

TRƯỜNG ĐẠI HỌC KHOA HỌC TỰ NHIÊN
KHOA CÔNG NGHỆ THÔNG TIN

Phạm Quốc Vương - Ngô Phi Hùng

HỆ THỐNG AI
HỖ TRỢ SÁNG TÁC NHẠC

KHÓA LUẬN TỐT NGHIỆP CỦ NHÂN
CHƯƠNG TRÌNH CHÍNH QUY

Tp. Hồ Chí Minh, tháng 07/2024

TRƯỜNG ĐẠI HỌC KHOA HỌC TỰ NHIÊN
KHOA CÔNG NGHỆ THÔNG TIN

Phạm Quốc Vương - 20120406

Ngô Phi Hùng - 20120486

HỆ THỐNG AI
HỖ TRỢ SÁNG TÁC NHẠC

KHÓA LUẬN TỐT NGHIỆP CỦ NHÂN
CHƯƠNG TRÌNH CHÍNH QUY

GIÁO VIÊN HƯỚNG DẪN

TS. Trần Duy Hoàng

Tp. Hồ Chí Minh, tháng 07/2024

Lời cam đoan

Nhóm xin cam đoan luận văn này là công trình nghiên cứu của riêng nhóm và những nội dung được trình bày trong luận văn này là hoàn toàn trung thực.

Những số liệu, bảng biểu phục vụ cho việc phân tích và dấn dắt đề tài này được thu thập từ các nguồn tài liệu khác nhau được ghi chú trong mục tài liệu tham khảo hoặc chú thích ngay bên dưới các bảng biểu.

Ngoài ra, đối với các tài liệu diễn giải để làm rõ thêm các luận điểm đã phân tích và trích dẫn trong phần phụ lục cũng được chú thích nguồn gốc dữ liệu.

Lời cảm ơn

Chúng em xin gửi lời cảm ơn chân thành và sâu sắc đến quý Thầy Cô đang công tác tại Trường Đại học Khoa học Tự Nhiên, Đại học Quốc gia Thành phố Hồ Chí Minh, đặc biệt là các Thầy Cô Khoa Công nghệ Thông tin. Trong suốt bốn năm qua, Thầy Cô đã nhiệt tình truyền đạt kiến thức và chia sẻ những kinh nghiệm quý báu, giúp chúng em xây dựng một nền tảng vững chắc để hoàn thành khóa luận tốt nghiệp này.

Đặc biệt, chúng em xin được bày tỏ lòng biết ơn sâu sắc đến thầy Trần Duy Hoàng, người đã trực tiếp hướng dẫn và hỗ trợ nhóm trong suốt quá trình thực hiện khóa luận. Nhờ có Thầy, chúng em đã có cơ hội tiếp cận với những kiến thức mới, trải nghiệm những điều mới mẻ và thú vị. Không chỉ dừng lại ở việc truyền đạt kiến thức chuyên môn, Thầy còn giúp chúng em hoàn thiện kỹ năng và chuẩn bị hành trang cho sự phát triển bản thân trong tương lai. Một lần nữa, chúng em xin gửi lời cảm ơn chân thành đến Thầy và chúc Thầy dồi dào sức khỏe.

Dù đã nỗ lực hoàn thành khóa luận trong phạm vi và khả năng cho phép, nhưng chắc chắn không thể tránh khỏi những thiếu sót. Chúng em rất mong nhận được sự cảm thông, góp ý và chỉ bảo tận tình từ quý Thầy Cô để hướng đến một khoá luận hoàn thiện nhất.

Chúng em chân thành cảm ơn!

TP. Hồ Chí Minh, ngày 01 tháng 07 năm 2024
Nhóm sinh viên thực hiện
Phạm Quốc Vương
Ngô Phi Hùng

Đề cương chi tiết

Xin xem ở trang kế tiếp.



fit@hcmus

TRƯỜNG ĐẠI HỌC KHOA HỌC TỰ NHIÊN
KHOA CÔNG NGHỆ THÔNG TIN

ĐỀ CƯƠNG KHOÁ LUẬN TỐT NGHIỆP
HỆ THỐNG AI HỖ TRỢ SÁNG TÁC NHẠC
(*AI-powered Support System for Music Composition*)

1 THÔNG TIN CHUNG

Người hướng dẫn:

- TS. Trần Duy Hoàng (Khoa Công nghệ Thông tin)

Nhóm sinh viên thực hiện:

1. Phạm Quốc Vương (MSSV: 20120406)
2. Ngô Phi Hùng (MSSV: 20120486)

Loại đề tài: Nghiên cứu

Thời gian thực hiện: Từ 01/2024 đến 07/2024

2 NỘI DUNG THỰC HIỆN

2.1 Giới thiệu về đề tài

Sau “cơn sốt” ChatGPT năm 2022, phần mềm phát triển dựa trên trí tuệ nhân tạo (AI) ra đời ngày càng nhiều và trở nên phổ biến, mang đến lợi ích cho hàng loạt lĩnh vực như lập trình, đồ họa, phim ảnh, truyền thông, v.v. Sản xuất âm nhạc trên máy tính cũng không ngoại lệ. Với nhiều kiểu ứng dụng như sinh nhạc theo dạng text-to-audio, text-to-midi¹, AI hỗ trợ phối trộn âm thanh (mixing), v.v, sự kết hợp giữa trí tuệ nhân tạo và sáng tạo âm nhạc mở ra nhiều cơ hội mới và tiềm năng hứa hẹn. Dựa trên xu hướng đó, nhóm chọn thực hiện đề tài “Hệ thống AI hỗ trợ sáng tác nhạc” với hướng “sinh nhạc theo dạng text-to-midi”. Bằng kỹ thuật chủ đạo là Masked Language Modeling và Next Sentence Prediction, nhóm mong muốn kết hợp các nghiên cứu đã có, cũng như kiến thức của bản thân, để cho ra một hệ thống AI hỗ trợ sáng tác nhạc, giúp khơi dậy và duy trì nguồn cảm hứng cho người làm nhạc trên máy tính nói riêng và người sáng tạo âm nhạc nói chung trong quá trình tạo ra các tác phẩm độc đáo và phong phú.

2.2 Mục tiêu đề tài

- Để giúp người sáng tác âm nhạc mở rộng phạm vi sáng tạo và có nguồn cảm hứng không giới hạn, nhóm đánh giá việc tạo ra “hệ thống AI hỗ trợ sáng tác nhạc” là một trong những cách nhanh và hiệu quả nhất.
- Khi thực hiện đề tài, nhóm muốn mang lại được các lợi ích: tăng cường hiệu suất công việc (thay vì phải dành nhiều thời gian cho việc tạo ra các giai điệu và hoà âm cơ bản, người sáng tác có thể tập trung vào các khâu phức tạp hơn như tô điểm cho giai điệu và hoà âm được công cụ tạo ra để có được thành quả tốt và đúng ý muốn nhất, giành thời gian tiết kiệm được cho khâu phối

¹MIDI - Musical Instrument Digital Interface: tương tự việc con người đọc các ký hiệu trên sheet nhạc, máy tính đọc các ký hiệu đó nhưng được biểu diễn ở dạng MIDI để hiểu được bản nhạc.

khí và các khâu phức tạp hơn sau đó), mở rộng khả năng và phạm vi sáng tạo nhờ việc cung cấp các nét nhạc với cách đi giai điệu và nhịp điệu mà người viết nhạc chưa từng nghĩ đến, hỗ trợ người không biết nhạc lý vẫn có thể sáng tác, và nhiều lợi ích khác.

- Đề tài có kết quả tốt sẽ mang ý nghĩa tăng cường sự sáng tạo trong việc sáng tác âm nhạc, thay đổi phương pháp sáng tác âm nhạc truyền thống, nâng cao hiệu suất và hiệu quả của ngành công nghiệp âm nhạc, cũng như góp một phần nhỏ vào lĩnh vực nghiên cứu và phát triển sự kết hợp giữa trí tuệ nhân tạo và âm nhạc.

2.3 Phạm vi của đề tài

2.3.1 Phạm vi chung

Đề tài này tập trung nghiên cứu và áp dụng các mô hình ngôn ngữ lớn vào lĩnh vực âm nhạc, kết hợp các mô hình và kiến thức về âm nhạc để giải quyết tác vụ sinh nhạc ở dạng midi dựa trên văn bản mô tả. Có thể chia đề tài thành hai phần chính:

- Nghiên cứu về các mô hình ngôn ngữ lớn: đánh giá và phân tích các mô hình ngôn ngữ lớn hiện đại như GPT (Generative Pre-trained Transformer), BERT (Bidirectional Encoder Representations from Transformers), và các biến thể, kỹ thuật được áp dụng trong những nghiên cứu được nhắc đến ở phần 2.4.
- Áp dụng kiến thức về âm nhạc vào giải quyết bài toán rút trích đặc trưng quan trọng từ bản nhạc như nhịp điệu, giai điệu, cao độ, trường độ, thang âm, v.v, từ đó, phát triển, cải tiến mô hình sinh nhạc ở dạng midi từ văn bản mô tả.

2.3.2 Các tập dữ liệu nhóm dự định sử dụng trong đề tài

- Hai bộ liệu huấn luyện của bài báo “MuseCoco - Generating Symbolic Music from Text” [1]. Bộ thứ nhất gồm các cặp “Câu mô tả các tính chất của bản

nhạc bằng ngôn ngữ tự nhiên - Câu mô tả được mã hoá (encode) theo định dạng xác định trước". Bộ thứ hai gồm các dòng dữ liệu chứa thông tin các đoạn nhạc được mã hoá (encode) theo định dạng được xác định trước.

- Bộ dữ liệu MusicCaps² từ bài báo “MusicLM: Generating Music From Text”[2] với các cặp “Khoá xác định bản nhạc trên YouTube - Đoạn văn bản mô tả các đặc điểm của bản nhạc” là dữ liệu chính.
- Dữ liệu do nhóm thu thập từ Hooktheory³ - trang web chuyên cung cấp sách điện tử, bài viết, thông kê, phần mềm giáo dục về lý thuyết âm nhạc, cũng như thông tin ký âm và hoà âm của hơn 40000 bản nhạc trên thế giới. Dữ liệu nhóm thu thập có thông tin chính gồm các cặp “Thông tin ký âm và hoà âm của một bản nhạc (có thể chuyển về dạng midi) - Đoạn văn bản nhận xét về bản nhạc và một số chỉ số đánh giá bản nhạc”.

2.4 Cách tiếp cận dự kiến

2.4.1 Một số nghiên cứu có sự tương đồng với đề tài

- MuseCoco - Generating Symbolic Music from Text[1]: MuseCoco là hệ thống được đề xuất giúp sinh nhạc ở dạng midi từ văn bản mô tả các đặc điểm mang tính kỹ thuật của bản nhạc.
- MusicLM: Generating Music From Text[2]: MusicLM là một mô hình được phát triển bởi Google Research, có khả năng tạo ra âm nhạc chất lượng cao dựa trên mô tả bằng văn bản. Ngoài ra, MusicLM có thể xử lý đồng thời cả văn bản và giai điệu, nghĩa là nó nhận đầu vào là các giai điệu được huýt sáo hoặc ngân nga và văn bản mô tả bổ sung để cho ra giai điệu mới.
- MUGEN: A Playground for Video-Audio-Text Multimodal Understanding and GENERation[3]: MuGen là mô hình được nghiên cứu để hiểu và sinh âm thanh

²Liên kết đến bộ dữ liệu tại Kaggle: <https://www.kaggle.com/datasets/googleai/musiccaps>, truy cập lần cuối 01/04/2024.
³Liên kết đến trang web: <https://www.hooktheory.com>, truy cập lần cuối 01/04/2024.

cho video game dựa trên đầu vào là video của một cảnh game và đoạn văn bản mô tả.

2.4.2 Nhận xét về các nghiên cứu đã nêu

- Tuy đều là các nghiên cứu sinh nhạc từ văn bản, nhưng MusicLM và MUGEN lại sinh nhạc ở định dạng âm thanh (text-to-audio), khác với hướng sinh nhạc ở dạng midi (text-to-midi) mà nhóm lựa chọn.
- MuseCoco tuy sinh nhạc ở dạng midi, phù hợp với hướng nghiên cứu của nhóm nhưng phần văn bản mô tả nhạc còn giới hạn trong việc mô tả các đặc tính kỹ thuật, không mô tả theo cảm xúc và cảm nhận tự nhiên của con người như MusicLM hay sinh nhạc theo kịch bản, mô tả ngữ cảnh như MuGen.
- Hầu hết các phương pháp tiếp cận chỉ thực hiện qua các bản demo nhỏ hoặc chưa phát triển thành một ứng dụng để giúp người dùng phổ thông có thể tiếp cận được.

2.4.3 Hướng phát triển nhóm đề xuất

- Nâng cao tính toàn diện của mô hình bằng việc kết hợp khả năng sinh text-to-midi của MuseCoco và khả năng hiểu mô tả nhạc theo cả phương diện kỹ thuật (giống MuseCoco) lẫn phương diện cảm xúc và cảm nhận của con người hoặc theo ngữ cảnh (giống MusicLM và MuGen).
- Xây dựng mô hình có bộ nhớ (có thể hỏi và trả lời với người dùng, nhớ ngữ cảnh cuộc trò chuyện để đưa ra các câu trả lời tiếp theo).
- Phát triển thành mô hình đa ngôn ngữ thay vì chỉ có tiếng Anh.
- Triển khai mô hình và phát triển một ứng dụng web giúp người dùng có thể dễ dàng tiếp cận.

2.4.4 Phương pháp tiếp cận dự kiến

- Tìm hiểu về lĩnh vực âm nhạc: các khái niệm trong âm nhạc như phách, nhịp, cao độ, trường độ, thang âm, v.v; các nguyên tắc hoà âm cơ bản; cách phối hợp các nhạc cụ trong bản nhạc; các bước để sáng tác ra một bản nhạc; v.v.
- Khảo sát, đánh giá cách tiếp cận của các nghiên cứu được đề cập ở phần 2.4.1.
- Thu thập dữ liệu từ các nguồn đã nêu ở phần 2.3.2: tải xuống thông thường với các bộ dữ liệu từ MuseCoco và MusicCaps; dùng Python để cào dữ liệu từ Hooktheory và kỹ thuật đa luồng để tăng hiệu suất cào.
- Viết công cụ chuyển đổi cấu trúc dữ liệu giữa các mô hình cần thử nghiệm.
- Tiếp cận các mô-hình-được-đào-tạo-trước (BERT, GPT2) để thử nghiệm các tác vụ trích xuất đối tượng và sinh văn bản.
- Nghiên cứu kỹ thuật Name Entity Recognition: Trích xuất từ văn bản người dùng nhập những thuộc tính được định nghĩa trước.
- Nghiên cứu kỹ thuật Masked Language Modeling: Đây là quá trình mô hình hóa ngôn ngữ bằng cách che đi ngẫu nhiên các từ trong câu đầu vào, sau đó chạy toàn bộ câu đã được xử lý như trên đi qua mô hình và để mô hình dự đoán các từ đã bị che là gì. Điều này cho phép mô hình học được một biểu diễn hai chiều của câu (từ trái qua phải và từ phải qua trái), từ những thuộc tính đã trích xuất tạo những tiền tố để mô hình có thể sinh ra câu văn bản mô tả nhạc hoàn chỉnh.
- Nghiên cứu kỹ thuật Next Sentence Prediction: Với đầu vào là một chuỗi các token tiền tố, mô hình phải dự đoán được chuỗi các token hậu tố cho chuỗi token tiền tố đó là gì. Bài toán điển hình cho kỹ thuật này là bài toán sinh thơ - mô hình nhận câu thơ đầu vào và sinh ra câu thơ tiếp theo cho câu thơ mà mô hình nhận được.

- Kết hợp các nghiên cứu có sẵn và ba kỹ thuật trên để cho ra mô hình hiểu được văn bản mô tả nhạc theo cả khía cạnh kỹ thuật và cảm xúc, cảm nhận tự nhiên của con người.
- Nghiên cứu phương pháp tối ưu và nâng cao chất lượng nhạc sinh ra (ở dạng midi) của các mô hình từ những nghiên cứu đã nêu dựa trên cách tiếp cận có sẵn của các nghiên cứu đó và hai kỹ thuật Masked Language Modeling và Next Sentence Prediction.
- Viết công cụ chuyển thông tin từ định dạng đầu ra của mô hình thành dạng midi và xây dựng ứng dụng web.

2.5 Kết quả dự kiến của đề tài

- Mô hình sinh nhạc bằng cách nhận đầu vào là văn bản mô tả đặc điểm bản nhạc bằng ngôn ngữ tự nhiên (tiếng Anh hoặc tiếng Việt).
- Ứng dụng web/Phần mềm sinh ra nhạc theo yêu cầu.

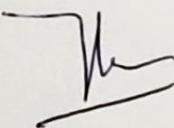
2.6 Kế hoạch thực hiện

Thời gian	Phạm Quốc Vương	Ngô Phi Hùng
01/2024	Giới thiệu các bước để làm ra một bản nhạc cơ bản và đưa ra ý tưởng ban đầu. Khảo sát các công trình nghiên cứu liên quan. Lập và bổ sung kế hoạch phát triển đề tài.	Khảo sát các công trình nghiên cứu liên quan. Xác định ý tưởng đề tài dựa trên khả năng đáp ứng của nhóm và nguồn lực các nghiên cứu liên quan. Lập kế hoạch phát triển đề tài.
02 - 03/2024	Thu thập và tiền xử lý dữ liệu. Viết công cụ chuyển đổi qua lại giữa các kiểu dữ liệu khác nhau mà những mô hình thử nghiệm cần sử dụng, từ đó có một cấu trúc dữ liệu chuẩn.	Phân tích dữ liệu, lọc và lựa chọn các nhãn cho mô hình. Thủ nghiêm, cài đặt pipeline trích xuất đặc trưng từ văn bản nhập vào theo tiêu chí: đặc trưng cần trích xuất thuộc trạng thái nào trong ba trạng thái “có tồn tại trong kết quả đầu ra mong muốn”, “không tồn tại trong kết quả đầu ra mong muốn” và “không được nhắc đến trong câu mô tả” đối với các thuộc tính về kỹ thuật như tốc độ của bản nhạc.
04/2024	Nghiên cứu hướng kết hợp các bộ dữ liệu với nhau. Hỗ trợ xử lý, mở rộng bộ dữ liệu huấn luyện.	Cài đặt và cấu hình máy chủ để huấn luyện mô hình. Cài đặt lại Masked Language Model cho tác vụ sinh ra các token, thử nghiêm với các mô hình dạng “pre-trained model” như BERT và GPT2.
05/2024	Xây dựng các chỉ số đánh giá mô hình cho ra kết quả đúng với câu mô tả đầu vào hay chưa.	Cải tiến mô hình với khả năng trích xuất các thuộc tính liên quan đến cảm xúc âm nhạc thay vì chỉ có các thuộc tính liên quan đến kỹ thuật; Tinh chỉnh mô hình.
06/2024	Phát triển frontend ứng dụng web. Hậu xử lý để chuyển kết quả mô hình thành file midi.	Phát triển frontend ứng dụng web. Viết API để trả ra kết quả sinh nhạc từ văn bản người dùng nhập vào của mô hình. Triển khai mô hình và ứng dụng.
07/2024	Viết báo cáo, soạn bài thuyết trình, và chuẩn bị bảo vệ.	Viết báo cáo, soạn bài thuyết trình, và chuẩn bị bảo vệ.

Tài liệu

- [1] P. Lu, X. Xu, C. Kang, B. Yu, C. Xing, X. Tan, and J. Bian, "Musecoco: Generating symbolic music from text," 2023.
- [2] A. Agostinelli, T. I. Denk, Z. Borsos, J. Engel, M. Verzetti, A. Caillon, Q. Huang, A. Jansen, A. Roberts, M. Tagliasacchi, M. Sharifi, N. Zeghidour, and C. Frank, "Musiclm: Generating music from text," 2023.
- [3] T. Hayes, S. Zhang, X. Yin, G. Pang, S. Sheng, H. Yang, S. Ge, Q. Hu, and D. Parikh, "Mugen: A playground for video-audio-text multimodal understanding and generation," 2022.

XÁC NHẬN
CỦA NGƯỜI HƯỚNG DẪN
(Ký và ghi rõ họ tên)


Trần Duy Hoàng

TP. Hồ Chí Minh, ngày 01 tháng 07 năm 2024
NHÓM SINH VIÊN THỰC HIỆN
(Ký và ghi rõ họ tên)


Ngô Phi Hùng Phạm Quốc Vượng

Mục lục

Nhận xét của GV hướng dẫn	i
Nhận xét của GV phản biện	ii
Lời cảm ơn	ii
Đề cương	iii
Mục lục	xiii
Tóm tắt	xviii
1 Giới thiệu	1
1.1 Đặt vấn đề	1
1.2 Mục tiêu đề tài	2
1.3 Cách tiếp cận	3
1.3.1 Hướng tiếp cận	3
1.3.2 Các bước tiếp cận	3
1.4 Đóng góp	4
1.5 Bố cục của báo cáo	5
2 Các công trình liên quan	6
2.1 Cơ sở lý thuyết	6
2.1.1 Lý thuyết âm nhạc	6
2.1.2 Kỹ thuật huấn luyện nền tảng	16

2.1.3	Fairseq	19
2.2	Các nghiên cứu liên quan	20
2.3	Dữ liệu huấn luyện	21
2.3.1	Nguồn dữ liệu	21
2.3.2	Kỹ thuật hỗ trợ thu thập	22
2.3.3	Các định dạng dữ liệu chính	23
2.4	Mô hình ngôn ngữ lớn	34
2.4.1	Transformer	34
2.4.2	BERT	37
2.4.3	GPT2	38
2.5	Các loại độ đo	39
2.5.1	Độ đo hiệu suất mô hình trích xuất đặc trưng . . .	39
2.5.2	Độ đo hiệu suất mô hình sinh nhạc	39
2.5.3	Độ đo nhạc tính	40
3	Phương pháp đề xuất	42
3.1	Xác định chi tiết vấn đề	42
3.2	Tổng quan về phương pháp	42
3.2.1	Kiến trúc tổng quát	42
3.2.2	Lí do lựa chọn	43
3.2.3	Các bước thực hiện	45
3.3	Chuẩn bị dữ liệu	47
3.3.1	Dữ liệu ngôn ngữ tự nhiên	47
3.3.2	Dữ liệu âm nhạc	51
3.4	Huấn luyện mô hình	55
3.4.1	Mô hình trích xuất đặc trưng câu văn bản	55
3.4.2	Mô hình sinh nhạc	57
3.5	Hậu xử lý dữ liệu	63
3.6	Phương pháp đánh giá mô hình	64
3.6.1	Đánh giá mô hình trích xuất đặc trưng	64
3.6.2	Đánh giá mô hình sinh nhạc	65

4 Kết quả thí nghiệm	66
4.1 Kết quả huấn luyện mô hình	66
4.1.1 Mô hình trích xuất đặc trưng	66
4.1.2 Mô hình sinh nhạc	68
4.2 Phần mềm demo	71
5 Kết luận	73
Tài liệu tham khảo	75

Danh sách hình

2.1	Ví dụ một đoạn nhạc nhịp $\frac{3}{4}$	9
2.2	Hình minh họa 12 nốt nhạc cơ bản với đàn piano	12
2.3	Kiến trúc LoRA	19
2.4	Một đoạn nhạc được lưu trữ dưới dạng MIDI khi hiển thị trên phần mềm làm nhạc	25
2.5	Minh họa dạng hình ảnh cho phương pháp token hoá âm nhạc CPWord[11]	30
2.6	Minh họa dạng hình ảnh cho phương pháp token hoá âm nhạc MIDI-Like[11]	30
2.7	Minh họa dạng hình ảnh cho phương pháp token hoá âm nhạc REMI[11]	30
2.8	Minh họa dạng hình ảnh cho phương pháp token hoá âm nhạc Structured MIDI Encoding[11]	30
2.9	Minh họa dạng hình ảnh cho phương pháp token hoá âm nhạc TSD[11]	31
2.10	Minh họa dạng hình ảnh cho phương pháp token hoá âm nhạc MMM[3]	31
2.11	Các giá trị nốt và tốc độ bản nhạc của một đoạn dữ liệu âm nhạc khi chuyển về dạng REMI trong Python[12]	32
2.12	Một đoạn dữ liệu âm nhạc dạng văn bản của MuseCoco	33
2.13	Kiến trúc Transformer	36
3.1	Kiến trúc tổng quát của mô hình	43

3.2	Quy trình thực nghiệm	46
3.3	Số lần xuất hiện trung bình của các giá trị cao độ (pitch) trong 29 038 đoạn nhạc từ bộ dữ liệu Hooktheory	54
3.4	Số lần xuất hiện trung bình của các giá trị cao độ (pitch) cơ bản trong 29 038 đoạn nhạc từ bộ dữ liệu Hooktheory . .	54
4.1	Số lần xuất hiện trung bình của các giá trị cao độ (GPT2 LoRA).	70
4.2	Số lần xuất hiện trung bình của các giá trị cao độ cơ bản (GPT2 LoRA).	70
4.3	Kiến trúc client-server	71
4.4	Giao diện tổng thể	72
4.5	Giao diện khi có nhạc sinh ra	72

Danh sách bảng

2.1	Các hình nốt thường gấp	7
2.2	Các dấu lặng thường gấp	7
2.3	Cách gọi tên 12 nốt nhạc cơ bản	13
2.4	Các thông điệp MIDI cơ bản	23
2.5	Đặc tả dữ liệu âm nhạc dạng văn bản của MuseCoco . . .	34
3.1	Tên và giá trị tương ứng của các nhãn trong mỗi câu template từ bài báo MuseCoco đã được rút gọn[9]	51
3.2	Các hyperparameter được sử dụng trong quá trình huấn luyện mô hình BERT	57
3.3	Các hyperparameter được sử dụng trong quá trình huấn luyện mô hình GPT-2 và kỹ thuật LoRA	61
3.4	Bảng tham số cấu hình cho mô hình GPT-2	62
3.5	Bảng tham số cấu hình cho LoraConfig.	62
3.6	Các tham số của mô hình Casual Linear Transformer . . .	63
4.1	Instrument Accuracy	67
4.2	Categories Accuracy	67
4.3	Các tham số cho hàm generate	68
4.4	So sánh kết quả giữa hai mô hình.	68

Tóm tắt

Sau cơn sốt ChatGPT vào năm 2022, các phần mềm phát triển dựa trên trí tuệ nhân tạo (AI) ngày càng nhiều và trở nên phổ biến, mang lại lợi ích cho nhiều lĩnh vực như lập trình, đồ họa, phim ảnh, truyền thông, v.v. Sản xuất âm nhạc trên máy tính cũng không ngoại lệ. Với nhiều ứng dụng như sinh nhạc theo dạng text-to-audio, text-to-midi¹, AI hỗ trợ phối trộn âm thanh (mixing), v.v, sự kết hợp giữa trí tuệ nhân tạo và sáng tạo âm nhạc mở ra nhiều cơ hội và tiềm năng hứa hẹn. Dựa trên xu hướng đó, nhóm chọn thực hiện đề tài “Hệ thống AI hỗ trợ sáng tác nhạc” với hướng “sinh nhạc theo dạng text-to-midi”. Bằng các kỹ thuật chủ đạo như LoRA, Masked Language Modelling, Casual Language Modelling, nhóm mong muốn kết hợp các nghiên cứu đã có và kiến thức của bản thân để tạo ra một hệ thống AI hỗ trợ sáng tác nhạc, giúp khơi dậy và duy trì nguồn cảm hứng cho người làm nhạc trên máy tính nói riêng và người sáng tạo âm nhạc nói chung trong quá trình tạo ra các tác phẩm độc đáo và phong phú.

