** HỌC VIỆN CÔNG NGHỆ BƯU CHÍNH VIỄN THÔNG KHOA CÔNG NGHỆ THÔNG TIN 1**

**BỘ MÔN NHẬP MÔN TRÍ TUỆ NHÂN TẠO**

**o0o**

****

**BÀI TẬP LỚN**

**Đề tài: Nhận dạng giọng nói bằng phương**

**pháp học sâu**

|  |  |
| --- | --- |
| Nhóm học phần | : N05 |
| Nhóm BTL | : 18 |

|  |  |
| --- | --- |
| **Đặng Tiến Dũng** | **MSV: B21DCCN263** |
| **Mai Văn Lịch** | **MSV: B21DCCN480** |
| **Vũ Thành Luân**  **Chu Đức Nam** | **MSV: B21DCCN502**  **MSV: B21DCCN538** |
| **Vũ Xuân Thịnh** | **MSV: B21DCCN694** |

*Hà Nội – 2024*

MỤC LỤC

[**I.** **Giới thiệu** 5](#_Toc167146679)

[**1. Đặt vấn đề** 5](#_Toc167146680)

[**2. Mục đích** 5](#_Toc167146681)

[**3. Phạm vi bài toán** 6](#_Toc167146682)

[**II.** **Cơ sở lý thuyết** 7](#_Toc167146683)

[**1.** **Âm thanh và biến đổi** 7](#_Toc167146684)

[**1.1 Biến đổi Fourier** 7](#_Toc167146685)

[**1.2. Spectrogram** 7](#_Toc167146686)

[**1.3.Chuẩn hóa Z-Score** 8](#_Toc167146687)

[**2.** **Mô hình CRNN** 9](#_Toc167146688)

[**2.1. Mạng RNN** 10](#_Toc167146689)

[**2.2. Hàm kích hoạt RELU** 10](#_Toc167146690)

[**2.3.** **Hàm kích hoạt tanh** 11](#_Toc167146691)

[**2.4. Hàm kích hoạt Sigmoid** 12](#_Toc167146692)

[**2.5. Hàm mất mát CTC** 13](#_Toc167146693)

[**2.6.** **Batch Normalization** 13](#_Toc167146694)

[**3.** **Thư viện Keras** 14](#_Toc167146695)

[**4.** **Ứng dụng xây model** 15](#_Toc167146696)

[**5. Phương pháp đánh giá** 17](#_Toc167146697)

[**III.** **Thực nghiệm và kết quả** 18](#_Toc167146698)

[**1.**  **Trình bày về dữ liệu** 18](#_Toc167146699)

[**1.1 Tập âm thanh ngắn .wav** 18](#_Toc167146700)

[**1.2. File metadata.csv** 18](#_Toc167146701)

[**1.3. Dữ liệu huấn luyện** 19](#_Toc167146702)

[**2.** **Thực hiện** 19](#_Toc167146703)

[**3.** **Kết quả** 24](#_Toc167146704)

[**4.** **Đánh giá và thảo luận** 25](#_Toc167146705)

[**5.** **Website thử nghiệm model** 27](#_Toc167146706)

[**IV.** **Kết luận** 28](#_Toc167146707)

[**V.** **Tài liệu tham khảo** 30](#_Toc167146708)

## 

## **Mục Lục Hình Ảnh**

[Hình 1 : Miên thời gian sang miền tần số 7](#_Toc167147503)

[Hình 2: Hình ảnh Spetrogram 8](#_Toc167147505)

[Hình 3: Hình ảnh Spetrogram chuẩn hóa 9](#_Toc167147507)

[Hình 4: Mô hình CNN 9](#_Toc167147508)

[Hình 5: Mô hình RNN 10](#_Toc167147510)

[Hình 6: Đồ thị ReLU 11](#_Toc167147512)

[Hình 7: Đồ thị Tanh 12](#_Toc167147514)

[Hình 8: Đồ thị Sigmoid 12](#_Toc167147516)

[Hình 9: Model 15](#_Toc167147519)

[Hình 10: Các file .wav 18](#_Toc167147521)

[Hình 11: File metadata.csv 18](#_Toc167147523)

[Hình 12: Số lượng dữ liệu huấn luyện và đánh giá 19](#_Toc167147525)

[Hình 13: Quá trình thực hiện 19](#_Toc167147526)

[Hình 14: Chuyển âm thanh sang dạng 1 chiều 20](#_Toc167147527)

[Hình 15: Biến đổi STFT 20](#_Toc167147528)

[Hình 16: Tensor spetrogram 21](#_Toc167147529)

[Hình 17: Tensor spetrogram sau khi chuẩn hóa 21](#_Toc167147530)

[Hình 18: Chuyển đổi label sang dạng số 22](#_Toc167147531)

[Hình 19: Cấu trúc mô hình 23](#_Toc167147532)

[Hình 20: Số lượng tham số mô hình 24](#_Toc167147533)

[Hình 21: Huấn luyện với 10 epochs 24](#_Toc167147534)

[Hình 22: Huấn luyện với 25 epochs 25](#_Toc167147535)

[Hình 23: Giá trị loss và val\_loss trong 10 epochs đầu tiên 26](#_Toc167147536)

[Hình 24: Giá trị loss và val\_loss trong 14-22 epochs 26](#_Toc167147537)

[Hình 25: Giá trị wer\_Score trong khoảng 25 epochs 27](#_Toc167147538)

# **Giới thiệu**

## **1. Đặt vấn đề**

- Nhận dạng giọng nói, còn được gọi là nhận dạng giọng nói tự động (ASR), là công nghệ cho phép máy diễn giải và hiểu lời nói của con người. Công nghệ này cung cấp một cách thuận tiện và hiệu quả hơn để người dùng tương tác với công nghệ. Thay vì gõ trên bàn phím hoặc màn hình cảm ứng, người dùng có thể chỉ cần nói lệnh hoặc nhập thông tin, thực hiện các tác vụ như gửi tin nhắn, tìm kiếm trên web hoặc điều khiển thiết bị rảnh tay và dễ dàng hơn.Nhận dạng giọng nói cũng đóng một vai trò quan trọng trong khả năng tiếp cận, cho phép những người khuyết tật hoặc khiếm khuyết ảnh hưởng đến khả năng nhập hoặc sử dụng các phương thức nhập liệu truyền thống để giao tiếp và truy cập thông tin dễ dàng hơn. Nhìn chung, công nghệ nhận dạng giọng nói có tiềm năng nâng cao trải nghiệm người dùng, tăng hiệu quả và cải thiện khả năng tiếp cận trong các khía cạnh khác nhau của cuộc sống hàng ngày của chúng ta.

- Ngày nay, công nghệ nhận dạng giọng nói phát triển nhanh chóng, được thúc đẩy bởi những tiến bộ trong trí tuệ nhân tạo, học sâu, xử lý ngôn ngữ tự nhiên và dữ liệu lớn. . Tuy nhiên, để tận dụng được tiềm năng của công nghệ này một cách tối đa, chúng ta cần hiểu rõ về cách thức  hoạt động công nghệ này.

- Trong bài báo cáo này, chúng tôi sẽ nghiên cứu về quy trình xử lý âm thanh và mô hình nhận dạng tiếng nói qua các bước: tiền xử lý âm thanh, xây dựng mô hình bằng RNN và huấn luyện mô hinh bằng hàm CTC Loss .

## **2. Mục đích**

**Mục tiêu của đề tài chuyển nhận diện giọng nói thành văn bản là:**

- Xây dựng mô hình: Sử dụng mạng nơ-ron tích chập (CNN) để trích xuất đặc trưng từ phổ tần số của âm thanh, sử dụng các lớp mạng nơ-ron hồi quy (RNN) để xử lý chuỗi đặc trưng và dự đoán văn bản tương ứng.

- Huấn luyện mô hình: Huấn luyện mô hình trên tập dữ liệu huấn luyện, sử dụng hàm mất mát CTC (Connectionist Temporal Classification) để xử lý các chuỗi đầu ra có độ dài khác nhau.

- Đánh giá hiệu suất: Đo lường độ chính xác của mô hình bằng chỉ số WER rồi in ra một số kết quả dự đoán so với văn bản gốc để kiểm tra trực quan.

## **3. Phạm vi bài toán**

- Phạm vi bài toán “Nhận diện âm thanh sử dụng mạng nơ-ron RNN”:

* Âm thanh: bài toán tập trung vào nhận diện các lời nói bằng Tiếng Anh.
* Dữ liệu vào: Thực hiện xử lý các file âm thanh để làm dữ liệu đầu vào, các file âm thanh được tiền xử lý trước khi đưa vào mô hình CNN.
* Mạng nơ-ron RNN: thiết kế và xây dựng một mạng RNN để nhận diện âm thanh. Ngoài kiến trúc mạng RNN, mô hình còn bao gồm các lớp tích chập (Convolutional Layer), Full Connected Layer và Softmax Layer.
* Connectionist Temporal Classification (CTC): dùng để thực hiện training và trả về kết quả đúng nhất.

