Voiceprint Recognition System using Deep Learning

1. Principles of Speaker Recognition

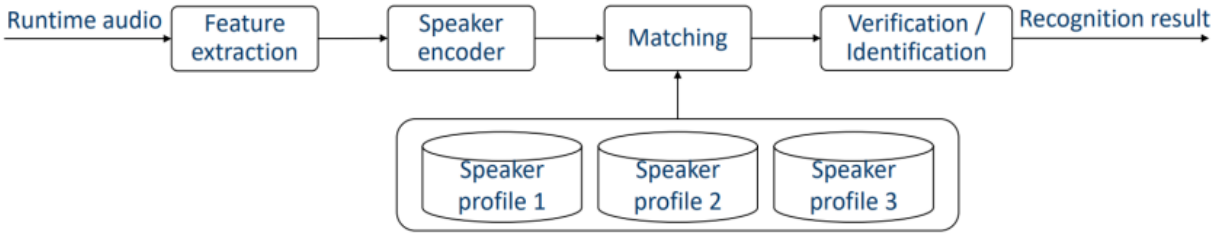


Figure 1.1. Basic speaker recognition system framework in a nutshell

2. Text-prompted speaker recognition system workflow

- Có hai cơ chế: Text-dependent và text-independent.
- **Text-dependent (TD)** là một phương pháp phổ biến trong nhận dạng giọng nói, giả định rằng đoạn văn bản nói luôn giống nhau hoặc có biến thể nhỏ. Điều này giúp đơn giản hóa một số vấn đề, như biến thể của mỗi âm vị là nhỏ (do chỉ cần mô hình một số âm vị), và biến thể về độ dài âm thanh cũng nhỏ (do giả định về cửa sổ độ dài cố định trong nhận dạng người nói theo văn bản cụ thể). Text-dependent thường được sử dụng trong các ứng dụng thực tế như các lệnh "Ok Google," "Hey Siri," "Alexa," hay mật khẩu thông qua giọng nói. Tuy nhiên, phương pháp này có nguy cơ bị tấn công bằng cách sử dụng bản ghi giọng nói của người dùng.
- **Text-independent (ID)**: Trong text-independent, hệ thống không biết "Đã nói gì." Trong khi đó, text-prompted là một hệ thống an ninh hơn, vì yêu cầu đoạn văn bản khi thử nghiệm thay đổi, có thể bao gồm cả mô hình nhận dạng giọng và mô hình xác minh người nói. Tuy nhiên, vì sự phức tạp cao hơn ⇒ text-prompted có thể đòi hỏi nhiều công sức hơn để triển khai. Cái này sẽ nhận dạng được người nói dựa trên những cụm từ tùy ý.

1. **Speaker Verification (Xác minh người nói):**
   - **Mục tiêu chính:** Xác định xem người nói có phải là người được xác định hay không, thường là trong ngữ cảnh xác minh danh tính hoặc bảo mật.
   - **Ưu điểm:** Đề cập đến người nói cụ thể, không nhất thiết phải hiểu nội dung nói.
   - **Ví dụ ứng dụng:** Các hệ thống bảo mật giọng nói, xác minh định danh qua giọng nói.
2. **Speech Recognition (Nhận dạng giọng):**
   - **Mục tiêu chính:** Chuyển đổi giọng nói thành văn bản hoặc lệnh máy tính.
   - **Ưu điểm:** Liên quan đến việc hiểu nội dung của những gì người nói.

- **Ví dụ ứng dụng:** Hệ thống nhận dạng giọng để điều khiển máy tính, trợ lý ảo như Siri, Google Assistant, dịch tiếng nói. **TẬP TRUNG VÀO VIỆC TIỀN XỬ LÝ, TRÍCH XUẤT ĐẶC TRƯNG, XÁC ĐỊNH NGỮ CẢNH VÀ HIỂU Ý NGHĨA**

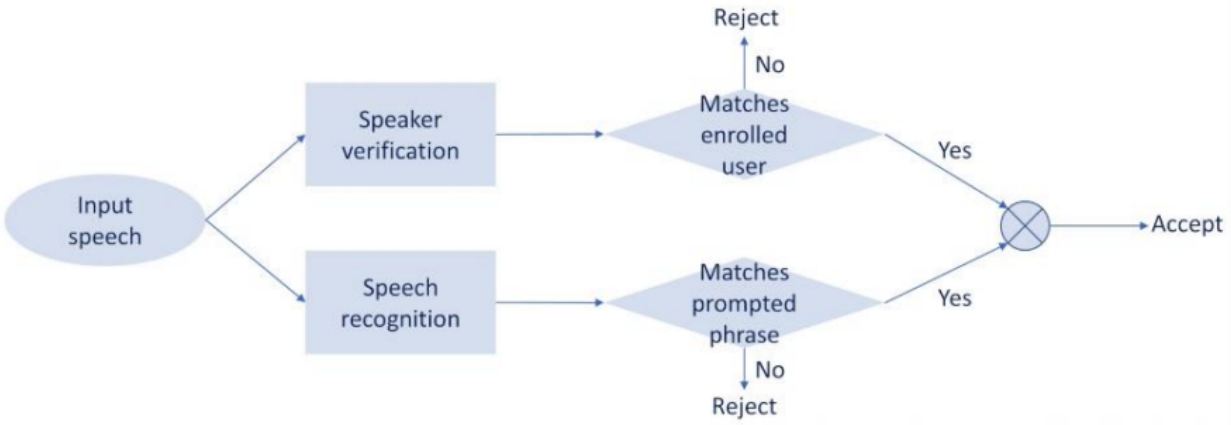


Figure 1.5. Text-prompted speaker recognition system workflow

3. Dataset:

- Sử dụng bộ dữ liệu Corpus LibriSpeech: một bộ sưu tập gần 1,000 giờ ghi âm sách nói, là một phần của dự án LibriVox.
- Dữ liệu huấn luyện được chia thành 3 phần gồm bộ dữ liệu 100 giờ, 360 giờ và 500 giờ, trong khi dữ liệu phát triển và kiểm thử được chia thành các nhóm 'clean' và 'other', tùy thuộc vào khả năng của hệ thống nhận dạng giọng nói tự động. Mỗi tập phát triển và kiểm thử có khoảng 5 giờ âm thanh. Bộ dữ liệu này cũng cung cấp các mô hình ngôn ngữ n-gram và các văn bản tương ứng được trích từ sách Project Gutenberg, bao gồm 803 triệu token và 977 nghìn từ duy nhất.
-

```
FeaturesDict({
    'chapter_id': int64,
    'id': string,
    'speaker_id': int64,
    'speech': Audio(shape=(None,), dtype=int16),
    'text': Text(shape=(), dtype=string),
})
```

- Link: https://www.tensorflow.org/datasets/catalog/librispeech?hl=vi#:~:text=LibriSpeech is a corpus of,been carefully segmented and aligned.

4. Model

- Bài viết đề cập đến model: LSTM, Attention và Transformer

- Trong kiến trúc Transformer, bộ mã hóa có nhiệm vụ ánh xạ một chuỗi đầu vào thành biểu diễn liên tục và đưa vào bộ giải mã. Bộ giải mã sau đó nhận đầu ra của bộ mã hóa cùng với đầu ra của bước giải mã trước đó để tạo ra chuỗi đầu ra. Bộ mã hóa sử dụng cơ chế tự chú ý để tập trung vào các phần quan trọng trong đầu vào. Bộ giải mã sử dụng cả cơ chế tự chú ý và cơ chế chú ý giữa bộ mã hóa và bộ giải mã, giúp mô hình hiệu quả hóa xử lý thông tin từ xa và duy trì thông tin lâu dài trong quá trình dự đoán chuỗi đầu ra.
- Nói đơn giản hơn, Trong kiến trúc Transformer, bộ mã hóa ánh xạ chuỗi đầu vào thành biểu diễn liên tục và đưa vào bộ giải mã; bộ giải mã sử dụng cả tự chú ý và chú ý giữa bộ mã hóa và bộ giải mã để tạo ra chuỗi đầu ra.

5. Inference

- "Sliding window inference" là một phương pháp được chọn cho hệ thống nhận diện người nói. Ý tưởng chính của loại suy luận này là chia toàn bộ chuỗi thành các cửa sổ trùng lặp, như minh họa dưới đây. Mỗi cửa sổ là một suy luận độc lập, sau đó, quá trình trung bình cộng được áp dụng trên tất cả các cửa sổ. Ưu điểm của phương pháp suy luận này là nó có thể được sử dụng cho cả mạng nơ-ron hồi quy (RNN) và các khối chú ý (attention blocks), hơn nữa, nó sử dụng thông tin của toàn bộ chuỗi có độ dài biến thiên.
-

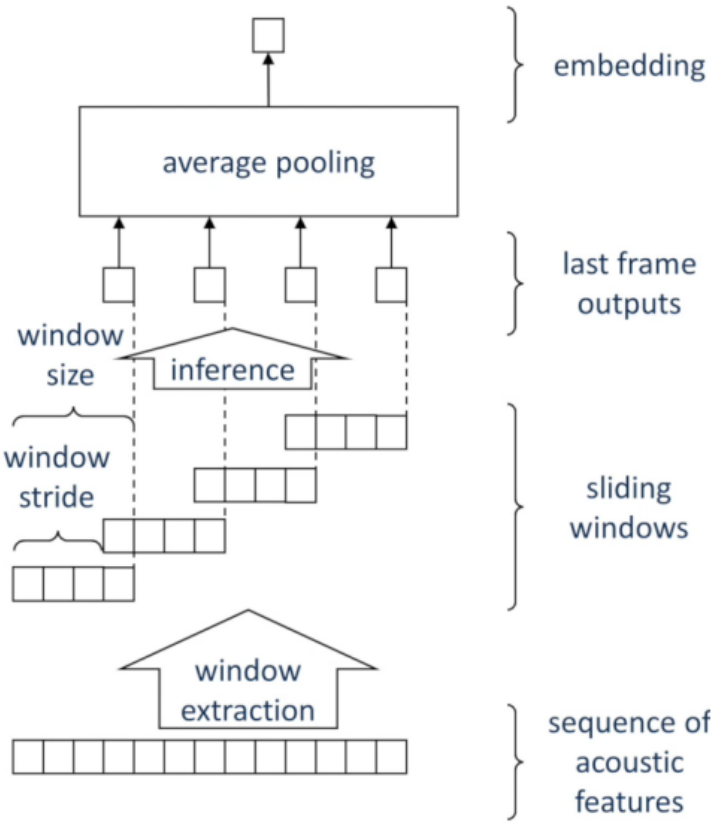


Figure 1.8. Sliding window inference main idea

6. THEORETICAL FOUNDATION AND TECHNOLOGY:

.