Trong quá trình nghiên cứu và thử nghiệm, nhóm đã thực nghiệm trên nhiều mô hình ngôn ngữ như BERT, GPT-2 để tối ưu hóa mô hình phù hợp với bài toán sinh nhạc. Nhóm đã đạt được một số thành công nhất định khi hoàn thiện pipeline để sinh ra bài hát ở dạng MIDI. Bằng việc sử dụng thêm kỹ thuật LoRA (Low-Rank Adaptation), nhóm đã tối ưu tài nguyên để phù hợp với lượng tài nguyên hiện có. Ngoài ra, nhóm cũng đề xuất phương pháp hậu xử lý dữ liệu theo hướng điền vào dữ liệu bị khuyết (fill

¹MIDI - Musical Instrument Digital Interface: tương tự việc con người đọc các ký hiệu trên sheet nhạc, máy tính đọc các ký hiệu đó nhưng được biểu diễn ở dạng MIDI để hiểu được bản nhạc.

missing data) để giải quyết trực tiếp từ các trường hợp sinh nhạc không đúng cấu trúc hoàn toàn. Cuối cùng, nhóm áp dụng mô hình vào sản phẩm thực tế là một trang web cho phép nhập vào văn bản ra lệnh (prompt) và nhận về bản nhạc tương ứng. Với những thành quả trên, nhóm mong rằng “Hệ thống AI hỗ trợ sáng tác nhạc” sẽ trở thành một công cụ hữu ích cho cộng đồng âm nhạc, và là một nguồn hữu ích cho những ai nghiên cứu trong cùng lĩnh vực cần tham khảo.

Chương 1

Giới thiệu

Chương này trình bày về lý do chọn đề tài, mục tiêu, cách tiếp cận, những giải pháp, đóng góp nghiên cứu và tổng quan cấu trúc cuốn luận này.

1.1 Đặt vấn đề

Lĩnh vực nghiên cứu về mô hình hóa âm nhạc hiện nay đã có nhiều thành tựu nổi bật với các công trình tiêu biểu như mô hình sinh nhạc MuseNet[14] từ OpenAI, MusicGen[2] từ Facebook, hay MuseCoco[9] từ Microsoft. Đó đều là những đóng góp quan trọng và mang tính nền tảng cho những người đi sau tiếp tục nghiên cứu, cải tiến, và phát triển.

Tuy nhiên, khi nghiên cứu và áp dụng các phương pháp này, một vấn đề phổ biến là sự khó khăn trong việc trích xuất và chuyển đổi các mô tả cảm tính của người dùng thành các đặc tính kỹ thuật cụ thể. Người dùng phổ thông thường mô tả bài hát dựa trên cảm xúc của họ. Các mô tả như "một bài nhạc chậm buồn" cần phải được chuyển đổi từ ngôn ngữ tự nhiên sang các giá trị kỹ thuật tương ứng như tốc độ, giọng và nhịp cụ thể của bản nhạc. Điều này gặp nhiều thách thức do các mô tả cảm tính thường không có định nghĩa rõ ràng và nhất quán, làm cho việc xác định các thông số kỹ thuật cụ thể trở nên khó khăn và phức tạp. Mặc dù người

dùng có thể diễn đạt cảm nhận này một cách dễ dàng, việc thiếu kiến thức kỹ thuật chuyên sâu khiến cho quá trình chuyển đổi các mô tả này thành các thông số kỹ thuật cụ thể trở nên khó khăn và làm cho việc sáng tác một bài nhạc theo ý muốn khó kiểm soát.

Do đó, vấn đề đặt ra là nghiên cứu và phát triển các phương pháp sinh MIDI mới, mà không phải âm thanh, có khả năng tạo ra các bản nhạc mới một cách tự động và có thể điều chỉnh theo phong cách âm nhạc đã cho trước. Đặc biệt đối với những người làm nhạc, việc tạo ra và điều chỉnh các bản nhạc theo ý tưởng thường gặp khó khăn khi chỉ sử dụng các mô hình hóa âm thanh thuần tuý bằng các tập tin âm thanh như mp3, wav, v.v. Phương pháp này sẽ kết hợp cả mô tả kỹ thuật chi tiết và các đặc điểm cảm xúc của bài nhạc để tạo ra các dữ liệu MIDI phù hợp với mục đích sáng tạo âm nhạc hiệu quả và sinh động hơn. Lý do chọn sinh MIDI thay vì audio là do tính tiện dụng và sự phù hợp với nhu cầu của những người làm nhạc, là lựa chọn tối ưu cho việc nghiên cứu và phát triển các phương pháp mô hình hóa âm nhạc hiện đại.

Mục tiêu của nghiên cứu là phát triển một hệ thống sinh MIDI đáp ứng được nhu cầu của người làm nhạc trong việc tạo ra các bản nhạc mới, từ đó mở ra những tiềm năng ứng dụng rộng rãi trong ngành công nghiệp âm nhạc và công nghệ giải trí.

1.2 Mục tiêu đề tài

Với mục tiêu chung là giúp người sáng tác âm nhạc mở rộng phạm vi sáng tạo và có nguồn cảm hứng không giới hạn, nhóm đánh giá việc sử dụng “Hệ thống AI hỗ trợ sáng tác nhạc” là một trong những cách nhanh và có hiệu quả tức thời nhất nhờ các lợi ích: tăng cường hiệu suất công việc (thay vì phải dành nhiều thời gian cho việc tạo ra các giai điệu và hoà âm cơ bản, người sáng tác có thể tập trung vào các khâu phức tạp hơn như tô điểm cho giai điệu và hoà âm do công cụ tạo ra để có thành quả tốt và đúng ý muốn nhất, giành thời gian tiết kiệm được cho khâu phối khí,

hoà trộn - mixing, và các khâu phức tạp hơn sau đó); mở rộng khả năng và phạm vi sáng tạo nhờ việc cung cấp các nét nhạc với cách đi giai điệu và nhịp điệu mà người viết nhạc có thể chưa từng nghĩ đến; hỗ trợ người không biết nhạc lý vẫn có thể sáng tác; v.v. Để đạt được mục tiêu chung đó, nhóm đưa ra các mục tiêu cụ thể như sau:

1.3 Cách tiếp cận

1.3.1 Hướng tiếp cận

- Nâng cao tính toàn diện của mô hình bằng việc kết hợp khả năng sinh text-to-midi và khả năng hiểu mô tả nhạc theo cả phương diện kỹ thuật lẫn theo ngữ cảnh.
- Xây dựng một mô hình có khả năng liên tục sinh nội dung dựa trên câu prompt trước đó (continuous generating).
- Phát triển thành mô hình đa ngôn ngữ thay vì chỉ có tiếng Anh.
- Hậu xử lý kết quả của mô hình để đảm bảo tính chính xác và mạch lạc.
- Triển khai mô hình và phát triển một ứng dụng web giúp người dùng có thể dễ dàng tiếp cận.

1.3.2 Các bước tiếp cận

1. Tìm hiểu về lĩnh vực âm nhạc: các khái niệm trong âm nhạc như phách, nhịp, cao độ, trường độ, thang âm, v.v; các nguyên tắc hoà âm cơ bản; cách phối hợp các nhạc cụ trong bản nhạc; các bước để sáng tác ra một bản nhạc; v.v.
2. Khảo sát, đánh giá các công trình nghiên cứu liên quan mà nhóm đề cập ở phần 2.2.

3. Thu thập dữ liệu từ các nguồn được nêu ở phần 2.3.1: tải xuống thông thường với các bộ dữ liệu được đóng gói sẵn; dùng Selenium (Python) để cào dữ liệu từ trang web và kỹ thuật đa luồng để tăng hiệu suất cào với dữ liệu chưa đóng gói sẵn.
4. Viết công cụ chuyển đổi cấu trúc dữ liệu giữa các mô hình cần thử nghiệm.
5. Tiếp cận các mô hình được đào tạo trước (BERT, GPT2) để thử nghiệm các tác vụ trích xuất đối tượng và sinh văn bản.
6. Nghiên cứu các kỹ thuật LoRA, Masked Language Modelling và Causal Language Modelling, kết hợp với các nghiên cứu hiện có, để phát triển mô hình có khả năng hiểu văn bản mô tả âm nhạc theo cả khía cạnh kỹ thuật và sinh nhạc dựa trên các khía cạnh đó.
7. Nghiên cứu phương pháp tối ưu và nâng cao chất lượng nhạc sinh ra (ở dạng midi) của các mô hình từ những nghiên cứu trước đó.
8. Phát triển công cụ chuyển đổi thông tin từ định dạng đầu ra của mô hình thành định dạng MIDI và thực hiện hậu xử lý kết quả.
9. Phát triển ứng dụng web, triển khai ứng dụng và mô hình.

1.4 Đóng góp

1. Mô hình sinh nhạc bằng cách nhận đầu vào là văn bản mô tả đặc điểm bản nhạc bằng ngôn ngữ tự nhiên (tiếng Anh hoặc tiếng Việt).
2. Giải pháp hậu xử lý kết quả của mô hình để đảm bảo định dạng dữ liệu đúng bằng phương pháp nội suy.
3. Ứng dụng web/Phần mềm sinh ra nhạc theo yêu cầu.

1.5 Bố cục của báo cáo

[Khóa luận](#)

[Khóa luận](#)

Phần còn lại của [bài viết](#) được tổ chức như sau: Bắt đầu từ phần 2, chúng tôi giới thiệu các cơ sở lý thuyết liên quan về các kỹ thuật cũng như các mô hình ngôn ngữ, các thông số kỹ thuật để đo lường và đánh giá. Tiếp theo, phần 3 sẽ giải thích chi tiết về các phương pháp và chiến lược đề xuất. Kết quả thực nghiệm được trình bày trong phần 4, và cuối cùng, chúng tôi rút ra kết luận chính và thảo luận về các phát triển tương lai có tiềm năng trong phần 5.

Chương 2

Các công trình liên quan

Chương trình này trình bày cơ sở lý thuyết, lý luận khoa học, phân tích các nghiên cứu từ các bài báo tham khảo, các vấn đề về mô hình ngôn ngữ lớn cần được cải tiến trong đó.

Các phương pháp nghiên cứu được sử dụng trong các đề tài này thuộc nhiều lĩnh vực khác nhau như học máy, học sâu, ngôn ngữ lớn, kết hợp các phương pháp **kỹ thuật kỹ thuật** như LoRA dùng để hoàn thiện các mô hình ngôn ngữ lớn, **kỹ thuật kỹ thuật** sinh dữ liệu; thử nghiệm mục tiêu hợp lý để sử dụng một cách có hiệu quả.

2.1 Cơ sở lý thuyết

2.1.1 Lý thuyết âm nhạc

Lý thuyết âm nhạc dài quá --> tóm tắt khái niệm trong 1 trang, còn lại đưa xuống phụ lục

Lý thuyết âm nhạc là nền tảng cho việc hiểu và sáng tạo âm nhạc. Nó bao gồm các khái niệm, nguyên tắc giúp người học nắm bắt và sử dụng các yếu tố âm nhạc một cách hiệu quả. Trong mục này, chúng tôi tập trung vào những lý thuyết cơ bản nhất, cần thiết nhất và thường xuyên sử dụng trong **Khóa luận**, nhằm giúp người đọc có cái nhìn tổng quát, giảm sự phức tạp trong quá trình hiểu các ý tưởng, cũng như không bị quá tải thông tin trong quá trình đọc.

2.1.1.1 Hình nốt, dấu lặng, và trường độ

Khi hát một bài hát bất kỳ, ta luôn rơi vào một trong hai trường hợp: hát theo lời bài hát, hoặc tạm ngưng ở vị trí không có lời. Để biểu diễn nốt nhạc được hát ở vị trí có lời, ta dùng hình nốt (xem các hình nốt thường gặp ở bảng 2.1). Để biểu diễn các vị trí tạm ngưng hát, ta dùng dấu lặng (xem các dấu lặng thường gặp ở bảng 2.2). Mỗi nốt hoặc dấu lặng được hát hoặc nghỉ trong thời gian bao lâu được xác định bằng trường độ tương ứng của hình nốt hoặc dấu lặng đó.

Tên hình nốt	Tên tiếng Anh	Cách viết khác	Ký hiệu
Nốt tròn	Whole note		○
Nốt trắng	Half note	$\frac{1}{2}$ note	♩
Nốt đen	Quarter note	$\frac{1}{4}$ note	♪
Nốt móc đơn	Eighth note	$\frac{1}{8}$ note	♫
Nốt móc đôi	Sixteenth note	$\frac{1}{16}$ note	♪♪
Nốt móc ba	Thirty-second note	$\frac{1}{32}$ note	♪♪♪

Bảng 2.1: Các hình nốt thường gặp

Tên dấu lặng	Tên tiếng Anh	Cách viết khác	Ký hiệu
Dấu lặng tròn	Whole rest		—
Dấu lặng trắng	Half rest	$\frac{1}{2}$ rest	—
Dấu lặng đen	Quarter rest	$\frac{1}{4}$ rest	♪
Dấu lặng móc đơn	Eighth rest	$\frac{1}{8}$ rest	♫
Dấu lặng móc đôi	Sixteenth rest	$\frac{1}{16}$ rest	♪♪
Dấu lặng móc ba	Thirty-second rest	$\frac{1}{32}$ rest	♪♪♪

Bảng 2.2: Các dấu lặng thường gặp

Mối tương quan trường độ giữa các hình nốt:

$$\textcircled{o} = 2\textcircled{d} = 4\textcircled{b} = 8\textcircled{m} = 16\textcircled{n} = 32\textcircled{h}$$

Mối tương quan trường độ giữa các dấu lặng:

$$\textcircled{-} = 2\textcircled{-} = 4\textcircled{x} = 8\textcircled{y} = 16\textcircled{z} = 32\textcircled{w}$$

Các hình nốt và dấu lặng có hậu tố giống nhau thì có trường độ bằng nhau. Ví dụ: nốt tròn có trường độ bằng dấu lặng tròn, nốt đen có trường độ bằng dấu lặng đen, v.v. Và quan trọng nhất, việc so sánh trường độ giữa các hình nốt và dấu lặng chỉ có ý nghĩa khi chúng nằm trong một hoặc các bản nhạc có cùng loại nhịp (time signature, xem mục 2.1.1.2) và cùng ký hiệu tốc độ (tempo, xem mục 2.1.1.3).

Ngoài các ký hiệu hình nốt và dấu lặng trên còn có dấu chấm dôi đặt bên phải hình nốt, làm tăng trường độ của hình nốt hoặc dấu lặng đó thêm $\frac{1}{2}$. Ví dụ:

$$\textcircled{o}\cdot = \textcircled{o} + \textcircled{d} = 3\textcircled{d}$$

Việc đặt liên tiếp các dấu chấm dôi bên phải một hình nốt hoặc dấu lặng sẽ làm tăng trường độ của hình nốt hoặc dấu lặng đó theo công thức:

$$duration(n) = \frac{d}{1} + \frac{d}{2} + \frac{d}{4} + \dots + \frac{d}{2^n},$$

với *duration* là trường độ, *n* là số dấu chấm dôi, *d* là trường độ nốt gốc. Ví dụ:

$$\textcircled{o}\cdot\cdot = \textcircled{o} + \textcircled{d} + \textcircled{b} = 7\textcircled{d}$$

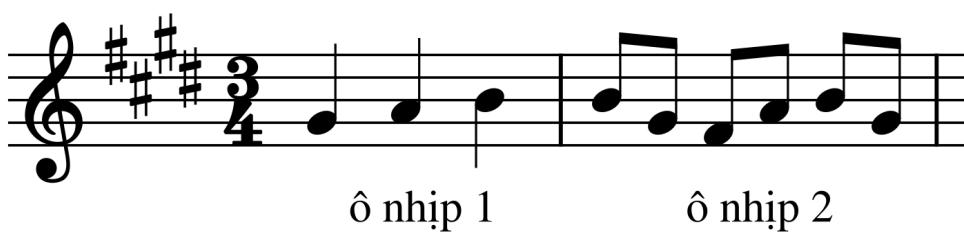
2.1.1.2 Ô nhịp và các loại nhịp

Ô nhịp (measure, hay còn gọi là bar) là các cụm nốt nhạc (và dấu lặng nếu có) được chia ra từ các nốt (và dấu lặng) ghi theo thứ tự trên sheet nhạc¹. Mỗi ô nhịp được chia thành nhiều phách, mỗi phách có trường độ được quy định dựa trên trường độ của phách đơn vị.

Thông tin trường độ phách đơn vị và độ dài mỗi ô nhịp được suy ra từ ký hiệu nhịp (time signature) của bài nhạc. Ký hiệu nhịp thường được đặt ở phần đầu của sheet nhạc, và có thể thay đổi trong suốt bản nhạc. Ký hiệu nhịp bao gồm hai con số, một số ở trên và một số ở dưới, giống như một phân số², với ý nghĩa:

- Số ở trên (numerator): Độ dài ô nhịp tính theo số *phách đơn vị*³.
- Số ở dưới (denominator): Trường độ của *phách đơn vị*.

Ví dụ với một nhịp thông dụng - nhịp $\frac{3}{4}$ (xem hình 2.1), số 3 ở trên cho biết mỗi ô nhịp có độ dài là 3 phách đơn vị, số 4 ở dưới cho biết mỗi phách đơn vị có trường độ bằng trường độ nốt $\frac{1}{4}$ (nốt đen). Ngoài nhịp $\frac{3}{4}$, còn có các nhịp thông dụng khác như $\frac{2}{4}$, $\frac{4}{4}$, $\frac{6}{8}$, v.v; hoặc các nhịp ít phổ biến hơn như $\frac{9}{8}$, $\frac{12}{8}$, v.v.



Hình 2.1: Ví dụ một đoạn nhạc nhịp $\frac{3}{4}$

¹Sheet nhạc là một loại văn bản bài nhạc. Trong đó, các dòng kẻ nhạc và nốt nhạc mô tả giai điệu, hoà âm, v.v, của bản nhạc là nội dung chính.

²Chỉ có điểm giống về cách ký hiệu, không áp dụng phép toán rút gọn như phân số.

³“Phách đơn vị” là thuật ngữ âm nhạc tạm thời do người viết đề xuất để phân biệt với thuật ngữ “phách” trong luận văn (xem chi tiết ở đoạn cuối của mục 2.1.1.2) do các thuật ngữ liên quan đến vấn đề này còn gây nhầm lẫn cho người mới. Thuật ngữ tạm thời này không tương đương với thuật ngữ “beat unit” (trong trường hợp người đọc tìm kiếm thông tin trong các tài liệu tiếng Anh).

Những thông tin được trình bày ở trên đã đủ để sử dụng cho việc lập trình tính toán trong luận văn. Tuy nhiên, *phách* và *phách đơn vị* của một nhịp là hai cụm từ thường bị nhầm lẫn, nên để tránh hai thuật ngữ này bị hiểu sai hoặc hiểu phiến diện, chúng tôi xin phép gợi mở thêm một số thông tin ở đoạn kế tiếp để người đọc có thể tìm hiểu khi có nhu cầu.

Trong lý thuyết âm nhạc, số ở trên của ký hiệu nhịp chỉ có ý nghĩa chỉ ra độ dài ô nhịp theo số *phách đơn vị*, không có ý nghĩa chỉ ô nhịp có bao nhiêu *phách*; số ở dưới chỉ có ý nghĩa chỉ ra trường độ mỗi *phách đơn vị* là hình nốt nào, không có ý nghĩa chỉ *phách* có trường độ là hình nốt nào. Với nhịp $\frac{3}{4}$, ta thấy rất đơn giản khi số phách và số phách đơn vị trong một ô nhịp đều là 3, mỗi phách và phách đơn vị đều có trường độ là nốt $\frac{1}{4}$, chúng thống nhất với nhau. Với trường hợp nhịp $\frac{6}{8}$, tuy số phách đơn vị trong mỗi ô nhịp là 6, mỗi phách đơn vị là nốt $\frac{1}{8}$, nhưng số phách trong mỗi ô nhịp lại là 2, mỗi phách có trường độ bằng 3 nốt $\frac{1}{8}$. Hoặc với nhịp $\frac{7}{8}$, trong khi số phách đơn vị trong ô nhịp và trường độ phách đơn vị thống nhất với ký hiệu nhịp, số phách trong ô nhịp lại là 3, trường độ của mỗi phách lần lượt là 3 nốt $\frac{1}{8}$, 3 nốt $\frac{1}{8}$, 2 nốt $\frac{1}{8}$, hoặc bất kỳ sự hoán đổi vị trí nào giữa 3 cụm nốt khác nhau về trường độ này. Để có thêm nhiều thông tin, người đọc có thể tìm kiếm với các từ khoá: “*nhip đơn và nhịp kép*”, “*simple vs. compound time signatures*”, “*division vs. subdivision in music*”, v.v.

2.1.1.3 Tốc độ bản nhạc

Tốc độ của bản nhạc (tempo) xác định tốc độ của bản nhạc, được đo bằng số lượng phách trong mỗi phút (BPM - beats per minute). Trên sheet nhạc, tốc độ được ký hiệu theo quy tắc:

$$\text{Hình nốt lũy làm chuẩn} = \text{Số phách mỗi phút}.$$

Hình nốt lũy làm chuẩn có thể là *phách* hoặc *phách đơn vị* của nhịp của bản nhạc (xem mục 2.1.1.2 để phân biệt hai khái niệm này) tùy theo loại nhịp. Ví dụ:

- Một bản nhạc nhịp $\frac{4}{4}$, ký hiệu tốc độ $\text{♩} = 128$, sẽ có 128 nhịp mỗi phút, mỗi nhịp tương ứng với một nốt $\frac{1}{4}$.
- Một bản nhạc nhịp $\frac{6}{8}$, tốc độ 100BPM, có hai cách ký hiệu nhịp: hoặc $\text{♩} = 100$ với ý nghĩa bài nhạc có tốc độ 100 nhịp mỗi phút, mỗi nhịp tương ứng với 1 nốt đen chấm dôi (bằng 3 nốt $\frac{1}{8}$) (tạm gọi là *cách 1 - dùng phách*); hoặc $\text{♩} = 100$ với ý nghĩa bài nhạc có tốc độ 100 nhịp mỗi phút, mỗi nhịp tương ứng với 1 nốt $\frac{1}{8}$ (tạm gọi là *cách 2 - dùng phách đơn vị*).

