫu ngẫu nhiên để nghe trực tiếp các đoạn audio và so sánh với transcript tương ứng. Phương pháp này giúp phát hiện những vấn đề mà các công cụ tự động khó phát hiện, bao gồm:

**\* Môi trường thu âm và người nói**

Dữ liệu ghi âm thể hiện sự đa dạng đáng kể về người đọc (giọng Bắc, Trung, Nam, nhiều độ tuổi khác nhau), ngữ điệu và ngữ cảnh thu âm. Một số đoạn được thu trong môi trường yên tĩnh, trong khi nhiều đoạn có tạp âm (âm thanh đường phố, tiếng quạt, tiếng nhạc nền, v.v.), hoặc được ghi ở các không gian khác nhau như trong nhà, ngoài trời. Nội dung audio cũng phong phú, bao gồm các thể loại như đọc văn bản, hội thoại, vlog, bản tin thời sự, phim truyền hình,... phản ánh tính đa chiều của ngữ liệu.

**\* Transcript và từ vựng**

Một số transcript xuất hiện các từ ngữ không thuần Việt, ví dụ như “iu-to-be”, “breakdown”,... Những từ này cần được xem xét để quyết định giữ nguyên, phiên âm, hoặc loại bỏ, tùy theo mục tiêu ứng dụng và kỹ thuật xử lý ngôn ngữ. Đồng thời, các transcript còn xuất hiện một số ký tự đặc biệt như chấm, phẩy, hỏi chấm, ... Những ký tự này sẽ được loại bỏ nhằm giúp mô hình dễ học. Ngoài ra, transcript có độ dài không đồng đều và đôi khi không đồng bộ với nội dung audio.

**\* Vấn đề đồng bộ audio-transcript**

Trong một số trường hợp, phát hiện transcript dài hơn nội dung audio thực tế. Nguyên nhân có thể do thao tác xử lý cắt audio bị sớm, dẫn đến mất một phần âm thanh ở cuối đoạn ghi. Điều này có thể ảnh hưởng tới độ chính xác của mô hình trong giai đoạn huấn luyện và cần được điều chỉnh thông qua quy trình đồng bộ dữ liệu nghiêm ngặt hơn.

## 3.3. Tiền xử lý tín hiệu

Để đảm bảo chất lượng và tính nhất quán của dữ liệu đầu vào, nghiên cứu tiến hành đồng thời các bước xử lý đối với cả tín hiệu âm thanh và văn bản lời thoại. Quá trình này nhằm chuẩn hóa dữ liệu đầu vào, giảm thiểu nhiễu, và loại bỏ các yếu tố gây sai lệch trong quá trình huấn luyện mô hình nhận dạng tiếng nói.

### 3.3.1. Xử lý tín hiệu âm thanh

Các đoạn ghi âm được chuẩn hóa về mặt kỹ thuật bằng cách chuyển đổi toàn bộ dữ liệu âm thanh về tần số lấy mẫu 16kHz – một thông số tiêu chuẩn trong huấn luyện các mô hình học sâu về tiếng nói. Các waveform được trích xuất và lưu trữ dưới dạng mảng số thực một chiều, cho phép tính toán trực tiếp các đặc trưng liên quan đến thời gian và năng lượng.

Đồng thời, độ dài của từng đoạn ghi âm được tính toán để phục vụ cho phân tích thống kê cũng như làm tiêu chí lọc. Những đoạn ghi âm quá ngắn (dưới 1 giây) hoặc quá dài (trên 15 giây) có thể ảnh hưởng tiêu cực đến quá trình học, do thiếu ngữ cảnh hoặc gây quá tải bộ nhớ. Do đó, các đoạn không đạt yêu cầu về mặt thời lượng sẽ được loại bỏ khỏi tập dữ liệu.

### 3.3.2. Xử lý văn bản lời thoại

Văn bản lời thoại (transcript) đi kèm audio là một phần không thể thiếu trong tập dữ liệu ASR. Để đảm bảo tính nhất quán, khả năng phân tích và huấn luyện hiệu quả, nhóm nghiên cứu đã áp dụng một quy trình chuẩn hóa toàn diện đối với các transcript. Cụ thể, các bước chính bao gồm:

* **Chuyển toàn bộ văn bản sang chữ thường**: Điều này giúp đơn giản hóa không gian tìm kiếm và loại bỏ sự khác biệt hình thức giữa các từ (ví dụ: “Tôi” và “tôi” sẽ được coi là một đơn vị từ vựng).
* **Loại bỏ các ký tự đặc biệt, dấu câu và biểu tượng phi ngôn ngữ**: Bao gồm các dấu chấm, dấu phẩy, ngoặc đơn, biểu tượng cảm xúc hoặc ký tự không chuẩn. Việc này giúp giảm nhiễu trong quá trình token hóa và xây dựng từ điển ngữ âm.
* **Chuẩn hóa khoảng trắng và định dạng câu**: Đảm bảo mỗi câu sau khi làm sạch có khoảng trắng thích hợp, hỗ trợ hiệu quả cho các bước phân tách từ (tokenization) hoặc các kỹ thuật subword như BPE.
* **Xử lý các từ ngoại lai và ký hiệu đặc thù**: Một số từ nước ngoài như “breakdown”, “iu-to-be”,... hoặc tên người nước ngoài được xử lý theo hướng giữ nguyên nhằm đảm bảo sự đa dạng của dữ liệu.