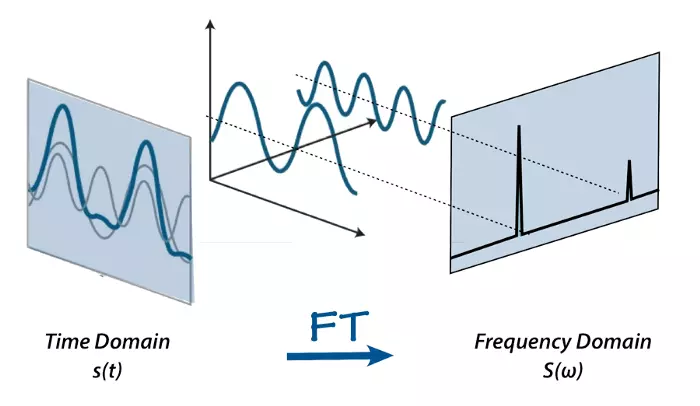
- Nhận dạng giọng nói có thể sử dụng nhiều phương pháp: HMMs, NLP, Speaker Diarization, DTW,… Tuy nhiên, phương pháp mạng nơ-ron với khả năng xử lý và biến đổi dữ liệu đầu vào bằng cách mô phỏng nhận thức tần số phi tuyến tính của hệ thống thính giác của con người được lựa chọn. Đồng thời, còn sử dụng thêm CTC giúp cho phép mạng nơ-ron trích xuất được đặc trưng của dữ liệu đầu vào và căn chỉnh để đạt được kết quả đúng của đầu ra.

# **Cơ sở lý thuyết**

## **Âm thanh và biến đổi**

### **1.1 Biến đổi Fourier**

- Với phép biến đổi Fourier, chúng ta chuyển đổi một tín hiệu từ miền thời gian sang miền tần số. Biến đổi Fourier không chỉ cung cấp các tần số có trong tín hiệu mà còn cung cấp độ lớn của mỗi tần số có trong tín hiệu.



Hình 1 : Miên thời gian sang miền tần số

### **1.2. Spectrogram**

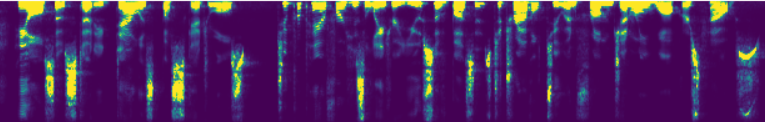
- Trong phần trước, chúng ta đã chia tín hiệu thành các giá trị tần số của nó, chúng sẽ đóng vai trò là features cho mạng nơ ron nhận dạng giọng nói. Nhưng khi áp dụng FFT cho tín hiệu của mình, nó chỉ cung cấp các giá trị tần số và chúng ta bị mất dấu thông tin thời gian. Do đó, chúng ta cần tìm một cách khác để tính toán các features sao cho các giá trị tần số và thời gian đều được quan sát.

- Spectrogram có thể giải quyết được vấn đề này.  
 - Biểu diễn trực quan các tần số của một tín hiệu nhất định với thời gian được gọi là Spectrogram.

**Cách tạo ra Spectrogram:**

- Khi sử dụng phép biến đổi Fourier rất nhanh (STFT - Short-Time Fourier Transform), bạn sẽ thu được một biểu đồ được gọi là "spectrogram".

STFT viết tắt của Short-Time Fourier Transform.  STFT là một kỹ thuật được sử dụng để phân tích nội dung tần số của một tín hiệu khi nó thay đổi theo thời gian. Kỹ thuật này thực hiện bằng cách lấy biến đổi Fourier của các phần nhỏ của tín hiệu tại các khoảng thời gian khác nhau



Hình 2: Hình ảnh Spetrogram

- Đây là Spectrogram. Rõ ràng, chúng ta không thể thấy rõ được các thông tin về tần số, biên độ mà Spectrogram thể hiện. Điều này được giải thích là do khả năng nhận thức âm thanh của con người. Hầu hết những âm thanh mà chúng ta nghe được đều tập trung xung quanh một dải tần số và biên độ khá hẹp. Vì vậy, trong nhiều bài toán (đặc biệt là nhận dạng giọng nói), Spectrogram không phải là sự lựa chọn hoàn hảo. Vì vậy ta cần thêm vài bước tính nữa để thu được

### **1.3.Chuẩn hóa Z-Score**

- Chuẩn hóa Z-score là một phương pháp quan trọng trong thống kê, giúp chúng ta hiểu và phân tích dữ liệu một cách hiệu quả. Đây là cách chuyển đổi dữ liệu sao cho chúng có mean (trung bình) bằng 0 và độ lệch chuẩn (standard deviation) bằng 1. Việc này giúp loại bỏ sự biến động tỷ lệ và giúp dễ dàng so sánh giữa các đặc trưng trong dữ liệu. Điều này có thể giúp chúng ta phát hiện xu hướng, biến thể, hoặc ngoại lệ trong dữ liệu một cách hiệu quả. Chuẩn hóa Z-score cũng hỗ trợ trong việc tính toán tương quan và giảm thiểu ảnh hưởng của biến động tự nhiên, từ đó tăng cường khả năng phân tích.

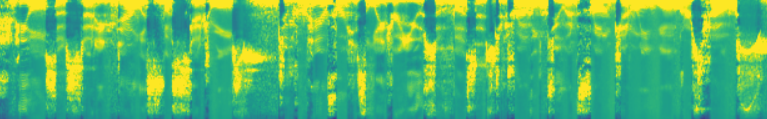
- Công thức:

Trong đó: z là giá trị Z-score.

x là giá trị thô (raw value) của điểm dữ liệu.

μ là giá trị trung bình của dữ liệu.

σ là độ lệch chuẩn của dữ liệu.



Hình 3: Hình ảnh Spetrogram chuẩn hóa

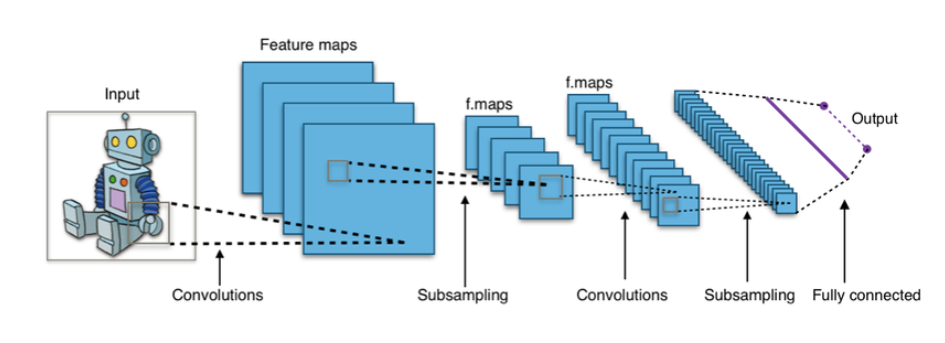
## **Mô hình CRNN**

- CNN là kiến trúc lý tưởng khi giải quyết vấn đề dữ liệu hình ảnh, một trong những mô hình Deep Learning tiên tiến. Nó giúp cho chúng ta xây dựng được những hệ thống thông minh với độ chính xác cao như hiện nay.

**Kiến trúc:**

- Mạng CNN là một tập hợp các lớp tích chập chồng lên nhau và sử dụng các hàm nonlinear activation như ReLU và tanh để kích hoạt các trọng số trong các node. Mỗi một lớp sau khi thông qua các hàm kích hoạt sẽ tạo ra các thông tin trừu tượng hơn cho các lớp tiếp theo.

- Mỗi một lớp sau khi thông qua các hàm kích hoạt sẽ tạo ra các thông tin trừu tượng hơn cho các lớp tiếp theo. Trong mô hình CNN, các lớp liên kết được với nhau thông qua cơ chế tích chập. Lớp tiếp theo là kết quả phép tính tích chập từ lớp trước đó, nhờ vậy mà ta có được các kết nối cục bộ. Như vậy mỗi neuron ở lớp kế tiếp sinh ra từ kết quả của filter áp đặt lên một vùng ảnh cục bộ của neuron trước đó. Mỗi một lớp được sử dụng các filter khác nhau thông thường có hàng trăm hàng nghìn filter như vậy và kết hợp kết quả của chúng lại. Ngoài ra có một số lớp khác như pooling/subsampling lớp dùng để chắt lọc lại các thông tin hữu ích hơn. Trong quá trình huấn luyện mạng (traning) CNN tự động học các giá trị qua các lớp filter



