TRƯỜNG ĐẠI HỌC CẦN THƠ

TRƯỜNG CÔNG NGHỆ THÔNG TIN & TRUYỀN THÔNG

LUẬN VĂN TỐT NGHIỆP ĐẠI HỌC

NGÀNH HỆ THỐNG THÔNG TIN

Đề tài

TÌM KIẾM NHẠC KHÔNG LỜI SỬ DỤNG CÁC PHƯƠNG PHÁP TRỰC QUAN PHỔ TẦN SỐ VÀ MẠNG NƠ-RON TÍCH CHẬP

Sinh viên: Lê Tuấn Anh

Mã số: B1906362

Khóa: K45

Cần Thơ, 12/2023

TRƯỜNG ĐẠI HỌC CẦN THƠ
TRƯỜNG CÔNG NGHỆ THÔNG TIN & TRUYỀN THÔNG

LUẬN VĂN TỐT NGHIỆP ĐẠI HỌC
NGÀNH HỆ THỐNG THÔNG TIN

Đề tài

TÌM KIẾM NHẠC KHÔNG LỜI SỬ DỤNG CÁC PHƯƠNG PHÁP TRỰC QUAN PHỔ TẦN SỐ VÀ MẠNG NƠ-RON TÍCH CHẬP

Người hướng dẫn
TS.Nguyễn Thanh Hải

Sinh viên thực hiện
Lê Tuấn Anh
Mã số: B1906362
Khóa: K45

Cần Thơ, 12/2023

ĐẠI HỌC CẦN THƠ
TRƯỜNG CÔNG NGHỆ THÔNG TIN & TRUYỀN THÔNG

CỘNG HÒA XÃ HỘI CHỦ NGHĨA VIỆT NAM
Độc lập-Tự do-Hạnh phúc

XÁC NHẬN CHỈNH SỬA LUẬN VĂN THEO YÊU CẦU CỦA HỘI ĐỒNG

Tên luận văn (Tiếng Việt và Tiếng Anh):
Tìm kiếm nhạc không lời sử dụng các phương pháp trực quan phổ tần số và mạng nơ-ron tích chập
Instrumental song classification using spectrograms and convolutional neural networks

Họ tên sinh viên: Lê Tuấn Anh MSSV: B1906362

Mã lớp: DI1995A2

Đã báo cáo tại hội đồng ngành: Hệ thống thông tin

Ngày báo cáo: 14/12/2023

Luận văn đã được chỉnh sửa theo góp ý của Hội đồng.

Cần Thơ, ngày … tháng 12 năm 2023

Giáo viên hướng dẫn

(Ký và ghi họ tên)

Nguyễn Thanh Hải

LỜI CẢM ƠN

Lời đầu tiên em xin gửi lời cảm ơn chân thành đến quý thầy cô Trường Đại học Cần Thơ, đặc biệt là quý thầy cô của Trường Công nghệ Thông tin và Truyền thông đã tận tình chỉ bảo, truyền đạt kiến thức cho em, giúp em có được nền tảng tốt nhất để vững bước trên con đường ở tương lai.

Em xin gửi lời cảm ơn chân thành đến Thầy Nguyễn Thanh Hải, cảm ơn thầy đã tận tình hướng dẫn, giúp đỡ, góp ý và động viên em trong suốt thời gian làm luận văn cũng như đã đồng hành, dẫn dắt giúp đỡ em trong suốt khoảng thời gian học đại học, cảm ơn thầy đã dành nhiều thời gian và công sức để hỗ trợ em trong suốt thời gian qua.

Em cảm ơn cô Bùi Đăng Hà Phương, thầy Trần Thanh Điện đã tham gia phản biện và góp ý giúp bài luận văn của em được hoàn thiện hơn.

Em cảm ơn cô Nguyễn Thị Thanh Hiền đã đồng hành, dẫn dắt và giúp đỡ em rất nhiều trong quá trình nghiên cứu đề tài, cảm ơn cô đã dành nhiều thời gian và công sức hỗ trợ em khoảng thời gian qua.

Em cũng xin gửi lời cảm ơn đến gia đình và bạn bè đã luôn bên cạnh ủng hộ em trong suốt thời gian vừa qua. Gia đình luôn tạo mọi điều kiện tốt nhất cho em học tập, song song với điều đó mỗi khi em gặp khó khăn thì gia đình luôn cho em lời khuyên và động viên em vượt qua mọi thứ. Cảm ơn tất cả những người bạn quý giá, đã bên cạnh sẻ chia những buồn vui cũng như khó khăn trong quá trình học tập và cuộc sống trong suốt khoảng thời gian qua.

Cuối cùng với lòng biết ơn sâu sắc nhất con xin cảm ơn gia đình, những người đã luôn luôn tin tưởng và ủng hộ con hết mình trong mọi việc, luôn dành những thứ tốt đẹp nhất cho con.

Bên cạnh những kết quả đã đạt được,khó tránh khỏi những sai sót và thiếu sót trong quá trình thực hiện đề tài. Rất mong quý thầy cô thông cảm, mong nhận được sự nhận xét và góp ý của các thầy, qua đó sẽ giúp em hoàn thiện đề tài này hơn nữa.Vì mỗi ý kiến đóng góp của quý thầy cô đều rất đáng trân trọng và là những kinh nghiệm, kiến thức có thể giúp em hoàn thiện bản thân mình hơn.

Bằng tất cả sự chân thành, một lần nữa xin kính chúc tất cả mọi người luôn vui khỏe, hạnh phúc và thành công hơn nữa trong tương lai.

Trân trọng!

Cần Thơ, ngày… tháng 12 năm 2023
Tác giả

Lê Tuấn Anh

NHẬN XÉT CỦA GIÁO VIÊN HƯỚNG DẪN

……
……
……
……
……
……
……
……
……
……
……
……
……
……
……
……
……
……
……
……
……
……
……
……
……
……
……
……
……
……
……
……
……
……
……

Cần Thơ, ngày… tháng 12 năm 2023
Giáo viên hướng dẫn

TS. Nguyễn Thanh Hải

NHẬN XÉT CỦA GIÁO VIÊN PHẢN BIỆN

……
……
……
……
……
……
……
……
……
……
……
……
……
……
……
……
……
……
……
……
……
……
……
……
……
……
……
……
……
……

Cần Thơ, ngày… tháng 12 năm 2023
Giáo viên phản biện

TÓM TẮT

Trong thời đại công nghệ thông tin phát triển nhanh chóng và cuộc sống ngày càng hiện đại, nhu cầu giải trí trở nên cần thiết hơn, đặc biệt là trong lĩnh vực thưởng thức âm nhạc. Việc đang tìm kiếm một bài hát là một điều cần thiết, bên cạnh đó bản quyền bài hát cũng là điều đáng quan tâm. Vì vậy, việc nhận diện một bài hát từ một đoạn nhạc nhỏ là cần thiết và hữu ích, được đề xuất để giúp người nghe có thể tìm kiếm được những bài nhạc không lời. Nghiên cứu này đề xuất một phương pháp phân loại và xác định các bài hát dựa trên các tính năng cụ thể mà mô hình học được từ dữ liệu âm nhạc. Ngôn ngữ lập trình Python được sử dụng để xây dựng mô hình Convolutional Neural Network (CNN) và Fully Connected Layer (FC) . Trước hết xử lý, các thư viện hỗ trợ được sử dụng để trích xuất dữ liệu âm thanh. Tập dữ liệu A bao gồm 100 bài hát không lời, trong khi tập dữ liệu B bao gồm 100 tệp âm thanh có cùng tên bài hát nhưng chơi bằng nhiều loại nhạc cụ khác nhau. Thư viện pydub sau đó cắt ngẫu nhiên các file âm thanh gốc thành các clip có thời lượng dưới 10 giây vì người dùng thường sử dụng một đoạn nhỏ cụ thể để tìm toàn bộ bản nhạc. Các tệp âm thanh gốc được chia thành các đoạn có độ dài khác nhau trong quá trình đào tạo, bao gồm 1 giây, 3 giây, 5 giây, 10 giây, 20 giây, 30 giây, 60 giây và 90 giây. Tiếp theo, phương pháp chuyển đổi tệp âm thanh thành ảnh phổ được thực hiện thông qua việc sử dụng thư viện librosa để chuyển đổi âm thanh về dạng ảnh phổ. Cuối cùng, hai mô hình CNN và FC được sử dụng để thực hiện phân loại bài hát. Kết quả cho thấy việc tăng cường dữ liệu bằng cách chia toàn bộ các bài hát thành các phần nhỏ dựa trên độ dài đã cải thiện đáng kể hiệu suất phân loại so với việc không sử dụng kỹ thuật này. Ngoài ra, CNN có tiềm năng tổng hợp cao hơn và mang lại hiệu suất tốt hơn hơn FC. Những kết quả này có ý nghĩa quan trọng đối với sự phát triển của hệ thống âm nhạc thương mại điện tử.

Từ khóa: Đặc tính âm thanh, Nhạc cụ, Tăng cường dữ liệu, Phân loại bài hát, Đặc trưng âm thanh, Trích xuất đặc trưng, Tìm kiếm bài hát, Ảnh phổ, Mạng nơ ron tích chập.

ABSTRACT

In the rapidly evolving era of information technology and an increasingly modern lifestyle, the need for entertainment becomes more crucial, especially in the realm of music appreciation. Searching for a song is essential, and alongside that, song copyright is also of interest. Therefore, identifying a song from a short music clip is necessary and beneficial, proposed to assist listeners in finding instrumental music. This study proposes a method for classifying and identifying songs based on specific features learned by the model from music data. Python programming language are employed to build a Convolutional Neural Network (CNN) model and Fully Connected Layer (FC). Pre-processing involves using supporting libraries to extract audio data. Dataset A consists of 100 instrumental songs, while Dataset B includes 100 audio files with the same song titles played on various musical instruments. The pydub library is then used to randomly cut the original audio files into clips lasting less than 10 seconds, as users often use a specific segment to identify the entire piece of music.The original audio files are divided into segments of varying lengths during training, including 1 second, 3 seconds, 5 seconds, 10 seconds, 20 seconds, 30 seconds, 60 seconds and 90 seconds. Next, the method of converting audio files into spectrogram images is implemented using the librosa library to transform sound into spectrogram images. Finally, CNN and FC are employed to classify the songs. The results show that data augmentation by dividing entire songs into smaller parts based on duration significantly improved the classification performance compared to not using this technique. Additionally, CNN demonstrates higher synthesis potential and better performance than FC. These results hold significant implications for the development of electronic music commerce systems.

Keywords: Acoustic characteristics, Instrumental music, Data augmentation, Song classification, Audio Features, Feature Extraction, Song Search, Spectrogram, Convolutional Neural Network .

MỤC LỤC

DANH MỤC HÌNH

DANH MỤC BẢNG

DANH MỤC TỪ CHUYÊN NGÀNH

Viết tắt	Diễn giải	Nghĩa Tiếng Việt
ACC	Accuracy	Đánh giá độ chính xác
AMT	Automatic Music Transcription	Chuyển đổi âm nhạc tự động
ANN	Approximate Nearest Neighbor	Thuật toán tìm kiếm lân cận gần nhất
API	Application Programming Interface	Giao diện lập trình ứng dụng
BG	Background Noise	Tiếng ồn nền
CNN	Convolutional Neural Network	Mạng nơ ron tích chập
CONV	Convolutional	Tích chập
CPU	Central Processing Unit	Ngôn ngữ đánh dấu siêu văn bản
CSS	Cascading Style Sheets	Tập tin định kiểu theo tầng
DRC	Dynamic Range Compression	Nén phạm vi động
FC	Fully Connected Layer	Lớp kết nối đầy đủ
GoogleNet	Google Inception Network	Mạng khởi đầu của Google
HTML	HyperText Markup Language	Ngôn ngữ đánh dấu siêu văn bản
MATLAB	MATrix LABoratory	Thí nghiệm ma trận
MEC	Music emotion classification	Phân loại cảm xúc âm nhạc
MFCC	Mel Frequency Cepstral Coefficients	Hệ số cepstral tần số mel
MIR	Music Information Retrieval	Trích xuất thông tin âm nhạc
ML	Machine Learning	Máy học
MRP	Multiresolution Recurrence Plots	Đồ thị tái phát đa độ phân giải
PiczakCNN	PiczakCNN	Mạng nơ ron tích chập Piczak
PS1-PS2	Pitch Shifting	Thay đổi tần số
RANSAC	Random Sample Consensus	Đồng thuận mẫu ngẫu nhiên
ReLU	Rectified Linear Unit	Đơn vị tuyến tính được chỉnh lại
SB-CNN	Spectral Bandwidth Convolutional Neural Network	Mạng nơ ron tích chập bằng thông phổ
Signal	Signal Augmentation	Tăng cường tín hiệu
SKM	Spectral Kernel Mapping	Ánh xạ nhân phổ
Spectro	Spectrogram Augmentation	Tăng cường phổ âm
Sr	Sample rate	Tần số mẫu
StandardIMG	Standard Image Augmentation	Tăng cường hình ảnh tiêu chuẩn
StandardSGN	Standard Signal Augmentation	Tiêu chuẩn tăng cường tín hiệu
SVM	Support Vector Machines	Máy vector hỗ trợ
TS	Time Stretching	Kéo dài thời gian
UIOWA MIS	University of Iowa Musical Instrument Samples	Mẫu nhạc cụ Đại học Iowa
VGG	Visual Geometry Group	Nhóm hình học trực quan
VGGNet	Very Deep Convolutional Networks	Mạng nơ ron tích chập rất sâu

LỜI CAM ĐOAN

Tôi xin cam đoan luận văn này được hoàn thành dựa trên kết quả nghiên cứu của tôi và các kết quả này là trung thực, không sao chép lấy từ bất kỳ một nguồn nào và dưới bất kỳ hình thức nào. Việc tham khảo các nguồn tài liệu đã được thực hiện trích dẫn và ghi nguồn tài liệu tham khảo đúng quy định.

Tôi hoàn toàn chịu trách nhiệm về kết quả, các số liệu được trình bày trong luận văn này.

Cần Thơ, ngày… tháng 12 năm 2023

Sinh viên thực hiện

Lê Tuấn Anh

CHƯƠNG 1. GIỚI THIỆU

1.1. Đặt vấn đề

Trong thế giới ngày nay, khi công nghệ ngày càng phát triển và internet trở thành một phần không thể thiếu trong cuộc sống hằng ngày, việc quản lý thông tin và tìm kiếm nhanh chóng trở thành một thách thức đối với người dùng. Trong lĩnh vực âm nhạc, đặc biệt là khi muốn xác định một bài hát dựa trên một đoạn nhạc ngắn, việc tìm kiếm bài hát là điều cần thiết, nơi có bản quyền của bài hát một mối quan tâm đáng kể. Trải qua những trải nghiệm cá nhân, tôi nhận thức được vấn đề này khi muốn tìm hiểu về một bản nhạc không lời thông qua một đoạn âm thanh tình cờ nghe được. Ngày nay, mỗi người dùng âm nhạc đều đã trải qua cảm giác tò mò và khám phá khi nghe một đoạn nhạc gây ấn tượng, nhưng việc tìm hiểu thông tin chi tiết về bài hát đó thường là một công việc khó khăn. Đặc biệt, khi đối mặt với những bản nhạc không lời hoặc không có lời ca để tra cứu, việc tìm kiếm trở nên thách thức hơn. Việc này thường đòi hỏi người nghe phải tiến hành tìm kiếm một cách thủ công trên internet, đọc các bình luận hoặc sử dụng các ứng dụng tìm kiếm âm nhạc, điều này có thể mất nhiều thời gian và công sức. Tuy nhiên, tôi cũng nhận ra rằng việc tìm kiếm tên bài hát không phải là một công việc đơn giản. Tôi đã phải đọc rất nhiều thông tin và tìm kiếm trên nhiều trang web để tìm được đáp án. Từ trải nghiệm này với sự tò mò và năng động của mình nên đề tài "***Tìm kiếm nhạc không lời sử dụng các phương pháp trực quan phổ tần số và mạng nơ-ron tích chập***" được đề xuất để giúp mọi người có thể tìm kiếm bài hát mà mình muốn nghe chỉ với 1 đoạn nhạc của bài hát không lời. Từ đó giúp cho việc tìm kiếm và nghe nhạc của mọi người sẽ trở nên dễ dàng hơn.

1.2. Các nghiên cứu liên quan

Nhạc cụ truyền thống mang đến không chỉ những giai điệu đáng nhớ mà còn thể hiện sự đa dạng và rõ nét trong việc tạo ra âm thanh. Tuy nhiên, đôi khi tôi vô tình nghe được những giai điệu này mà không biết tên bài hát hay những thông tin liên quan. Việc tìm kiếm và xác định những dấu vết truyền thống này đòi hỏi rất nhiều thời gian và công sức. Vì vậy, nhu cầu nghiên cứu và phát triển hệ thống nhận dạng nhạc cụ truyền thống dựa trên ứng dụng máy học (ML) ngày càng trở nên cần thiết. Điều này thu hút rất nhiều sự ủng hộ và quan tâm từ những người đam mê âm nhạc, mong muốn thấy hoạt động nghiên cứu, phát triển trong lĩnh vực này đạt được những thành tựu nổi bật, góp phần bảo tồn và phát huy giá trị âm nhạc của các nhạc cụ truyền thống trong xã hội hiện đại. Sau đây là một số nghiên cứu có thể kể đến như sau:

Nghiên cứu [1] đã đóng góp quan trọng cho nghiên cứu điển hình về công cụ tìm kiếm. Nghiên cứu này đã sử dụng Approximate Nearest Neighbours để xử lý trước các bài hát nhạc cụ và trích xuất các đặc điểm của các bản nhạc trong kho lưu trữ bằng cách sử dụng trích xuất tính năng Mel Frequency Cepstral Coefficients (MFCC). Phương pháp số hóa đường dẫn, trích xuất các đặc điểm của đường dẫn và xây dựng cây chỉ mục với độ dài khác nhau của mỗi MFCC và số chiều của vectơ được thu thập các bài hát được chơi bằng nhiều loại nhạc cụ để thử nghiệm. Nghiên cứu đã cho kết quả trên 100 track có độ dài khác

nhau, với tần số lấy mẫu là 16000 và độ dài mỗi MFCC là 13, cho kết quả tốt nhất, trong đó độ chính xác trên Top 1 là 36%, Top 5 là 4% và Top 10 là 44%. Bên cạnh đó, nghiên cứu áp dụng thuật toán ANN và trích xuất đặc trưng MFCC để thực hiện tìm kiếm bài hát theo hòa âm, giúp lựa chọn thông số phù hợp nhất để cho kết quả chính xác nhất và thời gian tìm kiếm nhanh nhất. Ngoài ra, xây dựng website tích hợp thuật toán giúp người dùng tìm kiếm bài hát dễ dàng hơn.

Nghiên cứu [2] là một công trình thiết yếu trong lĩnh vực nhận dạng âm thanh môi trường bằng mô hình mạng nơ-ron sâu, đặc biệt là CNN. Bài viết này đánh giá tiềm năng của mạng nơ ron tích chập trong việc phân loại các đoạn âm thanh ngắn về âm thanh môi trường. Một mô hình sâu bao gồm 2 lớp tích chập với lớp pooling tối đa và hai lớp Fully Connected được huấn luyện về cách biểu diễn dữ liệu âm thanh ở mức độ thấp (segmented spectrogram) với delta. Độ chính xác của mạng được đánh giá dựa trên ba bộ dữ liệu có sẵn công khai về ghi âm đô thị và môi trường. Mô hình này vượt trội hơn so với các phiên bản cở bản dựa trên hệ số cestral tần số mel và đạt được kết quả tương đương với các phương pháp tiếp cận tiên tiến khác.

Nghiên cứu [3] rất cần thiết cho việc trích xuất thông tin âm nhạc và nhận dạng nốt nhạc tự động. Nghiên cứu này tập trung vào việc chuyển đổi tín hiệu âm thanh thành biểu diễn tần số theo thời gian (quang phổ) và sử dụng mô hình học máy để dự đoán các nốt nhạc và nhạc cụ tương ứng. Hệ thống tự động trích xuất thông tin từ tín hiệu âm thanh như tên nốt nhạc và nhạc cụ cụ thể trong nhạc đa âm. Họ sử dụng các tính năng như cao độ và tần số cơ bản để mô tả các đặc điểm âm thanh trong biểu đồ phổ. Nghiên cứu này đã sử dụng các mô hình học máy như Support Vector Machines (SVM) để phân loại các nốt nhạc và nhạc cụ trong âm nhạc. Các mô hình đã được đào tạo trên các tập dữ liệu âm nhạc được chú thích bằng các nốt nhạc. Các tác giả đã tiến hành thử nghiệm trên bộ dữ liệu âm nhạc thực tế và so sánh kết quả dự đoán với thông tin nhãn thực tế. Kết quả cho thấy tính khả thi và tiềm năng của phương pháp trong việc hiện thực hóa các nốt nhạc, nhạc cụ từ tín hiệu âm thanh. Điều này hướng tới hướng phát triển Automatic Music Transcription (AMT) bằng cách sử dụng các mô hình học máy, mở ra con đường tự động hóa quá trình xử lý các bản ghi âm nhạc truyền thống. Những phát hiện này có ý nghĩa quan trọng đối với việc bảo tồn và phân tích âm nhạc cổ điển.

Nghiên cứu [4] đề xuất một sơ đồ phân biệt tín hiệu âm nhạc bài hát và nhạc cụ. Sử dụng các đặc điểm tương tự dựa trên hình ảnh phổ để phân loại tín hiệu âm thanh. Nghiên cứu đã quan sát thấy rằng hình ảnh quang phổ của tín hiệu nhạc cụ cho thấy các đỉnh ổn định hơn theo thời gian và điều này không phải là như vậy đối với một bài hát. Nó đã thúc đẩy nghiên cứu tìm kiếm các tính năng dựa trên hình ảnh phổ. Các đặc điểm theo ngữ cảnh đã được tính toán dựa trên mẫu xuất hiện của tần số quan trọng nhất theo thang thời gian và mẫu kết cấu tổng thể được tiết lộ bởi sự phân bố tần số thời gian của cường độ tín hiệu. Để phân loại, dựa vào kỹ thuật Random Sample Consensus (RANSAC) có khả năng xử lý nhiều loại dữ liệu trong một lớp. RANSAC tạo các mô hình cho từng lớp và thực hiện sản xuất mô hình phân loại dựa trên đó. Tác giả đã chuẩn bị một cơ sở dữ liệu âm nhạc bao

gồm 300 tệp nhạc cụ và 300 tệp bài hát. Mỗi tệp chứa âm thanh có thời lượng khoảng 40-45 giây. Tệp được lấy từ việc ghi đĩa CD, ghi các chương trình trực tiếp và tải xuống từ nhiều trang khác nhau trên Internet. Lấy mẫu dữ liệu tần số đã cho là 22050 Hz. Các mẫu có 16 bit và đơn sắc. Dữ liệu của các nhạc cụ khác nhau như sáo, guitar piano, trống đều có sẵn được lưu. Các bài hát cũng có nhiều thể loại như cổ điển, jazz, rock, Bhangra (cũng là thể loại Bắc Ấn Độ). Để tính toán các đặc tính, các tệp Âm thanh được chia thành các khung. Tương ứng với từng khung hình thường thu được. Mỗi khung bao gồm 256 mẫu trong đó có 128 mẫu chéo giữa hai khung liên tiếp. Kết quả đạt được: Neural Network: nhạc cụ 85%, Bài hát (80,33%), Tổng thể (82,67%), RANSAC: nhạc cụ (97%), Bài hát (93%), Tổng thể (95%). Trong nghiên cứu này, kiểm tra hiệu suất của RANSAC trên tập tính năng được tính toán. Hiệu suất Neural Network trên cùng một bộ tính năng và nhận thấy hiệu suất của RANSAC tốt hơn nhiều so với Neural Network.

Nghiên cứu [5] đề xuất phân loại âm thanh động vật tự động, khai thác các kĩ năng tăng cường dữ liệu khác nhau để đào tạo CNN. Nghiên cứu thử nghiệm và kết hợp hai kiến trúc CNN khác nhau: 1.GoogleNet: Cấu trúc này bao gồm 22 layer và 5 layer Pool, 2.VGGNet: Bao gồm 16 layer CONV/FC (VGG-16) hoặc 19 (VGG-19) các lớp CONV cực kỳ đồng nhất và sử dụng các bộ lọc tích chập rất nhỏ (3x3) với lớp POOL sau mỗi lần hai hoặc ba lớp CONV (thay vào sau mỗi lớp CONV). Bên cạnh đó, nghiên cứu đã thực nghiệm bốn giao thức tăng cường, hoạt động trên các tín hiệu âm thanh thô hoặc biểu diễn của chúng dưới dạng biểu đồ phổ: Standard Image Augmentation (StandardIMG), Standard Signal Augmentation (StandardSGN), Spectrogram Augmentation (Spectro), Signal Augmentation (Signal). Đối với Spectro và Signal nghiên cứu đã sử dụng các phương pháp cung cấp Audiogmenter một thư viện tăng cường dữ liệu âm thanh cho MATLAB. Nghiên cứu so sánh các phương pháp tiếp cận tốt nhất của mình với phương pháp hiện đại nhất, cho thấy rằng nghiên cứu đạt được tỷ lệ nhận dạng tốt nhất trên cùng một bộ dữ liệu mà không cần tối ưu hóa tham số đặc biệt. Bên cạnh đó, cho thấy rằng các CNN khác nhau có thể được đào tạo cho mục đích phân loại âm thanh động vật và sự kết hợp của chúng hoạt động tốt hơn so với các bộ phân loại độc lập. Nghiên cứu cũng muốn cảnh báo về việc sử dụng các giao thức tăng cường dữ liệu tiêu chuẩn: Các kỹ thuật tăng cường được phát triển riêng cho hình ảnh có thể vô dụng hoặc trong trường hợp xấu nhất là gây bất lợi khi phân loại tập dữ liệu cụ thể. Kết quả của nghiên cứu cho thấy StandardIMG hoạt động kém hơn so với phân loại không có tăng cường. Vì vậy, khi lựa chọn tăng cường dữ liệu cho một vấn đề phân loại, bản chất của tập dữ liệu để phân loại phải luôn được xem xét. Nghiên cứu này được coi là nghiên cứu lớn nhất về tăng cường dữ liệu cho CNN ở phân loại âm thanh động vật tập dữ liệu âm thanh sử dụng cùng một bộ phân loại và tham số.