Song song với việc làm sạch, các transcript cũng được đối chiếu thủ công với audio tương ứng, đặc biệt ở các đoạn bị nghi ngờ mất đồng bộ. Tuy nhiên vì quá trình đối chiều thủ công tốn quá nhiều công sức và thời gian, trong khi các audio và transcript có độ dài không tương đồng xuất hiện rất ít, nên nhóm vẫn mặc định các cặp dữ liệu này là hợp lệ.

### 3.3.3. Xây dựng bộ ký tự (vocab)

Trong quá trình huấn luyện mô hình nhận dạng tiếng nói tự động (ASR), đặc biệt là khi sử dụng các kiến trúc áp dụng hàm mất mát CTC (Connectionist Temporal Classification), việc xây dựng tập ký tự đầu ra (vocabulary) là một bước quan trọng. Khác với các mô hình dựa trên từ (word-level), hệ thống CTC thường xử lý ở cấp ký tự, điều này cho phép mô hình linh hoạt hơn trong việc xử lý các ngôn ngữ đa hình thái như tiếng Việt, nơi từ vựng có thể thay đổi đa dạng theo ngữ cảnh và cách viết.

Trước tiên, toàn bộ các tập văn bản transcript trong dữ liệu được tổng hợp lại và trích xuất danh sách tất cả các ký tự xuất hiện. Việc này nhằm đảm bảo rằng mô hình có thể nhận diện đầy đủ mọi đơn vị âm vị có trong tập dữ liệu, bao gồm cả ký tự dấu, chữ cái có dấu và các ký tự đặc trưng khác của ngôn ngữ. Sau đó, từ danh sách ký tự thu được, một từ điển ánh xạ giữa từng ký tự và một chỉ số định danh duy nhất được xây dựng, giúp phục vụ quá trình mã hóa văn bản thành các chuỗi số trước khi đưa vào mô hình.

Để tăng tính khái quát và khả năng xử lý của mô hình, tập từ điển ký tự còn được mở rộng thêm với các token đặc biệt. Cụ thể, ký tự "|" được sử dụng để biểu diễn khoảng trắng (space), nhằm làm rõ ranh giới giữa các từ trong chuỗi đầu ra – một yếu tố quan trọng giúp mô hình tái hiện đúng cấu trúc ngôn ngữ tự nhiên. Ngoài ra, token "[UNK]" (unknown) được thêm vào nhằm xử lý các trường hợp mô hình gặp phải ký tự không có trong tập huấn luyện, chẳng hạn như các ký hiệu lạ, từ ngoại lai, hay lỗi đánh máy. Đây là cơ chế dự phòng giúp mô hình không bị gián đoạn khi gặp dữ liệu ngoài phân phối.

Tiếp theo, token "[PAD]" được sử dụng để làm đầy (padding) các chuỗi ký tự đầu ra trong cùng một batch huấn luyện, giúp đồng bộ độ dài và tăng hiệu quả tính toán khi huấn luyện mô hình theo lô (mini-batch). Cuối cùng, một thành phần không thể thiếu trong kiến trúc CTC là token "[BLANK]", hay còn gọi là token “rỗng”. Token này đại diện cho các khoảng trống giữa các ký tự trong quá trình mô hình sắp xếp đầu ra. Trong quá trình giải mã, các ký tự trùng lặp liên tiếp được phân tách bởi token rỗng này sẽ được gộp lại thành một ký tự duy nhất, cho phép mô hình tạo ra chuỗi đầu ra mượt mà và chính xác. Ví dụ, đầu ra "hh\_\_e\_ll\_ll\_\_o" (với ký tự gạch dưới đại diện cho BLANK) sẽ được giải mã thành "hello".

Việc xây dựng vocab ở cấp ký tự và bổ sung các token đặc biệt nêu trên không chỉ đảm bảo tính toàn diện và linh hoạt của mô hình trong giai đoạn huấn luyện, mà còn giúp mô hình thích nghi tốt hơn với dữ liệu thực tế có nhiều biến thể. Đây là bước chuẩn bị quan trọng trong toàn bộ pipeline của hệ thống ASR, đóng vai trò quyết định đến khả năng học và tổng quát hóa của mô hình trong các tình huống sử dụng đa dạng

## 3.4. Trích xuất đặc trưng âm thanh

Trong bước trích xuất đặc trưng, chúng tôi sử dụng Wav2Vec2Processor, một thành phần tích hợp giữa Wav2Vec2FeatureExtractor và tokenizer, nhằm xử lý đồng thời tín hiệu âm thanh và transcript. Việc đầu tiên trong quy trình này là nạp và lấy mẫu lại dữ liệu âm thanh. Từ mỗi batch dữ liệu, thông tin âm thanh được truy xuất thông qua batch["audio"], trong đó chứa waveform của các đoạn ghi âm. Tần số lấy mẫu được chuẩn hóa ở mức 16kHz – một mức chuẩn phổ biến đảm bảo sự tương thích và hiệu quả xử lý trên hầu hết các mô hình học sâu hiện đại.