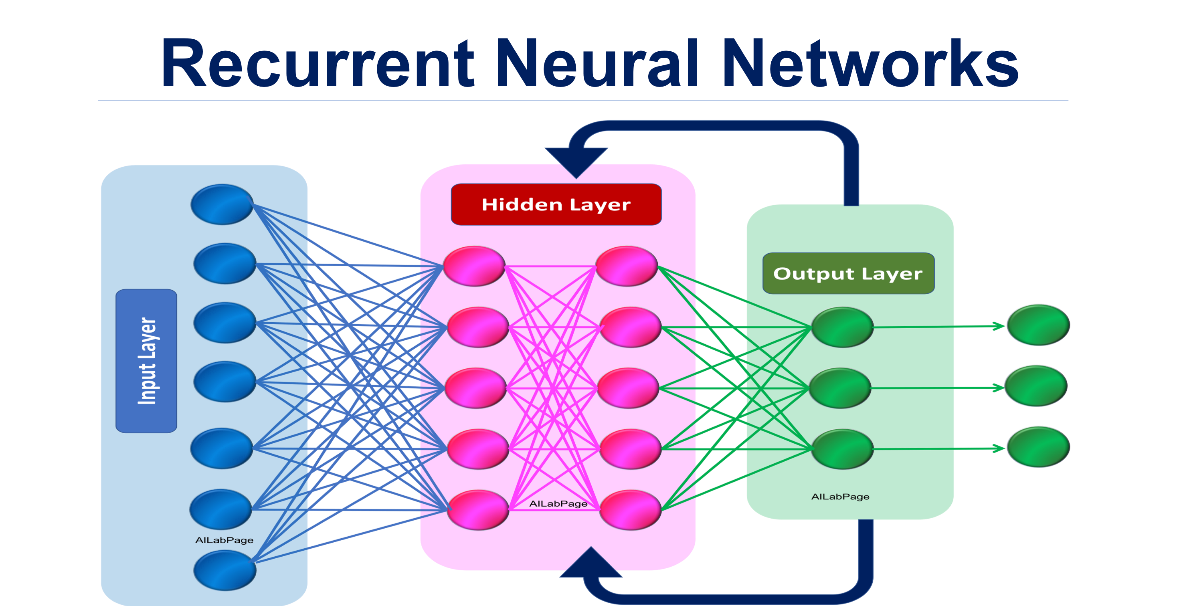
Hình 4: Mô hình CNN

### **. Mạng RNN**

- RNN là viết tắt của “Recurrent Neural Network” có thể dịch là “Mạng Nơ-ron Hồi Quy”. đây là một loại kiến trúc mạng nơ-ron trong lĩnh vực trí tuệ nhân tạo AI và học máy. RNN được thiết kế để xử lý dữ liệu chuỗi, nơi thông tin từ các bước thời gian trước đó được giữ lại để ảnh hưởng đến các bước thời gian sau.

- RNN được tạo thành từ các nơ-ron: các nút xử lý dữ liệu kết hợp cùng nhau để thực hiện các tác vụ phức tạp. Các nơ-ron được tổ chức dưới dạng lớp đầu vào, đầu ra và ẩn. Lớp đầu vào nhận thông tin để xử lý và lớp đầu ra cung cấp kết quả. Quá trình xử lý dữ liệu, phân tích và dự đoán diễn ra trong lớp ẩn.

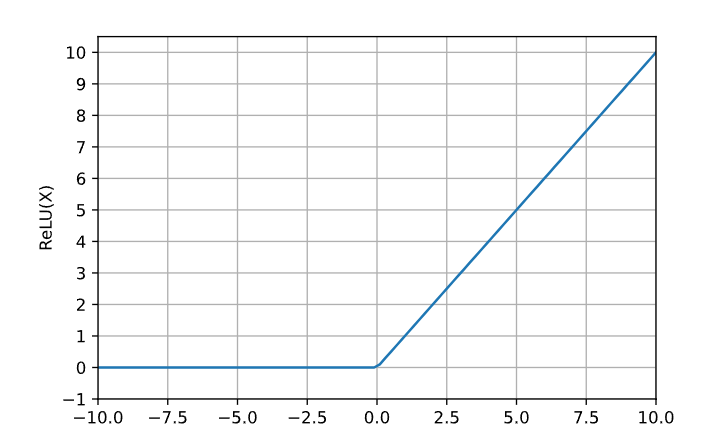
- Lớp ẩn của RNN chứa các trọng số (weights) và các tham số có thể được điều chỉnh trong quá trình huấn luyện để mô hình có thể học và bảo toàn thông tin quan trọng từ các bước thời gian trước đó. Tuy nhiên, trong RNN truyền thống, có một vấn đề là vanishing gradient problem, khiến cho mô hình khó học được thông tin từ những bước thời gian xa. Để giải quyết vấn đề này, các biến thể như LSTM và GRU được phát triển với các cơ chế kiểm soát thông tin tốt hơn.



Hình 5: Mô hình RNN

### **2.2 Hàm kích hoạt RELU**

- Hàm kích hoạt ReLU (Rectified Linear Activation) là một hàm kích hoạt phổ biến trong mạng nơ-ron và các mô hình học sâu. Được định nghĩa bởi f(x) = max(0, x), nơi x là đầu vào của hàm. Đơn giản là, nếu đầu vào là âm, nó sẽ trả về 0; nếu đầu vào là không âm, nó sẽ trả về giá trị đó. Hàm ReLU giúp giảm vấn đề biến mất đạo hàm và tăng tốc độ hội tụ trong quá trình huấn luyện mạng nơ-ron.



Hình 6: Đồ thị ReLU

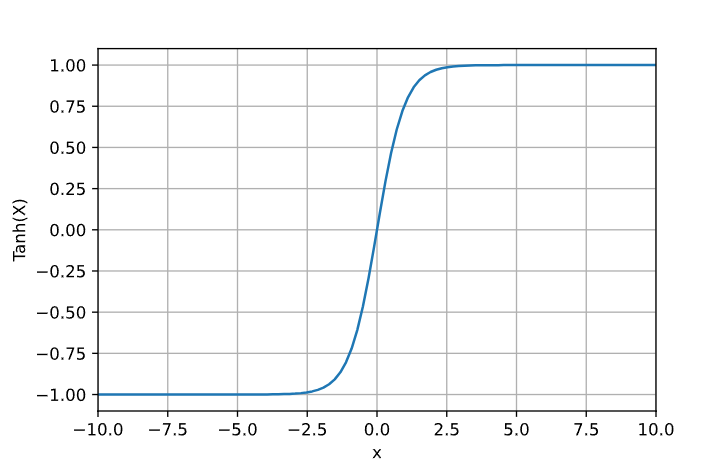
- Hàm ReLU có những ưu và nhược điểm:

+ Ưu điểm: Tốc độ hội tụ nhanh, tính toán nhanh vì đã loại bỏ hết giá trị âm

+ Nhược điểm: Với các node có giá trị nhỏ hơn 0, qua ReLU activation sẽ thành 0, hiện tượng đấy gọi là “Dying ReLU”. Nếu các node bị chuyển thành 0 thì sẽ không có ý nghĩa với bước linear activation ở lớp tiếp theo và các hệ số tương ứng từ node đấy cũng không được cập nhật với gradient descent.

### **Hàm kích hoạt tanh**

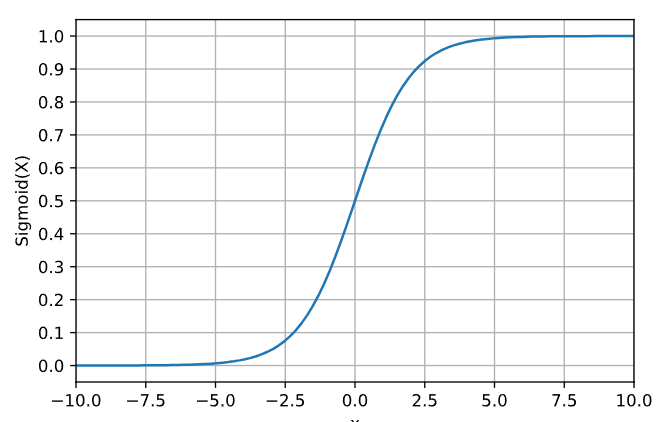
- Hàm tanh nhận đầu vào là một số thực và chuyển thành một giá trị trong khoảng (-1; 1). Cũng như Sigmoid, hàm Tanh bị bão hoà ở 2 đầu (gradient thay đổi rất ít ở 2 đầu). Tuy nhiên hàm Tanh lại đối xứng qua 0 nên khắc phục được một nhược điểm của Sigmoid.



Hình 7: Đồ thị Tanh

### **. Hàm kích hoạt Sigmoid**

- Hàm Sigmoid nhận đầu vào là một số thực và chuyển thành một giá trị trong khoảng (0;1). Đầu vào là số thực âm rất nhỏ sẽ cho đầu ra tiệm cận với 0, ngược lại, nếu đầu vào là một số thực dương lớn sẽ cho đầu ra là một số tiệm cận với 1. Nhược điểm: Hàm Sigmoid bão hòa và triệt tiêu gradient, không có trung tâm là 0 gây khó khăn trong hội tụ.



Hình 8: Đồ thị Sigmoid

### **Hàm mất mát CTC**

- Hàm CTC (Connectionist Temporal Classification) loss là một hàm mất mát được thiết kế đặc biệt cho các tác vụ nhận dạng chuỗi như nhận dạng giọng nói, nơi mà các dự đoán có thể có độ dài thay đổi và không được căn chỉnh chính xác với các đầu vào.

- Các thành phần:

+ Ký tự trống: Được dự đoán khi không có ký tự cụ thể nào được phát ra.

