**TRƯỜNG ĐẠI HỌC BÁCH KHOA HÀ NỘI**

**ĐỒ ÁN TỐT NGHIỆP**

**Mạng nơ-ron nhận dạng các từ rời rạc số đếm tiếng Việt từ không đến chín**

**NGUYỄN XUÂN TRƯỜNG**

truongnx.144794@sis.hust.edu.vn

**Ngành: Công nghệ thông tin**

|  |  |
| --- | --- |
| **Giảng viên hướng dẫn:** | PGS. TS. Trịnh Văn Loan  Chữ ký của GVHD |
| **Bộ môn:** | Kỹ thuật Máy Tính |
| **Viện:** | Công nghệ Thông tin và Truyền thông |
| **HÀ NỘI, 12/2019** | |

**ĐỀ TÀI TỐT NGHIỆP**

**MẠNG NƠ-RON NHẬN DẠNG CÁC TỪ RỜI RẠC SỐ ĐẾM TIẾNG VIỆT TỪ KHÔNG ĐẾN CHÍN**

Giáo viên hướng dẫn

Ký và ghi rõ họ tên

**Lời cảm ơn**

Trong quá trình làm đồ án tốt nghiệp em đã nhận được rất nhiều sự giúp đỡ để hoàn tất đồ án.

Trước tiên em xin gửi lời cảm ơn chân thành thầy Trịnh Văn Loan đã tận tình hướng dẫn, chỉ ra những sai xót để em có thể hoàn thành đồ án tốt nghiệp.

Xin gửi lời cảm ơn đến quý thầy cô Trường Đại học Bách khoa Hà Nội, đặc biệt là các thầy cô trong Viện Công nghệ Thông tin và Truyền thông đã truyền đạt kiến thức cho em suốt thời gian vừa qua.

Sau cùng xin gửi lời cảm ơn đến gia đình, bạn bè đã luôn ủng hộ, tạo động lực trong suốt quá trình học tập và làm đồ án tốt nghiệp.

Một lần nữa, xin chân thành cảm ơn!

**Tóm tắt nội dung đồ án**

**Chương 1:** Giới thiệu chung về bài toán nhận dạng tiếng nói, tìm hiểu về các khó khăn, các mô hình nhận dạng hiện nay, các bài toán nhận dạng tiếng nói.

Sinh viên thực hiện

Ký và ghi rõ họ tên

**Chương 2:** Trình bày nội dung lý thuyết về tiếng nói, xử lý tín hiệu tiếng nói và các vấn đề trong xử lý tiếng nói. Ngoài ra, nội dung còn đề cập đến đặc trưng và phương pháp trích trọn đặc trưng để đưa vào huấn luyện.

**Chương 3:** Tìm hiểu tổng quan về mạng nơ-ron, khái niệm cơ bản, cấu trúc, cách học. Giới thiệu về mạng nơ-ron tích chập.

**Chương 4:** Tiến hành phân thích bài toán, dữ liệu thu âm cùng tiêu chí đánh giá.

Trích xuất đặc trưng của mẫu. Thiết lập mạng nơ-ron để huấn luyện.

**Chương 5:** Xây dựng mô hình huấn luyện và tiến hành đánh giá trên tập Test.

**MỤC LỤC**

[CHƯƠNG 1. TỔNG QUAN VỀ ĐỀ TÀI 1](#_Toc29732902)

[1.1 Giới thiệu chung về bài toán liên quan tiếng nói 1](#_Toc29732903)

[1.2 Những khó khăn trong bài toán nhận dạng tiếng nói 1](#_Toc29732904)

[1.3 Các mô hình nhận dạng tiếng nói 1](#_Toc29732905)

[1.4 Phân loại bài toán nhận dạng tiếng nói theo độ phức tạp tăng dần 1](#_Toc29732906)

[1.5 Nhiệm vụ của đồ án 1](#_Toc29732907)

[CHƯƠNG 2. CƠ SỞ LÝ THUYẾT VỀ XỬ LÝ TIẾNG NÓI 3](#_Toc29732908)

[2.1 Tổng quan về tiếng nói 3](#_Toc29732909)

[2.1.1 Khái niệm về tiếng nói 3](#_Toc29732910)

[2.1.2 Cơ chế tạo tiếng nói 3](#_Toc29732911)

[2.1.3 Các thành phần cơ bản của ngữ điệu tiếng nói 4](#_Toc29732912)

[2.1.4 Biểu diễn tín hiệu tiếng nói 4](#_Toc29732913)

[2.1.5 Một số đặc điểm của ngữ âm tiếng Việt 6](#_Toc29732914)

[2.2 Tổng quan về xử lý tiếng nói 7](#_Toc29732915)

[2.2.1 Vai trò của xử lý tiếng nói 7](#_Toc29732916)

[2.2.2 Phương pháp xử lý tín hiệu tiếng nói thông qua phân tích phổ 8](#_Toc29732917)

[2.3 Một số đặc trưng thường được sử dụng trong nhận dạng tiếng nói 9](#_Toc29732918)

[CHƯƠNG 3. MẠNG NƠ-RON 11](#_Toc29732919)

[3.1 Tổng quan về mạng nơ-ron nhân tạo 11](#_Toc29732920)

[3.1.1 Perceptron 11](#_Toc29732921)

[3.1.2 Cấu tạo mạng nơ-ron và hoạt động cơ bản 13](#_Toc29732922)

[3.1.3 Dữ liệu cho mạng nơ-ron 14](#_Toc29732923)

[3.2 Mạng nơ-ron tích chập (Convolution Neural Network, CNNs) 15](#_Toc29732924)

[3.2.1 Tổng quan về mạng CNN 15](#_Toc29732925)

[3.2.2 Cách hoạt động của mạng CNN 15](#_Toc29732926)

[CHƯƠNG 4. GIẢI QUYẾT BÀI TOÁN 19](#_Toc29732927)

[4.1 Phân tích bài bài toán 19](#_Toc29732928)

[4.1.1 Bài toán và cách tiếp cận 19](#_Toc29732929)

[4.1.2 Tiêu chí đánh giá 19](#_Toc29732930)

[4.2 Dữ liệu 19](#_Toc29732931)

[4.2.1 Tập train 19](#_Toc29732932)

[4.2.2 Tập Test 20](#_Toc29732933)

[4.3 Chia dữ liệu từ dữ liệu ban đầu 21](#_Toc29732934)

[4.4 Tiến hành xử lý tín hiệu tiếng nói 22](#_Toc29732935)

[4.5 Xây dựng lớp Dataset để đưa vào mô hình huấn luyện 23](#_Toc29732936)

[4.6 Xây dựng mô hình huấn luyện 24](#_Toc29732937)

[4.6.1 Tính toán các tham số của mô hình huấn luyện: 26](#_Toc29732938)

[CHƯƠNG 5. TIẾN HÀNH HUẤN LUYỆN VÀ CÁC KẾT QUẢ ĐẠT ĐƯỢC 28](#_Toc29732939)

[5.1 Cách huấn luyện 28](#_Toc29732940)

[5.2 Biểu đồ accuracy của một số mô hình trên tập Validation 28](#_Toc29732941)

[5.3 Biểu đồ hàm loss của một số mô hình trên tập Train 29](#_Toc29732942)

[5.4 Biểu đồ hàm loss trên tập Validation một vài mô hình 30](#_Toc29732943)

[5.5 Ma trận lỗi của một số mô hình trên tập Test 31](#_Toc29732944)

[5.6 Bảng số liệu 32](#_Toc29732945)

[CHƯƠNG 6. KẾT LUẬN 34](#_Toc29732946)

[6.1 Kết luận 34](#_Toc29732947)

[6.2 Hướng phát triển của đồ án trong tương lai 34](#_Toc29732948)

[TÀI LIỆU THAM KHẢO 35](#_Toc29732949)

**DANH MỤC HÌNH VẼ**

[Hình 2.1 Bộ máy phát âm của com người 3](file:////Users/mac/Desktop/final/baocao_truongnx.docx#_Toc29732982)

[Hình 2.2 Biểu diễn tín hiệu dạng sóng của âm số sáu 5](#_Toc29732983)

[Hình 2.3 Biểu diễn dạng sóng của âm số hai 5](#_Toc29732984)

[Hình 2.4 Biểu diễn dưới dạng phổ tín hiệu 5](#_Toc29732985)

[Hình 2.5 Biểu diễn Spectrogram của âm số sáu 6](#_Toc29732986)

[Hình 2.6 Biểu diễn Specrogram của âm số hai 6](#_Toc29732987)

[Hình 2.7 Ứng dụng của xử lý tiếng nói 7](#_Toc29732988)

[Hình 2.8 Sơ đồ phân tích phổ 8](#_Toc29732989)

[Hình 2.9 Tác dụng của bộ lọc hiệu chỉnh 9](#_Toc29732990)

[Hình 2.10 Đặc trưng Spectrogram của âm tiết ‘sáu’ 10](#_Toc29732991)

[Hình 2.11 Đặc trưng Log Spectrogram 10](#_Toc29732992)

[Hình 3.1 Cấu tạo nơ-ron sinh học 11](#_Toc29732993)

[Hình 3.2 Perceptron-mô hình đơn giản của mộ mạng nơ-ron nhân tạo 12](#_Toc29732994)

[Hình 3.3 Mô hình MPL đơn giản 13](#_Toc29732995)

[Hình 3.4 Cấu trức mạng CNN đơn giản 15](#_Toc29732996)

[Hình 3.5 Con1d với stride=1 16](#_Toc29732997)

[Hình 3.6 Conv1d với stride=1, padding=2 16](#_Toc29732998)

[Hình 3.7 Một số bộ lọc thông dụng 17](#_Toc29732999)

[Hình 3.8 Cách hoạt động của bộ lọc 17](#_Toc29733000)

[Hình 3.9 MaxPool2d và Average Pooling 18](#_Toc29733001)

[Hình 4.1 Biểu đồ sự phân bố các lớp tập dữ liệu gốc 19](#_Toc29733002)

[Hình 4.2 Biểu đồ sự phân bố các lớp tập Train 20](#_Toc29733003)

[Hình 4.3 Sơ đồ mạng huấn luyện 26](#_Toc29733004)

[Hình 5.1 Biểu đồ Accuracy của một số mô hình trên tập Validation 29](#_Toc29733005)

[Hình 5.2 Biểu đồ hàm loss trên tập Train 30](#_Toc29733006)

[Hình 5.3 Biểu đồ hàm loss một số mô hình trên tập Validation 31](#_Toc29733007)

[Hình 5.4 Ma trận lỗi của một số mô hình huấn luyện 32](#_Toc29733008)

**DANH MỤC BẢNG**

[Bảng 4.1 Tổng các mẫu mỗi lớp trong tập Train và Validation 20](#_Toc29732977)

[Bảng 4.2 Biểu đồ sự phân bố các lớp tập dữ liệu tập Test 21](#_Toc29732978)

[Bảng 4.3 Tổng số các mẫu từng lớp trong tập Test 21](#_Toc29732979)

[Bảng 5.1 Bảng đánh giá mô hình trên tập Validation và Test 33](#_Toc29732980)

# TỔNG QUAN VỀ ĐỀ TÀI

## Giới thiệu chung về bài toán liên quan tiếng nói

Tiếng nói là phương thức giao tiếp cơ bản và chủ yếu của con người trong nhận dạng tiếng nói.

Nhận dạng tiếng nói nghĩa là làm cho máy tính hiểu, nhận biết được từ ngữ thông qua lời nói của con người.

Nhận dạng tiếng nói là quá trình phức tạp, gồm nhiều khâu biến đổi. Tín hiệu người phát ra là tín hiệu tương tự , thông qua quá trình lấy mẫu, lượng tử hoá và mã hoá để thu được các mẫu tín hiệu dạng số. Sau đó được trích trọn đặc trưng, những đặc trưng này sẽ đưa vào hệ thống để nhận dạng. Tuỳ thuộc yêu cầu mà đầu ra sẽ khác nhau.

## Những khó khăn trong bài toán nhận dạng tiếng nói

* Trong quá trình phát âm có người nói nhanh, có người nói chậm.
* Cùng một từ mà có thể có hai cách phát âm khác nhau.
* Trong quá trình thu âm, không thể tránh được các yêu tố làm nhiễu như nhiễu của môi trường, nhiễu của thiết bị thu, tạp âm, các khoảng lặng ảnh hưởng không nhỏ đến quá trình nhận dạng.
* Có những người bị phát âm ngọng, nói lắp, cũng ảnh hưởng đến kết quả nhận dạng.