Một trong những nghiên cứu đáng chú ý là nghiên cứu được thực hiện bởi các tác giả của nghiên cứu [6]. Không giống như các phương pháp truyền thống, nghiên cứu khám phá sơ đồ phân loại nhạc cụ bằng cách sử dụng các tính năng học được từ CNN. Tuy nhiên, một phương pháp khác để cung cấp thông tin pha vào mạng nơ ron đã được đề xuất trong nghiên cứu này. Vì chuỗi thời gian là chuỗi dữ liệu một chiều nên bộ lọc (kernel) được sử

dụng cho chuỗi thời gian trong CNN cũng phải là một chiều. Hạn chế này đã dẫn đến một hạn chế khi áp dụng tích chập bộ lọc vì thông tin ở một vị trí thời gian cụ thể chỉ có thể được tổng hợp một lần. Ngược lại, hình ảnh phổ hai chiều có thể được phân tích bằng bộ lọc hai chiều nhiều lần trên một vị trí tạm thời, cung cấp nhiều chiều hơn để phân tích. Sáng kiến nghiên cứu nhằm nâng cao hiệu suất của bộ phân loại thông qua các kỹ thuật tăng cường dữ liệu. Cụ thể, các nhà nghiên cứu đã dịch chuyển khối thời gian theo 13 mẫu, tạo ra 13 bộ dữ liệu Multiresolution Recurrence Plots (MRP) được dịch chuyển theo thời gian. Các bộ dữ liệu này sau đó được sử dụng để xây dựng bảy lớp hình ảnh RP, với mỗi lớp có độ dài gấp bốn lần so với khối trước đó trong chuỗi thời gian. Kích thước tạm thời của các lớp này được đặt ở mức 25, 27, 29, 211, 213, 215 và 217. Bằng cách kết hợp các MRP và hình ảnh phổ được đề xuất với mạng nhiều cột, họ đã thu được hiệu suất phân loại công cụ được cải thiện bằng UIOWA MIS cơ sở dữ liệu. Nghiên cứu cũng kiểm tra hiệu suất của mạng kết hợp cho nhiệm vụ phân loại đàn piano. Kết quả cho thấy việc phân loại âm sắc của nhạc cụ có thể được cải thiện bằng cách sử dụng dữ liệu MRP với hình ảnh phổ và bằng cách đưa dữ liệu và hình ảnh vào CNN nhiều cột.

Nghiên cứu này [7] có hai đóng góp chính: thứ nhất, nó đề xuất kiến trúc mạng nơ-ron tích chập sâu để phân loại âm thanh môi trường - mô hình SB-CNN, PiczakCNN, SKM. Thứ hai, nó đề xuất sử dụng tính năng tăng cường dữ liệu âm thanh để khắc phục vấn đề khan hiếm dữ liệu và khám phá ảnh hưởng của các mức tăng cường khác nhau đối với hiệu suất của kiến trúc CNN được đề xuất. Kiến trúc mạng nơ ron tích chập sâu (CNN) được đề xuất trong nghiên cứu này bao gồm ba lớp tích chập xen kẽ: 24, 48 và 48 filter với hai lớp pooling, theo sau là hai lớp fully connected (dense). Nghiên cứu đã thử nghiệm bốn cách tăng cường (biến dạng) dữ liệu âm thanh khác nhau: Time Stretching (TS), Pitch Shifting (PS1), Pitch Shifting (PS2), Dynamic Range Compression (DRC) and Background Noise (BG). Mỗi biến dạng được áp dụng trực tiếp vào tín hiệu âm thanh trước khi chuyển đổi nó thành biểu diễn đầu vào để huấn luyện mạng (log-mel-Spectrogram). Kết hợp với việc tăng cường dữ liệu, mô hình đề xuất đã đạt được kết quả tiên tiến về phân loại âm thanh môi trường. Nghiên cứu đã cải thiện hiệu suất bằng cách kết hợp mô hình sâu, dung lượng cao và tập dữ liệu đào tạo được tăng cường. Sự kết hợp này vượt trội hơn CNN được đề xuất mà không cần tăng cường và hình học từ điển mô hình "shallow" có cải tiến. Cuối cùng, nghiên cứu đã kiểm tra ảnh hưởng của từng mức tăng đối với độ chính xác phân loại của mô hình cho từng lớp và nhận thấy rằng mỗi mức tăng ảnh hưởng khác nhau đến độ chính xác của từng lớp, cho thấy hiệu suất của mô hình có thể được cải thiện hơn nữa bằng cách áp dụng tăng cường dữ liệu có điều kiện cho mỗi lớp.

Mặc dù nhiều nghiên cứu đã cố gắng đề xuất các phương pháp dựa trên CNN để thực hiện phát hiện bài hát và tăng cường dữ liệu bằng cách thay đổi vị trí thời gian của dữ liệu âm thanh để tạo ra các biến thể hoặc bản sao dữ liệu mới, nhưng hiệu quả của việc tăng mẫu bằng cách tách các bài hát gốc hoàn chỉnh thành các phần nhỏ hơn được huấn luyện bởi CNN vẫn chưa được làm rõ trên các bài hát nhạc cụ. Ngoài ra, nó cũng bắt nguồn từ việc yêu cầu tên bài hát nhưng thường chỉ có một đoạn nhỏ trong bài hát. Do đó, nghiên

cứu của chúng tôi nhằm mục đích đánh giá lợi ích tiềm năng của cách tiếp cận như vậy trong việc cải thiện nhiệm vụ phân loại bài hát bằng nhạc cụ và hỗ trợ tìm kiếm bài hát chỉ với một đoạn nhỏ của bài hát.

1.3. Mục tiêu đề tài

Mục tiêu chính của đề tài là nghiên cứu áp dụng phương pháp trực quan phổ tần số và Mạng nơ-ron tích chập (CNN) [8] để xây dựng hệ thống tìm kiếm nhạc không lời. Áp dụng các kỹ thuật tăng cường dữ liệu để cải thiện độ chính xác của mô hình được huấn luyện. Quá trình triển khai mô hình được thực hiện bằng ngôn ngữ lập trình Python và sử dụng các thư viện hỗ trợ như NumPy, SciPy, Matplotlib và TensorFlow.

1.4. Đối tượng và phạm vi nghiên cứu

1.4.1. Đối tượng nghiên cứu

Đối tượng nghiên cứu chính của đề tài là nghiên cứu các kiến trúc của mạng nơ ron tích chập.

1.4.2. Phạm vi đề tài

Đề tài nghiên cứu và xây dựng các mô hình máy học thực hiện việc sử dụng các phương pháp trực quan phổ tần số và mạng nơ-ron tích chập xây dựng website tìm kiếm bài hát dựa vào file âm thanh.

1.5. Nội dung đề tài

Đề tài được thực hiện bao gồm các nội dung chính như sau:

– **Thu thập dữ liệu âm thanh**: Bộ dữ liệu âm thanh đang dùng được lấy từ nghiên cứu [1] bằng cách sử dụng cùng một bộ dữ liệu mà các nghiên cứu trước đã sử dụng, nghiên cứu này có thể so sánh kết quả của mình với nghiên cứu trước đó. Điều này giúp xác nhận tính chính xác của kết quả và tăng cường khả năng so sánh của nghiên cứu.

– **Tìm hiểu và nghiên cứu các kiến trúc của mạng nơ ron tích chập để xây dựng hệ thống tìm kiếm nhạc không lời.**

- Tìm hiểu cách sử dụng thư viện Pydub [9] để chia nhỏ tệp âm thanh thành các phần bằng nhau và cách để cắt 1 đoạn ngẫu nhiên từ tệp âm thanh.
- Tìm hiểu cách trích xuất đặc trưng âm thanh thành mảng Numpy.
- Tìm hiểu cách chuyển đổi âm thanh thành dạng ảnh phổ.
- Tìm hiểu thư viện Python được xây dựng bằng TensorFlow và Keras, để xây dựng một mô hình mạng nơ-ron tích chập để học cách phân loại các bản nhạc không lời dựa vào dữ liệu âm thanh được chuyển đổi thành mảng Numpy và ảnh phổ.
- Nghiên cứu các tài liệu, bài báo khoa học, công trình nghiên cứu liên quan đến chủ đề mô hình CNN/FC để phân loại dữ liệu âm thanh từ phổ âm thanh và mảng Numpy.
- Nghiên cứu và áp dụng kỹ thuật tăng cường dữ liệu.
- Tìm hiểu độ đo Accuracy(ACC).

– **Tìm hiểu và sử dụng các công cụ, ngôn ngữ, và thư viện hỗ trợ:**

• Tìm hiểu và sử dụng các công cụ như Visual Studio Code để soạn thảo code.

• Tìm hiểu ngôn ngữ lập trình HTML, CSS, Python, framework và template hỗ trợ xây dựng website.

• Tìm hiểu Flask Framework tạo API cho website.

– **Xây dựng Website Tìm kiếm nhạc không lời sử dụng các phương pháp trực quan phổ tần số và mạng nơ-ron tích chập đã được đào tạo.**

• Tìm hiểu cách tích hợp mô hình đó lên website bằng cách sử dụng thư viện Flask của Python.

1.6. Những đóng góp chính có đề tài

Các kết quả chính đã đạt được:

– Nghiên cứu xây dựng kiến trúc mô hình CNN/FC dùng đánh giá hiệu suất phân loại trên dữ liệu âm thanh được trích xuất đặc trưng thành mảng Numpy.

– Nghiên cứu đã triển khai, đánh giá các mô hình CNN và FC cho nhiệm vụ phân loại bài hát dựa trên phổ âm của đoạn nhạc. Bên cạnh đó chúng tôi đã áp dụng các kỹ thuật tăng cường dữ liệu bằng cách chia bài hát thành nhiều đoạn có thể cải thiện độ chính xác của quá trình phân loại. Thực hiện triển khai mô hình CNN1/FC trên tập dữ liệu được tăng cường để so sánh hiệu suất giữa hai kiến trúc mạng trong nhiệm vụ phân loại bài hát dựa trên dữ liệu âm thanh.

– Nghiên cứu thực hiện so sánh kết quả thử nghiệm liên quan đến việc phân loại bài hát trên các đoạn âm thanh và toàn bộ bản nhạc.

– Nghiên cứu thực hiện xác định tầm quan trọng của việc chọn bài hát thử nghiệm có số giây nhỏ.

– Nghiên cứu thực nghiệm trên các mô hình CNN1, CNN2, CNN3. Bên cạnh đó, chúng tôi đã lược bỏ MaxPooling cho các mô hình, nó là minh chứng cho việc lựa chọn kiến trúc và xử lý mạng tập dữ liệu phù hợp cho bài toán phân loại bài hát.

– Về mặt khoa học, những kết quả trong nghiên cứu này đã được đề xuất viết một bài báo "***An Approach to Instrumental Song Classification Utilizing Spectrogram and Convolutional Neural Networks***" và nhận được sự chấp nhận cho trình bày tại Hội nghị quốc tế về trí tuệ nhân tạo và Trí tuệ tính toán AICI'2024 (Hanoi, Vietnam, January 13-14, 2024), đồng thời sẽ được xuất bản bởi Springer.

1.7. Bố cục của luận văn

Bố cục của luận văn gồm 5 phần chính như sau:

1. Giới Thiệu

Bao gồm các nội dung chi tiết sau:

– Đặt vấn đề

– Các nghiên cứu liên quan

– Mục tiêu đề tài

- Đối tượng và phạm vi đề tài
- Nội dung đề tài
- Những đóng góp chính của đề tài
- Bố cục luận văn

2. Mô tả bài toán

Bao gồm các nội dung chi tiết sau:

- Mô tả chi tiết bài toán
- Hướng tiếp cận giải quyết của đề tài

3. Thiết kế và cài đặt giải pháp

Bao gồm các nội dung chi tiết sau:

- Kiến trúc tổng quát hệ thống
- Xây dựng các mô hình
- Giải pháp cài đặt

4. Kiểm thử và đánh giá

Bao gồm các nội dung chi tiết sau:

- Kịch bản kiểm thử
- Kết quả kiểm thử

5. Kết luận và hướng phát triển

Bao gồm các nội dung chi tiết sau:

- Kết quả đạt được
- Hạn chế
- Hướng phát triển

1.8. Tổng kết chương

Trên đây là phần giới thiệu của đề tài nhằm giúp người đọc hiểu về đề tài, tìm hiểu được các nghiên cứu liên quan, đối tượng, nội dung, những đóng góp chính và bố cục của luận văn. Phần tiếp theo xin trình bày về bài toán và lựa chọn, đánh giá các giải pháp.

CHƯƠNG 2. MÔ TẢ BÀI TOÁN

2.1 Mô tả chi tiết bài toán

Nghiên cứu với mục tiêu xây dựng được các mô hình máy học và tích hợp chúng lên một website và sử dụng các giải thuật tìm kiếm để hỗ trợ người dùng có thể tìm ra bài hát mong muốn một cách dễ dàng chỉ với một đoạn nhạc. Để thực hiện mục tiêu trên, nghiên cứu đã xây dựng và huấn luyện các mô hình CNN và FC phân lớp.

Để thực hiện mục tiêu trên:

- Thu thập tập dữ liệu âm thanh.
- Tìm hiểu cách sử dụng thư viện Pydub để chia nhỏ tệp âm thanh.
- Tìm hiểu các thư viện liên quan đến xử lý âm thanh.
- Chuyển đổi dữ liệu âm thanh thành mảng Numpy.
- Chuyển đổi dữ liệu âm thanh sang dạng ảnh phổ.
- Nghiên cứu xây dựng và huấn luyện các mô hình phân lớp.
- Tìm hiểu độ đo Accuracy
- Nghiên cứu các kỹ thuật tăng cường dữ liệu để cải thiện độ chính xác của mô hình được huấn luyện.
- Tìm hiểu môi trường cài đặt.
- Tích hợp giải thuật lên website.

2.2 Hướng tiếp cận giải quyết của đề tài

2.2.1 Hướng tiếp cận 1: Phân lớp trên dữ liệu âm thanh chuyển đổi thành mảng Numpy

Nghiên cứu xây dựng kiến trúc mô hình CNN/FC dùng đánh giá hiệu suất phân loại trên dữ liệu âm thanh được chuyển đổi thành mảng Numpy. Để trích xuất đặc trưng từ dữ liệu âm thanh thành mảng Numpy, chúng tôi sử dụng các thư viện như Librosa trong ngôn ngữ lập trình Python. Quy trình bắt đầu bằng việc đọc dữ liệu từ tập dữ liệu âm thanh, sau đó tiến hành xử lý thông qua việc trích xuất đặc trưng dữ liệu âm thanh được chuyển đổi thành mảng Numpy. Tiếp theo, chúng tôi tiếp tục xây dựng kiến trúc mô hình CNN/FC.

Dữ liệu đầu vào được xác định bằng tham số input_shape=args.lengthaudio, thể hiện số lượng samples đã được chuyển đổi mảng Numpy. Tiếp đến là chọn đối số, áp dụng các bộ lọc 64 filter. Sau khi lớp tích chập được áp dụng, bước tiếp theo là sử dụng lớp MaxPooling để giảm kích thước của dữ liệu. Tiếp theo, chúng tôi thêm một lớp Dense vào mô hình với số đơn vị (units) là i_label và activation là softmax. Lớp Dense này đại diện cho mô hình, có nhiệm vụ phân loại bài hát dựa trên đặc trưng số hóa từ âm thanh. Cuối cùng, mô hình được biên soạn (compile) với hàm mất mát là categorical_crossentropy và tối ưu hóa bằng thuật toán Adam. Với các siêu tham số được sử dụng trong hướng tiếp cận 1: learning rate là 0.0001, batch size là 64, input_shape=args.lengthaudio, số lượng epochs (500), Numclass(100). Sau đó sẽ dựa trên kết quả huấn luyện mà chọn ra model tốt nhất.

2.2.2 Hướng tiếp cận 2: Phân lớp trên dữ liệu âm thanh chuyển đổi thành ảnh phổ

Trong phần này, chiến lược đề xuất phương pháp phân loại bài hát dựa trên các đặc điểm cụ thể mà mô hình học được từ dữ liệu âm nhạc. Ngôn ngữ lập trình Python được sử dụng để xây dựng mô hình CNN/FC, chuyển đổi bài toán qua nhận dạng hình ảnh với những giải thuật nhận dạng ảnh. Chúng tôi sẽ dùng các thư viện Librosa, Numpy, Keras, Matplotlib,… để chuyển các bài hát thành dạng ảnh phổ - một biểu đồ biểu thị tần số và thời gian để tiện xử lý với CNN/FC.

Với đầu vào của lớp tích chập là hình ảnh. Tiếp đến là chọn đối số, áp dụng các bộ lọc với các bước nhảy. Thực hiện tích chập cho hình ảnh và áp dụng hàm kích hoạt Rectified Linear Unit (ReLU) cho ma trận hình ảnh. Thực hiện Pooling để giảm kích thước cho hình ảnh thêm nhiều lớp tích chập sao cho phù hợp. Xây dựng đầu ra và dữ liệu đầu vào thành 1 lớp được kết nối đầy đủ. Sử dụng hàm kích hoạt để tìm đối số phù hợp để phân loại và đưa ra kết qua phù hợp. Với các siêu tham số learning rate là 0.0001, batch size là 64, số lượng epochs (500), Batch_Size(64), Image_Len(32), Numclass(100), optimizer là Adam. Sau đó sẽ dựa trên kết quả huấn luyện mà chọn ra model tốt nhất.

2.3 Tổng kết chương

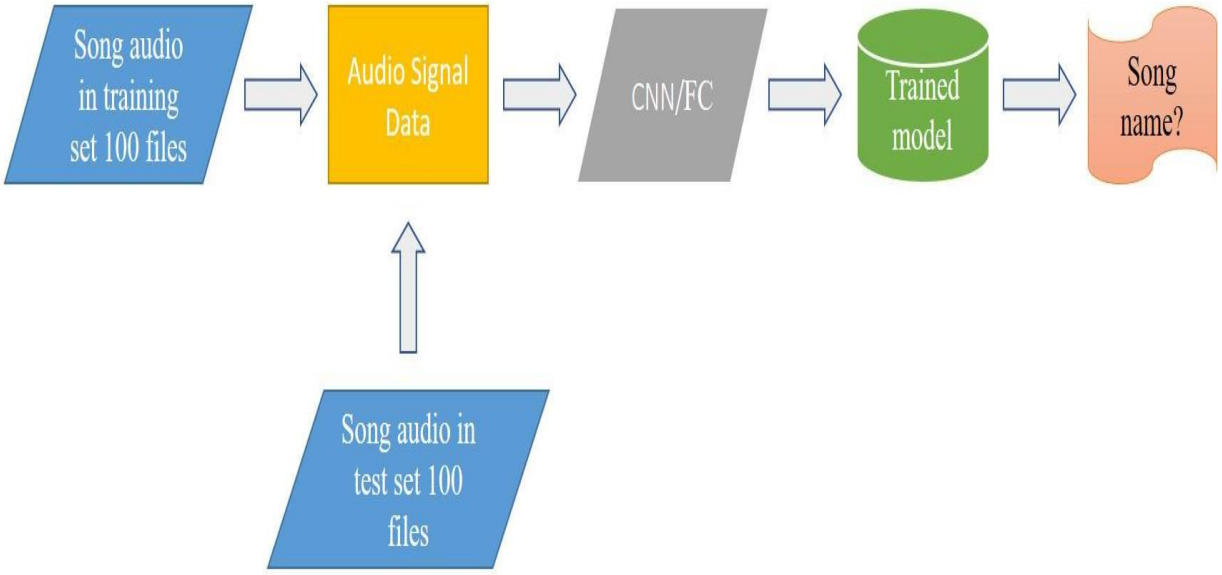
Trên đây là mô tả chi tiết, hướng tiếp cận và giải pháp dùng để thực hiện đề tài "***Tìm kiếm nhạc không lời sử dụng các phương pháp trực quan phổ tần số và Mạng nơ-ron tích chập***". Chi tiết về giải pháp sẽ được trình bày trong phần sau.

CHƯƠNG 3. THIẾT KẾ VÀ CÀI ĐẶT GIẢI PHÁP

3.1. Cách tiếp cận 1: Phân lớp trên dữ liệu âm thanh chuyển đổi thành mảng Numpy

Trong phần này trình bày các kĩ thuật chính trong xử lý tín hiệu âm thanh, biểu diễn tín hiệu thành mảng Numpy. Đầu tiên dữ liệu đầu vào của mô hình được đọc từ tập dữ liệu âm thanh, sau đó thực hiện xử lí dữ liệu bằng cách trích xuất đặc trưng. Tiếp theo, xây dựng kiến trúc mô hình CNN/FC với mục đích đánh giá hiệu suất trên dữ liệu âm thanh được trích xuất đặc trưng thành mảng Numpy.

Trong quy trình đầu tiên, tập dữ liệu A bao gồm 100 bài hát không lời, trong khi tập dữ liệu B bao gồm 100 tệp âm thanh có cùng tên bài hát nhưng được chơi bằng các loại nhạc cụ khác nhau. Sau đó, thư viện pydub cắt ngẫu nhiên các file âm thanh gốc thành các clip có độ dài dưới 10 giây trong tập kiểm thử vì người dùng thường sử dụng một đoạn mã cụ thể để tìm bản nhạc. Các tệp âm thanh gốc được chia thành các đoạn âm thanh có độ dài khác nhau trong tập huấn luyện bao gồm 5 giây, 10 giây. Tiếp theo, sử dụng thư viện librosa của python để chuyển đổi tín hiệu âm thanh thành mảng Numpy. Tiếp theo, mô hình CNN/FC được xây dựng sau khi có được trích xuất đặc trưng dữ liệu âm thanh. Kiến trúc mạng thần kinh này được thiết kế đặc biệt để hoạt động với dữ liệu 1D. Quy trình làm việc được trình bày trong **Hình 3. 1.**



Hình 3. 1: Kiến trúc tổng quan của hệ thống tiếp cận 1

3.1.1. Dữ liệu tín hiệu âm thanh

Trích xuất đặc trưng tệp âm thanh thành dữ liệu 1D là một quy trình quan trọng trong quá trình huấn luyện mô hình, cho phép biểu diễn tín hiệu thành mảng Numpy. (Minh họa **Hình 3. 2**).

```python
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
import sklearn
import librosa
import librosa.display
import warnings
warnings.filterwarnings('ignore')
import os
general_path = 'D:\\NLN\\audioData\\data\\wav\\test_train_random10s'
for subdir, _, files in os.walk(general_path):
    for file in files:
        if file.endswith('.wav'):
            parent_folder_name = os.path.basename(subdir)
            file_path = os.path.join(subdir, file)
            y, sr = librosa.load(file_path)
            audio_file, _ = librosa.effects.trim(y)
            output_dir = os.path.join('D:\\NLN\\audioData\\data\\test_train_random10s', parent_folder_name)
            os.makedirs(output_dir, exist_ok=True)
            output_text_path = os.path.join(output_dir, f'{os.path.splitext(file)[0]}.txt')
            np.savetxt(output_text_path, audio_file, fmt='%f', delimiter='\n')
```

Hình 3. 2: Chuyển đổi tệp âm thanh thành mảng NumPy

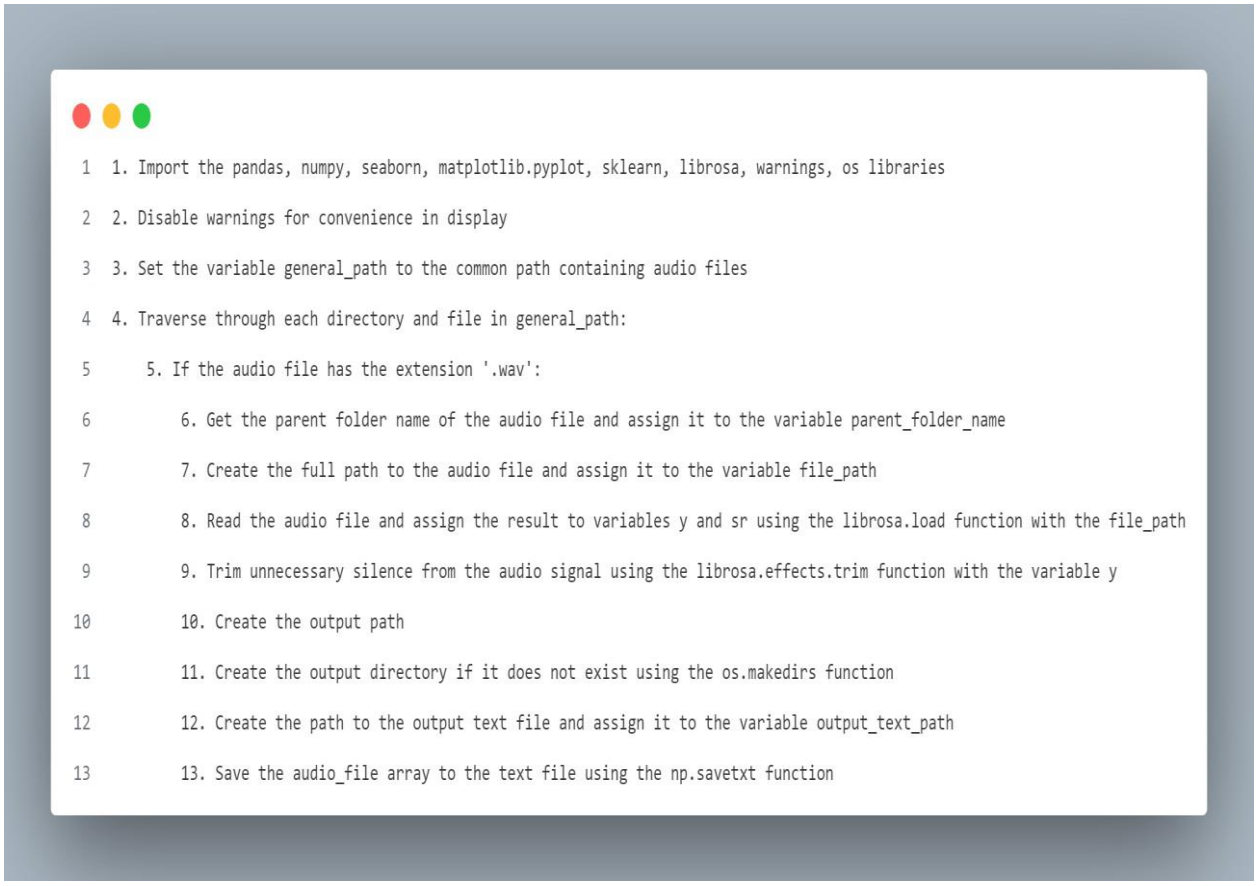
Trong nghiên cứu này sử dụng thư viện Librosa để lấy mẫu đoạn nhạc từ tệp âm thanh và chuyển đổi nó thành mảng Numpy (y) và tần số lấy mẫu (sr) (minh họa **Hình 3. 2**). Quá trình này để thuận tiện cho việc xử lý và phân tích tín hiệu âm thanh trong các bước tiếp theo của quy trình.

- **Biến "y" :** Mảng Numpy.
- **Biến "sr" :** Tần số lấy mẫu (Sample rate) đây là số lần mà một hệ thống lấy mẫu tín hiệu trong một khoảng thời gian nhất định, đo lường bằng Hz (Hertz) hoặc kHz (Kilohertz). Trong đó, Hz: là đơn vị đo tần số, thể hiện số lần xảy ra trong một giây. KHz (Kilohertz): Là đơn vị lớn hơn, đại diện cho 1,000 Hz. Do đó, 1 KHz = 1000 Hz.

Ví dụ: Nếu sample rate là 22050Hz, điều này có nghĩa là hệ thống đang lấy mẫu tín hiệu 22,050 lần mỗi giây. Sample rate thường được biểu diễn ở dạng KHz để giảm số lượng

chữ số và làm cho nó dễ đọc hơn. Ví dụ: 22.05 KHz có nghĩa là 22,050 Hz.