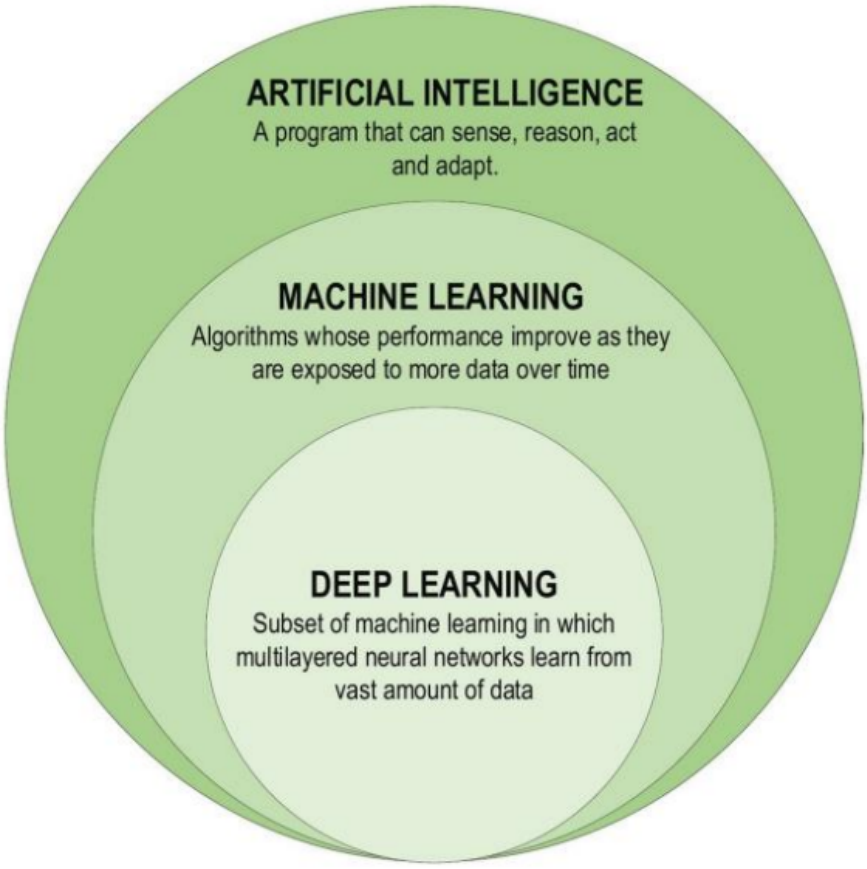


Figure 2.1. Deep Learning family

•

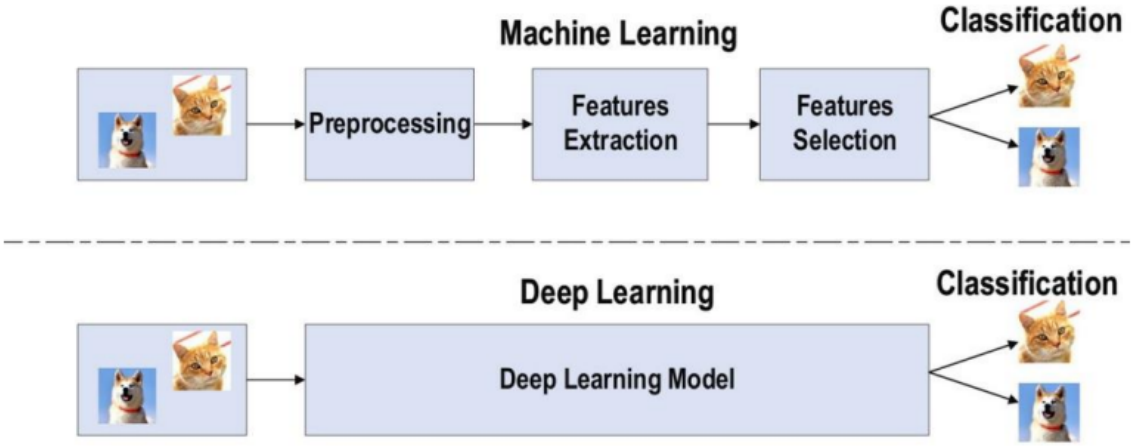


Figure 2.2. The difference between Deep Learning and traditional Machine Learning

-

Học Sâu Giám Sát (Deep Supervised Learning):

- **Đặc điểm Chính:** Xử lý dữ liệu có nhãn, trong môi trường có bộ dữ liệu đầu vào và đầu ra tương ứng (xt, yt) ~ ρ.
- **Phương pháp:** Sử dụng nhiều kỹ thuật học sâu giám sát như mạng nơ-ron hồi quy (RNNs), mạng nơ-ron tích chập (CNNs), và mạng nơ-ron sâu (DNNs). Các phương pháp như đơn vị hồi quy được kiểm soát (GRUs) và phương pháp lưu trữ ngắn hạn dài hạn (LSTM) thuộc danh mục RNN cũng được sử dụng.
- **Ưu điểm:** Khả năng thu thập dữ liệu hoặc tạo ra đầu ra từ kiến thức trước đó.
- **Nhược điểm:** Ranh giới quyết định có thể bị căng khi tập huấn luyện không có các mẫu thuộc lớp nên thuộc.

Học Sâu Bán Giám Sát (Deep Semi-Supervised Learning):

- **Đặc điểm Chính:** Quá trình học dựa trên bộ dữ liệu một phần có nhãn.
- **Phương pháp:** Có thể sử dụng mạng nơ-ron sinh đối nghịch (GANs), Học Sâu Tăng Cường (DRL) và RNNs (bao gồm GRUs và LSTMs) cho học bán giám sát.
- **Ưu điểm:** Giảm thiểu lượng dữ liệu có nhãn cần thiết.
- **Nhược điểm:** Đặc trưng đầu vào không liên quan trong dữ liệu huấn luyện có thể dẫn đến quyết định không chính xác.

Học Sâu Không Giám Sát (Deep Unsupervised Learning):

- **Đặc điểm Chính:** Thực hiện quá trình học mà không cần dữ liệu có nhãn.
- **Phương pháp:** Học đặc trưng quan trọng hoặc biểu diễn nội bộ để khám phá cấu trúc hoặc mối quan hệ chưa xác định trong dữ liệu đầu vào.
- **Nhược điểm:** Không thể cung cấp thông tin chính xác về phân loại dữ liệu và tính toán phức tạp. Phương pháp phổ biến là phân cụm (clustering).

Học Sâu Tăng Cường (Deep Reinforcement Learning):

- **Đặc điểm Chính:** Hoạt động thông qua tương tác với môi trường, tối ưu hóa phần thưởng trong một tình huống cụ thể.
- **Sự Khác Biệt với Học Giám Sát:** Trong học giám sát, dữ liệu đào tạo có câu trả lời, còn trong học tăng cường, không có câu trả lời, và tác nhân tăng cường quyết định hành động để thực hiện nhiệm vụ đã cho.

7. RNN:

Cơ Chế Hoạt Động của RNN:

- RNN hoạt động bằng cách lưu giữ thông tin từ các đầu ra trước đó và truyền nó lại vào mô hình để dự đoán đầu ra tiếp theo.
- Mỗi bước thời gian (time step), RNN nhận đầu vào hiện tại và thông tin từ bước thời gian trước đó, sau đó sinh ra đầu ra và lưu giữ thông tin để sử dụng cho bước thời gian tiếp theo.
- Mô hình có khả năng xử lý dữ liệu tuần tự như chuỗi thời gian hoặc văn bản.

Ưu Điểm của RNN:

1. **Xử Lý Dữ Liệu Tuần Tự:** RNN làm việc hiệu quả trên dữ liệu tuần tự như chuỗi thời gian hoặc ngôn ngữ tự nhiên.
2. **Bộ Nhớ Nội Tại:** RNN có khả năng nhớ thông tin từ các bước thời gian trước đó, giúp nó giữ lại thông tin quan trọng trong quá khứ.
3. **Kết Nối Thông Tin:** Có khả năng kết nối thông tin từ quá khứ với công việc hiện tại.

Nhược Điểm của RNN:

1. **Vanishing Gradient Problem:** Trong quá trình huấn luyện, gradient có thể trở nên rất nhỏ, làm cho mô hình không thể học được thông tin từ các bước thời gian xa.
2. **Computational Complexity:** Việc tính toán trong RNN có thể trở nên phức tạp khi chuỗi thời gian dài.
3. **Khả Năng Memory Hạn Chế:** Khả năng của RNN giữ lại thông tin có thể bị giảm khi khoảng cách giữa thông tin quan trọng và nơi cần nó tăng lên.

Kết Luận:
RNN là một kiến trúc mạng nơ-ron mạnh mẽ cho việc xử lý dữ liệu tuần tự và giữ lại thông tin từ quá khứ, nhưng nó cũng đối mặt với các thách thức như vanishing gradient và khả năng tính toán phức tạp.

Visualize: https://www.youtube.com/watch?v=D6vxZ5KMD18

8. LSTM:

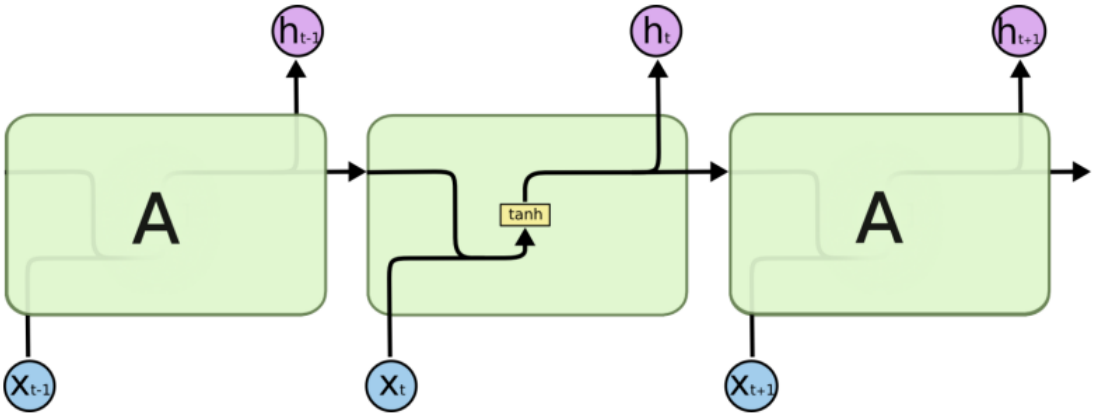
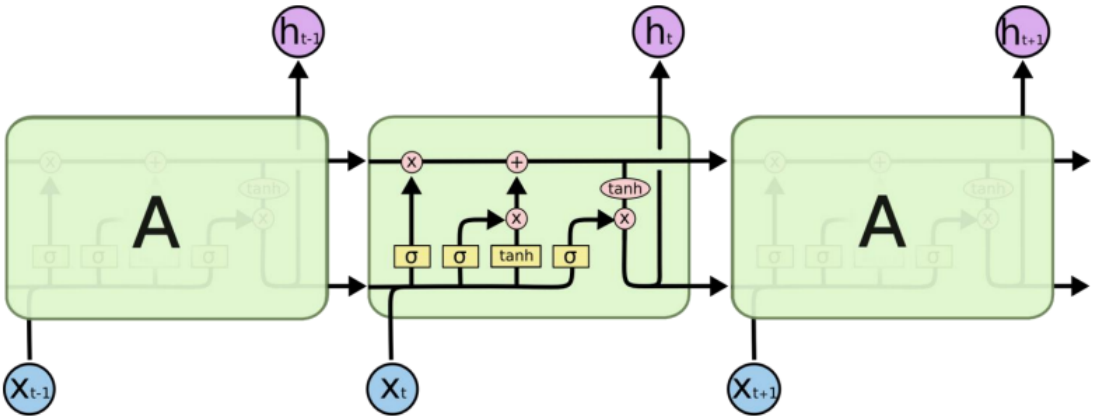


Figure 2.7. The repeating module in a standard RNN contains a single layer

LSTMs also have this chain like structure, but the repeating module has a different structure. Instead of having a single neural network layer, there are four, interacting in a very special way.