## Các mô hình nhận dạng tiếng nói

* Mô hình Markov ẩn (Hidden Markov Model : HMM) là mô hình thống kê trong đó hệ thống được mô hình hoá được cho là một quá trình Markov với các tham số không biết trước và nhiệm vụ là xác định các tham số ẩn từ các mô hình quan sát được, dựa trên sự thừa nhận này. Các mô hình được rút ra sau đó có thể sử dụng để thực hiện các phép phân tích kế tiếp.
* Mô hình mạng nơ-ron: hiện nay được phát triển và sử dụng một cách rộng rãi bởi kết quả mà nó đạt được trong nhiều lĩnh vực như Computer Vision, y học, nhận dạng tiếng nói,…

## Phân loại bài toán nhận dạng tiếng nói theo độ phức tạp tăng dần

* Nhận dạng từ riêng lẻ, từ vựng (<100), một người nói
* Từ vựng nhiều hơn (khoảng vài nghìn từ), một người nói
* Như trên nhưng cho hệ thống nhiều người nói
* Nhận dạng các từ đi với nhau, từ vựng ít (hàng chục từ)
* Nhận dạng câu văn ngắn, từ vựng hạn chế, một người nói
* Nhận dạng câu văn ngắn, từ vựng hạn chế và nhiều người nói
* Nhận dạng lời nói liên tục, một hoặc nhiều ngừoi nói

## Nhiệm vụ của đồ án

Tự thu mẫu các số đếm tiếng Việt từ không đến chín (hơn 1000 mẫu) với một kênh, tần số lấy mẫu 16 kHz, 16 bit/mẫu.

Tiến hành xử lý, trích xuất đặc trưng của các mẫu.

Huấn luyện theo mô hình mạng nơ-ron.

Tiến hành đánh giá mô hình với độ chính xác đạt từ 92% trở lên, có thể lên đến 100% đối với tập Test.

# CƠ SỞ LÝ THUYẾT VỀ XỬ LÝ TIẾNG NÓI

## Tổng quan về tiếng nói

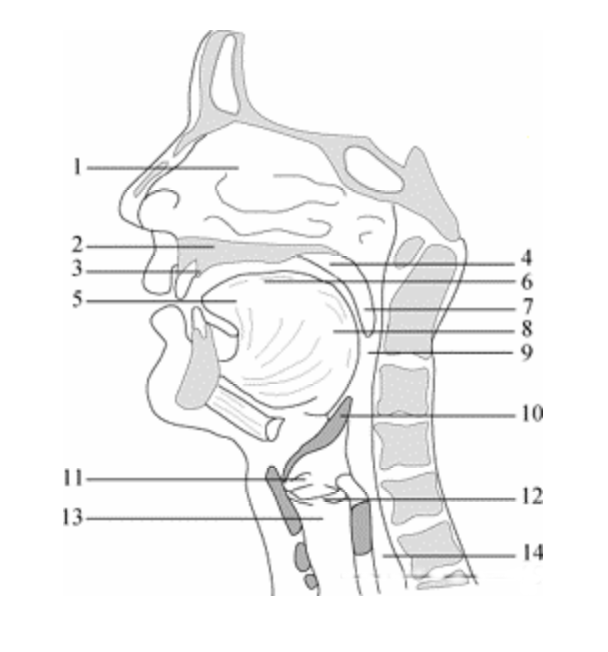
### Khái niệm về tiếng nói

Trong cuộc sống hàng ngày, tiếng nói là một phương tiện cơ bản và quan trọng trong giao tiếp giữa con người với nhau.

Cần phân biệt tiếng nói với các âm thanh khác bởi đặc tính âm học có nguồn gốc từ cơ chế tạo ra tiếng nói.

### Cơ chế tạo tiếng nói

Bộ máy phát âm bao gồm các thành phần riêng rẽ như phổi, khí quản, thanh quản và các đường dẫn miệng, mũi. Trong đó:

* Thanh quản chứa hai dây thanh có thể dao động tạo ra sự cộng hưởng cần thiết để tạo ra âm thanh.
* Tuyến âm là ống không đều bắt đầu từ môi, kết thúc bởi dây thanh hoặc thanh quản.
* Khoang mũi là ống không đều bắt đầu từ môi, kết thúc bởi vòm miệng, có độ dài ổn định khoảng 12cm đồi với người lớn.
* Vòm miệng là các nếp cơ chuyển động.

1. Hốc mũi
2. Vòm miệng
3. Ổ răng
4. Vòm miệng mềm
5. Đầu lưỡi
6. Thân lưỡi
7. Lưỡi gà
8. Cơ miệng
9. Yết hầu
10. Nắp đóng thanh quản
11. Dây thanh giả
12. Dây thanh
13. Thanh quản
14. Thực quản

Hình . Bộ máy phát âm của com người

Cơ chế phát âm: Trong quá trình tạo âm thanh không phải là âm mũi, vòm miệng mở, khoang mũi đóng lại, dòng khí sẽ chỉ đi qua khoang mũi. Khi phát âm mũi, vòm miệng hạ thấp và dòng khí sẽ chỉ đi qua mũi. Tuyến âm sẽ được kích thích bởi nguồn năng lượng tại chính thanh môn. Tiếng nói được tạo ra do tín hiệu từ nguồn từ thanh môn phát ra, đẩy không khí trong phổi tạo thành dòng khí, va chạm vào hai dây thanh trong tuyến âm. Hai dây thanh dao động sẽ tạo ra cộng hưởng, dao động âm sẽ được lan truyền theo tuyến âm (tính từ tuyến âm đến khoang miệng) và sau khi đi qua khoang mũi và môi, sẽ tạo ra tiếng nói.

### Các thành phần cơ bản của ngữ điệu tiếng nói

Ngữ điệu là một trong những thành phần tổng quát và quan trọng của tiếng nói, bởi tất cả các ngôn ngữ đều có ngữ điệu. Trong ngành ngôn ngữ học, ngữ điệu là một thành phần của của ngữ âm và được biểu diễn bởi các yếu tố vật lý như thời gian, cao độ, cường độ, phổ. Và một hệ thống ngữ điệu là sự thay đổi về cao độ, cường độ, thời gian ngắt nghỉ, phổ của một câu nhằm thể hiện một ý nghĩa, sắc thái tình cảm khi giao tiếp bằng tiếng nói.

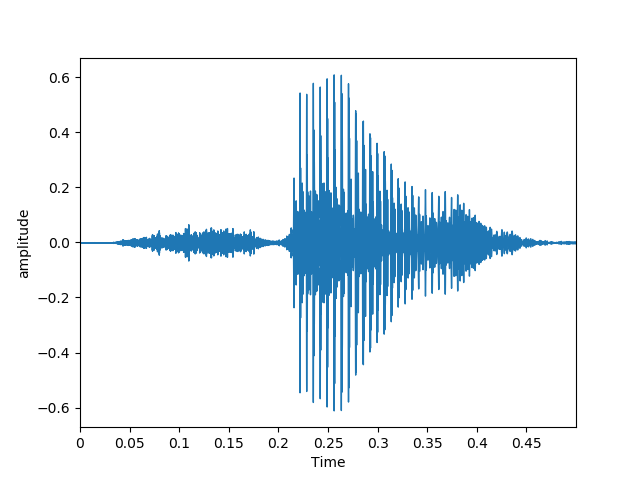
Các thành phần của ngữ điệu:

* Cao độ (pitch): trong số các yếu tố ngôn ngôn điệu, sự thay đổi cao độ rõ ràng, dễ thấy nhất. Các thay đổi này hợp thành đường cao độ của lời nói. Thuật ngữ cao độ còn dùng để chỉ tần số cơ bản F0 mà người nghe có thể nghe được, nói chung F0 và cao độ có thể coi là một. Bằng cách thay đổi độ căng của dây thanh, người nói có thể điều chỉnh âm tần số cơ bản. Bình thường tần số cơ bản của giọng nam thay đổi trong khoảng 80Hz đến 250Hz, trong khi giọng nữ là từ 120Hz đến 400Hz.
* Trường độ: Quan tâm đến độ dài của câu, của đoạn, của từ, của âm tiết, phầm âm không gió, trong một âm tiết, phần nguyên âm của âm tiết. Độ dài của âm tiết và lời nói phụ thuộc hoặc phụ thuộc lẫn nhau vào một số yếu tố như tốc độ nói, nhịp điệu, bản chất ngữ âm,… Trong phần lớn trường hợp, trường độ tuyệt đối của một yếu tố được ước lượng một cách dễ dàng.
* Cường độ: là một thuộc tính ngôn điệu đã được mô tả từ nghiên cứu đầu tiên về ngôn điệu trong ngữ âm học. Người ta cho rằng nó liên quan đến âm lượng cũng như các lực âm vị. Cả hai tính chất này đều ám chỉ đến dạng cảm nhận của ngôn điệu: âm tiết mạng trọng âm nổi bật hơn so với các âm tiết xung quanh, do âm lượng lớn hoặc các thuộc tính động của chúng.

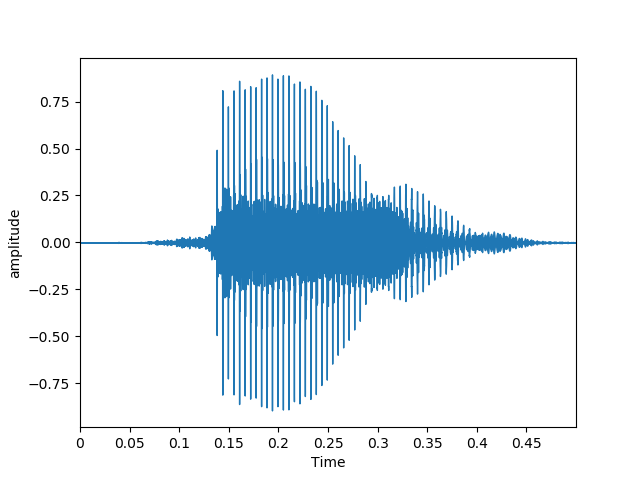
### Biểu diễn tín hiệu tiếng nói

Tiếng nói được coi là một dạng tín hiệu.

Biểu diễn dưới dạng sóng:



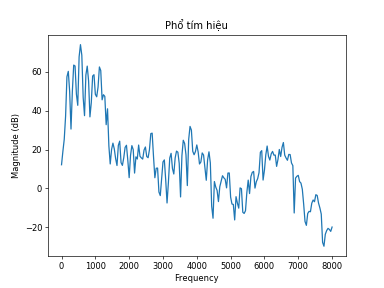
Hình . Biểu diễn tín hiệu dạng sóng của âm số sáu



Hình . Biểu diễn dạng sóng của âm số hai

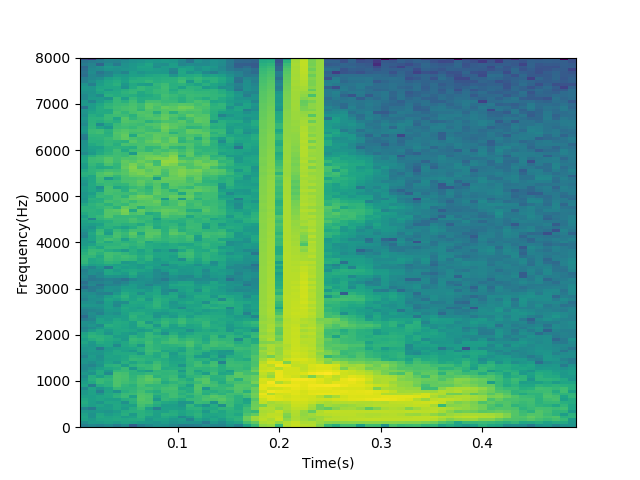
Tím hiệu tiếng nói được lưu trữ dưới dạng file .wav với các tần số lấy mẫu phổ biến 8.000Hz, 16.000Hz, 44.100Hz. Số bit/mẫu là 8/16 bit.

Biểu diễn dướng dạng phổ tín hiệu:

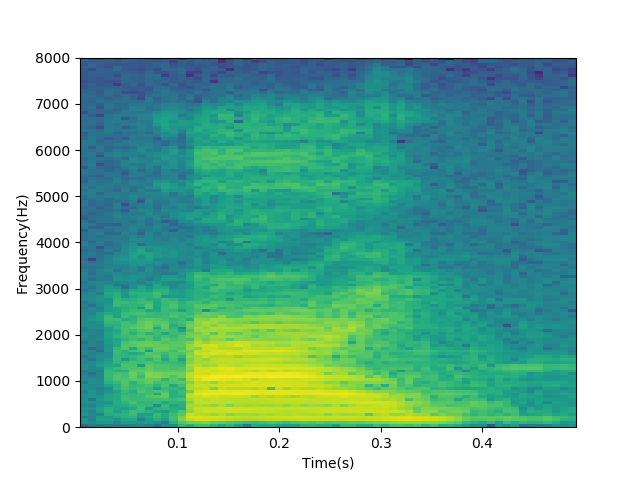


Hình . Biểu diễn dưới dạng phổ tín hiệu

Biểu diễn dưới dạng Spectrogram:



Hình . Biểu diễn Spectrogram của âm số sáu



Hình . Biểu diễn Specrogram của âm số hai

### Một số đặc điểm của ngữ âm tiếng Việt

* Đơn âm tiết:
* Có thanh điệu (6), biến đổi thanh điệu kèm theo biến đổi nghĩa.
* Không biến đổi hình thái.
* Hệ thống âm vị: 14 nguyên âm (11 nguyên âm đơn, 3 nguyên âm đôi) và 22 phụ âm.
* Cách phân biệt loại nguyên âm: phân biệt theo độ nâng của lưỡi và chuyển động lưỡi, phân biệt theo độ mở cửa miệng và chuyển động của lưỡi.
* Âm tắc: tiếng nổ, phát sinh do luồng khí từ phổi đi ra bị cản trở hoàn toàn, phải phá vỡ sự cản trở để thoát ra.
* Âm xát: Tiếng cọ xát, phát sinh do luồng khí đi ra bị cản trở không hoàn toàn (chỉ bị khó khăn), phải lách qua một khe hở nhỏ và trong khi thoát ra như vậy phải cọ xát vào thành của bộ phát âm.
* Phụ âm bên: đầu lưỡi tiếp xúc với lợi chặn lối thoát của của không khí, buộc nó phải lách quá khe hở ở hai bên cạnh lưỡi tiếp giáp với má mà ra ngoài tạo nên tiếng xát nhẹ (l).
* Luồng không khí thoát ra ngoài bị cản trở, tạo nên tiếng xát hay tiếng nổ, dạng tín hiệu không tuần hoàn gọi là tiếng động (ồn).
* Trong khi phát âm một số phụ âm, dây thanh cũng hoạt động đồng thời tạo nên tiếng thanh.
* Phụ âm có tỉ lệ tiếng động lớn hơn gọi là phụ âm ồn.
* Phụ âm có tiếng thanh lớn hơn gọi là phụ âm vang.