Bên cạnh đó, chúng tôi trình bày Mã giả (Pseudocode) tương ứng với việc chuyển đổi tệp âm thanh thành mảng Numpy (Minh họa **Hình 3. 3**), giúp chúng tôi diễn đạt ý tưởng một cách dễ dàng và dễ hiểu.



Hình 3. 3: Mã giả tương ứng Chuyển đổi âm thanh thành mảng Numpy

3.1.2. Phân lớp trên dữ liệu âm thanh chuyển đổi thành mảng Numpy

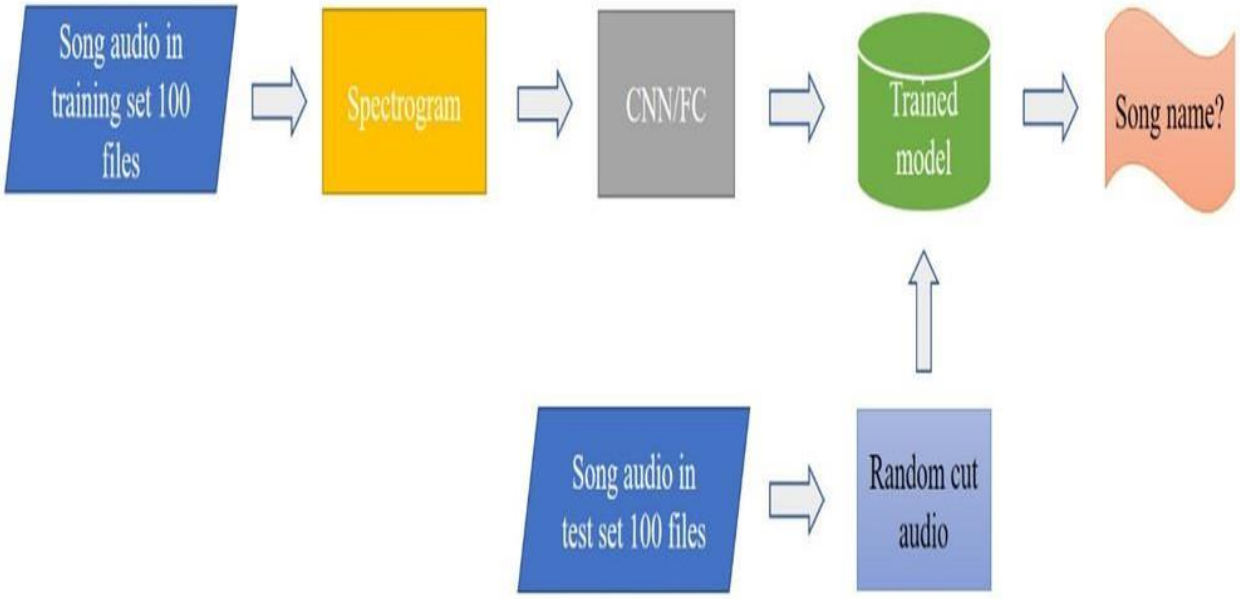
Nghiên cứu xây dựng kiến trúc mô hình CNN/FC dùng đánh giá hiệu suất phân loại trên dữ liệu âm thanh được chuyển đổi thành mảng Numpy. Dữ liệu đầu vào được xác định bằng tham số input_shape=args.lengthaudio, thể hiện số lượng samples đã được chuyển đổi mảng Numpy. Tiếp đến là chọn đối số, áp dụng các bộ lọc 64 filter. Sau khi lớp tích chập được áp dụng, bước tiếp theo là sử dụng lớp MaxPooling để giảm kích thước của dữ liệu. Tiếp theo, chúng tôi thêm một lớp Dense vào mô hình với số đơn vị (units) là i_label và activation là softmax. Lớp Dense này đại diện cho mô hình, có nhiệm vụ phân loại bài hát dựa trên đặc trưng số hóa từ âm thanh. Cuối cùng, mô hình được biên soạn (compile) với hàm mất mát là categorical_crossentropy và tối ưu hóa bằng thuật toán Adam. Với các siêu tham số được sử dụng trong nghiên cứu: learning rate là 0.0001, batch size là 64, input_shape=args.lengthaudio, số lượng epochs (500), Numclass(100). Sau đó sẽ dựa trên kết quả huấn luyện mà chọn ra model tốt nhất.

3.2. Cách tiếp cận 2: Phân lớp trên dữ liệu âm thanh chuyển đổi thành ảnh phổ

Trong phần này trình bày các kĩ thuật chính trong xử lý tín hiệu âm thanh, biểu diễn tín hiệu thành ảnh phổ. Đầu tiên dữ liệu đầu vào của mô hình được đọc từ tập dữ liệu âm thanh, sau đó thực hiện xử lí dữ liệu bằng cách trích xuất đặc trưng. Tiếp theo, xây dựng kiến trúc mô hình CNN/FC với mục đích đánh giá hiệu suất trên dữ liệu âm thanh được chuyển đổi thành ảnh phổ.

Trong quy trình đầu tiên, tập dữ liệu A bao gồm 100 bài hát không lời, trong khi tập dữ liệu B bao gồm 100 tệp âm thanh có cùng tên bài hát nhưng được chơi bằng các loại nhạc cụ khác nhau. Sau đó, thư viện pydub cắt ngẫu nhiên các file âm thanh gốc thành các clip có độ dài dưới 10 giây trong tập kiểm thử vì người dùng thường sử dụng một đoạn mã cụ thể để tìm bản nhạc. Các tệp âm thanh gốc được chia thành các đoạn âm thanh có độ dài khác nhau trong tập huấn luyện, bao gồm 1 giây, 3 giây, 5 giây, 10 giây, 20 giây, 30 giây, 60 giây và 90 giây. Tiếp theo, sử dụng thư viện Librosa của python để chuyển đổi âm thanh về dạng ảnh phổ với những vị trí màu sắc tươi sáng thể hiện tần số cao (Minh họa **Hình 3. 7**). Tiếp theo, mô hình CNN/FC được xây dựng sau khi có được phổ dữ liệu đặc trưng ảnh quang phổ.

Kiến trúc mạng thần kinh này được thiết kế đặc biệt để hoạt động với dữ liệu 2D. Ngoài ra, việc sử dụng mô hình FC giúp chúng tôi so sánh được khả năng tổng hợp và cách thức hoạt động trong giai đoạn kiểm thử. Quy trình làm việc được trình bày trong **Hình 3. 4** .



Hình 3. 4: Kiến trúc tổng quan của hệ thống tiếp cận 2

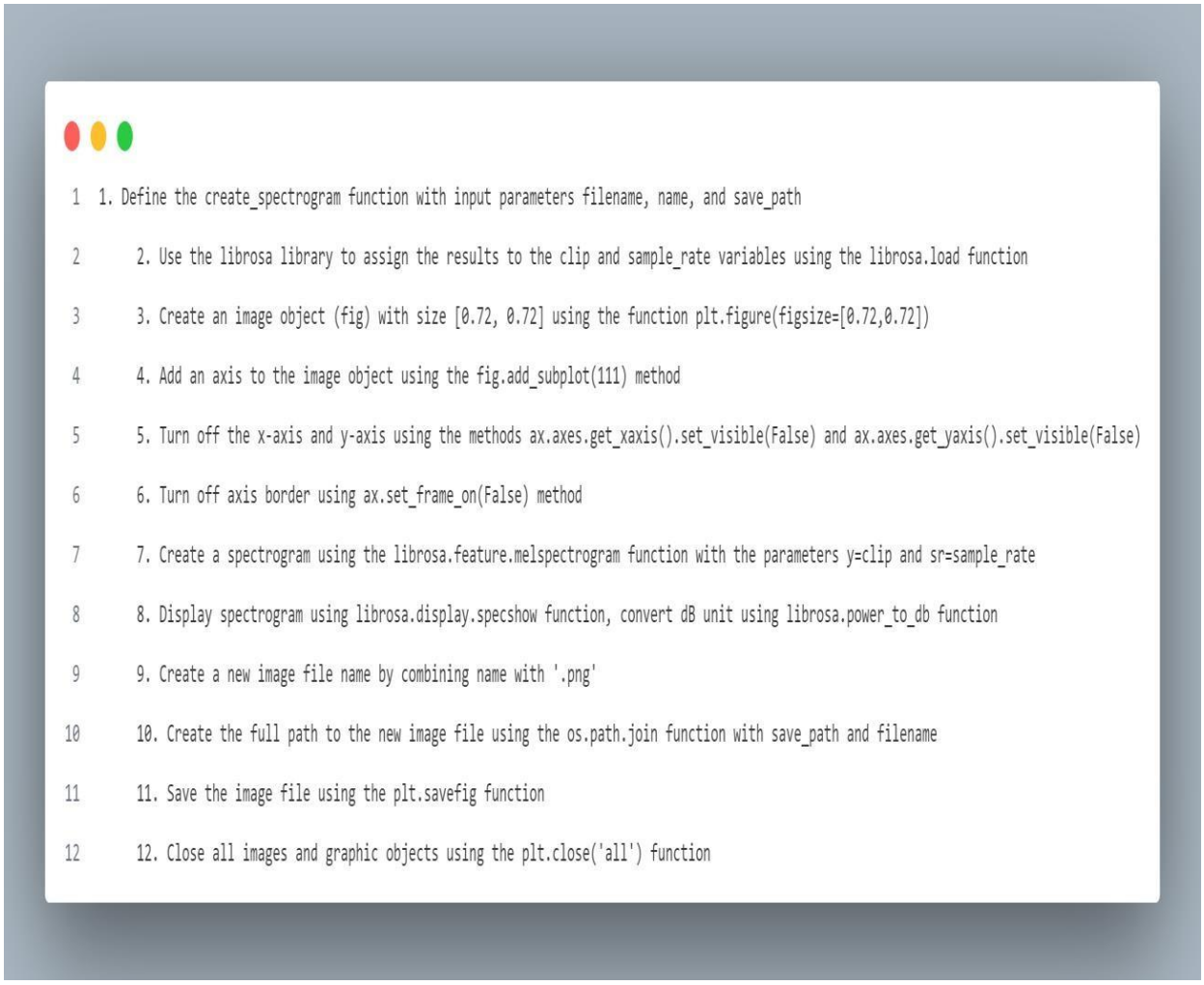
3.2.1. Tạo ảnh phổ

Biểu diễn trực quan các tần số của một tín hiệu nhất định với thời gian được gọi là Spectrogram [10]. Điều thú vị về những hình ảnh này là chúng tôi thực sự có thể sử dụng chúng như một công cụ chẩn đoán với Deep Learning và Computer Vision để huấn luyện mạng lưới thần kinh tích chập nhằm phân loại nhiều chủ đề khác nhau.

```
def create_spectrogram(filename, name, save_path):
    clip, sample_rate = librosa.load(filename, sr=None)
    fig = plt.figure(figsize=[0.72,0.72])
    ax = fig.add_subplot(111)
    ax.axis('off')
    ax.axes.get_xaxis().set_visible(False)
    ax.axes.get_yaxis().set_visible(False)
    ax.set_frame_on(False)
    spectrogram = librosa.feature.melspectrogram(y=clip, sr=sample_rate)
    librosa.display.specshow(librosa.power_to_db(spectrogram, ref=np.max))
    filename = name + '.png'
    save_file_path = os.path.join(save_path, filename)
    plt.savefig(save_file_path, dpi=400, bbox_inches='tight',pad_inches=0)
    plt.close('all')
```

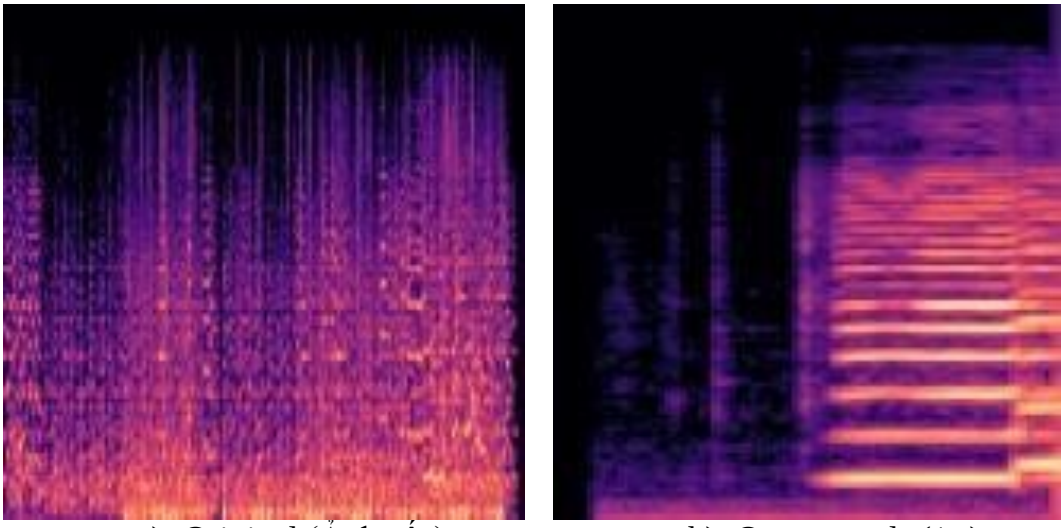
Hình 3. 5: Tạo Spectrogram từ tệp âm thanh

Phương pháp chuyển đổi dữ liệu âm thanh thành ảnh phổ được thực hiện thông qua việc sử dụng thư viện Librosa để chuyển đổi âm thanh về dạng ảnh phổ. (Minh họa **Hình 3. 5**). Ma trận 2D thu được là biểu đồ Spectrogram (minh họa **Hình 3. 7**).

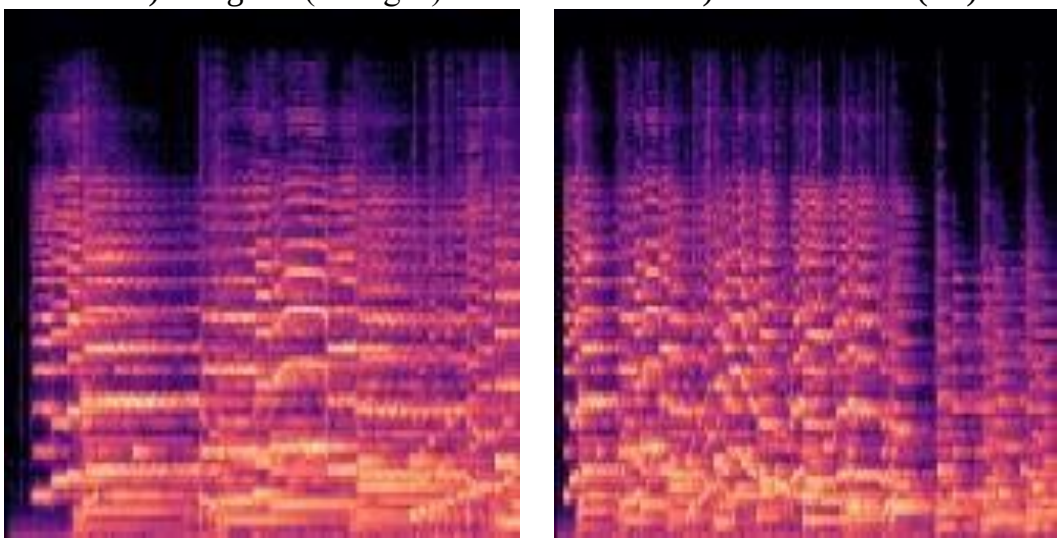


Hình 3. 6: Mã giả tương ứng Tạo spectrogram từ tệp âm thanh

Chúng tôi trình bày Mã giả (Pseudocode) tương ứng với việc chuyển đổi tệp âm thanh thành ảnh phổ. Giúp chúng tôi diễn đạt ý tưởng một cách dễ dàng và dễ hiểu. Trong đó, [0.72, 0.72] được truyền vào tham số figsize để xác định kích thước của hình ảnh, đại diện cho chiều rộng và chiều cao của hình ảnh trong đơn vị inch. Trong hàm add_subplot(), chúng ta truyền một tham số duy nhất là 111. Số 1 đầu tiên xác định số hàng của lưới subplot, số 1 thứ hai xác định số cột của lưới subplot và số 1 thứ ba xác định chỉ số của subplot hiện tại. Vì chúng tôi truyền 111 cho add_subplot(), điều này tạo ra một lưới subplot có 1 hàng và 1 cột và chỉ số của subplot hiện tại là 1. Do đó, chúng ta có một trục duy nhất trong hình ảnh (fig) và nó được gán cho biến ax.



a) Original (Ảnh gốc) b) One seconds (1 s)



c) 10 seconds (10 s) d) 30 seconds (30 s)

Hình 3. 7: Một số hình ảnh phổ minh họa có độ dài khác nhau được cắt từ một bài hát

Các hình ảnh (minh họa **Hình 3. 7)** cho thấy được sự khác biệt rõ khi ảnh phổ của các đoạn âm thanh riêng lẻ đến từ một bài hát từ tập dữ liệu thành 1 giây, 10 giây và 30 giây được biểu diễn dưới dạng màu sắc thường được ánh xạ từ giá trị cường độ âm thanh tại mỗi thời điểm và tần số cụ thể trong âm thanh, sự thay đổi trong cường độ âm thanh theo thời gian.

Ảnh phổ âm thanh được biểu diễn theo :

- **Trục X (Thời gian):** biểu đồ phổ âm thanh hiển thị thời gian trên trục ngang, vị trí cuối cùng của âm thanh sẽ ứng với thời điểm kết thúc của tệp âm thanh
- **Trục Y (Tần số) :** Trục dọc của biểu đồ biểu thị tần số. Vị trí cuối cùng trên trục này sẽ cho biết tần số cao nhất mà biểu đồ đang hiển thị.
- **Màu sắc hoặc Độ sáng:** Màu sắc thường được sử dụng để biểu thị cường độ âm thanh tại các tần số và thời điểm cụ thể.

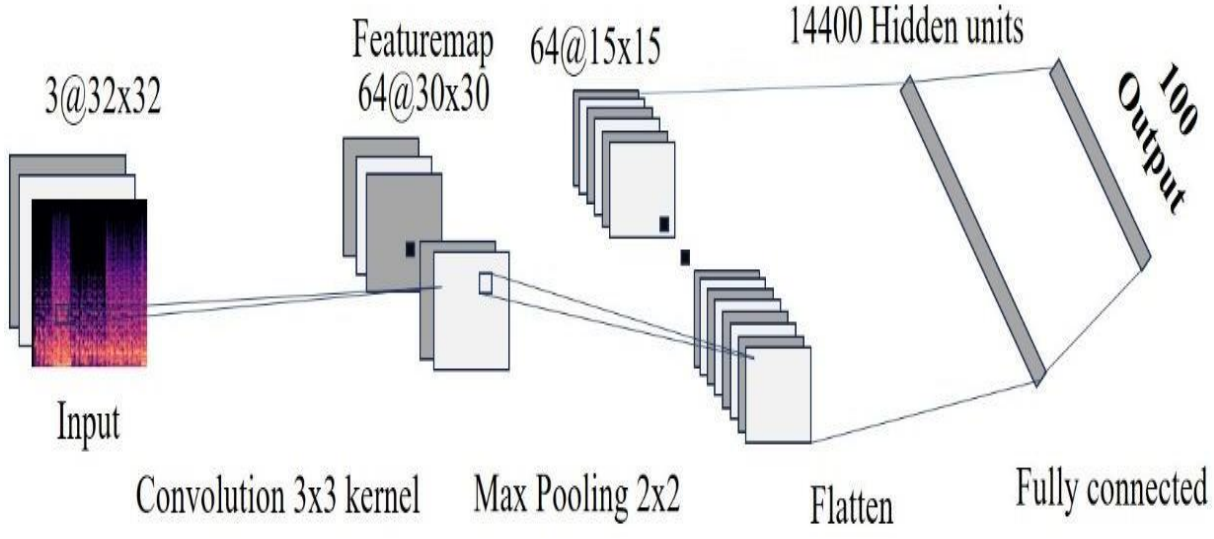
Bên cạnh đó, điểm đen thường biểu thị cường độ thấp hoặc không có âm thanh, trong khi màu đỏ (cam) thường được sử dụng để biểu thị các vùng có cường độ âm thanh cao.

Các màu đỏ hoặc cam thậm chí có thể biểu thị những phần của đoạn âm thanh có độ động lớn. Thông qua màu sắc, có thể quan sát sự biến động của cường độ âm thanh từ các vùng không có âm thanh đến các vùng có độ động lớn, sự thay đổi trong cường độ âm thanh theo thời gian.

3.2.2. Kiến trúc CNN

Xây dựng kiến trúc (minh họa **Hình 3. 8**) gọi tắt là mạng CNN1 (với một lớp chập). Kiến trúc mạng tích chập nông có thể hoạt động tốt trên biểu diễn hình ảnh tổng hợp như được trình bày trong [11] với kích thước đầu vào là 32x32. Các hình ảnh Spectrogram tạo ra trong nghiên cứu này cũng có kích thước 32x32, vì vậy chúng tôi sử dụng kiến trúc tương tự với [11] để thực hiện nhiệm vụ phân loại bài hát.

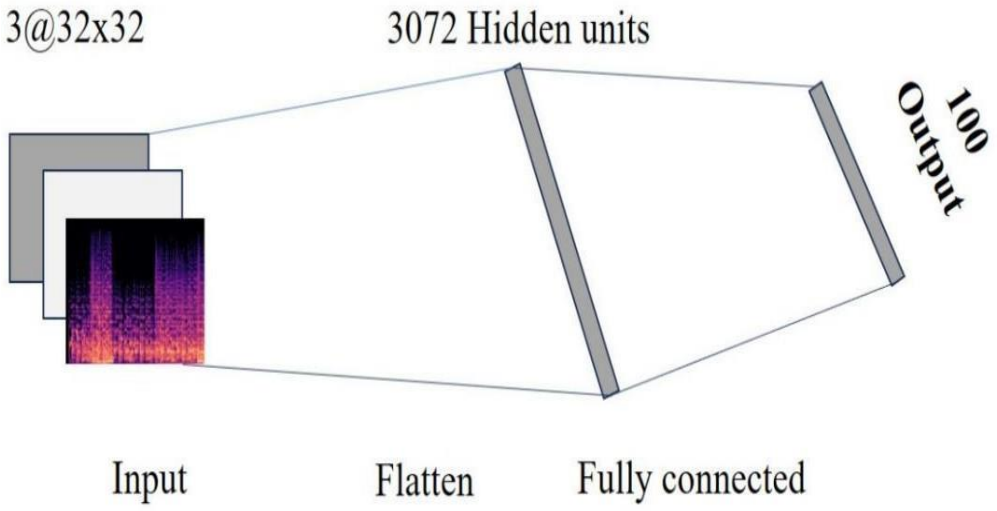
Dữ liệu đầu vào của mô hình là ảnh phổ. Hình ảnh này sau đó được truyền qua một loạt các lớp chập với 64 filter có kích thước 3x3 (với bước trượt 1). Hàm kích hoạt ReLU được áp dụng sau mỗi lớp tích chập để tạo ra đầu ra. Đầu vào có kích thước (IMAGE_LEN, IMAGE_LEN, 3), trong đó với kích thước đầu vào là 32x32, số 3 tương ứng với số lượng kênh màu: Trong ngữ cảnh RGB, mỗi ảnh có thể được biểu diễn bằng ba kênh màu (Red, Green, Blue), do đó có 3 kênh màu. Tiếp theo, một lớp composite tối đa (MaxPooling2D) được thêm vào mô hình. Lớp này thực hiện tổng hợp 2×2 trên đầu ra của lớp tích chập trước đó. Nó giúp giảm kích thước đầu ra và giữ lại các đặc điểm quan trọng. Tiếp theo, lớp phẳng (Flatten), được sử dụng để chuyển đổi đầu ra từ các lớp trước thành vectơ 1D để chuẩn bị cho lớp kết nối đầy đủ. Cuối cùng, một lớp được kết nối đầy đủ (Dense) với số lượng units bằng Num_Classes và sử dụng hàm kích hoạt softmax để tính xác suất của từng lớp.



Hình 3. 8: Kiến trúc mạng nơ-ron tích chập (CNN1) sử dụng trong nghiên cứu

3.2.3. Kiến trúc FC

Xây dựng kiến trúc (minh họa **Hình 3. 9**) là Fully Connected Layer (FC) một kiến trúc mạng nơ-ron mà trong đó mỗi nơ-ron của một lớp kết nối đầy đủ với tất cả các nơ-ron của lớp kế tiếp. Mạng kết nối đầy đủ không đặt ra các giả định cụ thể về cấu trúc của dữ liệu đầu vào. Điều này làm cho nó phù hợp với nhiều loại dữ liệu khác nhau mà không cần phải thực hiện tiền xử lý đặc biệt.



Hình 3. 9: Kiến trúc mạng FC sử dụng trong nghiên cứu

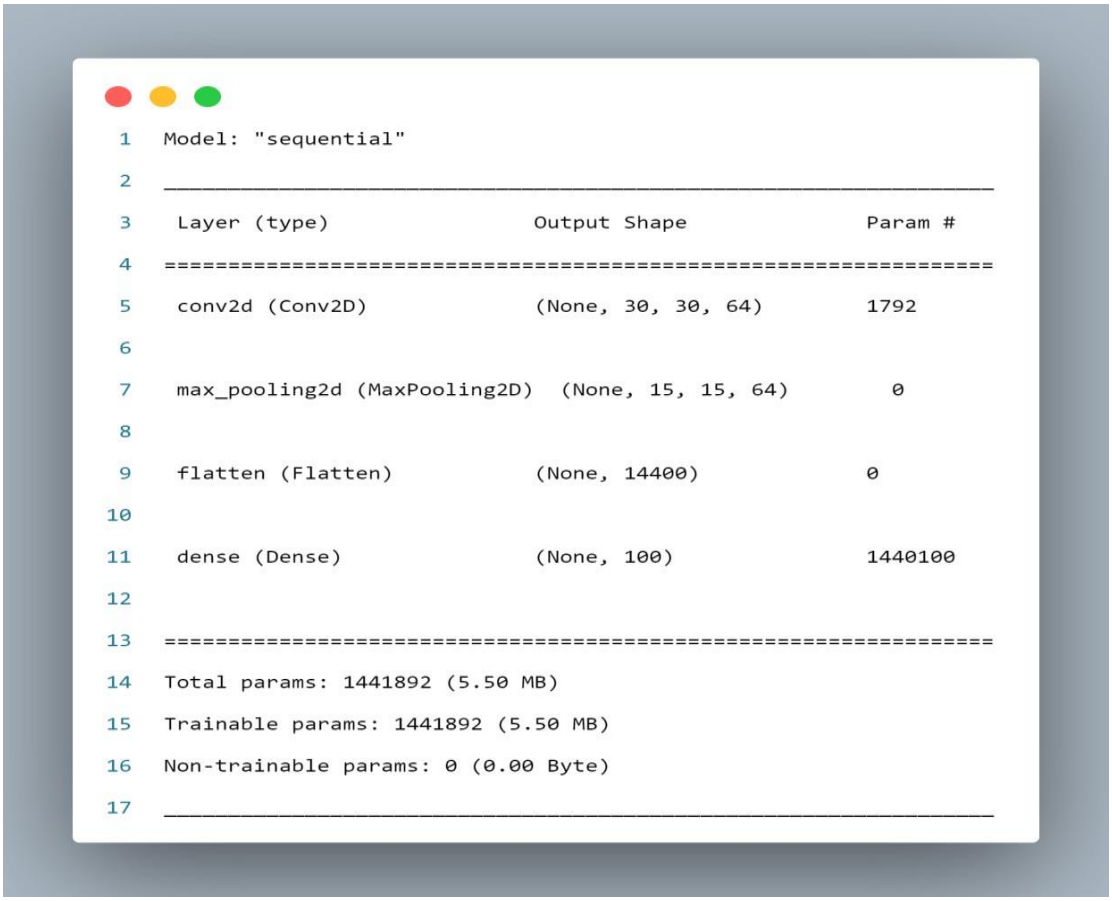
Các hình ảnh Spectrogram tạo ra trong nghiên cứu này đầu vào có kích thước (IMAGE_LEN, IMAGE_LEN, 3), trong đó với kích thước đầu vào là 32x32, Số 3 tương ứng với số lượng kênh màu: Trong ngữ cảnh RGB, mỗi ảnh có thể được biểu diễn bằng ba kênh màu (Red, Green, Blue), do đó có 3 kênh màu. Hình cho thấy mạng nhận được hình ảnh spectrogram làm đầu vào. Tiếp theo,lớp phẳng (Flatten), được sử dụng để chuyển đổi đầu ra từ các lớp trước thành vectơ 1D để chuẩn bị cho lớp kết nối đầy đủ. Cuối cùng, một lớp được kết nối đầy đủ (Dense) với số lượng units bằng Num_Classes và sử dụng hàm kích hoạt softmax để tính xác suất của từng lớp.