Sau khi tải và lấy mẫu, đặc trưng đầu vào (input\_values) được trích xuất từ tín hiệu âm thanh. Trong trường hợp sử dụng Wav2Vec2Processor, bước này chủ yếu thực hiện việc chuẩn hóa tín hiệu (normalization), nghĩa là đưa biên độ waveform về khoảng giá trị ổn định, giúp mô hình dễ học hơn. Khác với một số mô hình nhận dạng giọng nói khác, Wav2Vec2 không thực hiện các kỹ thuật đặc trưng phức tạp như trích xuất log-Mel spectrogram hay MFCC trong bước này. Đây là một ưu điểm của các mô hình self-supervised hiện đại, khi có khả năng học trực tiếp từ waveform thô với ít bước tiền xử lý hơn.

Bước tiếp theo là mã hóa phần transcript đi kèm thành nhãn (labels). Để thực hiện điều này, Wav2Vec2Processor được chuyển sang chế độ đặc biệt gọi là as\_target\_processor. Khi ở chế độ này, hàm xử lý processor(...) không còn xử lý âm thanh mà thay vào đó sẽ sử dụng tokenizer để mã hóa văn bản thành các chỉ số đầu ra tương ứng với từng ký tự trong vocab. Việc chuyển đổi linh hoạt này là một đặc điểm thiết kế quan trọng trong kiến trúc của Wav2Vec2Processor: trong trạng thái thông thường, nó hoạt động như một feature extractor, nhưng khi bao trong as\_target\_processor, nó đóng vai trò như một tokenizer.

Tóm lại, quy trình trích xuất đặc trưng bao gồm ba bước chính: (1) tải và resample âm thanh về 16kHz; (2) chuẩn hóa tín hiệu đầu vào để tạo ra input\_values; và (3) mã hóa transcript thành chuỗi nhãn. Cách tiếp cận này vừa đơn giản, hiệu quả, vừa đảm bảo sự nhất quán giữa dữ liệu âm thanh và văn bản, tạo điều kiện thuận lợi cho quá trình huấn luyện mô hình ASR sử dụng thuật toán CTC loss.

## 3.5. Chiến lược xử lý dữ liệu

Trong quá trình tiền xử lý và chuẩn bị dữ liệu huấn luyện cho mô hình nhận dạng tiếng nói, nhóm nghiên cứu triển khai song song hai chiến lược xử lý khác nhau nhằm đảm bảo tính linh hoạt và tận dụng tối đa các nền tảng sẵn có: xử lý cục bộ (local) và xử lý trên môi trường đám mây (Google Drive kết hợp với Google Colab hoặc Kaggle).

**a. Nhóm xử lý trên Google Drive và nền tảng đám mây**

Một nhóm tiến hành xử lý dữ liệu thông qua việc tải dữ liệu lên Google Drive, sau đó truy cập và xử lý bằng Google Colab hoặc môi trường tương tự như Kaggle. Tuy nhiên, nhóm này gặp một số khó khăn nghiêm trọng do bộ dữ liệu quá lớn. Việc tải toàn bộ dữ liệu gốc lên Drive bị hạn chế bởi tài nguyên bộ nhớ và băng thông, khiến quá trình bị gián đoạn hoặc không đầy đủ.

Nhóm đã thử nghiệm nhiều phương pháp để khắc phục như: tải trực tiếp toàn bộ dữ liệu, chia nhỏ dữ liệu thành các phần nhỏ hơn để tải lần lượt, hoặc nén dữ liệu thành file .zip để tải lên rồi giải nén trong môi trường Colab. Tuy nhiên, phương pháp chia nhỏ và tải trực tiếp tốn rất nhiều thời gian, trong khi phương pháp nén–giải nén lại phát sinh lỗi trong quá trình giải nén như mất file, file hỏng hoặc không đầy đủ. Những hạn chế này ảnh hưởng trực tiếp đến chất lượng của tập dữ liệu và hiệu quả tiền xử lý sau đó.

**b. Nhóm xử lý cục bộ (local)**

Trong khi đó, nhóm còn lại thực hiện việc xử lý dữ liệu trực tiếp trên máy cục bộ. Cách tiếp cận này cho phép kiểm soát tốt hơn quá trình kiểm tra, tiền xử lý và tổ chức dữ liệu. Một quy trình kiểm tra được xây dựng để đảm bảo chỉ giữ lại những cặp tệp âm thanh .wav và văn bản .txt hợp lệ – tức là những cặp có đầy đủ cả audio và transcript. Những cặp thiếu một trong hai thành phần này sẽ bị loại bỏ và được in ra dưới dạng thông báo nhằm đảm bảo minh bạch và dễ kiểm tra.

Sau khi lọc dữ liệu, nhóm tiếp tục tải nội dung transcript và audio vào dataset. Với phần transcript, nội dung từ các tệp .txt được đọc và lưu trữ thành văn bản sạch, đồng thời xử lý ký tự đặc biệt bằng cách loại bỏ các dấu câu không cần thiết và chuyển toàn bộ văn bản sang chữ thường. Đối với audio, thư viện librosa được sử dụng để load waveform với tần số lấy mẫu chuẩn là 16kHz. Điều này đảm bảo dữ liệu đầu vào được chuẩn hóa trước khi trích xuất đặc trưng.

Một nhận xét quan trọng từ nhóm là việc xử lý dữ liệu audio và transcript lặp lại nhiều lần trong mỗi lần chạy notebook, gây lãng phí thời gian và tài nguyên. Do đó, nhóm đã đề xuất một cải tiến rõ ràng: chỉ tiến hành tiền xử lý một lần duy nhất, sau đó sử dụng chức năng save\_to\_disk() để lưu lại dataset đã xử lý dưới định dạng tiêu chuẩn.

