TRƯỜNG ĐẠI HỌC BÁCH KHOA HÀ NỘI

**VIỆN ĐIỆN TỬ - VIỄN THÔNG**


# ĐỒ ÁN

# TỐT NGHIỆP ĐẠI HỌC

**Đề tài:**

## XÂY DỰNG HỆ THỐNG PHÁT HIỆN ÂM THANH BẤT THƯỜNG SỬ DỤNG AUTOENCODERS

Sinh viên thực hiện: NGUYỄN MINH HIẾU

Lớp ĐT03 – K60

Giảng viên hướng dẫn: TS. NGUYỄN THỊ KIM THOA

TS. HÀN HUY DŨNG

Hà Nội, 6-2020

TRƯỜNG ĐẠI HỌC BÁCH KHOA HÀ NỘI

**VIỆN ĐIỆN TỬ - VIỄN THÔNG**


# ĐỒ ÁN

# TỐT NGHIỆP ĐẠI HỌC

**Đề tài:**

## XÂY DỰNG HỆ THỐNG PHÁT HIỆN ÂM THANH BẤT THƯỜNG SỬ DỤNG AUTOENCODERS

Sinh viên thực hiện: NGUYỄN MINH HIẾU

Lớp ĐT03 – K60

Giảng viên hướng dẫn: TS. NGUYỄN THỊ KIM THOA

TS. HÀN HUY DŨNG

Cán bộ phản biện:

Hà Nội, 6-2020

# ĐÁNH GIÁ QUYỂN ĐỒ ÁN TỐT NGHIỆP

(Dùng cho giảng viên hướng dẫn)

Tên giảng viên đánh giá: T.S Nguyễn Thị Kim Thoa, T.S Hàn Huy Dũng

Họ và tên sinh viên: Nguyễn Minh Hiếu MSSV: 20151336

Tên đồ án: Xây dựng hệ thống phát hiện âm thanh bất thường sử dụng Autoencoders

**Chọn các mức điểm phù hợp cho sinh viên trình bày theo các tiêu chí dưới đây:**
Rất kém (1); Kém (2); Đạt (3); Giỏi (4); Xuất sắc (5)

| | | | | | | |
|---|---|---|---|---|---|---|
| **Có sự kết hợp giữa lý thuyết và thực hành (20)** | | | | | | |
| 1 | Nêu rõ tính cấp thiết và quan trọng của đề tài, các vấn đề và các giả thuyết (bao gồm mục đích và tính phù hợp) cũng như phạm vi ứng dụng của đồ án | 1 | 2 | 3 | 4 | 5 |
| 2 | Cập nhật kết quả nghiên cứu gần đây nhất (trong nước/quốc tế) | 1 | 2 | 3 | 4 | 5 |
| 3 | Nêu rõ và chi tiết phương pháp nghiên cứu/giải quyết vấn đề | 1 | 2 | 3 | 4 | 5 |
| 4 | Có kết quả mô phỏng/thực nghiệm và trình bày rõ ràng kết quả đạt được | 1 | 2 | 3 | 4 | 5 |
| **Có khả năng phân tích và đánh giá kết quả (15)** | | | | | | |
| 5 | Kế hoạch làm việc rõ ràng bao gồm mục tiêu và phương pháp thực hiện dựa trên kết quả nghiên cứu lý thuyết một cách có hệ thống | 1 | 2 | 3 | 4 | 5 |
| 6 | Kết quả được trình bày một cách logic và dễ hiểu, tất cả kết quả đều được phân tích và đánh giá thỏa đáng | 1 | 2 | 3 | 4 | 5 |
| 7 | Trong phần kết luận, tác giả chỉ rõ sự khác biệt (nếu có) giữa kết quả đạt được và mục tiêu ban đầu đề ra đồng thời cung cấp lập luận để đề xuất hướng giải quyết có thể thực hiện trong tương lai | 1 | 2 | 3 | 4 | 5 |
| **Kỹ năng viết quyển đồ án (10)** | | | | | | |
| 8 | Đồ án trình bày đúng mẫu quy định với cấu trúc các chương logic và đẹp mắt (bảng biểu, hình ảnh rõ ràng, có tiêu đề, được đánh số thứ tự và được giải thích hay đề cập đến; căn lề thống nhất, có dấu cách sau dấu chấm, dấu phảy v.v.), có mở đầu chương và kết luận chương, có liệt kê tài liệu tham khảo và có trích dẫn đúng quy định | 1 | 2 | 3 | 4 | 5 |
| 9 | Kỹ năng viết xuất sắc (cấu trúc câu chuẩn, văn phong khoa học, lập luận logic và có cơ sở, từ vựng sử dụng phù hợp v.v.) | 1 | 2 | 3 | 4 | 5 |
| **Thành tựu nghiên cứu khoa học (5)** *(chọn 1 trong 3 trường hợp)* | | | | | | |
| 10a | Có bài báo khoa học được đăng hoặc chấp nhận đăng/Đạt giải SVNCKH giải 3 cấp Viện trở lên/Có giải thưởng khoa học (quốc tế hoặc trong nước) từ giải 3 trở lên/Có đăng ký bằng phát minh, sáng chế | 5 | | | | |
| 10b | Được báo cáo tại hội đồng cấp Viện trong hội nghị SVNCKH nhưng không đạt giải từ giải 3 trở lên/Đạt giải khuyến khích trong các kỳ thi quốc gia và quốc tế khác về chuyên ngành (VD: TI contest) | 2 | | | | |
| 10c | Không có thành tích về nghiên cứu khoa học | 0 | | | | |
| **Điểm tổng** | | **/50** | | | | |
| **Điểm tổng quy đổi về thang 10** | | | | | | |

***Nhận xét khác*** *(về thái độ và tinh thần làm việc của sinh viên)*

.....

.....

.....

.....

.....

.....

Ngày: … / … / 20…

**Người nhận xét**

(Ký và ghi rõ họ tên)

# ĐÁNH GIÁ QUYỂN ĐỒ ÁN TỐT NGHIỆP

(Dùng cho cán bộ phản biện)

Giảng viên đánh giá:.....

Họ và tên sinh viên: Nguyễn Minh Hiếu MSSV: 20151336

Tên đồ án: Xây dựng hệ thống phát hiện âm thanh bất thường sử dụng Autoencoders

**Chọn các mức điểm phù hợp cho sinh viên trình bày theo các tiêu chí dưới đây:**
Rất kém (1); Kém (2); Đạt (3); Giỏi (4); Xuất sắc (5)

| | | | | | | |
|---|---|---|---|---|---|---|
| **Có sự kết hợp giữa lý thuyết và thực hành (20)** | | | | | | |
| 1 | Nêu rõ tính cấp thiết và quan trọng của đề tài, các vấn đề và các giả thuyết (bao gồm mục đích và tính phù hợp) cũng như phạm vi ứng dụng của đồ án | 1 | 2 | 3 | 4 | 5 |
| 2 | Cập nhật kết quả nghiên cứu gần đây nhất (trong nước/quốc tế) | 1 | 2 | 3 | 4 | 5 |
| 3 | Nêu rõ và chi tiết phương pháp nghiên cứu/giải quyết vấn đề | 1 | 2 | 3 | 4 | 5 |
| 4 | Có kết quả mô phỏng/thực nghiệm và trình bày rõ ràng kết quả đạt được | 1 | 2 | 3 | 4 | 5 |
| **Có khả năng phân tích và đánh giá kết quả (15)** | | | | | | |
| 5 | Kế hoạch làm việc rõ ràng bao gồm mục tiêu và phương pháp thực hiện dựa trên kết quả nghiên cứu lý thuyết một cách có hệ thống | 1 | 2 | 3 | 4 | 5 |
| 6 | Kết quả được trình bày một cách logic và dễ hiểu, tất cả kết quả đều được phân tích và đánh giá thỏa đáng | 1 | 2 | 3 | 4 | 5 |
| 7 | Trong phần kết luận, tác giả chỉ rõ sự khác biệt (nếu có) giữa kết quả đạt được và mục tiêu ban đầu đề ra đồng thời cung cấp lập luận để đề xuất hướng giải quyết có thể thực hiện trong tương lai | 1 | 2 | 3 | 4 | 5 |
| **Kỹ năng viết quyển đồ án (10)** | | | | | | |
| 8 | Đồ án trình bày đúng mẫu quy định với cấu trúc các chương logic và đẹp mắt (bảng biểu, hình ảnh rõ ràng, có tiêu đề, được đánh số thứ tự và được giải thích hay đề cập đến; căn lề thống nhất, có dấu cách sau dấu chấm, dấu phảy v.v.), có mở đầu chương và kết luận chương, có liệt kê tài liệu tham khảo và có trích dẫn đúng quy định | 1 | 2 | 3 | 4 | 5 |
| 9 | Kỹ năng viết xuất sắc (cấu trúc câu chuẩn, văn phong khoa học, lập luận logic và có cơ sở, từ vựng sử dụng phù hợp v.v.) | 1 | 2 | 3 | 4 | 5 |
| **Thành tựu nghiên cứu khoa học (5)** *(chọn 1 trong 3 trường hợp)* | | | | | | |
| 10a | Có bài báo khoa học được đăng hoặc chấp nhận đăng/Đạt giải SVNCKH giải 3 cấp Viện trở lên/Có giải thưởng khoa học (quốc tế hoặc trong nước) từ giải 3 trở lên/Có đăng ký bằng phát minh, sáng chế | 5 | | | | |
| 10b | Được báo cáo tại hội đồng cấp Viện trong hội nghị SVNCKH nhưng không đạt giải từ giải 3 trở lên/Đạt giải khuyến khích trong các kỳ thi quốc gia và quốc tế khác về chuyên ngành (VD: TI contest) | 2 | | | | |
| 10c | Không có thành tích về nghiên cứu khoa học | 0 | | | | |
| **Điểm tổng** | | **/50** | | | | |
| **Điểm tổng quy đổi về thang 10** | | | | | | |

***Nhận xét khác của cán bộ phản biện***

.....

.....

.....

.....

.....

.....

Ngày: … / … / 20…

**Người nhận xét**

(Ký và ghi rõ họ tên)

# LỜI NÓI ĐẦU

Để đáp ứng các nhu cầu về an toàn công cộng đang ngày càng gia tăng trên thế giới hiện nay, các hệ thống giám sát dựa trên video đã được phát triển để có thể tự động phát hiện người hoặc vật thể khả nghi, cũng như các hệ thống an ninh dựa trên âm thanh có khả năng phát hiện những sự kiện âm thanh bất thường. Các hệ thống giám sát dựa trên video hiện hoạt động rất hiệu quả, tuy nhiên bị hạn chế về vị trí lắp đặt camera nên khó có thể giám sát toàn bộ khu vực. Mặt khác, giám sát bằng âm thanh không có hạn chế về điểm mù, và chi phí lắp đặt micro thu âm rẻ hơn nhiều so với camera. Do đó, các hệ thống an ninh dựa trên âm thanh có thể bù đắp nhược điểm của hệ thống camera, từ đó nếu kết hợp hệ thống giám sát bằng âm thanh và video với nhau có thể tạo nên những hệ thống thông minh hơn.

Hiện nay các công nghệ về trí thông minh nhân tạo đã được nghiên cứu phát triển và ứng dụng ở nhiều lĩnh vực trong cuộc sống. Tuy nhiên, nghiên cứu về các hệ thống phát hiện bất thường dựa trên âm thanh là một đề tài còn mới, chưa thực sự phổ biến như các hệ thống giám sát bằng camera. Đứng trên góc nhìn là sinh viên ngành Điện tử - Viễn thông và được sự gợi ý đề tài từ thầy Hàn Huy Dũng, em đã tìm hiểu, nghiên cứu về lĩnh vực phát hiện âm thanh bất thường. Sau một thời gian dài làm việc nghiêm túc, em đã quyết định chọn đề tài *Xây dựng hệ thống phát hiện âm thanh bất thường sử dụng Autoencoders* làm đồ án tốt nghiệp của mình.

Em xin được gửi lời cảm ơn chân thành nhất tới TS. Hàn Huy Dũng đã định hướng đề tài và trực tiếp hướng dẫn em thực hiện đồ án. Ngoài ra, em cũng xin gửi lời cảm ơn TS. Nguyễn Thị Kim Thoa đã đưa ra những lời khuyên bổ ích mặt kiến thức để em hoàn thành đồ án này. Trong quá trình thực hiện, các sai sót là không thể tránh khỏi, em rất mong nhận được ý kiến đóng góp của các thầy cô giáo để đề tài được hoàn thiện hơn.

Hà Nội, tháng 06 năm 2020

**Sinh viên thực hiện**

**Nguyễn Minh Hiếu**

# LỜI CAM ĐOAN

Tôi là Nguyễn Minh Hiếu, mã số sinh viên 20151336, sinh viên lớp Điện tử 03, khóa K60. Người hướng dẫn là TS. Nguyễn Thị Kim Thoa và TS. Hàn Huy Dũng. Tôi xin cam đoan toàn bộ nội dung được trình bày trong đồ án *Xây dựng hệ thống phát hiện âm thanh bất thường sử dụng Autoencoders* là kết quả quá trình tìm hiểu và nghiên cứu của tôi. Các dữ liệu được nêu trong đồ án là hoàn toàn trung thực, phản ánh đúng kết quả đo đạc thực tế. Mọi thông tin trích dẫn đều tuân thủ các quy định về sở hữu trí tuệ; các tài liệu tham khảo được liệt kê rõ ràng. Tôi xin chịu hoàn toàn trách nhiệm với những nội dung được viết trong đồ án này.

Hà Nội, tháng 06 năm 2020

**Người cam đoan**

**Nguyễn Minh Hiếu**

# MỤC LỤC

# DANH MỤC KÝ HIỆU VÀ CHỮ VIẾT TẮT

| Từ viết tắt | Từ viết đầy đủ | Nghĩa tiếng Việt |
|---|---|---|
| ASD | Anomalous Sound Detection | Phát hiện âm thanh bất thường |
| AI | Artificial Intelligence | Trí thông minh nhân tạo |
| PAL | Perceptron Learning Algorithm | Thuật toán học perceptron |
| MLP | Multi-layer Perceptron | Perceptron đa tầng |
| ML | Machine Learning | Học máy |
| DL | Deep Learning | Học sâu |
| ASC | Acoustic Scene Classification | Phân loại ngữ cảnh âm thanh |
| AED | Acoustic Event Detection | Phát hiện sự kiện âm thanh |
| GMM | Gaussian Mixture Model | Mô hình hỗn hợp Gaussian |
| SVM | Support Vector Machine | Máy vector hỗ trợ |
| ANN | Artificial Neural Network | Mạng neural nhân tạo |
| FNN | Feedforward Neural Network | Mạng neural truyền thẳng |
| GD | Gradient Descent | Suy giảm gradient |
| SGD | Stochastic Gradient Descent | Suy giảm gradient ngẫu nhiên |
| MFBs | Mel Filter Banks | Các bank bộ lọc Mel |
| MFCCs | Mel Frequency Cepstral Coefficents | Các hệ số cepstral tần số Mel |
| STFT | Short-time Fourier Transform | Biến đổi Fourier thời gian ngắn |
| DFT | Discrete Fourier Transform | Biến đổi Fourier rời rạc |
| FFT | Fast Fourier Transform | Biến đổi Fourier nhanh |
| DCT | Discrete Cosine Transform | Biến đổi cosin rời rạc |
| AE | Autoencoder | Bộ mã hóa tự động |
| VAE | Variational Autoencoder | Bộ mã hóa tự động biến thể |
| CNN | Convolutional Neural Network | Mạng neural tích chập |
| CAE | Convolutional Autoencoder | Bộ mã hóa tự động tích chập |
| RNN | Recurrent Neural Network | Mạng neural truy hồi |
| CRAE | Convolutional Recurrent Autoencoder | Bộ mã hóa truy hồi tích chập |
| LSTM | Long Short-term Memory | Bộ nhớ dài ngắn hạn |
| GRU | Gated Recurrent Unit | Đơn vị truy hồi theo cổng |
| ROC | Receiver Operating Characteristic | Đặc trưng hoạt động của bộ thu |
| AUC | Area under the ROC Curve | Diện tích dưới đường cong ROC |
| CVAE | Convolutional Variational Autoencoder | Bộ mã hóa tự động biến thể tích chập |

# DANH MỤC HÌNH VẼ

# DANH MỤC BẢNG BIỂU

# TÓM TẮT ĐỒ ÁN

Đồ án này trình bày về hệ thống phát hiện âm thanh bất thường sử dụng mạng Autoencoders. Các mạng Autoencoder có khả năng tái tạo các mẫu âm thanh bình thường, sự sai khác giữa âm thanh gốc và âm thanh tái tạo được xác định bằng lỗi tái tạo. Các mẫu âm thanh được tái táo với lỗi tái tạo lớn sẽ được xác định là âm thanh bất thường. Bên cạnh các mô hình Autoencoders thông thường, đồ án này còn đề xuất một mô hình mạng mới trong bài toán phát hiện âm thanh bất thường. Mô hình mạng được đề xuất là *Covolutional Recurrent Autoencoder,* một mô hình dựa trên *Convolutional Neural Network* và *Recurrent Neural Network.*

Các đặc trưng của âm thanh gốc được trích xuất bằng cách tính toán phổ công suất và sử dụng các bộ lọc Mel Filters − những bộ lọc được xây dựng dựa trên khả năng nghe của tai người. Các đặc trưng trích xuất được gọi là Mel Filter Banks. Sau khi được chuẩn hóa bằng tiền xử lí, những đặc trưng này được đưa vào một mạng neural. Mạng neural được huấn luyện để có thể tái tạo dữ liệu đầu vào với sai khác nhỏ nhất bằng cách tối thiểu hóa lỗi tái tạo. Cuối cùng, một ngưỡng được tính toán qua những thí nghiệm từ trước, khi các mẫu âm thanh có lỗi tái tạo lớn hơn ngưỡng này thì chúng được xác định là âm thanh bất thường.

Tập dữ liệu được sử dụng là MIMII và âm thanh được sử dụng là tiếng máy bơm và tiếng thanh ray trượt. Để đánh giá hiệu quả của mô hình đề xuất, mô hình được so sánh với một số mô hình mạng thông thường khác trong bài toán phát hiện âm thanh bất thường. Kết quả thí nghiệm cho thấy mô hình đề xuất vượt trội so với các mô hình thông thường, đạt F1 lần lượt bằng 0.8457 và 0.9696 cho tiếng máy bơm và tiếng thanh ray trượt. Ngoài ra, AUC cho hai loại âm thanh trên đạt 0.8308 và 0.935. Đồ án này cũng nêu một số nhận xét về các mô hình dựa vào kết quả các tham số đánh giá và trình bày hướng nghiên cứu trong tương lai.

# ABSTRACT

This thesis presents an anomalous sound detection system based on autoencoders. Autoencoders have the ability to reconstruct normal audio samples, the difference between raw audio and reconstructed audio is measured by reconstruction error. The reconstructed audio samples with high reconstruction error are considered as anomalous sounds. Besides conventional autoencoders, this thesis also proposes a new network model of anomalous sound detection. The proposed model is Covolutional Recurrent Autoencoder, a model based on *Convolutional Neural Network* and *Recurrent Neural Network*.

Raw audio features are extracted by computing the power spectrum and applying Mel filters – filters which are built based on the hearing ability of human ear. The extracted features are called Mel filter banks. After being normalized by preprocessing, the features are fed into a neural network. The neural network is trained to reconstruct input data with the smallest difference by minimizing the reconstruction error. Eventually, a threshold is calculated through preliminary experiments, when the audio samples have reconstruction errors exceeding this threshold, they are considered as anomalous sounds.

The MIMII data set is used and the sounds are pump sound and slide rail sound. In order to evaluate the performance of the proposed model, it is compared with some other conventional models of anomalous sound detection. The experimental results demonstrate that the proposed model outperforms the conventional models, reachs the F1 of 0.8457 and 0.9696 for pump sound and slide rail sound, respectively. Moreover, AUC for these two types of sound achieves 0.8308 and 0.935. This thesis also states a number of remarks based on the results of evaluation metrics and presents future work.

# PHẦN MỞ ĐẦU

Trong một vài năm trở lại đây, bài toán phát hiện âm thanh bất thường đã nhận được nhiều sự chú ý, tuy nhiên vẫn chưa thực sự phổ biến. Việc phát hiện âm thanh bất thường có thể giúp phát hiện các mối nguy hiểm tiềm ẩn và từ đó có biện pháp xử lí kịp thời. Có thể kể ra một số ví dụ ứng dụng thực tế của phát hiện âm thanh bất thường như: giám sát trong chăn nuôi dựa vào âm thanh động vật [1], [2], giám sát hoạt động công nghiệp dựa vào âm thanh của máy móc trong nhà máy [3], [4], giám sát đường phố dựa vào âm thanh ngoài đường [5], [6], [7], [8].

Với những tiến bộ gần đây của Deep Learning, các phương pháp dựa trên mạng neural đã được nghiên cứu để phát hiện âm thanh bất thường. Tư tưởng chính của những phương pháp này là huấn luyện một mạng neural được gọi là Autoencoder chỉ sử dụng dữ liệu âm thanh bình thường [9], [10]. Autoencoder sẽ mã hóa dữ liệu đầu vào vào không gian ẩn, sau đó giải mã để khôi phục dữ liệu ban đầu. Mô hình Autoencoder đã huấn luyện sẽ được sử dụng để phát hiện bất thường bằng các tính toán lỗi tái tạo, thể hiện sự khác nhau giữa dữ liệu đầu vào và đầu ra của Autoencoder. Nếu lỗi tái tạo cao hơn một ngưỡng nhất định (được xác định bằng các thí nghiệm từ trước) thì âm thanh tại đầu vào được xác định là bất thường.

Trong đồ án này, em trình bày một số mô hình Autoencoder thông thường đã được sử dụng trong lĩnh vực phát hiện bất thường, đồng thời đề xuất một mô hình Autoencoder mới có tên Convolutional Recurrent Autoencoder và so sánh độ hiệu quả với các mô hình trước đó thông qua các thí nghiệm. Báo cáo đồ án được trình bày theo thứ tự các chương như sau:

**Chương 1. Giới thiệu chung** – Trình bày tổng quan về bài toán phát hiện âm thanh bất thường, trình bày sơ lược về trí thông minh nhân tạo, Machine Learning, Deep Learning và giới thiệu các phương pháp hiện có để giải quyết bài toán phát hiện âm thanh bất thường.

**Chương 2. Cơ sở lí thuyết mạng neural** – Trình bày những kiến thức cốt lõi của mạng neural nói chung, đặc biệt là cơ sở toán học của việc huấn luyện mạng neural.

**Chương 3. Mô hình hệ thống phát hiện âm thanh bất thường** – Trình bày mô hình hệ thống phát hiện âm thanh bất thường, lí thuyết về xử lí tín hiệu, cách trích xuất đặc trưng của dữ liệu âm thanh cũng như khuôn mẫu chung cho mạng neural trong bài toán phát hiện âm thanh bất thường.

**Chương 4. Các mô hình mạng neural sử dụng trong bài toán phát hiện âm thanh bất thường** – Trình bày cụ thể, chi tiết về kiến trúc các mô hình mạng neural được xây dựng để phát hiện âm thanh bất thường trong phạm vi đồ án.

**Chương 5. Thí nghiệm và kết quả** – Trình bày về tập dữ liệu được sử dụng, các tham số đánh giá, thiết lập thí nghiệm, kết quả thí nghiệm so sánh các mô hình và rút ra một số nhận xét.

# CHƯƠNG 1. GIỚI THIỆU CHUNG

Chương 1 giới thiệu tổng quan về bài toán phát hiện âm thanh bất thường, trình bày sơ lược về trí thông minh nhân tạo, Machine Learning, Deep Learning và xem xét các phương pháp hiện có để giải quyết bài toán phát hiện âm thanh bất thường.

## 1.1 Bài toán phát hiện âm thanh bất thường

Trong cuộc sống thường ngày, chúng ta gặp phải rất nhiều sự kiện âm thanh đa dạng như tiếng chó sủa, tiếng bước chân, tiếng vỡ kính, tiếng sấm sét, v.v. Việc nhận diện âm thanh môi trường sẽ cung cấp thông tin về sự đa dạng sinh học tại môi trường đó, hay phát hiện sự kiện âm thanh có thể được sử dụng trong cảnh báo và giám sát. Ngoài ra, các hệ thống nhận diện âm thanh còn được ứng dụng trong chăm sóc sức khỏe y tế, theo dõi và phân tích âm thanh đa phương tiện, v.v.

Phát hiện âm thanh bất thường (Anomalous Sound Detection – ASD) là một bài toán được xây dựng nhằm đáp ứng nhu cầu về giám sát bằng âm thanh. Thuật ngữ "âm thanh bất thường" nhằm để chỉ những âm thanh chưa từng được thấy, khác với những âm thanh "bình thường" đã được xác định từ trước. Lấy một ví dụ trong ứng dụng giám sát nhà máy dựa trên tiếng máy móc chạy, khi phát hiện có tiếng động lạ khi máy móc hoạt động (ví dụ: tiếng nổ, tiếng rơi vỡ đồ vật) thì hệ thống có thể phát cảnh báo nguy hiểm, hay khi nhận thấy tiếng máy hoạt động khác với bình thường thì hệ thống có thể nhắc nhở kiểm tra máy. Nhiệm vụ của các hệ thống phát hiện âm thanh bất thường nói chung, là phát hiện tất cả các âm thanh lạ xảy ra trong một môi trường âm thanh nhất định.

## 1.2 Sơ lược về trí thông minh nhân tạo

### *1.2.1 Giới thiệu*

Trong cuộc sống hiện đại ngày nay, chúng ta hẳn đã nghe nói về trí tuệ nhân tạo (Artificial Intelligence – AI). AI đã được ứng dụng trong hầu hết các lĩnh vực, ví dụ như nhận diện khuôn mặt, hệ thống gợi ý, trò chơi máy tính, v.v. Sau khi thế giới trải qua ba cuộc cách mạng công nghiệp, lần thứ nhất với sự phát minh ra động cơ hơi nước vào cuối thế kỉ XVIII, lần thứ hai là sự ra đời của năng lượng điện vào cuối thế kỉ XIX, và lần thứ ba khi công nghệ thông tin bùng nổ vào những năm 1980, thì hiện tại thế giới đang bước vào cuộc cách mạng công nghiệp lần thứ tư với sự nở rộ của AI.