+ Ký tự lặp lại: CTC có cơ chế hợp nhất các ký tự lặp lại thành một ký tự duy nhất khi xuất hiện liên tiếp

* Cơ chế hoạt động:

+ Căn chỉnh tất cả các khả năng: CTC tính toán tất cả các căn chỉnh có thể giữa chuỗi đầu vào và chuỗi đầu ra.

+ Tổng hợp xác suất: Tính tổng xác suất của tất cả các căn chỉnh hợp lệ dẫn đến chuỗi đầu ra mong muốn.

+ Đào tạo mô hình: Mô hình được đào tạo để tối đa hóa xác suất của chuỗi đầu ra đúng.

* Quá trình tính toán:

+ Forward-backward algorithm: Sử dụng thuật toán forward-backward để tính toán xác suất của tất cả các căn chỉnh khả thi một cách hiệu quả.

+ Biến thể dynamic programming: Tối ưu hóa quá trình tính toán xác suất tổng thể bằng cách sử dụng kỹ thuật lập trình động.

* Công thức:

+ Biểu thức tính toán CTC loss:

+ Biểu thức tính xác suất của tất cả căn chỉnh khả thi:

### **Batch Normalization**

* Batch Normalization là một kỹ thuật phổ biến trong học sâu (deep learning) nhằm cải thiện tốc độ và hiệu suất của mạng nơ-ron bằng cách chuẩn hóa đầu ra của mỗi lớp, giúp cho quá trình huấn luyện trở nên ổn định hơn
* Chuẩn hóa các giá trị đầu vào: là giá trị trung bình, là phương sai
* Chuyển đổi tuyến tính để duy trì khả năng biểu diễn mô hình
* Ưu điểm: Tăng tốc độ hội tụ, ổn định hơn, giảm overfitting.
* Nhược điểm: Phụ thuộc vào kích thước batch, độ phức tạp tính toán cao.

## **Thư viện Keras**

- Keras là một thư viện mã nguồn mở được xây dựng trên nền tảng của TensorFlow, Theano, hoặc Microsoft Cognitive Toolkit (CNTK), giúp việc xây dựng và huấn luyện các mô hình học sâu (deep learning) trở nên đơn giản và thuận tiện hơn.

- Trong dự án của tôi Keras sử dụng các lớp sau:

+ Conv2D: Lớp này thực hiện các phép tích chập 2D trên dữ liệu đầu vào, được sử dụng để trích xuất đặc trưng từ spetrogram.

* filters: Số lượng bộ lọc, biểu thị số lượng đặc trưng cần trích xuất.
* kernel\_size: Kích thước của kernel, xác định khu vực cửa sổ được sử dụng cho phép tích chập.
* strides: Bước nhảy của kernel khi di chuyển qua dữ liệu đầu vào.
* padding: Cách thức xử lý việc đệm cho đầu vào, "same" hoặc "valid".
* use\_bias: Xác định xem một tham số bias sẽ được sử dụng trong lớp hay không.

+ BatchNormalization: Lớp này thực hiện chuẩn hóa các batch của dữ liệu đầu vào, giúp tăng tốc quá trình huấn luyện và ổn định mô hình.

+ ReLU: Hàm kích hoạt Rectified Linear Unit, thường được sử dụng để tăng tính phi tuyến tính và khả năng học của mô hình.

+ GRU (Gated Recurrent Unit): Một loại lớp mạng nơ-ron tái phát (RNN) sử dụng để mô hình hóa dữ liệu chuỗi.

* units: Số lượng đơn vị trong GRU, tức là kích thước của không gian ẩn.
* activation: Hàm kích hoạt được sử dụng cho các đơn vị, thường là "tanh".
* recurrent\_activation: Hàm kích hoạt được sử dụng cho các cổng trong GRU, thường là "sigmoid".
* return\_sequences: Xác định liệu lớp sẽ trả về chuỗi đầu ra hoặc chỉ đầu ra cuối cùng.
* reset\_after: Xác định cách thức cập nhật trạng thái ẩn, thường là "True".

+ Bidirectional: Lớp này tạo ra một mạng nơ-ron tái phát hai chiều bằng cách kết hợp hai mạng nơ-ron tái phát một chiều và ghép kết quả.

* merge\_mode: Xác định cách thức ghép kết quả từ hai mạng nơ-ron tái phát một chiều, thường là "concat" hoặc "sum".

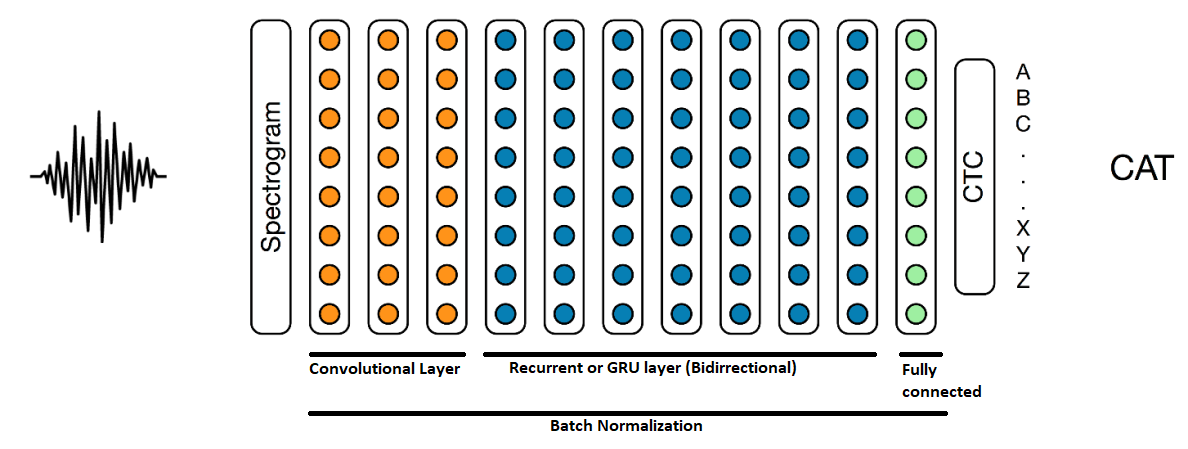
+ Dropout: Lớp này áp dụng kỹ thuật dropout để ngẫu nhiên loại bỏ một phần các đơn vị đầu ra trong quá trình huấn luyện, giúp chống lại việc quá khớp (overfitting).

+ Dense: Lớp này thực hiện phép toán fully connected, kết nối mỗi đầu vào với mỗi đầu ra trong các lớp liên kết hoàn toàn.

units: Số lượng đơn vị trong lớp, tức là kích thước của không gian đầu ra.

+ CTCLoss: Hàm mất mát Connectionist Temporal Classification (CTC), thường được sử dụng trong các vấn đề nhận dạng chuỗi.

## **Ứng dụng xây model**



Hình 9: Model

*Chú thích: input\_dim = fft\_length = 384, output\_dim= số lượng từ trong từ điển, rnn\_units= 512*

* Lớp input đầu vào:

+ Đầu vào là Spetrogram của âm thanh với kích thước (None, input\_dim). *với input\_dim là số lượng băng tần của tần số(hay chính là độ lớn trục y của spetrogram)*

* Lớp chuyển đổi chiều input

+ Thêm 1 chiều, chiều sâu của kênh để phù hợp với đầu vào cov2d

* Hai lớp mạng tích chập dùng để trích xuất thông tin

+ Hai lớp bao gồm các thuộc tính filter(số bộ lọc), kernel\_size(kích thước cửa sổ), strides(bước nhảy)…Được thiết lập tương ứng như đoạn code bên dưới.

+ Hai lớp đều xử lý đầu ra thông qua batchNormalization và hàm kích hoạt ReLU

* Lớp chuyển đổi chiều đầu ra để phù hợp với đầu vào của lớp RNN
* Lớp RNN

+ Sử dụng lớp GRU với hàm kích hoạt tanh cho đầu ra và hàm sigmoid cho trạng thái ẩn

+ Áp dụng lớp GRU theo 2 hướng (Bidirectional)

+ Tránh overfitting bằng cách thêm dropout

* Lớp kết nối đầy đủ(Dense)

+ Lớp kết nối đẩy đủ với rnn\_units \*2 đơn vị

+ Hàm kích hoạt ReLU

+ Sử dụng dropout để tránh overfitting

* Lớp phân loại

+ Lớp kết nối đầy đủ với output\_dim + 1 đơn vị. output\_dim là số lượng ký tự trong từ điển đầu ra, cộng thêm 1 để bao gồm ký tự trống cho CTC.

+ Áp dụng hàm kích hoạt softmax để chuyển đổi các giá trị đầu ra thành xác suất.

## **5. Phương pháp đánh giá**

- Mô hình sử dụng Word Error Rate (WER) làm phương pháp đánh giá chính. WER so sánh predicted output và target transcript, giữa từ với từ (hoặc giữa kí tự với kí tự) để tìm ra sự khác biệt giữa 2 câu. Sự khác biệt giữa 2 câu có thể có từ xuất hiện trong transcript nhưng không có trong prediction (được tính như là Deletion), 1 từ không có trong transcript nhưng lại được thêm vào prediction (coi như là Insertion), hoặc 1 từ được thay thế giữa prediction và transcript (xem như là Substitution).

