HANOI UNIVERSITY OF SCIENCE AND TECHNOLOGY
SCHOOL OF INFORMATION AND COMMUNICATION TECHNOLOGY

———————*———————

GRADUATION THESIS

SUBMITTED IN PARTIAL FULFILLMENT
OF THE REQUIREMENTS FOR THE DEGREE OF

ENGINEER

IN

INFORMATION TECHNOLOGY

TODO TITLE

Author : **Vi Thanh Dat**
Class : CNTT2.02 K61
Student ID : 20164803

Supervisor : TODO supervisor

HANOI, 5 - 2021

Abstract

Placeholder

Tóm tắt

Placeholder

Summary

The thesis is divided into 5 chapters:

- Chapter 1 describes the problem of training neural networks at scale, as well as the object detection problem that would be used as a case study.
- Chapter 2 presents the theoretical background as well as related works, including machine learning, object detection and distributed training.
- Chapter 3 details the design and implementation of BK.Synapse - the proposed framework - and a case study where it is applied to an object detection problem.
- Chapter 4 reports on the results of several experiments and benchmarks.
- Chapter 5 presents our conclusions and lays out plans for future works.

Contents

List of Figures

List of Tables

Chapter 1

Giới thiệu về bài toán nhận dạng người nói

Trong chương này, tác giả giới thiệu tổng quan về bài toán nhận dạng người nói, điểm qua các nghiên cứu liên quan và giới thiệu về bài toán trong đồ án.

1.1 Giới thiệu chung về bài toán nhận dạng người nói

Nhận dạng người nói (speaker recognition) là quá trình tự động nhận dạng người đang nói bằng cách sử dụng thông tin riêng biệt của người nói đó có trong tín hiệu âm thanh. Nhận dạng người nói được ứng dụng rộng rãi trong nhiều lĩnh vực khác nhau, tuy nhiên cũng gặp không ít khó khăn khi triển khai trong thực tế. Do vậy, nghiên cứu bài toán nhận dạng người nói rất được quan tâm bởi nhiều nhà khoa học trên thế giới. Một số ứng dụng của bài toán có thể kể đến như:

- Bảo mật cho các hệ thống tài chính, ngân hàng: người dùng dùng giọng nói kết hợp với các lớp bảo mật khác cho xác thực để tăng tính bảo mật khi giao dịch.
- Tăng trải nghiệm khách hàng trong tổng đài chăm sóc khách hàng.
- Xác định danh tính tội phạm trong an ninh khi thu được dữ liệu giọng nói.
- Kết hợp với các hệ thống nhận dạng tiếng nói để xây dựng ứng dụng gỡ băng cuộc họp.
- ...

Dựa vào ứng dụng, nhận dạng người được phân loại thành nhận định người nói (speaker identitfication) và xác minh người nói (speaker verfication) (Hình 1.1). Trong nhận định người nói, một đoạn tiếng nói từ một người không xác định được phân tích và so sánh với mô hình giọng nói của những người đã biết. Người này được nhận định là người mô hình giọng nói phù hợp nhất với câu nói đầu vào. Trong xác minh người nói, một người lạ xác nhận một danh tính đã biết; đoạn tiếng nói của người này được so sánh với mô hình giọng nói của danh tính đang được xác nhận. Nếu điểm tương đồng đủ tốt, nghĩa là trên một ngưỡng nào đó, danh tính của người lạ được chấp nhận. Ngưỡng cao khiến những kẻ mạo danh khó được chấp nhận bởi hệ thống, nhưng có nguy từ chối nhầm người dùng hợp lệ. Ngược lại, ngưỡng thấp cho phép chấp nhận người dùng hợp lệ một cách nhất quán, nhưng có nguy cơ chấp nhận những người giả mạo.

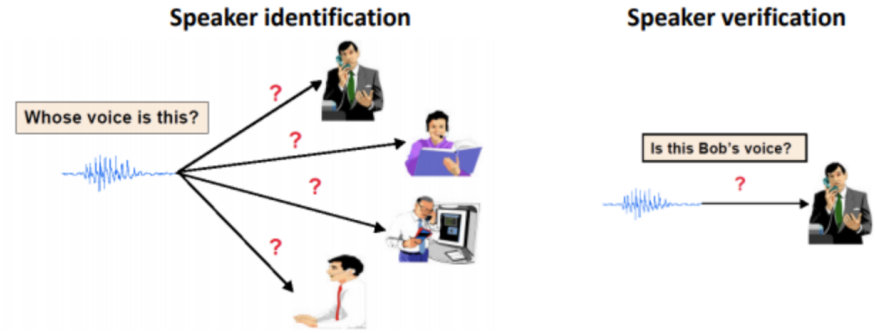


Figure 1.1: Nhận định người nói và xác minh người nói

Như mô tả trong hình 1.2, có ba pha trong quá trình nhận dạng người nói:

- Pha 1: phát triển (development). Trong pha này, mô hình có khả năng biểu diễn đặc trưng người nói được huấn luyện và tối ưu trên một cơ sở dữ liệu các đoạn tiếng nói.
- Pha 2: ghi danh (enrollment). Trong pha ghi danh, biểu diễn của người dùng mới được trích xuất bằng mô hình phát triển ở pha 1 và lưu trữ trong cơ sở dữ liệu để phục vụ cho pha 3.
- Pha 3: kiểm tra (testing). Với bài toán nhận định người nói, vec-tơ biểu diễn của câu nói đầu vào được so sánh với tất cả biểu diễn trong cơ sở dữ liệu để tìm ra người dùng tương đồng nhất. Trong bài toán xác thực người nói, được cung cấp danh tính đầu vào, hệ thống chỉ so sánh đoạn tiếng nói đầu vào và đoạn của danh tính trong hệ thống để đưa ra quyết định.

TODO Vẽ lại hình này và include development phase to make it more clear

Trong thực tế, hiệu năng của hệ thống nhận diện người nói bị suy giảm do sự

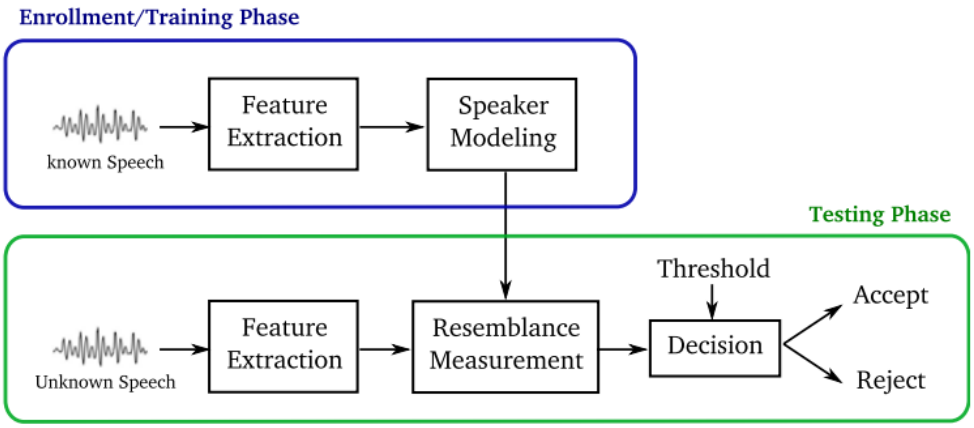


Figure 1.2: Tổng quan hệ thống nhận diện người nói

khác biệt của các kênh và phiên giữa tín hiệu giọng nói trong pha ghi danh và pha kiểm tra. Các yếu tố làm ảnh hưởng tới tín hiệu giọng nói bao gồm:

- Sử dụng các loại micrô khác nhau khi thu tín hiệu đăng ký và kiểm tra.
- Điều kiện tiếng ồn và độ vang của môi trường.
- Sự khác biệt trong giọng nói của người nói ở các giai đoạn khác nhau của độ tuổi, sức khoẻ, phong cách nói và trạng thái cảm xúc.
- Các kênh truyền như các loại điện thoại di động khác nhau, micrô, giao thức truyền âm thanh qua internet có thể làm thay đổi giọng nói.

Các phương pháp truyền thống sử dụng Gaussian Mixture Model TODO Đoạn này viết về dominated traditional approaches to overcome channel/session variabilities JFA, NAP, WCCN, LDA, PLDA

Dựa vào sự tương đồng của các câu nói đầu vào, phương pháp giải quyết bài toán nhận dạng người nói còn có thể chia thành nhóm dựa vào văn bản (text-dependent speaker recognition - TDSR) và nhóm không dựa vào văn bản(text-independent speaker recognition - TISR). Các hệ thống TDSR yêu cầu người nói cung cấp các đoạn tiếng nói có nội dung theo một từ hoặc câu được định sẵn; nội dung các câu nói phải được giữ nhất quán trong cả quá trình huấn luyện và nhận dạng. Ngược lại, TISR không yêu cầu người dùng phải thu theo bất cứ một văn bản nào. Với các đoạn tiếng nói ngắn, các hệ thống TDSR đã có thể đạt được hiệu suất nhận diện cao, trong khi các TISR yêu cầu các câu nói dài để huấn luyện các mô hình đáng tin cậy và đạt được hiệu suất tốt. Do sự bất tiện của các hệ thống TDSR, trong vài năm gần đây, cộng đồng nghiên cứu tập trung phát triển các mô hình học sâu end-to-end nhằm mục đích biểu diễn các đoạn tiếng nói ngắn.

1.2 Các nghiên cứu học sâu giải quyết bài toán nhận diện giọng nói

Các mô hình học sâu end-to-end hiện đại cho bài toán nhận diện giọng nói thường gồm ba phần chính (1.3):

- Một mạng nơ-ron làm công cụ trích xuất biểu diễn người nói cho một khung đặc trưng tiếng nói.
- Lớp tổng hợp dữ liệu: lớp này sử dụng biểu diễn của các khung âm thanh từ mạng nơ-ron để tổng hợp ra một vec-tơ duy nhất đại diện cho đoạn tiếng nói đầu vào.
- Một hàm mất mát để tối ưu toàn bộ hệ thống.

