**Vietnamese Lyrics Recognition from Live Music Using Whisper Small**

Nguyen Gia Huy

Email: [huyngia2003@gmail.com](mailto:huyngia2003@gmail.com)

Hanoi University of Science and Technology – Troy Univeristy

Computer Science

**Tóm tắt**

Trong bối cảnh các chương trình biểu diễn trực tiếp ngày càng phổ biến trên các nền tảng số, nhu cầu trích xuất và nhận diện lời bài hát từ các đoạn nhạc sống trở nên cấp thiết đối với nhiều ứng dụng thực tiễn như tạo phụ đề tự động, phân tích nội dung giải trí, và hỗ trợ tìm kiếm nhạc theo lời. Tuy nhiên, việc nhận dạng giọng hát trong môi trường nhiều tạp âm như concert, liveshow, hay chương trình truyền hình gặp rất nhiều thách thức do sự xuất hiện đồng thời của nhạc nền, tiếng khán giả, tiếng MC hoặc các yếu tố ngoại cảnh khác.

Đồ án này tập trung phát triển một hệ thống nhận diện lời bài hát tiếng Việt bằng cách ứng dụng mô hình Whisper Small của OpenAI. Hệ thống được fine-tune trên hơn 4.5 giờ dữ liệu âm thanh thu thập từ các video biểu diễn thực tế, trong đó âm thanh được xử lý và gán nhãn tỉ mỉ, phân loại thành các đoạn lời hát (lyric), đoạn nhạc nền (music) và đoạn trò chuyện (talk). Toàn bộ quy trình từ tiền xử lý, chuẩn hóa, trích xuất đặc trưng log-Mel spectrogram cho đến xây dựng pipeline huấn luyện đều được tối ưu hóa nhằm tăng khả năng nhận diện trong điều kiện có nhiều tạp âm.

Quá trình huấn luyện cho thấy mô hình đạt hiệu quả tốt với chỉ số Word Error Rate (WER) thấp nhất là 0.205, đặc biệt chính xác ở các đoạn lời hát rõ ràng và ít bị ảnh hưởng bởi tiếng nền. Hệ thống không chỉ giúp tự động hóa việc tạo phụ đề nhạc sống, mà còn có thể hỗ trợ các tác vụ khác như tìm kiếm nhạc qua trích xuất lời, phân tích nội dung biểu diễn, cũng như góp phần xây dựng cơ sở dữ liệu nhạc số tiếng Việt phong phú hơn.

Đề tài còn đặt nền móng cho việc nghiên cứu và ứng dụng các mô hình AI mạnh hơn trong tương lai, đồng thời đề xuất các hướng phát triển như mở rộng tập dữ liệu, tách nguồn âm thanh (source separation), và nâng cấp mô hình để nâng cao độ chính xác và khả năng ứng dụng thực tế.

Table of Contents

[Chương 1: Giới thiệu đề tài 3](#_Toc199428445)

[**1.1 Đặt vấn đề** 4](#_Toc199428446)

[**1.2 Mục tiêu và phạm vi đề tài** 4](#_Toc199428447)

[**1.3 Định hướng giải pháp** 5](#_Toc199428448)

[Chương 2: Cơ sở lý thuyết 7](#_Toc199428449)

[**2.1 Kiến thức nền tảng liên quan** 7](#_Toc199428450)

[**2.2 Tổng quan về mô hình Whisper** 9](#_Toc199428451)

[**2.3 Chỉ số đánh giá WER (Word Error Rate)** 10](#_Toc199428452)

[Chương 3: Phân tích và thiết kế hệ thống 12](#_Toc199428453)

[**3.1 Mô tả bài toán** 12](#_Toc199428454)

[**3.2 Quy trình xử lý dữ liệu** 12](#_Toc199428455)

[**3.3 Sơ đồ hệ thống, pipeline** 13](#_Toc199428456)

[**3.4 Mô hình huấn luyện** 15](#_Toc199428457)

[Chương 4: Thực nghiệm và đánh giá 17](#_Toc199428458)

[**4.1 Dữ liệu sử dụng** 17](#_Toc199428459)

[**4.2 Cách huấn luyện mô hình** 17](#_Toc199428460)

[**4.3 Các chỉ số đánh giá** 17](#_Toc199428461)