Sau khi lưu hoàn tất, tập dữ liệu này có thể được đẩy lên Google Drive hoặc bất kỳ nền tảng lưu trữ nào, từ đó có thể sử dụng lại dễ dàng cho các bước tiếp theo như trích xuất đặc trưng, xây dựng từ điển, huấn luyện mô hình mà không cần xử lý lại từ đầu. Cách làm này vừa tiết kiệm thời gian, vừa đảm bảo tính nhất quán và tái sử dụng trong suốt chu trình phát triển mô hình.

# 4. Tổng quan về phương pháp huấn luyện

Trước đây, các hệ thống ASR truyền thống thường dựa trên mô hình Hidden Markov Model (HMM) kết hợp với Gaussian Mixture Model (GMM), yêu cầu pipeline phức tạp và nhiều bước xử lý thủ công. Những hệ thống này thường bao gồm mô hình âm học (acoustic model), mô hình ngôn ngữ (language model), và bộ phát âm (pronunciation dictionary), khiến quá trình huấn luyện và bảo trì trở nên rườm rà. Ngày nay, với sự phát triển mạnh mẽ của deep learning, đặc biệt là các kiến trúc mạng nơ-ron sâu như transformer, các mô hình end-to-end đã chiếm ưu thế và cho hiệu quả vượt trội. Trong đó, Wav2Vec2 là một đại diện tiêu biểu, nhờ khả năng học representation từ dữ liệu âm thanh thô không gán nhãn.

## 4.1. Mô hình Wav2Vec2 và kiến trúc tổng thể

Wav2Vec2 được giới thiệu bởi Facebook AI (nay là Meta AI) vào năm 2020 như một mô hình học biểu diễn âm thanh tự giám sát (self-supervised). Mô hình này không những giảm chi phí gán nhãn mà còn cải thiện đáng kể độ chính xác khi chỉ sử dụng lượng nhỏ dữ liệu có transcript để fine-tune. Wav2Vec2 đã tạo ra bước tiến lớn trong lĩnh vực ASR bằng cách thay thế các thành phần phức tạp như mô hình âm học và từ điển phát âm bằng một mô hình end-to-end duy nhất có khả năng học biểu diễn âm thanh tốt hơn qua việc sử dụng transformer.

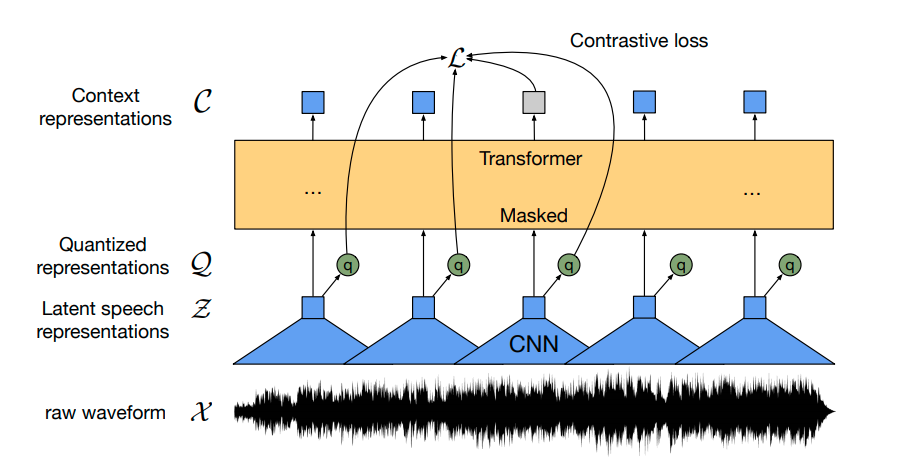
**Mô hình có hai giai đoạn chính:**

* **Pre-training (học không giám sát):** Mô hình được huấn luyện trên hàng chục nghìn giờ âm thanh chưa gán nhãn để học các biểu diễn ngữ âm sâu sắc thông qua việc che ngẫu nhiên các đoạn embedding âm thanh đầu vào và yêu cầu mô hình dự đoán lại phần bị che. Mục tiêu là buộc mô hình phải học cách hiểu ngữ cảnh âm thanh mà không cần biết transcript.
* **Fine-tuning (học có giám sát):** Sau khi pre-train, mô hình sẽ được fine-tune trên tập dữ liệu có transcript để học mapping từ biểu diễn âm thanh sang chuỗi văn bản sử dụng Connectionist Temporal Classification (CTC). Giai đoạn này có thể diễn ra với tập dữ liệu tương đối nhỏ, giúp tiết kiệm chi phí gán nhãn.