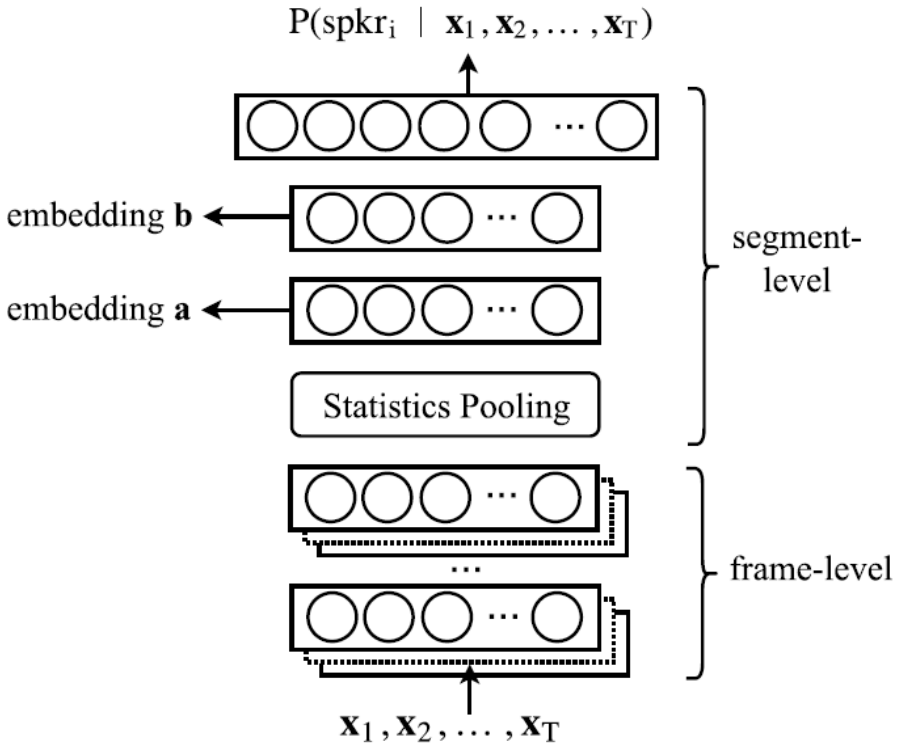


Figure 1.3: Tổng quan hệ thống nhận diện người nói [30]

Trong pha kiểm tra, mạng nơ-ron và lớp tổng hợp dữ liệu được sử dụng để tạo ra vec-tơ biểu diễn của đoạn tiếng nói kiểm tra. Hàm mất mát chỉ được sử dụng trong pha phát triển, và bị loại bỏ trong pha kiểm tra và pha ghi danh.

Phần lớn các nghiên cứu về nhận dạng người nói hướng tới cải thiện hệ thống qua nâng cao hiệu quả của lớp tổng hợp dữ liệu và hàm mất mát. Về mạng nơ-ron trích xuất đặc trưng, các nghiên cứu chủ yêu sử dụng các mạng xương

sống thành công trong các bài toán khác như phân loại hình ảnh (Ví dụ: mạng VGG [27] và mạng ResNet [11]) và nhận dạng tiếng nói (mạng TDNN [34] và mạng LSTM [12]).

Một vài nghiên cứu nổi bật cho lớp tổng hợp có thể kể đến như: tổng hợp trung bình (average pooling) [32], tổng hợp thống kê (statistical pooling) [30], tổng hợp dựa trên cơ chế tập trung (attentive pooling) [21, 42], hay mã hoá dựa trên từ điển NetVLAD/GhostVLAD trong [38].

Các nghiên cứu tiên phong [19, 28, 30, 32] ứng dụng mạng nơ-ron nhận tạo vào bài toán nhận dạng người nói học không gian biểu diễn bằng các hàm mất mát phân loại (classification loss). Trên cơ sở đó, các nghiên cứu thịnh hành [21, 23, 29] sử dụng hàm softmax để huấn luyện mô hình. Hàm softmax có khả năng học để phân loại người nói một cách hiêu quả, tuy nhiên, nó không được thiết kế để tối ưu tính tương đồng trong không gian embedding.

Để giải quyết điểm yếu của softmax, công trình của Liu và cộng sự [17] đề xuất hàm angular softmax (A-Softmax) cho bài toán nhận diện khuôn mặt bằng cách dùng độ tương đồng cô-sin làm đầu vào của hàm softmax. Các nghiên cứu sau đó áp dụng hàm A-Softmax trên bài toán nhận dạng người nói [31, 33] cho thấy sự vượt trội của hàm này so với softmax thông thường. Các phiên bản cải tiến của hàm A-Softmax như AM-Softmax [35] và AAM-Softmax [10] sử dụng hàm phạt biên trở nên phổ biến cho bài toán nhận dạng người nói do dễ dàng cài đặt và hiệu năng cao. Tuy nhiên, huấn luyện mô hình với AM-Softmax và AAM-Softmax khá khó khăn do hai hàm này rất nhạy cảm với sự điều chỉnh tham số.

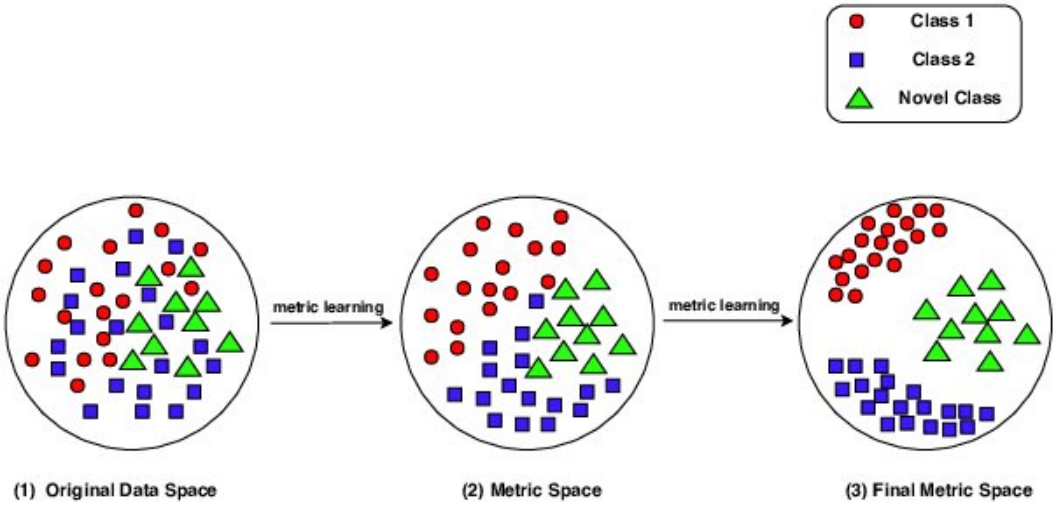


Figure 1.4: Không gian embedding với học biểu diễn[37]

Bên cạnh nhóm hàm mục tiêu phân loại, học biểu diễn (Representation learning

hay Metric learning) cũng là lựa chọn phổ biến cho bài toán nhận dạng người nói. Thay vì phân loại, mục tiêu của học biểu diễn là tối ưu không gian embedding 1.4. Trong bài toán nhận dạng người nói, biểu diễn của các câu nói từ cùng một người được kéo lại gần nhau trong không gian embedding, ngược lại câu của những người khác nhau bị đẩy ra xa.

Hai hàm học biểu diễn nổi tiếng là contrastive loss [7] và triplet loss [25] xuất phát từ bài toán nhận diện khuôn mặt cho kết quả hứa hẹn trong bài toán nhận diện người nói [22, 41]. Năm 2020, Chung và cộng sự [8] đề xuất hàm Angular Prototypical cho kết quả vượt trội khi so với các hàm mục tiêu phân loại.

1.3 Ngôn ngữ tài nguyên thấp

Ngôn ngữ tài nguyên thấp có thể hiểu là các ngôn ngữ

Chapter 2

Cơ sở lý thuyết

Chương 2 trình bày cơ sở lý thuyết về học máy, học sâu cùng TODO....

2.1 Học máy

2.1.1 Tổng quan về học máy

Học máy (Machine learning - ML) là một nhánh con của trí tuệ nhân tạo. Nghiên cứu học máy tập trung phát triển và xây dựng các thuật toán và kỹ thuật để giúp chương trình máy tính có thể "học" thi thực một tác nào đó từ kinh nghiệm. Trong [18], nhà tiên phong về học máy Tom M. Mitchell định nghĩa như sau: Một chương trình máy tính được nói là học từ kinh nghiệm E để thực thi một tác vụ T nếu khả năng của chương trình được đo bằng độ đo P tiến bộ với kinh nghiệm E. Rõ ràng hơn, học máy là những chương trình máy tính có thể tự học được dữ liệu mà không cần được lập trình một cách cụ thể.

Với các cách định nghĩa T, P, và E khác nhau, các thuật toán học máy có thể được phân vào các nhóm: học có giám sát, học không giám sát, học bán giám sát, và học tăng cường.

Học có giám sát là lớp các thuật toán sử dụng dữ liệu được gán nhãn từ trước để tìm mối liên hệ giữa đầu vào và đầu ra. Mục tiêu của học có giám sát là tạo ra mô hình có thể dự đoán được đầu ra cho các đầu vào mới mà mô hình chưa gặp bao giờ. Tập dữ liệu gán nhãn trên được gọi là dữ liệu huấn luyện. Cho tập dữ liệu huấn luyện gồm N ví dụ $D = \{(\boldsymbol{x}_1, y_1), ..., (\boldsymbol{x}_N, y_N)\}$ trong đó $\boldsymbol{x}_i$ là vec-tơ đặc trưng của ví dụ tứ i và y_i là nhãn tương ứng. Thuật toán học có giám sát tìm hàm ánh xạ $f : X \longrightarrow Y$ trong đó X là không gian vec-tơ đầu vào và Y là tập các nhãn.

Các thuật toán học có giảm sát thường được sử dụng để giải quyết hai loại bài toán: phân loại (classification) và hồi quy (regression). Bài toán phân loại có nhãn rời rạc thuộc một tập cho sẵn. Ví dụ: phân loại đồ vật, phân loại phương tiện giao thông, phân loại giới tính dựa trên giọng nói. Khác với bài toán phân loại, bài toán hồi quy lại có nhãn nằm trên một miền liên tục. Ví dụ: dự đoán giá nhà đất, dự đoán thị trường chứng khoán.