[**4.4 Nhận xét kết quả** 18](#_Toc199428462)

[Chương 5: Kết luận và hướng phát triển 20](#_Toc199428463)

[**5.1 Tóm tắt kết quả đạt được** 20](#_Toc199428464)

[**5.2 Hạn chế** 20](#_Toc199428465)

[**5.3 Định hướng tương lai** 20](#_Toc199428466)

**Chương 1: Giới thiệu đề tài**

**1.1 Đặt vấn đề**

Trong kỷ nguyên bùng nổ của công nghệ số và sự phát triển vượt bậc của các nền tảng truyền thông đa phương tiện, âm nhạc trực tiếp đang ngày càng chiếm vị trí quan trọng trong đời sống văn hóa hiện đại. Các sự kiện như concert, liveshow, hay chương trình truyền hình âm nhạc không chỉ mang đến trải nghiệm giải trí mà còn góp phần thúc đẩy giao lưu, kết nối cộng đồng và lưu giữ những giá trị nghệ thuật độc đáo. Song song với đó, nhu cầu **tự động nhận diện và trích xuất lời bài hát** từ các buổi biểu diễn trực tiếp phục vụ cho việc tạo phụ đề, tìm kiếm nhạc theo lời, xây dựng kho dữ liệu nhạc số, hay phân tích nội dung đang trở thành yêu cầu thực tế của nhiều cá nhân, doanh nghiệp và nền tảng số.

Tuy nhiên, so với việc nhận diện lời bài hát từ các bản thu âm chuẩn trong phòng, việc nhận diện trong môi trường biểu diễn trực tiếp đặt ra những thách thức lớn. Âm thanh thường bị pha trộn giữa giọng hát, nhạc nền, tiếng khán giả, thậm chí là các tiếng động ngoại cảnh và lời trò chuyện trên sân khấu. Những yếu tố này không chỉ làm giảm chất lượng tín hiệu đầu vào mà còn gây nhiễu loạn dữ liệu, ảnh hưởng trực tiếp đến độ chính xác của các hệ thống nhận dạng giọng nói truyền thống vốn chủ yếu được huấn luyện trên dữ liệu “sạch”. Đặc biệt đối với tiếng Việt, vốn có ngữ điệu và thanh điệu phong phú, bài toán nhận diện lời hát trong nhạc sống càng trở nên phức tạp và ít được nghiên cứu bài bản.

Chính vì vậy, việc nghiên cứu và phát triển một hệ thống nhận diện lời bài hát chính xác, hiệu quả trong môi trường biểu diễn thực tế có nhiều tạp âm không chỉ là bài toán kỹ thuật có giá trị ứng dụng cao, mà còn góp phần mở rộng ứng dụng của trí tuệ nhân tạo vào lĩnh vực âm nhạc, hỗ trợ bảo tồn và lan tỏa văn hóa âm nhạc Việt Nam trong thời đại mới.

**1.2 Mục tiêu và phạm vi đề tài**

**Mục tiêu của đề tài** là nghiên cứu và xây dựng một hệ thống tự động chuyển đổi âm thanh từ các buổi biểu diễn nhạc sống thành lời bài hát tiếng Việt một cách chính xác, bằng cách ứng dụng và fine-tune mô hình Whisper Small của OpenAI trên dữ liệu thực tế. Hệ thống hướng tới việc giải quyết các thách thức về tạp âm, nhạc nền và tính đa dạng của giọng hát trong môi trường concert, liveshow, giúp nâng cao khả năng nhận diện lời hát trong điều kiện âm thanh phức tạp.

**Phạm vi thực hiện của đề tài** bao gồm:

* **Thu thập và xử lý** trên 5 giờ dữ liệu âm thanh tiếng Việt lấy từ các video biểu diễn trực tiếp, trong đó 4.5 giờ dành cho huấn luyện mô hình và phần còn lại cho kiểm thử, đánh giá.
* **Tiền xử lý dữ liệu:** chuẩn hóa định dạng, cắt đoạn audio, gán nhãn lời hát, nhạc nền và đoạn nói.
* **Huấn luyện và fine-tune** mô hình Whisper Small trên tập dữ liệu tiếng Việt đã được chuẩn hóa.
* **Đánh giá mô hình** theo các tiêu chí khách quan như Word Error Rate (WER) và độ chính xác nhận diện lời hát.
* Đề tài **không mở rộng** sang các bài toán khác như nhận diện nhạc cụ, phân tích giai điệu, hoặc dịch lời bài hát sang ngôn ngữ khác.