Mô tả tổng quan:

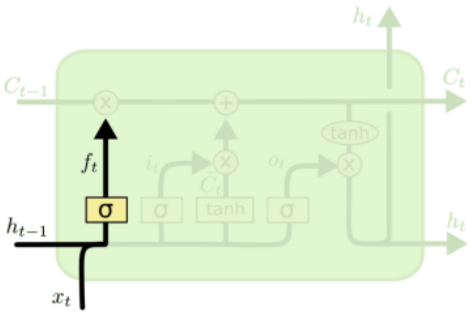
- **LSTMs** là một dạng đặc biệt của RNNs, được thiết kế đặc biệt để giải quyết vấn đề của việc học các phụ thuộc dài hạn.
- Cấu trúc của RNNs và LSTMs đều có dạng chuỗi các mô-đun lặp lại của mạng nơ-ron, nhưng mô-đun lặp lại trong LSTM có cấu trúc đặc biệt.

Cấu trúc mô-đun lặp lại trong LSTM:

1. **Trong RNN thông thường:**
   - Mô-đun lặp lại có cấu trúc đơn giản, thường chỉ là một tầng tanh (hình 2.7).
2. **Trong LSTM:**
   - Mô-đun lặp lại có cấu trúc phức tạp hơn với bốn tầng tương tác theo cách đặc biệt (hình 2.8).
   - Gồm bốn tầng: Forget Gate Layer, Input Gate Layer, Cell State, và Output Gate Layer.

Các bước trong LSTM:

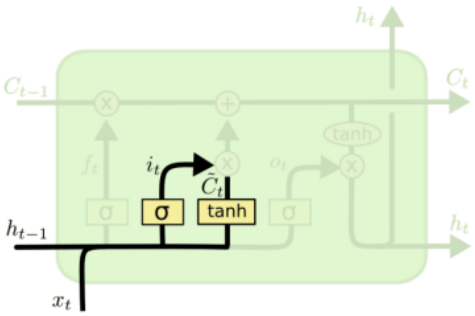
1. **Forget Gate Layer (Hình 2.9):**
   - Sử dụng tầng sigmoid để quyết định thông tin nào cần loại bỏ từ trạng thái tế bào (cell state) trước đó.
   - Số đầu ra là giữa 0 và 1, với 1 đại diện cho "giữ lại hoàn toàn" và 0 đại diện cho "loại bỏ hoàn toàn."
   -



$$f_t = \sigma\left(W_f \cdot [h_{t-1}, x_t] \; + \; b_f\right)$$

Figure 2.9. Step-by-Step LSTM Walk Through (1)

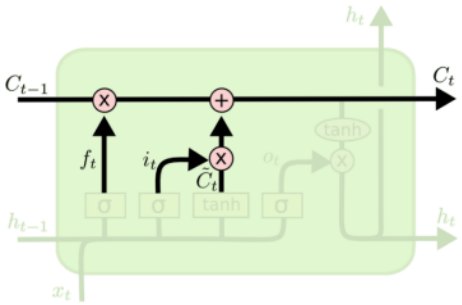
2. **Input Gate Layer (Hình 2.10):**
   - Sử dụng tầng sigmoid để quyết định thông tin mới nào sẽ được lưu trữ trong trạng thái tế bào.
   - Tầng tanh tạo một vector các giá trị ứng cử mới có thể được thêm vào trạng thái tế bào.
   -



$$i_t = \sigma\left(W_i \cdot [h_{t-1}, x_t] \; + \; b_i\right)$$
$$\tilde{C}_t = \tanh(W_C \cdot [h_{t-1}, x_t] \; + \; b_C)$$

Figure 2.10. Step-by-Step LSTM Walk Through (2)

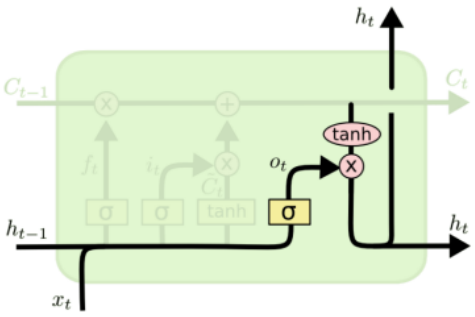
3. **Combine and Update (Hình 2.11):**
   - Kết hợp hai bước trước để tạo ra cập nhật cho trạng thái tế bào mới.
   - Sử dụng forget gate để quên thông tin và thêm vào các giá trị ứng cử mới đã được quyết định.
   -



$$C_t = f_t * C_{t-1} + i_t * \tilde{C}_t$$

Figure 2.11. Step-by-Step LSTM Walk Through (3)

4. **Output Gate Layer (Hình 2.12):**
   - Sử dụng tầng sigmoid để quyết định thông tin nào sẽ được xuất ra.
   - Đưa trạng thái tế bào qua hàm tanh và nhân với đầu ra của cổng sigmoid để chỉ xuất ra các phần quyết định giữ lại.
   -



$$o_t = \sigma\left(W_o\left[h_{t-1}, x_t\right] + b_o\right)$$
$$h_t = o_t * \tanh\left(C_t\right)$$

Figure 2.12. Step-by-Step LSTM Walk Through (4)

Ưu và Nhược Điểm:

- **Ưu điểm:** LSTMs giải quyết được vấn đề phụ thuộc dài hạn, giữ thông tin trong thời gian dài.
- **Nhược điểm:** Cấu trúc phức tạp và tăng độ phức tạp tính toán so với RNNs thông thường.

Visualize: https://www.youtube.com/watch?v=5dMXyiWddYs

9. Transformer:

Thành Phần Chính của Transformer:

1. **Encoder và Decoder:**
   - **Encoder:** Nhận dữ liệu đầu vào và tạo ra biểu diễn (embedding) của nó.
   - **Decoder:** Dự đoán đầu ra dựa trên biểu diễn của dữ liệu đầu vào.
2. **Tầng Tự-Chú Ý (Self-Attention Layer):**
   - **Chức năng chính:** Tính toán trọng số chú ý cho mỗi từ trong chuỗi dựa trên tất cả các từ khác trong chuỗi.

- **Quá trình:** Tính điểm chú ý (attention score) bằng cách sử dụng ba ma trận trọng số: Query, Key, và Value. Các điểm chú ý sau đó được tính toán và được sử dụng để tạo ra biểu diễn mới cho mỗi từ.

3. **Tầng Feed-Forward Neural Network:**
   - **Chức năng:** Thực hiện các phép biến đổi phi tuyến tính cho biểu diễn của từ được tạo ra bởi tầng tự-chú ý.
4. **Tầng Normalization và Residual Connection:**
   - **Normalization:** Chuẩn hóa giá trị đầu ra từ tầng feed-forward để ổn định quá trình học.
   - **Residual Connection:** Thêm giá trị đầu vào của tầng vào đầu ra của tầng để tránh vấn đề vanishing/exploding gradient.

Quy Trình Hoạt Động:

1. **Embedding Input:**
   - Mỗi từ trong câu được biểu diễn dưới dạng vector embedding.
2. **Encoder:**
   - **Tầng Tự-Chú Ý:** Tính trọng số chú ý cho mỗi từ trong chuỗi đầu vào.
   - **Tầng Feed-Forward:** Thực hiện biến đổi phi tuyến tính cho mỗi từ.
3. **Decoder:**
   - **Tầng Tự-Chú Ý:** Tính trọng số chú ý cho mỗi từ trong chuỗi đầu ra.
   - **Tầng Feed-Forward:** Thực hiện biến đổi phi tuyến tính cho mỗi từ.
4. **Tạo Ra Đầu Ra:**
   - Dựa vào biểu diễn của từ ở tầng decoder, mô hình dự đoán từ tiếp theo.

Ưu Điểm và Nhược Điểm:

- **Ưu Điểm:**
  - Hiệu quả xử lý phụ thuộc xa và đồng thời trong chuỗi dữ liệu.
  - Không mắc phải vấn đề vanishing/exploding gradient như các mô hình RNN.
  - Thích hợp cho việc xử lý ngôn ngữ tự nhiên, dịch máy, và nhiều ứng dụng khác.
- **Nhược Điểm:**
  - Yêu cầu nhiều tài nguyên tính toán và bộ nhớ, đặc biệt là đối với dữ liệu lớn.
  - Cần lưu trữ trọng số chú ý cho tất cả các vị trí trong chuỗi, tăng bộ nhớ yêu cầu.

Kết Luận:
Transformer đã đem lại đột phá đáng kể trong lĩnh vực xử lý ngôn ngữ tự nhiên và đã trở thành một trong những kiến trúc quan trọng trong nền deep learning. Sự kết hợp giữa tầng tự-chú ý và tầng feed-forward đã giúp mô hình hiệu quả hóa quá trình học và nắm bắt sự phụ thuộc phức tạp trong dữ liệu.

`Bảng so sánh:`

Đặc Điểm	RNN	LSTM	Transformer
Kiến Trúc	Dạng chuỗi tuần tự, các tầng lặp lại nhau.	Đặc biệt thiết kế để giải quyết vấn đề phụ thuộc xa.	Sử dụng cơ chế tự-chú ý và không cần chuỗi.
Xử Lý Phụ Thuộc Dài Hạn	Khó xử lý phụ thuộc dài hạn.	Thiết kế để giải quyết vấn đề phụ thuộc xa, nhưng vẫn có thể gặp vấn đề khi chuỗi quá dài.	Hiệu quả trong xử lý phụ thuộc xa, không gặp vấn đề phụ thuộc dài hạn.
Biểu Diễn Ngôn Ngữ	Tốt cho dữ liệu tuần tự, ngôn ngữ tự nhiên.	Cải thiện khả năng xử lý ngôn ngữ và phụ thuộc xa.	Hiệu quả cho xử lý ngôn ngữ và dữ liệu tuần tự.
Hiệu Năng	Có thể bị chậm đối với dữ liệu lớn.	Hiệu năng tốt hơn so với RNN, nhưng vẫn có thể gặp vấn đề với dữ liệu lớn.	Hiệu năng cao, khả năng xử lý đa vòng đồng thời.
Vấn Đề Gradient	Gặp vấn đề vanishing/exploding gradient.	Giảm vấn đề vanishing/exploding gradient thông qua cổng.	Không gặp vấn đề vanishing/exploding gradient.
Quá Trình Học	Quá trình học theo chiều thuận.	Quá trình học theo chiều thuận và ngược.	Quá trình học tương đối độc lập giữa các vị trí.
Ứng Dụng	Xử lý dữ liệu tuần tự, ngôn ngữ tự nhiên.	Xử lý ngôn ngữ, dịch máy, và ứng dụng yêu cầu xử lý phụ thuộc xa.	Xử lý ngôn ngữ, dịch máy, và nhiều ứng dụng đòi hỏi xử lý đa vòng đồng thời.