Trong **học không giám sát**, khác với học có giám sát, ta chỉ có dữ liệu đầu vào mà không có dữ liệu đầu ra. Mục tiêu chính của các thuật toán là tìm ra cấu trúc ẩn trong dữ liệu hoặc trích xuất đặc trưng chung của tập dữ liệu.

Học bán giám sát nằm giữa học có giám sát và không có giám sát. Học bán giám sát thường được sử dụng khi có một lượng lớn dữ liệu không có nhãn và một số ít dữ liệu có nhãn. Phương pháp học bán giám sát nổi tiếng nhất - tự huấn luyện (self-training) sử dụng mô hình huấn luyện trên tập có nhãn để sinh nhãn giả từ tập không nhãn nhằm tăng cường khả năng học của mô hình.

Hệ thống **học củng cố** được xem như là một tác tử trong một môi trường vô định và phức tạp. Với mỗi hành động, tác tử này nhận điểm thưởng hoặc phạt từ môi trường. Mục tiêu của học củng cố là các tác tử thông minh để tối đa điểm thưởng và thực hiện tác vụ của nó. Học củng cố hiện đang là lĩnh vực nghiên cứu tập trung nhiều nguồn lực và công sức với nhiều ứng dụng thực tế liên quan tới điều khiển robot hay vận hành nhà máy một cách tự động.

2.1.2 Mạng nơ-ron nhân tạo

Mạng nơ-ron nhân tạo (Artificial neural networks - ANN) là một mô hình tính toán mô phỏng cách nơ-ron trong não người phân tích và xử lý thông tin. Học sâu với nền tảng là ANN cho phép xử lý dữ liệu và thông tin với độ chính xác vượt xa các mô hình xác suất cổ điển và tốc độ vượt trội so với con người.

Nơ-ron sinh học gồm 3 phần chính: dendrite, thân tế bào (soma) và axon (Hình 2.2). Đầu tiên, tín hiệu đầu vào được thu thập bởi các khớp thần kinh của dendrite. Vai trò của soma là xử lý đầu vào và tổng hợp thông tin dựa vào độ quan trọng của tín hiệu. Sau đó, axon sẽ truyền thông tin đã được xử lý tới đầu ra. Thành phần cấu thành nên ANN là perceptron cũng có cách thức hoạt động tương tự. Perceptron bao gồm lớp đầu vào, tập trọng số cùng hàm kích hoạt để tổng hợp thông tin và lớp đầu ra, tương tự như dendrite, soma và axon.

[1]https://app.biorender.com/biorender-templates/t-5f5b7e6139954000b2bde860-neuron-anatomy

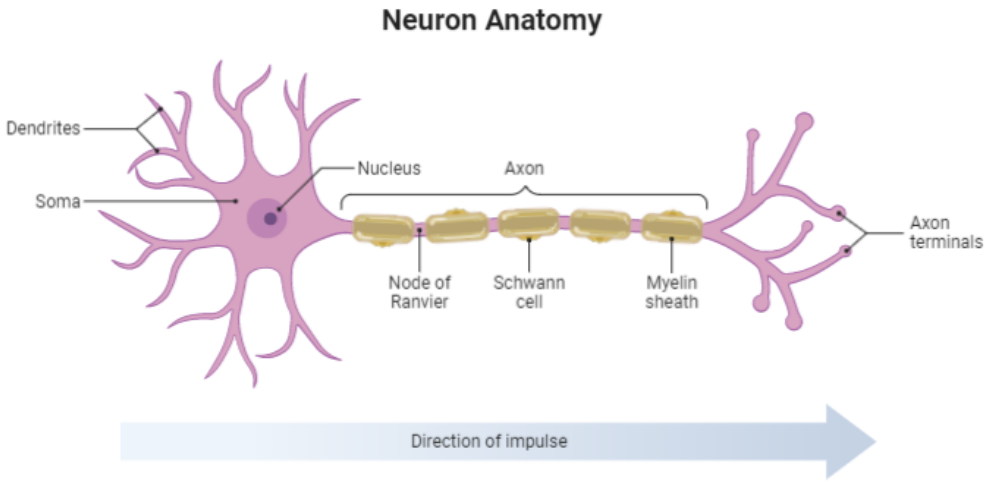


Figure 2.1: Cấu tạo nơ-ron sinh học [1]

Mạng nơ-ron là mô hình gồm nhiều lớp chồng lên nhau, mỗi lớp bao gồm nhiều perceptron (hay nơ-ron) tổng hợp thông tin từ lớp trước, mỗi lớp có một tập các nơ-ron độc lập. Lớp đầu tiên trong mạng được gọi là lớp đầu vào (input layer), lớp cuối cùng được gọi là lớp đầu ra (output layer), các lớp ở giữa được gọi là lớp ẩn (hidden layer) (Hình 2.2).

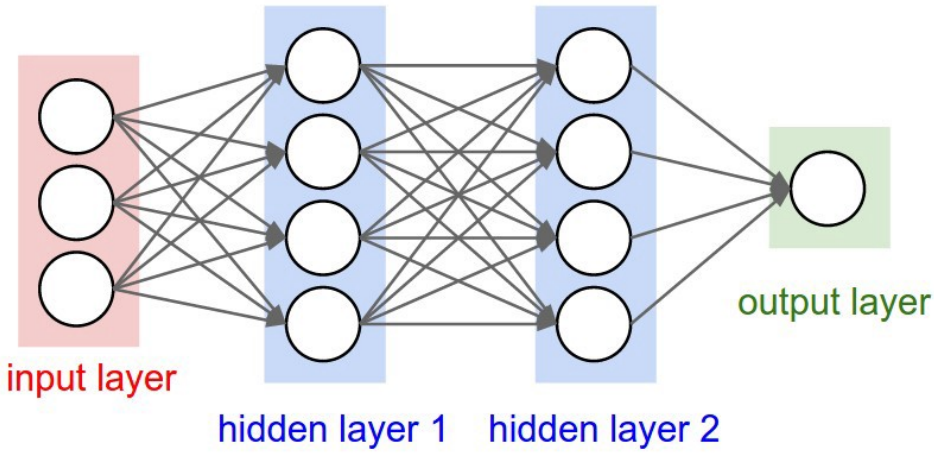


Figure 2.2: Cấu trúc của mạng nơ-ron nhân tạo[2]

Cho một mạng nơ-ron nhân tạo gồm L lớp, với $\boldsymbol{a}^i$ là tập các nơ-ron trong lớp thứ i, hai lớp nơ-ron liên tiếp có chỉ số i và $i+1$ được kết nối với nhau bằng một ma trận trọng số $\boldsymbol{W}^i$ và một vec-tơ bias $\boldsymbol{b}^i$. Ta có thể mô tả quá trình tính toán các nơ-ron từ lớp đầu vào và đầu ra dưới dạng toán học như sau:

$$\boldsymbol{a}^{(i)} = \sigma(\boldsymbol{W}^i \boldsymbol{a}^{(i-1)} + \boldsymbol{b}^i),\ 0 \leq i \leq L \tag{2.1}$$

[2]https://towardsdatascience.com/vanilla-neural-networks-in-r-43b028f415

Trong đó, σ được gọi là hàm kích hoạt (activation function). Hàm kích hoạt là một thành phần quan trọng không thể thiếu trong mạng nơ-ron nhân tạo. Để mạng nơ-ron nhân tạo có thể xấp xỉ hay học được những biến đổi phức tạp, σ phải là một hàm phi tuyến. Nếu không có hàm kích hoạt hay hàm kích hoạt là tuyến tính thì mạng nơ-ron dù có nhiều lớp đến đâu cũng có thể quy về một hàm tuyến tính và không có khả năng học được nhiều thông tin từ dữ liệu.

Có rất nhiều hàm kích hoạt khác nhau, ví dụ như hàm ReLU, hàm sigmoid, hàm Tanh, hàm softplus,... Trong các mạng nơ-ron hiện đại, hàm ReLU [20] được sử dụng phổ biến nhất do có nhiều lợi ích cho quá trình huấn luyện và tốc độ tính toán nhanh:

$$ReLU(x) = \max(0, x) \tag{2.2}$$

Quá trình lần lượt tính toán mạng nơ-ron từ lớp đầu vào, tới lớp ẩn và cuối cùng là lớp đầu ra được gọi là quá trình lan truyền tiến (feedforward).

Mạng nơ-ron học từ quá trình đối nghịch với lan truyền tiến gọi là lan truyền ngược (backpropagation). Với vec-tơ đặc trưng đầu vào $\boldsymbol{x}$ với nhãn tương ứng y, đầu tiên ta tính giá trị của nơ-ron lớp đầu ra $\boldsymbol{a}^{L-1}$ bằng quá trình lan truyền tiến. Sau đó, ta "lan truyền" sai khác giữa $\boldsymbol{a}^{L-1}$ và y tới toàn mạng để điều chỉnh các bộ tham số $\boldsymbol{W}^i$ và $\boldsymbol{b}^i$ với mục tiêu để giảm sai khác này.