3.2.4. Xây dựng các mô hình

❖ **Model CNN1**

Kiến trúc CNN1 (minh họa **Hình 3. 10**) bao gồm một lớp chập với 64 filter có kích thước 3x3 (với bước trượt 1) được áp dụng để trích xuất các đặc trưng từ hình ảnh quang phổ đầu vào. Hàm kích hoạt ReLU được áp dụng sau mỗi lớp tích chập để tạo ra đầu ra. Đầu vào có kích thước (IMAGE_LEN, IMAGE_LEN, 3), trong đó với kích thước đầu vào là 32x32, Số 3 tương ứng với số lượng kênh màu: Trong ngữ cảnh RGB, mỗi ảnh có thể được biểu diễn bằng ba kênh màu (Red, Green, Blue), do đó có 3 kênh màu. Tiếp theo, một lớp composite tối đa (MaxPooling2D) được thêm vào mô hình. Lớp này thực hiện tổng hợp 2×2 trên đầu ra của lớp tích chập trước đó. Nó giúp giảm kích thước đầu ra và giữ lại các đặc điểm quan trọng. Tiếp theo, lớp phẳng (Flatten), được sử dụng để chuyển đổi đầu ra từ các lớp trước thành vectơ 1D để chuẩn bị cho lớp kết nối đầy đủ. Cuối cùng, một lớp được kết nối đầy đủ (Dense) với số lượng units bằng Num_Classes và sử dụng hàm kích hoạt softmax để tính xác suất của từng lớp.

Bên cạnh đó, các siêu tham số quan trọng bao gồm learning rate (0.0001), số lượng epochs (500), Batch_Size(64), Image_Len(32), Numclass(100).



```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 conv2d (Conv2D)             (None, 30, 30, 64)        1792

 max_pooling2d (MaxPooling2D)  (None, 15, 15, 64)        0

 flatten (Flatten)           (None, 14400)             0

 dense (Dense)               (None, 100)               1440100

=================================================================
Total params: 1441892 (5.50 MB)
Trainable params: 1441892 (5.50 MB)
Non-trainable params: 0 (0.00 Byte)
_________________________________________________________________
```

Hình 3. 10: Các lớp được sử dụng cho model CNN1

❖ **Model CNN2**

Kiến trúc CNN2 (minh họa **Hình 3. 11**) bao gồm hai lớp chập với 64 và 128 filters, mỗi lớp có kích thước kernel là (3, 3) và sử dụng hàm kích hoạt ReLU được áp dụng sau mỗi lớp tích chập để tạo ra đầu ra. Các lớp MaxPooling2D sau mỗi lớp chập với pool size là (2, 2), giúp giảm kích thước của đầu ra và giữ lại các đặc điểm quan trọng. Tiếp theo, lớp phẳng (Flatten), được sử dụng để chuyển đổi đầu ra từ các lớp trước thành vectơ 1D để chuẩn bị cho lớp kết nối đầy đủ. Cuối cùng, một lớp được kết nối đầy đủ (Dense) với số lượng units bằng Num_Classes và sử dụng hàm kích hoạt softmax để tính xác suất của từng lớp.

Bên cạnh đó, các siêu tham số quan trọng bao gồm learning rate (0.0001), số lượng epochs (500), Batch_Size(64), Image_Len(32), Num_class(100).

```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 conv2d (Conv2D)             (None, 30, 30, 64)        1792

 max_pooling2d (MaxPooling2D)  (None, 15, 15, 64)        0

 conv2d_1 (Conv2D)           (None, 13, 13, 128)       73856

 max_pooling2d_1 (MaxPooling2D)  (None, 6, 6, 128)         0

 flatten (Flatten)           (None, 4608)              0

 dense (Dense)               (None, 100)               460900

=================================================================
Total params: 536548 (2.05 MB)
Trainable params: 536548 (2.05 MB)
Non-trainable params: 0 (0.00 Byte)
_________________________________________________________________
```

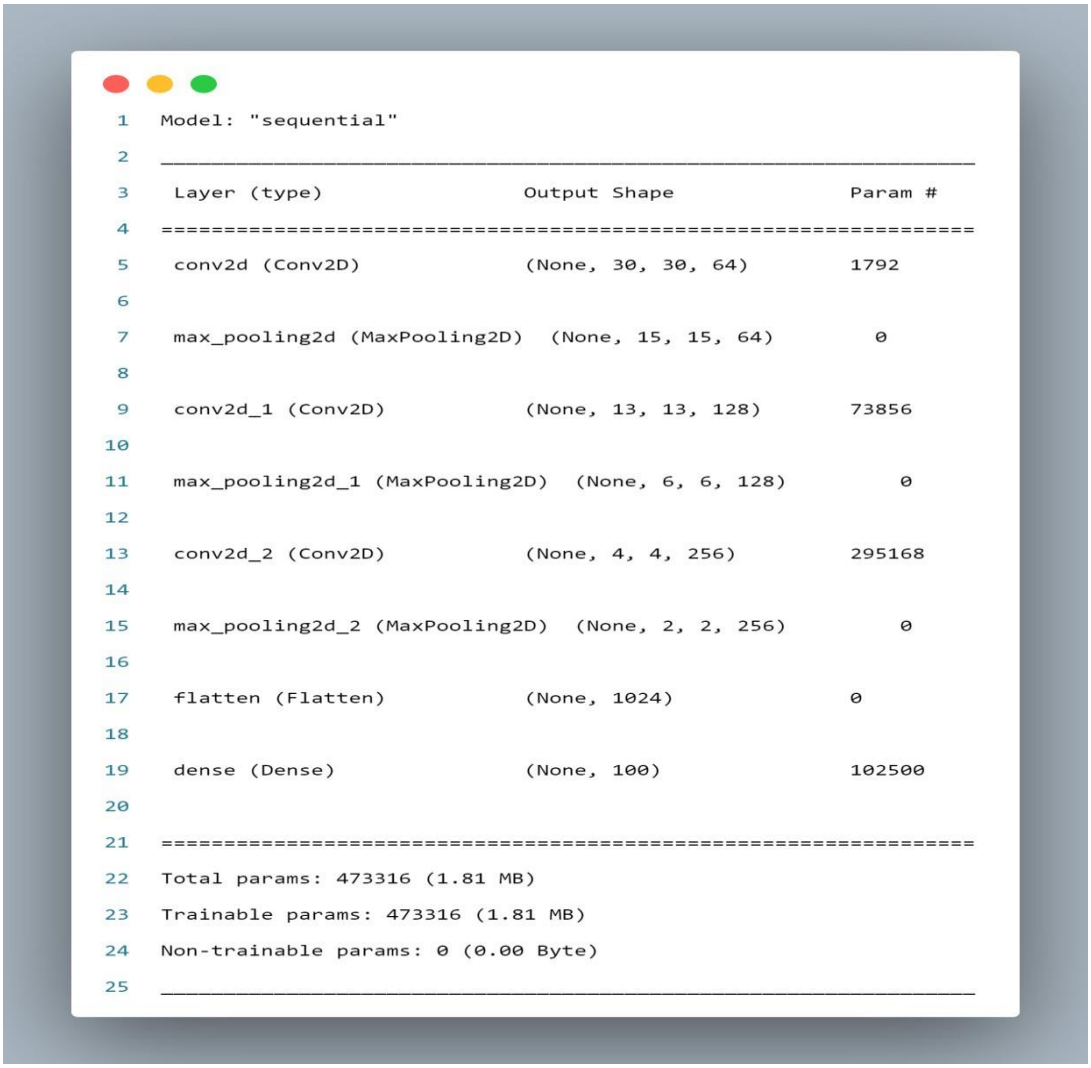
Hình 3. 11: Các lớp được sử dụng cho model CNN2

Mô hình CNN2 được mở rộng từ CNN1 với thêm một lớp chập và một lớp MaxPooling, gia tăng khả năng học đặc trưng từ dữ liệu. Siêu tham số được cấu hình để đảm bảo quá trình huấn luyện hiệu quả và tránh overfitting.

❖ **Model CNN3**

Kiến trúc CNN3 (minh họa **Hình 3. 13**) bao gồm ba lớp chập với 64, 128 và 256 filters, mỗi lớp có kích thước kernel là (3, 3) và sử dụng hàm kích hoạt ReLU được áp dụng sau mỗi lớp tích chập để tạo ra đầu ra. Các lớp MaxPooling2D sau mỗi lớp chập với pool size là (2, 2), giúp giảm kích thước của đầu ra và giữ lại các đặc điểm quan trọng. Tiếp theo, lớp phẳng (Flatten), được sử dụng để chuyển đổi đầu ra từ các lớp trước thành vectơ 1D để chuẩn bị cho lớp kết nối đầy đủ. Cuối cùng, một lớp được kết nối đầy đủ (Dense) với số lượng units bằng Num_Classes và sử dụng hàm kích hoạt softmax để tính xác suất của từng lớp.

Bên cạnh đó, các siêu tham số quan trọng bao gồm learning rate (0.0001), số lượng epochs (500), Batch_Size(64), Image_Len(32), Numclass(100).



```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 conv2d (Conv2D)             (None, 30, 30, 64)        1792

 max_pooling2d (MaxPooling2D)  (None, 15, 15, 64)        0

 conv2d_1 (Conv2D)           (None, 13, 13, 128)       73856

 max_pooling2d_1 (MaxPooling2D)  (None, 6, 6, 128)         0

 conv2d_2 (Conv2D)           (None, 4, 4, 256)         295168

 max_pooling2d_2 (MaxPooling2D)  (None, 2, 2, 256)         0

 flatten (Flatten)           (None, 1024)              0

 dense (Dense)               (None, 100)               102500

=================================================================
Total params: 473316 (1.81 MB)
Trainable params: 473316 (1.81 MB)
Non-trainable params: 0 (0.00 Byte)
_________________________________________________________________
```

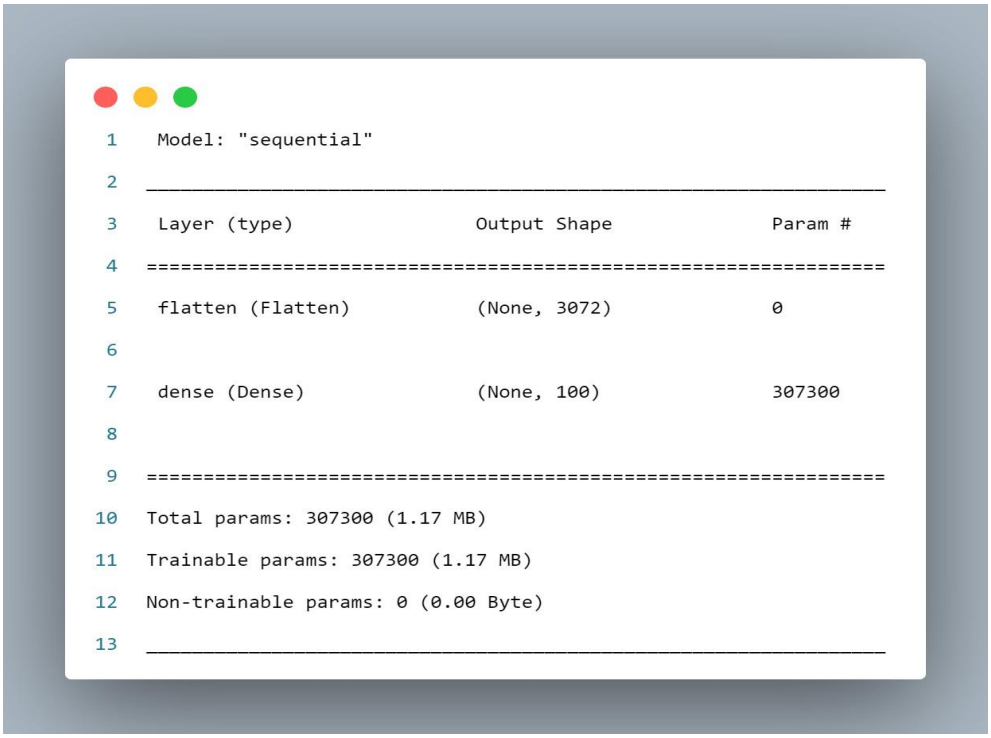
Hình 3. 12: Các lớp được sử dụng cho model CNN3

Mô hình CNN3 được mở rộng từ CNN1 với thêm hai lớp chập và hai lớp MaxPooling, gia tăng khả năng học đặc trưng từ dữ liệu. Siêu tham số được cấu hình để đảm bảo quá trình huấn luyện hiệu quả và tránh overfitting.

❖ **Model FC**

Kiến trúc FC (minh họa **Hình 3. 13**) bao gồm một lớp InputLayer với kích thước đầu vào là (Image_Len, Image_Len, 3) phản ánh kích thước của hình ảnh. Tiếp theo, lớp Flatten được sử dụng để làm phẳng feature map thành một vector 1D, chuẩn bị cho việc đưa vào lớp Dense. Cuối cùng, Lớp Dense với số lượng units bằng Num_Class(100) và hàm kích hoạt softmax được sử dụng để đảm bảo đầu ra là xác suất cho từng lớp.

Bên cạnh đó, các siêu tham số quan trọng bao gồm Learning Rate (0.0001), số lượng epochs (500), Batch_Size(64), Image_Len(32), Numclass(100).



```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 flatten (Flatten)           (None, 3072)              0

 dense (Dense)               (None, 100)               307300

=================================================================
Total params: 307300 (1.17 MB)
Trainable params: 307300 (1.17 MB)
Non-trainable params: 0 (0.00 Byte)
_________________________________________________________________
```

Hình 3. 13: Các lớp được sử dụng cho model FC

3.3. Giải pháp cài đặt

Hệ thống được thiết kế bằng ngôn ngữ lập trình Python và sử dụng các thư viện hỗ trợ như Annoy, Librosa, Os, Soundfile, Tqdm, Numpy, Pickle được sử dụng để xử lý và phân tích dữ liệu âm thanh. Ví dụ, thư viện librosa được sử dụng để trích xuất đặc trưng âm thanh như biểu diễn tần số và độ rộng mẫu. Thư viện numpy được sử dụng để tính toán và xử lý các ma trận số học.

3.4. Tổng kết chương

Chương này trình bày kiến trúc tổng quát của hệ thống, mô tả cách thức hệ thống làm việc, xây dựng các mô hình và giải pháp cài đặt hệ thống.

CHƯƠNG 4. KIỂM THỬ VÀ ĐÁNH GIÁ

4.1. Kịch bản kiểm thử

a) Kịch bản 1

Trong nghiên cứu, chúng tôi tiến hành thực hiện xây dựng kiến trúc mô hình CNN/FC dùng đánh giá hiệu suất phân lớp trên dữ liệu âm thanh được trích xuất đặc trưng thành mảng Numpy (trình bày ở **Mục 3.1.1**), chúng tôi sẽ kiểm thử v ới tập dữ liệu A bao gồm 100 bài hát không lời, trong khi tập dữ liệu B bao gồm 100 tệp âm thanh có cùng tên bài hát nhưng được chơi bằng các loại nhạc cụ khác nhau. Sau đó, thư viện pydub cắt ngẫu nhiên các file âm thanh gốc thành các clip có độ dài dưới 10 giây trong tập kiểm thử vì người dùng thường sử dụng một đoạn mã cụ thể để tìm bản nhạc. Các tệp âm thanh gốc được chia thành các đoạn âm thanh có độ dài khác nhau trong tập huấn luyện bao gồm 5 giây, 10 giây. Bước đầu tiên, chúng tôi chuyển đổi âm thanh thành mảng Numpy. Sau đó sử dụng mô hình CNN/FC để đánh giá hiệu suất phân lớp trên dữ liệu âm thanh được trích xuất đặc trưng thành mảng Numpy. Dữ liệu đầu vào được xác định bằng tham số input_shape=args.lengthaudio, thể hiện số lượng samples đã được chuyển đổi mảng Numpy. Tiếp đến là chọn đối số, áp dụng các bộ lọc 64 filter. Sau khi lớp tích chập được áp dụng, bước tiếp theo là sử dụng lớp MaxPooling để giảm kích thước của dữ liệu. Sau đó, chúng tôi thêm một lớp Dense vào mô hình với số đơn vị (units) là i_label và hàm kích hoạt được sử dụng trong lớp Dense là softmax, trong đó, i_label là số lượng lớp cần phân loại trong bài toán. Lớp Dense này đại diện cho " Fully Connected Layer " trong mô hình. Cuối cùng, mô hình được biên soạn (compile) với hàm mất mát là categorical_crossentropy và tối ưu hóa bằng thuật toán Adam.

Với các siêu tham số được sử dụng: learning rate là 0.0001, batch size là 64, input_shape=args.lengthaudio, số lượng epochs (500), Numclass(100). Sau đó sẽ dựa trên kết quả huấn luyện mà chọn ra model tốt nhất.

b) Kịch bản 2

Trong quá trình nghiên cứu, chúng tôi đã tiến hành thực nghiệm trên các mô hình CNN1, CNN2, CNN3 và FC để quét ảnh phổ, chúng tôi sẽ kiểm thử với tập dữ liệu A bao gồm 100 bài hát không lời, trong khi tập dữ liệu B bao gồm 100 tệp âm thanh có cùng tên bài hát nhưng được chơi bằng các loại nhạc cụ khác nhau. Sau đó, thư viện pydub cắt ngẫu nhiên các file âm thanh gốc thành các clip có độ dài dưới 10 giây trong tập kiểm thử vì người dùng thường sử dụng một đoạn mã cụ thể để tìm bản nhạc. Các tệp âm thanh gốc được chia thành các đoạn âm thanh có độ dài khác nhau trong tập huấn luyện, bao gồm 1 giây, 3 giây, 5 giây, 10 giây, 20 giây, 30 giây, 60 giây và 90 giây. Đối với hướng tiếp cận sử dụng mạng nơ ron tích chập để phân tính hình ảnh dựa trên ảnh phổ thì chúng tôi thực hiện 2 bước. Bước đầu tiên, chúng tôi chuyển đổi âm thanh thành dạng ảnh phổ (trình bày ở **Mục 3.2.1**). Bước kế tiếp, chúng tôi sẽ sử dụng các model mạng nơ ron tích chập để phân tích tập ảnh phổ để trả về kết quả gồm có model CNN1, model CNN2, model CNN3 và model FC.

Với các siêu tham số learning rate là 0.0001, batch size là 64, số lượng epochs (500), Batch_Size(64), Image_Len(32), Numclass(100), optimizer là Adam. Sau đó sẽ dựa trên kết quả huấn luyện mà chọn ra model tốt nhất.

4.1.1. Mô tả tập dữ liệu

a) Kịch bản 1

Việc nghiên cứu và thực nghiệm trên bộ dữ liệu được lấy từ nghiên cứu [1] bằng cách sử dụng cùng một bộ dữ liệu mà các nghiên cứu trước đã sử dụng, nghiên cứu này có thể so sánh kết quả của mình với nghiên cứu trước đó. Điều này giúp xác nhận tính chính xác của kết quả và tăng cường khả năng so sánh của nghiên cứu.

Xây dựng kỹ thuật để chia tệp âm thanh thành các đoạn nhỏ có thời lượng khác nhau, bao gồm 5 giây, 10 giây. Chúng tôi đã sử dụng thư viện pydub của Python. Quá trình bắt đầu bằng việc chọn độ dài của các bản âm thanh phụ, có thể là 5 giây, 10 giây. Sau đó, chúng tôi đi qua từng phần của file âm thanh gốc. Đối với mỗi phần, chúng tôi tính toán thời gian bắt đầu và kết thúc. Kiểm tra xem thời gian kết thúc có vượt quá độ dài của tệp âm thanh hay không và nếu có, hãy hạn chế không vượt quá thời gian đó. Sau đó, chúng tôi tạo tên cho tệp âm thanh phụ mới bằng cách kết hợp tên tệp gốc với số bộ phận. Chúng tôi sử dụng thời gian bắt đầu và kết thúc được tính toán để thực hiện cắt âm thanh từ tệp gốc và chuyển đổi âm thanh con thành tệp mới. Sau đó, mã tiếp tục bằng cách đặt đường dẫn thư mục đầu ra và lặp qua các tệp âm thanh trong thư mục đầu vào. Đối với mỗi tệp âm thanh, mã xác định đường dẫn tệp âm thanh đầu vào và thư mục đích cho phụ. Nếu thư mục đích chưa tồn tại, nó sẽ được tạo. Cuối cùng, chia tệp âm thanh thành nhiều phần nhỏ, mỗi phần có độ dài và lưu vào một thư mục.

Một phương pháp kỹ thuật được sử dụng để chia nhỏ các tệp âm thanh một cách ngẫu nhiên thành các độ dài tùy chỉnh là sử dụng thư viện pydub trong Python. Quá trình này bắt đầu bằng tổng thời gian của file âm thanh gốc. Sau đó, chọn ngẫu nhiên thời điểm bắt đầu cắt để tạo ra các phần âm thanh ngẫu nhiên. Khi các phần âm thanh đã được tạo, chúng tôi tạo một thư mục đầu ra dựa trên tên của tệp âm thanh gốc để lưu trữ các đoạn âm thanh được tách. Các bản âm thanh này được xuất và được tự động lưu vào thư mục đầu ra. Kết quả là một thư mục gồm các bản âm thanh ngẫu nhiên có độ dài tùy chỉnh từ tệp âm thanh gốc.

Trong **Bảng 4. 1**, Tập dữ liệu A bao gồm 100 bài hát không lời, trong khi tập dữ liệu B bao gồm 100 tệp âm thanh có cùng tên bài hát nhưng được chơi bằng các nhạc cụ khác nhau. Điều này mang lại sự đa dạng và phong phú cho bộ dữ liệu của chúng tôi để nghiên cứu và phân tích ảnh hưởng sự phân bố thời lượng đối với bài hát. AXstest và BXstest bao gồm các mẫu trong đó mỗi mẫu là một đoạn có độ dài X giây được trích xuất ngẫu nhiên từ một bài hát trong tập dữ liệu A và B tương ứng. Đồng thời, AYstrain bao gồm các chuỗi con của tệp âm thanh có độ dài Y giây được trích xuất từ A. Ví dụ: như được hiển thị trong **Bảng 4. 1** , A05strain chứa các phân đoạn âm thanh được cắt từ tập dữ liệu A, có cùng độ dài 5 giây. Tương tự, A10strain bao gồm các mẫu có độ dài là 10 giây.

Bảng 4. 1: Thông tin về các tập dữ liệu được sử dụng trong thí nghiệm: AXstest, BXstest bao gồm các mẫu trong đó mỗi mẫu là một đoạn có độ dài X giây được trích xuất ngẫu nhiên từ một bài hát trong tập dữ liệu A, B tương ứng, trong khi AYstrain bao gồm các chuỗi con của tệp âm thanh có độ dài Y giây được trích xuất từ A.

Datasets	The number of samples	The feature of samples
A10stest	100	220500
B10stest	100	220500
B05stest	100	110250
A05strain	5265	110250
A10strain	2660	220500

b) Kịch bản 2

Việc nghiên cứu và thực nghiệm trên bộ dữ liệu được lấy từ nghiên cứu [1] bằng cách sử dụng cùng một bộ dữ liệu mà các nghiên cứu trước đã sử dụng, nghiên cứu này có thể so sánh kết quả của mình với nghiên cứu trước đó. Điều này giúp xác nhận tính chính xác của kết quả và tăng cường khả năng so sánh của nghiên cứu.

Xây dựng kỹ thuật để chia tệp âm thanh thành các đoạn nhỏ có thời lượng khác nhau, bao gồm 1 giây, 3 giây, 5 giây, 10 giây, 20 giây, 30 giây, 60 giây và 90 giây. Chúng tôi đã sử dụng thư viện pydub của Python. Quá trình bắt đầu bằng việc chọn độ dài của các bản âm thanh phụ, có thể là 1 giây, 3 giây, 5 giây, 10 giây, 20 giây, 30 giây, 60 giây hoặc 90 giây. Sau đó, chúng tôi đi qua từng phần của file âm thanh gốc. Đối với mỗi phần, chúng tôi tính toán thời gian bắt đầu và kết thúc. Kiểm tra xem thời gian kết thúc có vượt quá độ dài của tệp âm thanh hay không và nếu có, hãy hạn chế không vượt quá thời gian đó. Sau đó, chúng tôi tạo tên cho tệp âm thanh phụ mới bằng cách kết hợp tên tệp gốc với số bộ phận. Chúng tôi sử dụng thời gian bắt đầu và kết thúc được tính toán để thực hiện cắt âm thanh từ tệp gốc và chuyển đổi bản âm thanh con thành tệp mới. Sau đó, mã tiếp tục bằng cách đặt đường dẫn thư mục đầu ra và lặp qua các tệp âm thanh trong thư mục đầu vào. Đối với mỗi tệp âm thanh, mã xác định đường dẫn tệp âm thanh đầu vào và thư mục đích cho phụ. Nếu thư mục đích chưa tồn tại, nó sẽ được tạo. Cuối cùng, chia tệp âm thanh thành nhiều phần nhỏ, mỗi phần có độ dài và lưu vào một thư mục.

Một phương pháp kỹ thuật được sử dụng để chia nhỏ các tệp âm thanh một cách ngẫu nhiên thành các độ dài tùy chỉnh là sử dụng thư viện pydub trong Python. Quá trình này bắt đầu bằng tổng thời gian của file âm thanh gốc. Sau đó, chọn ngẫu nhiên thời điểm bắt đầu cắt để tạo ra các phần âm thanh ngẫu nhiên. Khi các phần âm thanh đã được tạo, chúng tôi tạo một thư mục đầu ra dựa trên tên của tệp âm thanh gốc để lưu trữ các đoạn âm thanh được tách. Các bản âm thanh này được xuất và được tự động lưu vào thư mục đầu ra. Kết quả là một thư mục gồm các bản âm thanh ngẫu nhiên có độ dài tùy chỉnh từ tệp âm thanh gốc.