10. Sliding Window Inference

Sliding window inference là một phương pháp trong mô hình chuỗi và xử lý ngôn ngữ tự nhiên, sử dụng cửa sổ trượt để xử lý dữ liệu đầu vào theo kiểu chồng chéo. Trong nhiệm vụ mô hình chuỗi, nó chia chuỗi thành các cửa sổ nhỏ và xử lý chúng tuần tự, giúp nắm bắt sự phụ thuộc tại các vị trí khác nhau. Phương pháp này hữu ích cho dữ liệu có độ dài biến đổi, mặc dù có thể đưa ra chi phí tính toán. Cân nhắc kích thước cửa sổ và độ chồng là quan trọng để đạt được sự cân bằng giữa độ chính xác và hiệu suất.

11. SpecAugment

- Giống như kỹ thuật Augmentation cho hình ảnh nhưng ở đây là đối với âm thanh
- SpecAugment tương tự như các kỹ thuật tăng cường dữ liệu được sử dụng trong lĩnh vực xử lý hình ảnh, nhưng được thiết kế đặc biệt để áp dụng cho dữ liệu âm thanh, đặc biệt là cho các tác vụ liên quan đến xử lý giọng nói hoặc âm thanh. Bằng cách biến đổi các đặc trưng âm thanh, SpecAugment giúp mô hình học được các đặc điểm quan trọng và trở nên robust hơn đối với các biến đổi và nhiễu trong dữ liệu âm thanh.
- SpecAugment là một phương pháp tăng cường dữ liệu được áp dụng trực tiếp vào đầu vào đặc trưng của mạng nơ-ron (ví dụ: các hệ số filter bank).
- Chính sách tăng cường này bao gồm việc biến đổi đặc trưng bằng cách uốn cong các đặc trưng, che phủ các khối kênh tần số, và che phủ các khối bước thời gian. SpecAugment nhằm xây dựng một chính sách tăng cường tác động trực tiếp lên log mel spectrogram, giúp mạng học các đặc trưng hữu ích

1. **Time Warping (Biến đổi thời gian):** Áp dụng thông qua hàm `sparse_image_warp` của TensorFlow. Với một log mel spectrogram có τ bước thời gian, nó được xem xét như một hình ảnh trong đó trục thời gian là ngang và trục tần số là dọc. Một điểm ngẫu nhiên dọc theo đường ngang đi qua trung tâm của hình ảnh trong khoảng thời gian (W, τ−W) sẽ được uốn cong về phải hoặc về trái một khoảng w được chọn từ phân phối đồng đều từ 0 đến tham số uốn cong thời gian W.
2. **Frequency Masking (Che phủ tần số):** Áp dụng sao cho f kênh tần số mel liên tiếp [f0, f0+f) bị che phủ, trong đó f được chọn từ phân phối đồng đều từ 0 đến tham số che phủ tần số F, và f0 được chọn từ [0, ν−f). Ở đây, ν là số kênh tần số mel.
3. **Time Masking (Che phủ thời gian):** Áp dụng sao cho t bước thời gian liên tiếp [t0, t0+t) bị che phủ, trong đó t được chọn từ phân phối đồng đều từ 0 đến tham số che phủ thời gian T, và t0 được chọn từ [0, τ−t). Ở đây, τ là số bước thời gian.

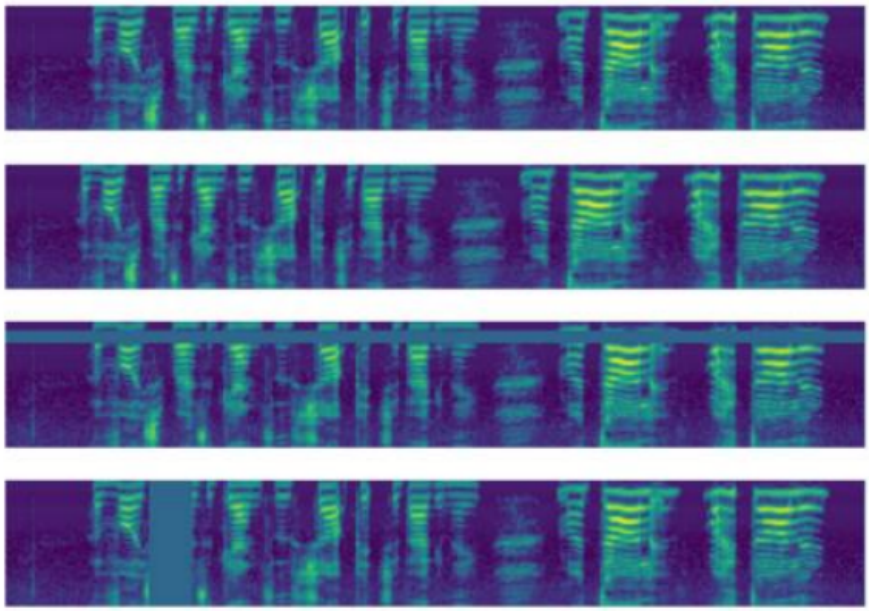


Figure 2.14. Augmentations applied to the base input, given at the top. From top to bottom, the figures depict the log mel spectrogram of the base input with no augmentation, time warp, frequency masking and time masking applied

12. Triplet Loss

- Đây là một hàm loss sử dụng để học sự tương đồng hoặc khoảng cách giữa các mẫu dữ liệu
- Thường được sử dụng trong nhận diện khuôn mặt (FaceNet)
- Mục tiêu chính của Triplet Loss là đảm bảo rằng các cặp không giống nhau (dissimilar) cách xa nhau ít nhất một giá trị biên (margin) so với các cặp giống nhau (similar).

$$\sum_{i}^{N}\left[\left|\left|f\left(x_i^a\right)-f\left(x_i^p\right)\right|\right|_2^2-\left|\left|f\left(x_i^a\right)-f\left(x_i^n\right)\right|\right|_2^2+\propto\right]$$

Figure 2.15. Triplet loss formula

- **f(x):** Là hàm ánh xạ đầu vào x thành một vector n chiều (n-dimensional).
- **i:** Đại diện cho dữ liệu thứ i.
- **a (anchor), p (positive), n (negative):** Các chỉ số cho anchor, positive và negative samples, tương ứng.
- **w:** Là vector nhúng được sinh ra bởi hàm ánh xạ.

Đây là cách giải thích chi tiết của công thức:

- $f(x_i^a)$: Là hàm biểu diễn (embedding function) của một điểm dữ liệu anchor x_i^a.
- $f(x_i^p)$: Là hàm biểu diễn của một điểm dữ liệu tích cực (positive) x_i^p, có nghĩa là điểm dữ liệu này cùng lớp với điểm dữ liệu anchor.
- $f(x_i^n)$: Là hàm biểu diễn của một điểm dữ liệu tiêu cực (negative) x_i^n, có nghĩa là điểm dữ liệu này thuộc một lớp khác so với điểm dữ liệu anchor.

Mục tiêu của Triplet Loss là:

1. Minimize khoảng cách (được đo bằng norm L2, hay Euclidean distance) giữa anchor và positive, tức là $\|f(x_i^a) - f(x_i^p)\|_2^2$.
2. Maximize khoảng cách giữa anchor và negative, tức là $\|f(x_i^a) - f(x_i^n)\|_2^2$.
3. Đảm bảo rằng khoảng cách giữa anchor và negative lớn hơn khoảng cách giữa anchor và positive ít nhất là một lượng α (được gọi là margin).

Triplet Loss được tính bằng cách lấy tổng (sum) qua tất cả các triplets có sẵn trong tập dữ liệu huấn luyện. Cho mỗi triplet, hàm mất mát sẽ tính toán như sau:

$$L = \sum_i \left[\|f(x_i^a) - f(x_i^p)\|_2^2 - \|f(x_i^a) \downarrow {}^{e}(x_i^n)\|_2^2 + \alpha\right]$$

Giá trị α là một hyperparameter quan trọng mà bạn phải chọn trước khi huấn luyện. Nó quyết định mức độ "đẩy" các điểm dữ liệu tiêu cực ra xa khỏi điểm dữ liệu anchor so với các điểm dữ liệu tích cực.

Trong hàm mất mát này, phần bên trong ngoặc vuông chỉ được tính khi nó dương, nghĩa là chỉ khi khoảng cách từ anchor đến positive cộng với margin α nhỏ hơn khoảng cách từ anchor đến negative. Nếu điều kiện này không được thoả mãn (khoảng cách từ anchor đến negative đã đủ lớn), giá trị trong ngoặc vuông sẽ được coi là 0 và không có lỗi nào được tính cho triplet đó. Điều này được gọi là "positive semi-definite", đảm bảo rằng hàm mất mát không bao giờ âm.

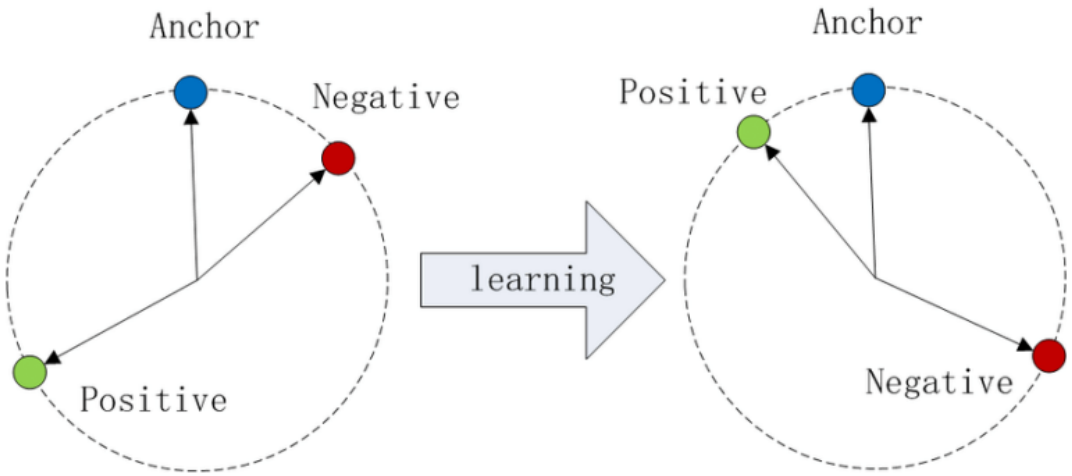


Figure 2.16 The objective of triplet loss

Trong Triplet Loss, quá trình sử dụng các nhóm ba mục gọi là triplets, mỗi triplet bao gồm:

1. **Anchor (mẫu gốc):** Là một mẫu cố định với một danh tính nhất định.
2. **Positive (mẫu tương đồng):** Là một mẫu gần với anchor, thường là một mẫu cùng lớp hoặc cùng phân loại.
3. **Negative (mẫu không tương đồng):** Là một mẫu xa với anchor, không cùng lớp hoặc phân loại với anchor.