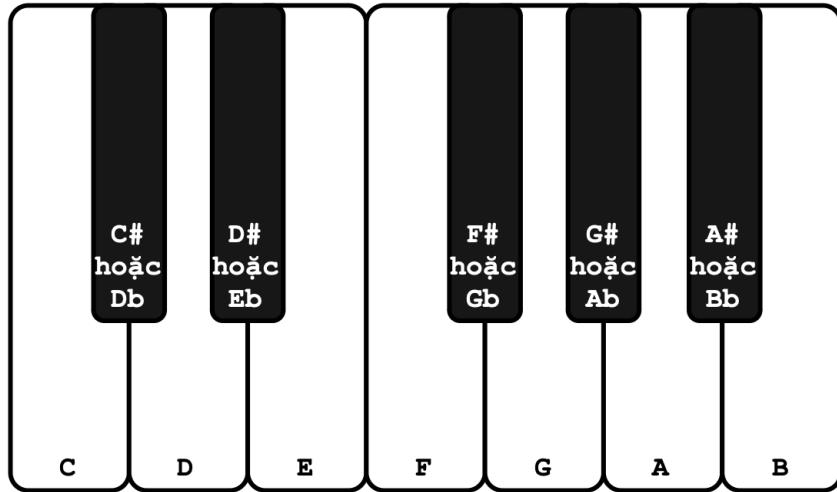
Về bản chất (xem mục 2.1.1.2, đoạn cuối, về *phách và phách đơn vị*), với nhịp $\frac{6}{8}$, cách ký hiệu thứ nhất là phù hợp hơn. Tuy nhiên, hiện nay, nhiều phần mềm làm việc với âm nhạc cho phép sử dụng cả hai cách ký hiệu như phần mềm soạn thảo sheet nhạc MuseScore 4 (soạn thảo âm nhạc bằng các ký hiệu âm nhạc), hoặc hoạt động mặc định theo cách ký hiệu thứ hai như phần mềm sản xuất âm nhạc FL Studio 21 (soạn thảo âm nhạc dựa trên MIDI). Trong luận văn này, chúng tôi chọn lập trình dựa trên cách ký hiệu thứ hai với lí do cách này có sự tương đồng cao với các số trên ký hiệu nhịp, tạo sự thuận tiện cho việc tính toán với các loại nhịp phức tạp.

2.1.1.4 Cao độ

Khi một câu nhạc được hát lên, ngoài diễn đạt mỗi từ trong lời bài hát cần ngân dài bao lâu - tương ứng với trường độ của nốt nhạc (xem mục 2.1.1.1), người hát còn diễn đạt mỗi từ đó được hát ở những nốt cao thấp thế nào - tương ứng với cao độ của nốt nhạc. Ở các cấp giáo dục phổ thông, chúng ta được biết 7 nốt nhạc cơ bản: Đô, Ré, Mi, Fa, Sol, La, và Si, ký hiệu bằng chữ cái tiếng Anh lần lượt là C, D, E, F, G, A, và B, tương ứng với các phím màu trắng của đàn piano. Tuy nhiên, trên piano còn có các phím màu đen (xem hình minh họa 2.2⁴). Đây là lúc ta cần đến ký hiệu dấu thăng (♯, tiếng Anh là sharp) và dấu giáng (♭,

⁴Trên các loại đàn phím (keyboard) như piano, những cụm phím trắng và đen theo bố cục này được xếp liên tiếp nhau để tạo thành dây phím của đàn.

tiếng Anh là flat) để có thể gọi tên các nốt nhạc tương ứng với các phím màu đen này.



Hình 2.2: Hình minh họa 12 nốt nhạc cơ bản với đàn piano

Hai phím đàm trắng hoặc đen bất kỳ nằm kế nhau trên đàn piano có khoảng cách cao độ là $\frac{1}{2}$ cung⁵ (hay còn gọi là nửa cung⁶). Bằng cách thêm vào bên phải của tên nốt nhạc một dấu thăng với tác dụng tăng cao độ lên nửa cung, hoặc dấu giáng để giảm cao độ xuống nửa cung, ta có thể gọi tên các nốt đen một cách dễ dàng. Xem cách gọi tên 12 nốt nhạc cơ bản ở bảng 2.3.

⁵Cung - whole tone hoặc whole step, là đại lượng để tính khoảng cách cao độ giữa hai nốt nhạc.

⁶Trong thuật ngữ tiếng Anh, nửa cung được gọi là semitone hoặc half step.

Ký hiệu	Cách đọc tiếng Anh	Cách đọc tiếng Việt
C	C	Đô
C♯ (hoặc D♭)	C sharp (hoặc D flat)	Đô thăng (hoặc Rê giáng)
D	D	Rê
D♯ (hoặc E♭)	D sharp (hoặc E flat)	Rê thăng (hoặc Mi giáng)
E	E	Mi
F	F	Fa
F♯ (hoặc G♭)	F sharp (hoặc G flat)	Fa thăng (hoặc Sol giáng)
G	G	Sol
G♯ (hoặc A♭)	G sharp (hoặc A flat)	Sol thăng (hoặc La giáng)
A	A	La
A♯ (hoặc B♭)	A sharp (hoặc B flat)	La thăng (hoặc Si giáng)
B	B	Si

Bảng 2.3: Cách gọi tên 12 nốt nhạc cơ bản

Ở bảng trên, về mặt lý thuyết, các nốt C, E, F, và B có tồn tại cách ký hiệu biểu diễn bằng nốt kế nó và dấu hoá, ví dụ: C♭ tương đương với B, E♯ tương đương với F, v.v. Tuy nhiên, việc áp dụng cách ký hiệu như ví dụ cho bốn nốt trên rất ít khi được sử dụng, do gây ra một số vấn đề phức tạp khi đọc và chơi bản nhạc (sẽ được giải thích ở mục 2.1.1.5 về thang âm), nên chúng tôi không liệt kê để bảng sát với thực tế.

2.1.1.5 Thang âm

Khi một người đang hát, nếu người nghe cảm thấy một nốt nhạc có cao độ không hoà quyện và chêch hẵn khỏi nhạc nền, ta thường nói người hát đang “hát sai tone”, “hát bị chênh cao độ”, hoặc “hát bị ngang”. Các nốt

gây ra cảm giác không đúng đó cho người nghe chính là những nốt bị lệch khỏi thang âm của bài nhạc. Giai điệu và hoà âm của những bản nhạc phổ biến trên thế giới lẫn Việt Nam hiện tại đều được xây dựng dựa trên một hoặc một vài thang âm⁷ - công cụ để xác định nốt nhạc nào nên và không nên sử dụng trong bản nhạc, hay nói cách khác, có thể xác định nốt nhạc “sai tone” bằng cách sử dụng thang âm.

Thang âm (scale) là một dãy các nốt nhạc theo một “trật tự về khoảng cách cao độ” xác định. Thang âm cơ bản và phổ biến nhất là thang âm trưởng (major scale) và thang âm thứ (minor scale). Một thang âm trưởng bao gồm các nốt nhạc theo thứ tự: cung – cung – nửa cung – cung – cung – cung – nửa cung. Ví dụ: thang âm Đồ trưởng (C major) gồm các nốt: Đồ, Rê, Mi, Fa, Sol, La, Si, Đồ.

2.1.1.6 Quãng nhạc

Quãng nhạc (interval, gọi tắt là quãng) là khoảng cách cao độ giữa hai nốt nhạc giữa hai cao độ. Các quãng cơ bản bao gồm:

- Quãng 1 (Unison): Cùng một cao độ.
- Quãng 2 (Second): Khoảng cách 1 cung (tone) hoặc $\frac{1}{2}$ cung (semi-tone).
- Quãng 3 (Third): Khoảng cách 2 cung hoặc 1,5 cung.
- Quãng 4 (Fourth): Khoảng cách 2,5 cung.
- Quãng 5 (Fifth): Khoảng cách 3,5 cung.
- Quãng 6 (Sixth): Khoảng cách 4,5 cung hoặc 4 cung.
- Quãng 7 (Seventh): Khoảng cách 5,5 cung hoặc 5 cung.

⁷Những bản nhạc này, thuộc kiểu âm nhạc có điệu thức (tonal music), dựa trên một hoặc một vài thang âm để xây dựng, cũng như lí luận, giai điệu và hoà âm. Ngược lại với loại âm nhạc trên là âm nhạc phi điệu thức (atonal music) không dựa trên một thang âm cố định nào để xây dựng bài nhạc mà dựa trên toàn bộ các nốt nhạc có thể có.

- Quãng 8 (Octave): Khoảng cách 6 cung, nốt cùng tên ở cao độ gấp đôi.

2.1.1.7 Các loại hợp âm hai và ba nốt

Hợp âm (chord) là sự kết hợp của ba hoặc nhiều nốt nhạc được vang lên đồng thời. Các hợp âm cơ bản nhất là hợp âm hai nốt (dyad) và hợp âm ba nốt (triad). Hợp âm ba nốt bao gồm ba loại chính:

- Hợp âm trưởng (Major chord): Bao gồm nốt gốc (root), quãng ba trưởng (major third), và quãng năm đúng (perfect fifth). Ví dụ: Hợp âm Đồ trưởng (C Major) gồm các nốt: Đồ (C), Mi (E), Sol (G).
- Hợp âm thứ (Minor chord): Bao gồm nốt gốc, quãng ba thứ (minor third), và quãng năm đúng. Ví dụ: Hợp âm La thứ (A Minor) gồm các nốt: La (A), Dô (C), Mi (E).
- Hợp âm giảm (Diminished chord): Bao gồm nốt gốc, quãng ba thứ, và quãng năm giảm (diminished fifth). Ví dụ: Hợp âm Si giảm (B Diminished) gồm các nốt: Si (B), Rê (D), Fa (F).
- Hợp âm tăng (Augmented chord): Bao gồm nốt gốc, quãng ba trưởng, và quãng năm tăng (augmented fifth). Ví dụ: Hợp âm Đồ tăng (C Augmented) gồm các nốt: Đồ (C), Mi (E), Sol (G).

2.1.1.8 Các loại hợp âm từ bốn nốt trở lên

Hợp âm từ bốn nốt trở lên phức tạp hơn và tạo ra các âm sắc đa dạng hơn. Các hợp âm này bao gồm:

- Hợp âm bảy (Seventh chord): Bao gồm bốn nốt nhạc. Các loại phổ biến:
 - Hợp âm bảy trưởng (Major Seventh): Nốt gốc, quãng ba trưởng, quãng năm đúng, và quãng bảy trưởng. Ví dụ: Đồ bảy trưởng (Cmaj7) gồm Đồ (C), Mi (E), Sol (G), Si (B).

- Hợp âm bảy thứ (Minor Seventh): Nốt gốc, quãng ba thứ, quãng năm đúng, và quãng bảy thứ. Ví dụ: La bảy thứ (Am7) gồm La (A), Đô (C), Mi (E), Sol (G).
 - Hợp âm bảy át (Dominant Seventh): Nốt gốc, quãng ba trưởng, quãng năm đúng, và quãng bảy thứ. Ví dụ: Sol bảy át (G7) gồm Sol (G), Si (B), Rê (D), Fa (F).
 - Hợp âm bảy giảm (Diminished Seventh): Nốt gốc, quãng ba thứ, quãng năm giảm, và quãng bảy giảm. Ví dụ: Si bảy giảm (Bdim7) gồm Si (B), Rê (D), Fa (F), Lab (Ab).
- Hợp âm chín (Ninth chord): Bao gồm năm nốt nhạc, thêm nốt thứ chín vào hợp âm bảy.
 - Hợp âm mười một (Eleventh chord): Bao gồm sáu nốt nhạc, thêm nốt thứ mười một vào hợp âm chín.
 - Hợp âm mười ba (Thirteenth chord): Bao gồm bảy nốt nhạc, thêm nốt thứ mười ba vào hợp âm mười một.

2.1.2 Kỹ thuật huấn luyện nền tảng

2.1.2.1 Masked Language Modelling

Masked Language Modelling (MLM) là một phương pháp huấn luyện mô hình ngôn ngữ, nổi bật với việc sử dụng trong mô hình BERT của Google. MLM hoạt động bằng cách che đi một phần các từ trong câu đầu vào và yêu cầu mô hình dự đoán các từ bị che giấu đó dựa trên ngữ cảnh xung quanh. MLM buộc mô hình phải học cách biểu diễn từ ngữ theo ngữ cảnh của chúng, thay vì chỉ dựa vào ý nghĩa riêng lẻ của từng từ. Điều này giúp mô hình nắm bắt được các mối quan hệ phức tạp giữa các từ trong câu và xây dựng một biểu diễn ngôn ngữ phong phú hơn.

2.1.2.2 Casual Language Modelling

Casual Language Modelling (CLM) là một kĩ thuật xử lý trong lĩnh vực ngôn ngữ tự nhiên, đây là một dạng dùng trong các autoregressive model và dùng để dự đoán từ tiếp theo dựa trên chuỗi từ trước đó, thông thường thì CLM sẽ được sử dụng trong các mô hình như GPT 2, GPT 3 dành cho các tác vụ sinh văn bản hay tóm tắt. Lấy một ví dụ thì giả sử cung cấp cho mô hình câu hướng dẫn (prompt) là “hôm nay trời ...” thì nó sẽ dự đoán từ tiếp theo có thể xuất hiện là “nắng”, “mưa”.

Casual Language Modelling hoạt động dựa trên cách phân tích trên một lượng dữ liệu lớn văn bản để rút ra các mẫu có sẵn hay các quy luật trong ngôn ngữ, nó có thể phát hiện ra các cặp từ vựng thường đi cùng nhau. Từ đó có thể sinh ra dữ liệu văn bản một cách hợp lý nhất.

Trong khóa luận này, dữ liệu âm nhạc cũng tuân theo một quy luật nhất định, thì qua đó, chúng tôi đã tận dụng CLM để tạo ra một mô hình sinh nhạc hiệu quả. Khi triển khai CLM thì nó được huấn luyện bằng cách đưa vào một lượng dữ liệu lớn văn bản để mô hình có thể học được cấu trúc ngữ pháp và ngữ nghĩa của ngôn ngữ.

2.1.2.3 LoRA

LoRA ra đời vào năm 2021 do đội ngũ Microsoft Research phát hành thông qua bài báo LoRA: Low-Rank Adaptation of Large Language Models, sau khi các mô hình ngôn ngữ lớn ra đời thì LoRA được đề xuất như một phương pháp hiệu quả để tinh chỉnh các mô hình ngôn ngữ lớn bằng cách tận dụng Low-rank. Từ khi giới thiệu thì LoRA đã nhanh chóng trở thành một kĩ thuật phổ biến để tinh chỉnh các mô hình ngôn ngữ lớn.

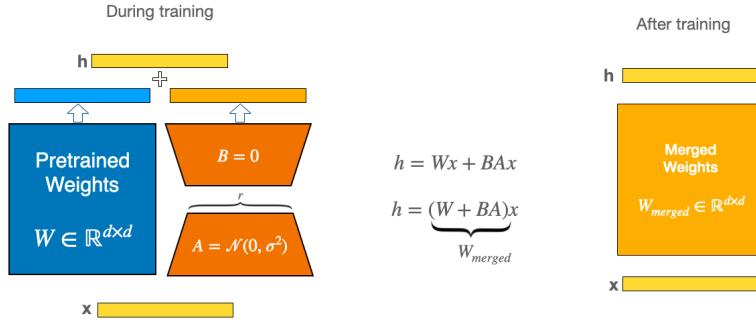
Điều làm LoRA trở nên nổi bật chính là hiệu quả về chi phí mà không làm giảm hiệu suất đáng kể bởi thông thường việc huấn luyện một mô hình ngôn ngữ lớn là rất tốn kém chi phí bởi nó đòi hỏi nhiều tài nguyên. Với sự gia tăng chóng mặt của các mô hình thì LoRA được kì vọng giúp có thể dễ dàng tiếp cận các mô hình này.

LoRA tận dụng khái niệm low-rank matrices (ma trận cấp thấp) để huấn luyện và xử lý mô hình hiệu quả và nhanh chóng hơn. Thông thường các mô hình ngôn ngữ lớn thì tốn nhiều bộ nhớ về GPU như GPT 3 với 175 tỷ tham số, hay các mô hình Llama 70 tỷ tham số, việc huấn luyện và tinh chỉnh các mô hình ngôn ngữ này khá tốn kém và có thể tiêu tốn hàng triệu đô la với các mô hình có kích thước như GPT.

Không giống như việc tinh chỉnh truyền thống toàn bộ mô hình thì LoRA chỉ tập trung vào việc huấn luyện lại tập con tham số trên các lớp nhất định, từ đó giảm chi phí bộ nhớ. LoRA tận dụng các ma trận cấp thấp bởi theo ta được biết thì ma trận cấp thấp có thể phân tích được thành tích hai ma trận nhỏ hơn. Điều này giúp biểu diễn được các thông tin phức tạp với ít tham số hơn. Vậy nên có thể huấn luyện các ma trận này và điều chỉnh trên một tác vụ cụ thể mà không cần phải cập nhật lại toàn bộ mô hình.

Đi sâu vào cơ chế của LoRA, thông thường sẽ gồm các bước:

1. Phân rã ma trận có trọng số lớn. Các mô hình như GPT sử dụng ma trận trọng số không lồ để lưu trữ các tham số. Kỹ thuật LoRA cho phép phân rã các ma trận này thành các ma trận nhỏ hơn thông qua kỹ thuật phân rã ma trận cấp thấp, giúp xấp xỉ một ma trận lớn bằng tích của hai ma trận nhỏ hơn. Điều này làm giảm đáng kể số lượng tham số cần huấn luyện, thường chỉ khoảng [1 số lượng tham số chỉ 1?](#)
2. Huấn luyện ma trận cấp thấp, tất cả các ma trận cấp thấp mới được thêm vào đều được huấn luyện, giúp quá trình nhanh chóng và hiệu quả.
3. Dự đoán kết quả với các bộ trọng số cộng thêm. Sau khi huấn luyện, LoRA không thay đổi mô hình gốc. Trong quá trình huấn luyện, trọng số mới sẽ được cộng thêm vào trọng số ban đầu của mô hình.



Hình 2.3: Kiến trúc LoRA

Ưu điểm khi sử dụng LoRA để tinh chỉnh và huấn luyện mô hình ngoài tối ưu tài nguyên ra còn vì ma trận cấp thấp có kích thước nhỏ nên việc thêm chúng vào mô hình ban đầu sẽ không gây ra độ trễ. Có thể chuyển đổi linh hoạt giữa các tác vụ bởi do ma trận cấp thấp có kích thước nhỏ nên chúng có thể dễ dàng thay đổi linh hoạt khi cần thiết giữa các tác vụ và người dùng khác nhau. Nên từ điều này có thể tạo ra một mô hình đa chức năng đáp ứng nhiều nhu cầu của người dùng mà tiết kiệm với nguồn tài nguyên.

Khi áp dụng LoRA thì chúng tôi đã sử dụng thư viện PEFT (Parameter-Efficient Fine-Tuning) - một phương pháp nhằm mục tiêu giảm số lượng tham số cần cập nhật trong quá trình tinh chỉnh mà vẫn duy trì hiệu suất cao.

2.1.3 Fairseq

Fairseq, một bộ công cụ học sâu phát triển bởi Facebook AI Research (FAIR), được thiết kế đặc biệt để hỗ trợ nghiên cứu và thử nghiệm các mô hình sequence-to-sequence, đóng một vai trò quan trọng trong lĩnh vực xử lý ngôn ngữ tự nhiên (NLP). Fairseq cung cấp một khung làm việc hiệu quả và linh hoạt cho việc triển khai, huấn luyện và thử nghiệm các mô hình NLP tiên tiến như mô hình dịch máy tự động, nhận dạng giọng nói, tóm tắt văn bản và tạo chú thích hình ảnh. Được trang bị các tính năng như huấn luyện song song trên nhiều GPU, hỗ trợ đa ngôn ngữ, và tối ưu

hóa các quá trình huấn luyện với kỹ thuật mới nhất, Fairseq cho phép các nhà nghiên cứu và nhà phát triển nâng cao chất lượng và hiệu quả của các mô hình học sâu.

Trong các ứng dụng thực tế, Fairseq đã chứng minh khả năng đáp ứng với các yêu cầu khắt khe của các hệ thống NLP hiện đại. Từ việc dịch ngôn ngữ tự động với độ chính xác cao đến phát triển các hệ thống nhận dạng giọng nói tiên tiến và tạo ra bản tóm tắt văn bản súc tích, Fairseq đã giúp đẩy nhanh quá trình nghiên cứu và triển khai các giải pháp NLP hiệu quả. Điều này không chỉ củng cố vị thế của FAIR như một trung tâm nghiên cứu AI hàng đầu, mà còn tăng cường khả năng cạnh tranh của Facebook trong lĩnh vực công nghệ AI toàn cầu.

2.2 Các nghiên cứu liên quan

MuseCoco - Generating Symbolic Music from Text[9]: MuseCoco là hệ thống được đề xuất giúp sinh nhạc ở dạng midi từ văn bản mô tả các đặc điểm mang tính kỹ thuật của bản nhạc, ví dụ: “Bản nhạc có nhịp 4/4, viết ở giọng la thứ, có các nhạc cụ piano, guitar và sáo kết hợp với nhau”.

MusicLM: Generating Music From Text[1]: MusicLM là một mô hình được phát triển bởi Google Research, có khả năng tạo ra âm nhạc chất lượng cao dựa trên mô tả bằng văn bản, ví dụ: “Bản nhạc Pop có giọng nữ nhẹ nhàng hát trên phần đệm lead synth⁸ đầy đặn, giai điệu piano êm dịu, tiếng kèn đồng ngân dài, và tiếng trống mạnh mẽ, cùng cảm giác buồn, luyến nhớ, làm người nghe liên tưởng đến những bản nhạc thường phát trên radio.”. Ngoài ra, MusicLM có thể xử lý đồng thời cả văn bản và giai điệu, nghĩa là nó nhận đầu vào là các giai điệu được huýt sáo hoặc ngân nga và văn bản mô tả bổ sung để cho ra giai điệu mới.

MUGEN: A Playground for Video-Audio-Text Multimodal

⁸Còn gọi là lead synthesizer - một loại nhạc cụ điện tử dùng để đệm các tuyến giai điệu chính của bản nhạc.

Understanding and GENeration[6]: MuGen là mô hình được nghiên cứu để hiểu và sinh âm thanh cho video game dựa trên đầu vào là video của một cảnh game và đoạn văn bản mô tả, ví dụ: “Nhân vật chạy đến bên phải để thu thập đồng xu. Sau đó, nhân vật bị nảy lên và rơi trúng quái vật ốc sên khiến nó bị tiêu diệt.”.

Nhận xét về các nghiên cứu nêu trên:

- Tuy đều là các nghiên cứu sinh nhạc từ văn bản, nhưng MusicLM và MUGEN lại sinh nhạc ở định dạng âm thanh (text-to-audio), khác với hướng sinh nhạc ở dạng midi (text-to-midi) mà nhóm lựa chọn.
- Hầu hết các phương pháp tiếp cận chỉ thực hiện qua các bản demo nhỏ hoặc chưa phát triển thành một ứng dụng để giúp người dùng phổ thông có thể tiếp cận được.

2.3 Dữ liệu huấn luyện

2.3.1 Nguồn dữ liệu

- Hai bộ liệu huấn luyện của bài báo **MuseCoco - Generating Symbolic Music from Text**[9]. Bộ thứ nhất gồm các cặp “Câu mô tả các tính chất của bản nhạc bằng ngôn ngữ tự nhiên - Câu mô tả được mã hoá (encode) theo định dạng xác định trước”. Bộ thứ hai gồm các dòng dữ liệu chứa thông tin các đoạn nhạc được mã hoá (encode) theo định dạng được xác định trước.
- Bộ dữ liệu MusicCaps⁹ từ bài báo **MusicLM: Generating Music From Text**[1] với các cặp “Khoá xác định bản nhạc trên YouTube - Đoạn văn bản mô tả các đặc điểm của bản nhạc” là dữ liệu chính.

⁹Liên kết đến bộ dữ liệu tại Kaggle: <https://www.kaggle.com/datasets/googleai/musiccaps>, truy cập lần cuối 01/04/2024.

- Dữ liệu do nhóm thu thập từ **Hooktheory**¹⁰ - trang web chuyên cung cấp sách điện tử, bài viết, thống kê, phần mềm giáo dục về lý thuyết âm nhạc, cũng như thông tin ký âm và hoà âm của hơn 40000 bản nhạc trên thế giới. Dữ liệu nhóm thu thập có thông tin chính gồm các cặp “Thông tin ký âm và hoà âm của một bản nhạc (có thể chuyển về dạng midi) - Đoạn văn bản nhận xét về bản nhạc và một số chỉ số đánh giá bản nhạc”.