Trong **Bảng 4. 2**, Tập dữ liệu A bao gồm 100 bài hát không có lời bài hát, trong khi tập dữ liệu B bao gồm 100 tệp âm thanh có cùng tên bài hát nhưng được chơi bằng các nhạc cụ khác nhau. Điều này mang lại sự đa dạng và phong phú cho bộ dữ liệu của chúng

tôi để nghiên cứu và phân tích ảnh hưởng của sự phân bố thời lượng đối với bài hát. AXstest và BXstest bao gồm các mẫu trong đó mỗi mẫu là một đoạn có độ dài X giây được trích xuất ngẫu nhiên từ một bài hát trong tập dữ liệu A và B tương ứng. Đồng thời, AYstrain bao gồm các chuỗi con của tệp âm thanh có độ dài Y giây được trích xuất từ A. Ví dụ: như được hiển thị trong **Bảng 4. 2**, A03strain chứa các phân đoạn âm thanh được cắt từ tập dữ liệu A, có cùng độ dài 3 giây. Tương tự, A01strain, A05strain, A10strain, A30strain, A60strain và A90strain bao gồm các mẫu có độ dài lần lượt là 1, 5, 10, 30, 60 và 90 giây.

Bảng 4. 2: Thông tin về các tập dữ liệu được sử dụng trong thí nghiệm: AXstest, BXstest bao gồm các mẫu trong đó mỗi mẫu là một đoạn có độ dài X giây được trích xuất ngẫu nhiên từ một bài hát trong tập dữ liệu A, B tương ứng, trong khi AYstrain bao gồm các chuỗi con của tệp âm thanh có độ dài Y giây được trích xuất từ A.

Datasets	The number of samples
A	100
B	100
A10stest	100
B10stest	100
B05stest	100
A01strain	26122
A03strain	8738
A05strain	5265
A10strain	2660
A20strain	1353
A30strain	919
A60strain	481
A90strain	346

4.1.2. Môi trường thực nghiệm

Nghiên cứu được thực hiện dựa trên nền tảng ngôn ngữ Python sử dụng các thư viện của nó và Anaconda giúp đơn giản hóa việc cài đặt, quản lý và triển khai packages (Numpy, Scipy, Librosa,…) là nền tảng mã nguồn mở về khoa học dữ liệu trên Python. Cùng với Keras - một mã nguồn mở cho mạng nơ-ron hỗ trợ chạy mô hình trên CPU. Ngoài ra còn có tìm kiếm bài hát trên nền tảng website.

4.1.3. Cơ sở đánh giá

Đề tài xây dựng và đánh giá mô hình phân lớp, nghiên cứu thực hiện đánh giá trên độ đo Accuracy (ACC). Chi tiết về các độ đo được trình bày bên dưới:

- **Accuracy (ACC):** độ chính xác, là tỷ lệ giữa số mẫu dữ liệu được dự đoán chính xác trên tổng số mẫu dữ liệu thực hiện dự đoán.

4.2. Kết quả kiểm thử

4.2.1. Phân lớp trên dữ liệu âm thanh chuyển đổi thành mảng 1 chiều

Trong nghiên cứu này, tập dữ liệu A bao gồm 100 bài hát không có lời bài hát, trong khi tập dữ liệu B bao gồm 100 tệp âm thanh có cùng tên bài hát nhưng được chơi bằng các loại nhạc cụ khác nhau. Sau đó, thư viện pydub cắt ngẫu nhiên các file âm thanh gốc thành các clip có độ dài dưới 10 giây do người dùng thường sử dụng một đoạn mã cụ thể để tìm toàn bộ bản nhạc. Các tệp âm thanh gốc được chia thành các đoạn âm thanh có độ dài khác nhau trong tập huấn luyện, bao gồm 5 giây, 10 giây. Tiếp theo, sử dụng thư viện Librosa của python để chuyển đổi tín hiệu âm thanh thành mảng Numpy. Tiếp theo, mô hình CNN/FC được xây dựng sau khi có được trích xuất đặc trưng dữ liệu âm thanh. Kiến trúc mạng thần kinh này được thiết kế đặc biệt để hoạt động với dữ liệu 1D. Sự đa dạng trong các bộ thử nghiệm này đã mang lại những kết quả thú vị, giúp chúng tôi có cơ sở để so sánh và phân tích. Việc lựa chọn các đoạn âm thanh có độ dài khác nhau đóng vai trò quan trọng trong việc tạo sự đa dạng và chân thực cho quá trình luyện tập. Điều này giúp mô hình tiếp xúc với nhiều tình huống âm nhạc khác nhau và tìm hiểu các đặc điểm âm nhạc quan trọng trong nhiều khung thời gian khác nhau. Điều này có ý nghĩa thực tiễn quan trọng vì người nghe thường có thể nghe ngẫu nhiên bất kỳ đoạn nhạc nào trong bài hát. Kết quả được trình bày trong (**Bảng 4. 3, Bảng 4. 4 và Bảng 4. 5)** dưới đây:

Trong **Bảng 4. 3** bên dưới, chúng tôi trình bày kết quả về hiệu suất phân loại bài hát từ một đoạn trích được cắt ngẫu nhiên từ 100 bài hát trong tập dữ liệu gốc "A" có độ dài 10 giây. Trong quá trình đào tạo, chúng tôi đã xây dựng tập dữ liệu huấn luyện trong tập dữ liệu "A" đoạn âm thanh có độ dài 10 giây. Mục tiêu của chúng tôi là đánh giá hiệu suất phân loại bài hát dựa trên độ dài của đoạn trích trong bài hát. Ngoài ra, chúng tôi đã áp dụng phương pháp là mô hình CNN/FC cho bộ dữ liệu này trong nhiệm vụ phân loại bài hát dựa trên dữ liệu âm thanh.

Bảng 4. 3: Hiệu suất phân loại bài hát của một đoạn trích từ bài hát được cắt ngẫu nhiên từ 100 bài hát trong tập dữ liệu gốc "A" có độ dài 10 giây

	Models	**CNN**		**FC**	
Roles	**Datasets**	**Accuracy**	**Time(seconds)**	**Accuracy**	**Time(seconds)**
Training set	A10strain	0.99591	2781.25371	0.99959	2188.65752
Test set	A10stest	0.11000		0.09000	

Trong **Bảng 4. 4** bên dưới, chúng tôi trình bày kết quả về hiệu suất phân loại bài hát từ một đoạn trích được cắt ngẫu nhiên từ 100 bài hát trong tập dữ liệu gốc "A" có độ dài 10 giây. Trong quá trình đào tạo, chúng tôi đã xây dựng tập dữ liệu huấn luyện trong tập dữ liệu "B" đoạn âm thanh có độ dài 10 giây. Mục tiêu của chúng tôi là đánh giá hiệu suất phân loại bài hát dựa trên độ dài của đoạn trích trong bài hát. Ngoài ra, chúng tôi đã áp dụng phương pháp là mô hình CNN/FC cho bộ dữ liệu này trong nhiệm vụ phân loại bài hát dựa trên dữ liệu âm thanh.

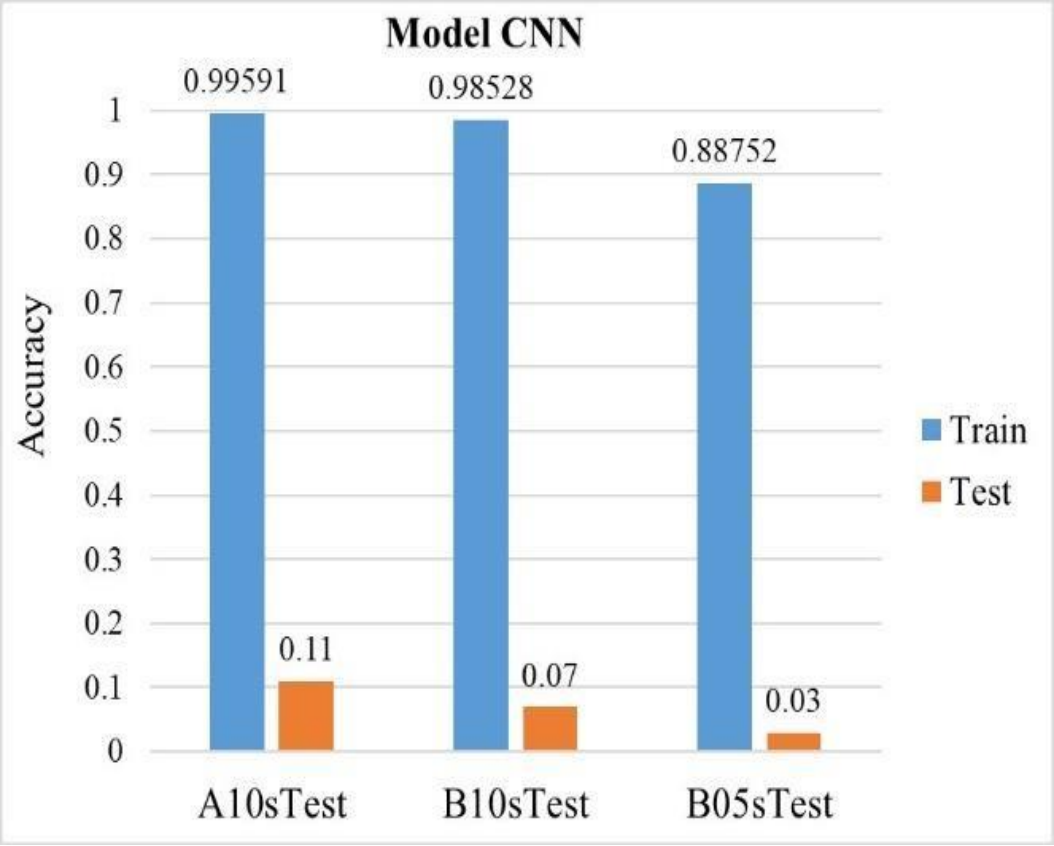
Bảng 4. 4: Hiệu suất phân loại bài hát của một đoạn trích từ bài hát được cắt ngẫu nhiên từ 100 bài hát trong tập dữ liệu gốc "B" có độ dài 10 giây

	Models	CNN		FC	
Roles	**Datasets**	**Accuracy**	**Time(seconds)**	**Accuracy**	**Time(seconds)**
Training set	A10strain	0.98528	2861.36575	0.99959	2520.18544
Test set	B10stest	0.07000		0.07000	

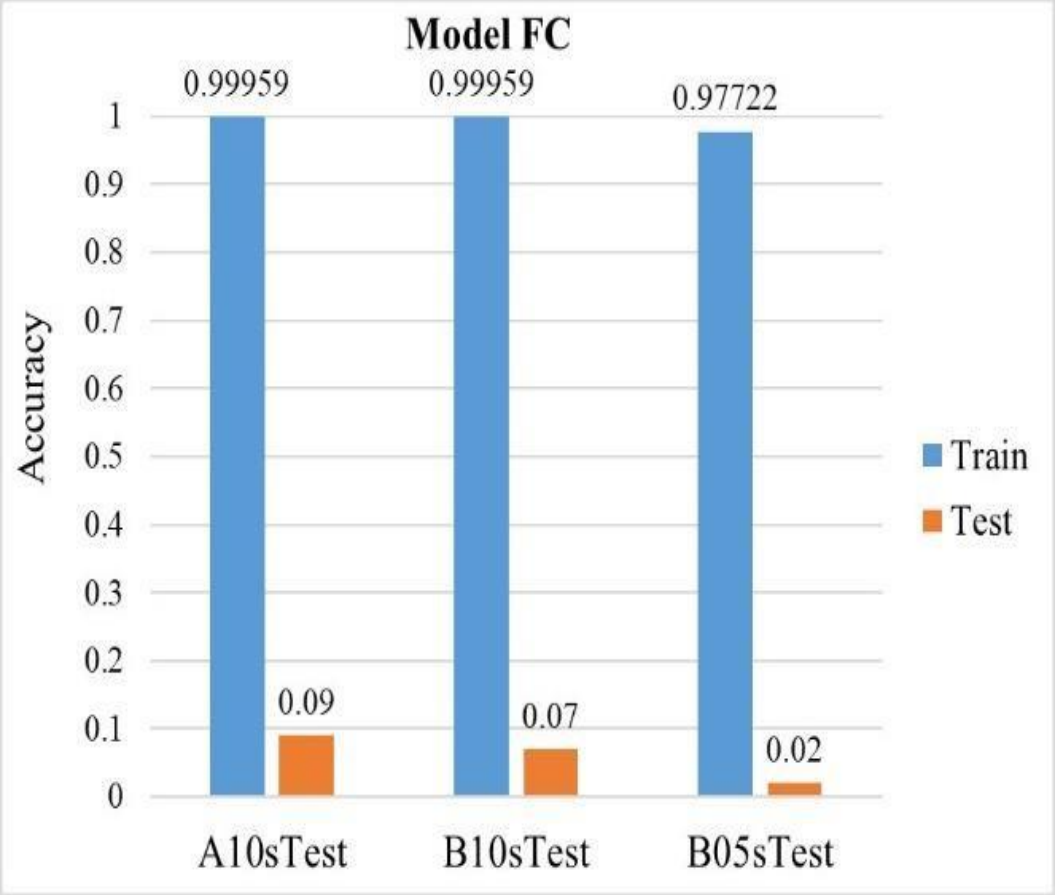
Trong **Bảng 4. 5** bên dưới, chúng tôi trình bày kết quả về hiệu suất phân loại bài hát từ một đoạn trích được cắt ngẫu nhiên từ 100 bài hát trong tập dữ liệu gốc "A" có độ dài 5 giây. Trong quá trình đào tạo, chúng tôi đã xây dựng tập dữ liệu huấn luyện trong tập dữ liệu "B" đoạn âm thanh có độ dài 5 giây. Mục tiêu của chúng tôi là đánh giá hiệu suất phân loại bài hát dựa trên độ dài của đoạn trích trong bài hát. Ngoài ra, chúng tôi đã áp dụng phương pháp là mô hình CNN/FC cho bộ dữ liệu này trong nhiệm vụ phân loại bài hát dựa trên dữ liệu âm thanh.

Bảng 4. 5: Hiệu suất phân loại bài hát của một đoạn trích từ bài hát được cắt ngẫu nhiên từ 100 bài hát trong tập dữ liệu gốc "B" có độ dài 5 giây

	Models	CNN		FC	
Roles	**Datasets**	**Accuracy**	**Time(seconds)**	**Accuracy**	**Time(seconds)**
Training set	A05strain	0.88752	1881.89070	0.97722	951.68599
Test set	B05stest	0.03000		0.02000	



Hình 4. 1: Hiệu suất phân loại bài hát mô hình CNN



Hình 4. 2: Hiệu suất phân loại bài hát mô hình FC

Trên **Hình 4. 2 và Hình 4. 2** trình bày hiệu suất phân loại bài hát trên mô hình CNN/FC dựa vào kết quả đã trình bày trong **Bảng 4. 3, Bảng 4. 4 và Bảng 4. 5.** Sự đa dạng các bộ thử nghiệm đã mang lại kết quả cho việc so sánh và phân tích. Kết quả mô hình FC đem lại trên bộ dữ liệu A10sTest có độ chính xác trên tập kiểm tra là 9%, B10sTest có độ chính xác 7% bên cạnh đó trên tập dữ liệu B5sTest kết quả kém hơn là 2%. Bên cạnh đó mô hình CNN cho kết quả đem lại trên bộ dữ liệu A10sTest có độ chính xác trên tập kiểm tra là 11%, B10sTest có độ chính xác 7% bên cạnh đó trên tập dữ liệu B5sTest kết quả kém hơn là 3%. Từ những kết quả đem lại sẽ là minh chứng cho việc lựa chọn phương pháp xử lý dữ liệu âm thanh cũng như lựa chọn kiến trúc và xử lý mạng tập dữ liệu phù hợp cho bài toán phân loại bài hát trình bày ở **Mục 4.3**.

4.2.2. Phân lớp trên dữ liệu âm thanh chuyển đổi thành ảnh phổ

a) Độ dài của đoạn trích từ bài hát có thể ảnh hưởng đến hiệu suất phân loại bài hát

Trong nghiên cứu, tập dữ liệu "A" cùng với các bản âm thanh có độ dài khác nhau như 1 giây, 3 giây, 5 giây, 10 giây, 20 giây, 30 giây, 60 giây và 90 giây. Các bản âm thanh này được cắt bớt từ các bài hát trong tập dữ liệu "A" để tạo ra một tập hợp đa dạng và toàn diện trong quá trình đào tạo mô hình. Ngoài ra, chúng tôi đã tạo ba bộ dữ liệu thử nghiệm khác nhau: bộ đầu tiên gồm 100 mẫu được cắt ngẫu nhiên từ 100 bài hát trong tập dữ liệu gốc "A" có độ dài 10 giây, tập 100 mẫu thứ hai được cắt ngẫu nhiên từ 100 bài hát trong tập dữ

liệu "B" cũng dài 10 giây và tập thứ ba bao gồm 100 đoạn cắt ngẫu nhiên từ 100 bài hát trong tập dữ liệu gốc "B" dài 5 giây. Sự đa dạng trong các bộ thử nghiệm này đã mang lại những kết quả thú vị, giúp chúng tôi có cơ sở để so sánh và phân tích. Việc lựa chọn các đoạn âm thanh có độ dài khác nhau đóng vai trò quan trọng trong việc tạo sự đa dạng và chân thực cho quá trình luyện tập. Điều này giúp mô hình tiếp xúc với nhiều tình huống âm nhạc khác nhau và tìm hiểu các đặc điểm âm nhạc quan trọng trong nhiều khung thời gian khác nhau. Điều này có ý nghĩa thực tiễn quan trọng vì người nghe thường có thể nghe ngẫu nhiên bất kỳ đoạn nhạc nào trong bài hát.

Ý tưởng của chúng tôi là kiểm tra xem bao nhiêu giây âm thanh sẽ phù hợp nhất với mô hình nhận dạng và có thể phân loại bài hát một cách chính xác nhất. Bằng cách chọn ngẫu nhiên các phân đoạn thời gian từ mỗi bài hát và đưa chúng vào mô hình, chúng tôi có thể xác định khoảng thời gian mà mô hình có thể xác định các đặc điểm quan trọng của bài hát. Chúng tôi nhận thấy rằng độ dài của một đoạn trích từ một bài hát có thể ảnh hưởng đáng kể đến hiệu suất phân loại bài hát. Điều này có ý nghĩa quan trọng trong việc triển khai thực tế vì người nghe thường có thể nghe ngẫu nhiên bất kỳ đoạn nhạc nào trong bài hát. Qua quá trình thử nghiệm này, chúng tôi sẽ có cái nhìn rõ ràng hơn về khả năng phân loại và nhận biết khi áp dụng các đoạn âm thanh khác nhau. Điều này giúp người mẫu được tiếp xúc với nhiều tình huống khác nhau và hiểu được các đặc điểm âm nhạc quan trọng trong nhiều khả năng thời gian khác nhau.

Tiếp theo, mô hình CNN1 và FC được xây dựng sau khi có được phổ dữ liệu đặc trưng ảnh quang phổ. Kiến trúc mạng thần kinh này được thiết kế đặc biệt để hoạt động với dữ liệu 2D. Ngoài ra, việc sử dụng mô hình FC giúp chúng tôi so sánh được khả năng tổng hợp và cách thức hoạt động trong giai đoạn kiểm thử. Kết quả được trình bày trong **Bảng 4. 6, Bảng 4. 8 và Bảng 4. 10** dưới đây:

- **Kết quả kiểm thử model CNN1/FC:**

Trong **Bảng 4. 6** dưới đây, chúng tôi trình bày kết quả về hiệu suất phân loại bài hát từ một đoạn trích được cắt ngẫu nhiên từ 100 bài hát trong tập dữ liệu gốc "A" có độ dài 10 giây. Trong quá trình đào tạo, chúng tôi đã xây dựng tập dữ liệu huấn luyện bằng cách kết hợp bản gốc của các bài hát trong tập dữ liệu "A", cùng với việc thêm các đoạn âm thanh có độ dài khác nhau, bao gồm 1 giây, 3 giây, 5 giây, 10 giây, 20 giây, 30 giây, 60 giây và 90 giây. Mục tiêu của chúng tôi là đánh giá hiệu suất phân loại bài hát dựa trên độ dài của đoạn trích trong bài hát. Ngoài ra, chúng tôi đã áp dụng hai phương pháp là mô hình CNN1 và FC cho bộ dữ liệu này để so sánh hiệu suất giữa hai kiến trúc mạng trong nhiệm vụ phân loại bài hát dựa trên dữ liệu âm thanh. Hơn nữa, chúng tôi nhấn mạnh các kết quả ấn tượng trên tập dữ liệu để tạo điều kiện thuận lợi cho quan sát và so sánh chính xác hiệu suất phân loại bài hát giữa mô hình CNN1 và FC.

Bảng 4. 6: Hiệu suất phân loại bài hát của một đoạn trích từ bài hát được cắt ngẫu nhiên từ 100 bài hát trong tập dữ liệu gốc "A" có độ dài 10 giây

	Models	**CNN1**		**FC**	
Roles	**Datasets**	**Accuracy**	**Time(seconds)**	**Accuracy**	**Time(seconds)**
Training set	A	1	30.08828	1	1.49742
Test set	A10stest	0.09000		0.03000	
Training set	A01strain	0.83187	9327.31516	0.66700	264.54783
Test set	A10stest	**0.78000**		**0.52000**	
Training set	A03strain	0.94899	5834.37020	0.79062	82.31451
Test set	A10stest	0.70000		0.45000	
Training set	A05strain	0.98232	3272.99490	0.75046	40.92374
Test set	A10stest	0.72000		0.44000	
Training set	A10strain	0.99872	2236.10187	0.86803	32.87552
Test set	A10stest	0.54000		0.32000	
Training set	A20strain	0.88471	1363.20650	0.88614	11.72585
Test set	A10stest	0.46000		0.22000	
Training set	A30strain	0.90288	4467.43876	0.92708	8.13230
Test set	A10stest	0.28000		0.14000	
Training set	A60strain	0.95632	1118.97285	0.55711	5.56170
Test set	A10stest	0.22000		0.07000	
Training set	A90strain	0.98263	441.26702	0.61843	7.03571
Test set	A10stest	0.14000		0.06000	

- **Đề xuất không sử dụng MaxPooling cho model CNN1:**

Bảng 4. 7: Hiệu suất phân loại bài hát của một đoạn trích từ bài hát được cắt ngẫu nhiên từ 100 bài hát trong tập dữ liệu gốc "A" có độ dài 10 giây

	Models	**CNN1**	
Roles	**Datasets**	**Accuracy**	**Time(seconds)**
Training set	A	1	50.57231
Test set	A10stest	0.07000	
Training set	A01strain	0.90153	7086.30159
Test set	A10stest	**0.73000**	
Training set	A03strain	0.98729	3104.47736
Test set	A10stest	0.66000	
Training set	A05strain	0.98879	1980.30008
Test set	A10stest	0.69000	
Training set	A10strain	0.99360	1211.94736
Test set	A10stest	0.56000	
Training set	A20strain	0.99113	987.49179
Test set	A10stest	0.38000	
Training set	A30strain	0.99020	488.22931
Test set	A10stest	0.20000	
Training set	A60strain	0.99792	264.24090
Test set	A10stest	0.19000	
Training set	A90strain	1	803.71535
Test set	A10stest	0.11000	

Trong **Bảng 4. 8** dưới đây, chúng tôi trình bày kết quả về hiệu suất phân loại bài hát từ một đoạn trích được cắt ngẫu nhiên từ 100 bài hát trong tập dữ liệu gốc "B" có độ dài 10 giây. Trong quá trình đào tạo, chúng tôi đã xây dựng tập dữ liệu huấn luyện bằng cách kết hợp bản gốc của các bài hát trong tập dữ liệu "A", cùng với việc thêm các đoạn âm thanh có độ dài khác nhau, bao gồm 1 giây, 3 giây, 5 giây, 10 giây, 20 giây, 30 giây, 60 giây và 90 giây. Mục tiêu của chúng tôi là đánh giá hiệu suất phân loại bài hát dựa trên độ dài của đoạn trích trong bài hát. Ngoài ra, chúng tôi đã áp dụng hai phương pháp là mô hình CNN1 và FC cho bộ dữ liệu này để so sánh hiệu suất giữa hai kiến trúc mạng trong nhiệm vụ phân loại bài hát dựa trên dữ liệu âm thanh. Hơn nữa, chúng tôi nhấn mạnh các kết quả ấn tượng trên tập dữ liệu để tạo điều kiện thuận lợi cho quan sát và so sánh chính xác hiệu suất phân loại bài hát giữa mô hình CNN1 và FC.

Bảng 4. 8: Hiệu suất phân loại bài hát của một đoạn trích từ bài hát được cắt ngẫu nhiên từ 100 bài hát trong tập dữ liệu gốc "B" có độ dài 10 giây

	Models	**CNN1**		**FC**	
Roles	**Datasets**	**Accuracy**	**Time(seconds)**	**Accuracy**	**Time(seconds)**
Training set	A	1	22.88092	0.99000	1.53298
Test set	B10stest	0.07000		0.03000	
Training set	A01strain	0.99340	2861.09413	0.75986	299.85230
Test set	B10stest	**0.21000**		**0.16000**	
Training set	A03strain	0.92056	883.15492	0.79060	81.15322
Test set	B10stest	0.20000		0.13000	
Training set	A05strain	0.98532	619.05232	0.75041	48.05398
Test set	B10stest	**0.21000**		0.15000	
Training set	A10strain	0.98574	398.18025	0.70118	32.92578
Test set	B10stest	0.19000		0.11000	
Training set	A20strain	0.99708	270.01730	0.85800	10.92764
Test set	B10stest	0.15000		0.10000	
Training set	A30strain	0.94880	353.66772	0.91408	8.81605
Test set	B10stest	0.14000		0.07000	
Training set	A60strain	0.97080	241.07309	0.46360	5.68537
Test set	B10stest	0.12000		0.05000	
Training set	A90strain	1	269.31917	0.50280	7.22746
Test set	B10stest	0.08000		0.06000	

- **Đề xuất không sử dụng MaxPooling cho model CNN1:**

Bảng 4. 9: Hiệu suất phân loại bài hát của một đoạn trích từ bài hát được cắt ngẫu nhiên từ 100 bài hát trong tập dữ liệu gốc "B" có độ dài 10 giây

	Models	**CNN1**	
Roles	**Datasets**	**Accuracy**	**Time(seconds)**
Training set	A	1	15.01418
Test set	B10stest	0.06000	
Training set	A01strain	0.67514	4858.55303
Test set	B10stest	**0.24000**	
Training set	A03strain	0.97379	1370.47453
Test set	B10stest	0.19000	
Training set	A05strain	0.97549	712.63858
Test set	B10stest	0.22000	
Training set	A10strain	0.98120	464.32316
Test set	B10stest	0.21000	
Training set	A20strain	0.99334	237.28927
Test set	B10stest	0.15000	
Training set	A30strain	0.95321	158.23603
Test set	B10stest	0.10000	
Training set	A60strain	0.97505	668.18812
Test set	B10stest	0.10000	
Training set	A90strain	0.99711	451.51238
Test set	B10stest	0.07000	

Trong **Bảng 4. 10** dưới đây, chúng tôi trình bày kết quả về hiệu suất phân loại bài hát từ một đoạn trích được cắt ngẫu nhiên từ 100 bài hát trong tập dữ liệu gốc "B" có độ dài 5 giây. Trong quá trình đào tạo, chúng tôi đã xây dựng tập dữ liệu huấn luyện bằng cách kết hợp bản gốc của các bài hát trong tập dữ liệu "A", cùng với việc thêm các đoạn âm thanh có độ dài khác nhau, bao gồm 1 giây, 3 giây, 5 giây, 10 giây, 20 giây, 30 giây, 60 giây và 90 giây. Mục tiêu của chúng tôi là đánh giá hiệu suất phân loại bài hát dựa trên độ dài của đoạn trích trong bài hát. Ngoài ra, chúng tôi đã áp dụng hai phương pháp là mô hình CNN1 và FC cho bộ dữ liệu này để so sánh hiệu suất giữa hai kiến trúc mạng trong nhiệm vụ phân loại bài hát dựa trên dữ liệu âm thanh. Hơn nữa, chúng tôi nhấn mạnh các kết quả ấn tượng trên tập dữ liệu để tạo điều kiện thuận lợi cho quan sát và so sánh chính xác hiệu suất phân loại bài hát giữa mô hình CNN1 và FC.