## Tổng quan về xử lý tiếng nói

### Vai trò của xử lý tiếng nói

Tiếng nói là phương tiện giao tiếp cơ bản và chủ yếu giữa con người trong cuộc sống hàng ngày. Hiện nay tiếng còn còn được áp dụng vào việc giao tiếp giữa người và máy, hướng tới thay thế các phương thức giao tiếp truyền thống nhứ sử dụng chuột, bàn phím, màn hình,… Và xử lý tiếng nói giữa vai trò quan trọng trong vấn đề giao tiếp này.

Xử lý tiếng nói là sự nghiên cứu tiếng nói của con người dưới dạng tín hiệu và các phương pháp xử lý tín hiệu này.

Xử lý thông tin chứa trong tín hiệu tiếng nói nhằm truyền, lưu trữ tín hiệu này hoặc tổng hợp, nhận dạng tiếng nói. Xử lý tiếng nói có thể chia thành các mục:

Hình . Ứng dụng của xử lý tiếng nói

* Nhận dạng tiếng nói: phân tích và xử lý về mặt nội dung ngôn ngữ của tín hiệu tiếng nói. Mục đích là để chuyển nội dung nói thành tín hiệu đầu vào của máy tính, giúp cho máy tính có thể xử lý và tương tác được với người nói.
* Nhận dạng người nói: mục đích là để nhận dạng người nói là ai và/hoặc là xác minh liệu người đang nói có đúng là người mà máy tính biết được hay không
* Tăng chất lượng tiếng nói: nhằm tăng sự cảm nhận của người nghe về chất lượng tiếng nói. Gồm: giảm nhiễu ồn của tín hiệu tiếng nói, giảm/khử tiếng vọng, …
* Mã hoá tiếng nói: là một dạng của nén dữ liệu có vai trò quan trọng trong lĩnh vực viễn thông. Ví dụ như truyền tiếng nói qua internet (voiIP), việc nén dữ liệu tiếng nói đều là điều bắt buộc để giảm băng thông đường truyền.
* Tổng hợp tiếng nói: là tạo tiếng nói một cách nhân tạo nhờ máy tính
* Phân tích giọng nói: được ứng dụng chủ yếu trong y học, nhằm phát hiện ra khuyết tật hay vấn đề của dây thanh âm, thanh quản, …
* Định vị nguồn âm thanh: xử lý tiếng nói để xác định vị trí nguồn phát ra âm. Nó được ứng dụng trong hội nghị hình thoại, khi vị trí người nói trong hội nghị được xác định, máy ghi sẽ được lập trình tự động quay đến chỗ vị trí đó và gửi hình ảnh đi. Ngoài ra định vj nguồn âm thanh còn ứng dụng trong các kỹ thuật tăng chất lượng tiếng nói, trong theo dõi an ninh, …

### Phương pháp xử lý tín hiệu tiếng nói thông qua phân tích phổ

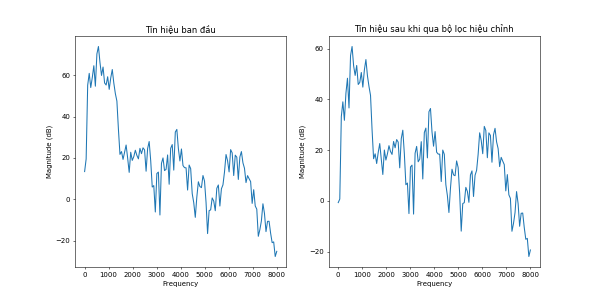
Khi xử lý tín hiệu nói chung và tín hiệu tiếng nói nói riêng thì chúng ta không làm việc trên miền thời gian, thay vào đó ta sẽ xử lý trên miền tần số. Miền tần số đề cập đến việc phân tích các hàm, tín hiệu liên quan đến tần số thay vì thời gian. Tức là miền thời gian sẽ biểu diễn tín hiệu thay đổi như thế nào theo thời gian còn miền tần số cho biết mức độ tín hiệu nằm trong mỗi dải tần số cho trước trong một dải tần số.

Một hàm hoặc tín hiệu nhất định có thể chuyển đổi giữa các miền thời gian và tần số thông qua hàm biến đổi. Ví dụ ta có có biến đổi Fourier có tác dụng biến đổi hàm thời gian thành tổng hoặc tích phân của các sóng hình sin có tần số khác nhau, mỗi sóng sẽ đại diện cho một thành phần tần số. Phổ của các thành phần tần số biểu diễn miền tần số của tín hiệu. Biến đổi Fourier ngược thì ngược lại, biến đổi hàm trên miền tần số trời lại về hàm trên miền thời gian. Phân tích phổ là một công cụ phổ biến thường được dùng để biểu diễn tín hiệu trên miền tần số. Trong xử lý tín hiệu số nói chung 2 phép biến đổi hay sử dụng là phép biến đổi Fourier (Fourier Transform, FT) và biến đổi Z ( Z – Transform, ZT) của các tín hiệu rời rạc.

Ta có sơ đồ tiến hành phân tích phổ từ tín hiệu tiếng nói:

Hình . Sơ đồ phân tích phổ

* Bộ lọc hiệu chỉnh: , với . Bộ lọc hiệu chỉnh có tác dụng tăng cường độ lớn của các tần số cao ngoài ra còn có tác dụng cân bằng tần số của phổ.



Hình . Tác dụng của bộ lọc hiệu chỉnh

* Sau khi qua bộ lọc, tín hiệu được chia thành các frames. Mỗi frame có độ dài khoảng 25ms và gối chồng lên nhau, tức là đầu frame sau sẽ chồng lên frame trước khoảng 10ms; nếu không gối lên nhau việc biên độ bị giảm đột ngộ sẽ khiến gây nhiễu ở tần số cao. Sau khi chia thành các frame thì từng frame sẽ đều độc lập thực hiện các phép biến đổi như nhau trong các bước tiếp theo.
* Cửa sổ Hamming: cửa sổ sẽ giúp cho thông tin trong tần số gốc tốt hơn khi gặp nhiễu; ngoài cửa sổ Hamming thì có thể sử dụng cửa sổ Hanning.

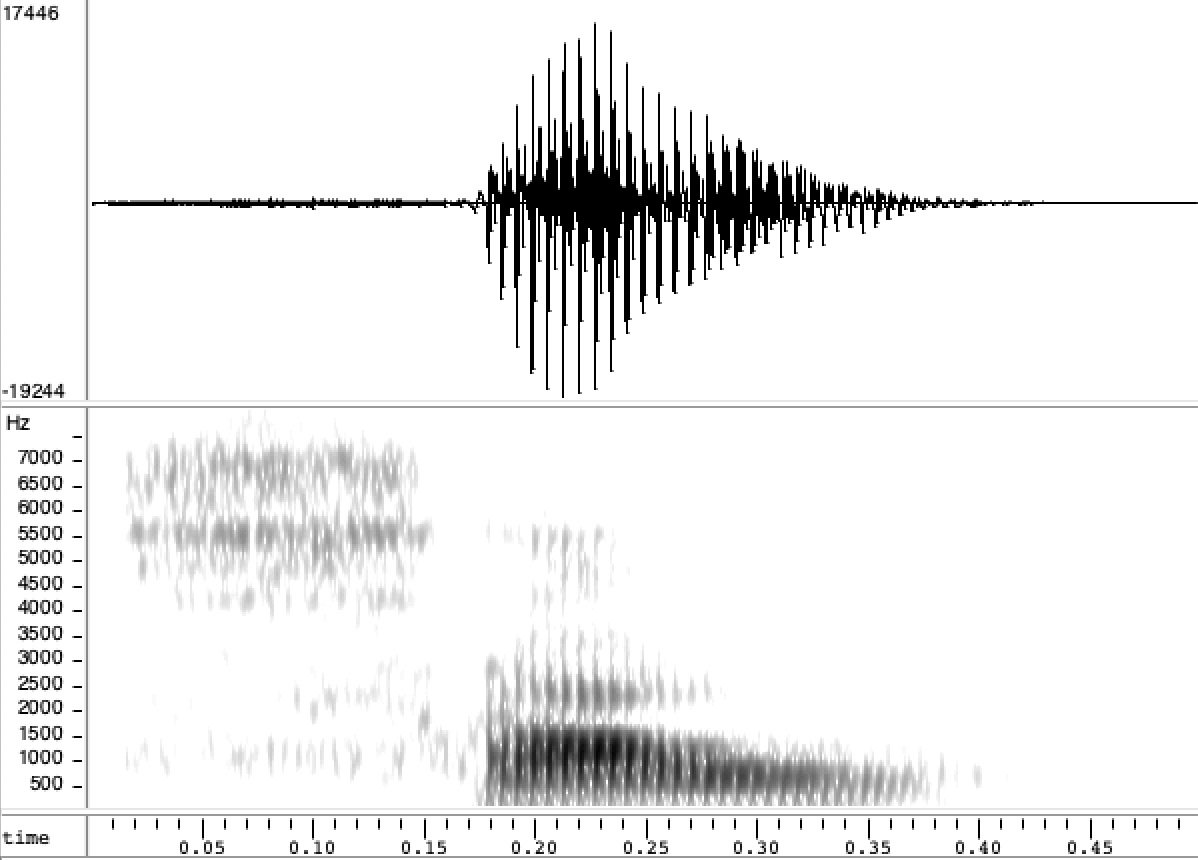
* FFT (Fast Furier Transform): biến đổi FFT là một thuật toán rất hiệu quả để tính toán biến đổi Fourier rời rạc (DFT) và biến đổi ngược. Sau bước này ta thu được phổ:

* Sau cùng lấy Log để đưa đơn vị theo trục tung của Phổ thành Ben (B). Thông thường, người ta sẽ sử dụng đơn vị Decibel (dB) với

1B = 1/10dB.

## Một số đặc trưng thường được sử dụng trong nhận dạng tiếng nói

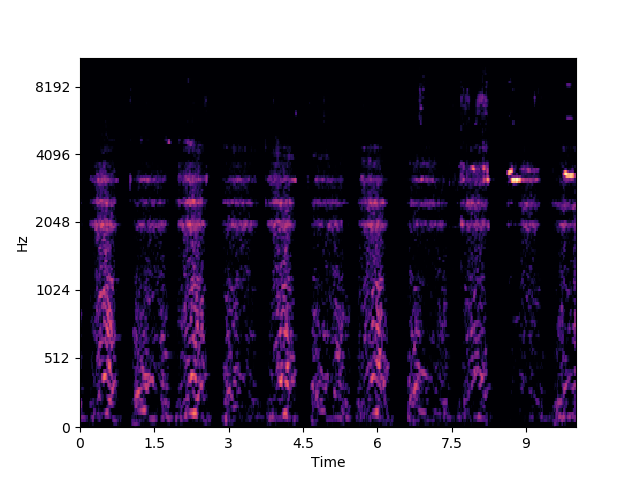
Trong xử lý tín hiệu, Phổ hay còn gọi là Spectrum thể hiện mối qua hệ giữa tần số và biên độ. Thông thường người ta sẽ lấy Logarithmic độ lớn của tần số để chuyển sang đơn vị Decibel. Vấn đề là Phổ chỉ đại diện cho 1 frame trong tín hiệu được đưa vào. Nên để tìm đặc trưng cho cả đoạn tín hiệu tiếng nói ta sẽ sử dụng đặc trưng Spectrogram. Spectrogram biểu hiện mối quan hệ giữa tần số và thời gian:



Hình . Đặc trưng Spectrogram của âm tiết ‘sáu’



Theo như ta thấy, Spectrogram thể mối quan hệ giữa tần số và thời gian trong cả đoạn tín hiệu giọng nói nhưng hoàn toàn không rõ ràng. Như trên hình vẽ nó chỉ bao gồm vài chấm nhỏ, việc lấy đặc trưng gần như là không thể. Nên người ta đã lấy Log và đạt kết quả như hình:



Hình . Đặc trưng Log Spectrogram

Theo như hình trên, việc lấy đặc trưng của âm thanh hoàn toàn là có thể. Log Spectrogram sẽ trả về kết quả như là một tấm ảnh, ta có thể đưa quy từ bài toán nhận dạng tiếng nói về bài toán nhận dạng ảnh. Từ tín hiệu âm thanh, tiến hành xử lý, trích xuất đặc trưng Log Spectrogram và đưa vào mạng nơ-ron để huấn luyện.