**1.3 Định hướng giải pháp**

Để giải quyết bài toán nhận diện lời bài hát trong môi trường biểu diễn trực tiếp nhiều tạp âm, đề tài lựa chọn sử dụng **mô hình Whisper Small** – một mô hình nhận dạng giọng nói mã nguồn mở nổi bật do OpenAI phát triển, nổi tiếng với khả năng chịu nhiễu tốt và hỗ trợ đa ngôn ngữ, bao gồm tiếng Việt.

Quy trình thực hiện gồm các bước chính:

* **Thu thập dữ liệu** từ các video nhạc sống thực tế, đảm bảo đa dạng về thể loại và môi trường biểu diễn.
* **Tiền xử lý dữ liệu:** chuẩn hóa định dạng âm thanh (về 16kHz, mono), phân đoạn audio thành các đoạn nhỏ (30 giây), gán nhãn chi tiết cho từng đoạn (lyric, music, talk) để hỗ trợ quá trình huấn luyện.
* **Huấn luyện và fine-tune mô hình Whisper Small** trên tập dữ liệu tiếng Việt đã tiền xử lý, tối ưu hóa tham số phù hợp với môi trường nhạc sống.
* **Đánh giá và tinh chỉnh mô hình** bằng các chỉ số như WER, độ chính xác, nhằm đảm bảo mô hình có khả năng nhận diện tốt trong điều kiện âm thanh phức tạp.
* **Kiểm thử mô hình** trên các trường hợp thực tế để đánh giá tính ứng dụng, từ đó đề xuất các hướng cải tiến về sau.

Các công cụ chính được sử dụng trong đồ án gồm: **Python** (ngôn ngữ lập trình), **PyTorch** (framework deep learning), và các thư viện hỗ trợ fine-tune, xử lý dữ liệu âm thanh. Mô hình sau khi huấn luyện được triển khai thử nghiệm trên các tập dữ liệu mới để đánh giá khả năng ứng dụng thực tiễn.

**Chương 2: Cơ sở lý thuyết**

**2.1 Kiến thức nền tảng liên quan**

**2.1.1 Tiền xử lý âm thanh: Vì sao cần thiết?**

Âm thanh thu được từ môi trường biểu diễn thực tế thường chứa rất nhiều tạp âm như tiếng ồn môi trường, nhạc nền lớn, tiếng khán giả, hoặc các đoạn nói chồng lấn. Nếu không tiền xử lý, dữ liệu sẽ thiếu nhất quán, làm giảm hiệu quả và độ chính xác của mô hình. Việc tiền xử lý âm thanh với các bước như **chuẩn hóa tần số lấy mẫu về 16kHz, chuyển sang mono, loại bỏ nhiễu** giúp:

* **Chuẩn hóa đầu vào**: đảm bảo mọi file audio có cùng định dạng, giúp mô hình học dễ dàng và tăng tính ổn định.
* **Làm nổi bật tín hiệu giọng hát**: giảm bớt tạp âm, giúp đặc trưng quan trọng (giọng hát) trở nên rõ ràng hơn.
* **Giảm tải tính toán**: dữ liệu được tối ưu, tiết kiệm bộ nhớ và tăng tốc huấn luyện.

Nếu bỏ qua bước này, mô hình dễ bị “nhiễu loạn” bởi những yếu tố không liên quan đến lời bài hát, dẫn đến chất lượng kết quả kém.

**2.1.2 Tại sao sử dụng đặc trưng MFCC?**

**MFCC (Mel-Frequency Cepstral Coefficients)** là một dạng đặc trưng thường dùng trong xử lý âm thanh, đặc biệt phù hợp với tiếng nói. MFCC mô phỏng cách tai người cảm nhận tần số, nhờ đó giúp mô hình học tốt hơn các đặc trưng hữu ích từ tín hiệu âm thanh.