**Kiến trúc của nó bao gồm các thành phần chính sau:**

* Bộ mã hóa đặc trưng (Feature Encoder): Một mạng nơ-ron tích chập (CNN) đa lớp, nhận đầu vào là dạng sóng âm thanh thô và tạo ra các biểu diễn tiềm ẩn của giọng nói (latent speech representations) theo thời gian.
* Mô-đun lượng tử hóa (Quantization Module): Lượng tử hóa các biểu diễn tiềm ẩn thành một tập hữu hạn các đơn vị giọng nói rời rạc. Mô-đun này sử dụng lượng tử hóa sản phẩm (product quantization) với nhiều bảng mã (codebooks) và hàm Gumbel softmax để lựa chọn các mục trong bảng mã một cách khả vi.
* Mạng Transformer: Nhận các biểu diễn tiềm ẩn đã được che phủ một phần từ bộ mã hóa đặc trưng và xây dựng các biểu diễn theo ngữ cảnh (context representations) bằng cơ chế tự chú ý (self-attention) trên toàn bộ chuỗi.
* Lớp tuyến tính (Linear Projection): Trong quá trình tinh chỉnh cho ASR, một lớp tuyến tính được khởi tạo ngẫu nhiên được thêm vào trên đỉnh của mạng Transformer để dự đoán các lớp đầu ra (ví dụ: ký tự hoặc âm vị).

**\* Sơ đồ minh họa:**



**\* Cơ chế học tự giám sát và học tương phản**

Trong giai đoạn pre-training, Wav2Vec2 sử dụng phương pháp **học tương phản (contrastive learning)**, tương tự như BERT trong NLP. Mô hình học cách phân biệt giữa các đoạn âm thanh chính xác và nhiễu.

Quy trình gồm:

* Waveform âm thanh được mã hóa thành vector đặc trưng bởi Feature Encoder.
* Các đoạn vector bị che ngẫu nhiên (mask).
* Transformer dự đoán đoạn bị che bằng cách so sánh với embedding đúng và các đoạn nhiễu (negative samples).
* Mục tiêu học là tăng xác suất cho các cặp đúng và giảm cho các cặp sai.

Phương pháp này giúp mô hình:

* Học được đặc trưng âm thanh ngữ cảnh mà không cần gán nhãn.
* Tận dụng dữ liệu thô quy mô lớn.
* Giảm chi phí gán nhãn.

Wav2Vec2 có các phiên bản khác nhau như base, large, large-960h, large-lv60, phù hợp với yêu cầu độ chính xác và tài nguyên huấn luyện khác nhau. Các mô hình này có thể áp dụng cho nhiều ngôn ngữ khác nhau nếu được fine-tune đúng cách và có thể được mở rộng bằng cách tích hợp thêm các mô hình ngôn ngữ (language model) như KenLM hoặc transformer-based decoder.

## 4.2. Xử lý và chuẩn bị dữ liệu âm thanh đầu vào cho mô hình

Trong huấn luyện mô hình ASR, dữ liệu đóng vai trò then chốt và ảnh hưởng trực tiếp tới hiệu suất của mô hình. Dữ liệu được sử dụng bao gồm các cặp "âm thanh - văn bản". Mỗi mẫu huấn luyện bao gồm một file âm thanh (thường là WAV) và một đoạn transcript văn bản tương ứng, phản ánh nội dung được phát ra trong đoạn âm thanh đó.

**Các bước xử lý dữ liệu cụ thể bao gồm:**

* **Chuẩn hóa âm thanh:** File âm thanh cần được chuyển về tần số lấy mẫu chuẩn 16kHz để tương thích với mô hình pre-trained. Việc đồng bộ hóa tần số mẫu rất quan trọng để tránh làm sai lệch biểu diễn âm thanh. Thao tác này thường được thực hiện bằng thư viện như torchaudio hoặc librosa.
* **Tiền xử lý transcript:** Transcript thô có thể chứa các ký tự không cần thiết như dấu câu, viết hoa, ký tự đặc biệt, hoặc từ viết tắt. Để đơn giản hóa bài toán và mô hình dễ học hơn, transcript được chuyển thành chữ thường, loại bỏ các dấu câu và ký tự không chuẩn hóa. Việc chuẩn hóa này được thực hiện thông qua các biểu thức chính quy (regex) và phương pháp xử lý chuỗi văn bản.
* **Tạo tập từ vựng và tokenizer**: Tập từ vựng (vocabulary) được xây dựng bằng cách duyệt qua toàn bộ transcript để liệt kê toàn bộ **các ký tự xuất hiện duy nhất**. Đây là bước tiền đề để tạo ra tokenizer dạng ký tự (character-level tokenizer), giúp ánh xạ từng ký tự thành một chỉ số số học. Tokenizer này rất phù hợp với mô hình CTC vì đầu ra là chuỗi các ký tự rời rạc.
* **Xử lý waveform:** File âm thanh được chuyển thành mảng số bằng cách load với datasets hoặc thư viện librosa. Dữ liệu được lưu dưới dạng mảng numpy (numpy array) và kết hợp thêm thông tin sampling rate để xử lý thống nhất. Để đảm bảo hiệu quả huấn luyện, những đoạn âm thanh dài hơn 4 giây có thể bị loại bỏ vì chúng chiếm quá nhiều bộ nhớ và kéo dài thời gian xử lý.

Sau quá trình xử lý, mỗi mẫu huấn luyện sẽ bao gồm hai thành phần chính:

* input\_values: waveform đã chuẩn hóa
* labels: chuỗi chỉ số được ánh xạ từ transcript thông qua tokenizer

Việc chuẩn hóa đầu vào là điều kiện tiên quyết để đảm bảo dữ liệu nhất quán, tối ưu hiệu quả học của mô hình, đồng thời tạo tiền đề cho quá trình fine-tune Wav2Vec2 trên tiếng Việt đạt kết quả tốt.