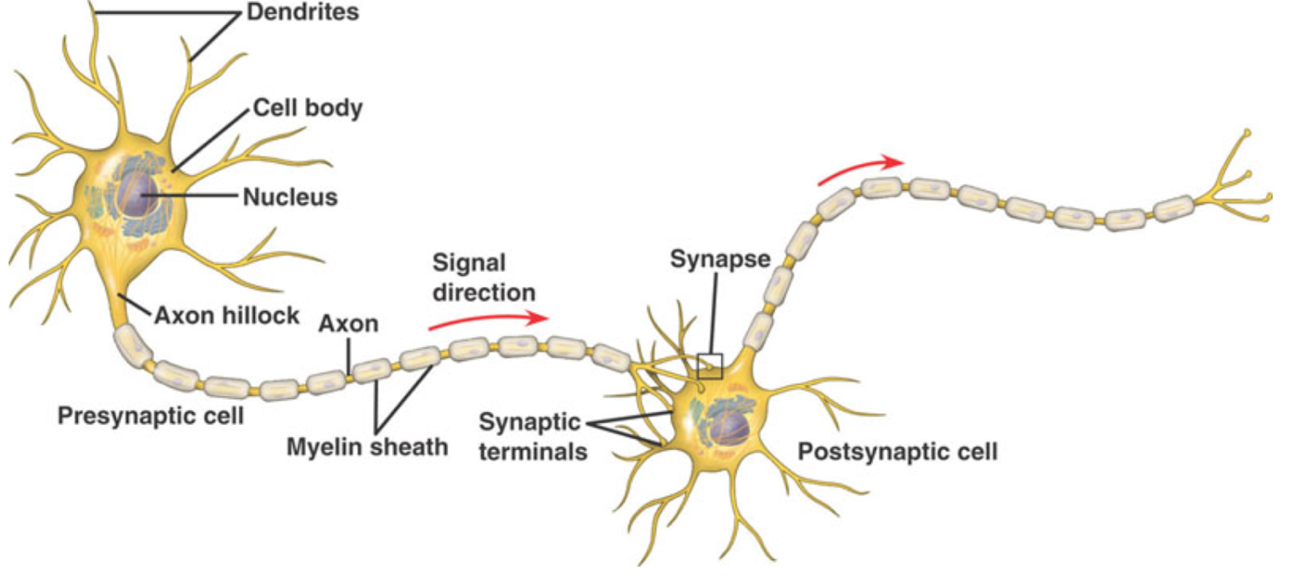
# MẠNG NƠ-RON

## Tổng quan về mạng nơ-ron nhân tạo

### Perceptron

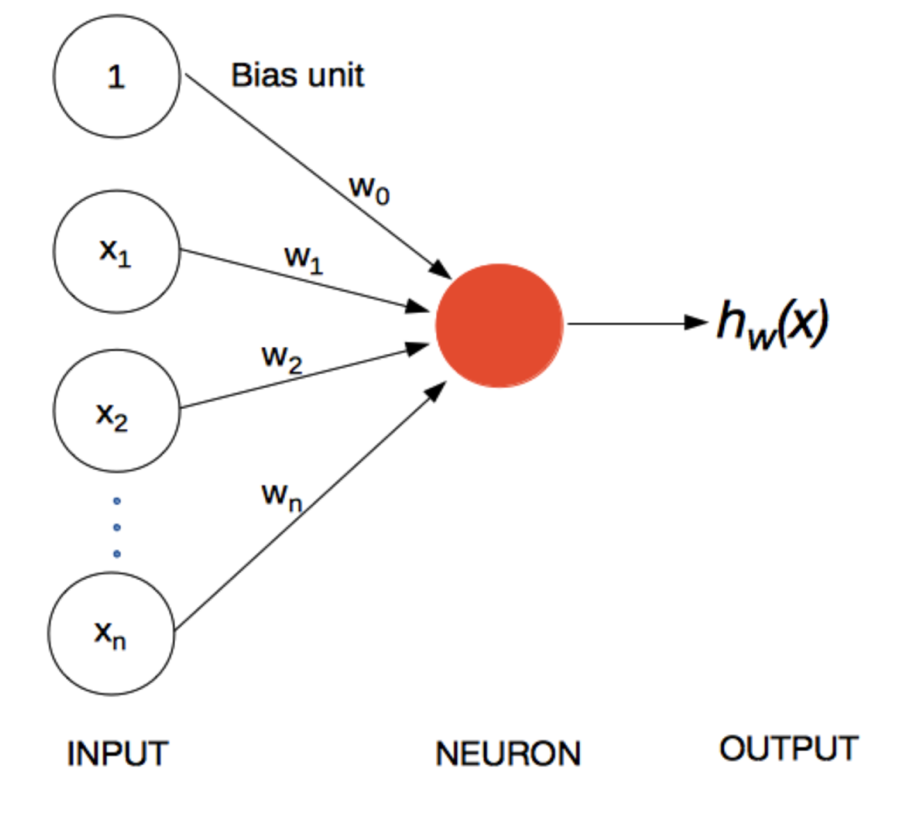
Về mặt sinh học, nơ-ron là đơn vị cấu tạo cơ bản của hệ thần kinh của hầu hết các loài động vật và là thành phần quan trọng bậc nhất của não. Mạng nơ-ron là tập hợp các dây thần kinh kết nối với nhau. Ngày nay, thuật ngữ này còn để chỉ mạng nơ-ron nhân tạo được dùng để thiết kế mô hình một số tính chất của mạng nơ-ron tự nhiên.

Việc nói các mạng nơ-ron mô phỏng mà cách não chúng ta hoạt động là chưa đúng. Vì hiện nay, mạng nơ-ron chỉ sử dụng các nguyên lý hoạt động đơn giản nhất của các nơ-ron thần kinh dựa trên nghiên cứu từ những năm 1940. Từ đó đến bây giờ, các nhà khoa học đã khám phá ra rằng các nơ-ron trong hệ thần kinh hoạt động phức tạp hơn rất nhiều. Dù chỉ với những khái niệm đơn giản ấy đã đem lại những thành công to lớn trong lĩnh vựng trí tuệ nhân tạo như hiện nay. Mô hình mạng nơ-ron đầu tiên được gọi là Perceptron, là nền tảng cho các mô hình mạng nơ-ron hiện nay. Trước hết hãy xem các nơ-ron thần kinh hoạt động:



Hình . Cấu tạo nơ-ron sinh học

Một nơ-ron nhận tín hiệu điện có chứa các thông tin từ các synapse của một hay nhiều nơ-ron khác thông qua gai (dendirt). Các giá trị đầu vào (**input**) sẽ nhận được tích tụ lại trong thân nơ-ron (cell body). Nếu tổng này vượt quá ngưỡng (threshold) nhất định thì nơ-ron sẽ phát ra một tín hiệu điện đầu ta (**output**) truyền qua sợi trục (axon) tới các synapse. Lúc này nơ-ron ở trạng thái gọi là là **firing** hay **activated**. Tín hiệu output nãy sẽ được truyền sang các nơ-ron khác nhờ vào sự liên kết giữ synapse và dendirt, độ mạnh yếu của các liên kết sẽ quyết định lượng thông tin được truyền sang. Cứ như thế quá trình này khi diễn ra đồng thời giữa một mạng lưới nơ-ron sẽ tạo nên sự hoạt động của hệ thần kinh trong bộ não người. Vào năm 1958, nhà khoa học Frank Rosenblatt đã dựa trên nguyên lý trên để đề xuất ra mô hình Perceptron.



Hình . Perceptron-mô hình đơn giản của mộ mạng nơ-ron nhân tạo

Trong mô hình trên, hình tròn màu đỏ đại diện cho thân nơ-ron. Các giá trị đầu vào là đại diện cho các giá trị truyền từ các synapse của nơ-ron khác. Các giá trị này được nhân với (được gọi là các **weights**) tương ứng trước khi truyền vào thân nơ-ron. Đây là cách Perceptron mô phỏng độ mạnh yếu giữa các liên kết synapse-dendirt để điều chỉnh lượng thông tin được truyền sang. Nếu ta gọi tổng của các tín hiệu được hội tụ tại thân nơ-ron là thì ta sẽ có:

Cũng giống như nguyên lý hoạt động của một nơ-ron sinh học đã nói ở trên, nếu tín hiệu này lớn hơn một ngưỡng threshold thì nơ-ron sẽ vào trạng thái **activated** và xuất ra một tín hiệu output là . Quá trình này được mô tả bởi **activation** **function**. Có thể mô tả đơn giản như sau:

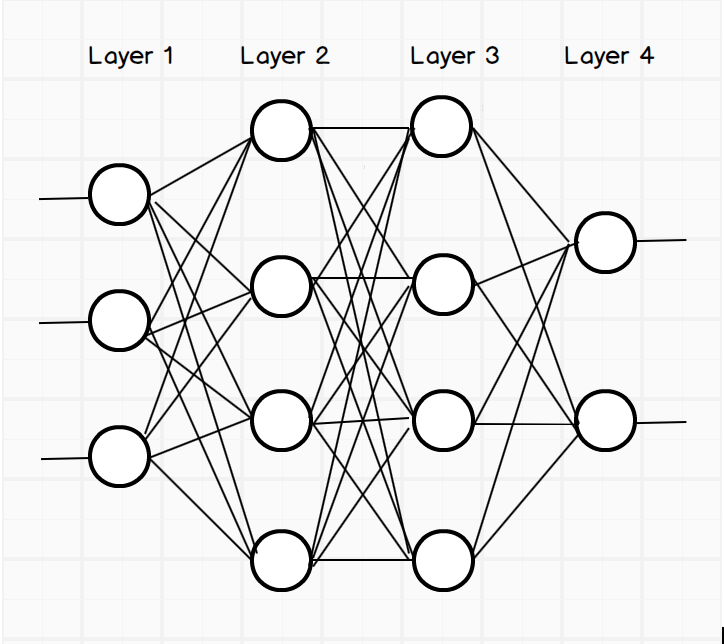
Nghĩa là nếu tổng **input** lớn hơn hoặc bằng ngưỡng thì giá trị **output** sẽ là 1, còn ngược lại thì **output** mang giá trị 0. Nếu đặt ta có biểu thức mới:

Do vậy với bất kể giá trị nào của ngưỡng ta cũng có thể tính tổng lượng tín hiệu đầu vào của cũng như cộng thêm giá trị mà ta có thể coi là tích của một đầu vào với giá trị bằng 1 với **weight** là như trong Hình 3.1 được gọi là **bias unit.** Sau đó so sánh tổng này với 0 để quyết định giá trị đầu ra h.

Như ta thấy, việc sử dụng Perceptron có thể áp dụng cho bài toán phân lớp nhị phân, trong đó ta sẽ dựa trên các giá trị đầu vào quyết định chia đối tượng vào nhóm 1 hay nhóm 2 cho trước. Ví dụ như phân biệt ảnh có người hay không có người, đầu vào sẽ là ảnh được biểu hiện dưới dạng các pixel, từ các pixel suy ra đặc điểm. Đưa vào Perceptron để tìm ra kết quả. Do đó việc tìm các **weight** cho Perceptron học các tìm ra giá trị chính xác nhất. Các tìm phổ biến và hiệu quả là sử dụng thuật toán **Gradient Descent** được giới thiệu ở phần tiếp theo.

### Cấu tạo mạng nơ-ron và hoạt động cơ bản

Perceptron là mô hình căn bản của mạng nơ-ron nhân tạo. Từ cơ sở đó người ta phát triển lên những mô hình mạng phức tạp như hiện nay. Mạng có tên là Mutil-layer Perceptron (MLP). Đúng như tên gọi, bởi nó là tập hợp các Perceptron được chia làm nhiều lớp.



Hình . Mô hình MPL đơn giản

Hình vẽ trên mô tả một mạng nơ-ron đơn giản với 4 lớp, layer 1 được gọi là lớp đầu vào, các layer 2 và 3 ở giữa là các lớp ẩn còn lớp 4 được là lớp đầu ra. Mỗi lớp gồm các node hay còn gọi là các nơ-ron.

Từ tập dữ liệu ban đầu, thông qua các phép biến đổi từ những dữ liệu đầu vào sẽ được chuyển thành một vector đặc trưng. Sự biến đổi đổi gọi là quán trình *preprocess,* mỗi bài toán khác nhau có cách biến đổi khác nhau; bởi đầu vào khác nhau . Mạng Nơ-ron nhận đầu vào là vector đặc trưng, đi tiếp vào các lớp ẩn và rồi kết thúc ở lớp đầu ra với giá trị là kết quả dự đoán.

Sơ đồ trên được gọi là một mạng fully connected bởi các nơ-ron lớp trước đều liên kết với các nơ-ron lớp sau. Trên mỗi kết nối có các trọng số *weight* quyết định độ mạnh của tín hiệu từ lớp trước đến lớp sau. Trong quá trình huấn luyện (Training) các trọng số *weight* sẽ được điều chỉnh sao cho đầu ra phù hợp với kết quả mong muốn.