* **Trích xuất đặc trưng quan trọng**: MFCC giúp tập trung vào những yếu tố chính của giọng nói, loại bỏ nhiễu và thông tin dư thừa.
* **Chuyển đổi sóng âm sang dạng phù hợp**: Tín hiệu sóng âm ban đầu rất phức tạp, việc chuyển thành MFCC hoặc **log-Mel spectrogram** (như Whisper sử dụng) giúp mạng học sâu tiếp nhận và xử lý hiệu quả hơn.
* **Đặc biệt phù hợp cho hát**: Các mẫu âm thanh kéo dài, biến đổi phức tạp của giọng hát cũng có thể được nhận diện rõ qua biểu diễn log-Mel.

**2.1.3 Mô hình học sâu (Deep Learning Model) là gì?**

**Học sâu (Deep Learning)** là lĩnh vực sử dụng nhiều lớp mạng nơ-ron nhân tạo để tự động học đặc trưng và quy luật phức tạp từ dữ liệu lớn.

* Trong bài toán ASR, các mô hình học sâu như **RNN (Recurrent Neural Network)**, **Transformer** hoặc **Conformer** giúp ánh xạ trực tiếp đặc trưng âm thanh sang chuỗi văn bản mà không cần bước trích chọn thủ công.
* **Khả năng học phi tuyến**: Mô hình học sâu có thể nhận diện các mẫu âm thanh đa dạng, không tuyến tính, từ lời nói thường đến lời hát trong nhiều môi trường nhiễu.
* **Tính thích nghi**: Học sâu có thể thích nghi với các biến thể về phát âm, ngữ điệu, giọng hát và nhạc nền.

**2.1.4 Quá trình giải mã (Decoding) trong ASR**

Sau khi mô hình học sâu dự đoán xác suất của từng token hoặc ký tự ở mỗi thời điểm, cần chuyển các xác suất này thành chuỗi văn bản cuối cùng, gọi là **giải mã (decoding)**.

* **Beam search** là phương pháp phổ biến nhất: duyệt nhiều nhánh dự đoán, chọn dãy từ có tổng xác suất cao nhất, giúp kết quả tự nhiên và chính xác hơn.
* **Greedy decoding**: đơn giản hơn, luôn chọn token có xác suất cao nhất tại mỗi bước, nhưng dễ bỏ lỡ chuỗi tốt nhất.
* **Kết hợp với mô hình ngôn ngữ**: giúp đầu ra hợp lý, phù hợp ngữ cảnh, đặc biệt khi nhận diện lời hát hoặc các đoạn hội thoại phức tạp.

**2.2 Tổng quan về mô hình Whisper**

**2.2.1 Kiến trúc và các thành phần chính**

Whisper là mô hình nhận dạng giọng nói tự động (ASR) dựa trên kiến trúc Transformer, được thiết kế để xử lý hiệu quả nhiều loại âm thanh trong môi trường thực tế, đặc biệt là khi có tạp âm. Mô hình bao gồm các thành phần chính sau:

* **Feature Extractor:** Chuyển đổi tín hiệu âm thanh đầu vào thành **log-Mel spectrogram** – dạng đặc trưng phổ biến giúp mô hình nhận biết được các yếu tố quan trọng của tín hiệu giọng nói và nhạc.
* **Encoder (Transformer):** Nhận log-Mel spectrogram và học các đặc trưng tổng quát của toàn bộ đoạn âm thanh, cho phép mô hình hiểu ngữ cảnh dài và các mối liên hệ phức tạp trong tín hiệu.
* **Decoder (Transformer):** Sinh ra văn bản dự đoán từ đặc trưng đã mã hóa. Decoder có khả năng sử dụng các token đặc biệt để chỉ định tác vụ (ví dụ: dịch, phiên âm) hoặc xác định ngôn ngữ đầu ra.
* **Tokenizer:** Biến đổi văn bản thành dãy số (token) để đưa vào mô hình, đồng thời hỗ trợ nhiều ngôn ngữ khác nhau, trong đó có tiếng Việt.
* **Forced Decoder IDs:** Cơ chế buộc mô hình sinh ra đầu ra bằng đúng ngôn ngữ mong muốn (ở đây là tiếng Việt), giúp tăng độ chính xác trong các môi trường đa ngôn ngữ.