2.3.2 Kỹ thuật hỗ trợ thu thập

2.3.2.1 Tự động hóa trình duyệt

Tự động hóa trình duyệt là quá trình sử dụng các công cụ và thư viện phần mềm để tự động hóa các tác vụ trên trình duyệt web. Kỹ thuật này thường được sử dụng để kiểm thử phần mềm (testing), thu thập dữ liệu (crawling), và thực hiện các tác vụ lặp đi lặp lại trên web mà không cần sự can thiệp của con người. Công cụ tiêu biểu cho kỹ thuật này là Selenium. Chúng tôi sử dụng thư viện tương ứng Python để thu thập dữ liệu từ các nguồn web.

2.3.2.2 Lập trình đa luồng

Lập trình đa luồng (multithreading) là một kỹ thuật lập trình cho phép một ứng dụng thực hiện nhiều công việc đồng thời bằng cách sử dụng cùng lúc nhiều luồng (thread) độc lập trong một tiến trình (process) duy nhất. Trong khoá luận này, nhóm sử dụng kỹ thuật trên để tạo ra nhiều luồng Selenium chạy đồng thời nhằm tăng hiệu suất cào dữ liệu.

2.3.2.3 Xử lý theo lô

Xử lý theo lô (batch processing) là một phương pháp lập trình mà trong đó, danh sách các công việc được chia thành nhiều lô (batch), một lô được

¹⁰Liên kết đến trang web: <https://www.hooktheory.com>, truy cập lần cuối 01/04/2024.

thực thi khi lô trước đã hoàn thành, khác với việc xử lý từng công việc theo thứ tự từ đầu đến cuối. Nhóm sử dụng ý tưởng của phương pháp này để kết hợp với lập trình trình đa luồng, nhằm tạo được “các lô cào dữ liệu bằng Selenium”, mỗi công việc trong lô được chạy song nhau nhằm tăng hiệu suất cào.

2.3.3 Các định dạng dữ liệu chính

2.3.3.1 MIDI

MIDI (Musical Instrument Digital Interface) là một giao thức kỹ thuật số được phát triển vào đầu thập niên 1980 để cho phép các nhạc cụ điện tử, máy tính và các thiết bị âm thanh khác giao tiếp và kiểm soát lẫn nhau. MIDI không truyền âm thanh thực sự, mà truyền các sự kiện âm nhạc như nốt nhạc (note), tốc độ nhấn/lực nhấn phím đàn (velocity), và các thông số điều khiển khác.

Các thành phần cơ bản của MIDI gồm:

1. Giao thức truyền dữ liệu: Cho phép truyền các thông điệp MIDI với tốc độ 31.25 kbps.
2. Thông điệp MIDI: Gồm các byte trạng thái (status byte) và byte dữ liệu (data byte). Xem một số thông điệp cơ bản được nhóm tóm tắt lại từ tài liệu của MIDI Association[10] ở bảng 2.4.

Thông điệp	Mô tả
Note On/Off	Cho biết khi nào một nốt nhạc bắt đầu phát (on) hoặc dừng (off) với lực nhấn phím đàn (velocity) bao nhiêu.
Control Change	Cho biết các thông số tương ứng (âm lượng, độ ngân dài nốt, v.v) bị thay đổi giá trị.
Program Change	Cho biết chương trình nhạc cụ bị thay đổi.
Pitch Bend	Cho biết sự thay đổi giá trị của cần vặn/cần gạt pitch bend để thay đổi cao độ của nốt nhạc.

Bảng 2.4: Các thông điệp MIDI cơ bản

3. Kết nối: Gồm các cổng kết nối cho phép truyền thông điệp MIDI giữa các thiết bị:

- MIDI In: Nhận dữ liệu.
- MIDI Out: Gửi dữ liệu.
- MIDI Thru: Chuyển tiếp dữ liệu từ “MIDI In” đến các thiết bị khác.

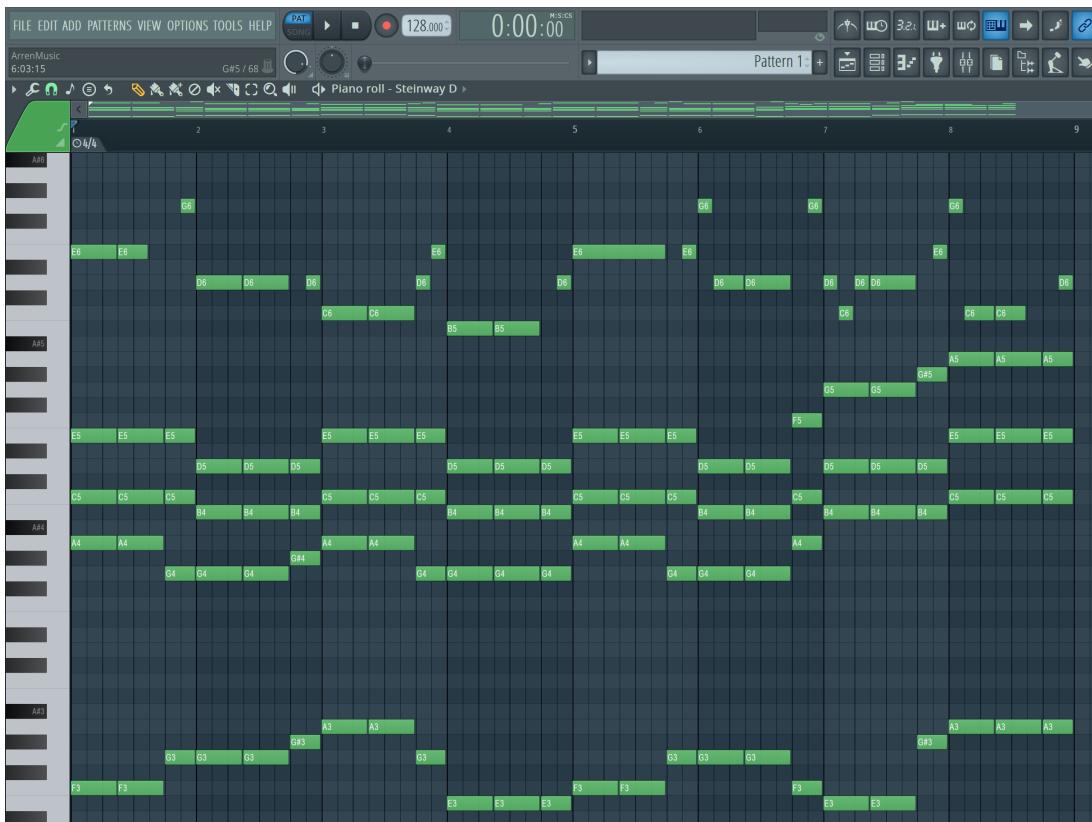
Lợi ích mà MIDI mang lại (nhất là với quá trình làm việc với âm nhạc số):

- Tính tương thích: MIDI là một tiêu chuẩn quốc tế được áp dụng rộng rãi, nên các thiết bị từ các nhà sản xuất khác nhau có thể làm việc cùng nhau dễ dàng.
- Tính linh hoạt: MIDI cho phép điều khiển và chỉnh sửa âm nhạc một cách dễ dàng, từ việc thay đổi âm sắc đến chỉnh sửa các ghi chú nhạc.
- Hiệu quả: Dữ liệu MIDI rất nhỏ gọn, nên việc truyền và lưu trữ rất hiệu quả.

Ngày nay, MIDI trở thành tiêu chuẩn trong nhiều phần mềm, thiết bị liên quan đến âm nhạc với đa dạng ứng dụng như:

- Sản xuất âm nhạc: MIDI cho phép các nhạc sĩ và nhà sản xuất kết nối và điều khiển nhiều thiết bị âm thanh, từ đàn tổng hợp âm thanh (synthesizer), máy chơi trống (drum machine), đến phần mềm âm nhạc trên máy tính.
- Biểu diễn trực tiếp: MIDI được sử dụng để đồng bộ hóa các thiết bị trên sân khấu, điều khiển ánh sáng, và các hiệu ứng đặc biệt.
- Học tập và giảng dạy âm nhạc: MIDI hỗ trợ các phần mềm giáo dục âm nhạc, cho phép học sinh thực hành và cải thiện kỹ năng chơi nhạc cụ.

Trong khoá luận này, chúng tôi sử dụng định dạng tập tin MIDI - một trong các thể hiện của giao thức trên, và các kiểu dữ liệu được biến đổi dựa trên định dạng vừa nêu. Loại tập tin này chứa dữ liệu MIDI, cho phép lưu trữ và phát lại các sự kiện âm nhạc đã được ghi lại, giúp người dùng chia sẻ, chỉnh sửa, và phát lại các bản nhạc trên nhiều thiết bị và phần mềm khác nhau một cách dễ dàng. Xem hình minh họa 2.13.



Hình 2.4: Một đoạn nhạc được lưu trữ dưới dạng MIDI khi hiển thị trên phần mềm làm nhạc

Ngoài lí do về tính tiện dụng cho đối tượng người dùng hướng đến là người làm nhạc trên máy tính, nhóm chọn định dạng tập tin MIDI mà không phải tập tin âm thanh vì năm nguyên nhân chi tiết sau:

1. Tính cấu trúc cao:

- **Ưu điểm của tập tin MIDI:** Tập tin MIDI chứa thông tin về các nốt nhạc, độ dài, lực nhấn phím đàn, và các điều khiển

khác. Điều này cung cấp cho mô hình một cấu trúc rõ ràng và có tổ chức, giúp dễ dàng phân tích, học tập, và chỉnh sửa. Ngoài ra, tập tin MIDI hoàn toàn có thể chuyển thành dữ liệu âm thanh.

- **Điểm yếu của tập tin âm thanh:** Tập tin âm thanh chỉ chứa sóng âm thanh liên tục, không có thông tin trực tiếp về cấu trúc âm nhạc như nốt nhạc hay nhịp điệu. Mô hình phải trích xuất các tính năng này từ dữ liệu sóng âm, một quá trình phức tạp và dễ bị lỗi. Hơn nữa, tập tin âm thanh không thể chỉnh sửa sâu đến từng nốt nhạc như MIDI, dẫn đến hệ quả nhạc sinh ra không thể tùy chỉnh sâu theo ý muốn của người dùng.

2. Kích thước dữ liệu nhỏ gọn:

- **Ưu điểm của tập tin MIDI:** Tập tin MIDI có kích thước rất nhỏ so với tập tin âm thanh, giúp tiết kiệm không gian lưu trữ và giảm chi phí xử lý dữ liệu.
- **Điểm yếu của tập tin âm thanh:** Tập tin âm thanh có kích thước lớn, đặc biệt là các tập tin chất lượng cao (ví dụ: wav, flac), dẫn đến việc tiêu tốn nhiều không gian lưu trữ và tài nguyên xử lý khi huấn luyện mô hình.

3. Tính đa năng:

- **Ưu điểm của tập tin MIDI:** Tập tin MIDI là một tiêu chuẩn quốc tế và có thể được sử dụng trên nhiều nền tảng và thiết bị khác nhau mà không cần chuyển đổi định dạng.
- **Điểm yếu của tập tin âm thanh:** Các định dạng âm thanh có thể khác nhau (mp3, wav, flac) và đôi khi không tương thích với một số phần mềm hoặc thiết bị, đòi hỏi phải chuyển đổi định dạng, gây mất thời gian và có thể giảm chất lượng.

4. Không bị ảnh hưởng bởi tín hiệu nhiễu và tạp âm:

- **Ưu điểm của tập tin MIDI:** Tập tin MIDI không chứa các tập âm hay nhiều âm thanh như trong tập tin âm thanh, giúp mô hình tập trung vào các yếu tố âm nhạc chính xác hơn.
- **Điểm yếu của tập tin âm thanh:** Tập tin tập tin thường chứa nhiều và tạp âm từ môi trường ghi âm, đòi hỏi các bước xử lý và làm sạch dữ liệu phức tạp trước khi có thể sử dụng cho huấn luyện mô hình.

5. Khả năng phân tích và hiểu biết cao:

- **Ưu điểm của tập tin MIDI:** Dữ liệu MIDI dễ dàng phân tích hơn vì các thông tin về nốt nhạc, nhịp điệu và hòa âm đã được xác định rõ ràng.
- **Điểm yếu của tập tin âm thanh:** Tập tin âm thanh yêu cầu các kỹ thuật phức tạp để trích xuất thông tin nhạc lý, như nhận diện nốt nhạc, tách nhạc cụ và phân tích hòa âm, các bước này thường tốn nhiều thời gian và dễ gặp sai sót.

2.3.3.2 Lớp MidiFile của thư viện miditoolkit trong Python

Vì tập tin MIDI có cấu trúc tương đối phức tạp, nhiều thư viện Python được ra đời để đơn giản hóa các thao tác trên loại tập tin này, tiêu biểu là `miditoolkit`¹¹. Trong thư viện trên, lớp `MidiFile` là cấu trúc dữ liệu trọng tâm. Lớp này cho phép lập trình viên đọc, sửa đổi và ghi lại tập tin MIDI một cách dễ dàng, linh hoạt. Trong phạm vi khoá luận, chúng tôi tận dụng bốn thuộc tính sau của lớp `MidiFile` để làm việc:

1. `ticks_per_beat (int)`: Số tick (đơn vị thời gian trong tập tin MIDI) tương ứng với mỗi nhịp (đơn vị thời gian trong lý thuyết âm nhạc).

¹¹Liên kết đến thư viện `miditoolkit` trong Python:
<https://pypi.org/project/miditoolkit/0.1.17/>, truy cập lần cuối 03/06/2024.

2. `tempo_changes` (`list[TempoChange]`): Danh sách các lần thay đổi tốc độ bản nhạc (`TempoChange`) theo thời gian trong tập tin MIDI. Trong đó `TempoChange` gồm các thuộc tính:

- `tempo` (`int`): Tốc độ bản nhạc.
- `time` (`int`): Thời điểm thay đổi tốc độ bản nhạc; tính theo đơn vị tick.

3. `time_signature_changes` (`list[TimeSignature]`): Danh sách các lần thay đổi nhịp (`TimeSignature`) của bản nhạc theo thời gian trong tập tin MIDI. Trong đó `TimeSignature` gồm các thuộc tính:

- `numerator` (`int`): Số đơn vị nhịp trong một ô nhịp.
- `denominator` (`int`): Độ dài (trường độ) của một đơn vị nhịp.
- `time` (`int`): Thời điểm thay đổi nhịp của bản nhạc; tính theo đơn vị tick.

4. `instruments` (`list[Instrument]`): Danh sách các đối tượng `Instrument`, mỗi đối tượng này đại diện cho một nhạc cụ trong tập tin MIDI, chứa các nốt nhạc, và gồm các thuộc tính chính sau:

- `program` (`int`): Số nguyên xác định số chương trình của nhạc cụ theo chuẩn **General MIDI**¹², có giá trị từ 0 đến 127 (tương ứng với các nhạc cụ từ số chương trình 1 đến 128 trong chuẩn được nêu).
- `is_drum` (`bool`): Cho biết nhạc cụ có thuộc nhóm các loại trống hay không.
- `name` (`str`): Tên của nhạc cụ.

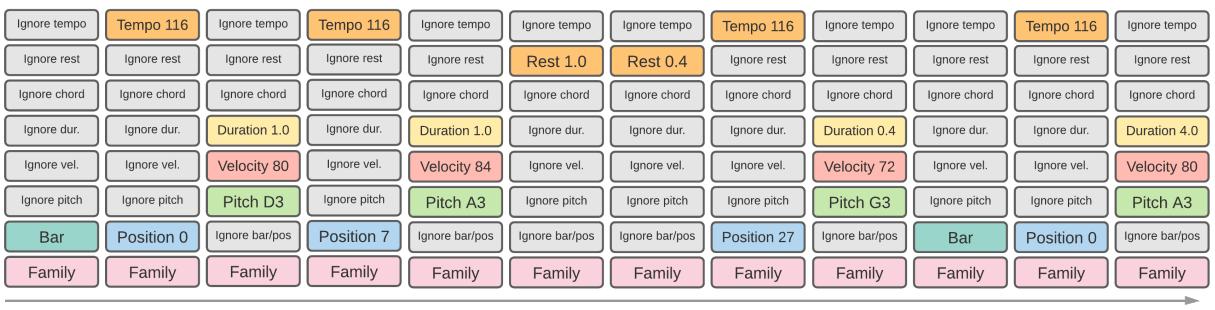
¹²Chi tiết danh sách nhạc cụ ở tài liệu do **MIDI Association** cung cấp: **RP-003_General_MIDI_System_Level_1_Specification_96-1-4_0.1.pdf**, trang 25, mục **General MIDI Sound Set (Table 2)**, tại liên kết <https://midi.org/general-midi-level-1>, truy cập lần cuối 03/06/2024.

- **notes** (list[Note]): Danh sách các đối tượng Note. Mỗi đối tượng này đại diện cho một nốt nhạc, gồm bốn thuộc tính:
 - + **pitch** (int): Cao độ của nốt nhạc theo chuẩn MIDI, có giá trị từ 0 đến 127, trong đó $\text{pitch} = 60$ tương ứng với nốt C4 (middle C) trên đàn piano¹³.
 - + **velocity** (int): Tốc độ nhấn/lực nhấn phím đàn, có giá trị từ 0 đến 127.
 - + **start** (int): Thời gian bắt đầu của nốt nhạc; tính theo đơn vị tick.
 - + **end** (int): Thời gian kết thúc của nốt nhạc; tính theo đơn vị tick.

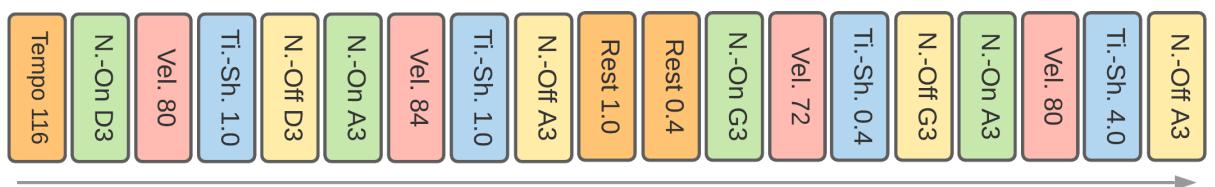
2.3.3.3 REMI

Để token hoá âm nhạc, ta có nhiều phương pháp như CPWord[7] (hình 2.5), MIDI-Like[13] (hình 2.6), MMM[3] (hình 2.10), REMI[8] (hình 2.7), Structured MIDI Encoding[5] (hình 2.8), và TSD[4] (hình 2.9). Trong đó, REMI - REvamped MIDI-derived events, là dạng biểu diễn sự kiện MIDI được bài báo “Pop Music Transformer: Beat-based Modeling and Generation of Expressive Pop Piano Compositions”[8] đề xuất vào năm 2020 để chuyển đổi các bản nhạc MIDI thành những token rời rạc tương tự văn bản.

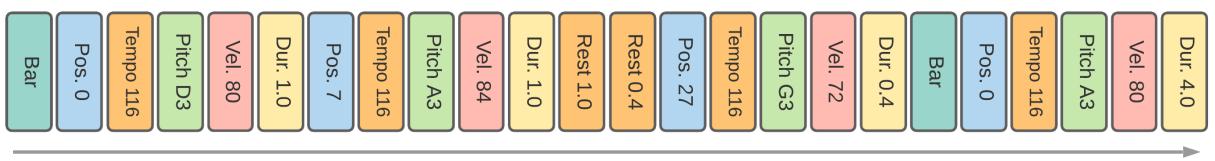
¹³Một số giá trị tương ứng giữa pitch và tên nốt nhạc ở tài liệu do **MIDI Association** cung cấp: **General_MIDI_Level_2_07-2-6_1.2a.pdf**, trang 32, mục **8. Appendix B: GM 2 Percussion Sound Set**, cột **NOTE#**, tại liên kết <https://midi.org/general-midi-level-2-2>, truy cập lần cuối 03/06/2024.



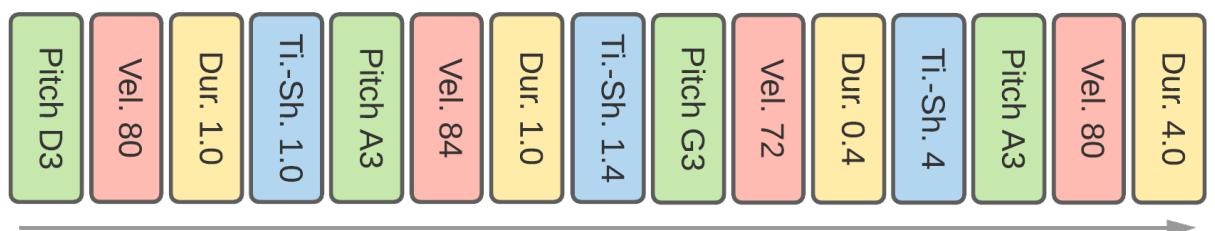
Hình 2.5: Minh họa dạng hình ảnh cho phương pháp token hoá âm nhạc CPWord[11]



Hình 2.6: Minh họa dạng hình ảnh cho phương pháp token hoá âm nhạc MIDI-Like[11]



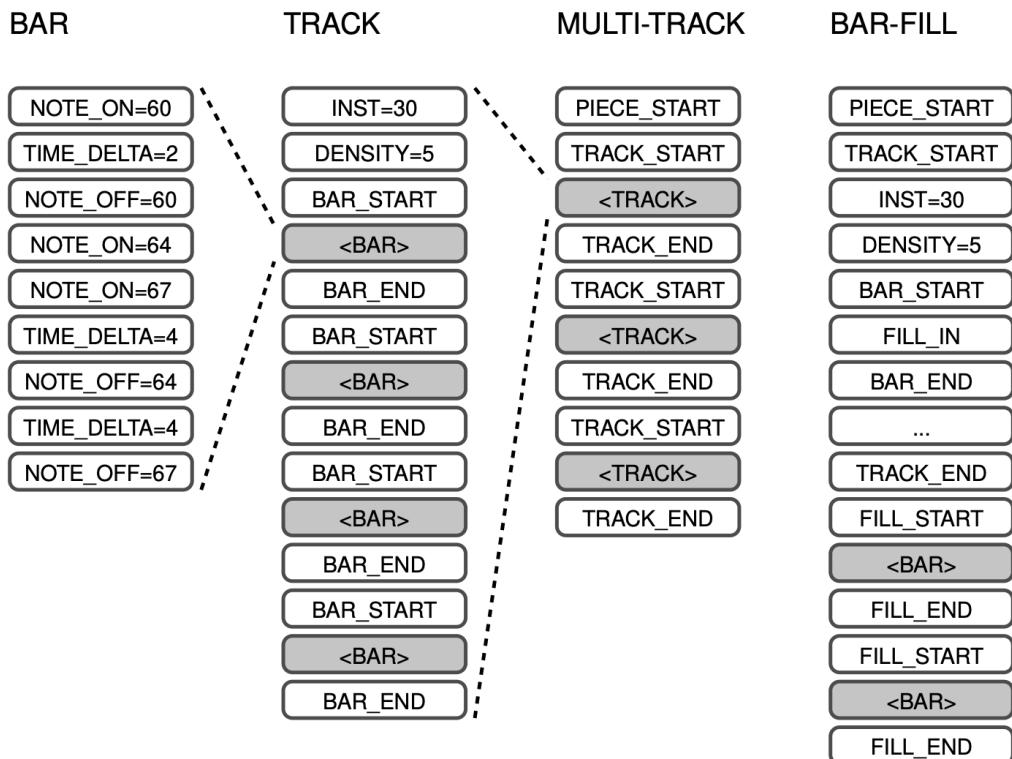
Hình 2.7: Minh họa dạng hình ảnh cho phương pháp token hoá âm nhạc REMI[11]



Hình 2.8: Minh họa dạng hình ảnh cho phương pháp token hoá âm nhạc Structured MIDI Encoding[11]



Hình 2.9: Minh họa dạng hình ảnh cho phương pháp token hoá âm nhạc TSD[11]



Hình 2.10: Minh họa dạng hình ảnh cho phương pháp token hoá âm nhạc MMM[3]