Sau khi đi qua một lớp, trước khi đi vào lớp tiếp theo kết quả sẽ được đi qua hàm *activation.* Hàm *activation* sẽ loại bỏ sự tuyến tính của mô hình huấn luyện. Một số hàm *activation*:

* Sigmoid:
* Tanh:
* ReLU:

Trong quá trình huấn luyện: hàm *loss* là chìa khoá khiến cho việc sử dụng mạng Nơ-ron đạt hiệu quả cao. Nó giúp tính toán hiệu suất của việc huấn lyện trên tập Train, nhờ đó có thể hiệu chỉnh các tham số *weight* sao cho phù hợp. Một số hàm *loss*:

* CrossEntropyLoss
* MSELoss

*Optimizer*: huấn luyện mô hình bằng cách đưa dữ liệu vào, tính toán hàm *loss* để xác định sự khác biệt giữa kết quả mà mô hình tính ra và kết quả mong muốn. Và sử dụng các thông tin này để điều chỉnh các tham số *weight* sao cho hàm *loss* đạt giá trị nhỏ nhất có thể. Việc cập nhật các weight sẽ do *optimizer* đảm nhiệm. Một số *optimizer:*

* Adam
* SGD (Stochastic Gradient Descent)
* RMSProp
* AdaGrad

### Dữ liệu cho mạng nơ-ron

Trước hết để giải quyết được bài toán đặt ra bằng mạng Nơ-ron vấn đề luôn là tập dữ liệu đầu vào. Dữ liệu bao nhiêu cho đủ? Câu trả lời là càng nhiều mô hình mạng sẽ học được càng nhiều mẫu hay nói cách khác là nó có khả năng nhận biết càng chính xác. Hiện nay với mục đích học tập thì có rất nhiều trang cung cấp có ta khối lượng dữ liệu khổng lồ như: MNIST, Kaggle,…

Có dữ liệu rồi chúng ta làm gì tiếp theo? Tiếp theo chúng ta cần phân chia tập dữ liệu để huần luyện (tập Train), đánh giá trong quá trình huấn luyện (tập Validation) và tập đánh giá mô hình sau quá trình huấn luyện (tập Test). Thông thường chúng ta có thể chia tỉ lệ 60:20:20 lần lượt cho các tập Train, Validation và Test. Nhưng khi chia tập Train cần chú ý số lượng mẫu của các lớp cần phải không chênh lệch quá nhiều; nó cũng giống như việc học đều thì hiểu biết đều còn học không cân bằng sẽ dẫn đến việc nhận biết kém. Nếu việc cân bằng dữ liệu tập Train là không thể, chúng ta có thể tăng cường dữ liệu; bằng các biến đổi dữ liệu gốc đi, ví dụ như trong nhận dạng ảnh có thể dịch ảnh và tạo thành mẫu mới.

## Mạng nơ-ron tích chập (Convolution Neural Network, CNNs)

### Tổng quan về mạng CNN

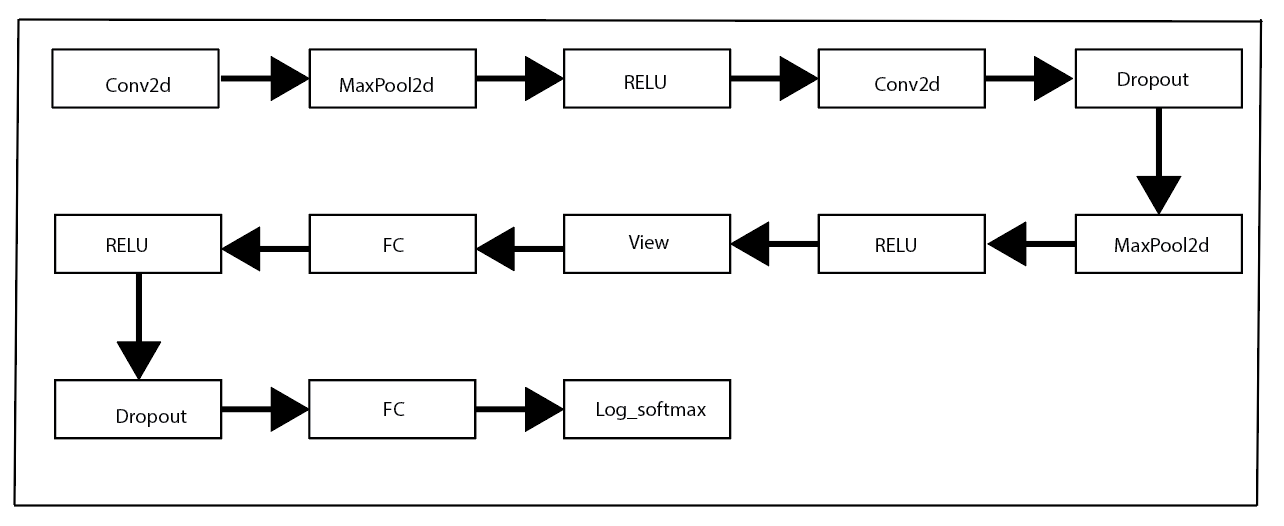
Gần đây trong lĩnh vực về Thị giác máy tính (Computer Vision) đã đạt được những thành tựu ấn tượng; ví dụ như: hệ thống nhận diện khuôn mặt, phát triển xe tự hành,… Nhờ có mạng nơ-ron tích chập đem lại một hiệu quả tốt hơn so với mạng nơ-ron truyền thống.

Mạng nơ-ron thích chập là một trong những mô hình tiên tiến giúp cho chúng ta có thể xây dựng được các hệ thống đạt độ hiệu quả chính xác gần như bằng con người.

Một số mạng nơ-ron tích chập được phát triển từ lâu:

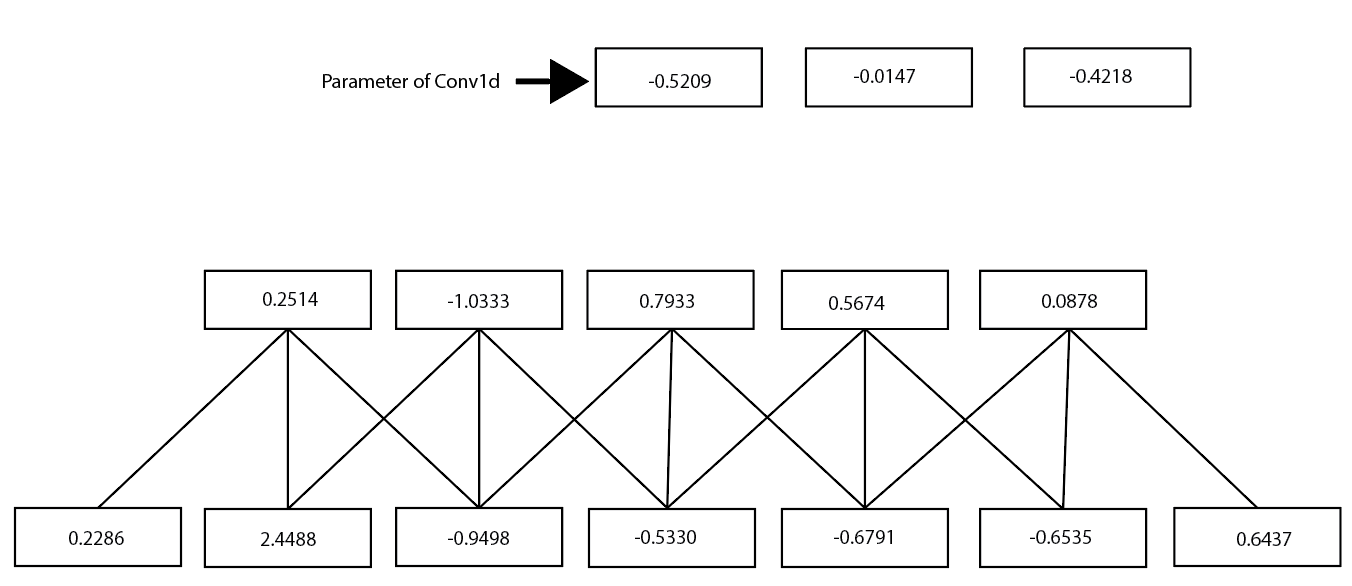
* AlexNet
* GoogleLeNet
* VGG
* ResNet

### Cách hoạt động của mạng CNN



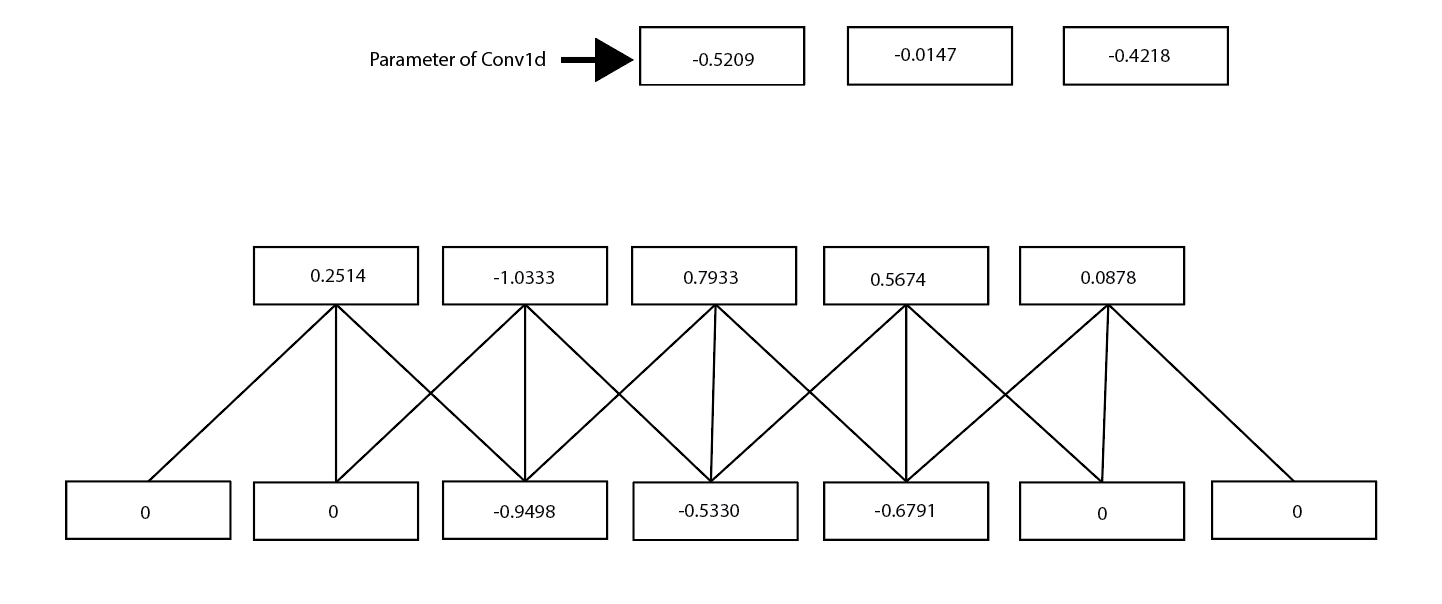
Hình . Cấu trức mạng CNN đơn giản

* **Conv2d** ( Convolution 2-dimension) : Là tấm lọc 2 chiều, để hiểu cách thức hoạt động của nó trước hết là hãy xem cách **conv1d** ( có kích thước là 3) hoạt động với một mảng 7 phần tử :



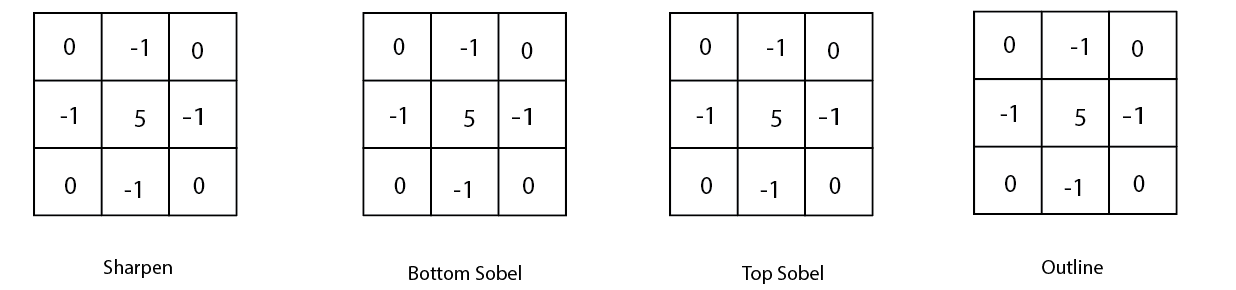
Hình . Con1d với stride=1

Theo sơ đồ, ta có giá trị Output1: ( -0.5209 x 0.2286) + ( -0.0147 x 2.448) + ( -0.4218 x -0.9498) = 0.2514. Số phần tử bị bỏ qua hay bước nhảy được gọi là **stride** thường có giá trị là 1. Còn một thông số quan trọng nữa gọi là **padding** được sử dụng khi tấm lọc không được sử dụng cho toàn bộ dữ liệu đưa vào hoặc không đủ tham số truyền vào, ví dụ:



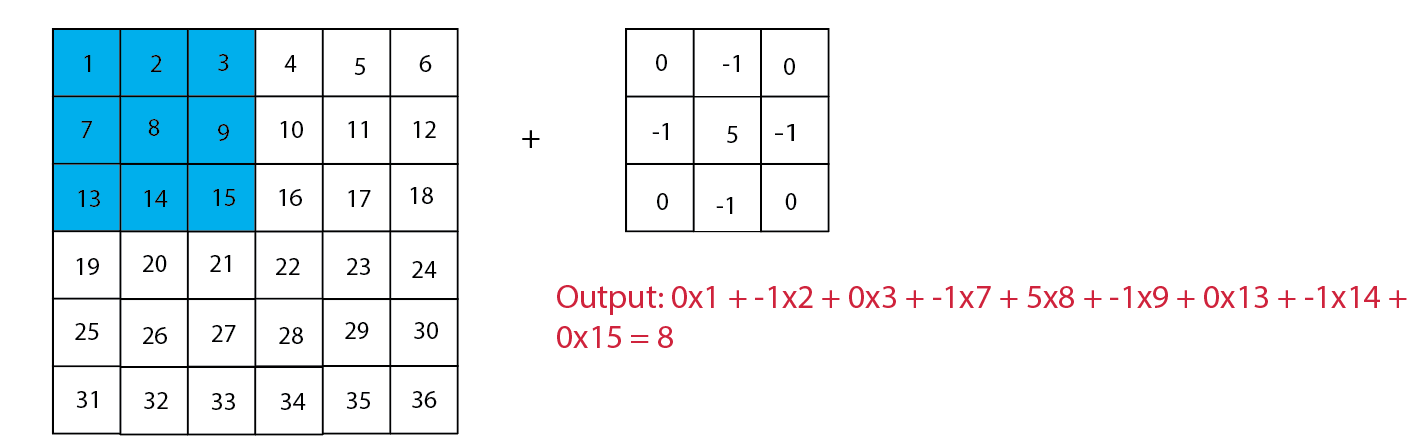
Hình . Conv1d với stride=1, padding=2

Bây giờ hãy xem cách **conv2d** hoạt động: nó hoạt động như một bộ lọc ảnh kích thước là 3x3, có khá nhiều dạng của bộ lọc này :



Hình . Một số bộ lọc thông dụng

Minh minh hoạ cách hoạt động của bộ lọc cho phần màu xanh:

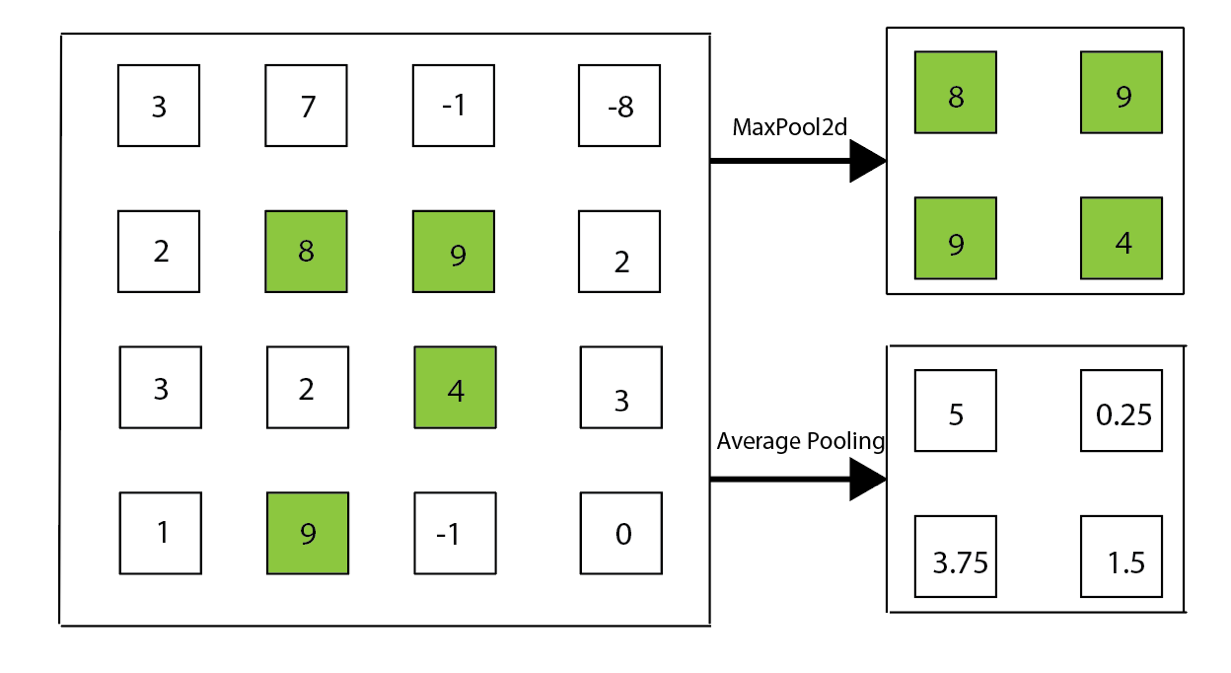


Hình . Cách hoạt động của bộ lọc

Nhưng thực tế trong quá trình chạy thì các giá trị của cửa sổ lọc được tạo ngẫu nhiên và việc điều chỉnh tham số sẽ là công việc của gradient descent và backpropagation. Trong quá trình học cửa sổ lọc sẽ chịu trách nhiệm nhận diện đường thẳng, đường cong, các đặc trưng của đối tượng. Chúng ta có thể lựa chọn kích thước **kernel\_size** của cửa sổ lọc bao gồm: 1, 3, 5 và 7.

* **Pooling:** Đây là bước gộp các layers sau convolution layers, nhằm giảm thiểu kích thước dữ liệu. Layer này có 2 đặc trưng:
  + Giảm kích thước dữ liệu cần xử lý.
  + Thuật toán tập chung xử lý ở những điểm quan trọng. Ví dụ như thuật toán nhận diện khuôn mặt thì cần phải phát hiện khuôn mặt trong ảnh bất kể đối tượng có ở vị trí nào trong ảnh đi chăng nữa.

Pooling khá giống với khái niệm cửa số lọc, nhưng điều khác biệt là nó sẽ chỉ dựa vào dữ liệu của layers trước để tính toán. Ngoài **MaxPool2d** thì còn có **Average Pooling.** Lấy một ví dụ đơn giản như sau:



Hình . MaxPool2d và Average Pooling

Nhờ đó đã đem lại sự mạnh mẽ trong khả năng nhận diện hình ảnh của mạng CNN.

# GIẢI QUYẾT BÀI TOÁN

## Phân tích bài bài toán

### Bài toán và cách tiếp cận

Bài toán thực hiện phân lớp giọng nói của các số đếm tiếng Việt từ chín đến chín bởi một người nói.

Cách tiếp cận bài toán: Huấn luyện một mô hình duy nhất để phân lớp 10 âm các số đếm tiếng Việt từ không đến chín.

### Tiêu chí đánh giá

Mô hình được đánh giá được tính bằng giá trị của Accuracy(%)

Accuracy =

## Dữ liệu

Tập dữ liệu được tự thu âm bằng microphone trên laptop. Tiến hành tạo ra các số đếm tiếng việt từ chín đến chín một cách ngẫu nhiên sau đó phát âm và thu âm lại. Các mẫu thu cần đảm bảo thu trọn vẹn âm và không chứa quá nhiều khoảng lặng.

Tập dữ liệu sau khi thu âm gồm 1206 tệp âm thanh, độ dài mỗi tệp là 0.5s, được thu với tần số lấy mẫu là 16 KHz, 16bit/mẫu, 1 kênh; bao gồm các số đếm tiếng việt từ không đến chín bởi một người nói.

Tiến hành chia tập dữ liệu đầu vào theo tỉ lệ 70:30. 70% cho quá trình huấn luyện và tiến hành đánh giá trên 30% (tập Test).

### Tập train

Ta có biểu đồ tập dữ liệu gốc:

Hình . Biểu đồ sự phân bố các lớp tập dữ liệu gốc

Từ tập huấn luyện ta chia với tỉ lệ 80:20 cho việc huấn luyện (tập Train) và đánh giá trong quá trình huấn luyện (tập Validation).

Ta có biểu đồ phân bố các lớp tập Train:

Hình . Biểu đồ sự phân bố các lớp tập Train

Nhận xét: Sự phân bố về số lượng của các lớp là khá đồng đều trong tập Train.

|  |  |
| --- | --- |
| Lớp | Tổng số mẫu |
| 0 | 83 |
| 1 | 84 |
| 2 | 87 |
| 3 | 85 |
| 4 | 87 |
| 5 | 81 |
| 6 | 85 |
| 7 | 84 |
| 8 | 87 |
| 9 | 87 |
|  | Tổng = 850 |

Bảng . Tổng các mẫu mỗi lớp trong tập Train và Validation

### Tập Test

Ta có biểu đồ tập Test:

Bảng . Biểu đồ sự phân bố các lớp tập dữ liệu tập Test

|  |  |
| --- | --- |
| Lớp | Tổng số mẫu |
| 0 | 35 |
| 1 | 38 |
| 2 | 36 |
| 3 | 40 |
| 4 | 36 |
| 5 | 36 |
| 6 | 34 |
| 7 | 37 |
| 8 | 30 |
| 9 | 34 |
|  | Tổng = 356 |

Bảng . Tổng số các mẫu từng lớp trong tập Test

## Chia dữ liệu từ dữ liệu ban đầu

Hàm lấy thông tin về các mẫu của từng nhãn

|  |
| --- |
| # return list file each label  def compute\_label(path):  list\_data\_each\_label = []  for i in range(10):  files = glob.glob('{}/{}\_\*.wav'.format(path, i))  list\_data\_each\_label.append(files)  return list\_data\_each\_label |

Sau khi có được danh sách các mẫu trong từng nhãn tiến hành phân chia dữ liệu.

Phân chia dữ liệu ra (Train+Validation) và Test:

|  |
| --- |
| # train 70% test 30%  def make\_train\_and\_test(list\_data, flag):  random.shuffle(list\_data)  flag = flag+ random.randrange(-4, 4)  # copy to train+val path  for file in list\_data[:flag]:  dst = DATA\_TO\_TRAIN\_PATH + '/' + file.split('/')[-1]  shutil.copy(file, dst)  # copy to test path  for file in list\_data[flag:]:  dst = TEST\_PATH + '/' + file.split('/')[-1]  shutil.copy(file, dst) |

Tiếp tục phân chia (Train + Validation) thành 2 tập riêng Train và Validation.

|  |
| --- |
| # 80% and train 20% validation  def make\_train\_and\_validation(list\_data):  flag = int(0.8 \* len(list\_data))  random.shuffle(list\_data)  # copy to train path  for file in list\_data[:flag]:  dst = TRAIN\_PATH + '/' + file.split('/')[-1]  shutil.copy(file, dst)  # copy to val path  for file in list\_data[flag:]:  dst = VAL\_PATH + '/' + file.split('/')[-1]  shutil.copy(file, dst) |

## Tiến hành xử lý tín hiệu tiếng nói

Trích xuất trọn vẹn đặc trưng Log Spectrogram các mẫu (mỗi mẫu 0.5s) thu được một ma trận. Coi đó như là ảnh một kênh và quy bài toán về nhận dạng ảnh.