Ý tưởng về AI đã có từ đầu thế kỉ XX, nhưng phải đến năm 1956, tại hội nghị Dartmouth được tổ chức bởi John McCarthy tại Hoa Kỳ, thuật ngữ AI mới lần đầu được nhắc đến [11]. Một trong những nền tảng cốt lõi của AI là thuật toán học perceptron (Perceptron Learning Algorithm – PAL) giúp giải quyết các bài toán phân loại nhị phân. Tuy nhiên, vào năm 1969, Marvin Minsky và Seymour Papert đã chứng minh không thể sử dụng PLA nếu dữ liệu không tách biệt tuyến tính [12]. Phát hiện này khiến sự phát triển AI bị chững lại gần 20 năm. Mãi đến năm 1986, Geoffrey Hinton mới chứng minh một mạng perceptron đa tầng (Multi-layer Perceptron – MLP) có thể được huấn luyện dựa trên thuật toán lan truyền ngược (Backpropagation) [13], giúp giải quyết những hạn chế của Perceptron về việc chỉ biểu diễn được các quan hệ tuyến tính. Tuy nhiên, việc nghiên cứu AI vẫn còn nhiều khó khăn vì khả năng tính toán của máy tính vẫn còn hạn chế. Từ đó đến khoảng cuối thập niên 2000, AI vẫn được phát triển nhưng chưa thu hút được nhiều sự chú ý cũng như ứng dụng nhiều trong đời sống. Phải đến khoảng thời gian 10 năm trở lại đây, khi phần cứng máy tính có sự đột phá về khả năng xử lí và lưu trữ dữ liệu, thì AI mới thực sự bùng nổ và có mặt trong mọi khía cạnh đời sống.

### *1.2.2 Học máy*

Học máy (Machine Learning – ML) là một cách tiếp cận để thực hiện AI. Theo định nghĩa của Arthur L. Samuel vào năm 1959, ML là một lĩnh vực con của khoa học máy tính, làm cho máy tính có khả năng học mà không cần lập trình cụ thể [14]. ML về cơ bản là sử dụng thuật toán để phân tích dữ liệu, học từ nó thông qua việc huấn luyện và đưa ra quyết định hoặc dự đoán tùy vào yêu cầu từng bài toán, thay vì các phần mềm được lập trình với các lệnh cụ thể để thực hiện một nhiệm vụ cụ thể. Bảng 1.1 liệt kê một số bài toán phổ biến trong ML.

**Bảng 1.1 Một số bài toán phổ biến trong ML [15]**

| Bài toán | Mục đích | Ví dụ |
|---|---|---|
| Phân loại (Classification) | Chỉ ra nhãn của một điểm dữ liệu. Nhãn được chia thành các lớp | Nhận dạng chữ viết tay |
| Hồi quy (Regression) | Bài toán trong đó nhãn là số thực | Ứng dụng dự đoán độ tuổi dựa trên ảnh chụp khuôn mặt |
| Phân cụm (Clustering) | Chia dữ liệu thành các nhóm dựa trên sự tương quan của chúng | Phân loại khách hàng dựa trên hành vi mua sắm |
| Hoàn thiện (Completion) | Hoàn thiện dữ liệu với các giá trị còn thiếu | Hệ thống gợi ý |
| Một số bài toán khác: xếp hạng (Ranking), giảm nhiễu (Denoising), khôi phục thông tin (Information Retrieval), v.v | | |

### *1.2.3 Học sâu*

Học sâu (Deep Learning – DL) là một kĩ thuật để triển khai ML. Về bản chất, DL là thuật ngữ dùng để chỉ mạng neural với rất nhiều lớp (chi tiết về mạng neural sẽ được trình bày trong chương 2). Lấy một ví dụ của DL, chương trình chơi cờ vây Google AlphaGo đã học cách chơi và được huấn luyện để ngày một chơi tốt hơn, nó điều chỉnh mạng lưới thần kinh của mình bằng cách chơi với chính nó rất rất nhiều ván cờ [16].

DL đã tạo điều kiện cho nhiều ứng dụng thực tế của ML và từ đó mở rộng lĩnh vực tổng thể của AI. Xe không người lái, chăm sóc y tế dự phòng tốt hơn, các đề xuất phim tốt hơn là một số ứng dụng quan trọng của DL trong cuộc sống.

## 1.3 Các phương pháp hiện có để giải quyết bài toán phát hiện âm thanh bất thường

Phương pháp giải quyết bài toán ASD có thể chia thành hai loại: phương pháp có giám sát và phương pháp không giám sát. Phương pháp có giám sát sử dụng các nhãn dữ liệu được gán bằng tay, xác định âm thanh bất thường ở chỗ nào, bao gồm các bài toán phân loại ngữ cảnh âm thanh (Acoustic Scene Classification – ASC) [17], [18], và phát hiện sự kiện âm thanh (Acoustic Event Detection – AED) [19], [20], [21]. Nhiệm vụ của ASC là phân loại các đoạn âm thanh thành từng loại ngữ cảnh, trong khi AED xác định thời điểm bắt đầu và kết thúc của các sự kiện âm thanh. Các phương pháp này có khả năng nhận diện nhiều dạng âm thanh khác nhau, nhưng chúng đòi hỏi phải có nhãn dữ liệu. Ngược lại, các phương pháp không giám sát không yêu cầu có nhãn trước, do đó được sử dụng phổ biến hơn. Một phương pháp không giám sát là phát hiện điểm thay đổi (Change Point Detection) [22], [23], [24], được thực hiện bằng cách so sánh mô hình tại thời điểm hiện tại với thời điểm trước đó để tính toán sự tương đồng giữa hai thời điểm, nếu xảy ra sự không tương đồng thì tại đó xác định điểm bất thường. Tuy nhiên, đối với không gian công cộng, các âm thanh có thể xảy ra với độ biến động cao và không cố định, do đó các điểm thay đổi có thể không phải bất thường (ví dụ: tiếng xe cấp cứu). Một phương pháp không giám sát khác là phát hiện ngoại lệ (Outlier Detection) [25], [26], [27], thực hiện mô hình hóa các mẫu âm thanh bình thường của môi trường, sau đó phát hiện các âm thanh không tương thích với mô hình và xác định đó là bất thường. Mô hình hỗn hợp Gaussian (Gaussian Mixture Model – GMM) [28], máy vector hỗ trợ (Support Vector Machine – SVM) [29] là các mô hình điển hình cho phương pháp này.

Thời gian gần đây khi DL nở rộ, những mạng neural được nghiên cứu và sử dụng ngày càng nhiều để giải quyết bài toán phát hiện âm thanh bất thường. Trong số đó, những mạng neural được sử dụng phổ biến nhất là Autoencoder và các biến thể của nó. Autoencoder sẽ được huấn luyện để có thể nén và tái tạo lại âm thanh bình thường, sau đó được sử dụng để tái tạo âm thanh, nếu gặp phải âm thanh bất thường sẽ sinh ra lỗi tái tạo lớn. Phương pháp sử dụng Autoencoder được ưa chuộng vì không yêu cầu có nhãn dữ liệu huấn luyện, tiết kiệm được nhiều chi phí. Đồ án này chỉ tập trung nghiên cứu về các mạng Autoencoder.

## 1.4 Kế hoạch thực hiện đồ án

Bảng 1.2 thể hiện kế hoạch thực hiện đồ án tốt nghiệp này trong khoảng thời gian 17 tuần, bắt đầu từ ngày 02/03/2020 đến ngày 26/06/2020.

**Bảng 1.2 Kế hoạch thực hiện đồ án**

| STT | Công việc | Thời gian |
|---|---|---|
| 1 | Tìm hiểu bài toán phát hiện âm thanh bất thường; tìm hiểu Machine Learning, Deep Learning, ngôn ngữ lập trình Python và các kiến thức, công cụ liên quan | 05 tuần (02/03/2020 – 05/04/2020) |
| 2 | Tìm hiểu các phương pháp hiện có để phát hiện âm thanh bất thường, đặc biệt là các phương pháp sử dụng mạng Autoencoder | 04 tuần (06/04/2020 – 03/05/2020) |
| 3 | Tìm hiểu các mạng Autoencoders mới được sử dụng để phát hiện bất thường trong các lĩnh vực khác nhưng chưa được sử dụng trong lĩnh vực âm thanh | 03 tuần (04/05/2020 – 24/05/2020 |
| 4 | Triển khai thí nghiệm với tập dữ liệu sử dụng các mạng đã nghiên cứu và xây dựng, từ đó rút ra các nhận xét | 03 tuần (25/05/2020 – 14/06/2020) |
| 5 | Tổng hợp kết quả, đưa ra kết luận, xác định hướng phát triển, viết báo cáo | 02 tuần (15/06/2020 – 26/06/2020) |

## 1.5 Kết luận chương

Chương 1 đã giới thiệu về bài toán phát hiện âm thanh bất thường, trình bày tổng quát nhất về AI, ML và DL, đây là những nền tảng cơ bản nhất để có thể thực hiện đề tài này. Ngoài ra, chương này cũng đã trình bày sơ lược về các phương pháp phát hiện âm thanh bất thường và đặc điểm của từng phương pháp.

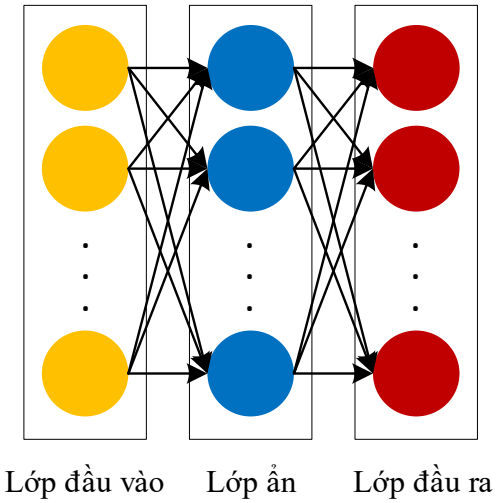
# CHƯƠNG 2. CƠ SỞ LÍ THUYẾT MẠNG NEURAL

Chương 2 giới thiệu về mạng neural nhân tạo, trình bày nguyên lí hoạt động của mạng neural, hàm mất mát, thuật toán lan truyền ngược, cách huấn luyện mạng neural, các hiện tượng Vanishing Gradient và quá khớp khi huấn luyện mạng neural.

## 2.1 Giới thiệu

Mạng neural, tên đầy đủ là mạng neural nhân tạo (Artificial Neural Network – ANN), là một mô hình mô phỏng được lấy cảm hứng từ các neuron sinh học trong hệ thống thần kinh của con người và động vật, được triển khai trên máy tính nhằm giải quyết một bài toán cụ thể.

Về cơ bản, mạng neural gồm ba lớp: lớp đầu vào, lớp ẩn và lớp đầu ra. Kiến trúc cơ bản của mạng neural được thể hiện trên Hình 2.1.




**Hình 2.1 Kiến trúc cơ bản của mạng neural**

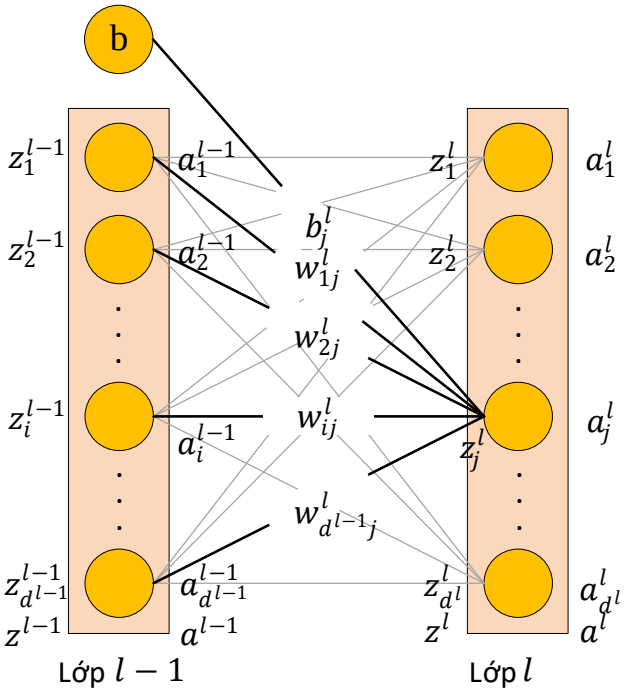
Mạng neural luôn có một lớp đầu vào, một lớp đầu ra, một hoặc nhiều lớp ẩn. Trong mỗi lớp có thể có một hoặc nhiều neurons, các neurons này hoạt động giống neuron sinh học, chúng nhận dữ liệu từ các neurons lớp trước đó, xử lí và truyền dữ liệu sang các neurons lớp sau. Các kết nối giữa các neurons của các lớp được thể hiện qua các trọng số và mạng neural sẽ "học" dựa vào một thuật toán học để cập nhật các trọng số này sao cho khớp với dữ liệu huấn luyện. Số lớp của một mạng neural được tính bằng tổng số lớp ẩn và lớp đầu ra. Số neurons lớp đầu vào thường bằng số thành phần đặc trưng của dữ liệu, còn số neurons lớp đầu ra được xác định tùy thuộc vào từng bài toán, ví dụ bằng

số lớp cần phân loại trong bài toán phân loại. Số lớp và số neurons trong mỗi lớp được gọi chung các siêu tham số của mạng. Mạng neural như trên Hình 2.1 còn được gọi là mạng neural truyền thẳng (Feedforward Neural Network – FNN) do dữ liệu được truyền thẳng từ đầu đến cuối mà không quay lại ở điểm nào.

## 2.2 Nguyên lí hoạt động

### *2.2.1 Kết nối giữa hai lớp mạng neural liên tiếp*

Mạng neural hoạt động dựa trên việc truyền dữ liệu từ lớp đầu tiên đến lớp cuối cùng của mạng. Hình 2.2 minh họa kết nối giữa hai lớp mạng neural liên tiếp.




**Hình 2.2 Kết nối giữa hai lớp neural liên tiếp**

Xét lớp $l-1$ và lớp $l$ là hai lớp các neurons liên tiếp, mỗi lớp có lần lượt $d^{l-1}$ và $d^l$ neurons. Kí hiệu $a_i^{l-1}$ là đầu ra neuron thứ i lớp $l-1$, $z_j^l$ và $a_j^l$ lần lượt là đầu vào và đầu ra của neuron thứ j lớp thứ $l$. Giá trị $z_j^l$ được tính như sau:

$$z_j^l = \sum_{i=1}^{d^{l-1}} w_{ij}^l a_i^{l-1} + b_j^l \tag{2.1}$$

Trong đó:

- $w_{ij}^l$: trọng số của kết nối từ neuron thứ i lớp $l-1$ đến neuron thứ j lớp $l$
- $b_j^l$: bias của lớp $l$ kết nối với neurn thứ j lớp $l$

Giá trị đầu ra $a_j^l$ của neuron có quan hệ với đầu vào như sau:

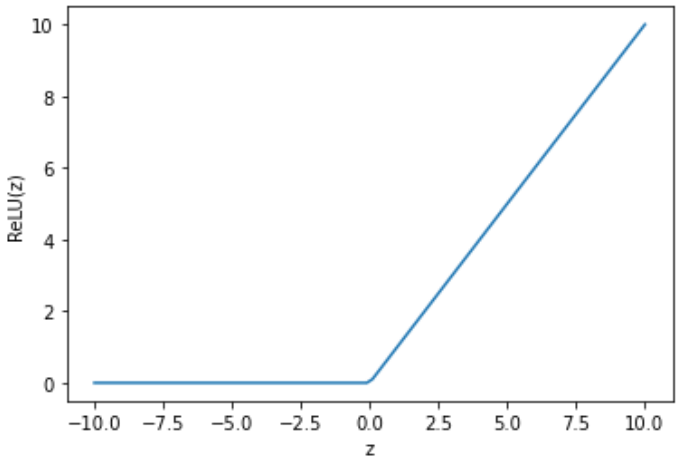
$$a_j^l = f\left(z_j^l\right) \tag{2.2}$$

Trong đó $f$ là một hàm kích hoạt phi tuyến của neuron. Hàm ReLU (Rectified Linear Unit), hàm tanh và hàm Sigmoid là ba hàm kích hoạt được sử dụng phổ biến. Dưới đây là công thức và đồ thị của chúng [15]:

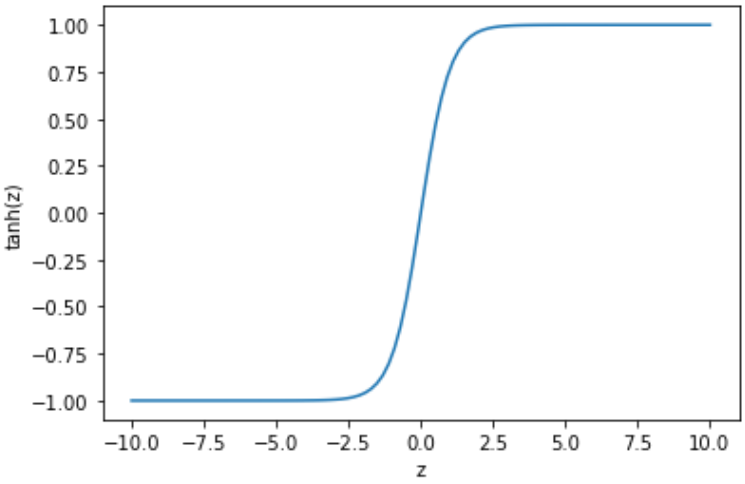
$$ReLU(z) = \max(0, z) \tag{2.3}$$

$$\tanh(z) = \frac{e^z - e^{-z}}{e^z + e^{-z}} \tag{2.4}$$

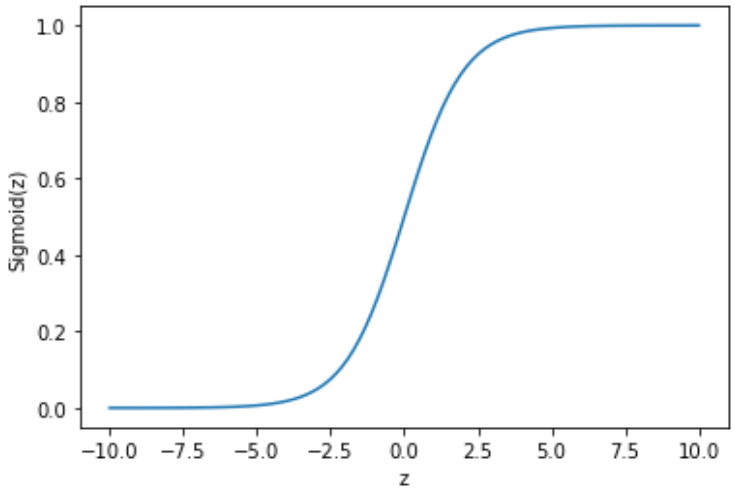
$$Sigmoid(z) = \frac{1}{1 + e^{-z}} \tag{2.5}$$




**Hình 2.3 Đồ thị hàm ReLU**




**Hình 2.4 Đồ thị hàm tanh**




**Hình 2.5 Đồ thị hàm Sigmoid**

Hàm ReLU đã được chứng minh giúp tốc độ huấn luyện mạng neural nhanh hơn nhiều so với hàm tanh [30], do đó thường được sử dụng làm hàm kích hoạt ở lớp đầu vào và các lớp ẩn. Trong khi đó, đầu ra hàm Sigmoid luôn nằm trong khoảng (0, 1) nên thường được sử dụng ở lớp đầu ra để tính toán xác suất một điểm dữ liệu thuộc về lớp nào trong bài toán phân loại.

Kí hiệu $z^l$, $a^l$, $W^l$, $b^l$ lần lượt là ma trận đầu vào, ma trận đầu ra của lớp $l$, ma trận trọng số của các kết nối từ lớp $l-1$ đến lớp $l$ và ma trận bias của lớp $l$. Giả sử mạng neural có L lớp, ta có mối quan hệ sau:

$$a^l = f(z^l) = f(W^l a^{l-1} + b^l), \qquad l = 1,2,\dots,L \tag{2.6}$$

Trong đó, $a^{l-1} \in R^{d^{l-1}\times 1}$, $W^l \in R^{d^l \times d^{l-1}}$, $b^l \in R^{d^l \times 1}$ và $a^l, z^l \in R^{d^l \times 1}$.

Có tất cả L ma trận trọng số và L ma trận bias cho một mạng neural L lớp. Các ma trận này được cập nhật qua một thuật toán học, làm cho mạng neural có thể xấp xỉ đầu vào và đầu ra tương ứng một các chính xác nhất có thể. Các trọng số và bias của mạng neural đều được gọi chung là các tham số mô hình.

### *2.2.2 Hàm mất mát*

Mối quan hệ giữa nhãn thực của dữ liệu và đầu ra của mạng neural được biểu diễn thông qua hàm mất mát. Mạng neural hoạt động tốt khi hàm mất mát nhỏ, vì vậy hàm mất mát cần phải được tối thiểu hóa.

Gọi $\theta$ là tập các tham số mô hình, $\theta^*$ là bộ tham số của mô hình có hàm mất mát tối ưu, kí hiệu hàm mất mát của mô hình là $J(\theta)$, ta cần giải bài toán tối ưu sau [15]:

$$\theta^* = \underset{\theta}{argmin}\, J(\theta) \tag{2.7}$$

### *2.2.3 Thuật toán lan truyền ngược và suy giảm gradient*

Để tối thiểu hóa hàm mất mát, việc bắt buộc là phải tính đạo hàm. Tuy nhiên, việc tính toán đạo hàm của hàm mất mát là cực kì phức tạp, do đó một phương pháp phổ biến được sử dụng để tính đạo hàm ngược từ lớp cuối cùng lên lớp đầu tiên có tên là lan truyền ngược (Backpropagation). Việc tính toán đạo hàm theo kiểu chuỗi này được dựa trên quy tắc chuỗi cho đạo hàm của hàm hợp, do đầu ra của mạng neural được tính bằng hợp của rất nhiều hàm kích hoạt giữa các lớp của mạng.

Gọi J là một hàm mất mát bất kì, đạo hàm của J theo một thành phần của ma trận trọng số lớp đầu ra được tính là:

$$\frac{\partial J}{\partial w_{ij}^L} = \frac{\partial J}{\partial z_j^L} \cdot \frac{\partial z_j^L}{\partial w_{ij}^L} \tag{2.8}$$

Đặt $e_j^L = \frac{\partial J}{\partial z_j^L}$. Do $\frac{\partial z_j^L}{\partial w_{ij}^L} = \frac{\partial (w_{ij}^L a_i^{L-1} + b_j^L)}{\partial w_{ij}^L} = a_i^{L-1}$ nên ta có:

$$\frac{\partial J}{\partial w_{ij}^L} = e_j^L a_i^{L-1} \tag{2.9}$$

Trong công thức (2.9), $e_j^L$ là một đại lượng có thể được tính toán không quá phức tạp vì chỉ là đạo hàm của J theo đầu vào của một neuron. Tương tự, đạo hàm của J theo một thành phần của ma trận trọng số thuộc lớp $l$ bất kì được xác định như sau:

$$\frac{\partial J}{\partial w_{ij}^l} = \frac{\partial J}{\partial z_j^l} \cdot \frac{\partial z_j^l}{\partial w_{ij}^l} = e_j^l a_i^{l-1} \tag{2.10}$$

Với $e_j^l = \frac{\partial J}{\partial z_j^l} = \frac{\partial J}{\partial a_j^l} \cdot \frac{\partial a_j^l}{\partial z_j^l}$. Do $a_j^l$ đóng góp vào việc tính các giá trị đầu vào $z_k^{l+1}$ của các neuron lớp $l+1$, ta có:

$$\frac{\partial J}{\partial a_j^l} = \sum_{k=1}^{d^{l+1}} \frac{\partial J}{\partial z_k^{l+1}} \cdot \frac{\partial z_k^{l+1}}{\partial a_j^l} \tag{2.11}$$

Ngoài ra, vì $a_j^l = f(z_j^l)$ nên:

$$\frac{\partial a_j^l}{\partial z_j^l} = f'(z_j^l) \tag{2.12}$$

Thay (2.11) và (2.12) vào công thức tính $e_j^l$, ta thu được:

$$e_j^l = \left(\sum_{k=1}^{d^{l+1}} \frac{\partial J}{\partial z_k^{l+1}} \cdot \frac{\partial z_k^{l+1}}{\partial a_j^l}\right) f'(z_j^l) = \left(\sum_{k=1}^{d^{l+1}} e_k^{l+1} w_{jk}^{l+1}\right) f'(z_j^l) \tag{2.13}$$

Như vậy, việc tính mỗi giá trị $e_j^l$ lại phụ thuộc vào các giá trị $e_k^{l+1}$, vì vậy các giá trị này cần được tính ngược từ cuối mạng neural lên đầu. Tương tự với trọng số, đối với đạo hàm của J theo bias, ta có:

$$\frac{\partial J}{\partial b_j^l} = e_j^l \tag{2.14}$$

Sau khi tính được đạo hàm của J theo từng ma trận trọng số $W^l$ và ma trận bias $b^l$, các trọng số và bias được cập nhật theo phương pháp suy giảm gradient (Gradient Descent – GD) như sau [15]:

$$W^l \leftarrow W^l - \eta \nabla_{W^l} J \tag{2.15}$$

$$b^l \leftarrow b^l - \eta \nabla_{b^l} J \tag{2.16}$$

Trong đó:

- $\eta$: được gọi là tốc độ học, luôn lớn hơn 0
- $\nabla_{W^l} J$: đạo hàm của J theo $W^l$
- $\nabla_{b^l} J$: đạo hàm của J theo $b^l$

## 2.3 Một số khái niệm khác

### *2.3.1 Suy giảm gradient ngẫu nhiên*

Thuật toán suy giảm gradient ngẫu nhiên (Stochastic Gradient Descent – SGD) là một biến thể của GD. Thuật toán SGD chỉ tính đạo hàm của hàm mất mát dựa trên một điểm dữ liệu rồi cập nhật các tham số mô hình $\theta$ theo đạo hàm này. Sau khi duyệt qua tất cả các điểm dữ liệu, thuật toán lặp lại quá trình trên. Mỗi lần duyệt qua tất cả các

điểm dữ liệu được gọi là một epoch, mỗi epoch gồm N lần cập nhật $\theta$ với N là số điểm dữ liệu [15].