Để đo đạc sự sai khác trong đầu ra của mạng và nhãn đầu vào, ta sử dụng một hàm mất mát. Dựa vào tác vụ khác nhau của bài toán ta cần sử dụng các hàm mất mát khác nhau. Thông thường, bài toán hồi quy sử dụng hàm trung bình bình phương sai số (mean square error - MSE), còn bài toán phân loại sử dụng hàm entropy chéo (cross entropy - CE). Với tập dữ liệu $D = \{(\boldsymbol{x}_1, y_1), ..., (\boldsymbol{x}_N, y_N)\}$, giả sử lớp cuối cùng của mạng có một nơ-ron, gọi a_i^{L-1} là giá trị của nơ-ron đầu ra tương ứng với dữ liệu đầu vào $\boldsymbol{x}_i$. Hàm mất mát MSE được tính như sau:

$$MSE = \frac{1}{N}\sum_{i=1}^{N}(y_i - a_i^{L-1})^2 \tag{2.3}$$

hàm CE được tính theo công thức:

$$CE = -\frac{1}{N}\sum_{i=1}^{N} a_i^{L-1} log(y_i) \tag{2.4}$$

Để điều chỉnh các tham số, ta cần phải biết giá trị cập nhật cho mỗi tham số. Dựa trên giá trị mất mát, các tín hiệu mất mát của nơ-ron trong lớp thứ i được tính toán dựa trên lỗi mà lớp thứ $i+1$ gây ra. Tín hiệu mất mát này được lan truyền từ lớp đầu ra về tới lớp đầu vào, sau đó thực hiện cập nhật các tham số $\boldsymbol{W}, \boldsymbol{b}$ nhằm giảm các giá trị lỗi. Các giá trị lỗi của các bộ tham số được gọi là gradient.

Sau khi có bộ gradient, ta có thể cập nhật mạng bằng cách đi ngược lại với hướng của gradient. Giá trị mất mát trên bộ dữ liệu sẽ giảm nếu gradient được cập nhật với bước đủ nhỏ. Bằng cách lặp đi lặp lại quá trình lan truyền tiến, lan truyền ngược và cập nhật tham số bằng gradient, bộ tham số có thể hội tụ tại một điểm cực tiểu của hàm mất mát. Phương pháp tối ưu lặp này được gọi là gradient descent (Hình 2.3).

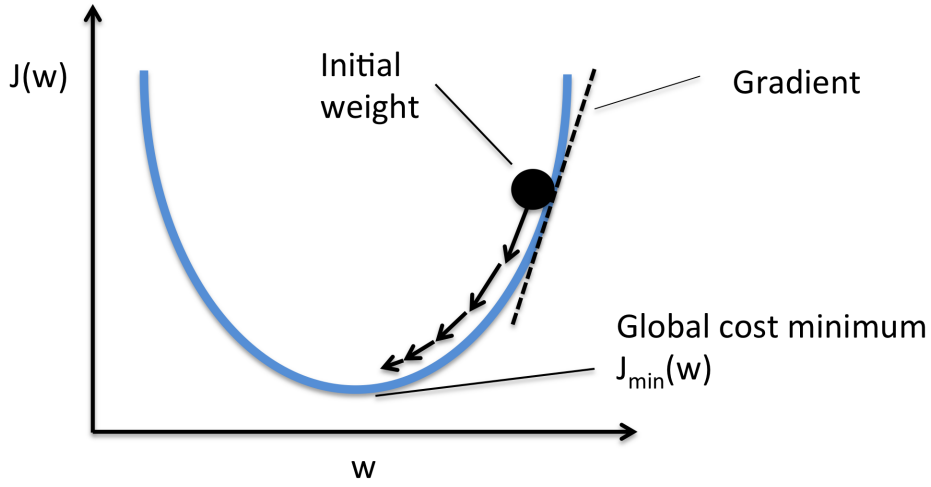


Figure 2.3: Phương pháp gradient descent [3]

Hệ số học (learning rate) được điều chỉnh để kiểm soát bước cập nhật trong gradient descent. Với hệ số học lớn, bước đi của gradient descent trở nên lớn hơn và có thể gặp khó khăn hội tụ khi gần điểm cực tiểu. Hệ số học nhỏ hơn giúp mô hình dễ hội tụ hơn tuy nhiên mất nhiều vòng lặp hơn.

Phương pháp gradient descent truyền thống sử dụng gradient cho cả bộ dữ liệu. Tuy vậy, với các bộ dữ liệu cực lớn, điều này là không thể. Do vậy, ta phải xấp xỉ gradient của cả bộ dữ liệu bằng cách chia nhỏ bộ dữ liệu gốc thành các lô

[3] https://machinelearningnotepad.wordpress.com/2018/04/15/gradient-descent

nhỏ (mini-batch) và thực hiện việc cập nhật trọng số trên các lô này. Phương pháp xấp xỉ này được gọi là stochastic gradient descent - SGD. Gần đây, nhiều phương pháp biến thể của SGD được phát triển có thể kể đến như RMSprop, Adadelta [40] và Adam [14] giúp tăng tốc độ hội tụ cũng như tăng cường hiệu năng khi huấn luyện mô hình.

2.2 Học sâu

Học sâu là một lớp các mô hình học máy được xây dựng dựa trên mạng nơ-ron nhân tạo. Nếu mạng nơ-ron nhân tạo chỉ gồm vài lớp nơ-ron, các mạng học sâu được thiết kế để mở rộng ra hàng chục, trăm, thậm chí hàng nghìn lớp nơ-ron. Điều này giúp các mô hình học sâu giải quyết được nhiều bài toán phức tạp với độ chính xác ngang bằng con người. Huấn luyện mô hình học sâu yêu cầu một lượng dữ liệu lớn và tài nguyên tính toán lớn. Trong thập kỉ vừa qua với sự bùng nổ của dữ liệu và tài nguyên tính toán, nghiên cứu học sâu trở thành tâm điểm của lĩnh vực trí tuệ nhân tạo.

2.2.1 Mạng nơ-ron tích chập

Mạng nơ-ron tích chập (Convolutional neural networks), hay mạng tích chập, được đề xuất lần đầu vào năm 1989 bởi Yan LeCun [16] để giải quyết bài toán nhận dạng chữ viết tay. Mạng tích chập được thiết kế với mục tiêu xử lý dữ liệu dạng bảng - lưới, ví dụ như dữ liệu chuỗi thời gian có thể biểu diễn dưới dạng bảng 1D hay ảnh là dữ liệu 2D của các điểm ảnh. Năm 2012, Alex Krizhevsky xây dựng một mạng tích chập (AlexNet [15]) và tăng tốc quá trình huấn luyện sử dụng GPU. Mô hình đề xuất của Krizhevsky đứng đầu trong bảng xếp hạng trong cuộc thi phân loại ảnh ImageNet [9] với độ chính xác vượt hơn 10% các đội tham gia. Hiện nay, mạng tích chập được áp dụng xử lý nhiều bài toán phức tạp trong xử lý ảnh, xử lý ngôn ngữ tự nhiên, xử lý tín hiệu âm thanh, ...

Thời điểm hiện tại đã có rất nhiều kiến trúc mạng nơ-ron tích chập khác nhau được xây dựng, tuy nhiên chúng đều được cấu thành từ các lớp thành phần (Hình 2.4), bao gồm: lớp tích chập (convolutional layer), lớp tổng hợp (pooling layer) và lớp kết nối đầy đủ (fully connected layer).

[4]https://www.upgrad.com/blog/basic-cnn-architecture

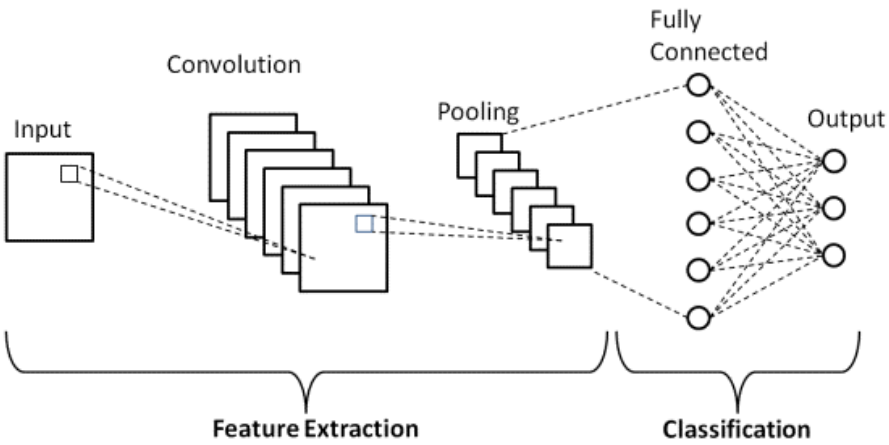


Figure 2.4: Các thành phần cơ bản của mạng tích chập [4]

Lớp tích chập

Lớp tích chập là lớp được sử dụng nhiều nhất trong mạng nơ-ron tích chập, mục tiêu của lớp tích chập là học các đặc trưng cục bộ từ dữ liệu đầu vào (Hình 2.5). Cái tên lớp tích chập xuất phát từ việc lớp sử dụng phép biến đổi toán học tuyến tính gọi là tích chập (convolution).

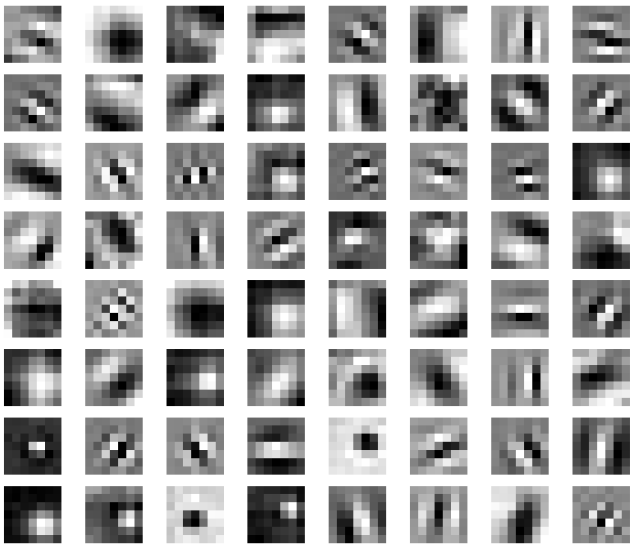


Figure 2.5: Các đặc trưng học được trong lớp tích chập [5]

Ban đầu, phép tích chập được được sử dụng phổ biến trong xử lý tín hiệu. Về sau nguyên lý biến đổi trong tích chập mới được áp dụng trong lĩnh vực xử lý hình ảnh. Công thức tích chập

[5]https://debuggercafe.com/visualizing-filters-and-feature-maps-in-convolutional-neural-networks-using-pytorch

Lớp tổng hợp

Lớp kết nối đầy đủ

2.3 Metric Learning

2.3.1 Contrastive loss

2.3.2 Triplet loss

2.4 Trích xuất thông tin

Xử lý tín hiệu đóng vai trò quan trọng trong bất kì hệ thống tiếng nói nào, từ nhận dạng tiếng nói tới nhận dạng người nói. Hai trong những đặc trưng âm thanh phổ biến nhất được sử dụng là filter banks và Mel-Frequency Cepstral Coefficients (MFCCs).