**2.2.2 Vì sao chọn Whisper Small?**

Whisper Small là một phiên bản thu nhỏ trong họ mô hình Whisper, với **244 triệu tham số**. Lựa chọn Whisper Small trong đề tài này dựa trên các lý do sau:

* **Cân bằng giữa hiệu năng và tài nguyên:** Mô hình có độ chính xác cao so với các mô hình nhỏ hơn (tiny, base), nhưng vẫn đủ nhẹ để triển khai trên máy tính cá nhân, Google Colab hoặc các máy chủ GPU phổ thông.
* **Phù hợp với lượng dữ liệu giới hạn:** Kích thước nhỏ giúp việc fine-tune trên tập dữ liệu không quá lớn (như đề tài chỉ có ~5 giờ dữ liệu) trở nên hiệu quả, giảm nguy cơ quá khớp (overfitting).
* **Tối ưu cho nghiên cứu và ứng dụng thực tế:** Thời gian huấn luyện nhanh hơn, chi phí thấp, dễ dàng mở rộng hoặc thử nghiệm với các cấu hình khác nhau.

**2.2.3 Ưu điểm của Whisper**

* **Khả năng chịu nhiễu tốt:** Whisper được huấn luyện trên dữ liệu đa dạng, bao gồm nhiều đoạn âm thanh có tạp âm, nhạc nền, tiếng nói đan xen… giúp mô hình hoạt động ổn định trong môi trường thực tế như concert, liveshow.
* **Hỗ trợ đa ngôn ngữ:** Whisper đã được huấn luyện trên hàng trăm ngôn ngữ, trong đó tiếng Việt được hỗ trợ khá tốt, phù hợp cho các ứng dụng nhận diện lời bài hát Việt Nam.
* **Tự động nhận diện tác vụ và ngôn ngữ:** Mô hình có thể tự phát hiện ngôn ngữ đầu vào, tự chuyển đổi giữa các tác vụ như phiên âm, dịch… giúp tăng sự linh hoạt và hiệu quả khi triển khai cho nhiều mục đích khác nhau.

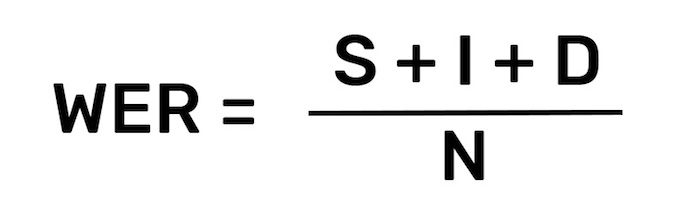
**2.3 Chỉ số đánh giá WER (Word Error Rate)**

**2.3.1 WER là gì?**

* **WER** là chỉ số chuẩn để đánh giá chất lượng hệ thống nhận dạng tiếng nói (ASR).
* WER cho biết tỷ lệ từ bị nhận sai so với tổng số từ chuẩn, càng thấp càng tốt.

**2.3.2 Công thức tính WER**

WER được tính như sau:



Trong đó:

* **S** = Số từ bị **thay thế** (substitutions)
* **D** = Số từ bị **bỏ sót** (deletions)
* **I** = Số từ **thừa** (insertions)
* **N** = Tổng số từ trong câu đúng (reference)

**Ví dụ:**  
Câu chuẩn: “Tôi yêu âm nhạc”  
Câu dự đoán: “Tôi thích nhạc”

* S = 1 (“yêu” → “thích”)
* D = 1 (“âm” bị bỏ)
* I = 0
* N = 3

**A number with black numbers

AI-generated content may be incorrect.**

**Chương 3: Phân tích và thiết kế hệ thống**

**3.1 Mô tả bài toán**

Bài toán đặt ra là xây dựng một hệ thống có thể nhận diện chính xác lời bài hát tiếng Việt từ các đoạn âm thanh được ghi lại trong các buổi biểu diễn trực tiếp (concert, liveshow, sân khấu truyền hình,...). Đặc điểm của bài toán là:

* Dữ liệu đầu vào là audio có nhiều tạp âm, tiếng nhạc nền, tiếng khán giả.
* Mục tiêu là xuất ra đoạn văn bản lời bài hát tương ứng với phần có giọng hát.
* Hệ thống phải phân biệt được giữa lời hát, lời nói không liên quan, và nhạc cụ.