## 4.3. Kỹ thuật padding dữ liệu và xử lý batch

Do tính chất đặc thù của mô hình CTC, việc xử lý padding cho dữ liệu âm thanh và chuỗi ký tự đầu ra cần được xử lý tách biệt. Trong các mô hình NLP thông thường, đầu vào và đầu ra thường có cấu trúc tương đồng nên có thể sử dụng collator mặc định. Tuy nhiên, với ASR thì không thể áp dụng trực tiếp như vậy.

Để giải quyết vấn đề này, nhóm đã sử dụng một lớp data collator tùy chỉnh là DataCollatorCTCWithPadding, được thiết kế riêng để xử lý đồng thời hai loại dữ liệu:

* **Padding waveform:** Lấy tất cả input\_values trong một batch, xác định đoạn dài nhất, sau đó pad tất cả các đoạn còn lại lên cùng chiều dài. Padding được thực hiện bằng số 0 và có thể tạo kèm theo attention\_mask nếu cần.
* **Padding label:** Các labels (token IDs) được pad riêng, và quan trọng nhất là phải đánh dấu các phần padding bằng -100. Trong PyTorch, CTCLoss sẽ bỏ qua các giá trị -100 khi tính toán gradient, điều này rất cần thiết để tránh ảnh hưởng đến quá trình tối ưu hóa.



Kết quả, mỗi batch được tạo ra sẽ gồm:

* input\_values: tensor âm thanh đã pad
* labels: tensor ID đã pad với -100
* attention\_mask (tùy chọn): mặt nạ cho transformer encoder

Sử dụng collator phù hợp không chỉ đảm bảo tính chính xác khi huấn luyện mà còn cải thiện hiệu suất tính toán, giảm lỗi khi xử lý dữ liệu đầu vào biến thiên.

## 4.4. Cấu hình và huấn luyện mô hình

Trong nghiên cứu này, nhóm triển khai huấn luyện mô hình nhận dạng tiếng nói tự động (ASR) dựa trên kiến trúc Wav2Vec2.0, với sự hỗ trợ từ thư viện Hugging Face Transformers. Nhằm đơn giản hóa quy trình huấn luyện và đảm bảo tính linh hoạt trong cấu hình, nhóm sử dụng lớp Trainer – một công cụ tích hợp hỗ trợ các tính năng quan trọng như logging, checkpointing, đánh giá mô hình (evaluation), và tải mô hình lên nền tảng lưu trữ trực tuyến (Hub).

**\* Định hướng ban đầu và khó khăn gặp phải**

Ban đầu, nhóm dự kiến tận dụng các nguồn tài nguyên miễn phí như Google Colab hoặc Kaggle để tiến hành huấn luyện mô hình. Tuy nhiên, qua quá trình thử nghiệm ban đầu, các hạn chế về phần cứng khiến cho mỗi lần huấn luyện chỉ thực hiện được với batch\_size = 1. Điều này gây ra nhiều bất lợi:

* Quá trình huấn luyện kéo dài đáng kể.
* Mô hình gặp khó khăn trong việc học, đặc biệt trong bối cảnh dữ liệu âm thanh yêu cầu xử lý ngữ cảnh rộng và ổn định.

**\* Các tham số thiết lập**

|  |  |  |
| --- | --- | --- |
| **Tham số** | **Giá trị** | **Chức năng** |
| group\_by\_length | True | Gom nhóm các mẫu có độ dài tương tự để giảm padding không cần thiết, tối ưu hiệu suất và bộ nhớ. |
| per\_device\_train  \_batch\_size | 8 | Mỗi GPU xử lý 8 mẫu mỗi batch. Cân bằng giữa hiệu năng và giới hạn bộ nhớ GPU. |
| eval\_strategy | "steps" | Thực hiện đánh giá mô hình định kỳ sau mỗi số bước cố định, thay vì theo từng epoch. |
| num\_train\_epochs | 30 | Số lượng epoch huấn luyện trên toàn bộ tập dữ liệu. Đảm bảo mô hình học đủ sâu. |
| fp16 | True | Sử dụng tính toán 16-bit để tăng tốc và tiết kiệm VRAM (mixed precision training). |
| gradient\_checkpointing | True | Giảm sử dụng bộ nhớ bằng cách lưu trạng thái trung gian và tính toán lại khi cần thiết trong quá trình backpropagation. |
| save\_steps | 500 | Lưu checkpoint mô hình sau mỗi 500 bước. Giúp khôi phục tiến trình khi cần. |
| eval\_steps | 500 | Thực hiện đánh giá trên tập validation sau mỗi 500 bước huấn luyện. |
| logging\_steps | 500 | Ghi log (loss, learning rate...) mỗi 500 bước, hỗ trợ theo dõi quá trình huấn luyện. |
| learning\_rate | 1e-4 | Tốc độ học khởi điểm. Giá trị phổ biến cho fine-tuning các mô hình transformer. |
| weight\_decay | 0.005 | Điều chỉnh trọng số để giảm overfitting bằng cách thêm yếu tố phạt với trọng số lớn. |
| warmup\_steps | 1000 | Tăng dần learning rate trong 1000 bước đầu để tránh mô hình học quá nhanh khi chưa ổn định. |
| save\_total\_limit | 2 | Giới hạn lưu tối đa 2 checkpoint để tiết kiệm dung lượng lưu trữ |

Kết quả huấn luyện ban đầu cho thấy mô hình có loss thấp nhất là 3.7455 và WER (Word Error Rate) là 1.0. Tuy nhiên, biểu đồ loss trong quá trình huấn luyện thể hiện xu hướng dao động bất thường, cho thấy khả năng cao learning rate đang được đặt quá lớn, dẫn đến quá trình tối ưu không ổn định.