- Ví dụ:

|  |  |
| --- | --- |
| Original Transcript | Model Prediction |
| Hello it is a great day!  it là một Deletion (bị xóa mất) | Hello is a are green day!  are là một Insertion (được thêm vào)  green là một Substitution (được thay thế từ great) |

- Công thức WER tính tỉ lệ phần số từ sự khác biệt giữa transcript và prediction.

Trong đó:  - S là số Substitution

- D là số Deletion

- I là số Insertion

- C là số từ đúng trong Prediction

- N là số lượng từ trong Transcript

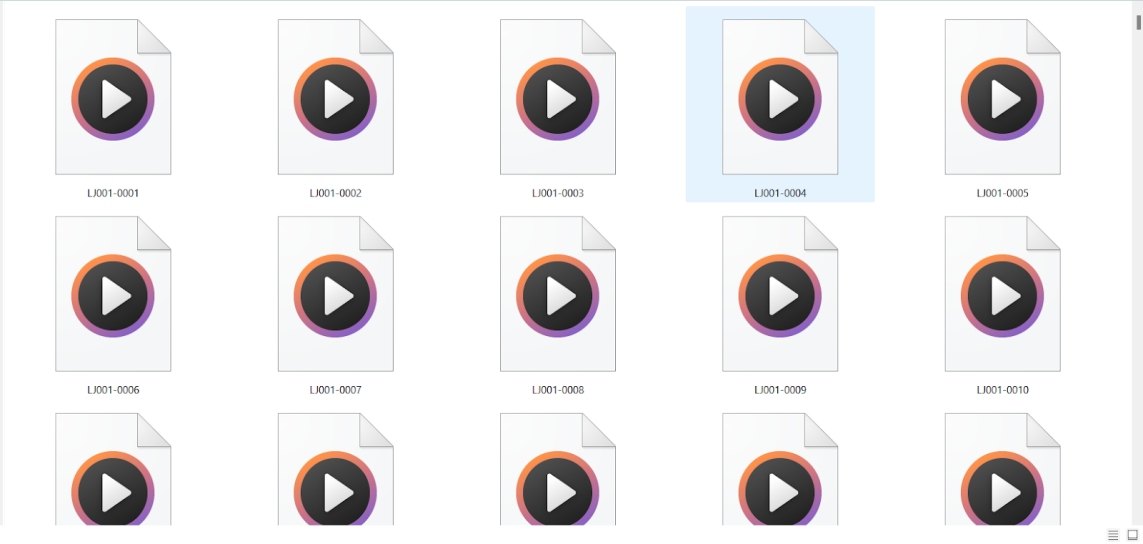
# **Thực nghiệm và kết quả**

## **Trình bày về dữ liệu**

### **1.1 Tập âm thanh ngắn .wav**

- Đây là tập dữ liệu về giọng nói trên phạm vi công cộng bao gồm 13.100 đoạn âm thanh ngắn của một người đọc đoạn văn từ 7 cuốn sách phi hư cấu. Một phiên âm được cung cấp cho mỗi clip. Các clip có độ dài khác nhau từ 1 đến 10 giây và có tổng thời lượng khoảng 24 giờ.

- Mỗi tệp âm thanh là một PCM WAV 16 bit một kênh với tốc độ mẫu là 22.050 Hz.



Hình 10: Các file .wav

### **1.2. File metadata.csv**

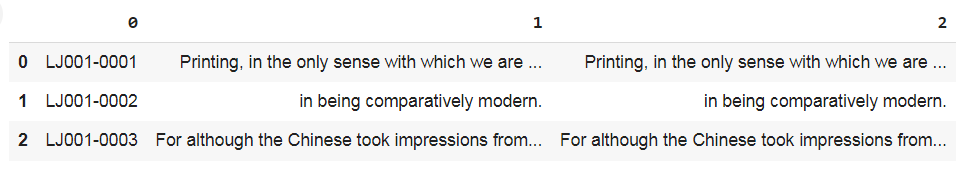
- Nhãn (bản ghi) cho mỗi tệp âm thanh .wav là một chuỗi được cung cấp trong tệp metadata.csv. Các trường là:

**(0) ID**: đây là tên của file .wav tương ứng

**(1) Phiên âm**: lời nói của người đọc (UTF-8)

**(2) Phiên âm chuẩn hóa**: phiên âm với số, thứ tự và đơn vị tiền tệ được mở rộng thành từ đầy đủ (UTF-8).

Đối với bản demo này, chúng tôi sẽ sử dụng trường "Phiên âm chuẩn hóa".



Hình 11: File metadata.csv

### **1.3. Dữ liệu huấn luyện**

Chia dữ liệu thành hai tập: Huấn luyện và kiểm tra (Training và Validation)

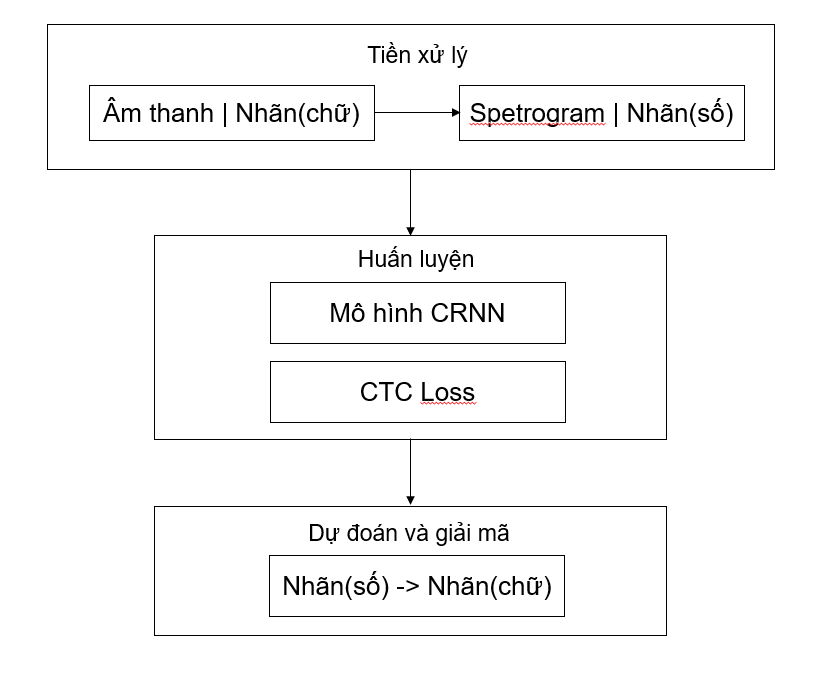
* Trainning: Chiếm 90% tổng số dữ liệu âm thanh (11790)
* Validation: Chiểm 10% tổng số dữ liệu âm thanh (1310)



Hình 12: Số lượng dữ liệu huấn luyện và đánh giá

## **Thực hiện**

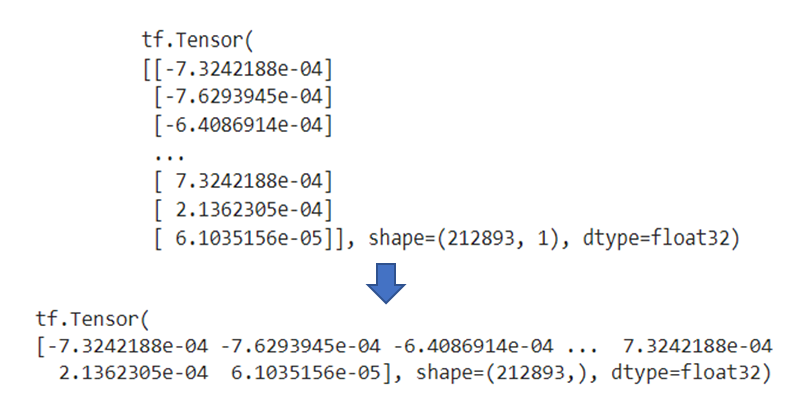
- Quá trình thực hiện:



Hình 13: Quá trình thực hiện

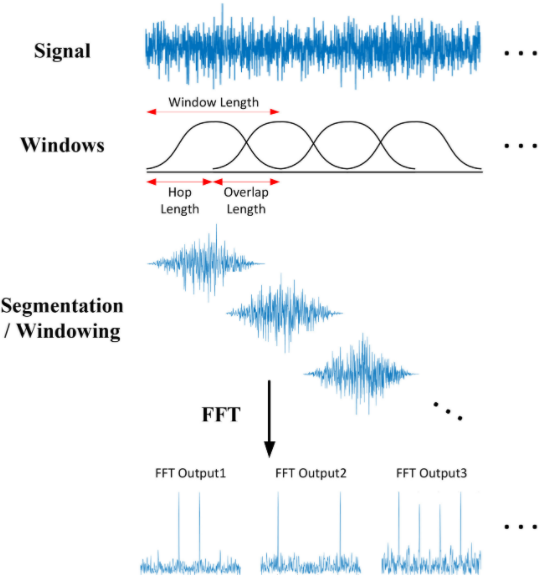
**Bước 1: Tiền xử lý dữ liệu**

- Khi âm thanh được lưu trong một tệp(cụ thể của nhóm em là file “.wav”), nó sẽ ở định dạng nén. Khi tệp được tải, nó sẽ được giải nén và chuyển đổi thành một mảng Numpy, được biểu diễn dưới dạng số thực rồi biểu diễn dưới dạng 1 chiều.