**3.2 Quy trình xử lý dữ liệu**

Quy trình xử lý dữ liệu bao gồm các bước chính như sau:

1. **Thu thập dữ liệu:**
   * Nguồn dữ liệu được lấy từ YouTube, chủ yếu là các video biểu diễn trực tiếp như concert, liveshow tiếng Việt.
   * Âm thanh được trích xuất từ video dưới định dạng WAV, chuẩn hóa về **16kHz**, **mono channel** để phù hợp với đầu vào của mô hình Whisper.
2. **Phân đoạn audio:**
   * Toàn bộ file âm thanh được chia nhỏ thành các đoạn có độ dài **30 giây**, giúp mô hình dễ xử lý, đồng thời phù hợp với giới hạn bộ nhớ khi huấn luyện.
3. **Gán nhãn lời hát:**
   * Các đoạn audio được gán nhãn bằng cách đối chiếu với phụ đề có sẵn hoặc thực hiện thủ công.
   * Mỗi đoạn được phân loại theo ba nhóm:
     + **lyric**: đoạn có giọng hát và lời bài hát rõ ràng.
     + **music**: đoạn chỉ có nhạc nền, không có lời hát.
     + **talk**: đoạn ca sĩ hoặc MC giao lưu, trò chuyện với khán giả.
   * Việc phân loại này giúp mô hình học cách phân biệt lời bài hát với các dạng âm thanh khác, từ đó nâng cao độ chính xác.

**3.3 Sơ đồ hệ thống, pipeline**

A diagram of a program

AI-generated content may be incorrect.

**3.4 Mô hình huấn luyện**

Mô hình được sử dụng trong đồ án là **Whisper Small**, một kiến trúc encoder-decoder dựa trên Transformer do OpenAI phát triển. Để phù hợp với tiếng Việt và môi trường biểu diễn trực tiếp nhiều tạp âm, mô hình đã được fine-tune trên hơn 4.5 giờ dữ liệu âm thanh từ các concert và liveshow tiếng Việt.

**Cấu hình huấn luyện cụ thể như sau:**

* **Mô hình gốc:** openai/whisper-small
* **Ngôn ngữ:** tiếng Việt (language="vietnamese", task="transcribe")
* **Kích thước batch:** 4 (tăng hiệu quả huấn luyện khi dùng gradient\_accumulation)
* **Epochs:** 50 (kèm EarlyStopping để dừng sớm nếu không cải thiện)
* **Learning rate:** 3e-4
* **Loss function:** Cross Entropy Loss (với padding là -100)
* **Optimizer:** AdamW
* **Chiến lược đánh giá và lưu mô hình:** mỗi epoch
* **Tham số early stopping:** dừng sớm nếu không cải thiện WER sau 5 epoch liên tiếp
* **Tăng tốc bằng:** fp16 và gradient accumulation (8 bước)
* **Chỉ số đánh giá chính:** Word Error Rate (WER)

**Chuẩn bị dữ liệu:**

* Âm thanh được tiền xử lý và chuyển sang dạng spectrogram bởi WhisperProcessor.
* Văn bản được mã hóa bằng tokenizer của Whisper, có chuẩn hóa chữ thường, bỏ dấu câu và khoảng trắng dư.
* Các đoạn dữ liệu được gán nhãn cụ thể gồm lyric, music, và talk, giúp mô hình học phân biệt các loại tín hiệu.

**Huấn luyện và đánh giá:**

* Dữ liệu được chia thành tập huấn luyện (90%) và kiểm thử (10%).
* Mô hình được huấn luyện bằng Seq2SeqTrainer với data collator tùy chỉnh, đảm bảo pad đúng và loại bỏ token BOS khi cần.
* Sau khi huấn luyện, mô hình được kiểm thử và đánh giá dựa trên WER với văn bản đã được chuẩn hóa.
* Mô hình cuối cùng được lưu trữ và đẩy lên Hugging Face Hub tại địa chỉ:  
  **huybunn/whisper-small-vietnamese-lyrics-transcription**

**Chương 4: Thực nghiệm và đánh giá**

**4.1 Dữ liệu sử dụng**

Tập dữ liệu gồm hơn **5 giờ audio tiếng Việt** được trích xuất từ các video biểu diễn trực tiếp (concert, liveshow), trong đó:

* **4.5 giờ** dùng để huấn luyện
* **0.5 giờ** dùng để kiểm tra  
  Dữ liệu được chia thành các đoạn **30 giây**, chuẩn hóa về định dạng **WAV 16kHz mono**, và gán nhãn gồm lyric, talk, và music. Văn bản được đồng bộ thủ công với phần âm thanh để đảm bảo độ chính xác khi huấn luyện.