Mục tiêu của Triplet Loss là tối thiểu hóa khoảng cách giữa Anchor và Positive, đồng thời tối đa hóa khoảng cách giữa Anchor và Negative. Điều này đồng nghĩa với việc mô hình cố gắng giảm sự khác biệt giữa các mẫu tương đồng (Positive) trong không gian nhúng, trong khi tăng cường sự khác biệt giữa các mẫu không tương đồng (Negative).

Ví dụ, nếu Anchor là một hình ảnh của một con mèo, Positive có thể là một hình ảnh khác của cùng một con mèo, trong khi Negative có thể là hình ảnh của một con chó hoặc một đối tượng khác không liên quan. Mô hình sẽ cố gắng làm cho khoảng cách giữa Anchor và Positive nhỏ hơn, trong khi làm cho khoảng cách giữa Anchor và Negative lớn hơn một giá trị biên (margin). Điều này giúp tạo ra một không gian nhúng sao cho các mẫu tương đồng được đặt gần nhau, và các mẫu không tương đồng được đặt xa nhau.

13. Equal Error Rate (EER)

Trong hệ thống nhận dạng sinh trắc học:

False Acceptance Rate (FAR):

- FAR đo lường tỷ lệ mà hệ thống nhận dạng sinh trắc học chấp nhận một người dùng không được ủy quyền hoặc người giả mạo một cách không đúng.
- Một FAR thấp là mong muốn vì nó chỉ ra xác suất thấp của việc truy cập không được ủy quyền.

False Rejection Rate (FRR):

- FRR đo lường tỷ lệ mà hệ thống nhận dạng sinh trắc học từ chối một người dùng hợp lệ hoặc cá nhân đúng đắn một cách không đúng.
- Một FRR thấp cũng là mong muốn để đảm bảo rằng người dùng hợp lệ không bị từ chối truy cập.

FAR và FRR có xu hướng ngược chiều nhau bởi vì threshold của hệ thống. Quan định quyết định tính hiệu quả của một hệ thống bảo mật sinh trắc.

Equal Error Rate (EER):

- EER là một thước đo hiệu suất thường được sử dụng trong hệ thống nhận dạng sinh trắc học.
- Nó là một giá trị ngưỡng độc lập, cung cấp một đánh giá cân bằng về hiệu suất của hệ thống bằng cách xem xét cả FAR và FRR.
- EER là điểm mà FAR và FRR bằng nhau, tức là hệ thống đưa ra số lượng phê duyệt sai và từ chối sai bằng nhau.
- EER thường được xem xét là một chỉ số quan trọng về hiệu suất và độ chính xác tổng thể của hệ thống nhận dạng sinh trắc học.

Những thông số này giúp đánh giá khả năng của hệ thống trong việc phân biệt giữa người dùng được ủy quyền và người giả mạo, đồng thời đóng vai trò quan trọng trong việc đánh giá tính bảo mật và khả năng sử dụng của hệ thống.

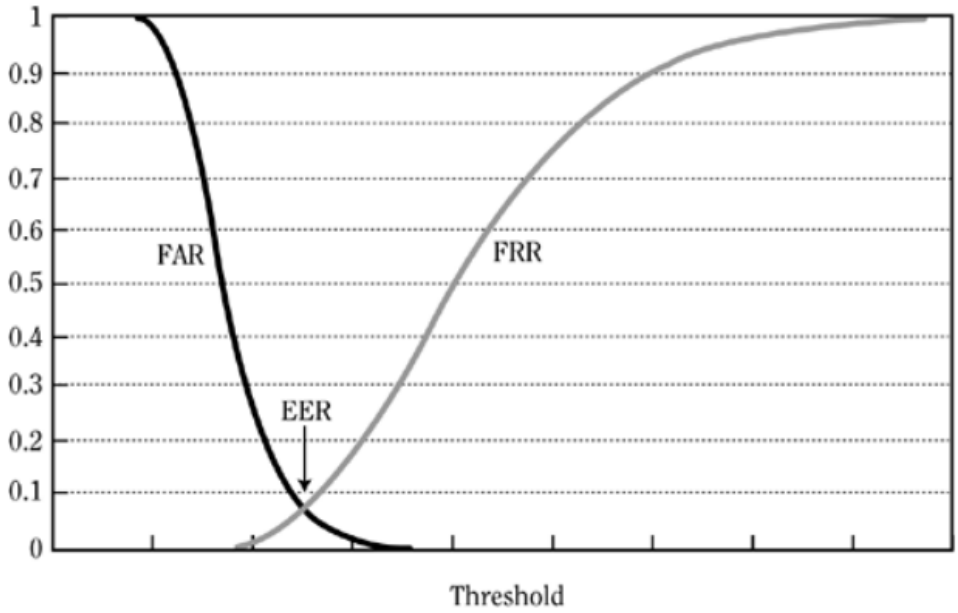


Figure 2.17. FAR, FRR and EER

14. Mel Frequency Cepstral Coefficients (MFCC)

Hệ số Cepstral Tần Số Mel (MFCCs):

- **Định nghĩa:** MFCCs là một phương pháp trích xuất đặc trưng cho tín hiệu âm thanh, thường được sử dụng trong xử lý tín hiệu giọng nói và âm thanh.
- **Bước Tính Toán:**
  1. **Tiền-emphasis:** Tín hiệu âm thanh trải qua tiền-emphasis để tăng cường các tần số cao, thường thông qua bộ lọc high-pass bậc một.
  2. **Chia khối theo khung (Frame Blocking):** Tín hiệu được tiền-emphasis được chia thành các khung ngắn, chồng lên nhau, thường kéo dài từ 20-40 mili giây. Điều này giúp phân tích các đặc tính phổ qua thời gian.
  3. **Tạo cửa sổ (Windowing):** Mỗi khung được nhân với một hàm cửa sổ, như cửa sổ Hamming, để giảm rò rỉ phổ và giảm thiểu hiện tượng nghệ thuật ở mép khung.
  4. **Biến đổi Fourier Nhanh (FFT):** Biến đổi Fourier được áp dụng cho mỗi khung đã tạo cửa sổ để chuyển đổi từ miền thời gian sang miền tần số. Điều này tạo ra một phổ công suất biểu thị sự phân phối công suất qua các tần số khác nhau.
  5. **Bộ lọc Mel (Mel Filterbank):** Phổ công suất được chuyển qua một bộ lọc trên thang Mel, một thang thị giác dựa trên âm thanh của người. Bước này mô phỏng sự nhạy cảm của hệ thống thính giác của con người đối với các tần số khác nhau. Mỗi bộ lọc trong bộ lọc là hình tam giác, với đáp ứng bằng 1 tại tần số trung tâm, giảm dần theo hình tuyến tính đến 0 cho đến khi đạt đến tần số trung tâm của hai bộ lọc kế tiếp.

- **Ý nghĩa:** MFCCs là hiệu quả trong việc thu thập các đặc trưng quan trọng của tín hiệu âm thanh cho các nhiệm vụ như nhận dạng giọng nói. Chúng cung cấp một biểu diễn ngắn gọn của các đặc tính phổ, nhấn mạnh thông tin quan trọng theo cảm nhận và cho phép xử lý hiệu quả trong các ứng dụng liên quan đến phân tích âm thanh.

15. K-means clustering

K-means Clustering:

- **Loại thuật toán:** K-means clustering là một thuật toán học máy không giám sát.
- **Mục tiêu:** Phân chia n quan sát thành k cụm, trong đó mỗi quan sát thuộc về cụm có trung bình gần nhất (trung tâm cụm hoặc trọng tâm cụm).
- **Ứng dụng:** Sử dụng để tạo ra các cụm dữ liệu và dự đoán cụm cho các quan sát mới.
- **Quy trình:**
  1. Chọn k trung bình ngẫu nhiên ban đầu.
  2. Gán từng quan sát vào cụm có trung bình gần nhất.
  3. Cập nhật trung bình cho mỗi cụm dựa trên quan sát trong cụm đó.
  4. Lặp lại bước 2 và 3 cho đến khi sự hội tụ (kết thúc).
- **Ưu điểm:** Dễ hiểu, thực hiện nhanh chóng, phù hợp cho dữ liệu lớn.
- **Nhược điểm:** Nhạy cảm với giá trị trung bình ban đầu, không đảm bảo tìm ra kết quả tối ưu toàn cầu.
- **Ứng dụng thực tế:** Phân loại hình ảnh, phân khúc thị trường, phân loại văn bản,...
- **Chú ý:** Kết quả có thể thay đổi dựa trên lựa chọn ban đầu của k và giá trị trung bình.