Hình 14: Chuyển âm thanh sang dạng 1 chiều

* Sau đó chúng em thực hiện phép biến đổi short-time Fourier transform (STFT) cho tín hiệu âm thanh để thu được spectrogram.



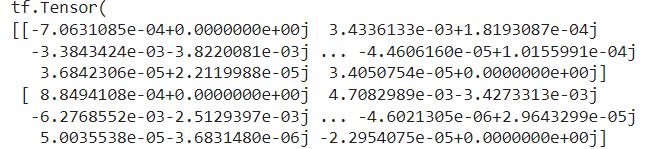
Hình 15: Biến đổi STFT

- Window Length (Độ dài cửa sổ): Đây là độ dài của cửa sổ mẫu được áp dụng cho mỗi frame tín hiệu âm thanh để tính toán biến đổi Fourier

- Hop Length (Bước nhảy): Đây là bước di chuyển giữa các frame liên tiếp trong quá trình tính toán STFT.

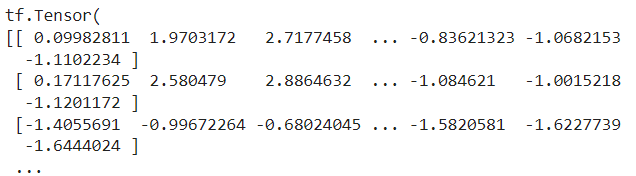
- Overlap Length (Độ dài chồng lấn): Đây là phần chồng lên giữa các frame liên tiếp khi tính toán STFT.

- Kết quả được tính toán từ tín hiệu âm thanh bằng cách sử dụng phép biến đổi Fourier ngắn thời gian (STFT) là một tensor biểu diễn một spectrogram như hình bên dưới:



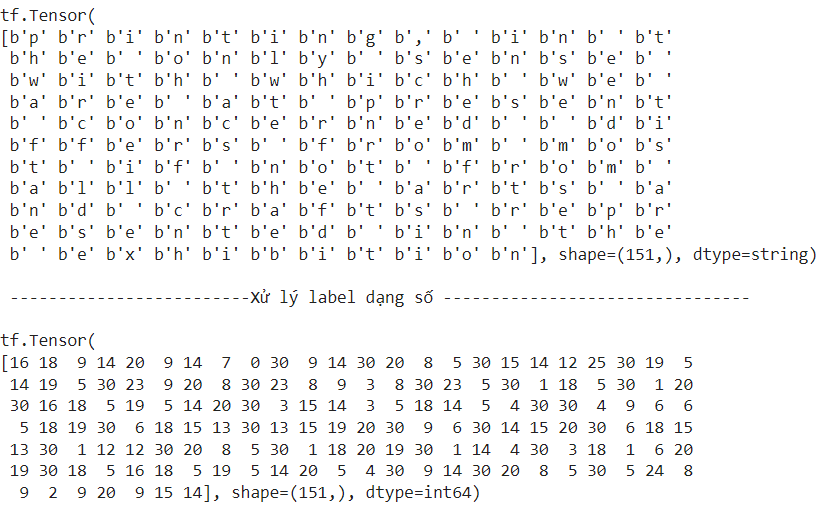
Hình 16: Tensor spetrogram

- Sau đó, chúng em chuẩn hóa bằng cách tính toán giá trị tuyệt đối để loại bỏ dấu của các giá trị và lấy căn bậc hai của spectrogram để làm giá trị biên độ không còn biến động nhiều rồi thực hiện chuẩn hóa Z-score để giúp cải thiện tính ổn định, hiệu suất và khả năng tổng quát hóa của mô hình học máy trong quá trình huấn luyện và dự đoán.



Hình 17: Tensor spetrogram sau khi chuẩn hóa

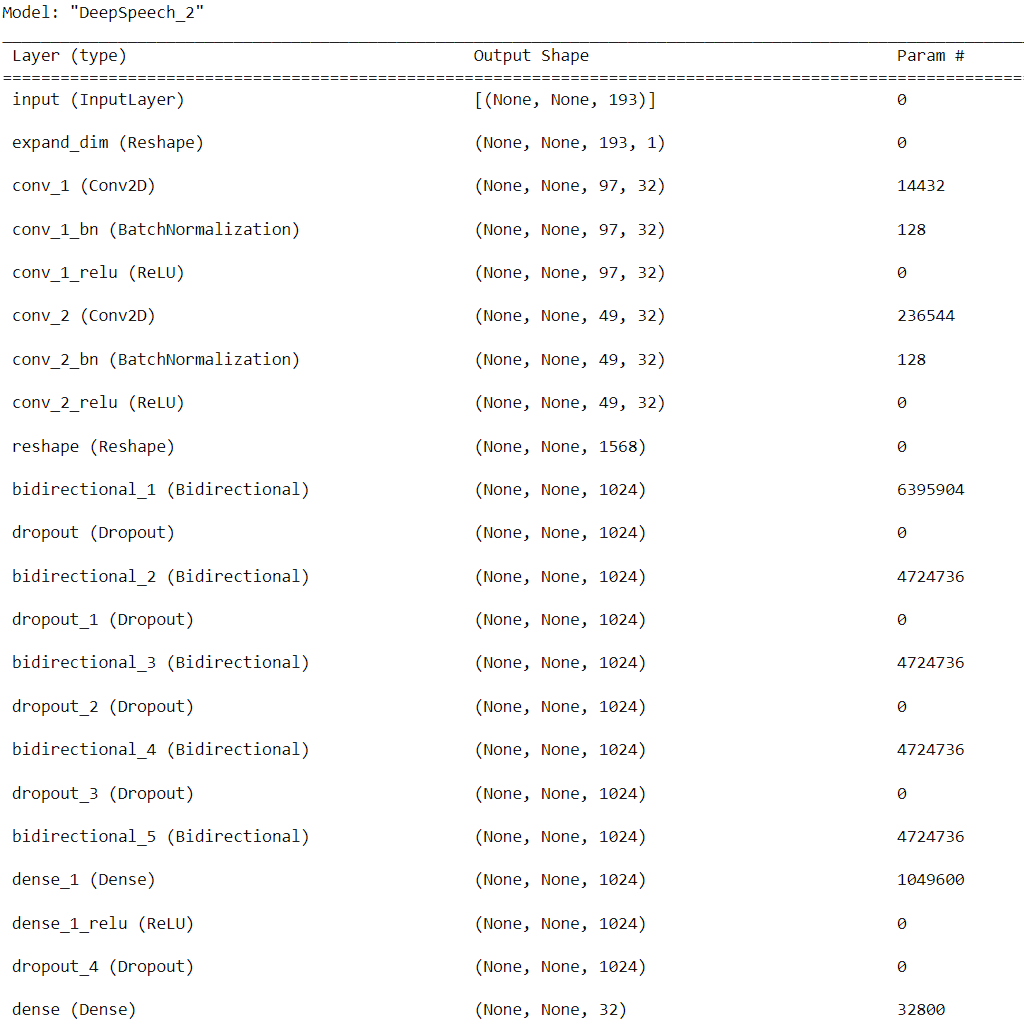
- Bây giờ chúng em xử lý chuỗi nhãn theo các bước sau: đầu tiên, chuỗi nhãn được chuyển đổi thành chữ thường để đảm bảo tính thống nhất trong dữ liệu đầu vào. Tiếp theo, chuỗi nhãn được tách thành các thành phần đơn vị Unicode để xử lý từng phần tử của nhãn một cách độc lập. Sau đó, các ký tự trong chuỗi nhãn được ánh xạ thành các số tương ứng, làm cho dữ liệu có thể được truyền vào mô hình dưới dạng số.



Hình 18: Chuyển đổi label sang dạng số

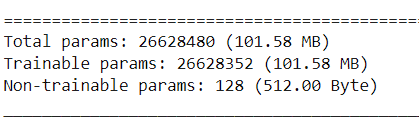
**Bước 2: Xây model**

* Sử dụng mạng tích chập và mạng hồi quy để thực hiện xây dựng mô hình gồm các lớp như bên dưới.