So với hầu hết các kiểu biểu diễn âm nhạc dựa trên MIDI từng áp dụng trong những mô hình sinh nhạc bằng Transformer trước đó, REMI cung cấp cho các mô hình dạng chuỗi (sequence model) một ngữ cảnh nhịp điệu (metrical context) bằng cách thêm thông tin vạch nhịp giữa các ô nhịp của bản nhạc, thay vì chỉ có thông tin thời gian nốt nhạc bắt đầu phát và ngừng phát (note on/note off), để phân chia rõ ràng các ô nhịp, giúp

mô-hình mô-hình-hoá các mẫu nhịp điệu (rhythmic pattern - một chuỗi nhịp điệu được lặp lại theo một thứ tự cụ thể) tốt hơn. Từ đó giúp mô hình cho ra các bản nhạc ổn định về mặt nhịp điệu hơn so với các mô hình học thông tin nốt nhạc bắt đầu phát và dừng phát chỉ dựa vào thời gian.

```

note_items, tempo_items = utils.read_items('./data/evaluation/000.midi')

print(*note_items, sep='\n')

Item(name=Note, start=956, end=1530, velocity=55, pitch=59)
Item(name=Note, start=1420, end=1998, velocity=57, pitch=60)
Item(name=Note, start=1885, end=3960, velocity=71, pitch=62)
Item(name=Note, start=1921, end=2519, velocity=58, pitch=43)
Item(name=Note, start=2410, end=4109, velocity=62, pitch=50)
Item(name=Note, start=2886, end=5285, velocity=69, pitch=59)
Item(name=Note, start=3372, end=3848, velocity=64, pitch=67)
Item(name=Note, start=3848, end=5910, velocity=71, pitch=67)
Item(name=Note, start=5285, end=5872, velocity=68, pitch=59)
Item(name=Note, start=5761, end=6904, velocity=66, pitch=43)
Item(name=Note, start=5761, end=6723, velocity=72, pitch=60)
Item(name=Note, start=6247, end=6795, velocity=57, pitch=52)
Item(name=Note, start=6723, end=8710, velocity=66, pitch=60)
Item(name=Note, start=7198, end=7673, velocity=68, pitch=69)

print(*tempo_items[:10], sep='\n')

Item(name=Tempo, start=0, end=None, velocity=None, pitch=120)
Item(name=Tempo, start=480, end=None, velocity=None, pitch=23)
Item(name=Tempo, start=960, end=None, velocity=None, pitch=146)
Item(name=Tempo, start=1440, end=None, velocity=None, pitch=139)
Item(name=Tempo, start=1920, end=None, velocity=None, pitch=146)
Item(name=Tempo, start=2400, end=None, velocity=None, pitch=142)
Item(name=Tempo, start=2880, end=None, velocity=None, pitch=146)
Item(name=Tempo, start=3360, end=None, velocity=None, pitch=142)
Item(name=Tempo, start=3840, end=None, velocity=None, pitch=146)
Item(name=Tempo, start=4320, end=None, velocity=None, pitch=142)

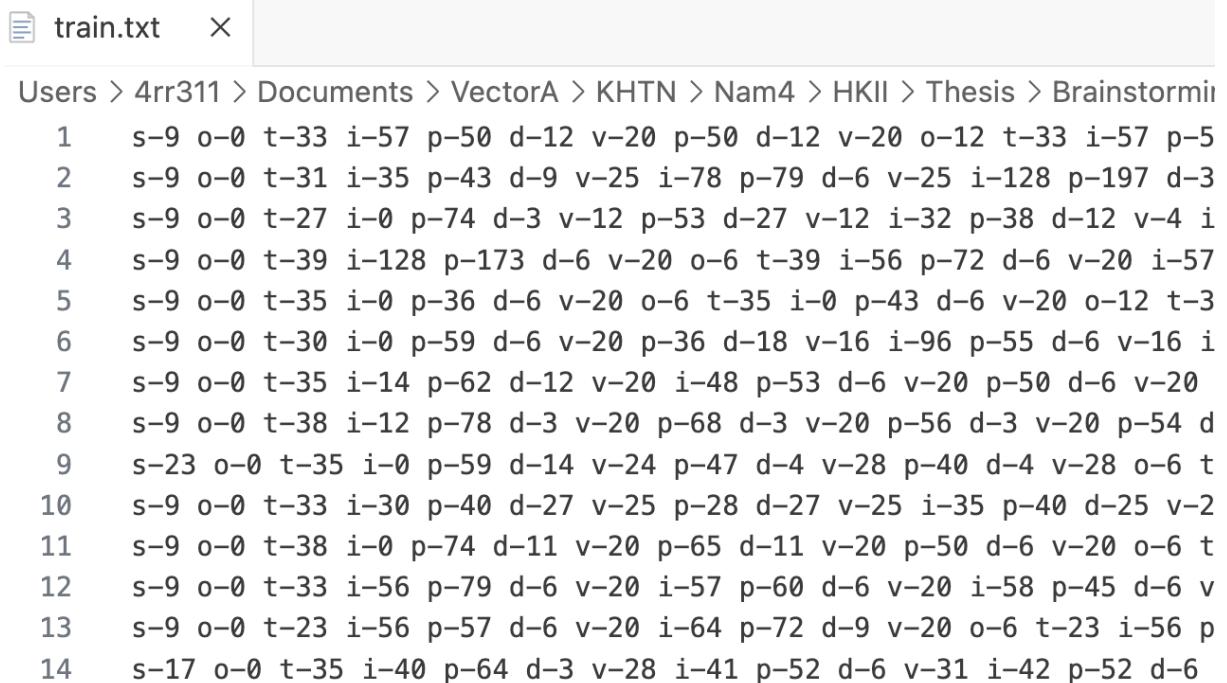
```

Hình 2.11: Các giá trị nốt và tốc độ bản nhạc của một đoạn dữ liệu âm nhạc khi chuyển về dạng REMI trong Python[12]

2.3.3.4 Dữ liệu âm nhạc dạng văn bản của MuseCoco

Bài báo **MuseCoco - Generating Symbolic Music from Text**[9] dựa vào định dạng REMI (đã nêu ở mục 2.3.3.3) để tạo ra âm nhạc dưới dạng token hoá và có thể biểu diễn ở dạng văn bản (xem ví dụ ở hình 2.12).

Nhóm tái sử dụng cấu trúc này để xây dựng bộ dữ liệu âm nhạc từ các bộ dữ mà nhóm có.



The screenshot shows a text editor window with a tab labeled "train.txt". The file path is shown at the top: "Users > 4rr311 > Documents > VectorA > KHTN > Nam4 > HKII > Thesis > Brainstorming". The content of the file is a sequence of 14 lines, each representing a musical event. Each line is composed of several tokens separated by spaces, representing parameters like start time, duration, pitch, and velocity. The tokens are lowercase letters and numbers, such as "s-9 o-0 t-33 i-57 p-50 d-12 v-20".

```
1 s-9 o-0 t-33 i-57 p-50 d-12 v-20 p-50 d-12 v-20 o-12 t-33 i-57 p-5
2 s-9 o-0 t-31 i-35 p-43 d-9 v-25 i-78 p-79 d-6 v-25 i-128 p-197 d-3
3 s-9 o-0 t-27 i-0 p-74 d-3 v-12 p-53 d-27 v-12 i-32 p-38 d-12 v-4 i
4 s-9 o-0 t-39 i-128 p-173 d-6 v-20 o-6 t-39 i-56 p-72 d-6 v-20 i-57
5 s-9 o-0 t-35 i-0 p-36 d-6 v-20 o-6 t-35 i-0 p-43 d-6 v-20 o-12 t-3
6 s-9 o-0 t-30 i-0 p-59 d-6 v-20 p-36 d-18 v-16 i-96 p-55 d-6 v-16 i
7 s-9 o-0 t-35 i-14 p-62 d-12 v-20 i-48 p-53 d-6 v-20 p-50 d-6 v-20
8 s-9 o-0 t-38 i-12 p-78 d-3 v-20 p-68 d-3 v-20 p-56 d-3 v-20 p-54 d
9 s-23 o-0 t-35 i-0 p-59 d-14 v-24 p-47 d-4 v-28 p-40 d-4 v-28 o-6 t
10 s-9 o-0 t-33 i-30 p-40 d-27 v-25 p-28 d-27 v-25 i-35 p-40 d-25 v-2
11 s-9 o-0 t-38 i-0 p-74 d-11 v-20 p-65 d-11 v-20 p-50 d-6 v-20 o-6 t
12 s-9 o-0 t-33 i-56 p-79 d-6 v-20 i-57 p-60 d-6 v-20 i-58 p-45 d-6 v
13 s-9 o-0 t-23 i-56 p-57 d-6 v-20 i-64 p-72 d-9 v-20 o-6 t-23 i-56 p
14 s-17 o-0 t-35 i-40 p-64 d-3 v-28 i-41 p-52 d-6 v-31 i-42 p-52 d-6
```

Hình 2.12: Một đoạn dữ liệu âm nhạc dạng văn bản của MuseCoco

Mỗi cặp “`thuộc_tính - giá_trị`” trong định dạng trên phân cách nhau bởi một khoảng trắng; `thuộc_tính` và `giá_trị` phân cách nhau bằng dấu gạch nối; `giá_trị` được chuẩn hoá (normalize) từ các giá trị MIDI tương ứng trong lớp `MidiFile` (đã nêu ở mục 2.3.3.2) bằng bộ chuẩn hoá của bài báo MuseCoco[9]. Xem đặc tả dữ liệu ở bảng 2.5.

Thuộc tính	Tên đầy đủ	Mô tả	Thuộc tính liền sau
s	Time Signature	Nhịp của bản nhạc	b, o.
o	Position	Thời điểm xuất hiện của sự kiện	t.
t	Tempo	Tốc độ bản nhạc	i.
i	Instrument	Nhạc cụ	p.
p	Pitch	Cao độ nốt nhạc	d.
d	Duration	Độ dài nốt nhạc	v.
v	Velocity	Tốc độ nhấn/Lực nhấn phím đàn	i, b, p, o.
b	Bar	Vạch nhịp	s.

Bảng 2.5: Đặc tả dữ liệu âm nhạc dạng văn bản của MuseCoco

2.4 Mô hình ngôn ngữ lớn

Trong khoá luận này, nhóm chúng tôi đã áp dụng và thử nghiệm trên các mô hình ngôn ngữ lớn, bao gồm Transformer gốc, BERT và GPT2. Mục tiêu của chúng tôi là khám phá và đánh giá khả năng và hiệu quả của từng mô hình trong việc xử lý và hiểu ngôn ngữ tự nhiên, cũng như khả năng ứng dụng của chúng trong các tác vụ NLP cụ thể. Transformer, một kiến trúc mạng nơ-ron được sử dụng rộng rãi, là nền tảng cho cả BERT và GPT2. BERT được thiết kế để hiểu ngữ cảnh hai chiều của một từ trong văn bản, trong khi GPT2 được tối ưu hóa cho các tác vụ sinh văn bản. Bằng cách triển khai và thử nghiệm các mô hình này trên một loạt các tập dữ liệu và trong các tình huống khác nhau, chúng tôi mong muốn đưa ra những hiểu biết sâu sắc về cách các tiến bộ trong công nghệ ngôn ngữ máy tính có thể được tận dụng để cải thiện và tối ưu hóa chúng. Chúng ta sẽ tìm hiểu kỹ hơn ở phần bên dưới.

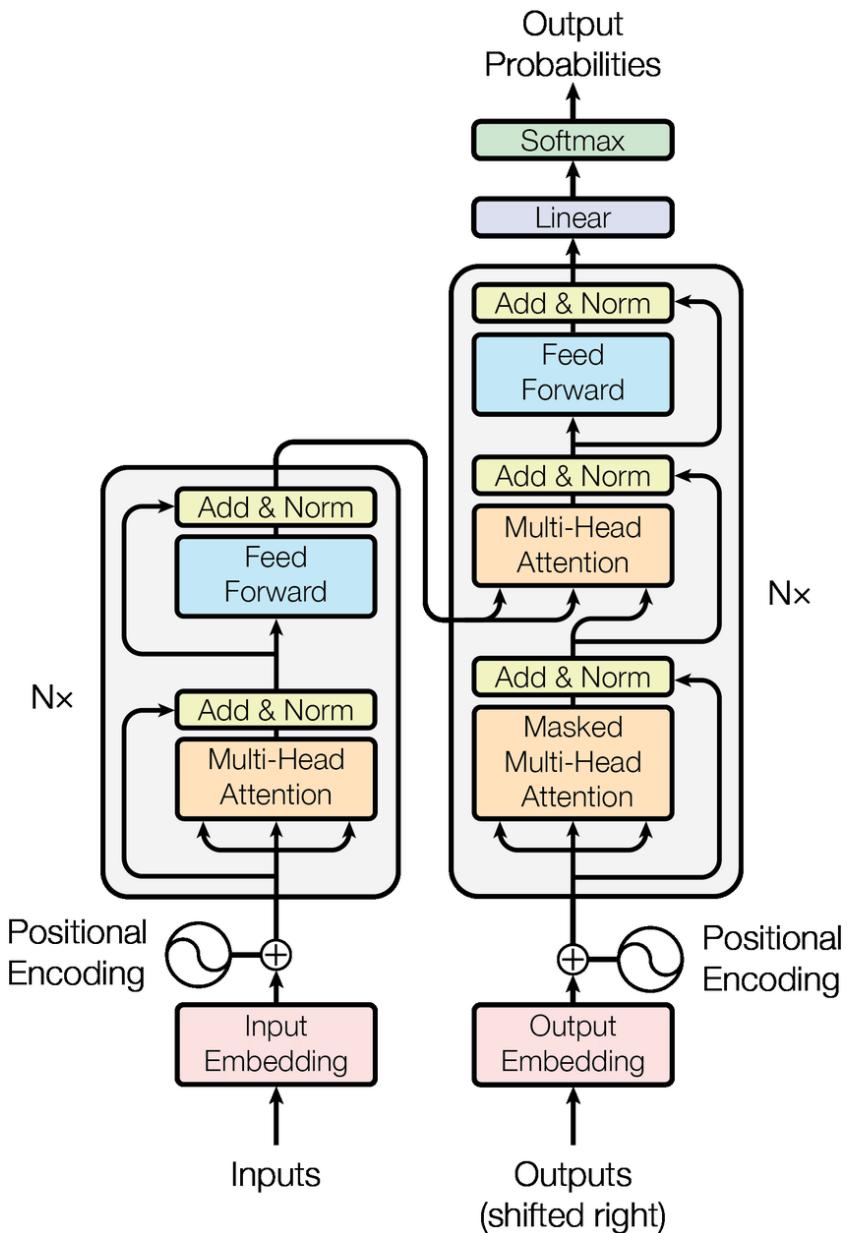
2.4.1 Transformer

Transformer là một kiến trúc mạng nơ-ron được giới thiệu vào năm 2017 bởi đội nghiên cứu của Google. Kiến trúc này đã tạo ra một bước

ngoặt trong lĩnh vực xử lý ngôn ngữ tự nhiên, đạt được những tiến bộ đáng kể trong nhiều tác vụ khác nhau. Không giống như các mô hình tuần tự truyền thống như RNN và LSTM, vốn xử lý các token tuần tự khiến mô hình huấn luyện chậm và hạn chế trong việc biểu diễn sự phụ thuộc giữa các từ xa nhau trong một câu, Transformer xử lý tất cả các token song song, triệt để giải quyết vấn đề này. Transformer cũng là nền tảng cho các mô hình hiện đại như GPT và BERT.

Kiến trúc Transformer có các đặc điểm chính như sau:

- Đầu tiên, Transformer không sử dụng các kiến trúc hồi quy như RNN hay LSTM mà Transformer sử dụng cơ chế self-attention. Đây là thành phần cốt lõi giúp Transformer tạo ra được sự khác biệt. Nó có phép biểu diễn mối quan hệ giữa một với với các từ còn lại. Có nhiều cơ chế attention khác nhau như self-attention, multi-head attention.
- Bộ mã hóa (Encoder): Bộ mã hóa bao gồm nhiều lớp, mỗi lớp gồm hai thành phần chính là self-attention và mạng nơ-ron truyền thẳng (Feed Forward Network). Ngoài ra, còn có các lớp kết nối tắt (Residual Connection) và chuẩn hóa theo lớp (Layer Normalization). Bộ mã hóa được thiết kế để xử lý đầu vào và tạo ra biểu diễn thông tin.
- Bộ giải mã (Decoder): Bộ giải mã bao gồm nhiều lớp tương tự như bộ mã hóa, nhưng có thêm một lớp multi-head attention để chú ý đến đầu ra của bộ mã hóa (Encoder). Bộ giải mã chịu trách nhiệm tạo ra đầu ra mong muốn.
- Mã hóa vị trí: Vì Transformer không có khái niệm về thứ tự của các từ, nên cần một cách để biểu diễn vị trí của mỗi từ trong câu. Điều này được thực hiện bằng cách thêm một vectơ mã hóa vị trí vào biểu diễn của mỗi từ.



Hình 2.13: Kiến trúc Transformer

Có hai nguyên nhân chính khiến Transformer ưu việt hơn so với các kiến trúc trước đó. Thứ nhất, Transformer đã cho ra kiến trúc để có thể tính toán song song cho tất cả các từ trong câu, giúp tăng tốc độ huấn luyện và suy luận. Thứ hai, Cơ chế attention cho phép Transformer nắm bắt được mối quan hệ giữa các từ và hiểu được ngữ cảnh của từng từ, từ đó hiểu được bối cảnh trong câu.

Transformer đã đạt được hiệu suất vượt trội so với các mô hình trước

đó trong nhiều tác vụ xử lý ngôn ngữ tự nhiên. Transformer ra đời giúp giải quyết được nhiều tác vụ như dịch máy, tóm tắt văn bản, v.v. Trong khóa luận này chúng tôi cũng đã áp dụng những kinh nghiệm của Transformer để xây dựng mô hình sinh nhạc.

2.4.2 BERT

BERT (Bidirectional Encoder Representations from Transformers) là một mô hình được Google phát triển và ra mắt vào năm 2018, đem lại sự đột phá trong lĩnh vực AI, đặc biệt là xử lý ngôn ngữ tự nhiên (NLP). Về sau, BERT còn được ứng dụng trong các lĩnh vực liên quan đến xử lý hình ảnh và âm thanh, mở rộng tầm ảnh hưởng của nó trong thế giới AI.

Kiến trúc của BERT dựa trên mô hình Transformer, sử dụng hoàn toàn các bộ mã hóa (encoder) của Transformer mà không bao gồm phần giải mã (decoder). Kiến trúc này bao gồm nhiều lớp encoder, mỗi lớp gồm các thành phần chính sau:

- Multihead Self Attention: Cơ chế này cho phép mô hình chú ý đến các từ khác trong câu khi đang xử lý một từ cụ thể, giúp hiểu rõ bối cảnh và ngữ cảnh của từ.
- Feed Forward Neural Network: Kết quả từ lớp self-attention sẽ được đi qua mạng FFN để tiếp tục xử lý. Mỗi lớp encoder có một mạng FFN riêng biệt.
- Layer Normalization và Residual Connections: Những cơ chế này giúp mô hình ổn định trong quá trình huấn luyện, tăng khả năng học tập và giảm thiểu vấn đề biến mất gradient.
- Pooler: Phần này nằm ở cuối mô hình và được sử dụng để tổng hợp thông tin từ các token đầu ra của encoder, qua đó biểu diễn toàn bộ câu dựa trên các biểu diễn này.

BERT có khả năng thu thập và hiểu ngữ cảnh một cách đa chiều, không chỉ dựa vào ngữ cảnh trước mà còn ngữ cảnh sau của một từ, một cải tiến đáng kể so với các mô hình NLP trước đó chỉ học một chiều.

Trong quá trình huấn luyện, BERT tập trung vào hai mục tiêu chính: Masked Language Model - Phương pháp này dự đoán các token bị che dựa trên ngữ cảnh xung quanh, giúp mô hình hiểu được mối liên kết và ảnh hưởng của các từ lén nhau và Next Sentence Prediction - Kỹ thuật này dự đoán mối liên hệ giữa hai câu, giúp cải thiện khả năng hiểu ngữ cảnh toàn câu và mối liên kết giữa các câu trong văn bản.

BERT đã đạt được thành tựu vượt trội so với các mô hình trước đó trong nhiều tác vụ NLP, từ phân tích cảm xúc đến trả lời câu hỏi, và tiếp tục là một nền tảng quan trọng trong nghiên cứu và ứng dụng AI hiện nay.

2.4.3 GPT2

Một trong những công nghệ đột phá trong lĩnh vực xử lý ngôn ngữ tự nhiên, đạt được những thành tựu nổi tiếng và gây tiếng vang toàn cầu đó là GPT3.5, GPT4 hay GPTo. Nền móng của những mô hình trên xuất phát từ GPT2 do OpenAI phát triển vào năm 2019. Tại thời điểm ra đời, GPT2 mang tính cải tiến mạnh mẽ so với các mô hình trước đó, và mang tiềm năng để đạt được các thành tựu như ngày hôm nay. Được mệnh danh là “ma thuật” do khả năng sinh văn bản tự nhiên đầy ấn tượng, nổi tiếng với khả năng học sâu và phân tích ngôn ngữ một cách toàn diện.

GPT2 dựa trên kiến trúc của Transformer. Có nhiều phiên bản với số lượng tham số khác nhau từ 124 triệu đến 1.5 tỷ tham số, nó có quy mô vượt trội so với các mô hình trước đó, với lượng tham số lớn như vậy thì mô hình mở ra tiềm năng to lớn và khả năng ứng dụng trên các tác vụ đặc biệt. Từ đó mô hình có thể áp dụng cho những nhiệm vụ cụ thể, đặc biệt là sinh ra văn bản tự nhiên với độ chính xác cao.

GPT2 được huấn luyện trên một tập dữ liệu khổng lồ từ internet, bao gồm nhiều loại văn bản khác nhau. Quá trình pre-training giúp mô hình

học được cấu trúc ngữ pháp, ngữ nghĩa và các mối quan hệ ngữ cảnh. Sau đó, mô hình có thể được fine-tune trên các tập dữ liệu nhỏ hơn và chuyên biệt cho các nhiệm vụ cụ thể như dịch máy, tóm tắt văn bản hoặc trả lời câu hỏi, hay sinh nhạc.

GPT2 đòi hỏi rất nhiều tài nguyên tính toán để huấn luyện và triển khai, điều này có thể là một rào cản đối với một số tổ chức và cá nhân.

GPT2 đã mở đường cho sự ra đời của các mô hình ngôn ngữ lớn hơn và mạnh mẽ hơn như GPT3 và GPT4. Những mô hình này tiếp tục cải tiến về quy mô và hiệu suất, mang lại nhiều ứng dụng và khả năng mới cho lĩnh vực xử lý ngôn ngữ tự nhiên.

2.5 Các loại độ đo

Chuyển phần này xuống thực nghiệm luôn chứ mất công khi xem kết quả thực nghiệm lại phải lên đây đọc lại. Mình có thể mô tả tất cả các độ đo và những độ đo mình sử

2.5.1 Độ đo hiệu suất mô hình trích xuất đặc trưng

Vì đây là một mô hình phân loại đa nhãn, nhóm tập trung vào các chỉ số đánh giá hiệu quả đặc thù cho loại mô hình này, khác với phân loại đơn nhãn. Cụ thể:

- Micro: Tính toán các số liệu accuracy trên toàn bộ các dự đoán, coi tất cả nhãn như một tập hợp lớn. Ưu tiên khi quan tâm đến hiệu suất tổng thể.
- Macro: Tính toán các số liệu cho từng nhãn riêng lẻ, rồi lấy trung bình. Ưu tiên khi muốn đảm bảo mỗi nhãn đều được đánh giá công bằng, không bị các nhãn lớn lấn át.

2.5.2 Độ đo hiệu suất mô hình sinh nhạc

2.5.2.1 Độ đo ASA [cite?](#)

Dựa trên bài báo MuseCoCo đề xuất độ đo ASA (Average Sample-wise Accuracy) gọi là Độ Chính Xác Trung Bình Mẫu được tính bằng cách xác

định tỷ lệ các thuộc tính dự đoán đúng trong mỗi mẫu, sau đó tính toán độ chính xác dự đoán trung bình trên toàn bộ tập kiểm tra.

2.5.2.2 Accuracy

Accuracy là tỷ lệ giữa số lượng dự đoán đúng và tổng số dự đoán. Công thức:

$$accuracy = \frac{\text{số dự đoán đúng}}{\text{tổng số dự đoán}}$$

Trong bối cảnh sinh nhạc, accuracy là một độ đo quan trọng để đánh giá mức độ hiệu quả của mô hình trong việc dự đoán chính xác các thuộc tính về âm nhạc.

2.5.3 Độ đo nhạc tính cite?

2.5.3.1 Số lần xuất hiện trung bình của mỗi cao độ trong một đoạn nhạc

Đây là độ đo nhằm hỗ trợ xác định sự phân phối cao độ (pitch) của 128 cao độ mà tập tin MIDI hỗ trợ (đã nêu trong phần 2.3.3.2 về lớp MidiFile của thư viện miditoolkit, ở lớp instruments, thuộc tính notes). Công thức:

$$average_pitch_count = \frac{\text{số lần xuất hiện của cao độ đang xét}}{\text{tổng số nốt trong tập tin MIDI}}$$

2.5.3.2 Số lần xuất hiện trung bình của mỗi cao độ cơ bản trong một đoạn nhạc

Đây là độ đo nhằm hỗ trợ xác định sự phân phối cao độ (pitch) của 12 nốt nhạc cơ bản (đã nêu ở mục 2.1.1.4, hình 2.2). Công thức:

$$average_pitch_class_count = \frac{\text{số lần xuất hiện}}{\text{tổng số nốt trong tập tin MIDI}},$$

với số lần xuất hiện được tính bằng số các nốt có cao độ đồng dư với cao độ đang xét theo modulo 12.