SGD là một thuật toán đơn giản hơn GD, đi kèm với đó là sai số nhưng tốc độ hội tụ của hàm mất mát nhanh hơn. SGD phù hợp với các bài toán có lượng cơ sở dữ liệu lớn.

### *2.3.2 Suy giảm gradient theo mini-batch*

Thuật toán suy giảm gradient theo mini-batch (Mini-batch GD) sử dụng k điểm dữ liệu để cập nhật $\theta$ trong một vòng lặp (trừ vòng lặp cuối nếu N không chia hết cho k) thay vì một điểm dữ liệu như SGD. Mỗi epoch bao gồm khoảng $\frac{N}{k}$ vòng lặp, mỗi vòng lặp cập nhật $\theta$ được gọi là một mini-batch và k được gọi là batch size. Mini-batch GD được sử dụng hầu hết trong các thuật toán DL [15].

### *2.3.3 Vanishing Gradient*

Vanishing Gradient là một hiện tượng không mong muốn khi huấn luyện mạng neural. Hiện tượng này xảy ra do giới hạn tính toán của máy tính, khi sử dụng thuật toán lan truyền ngược để tính đạo hàm của hàm mất mát, cần phải nhân rất nhiều các giá trị nhỏ hơn 1 với nhau, khiến cho nhiều đạo hàm thành phần ngày càng gần về 0. Số thực được biểu diễn trong máy tính có một giới hạn để biểu diễn số thực nhỏ nhất khác 0, nếu các giá trị tính toán nhỏ hơn ngưỡng này sẽ được biểu diễn bằng 0, do đó các tham số mô hình sẽ không được cập nhật [15].

Để tránh hiện tượng Vanishing Gradient, các tham số mô hình khởi tạo có thể được tiền huấn luyện để có giá trị khởi tạo tốt, hoặc một cách đơn giản hơn là lấy logarithm của dữ liệu đặc trưng, làm cho giá trị của chúng lớn hơn rất nhiều. Những cách làm này đã giúp giải quyết phần nào vấn đề Vanishing Gradient, tuy vẫn chưa thật sự triệt để.

### *2.3.4 Quá khớp*

Khi huấn luyện các mạng neural, mô hình được huấn luyện sao cho càng khớp với dữ liệu càng tốt. Tuy nhiên, khi mô hình quá khớp với dữ liệu huấn luyện, một hiện tượng không mong muốn là mô hình không có kết quả tốt với dữ liệu kiểm tra. Hiện tượng này được gọi là quá khớp (Overfitting), dùng để chỉ mô hình tìm được không có tính tổng quát.

Một mô hình được coi là tốt nếu cả lỗi khi huấn luyện và lỗi khi kiểm tra đều nhỏ. Để khắc phục hiện tượng quá khớp, một phương pháp được sử dụng phổ biến là dùng thêm tập xác thực bên cạnh tập huấn luyện và tập kiểm tra để điều chỉnh các siêu tham số của mạng. Khi huấn luyện, lỗi huấn luyện và lỗi xác thực đều được tính toán, nếu lỗi huấn luyện vẫn có xu hướng giảm nhưng lỗi xác thực có xu hướng tăng thì ta kết thúc quá trình huấn luyện. Kĩ thuật này được gọi là kết thúc sớm (Early Stopping) [15].

## 2.4 Kết luận chương

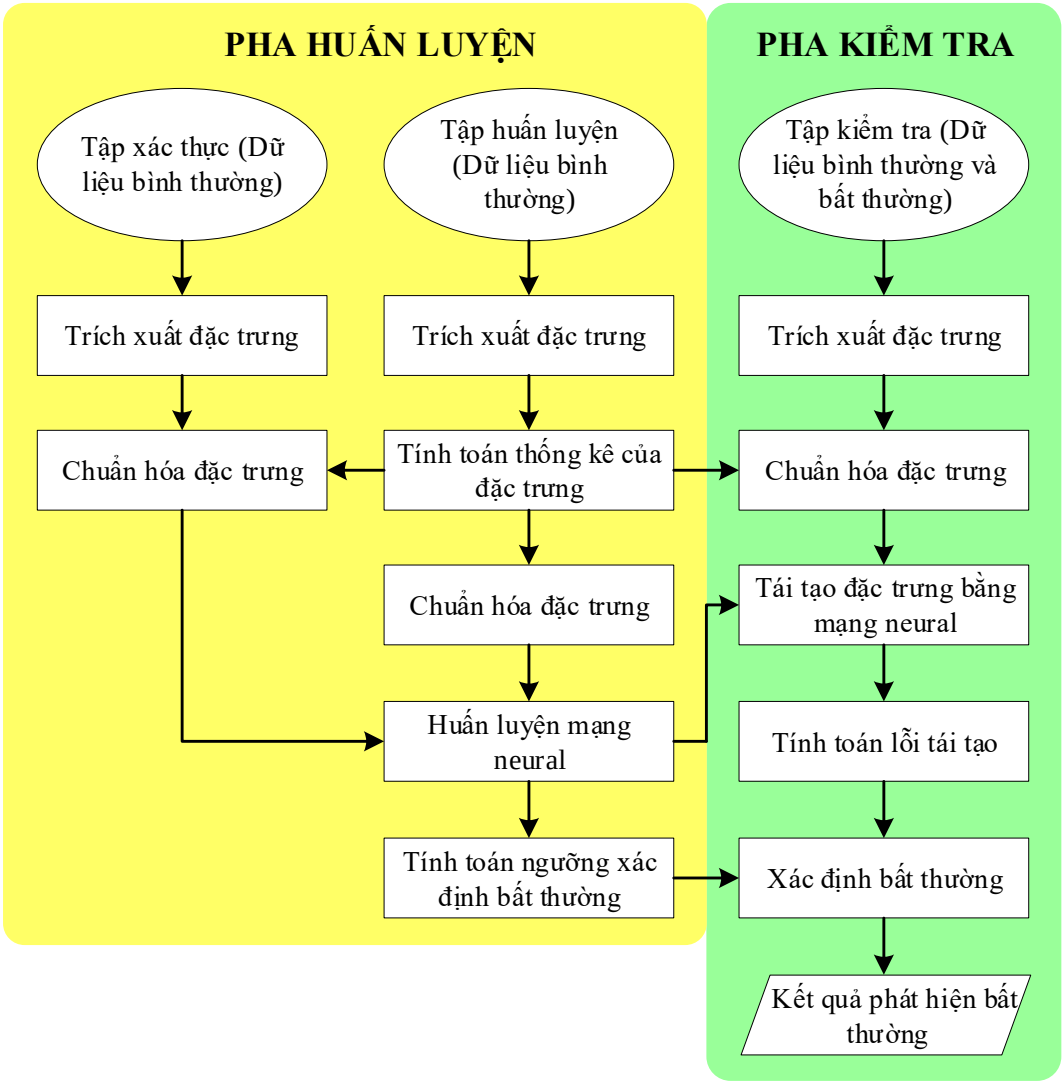
Chương 2 đã trình bày những kiến thức của mạng neural, phương pháp huấn luyện mạng neural bằng thuật toán lan truyền ngược và một số vấn đề khác liên quan đến quá trình huấn luyện mạng neural. Những kiến thức này là cơ sở cố lõi để có thể sử dụng mạng neural giải quyết bài toán phát hiện âm thanh bất thường.

# CHƯƠNG 3. MÔ HÌNH HỆ THỐNG PHÁT HIỆN ÂM THANH BẤT THƯỜNG

Chương 3 trình bày về mô hình hệ thống phát hiện âm thanh bất thường, lí thuyết về xử lí tín hiệu âm thanh, cách trích xuất đặc trưng của dữ liệu âm thanh và khuôn mẫu chung cho các mạng neural trong bài toán phát hiện âm thanh bất thường.

## 3.1 Tổng quan hệ thống

Hình 3.1 thể hiện tổng quan về hệ thống phát hiện âm thanh bất thường.




**Hình 3.1 Tổng quan hệ thống phát hiện âm thanh bất thường**

Trong pha huấn luyện, ban đầu, dữ liệu âm thanh được lấy ra từ tập huấn luyện (chỉ gồm dữ liệu âm thanh bình thường) và đưa vào khối trích xuất đặc trưng để tính toán Mel Filter Banks (đặc trưng của dữ liệu âm thanh). Sau đó, các tham số thống kê (kì vọng, phương sai, độ lệch chuẩn, giá trị lớn nhất, giá trị nhỏ nhất) của các đặc trưng được tính toán và được sử dụng để chuẩn hóa sao cho các giá trị đặc trưng đều nằm trong khoảng (0, 1). Những tham số thống kê này cũng được dùng để chuẩn hóa đặc trưng cho dữ liệu trong tập xác thực và tập kiểm tra. Tiếp theo, mạng neural được huấn luyện để tái tạo dữ liệu đầu vào sử dụng các đặc trưng của tập huấn luyện, trong khi đó tập xác thực được dùng để điều chỉnh các siêu tham số của mạng neural. Cuối cùng, một ngưỡng được tính toán dùng để phát hiện dữ liệu bất thường ở pha kiểm tra.

Trong pha kiểm tra, dữ liệu gồm cả âm thanh bình thường và bất thường cũng được trích xuất đặc trưng và chuẩn hóa sử dụng các tham số thống kê của dữ liệu huấn luyện. Các đặc trưng đã chuẩn hóa được đưa vào mạng neural đã được huấn luyện để tái tạo dữ liệu, sau đó lỗi tái tạo thể hiện sự khác nhau giữa dữ liệu gốc và dữ liệu tái tạo sẽ được tính toán. Nếu một mẫu dữ liệu âm thanh có lỗi tái tạo này lớn hơn ngưỡng đã tính toán từ trước thì âm thanh tại đó được xác định là bất thường.

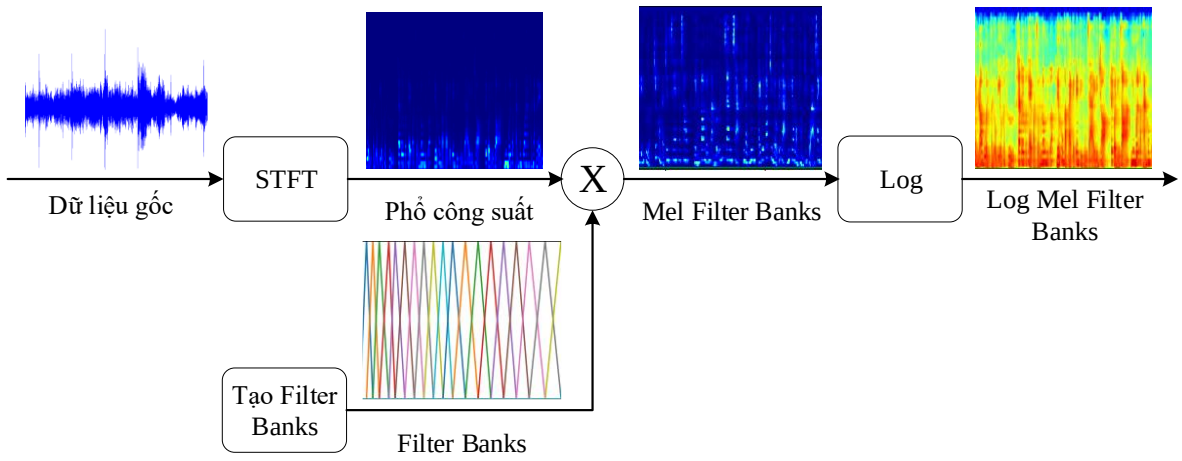
## 3.2 Trích xuất đặc trưng

### *3.2.1 Tổng quát*

Trích xuất đặc trưng là bước đầu tiên trong bất kì hệ thống nhận diện âm thanh nào, hiểu đơn giản là xác định các thành phần của tín hiệu âm thanh mà có thể dùng để xác định các tính chất âm thanh và loại bỏ những thành phần không có ý nghĩa khác như nhiễu nền.

Mel Filter Banks (MFBs), hay còn được gọi là Mel-band Energies, là một đặc trưng được sử dụng rất rộng rãi trong lĩnh vực nhận diện âm thanh. Nếu biến đổi Mel Filter Banks thêm một vài bước, đặc trưng thu được được gọi là Mel Frequency Cepstral Coefficents (MFCCs). MFCCs lần đầu được đưa ra bởi S. Davis và P. Mermelstein vào năm 1980 [31] và từ đó trở nên phổ biến. Tuy nhiên gần đây, khi DL phát triển mạnh, đặc trưng MFBs được ưa chuộng hơn, điều này sẽ được giải thích ở mục 3.2.3. Trong đồ án này, đặc trưng MFBs  được sử dụng.

Hình 3.2 thể hiện sơ đồ tổng quát khối trích xuất đặc trưng.



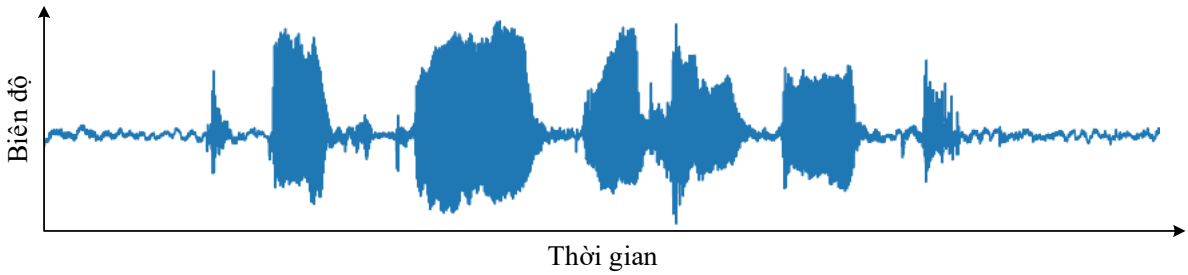

**Hình 3.2 Sơ đồ tổng quát khối trích xuất đặc trưng**

Dữ liệu âm thanh gốc ban đầu được đưa vào khối STFT để tính toán phổ công suất của tín hiệu. Tiếp theo, khối tạo Filter Banks tạo ra các bộ lọc được gọi là Filter Banks, là các bộ lọc tam giác được xây dựng dựa trên hoạt động tai người. Tiếp tục thực hiện nhân các bộ lọc này với phổ công suất của tín hiệu để thu được các đặc trưng MFBs Cuối cùng, lấy logarithm của MFBs, ta thu được Log MFBs. Dưới đây sẽ trình bày cụ thể từng bước để trích xuất đặc trưng MFBs của dữ liệu âm thanh.

### *3.2.2 Biến đổi STFT*

Biến đổi Fourier thời gian ngắn (Short-time Fourier Transform – STFT) được sử dụng để tính toán phổ công suất của tín hiệu âm thanh, là bước đầu tiên của quá trình trích xuất đặc trưng MFBs. Về bản chất, STFT là biến đổi Fourier được thực hiện trên các đoạn thời gian ngắn bằng nhau.

Giả sử ban đầu ta có tín hiệu thực liên tục x(t) có độ dài L s như trên Hình 3.3.

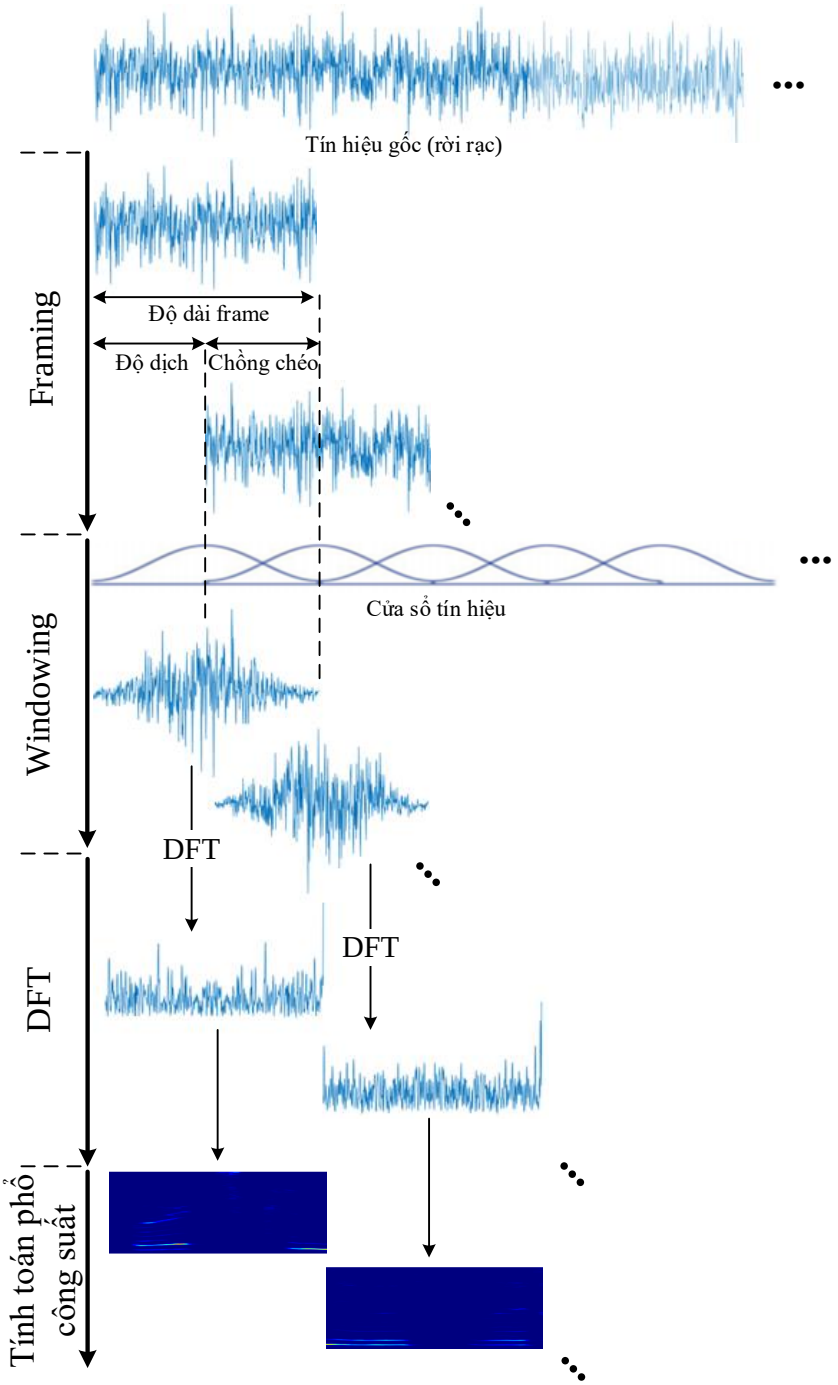



**Hình 3.3 Tín hiệu âm thanh ban đầu**

Tín hiệu thực liên tục x(t) sau khi được lấy mẫu với tốc độ $f_s$ Hz sẽ trở thành tín hiệu thực rời rạc được biểu diễn dưới dạng ma trận X như sau:

$$X = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_{Lf_s} \end{bmatrix} \in R^{Lf_s \times 1} \tag{3.1}$$

Để thực hiện STFT, tín hiệu được chia thành các frames nhỏ (Framing), sau đó nhân các frames với cửa sổ tín hiệu (Windowing), cuối cùng thực hiện biến đổi Fourier rời rạc (DFT) và tính phổ công suất của tín hiệu trên từng frame này (Hình 3.4).




**Hình 3.4 Các bước thực hiện STFT và tính toán phổ công suất của tín hiệu**

*3.2.2.1 Framing*

Framing là bước chia tín hiệu ban đầu thành các frames nhỏ có độ dài bằng nhau. Lí do của việc cần phải thực hiện Framing là do tần số của tín hiệu thay đổi theo thời gian, nếu thực hiện biến đổi Fourier trên toàn miền thời gian sẽ không có ý nghĩa do đã làm mất thông tin về sự thay đổi tần số theo thời gian. Để giải quyết vấn đề này, tần số tín hiệu được giả sử không thay đổi trên một khoảng thời gian rất ngắn. Vì vậy, tín hiệu được chia thành các frames ngắn và ta sẽ thực hiện biến đổi Fourier trên các frames này.

Đặt độ dài mỗi frame là $l_{fr}$ s, độ dịch giữa hai frames liên tiếp là $s_{fr}$ s. Giá trị điển hình cho hai tham số này là mỗi frame dài 0.025 s, độ dịch là 0.01 s, như vậy hai frames liên tiếp sẽ có độ dài chồng lấn (overlap) là 0.015 s. Một bước cần thực hiện trước khi framing là đệm (padding) thêm các giá trị 0 vào đầu, cuối hoặc cả hai đầu của tín hiệu để đảm báo tín hiệu ban đầu được chia thành các frames có độ dài bằng nhau,. Độ dài padding được tính toán như sau:

$$l_{pad} = L - l_{fr} - s_{fr}\left\lfloor \frac{L - l_{fr}}{s_{fr}} \right\rfloor \tag{3.2}$$

Trong đó, $\lfloor \cdot \rfloor$ là phép làm tròn xuống.

Số frames tạo ra là:

$$N_{fr} = \left\lceil \frac{L - l_{fr}}{s_{fr}} \right\rceil \tag{3.3}$$

Trong đó, $\lceil \cdot \rceil$ là phép làm tròn lên.

Số samples trong một frame là:

$$n_s = f_s l_{fr} \tag{3.4}$$

Như vậy, tín hiệu ban đầu đã được chia thành $N_{fr}$ frames, mỗi frame gồm có $n_s$ samples. Ma trận biểu diễn các frames là:

$$X_{fr} = \begin{bmatrix} x_{11} & x_{12} & \cdots & x_{1n_s} \\ x_{21} & x_{22} & \cdots & x_{2n_s} \\ \vdots & \vdots & \ddots & \vdots \\ x_{N_{fr}1} & x_{N_{fr}2} & \dots & x_{N_{fr}n_s} \end{bmatrix} \in R^{N_{fr} \times n_s} \tag{3.5}$$

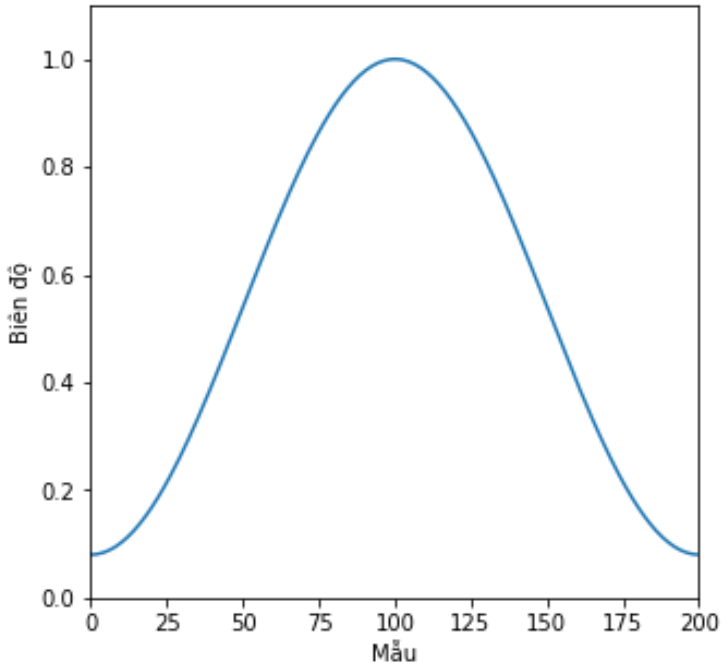
Trong đó, mỗi hàng của ma trận $X_{fr}$ biểu diễn một frame với $n_s$ samples.

*3.2.2.2 Windowing*

Windowing là bước nhân một cửa sổ tín hiệu với các frame thu được từ bước Framing. Một cửa sổ được sử dụng rất phổ biến trong xử lí tín hiệu số là cửa sổ Hamming. Công thức của cửa sổ Hamming như sau [32]:

$$w[n] = 0.54 - 0.46\cos\frac{2\pi n}{N_w - 1}, 0 \le n \le N_w - 1 \quad (3.6)$$

Trong đó $N_w$ được gọi là kích thước cửa sổ. Hình 3.5 minh họa một cửa sổ Hamming với $N_w = 200$.




**Hình 3.5 Cửa sổ Hamming với $N_w$=200**

Tiếp theo các frame được nhân với cửa sổ Hamming. Lí do của việc cần phải có bước Windowing là để chống lại giả định của biến đổi Fourier rằng dữ liệu là vô hạn. Cửa sổ Hamming có biên độ giảm dần ở hai đầu, do đó có thể gây suy giảm tín hiệu trong mỗi frame, đây là lí do ở bước Framing cần có một khoảng overlap giữa các frames.

Cửa sổ Hamming với kích thước $N_w = n_s$ có dạng ma trận như sau:

$$W_{Ham} = [w_1 \quad w_2 \quad \cdots \quad w_{N_w}] \quad (3.7)$$

Nhân từng hàng của $X_{fr}$ với $W_{Ham}$ theo kiểu nhân từng phần tử tương ứng với nhau, ta thu được ma trận $F_w$ biểu diễn các frames sau bước windowing:

$$F_W = \begin{bmatrix} x_{11}w_1 & x_{12}w_2 & \cdots & x_{1n_s}w_{N_w} \\ x_{21}w_1 & x_{22}w_2 & \cdots & x_{2n_s}w_{N_w} \\ \vdots & \vdots & \ddots & \vdots \\ x_{N_{fr}1}w_1 & x_{N_{fr}2}w_2 & \cdots & x_{N_{fr}n_s}w_{N_w} \end{bmatrix}$$

$$= \begin{bmatrix} f_{11} & f_{12} & \cdots & f_{1n_s} \\ f_{21} & f_{22} & \cdots & f_{2n_s} \\ \vdots & \vdots & \ddots & \vdots \\ f_{N_{fr}1} & f_{N_{fr}2} & \cdots & f_{N_{fr}n_s} \end{bmatrix} \in R^{N_{fr} \times n_s} \quad (3.8)$$

*3.2.2.3 Biến đổi Fourier và phổ công suất của tín hiệu*

Công suất của tín hiệu mô tả sự phân bố năng lượng của tín hiệu trên một đơn vị thời gian, và phổ công suất của tín hiệu mô tả phân phối công suất thành các thành phần tần số tạo thành tín hiệu đó. Nói cách khác, phổ công suất cung cấp thông tin về vị trí tần số tập trung năng lượng của tín hiệu.