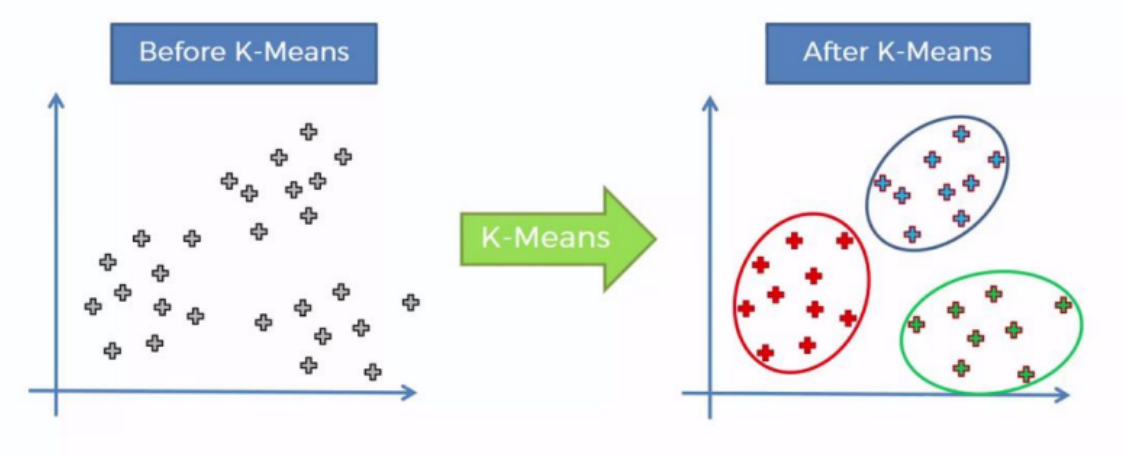


Figure 2.22 K-means clustering

16. Proposed Local Equal Error Rate Threshold (EER Threshold) per speaker approach

- Một ngưỡng cố định có thể không phản ánh đúng sự phân tán khác nhau của vectơ nhúng từ các người nói khác nhau.
- Hiểu nôm na: Với mỗi người, sẽ thực hiện 1000 lần lấy bộ triplet với anchor và positive là chính họ, còn negative là người khác. Rồi tối ưu bằng cách tìm điểm bằng nhau giữa FAR và FRR.
- Mỗi người sẽ có 1 EER khác nhau

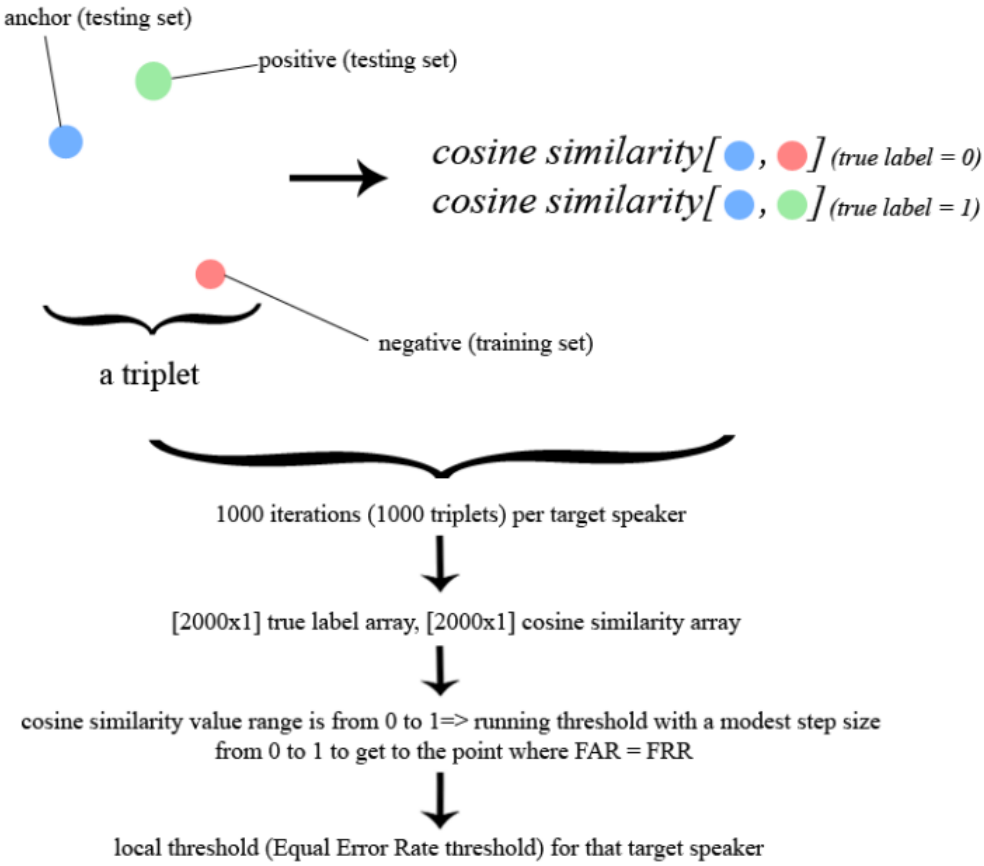


Figure 2.25. Process of setting a local similarity threshold

17. Technology

- Pytorch: PyTorch là một framework học sâu có đầy đủ tính năng, thường được sử dụng cho nhận diện hình ảnh và xử lý ngôn ngữ. Viết bằng Python, PyTorch hỗ trợ tốt GPU và sử dụng auto-differentiation giúp sửa đổi đồ thị tính toán ngay lập tức. Nó được phát triển bởi Facebook AI Research, có backend từ Torch và frontend Python, tập trung vào việc thử nghiệm nhanh và prototyping. Được giới thiệu vào năm 2017, PyTorch được ưa chuộng với các nhà phát triển machine learning.

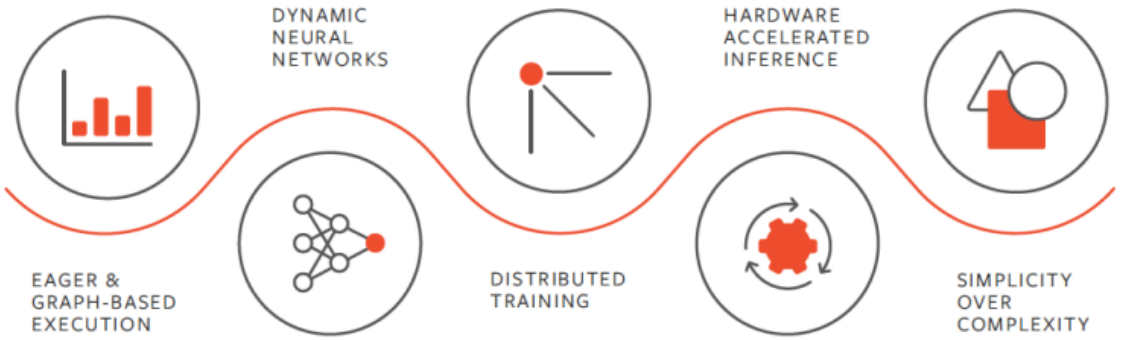


Figure 2.26 Pytorch features

- Librosa: Librosa là một thư viện Python quan trọng dành cho phân tích âm nhạc và âm thanh, giúp các nhà phát triển xây dựng ứng dụng làm việc với định dạng tệp âm thanh và âm nhạc bằng Python. Thư viện này thường được sử dụng khi làm việc với dữ liệu âm thanh, như trong việc tạo âm nhạc (sử dụng LSTM) hay nhận dạng giọng nói tự động.
- Django và DRF: ...
- Custom Tkinter: CustomTkinter là một mở rộng của module Tkinter trong Python, mang đến các yếu tố giao diện người dùng (UI) phong phú hơn so với Tkinter thông thường. Có thể tùy chỉnh các yếu tố như nút để thêm hình ảnh, làm tròn các cạnh, thêm viền, và nhiều tùy chọn khác.

18. Training data proprocessing

- Trong mỗi bước của quá trình huấn luyện, một lô dữ liệu với kích thước lô bằng 8 sẽ được đưa vào các mô hình. Mỗi trong tám phần tử đầu vào này là một bộ ba gồm các đặc trưng của anchor, positive và negative. Các đặc trưng này đã được cắt tỉa để tuân thủ với chiều dài chuỗi được chọn cho các mô hình sử dụng.
-

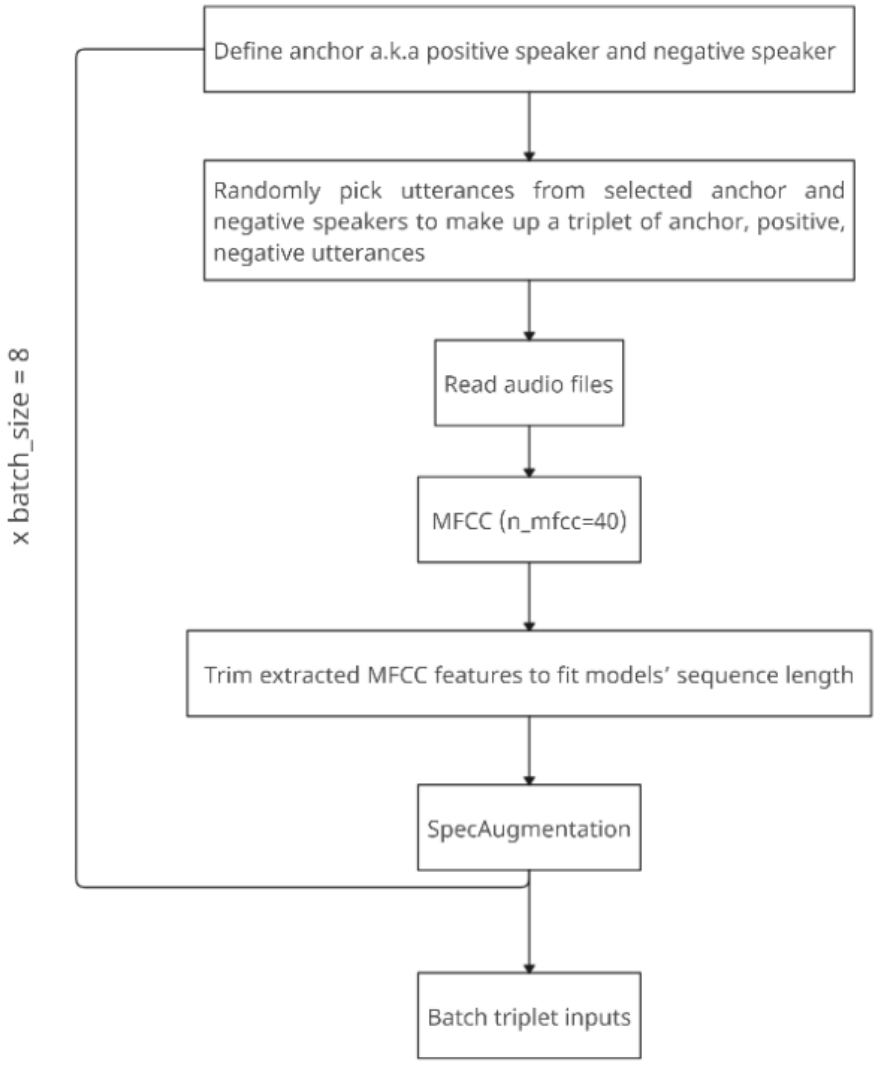


Figure 4.1 Block diagram of data preprocessing pipeline (Training stage)

1. **Định nghĩa Anchor và Speaker**: Đầu tiên, bạn chọn các speaker sẽ đóng vai trò là anchor (điểm neo) cũng như là positive (cùng người nói với anchor) và negative speaker (người nói khác).
2. **Chọn Utterances Ngẫu Nhiên**: Từ mỗi speaker đã chọn, bạn chọn các utterances (đoạn nói) một cách ngẫu nhiên để tạo nên một triplet, bao gồm một utterance của anchor, một utterance tích cực (cùng người nói với anchor), và một utterance tiêu cực (người nói khác).
3. **Đọc File Âm thanh**: Sau đó, hệ thống sẽ đọc các file âm thanh tương ứng với các utterances đã chọn.
4. **Trích xuất Tính năng MFCC**: Sử dụng Mel-Frequency Cepstral Coefficients (MFCC) để trích xuất các đặc trưng từ các file âm thanh. Số lượng coefficients được đặt là 40 (n_mfcc=40), đây là một giá trị phổ biến được sử dụng để trích xuất thông tin âm thanh quan trọng.
5. **Cắt Tính năng MFCC**: Tính năng MFCC sau khi được trích xuất có thể cần phải được cắt bớt hoặc padding để phù hợp với chiều dài chuỗi của mô hình mà bạn đang huấn luyện.