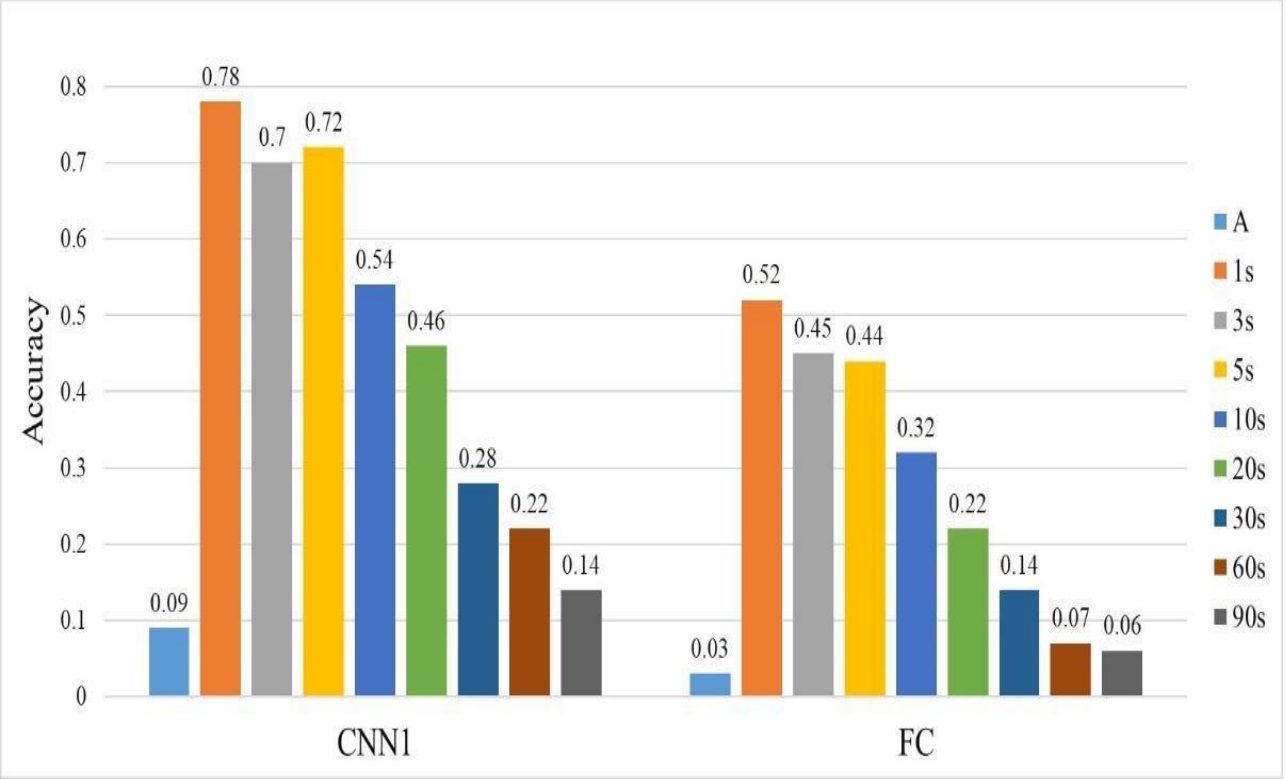
Bảng 4. 10: Hiệu suất phân loại bài hát của một đoạn trích từ bài hát được cắt ngẫu nhiên từ 100 bài hát trong tập dữ liệu gốc "B" có độ dài 5 giây

	Models	CNN1		FC	
Roles	**Datasets**	**Accuracy**	**Time(seconds)**	**Accuracy**	**Time(seconds)**
Training set	A	1	28.29685	0.94000	1.51251
Test set	B05stest	0.05000		0.03000	
Training set	A01strain	0.62363	2755.08256	0.68056	265.08607
Test set	B05stest	**0.20000**		0.17000	
Training set	A03strain	0.92059	825.94678	0.79068	75.16582
Test set	B05stest	0.18000		0.15000	
Training set	A05strain	0.97811	600.66608	0.78130	39.75371
Test set	B05stest	**0.20000**		**0.20000**	
Training set	A10strain	0.97516	341.46124	0.76508	30.49612
Test set	B05stest	0.16000		0.10000	
Training set	A20strain	0.97412	273.68412	0.86509	11.19394
Test set	B05stest	0.15000		0.09000	
Training set	A30strain	0.96300	374.43400	0.86505	8.55820
Test set	B05stest	0.11000		0.07000	
Training set	A60strain	1	105.92679	0.50518	5.46766
Test set	B05stest	0.12000		0.06000	
Training set	A90strain	0.99717	266.49906	0.92190	7.31094
Test set	B05stest	0.08000		0.06000	

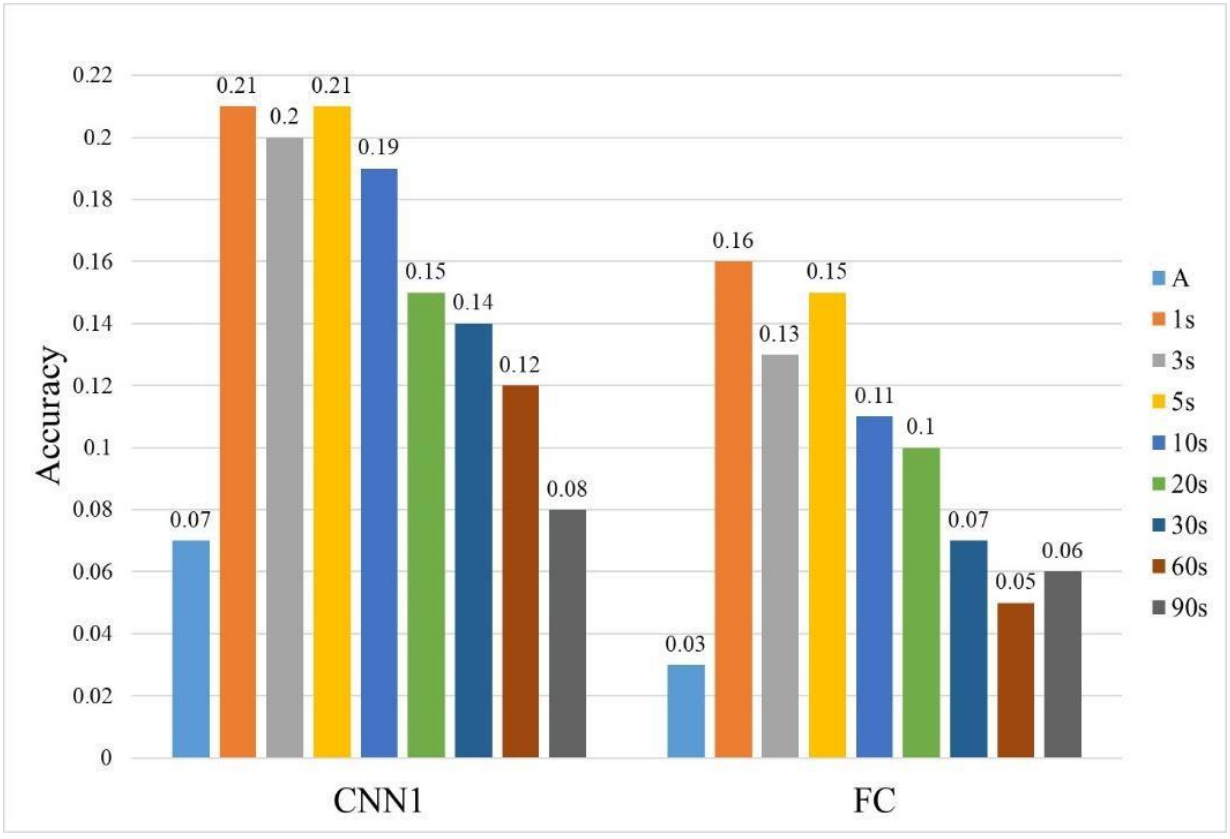
- **Đề xuất không sử dụng MaxPooling cho model CNN1:**

Bảng 4. 11: Hiệu suất phân loại bài hát của một đoạn trích từ bài hát được cắt ngẫu nhiên từ 100 bài hát trong tập dữ liệu gốc "B" có độ dài 5 giây

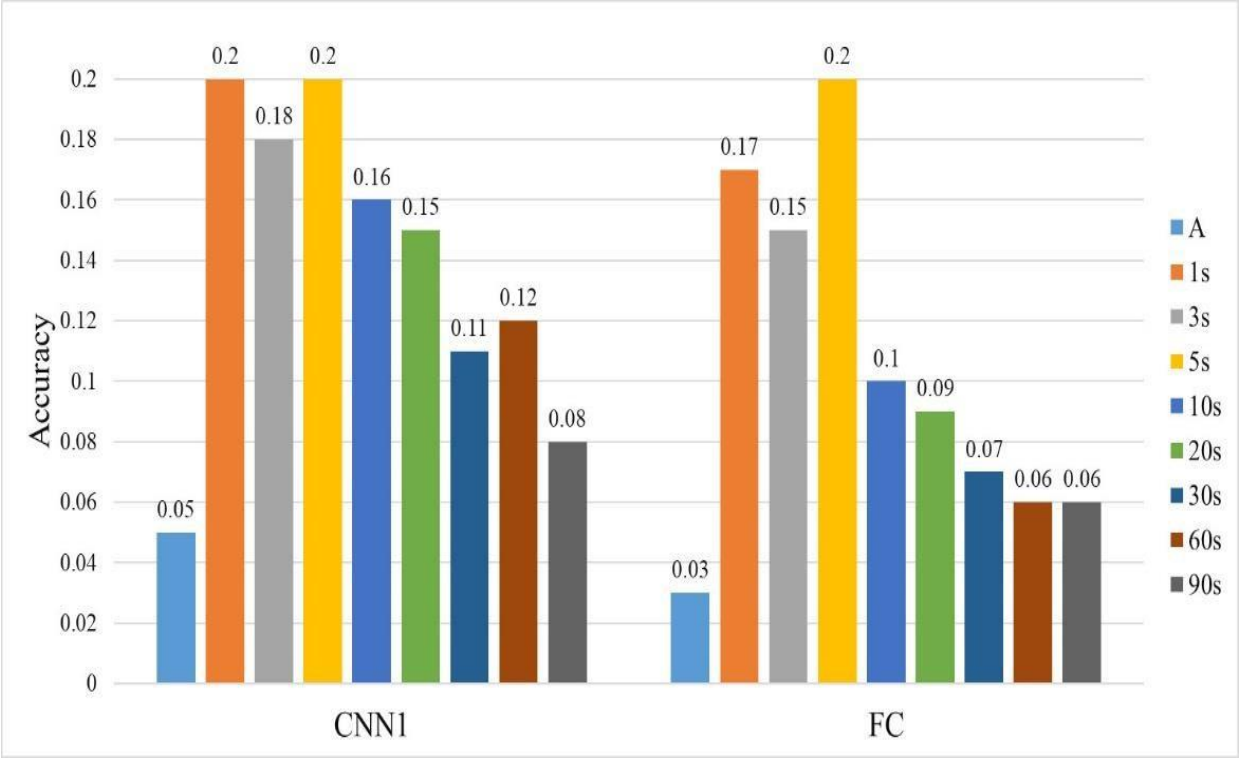
	Models	CNN1	
Roles	**Datasets**	**Accuracy**	**Time(seconds)**
Training set	A	1	24.31946
Test set	B05stest	0.08000	
Training set	A01strain	0.67514	3208.34429
Test set	B05stest	0.19000	
Training set	A03strain	0.98455	1068.13810
Test set	B05stest	**0.21000**	
Training set	A05strain	0.99259	722.82252
Test set	B05stest	0.23000	
Training set	A10strain	0.96165	405.01790
Test set	B05stest	0.20000	
Training set	A20strain	0.95343	214.54483
Test set	B05stest	0.14000	
Training set	A30strain	0.97714	293.59361
Test set	B05stest	0.09000	
Training set	A60strain	0.98960	684.65220
Test set	B05stest	0.10000	
Training set	A90strain	0.99132	442.69546
Test set	B05stest	0.11000	



Hình 4. 3: Kết quả đánh giá hiệu suất phân loại bài hát của một đoạn trích từ bài hát được cắt ngẫu nhiên từ 100 bài hát trong tập dữ liệu gốc "A" có độ dài 10 giây



Hình 4. 4: Kết quả đánh giá hiệu suất phân loại bài hát của một đoạn trích từ bài hát được cắt ngẫu nhiên từ 100 bài hát trong tập dữ liệu gốc "B" có độ dài 10 giây



Hình 4. 5: Kết quả đánh giá hiệu suất phân loại bài hát của một đoạn trích từ bài hát được cắt ngẫu nhiên từ 100 bài hát trong tập dữ liệu gốc "B" có độ dài 5 giây

Tiếp theo, chúng tôi đã so sánh hai phương pháp là mô hình CNN1 và FC trình bày **Bảng 4. 6, Bảng 4. 8 và Bảng 4. 10** hay biểu đồ cột thể hiện kết quả đánh giá của mô hình CNN1 và FC trình bày **Hình 4. 4, Hình 4. 5 và Hình 4. 5**. Sự đa dạng các bộ thử nghiệm đã mang lại kết quả vượt trội cho việc so sánh và đánh giá. Cả mô hình CNN1 và FC đều cho kết quả cao trên tập dữ liệu 1 giây, Tập dữ liệu kiểm thử (A10sTest) Mô hình FC đạt 52% bên cạnh đó mô hình CNN1 cho kết quả tốt hơn 78%. Bên cạnh đó Tập dữ liệu kiểm thử (B10sTest) mô hình FC đạt 16%, mô hình CNN1 cho kết quả tốt hơn 21%. Ngoài ra, chúng tôi còn thực nghiệm trên (B05sTest) mô hình FC đạt 20%, mô hình CNN1 đạt 20%. Khám phá này rất ấn tượng và thú vị cho việc phân loại bài hát vì trên thực tế, người nghe có thể nghe bất kì phần nào của bản nhạc một cách ngẫu nhiên. Bên cạnh đó, mô hình CNN1 vẫn thể hiện sự vượt trội so với FC, khả năng khái quát hóa và xử lý dữ liệu phức tạp của CNN1 khiến nó trở thành một công cụ phân bài hát mạnh mẽ.

Bên cạnh đó chúng tôi đề xuất sử dụng các model mạng nơ ron tích chập để phân tích tập ảnh phổ để trả về kết quả gồm có model CNN2 và model CNN3. Nhằm mục đích đánh giá và so sánh hiệu suất của các mô hình trong việc quét ảnh phổ và phân loại bài hát. Bằng cách này, chúng tôi có thể hiểu rõ hơn về cách mỗi mô hình xử lý thông tin âm thanh và khả năng phân loại các đoạn nhạc từ bộ dữ liệu đã chọn. Điều này có ý nghĩa quan trọng trong việc so sánh để xác định mô hình nào có hiệu suất tốt nhất trong việc phân loại bài hát và kiểm tra xem bao nhiêu giây âm thanh sẽ phù hợp nhất với mô hình một cách chính xác nhất. Hơn nữa, chúng tôi nhấn mạnh các kết quả ấn tượng trên tập dữ liệu để tạo điều kiện thuận lợi cho quan sát và so sánh chính xác hiệu suất phân loại bài hát giữa các mô hình. Kết quả được trình bày trong **Bảng 4. 12, Bảng 4. 14, Bảng 4. 16, Bảng 4. 18, Bảng 4. 20 và Bảng 4. 22** dưới đây:

- **Kết quả kiểm thử model CNN2:**

Bảng 4. 12: Hiệu suất phân loại bài hát của một đoạn trích từ bài hát được cắt ngẫu nhiên từ 100 bài hát trong tập dữ liệu gốc "A" có độ dài 10 giây

	Models	**CNN2**	
Roles	**Datasets**	**Accuracy**	**Time(seconds)**
Training set	A	1	19.36469
Test set	A10stest	0.06000	
Training set	A01strain	0.88856	6925.14607
Test set	A10stest	0.68000	
Training set	A03strain	0.94873	2741.23508
Test set	A10stest	0.69000	
Training set	A05strain	0.97853	1945.13292
Test set	A10stest	**0.73000**	
Training set	A10strain	0.99398	1385.57440
Test set	A10stest	0.62000	
Training set	A20strain	0.99113	881.32294
Test set	A10stest	0.36000	
Training set	A30strain	0.99238	482.63454
Test set	A10stest	0.20000	
Training set	A60strain	0.99584	272.36956
Test set	A10stest	0.19000	
Training set	A90strain	1	52.08077
Test set	A10stest	0.07000	

- **Đề xuất không sử dụng MaxPooling cho model CNN2:**

Bảng 4. 13: Hiệu suất phân loại bài hát của một đoạn trích từ bài hát được cắt ngẫu nhiên từ 100 bài hát trong tập dữ liệu gốc "A" có độ dài 10 giây

	Models	**CNN2**	
Roles	**Datasets**	**Accuracy**	**Time(seconds)**
Training set	A	1	34.65281
Test set	A10stest	0.07000	
Training set	A01strain	0.64493	8006.47439
Test set	A10stest	0.60000	
Training set	A03strain	0.81323	7786.50338
Test set	A10stest	**0.63000**	
Training set	A05strain	0.96562	3475.95747
Test set	A10stest	0.57000	
Training set	A10strain	0.85225	1887.94842
Test set	A10stest	0.54000	
Training set	A20strain	0.85291	1447.67689
Test set	A10stest	0.47000	
Training set	A30strain	0.93688	1232.25305
Test set	A10stest	0.28000	
Training set	A60strain	0.63617	263.64767
Test set	A10stest	0.19000	
Training set	A90strain	0.83815	158.88676
Test set	A10stest	0.16000	

Bảng 4. 14: Hiệu suất phân loại bài hát của một đoạn trích từ bài hát được cắt ngẫu nhiên từ 100 bài hát trong tập dữ liệu gốc "B" có độ dài 10 giây

	Models	**CNN2**	
Roles	**Datasets**	**Accuracy**	**Time(seconds)**
Training set	A	1	18.58217
Test set	B10stest	0.05000	
Training set	A01strain	0.99349	12747.62896
Test set	B10stest	0.19000	
Training set	A03strain	0.69993	2192.50731
Test set	B10stest	0.16000	
Training set	A05strain	0.98746	1432.57429
Test set	B10stest	0.20000	
Training set	A10strain	0.65037	717.73999
Test set	B10stest	**0.23000**	
Training set	A20strain	0.94161	817.81520
Test set	B10stest	0.13000	
Training set	A30strain	0.99782	471.96043
Test set	B10stest	0.08000	
Training set	A60strain	1	1393.22631
Test set	B10stest	0.10000	
Training set	A90strain	0.76589	925.25066
Test set	B10stest	0.08000	

Đề xuất không sử dụng MaxPooling cho model CNN2:

Bảng 4. 15: Hiệu suất phân loại bài hát của một đoạn trích từ bài hát được cắt ngẫu nhiên từ 100 bài hát trong tập dữ liệu gốc "B" có độ dài 10 giây

	Models	**CNN2**	
Roles	**Datasets**	**Accuracy**	**Time(seconds)**
Training set	A	1	29.65235
Test set	B10stest	0.04000	
Training set	A01strain	0.99364	7188.96293
Test set	B10stest	0.17000	
Training set	A03strain	0.99507	1915.17336
Test set	B10stest	**0.20000**	
Training set	A05strain	0.99506	1416.99897
Test set	B10stest	0.16000	
Training set	A10strain	0.66240	834.27992
Test set	B10stest	**0.20000**	
Training set	A20strain	0.99630	932.73243
Test set	B10stest	0.10000	
Training set	A30strain	0.99891	890.75699
Test set	B10stest	0.10000	
Training set	A60strain	0.72557	151.99271
Test set	B10stest	0.09000	
Training set	A90strain	0.99711	343.12301
Test set	B10stest	0.07000	

Bảng 4. 16: Hiệu suất phân loại bài hát của một đoạn trích từ bài hát được cắt ngẫu nhiên từ 100 bài hát trong tập dữ liệu gốc "B" có độ dài 5 giây

	Models	**CNN2**	
Roles	**Datasets**	**Accuracy**	**Time(seconds)**
Training set	A	1	17.57752
Test set	B05stest	0.06000	
Training set	A01strain	0.99364	11429.05290
Test set	B05stest	0.21000	
Training set	A03strain	0.95056	2229.15640
Test set	B05stest	0.19000	
Training set	A05strain	0.99430	1645.10290
Test set	B05stest	**0.23000**	
Training set	A10strain	0.99586	825.98367
Test set	B05stest	0.15000	
Training set	A20strain	0.92313	874.64034
Test set	B05stest	0.10000	
Training set	A30strain	0.95865	525.67595
Test set	B05stest	0.09000	
Training set	A60strain	0.81496	1456.69429
Test set	B05stest	0.09000	
Training set	A90strain	0.97976	933.29681
Test set	B05stest	0.08000	

Đề xuất không sử dụng MaxPooling cho model CNN2:

Bảng 4. 17: Hiệu suất phân loại bài hát của một đoạn trích từ bài hát được cắt ngẫu nhiên từ 100 bài hát trong tập dữ liệu gốc "B" có độ dài 5 giây

	Models	**CNN2**	
Roles	**Datasets**	**Accuracy**	**Time(seconds)**
Training set	A	1	30.20783
Test set	B05stest	0.04000	
Training set	A01strain	0.99345	7008.92388
Test set	B05stest	0.17000	
Training set	A03strain	0.99542	2006.20832
Test set	B05stest	0.13000	
Training set	A05strain	0.99525	1438.83172
Test set	B05stest	**0.22000**	
Training set	A10strain	0.94135	985.43317
Test set	B05stest	0.11000	
Training set	A20strain	0.99704	897.66513
Test set	B05stest	0.12000	
Training set	A30strain	0.99782	807.24976
Test set	B05stest	0.10000	
Training set	A60strain	0.75052	215.16731
Test set	B05stest	0.11000	
Training set	A90strain	0.84104	374.01006
Test set	B05stest	0.08000	

➢ **Kết quả kiểm thử model CNN3:**

Bảng 4. 18: Hiệu suất phân loại bài hát của một đoạn trích từ bài hát được cắt ngẫu nhiên từ 100 bài hát trong tập dữ liệu gốc "A" có độ dài 10 giây

	Models	**CNN3**	
Roles	**Datasets**	**Accuracy**	**Time(seconds)**
Training set	A	1	47.63179
Test set	A10stest	0.05000	
Training set	A01strain	0.99349	28152.80999
Test set	A10stest	0.68000	
Training set	A03strain	0.85225	4786.76237
Test set	A10stest	0.65000	
Training set	A05strain	0.99335	3242.29634
Test set	A10stest	**0.77000**	
Training set	A10strain	0.99210	2021.96437
Test set	A10stest	0.66000	
Training set	A20strain	0.99630	1405.30974
Test set	A10stest	0.49000	
Training set	A30strain	0.99129	1118.00821
Test set	A10stest	0.24000	
Training set	A60strain	0.99584	497.85835
Test set	A10stest	0.19000	
Training set	A90strain	0.93352	2213.79315
Test set	A10stest	0.09000	

- **Đề xuất không sử dụng MaxPooling cho model CNN3:**

Bảng 4. 19: Hiệu suất phân loại bài hát của một đoạn trích từ bài hát được cắt ngẫu nhiên từ 100 bài hát trong tập dữ liệu gốc "A" có độ dài 10 giây

	Models	**CNN3**	
Roles	**Datasets**	**Accuracy**	**Time(seconds)**
Training set	A	1	192.18815
Test set	A10stest	0.06000	
Training set	A01strain	0.65798	11746.26832
Test set	A10stest	0.56000	
Training set	A03strain	0.74170	7616.04523
Test set	A10stest	**0.76000**	
Training set	A05strain	0.89363	5631.59597
Test set	A10stest	0.71000	
Training set	A10strain	0.80864	5420.31584
Test set	A10stest	0.62000	
Training set	A20strain	0.87657	1599.33365
Test set	A10stest	0.51000	
Training set	A30strain	0.98367	1364.70751
Test set	A10stest	0.34000	
Training set	A60strain	0.41580	286.60541
Test set	A10stest	0.14000	
Training set	A90strain	0.99132	200.30155
Test set	A10stest	0.16000	

Bảng 4. 20: Hiệu suất phân loại bài hát của một đoạn trích từ bài hát được cắt ngẫu nhiên từ 100 bài hát trong tập dữ liệu gốc "B" có độ dài 10 giây

	Models	**CNN3**	
Roles	**Datasets**	**Accuracy**	**Time(seconds)**
Training set	A	1	33.71978
Test set	B10stest	0.04000	
Training set	A01strain	0.99376	20022.01427
Test set	B10stest	**0.25000**	
Training set	A03strain	0.99530	5499.92404
Test set	B10stest	0.17000	
Training set	A05strain	0.97473	3206.42217
Test set	B10stest	**0.25000**	
Training set	A10strain	0.56842	1629.16142
Test set	B10stest	0.19000	
Training set	A20strain	0.99704	1426.49945
Test set	B10stest	0.13000	
Training set	A30strain	0.83460	858.87496
Test set	B10stest	0.10000	
Training set	A60strain	0.80249	421.42144
Test set	B10stest	0.12000	
Training set	A90strain	1	2373.86132
Test set	B10stest	0.08000	

Đề xuất không sử dụng MaxPooling cho model CNN3:

Bảng 4. 21: Hiệu suất phân loại bài hát của một đoạn trích từ bài hát được cắt ngẫu nhiên từ 100 bài hát trong tập dữ liệu gốc "B" có độ dài 10 giây