Phổ công suất của tín hiệu đã lấy mẫu được tính toán nhờ biến biến đổi Fourier, cụ thể là biến đổi Fourier rời rạc (Discrete Fourier Transform – DFT). Công thức của DFT N điểm cho tín hiệu rời rạc $x_n$ như sau [32]:

$$X_k = \sum_{n=0}^{N-1} x_n e^{-\frac{i2\pi}{N}kn} \quad (3.9)$$

Trong đó, $X_k$ là hệ số DFT thứ k. Giá trị của N được đặt bằng một lũy thừa cơ số 2, điển hình như 512, 1024, 2048, nhằm mục đích tối ưu thuật toán tính toán biến đổi Fourier nhanh (Fast Fourier Transform – FFT).

Khi thực hiện DFT cho tín hiệu thực, đầu ra của biến đổi có dạng đối xứng, trong đó các tần số âm là các liên hợp phức của các tần số dương tương ứng, nên các tần số âm sẽ bị loại bỏ. Kết quả biến đổi DFT cho mỗi frame sẽ chỉ gồm $\frac{N}{2} + 1$ thành phần tần số dương, tương ứng với các tần số từ 0 đến $\frac{f_s}{2}$.

Tiếp theo, phổ công suất của từng frame được tính như sau:

$$P_i = \frac{|DFT(f_i)|^2}{N} \quad (3.10)$$

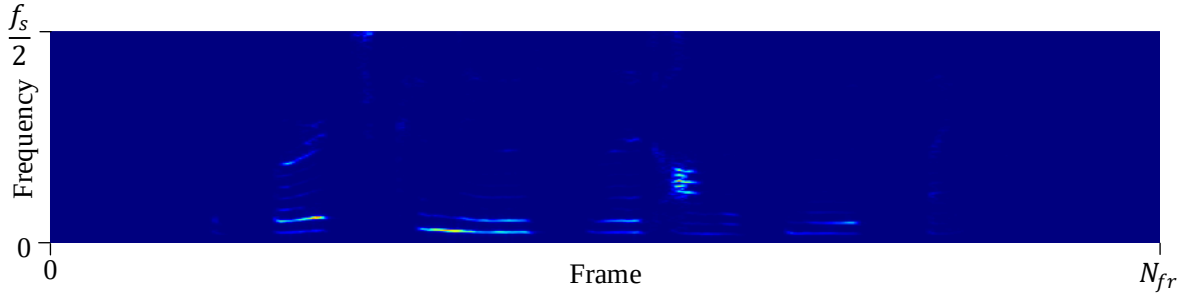
Trong đó, $P_i$ là phổ công suất của frame $f_i$.

Ma trận biểu diễn phổ công suất các frames sau khi thực hiện biến đổi DFT là:

$$P = \frac{1}{N}\begin{bmatrix} |DFT(f_1)|^2 \\ |DFT(f_2)|^2 \\ \vdots \\ \left|DFT\left(f_{N_{fr}}\right)\right|^2 \end{bmatrix} = \begin{bmatrix} p_{11} & p_{12} & \cdots & p_{1\left(\frac{N}{2}+1\right)} \\ p_{21} & p_{22} & \cdots & p_{2\left(\frac{N}{2}+1\right)} \\ \vdots & \vdots & \ddots & \vdots \\ p_{N_{fr}1} & p_{N_{fr}2} & \cdots & p_{N_{fr}\left(\frac{N}{2}+1\right)} \end{bmatrix} \in R^{N_{fr}\times\left(\frac{N}{2}+1\right)} \quad (3.11)$$

Trong đó, $f_i = [f_{i1} \quad f_{i2} \quad \cdots \quad f_{in_s}]$ là hàng thứ i của ma trận $F_w$.

Hình 3.6 minh họa phổ công suất của các frames.




**Hình 3.6 Phổ công suất của các frames**

Hình 3.6 được gọi là spectrogram, là một ảnh minh họa phổ công suất của tín hiệu. Có thể thấy, đối với tín hiệu đang xét, phổ công suất của tín hiệu tập trung tại các tần số thấp và giảm dần ở các tần số cao hơn.

### *3.2.3 Mel Filter Banks*

Phổ công suất đã cho thấy sự tập trung năng lượng của tín hiệu nằm ở các thành phần tần số nào, nhưng không cho thấy sự khác nhau giữa các tần số gần nhau. Các bộ lọc Filter Banks được tạo ra để giải quyết vấn đề này. Đây là các bộ lọc tam giác nhằm bắt chước hoạt động nhận thức về âm thanh của tai người, bằng các phân biệt rõ ràng ở các tần số thấp và ít phân biệt ở các tần số cao hơn, do tai người nhận biết các thay đổi nhỏ tốt hơn ở tần số thấp so với tần số cao [32].

Thang đo tần số Mel được xây dựng để tạo ra Filter Banks. Thang đo Mel liên quan đến tần số cảm nhận hoặc cao độ của âm thuần so với tần số thực tế của nó, làm cho các đặc trưng sinh ra phù hợp hơn với những gì con người nghe thấy. Dưới đây là hai công thức chuyển đổi giữa thang đo Mel (m) và thang đo Hz (f) [32].

$$m = 2595 \log\left(1 + \frac{f}{700}\right) \quad (3.12)$$

$$f = 700\left(10^{\frac{m}{2595}} - 1\right) \tag{3.13}$$

Trước khi tính toán các bộ lọc, ta cần chọn tần số ngưỡng dưới $f_{lower}$ và tần số ngưỡng trên $f_{upper}$, thông thường $f_{lower} = 0$ và $f_{upper} = \frac{f_s}{2}$. Sau đó hai tần số này được chuyển sang thang đo Mel sử dụng công thức (3.12). Giả sử số lượng bộ lọc sử dụng là $N_f$, ta cần chia khoảng tần số $[f_{lower}, f_{upper}]$ thành $N_f + 2$ mốc tần số với khoảng cách giữa hai mốc tần số liên tiếp là bằng nhau. Các mốc tần số này tiếp tục được chuyển về thang đo Hz bằng công thức (3.13). Ma trận B dưới đây biểu diễn các mốc tần số tính toán được:

$$B = \begin{bmatrix} b_1 & b_2 & \cdots & b_{N_f+2} \end{bmatrix} \tag{3.14}$$

Trong đó, $b_1 = f_{lower}$ và $b_{N_f+2} = f_{upper}$. Tiếp theo, ta cần chuyển các mốc tần số này cho tương ứng với các điểm DFT:

$$f_m = \frac{NB}{f_s} = \begin{bmatrix} \frac{Nb_1}{f_s} & \frac{Nb_2}{f_s} & \cdots & \frac{Nb_{N_f+2}}{f_s} \end{bmatrix} \tag{3.15}$$

Các bộ lọc tam giác của Filter Banks được xác định theo công thức sau [32]:

$$H_m(k) = \begin{cases} 0 & k < f_m(m-1) \\ \dfrac{k - f_m(m-1)}{f_m(m) - f_m(m-1)} & f_m(m-1) \le k < f_m(m) \\ 1 & k = f_m(m) \\ \dfrac{f_m(m+1) - k}{f_m(m+1) - f_m(m)} & f_m(m) < k \le f_m(m+1) \\ 0 & k > f_m(m+1) \end{cases} \tag{3.16}$$

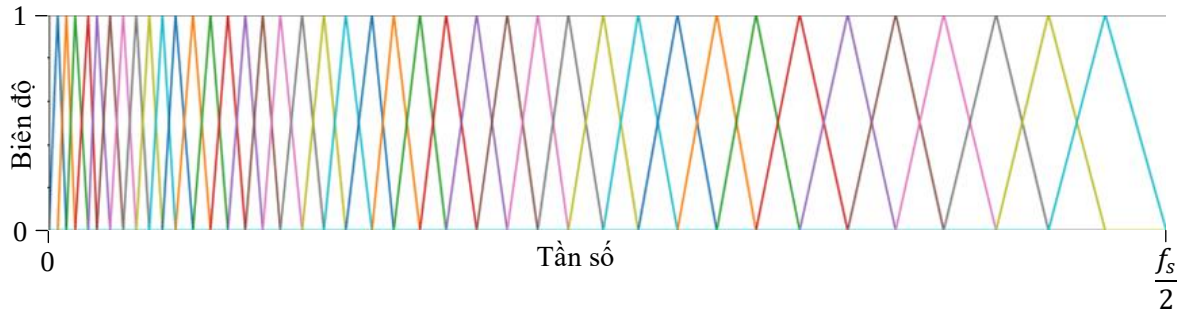
Trong đó:

- $H_m(k)$: giá trị thứ k của bộ lọc thứ m
- $f_m(m)$: Giá trị thứ phần tử thứ m+1 của $f_m$ trong (3.15)

Các bộ lọc này được biểu diễn dưới dạng ma trận như sau:

$$H = \begin{bmatrix} h_{11} & h_{12} & \cdots & h_{1N_f} \\ h_{21} & h_{22} & \cdots & h_{2N_f} \\ \vdots & \vdots & \ddots & \vdots \\ h_{\left(\frac{N}{2}+1\right)1} & h_{\left(\frac{N}{2}+1\right)2} & \cdots & h_{\left(\frac{N}{2}+1\right)N_f} \end{bmatrix} \in R^{\left(\frac{N}{2}+1\right)\times N_f} \tag{3.17}$$

Trong đó, mỗi cột của H biểu diễn một bộ lọc. Số lượng bộ lọc trên thực tế thường được chọn là 40. Hình 3.7 minh họa 40 bộ lọc Filter Banks.



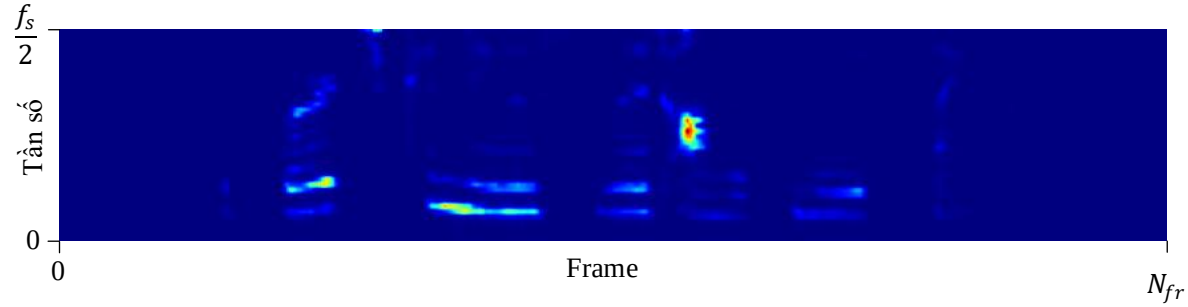

**Hình 3.7 40 bộ lọc Filter Banks**

Sau khi nhân các bộ lọc này với phổ công suất, ta thu được các đặc trưng Mel Filter Banks (MFBs), được thể hiện bằng ma trận M như sau:

$$M = \begin{bmatrix} p_{11} & p_{12} & \cdots & p_{1\left(\frac{N}{2}+1\right)} \\ p_{21} & p_{22} & \cdots & p_{2\left(\frac{N}{2}+1\right)} \\ \vdots & \vdots & \ddots & \vdots \\ p_{N_{fr}1} & p_{N_{fr}2} & \cdots & p_{N_{fr}\left(\frac{N}{2}+1\right)} \end{bmatrix} \begin{bmatrix} h_{11} & h_{12} & \cdots & h_{1N_f} \\ h_{21} & h_{22} & \cdots & h_{2N_f} \\ \vdots & \vdots & \ddots & \vdots \\ h_{\left(\frac{N}{2}+1\right)1} & h_{\left(\frac{N}{2}+1\right)2} & \cdots & h_{\left(\frac{N}{2}+1\right)N_f} \end{bmatrix}$$

$$= \begin{bmatrix} m_{11} & m_{12} & \cdots & m_{1N_f} \\ m_{21} & m_{22} & \cdots & m_{2N_f} \\ \vdots & \vdots & \ddots & \vdots \\ m_{N_{fr}1} & m_{N_{fr}2} & \cdots & m_{N_{fr}N_f} \end{bmatrix} \in R^{N_{fr} \times N_f} \quad (3.18)$$

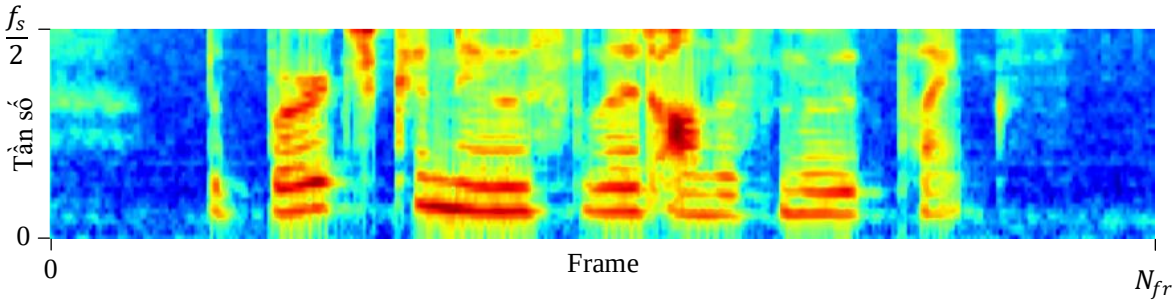
Trong đó, mỗi hàng của ma trận M biểu diễn $N_f$ giá trị đặc trưng MFBs của một frame. Các đặc trưng này được minh họa trên Hình 3.8.




**Hình 3.8 Đặc trưng Mel Filter Banks**

Cuối cùng, ta cần lấy logarithm của MFBs để thu được Log Mel Filter Banks. Vì các bộ lọc Filter Banks có giá trị nằm trong đoạn [0, 1], nên sau khi nhân với phổ công suất

sẽ sinh ra các đặc trưng MFBs có giá trị rất nhỏ. Điều này có thể dẫn tới hiện tượng Vanishing Gradient khi huấn luyện mạng neural, làm cho các trọng số của mạng không còn được cập nhật nữa. Ngoài ra, việc lấy logarithm là một phép phi tuyến tính hóa, làm cho các đặc trưng khớp hơn với những gì con người nghe thấy, bởi tai người không nghe thấy được âm thanh trên quy mô tuyến tính. Để cho thuận tiện, ta vẫn gọi đặc trưng sau khi lấy logarithm (Hình 3.9) là MFBs.




**Hình 3.9 Đặc trưng Log Mel Filter Banks**

Như vậy, tín hiệu x(t) ban đầu được chia thành các frames nhỏ, mỗi frames được trích xuất đặc trưng là $N_f$ hệ số Mel Filter Banks, biểu diễn tần số từ 0 đến $\frac{f_s}{2}$. Các đặc trưng này có đặc điểm là có sự tương quan rất cao, điều này gây khó khăn cho một số thuật toán ML. Để khắc phục điều này, biến đổi Cosine rời rạc (Discrete Cosine Transform – DCT) được thực hiện để giải tương quan các hệ số MFBs, tạo ra các đặc trưng MFCCs. Với sự ra đời của DL, MFCCs không còn là lựa chọn tốt khi các mạng neural ít bị ảnh hưởng bởi dữ liệu có tính tương quan cao. Ngoài ra, biến đổi DCT được sử dụng để tính toán MFCCs là một biến đổi tuyến tính, do đó nó có thể làm mất một số thông tin trong các tín hiệu âm thanh có tính phi tuyến tính cao.

STFT cũng là một biến đổi tuyến tính, tuy nhiên khi thực hiện STFT, tín hiệu đã được giả sử không thay đổi trong khoảng thời gian ngắn, nên tính tuyến tính của STFT không gây ra ảnh hưởng nghiêm trọng. Gần đây, một số nghiên cứu đã có ý tưởng loại bỏ biến đổi STFT và sử dụng mạng neural học trực tiếp từ tín hiệu trong miền thời gian, điển hình là WaveNet [33]. Tuy nhiên, điều này sẽ làm tăng lượng dữ liệu và tăng độ phức tạp của mô hình cần thiết để có thể đạt được hiệu quả tương tự như khi sử dụng STFT nên cách làm này vẫn chưa phổ biến.

## 3.3 Chuẩn hóa đặc trưng

Sau khi trích xuất được các đặc trưng MFBs, các đặc trưng cần được chuẩn hóa về một khoảng giá trị dữ trước khi đưa vào mạng neural, giúp mạng neural khi huấn luyện không bị phụ thuộc quá vào một vài đặc trưng nào đó. Rescaling là phương pháp chuẩn hóa đơn giản nhất, đưa tất cả thành phần đặc trưng về đoạn [0, 1]. Công thức chuẩn hóa đặc trưng thứ i (tương ứng bộ lọc Filter Bank thứ i) là [15]:

$$m_i' = \frac{m_i - \min(m_i)}{\max(m_i) - \min(m_i)}, \qquad 1 \le i \le N_f \tag{3.19}$$

Trong đó, $m_i$ và $m_i'$ lần lượt là giá trị đặc trưng MFBs thứ i ban đầu và sau khi chuẩn hóa, $\min(m_i)$ và $\max(m_i)$ lần lượt là giá trị nhỏ nhất và lớn nhất của đặc trưng MFBs thứ i xét trên toàn bộ tập huấn luyện. Về biểu diễn ma trận, $m_i$ tương ứng với cột thứ i của ma trận M.

Một phương pháp chuẩn hóa khác được gọi là Standardization, sẽ làm cho mỗi thành phần đặc trưng đều có phân phối chuẩn với kì vọng bằng 0 và phương sai bằng 1. Công thức chuẩn hóa là [15]:

$$m_i' = \frac{m_i - \mu_i}{\sigma_i}, \qquad 1 \le i \le N_f \tag{3.20}$$

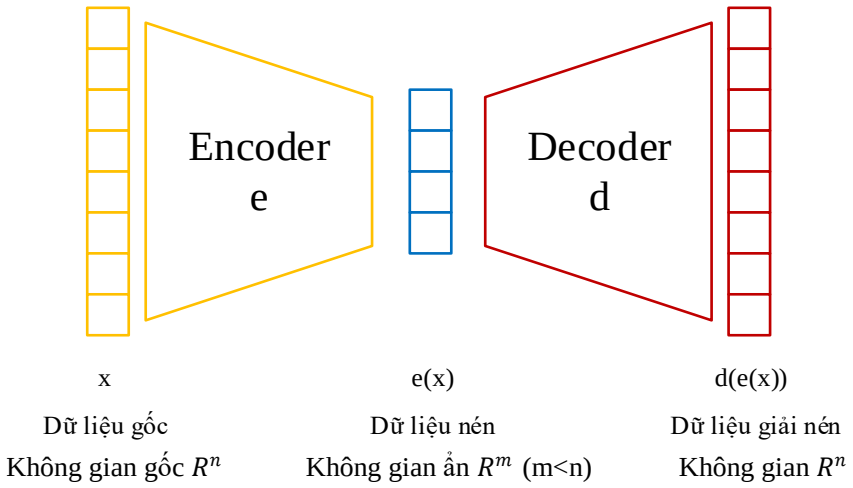
Trong đó, $\mu_i$ và $\sigma_i$ lần lượt là kì vọng và độ lệch chuẩn của đặc trưng MFBs thứ i xét trên toàn bộ tập huấn luyện, nghĩa là trên cột thứ i của ma trận M.

Đồ án này sẽ sử dụng phương pháp Rescaling để chuẩn hóa dữ liệu.

## 3.4 Mạng neural trong bài toán phát hiện âm thanh bất thường

Các mạng neural trong bài toán phát hiện âm thanh bất thường đều được xây dựng dựa trên nguyên lí giảm chiều dữ liệu. Theo khái niệm trong ML, giảm chiều dữ liệu là một tiến trình giảm số lượng đặc trưng của dữ liệu [15]. Số lượng đặc trưng có thể giảm bằng cách lựa chọn những đặc trưng chính hoặc bằng cách trích xuất, nghĩa là tạo ra các đặc trưng mới dựa trên những đặc trưng sẵn có, sao cho số lượng đặc trưng giảm đi. Việc giảm chiều dữ liệu này đặc biệt hữu ích trong những trường hợp mà dữ liệu cần có ít chiều (ví dụ: lưu trữ dữ liệu, tính toán khối lượng lớn, …). Mặc dù có rất nhiều phương pháp khác nhau để giảm chiều dữ liệu, nhưng chúng đều dựa trên một khuôn mẫu, bao gồm bộ mã hóa (Encoder) và bộ giải mã (Decoder).

Một cách tổng quát nhất, Encoder có nhiệm vụ tạo ra sự biểu diễn của các đặc trưng mới dựa trên sự biểu diễn của các đặc trưng cũ (có thể bằng cách chọn lựa hoặc trích xuất). Ở chiều ngược lại, Decoder tái tạo lại những đặc trưng ban đầu dựa vào các đặc trưng mới được tạo ra bởi Encoder. Việc giảm chiều dữ liệu có thể được hiểu như việc nén dữ liệu từ không gian ban đầu thành không gian mã hóa hay không gian ẩn, sau đó giải nén để thu được dữ liệu ban đầu. Việc nén và giải nén dữ liệu chắc chắn sẽ có mất mát dữ liệu, khi mà một phần thông tin bị mất trong quá trình nén và sẽ không thể được khôi phục khi giải nén. Nguyên lí giảm chiều dữ liệu được minh họa trên Hình 3.10.




**Hình 3.10 Nguyên lí giảm chiều dữ liệu**

Trên Hình 3.10, dữ liệu gốc x trong không gian gốc $R^n$ được nén bằng Encoder e thành dữ liệu e(x) trong không gian ẩn $R^m$, với m<n vì mục đích giảm chiều dữ liệu. Sau đó, dữ liệu nén được giải nén bằng Decoder d, thu được dữ liệu d(e(x)) trong không gian $R^n$ giống không gian gốc. Trong trường hợp lí tưởng khi x = d(e(x)), dữ liệu thu được giống hệt dữ liệu gốc vì không bị mất mát thông tin, tuy nhiên trên thực tế sự mất mát thông tin luôn xảy ra, khi đó x ≠ d(e(x)). Công việc cần làm ở đây là tìm ra cặp Encoder/Decoder sao cho việc mất mát là tối thiểu. Cụ thể là, cần tìm Encoder có khả năng giữ được tối đa thông tin khi nén, và từ đó mất mát là tối thiểu khi giải nén. Sự mất mát thông tin được xác định bằng lỗi tái tạo giữa dữ liệu gốc và dữ liệu thu được sau khi giải nén, kí hiệu là $\epsilon(x, d(e(x)))$. Kí hiệu E và D lần lượt là tập Encoder và Decoder, e* và d* lần lượt là Encoder và Decoder cần tìm, ta cần giải bài toán tối ưu:

$$(e^*, d^*) = \underset{(e,d) \in (E,D)}{argmin} \epsilon\left(x, d\left(e(x)\right)\right) \tag{3.21}$$

Phương trình (3.21) được hiểu là, trong tập các Encoders E và Decoders D, cần tìm được Encoder e* và Decoder d* sao cho lỗi tái tạo có giá trị nhỏ nhất. Mô hình như trên Hình 3.10 là mô hình chung cho các mạng neural trong bài toán phát hiện âm thanh bất thường nói riêng và bài toán phát hiện bất thường nói chung.

## 3.5 Tính toán ngưỡng xác định bất thường

Để xác định một điểm dữ liệu có phải là bất thường hay không, dữ liệu được đưa qua mạng neural đã huấn luyện và tính toán lỗi tái tạo, nếu lỗi này lớn hơn một ngưỡng nhất định thì điểm dữ liệu được xác định là bất thường. Việc tính toán ngưỡng này cần được thực hiện trước pha kiểm tra. Ngưỡng xác định bất thường $\tau$ được tính toán như sau:

$$\tau = \mu_\epsilon + \beta\sigma_\epsilon \tag{3.22}$$

Trong đó, $\mu_\epsilon$ và $\sigma_\epsilon$ lần lượt là kì vọng và độ lệch chuẩn của lỗi tái tạo tính trên tập xác thực, $\beta$ là một siêu tham số được xác định từ những thí nghiệm từ trước.

## 3.6 Kết luận chương

Chương 3 đã trình bày chi tiết về mô hình hệ thống phát hiện âm thanh bất thường, phương pháp trích xuất đặc trưng MFB cho dữ liệu âm thanh, đưa ra mô hình chung cho các mạng neural trong bài toán phát hiện âm thanh bất thường và cách tính ngưỡng xác định bất thường.

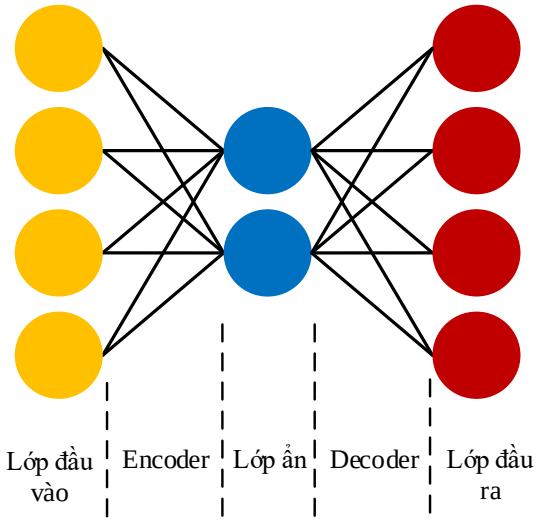
# CHƯƠNG 4. CÁC MÔ HÌNH MẠNG NEURAL SỬ DỤNG TRONG BÀI TOÁN PHÁT HIỆN ÂM THANH BẤT THƯỜNG

Chương 4 trình bày cụ thể về kiến trúc các mô hình mạng neural được sử dụng trong bài toán phát hiện âm thanh bất thường nằm trong phạm vi đồ án này, bao gồm: Autoencoder, Variational Autoencoder, Convolutional Autoencoder và mô hình mạng đề xuất Convolutional Recurrent Autoencoder.

## 4.1 Autoencoder

### *4.1.1 Giới thiệu*

Ý tưởng về bộ mã hóa tự động (Autoencoder – AE) đã có từ cuối thế kỉ XX [34], tuy nhiên chỉ thực sự phát triển trong khoảng 10 năm trở lại đây với sự ra đời của DL [35]. Về cơ bản, AE có mô hình giống như trên Hình 3.10, với Encoder và Decoder là các mạng neural, và việc tìm ra mô hình AE tốt nhất được thực hiện dựa trên một thuật toán tối ưu. Một kiến trúc AE cơ bản được thể hiện trên Hình 4.1.