Chương 3

Phương pháp đề xuất

Chương này mô tả chi tiết phương pháp thực hiện, các chiến lược để xuất trong khoá luận.

3.1 Xác định chi tiết vấn đề

Đây là bài toán biến đổi ngôn từ tự nhiên thành âm nhạc, nghĩa là từ một câu đầu vào cụ thể, mô hình sẽ tạo ra một bản nhạc tương ứng. Thách thức của bài toán này là sản phẩm âm nhạc phải không chỉ phù hợp với nội dung và cảm xúc của câu văn đầu vào mà còn phải có tính thẩm mỹ, sáng tạo và hấp dẫn về mặt âm nhạc. Để đạt được điều này, việc lựa chọn kiến trúc mô hình phù hợp và sở hữu nguồn dữ liệu chất lượng cao là rất quan trọng. Phần bên dưới sẽ đi sâu về phần chi tiết về kiến trúc mô hình, phương pháp huấn luyện cũng như dữ liệu.

3.2 Tổng quan về phương pháp

3.2.1 Kiến trúc tổng quát

được mô tả trong Hình 3.1.

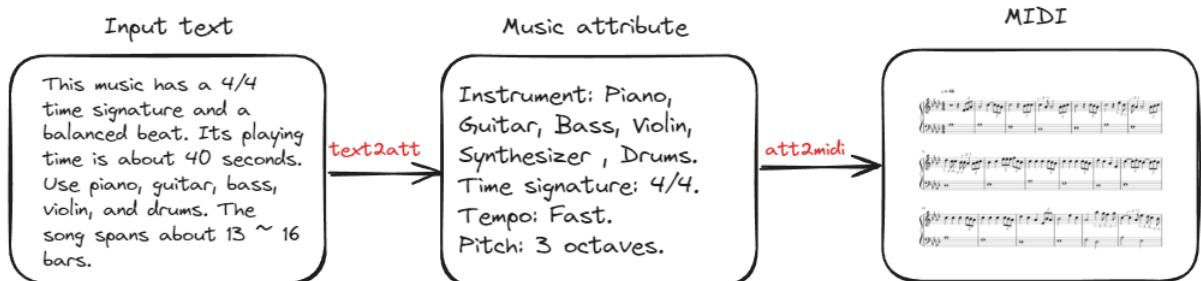
Kiến trúc nhóm sử dụng gồm hai mô hình tương ứng với hai **giai đoạn**:

Giai đoạn 1: Trích xuất các thuộc tính âm nhạc. Giai đoạn này tập trung vào việc xác định các đặc trưng âm nhạc dựa trên sở thích và yêu

cầu của nhạc sĩ, nhà sản xuất âm nhạc, hoặc người yêu thích âm nhạc. Chúng tôi thu thập thông tin về các yêu cầu như loại nhạc cụ được yêu thích (ví dụ, piano hoặc guitar), tốc độ của bản nhạc (nhanh hay chậm), và cao độ của giai điệu. Từ những thông tin này, chúng tôi tổng hợp và gán nhãn cho dữ liệu âm nhạc để phản ánh các thuộc tính này.

Giai đoạn 2: Sinh nhạc từ các nhãn đã trích xuất. Các thuộc tính âm nhạc đã được gán nhãn trong giai đoạn đầu được sử dụng để tạo ra câu dẫn (prompt) cho mô hình ngôn ngữ. Mô hình này sau đó sử dụng câu dẫn để sinh ra bản nhạc phù hợp với các đặc trưng đã được nêu. Quá trình này không chỉ nhắm đảm bảo bản nhạc tạo ra phải chính xác về mặt ngữ cảnh mà còn cần phải sáng tạo và hấp dẫn về mặt âm nhạc, phù hợp với yêu cầu thẩm mỹ.

Mỗi giai đoạn đều có vai trò quan trọng trong việc đạt được mục tiêu cuối cùng của dự án: tạo ra bản nhạc không chỉ thỏa mãn yêu cầu kỹ thuật mà còn đáp ứng được yếu tố nghệ thuật và cảm xúc.



Hình 3.1: Kiến trúc tổng quát của mô hình

Font chữ trong figure hơi khó đọc

3.2.2 Lí do lựa chọn

Trong nghiên cứu này, chúng tôi quyết định sử dụng kiến trúc hai lớp để giải quyết bài toán sinh nhạc từ ngôn ngữ tự nhiên. Câu hỏi đặt ra là tại sao chúng tôi lại chọn kiến trúc này thay vì các kiến trúc khác thường được sử dụng trong các mô hình ngôn ngữ.

Thông thường, các mô hình ngôn ngữ lớn thường sử dụng một trong

hai loại kiến trúc: mô hình decoder như GPT để sinh văn bản, hoặc mô hình encoder như BERT để tóm tắt và trích xuất thông tin từ văn bản. Tuy nhiên, một nhược điểm lớn của các mô hình này là lượng tài nguyên tính toán và dữ liệu cần thiết để huấn luyện lại từ đầu là cực kỳ lớn. Ví dụ, việc huấn luyện từ đầu một mô hình ngôn ngữ lớn không chỉ tốn kém chi phí mà còn đòi hỏi lượng tài nguyên khổng lồ, điều này là không khả thi trong nhiều trường hợp cụ thể và đôi khi không mang lại hiệu quả cao cho các tác vụ chuyên biệt.

Trong lĩnh vực âm nhạc, các thuộc tính phổ biến thường chỉ tập trung vào một lượng nhất định, và chúng tôi cũng chỉ tập trung vào những thuộc tính này. Các thông tin của một bản nhạc thường chỉ bao gồm một số lượng từ vựng nhất định, giúp giảm bớt độ phức tạp khi xử lý và huấn luyện mô hình.

Trong khi đó, các thuộc tính âm nhạc mà chúng tôi muốn mô hình hóa lại tập trung vào một lượng từ vựng nhất định và không cần đến toàn bộ bộ từ vựng rộng lớn như trong các mô hình GPT3.5 hay Llama2. Việc này cho phép chúng tôi sử dụng các mô hình nhỏ hơn và hiệu quả hơn như GPT-2 và Transformer casual language model, đồng thời tận dụng bộ từ điển REMI đã được tối ưu hóa cho âm nhạc, giảm số lượng từ vựng từ hơn 50,000 xuống còn 1,253 từ.

Ngoài ra, việc áp dụng kỹ thuật LoRA (Low-Rank Adaptation) cho phép chúng tôi tinh chỉnh mô hình với chỉ một phần nhỏ các tham số của nó, giảm bớt nhu cầu về tài nguyên trong khi vẫn duy trì hiệu suất cao. Điều này cũng hỗ trợ cho việc huấn luyện và triển khai mô hình trở nên linh hoạt và hiệu quả hơn.

Cuối cùng, sự độc lập của hai mô hình trong kiến trúc hai lớp này cũng làm cho việc xử lý dữ liệu trở nên đơn giản hơn, vì dữ liệu có thể được quản lý một cách hiệu quả hơn, tập trung vào các đặc tính kỹ thuật thay vì phải đối mặt với sự phức tạp của ngôn ngữ tự nhiên và các cách mô tả âm nhạc khác nhau. Việc này giúp cải thiện độ chính xác và giảm thiểu nhiễu trong quá trình huấn luyện và sinh sản phẩm cuối cùng. Từ bài toán

sinh nhạc bằng ngôn ngữ tự nhiên ban đầu, chúng tôi đã chuyển đổi được thành hai bài toán nhỏ hơn.

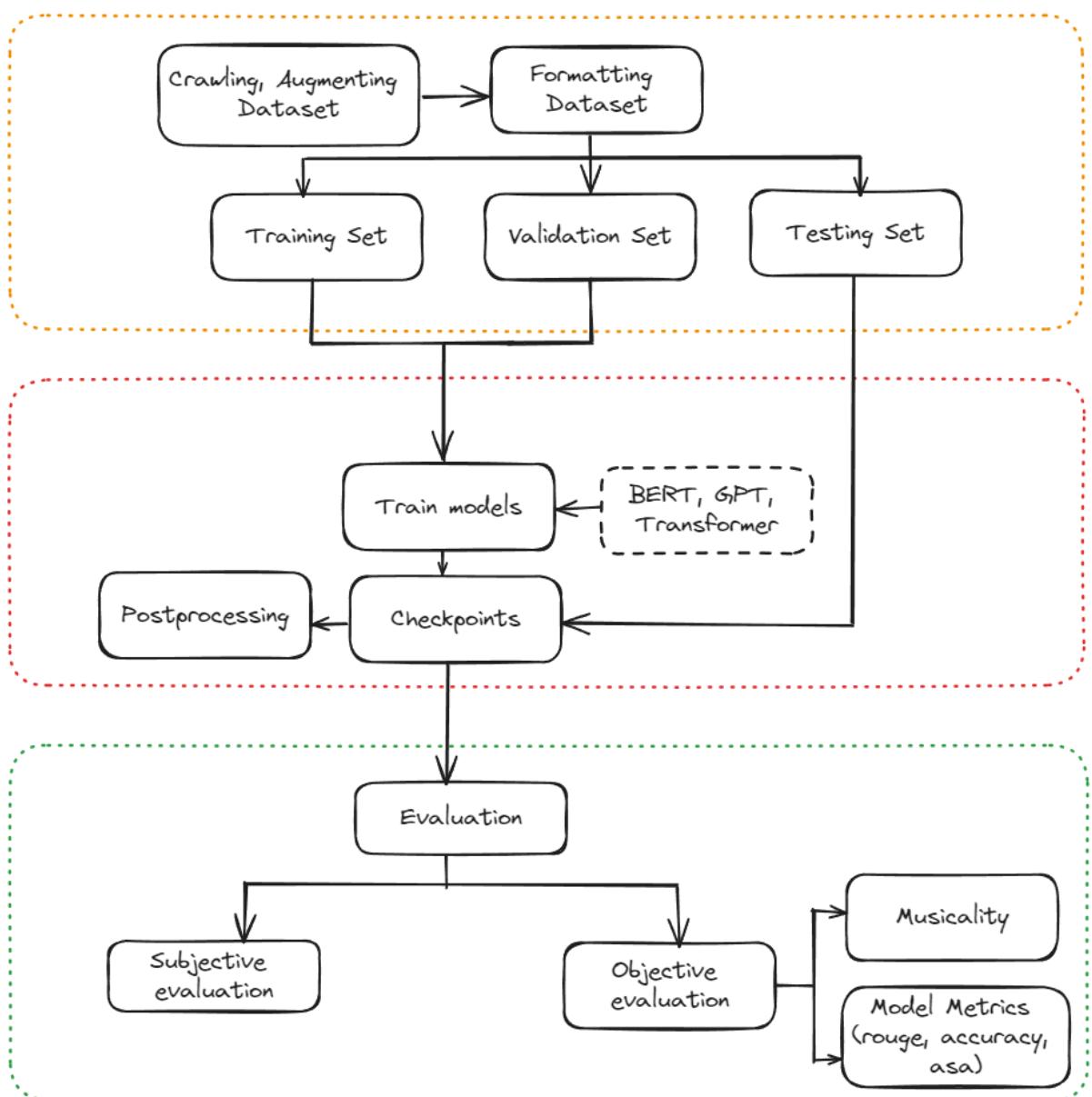
Nhờ vào những lựa chọn này, dự án không chỉ đạt được mục tiêu ban đầu về sinh nhạc từ ngôn ngữ tự nhiên mà còn đảm bảo hiệu quả và khả thi về mặt tài nguyên, mở ra khả năng áp dụng rộng rãi.

3.2.3 Các bước thực hiện

Các bước chính gồm xử lý dữ liệu, trích xuất thông tin từ câu nhập vào, và sinh nhạc được trực quan hóa qua sơ đồ ở hình 3.2. Tóm tắt các bước này gồm bảy ý chính:

1. Thu thập, tiền xử lý dữ liệu và trực quan hóa dữ liệu. Bao gồm việc thu thập dữ liệu thô, làm sạch và chuẩn hóa dữ liệu để phù hợp với nhu cầu của mô hình. Các bước này cũng bao gồm việc trực quan hóa dữ liệu để hiểu rõ hơn về cấu trúc và mối quan hệ trong dữ liệu.
2. Huấn luyện mô hình trích xuất đặc trưng âm nhạc: Phát triển và huấn luyện một mô hình để xác định và trích xuất các đặc trưng âm nhạc quan trọng từ dữ liệu đầu vào. Các đặc trưng này bao gồm nhãn về nhạc cụ, tốc độ bài hát, tông cao, và các yếu tố khác.
3. Huấn luyện mô hình sinh âm nhạc: Sử dụng các đặc trưng đã trích xuất để huấn luyện một mô hình sinh âm nhạc có khả năng tạo ra các bản nhạc mới dựa trên các câu dẫn đã được định nghĩa.
4. Đánh giá hiệu suất của mô hình: Kiểm tra hiệu suất của mô hình dựa trên các tiêu chí đã được đề cập trong chương trước, như độ chính xác của các đặc trưng âm nhạc được sinh ra và tính thẩm mỹ của bản nhạc.
5. So sánh và đánh giá kết quả từ các mô hình: So sánh hiệu suất giữa các mô hình khác nhau và các phương pháp huấn luyện để chọn ra mô hình tối ưu nhất.

6. Chuyển đổi các đặc trưng thành câu dẫn: Biến các đặc trưng âm nhạc thành câu dẫn sử dụng bộ từ điển đã cho trước và đưa chúng vào mô hình sinh nhạc.
7. Hậu xử lý (fill-missing) các cấu trúc lỗi nếu có, đảm bảo kết quả cuối cùng là một bài nhạc hoàn chỉnh.



Hình 3.2: Quy trình thực nghiệm

3.3 Chuẩn bị dữ liệu

Chuẩn bị dữ liệu là bước vô cùng quan trọng trong việc xây dựng các mô hình trí tuệ nhân tạo. Trong phần này, chúng tôi tập trung vào việc thu thập và tiền xử lý dữ liệu ngôn ngữ tự nhiên và dữ liệu âm nhạc, với các mục tiêu cụ thể liên quan đến các bộ dữ liệu MuseCoco, và Hooktheory (đã giới thiệu ở phần 2.3.1).

3.3.1 Dữ liệu ngôn ngữ tự nhiên

Mục tiêu đầu ra của công đoạn chuẩn bị dữ liệu ngôn ngữ tự nhiên là **các mẫu câu (template) mô tả âm nhạc** bằng tiếng Anh và tiếng Việt. Giá trị của các thuộc tính âm nhạc trong câu mô tả bị che bằng các nhãn (label) tương ứng. Dữ liệu này sẽ hỗ trợ trong quá trình huấn luyện các mô hình dạng Masked Language Modelling (kỹ thuật được giới thiệu ở mục 2.1.2.1).

Ví dụ câu template tiếng Anh: “The use of [KEY] key in this music creates a distinct atmosphere, while its exceptionally energetic beat adds to its unique character. The track has a duration of [TM1] seconds and is performed at a moderate speed.”

Ví dụ câu template tiếng Việt: “Bản nhạc có phạm vi cao độ giới hạn là [RANGE] quãng tám, nhấn mạnh vào các sắc thái của giai điệu và nhịp điệu. Sử dụng giọng [KEY], độ dài [TM1] giây.”

3.3.1.1 Thu thập

Để đáp ứng nhu cầu dữ liệu tiếng Anh, nhóm tận dụng lại bộ dữ liệu các câu template do nhóm tác giả bài báo MuseCoco[9] tạo ra bằng việc sử dụng ChatGPT trong quá trình nghiên cứu. Để đáp ứng nhu cầu dữ liệu tiếng Việt, nhóm thực hiện các thao tác xử lý, biến đổi để có được dữ liệu mong muốn (sẽ được nêu ở mục 3.3.1.2 kế tiếp).

3.3.1.2 Tiết xử lý

Để có được dữ liệu tiếng Việt, nhóm thực hiện dịch từ bộ dữ liệu tiếng Anh. Tuy nhiên, trong quá trình dịch thuận tuý, nhóm nhận thấy có hai vấn đề phát sinh:

1. Các từ chuyên ngành dễ bị dịch sai nghĩa.
2. Các nhãn khuyết đáng lẽ phải được giữ nguyên ở tiếng Anh để huấn luyện mô hình nhưng lại bị dịch sang tiếng Việt.

Do đó, nhóm đưa ra kỹ thuật dịch mới bằng cách chen vào các “ký tự gây nhiễu quá trình dịch”. Để làm được điều đó, trước hết, nhóm đưa ra cấu trúc dữ liệu (tạm gọi là “đơn vị hỗ trợ dịch thuật”) có dạng như sau:

```
1 {  
2     "attribute": "[TIME_SIGNATURE]" ,  
3     "prefix": "",  
4     "postfix": "time signature",  
5     "vietnamese_prefix": "",  
6     "vietnamese_postfix": "nhịp"  
7 }
```

Cấu trúc trên được xây dựng tập trung vào khoá **attribute** (tương ứng với nhãn khuyết). Khoá **prefix** và **postfix** tương ứng với những từ đứng trước và sau giá trị **attribute** trong câu template tiếng Anh. Khoá **vietnamese_prefix** và **vietnamese_postfix** tương ứng với những từ đứng trước và sau giá trị **attribute** trong câu template tiếng Việt, có ý nghĩa lần lượt tương đương với **postfix** và **prefix**.

Bằng việc sử dụng cấu trúc trên, quy trình dịch thuật diễn ra theo 8 bước sau:

1. Liệt kê các nhãn khuyết có trong dữ liệu tiếng Anh.

2. Lập danh sách các “đơn vị hỗ trợ dịch thuật” (thực thủ công).
3. Tạo hai từ điển để lưu các cặp “Chuỗi gốc - Chuỗi đã thả nhiều” chưa lần thả nhiều 1 và 2.
4. Với mỗi đơn vị hỗ trợ, thực hiện các bước sau:
 - (a) Thả nhiều lần 1 vào **attribute** bằng cách: Cứ mỗi ký tự của chuỗi, lần lượt chèn các số nguyên từ 1 đến 9 và lặp lại từ 0 mỗi lần vượt qua 9 (bắt đầu từ sau ký tự thứ hai của chuỗi và kết thúc ghi gấp dấu đóng ngoặc). Ví dụ:
“[TIME_SIGNATURE]” sau khi thả nhiều sẽ được
“[T1I2M3E4_5S6I7G8N9A0T1U2R3E4]”.
 - (b) Nối chuỗi để được cụm chứa nhiều lần 1 theo định dạng:
“prefix attribute_{chứa nhiều} postfix”.
 - (c) Lưu kết quả thả nhiều vào từ điển thả nhiều lần 1 với khoá là chuỗi chưa thả nhiều, giá trị là chuỗi sau khi thả nhiều lần 1.
 - (d) Thả nhiều lần 2 vào cụm chứa nhiều lần 1 bằng cách: Cứ mỗi hai ký tự của chuỗi, lần lượt chèn các số nguyên từ 0 đến 9 và lặp lại từ 0 mỗi lần vượt qua 9 (bắt đầu từ sau ký tự thứ hai của chuỗi). Sau đó cho cặp mở/đóng ngoặc vuông vào đầu và cuối chuỗi. Ví dụ:
“[T1I2M3E4_5S6I7G8N9A0T1U2R3E4] time signature” sau khi thả nhiều sẽ được
“[[T01I12M23E34_45S56I67G78N89A90T01U12R23E34]4
t5im6e 7si8gn9at0ur1e2]”.
 - (e) Lưu kết quả thả nhiều vào từ điển thả nhiều lần 2 với khoá là chuỗi chưa thả nhiều, giá trị là chuỗi sau khi thả nhiều lần 2.
 - (f) Tìm trong dữ liệu tiếng Anh các vị trí khớp chuỗi chưa thả nhiều và thay thế bằng chuỗi sau khi thả nhiều lần 2.
5. Lưu kết quả thả nhiều thành tập tin csv.

6. Sử dụng Google Sheets với hàm:

`GOOGLETRANSLATE(văn bản; ngôn ngữ nguồn; ngôn ngữ đích)`
để dịch văn bản tiếng Anh thành tiếng Việt và tải kết quả dịch về dưới dạng tập tin excel (xlsx).

7. Sử dụng bộ từ điển thả nhiều lần 1 và lần 2 truy ngược lại cụm từ tiếng Anh gốc để tìm “đơn vị hỗ trợ dịch thuật” tương ứng.
8. Thay cụm từ nhiều bằng chuỗi có định dạng “`vietnamese_prefix attribute vietnamese_postfix`” để hoàn chỉnh câu template tiếng Việt.

Bằng cách sử dụng kỹ thuật này, nhóm đã có thể đảm bảo rằng quá trình dịch thuật diễn ra chính xác hơn. Đảm bảo các nhãn khuyết được giữ nguyên và từ chuyên ngành không bị dịch sai.

Sau khi thu thập, xử lý và tổng hợp các bộ dữ liệu, nhóm có được một bộ dữ liệu câu template tiếng Anh và một bộ tiếng Việt tương ứng. Với phần giá trị cho các nhãn khuyết của các câu template, nhóm tái sử dụng lại từ dữ liệu của MuseCoco[9] do đã khá đa dạng (xem bảng 3.1).

Tên nhãn	Giá trị
Nhạc cụ	28 instruments: piano, keyboard, percussion, organ, guitar, bass, violin, viola, cello, harp, strings, voice, trumpet, trombone, tuba, horn, brass, sax, oboe, bassoon, clarinet, piccolo, flute, pipe, synthesizer, ethnic instrument, sound effect, drum. Mỗi nhạc cụ: 0: Được chơi, 1: Không được chơi, 2: NA
Pitch	Range: 0-11: octaves, 12: NA.
Rhythm Danceability (Nhịp điệu)	0: danceable, 1: not danceable, 2: NA.
Bar	0: 1-4 bars, 1: 5-8 bars, 2: 9-12 bars, 3: 13-16 bars, 4: NA.
Time Signature	0: 4/4, 1: 2/4, 2: 3/4, 3: 1/4, 4: 6/8, 5: 3/8, 6: các nhịp khác, 7: NA.
Key	0: major, 1: minor, 2: NA.
Tempo	0: chậm (≤ 76 BPM), 1: trung bình (76-120 BPM), 2: nhanh (≥ 120 BPM), 3: NA.
Time	0: 0-15s, 1: 15-30s, 2: 30-45s, 3: 45-60s, 4: >60 s, 5: NA.

Bảng 3.1: Tên và giá trị tương ứng của các nhãn trong mỗi câu template từ bài báo MuseCoco đã được rút gọn[9]

3.3.2 Dữ liệu âm nhạc

Mục tiêu của công đoạn chuẩn bị dữ liệu âm nhạc là tạo ra một bộ dữ liệu dưới định dạng MuseCoco (đã giới thiệu ở mục 2.3.3.4) từ các bản nhạc hiện đại và có tính cập nhật so với các bộ dữ liệu trước đây vốn tập trung vào nhạc cổ điển, rock, và pop theo phong cách của nhiều thập niên trước.

3.3.2.1 Thu thập

Bằng việc kết hợp lập trình đa luồng, xử lý theo lô, và tự động hóa trình duyệt, nhóm thực hiện cào dữ liệu theo các bước sau:

- Thiết lập số đoạn nhạc trong một lô.
- Thiết lập thời gian tạm dừng giữa các lô để tránh rate limiting.

3. Lập danh sách `id` của toàn bộ những đoạn nhạc cần thu thập.
4. Lập `webdriver_pool` để lưu các biến `Selenium webdriver` đang sử dụng.
5. Lặp lại các bước sau cho đến khi danh sách trống:
 - (a) Tạo một lô các đoạn nhạc theo số đoạn nhạc mỗi lô đã thiết lập.
 - (b) Gán gán các đoạn nhạc cần cào cho các `webdriver` (đồng thời bắt đầu chạy luồng tương ứng) đến khi `webdriver_pool` có số lượng bằng số đoạn nhạc mỗi lô đã quy định.
 - (c) Chạy lệnh thiếp lập thời gian đợi tối đa cho các luồng tương ứng với từng `webdriver` trong `webdriver_pool`.
 - (d) Kích hoạt chế độ chờ đến khi tất cả các luồng đạt thời gian đợi tối đa hoặc đã hoàn thành mới cho phép sang vòng lặp mới.
 - (e) Tạm dừng theo thời gian chờ giữa các lô đã thiết lập.
 - (f) Làm trống `webdriver_pool` và cập nhật danh sách các đoạn nhạc cần thu thập.

3.3.2.2 Tiền xử lý

Dữ liệu được thu thập từ file MIDI được xử lý theo các bước sau để chuẩn hóa thành dạng source-target hay command và music.