Trích xuất filter banks và MFCCs tuân theo quy trình khá tương tự nhau, tính toán filter banks yêu cầu ít hơn MFCCs một số bước. Quy trình trích xuất MFCCs gồm 6 bước chính (Hình 2.6): nhấn mạnh (pre-emhphasis), cắt khung (framing), cửa sổ (windowing), biến đổi Fourier (Fourier transform), filter bank và cuối cùng là biến đổi cô sin rời rạc (Discrete cosine transform) để có kết quả là MFCC.

Bước 1: Nhấn mạnh

Trong bước này, tín hiệu đầu vào được đưa qua một bộ lọc để khuếch đại các tín hiệu với tần số cao. Tín hiệu đầu ra của bộ lọc y_t tại thời gian t được tính như sau:

$$y_t = x_y - \alpha x_{t-1} \tag{2.5}$$

Trong đó, x là dãy tín hiệu đầu vào và α là hệ số của bộ lọc. α thường nhận giá trị 0.95 hoặc 0.97.

Bước 2: Cắt khung

Bước 3: Nhấn mạnh

Bước 4: Nhấn mạnh

Bước 5: Nhấn mạnh

Bước 6: Nhấn mạnh

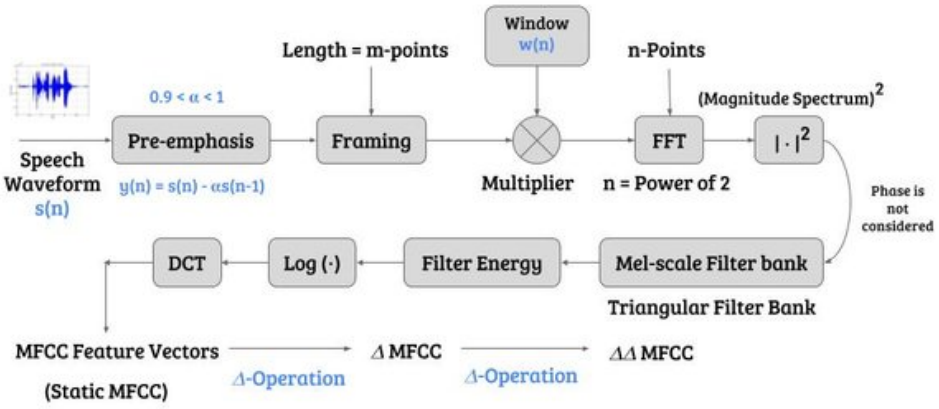


Figure 2.6: Thuật toán trich xuất MFCCs [5]

Chapter 3

Nhận dạng người nói tiếng Việt

Trong chương này, tác giả trình bày phương pháp học sâu để giải quyết bài toán nhận diện người nói. Ngoài ra tác giả cũng đề xuất phương pháp để cải tiến mô hình hiện tại để đạt được kết quả cao hơn trên tiếng Việt.

3.1 Hệ thống cơ sở

Hệ thống tác giả sử dụng trong đồ án tuân theo hệ thống ba pha được mô tả như trong 1.2, đây là sự kết hợp giữa mô hình ResNet, lớp tổng hợp thống kê tập trung (attentive statistic pooling - ASP) và hàm mất mát Angular Prototypical (AP). Kiến trúc của hệ thống được miêu tả trong Hình 3.1.

Đầu vào của hệ thống là **TODO**....

Trong các mục tiếp theo, tác giả sẽ trình bày chi tiết thành phần của hệ thống.

3.1.1 Biểu diễn khung âm thanh bằng mạng ResNet

Đầu vào của mạng ResNet là

3.1.2 Tổng hợp thống kê tập trung

Trong thực tế, không phải khung âm thanh nào cũng chứa nhiều thông tin của người nói do độ dài một khung rất ngắn thông thường chỉ 25 mili giây. Ví dụ, một khung có thể chứa nhiều tiếng ồn, hoặc không hề chứa giọng nói. Do vậy, hệ thống sử dụng tổng hợp thống kê tập trung để đánh trọng số cho biểu diễn của các khung với mong muốn tăng thông tin của những khung nhiều ý nghĩa

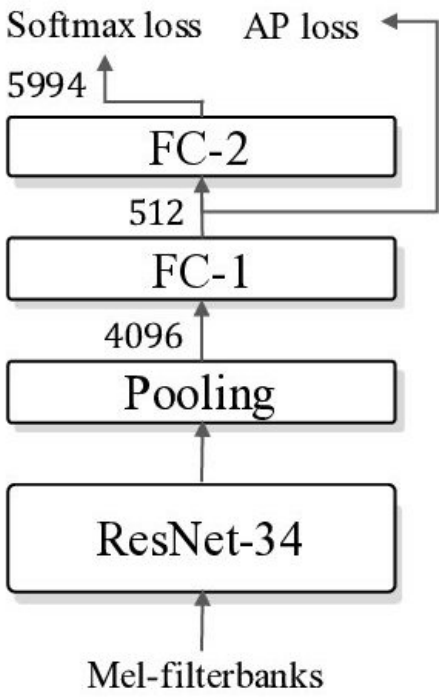


Figure 3.1: Tổng quan hệ thống sử dụng hàm mất mát Angular Prototypical

và giảm thông tin của những khung ít ý nghĩa. Từ đó có thể phân biệt giọng nói một cách hiệu quả hơn.

ASP nhận đầu vào là tập các vec-tơ biểu diễn khung $\boldsymbol{h}_t$ $(t = 1, ..., T)$. Vec-tơ biểu diễn của toàn đoạn âm thanh được tính qua 2 bước: đánh trọng số cho từng khung bằng cơ chế tập trung và tổng hợp thông tin thống kê dựa trên trọng số tính được.

Cơ chế tập trung

Bằng cơ chế tập trung, trọng số của từng khung có thể được tính như sau:

$$e_t = \boldsymbol{v}^T f(\boldsymbol{W}\boldsymbol{h}_t + b) + k \tag{3.1}$$

$$\alpha_t = \frac{\alpha_t}{\sum_{\rho}^{T} \alpha_\rho} \tag{3.2}$$

Trong công thức 3.1, với $\boldsymbol{v}, \boldsymbol{W}$ là các ma trận trọng số có thể học được, f là hàm phi tuyến như thanh hoặc ReLu, ta tính được điểm cho mỗi khung e_t. Sau đó, điểm của mỗi khung được chuẩn hoá trên tất cả các khung để thu được trọng số tập trung bằng hàm softmax như trong 3.2.

Tổng hợp thống kê

Sau khi có được trọng số của các khung, ta tính vec-tơ trung bình có trọng số:

$$\boldsymbol{\mu} = \sum_{t}^{T} \alpha_t \ \boldsymbol{h}_t \tag{3.3}$$

Bằng cách tính này, vec-tơ biểu diễn của đoạn âm thanh tập trung hơn vào những khung tiếng nói có ý nghĩa cao. Ngoài ra, các trọng số tập trung còn được sử dụng để tính độ lệch chuẩn có trọng số:

$$\boldsymbol{\sigma} = \sqrt{\sum_{t}^{T} \alpha_t \boldsymbol{h}_t \odot \boldsymbol{h}_t - \boldsymbol{\mu} \odot \boldsymbol{\mu}} \tag{3.4}$$

Với $\odot$ là phép nhân Hadamard, $\boldsymbol{\mu}$ là vec-tơ trung bình có trọng số tính trong công thức 3.3. Sau khi hoàn tất quá trình tính toán vec-tơ trung bình và độ lệch chuẩn có trọng số, $\boldsymbol{\mu}$ và $\boldsymbol{\sigma}$ được ghép lại để biểu diễn cho một đoạn tiếng nói. Bằng cách này, mọi đoạn âm thanh dài ngắn đều có vec-tơ biểu diễn với số chiều như nhau, được tổng hợp từ những khung âm thanh có ý nghĩa nhất trong câu.

3.1.3 Hàm mất mát Angular Prototypical

Trong thực tế, ta cần tổng hợp từ một số câu nói nhất định để tạo vec-tơ biểu diễn người nói. Do vậy, Chung và cộng sự [8] đề xuất hàm mất mát AP tối ưu không gian biểu diễn dựa trên nguyên mẫu (prototype) của người nói. Mỗi người nói có một nguyên mẫu và một câu truy vấn, mục tiêu của AP là đẩy xa truy vấn của một người ra xa nguyên mẫu của những người khác và kéo nó lại gần nguyên mẫu của người đó (Hình 3.2).