**Hình 4.1 Kiến trúc AE cơ bản**

Trên Hình 4.1 là một mạng neural đơn giản với một lớp ẩn với hai neurons, hai lớp đầu vào và đầu ra đều có bốn neurons. Lớp ẩn có số neurons ít hơn lớp đầu vào và đầu ra, do đó có thể biểu diễn dữ liệu trên số chiều ít hơn dữ liệu gốc. Đây là kiến trúc cơ bản nhất của AE, có một lớp đầu vào, một lớp đầu ra, có thể có một hoặc nhiều lớp ẩn,

số neurons ở tầng đầu vào và tầng đầu ra luôn bằng nhau. AE là mô hình học không giám sát, vì nó không yêu cầu có nhãn dữ liệu để cho mạng có thể học.

### *4.1.2 Hoạt động của AE*

Xét một AE có một lớp ẩn với m neurons, lớp đầu vào và lớp đầu ra có n neurons. Giả sử vector dữ liệu ban đầu là $x \in R^n$. Gọi $W_{m\times n}$, $b_m$ và $\sigma$ lần lượt là ma trận trọng số, ma trận bias và hàm kích hoạt của Encoder. Đặt z là vector dữ liệu ở tầng ẩn, ta có:

$$z = \sigma(Wx + b) \in R^m \tag{4.1}$$

Tương tự với Encoder, gọi $W'_{n\times m}$, $b'_n$ và $\sigma'$ lần lượt là ma trận trọng số, ma trận bias và hàm kích hoạt của Bộ giải mã, ta thu được vector dữ liệu đầu ra là:

$$x' = \sigma'(W'z + b') \in R^n \tag{4.2}$$

Lỗi tái tạo trên toàn bộ N điểm dữ liệu huấn luyện được tính như sau:

$$\mathcal{L}(x, x') = \frac{1}{N}\|x - x'\|_2^2 = \frac{1}{N}\|x - \sigma'(W'\sigma(Wx + b) + b')\|_2^2 \tag{4.3}$$

Với N là tổng số mẫu huấn luyện. Lỗi tái tạo này có thể được tối thiểu nhờ thuật toán lan truyền ngược được áp dụng cho mạng neural.

Về lí thuyết, khi có càng nhiều lớp ẩn thì AE càng có thể giảm số chiều của dữ liệu đi nhiều lần, thậm chí còn một chiều mà vẫn có thể tái tạo dữ liệu với mất mát nhỏ. Tất nhiên, việc tái tạo dữ liệu với mất mát càng nhỏ phải đánh đổi bằng sự phức tạp của mạng. Ngoài ra, việc chọn số neurons ở lớp ẩn cũng rất quan trọng, nếu chọn quá ít thì khả năng tái tạo dữ liệu sẽ bị giới hạn, nhưng nếu chọn quá nhiều thì mạng sẽ không chọn được những đặc trưng quan trọng của dữ liệu.

### *4.1.3 Những hạn chế của AE*

AE là mạng neural đơn giản nhất dùng để phát hiện bất thường, nó có một số hạn chế nhất định. Hạn chế lớn nhất của AE là nó chỉ có khả năng tái tạo lại những dữ liệu tương tự với dữ liệu đã được huấn luyện. AE cũng rất dễ bị hiện tượng quá khớp vì mục đích của nó là tối thiểu nhất có thể sự sai khác giữa đầu vào và đầu ra. AE không tính đến phân bố dữ liệu của lớp ẩn, nên nó có phần cứng nhắc khi tái tạo dữ liệu. Vì thế, AE không thể được sử dụng như một mô hình sinh, bởi nếu tách riêng Decoder ra để tạo dữ liệu mới, thì chưa chắc  nó có thể tạo được dữ liệu như mong muốn nếu đầu vào của Decoder có sai khác (thậm chí rất nhỏ) so với dữ liệu đã được nén khi huấn luyện.

## 4.2 Variational Autoencoder

### *4.2.1 Giới thiệu*

Để khắc phục những nhược điểm của AE, vào năm 2013, Diederik P. Kingma và Max Welling đã đưa ra bộ mà hóa tự động biến thể (Variational Autoencoder – VAE) [36]. VAE hoạt động cũng giống như AE, nhưng dữ liệu ở tầng ẩn được biểu diễn dưới dạng một phân phối xác suất hay vì các neurons cố định, do đó tránh được hiện tượng quá khớp và có khả năng sinh dữ liệu mới.

Đối với VAE, Bộ mã hóa được huấn luyện để tính toán kì vọng và phương sai mô tả phân phối Gaussians (hay phân phối chuẩn), từ đó biểu diễn dữ liệu của không gian ẩn. Hàm mất mát của VAE là kết hợp của hai phần: phần tái tạo giống như của AE, và phần regularization sử dụng để xây dựng phân phối của không gian ẩn gần với phân phối chuẩn. Phần regularization này được thể hiện dưới dạng phân kì Kullback-Leibler (Kulback-Leibler Divergence) giữa phân phối tìm được và phân phối chuẩn.

### *4.2.2 Cơ sở toán học của VAE*

Gọi x là biến ngẫu nhiên biểu diễn dữ liệu ban đầu, z là biến ngẫu nhiên trong không gian ẩn. Giả sử z được lấy mẫu từ phân phối $p_{\theta^*}(z)$, x được lấy mẫu từ phân phối có điều kiện $p_{\theta^*}(x|z)$, trong đó $\theta$ là tham số (kì vọng, phương sai) của phân phối. Ngoài ra, giả sử rằng $p_{\theta^*}(z)$ và $p_{\theta^*}(x|z)$ thuộc các họ tham số của phân phối $p_\theta(z)$ và $p_\theta(x|z)$ mà hàm mật độ xác suất của chúng khả vi tại mọi điểm. Khác với Encoder và Decoder của AE ở dạng xác định, thì Encoder và Decoder của VAE sẽ ở dạng xác suất. Encoder của VAE được xác định bởi $p_\theta(z|x)$ mô tả phân phối của biến ngẫu nhiên z. Trong khí đó, Decoder của VAE được xác định bởi $p_\theta(x|z)$ mô tả phân phối của biến ngẫu ngẫu nhiên được tái tạo lại.

Theo định lí Bayes trong xác suất, ta có:

$$p_\theta(z|x) = \frac{p_\theta(x|z)p_\theta(z)}{p_\theta(x)} \tag{4.4}$$

Vì $p_\theta(x) = \int p_\theta(x|z)p_\theta(z)dz$ nên:

$$p_\theta(z|x) = \frac{p_\theta(x|z)p_\theta(z)}{\int p_\theta(x|z)p_\theta(z)dz} \tag{4.5}$$

Với những giả sử vừa nêu trên, thì việc tính toán $p_\theta(z|x)$ thực sự khó khăn, vì ta chưa biết $\theta^*$ và z, vì thế ở đây cần sử dụng kĩ thuật xấp xỉ. Gọi $q_\phi(z|x)$ là một mô hình xấp xỉ với $p_\theta(z|x)$. Trong không gian ẩn, dữ liệu được lấy mẫu và đưa vào Decoder để sinh dữ liệu.

Một phương pháp phổ biến để xấp xỉ hai phân phối là phân kì Kullback-Leibler. Hàm mất mát do xấp xỉ phân phối bằng phân kì Kullback-Leibler là [36]:

$$KL\left(q_\phi(z|x), p_\theta(z|x)\right) = \sum q_\theta(z|x) log \frac{q_\phi(z|x)}{p_\theta(z|x)} \tag{4.6}$$

Biến đổi (4.6) một chút, ta được:

$$KL\left(q_\phi(z|x), p_\theta(z|x)\right) = E\big(\log q_\phi(z|x) - \log p_\theta(z|x)\big) \tag{4.7}$$

Thay (4.4) vào (4.7), ta có:

$$KL\left(q_\phi(z|x), p_\theta(z|x)\right) = E\big(\log q_\phi(z|x) - \log p_\theta(z) - \log p_\theta(x|z) + \log p_\theta(x)\big) \tag{4.8}$$

Tiếp tục biến đổi (4.8), thu được:

$$KL\left(q_\phi(z|x), p_\theta(z|x)\right) = E(\log p_\theta(x) - \log p_\theta(x|z)) + KL\left(q_\phi(z|x), p_\theta(z)\right) \tag{4.9}$$

Do kì vọng được lấy theo biến z, nên $p_\theta(x)$ không phụ thuộc vào z, do đó:

$$\log p_\theta(x) = KL\left(q_\phi(z|x), p_\theta(z|x)\right) + E(\log p_\theta(x|z)) - KL\left(q_\phi(z|x), p_\theta(z)\right) \tag{4.10}$$

Ở đây, $\log p_\theta(x)$ được gọi là Marginal Likelihood.

Đặt $\mathcal{L}(\theta, \phi) = E(\log p_\theta(x|z)) - KL\left(q_\phi(z|x), p_\theta(z)\right)$, ta có thể viết lại (4.10):

$$\log p_\theta(x) = KL\left(q_\phi(z|x), p_\theta(z|x)\right) + \mathcal{L}(\theta, \phi) \tag{4.11}$$

Vì số hạng đầu tiên của (4.11) luôn không âm, nên $\log p_\theta(x) \geq \mathcal{L}(\theta, \phi)$, và $\mathcal{L}(\theta, \phi)$ được gọi là cận dưới của Marginal Likelihood $\log p_\theta(x)$. Mục đích của VAE là tối ưu $\mathcal{L}(\theta, \phi)$ cho cả hai tham số $\theta$ và $\phi$.

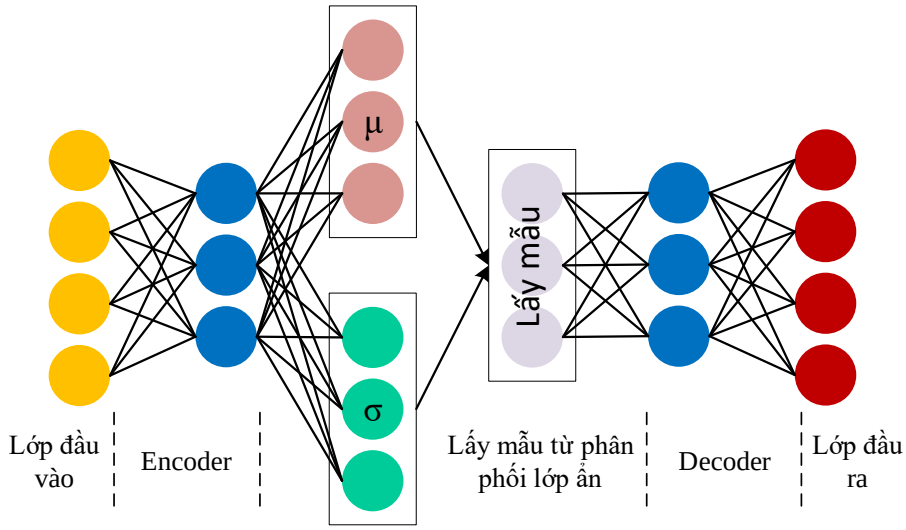
Số hạng thứ nhất của $\mathcal{L}(\theta, \phi)$, $E(\log p_\theta(x|z))$ là một ước lượng hợp lí cực đại (Maximum Likelihood Estimation – MLE), có vai trò giống như hàm mất mát của AE. Nhưng thay vì tối thiểu hóa, ta mong muốn tối đa hóa số hạng này vì nó thể hiện phân phối của biến ngẫu nhiên được sinh ra từ Decoder.

Số hạng thứ hai của $\mathcal{L}(\theta, \phi)$, $KL\left(q_\phi(z|x), p_\theta(z)\right)$ thể hiện sự khác nhau của hai phân phối $q_\phi(z|x)$ và $p_\theta(z)$, do đó ta mong muốn số hạng này càng nhỏ càng tốt. Để đơn giản nhất, cho rằng $p_\theta(z)$ tuân theo phân phối chuẩn $\mathcal{N}(0, 1)$, từ đó $q_\phi(z|x)$ cần được tối ưu để gần với phân phối chuẩn $\mathcal{N}(0, 1)$ nhất có thể. Giả sử $q_\phi(z|x)$ tuân theo phân phối Gaussian với kì vọng và phương sai lần lượt là μ và $\sigma^2$. Sử dụng thuật toán tối ưu Bayes biến thể mã hóa tự động (Auto-encoding Variational Bayesian – AEVB) được các tác giả trình bày trong [36], $KL\left(q_\phi(z|x), p_\theta(z)\right)$ được tối ưu như sau:

$$KL\left(q_\phi(z|x), p_\theta(z)\right) = -\frac{1}{2}(1 - \mu^2 - \sigma^2 + \log \sigma^2) \tag{4.12}$$

### *4.2.3 Triển khai VAE trên mạng neural*

Hình 4.2 thể hiện một kiến trúc mạng neural cơ bản của VAE.




**Hình 4.2 Kiến trúc VAE cơ bản**

Các lớp đầu vào, đầu ra, phần Encoder và Decoder của VAE cũng giống như AE, điểm khác biệt duy nhất nằm ở lớp ẩn. Thay vì được biểu diễn là một lớp thông thường như AE, lớp ẩn của VAE được biểu diễn dưới dạng các phân phối với kì vọng và độ lệch chuẩn được thể hiện dưới dạng các neurons, với số neurons bằng số chiều lớp ẩn. Sau đó, lớp tiếp theo có các neurons là các mẫu được lấy từ các phân phối này.

## 4.3 Convolutional Autoencoder

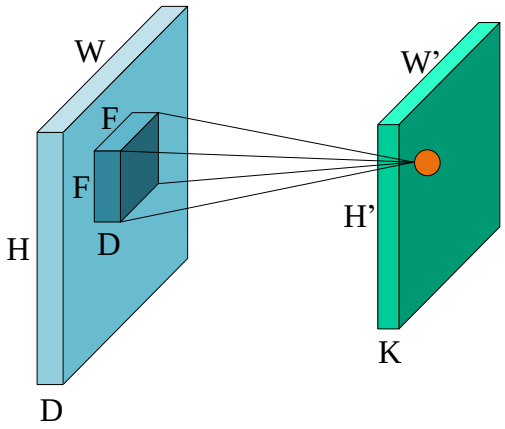
### *4.3.1 Giới thiệu*

Các mạng neural AE và VAE được trình bày ở trên đều nhận dữ liệu đầu vào là các hệ số MFBs của một frame âm thanh. Tuy nhiên, dựa vào Hình 3.9, ta nhận thấy hoàn toàn có thể sử dụng dữ liệu dưới dạng ảnh để huấn luyện mạng neural. Mỗi frame bao gồm các hệ số MFBs, nếu ghép các frames lại ta sẽ thu được dữ liệu dạng ảnh. Ảnh ghép được chứa dữ liệu về MFBs của nhiều frames cạnh nhau, thay vì chỉ một frame đơn lẻ. Một loại mạng neural chuyên dùng để xử lí dữ liệu ảnh là mạng neural tích chập (Convolutional Neural Network – CNN), và bộ mã hóa tự động tích chập (Convolutional Autoencoder – CAE) sẽ được xây dựng dựa vào các lớp Convolution trong CNN [37]. Sử dụng CAE để tái tạo dữ liệu ảnh MFBs hoàn toàn có thể mang lại kết quả tốt hơn so với AE và VAE chỉ tái tạo từng MFBs của từng frame.

Kiến trúc của CAE cũng bao gồm hai phần Encoder và Decoder như AE và VAE, nhưng thay vì là các lớp neuron một chiều, Encoder và Decoder của CAE lần lượt là các lớp tích chập (Convolution) và lớp tích chập chuyển vị (Transposed Convolution). Các lớp này cũng bao gồm các neurons như các lớp thông thường, nhưng các neurons được sắp xếp theo ba chiều, giống như ba chiều của một bức ảnh (chiều rộng, chiều cao và chiều sâu là các kênh màu). Ngoài ra, thay vì các kết nối giữa các neurons, các lớp Convolution sử dụng bộ lọc dịch để tính toán.

### *4.3.2 Lớp Convolution*

Lớp Convolution được sử dụng làm phần Encoder của CAE. Cấu tạo lớp Convolution được minh họa trên Hình 4.3.




**Hình 4.3 Lớp Convolution**

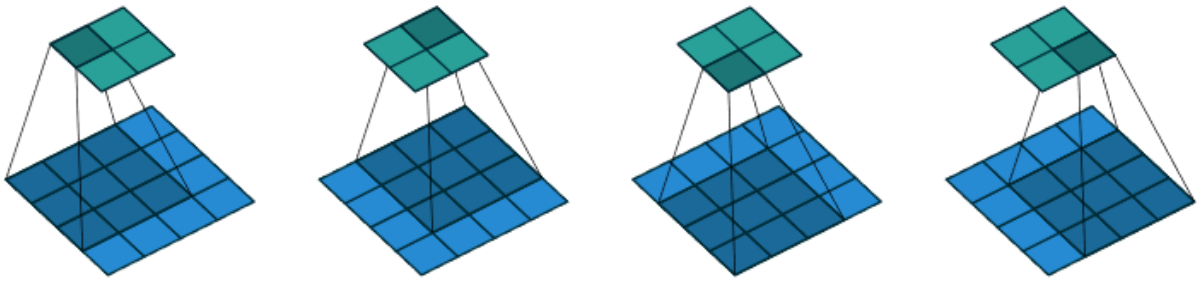
Lớp Convolution có kích thước $W \times H \times D$, nghĩa là W pixels chiều rộng, H pixels chiều cao và D kênh màu. Lớp Convolution tính toán nhờ một tập các bộ lọc dịch kích thước $F \times F \times D$, chứa các tham số mà mạng có thể học được. Mỗi bộ lọc này được dịch dọc theo chiều rộng và chiều cao của lớp và tính tích vô hướng giữa các tham số bộ lọc và các giá trị điểm ảnh tại vị trí bộ lọc dịch đến. Khi dịch bộ lọc hết toàn bộ lớp, một lớp hai chiều kích thước $W' \times H'$ được tạo ra với mỗi giá trị neuron là kết quả của phép tích chập tại một vị trí của bộ lọc ở lớp trước. Sau khi thực hiện dịch và tính toán trên K bộ lọc, lớp được tạo ra là một lớp ba chiều kích thước $W' \times H' \times K$. Lớp này lại sử dụng một tập các bộ lọc khác và các lớp tiếp theo cũng được tính toán tương tự.

Độ dịch của mỗi bộ lọc được gọi là bước nhảy, kí hiệu là S. Rõ ràng các pixels nằm ở ngoài cùng của lớp ít được bộ lọc dịch đến hơn các pixels phía trong, nên kích thước lớp tiếp theo luôn nhỏ hơn kích thước lớp trước. Tính chất này của lớp Convolution được tận dụng để giảm chiều dữ liệu. Cụ thể hơn, từ lớp trước có kích thước $W \times H \times D$, sử dụng K bộ lọc kích thước $F \times F \times D$ và bước nhảy bằng S, lớp tiếp theo sẽ có kích thước $W' \times H' \times K$ với $W'$ và $H'$ được tính như sau:

$$W' = \frac{W - F}{S} + 1 \tag{4.13}$$

$$H' = \frac{H - F}{S} + 1 \tag{4.14}$$

Hình 4.4 minh họa ví dụ lớp Convolution với $W = H = 4$, $F = 3$, $S = 1$. Theo các công thức (4.13) và (4.14), ta có $W' = H' = (4 - 3)/1 + 1 = 2$.

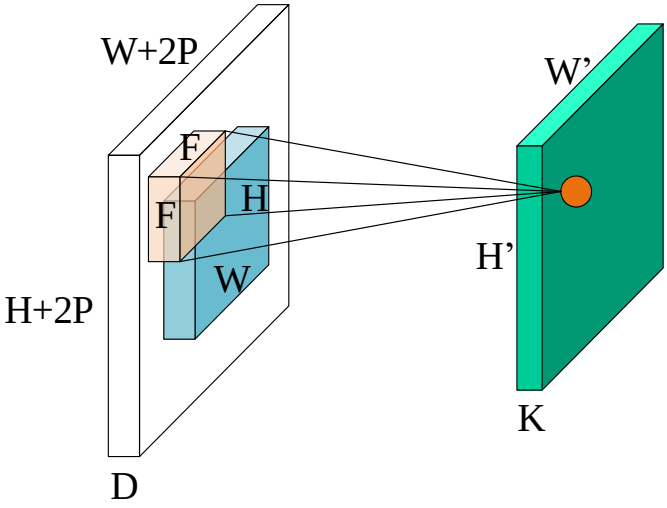


**Hình 4.4 Ví dụ lớp Convolution với W=H=4, F=3, S=1 [38]**

Đối với CAE thì F, S và K được gọi là các siêu tham số, thông thường thì $F = 3$ và $S = 1$. Mỗi bộ lọc bao gồm $F \cdot F \cdot D$ trọng số và K bộ lọc sẽ bao gồm $F \cdot F \cdot D \cdot K$ trọng số và K biases. Số lượng tham số mô hình của CAE lớn hơn rất nhiều so với AE và VAE, nên tốc độ huấn luyện cũng chậm hơn nhưng đổi lại cũng mang tới kết quả tốt hơn.

### *4.3.3 Lớp Convolution chuyển vị*

Lớp Convolution chuyển vị, có cấu tạo ngược với lớp Convolution khi kích thước lớp sau lớn hơn kích thước lớp trước. Lớp Convolution chuyển vị được sử dụng làm phần Decoder của CAE. Hình 4.5 thể hiện cấu tạo lớp Convolution chuyển vị.



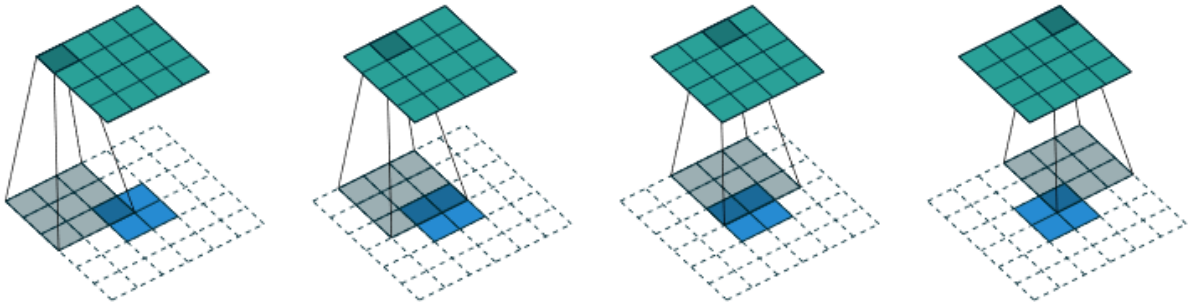

**Hình 4.5 Lớp Convolution chuyển vị**

Kích thước đầu vào lớp Convolution chuyển vị là $W \times H \times D$. Lớp Convolution chuyển vị cũng sử dụng các bộ lọc dịch để tính giá trị các neurons lớp kế tiếp giống lớp Convolution. Tuy nhiên để có thể sử dụng lớp Convolution chuyển vị làm phần Decoder của CAE, ta cần áp dụng kĩ thuật đệm không (zero-padding), thêm các giá trị 0 vào xung quanh ảnh đầu vào, làm tăng kích thước lớp, từ đó làm cho kích thước lớp sau lớn hơn kích thước đầu vào ban đầu. Trên Hình 4.5, khối màu xanh nước biển biểu diễn đầu vào ban đầu của lớp hiện tại, phần màu trắng xung quanh biểu diễn các giá trị zero-padding và khối màu hồng biểu diễn bộ lọc kích thước $F \times F \times D$. Đặt kích thước zero-padding (theo một cạnh) là P, ta có kích thước lớp sau khi đệm là $(W + 2P) \cdot (H + 2P) \cdot D$. Với số bộ lọc sử dụng là K, bước nhảy là S, lớp kế tiếp sẽ có kích thước $W' \times H' \times K$ với:

$$W' = \frac{W + 2P - F}{S} + 1 \tag{4.15}$$

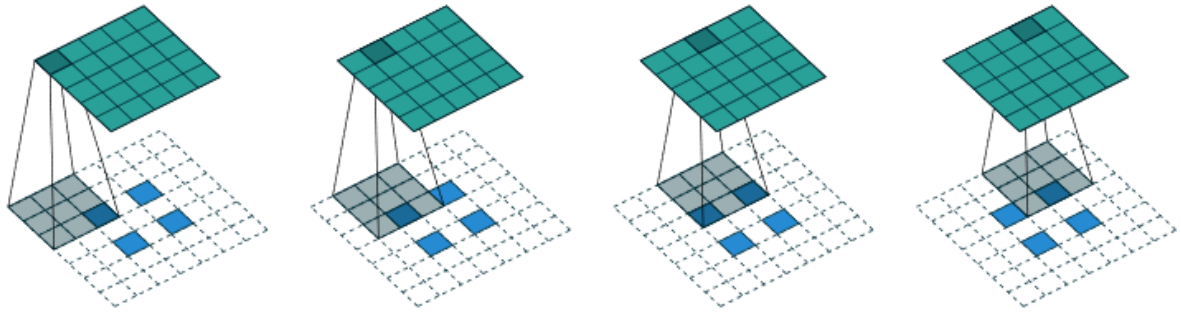
$$H' = \frac{H + 2P - F}{S} + 1 \tag{4.16}$$

Hình 4.6 là một ví dụ lớp Convolution chuyển vị với $W = H = 2$, $F = 3$, $S = 1$, $P = 2$. Theo các công thức (4.15) và (4.16), $W' = H' = (2 + 2.2 - 3)/1 + 1 = 4$.