- Đầu tiên, sử dụng thư viện midi_data_extractor để trích xuất thông tin metadata từ một file MIDI. Metadata này sau đó được ánh xạ với bộ từ điển âm nhạc để tìm ra các thông tin như thời lượng bài nhạc, tốc độ, nhạc cụ được chơi, và các chi tiết khác.
- Phần command được tạo ra bằng cách ghép những metadata này thành các câu prompt phù hợp với bài nhạc. Đây là bản command chính để điều khiển sinh nhạc.
- Sau đó, file MIDI được chuẩn hóa thành dạng văn bản với các sự kiện REMI, chi tiết hoặc dựa trên các đặc điểm âm nhạc. Đây là phần biểu diễn cho target của tập dữ liệu.

Dưới đây là ví dụ về các thông tin trong metadata của một file MIDI:

I1s2 : (11, 4)

I4 : (11, False)

R3 : 1

B1s1 : (16, 3)

TS1s1 : (4, 4)

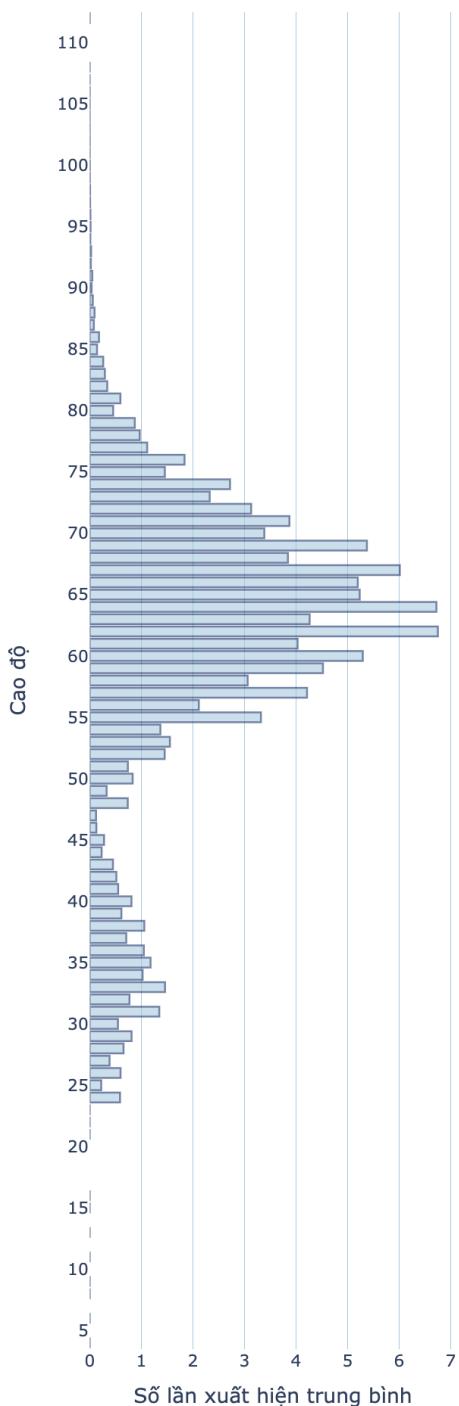
K1 : major

T1s1 : (114.03503592196344, 1)

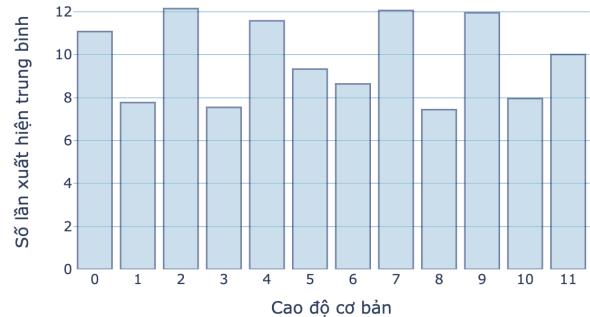
P4 : 3

TM1 : (33.45463058047076, 2)

3.3.2.3 Khai phá dữ liệu



Hình 3.3: Số lần xuất hiện trung bình của các giá trị cao độ (pitch) trong 29 038 đoạn nhạc từ bộ dữ liệu Hooktheory



Hình 3.4: Số lần xuất hiện trung bình của các giá trị cao độ (pitch) cơ bản trong 29 038 đoạn nhạc từ bộ dữ liệu Hooktheory

Số lượng nốt nhạc trung bình trong một đoạn nhạc từ Hooktheory ($n = 29038$): 117.50420139127878.

Hình 3.3 cho thấy có sự tập trung cao độ nhiều ở giữa, nguyên nhân là do hợp âm - thành phần chứa nhiều nốt nhất - tập trung ở khu vực này. Ở phía dưới ít hơn vì đó là âm vực của bass, tại mỗi thời điểm chỉ có một nốt đảm nhận vai trò bass. Phía trên cao cũng ít hơn vì đó là âm vực của giai điệu, tại mỗi thời điểm thường chỉ có một nốt giai điệu.

3.4 Huấn luyện mô hình

3.4.1 Mô hình trích xuất đặc trưng câu văn bản

Dự án này ứng dụng mô hình BERT, vốn là một mô hình thuộc loại encoder có khả năng trích xuất đặc trưng văn bản hiệu quả, để dự đoán các thuộc tính âm nhạc từ câu văn nhập vào. Dựa vào cấu trúc của mô hình BERT, nhóm nghiên cứu đã phát triển “MusicBert”, một biến thể được tinh chỉnh để phù hợp với nhiệm vụ phân loại các thuộc tính âm nhạc đa dạng. Nhóm đã áp dụng BERT cho việc dự đoán các thuộc tính xuất hiện trong câu như, có tiếng đàn piano hay không, thời lượng bản nhạc là bao lâu, tốc độ nhanh khoảng bao nhiêu, có được đề cập hay không. Kết quả sau khi được dự đoán sẽ đi qua một hàm Softmax để phân loại vào các nhánh cho trước. Gồm có hai loại thuộc tính là thuộc tính định tính và định lượng. Thuộc tính nhạc cụ (định tính) sẽ được phân loại theo 3 nhãn gồm: có xuất hiện, không xuất hiện, không được đề cập. Thuộc tính định lượng đo về đại lượng sẽ được sắp xếp theo từng mức độ đã quy định sẵn ví dụ như thời lượng thì sẽ có từ 0-15s hay 15-30s.

Từ kiến trúc BERT gốc ban đầu, chúng tôi đã lên ý tưởng, tham khảo từ nhiều nguồn và đã cài đặt để phù hợp cho các tác vụ phân loại nhiều lớp như sau:

- Token [CLS]: Được thêm vào trước khi huấn luyện, giúp phân loại với 53 nhãn tương ứng cho các thuộc tính nhạc cụ, tốc độ bài hát, và thời lượng.
- Lớp Pooler: Chịu trách nhiệm lấy ra các vector đại diện cho token [CLS] từ Embeddings.
- Hàm mất mát: Sử dụng CrossEntropyLoss cho việc phân loại đa nhãn.

- Lớp Softmax: Được sử dụng ở cuối để xác định xác suất của mỗi nhãn.
- Những lớp còn lại giống với mô hình gốc.

Mô hình trích xuất đặc trưng âm nhạc được tinh chỉnh và huấn luyện từ bộ trọng số của mô hình gốc là bert-large-multilingual-uncased, mô hình này hỗ trợ đa ngôn ngữ. Do đó, chúng tôi đã sử dụng tập dữ liệu tiếng Anh và tiếng Việt để huấn luyện.

Trước khi đưa dữ liệu vào mô hình, cần phải qua bước tiền xử lý. Sử dụng tokenizer để mã hóa văn bản đầu vào đi kèm theo các tham số như padding, max-length, và truncation. Nhãn là một chuỗi one-hot tương ứng với giá trị nào được đánh dấu là 1 thì được phân loại vào lớp đó. Kết quả sẽ trả ra đầu vào text đã được mã hóa để mô hình có thể hiểu được. Dữ liệu được chia theo tỷ lệ 80:10:10, lần lượt là tập train, tập valid, và tập test.

Hàm datacollator, hay gom dữ liệu, là thao tác để tổ chức dữ liệu huấn luyện trước khi đưa vào mô hình. Hàm này nhận danh sách các đặc trưng của dữ liệu đầu vào và trả về một từ điển chứa các tensors để sử dụng trong quá trình huấn luyện. Nhóm đã xác định dữ liệu nhãn và tạo tensors cho chúng. Các nhãn không phải là None và là một tensor sẽ được chuyển đổi sang kiểu dữ liệu phù hợp (long hoặc float) và chuyển thành tensor.

Chúng tôi đã sử dụng CrossEntropyLoss để tính toán hàm mất mát, đồng thời đánh giá được hiệu suất và độ chính xác của mô hình đưa ra. Ngoài ra, chúng tôi sử dụng thuật toán tối ưu hóa Adam với hệ số beta1 là 0.9 và beta2 là 0.999. Learning rate ban đầu được đặt là 0.0001. Mô hình được huấn luyện trên 100 epochs để đạt được kết quả tốt nhất.

Với việc huấn luyện mô hình đa ngôn ngữ, chúng tôi chọn cách thức huấn luyện các mô hình riêng biệt cho từng ngôn ngữ, sử dụng dữ liệu huấn luyện chỉ thuộc ngôn ngữ đó. Mỗi mô hình trung vào việc học các đặc trưng riêng của từng ngôn ngữ sẽ dễ đạt hiệu suất tốt hơn trên từng ngôn ngữ cụ thể (tiếng Anh hoặc tiếng Việt).

Bảng 3.2 thể hiện chi tiết từng tham số mà nhóm đã lựa chọn.

Siêu tham số	Giá trị	Ý nghĩa
epoch	100	Số lượng epoch xác định số lần mô hình sẽ được huấn luyện trên toàn bộ dữ liệu.
batch size	32	Kích thước của mỗi batch dữ liệu đưa vào mô hình trong mỗi bước huấn luyện.
optimizer	AdamW	Bộ tối ưu hóa sử dụng để cập nhật các tham số của mô hình trong quá trình huấn luyện.
learning rate	2e-5	Tốc độ học xác định mức độ điều chỉnh các trọng số của mô hình sau mỗi bước huấn luyện.
max sequence length	128	Chiều dài tối đa của chuỗi đầu vào, các chuỗi dài hơn sẽ bị cắt ngắn.
warmup steps	500	Số bước đầu tiên trong đó tốc độ học tăng dần đến giá trị tối đa đã định.
dropout rate	0.1	Tỷ lệ dropout sử dụng trong quá trình huấn luyện để tránh overfitting.
gradient accumulation steps	2	Số bước gradient accumulation trước khi cập nhật trọng số mô hình.
weight decay	0.01	Hệ số điều chỉnh tỷ lệ suy giảm trọng số, giúp tránh overfitting.

Bảng 3.2: Các hyperparameter được sử dụng trong quá trình huấn luyện mô hình BERT

3.4.2 Mô hình sinh nhạc

Khi bắt đầu với đề tài nghiên cứu về mô hình ngôn ngữ, một trong những thách thức lớn nhất là lựa chọn kiến trúc và mô hình sao cho phù

hợp với nguồn tài nguyên sẵn có nhưng vẫn đảm bảo cho ra kết quả tốt. Điều này đặt ra một bài toán cân bằng giữa hiệu suất của mô hình và tài nguyên tính toán. Nếu mô hình ngôn ngữ gốc chưa đủ mạnh, kết quả sẽ không đáp ứng được mục tiêu đề ra. Ngược lại, nếu sử dụng mô hình với nhiều tham số, lượng tài nguyên cần thiết sẽ rất lớn. Bài luận này sẽ thảo luận về các yếu tố cần xem xét khi lựa chọn mô hình ngôn ngữ, cũng như các giải pháp tiềm năng để tối ưu hóa tài nguyên và hiệu suất. Trong bối cảnh này, sự ra đời của Low-Rank Adaptation (LoRA) đã mang lại một giải pháp hiệu quả để giải quyết vấn đề này. Bài luận này sẽ thảo luận về các yếu tố cần xem xét khi lựa chọn mô hình ngôn ngữ, vai trò của LoRA, và cách LoRA có thể giúp tối ưu hóa tài nguyên và hiệu suất.

Trong quá trình huấn luyện, nhóm nghiên cứu của chúng tôi đã tiến hành thử nghiệm với hai phương pháp khác nhau để xác định phương pháp tối ưu nhất cho mục tiêu của dự án. Hai mô hình chính được sử dụng bao gồm:

- GPT-2 LoRA: Sử dụng kiến trúc GPT-2 đã được tối ưu bằng cách áp dụng phương pháp Low-Rank Adaptation (LoRA). Mô hình này được thiết kế để tăng khả năng mở rộng và linh hoạt của GPT-2 mà không làm tăng đáng kể số lượng tham số cần huấn luyện, giúp giảm thiểu yêu cầu về tài nguyên tính toán mà vẫn giữ được hiệu suất cao.
- Transformer Seq2Seq Sử Dụng Fairseq: Là một phương pháp tiếp cận khác sử dụng kiến trúc Transformer dạng sequence-to-sequence, được triển khai qua bộ công cụ Fairseq của Facebook AI Research. Kiến trúc Seq2Seq này được thiết kế để xử lý các nhiệm vụ biến đổi chuỗi, như dịch máy hoặc tổng hợp văn bản, và được tối ưu hóa để cung cấp một giải pháp hiệu quả cho việc sinh nhạc dựa trên ngôn ngữ tự nhiên.

3.4.2.1 Chuẩn bị tokenizer

Trong bài toán sinh nhạc, việc chuẩn bị dữ liệu đầu vào cho mô hình là một bước quan trọng để đảm bảo hiệu suất và độ chính xác của quá trình huấn luyện. Một phần không thể thiếu trong giai đoạn này là tạo ra một bộ tokenizer đặc trưng cho bộ dữ liệu âm nhạc. Tokenizer là công cụ phân tích và chuyển đổi dữ liệu đầu vào thành các đơn vị nhỏ hơn, gọi là token, để mô hình có thể xử lý hiệu quả.

Bộ dữ liệu âm nhạc có một bộ từ vựng và kiểu dữ liệu đặc trưng riêng, bao gồm các thuộc tính như nhạc cụ, cao độ, nhịp điệu, cường độ nhịp, ô nhịp, nhịp, khóa, tốc độ, thời gian, nghệ sĩ, thể loại, và cảm xúc. Mỗi thuộc tính này cần được mã hóa một cách chính xác để mô hình có thể hiểu và học từ dữ liệu.

Ví dụ, thuộc tính “nhạc cụ” có thể bao gồm 28 nhạc cụ khác nhau như piano, guitar, violin, và trống. Mỗi nhạc cụ cần được gán một mã số duy nhất để phân biệt trong quá trình mã hóa. Tương tự, thuộc tính “cao độ” có thể có phạm vi quãng tám từ 0 đến 11, trong khi “nhịp” có thể có các giá trị như $\frac{3}{4}$, $\frac{4}{4}$, và $\frac{6}{8}$. Mỗi giá trị này cần được chuyển đổi thành các token tương ứng.

Sau khi chuẩn bị được tập dữ liệu gồm các từ giả, bước tiếp theo là huấn luyện một tokenizer. Tokenizer được huấn luyện từ đầu để hiểu và xử lý các token âm nhạc. Quá trình này tương tự như huấn luyện một mô hình ngôn ngữ thông thường, nhưng ngôn ngữ ở đây là ngôn ngữ âm nhạc. Nhóm chúng tôi sử dụng bộ REMI do có các thư viện phù hợp để chuẩn hóa.

3.4.2.2 GPT-2 LoRA

Trong quá trình huấn luyện mô hình GPT-2 sử dụng kỹ thuật LoRA (Low-Rank Adaptation), chúng tôi đã thực hiện các bước chi tiết từ chuẩn bị dữ liệu, token hóa, cấu hình mô hình, áp dụng kỹ thuật LoRA, huấn luyện và đánh giá kết quả. Dưới đây là mô tả chi tiết từng bước và bảng

các hyperparameter sử dụng trong quá trình huấn luyện.

Dữ liệu được token hóa bằng cách sử dụng tokenizer với các tham số như độ dài tối đa và padding. Hàm tokenize sẽ xử lý dữ liệu đầu vào để phù hợp với mô hình. Dữ liệu được chia theo tỷ lệ 80:10:10, lần lượt là tập train, tập valid, và tập test.

Mô hình GPT-2 được cấu hình với các tham số như số lớp transformer, số lượng head trong multi-head attention, kích thước embedding, và các token đã được đặt sẵn. Với cách cấu hình này, mô hình được thiết lập để tận dụng các khả năng của GPT-2, đồng thời tối ưu hóa việc huấn luyện và dự đoán mà không cần phải cấu hình toàn bộ mô hình từ đầu. Thay vào đó, chúng ta chỉ sử dụng các lớp transformer với các tham số được cấu hình cụ thể để đạt hiệu quả cao nhất. Các tham số này đảm bảo rằng mô hình có đủ khả năng để học và dự đoán chính xác.

Kỹ thuật LoRA được cấu hình để cải thiện hiệu suất của mô hình. Chúng tôi thiết lập các tham số như r , lora_alpha, lora_dropout, và các module đích. Kỹ thuật này giúp giảm số lượng tham số cần huấn luyện và tăng tốc quá trình huấn luyện. Theo nghiên cứu từ bài báo QLoRA, các module cần được tinh chỉnh với LoRA để đạt được hiệu suất tương đương với việc tinh chỉnh toàn bộ mô hình bao gồm: gate_proj, down_proj, up_proj, q_proj, v_proj, k_proj, và o_proj. Các module này đều là các lớp tuyến tính và việc tinh chỉnh tất cả các lớp tuyến tính này là cần thiết để đạt được hiệu suất tối ưu.

Chúng tôi cấu hình các tham số huấn luyện như số epoch, batch size, optimizer, learning rate, và các chiến lược lưu và đánh giá mô hình. Sử dụng lớp Trainer của transformers, mô hình được huấn luyện với các dữ liệu đã token hóa và cấu hình huấn luyện đã thiết lập. Tổng số lượng tham số là 203 triệu.

Xem chi tiết các giá trị thông số ở bảng 3.3, 3.4, và 3.5.

Tham số	Giá trị	Ý nghĩa
epoch	10	Số lượng epoch xác định số lần mô hình sẽ được huấn luyện trên toàn bộ dữ liệu.
batch size	8	Kích thước của mỗi batch dữ liệu đưa vào mô hình trong mỗi bước huấn luyện.
optimizer	AdamW	Bộ tối ưu hóa sử dụng để cập nhật các tham số của mô hình trong quá trình huấn luyện.
learning rate	2e-4	Tốc độ học xác định mức độ điều chỉnh các trọng số của mô hình sau mỗi bước huấn luyện.
max sequence length	2048	Chiều dài tối đa của chuỗi đầu vào, các chuỗi dài hơn sẽ bị cắt ngắn.
warmup steps	2000	Số bước đầu tiên trong đó tốc độ học tăng dần đến giá trị tối đa đã định.
dropout rate	0.1	Tỷ lệ dropout sử dụng trong quá trình huấn luyện để tránh overfitting.
gradient accumulation steps	2	Số bước gradient accumulation trước khi cập nhật trọng số mô hình.
weight decay	0.01	Hệ số điều chỉnh tỷ lệ suy giảm trọng số, giúp tránh overfitting.
lora r	16	Hệ số r trong kỹ thuật LoRA xác định số chiều của ma trận low-rank.
lora alpha	12	Tham số alpha trong kỹ thuật LoRA, điều chỉnh mức độ ảnh hưởng của ma trận low-rank.
lora dropout	0.1	Tỷ lệ dropout áp dụng trong kỹ thuật LoRA để tránh overfitting.

Bảng 3.3: Các hyperparameter được sử dụng trong quá trình huấn luyện mô hình GPT-2 và kỹ thuật LoRA

Tham số	Giá trị	Ý nghĩa
n_layer	20	Số lượng lớp Transformer
n_head	16	Số lượng head của multi-head attention
n_emb	1024	Kích thước của vector embedding
vocab_size	1253	Kích thước từ vựng, tổng số lượng token
n_positions	2048	Số lượng vị trí tối đa trong một chuỗi

Bảng 3.4: Bảng tham số cấu hình cho mô hình GPT-2

Tham số	Giá trị	Ý nghĩa
r	16	Hệ số giảm chiều của các lớp được điều chỉnh
lora_alpha	12	Hệ số mở rộng (scaling factor) cho Lora
lora_dropout	0.1	Xác suất dropout áp dụng cho các lớp Lora
target_modules	[q_proj, k_proj, v_proj, out_proj, fc_in, fc_out, wte, lm_head]	Danh sách các mô-đun đích để áp dụng LoRA
task_type	CAUSAL_LM	Loại tác vụ mà mô hình được áp dụng

Bảng 3.5: Bảng tham số cấu hình cho LoraConfig.

3.4.2.3 Transformer Seq2Seq Sử Dụng Fairseq

Để thống nhất tham số giữa hai mô hình, nhóm đã thiết lập một loạt các thí nghiệm sử dụng cùng tham số với mô hình trước. Mô hình tiếp theo được huấn luyện và đánh giá trên cùng một tập dữ liệu. Các bài kiểm tra được thiết kế để đánh giá khả năng của mỗi mô hình trong việc xử lý và tạo ra âm nhạc phù hợp với các yêu cầu đặt ra, từ đó so sánh hiệu suất của chúng dựa trên các tiêu chuẩn như độ chính xác, tốc độ xử lý và hiệu quả sử dụng tài nguyên. Chúng tôi đã tham khảo cách cài đặt từ bài báo MuseCoco[9] để cấu hình theo bộ tham số của mình.

Trước hết, dữ liệu cần được chuẩn bị dưới dạng cặp câu “source - target”. Sau đó, tách bộ dữ liệu ra thành từng tập training và validation. Sử dụng bộ tokenizer đã chuẩn bị sẵn để token hoá dữ liệu. Fairseq hỗ trợ nhiều kiến trúc Transformer khác nhau, bao gồm cả Linear Casual Attention, phù hợp cho bài toán sinh nhạc. Tổng số lượng tham số là 203 triệu.

Việc chọn các siêu tham số phù hợp là yếu tố quan trọng trong việc huấn luyện mô hình ngôn ngữ để đạt được hiệu suất tối ưu. Các siêu tham số chính mà nhóm sử dụng trong quá trình huấn luyện được thể hiện ở bảng 3.6.

Tham số	Giá trị	Mô tả
n_layer	20	Số lớp trong Transformer.
n_head	16	Số lượng head attention.
n_emb	1024	Kích thước vector embeddings.
FFN size	2048	Kích thước của Feed Forward Network.
dropout ratio	0.1	Tỷ lệ dropout để tránh overfitting.
optimizer	AdamW	Optimizer sử dụng để huấn luyện mô hình.
β_1	0.9	Tham số β_1 của Adam optimizer.
β_2	0.98	Tham số β_2 của Adam optimizer.
ϵ	10^{-9}	Tham số ϵ của Adam optimizer.
learning rate	2×10^{-4}	Tốc độ học của mô hình.
warmup steps	2000	Số bước đầu tiên để tốc độ học tăng dần đến giá trị tối đa đã định.

Bảng 3.6: Các tham số của mô hình Casual Linear Transformer

3.5 Hậu xử lý dữ liệu

Như chúng ta đã biết, dù mô hình có tốt đến đâu, vẫn luôn tồn tại rủi ro rằng đầu ra của mô hình không thể mã hóa thành bản nhạc do các sai

sót khi sinh ra. Để đảm bảo mô hình luôn cho ra kết quả hợp lệ, nhóm chúng tôi đã đề xuất một thuật toán kiểm tra và sửa lỗi.

File âm nhạc cũng là một dạng của dữ liệu chuỗi thời gian (timeseries). Từ ý tưởng đó, chúng tôi kiểm tra từng vị trí tương ứng với các thời điểm có sự kiện âm thanh như tempo thay đổi, nhịp thay đổi, hoặc note mới xuất hiện. Chúng tôi xem xét từng sự kiện đó có tuân theo các quy luật hay không. Các quy luật này được ánh xạ vào bộ quy tắc REMI.

Sau khi kiểm tra toàn bộ các sự kiện, nếu có sự kiện nào bị lỗi, thông thường chúng sẽ bị bỏ qua, nhưng điều này có thể làm mất tính mạch lạc của bài nhạc. Để khắc phục điều này, chúng tôi đề xuất sửa lỗi bằng cách thay thế cấu trúc của sự kiện lỗi bằng cấu trúc của sự kiện gần nhất đúng. Tuy nhiên, giá trị của sự kiện lỗi sẽ được điền vào cấu trúc mới này để duy trì tính chính xác của thông tin.

Nhờ vào thuật toán này, chúng tôi có thể giảm thiểu các sai sót và đảm bảo đầu ra của mô hình luôn hợp lệ và mạch lạc.