Xét một mini-batch gồm M đoạn tiếng nói từ mỗi N người nói, gọi $\boldsymbol{x}_{i,j}$ là vec-tơ biểu diễn của đoạn tiếng nói thứ j của người thứ i, $1 \leq i \leq N, 1 \leq j \leq M$. Giả sử truy vấn của một người là câu cuối cùng của người đó $\boldsymbol{x}_{i,M}$, nguyên mẫu của một người nói được tính toán như sau:

$$\boldsymbol{c}_i = \frac{1}{M-1} \sum_{m=1}^{M-1} \boldsymbol{x}_{i,m} \tag{3.5}$$

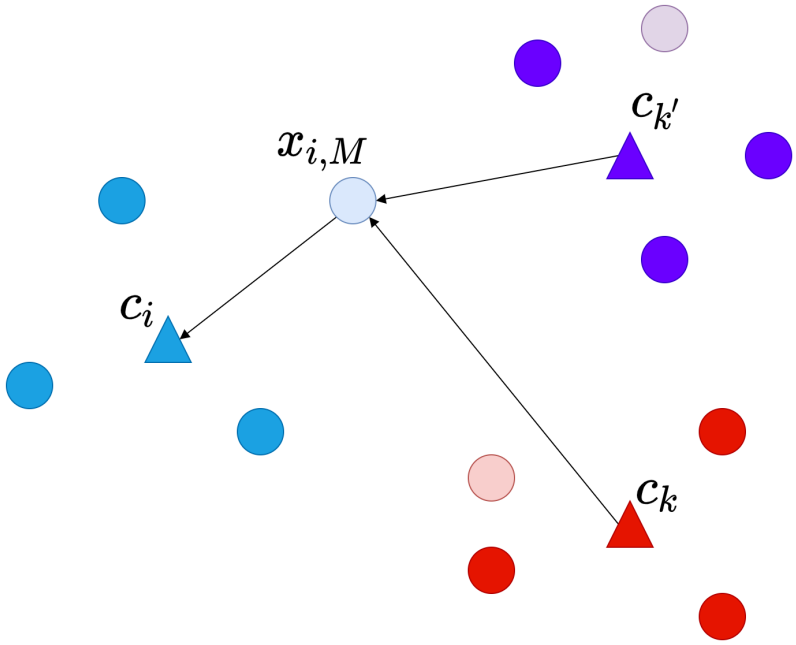


Figure 3.2: Hàm mất mát Angular Prototypical

Trong AP, độ tương đồng cô-sin được sử dụng để làm độ đo. Độ tương đồng được tính theo công thức 3.6 với hệ số scale w và bias b. Hai hệ số này giúp mô hình hội tụ ổn định hơn và khái quát hoá tốt hơn với thay đổi trong đặc trưng đầu vào [36].

$$\boldsymbol{S}_{i,k} = w \cdot cos(\boldsymbol{x}_{i,M}, \boldsymbol{c}_k) + b \tag{3.6}$$

Trong quá trình huấn luyện, câu truy vấn của mỗi người được phân loại dựa trên độ tương đồng đối với N nguyên mẫu trong mini-batch:

$$L_P = -\frac{1}{N}\sum_{i=1}^{N} log\frac{e^{\boldsymbol{S}_{i,i}}}{\sum_{k=1}^{N} e^{\boldsymbol{S}_{i,k}}} \tag{3.7}$$

Trong công thức 3.7, $\boldsymbol{S}_{i,i}$ là độ tương đồng của truy vấn người i và nguyên mẫu của chính người đó. Bằng việc sử dụng hàm softmax, $\boldsymbol{S}_{i,i}$ được đẩy gần hơn tới 1 và mô hình bị "phạt" nặng hơn nếu độ tương đồng của truy vấn người i tới nguyên mẫu của người khác lớn.

3.2 Đề xuất cải tiến mô hình

Trong phần này, đề xuất ...

3.2.1 Transfer learning

Transfer learning là một kỹ thuật trong học máy khi mà một mô hình được huấn luyện cho một tác vụ nhất định được sử dụng làm điểm bắt đầu cho một tác vụ khác. Transfer learning cho phép rút ngắn quá trình huấn luyện và gia tăng hiệu năng cho quá trình huấn luyện mô hình trên tác vụ mới với ít dữ liệu hơn đáng kể.

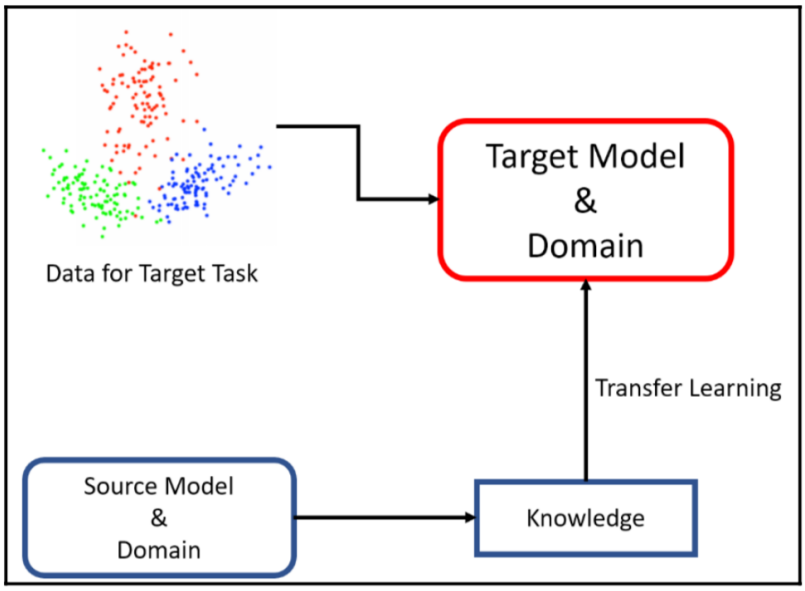


Figure 3.3: Sơ đồ mô tả transfer learning sử dụng kiến thức hiện có cho các tác vụ mới [24]

Cụ thể hơn, transfer learning hỗ trợ việc huấn luyện tác vụ mục tiêu theo những cách sau:

- Hiệu suất cơ sở tốt hơn (higher start): khi ta tăng cường kiến thức của mô hình mới với kiến thức từ mô hình gốc, hiệu suất cơ sở có thể cải thiện nhờ việc chuyển giao kiến thức.
- Thời gian huấn luyện ngắn hơn (higher slope): tốc độ hội tụ của mô hình mới có thể nhanh hơn dẫn tới thời gian huấn luyện ngắn hơn.
- Kết quả cuối cùng tốt hơn (higher asymptote): hiệu suất cuối cùng cao hơn có thể đạt được bằng việc sử dụng transfer learning.

Huấn luyện mô hình học sâu cần một lượng lớn tài nguyên tính toán và dữ liệu, do đó transfer learning được sử dụng rộng rãi trong cộng đồng nghiên cứu học sâu cho bài toán thị giác máy tính hay xử lý ngôn ngữ tự nhiên. TODO...

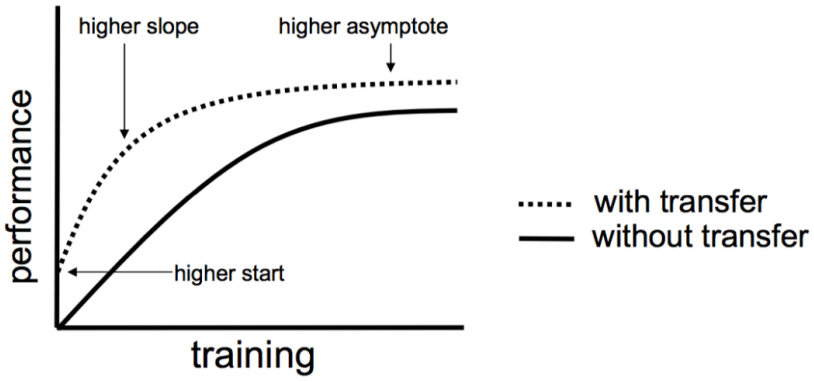


Figure 3.4: Lợi ích của transfer learning đối với việc huấn luyện mô hình [4]

3.2.2 Stochastic gradient descent khái quát hoá tốt hơn ADAM

3.2.3 Angular margin tăng khoảng cách giữa các người nói trong không gian embedding

Chapter 4

Thực nghiệm và đánh giá

4.1 Cơ sở dữ liệu

Để xây dựng cơ sở dữ liệu cho phần thực nghiệm, tác giả tổng hợp và sàng lọc dữ liệu từ ba bộ dữ liệu công khai: ZaloAI, VIVOS Corpus và VLSP ASR 2020.

Bắt đầu từ năm 2018, ZaloAI challenge [3] là một cuộc thi thường niên tập trung vào trí tuệ nhân tạo do Zalo Group, VNG tổ chức. Năm 2020, ZaloAI challenge thử thách các nhà phát triển và kỹ sư học máy Việt Nam với ba bài toán: tóm tắt tin tức, phát hiện biển báo giao thông, và xác thực người nói. Bộ dữ liệu huấn luyện công khai của bài toán xác thực giọng nói bao gồm 400 danh tính tiếng Việt thu thập từ chương trình truyền hình Bạn muốn hẹn hò. Mỗi danh tính có trung bình *26.4* câu nói với phân phối mô tả trong Hình 4.1. Bộ dữ liệu tuy đa dạng về mặt độ tuổi giới tính tuy nhiên vẫn còn vấn đề như: trùng lặp danh tính, nhiễu âm thanh như nhạc nền, người nói phía sau, câu nói của danh tính này lẫn vào danh tính kia, ...

VIVOS là tập giọng nói tiếng Việt phục vụ cho bài toán nhận dạng tiếng nói thu thập bởi phòng thí nghiệm khoa học máy tính AILAB từ trường Đại học Khoa học Tự nhiên - Đại học Quốc Gia TP.HCM [2]. Tuy chủ đích của bộ dữ liệu là dành cho nhận dạng tiếng nói nhưng lại có nhãn danh tính cụ thể nên có thể sử dụng cho bài toán nhận dạng người nói. Tập huấn luyện VIVOS bao gồm 40 danh tính với trung bình *253.5* câu nói mỗi người. Tập kiểm thử có 19 danh tính không trùng với tập huấn luyện với trung bình *40* câu nói mỗi người. Chất lượng dữ liệu của VIVOS rất tốt do điều kiện thu âm được kiểm soát nên không yêu cầu xử lý gì thêm.