**Hình 4.6 Ví dụ lớp Convolution chuyển vị với với W=H=2, F=3, S=1, P=2 [38]**

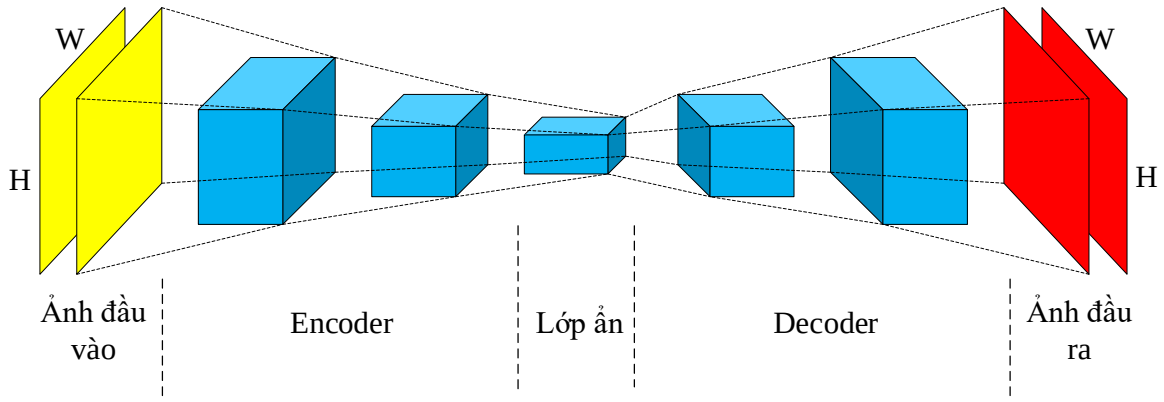
Bên cạnh cách đệm các giá trị 0 vào xung quanh ảnh ban đầu, một cách đệm khác là thêm các giá trị 0 xen kẽ giữa các pixels đầu vào kết hợp với zero-padding thông thường. Kí hiệu P' là kích thước zero-padding xen kẽ giữa các pixels. Hình 4.7 minh họa một ví dụ lớp Convolution chuyển vị có zero-padding xen kẽ với $W = H = 2$, $F = 3$, $S = 1$, $P = 2$, $P' = 1$. Kích thước lớp tiếp theo là: $W' = H' = (2 + 2.2 + 1 - 3)/1 + 1 = 5$.



**Hình 4.7 Ví dụ lớp Convolution chuyển vị sử dụng zero-padding xen kẽ với W=H=2, F=3, S=1, P=2, P'=1 [38]**

### *4.3.4 Kiến trúc CAE*

CAE dược tạo ra từ Encoder là các lớp Convolution ghép với Decoder là các lớp Convolution chuyển vị. Kiến trúc của CAE được thể hiện trên Hình 4.8.




**Hình 4.8 Kiến trúc CAE**

Đối với dữ liệu là các đặc trưng MFBs, ảnh đầu vào có kích thước $W \times H \times D$, với W là số frames ghép lại với nhau, H là số Mel Filter Banks của mỗi frame và D là số kênh âm thanh. Ảnh được tái tạo ở đầu ra có kích thước giống hệt ảnh đầu vào. Để cho đơn giản, ta chỉ xét trên một kênh của ảnh MFBs. Mỗi kênh của ảnh đầu vào có thể được biểu diễn dưới dạng ma trận X:

$$X = \begin{bmatrix} x_{11} & x_{12} & \cdots & x_{1W} \\ x_{21} & x_{22} & \cdots & x_{2W} \\ \vdots & \vdots & \ddots & \vdots \\ x_{H1} & x_{H2} & \cdots & x_{HW} \end{bmatrix} \tag{4.17}$$

Tương tự, ma trận X' biểu diễn mỗi kênh của ảnh đầu ra:

$$X' = \begin{bmatrix} x'_{11} & x'_{12} & \cdots & x'_{1W} \\ x'_{21} & x'_{22} & \cdots & x'_{2W} \\ \vdots & \vdots & \ddots & \vdots \\ x'_{H1} & x'_{H2} & \cdots & x'_{HW} \end{bmatrix} \tag{4.18}$$

Lỗi tái tạo trên toàn bộ N điểm dữ liệu huấn luyện được xác định là:

$$\mathcal{L}(X, X') = \frac{1}{NHW} \|X - X'\|_2^2 = \frac{1}{NHW} \sum_{i=1}^{W} \sum_{j=1}^{H} \left(x_{ij} - x'_{ij}\right)^2 \tag{4.19}$$

Ảnh đầu ra được tính toán dựa vào các trọng số và bias của các bộ lọc. Một bộ lọc kích thước $F \times F \times D$ có ma trận trọng số (chỉ xét trên một kênh) như sau:

$$Q = \begin{bmatrix} w_{11} & w_{12} & \cdots & w_{1F} \\ w_{21} & w_{22} & \cdots & w_{2F} \\ \vdots & \vdots & \ddots & \vdots \\ w_{F1} & w_{F2} & \cdots & w_{FF} \end{bmatrix} \tag{4.20}$$

Các trọng số này (và bias của mỗi bộ lọc) cũng sẽ được cập nhật nhờ thuật toán lan truyền ngược như trong các mạng FNN, nhằm mục đích tối thiểu hóa lỗi tái tạo.

Cũng giống như AE, CAE càng có nhiều lớp thì tái tạo ảnh càng chính xác. Tuy nhiên như đã trình bày ở trên, số lượng tham số của CAE là rất lớn, nên việc lựa chọn các siêu tham số, đặc biệt là kích thước bộ lọc F và số lượng bộ lọc K, cần được tính toán cẩn thận. Nếu F và K quá nhỏ có thể dẫn đến mất mát thông tin đặc trưng, ngược lại nếu quá lớn sẽ làm cho mô hình phức tạp và tốn rất nhiều thời gian huấn luyện.

## 4.4 Convolutional Recurrent Autoencoder

### *4.4.1 Giới thiệu*

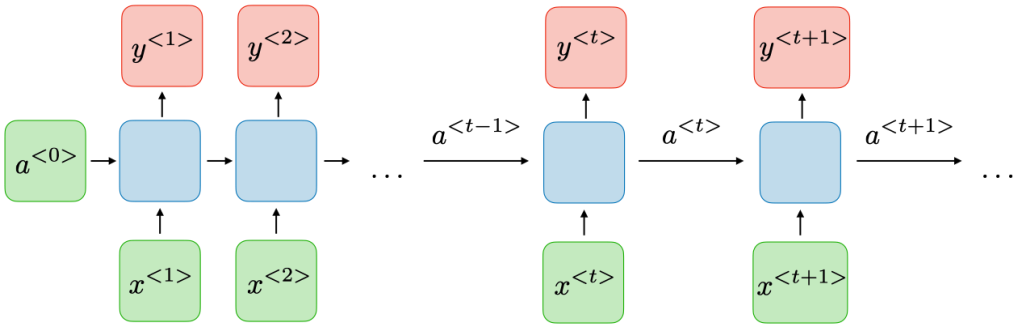
Từ đầu đến giờ, các mạng neural đã được xây dựng với độ phức tạp tăng dần, từ AE đến VAE và CAE. Độ phức tạp của mạng càng tăng thì độ hiệu quả của mạng cũng tăng theo, và mạng CAE mới xây dựng là mạng có độ phức tạp cao nhất. Tuy nhiên, tất cả các mạng trên đều có một điểm chung là không xét đến yếu tố thời gian của âm thanh. Về bản chất, âm thanh là một chuỗi các frames theo thời gian, do đó dữ liệu âm thanh là dữ liệu có các dạng chuỗi thời gian. Các mạng neural đã trình bày ở trên chỉ cố gắng tái tạo đầu vào bằng cách học từ một mẫu huấn luyện tại một thời điểm mà không học từ các mẫu trước đó. Một frame âm thanh có thể phụ thuộc vào các frames trước đó, vì vậy việc không xét đến tính thời gian của dữ liệu dạng chuỗi thời gian như dữ liệu âm thanh là một hạn chế của các mạng neural đã xây dựng.

Mạng neural truy hồi (Recurrent Neural Network – RNN) được tạo ra để xử lí dữ liệu dạng chuỗi thời gian. RNN bao gồm các mạng nhỏ được ghép lại thành chuỗi, đầu vào của mỗi mạng là một điểm dữ liệu và đầu ra của mỗi mạng lại được đưa vào mạng kế tiếp. Như vậy, RNN không chỉ học từ một điểm dữ liệu tại một thời điểm mà còn học từ các điểm dữ liệu trước đó của chuỗi. Bộ mã hóa tự động truy hồi tích chập (Convolutional Recurrent Autoencoder – CRAE) được mà đồ án này đề xuất là kết hợp của CAE và RNN, giúp giải quyết vấn đề về tính thời gian của các mạng neural trước.

### *4.4.2 Recurrent Neural Network*

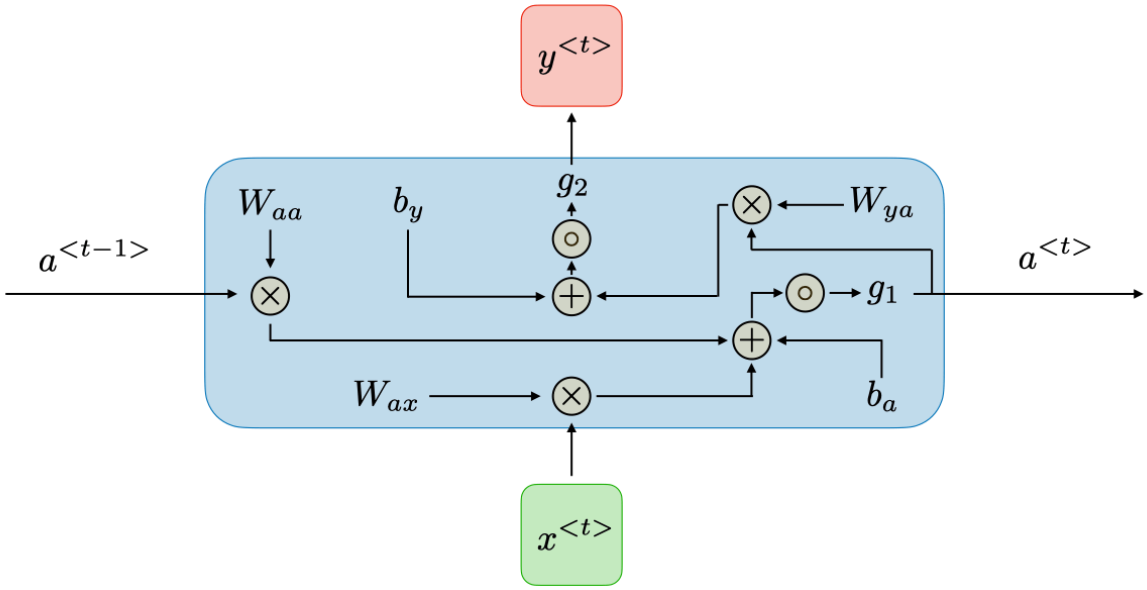
#### *4.4.2.1 Kiến trúc điển hình*

RNN là loại mạng neural cho phép đầu ra trước đó có thể được sử dụng làm đầu vào và có những trạng thái ẩn. Kiến trúc điển hình của RNN được thể hiện trên Hình 4.9.




**Hình 4.9 Kiến trúc điển hình của RNN [39]**

Trên Hình 4.9, các khối màu xanh nước biển được gọi là mạng con (hay cell) của RNN, $x^{<t>}$ và $y^{<t>}$ lần lượt là đầu vào và đầu ra của một cell tại thời điểm t, $a^{<t-1>}$ và $a^{<t>}$ được gọi là activation đầu vào và đầu ra của cell tại thời điểm t. Cấu trúc một cell được thể hiện trên Hình 4.10.




**Hình 4.10 Cấu trúc một cell của RNN [39]**

Tại thời điểm t, activation $a^{<t>}$ và đầu ra $y^{<t>}$ được xác định như sau [39]:

$$a^{<t>} = g_1(W_{aa}a^{<t-1>} + W_{ax}x^{<t>} + b_a) \quad (4.21)$$

$$y^{<t>} = g_2(W_{ya}a^{<t>} + b_y) \quad (4.22)$$

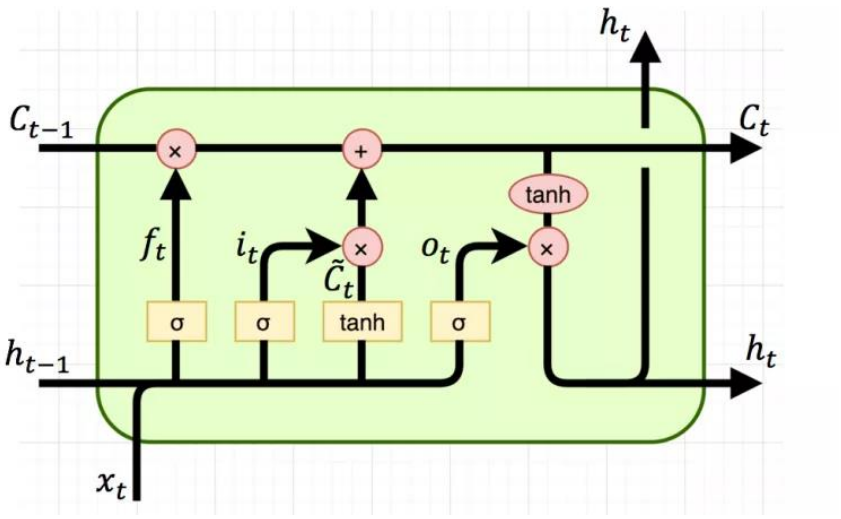
Trong đó $W_{aa}$, $W_{ax}$, $W_{ya}$, $b_a$, $b_y$ là các ma trận trọng số và bias giống như của mạng neural thông thường, $g_1$ và $g_2$ là các hàm kích hoạt.

Mặc dù có ưu điểm xử lí được các thông tin theo thời gian, RNN có một nhược điểm là tốc độ huấn luyện chậm và khó có thể truy cập thông tin từ thời điểm rất xa trước đó.

Cell trên Hình 4.10 chỉ là một loại cell điển hình của RNN. Mạng RNN cũng được tối ưu bằng thuật toán lan truyền ngược, và do phải xử lí dữ liệu từ các thời điểm trước nên RNN rất dễ gặp phải hiện tượng Vanishing Gradient khi giá trị cập nhật ngày càng tiến dần về 0 khi truyền về các cell trước đó. Vì thế, RNN sẽ quên những gì đã thấy khi gặp chuỗi dài nên được gọi là có bộ nhớ ngắn hạn (Short-term Memory). Hai loại mạng có tên bộ nhớ dài ngắn hạn (Long Short-term Memory – LSTM) và đơn vị truy hồi theo cổng (Gated Recurrent Unit – GRU) được tạo ra để giải quyết vấn đề này của RNN.

*4.4.2.2 LSTM*

Cấu trúc của một cell LSTM được thể hiện trên Hình 4.11.




**Hình 4.11 Cấu trúc cell LSTM [40]**

Dữ liệu trong cell LSTM đi theo chiều từ trái sang phải, với $x_t$ là đầu vào hiện tại, $c_{t-1}$ và $h_{t-1}$ là đầu ra của cell trước, được gọi là trạng thái cell và trạng thái ẩn (theo thứ tự) của LSTM. $x_t$ và $h_{t-1}$ được ghép lại với nhau và đi vào đường dưới cùng. Trên Hình 4.11, kí hiệu $\otimes$ dùng để chỉ phép nhân theo từng phần tử của ma trận, $\sigma$ là hàm kích hoạt sigmoid. LSTM có ba loại cổng có tên là cổng quên (forget gate), cổng đầu vào (input gate) và cổng đầu ra (output gate). Các cổng này được sử dụng để tính toán các trạng thái cell và trạng thái ẩn của LSTM.

Đầu ra của forget gate $f_t$ được tính như sau:

$$f_t = \sigma\left(U_f \times x_t + W_f \times h_{t-1} + b_f\right) \tag{4.23}$$

Trong đó, $U_f$ và $b_f$ là ma trận trọng số và ma trận bias của $x_t$ trong forget gate, $W_f$ là ma trận trọng số của $h_{t-1}$ trong input gate.

Tiếp theo tại input gate, đầu ra $i_t$ của cổng này là:

$$i_t = \sigma(U_i \times x_t + W_i \times h_{t-1} + b_i) \tag{4.24}$$

Trong đó, $U_i$ và $b_i$ là ma trận trọng số và ma trận bias của $x_t$ trong input gate, $W_i$ là ma trận trọng số của $h_{t-1}$ trong input gate.

Tại output gate, đầu ra $o_t$ được xác định như sau:

$$o_t = \sigma(U_o \times x_t + W_o \times h_{t-1} + b_o) \tag{4.25}$$

Trong đó, $U_o$ và $b_o$ là ma trận trọng số và ma trận bias của $x_t$ trong output gate, $W_o$ là ma trận trọng số của $h_{t-1}$ trong output gate.

Trạng thái cell $c_t$ được tính như sau:

$$c_t = f_t \times c_{t-1} + i_t \times \tilde{c}_t \tag{4.26}$$

Với:

$$\tilde{c}_t = \tanh(U_c \times x_t + W_c \times h_{t-1} + b_c) \tag{4.27}$$

Trong đó, $U_c$ và $b_c$ là ma trận trọng số và ma trận bias của $x_t$ đối với trạng thái cell, $W_c$ là ma trận trọng số của $h_{t-1}$ đối với trạng thái cell.

Cuối cùng, trạng thái ẩn $h_t$ được xác định:

$$h_t = o_t \times \tanh(c_t) \tag{4.28}$$

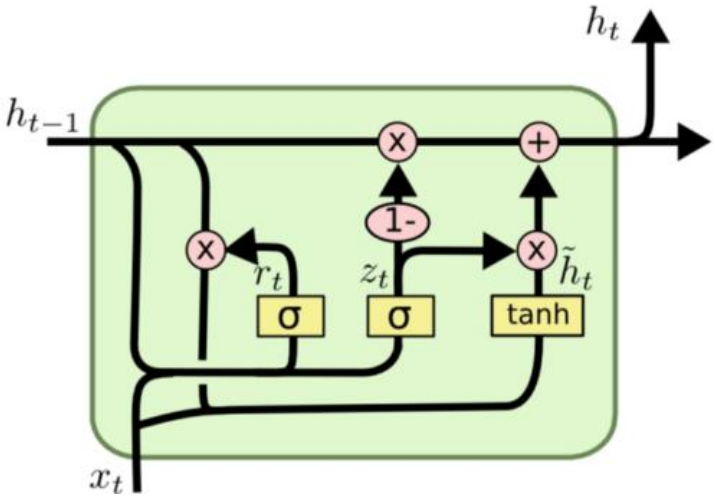
Khi thực hiện thuật toán lan truyền ngược cho RNN, ta cần phải tính $\frac{\partial c_t}{\partial c_{t-1}}$. Từ công thức (4.26), ta có:

$$\frac{\partial c_t}{\partial c_{t-1}} = f_t \tag{4.29}$$

Như vậy, thành phần $i_t \times \tilde{c}_t$ đã bị loại bỏ và chỉ giữ lại thành phần $f_t$. Khi mang thông tin trên trạng thái cell thì ít khi cần phải quên giá trị cell cũ nên $f_t \approx 1$, dẫn đến đạo hàm ít bị suy giảm nên giảm thiểu được Vanishing Gradient. Một cách tổng quát, forget gate có vai trò loại bỏ các thành phần không cần thiết (thông qua huấn luyện) nhằm giảm tải "bộ nhớ" cho RNN, giúp mạng nhớ được các thông tin từ xa trước đó. Input gate là cổng xác định lượng thông tin mới nhận vào và cập nhật trạng thái cell. Cuối cùng, output gate xác định trạng thái ẩn mới, cả trạng thái cell và trạng thái ẩn đều được đưa tới mốc thời gian tiếp theo.

*4.4.2.3 GRU*

GRU là một loại cell khác trong mạng RNN, là một phiên bản mới hơn của LSTM. GRU loại bỏ trạng thái cell và chỉ sử dụng trạng thái ẩn để truyền thông tin. GRU chỉ có hai loại cổng là cổng đặt lại (reset gate) và cổng cập nhật (update gate). Cấu trúc cell GRU được thể hiện trên Hình 4.12.




**Hình 4.12 Cấu trúc cell GRU [40]**

Về cơ bản thì cấu trúc GRU cũng khá giống với LSTM. Đối với GRU, $r_t$, $z_t$ được gọi lần lượt là đầu ra của các cổng đặt lại (reset gate) và cổng cập nhật (update gate). Chúng được tính toán như sau:

$$r_t = \sigma(U_r \times x_t + W_r \times h_{t-1}) \tag{4.30}$$

$$z_t = \sigma(U_z \times x_t + W_z \times h_{t-1}) \tag{4.31}$$

Trong đó, $U_r$ và $W_r$ là các ma trận trọng số của $x_t$ và $h_{t-1}$ trong reset gate, $U_z$ và $W_z$ là các ma trận trọng số của $x_t$ và $h_{t-1}$ trong update gate.

Trạng thái ẩn $h_t$ của GRU được xác định là:

$$h_t = (1 - z_t) \times h_{t-1} + z_t \times \tilde{h}_t \tag{4.32}$$

Với:

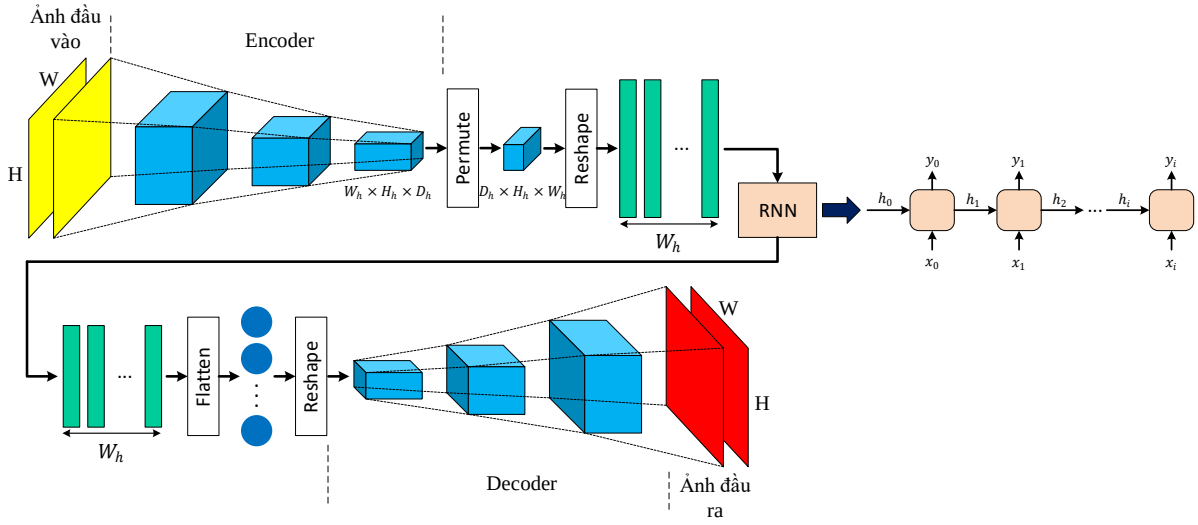
$$\tilde{h}_t = \tanh(U_h \times x_t + W_h \times h_{t-1} \times r_t) \tag{4.33}$$

Trong đó, $U_h$ và $W_h$ là các ma trận trọng số của $x_t$ và $h_{t-1}$ đối với trạng thái ẩn.

Update gate của GRU là cổng hoạt động tương tự như forget gate và input gate của LSTM, nó xác định lượng thông tin cần bỏ đi và lượng thông tin mới được thêm vào. Reset gate là một cổng khác cũng dùng để xác định lượng thông tin trong quá khứ cần quên đi. So với LSTM, GRU có ít thành phần hơn trong cell, do đó tốc huấn luyện nhanh hơn nhưng không đáng kể. Trên thực tế, độ hiệu quả của LSTM và GRU là khá ngang nhau và chúng đều được sử dụng phổ biến trong các mạng RNN.

### *4.4.3 Kiến trúc CRAE*

Như đã giới thiệu ở mục 4.4.1, mạng CRAE là sự kết hợp của CAE và RNN, cụ thể là thêm RNN vào giữa hai phần Encoder và Decoder của CAE. Kiến trúc của CRAE được thể hiện trên Hình 4.13.




**Hình 4.13 Kiến trúc CRAE**

Phần Encoder và Decoder của CRAE hoàn toàn giống với CAE. Tuy nhiên, sau khi các ảnh MFBs đi qua phần Encoder, thì thay vì được đưa sang phần Encoder để khôi phục thì chúng được đưa vào một mạng RNN. Cụ thể hơn, sau lớp Convolution cuối cùng, ảnh được hoán vị chiều để xác định chiều thời gian và chiều đặc trưng. Ví dụ, ảnh MFBs sau lớp Convolution cuối cùng có kích thước $W_h \times H_h \times D_h$ sau đi qua khối Permute sẽ có kích thước $D_h \times H_h \times W_h$. Tiếp theo, ảnh được reshaped để tạo ra chuỗi có $W_h$ timesteps và $D_h \times H_h$ đặc trưng. Các timesteps này tiếp tục được đưa vào các cells của RNN, đầu ra của mỗi cell là các timesteps với số đặc trưng là số neuron units của RNN. Các khối Flatten và Reshape tiếp theo chuyển các timesteps này về dạng ba chiều để đưa vào các lớp Convolution chuyển vị trong phần Decoder của CRAE để khôi phục lại ảnh MFB ban đầu.

Hàm lỗi tái tạo của CRAE có dạng giống như của CAE trong công thức (4.19). Mạng CRAE đã tận dụng được ưu điểm của RNN khi đưa RNN vào lớp ẩn của CAE, giúp tái tạo ảnh MFBs tốt hơn. Mạng này có kiến trúc phức tạp nhất trong các mạng đã xây dựng, nhưng tương xứng với đó là độ hiệu quả (sẽ được thể hiện ở mục 5.4). CRAE là một mạng mới, hiện nay mới chỉ được ứng dụng trong lĩnh vực hệ thống dòng chảy chất lỏng [41], [42].

## 4.5 Kết luận chương

Chương 4 đã trình bày về các kiến trúc mạng để phát hiện âm thanh bất thường, đi từ đơn giản đến phức tạp. Bắt đầu từ AE, một mạng neural đơn giản được xây dựng dựa trên các lớp kết nối đầy đủ (fully connected), sau khi biểu diễn lớp ẩn bằng phân phối, ta có được mạng VAE. Nếu thay đổi các lớp của AE từ một chiều thành ba chiều, ta thu được mạng CAE. Cuối cùng, nếu đưa mạng RNN vào giữa hai phần Encoder và Decoder của CAE, ta xây dựng được mạng CRAE. Mạng càng phức tạp thì tốc độ huấn luyện càng chậm, nhưng đi kèm là độ hiệu quả tăng theo, điều này sẽ được chứng minh bằng các thí nghiệm ở chương tiếp theo.