Trước hết từ các tín hiệu thu đưa vào bộ lọc hiệu chỉnh:

|  |
| --- |
| # x(n) = x(n) – alpha\*x(x-1)  def pre\_emphesis(data, alpha=0.97):  return np.append(data[0], data[1:] - alpha\*data[:-1]) |

Sau đó ta sử dụng thư viện scipy để trích chọn đặc trưng Spectrogram, với các tham số: **window\_size**=0.02 là chiều dài frame(s), **step\_size**=0.01 là chiều dài chồng nhau của các frames(s), **nfft**=512, **eps=**1e-10 để tránh lấy phải giá trị log(0).

|  |
| --- |
| # get log spectrogram  def log\_spectrogram(data, sample\_rate, window\_size=0.02, step\_size=0.01, nfft=512, eps=1e-10):  window\_length = int(sample\_rate\*window\_size)  hop\_length = int(sample\_rate\*step\_size)  freqs, time, spec = signal.spectrogram(data, fs=sample\_rate, window='hann', nfft=nfft, nperseg=window\_length,  noverlap=hop\_length, detrend=False)  log\_spec = np.log10(spec.astype(np.float32)+eps)  return freqs, time, log\_spec |

Log Spectrogram thu được sẽ là một ma trận có kích cỡ 257x49. Ta chuyển nó thành ảnh một kênh với kích thước 257x50 để đưa vào mạng Nơ-ron tích chập:

|  |
| --- |
| # get log spectrogram with shape (257,50)  def convert2model(file):  sr, data = wavfile.read(file)  data = pre\_emphesis(data)  \_, \_, log\_spectro = log\_spectrogram(data, sample\_rate=sr)  log\_spectro = np.resize(log\_spectro, (1, 257, 50))  return log\_spectro |

## Xây dựng lớp Dataset để đưa vào mô hình huấn luyện

|  |
| --- |
| class VoiceDataset(Dataset):  def \_\_init\_\_(self, path):  files = Path(path).glob('\*.wav')  self.items = [(str(f), f.name.split('\_')[0]) for f in files]  self.length = len(self.items)  self.trans = transforms.Compose([  transforms.ToTensor()  ])  def \_\_getitem\_\_(self, index):  filename, label = self.items[index]  image = convert2model(filename)  tensor\_image = self.trans(image)  # convert to shape (257, 49, 1)  return tensor\_image.permute(1, 2, 0), int(label)  def \_\_len\_\_(self):  return self.length |

## Xây dựng mô hình huấn luyện

Xây dựng mô hình mạng Nơ-ron rích chập để huấn luyện:

|  |
| --- |
| class Net(nn.Module):  def \_\_init\_\_(self, num\_classes=10):  super(Net, self).\_\_init\_\_()  self.features = nn.Sequential(  nn.Conv2d(1, 64, kernel\_size=7, stride=1, padding=2),  nn.ReLU(),  nn.BatchNorm2d(64),  nn.MaxPool2d(kernel\_size=5, stride=2),  nn.Conv2d(64, 256, kernel\_size=5, padding=2),  nn.ReLU(),  nn.BatchNorm2d(256),  nn.MaxPool2d(kernel\_size=3, stride=2),  nn.Conv2d(256, 512, kernel\_size=3, stride=1),  nn.ReLU(),  nn.BatchNorm2d(512),  nn.Conv2d(512, 256, kernel\_size=3, padding=1),  nn.ReLU(),  nn.Conv2d(256, 256, kernel\_size=3, padding=1),  nn.MaxPool2d(kernel\_size=3, stride=2)  )  self.avgpool = nn.AdaptiveAvgPool2d((6, 6))  self.classifer = nn.Sequential(  nn.Dropout(),  nn.Linear(256\*6\*6, 1000),  nn.ReLU(),  nn.Dropout(),  nn.Linear(1000, 100),  nn.ReLU(),  nn.Linear(100, num\_classes)  )  def forward(self, x):  x = self.features(x)  x = self.avgpool(x)  x = torch.flatten(x, 1)  x = self.classifer(x)  return F.log\_softmax(x, dim=1) |

Sơ đồ mạng :

Hình . Sơ đồ mạng huấn luyện

### Tính toán các tham số của mô hình huấn luyện:

* Input: input là một ảnh nê không có tham số nào cả.
* Con2d(L, K): đầu vào có L kênh vào và K kênh ở đầu ra, bộ lọc (NxM) thì tổng số tham số là M\*N\*K\*L. Ngoài ta còn các tham số bias nên tổng số tham số của Con2d là: (M\*N\*L+1)\*K
* Pooling: do chỉ có tác dụng làm giảm kích thước và giữ đặc trưng ảnh nên nó không có tham số.
* FC (fully connected): với số lượng đầu vào là N, đầu ra là M ta có tổng số weight là M\*N. Cộng thêm với các tham số bias thì tổng sẽ là (N+1)\*M
* Flatten: chỉ có tác dụng làm ‘phẳng’ dữ liệu nên không có tham số nào cả.
* Log\_softmax: có tác dụng trả về kết quả, không có tham số nào cả.

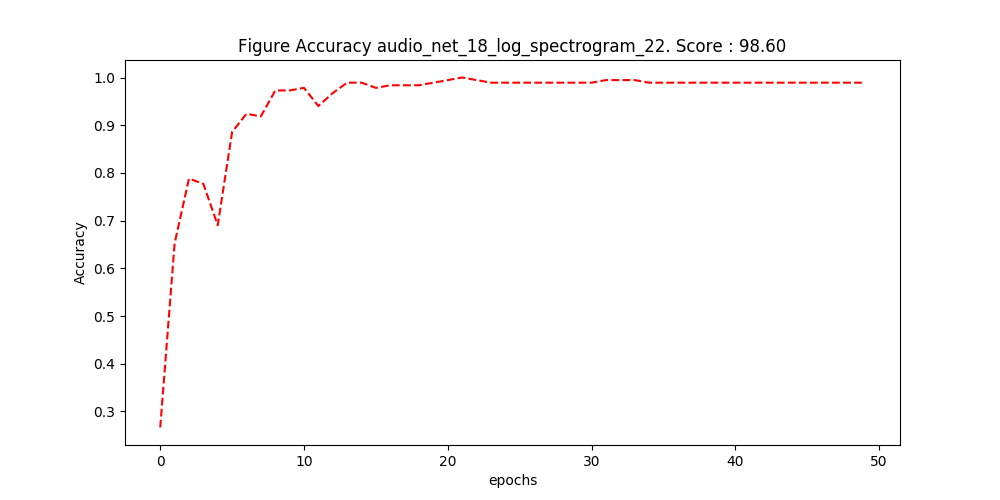
|  |  |  |
| --- | --- | --- |
|  | Kích thước bộ lọc | Tổng tham số |
| Input [257x29x1] | Không có | 0 |
| Conv2d(1, 64) | (7x7) | 3.137 |
| Conv2d(64, 256) | (5x5) | 409.664 |
| Conv2d(256, 512) | (3x3) | 1.179.905 |
| Conv2d(512, 256) | (3x3) | 1.180.160 |
| Con2d(256, 256) | (3x3) | 590.080 |
| FC(9216,1000) | Không có | 9.225.216 |
| FC(1000, 100) | Không có | 101.000 |
| FC(100, 10) | Không có | 1.100 |
|  |  | Tổng: 12.687.262 |

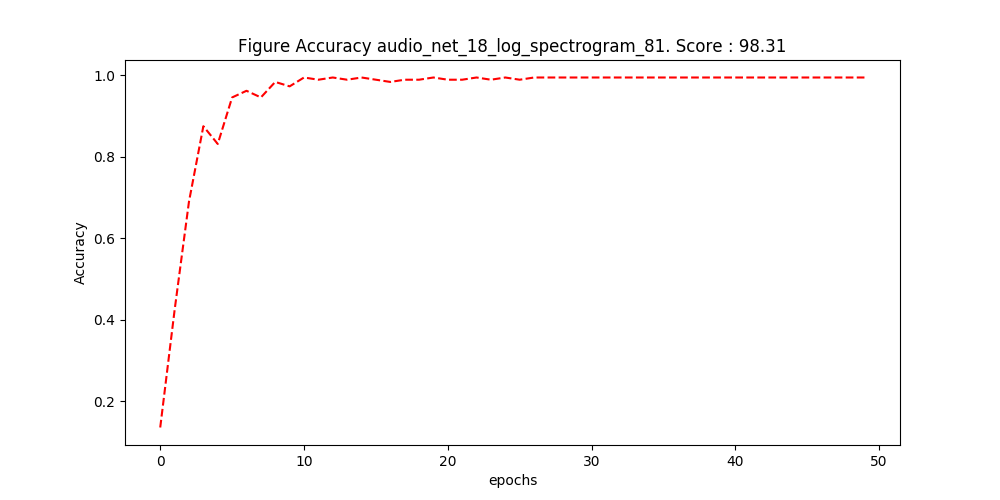
# TIẾN HÀNH HUẤN LUYỆN VÀ CÁC KẾT QUẢ ĐẠT ĐƯỢC

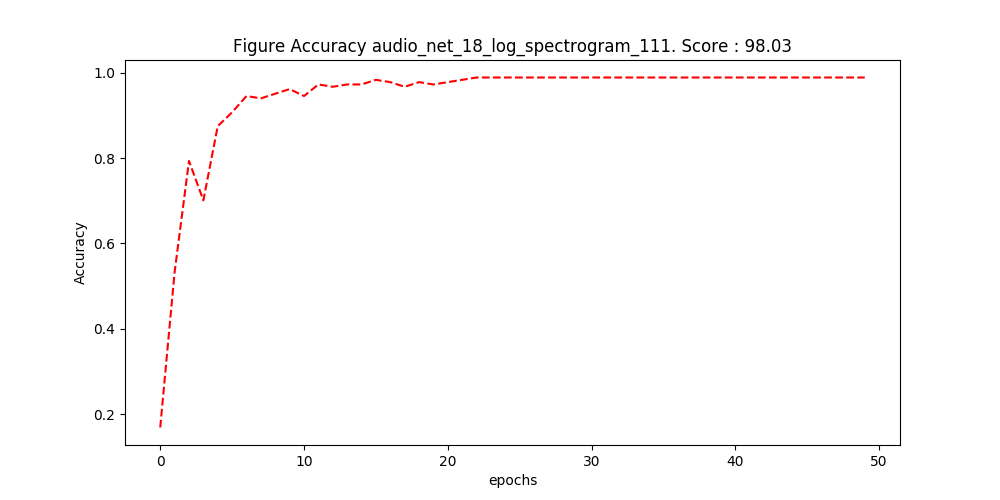
## Cách huấn luyện

Tiến hành huấn luyện mô hình mạng đã được xây dựng như ở chương 4 bằng cách: khởi tạo 10 mô hình, huấn luyện riêng 10 mô hình với các **random seed** khác nhau. Tiến hành đánh giá thông qua tập Test và lấy mô hình có Accuracy cao nhất trên tập Test.