Hình 19: Cấu trúc mô hình

* Tham số mô hình: Tham số tham gia học, không học như hình bên dưới



Hình 20: Số lượng tham số mô hình

**Bước 3: Xây dựng hàm giải mã**: Giải mã kí tự từ dạng số 🡪 dạng chữ

**Bước 4 : Xây dựng hàm đánh giá**: CTC loss hàm đánh giá giúp mô hình huấn luyện. Wer\_Score(số lượng từ dự đoán sai so với nhãn) dùng đề đánh giá độ chính xác của mô hình

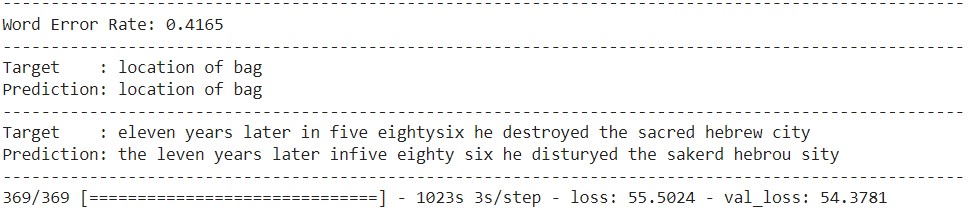
**Bước 5: Huấn luyện(huấn luyện trên 2 số lượng epochs 10 và 20).**

## **Kết quả**

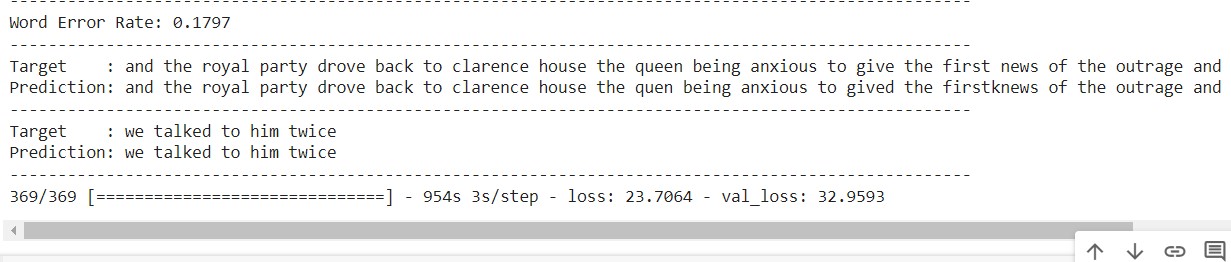
Các tham số:

* epochs: Kỉ nguyên huấn luyện
* Loss: Giá trị hàm CTC trên tập dữ liệu huấn luyện
* Val\_loss: Giá trị hàm CTC trên tập dữ liệu đánh giá

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Kết quả thu được sau khi huấn luyện nhiều epochs khác nhau** | | | | |
| Số epochs | Thời gian | Loss | Val\_loss | Wer\_score |
| 10 | 6 giờ | 55.5024 | 54.3781 | 0.4165 |
| 25 | 13 giờ | 23.7064 | 32.9593 | 0.1797 |



Hình 21: Huấn luyện với 10 epochs



Hình 22: Huấn luyện với 25 epochs

## **Đánh giá và thảo luận**

🡪 Mô hình sau khi huấn luyện qua 20 kỉ nguyên đạt được kết quả wer\_score giao động từ 15-17% tương ứng với dự đoán số kí tự chính xác đạt 83-85% . Dựa vào sơ đồ về giá trị loss, val\_loss và wer\_Score có những đánh giá sau:

* Từ epoch thứ 22 giá trị loss trên tập đánh giá và trên tập huấn luyện đang có xu hướng đi ngược hướng nhau, cho thấy mô hình có thể dẫn đến overfitting 🡪 Tạm dừng huấn luyện.
* Tuy 2 giá trị giữa loss và val\_loss đang có xu hướng xa nhau nhưng biểu đồ wer\_score cho thấy giá trị vẫn đang giảm 🡪 Vẫn có thể train model thêm số lượng hữu hạn kỉ nguyên tiếp theo.

🡪 Huấn luyện mô hình dừng ở epoch thứ 25 nguyên nhân do:

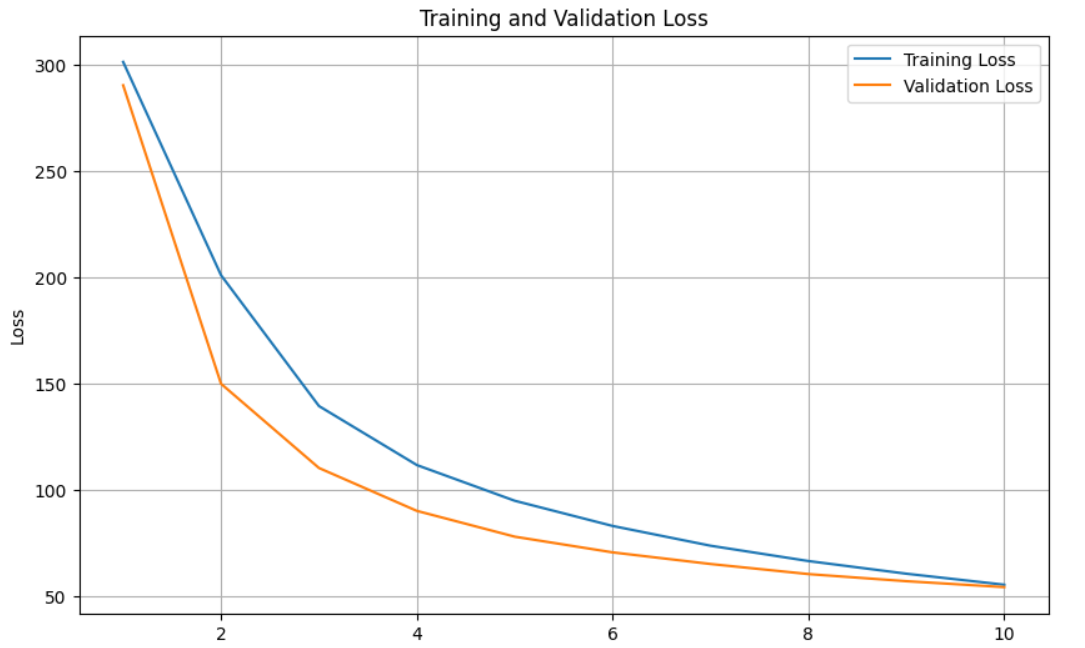
* Độ chính xác giao động từ 83-85% được đánh giá ở mức ổn
* Chi phí thơi gian để huấn luyện khá cao nên tiếp tục huấn luyện khả năng thu về kết quả không tương xứng với chi phí.

🡪 Điểm yếu của mô hình:

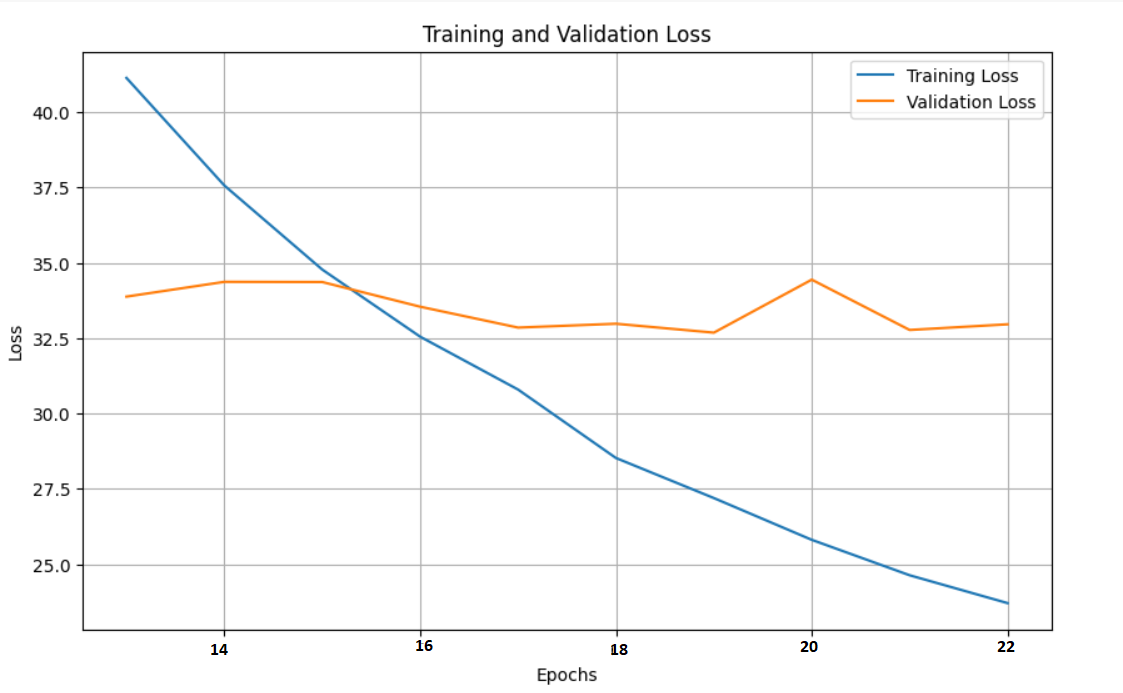
* Được huấn luyện bởi tập dữ liệu âm thanh đã qua xử lý và ít nhiễu, khó nhận biết được những âm thanh có nhiều tạp âm.
* Mô hình chỉ được huấn luyện cho ngôn ngữ tiếng anh nên hạn chế về nhận diện ngôn ngữ.
* Tỷ lệ dữ đoán 83-85% vẫn chưa được coi là tỷ lệ chính xác cao.