# CHƯƠNG 5. THÍ NGHIỆM VÀ KẾT QUẢ

Chương 5 giới thiệu về tập dữ liệu MIMII được sử dụng trong đề tài phát hiện âm thanh bất thường, sau đó đưa ra các tham số đánh giá độ hiệu quả của mỗi mạng và trình bày về thiết lập thí nghiệm cho các mạng neural. Cuối cùng, các kết quả thí nghiệm được thể hiện và từ đó rút ra các nhận xét.

## 5.1 Tập dữ liệu

Để thực hiện thí nghiệm so sánh hiệu quả của các mô hình mạng đã xây dựng trong việc phát hiện âm thanh bất thường, tập dữ liệu MIMII [43] được sử dụng. Tập dữ liệu này bao gồm các segments âm thanh bình thường và bất thường cho bốn loại máy: van (valve), máy bơm (pump), quạt (fan) và thanh ray trượt (slide rail); mỗi loại máy gồm các ID đánh số từ 00 đến 06. Âm thanh trong tập dữ liệu chỉ có một kênh, tốc độ lấy mẫu là 16 kHz, mỗi segment âm thanh dài 10 s. Bảng 5.1 thể hiện chi tiết số lượng segments âm thanh bình thường và bất thường cho từng ID của từng loại máy.

**Bảng 5.1 Chi tiết tập dữ liệu MIMII [43]**

| Loại máy | ID | Số segments bình thường | Số segments bất thường | Loại máy | ID | Số segments bình thường | Số segments bất thường |
|---|---|---|---|---|---|---|---|
| Van | 00 | 991 | 119 | Quạt | 00 | 1011 | 407 |
| | 01 | 869 | 120 | | 01 | 1034 | 407 |
| | 02 | 708 | 120 | | 02 | 1016 | 359 |
| | 03 | 963 | 120 | | 03 | 1012 | 358 |
| | 04 | 1000 | 120 | | 04 | 1033 | 348 |
| | 05 | 999 | 400 | | 05 | 1109 | 349 |
| | 06 | 992 | 120 | | 06 | 1015 | 361 |
| Máy bơm | 00 | 1006 | 143 | Thanh ray trượt | 00 | 1068 | 356 |
| | 01 | 1003 | 116 | | 01 | 1068 | 178 |
| | 02 | 1005 | 111 | | 02 | 1068 | 267 |
| | 03 | 706 | 113 | | 03 | 1068 | 178 |
| | 04 | 702 | 100 | | 04 | 534 | 178 |
| | 05 | 1008 | 248 | | 05 | 534 | 178 |
| | 06 | 1036 | 102 | | 06 | 534 | 89 |
| Tổng | **Số segments bình thường** | | | **Số segments bất thường** | | | |
| | 26092 | | | 6065 | | | |

Đối với từng loại máy, âm thanh khi hoạt động bình thường và khi được coi là bất thường là khác nhau. Bảng 5.2 mô tả âm thanh từng loại máy khi hoạt động bình thường và một số ví dụ âm thanh máy được coi là bất thường.

**Bảng 5.2 Hoạt động bình thường và bất thường của từng loại máy [43]**

| Loại máy | Hoạt động bình thường | Ví dụ hoạt động bất thường |
|---|---|---|
| Van | Mở/đóng van với timing khác nhau | Nhiễm bẩn |
| Máy bơm | Hút/xả nước vào bể | Rò rỉ, nhiễm bẩn, tắc nghẽn |
| Quạt | Quay bình thường | Mất cân bằng, thay đổi điện thế, tắc nghẽn |
| Thanh ray trượt | Trượt thanh ray lặp đi lặp lại với tốc độ khác nhau | Hư hỏng thanh ray, vành đai lỏng lẻo, không có dầu mỡ bôi trơn |

## 5.2 Các tham số đánh giá

### *5.2.1 Các tham số thống kê trung gian*

Để so sánh kết quả xác định một frame âm thanh là bất thường hay không với nhãn thực tế của nó, các tham số thống kê theo đoạn (segment-based) được sử dụng. Dựa trên kết quả xác định bất thường, bốn tham số sau đây được định nghĩa trong Bảng 5.3:

**Bảng 5.3 Các tham số thống kê trung gian**

| | Nhãn | Dự đoán |
|---|---|---|
| **True positive (TP)** | Bất thường | Bất thường |
| **False positive (FP)** | Bình thường | Bất thường |
| **True negative (TN)** | Bình thường | Bình thường |
| **False negative (FN)** | Bất thường | Bình thường |

### *5.2.2 Precision, Recall và F-Score*

Precision (P) và Recall (R) được đưa ra ở [44] để đo độ hiệu quả của việc khôi phục thông tin, nhưng cũng có thể được sử dụng trong các bài toán phân loại. Chúng được định nghĩa như sau:

$$P = \frac{TP}{TP + FP} \tag{5.1}$$

$$R = \frac{TP}{TP + FN} \tag{5.2}$$

F-score (còn được gọi là F1-score hay F1) được tính toán dựa trên P và R [45]:

$$F1 = \frac{2PR}{P + R} \tag{5.3}$$

Thay (5.1) và (5.2) vào (5.3), công thức tính F1 có thể được viết lại:

$$F1 = \frac{2TP}{2TP + FP + FN} \tag{5.4}$$

### *5.2.3 ROC AUC*

Receiver Operating Characteristic (ROC) là một đồ thị cho thấy độ hiệu quả của một mô hình phân loại ở tất cả các ngưỡng phân loại. Đồ thị này biểu diễn hai thông số là True Positive Rate (TPR) và False Positive Rate (FPR).

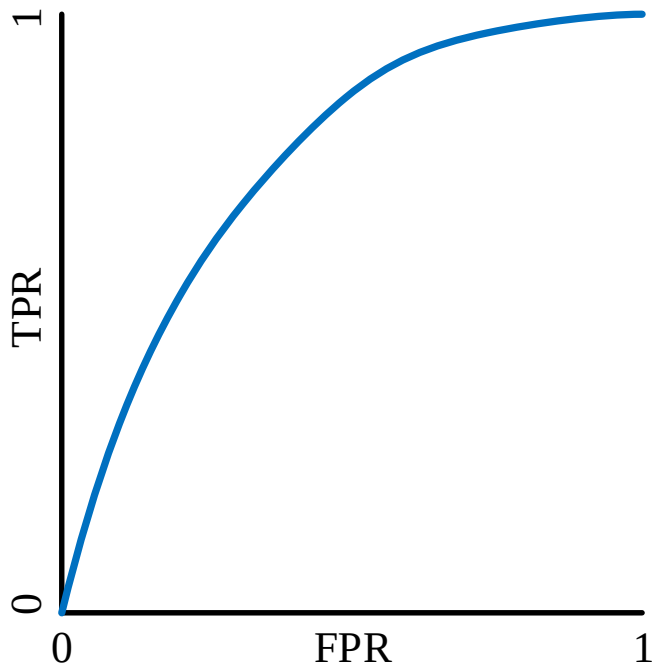
TPR là một cách gọi khác của Recall, do đó công thức tính của nó là:

$$TPR = \frac{TP}{TP + FN} \tag{5.5}$$

FPR được định nghĩa như sau:

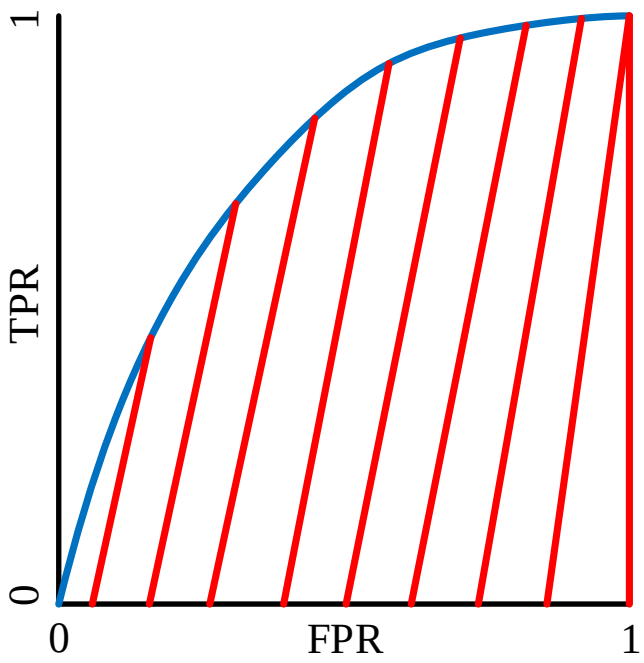
$$FPR = \frac{FP}{FP + TN} \tag{5.6}$$

Đường cong ROC sẽ vẽ mối quan hệ của TPR với FPR ở các ngưỡng phân loại. Dạng của đường cong ROC được thể hiện trên Hình 5.1.




**Hình 5.1 Đường cong ROC**

Để tính toán từng điểm trên đường cong ROC, ta có thể đánh giá mô hình rất nhiều lần với các ngưỡng phân loại khác nhau, nhưng việc này là không hiệu quả. Thay vào đó, một thông số khác có thể cung cấp thông tin này, được gọi là Area Under the ROC Curve (AUC). AUC chính là diện tích của miền nằm dưới đường cong ROC như được thể hiện trên Hình 5.2.




**Hình 5.2 AUC**

Trên Hình 5.2, AUC là diện tích miền gạch chéo màu đỏ. AUC cung cấp một thước đo tổng hợp về hiệu quả mô hình trên tất cả các ngưỡng phân loại có thể. AUC có thể được hiểu như là xác suất mô hình xếp một điểm positive ngẫu nhiên cao hơn một điểm negative ngẫu nhiên. AUC có giá trị từ 0 đến 1, một mô hình phân loại tốt sẽ có giá trị AUC gần với 1.

## 5.3 Thiết lập thí nghiệm

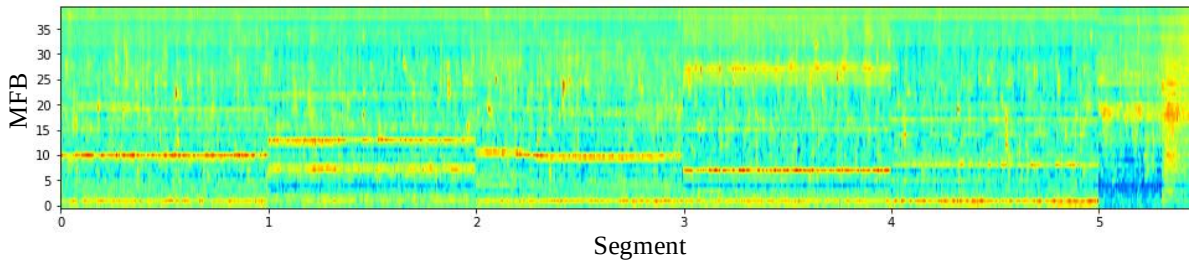
### *5.3.1 Thiết lập tập dữ liệu*

Tập dữ liệu MIMII có bốn loại tiếng máy, mỗi loại lại gồm có 07 ID khác nhau. Đồ án này chỉ thí nghiệm trên tiếng máy bơm ID 00 và tiếng thanh ray trượt ID 00. Việc chia tập huấn luyện, tập xác thực và tập kiểm tra cho hai loại tiếng máy được thực hiện theo DCASE2020 Challenge Task 2 [46] (Bảng 5.4).

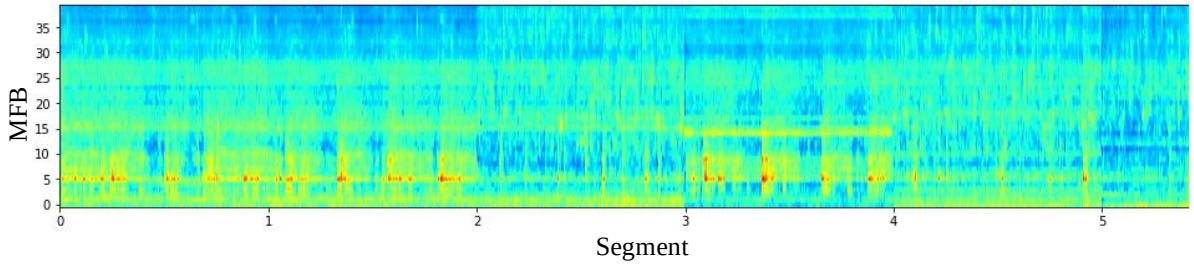
**Bảng 5.4 Thiết lập tập dữ liệu**

| | **Máy bơm** | **Thanh ray trượt** |
|---|---|---|
| **Tập huấn luyện** | 816 segments bình thường | 871 segments bình thường |
| **Tập xác thực** | 90 segments bình thường | 97 segments bình thường |
| **Tập kiểm tra** | 143 segments bất thường,<br>100 segments bình thường | 356 segments bất thường,<br>100 segments bình thường |

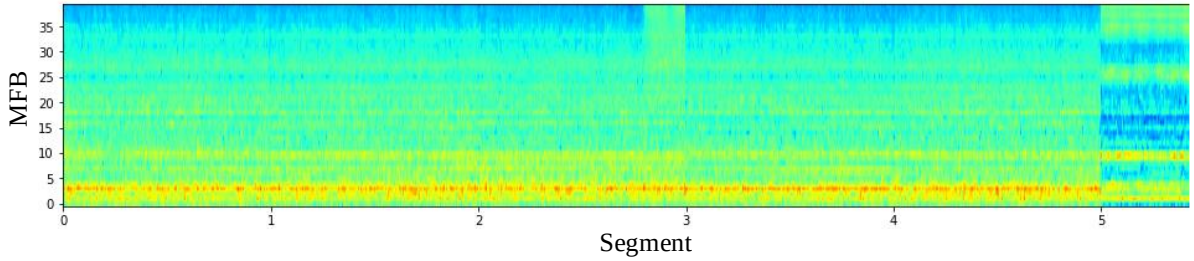
Hình 5.3, Hình 5.4, Hình 5.5 và Hình 5.6 lần lượt minh họa ví dụ MFBs cho âm thanh bình thường của tiếng máy bơm và tiếng thanh ray trượt.



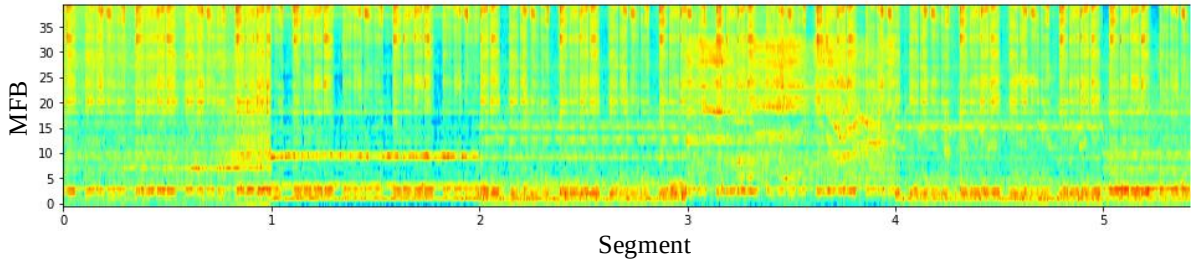

**Hình 5.3 Ví dụ MFBs cho âm thanh bình thường của máy bơm**




**Hình 5.4 Ví dụ MFBs cho âm thanh bất thường của máy bơm**




**Hình 5.5 Ví dụ MFBs cho âm thanh bình thường của thanh ray trượt**




**Hình 5.6 Ví dụ MFBs cho âm thanh bất thường của thanh ray trượt**

Hình 5.3, Hình 5.4, Hình 5.5 và Hình 5.6 cho thấy MFBs cho âm thanh bình thường và bất thường là khác nhau nên có thể phân biệt được. MFBs cho âm thanh bình thường của tiếng máy trong tập dữ liệu này không có độ biến động cao theo thời gian nên việc tái tạo âm thanh bình thường và phát hiện âm thanh bất thường là không quá phức tạp.

### *5.3.2 Thiết lập các mô hình mạng*

Để so sánh hiệu quả của các mô hình phát hiện âm thanh bất thường bằng các thí nghiệm trên tập dữ liệu, các mô hình được triển khai bằng ngôn ngữ lập trình Python sử dụng thư viện Keras. Keras là một thư viện cung cấp bởi Google, chạy trên nền lõi TensorFlow, chuyên phục vụ cho việc xây dựng các mô hình Deep Learning. Keras được sử dụng rộng rãi trong doanh nghiệp và cộng đồng nghiên cứu nhờ cung cấp trải nghiệm người dùng tốt và giúp dễ dàng chuyển đổi thiết kế thành sản phẩm. Ngoài ra, Keras còn hỗ trợ huấn luyện trên nhiều GPU phân tán, hỗ trợ đa backend engines và không giới hạn hệ sinh thái.

Tiếp theo, các mô hình mạng đã xây dựng sẽ được thiếp lập, bao gồm AE, VAE, CAE và CRAE.

Mạng AE chỉ gồm các lớp fully conntected (FC) với số neurons phần Encoder giảm dần theo lũy thừa cơ số 2. Từ 40 hệ số MFBs đầu vào, đi qua lần lượt các lớp FC(32), FC(16), FC(8), FC(4) và FC(2), số đặc trưng chỉ còn hai chiều. Phần Decoder có các lớp FC đối xứng với phần Encoder, gồm các lớp FC(4), FC(8), FC(16), FC(32) và FC(40) để khôi phục 40 hệ số MFBs lúc đầu [47].

Mạng VAE được thiết lập giống hệt AE ở phần Encoder và Decoder, tuy nhiên lớp ẩn FC(2) được thay thế bằng hai phân phối chuẩn để biểu diễn không gian ẩn có số chiều bằng 2 [47].

Mạng CAE nhận đầu vào là ảnh MFBs kích thước 157x40x1, là 157 frames (tương ứng với 10 s của một segment) ghép với nhau, chiều thứ ba bằng 1 do âm thanh trong tập dữ liệu chỉ có một kênh. Phần Encoder và Decoder của CAE là ba lớp Convolution (Conv2D) với số bộ lọc lần lượt là 128, 64, 32, kích thước bộ lọc là 3x3. Phần Decoder là ba lớp Convolution chuyển vị (Conv2DT) với số bộ lọc lần lượt là 64, 128, 1. Lớp cuối cùng sau phần Decoder chỉ có một bộ lọc và được zero-padding do mục đích tái tạo ảnh đầu vào chỉ có một kênh [48].

Thiết lập phần Encoder và Decoder của CRAE là giống với CAE. Phần lớp ẩn của CRAE được bổ sung một mạng RNN hai chiều với các cell là LSTM. Để lựa chọn thiết lập tốt nhất cho mạng RNN, các thí nghiệm với thiết lập mạng RNN khác nhau được tiến hành. Bảng 5.5 thể hiện kết quả F1 và AUC trung bình 10 lần huấn luyện độc lập cho các thiết lập khác nhau cho mạng RNN.

**Bảng 5.5 Kết quả F1 và AUC cho các thiết lập mạng RNN trong CRAE**

| | Máy bơm | | Thanh ray trượt | |
|---|---|---|---|---|
| | **F1** | **AUC** | **F1** | **AUC** |
| **LSTM(32)** | 0.8293 | 0.8274 | 0.9685 | 0.9244 |
| **LSTM(64)** | 0.8457 | 0.8308 | 0.9696 | 0.935 |
| **LSTM(128)** | 0.8047 | 0.8136 | 0.9553 | 0.9132 |

Kết quả F1 và AUC từ Bảng 5.5 cho thấy lựa chọn LSTM(64) làm cell cho mạng RNN trong CRAE là thiết lập tốt nhất.

Ngoài ra, các mô hình đã xây dựng cũng được so sánh với mô hình trong Baseline System (BS) của DCASE2020 Challenge Task 2 [46]. BS sử dụng số bộ lọc Filter Banks bằng 64 và ghép 10 frames liên tục thành đầu vào của mạng, do đó lớp đầu vào có 640 neurons. Mô hình BS này là một AE với Encoder gồm bốn lớp FC(128), lớp ẩn là FC(8) và phần Decoder cũng là bốn lớp FC(128) như phần Encoder và có thêm lớp FC(640). Bảng 5.6 dưới đây tổng hợp lại thiết lập cho các mô hình mạng.

**Bảng 5.6 Tổng hợp thiết lập các mô hình mạng**

| **Mạng** | **Encoder** | **Lớp ẩn** | **Decoder** |
|---|---|---|---|
| **AE** | Input size=40,<br>FC(32), FC(16), FC(8), FC(4) | FC(2) | FC(4), FC(8), FC(16), FC(32), FC(40) |
| **VAE** | Input size=40,<br>FC(32), FC(16), FC(8), FC(4) | Latent dim=2 | FC(4), FC(8), FC(16), FC(32), FC(40) |
| **CAE** | Input size=157x40x1<br>Conv2D(128, K=3x3),<br>Conv2D(64, K=3x3), | Conv2D(32, K=3x3) | Conv2DT(64, K=3x3),<br>Conv2DT(128, K=3x3),<br>Conv2DT(1, K=3x3, padding) |
| **CRAE** | Input size=157x40x1,<br>Conv2D(128, K=3x3),<br>Conv2D(64, K=3x3),<br>Conv2D(32, K=3x3) | Bidirectional(<br>LSTM(64)) | Conv2DT(64, K=3x3),<br>Conv2DT(128, K=3x3),<br>Conv2DT(1, K=3x3, padding) |
| **BS** | Input size=640,<br>FC(128), FC(128), FC(128),<br>FC(128) | FC(8) | FC(128), FC(128), FC(128),<br>FC(128), FC(640) |

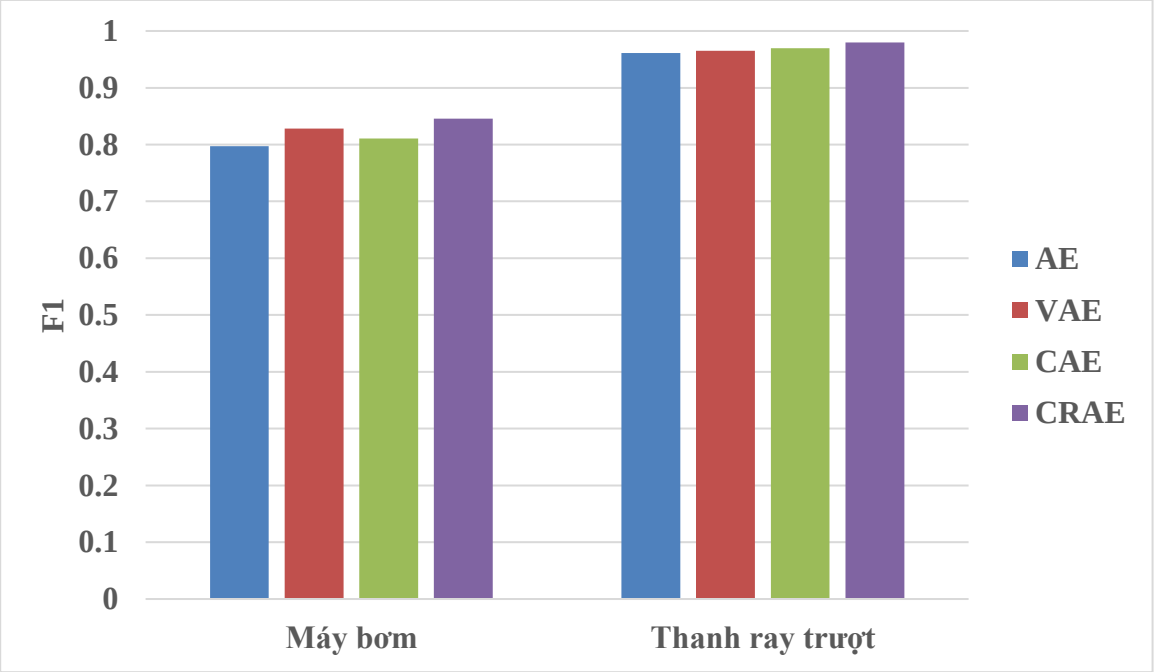
Hàm kích hoạt được áp dụng ở tất cả các lớp của các mô hình mạng là hàm ReLU ngoại trừ ở lớp đầu ra là hàm Sigmoid do giá trị các đặc trưng MFBs đều nằm trong khoảng (0, 1). Các mạng neural được huấn luyện bằng thuật toán tối ưu Adam [49] và được chia thành các mini-batch để huấn luyện trong mỗi epoch, kích thước batch size được đặt bằng 128. Kĩ thuật Early Stopping được sử dụng để dừng quá trình huấn luyện nếu lỗi xác thực không giảm trong 10 epochs liên tiếp, nền tảng triển khai là Google Colab có GPU hỗ trợ.

## 5.4 Kết quả thí nghiệm

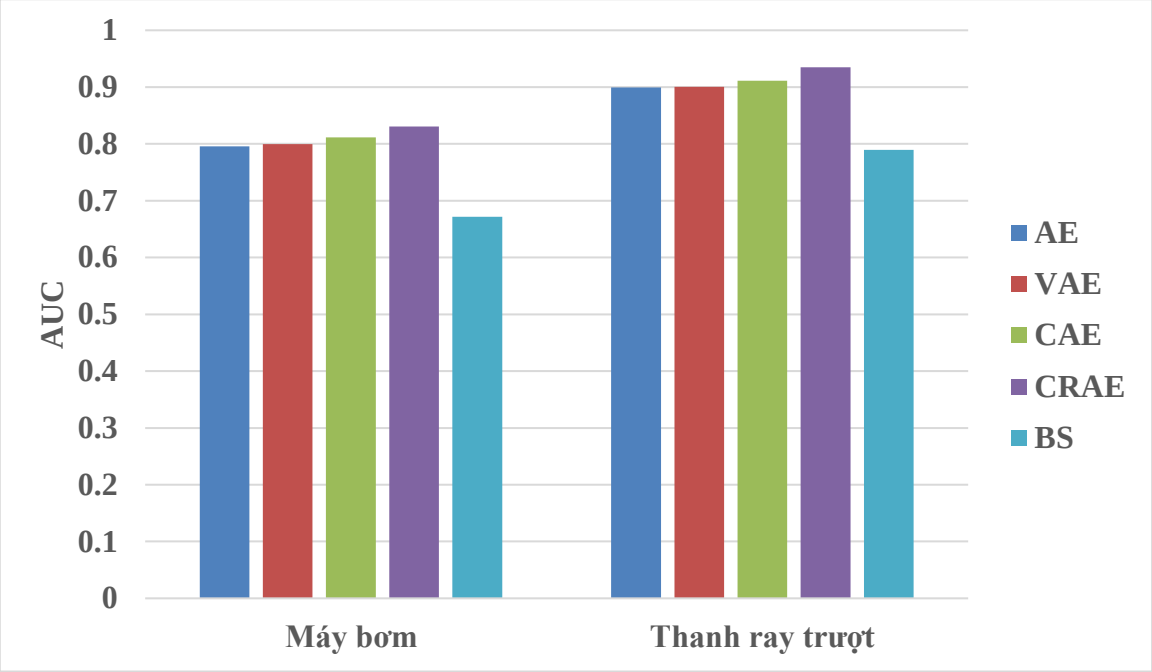
Sau khi hoàn thành thiết lập, các thí nghiệm phát hiện âm thanh bất thường trên tập dữ liệu MIMII được thực hiện. Bảng 5.7 thể hiện chi tiết kết quả F1 và AUC trung bình của 10 lần huấn luyện độc lập cho các mô hình. Để trực quan hóa kết quả thí nghiệm nhằm dễ dàng so sánh độ hiệu quả giữa các mô hình, Hình 5.7 và Hình 5.8 lần lượt thể hiện kết quả F1 và AUC của các mô hình dưới dạng biểu đồ cột. Ngoài ra, Hình 5.9 và Hình 5.10 lần lượt minh họa đường cong ROC của các mô hình cho hai dữ liệu tiếng máy bơm và tiếng thanh ray trượt. DCASE2020 Challenge Task 2 chỉ đưa ra kết quả AUC của BS cho tập dữ liệu này.