3.6 Phương pháp đánh giá mô hình

3.6.1 Đánh giá mô hình trích xuất đặc trưng

Bản chất của bài toán trích xuất đặc trưng là một dạng của mô hình phân loại đa nhãn, trong đó mỗi mẫu dữ liệu có thể thuộc về nhiều nhãn khác nhau cùng một lúc. Do đó, việc đánh giá mô hình phân loại đa nhãn cũng sử dụng các độ đo tương tự như các mô hình phân loại khác, bao gồm accuracy, precision, recall và F1-score. Những độ đo này giúp đánh giá được hiệu suất của mô hình dựa trên mức độ chính xác của các nhãn mà mô hình dự đoán được so với nhãn thực tế. Accuracy trong bối cảnh này được tính dựa trên tỷ lệ các nhãn đúng mà mô hình dự đoán được so với tổng số nhãn. Chúng tôi đã tính toán trên từng loại nhãn để đánh giá chất lượng của mô hình.

3.6.2 Đánh giá mô hình sinh nhạc

Mô hình sinh nhạc cơ bản cũng là mô hình phân loại. Việc đánh giá được thực hiện theo hướng kiểm tra đầu ra có giống với câu prompt đầu vào hay không. Bên cạnh đó, cần phải xem xét tính nhạc của bài nhạc. Như vậy, nhóm sẽ đánh giá dựa trên hai yếu tố: khách quan và chủ quan.

Phương pháp đánh giá chủ quan được đánh giá bằng cách đánh giá từ quan điểm cá nhân, dựa trên cảm nhận và ảnh hưởng mà âm nhạc tạo ra đối với người nghe.

Yếu tố khách quan dựa trên các độ đo, gồm có hai loại: độ đo về hiệu suất mô hình và độ đo về nhạc lý dựa trên lý thuyết âm nhạc. Để đánh hiệu suất mô hình, chúng tôi đã sử dụng các thư viện nhằm trích xuất ngược lại các đặc trưng của một tập tin MIDI như các nhạc cụ xuất hiện trong bài, thời gian, tốc độ để so khớp với câu prompt đầu vào, sau đó tính toán độ chính xác dựa trên các thuộc tính đó.

Sau khi đánh giá hiệu suất của mô hình, chúng tôi còn đánh giá chất lượng bản nhạc sinh ra. Mặc dù bản nhạc có thể ra đúng về mặt cấu trúc dữ liệu nhưng không đảm bảo được chất lượng về nhạc lý thì mô hình vẫn không tốt. Chúng tôi đã sử dụng các thư viện để đo các thông số về sự sáng tạo của bản nhạc, sự phân bố nốt nhạc thường được dùng để thấy xu hướng sử dụng nốt nhạc và cách tạo sắc thái âm nhạc trong tác phẩm, và chỉ số trung bình số lượng nốt nhạc trong mỗi bài hát phản ánh phong cách sáng tác hoặc biểu diễn. Bên cạnh đó, chúng tôi cũng đưa ra trực quan về sự so sánh giữa nhạc do mô hình sinh ra và nhạc do các nghệ sĩ sáng tác nhằm có cái nhìn tổng quát.

Chương 4

Kết quả thí nghiệm

Chương này trình bày về kết quả thu được sau quá trình huấn luyện các mô hình.

4.1 Kết quả huấn luyện mô hình

4.1.1 Mô hình trích xuất đặc trưng

Kết quả được tổng hợp sau quá trình huấn luyện và dự đoán với bộ checkpoint tốt nhất và thực hiện đánh giá trên tập kiểm tra gồm 1000 mẫu.

Với các thuộc tính âm nhạc liên quan đến các loại nhạc cụ (instrument), khi đánh giá, chúng tôi chỉ tập trung vào các loại nhạc cụ phổ biến trong dữ liệu huấn luyện. Kết quả được đo lường và cho thấy độ chính xác rất cao (xem bảng 4.1).

Cụ thể, các loại nhạc cụ như accordion, brass, celesta, choir, guitar, harmonica, organ, piano, synth, viola, violin, và voice đều đạt độ chính xác cao, dao động từ 0.90 đến 0.99 trong cả tiếng Anh và tiếng Việt. Điều này cho thấy mô hình trích xuất đặc trưng của chúng tôi hoạt động tốt và nhất quán với các loại nhạc cụ này, bất kể ngôn ngữ đánh giá.

Instrument	Accuracy (Tiếng Anh)	Accuracy (Tiếng Việt)
accordion	0.94	0.93
brass	0.98	0.96
celesta	0.91	0.92
choir	0.95	0.97
guitar	0.99	0.93
harmonica	0.97	0.94
organ	0.90	0.91
piano	0.96	0.95
synth	0.92	0.94
viola	0.91	0.90
violin	0.93	0.92
voice	0.95	0.96

Bảng 4.1: Instrument Accuracy

Với các thuộc tính âm nhạc khác, kết quả được đo lường cho thấy độ chính xác rất cao, các thuộc tính như “Time Signature” và “Pitch Range” đạt độ chính xác cao nhất, lần lượt là 0.94 (tiếng Anh) và 0.94 (tiếng Việt). Điều này cho thấy mô hình có khả năng phân loại tốt các yếu tố liên quan đến nhịp điệu và phạm vi âm thanh. Các thuộc tính như “Rhythm Danceability” và “Tempo” có độ chính xác thấp hơn, dao động từ 0.85 đến 0.93, nhưng vẫn cho thấy mức độ chính xác cao và đáng tin cậy (xem bảng 4.2).

Category	Accuracy (Tiếng Anh)	Accuracy (Tiếng Việt)
Rhythm Danceability	0.85	0.87
Bar	0.91	0.89
Time Signature	0.94	0.92
Key	0.88	0.90
Tempo	0.93	0.85
Pitch Range	0.90	0.94

Bảng 4.2: Categories Accuracy

4.1.2 Mô hình sinh nhạc

Mô hình sinh nhạc bao gồm hai loại là GPT2 LoRA và Fairseq Transformer được huấn luyện với 50000 steps bằng bộ tham số ở bảng 4.3. Sau khi cho mô hình sinh ra 500 mẫu, nhóm đo kết quả trên các thuộc tính quyết định tính chất của một bài hát (xem bảng 4.4).

4.1.2.1 Đánh giá dựa trên các tiêu chí của mô hình.

Tham số	Giá trị	Ý nghĩa
num_beams	5	Số lượng beams cho beam search giúp tạo ra các chuỗi chất lượng cao hơn.
no_repeat_ngram_size	2	Đảm bảo rằng các n-gram có kích thước được chỉ định không lặp lại trong văn bản được tạo ra.
early_stopping	True	Nếu đặt thành True, beam search sẽ dừng khi ít nhất num_beams câu hoàn thành mỗi batch.
temperature	0.2	Kiểm soát sự ngẫu nhiên của quá trình tạo ra văn bản.

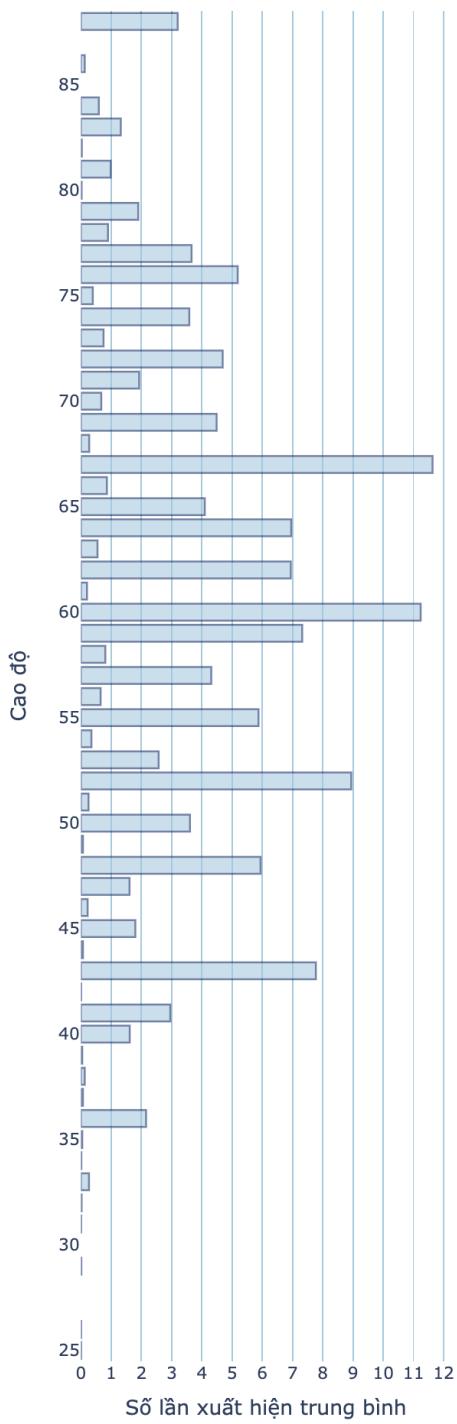
Bảng 4.3: Các tham số cho hàm generate

Độ đo	LoRA GPT2	Fairseq Transformer seq2seq
ASA	0.64	0.57
Nhạc cụ	0.67	0.6
Pitch Range	0.68	0.52
Rhythm Danceability	0.96	0.89
Bar	0.51	0.42
Time Signature	0.37	0.43
Key	0.57	0.51
Tempo	0.69	0.61

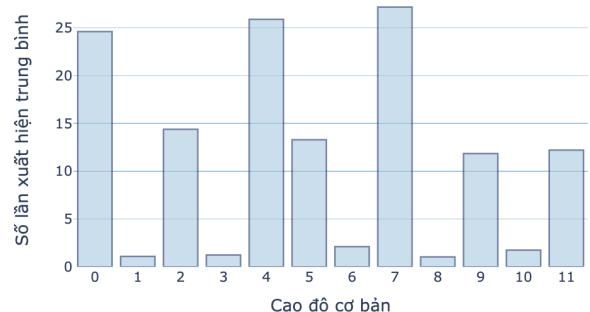
Bảng 4.4: So sánh kết quả giữa hai mô hình.

Độ chính xác của các thuộc tính âm nhạc giữa hai mô hình LoRA GPT2 và Fairseq Transformer seq2seq thể hiện ở bảng 4.4 cho thấy mô hình LoRA GPT2 vượt trội hơn về hầu hết các độ đo. Đặc biệt, trong các độ đo ASA và Rhythm Danceability, LoRA GPT2 có độ chính xác rất cao, lần lượt là 0.64 và 0.96. Tuy nhiên, đối với độ đo Time Signature, Fairseq Transformer seq2seq có kết quả cao hơn một chút với giá trị 0.43 so với 0.37 của LoRA GPT2. Nhìn chung, mô hình LoRA GPT2 thể hiện hiệu quả tốt hơn trong việc dự đoán các thuộc tính âm nhạc phổ biến.

4.1.2.2 Đánh giá theo nhạc tính.



Hình 4.1: Số lần xuất hiện trung bình của các giá trị cao độ (GPT2 LoRA).



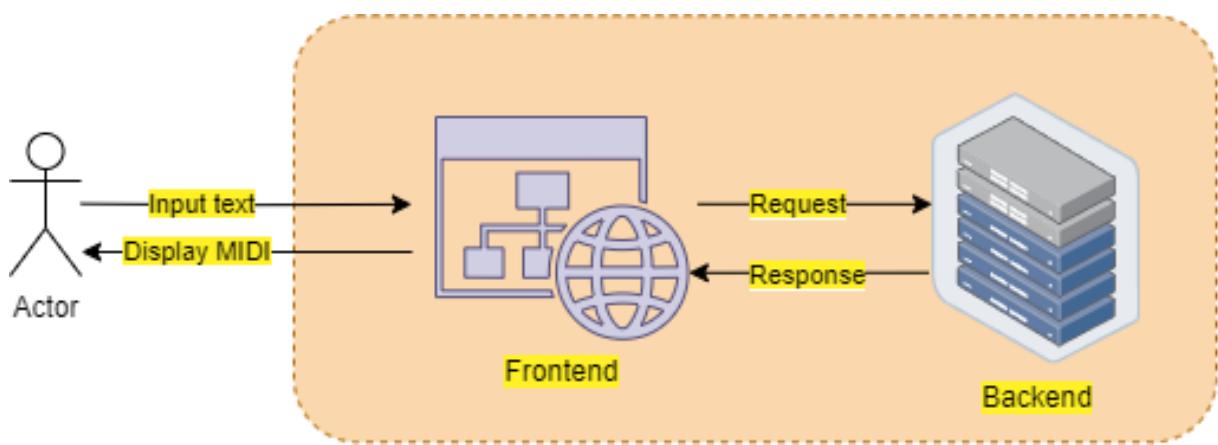
Hình 4.2: Số lần xuất hiện trung bình của các giá trị cao độ cơ bản (GPT2 LoRA).

Số lượng nốt nhạc trung bình trong một đoạn nhạc từ GPT2 LoRA: **136.66666666666667**. 136.67

Hình 4.1 cho thấy số lượng nốt trung bình được phân bố khá đều cho mỗi cao độ cơ bản.

4.2 Phần mềm demo

Ứng dụng sinh nhạc sử dụng kiến trúc client-server là một hệ thống cho phép người dùng nhập vào văn bản để tạo ra nhạc dựa trên nội dung văn bản đó. Hệ thống này sử dụng mô hình ngôn ngữ để phân tích và chuyển đổi văn bản thành nhạc. Kiến trúc client-server giúp tách biệt các thành phần xử lý, đảm bảo hiệu suất và dễ dàng mở rộng hệ thống.

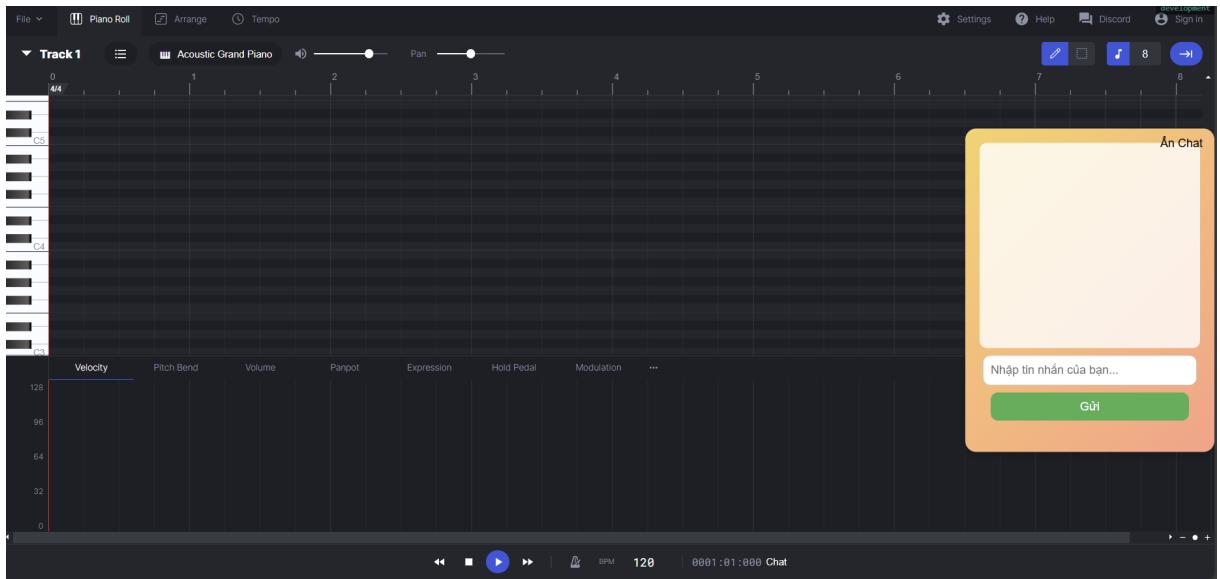


Hình 4.3: Kiến trúc client-server

Hai thành phần chính của kiến trúc:

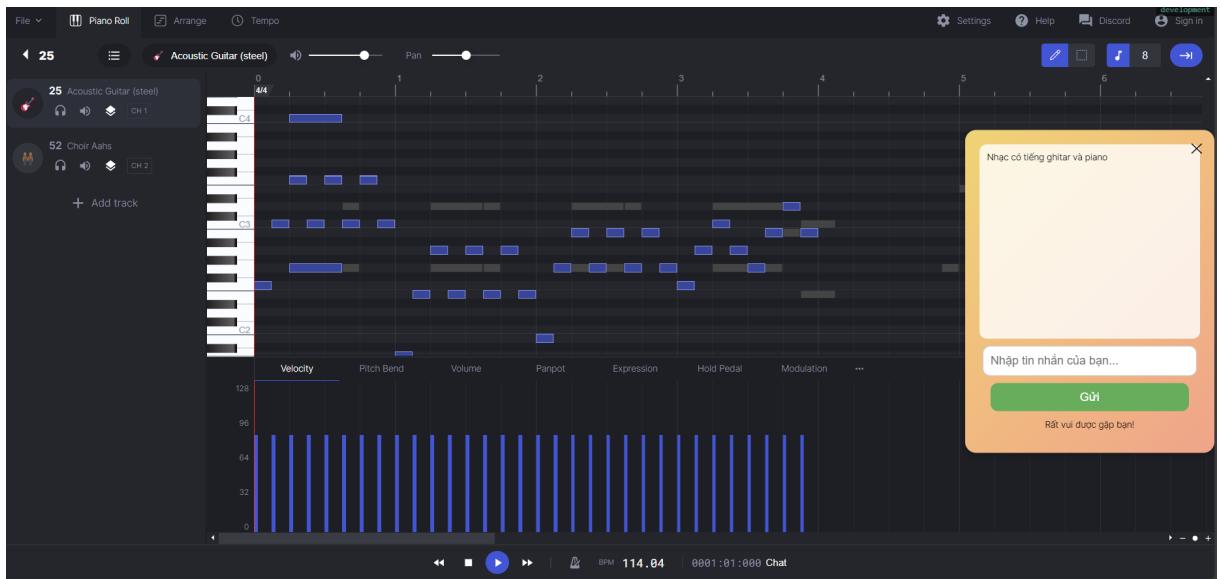
1. Client: Nhóm sử dụng giao diện chỉnh sửa tập tin MIDI từ dự án mã nguồn mở Signal¹, và bổ sung thêm chức năng nhập văn bản để sinh nhạc. Ở giao diện này người dùng có thể nhập mô tả âm nhạc bằng ngôn ngữ tự nhiên để sinh nhạc, hoặc sử dụng như ứng dụng soạn nhạc bằng MIDI (xem hình 4.4).
2. Server: Được xây dựng với Python FastAPI, giúp xử lý các yêu cầu từ client, bao gồm phân tích văn bản và sinh nhạc dựa trên văn bản đó.

¹Đường dẫn github: <https://github.com/ryohey/signal> (truy cập lần cuối: 30/06/2024).



Hình 4.4: Giao diện tổng thể

Với chức năng tạo nhạc từ câu mô tả do người dùng nhập vào, sau khi nhấn “Gửi”, một bài nhạc sẽ xuất hiện trong khu vực hiển thị MIDI. Người dùng cũng có thể dễ dàng tùy chỉnh các nốt nhạc, nhạc cụ, và các thông số khác trên giao diện này (xem hình 4.5).



Hình 4.5: Giao diện khi có nhạc sinh ra

Chương 5

Kết luận

Qua quá trình nghiên cứu và thực hiện khóa luận này, chúng tôi đã thực nghiệm phương pháp sử dụng hai mô hình nhỏ gọn hơn để xử lý tác vụ sinh nhạc từ câu ngôn ngữ tiếng Anh hoặc tiếng Việt một cách hiệu quả, thay vì sử dụng một mô hình lớn như truyền thống. Chúng tôi đã thử nghiệm trên nhiều kiến trúc mô hình và điều chỉnh nhiều tham số để đưa ra cái nhìn tổng quát và chi tiết về khả năng sinh nhạc của kiến trúc này cũng như việc so sánh các mô hình Transformer để biết được mô hình nào phù hợp hơn cho mỗi tác vụ trong kiến trúc. Kết quả cho thấy việc sử dụng các kỹ thuật casual language modelling trong các mô hình ngôn ngữ sẽ phù hợp với tác vụ sinh nhạc.

Các kỹ thuật về giảm ma trận LoRA giúp việc huấn luyện trở nên dễ dàng hơn. Việc tinh chỉnh LoRA cũng được thử nghiệm qua các mô-đun nhằm tối ưu hóa việc huấn luyện và cho ra kết quả tốt. Thực nghiệm cho thấy việc tinh chỉnh trên các lớp tuyến tính là cần thiết để đạt được hiệu suất tương đương với việc tinh chỉnh toàn bộ mô hình (các lớp bao gồm gate_proj, down_proj, up_proj, q_proj, v_proj, k_proj, o_proj) đóng vai trò quan trọng trong việc biến đổi và kết hợp thông tin. Bên cạnh đó, nhóm đã thực nghiệm tinh chỉnh trên nhiều bộ tham số vừa đề cập nhằm cho ra kết quả tốt nhất.

Để đảm bảo kết quả chương trình luôn cho ra bản nhạc hoàn chỉnh, nhóm đã đưa ra kỹ thuật kiểm tra tính đúng đắn trong bài hát và điền

chỗ còn thiếu bằng phương pháp nội suy dựa trên nguyên tắc nhạc lý và ràng buộc của cấu trúc dữ liệu. Khâu này diễn ra sau khi mô hình sinh ra nhạc để đảm bảo kết quả đủ tốt và ứng dụng được trong sản phẩm.

Mặc dù các phương pháp trên đều cho kết quả tương đối tốt, vẫn cần nghiên cứu thêm để điều chỉnh các bộ tham số trên các tập dữ liệu phong phú và đa dạng hơn nhằm đạt được mô hình hỗ trợ nhiều thể loại, màu sắc trong âm nhạc. Các thí nghiệm trên chỉ thực hiện trong quá trình huấn luyện bằng LoRA nhưng vẫn cho kết quả tốt, thể hiện tiềm năng việc huấn luyện với mô hình gốc và đầy đủ sẽ cho ra kết quả tốt hơn.

Tài liệu tham khảo

Tiếng Anh

- [1] Agostinelli, Andrea et al. *MusicLM: Generating Music From Text*. 2023. arXiv: 2301.11325 [cs.SD].
- [2] Copet, Jade et al. “Simple and Controllable Music Generation”. In: *Thirty-seventh Conference on Neural Information Processing Systems*. 2023.
- [3] Ens, Jeff and Pasquier, Philippe. *MMM: Exploring Conditional Multi-Track Music Generation with the Transformer*. 2020. arXiv: 2008.06048 [cs.SD].
- [4] Fradet, Nathan et al. “Byte Pair Encoding for Symbolic Music”. In: *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*. Ed. by Bouamor, Houda, Pino, Juan, and Bali, Kalika. Singapore: Association for Computational Linguistics, Dec. 2023, pp. 2001–2020. DOI: 10.18653/v1/2023.emnlp-main.123. URL: <https://aclanthology.org/2023.emnlp-main.123>.
- [5] Hadjeres, Gaëtan and Crestel, Léopold. *The Piano Inpainting Application*. 2021. arXiv: 2107.05944 [cs.SD]. URL: <https://arxiv.org/abs/2107.05944>.
- [6] Hayes, Thomas et al. *MUGEN: A Playground for Video-Audio-Text Multimodal Understanding and GENeration*. 2022. arXiv: 2204.08058 [cs.CV].

- [7] Hsiao, Wen-Yi et al. “Compound Word Transformer: Learning to Compose Full-Song Music over Dynamic Directed Hypergraphs”. In: *Proceedings of the AAAI Conference on Artificial Intelligence* 35.1 (2021), pp. 178–186. DOI: 10.1609/aaai.v35i1.16091. URL: <https://ojs.aaai.org/index.php/AAAI/article/view/16091>.
- [8] Huang, Yu-Siang and Yang, Yi-Hsuan. *Pop Music Transformer: Beat-based Modeling and Generation of Expressive Pop Piano Compositions*. 2020. arXiv: 2002.00212 [cs.SD].
- [9] Lu, Peiling et al. *MuseCoco: Generating Symbolic Music from Text*. 2023. arXiv: 2306.00110 [cs.SD].
- [10] MIDI Association. *Summary of MIDI 1.0 Messages*. URL: <https://midi.org/summary-of-midi-1-0-messages> (visited on 06/30/2024).
- [11] MIDITok. *Tokenizations*. URL: <https://miditok.readthedocs.io/en/latest/tokenizations.html> (visited on 06/30/2024).
- [12] Music, Yating. *midi2remi.ipynb*. 2024. URL: <https://github.com/YatingMusic/remi/blob/master/midi2remi.ipynb> (visited on 06/30/2024).
- [13] Oore, Sageev et al. “This Time with Feeling: Learning Expressive Musical Performance”. In: *Neural Computing and Applications* 32 (2018), 955–967. URL: <https://link.springer.com/article/10.1007/s00521-018-3758-9>.
- [14] Payne, Christine. *MuseNet*. 2019. URL: <https://openai.com/blog/musenet/> (visited on 06/30/2024).