6. **SpecAugmentation**: Áp dụng SpecAugmentation, một kỹ thuật tăng cường dữ liệu âm thanh, để làm cho mô hình mạnh mẽ hơn với các biến thể âm thanh và giảm overfitting. Điều này thường bao gồm việc thay đổi ngẫu nhiên các đặc trưng về mặt thời gian và tần số để tạo ra sự đa dạng trong dữ liệu.
7. **Tạo Batches Triplet**: Cuối cùng, các triplets được tổ chức thành các batches để đưa vào mô hình huấn luyện. Một batch có kích thước x được đề cập ở góc trái trên cùng của sơ đồ, nghĩa là mỗi batch sẽ chứa x triplets để xử lý cùng một lúc trong quá trình huấn luyện.

Trước khi tiến hành huấn luyện mô hình, quá trình tiền xử lý dữ liệu được thực hiện trong sáu bước liên tiếp. Kết quả cuối cùng của quy trình này là một lô dữ liệu chứa các bộ ba được ghép từ tám (kích thước lô). Mỗi bộ ba bao gồm ba tệp âm thanh khác nhau: hai trong số đó do người nói làm anchor, và người nói tiêu cực nói cái còn lại.

Trước đó, từ tập dữ liệu mục tiêu (trong trường hợp này là tập dữ liệu LibriSpeech), tác giả tạo ra một từ điển, trong đó khóa là danh tính của người nói (duy nhất cho mỗi người nói), và giá trị tương ứng là đường dẫn thư mục lưu trữ tất cả các đoạn nói của người đó. Dựa trên danh sách này của tất cả các người nói trong tập huấn luyện, hai người nói được chọn ngẫu nhiên để làm anchor/positive speaker và negative speaker trong một bộ ba. Bây giờ có thể có được một bộ ba, được hình thành từ ba đường dẫn tệp của các đoạn nói từ người nói anchor và người nói tiêu cực. Tiếp theo trong quy trình tiền xử lý dữ liệu được đề xuất này, tác giả sẽ đọc tín hiệu âm thanh trong bộ ba đó và chuyển đổi chúng từ định dạng analog sang kỹ thuật số với tần số lấy mẫu là 16kHz

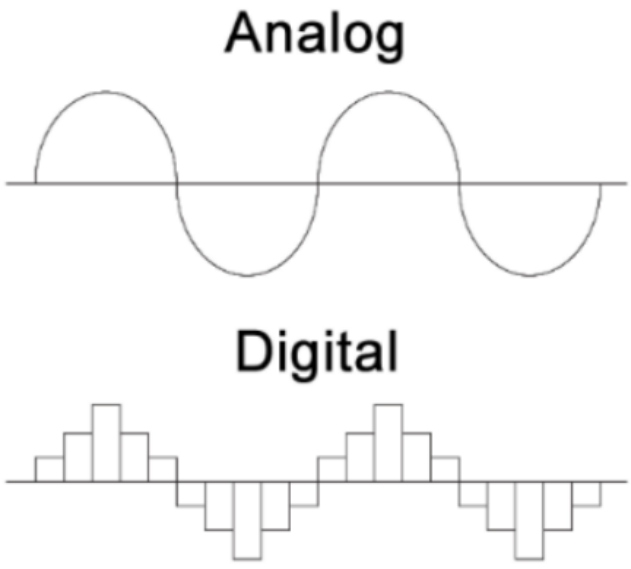


Figure 4.2 Analog signal to digital signal

Tác giả thực hiện quy trình tiền xử lý dữ liệu như sau:

1. **Tạo từ điển (dictionary) của dataset:** Tác giả tạo một từ điển trong đó khóa (key) là ID của người nói (unique cho mỗi người nói), và giá trị tương ứng là đường dẫn thư mục chứa tất cả các đoạn nói của

người đó.

2. **Chọn ngẫu nhiên 2 người nói làm người nói chính (anchor/positive) và người nói phụ (negative):** Tác giả chọn ngẫu nhiên 2 người nói từ danh sách các người nói trong tập huấn luyện để tạo một bộ ba. Ba tệp âm thanh tương ứng với người nói chính và người nói phụ được kết hợp để tạo thành một bộ ba.
3. **Đọc tín hiệu âm thanh và chuyển đổi thành định dạng số:**
   - Tác giả đọc tín hiệu âm thanh trong bộ ba.
   - Chuyển đổi tín hiệu từ dạng analog sang dạng số với tần số lấy mẫu là 16kHz.
4. **Trích xuất MFCCs:**
   - Sử dụng thư viện Librosa, tác giả trích xuất các hệ số cepstral Mel-frequency (MFCCs) từ tín hiệu âm thanh.
   - Đặt `n_mfcc` là 40, tức là sẽ có 40 hệ số MFCC tương ứng với mỗi cửa sổ.
5. **Chuẩn hóa độ dài chuỗi đầu vào:**
   - Đặt độ dài chuỗi đầu vào là 100. Mỗi tệp âm thanh được biến đổi thành ma trận kích thước 40×100.
   - Nếu kích thước ban đầu vượt quá 100, cắt đi phần dư. Nếu ngắn hơn 100, bỏ qua bộ ba hiện tại và chọn lại một bộ ba mới.
6. **SpecAugmentation:**
   - Áp dụng thao tác che phủ thời gian và tần số (SpecAugmentation) lên các đặc trưng đã cắt của bộ ba.
   - 30% dữ liệu huấn luyện được tăng cường.
7. **Lặp lại bước 2-6 cho số lô lần (batch size) cần thiết (8 trong trường hợp này):**
   - Lặp lại quy trình trên để có được tất cả các bộ ba và tạo thành một đầu vào đóng gói là ma trận kích thước (8 × 3) x 40 × 100.

Những bước trên làm phong phú đối với dữ liệu và chuẩn bị đầu vào cho việc huấn luyện mô hình nhận dạng giọng nói.

19. Training models

- **Mục Tiêu:**
  - Trích xuất các đặc điểm quan trọng nhất của giọng nói để phân biệt giọng nói của mỗi người.
- **Kiến Trúc Mô Hình:**
  - Sử dụng Long Short-term Memory (LSTM) và Transformer làm cơ sở chính.
  - Thực hiện thí nghiệm thủ công trên các biến thể của LSTM và Transformer để chọn ra mô hình tối ưu nhất.
- **Đánh Giá Mô Hình:**
  - Mỗi mô hình đã được huấn luyện ít nhất 50,000 bước trước khi đánh giá.

- Các mô hình được mô tả dưới đây là những mô hình cho hiệu suất tốt nhất trong các giai đoạn đánh giá.

- **Tên Gọi Các Mô Hình:**
  - Vì mục đích so sánh giữa các mô hình đã huấn luyện, tác giả gọi chúng là Jennie, Jisoo, Lisa, Rosé, Momo và Tzuyu.

20. Models

1. **Jennie (LSTM-based):**
   - LSTM-based model.
   - 1 LSTM neural network with a hidden size of 64 and 3 stacked layers.
   - Trained for 100,000 steps.
   - Total parameters: 252,928.
   - Sử dụng các kỹ thuật như LSTM, ReLU activation, và concatenate layer.
2. **Jisoo (LSTM-based):**
   - Giống hệt kiến trúc của Jennie.
   - Huấn luyện trong 50,000 bước.
   - Mục đích để so sánh với Jennie.
3. **Lisa (LSTM-based):**
   - LSTM-based model với 2 LSTM neural networks.
   - Mỗi LSTM có hidden size là 64, 3 layers, và bidirectional.
   - Có 2 linear layers.
   - Huấn luyện trong 100,000 bước.
   - Total parameters: 583,936.
4. **Rosé (LSTM-based):**
   - Mô hình đơn giản hóa của Lisa.
   - Giữ nguyên route thứ hai và thứ ba để kết hợp vào output cuối cùng.
   - Huấn luyện trong 100,000 bước.
   - Total parameters: 550,912.
5. **Momo (Transformer - Encoder-Decoder):**
   - Mô hình Transformer sử dụng kiến trúc Encoder-Decoder.
   - Chiều của các lớp transformer là 32.
   - Trong encoder, có 2 lớp transformer với số lượng heads là 8.
   - Huấn luyện trong 100,000 bước.
   - Total parameters: 418,112.

6. **Tzuyu (Transformer - Encoder-Decoder):**
    - Giống hệt kiến trúc của Momo.
    - Được huấn luyện lâu hơn với 200,000 bước.

21. Loss function

Trong giai đoạn huấn luyện, tác giả đã sử dụng triplet loss để điều chỉnh các mô hình. Công thức của triplet loss đã được đề cập trước đó ở chương 2, và đây là triển khai bằng Python của triplet loss cho một triplet cụ thể, với siêu tham số alpha được đặt là 0.1:

```
pythonCopy code
def my_triplet_loss(anchor, positive, negative):
    cos = nn.CosineSimilarity(dim=-1, eps=1e-6)
    return torch.max(cos(anchor, negative) - cos(anchor, positive) + 0.1, tor
ch.tensor(0.0).to('cpu'))
```

Một hàm mất mát là một hàm so sánh giữa giá trị đầu ra dự đoán và mục tiêu; đo lường khả năng mô hình mạng neural mô hình dữ liệu huấn luyện. Trong quá trình huấn luyện, mục tiêu là giảm thiểu mất mát này giữa đầu ra dự đoán và mục tiêu bằng cách lấy đạo hàm, tối ưu hóa trọng số để đưa hàm mất mát về giá trị tối thiểu của nó.

22. Evaluate trained models

1. **Đánh Giá Mô Hình:**
    - Tác giả sử dụng Equal Error Rate (EER) làm đánh giá mô hình, là một phép đo độ đồng đều không phụ thuộc vào ngưỡng.
    - 10,000 triplet ngẫu nhiên được lấy mẫu và đánh giá trên mô hình, tạo ra 20,000 cặp nhãn và điểm số.
    - EER được sử dụng để xác định ngưỡng mục tiêu, nơi False Acceptance Rate (FAR) bằng False Rejection Rate (FRR).
2. **Quy Trình Huấn Luyện:**
    - Quy trình huấn luyện gồm ba bước chính: tiền xử lý dữ liệu, huấn luyện, đánh giá.
3. **Kết Quả Huấn Luyện:**
    - Hiển thị biểu đồ mất mát của 6 mô hình (Jennie, Jisoo, Lisa, Rosé, Momo, Tzuyu).
    - Tóm tắt kết quả huấn luyện trong Bảng 4.1:

Tên Mô Hình	Kiến Trúc Cơ Bản	Kích Thước Nhúng	EER	Ngưỡng EER	Bước Huấn Luyện

Jennie	LSTM	[128×1]	0.071	0.577	100,000
Jisoo	LSTM	[128×1]	0.08115	0.666	50,000
Lisa	LSTM, Feedforward NN	[128×1]	0.0839	0.59	100,000
Rosé	LSTM	[128×1]	0.0836	0.608	100,000
Momo	Transformer	[32×1]	0.1008	0.79	100,000
Tzuyu	Transformer	[32×1]	0.0957	0.698	200,000

4. **Kết Luận:**
   - Jennie là mô hình xuất sắc nhất với EER thấp nhất là 0.071 so với các mô hình khác.