Bộ dữ liệu VLSP ASR 2020 [1] nằm trong chiến dịch đánh giá năm 2020 của

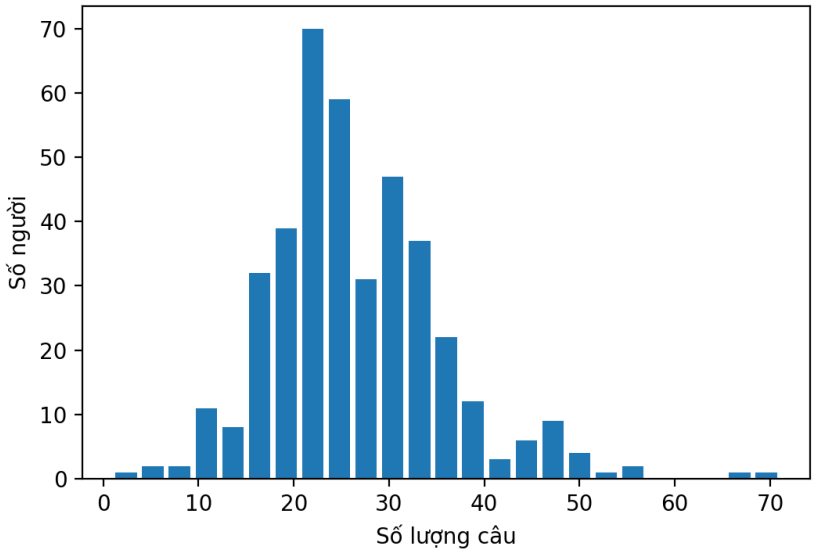


Figure 4.1: Biểu đồ phân phối số danh tính theo số câu nói của bộ dữ liệu ZaloAI

Hiệp hội xử lý Ngôn ngữ và Tiếng nói tiếng Việt. Giống như VIVOS, VLSP ASR 2020 được thu thập và thiết kế cho bài toán nhận diện giọng nói nhưng có nhãn danh tính cho các câu nói. Tổng số danh tính trong VLSP ASR 2020 là 567 người với trung bình 22.3 câu mỗi người. Tuy nhiên, dữ liệu danh tính của bộ dữ liệu lại không được chuẩn xác và có nhiều vấn đề tương tự như bộ ZaloAI. Những vấn đề này được giải quyết bằng phương pháp mô tả trong 4.1.1.

4.1.1 Cải thiện chất lượng bộ dữ liệu

Hai bộ ZaloAI và VLSP có tổng cộng gần 1 nghìn danh tính và hơn 20 nghìn câu. Do vậy, việc kiểm tra dữ liệu rất khó khăn và tốn thời gian. Việc này còn trở nên khó khăn hơn khi đánh giá bằng tai người, ví dụ để phân biệt giọng của 2 người cùng là nam, giọng trầm miền bắc thì cần sự tập trung cao độ để tìm điểm khác biệt. Vì thế, đồ án phân tích ma trận tương đồng của các câu nói để tìm ra sự không nhất quán từ đó thu hẹp phạm vi kiểm tra.

Cho một tập biểu diễn n đoạn âm thanh đầu vào $\boldsymbol{V} = \{\boldsymbol{v}_0, \boldsymbol{v}_1, ..., \boldsymbol{v}_{n-1}\}$, sử dụng độ tương đồng cô-sin, ma trận tương đồng cho các đoạn tiếng nói được tính theo công thức 4.1. Ví dụ một ma trận tương đồng trong Hình 4.2, đường chéo chính có giá trị tương đồng là 1 do so sánh mỗi câu với chính câu đó.

$$\boldsymbol{S}_{i,j} = cos(\boldsymbol{v}_i, \boldsymbol{v}_j) = \frac{\boldsymbol{v}_i \cdot \boldsymbol{v}_j}{\|\boldsymbol{v}_i\| \|\boldsymbol{v}_j\|}, 0 \leq i, j \leq n \tag{4.1}$$

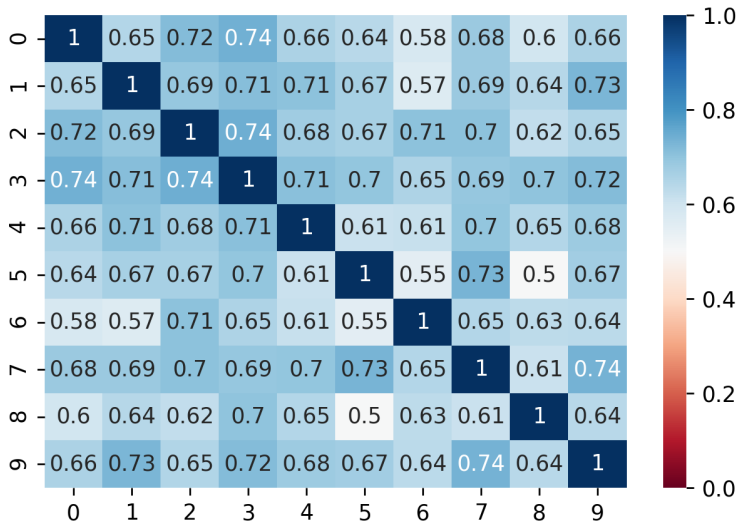


Figure 4.2: Ma trận tương đồng cho một tập 10 đoạn âm thanh của một người

Ma trận tương đồng được áp dụng khá rộng rãi, trong đó, phổ biến nhất là phân tích âm nhạc dựa trên nội dung [26], phân tích văn bản [13] và tin sinh [6]. Các kĩ thuật được sử dụng chủ yếu là phân cụm và phân đoạn. Trong đồ án, tác giả chỉ sử dụng phân tích đơn thuần để tìm ra các người nói, câu nói có khả năng bị gán nhãn sai.

Loại bỏ người nói không hợp lệ

Việc loại bỏ một danh tính có thể do nhiều lý do: nhiều câu nói không thuộc về người đó, chất lượng âm thanh kém, môi trường xung quanh ồn ào, tệp âm thanh bị hư hại qua đường truyền hoặc thiết bị, ... Các nguyên nhân này dẫn đến việc chất giọng của danh tính không được đảm bảo gây bất lợi cho việc huấn luyện mô hình. Một số danh tính có số lượng câu có vấn đề lớn, làm sạch và loại bỏ từng câu bằng việc nghe rất tốn thời gian và công sức. Do vậy, việc loại bỏ hẳn những danh tính này là cần thiết. Khi nhìn vào ma trận tương đồng của một danh tính, có thể thấy được và loại bỏ những danh tính không hợp lệ. Ma trận tương đồng của một người hợp lệ và bị loại bỏ có thể được thấy trong Hình 4.2 và Hình 4.4 tương ứng.

Loại bỏ đoạn tiếng nói không hợp lệ

Các đoạn tiếng nói không hợp lệ bao gồm sai nhãn danh tính, độ dài quá ngắn, tiếng ồn xung quanh quá lớn hay trong một đoạn có giọng của nhiều người khác

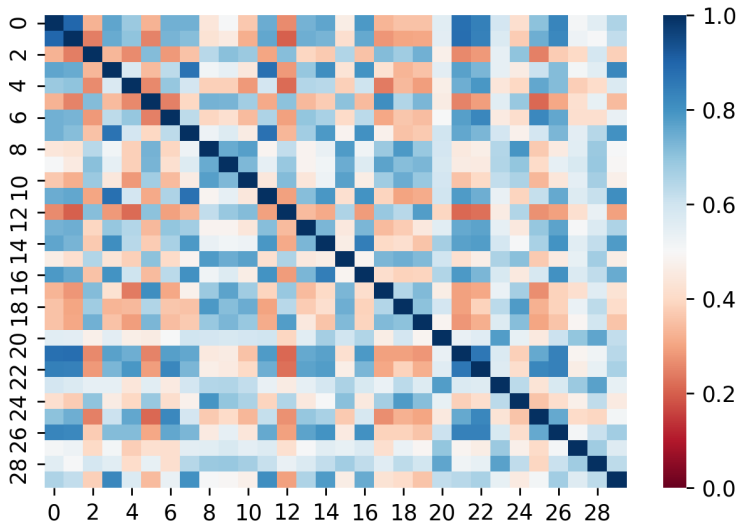


Figure 4.3: Ma trận tương đồng của một danh tính bị loại bỏ

nhau. Các đoạn này làm cho việc huấn luyện mô hình gặp khó khăn và giảm chất lượng của mô hình đầu ra. Lấy ma trận tương đồng của một người có câu nói không hợp lệ (Hình 4.4) làm ví dụ, để lọc ra đoạn có chỉ số 6 khá đơn giản bằng cách lấy một ngưỡng thấp (ví dụ 0.3). Những câu có độ tương đồng so với những câu khác của một danh tính mà dưới ngưỡng này ta sẽ xem là không hợp lệ. Tuy nhiên, cách này không hợp lý với những câu như câu chỉ số 2 trong Hình 4.4, có điểm nằm trong khoảng 0.4 - 0.6. Tuy có điểm tương đồng khá cao nhưng những câu này cũng cần được kiểm tra. Những câu này có thể được tìm thấy bằng cách phát hiện ngoại lệ sử dụng khoảng trong tứ phân vị (Interquartile range) [39].

Với, Q1, Q3 lần lượt là tứ phân vị thứ nhất và thứ ba của tập điểm trung bình của các câu $a_i = \frac{1}{n}\sum_{j=0,j\neq i}^{n-1} \boldsymbol{S}_{i,j}$, dựa trên khoảng trong tứ phân vị, đoạn điểm tương đồng hợp lệ cho tập điểm $\boldsymbol{a}$ được tính như sau:

$$a_{min} = Q1 - 1.5 * IQR;\ a_{max} = Q3 + 1.5 * IQR \tag{4.2}$$

$$IQR = Q3 - Q1 \tag{4.3}$$

Các câu có điểm trung bình a_i nằm ngoài đoạn $[a_{min}, a_{max}]$ được đánh dấu và cần nghe lại để quyết định có loại bỏ hay không.