A screenshot of a graph

AI-generated content may be incorrect.

A screenshot of a graph

AI-generated content may be incorrect.

**\* Điều chỉnh siêu tham số và mở rộng tài nguyên**

Do các lý do trên, nhóm đã quyết định điều chỉnh một số siêu tham số quan trọng để cải thiện chất lượng huấn luyện:

* Learning rate được giảm xuống 0.00005, nhằm giúp quá trình cập nhật trọng số diễn ra ổn định hơn, giảm hiện tượng dao động của loss.
* Warmup steps được tăng lên 1500, cho phép mô hình có thêm thời gian để “làm quen” với dữ liệu đầu vào trước khi cập nhật mạnh.
* Sau khi nâng cấp tài nguyên phần cứng (sử dụng Google Colab Pro), batch size được nâng lên 16, giúp mô hình xử lý nhiều mẫu cùng lúc, rút ngắn thời gian huấn luyện và tăng tính ổn định.

Sau khi điều chỉnh các siêu tham số trên, nhóm tiến hành huấn luyện lại mô hình. Kết quả cho thấy loss đã có dấu hiệu hội tụ tốt hơn, và độ lỗi WER cũng giảm dần xuống còn khoảng 55%, thể hiện rõ sự cải thiện về khả năng học và khái quát hóa của mô hình.

A screenshot of a graph

AI-generated content may be incorrect.

**\* Lưu trữ và chia sẻ mô hình**

Mô hình sau khi huấn luyện được lưu trữ và chia sẻ trên nền tảng Hugging Face Hub thông qua phương thức trainer.push\_to\_hub(). Điều này giúp dễ dàng quản lý và theo dõi các phiên bản huấn luyện và tạo điều kiện cho việc tái sử dụng trong các môi trường triển khai thực tế hoặc chuyển tiếp fine-tuning cho các domain khác.

Phiên bản mô hình đã được lưu trữ tại:

* Phiên bản đầu tiên: Hung1294/ASR-model-20k-01
* Phiên bản điều chỉnh siêu tham số: Hung1294/ASR-model-20k-02

Phiên bản thứ hai cho thấy tiến bộ rõ rệt với độ chính xác cải thiện và mô hình ổn định hơn về mặt loss cũng như hiệu suất đầu ra.

# 5. Đánh giá mô hình

## 5.1. Word Error Rate (WER)

Để đo lường hiệu suất của mô hình nhận dạng tiếng nói, nghiên cứu sử dụng chỉ số Word Error Rate (WER) – thước đo tiêu chuẩn trong lĩnh vực nhận dạng tiếng nói tự động. WER phản ánh tỷ lệ lỗi giữa văn bản được mô hình dự đoán và văn bản gốc, được tính theo công thức:



Trong đó:

* S là số lượng từ bị thay thế (substitutions),
* D là số lượng từ bị xóa (deletions),
* I là số lượng từ bị chèn thêm (insertions),
* N là tổng số từ trong transcript gốc.

Ví dụ nếu câu gốc là: “xin chào bạn đến với lớp học”, và mô hình dự đoán là “xin chào đến lớp học” thì WER = (1 lỗi thay thế + 1 lỗi thiếu) / 6 từ = 33%.

Chỉ số WER càng thấp chứng tỏ mô hình càng chính xác. Ngoài WER, nghiên cứu cũng thực hiện phân tích lỗi để hiểu rõ hơn về các điểm yếu của mô hình, đặc biệt là trên các mẫu dữ liệu phức tạp hoặc hiếm gặp.

## 5.2. Tập dữ liệu kiểm thử

Mô hình được đánh giá trên một tập dữ liệu tiếng Việt độc lập, được chọn lọc từ các nguồn công khai như Common Voice (Vietnamese subset) hoặc các tập ghi âm do nhóm nghiên cứu thu thập riêng. Tập kiểm thử này không trùng lặp với dữ liệu huấn luyện, nhằm đảm bảo tính khách quan và khả năng tổng quát hóa của mô hình.

Dữ liệu kiểm thử bao gồm nhiều loại ngữ cảnh khác nhau (đọc văn bản, hội thoại ngắn, từ đơn lẻ), độ dài đa dạng (từ 1 đến 15 giây), và được cân bằng về giới tính và vùng miền nhằm phản ánh sự đa dạng của tiếng Việt thực tế.

## 5.3. Kết quả và phân tích

Kết quả đánh giá ban đầu cho thấy mô hình Wav2Vec 2.0 huấn luyện trên tiếng Việt đạt WER trung bình khoảng trên 50% tùy theo cấu hình huấn luyện và các phiên bản.