	Models	**CNN3**	
Roles	**Datasets**	**Accuracy**	**Time(seconds)**
Training set	A	0.75999	156.26325
Test set	B10stest	0.05000	
Training set	A01strain	0.85602	8026.62791
Test set	B10stest	0.16000	
Training set	A03strain	0.99553	2802.10242
Test set	B10stest	0.14000	
Training set	A05strain	0.99506	2241.56251
Test set	B10stest	0.20000	
Training set	A10strain	0.99586	1220.19859
Test set	B10stest	**0.22000**	
Training set	A20strain	0.99630	1112.59380
Test set	B10stest	0.16000	
Training set	A30strain	0.99891	919.91661
Test set	B10stest	0.10000	
Training set	A60strain	0.38461	171.71813
Test set	B10stest	0.08000	
Training set	A90strain	0.99711	367.28822
Test set	B10stest	0.06000	

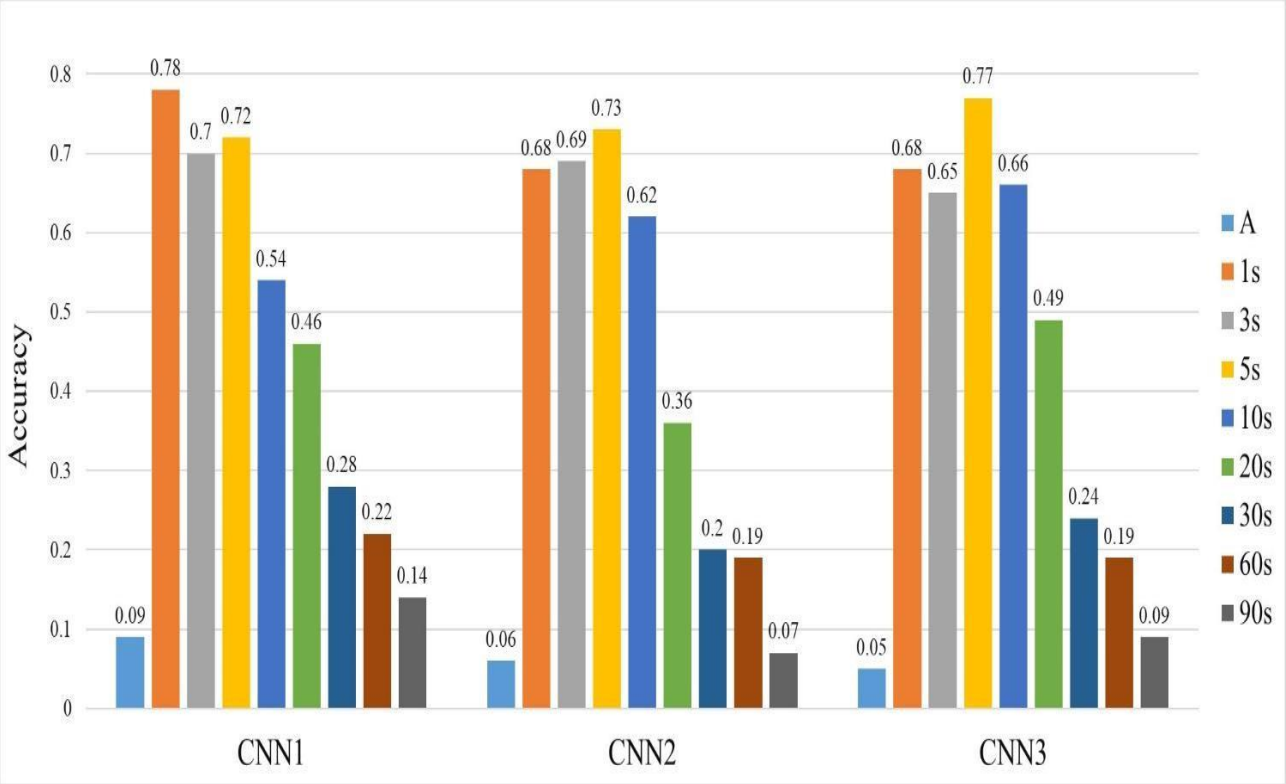
Bảng 4. 22: Hiệu suất phân loại bài hát của một đoạn trích từ bài hát được cắt ngẫu nhiên từ 100 bài hát trong tập dữ liệu gốc "B" có độ dài 5 giây

	Models	**CNN3**	
Roles	**Datasets**	**Accuracy**	**Time(seconds)**
Training set	A	1	33.25303
Test set	B05stest	0.04000	
Training set	A01strain	0.99330	23002.75389
Test set	B05stest	0.22000	
Training set	A03strain	0.73003	6676.48461
Test set	B05stest	0.17000	
Training set	A05strain	0.96163	3420.87633
Test set	B05stest	**0.25000**	
Training set	A10strain	0.99548	1522.49604
Test set	B05stest	0.21000	
Training set	A20strain	0.86622	1434.90717
Test set	B05stest	0.13000	
Training set	A30strain	0.99782	964.72054
Test set	B05stest	0.09000	
Training set	A60strain	0.99792	388.16388
Test set	B05stest	0.12000	
Training set	A90strain	1	2204.39398
Test set	B05stest	0.05000	

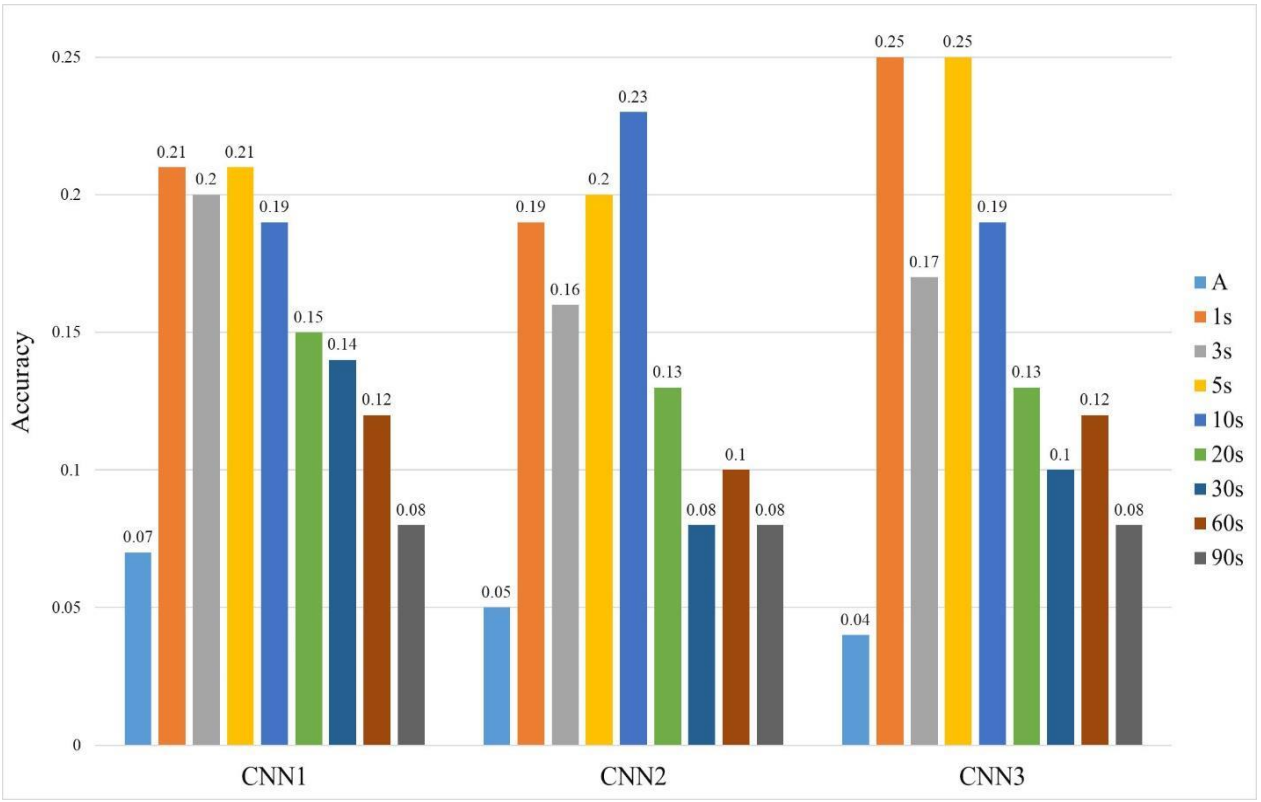
Đề xuất không sử dụng MaxPooling cho model CNN3:

Bảng 4. 23: Hiệu suất phân loại bài hát của một đoạn trích từ bài hát được cắt ngẫu nhiên từ 100 bài hát trong tập dữ liệu gốc "B" có độ dài 5 giây

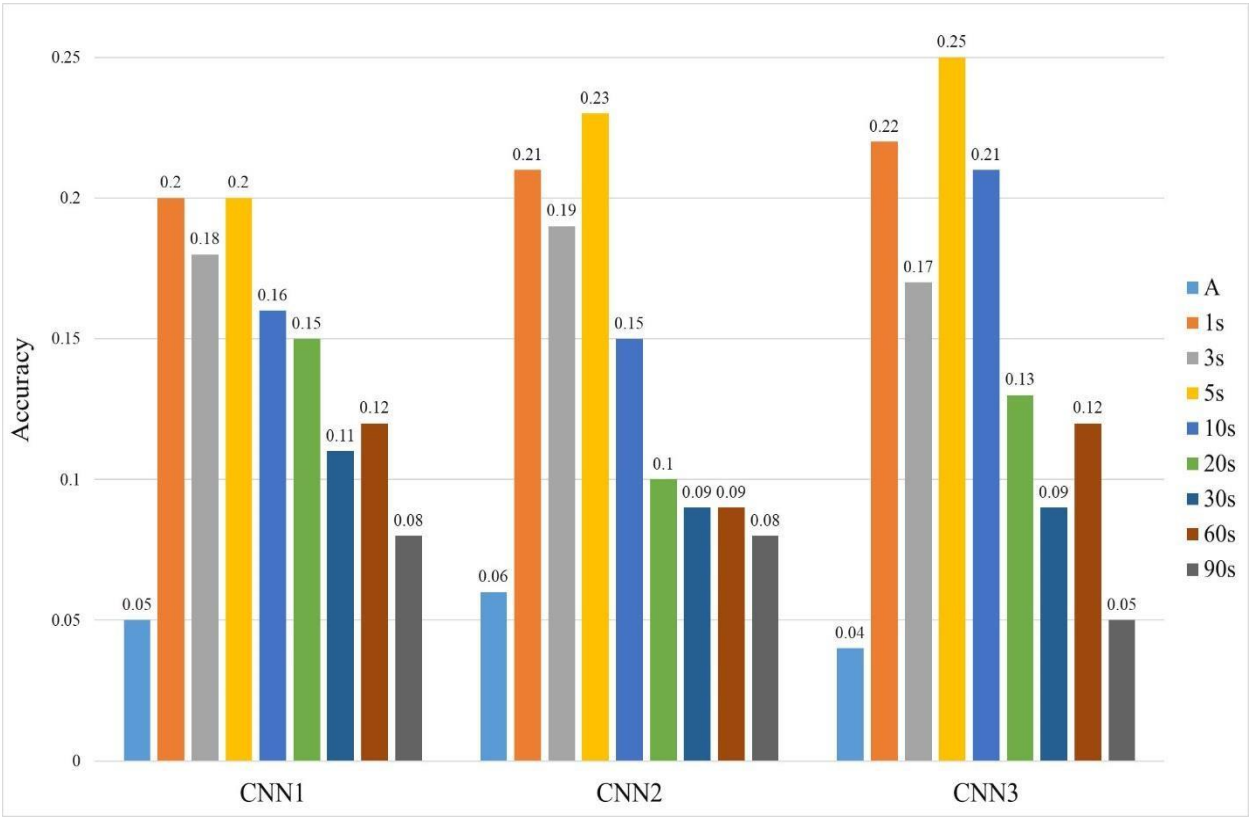
	Models	**CNN3**	
Roles	**Datasets**	**Accuracy**	**Time(seconds)**
Training set	A	0.65000	171.87230
Test set	B05stest	0.08000	
Training set	A01strain	0.93316	7839.65033
Test set	B05stest	0.17000	
Training set	A03strain	0.99542	2828.67322
Test set	B05stest	0.22000	
Training set	A05strain	0.99506	2121.47588
Test set	B05stest	**0.24000**	
Training set	A10strain	0.98308	1159.97878
Test set	B05stest	0.14000	
Training set	A20strain	0.99630	1280.85212
Test set	B05stest	0.17000	
Training set	A30strain	0.99782	1091.20614
Test set	B05stest	0.09000	
Training set	A60strain	0.14968	233.94862
Test set	B05stest	0.04000	
Training set	A90strain	1	390.54828
Test set	B05stest	0.05000	



Hình 4. 6: Kết quả đánh giá hiệu suất phân loại bài hát của một đoạn trích từ bài hát được cắt ngẫu nhiên từ 100 bài hát trong tập dữ liệu gốc "A" có độ dài 10 giây



Hình 4. 7: Kết quả đánh giá hiệu suất phân loại bài hát của một đoạn trích từ bài hát được cắt ngẫu nhiên từ 100 bài hát trong tập dữ liệu gốc "B" có độ dài 10 giây



Hình 4. 8: Kết quả đánh giá hiệu suất phân loại bài hát của một đoạn trích từ bài hát được cắt ngẫu nhiên từ 100 bài hát trong tập dữ liệu gốc "B" có độ dài 5 giây

Từ những kết quả thực nghiệm, biểu đồ cột thể hiện kết quả đánh giá của các mô hình CNN1, CNN2 và CNN3 trình bày ở **Hình 4. 7, Hình 4. 8 và Hình 4. 8** chúng tôi có thể thấy rằng kết quả hiệu suất phân loại cho ra kết quả đạt độ chính xác ấn tượng trên các đoạn nhạc từ bộ dữ liệu đã chọn.

- Kết quả đánh giá hiệu suất phân loại bài hát của một đoạn trích từ bài hát được cắt ngẫu nhiên từ 100 bài hát trong tập dữ liệu gốc "A" có độ dài 10 giây:
  - Mô hình CNN1 đạt 78% trên bộ kiểm thử 1s.
  - Mô hình CNN2 đạt 73% trên bộ kiểm thử 5s.
  - Mô hình CNN3 đạt 77% trên bộ kiểm thử 5s.
- Kết quả đánh giá hiệu suất phân loại bài hát của một đoạn trích từ bài hát được cắt ngẫu nhiên từ 100 bài hát trong tập dữ liệu gốc "B" có độ dài 10 giây:
  - Mô hình CNN1 đạt 21% trên bộ kiểm thử 1s và 5s.
  - Mô hình CNN2 đạt 23% trên bộ kiểm thử 10s.
  - Mô hình CNN3 đạt 25% trên bộ kiểm thử 1s và 5s.
- Kết quả đánh giá hiệu suất phân loại bài hát của một đoạn trích từ bài hát được cắt ngẫu nhiên từ 100 bài hát trong tập dữ liệu gốc "B" có độ dài 5 giây:
  - Mô hình CNN1 đạt 20% trên bộ kiểm thử 1s và 5s.
  - Mô hình CNN2 đạt 23% trên bộ kiểm thử 10s.
  - Mô hình CNN3 đạt 25% trên bộ kiểm thử 10s.

Bên cạnh đó chúng tôi đã nghiên cứu thực nghiệm trên các mô hình CNN1, CNN2 và CNN3 chúng tôi đã lược bỏ MaxPooling cho các mô hình để có kết quả minh chứng cho việc lựa chọn kiến trúc và xử lý mạng tập dữ liệu phù hợp cho bài toán phân loại bài hát. Kết quả trình bày **Bảng 4. 7, Bảng 4. 9, Bảng 4. 11, Bảng 4. 13, Bảng 4. 15, Bảng 4. 17, Bảng 4. 19, Bảng 4. 21 và Bảng 4. 23**. Kết quả khi lược bỏ Maxpooling không thực sự khả thi khi đem lại kết quả kém hơn. Điều này cho thấy vai trò quan trọng của MaxPooling trong việc cải thiện hiệu suất của mạng trong bài toán phân loại bài hát.

Nhằm mục đích đánh giá và so sánh hiệu suất của các mô hình trong việc quét ảnh phổ và phân loại bài hát. Bằng cách này, chúng tôi có thể hiểu rõ hơn về cách mỗi mô hình xử lý thông tin âm thanh và khả năng phân loại các đoạn nhạc từ bộ dữ liệu đã chọn. Điều này có ý nghĩa quan trọng trong việc so sánh để xác định mô hình nào có hiệu suất tốt nhất trong việc phân loại bài hát và kiểm tra xem bao nhiêu giây âm thanh sẽ phù hợp nhất với mô hình một cách chính xác nhất và cũng một lần nữa minh chứng về việc lựa chọn kiến trúc và xử lý mạng tập dữ liệu phù hợp cho bài toán phân loại bài hát.

b) Tăng cường dữ liệu về bài hát

Chúng tôi đã đề xuất đánh giá hiệu suất trên tập dữ liệu A trong cả hai trường hợp:

- **Khi không áp dụng tăng cường dữ liệu:** từ tập dữ liệu A.
- **Khi áp dụng tăng cường dữ liệu:** từ tập dữ liệu A gốc và dữ liệu bổ sung được thêm vào dưới dạng các đoạn có độ dài 1 giây, 3 giây, 5 giây, 10 giây, 20 giây, 30 giây, 60 giây và 90 giây.

Chúng tôi cũng đã thực hiện triển khai mô hình CNN1 và FC trên tập dữ liệu nâng cao để so sánh hiệu suất giữa hai kiến trúc mạng trong nhiệm vụ phân loại bài hát dựa trên dữ liệu âm thanh. Ví dụ: như trong

Bảng 4. 24 A03strain+A chứa các đoạn âm thanh có độ dài 3 giây, riêng biệt với tập dữ liệu A gốc đã được thêm vào. Tương tự, A01strain+A, A05strain+A, A10strain+A, A20strain+A, A30strain+A, A60strain+A, A90strain+A bao gồm tập dữ liệu gốc A được tăng cường dữ liệu là các mẫu của độ dài lần lượt là 1, 3, 5, 10, 20, 30, 60, 90 giây. Ngoài ra, tại cột **Enhanced Samples**, chúng tôi tính tỷ lệ phần trăm mẫu tăng, trong tập dữ liệu kiểm nghiệm là bao nhiêu %.

Bảng 4. 24: Thông tin về tập dữ liệu được sử dụng trong các thử nghiệm: AXs+A thử nghiệm bao gồm các mẫu trong đó mỗi mẫu là một đoạn có độ dài X giây được trích xuất ngẫu nhiên từ một bài hát trong tập dữ liệu A tương ứng cộng với A gốc, trong khi AY bao gồm chuỗi con của tệp âm thanh có độ dài Y giây được trích xuất từ A.

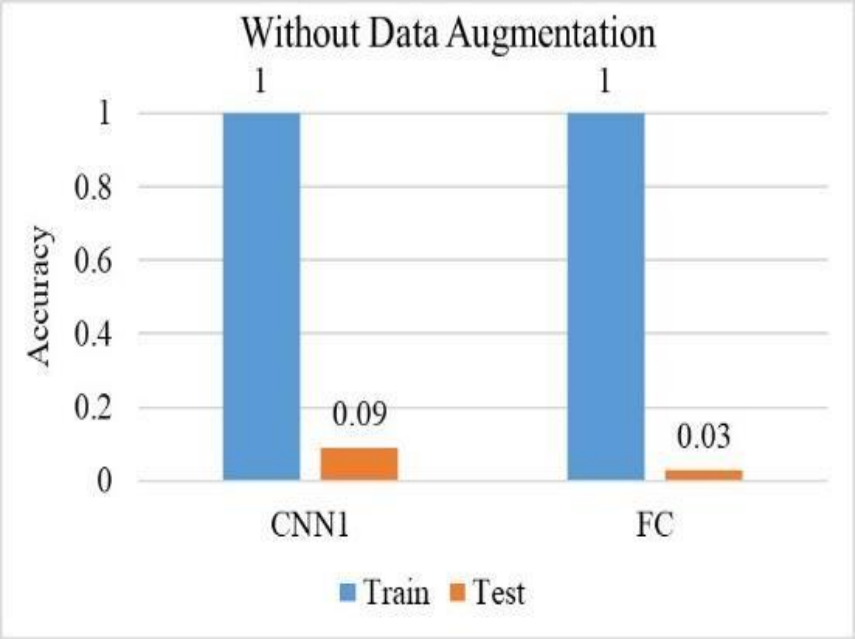
A with	Sample in A	Increased number	% Enhanced samples	Total
A01strain	100	25222	96.18644	26222
A03strain	100	8738	98.86855	8838
A05strain	100	5265	98.13602	5365
A10strain	100	2660	96.37683	2760
A20strain	100	1353	93.11764	1453
A30strain	100	919	90.18646	1019
A60strain	100	481	82.78826	581
A90strain	100	346	77.57847	446

Như được hiển thị trong **Bảng 4. 25** dưới đây, chúng tôi đã tiến hành đánh giá hiệu suất trên cùng một tập dữ liệu cả khi thực hiện tăng cường dữ liệu và khi không thực hiện tăng cường dữ liệu. Đồng thời, khi thực hiện tăng cường dữ liệu, mô hình CNN1 thể hiện sự vượt trội so với mô hình FC sau khi thực hiện nâng cao dữ liệu, nêu bật tầm quan trọng của việc lựa chọn kiến trúc và xử lý mạng tập dữ liệu phù hợp cho bài toán phân loại bài hát. Hơn nữa, chúng tôi nhấn mạnh các kết quả ấn tượng trên tập dữ liệu để tạo điều kiện thuận lợi cho quan sát và so sánh chính xác hiệu suất phân loại bài hát giữa mô hình CNN1 và FC.

Bảng 4. 25: Hiệu suất phân loại bài hát có sử dụng và không tăng cường dữ liệu

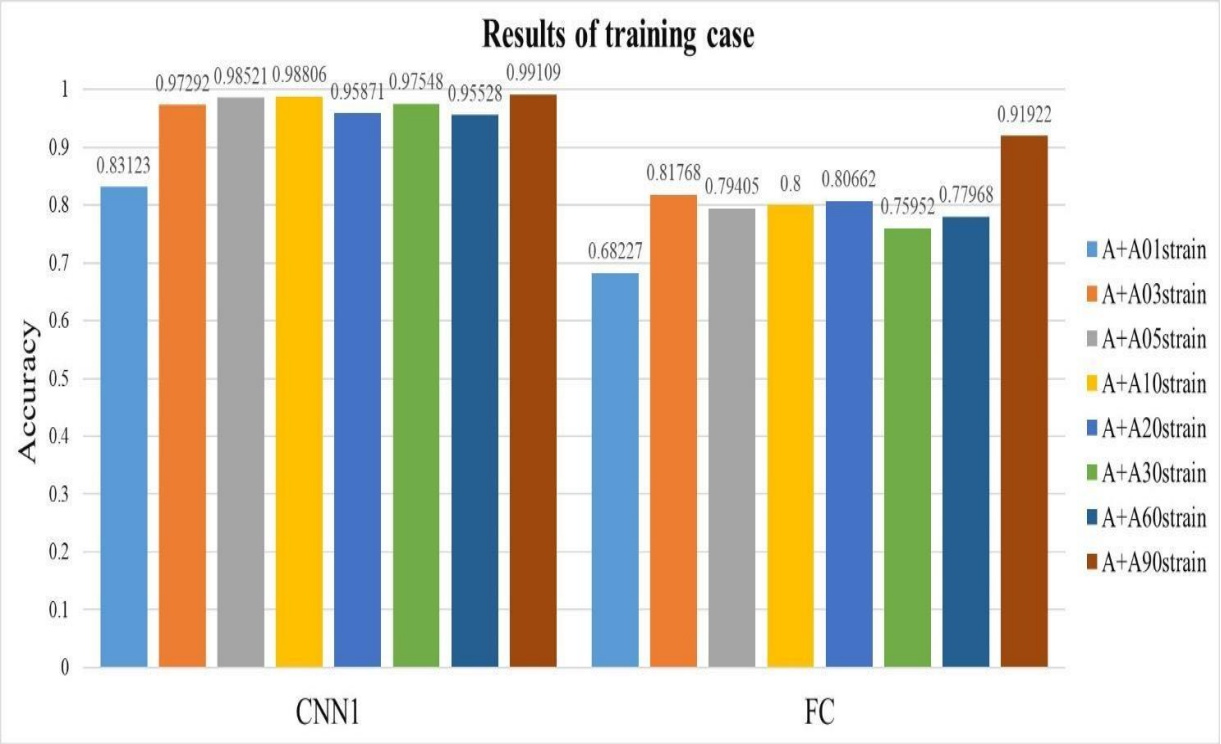
		CNN1		FC	
Roles	**Datasets**	**Accuracy**	**Time(seconds)**	**Accuracy**	**Time(seconds)**
Without data augmentation					
Training set	A	1	30.08828	1	1.49742
Test set	A10stest	0.09000		0.03000	
Using data augmentation					
Training set	A + A01strain	0.83123	5622.66006	0.68227	231.35512
Test set	A10stest	**0.80000**		**0.52000**	
Training set	A + A03strain	0.97292	1799.65929	0.81768	69.41635
Test set	A10stest	0.69000		0.48000	
Training set	A + A05strain	0.98521	1564.57873	0.79405	49.98381
Test set	A10stest	0.73000		0.50000	
Training set	A + A10strain	0.98806	847.49260	0.80000	31.08953
Test set	A10stest	0.55000		0.30000	
Training set	A + A20strain	0.95871	435.14626	0.80662	19.95024
Test set	A10stest	0.47000		0.22000	
Training set	A + A30strain	0.97548	167.96229	0.75952	9.96315
Test set	A10stest	0.33000		0.13000	
Training set	A + A60strain	0.95528	183.13637	0.77968	8.12120
Test set	A10stest	0.26000		0.08000	
Training set	A + A90strain	0.99109	472.36203	0.91922	5.90733
Test set	A10stest	0.17000		0.09000	

Đầu tiên, so sánh trong tập dữ liệu huấn luyện không sử dụng tăng cường dữ liệu, chúng tôi có thể thấy rằng kết quả rất thấp, trên mô hình CNN1 đạt 9% còn mô hình FC chỉ đạt 3%. Từ **Bảng 4. 25** hay biểu đồ cột thể hiện kết quả đánh giá của mô hình CNN1/FC trình bày ở **Hình 4. 9,** chúng tôi cũng có thể thấy, mặc dù CNN1 cho độc chính xác cao hơn FC, tuy nhiên điều này không có đem lại nhiều kết quả ấn tượng nếu triển khai thực tế. Nhưng bên cạnh đó, cho thấy mô hình CNN1 đem lại kết quả tốt hơn mô hình FC trong việc phân loại bài hát và cũng một lần nữa là minh chứng về việc lựa chọn kiến trúc và xử lý mạng tập dữ liệu phù hợp cho bài toán phân loại bài hát.