23. Enrollment stage

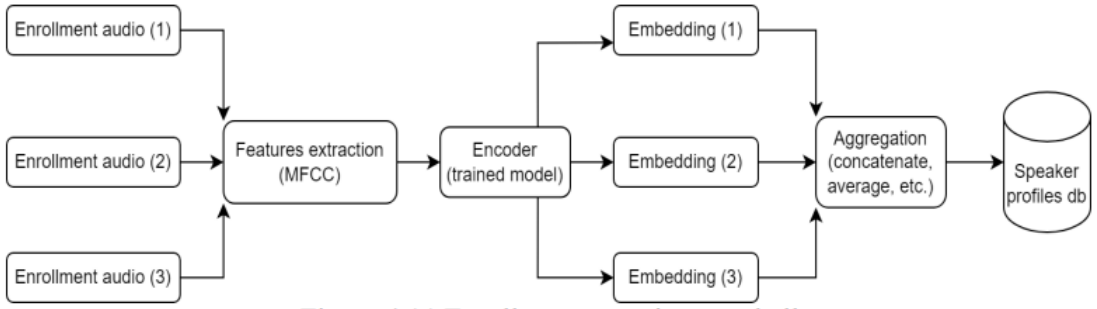


Figure 4.11 Enrollment stage in a nutshell

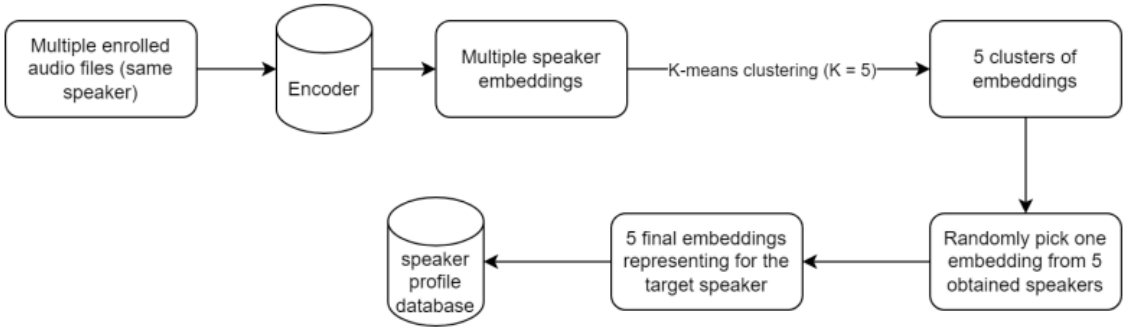


Figure 4.12 Using K-means clustering to compute speaker profile

Giai đoạn Đăng Ký (Enrollment) (Hình 4.11):

1. **Thu Thập Âm thanh Đăng ký**: Mỗi người nói cung cấp nhiều đoạn phát ngôn (utterances), và mỗi đoạn phát ngôn này được thu thập dưới dạng file âm thanh.
2. **Trích Xuất Đặc Trưng**: Tính năng MFCC được trích xuất từ mỗi file âm thanh. MFCC là một bộ đặc trưng phổ biến để phân tích và hiểu thông tin âm thanh.

3. **Mã hóa Đặc Trưng**: Một mô hình đã được huấn luyện (encoder) nhận đầu vào là các đặc trưng MFCC và biến chúng thành các vector nhúng (embeddings).
4. **Tổng hợp**: Các vector nhúng này sau đó được tổng hợp lại (ví dụ, thông qua việc nối các vector, lấy trung bình, v.v.) để tạo thành một hồ sơ người nói duy nhất sẽ được lưu trữ trong cơ sở dữ liệu hồ sơ người nói.

Sử dụng K-means để Tính Toán Hồ sơ Người Nói (Hình 4.12):

1. **Nhiều File Âm thanh**: Có nhiều file âm thanh đăng ký cho mỗi người nói.
2. **Mã hóa**: Mỗi file âm thanh được mã hóa thành một vector nhúng.
3. **K-means Clustering**: Sử dụng thuật toán K-means để phân cụm các vector nhúng vào K (trong trường hợp này là 5) cụm.
4. **Chọn Vector Đại Diện**: Từ mỗi cụm, chọn một vector nhúng làm đại diện thay vì lấy trung bình của tất cả các vector trong cụm. Điều này để đảm bảo vector được chọn vẫn giữ được đặc tính độc đáo của người nói, vì việc lấy trung bình có thể làm mất đi thông tin quan trọng do nội dung nói có thể khác nhau giữa các vector nhúng.
5. **Hồ sơ Người Nói**: 5 vector nhúng cuối cùng này sau đó được dùng để định hình hồ sơ người nói cho hệ thống.
6. **Lưu Trữ**: Hồ sơ người nói này và ngưỡng EER cục bộ (nếu tính toán được) sẽ được lưu trữ trong cơ sở dữ liệu để sử dụng trong việc nhận dạng giọng nói sau này.

Ngoài ra, tác giả cũng đề cập rằng dù việc sử dụng ngưỡng EER cục bộ là khả thi, nhưng việc sử dụng một ngưỡng EER toàn cầu (global EER threshold) đã được chứng minh là đủ tốt trong các thí nghiệm thực tế của họ. Điều này có nghĩa là một ngưỡng đơn lẻ có thể được sử dụng cho tất cả người nói thay vì cần phải xác định một ngưỡng

24. Recognition stage:

Giai đoạn Nhận Dạng (4.3 Recognition Stage):

1. **Giai đoạn Nhận Dạng - Workflow (4.3.1 Recognition Stage Workflow):**
    - **Nhận Audio Runtime**: Âm thanh runtime (đoạn phát ngôn cần được nhận dạng) được đưa vào hệ thống.
    - **Trích Xuất Đặc Trưng**: Đặc trưng được trích xuất từ âm thanh runtime, giống như trong giai đoạn đăng ký, thường là MFCC.
    - **Mã hóa Đặc Trưng**: Mã hóa (Encoder) nhận đặc trưng và trả về một vector nhúng đại diện cho đặc trưng đó.
    - **So Sánh với Hồ sơ Đã Đăng Ký**: Vector nhúng nhận được sau đó được so sánh với các hồ sơ người nói đã đăng ký trước đó. Người nói mục tiêu được xác định là người có hồ sơ có điểm tương đồng cao nhất với âm thanh đầu vào.
    - **Sử Dụng Ngưỡng**: Có thể sử dụng ngưỡng cục bộ cho từng người nói hoặc một ngưỡng toàn cục trong quá trình nhận dạng.

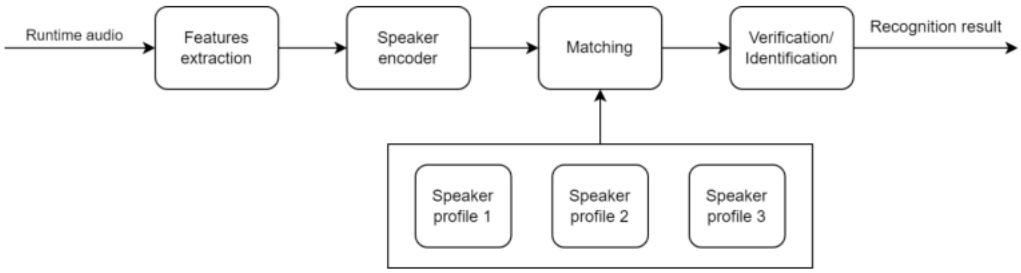


Figure 4.13 Workflow of Recognition stage

2. **Sử Dụng Cửa Sổ Trượt (4.14 Sliding Window Inference):**
    - Trái ngược với việc chỉ sử dụng một phần đặc trưng cắt giảm có độ dài được xác định trước, cả hai giai đoạn đăng ký và nhận dạng đều sử dụng cửa sổ trượt khi suy luận.
    - Cửa sổ trượt phân chia đặc trưng đã trích xuất của một file âm thanh thành các khung chồng lên nhau và đưa chúng qua bộ mã hóa.
    - Lấy trung bình tất cả các vector nhúng kết quả là đầu ra cuối cùng, đại diện cho tín hiệu đầu vào.

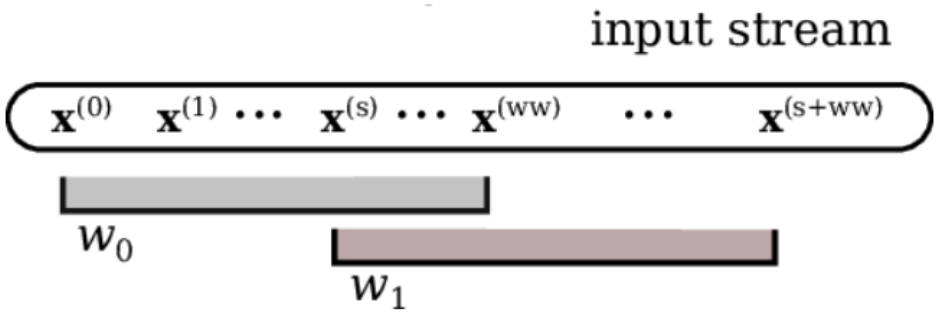


Figure 4.14 Sliding window inference

3. **Kết Quả Kiểm Thử Giai Đoạn Nhận Dạng (4.3.2 Recognition Stage Testing Result):**
    - Căn cứ vào kết quả đánh giá trong giai đoạn huấn luyện, "Jennie" là mô hình nổi bật nhất.
    - Trong phần này, mô hình "Jennie" được sử dụng để giải quyết vấn đề giữa việc sử dụng một ngưỡng toàn cục thông thường hay ngưỡng cục bộ cho từng người nói.
    - Đối với mỗi người nói trong tập kiểm thử, họ được coi là neo của 1000 triplet, phần tiêu cực của các triplet này được chọn ngẫu nhiên từ tập dữ liệu huấn luyện.
    - Thay vì sử dụng Tỷ Lệ Lỗi Bằng Nhau (Equal Error Rate - EER) để đánh giá hiệu suất mô hình như thường lệ, lần này, độ chính xác được sử dụng để đánh giá, làm cho kết quả đánh giá trở nên dễ hiểu và toàn diện hơn.
    - Bảng 4.2 cho thấy sự so sánh giữa ngưỡng toàn cục và ngưỡng cục bộ cho mỗi người:

Table 4.2 Comparison of global and local EER threshold

	Accuracy
Global threshold	92.81%
Local threshold per speaker	93.645%