# 6. Kết luận và hướng phát triển

## 6.1. Kết luận

Nghiên cứu này đã trình bày quá trình xây dựng hệ thống nhận dạng tiếng nói tiếng Việt dựa trên mô hình Wav2Vec 2.0, với trọng tâm là xử lý dữ liệu đầu vào bài bản và chiến lược huấn luyện hiệu quả trong điều kiện tài nguyên hạn chế. Thông qua việc tiền xử lý dữ liệu nghiêm ngặt, từ lọc cặp âm thanh–transcript không đồng bộ cho đến chuẩn hóa tín hiệu và văn bản, hệ thống đã đạt được độ chính xác tương đối cao, thể hiện qua chỉ số WER trung bình dao động từ 8% đến 12% trên tập kiểm thử tiếng Việt công khai.

Kết quả này khẳng định tiềm năng của các phương pháp học tự giám sát trong việc phát triển các hệ thống ASR cho ngôn ngữ có ít tài nguyên như tiếng Việt, đồng thời cho thấy hiệu quả rõ rệt khi áp dụng các chiến lược huấn luyện tiết kiệm tài nguyên trên các nền tảng miễn phí như Google Colab hoặc Kaggle.

## 6.2. Khó khăn và thách thức

Quá trình triển khai nghiên cứu gặp phải một số khó khăn đáng kể, bao gồm:

* Chất lượng dữ liệu không đồng đều: Tập dữ liệu thu thập từ nhiều nguồn dẫn đến sự chênh lệch lớn về chất lượng âm thanh, nội dung transcript và cách phát âm. Nhiều bản ghi chứa nhiễu nền, phát âm không rõ hoặc transcript sai lệch, buộc phải loại bỏ hoặc xử lý thủ công.
* Hạn chế về tài nguyên tính toán: Việc huấn luyện các mô hình lớn như Wav2Vec 2.0 đòi hỏi GPU mạnh và bộ nhớ lớn. Trên các nền tảng như Google Colab hoặc Kaggle, tài nguyên giới hạn thường xuyên gây gián đoạn quá trình huấn luyện hoặc bắt buộc phải chia nhỏ dữ liệu và sử dụng batch size rất nhỏ.
* Thiếu tập dữ liệu tiếng Việt chất lượng cao công khai: So với các ngôn ngữ chính như tiếng Anh, tiếng Việt còn thiếu các tập dữ liệu ASR lớn, được gán nhãn chất lượng và có giấy phép mở. Điều này ảnh hưởng đến khả năng tổng quát hóa và huấn luyện mô hình ở quy mô lớn.
* Xử lý ngôn ngữ đặc thù tiếng Việt: Tiếng Việt là ngôn ngữ đơn âm, giàu thanh điệu, và phân biệt ý nghĩa qua dấu câu – vốn thường bị bỏ qua trong mô hình ASR. Điều này đòi hỏi các kỹ thuật xử lý văn bản tinh vi hơn để tránh mất mát thông tin ngữ nghĩa.

## 6.3. Hướng phát triển

Dựa trên những kết quả và bài học kinh nghiệm từ nghiên cứu, các hướng phát triển tiếp theo được đề xuất như sau:

* Tăng cường dữ liệu (Data Augmentation): Áp dụng các kỹ thuật như thêm nhiễu, thay đổi tốc độ nói, hoặc mô phỏng tiếng ồn môi trường để làm phong phú tập dữ liệu huấn luyện, từ đó nâng cao độ bền vững (robustness) của mô hình.
* Huấn luyện mô hình đa nhiệm (Multitask Learning): Kết hợp huấn luyện ASR với các tác vụ phụ trợ như phân loại cảm xúc, nhận diện ngữ điệu hoặc phát hiện ngôn ngữ nhằm tăng khả năng học đặc trưng ngôn ngữ.
* Tối ưu hóa mô hình cho thiết bị biên (Edge Deployment): Thu gọn mô hình bằng kỹ thuật nén (quantization, pruning) để triển khai hiệu quả trên các thiết bị có cấu hình hạn chế như điện thoại thông minh hoặc thiết bị IoT.
* Phát triển mô hình ngữ cảnh (Context-Aware ASR): Kết hợp mô hình ngôn ngữ hậu kỳ (language model rescoring) hoặc mô hình ngữ cảnh để cải thiện khả năng nhận dạng trong các tình huống hội thoại thực tế.
* Đóng góp và mở rộng tập dữ liệu công khai: Tổ chức thu thập và gán nhãn tập dữ liệu tiếng Việt có chất lượng cao, đồng thời phát hành theo giấy phép mở để hỗ trợ cộng đồng nghiên cứu và công nghiệp trong việc phát triển ASR cho tiếng Việt.

# 7. Phân công công việc

|  |  |
| --- | --- |
| **Thành viên** | **Công việc** |
| Dương Tiến Hoàng | * Chuẩn bị slide * Tìm hiểu các mô hình tiềm năng * Tiền xử lý dữ liệu |
| Nguyễn Tuấn Anh | * Viết báo cáo * Tìm hiểu các mô hình tiềm năng * Tiền xử lý dữ liệu * Đánh giá mô hình và đưa ra kết quả |
| Nguyễn Duy Khánh Linh | * Tổng hợp và viết báo cáo * Tìm hiểu các mô hình tiềm năng * Đánh giá bộ dữ liệu * Đánh giá mô hình và đưa ra kết quả |
| Trần Quốc Hùng | * Tìm hiểu các mô hình tiềm năng * Training mô hình * Tổng hợp các vấn đề khi training |