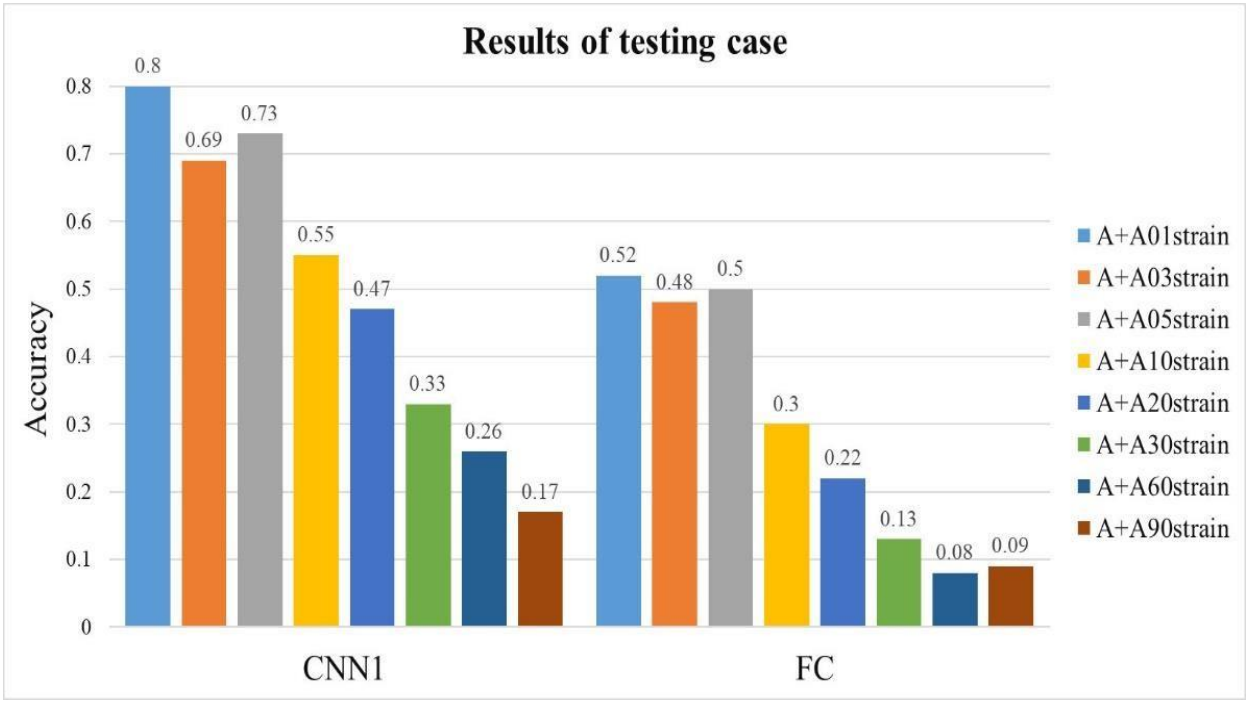


Hình 4. 9: Kết quả đánh giá việc không tăng cường dữ liệu của mô hình CNN1/FC

Bên cạnh đó, chúng tôi so sánh hiệu quả của phương pháp tăng cường dữ liệu, nhìn vào **Bảng 4. 25** hay biểu đồ cột thể hiện kết quả đánh giá của mô hình CNN1 và FC trình bày ở **Hình 4. 11 và Hình 4. 11** không khó để nhận ra rằng các mô hình được huấn luyện trên tập dữ liệu được áp dụng phương pháp tăng cường dữ liệu cho kết quả tốt hơn so với việc không áp dụng tăng cường dữ liệu. Độ chính xác của các mô hình đã được tăng trên cả tập huấn luyện và kiểm tra. Trên mô hình CNN1 đạt 80% bên cạnh đó mô hình FC cũng đạt kết quả ấn tượng 52%. Từ đó có thể thấy, phương pháp tăng cường dữ liệu thực sự hiệu quả trong việc xây dựng phương pháp phân loại bài hát. Đồng thời, khi thực hiện tăng cường dữ liệu, mô hình CNN1 thể hiện sự vượt trội so với mô hình FC sau khi thực hiện nâng cao dữ liệu, nêu bật tầm quan trọng của việc lựa chọn kiến trúc và xử lý mạng tập dữ liệu phù hợp cho bài toán phân loại bài hát.



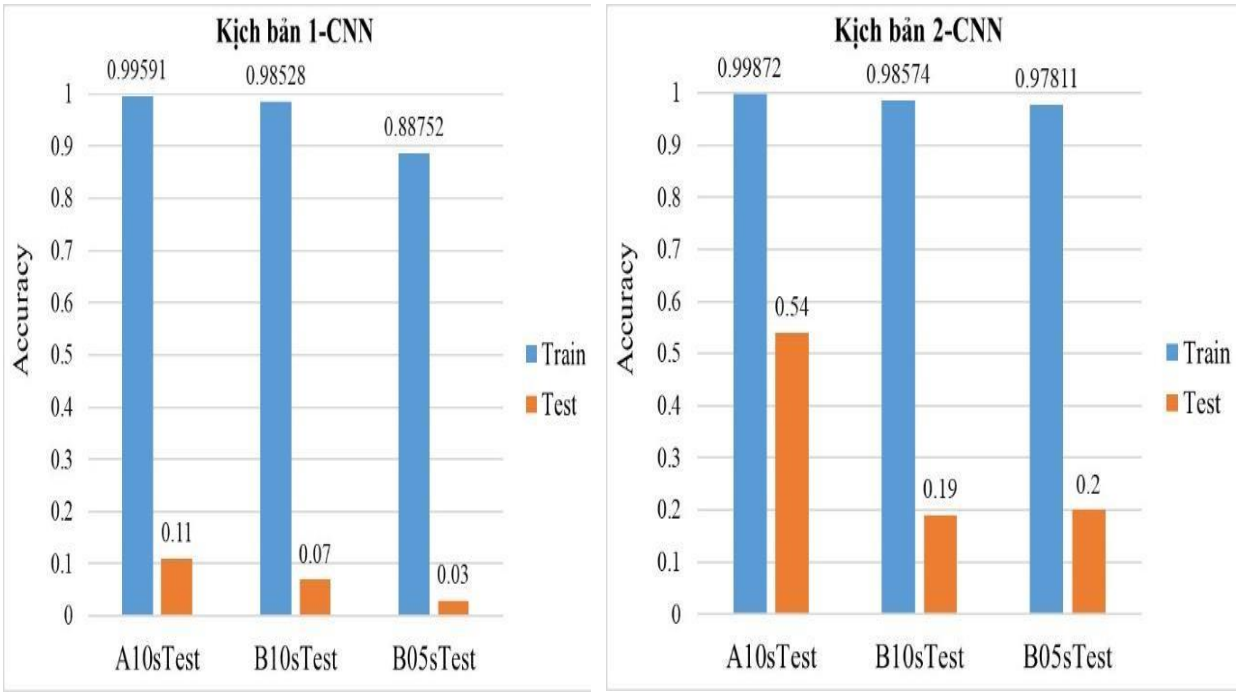
Hình 4. 10: Kết quả huấn luyện việc tăng cường dữ liệu của mô hình CNN1/FC



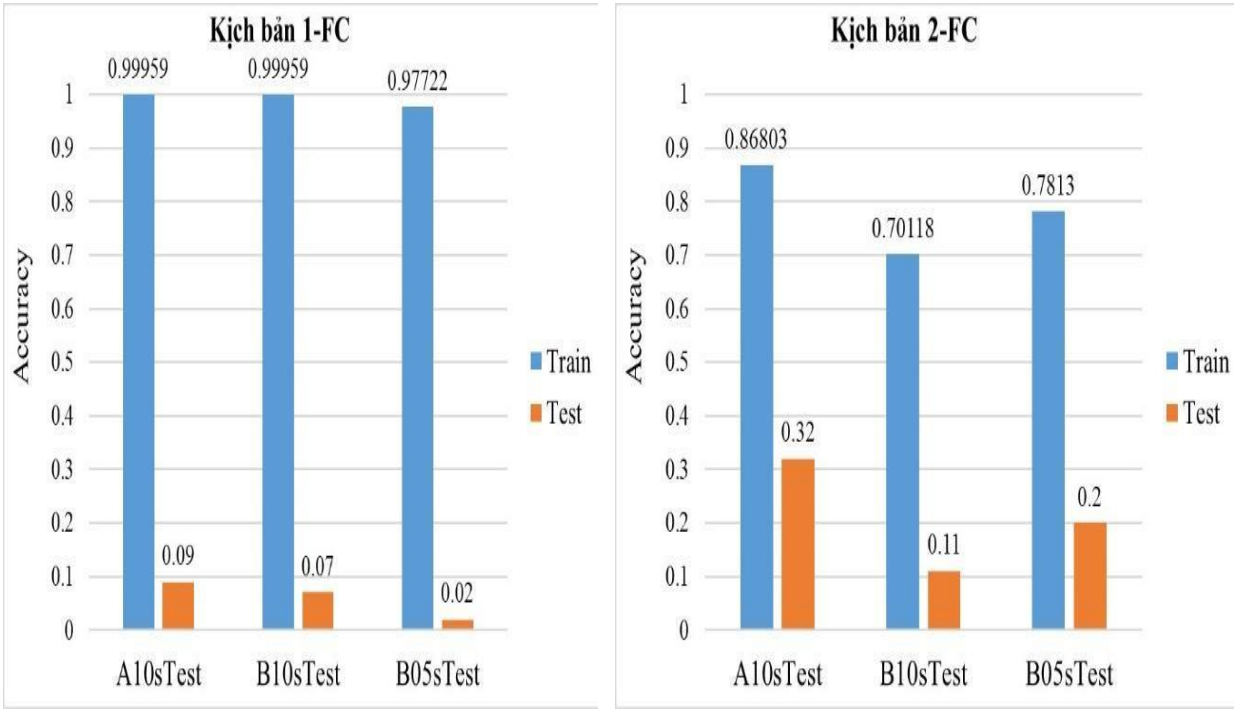
Hình 4. 11: Kết quả kiểm thử việc tăng cường dữ liệu của mô hình CNN1/FC

4.3. So sánh các thuật toán phân loại

Trong nghiên cứu này chúng tôi đã tiến hành thực nghiệm trên 2 kịch bản. Nhằm xác định tầm quan trọng của việc lựa chọn phương pháp xử lý dữ liệu âm thanh phù hợp là một yếu tố quan trọng để nâng cao hiệu suất của mô hình trong việc phân loại bài hát. Kết quả nghiên cứu là minh chứng rõ sự vượt trội khi lựa chọn phương pháp xử lý dữ liệu âm thanh phù hợp, cụ thể là sử dụng dữ liệu ảnh phổ âm thanh so với phương pháp trích xuất đặc trưng thành mảng Numpy. (Minh họa **Hình 4. 12 và Hình 4. 13**).



Hình 4. 12: So sánh hai phương pháp xử lý dữ liệu âm thanh được đánh giá bởi mô hình CNN



Hình 4. 13: So sánh hai phương pháp xử lý dữ liệu âm thanh được đánh giá bởi mô hình FC

Chúng tôi thực nghiệm trên cùng bộ dữ liệu với 2 kịch bản có thể thấy rõ rằng hiệu suất trên kịch bản 1 (trình bày ở **Bảng 4. 3, Bảng 4. 4 và Bảng 4. 5**) hay biểu đồ cột thể hiện kết quả đánh giá của mô hình CNN/FC trình bày ở **Hình 4. 2 và Hình 4. 2** không đem lại kết quả khả thi trong khi kết quả mô hình FC đem lại trên bộ dữ liệu A10sTest có độ chính xác trên tập kiểm tra là 9%, B10sTest có độ chính xác 7% bên cạnh đó trên tập dữ liệu B5sTest kết quả kém hơn là 2%. Bên cạnh đó mô hình CNN cho kết quả đem lại trên bộ dữ liệu A10sTest có độ chính xác trên tập kiểm tra là 11%, B10sTest có độ chính xác 7% bên cạnh đó trên tập dữ liệu B5sTest kết quả kém hơn là 3%.

Ngược lại, kết quả kịch bản 2 đem lại sự vượt trội hơn hẳn. Tập dữ liệu kiểm thử (A10sTest) Mô hình FC đạt 32% bên cạnh đó mô hình CNN cho kết quả tốt hơn 54%. Bên cạnh đó Tập dữ liệu kiểm thử (B10sTest) mô hình FC đạt 11%, mô hình CNN1 cho kết quả tốt hơn 19%. Ngoài ra, chúng tôi còn thực nghiệm trên (B05sTest) mô hình FC đạt 20%, mô hình CNN1 đạt 20%. Bên cạnh đó, ở kịch bản 2 chúng tôi đề xuất sử dụng các model mạng nơ ron tích chập để phân tích tập ảnh phổ để trả về kết quả gồm có model CNN2, model CNN3 trình bày ở ở **Hình 4. 7, Hình 4. 8 và Hình 4. 8.** Bên cạnh đó chúng tôi đã nghiên cứu thực nghiệm trên các mô hình CNN1, CNN2 và CNN3 chúng tôi đã lược bỏ MaxPooling cho các mô hình để có kết quả minh chứng cho việc lựa chọn kiến trúc và xử lý mạng tập dữ liệu phù hợp cho bài toán phân loại bài hát. Kết quả trình bày **Bảng 4. 7, Bảng 4. 9, Bảng 4. 11, Bảng 4. 13, Bảng 4. 15, Bảng 4. 17, Bảng 4. 19, Bảng 4. 21 và Bảng 4. 23**. Kết quả khi lược bỏ Maxpooling không thực sự khả thi khi đem lại kết quả kém hơn. Điều này cho thấy vai trò quan trọng của MaxPooling trong việc cải thiện hiệu suất của mạng trong bài toán phân loại bài hát.

Ngoài các kết quả kiểm tra trên, chúng tôi còn đánh giá hiệu suất trên cả hai trường hợp: không áp dụng tăng cường dữ liệu và áp dụng tăng cường dữ liệu. Trong tập dữ liệu huấn luyện không sử dụng tăng cường dữ liệu, có thể thấy rằng kết quả rất thấp, trên mô hình CNN1 đạt 9% còn mô hình FC chỉ đạt 3%. Trong khi mô hình được huấn luyện trên tập dữ liệu được áp dụng phương pháp tăng cường dữ liệu cho kết quả tốt hơn so với việc không áp dụng tăng cường dữ liệu. Độ chính xác của các mô hình đã được tăng trên cả tập huấn luyện và kiểm tra. Trên mô hình CNN1 đạt 80% bên cạnh đó mô hình FC cũng đạt kết quả ấn tượng 52%. Kết quả cho thấy việc tăng cường dữ liệu bằng cách chia tập dữ liệu thành các phần nhỏ dựa trên độ dài đã cải thiện đáng kể hiệu suất phân loại so với việc không sử dụng kỹ thuật này. Điều này góp phần đáng kể vào quá trình phân loại bài hát và tạo ra sự gia tăng đáng kể về hiệu suất của mô hình phân loại. Ngoài các nghiên cứu trên, chúng tôi đã trình bày kết quả thử nghiệm trên các mô hình CNN1, CNN2, CNN3 và FC, cung cấp cái nhìn chi tiết về so sánh giữa kiến trúc mạng trong nhiệm vụ phân loại bài hát dựa trên dữ liệu âm thanh. Chúng tôi nhận thấy rằng FC có tiềm năng tổng hợp thấp hơn và dẫn đến hiệu suất kém hơn CNN trên bộ thử nghiệm, đây là một phát hiện quan trọng. Khả năng mở rộng của phương pháp đề xuất là một mối quan tâm đáng kể. Với khả năng số lượng lớp học có thể thay đổi khi có thêm bài hát mới, bài viết sẽ đề cập đến cách mô hình có thể thích ứng với tình huống này. Công việc trong tương lai có thể khám phá các giải pháp cho thách thức về khả năng mở rộng này.

Từ những kết quả đến từ 2 kịch bản trên có thể thấy rõ sự vượt trội hơn hẳn khi lựa chọn phương pháp xử lý dữ liệu âm thanh đóng vai trò quan trọng trong quá trình huấn luyện

mô hình. Kịch bản 2 - sử dụng dữ liệu đầu vào là ảnh phổ đã đem lại hiệu suất vượt trội hơn kịch bản 1- sử dụng trích xuất đặc trưng âm thanh thành mảng Numpy. Như vậy, việc sử dụng mạng nơ-ron tích chập và lựa chọn phương pháp xử lý dữ liệu âm thanh phù hợp là một yếu tố quan trọng để nâng cao hiệu suất của mô hình trong việc phân loại bài hát. Kết quả này cung cấp hướng đi cụ thể và hữu ích cho các nghiên cứu và ứng dụng trong lĩnh vực xử lý âm thanh và nhận dạng bài hát.

4.4. Tổng kết chương

Chương này đã trình bày các phương pháp đánh giá, kịch bản kiểm thử và kết quả kiểm thử của việc xây dựng các mô hình trong việc phân loại bài hát. Qua đó, có thể thấy nghiên cứu đã có những kết quả thú vị bên cạnh việc sử dụng mạng nơ-ron tích chập và lựa chọn phương pháp xử lý dữ liệu âm thanh phù hợp là một yếu tố quan trọng để nâng cao hiệu suất của mô hình trong việc phân loại bài hát. Thông qua so sánh, chúng tôi phát hiện ra rằng CNN có khả năng tích hợp cao hơn và hoạt động tốt hơn FC trong giai đoạn thử nghiệm. Đây là phát hiện quan trọng khả năng mở rộng của phương pháp đề xuất là một mối quan tâm đáng kể.

Chương tiếp theo trình bày về kết quả đạt được, hạn chế và hướng phát triển của đề tài.

CHƯƠNG 5. KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN

5.1. Kết luận

5.1.1 Kết quả đạt được

Sau nhiều tháng nghiên cứu và thực hiện đề tài "***Tìm kiếm nhạc không lời sử dụng các phương pháp trực quan phổ tần số và Mạng nơ-ron tích chập***" đã đạt được những kết quả như sau:

– Chúng tôi đã thực hiện nghiên cứu nhằm xác định tầm quan trọng của việc lựa chọn phương pháp xử lý dữ liệu âm thanh phù hợp là một yếu tố quan trọng để nâng cao hiệu suất của mô hình trong việc phân loại bài hát. Kết quả từ việc thực nghiệm nghiên cứu qua hai mô hình CNN/FC trên cùng bộ dữ liệu qua 2 kịch bản, cho thấy rõ sự vượt trội khi lựa chọn phương pháp xử lý dữ liệu âm thanh phù hợp, cụ thể là sử dụng dữ liệu ảnh phổ âm thanh so với phương pháp trích xuất đặc trưng thành mảng Numpy. Kết quả nghiên cứu của chúng tôi cung cấp bằng chứng rõ ràng về tầm quan trọng của việc lựa chọn phương pháp xử lý dữ liệu âm thanh phù hợp. Nó đề cao vai trò của ảnh phổ âm thanh trong cải thiện hiệu suất của mô hình phân loại bài hát. Kết quả này có thể được áp dụng và ứng dụng trong các lĩnh vực liên quan đến xử lý âm thanh và nhận dạng tín hiệu âm thanh.

– Chúng tôi đề xuất phương pháp sử dụng mạng CNN để phân loại bài hát dựa trên phổ âm của đoạn nhạc. Ngoài ra, chúng tôi đã thực hiện nghiên cứu để xác định tầm quan trọng của việc chọn một bài hát thử nghiệm có số giây nhỏ so với toàn bộ bản nhạc. Kết quả cho thấy việc chọn một đoạn thử nghiệm ngắn hơn đã cải thiện độ chính xác của quá trình phân loại. Điều này có thể giải thích bởi việc các đoạn ngắn thường chỉ chứa một phần nhỏ của bài hát, tập trung vào các đặc điểm độc đáo của những bài hát cụ thể.Điều này chứng tỏ rằng tập trung vào các phần nhỏ của một bản nhạc có thể giúp mô hình học được đặc điểm cụ thể và phân loại của mỗi bài hát, từ đó cải thiện khả năng phân loại.

– Chúng tôi đề xuất rằng việc tăng cường dữ liệu bằng cách chia bài hát thành nhiều đoạn có thể cải thiện độ chính xác của quá trình phân loại và rằng CNN1 hoạt động tốt hơn FC trên cả hai bộ dữ liệu. Chúng tôi phát hiện rằng việc thực hiện tăng cường dữ liệu bằng cách chia bộ dữ liệu thành các đoạn cải thiện hiệu suất phân loại hơn so với việc không áp dụng kỹ thuật này. Phương pháp này đã tăng đáng kể hiệu suất của mô hình phân loại. Phương pháp tăng cường dữ liệu bao gồm bộ dữ liệu A và dữ liệu bổ sung được thêm vào dưới dạng các đoạn có độ dài khác nhau, giúp mô hình tập trung vào việc học và phân loại dựa trên đặc điểm độc đáo của mỗi phần dữ liệu trong mỗi bài hát. Kết quả dẫn đến một phân loại chính xác hơn so với trường hợp không sử dụng tăng cường dữ liệu.

– Bên cạnh đó, chúng tôi đã nghiên cứu thực nghiệm trên các mô hình CNN1, CNN2 và CNN3 chúng tôi đã lược bỏ MaxPooling cho các mô hình để có kết quả minh chứng cho việc lựa chọn kiến trúc và xử lý mạng tập dữ liệu phù hợp cho bài toán phân loại bài hát. Kết quả khi lược bỏ MaxPooling không thực sự khả thi khi đem lại kết quả kém hơn. Điều này cho thấy vai trò quan trọng của MaxPooling trong việc cải thiện hiệu suất của mạng trong bài toán phân loại bài hát.

– Cả hai mô hình CNN và FC được áp dụng trong nghiên cứu này. Thông qua so sánh, chúng tôi phát hiện ra rằng CNN có khả năng tích hợp cao hơn và hoạt động tốt hơn trong giai đoạn thử nghiệm. Đây là phát hiện quan trọng khả năng mở rộng của phương pháp đề xuất là một mối quan tâm đáng kể. Với khả năng số lượng lớp học có thể thay đổi khi thêm bài hát mới, bài viết sẽ đề cập đến cách mô hình có thể thích ứng với tình huống này. Công việc tương lai có thể khám phá các giải pháp cho thách thức về khả năng mở rộng này.

– Về mặt khoa học, những kết quả trong nghiên cứu này đã được đề xuất viết một bài báo "***An Approach to Instrumental Song Classification Utilizing Spectrogram and Convolutional Neural Networks***" và nhận được sự chấp nhận cho trình bày tại Hội nghị quốc tế về trí tuệ nhân tạo và Trí tuệ tính toán AICI'2024 (Hanoi, Vietnam, January 13-14, 2024), đồng thời sẽ được xuất bản bởi Springer.

5.1.2 Hạn chế

Sau thời gian nghiên cứu và thực hiện thì đề tài vẫn còn những hạn chế:

– Phương pháp sử dụng mạng CNN để phân loại bài hát dựa trên phổ âm của đoạn nhạc cho kết quả khá tốt, tuy nhiên nó chưa đạt đến mức ấn tượng. Có thể có một số nguyên nhân giải thích cho điều này, lí do có thể lựa chọn kiến trúc mô hình và cách xử lý dữ liệu đầu vào phù hợp, mô hình được xây dựng với đầu vào là hình ảnh có kích thước 32 x 32, nhỏ hơn nhiều so với kích thước do giới hạn cấu hình máy tính (500 pixels) có thể dẫn đến mất thông tin quan trọng trong quá trình huấn luyện.

– Xây dựng website tích hợp các mô hình đã huấn luyện.

5.2. Hướng phát triển

Chạy thực nghiệm các mô hình hiện có trên nhiều bộ tham số khác nhau, thay đổi kích thước ảnh, đồng thời nghiên cứu và xây dựng thêm các mô hình cho việc phân loại bài hát để cải thiện hiệu suất.

Phát triển thêm chức năng voice cho trang web. Xây dựng thêm nhiều chức năng khác cho trang web.

Tăng cường thu thập thêm nhiều dữ liệu để cho quá trình huấn luyện được tốt hơn.

Tiếp tục nghiên cứu để phát triển phần xử lý ảnh với CNN. Xây dựng các mô hình máy học tốt hơn.

TÀI LIỆU THAM KHẢO

[1] H. T. Nguyen, L. D. Vo, and T. T. Tran, "Approximate Nearest Neighbour-based Index Tree: A Case Study for Instrumental Music Search," *Applied Computer Systems*, vol. 28, no. 1, pp. 156–162, Jun. 2023, doi: 10.2478/acss-2023-0015.

[2] K. J. Piczak, "Environmental sound classification with convolutional neural networks," in *2015 IEEE 25th International Workshop on Machine Learning for Signal Processing (MLSP)*, Sep. 2015, pp. 1–6. doi: 10.1109/MLSP.2015.7324337.

[3] E. Benetos, S. Dixon, Z. Duan, and S. Ewert, "Automatic Music Transcription: An Overview," *IEEE Signal Processing Magazine*, vol. 36, pp. 20–30, Jan. 2019, doi: 10.1109/MSP.2018.2869928.

[4] A. Ghosal, R. Chakraborty, B. C. Dhara, and S. K. Saha, "Song/instrumental classification using spectrogram based contextual features," in *Proceedings of the CUBE International Information Technology Conference*, in CUBE '12. New York, NY, USA: Association for Computing Machinery, Tháng Chín 2012, pp. 21–25. doi: 10.1145/2381716.2381722.

[5] L. Nanni, G. Maguolo, and M. Paci, "Data augmentation approaches for improving animal audio classification," *Ecological Informatics*, vol. 57, p. 101084, May 2020, doi: 10.1016/j.ecoinf.2020.101084.

[6] T. Park and T. Lee, "Musical instrument sound classification with deep convolutional neural network using feature fusion approach," Dec. 2015.

[7] J. Salamon and J. P. Bello, "Deep Convolutional Neural Networks and Data Augmentation for Environmental Sound Classification," *IEEE Signal Processing Letters*, vol. 24, no. 3, pp. 279–283, Mar. 2017, doi: 10.1109/LSP.2017.2657381.

[8] "A robust deformed convolutional neural network (CNN) for image denoising - Zhang - 2023 - CAAI Transactions on Intelligence Technology - Wiley Online Library." Accessed: Oct. 21, 2023. [Online]. Available: https://ietresearch.onlinelibrary.wiley.com/doi/full/10.1049/cit2.12110

[9] "Working with wav files in Python using Pydub - GeeksforGeeks." Accessed: Oct. 21, 2023. [Online]. Available: https://www.geeksforgeeks.org/working-with-wav-files-in-python-using-pydub/

[10] "XỬ LÝ DỮ LIỆU ÂM THANH." Accessed: Nov. 24, 2023. [Online]. Available: https://viblo.asia/p/xu-ly-du-lieu-am-thanh-Qpmlezg95rd

[11] T. Hai Nguyen, E. Prifti, N. Sokolovska, and J.-D. Zucker, "Disease Prediction Using Synthetic Image Representations of Metagenomic Data and Convolutional Neural Networks," in *2019 IEEE-RIVF International Conference on Computing and Communication Technologies (RIVF)*, Mar. 2019, pp. 1–6. doi: 10.1109/RIVF.2019.8713670.