🡪 Khắc phục:

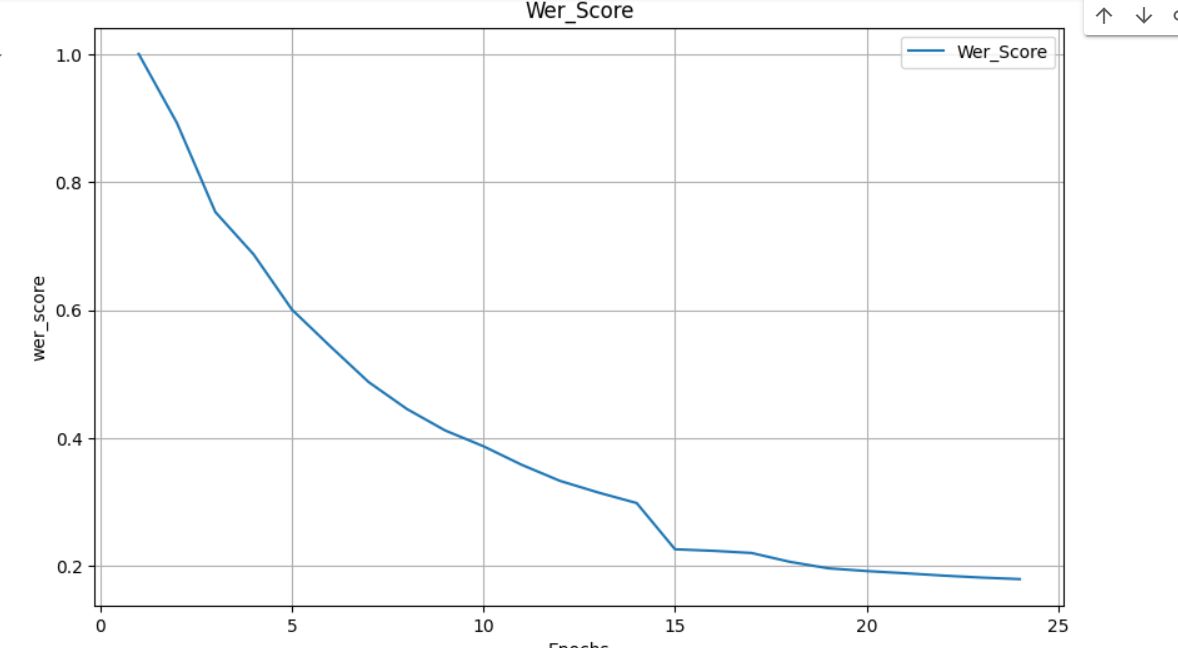
* Thu thập thêm nhiều dữ liệu đa dạng hơn về ngôn ngữ, âm thanh thực tế.
* Sử dụng các mô hình kiến trúc tiên tiến hơn như Transformer, điều chỉnh tối ưu về mặt tham số hơn.



Hình 23: Giá trị loss và val\_loss trong 10 epochs đầu tiên



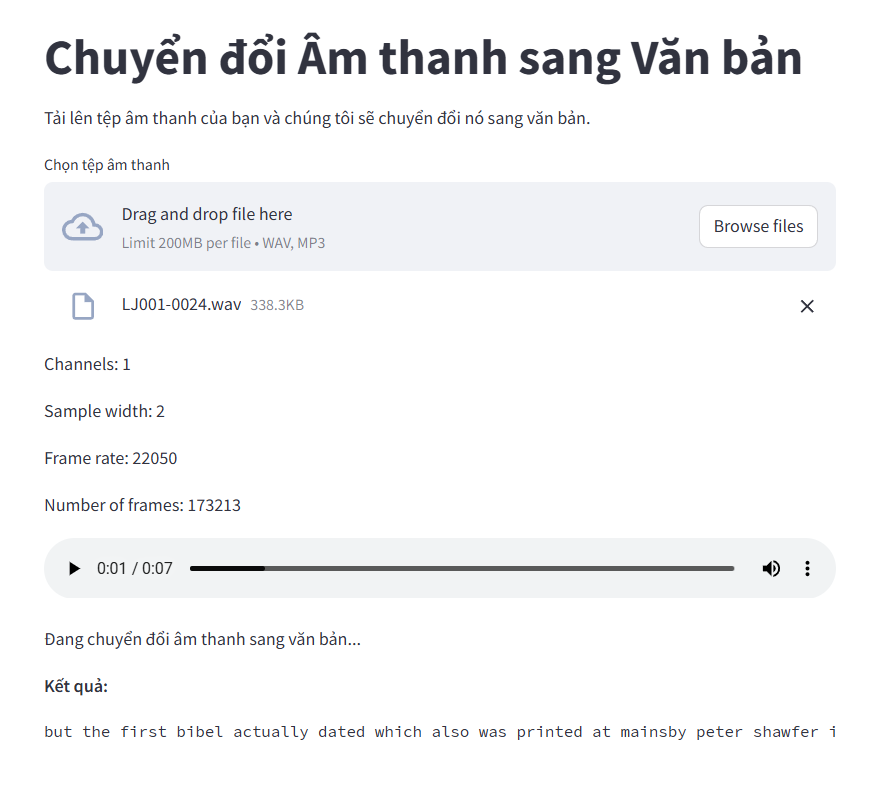
Hình 24: Giá trị loss và val\_loss trong 14-22 epochs



Hình 25: Giá trị wer\_Score trong khoảng 25 epochs

## **Website thử nghiệm model**

* Website có chứng năng upload file âm thanh wav/mp3 và thực hiện dự đoán chữ của người nói trong file âm thanh.



# **Kết luận**

- Qua quá trình khảo sát đã đạt được những kết quả sau:

+ Tìm hiểu yêu cầu và ứng dụng: Chúng tôi đã phân tích chi tiết các yêu cầu của bài toán và xác định các ứng dụng thực tế như trợ lý ảo và dịch vụ khách hàng.

+ Nắm bắt mô hình CRNN và tiền xử lý âm thanh: Đã nghiên cứu kỹ mô hình CRNN và các kỹ thuật tiền xử lý âm thanh để tối ưu hóa hiệu suất nhận diện.

+ Chức năng hệ thống: Hệ thống đã hỗ trợ nhận diện âm thanh và chuyển đổi thành văn bản với độ chính xác khá cao(83-85%) trong một số ngữ cảnh.

- Những hạn chế còn tồn tại:

+ Dự đoán chưa chính xác hoàn toàn: Hệ thống gặp khó khăn trong các môi trường âm thanh phức tạp và giọng nói đa dạng.

+ Chưa tối ưu hóa công nghệ: Hiệu suất và thời gian xử lý chưa đạt mức tối ưu.

+ Xử lý chưa nhanh: Thời gian xử lý vẫn còn chậm, đặc biệt với các đoạn âm thanh dài.

- Hướng phát triển:

+ Nâng cấp và cập nhật: Tiếp tục theo dõi công nghệ mới và cập nhật tính năng để cải thiện trải nghiệm người dùng.

+ Cải thiện tốc độ: Tối ưu hóa mã nguồn và thuật toán, triển khai trên nền tảng phần cứng mạnh mẽ hơn.

+ Tăng cường dự đoán chính xác: Huấn luyện mô hình trên dữ liệu đa dạng hơn và áp dụng các kỹ thuật học máy tiên tiến.

🡪 Chức năng hệ thống: Hệ thống đã hỗ trợ nhận diện âm thanh và chuyển đổi thành văn bản với độ chính xác khá cao trong một số ngữ cảnh.

**Mã nguồn dự án:**

* Link notebook: <https://colab.research.google.com/drive/1GQeLgDeAT39vn0s4Ckx8utIF6P2lr5r0?usp=sharing>
* Link Dataset:

<https://keithito.com/LJ-Speech-Dataset/>

# **Tài liệu tham khảo**

* + - 1. <https://nvidia.github.io/OpenSeq2Seq/html/speechrecognition.html#models>

(Cấu trúc mô hình)

* + - 1. <https://www.analyticsvidhya.com/blog/2021/08/beginners-guide-to-convolutional-neural-network-with-implementation-in-python/>

(Kiến trúc mô hình CNN)

* + - 1. <https://distill.pub/2017/ctc/>

(Sử dụng CTC loss với nhận dạng giọng nói)

* + - 1. <https://www.analyticsvidhya.com/blog/2019/07/learn-build-first-speech-to-text-model-python/>  
         (Cấu trúc của âm thanh và biến đổi)
      2. https://keras.io/examples/audio/ctc\_asr/  
         (Sử dụng keras trong mô hình)
      3. <https://viblo.asia/p/xu-ly-du-lieu-am-thanh-Qpmlezg95rd>

(Tiền xử lý dữ liệu âm thanh)

* + - 1. <https://www.sciencedirect.com/topics/computer-science/word-error-rate>

(Phương pháp đánh giá Word Error Rate)