## Biểu đồ accuracy của một số mô hình trên tập Validation

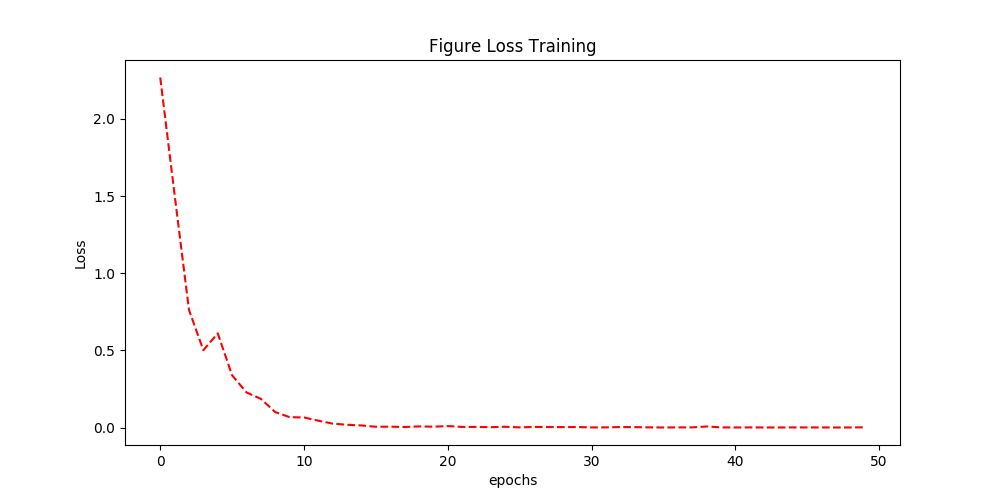


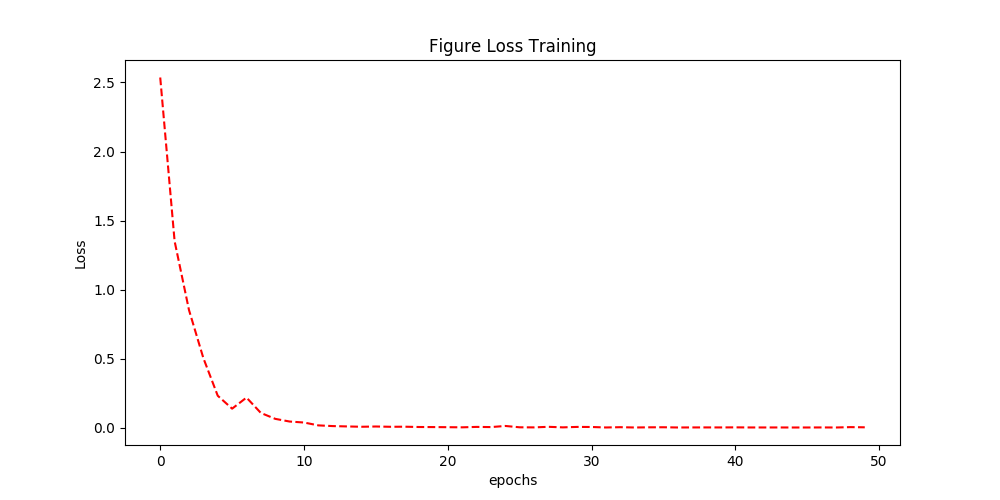


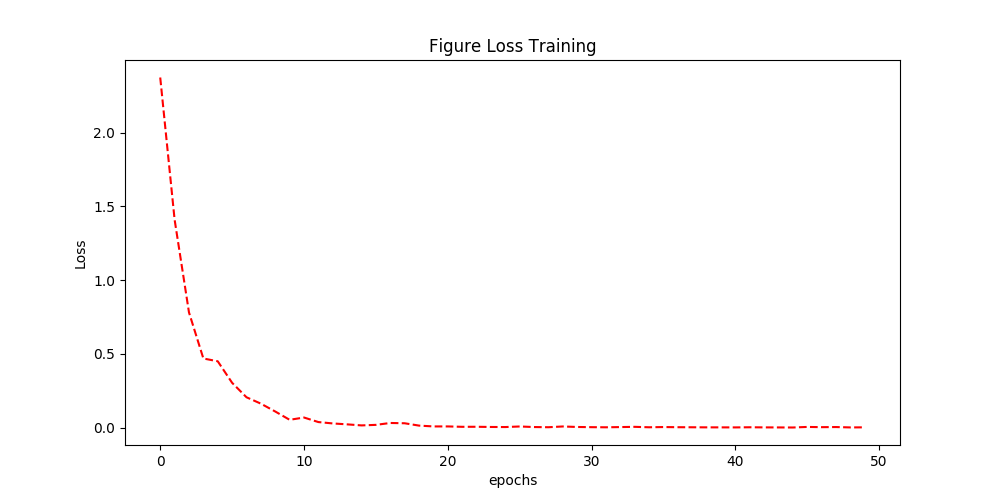


Hình . Biểu đồ Accuracy của một số mô hình trên tập Validation

## Biểu đồ hàm loss của một số mô hình trên tập Train

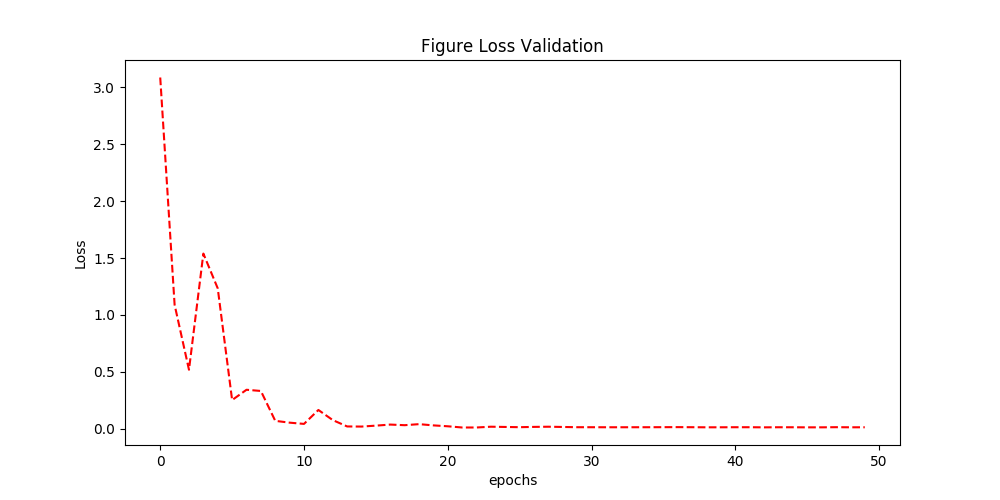


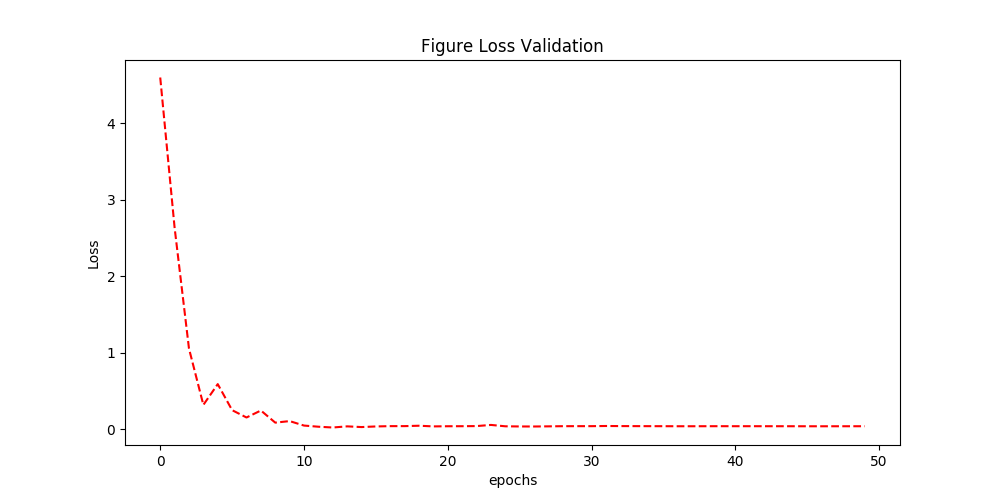


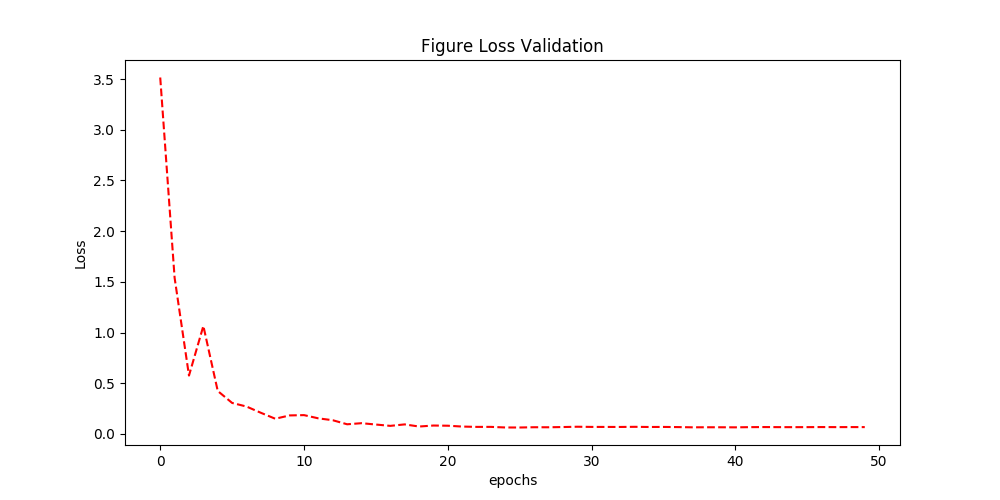


Hình . Biểu đồ hàm loss trên tập Train

## Biểu đồ hàm loss trên tập Validation một vài mô hình



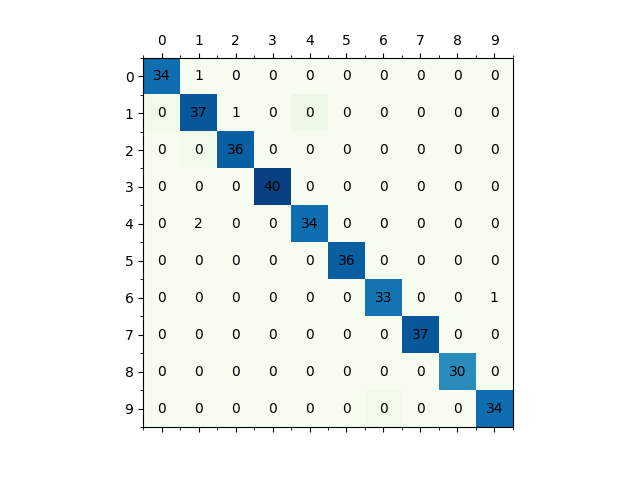


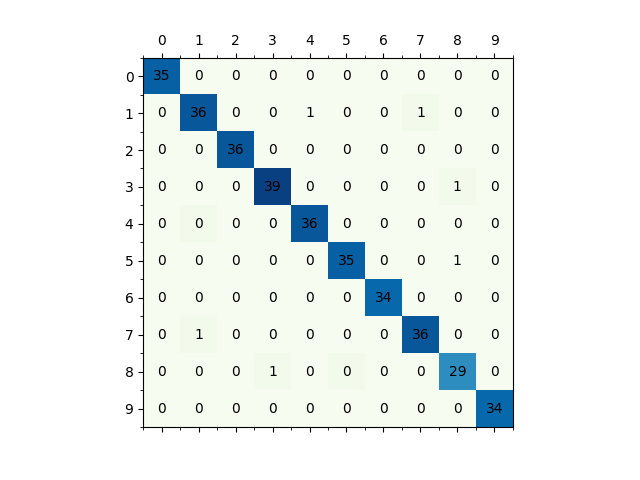


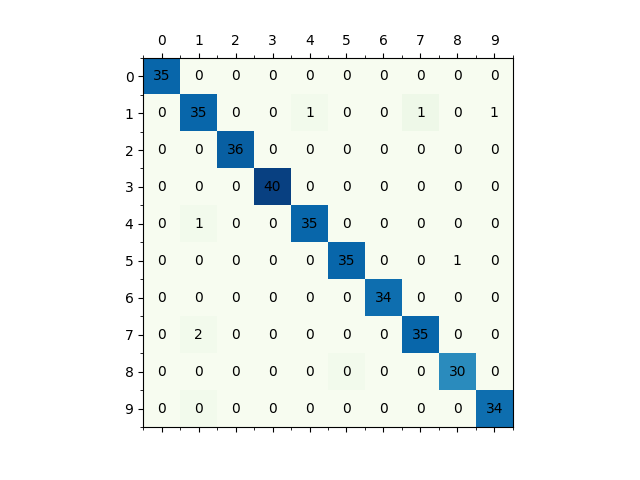
Hình . Biểu đồ hàm loss một số mô hình trên tập Validation

## Ma trận lỗi của một số mô hình trên tập Test

Ta sử dụng ma trận lỗi C với kích thước 10x10 là ma trận chỉ ra tổng số mẫu nhãn . Từ đó chúng ta có thể đưa ra những phương pháp nhằm cải thiện dữ liệu đầu vào và tăng độ chính xác cho mô hình. Tại các trị trong C[i, j] với i=j thì sẽ là số mẫu đúng với lớp của nó, còn với i khác j thì giá trị tại C[i, j] sẽ thể hiện mẫu lớp j bị nhầm với mẫu lớp i.







Hình . Ma trận lỗi của một số mô hình huấn luyện

Ta thấy trong ma trận thứ 3, ô C[7, 1] có giá trị 2. Điều này chứng tỏ một số mẫu lớp 1 bị nhầm với lớp số 7.

## Bảng số liệu

Bảng Accuracy trên tập Validation và tập Test với 10 mô hình với các random seed khác nhau:

|  |  |  |
| --- | --- | --- |
| Mô hình | Validation (%) | Test (%) |
| 1 | 97,28 | 97,19 |
| 2 | 100 | 99,44 |
| 3 | 100 | 99,16 |
| 4 | 99,91 | 97,95 |
| 5 | 99,46 | 98,31 |
| 6 | 98,91 | 98,03 |
| 7 | 99,91 | 97,19 |
| 8 | 98,91 | 97,47 |
| 9 | 98,91 | 98,60 |
| 10 | 100 | 98,60 |
| MAX | **100** | **99,44** |
| MEAN | **99,14** | **97,29** |

Bảng . Bảng đánh giá mô hình trên tập Validation và Test

# KẾT LUẬN

## Kết luận

Kết quả: xét về tổng thể đồ án đã hoàn thành được những mục tiêu đề ra bao gồm:

* Tiến hành thu âm tổng cộng 1206 file âm thanh của các số đếm tiếng Việt từ không đến chín.
* Xử lý tín hiệu tiếng nói, trích trọn đặc trưng Log Spectrogram để đưa vào mạng nơ-ron huấn luyện.
* Xây dựng mô hình mạng nơ-ron huấn luyện giải quyết bài toán nhận dạng tiếng nói.

Tồn tại: Việc thu âm các mẫu ở điều kiện khá lý tưởng, ít tạp âm, ít khoảng lặng. Việc lấy mẫu như này sẽ khiến mô hình huấn luyện có thể học dễ hơn. Nhưng khi ra ngoài thực tế thì sẽ không đạt được hiệu quả cao như hiện nay.

## Hướng phát triển của đồ án trong tương lai

Bài toán nhận dạng tiếng nói có rất nhiều mức độ, đề tại thực hiện được coi là bài toán có thể coi là đơn gỉản. Hướng phát triển tiếp theo sẽ nâng cao mức độ của bài toán nhận dạng tiếng nó để có thể áp dụng vào thực tế, từ đó đưa vào xây dựng các hệ thống liên quan đến lĩnh vực xử lý tiêng nói. Các mức độ tiếp theo:

* Nhận dạng nhiều từ riêng lẻ hơn (vài nghìn từ) , một người nói.
* Nhận dạng nhiều từ riêng lẻ (vài nghìn từ), một người nói.
* Nhận dạng nhiều từ lẻ nhưng cho nhiều nhiều nói.
* Nhận dạng câu văn ngắn, từ vựng hạn chế, một người nói.
* Nhận dạng câu văn ngắn, từ ngữ hạn chế nhiều người nói.
* Nhận dạng lời nói liên tục, một hoặc nhiều người nói.

# TÀI LIỆU THAM KHẢO

|  |  |
| --- | --- |
| [1] | O. R. Tom Bäckström, "https://wiki.aalto.fi/," 04 10 2019. [Online]. Available: https://wiki.aalto.fi/display/ITSP/Introduction+to+Speech+Processing. |
| [2] | P. T. T. V. Loan, Tài liệu xử lý tiếng nói, 2013. |
| [3] | karpathy@cs.stanford.edu, "http://cs231n.github.io," [Online]. Available: http://cs231n.github.io/convolutional-networks/. |