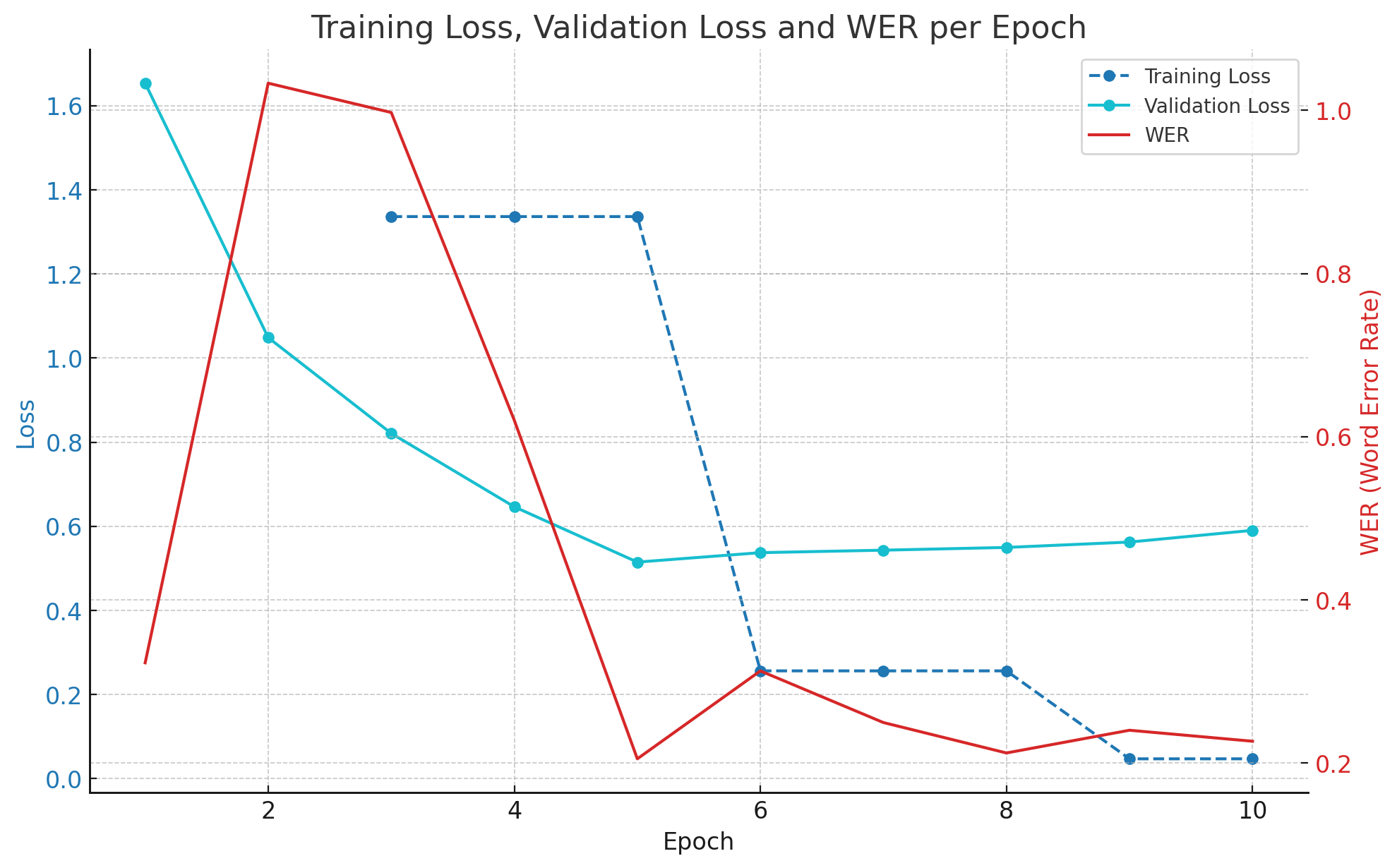
**4.2 Cách huấn luyện mô hình**

Mô hình Whisper Small được fine-tune bằng thư viện Transformers của Hugging Face với cấu hình như sau:

* Batch size: 4
* Epochs: 50 (dừng sớm sau 10 epoch)
* Optimizer: AdamW
* Learning rate: 3e-4
* Chiến lược lưu và đánh giá: theo mỗi epoch
* Sử dụng fp16 và gradient\_accumulation để tối ưu hiệu suất trên GPU
* Tự động chọn mô hình tốt nhất dựa theo chỉ số **WER thấp nhất**

**4.3 Các chỉ số đánh giá**

Kết quả huấn luyện theo từng epoch được ghi nhận như sau:



| **Epoch** | **Training Loss** | **Validation Loss** | **WER** |
| --- | --- | --- | --- |
| 1 | No log | 1.653314 | 0.322856 |
| 2 | No log | 1.048465 | 1.033229 |
| 3 | 1.336200 | 0.820843 | 0.997306 |
| 4 | 1.336200 | 0.646260 | 0.620117 |
| 5 | 1.336200 | 0.514788 | **0.205209** |
| 6 | 0.256200 | 0.537218 | 0.312977 |
| 7 | 0.256200 | 0.543139 | 0.249663 |
| 8 | 0.256200 | 0.549626 | 0.212393 |
| 9 | 0.047100 | 0.562417 | 0.240233 |
| 10 | 0.047100 | 0.590277 | 0.226762 |

**4.4 Nhận xét kết quả**

* **WER tốt nhất đạt 0.205 (~20.5%)** tại epoch thứ 5, thể hiện khả năng nhận diện lời bài hát trong điều kiện có tạp âm là khá tốt với mô hình nhỏ như Whisper Small.
* Sau epoch 5, dù loss tiếp tục giảm, WER dao động nhẹ và không cải thiện rõ rệt → early stopping là hợp lý.
* Các giá trị WER ban đầu khá cao (epoch 2–3 > 1.0), có thể do mô hình chưa thích nghi với dữ liệu tiếng Việt và âm thanh có nhạc nền.
* Việc mô hình nhanh chóng học được đặc trưng lời hát cho thấy dữ liệu gán nhãn có chất lượng tốt và phù hợp với bài toán.

**Chương 5: Kết luận và hướng phát triển**

**5.1 Tóm tắt kết quả đạt được**

Đề tài đã xây dựng thành công một hệ thống nhận diện lời bài hát tiếng Việt trong môi trường biểu diễn trực tiếp (concert, liveshow...) bằng cách fine-tune mô hình Whisper Small.  
Hệ thống đã được huấn luyện trên hơn 4.5 giờ dữ liệu thực tế, với kết quả **Word Error Rate (WER) thấp nhất đạt 0.205**, cho thấy khả năng nhận dạng lời hát trong điều kiện có nhiều tạp âm là khả thi. Mô hình đã được đẩy lên Hugging Face Hub để phục vụ sử dụng và nghiên cứu tiếp theo.

**5.2 Hạn chế**

* **Dung lượng dữ liệu huấn luyện còn khiêm tốn** so với yêu cầu của mô hình học sâu, dễ gây overfitting.
* Một số đoạn có nhạc nền lớn hoặc lời hát không rõ dẫn đến lỗi nhận dạng cao.
* Mô hình chưa phân biệt tốt giữa lời hát, lời nói giao lưu và nhạc nền nếu không có nhãn rõ ràng.
* Chưa áp dụng các kỹ thuật tách nhạc (source separation) để làm sạch đầu vào.

**5.3 Định hướng tương lai**

Để cải thiện chất lượng và mở rộng ứng dụng, các hướng phát triển tiềm năng bao gồm:

* **Tăng khối lượng và đa dạng dữ liệu**, bao gồm nhiều giọng hát, thể loại nhạc, và môi trường biểu diễn khác nhau.
* **Tách giọng hát khỏi nhạc nền** bằng các mô hình như Demucs hoặc Spleeter trước khi đưa vào ASR.
* **Thử nghiệm các mô hình lớn hơn** như Whisper Medium hoặc Large để cải thiện độ chính xác.
* **Huấn luyện mô hình phân loại đoạn âm thanh** (lyric, talk, music) kết hợp với ASR để tăng độ chính xác và phân đoạn tự động.
* **Tối ưu hóa inference** cho mục đích ứng dụng thực tế như đồng bộ lời bài hát trực tiếp, tìm kiếm nhạc theo lời, hoặc phụ đề biểu diễn.