**Bảng 5.7 Kết quả F1 và AUC cho các mô hình mạng**

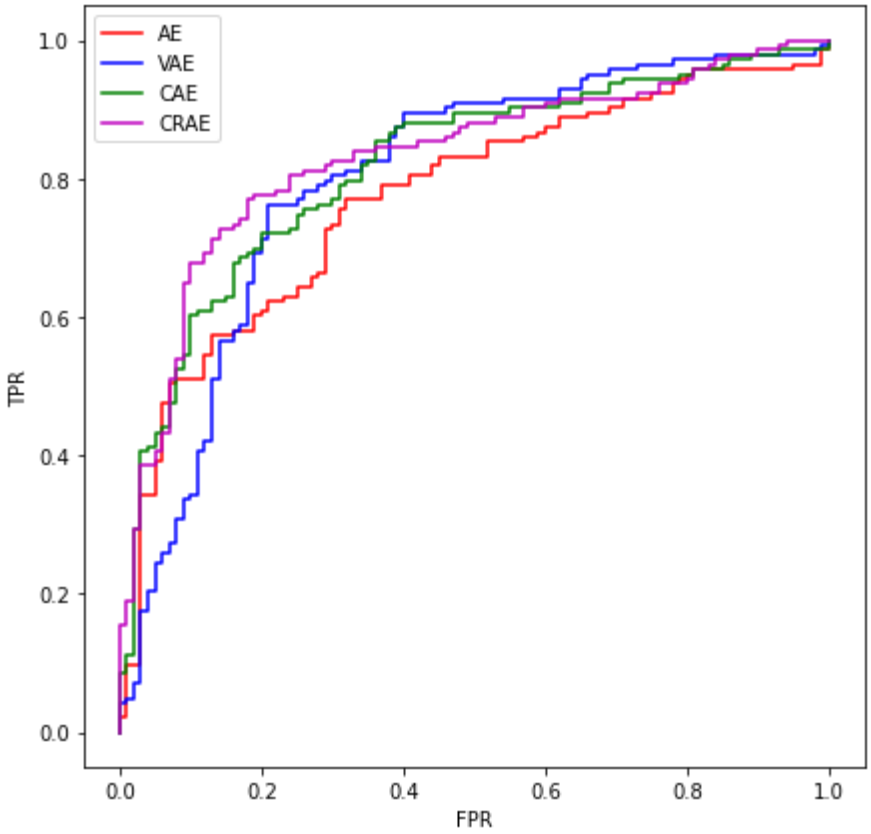
| | Máy bơm | | Thanh ray trượt | |
|---|---|---|---|---|
| | **F1** | **AUC** | **F1** | **AUC** |
| **AE** | 0.7971 | 0.796 | 0.9544 | 0.8994 |
| **VAE** | 0.8231 | 0.7996 | 0.9654 | 0.9003 |
| **CAE** | 0.8106 | 0.8115 | 0.9697 | 0.9113 |
| **CRAE** | 0.8457 | 0.8308 | 0.9696 | 0.935 |
| **BS** | | 0.6715 | | 0.7897 |



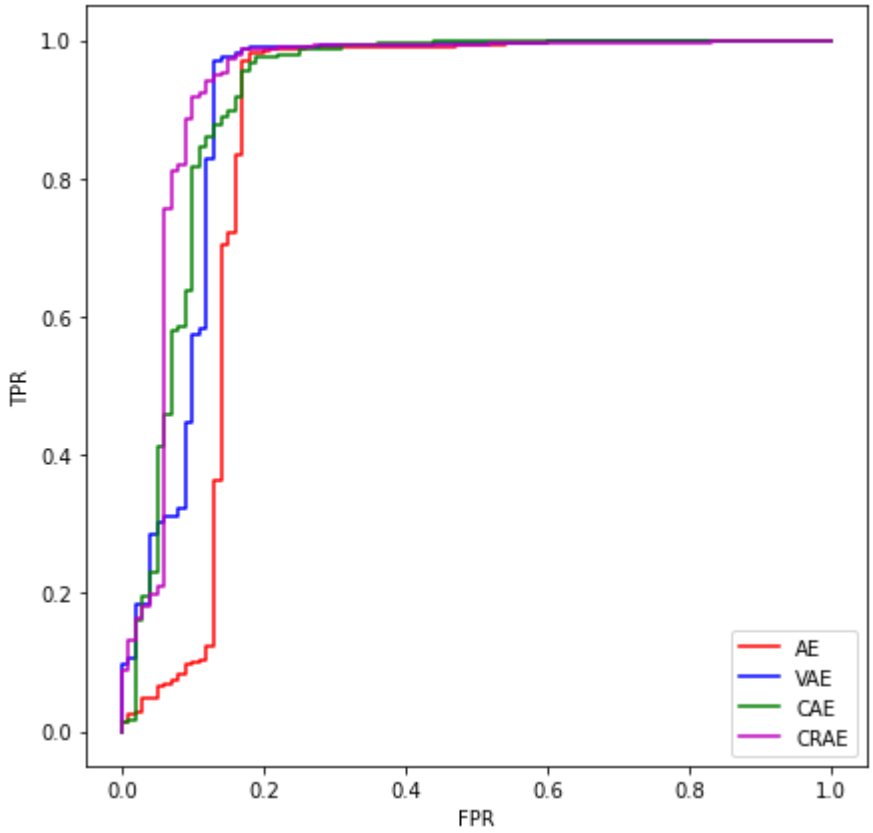

**Hình 5.7 So sánh F1 của các mô hình**




**Hình 5.8 So sánh AUC của các mô hình**




**Hình 5.9 Đường cong ROC của các mô hình cho tiếng máy bơm**




**Hình 5.10 Đường cong ROC của các mô hình cho tiếng thanh ray trượt**

Nhìn chung, kết quả của BS kém hơn khá nhiều so với các mô hình đã xây dựng trong đồ án. BS chỉ là một khuôn mẫu cho các ứng viên tham gia cuộc thi nên không được tối ưu. Việc các lớp FC(128) liên tục được nối với nhau trong khi lớp ẩn đột ngột giám xuống thành FC(8) khiến cho mạng không chọn được những đặc trưng quan trọng nhất của dữ liệu bình thường, từ đó dẫn đến phát hiện bất thường là không tốt. Phần này chỉ đi sâu vào so sánh các mô hình đã xây dựng với nhau.

Đối với kết quả F1 của tiếng máy bơm, CRAE cho F1 tốt nhất và AE cho F1 kém nhất. Điều này là hợp lí vì AE là mô hình đơn giản nhất và CRAE là mô hình phức tạp nhất. Tuy nhiên, F1 của VAE và CAE lại khá ngang nhau, thậm chí CAE còn có F1 nhỏ hơn mặc dù mô hình phức tạp hơn. Điều này được giải thích là do CAE có lợi thế xử lí dữ liệu ảnh ba chiều, nhưng lại không có tính phân phối dữ liệu lớp ẩn như VAE nên thiếu đi tính linh hoạt khi tái tạo dữ liệu.

Khác với F1 của tiếng máy bơm có sự khác nhau giữa các mô hình, F1 của tiếng thanh ray trượt gần như không thay đổi từ mô hình đơn giản AE đến mô hình phức tạp CRAE. Tiếng thanh ray trượt có âm thanh hoạt động bình thường và bất thường khác nhau rõ hơn so với tiếng máy bơm, nên một mô hình đơn giản nhất như AE cũng có thể

phân biệt âm thanh bất thường tốt, việc tăng độ phức tạp mô hình không mang lại hiệu quả đáng kể.

Đối với kết quả AUC cho tiếng máy bơm, nhìn chung xu hướng của AUC tăng theo độ phức tạp mô hình. CRAE vẫn vượt trội so với ba mô hình còn lại, tuy nhiên có thể thấy AUC của VAE gần như không cải thiện gì so với AE. Tham số AUC có tính đến cả TN thay vì chỉ có TP, FP, FN, nên việc tái tạo chính xác dữ liệu bình thường cũng quan trọng không kém việc phát hiện bất thường khi tính toán AUC. Cả AE và VAE đều có khả năng tái tạo dữ liệu bình thường tốt, nhưng VAE nhỉnh hơn ở khả năng xác định bất thường do tối ưu thêm phần phân phối của dữ liệu bình thường. Vì vậy, F1 của VAE cao hơn AE nhưng AUC của hai mô hình là ngang nhau.

Kết quả AUC cho tiếng thanh ray trượt cũng có tương quan giữa các mô hình giống như F1 cho tiếng loại này. Không có sự khác biệt đáng kể của AUC giữa các mô hình, do âm thanh bất thường của thanh ray trượt có sự khác biệt rõ rệt so với âm thanh bình thường, nên việc phân biệt giữa âm thanh bình thường và bất thường là đơn giản hơn nhiều so với tiếng máy bơm. Mặc dù CRAE vẫn có kết quả AUC tốt nhất, nhưng độ hiệu quả tăng lên là không nhiều.

Như vậy, qua các thí nghiệm thực hiện, CRAE đã cho thấy sự vượt trội của nó so với các mô hình còn lại, mặc dù sự vượt trội này không thực sự đáng kể. Đối với dữ liệu là âm thanh tiếng máy bơm và tiếng thanh ray trượt, âm thanh của chúng hoạt động bình thường khá ổn định và ít có sự biến động, nên việc dùng mô hình phức tạp hơn chỉ có thể cải thiện khả năng phát hiện bất thường lên một chút. Ngoài ra, cả CAE và CRAE còn có một điểm yếu là thiếu đi tính linh hoạt do không xét đến phân phối dữ liệu ở lớp ẩn, điều này cũng hạn chế phần nào khả năng tái tạo của mô hình.

## 5.5 Kết luận chương

Chương 5 đã trình bày về tập dữ liệu, các tham số đánh giá độ hiệu quả của các mô hình, các thiết lập thí nghiệm và cuối cùng đưa ra kết quả thí nghiệm. Kết quả thí nghiệm đã cho thấy mô hình mạng đề xuất CRAE có khả năng phát hiện âm thanh bất thường tốt hơn các mạng thông thường là AE, VAE và CAE. Tuy nhiên, do tính chất của dữ liệu âm thanh bình thường trong tập dữ liệu có độ ổn định cao nên sự hiệu quả tăng lên từ CRAE so với các mô hình khác là không quá nhiều.

# KẾT LUẬN

## Kết luận chung

Đồ án này đã trình bày cơ sở lí thuyết về phát hiện âm thanh bất thường bằng mạng neural, đề xuất mô hình mạng CRAE và thực hiện thí nghiệm để chứng minh sự hiệu quả của mô hình. Cụ thể, đồ án đã nêu lên và phân tích kĩ từng bước trích xuất đặc trưng MFBs của dữ liệu âm thanh, xây dựng các loại mô hình mạng Autoencoder là AE, VAE, CAE và CRAE được sử dụng cho các bài toán phát hiện âm thanh bất thường. Các kết quả thí nghiệm trên tập dữ liệu MIMII đã cho thấy sự vượt trội của mạng CRAE trong khả năng phát hiện âm thanh bất thường so với các mạng còn lại.

## Hướng phát triển

Mô hình mạng CRAE được đề xuất vẫn còn hạn chế ở lớp ẩn khi không tính đến phân phối đặc trưng ở lớp này. Để khắc phục hạn chế này, trong tương lai em sẽ thử nghiệm mô hình Bộ mã hóa tự động biến thể tích chập (Convolutional Variational Autoencoder – CVAE), là kết hợp của CAE và VAE khi đưa thêm phân phối vào lớp ẩn. CVAE đã được triển khai để phát hiện bất thường trong video [50] và trong lĩnh vực Internet of Things (IoT) [51] nhưng chưa được triển khai với dữ liệu âm thanh. Một số mạng neural khác như mạng neural nội suy [47] và một biến thể khác của AE sử dụng Neyman-Pearson Lemma [52] cũng sẽ được xem xét để so sánh với các mạng đã xây dựng trong đồ án. Ngoài ra, do âm thanh bình thường của máy mó c trong tập dữ liệu MIMII có độ ổn định cao, nên CRAE chưa thể hiện được khả năng vượt trội đáng kể so với các mô hình đơn giản. Vì vậy, bên cạnh âm thanh máy móc, em cũng sẽ thực hiện thêm thí nghiệm đối với tập dữ liệu âm thanh đường phố có độ biến động cao ở trong [33] để khai thác tốt hơn tiềm năng của CRAE.

# TÀI LIỆU THAM KHẢO


[1] P. Coucke, B. D. Ketelaere and J. D. Baerdemaeker, "Experimental analysis of the dynamic, mechanical behavior of a chicken egg," *Journal of Sound and Vibration,* vol. 266, pp. 711-721, 2003.

[2] Y. Chung, S. Oh, J. Lee, D. Park, H. H. Chang and S. Kim, "Automatic Detection and Recognition of Pig Wasting Diseases Using Sound Data in Audio Surveillance Systems," *Sensors,* p. 12929–12942, 2013.

[3] A. Yamashita, T. Hara and T. Kaneko, "Inspection of Visible and Invisible Features of Objects with Image and Sound Signal Processing," in *Proceedings of the 2006 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS2006)*, Beijing, China, October 2006.

[4] Y. Koizumi, S. Saito, H. Uematsu and N. Harada, "Optimizing Acoustic Feature Extractor for Anomalous Sound Detection Based on Neyman-Pearson Lemma," in *Proceedings of European Signal Processing Conference (EUSIPCO)*, Greek island of Kos, September 2017.

[5] C. Clavel, T. Ehrette and G. Richard, "Events Detection for an Audio-Based Surveillance System," in *Proceedings of 2005 IEEE International Conference on Multimedia and Expo*, Amsterdam, Netherlands, July 2005.

[6] G. Valenzise, L. Gerosa, M. Tagliasacchi, F. Antonacci and A. Sarti, "Scream and Gunshot Detection and Localization for Audio-Surveillance," in *Proceedings of Advanced Video and Signal based Surveillance*, London, UK, September 2007.

[7] S. Ntalampiras, I. Potamitis and N. Fakotakis, "Probabilistic Novelty Detection for Acoustic Surveillance Under Real-World Conditions," *IEEE Transactions on Multimedia ,* vol. 13, no. 4, pp. 713-719, August 2011.

[8] P. Foggia, N. Petkov, A. Saggese, N. Strisciuglio and M. Vento, "Audio Surveillance of Roads: A System for Detecting Anomalous Sounds," *IEEE Transactions on Intelligent Transportation Systems,* vol. 17, no. 1, pp. 279-288, January 2016.

[9] M. Sakurada and T. Yairi, "Anomaly Detection Using Autoencoders with Nonlinear Dimensionality Reduction," in *Proceedings of the MLSDA 2014 2nd Workshop on Machine Learning for Sensory Data Analysis*, New York, USA, December 2014.

[10] E. Marchi, F. Vesperini, F. Eyben, S. Squartini and B. Schuller, "A novel approach for automatic acoustic novelty detection using a denoising autoencoder with bidirectional LSTM neural networks," in *Acoustics, Speech and Signal Processing (ICASSP), 2015 IEEE International Conference on*, Brisbane, QLD, Australia , April 2015.

[11] J. McCarthy, M. L. Minsky, N. Rochester and C. Shannon, "A Proposal for the Dartmouth Summer Research Project on Artificial Intelligence," Nikos Drakos, Computer Based Learning Unit, University of Leeds, England, 1956.

[12] M. Minsky and S. Papert, Perceptrons: An Introduction to Computational Geometry, Cambridge, Massachusetts, USA: MIT Press, 1969.

[13] D. E. Rumelhart, G. E. Hinton and R. J. Williams, "Learning representations by back-propagating errors," in *Neurocomputing: foundations of research*, Cambridge, Massachusetts, USA, MIT Press, 1988, pp. 696-699.

[14] A. L. Samuel, "Some Studies in Machine Learning Using the Game of Checkers," *IBM Journal of Research and Development,* vol. 3, no. 3, pp. 210-229, July 1959.

[15] V. H. Tiệp, Machine Learning cơ bản, Hà Nội, VIệt Nam: NXB Khoa học kỹ thuật, 2018.

[16] M. Copeland, "What's the Difference Between Artificial Intelligence, Machine Learning and Deep Learning?," NVIDIA, 29 July 2016. [Online]. Available: https://blogs.nvidia.com/blog/2016/07/29/whats-difference-artificial-intelligence-machine-learning-deep-learning-ai/.

[17] H. Eghbal-Zadeh, B. Lehner, M. Dorfer and G. Widmer, "CP-JKU Submissions for DCASE-2016: a Hybrid Approach Using Binaural I-Vectors and Deep Convolutional Neural Networks," DCASE2016 Challenge, Tech. Rep., Budapest, Hungary, September 2016.

[18] Y. Han and J. Park, "Convolutional Neural Networks with Binaural Representations and Background Subtraction for Acoustic Scene Classification," DCASE2017 Challenge, Tech. Rep., Munich, Germany, September 2017.

[19] T. Komatsu, T. Toizumi, R. Kondo and Y. Senda, "Acoustic Event Detection Method Using Semi-Supervised Non-Negative Matrix Factorization with Mixtures of Local Dictionaries," in *Proceedings of the Detection and Classification of Acoustic Scenes and Events 2016 Workshop*, Budapest, Hungary, September 2016.

[20] T. Hayashi, S. Watanabe, T. Toda, T. Hori, J. L. Roux and K. Takeda, "Duration-Controlled LSTM for Polyphonic Sound Event Detection," *IEEE/ACM Transactions on Audio, Speech, and Language Processing,* vol. 25, no. 11, pp. 2059 - 2070, August 2017.

[21] S. Adavanne, A. Politis and T. Virtanen, "Multichannel Sound Event Detection Using 3D Convolutional Neural Networks for Learning Inter-channel Features," in *arXiv preprint arXiv:1801.09522, 2018*, January 2018.

[22] M. Basseville and I. V. Nikiforov, Detection of Abrupt Changes: Theory and Application, vol. 104, New Jersey, USA: Prentice Hall Englewood Cliffs, 1993.

[23] S. Liu, M. Yamada, N. Collier and M. Sugiyama, "Change-Point Detection in Time-Series Data by Relative Density-Ratio Estimation," *Neural Networks,* vol. 43, pp. 72-83, 2013.

[24] Y. Ono, Y. Onishi, T. Koshinaka, S. Takata and O. Hoshuyama, "Anomaly detection of motors with feature emphasis using only normal sounds," in *Acoustics, Speech, and Signal Processing (ICASSP), International Conference on*, Vancouver, Canada, May 2013.

[25] L. Tarassenko, P. Hayton, N. Cerneaz and M. Brady, "Novelty detection for the identification of masses in mammograms," in *1995 Fourth International Conference on Artificial Neural Networks*, Cambridge, UK, June 1995.

[26] J. Foote, "Automatic audio segmentation using a measure of audio," in *Multimedia and Expo, 2000. ICME 2000. 2000 IEEE*, New York, USA, August 2000.

[27] M. Markou and S. Singh, "Novelty detection: a review-part 1: statistical approaches," *Signal Processing,* vol. 83, no. 12, pp. 2481-2497, December 2003 .

[28] D. W. Scott, "Outlier Detection and Clustering by Partial Mixture Modeling," in *COMPSTAT 2004 Proceedings in Computational Statistics*, 2004.

[29] J. Ma and S. Perkins, "Time-series novelty detection using one-class support vector machines," in *Proceedings of the International Joint Conference on Neural Networks*, Oregon, USA, July 2003.

[30] A. Krizhevsky, I. Sutskever and G. E. Hinton, "ImageNet Classification with Deep Convolutional Neural Networks," in *Advances in Neural Information Processing Systems 25*, Nevada, USA, December 2012.

[31] S. Davis and P. Mermelstein, "Comparison of Parametric Representations for Monosyllabic Word Recognition in Continuously Spoken Sentences," *IEEE Transactions on Acoustics, Speech, and Signal Processing,* vol. 28, no. 4, pp. 357-366, August 1980.

[32] X. Huang, A. Acero and H. Hon., Spoken Language Processing: A guide to theory, algorithm, and system development, Prentice Hall, 2001.

[33] T. Hayashi, T. Komatsu, R. Kondo, T. Toda and K. Takeda, "Anomalous Sound Event Detection Based on WaveNet," in *2018 26th European Signal Processing Conference*, Rome, Italy, September 2018.

[34] R. S. Z. Geoffrey E. Hinton, "Autoencoders, Minimum Description Length and Helmholtz Free Energy," in *Advances in Neural Information Processing Systems 6*, Denver, Colorado, USA, November 1993.

[35] P. Baldi, "Autoencoders, Unsupervised Learning, and Deep Architectures," in *Proceedings of ICML Workshop on Unsupervised and Transfer Learning*, Bellevue, Washington, USA, July 2011.

[36] D. P. Kingma and M. Welling, "Auto-Encoding Variational Bayes," in *International Conference on Learning Representations*, Banff, Canada, April 2014.

[37] T. Duman, B. Bayram and G. İnce, "Acoustic Anomaly Detection Using Convolutional Autoencoders in Industrial Processes," in *14th International Conference on Soft Computing Models in Industrial and Environmental Applications*, Seville, Spain, May 2019.

[38] V. Dumoulin and F. Visin, "A guide to convolution arithmetic for deep learning," arXiv:1603.07285 [stat.ML], March 2016.

[39] S. Amidi and A. Amidi, "Recurrent Neural Networks cheatsheet," 2019. [Online]. Available: https://stanford.edu/~shervine/teaching/cs-230/cheatsheet-recurrent-neural-networks.

[40] S. Rathor, "Simple RNN vs GRU vs LSTM :- Difference lies in More Flexible control," 3 June 2018. [Online]. Available: https://medium.com/@saurabh.rathor092/simple-rnn-vs-gru-vs-lstm-difference-lies-in-more-flexible-control-5f33e07b1e57.

[41] F. J. Gonzalez and M. Balajewicz, "Deep convolutional recurrent autoencoders for learning low-dimensional feature dynamics of fluid systems," *arXiv:1808.01346v2 [math.DS],* August 2018.

[42] S. R. Bukka, A. R. Magee and R. K. Jaiman, "Deep Convolutional Recurrent Autoencoders for Flow Field Prediction," *arXiv:2003.12147 [physics.flu-dyn],* March 2020.

[43] H. Purohit, R. Tanabe, K. Ichige, T. Endo, Y. Nikaido, K. Suefusa and Y. Kawaguchi, "MIMII Dataset: Sound Dataset for Malfunctioning Industrial Machine Investigation and Inspection," in *Detection and Classification of Acoustic Scenes and Events 2019*, New York, USA, October 2019.

[44] C. Rijsbergen, Information Retrieval, Newton, MA, USA: Butterworth-Heinemann, 1979.

[45] A. Mesaros, T. Heittola and T. Virtanen, "Metrics for polyphonic sound event detection," *Applied Sciences,* vol. 6, no. 6, 2016.

[46] Y. Koizumi, Y. Kawaguchi, K. Imoto, T. Nakamura, Y. Nikaido, R. Tanabe, H. Purohit, K. Suefusa, T. Endo, M. Yasuda and N. Harada, "Description and Discussion on DCASE2020 Challenge Task2: Unsupervised Anomalous Sound Detection for Machine Condition Monitoring," in *arXiv:2006.05822 [eess.AS]*, 2020.

[47] K. Suefusa, T. Nishida, H. Purohit, R. Tanabe, T. Endo and Y. Kawaguchi, "Anomalous Sound Detection Based on Interpolation Deep Neural Network," in

*2020 IEEE International Conference on Acoustics, Speech and Signal Processing*, Barcelona, Spain, May 2020 .

[48] R. Müller, F. Ritz, S. Illium and C. Linnhoff-Popien, "Acoustic Anomaly Detection for Machine Sounds based on Image Transfer Learning," in *arXiv:2006.03429 [eess.AS]*, June 2020.

[49] D. P. Kingma and J. Ba, "Adam: A Method for Stochastic Optimization," in *3rd International Conference on Learning Representations*, San Diego, CA, USA, May 215.

[50] Y. Fan, G. Wen, D. Li, S. Qiu and M. D. Levine, "Video Anomaly Detection and Localization via Gaussian Mixture Fully Convolutional Variational Autoencoder," in *arXiv:1805.11223 [cs.CV]*, 2018.

[51] D. Kim, H. Yang, M. Chung, S. Cho, H. Kim, M. Kim, K. Kim and E. Kim, "Squeezed Convolutional Variational AutoEncoder for Unsupervised Anomaly Detection in Edge Device Industrial Internet of Things," in *2018 International Conference on Information and Computer Technologies*, DeKalb, Illinois, USA, March 2018.

[52] Y. Koizumi, S. Saito, H. Uematsum, Y. Kawachi and N. Harada, "Unsupervised Detection of Anomalous Sound Based on Deep Learning and the Neyman–Pearson Lemma," *IEEE/ACM Transactions on Audio, Speech, and Language Processing,* vol. 27, no. 1, pp. 212-224, January 2019.