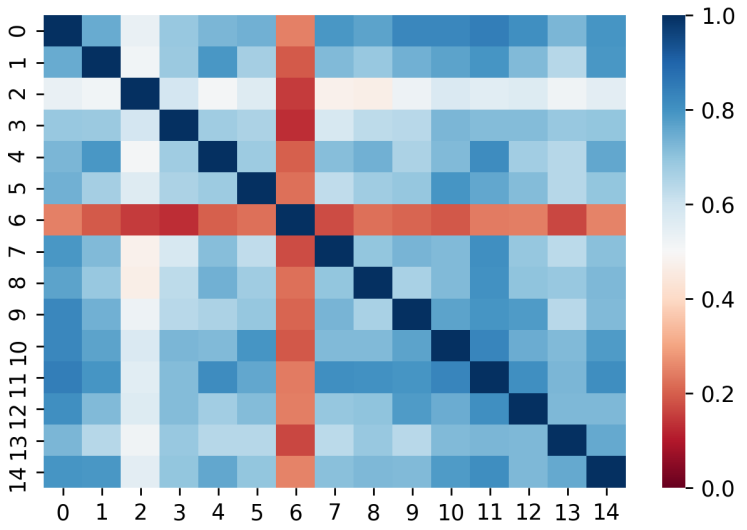


Figure 4.4: Ma trận tương đồng của một danh tính có đoạn âm thanh không hợp lệ

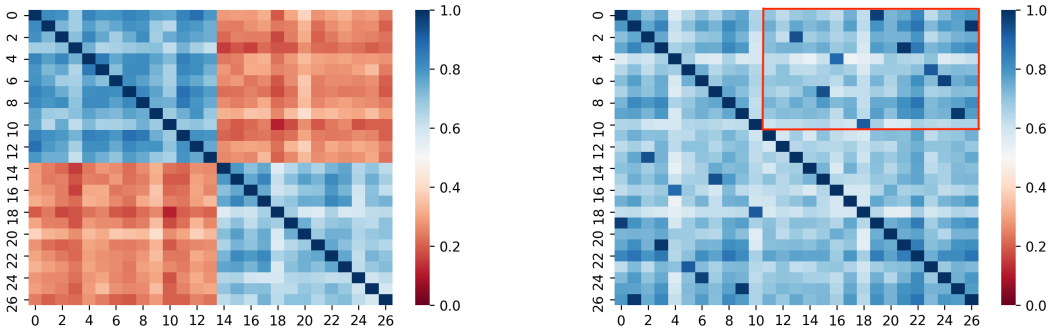
Hợp nhất người nói có cùng danh tính

Do các bộ dữ liệu được thu thập một cách độc lập, có khả năng người nói trong bộ dữ liệu này trùng với bộ kia. Hơn nữa, một người đã tồn tại trong cơ sở dữ liệu cũng có khả năng được yêu cầu thu lại. Các cặp người nói trùng danh tính có thể được tìm dựa vào ma trận tương đồng chéo. Từ Hình a mô tả ma trận tương đồng của người nói 52-M-31 và 64-M-30, có thể thấy rõ đây là 2 người khác nhau do ma trận tương đồng chéo (phần màu đỏ) có điểm tương đồng rất thấp. Ngược lại, trong Hình 4.5, 2 người có nhãn khác nhau là 64-M-30 và 636-M-30 thực chất là cùng một người với ma trận tương đồng chéo nằm trong ô màu đỏ. Có nhiều cặp câu điểm tương đồng cao (ô xanh đậm) nhưng không đạt tới 1.0 như trên đường chéo chính do cơ bản có cùng nội dung nhưng khác biệt đến từ sự biến đổi nhất định trong quá trình xử lý. Các cặp người nói có giá trị trung bình của ma trận tương đồng chéo lớn hơn 0.7 yêu cầu được nghe lại và ra quyết định để hợp nhất.

Bảng 4.1 tổng kết thông tin sàng lọc dữ liệu. Số người nói bị loại bỏ chiếm TODO stat tổng số người nói, số người nói được hợp nhất chiếm TODO stat tổng số người nói, số câu nói bị loại bỏ chiếm TODO stat tổng số câu.

Table 4.1: placeholder

Bộ dữ liệu	Số danh tính loại bỏ	Số danh tính hợp nhất	Số câu loại bỏ
ZaloAI	0	51	1,066
VIVOS	0	0	2
VLSP	65	33	549
Tổng	65	84	1,617



(a) Ma trận tương đồng của 52-M-31 và 64-M-30 (b) Ma trận tương đồng của 64-M-30 và 636-M-30

Figure 4.5

4.1.2 Bộ dữ liệu thực nghiệm

Sau khi loại bỏ các danh tính không phù hợp, hợp nhất người nói có cùng danh tính và loại bỏ các câu vấn đề, bộ dữ liệu thực nghiệm đã có chất lượng tương đối tốt. Tổng số lượng người nói là 1110, chia thành 3 tập: tập huấn luyện (training set) gồm 1031 người nói, tập kiểm thử (validation set) gồm 20 người, tập kiểm tra (test set) gồm 59 người nói. Người nói trong tập kiểm thử và 40 người trong tập kiểm tra được lấy ngẫu nhiên trong bộ ZaloAI với điều kiện cân bằng giới tính nam - nữ. 19 người còn lại trong tập kiểm tra là tập kiểm thử của bộ dữ liệu VIVOS.

4.2 Chi tiết cài đặt thực nghiệm

Thông số huấn luyện mô hình

Các thực nghiệm trong mục tiếp theo đều được chạy trên cùng bộ thông số như sau:

- Mạng trích xuất đặc trưng: ResNet
- Lớp tổng hợp: Tổng hợp thống kê tập trung
- Batch size: 100

Môi trường lập trình

Để cài đặt thực nghiệm, tác giả sử dụng ngôn ngữ lập trình Python kết hợp với thư viện PyTorch phiên bản 1.7.1. PyTorch là thư mã nguồn mở của Facebook

được xây dựng trên ngôn ngữ lập trình Lua. PyTorch cho phép người dùng xây dựng, tuỳ biến mô hình ở cả cấp cao và cấp thấp với thiết kế trực quan.

Môi trường thực nghiệm

Để thực hiện huấn luyện các mô hình, tác giả sử dụng Google Colaboratory: Hệ điều hành Ubuntu 18.04, 2vCPU Intel Xeon 2.2 Ghz, RAM 25GB, GPU Tesla T4 15GB.

4.3 Kết quả thực nghiệm và đánh giá

Trong phần này tác giả sử dụng tỉ lệ lỗi bằng nhau (Equal error rate - EER) để đánh giá hiệu năng của các mô hình thực nghiệm. EER là điểm nằm trên đường ROC mà có tỉ lệ dương tính giả (ví dụ bình thường được coi là bất thường) và tỉ lệ âm tính giả (ví dụ bất thường được xem là bình thường). Hệ thống nhận dạng càng tốt thì có EER càng nhỏ.

Tác giả thực hiện nhiều trường hợp thực nghiệm khác nhau với mục tiêu huấn luyện mô hình nhận dạng người nói một cách có hiệu quả trên tiếng Việt. . Thực nghiệm 5 kiểm tra tính hiệu quả của hàm mất mát AP với margin khác nhau. Mặc định, các thông số TODO ...:

- Các mô hình được finetune từ mô hình train sẵn trong [].

Thực nghiệm 1: làm sạch dữ liệu

Bảng 4.2 mô tả kết quả thực nghiệm với dữ liệu ban đầu và dữ liệu đã được sàng học như đã trình bày trong 4.1.1. Như có thể thấy, kết quả huấn luyện trên tập đã sàng lọc cải thiện 0.93% EER so với dữ liệu gốc. Do vậy, các thực nghiệm về sau sẽ sử dụng bộ dữ liệu đã qua sàng lọc.

Table 4.2: placeholder

Dữ liệu	**EER trên tập kiểm tra**
Gốc	6.790%
Cải thiện chất lượng	5.860%

Thực nghiệm 2: Phương thức huấn luyện

Trong thực nghiệm này, tác giả tiến hành khảo sát các phương thức huấn luyện mô hình. Bảng [**?**] mô tả kết quả với các trường hợp khác nhau. Do số người nói trong bộ dữ liệu tiếng Việt ít hơn hẳn so với người nói tiếng Anh trong bộ dữ liệu VoxCeleb, huấn luyện không tập trung đủ để tìm ra các điểm hữu dụng phân biệt người nói tiếng Việt, dẫn đến mô hình huấn luyện từ đầu kết hợp hai bộ dữ liệu đạt kết quả tệ hơn. Có thể thấy finetune bằng riêng dữ liệu tiếng Việt cho kết quả vượt trội so với huấn luyện từ đầu bằng bộ dữ liệu tiếng Việt hoặc kết hợp VoxCeleb (dữ liệu tiếng Anh) với EER 4.775%. Các thử nghiệm phía sau sử dụng phương pháp finetune.

Table 4.3: placeholder

Phương pháp	**EER trên tập kiểm tra**
Pretrain	10.60%
Scratch + vn data + VoxCeleb	6.294%
Scratch + vn data	5.860%
Finetune	**4.775%**

Thực nghiệm 3: Khử nhiễu đầu vào

Table 4.4: placeholder

Dữ liệu	**EER trên tập kiểm tra**
Nhiễu âm thanh	4.775%
Khử nhiễu âm thanh	**4.215%**

Thực nghiệm 4: Hàm tối ưu

Table 4.5: placeholder

Dữ liệu	**EER trên tập kiểm tra**
ADAM	4.215%
SGD	**2.930%**

Thực nghiệm 5: Hàm mất mát

- Visualizing embedding space with t-SNE (some speakers) - DET curve

Table 4.6: placeholder

hmm	**EER trên tập kiểm tra**
AP	2.930%
AP (m=0.1)	2.789%
AP (m=0.2)	**2.749%**
AP (m=0.3)	2.754%
AP (m=0.4)	2.804%
AP (m=0.5)	2.892%

Chapter 5

Conclusions

.

Bibliography

[1] Automatic speech recognition for vietnamese. `https://vlsp.org.vn/vlsp2020/eval/asr`. Truy cập vào: 10-05-2021.

[2] Vivos corpus. `https://ailab.hcmus.edu.vn/vivos`. Truy cập vào: 10-05-2021.

[3] Zalo ai challenge. `https://challenge.zalo.ai/`. Truy cập vào: 10-05-2021.

[4] Yusuf Aytar and Andrew Zisserman. Tabula rasa: Model transfer for object category detection. In *2011 international conference on computer vision*, pages 2252–2259. IEEE, 2011.

[5] Bidhan Barai, Debayan Das, Nibaran Das, Subhadip Basu, and Mita Nasipuri. An asr system using mfcc and vq/gmm with emphasis on environmental dependency. In *2017 IEEE Calcutta Conference (CALCON)*, pages 362–366. IEEE, 2017.

[6] A Bustamam, F Zubedi, and Titin Siswantining. Implementation χ–sim cosimilarity and agglomerative hierarchical to cluster gene expression data of lymphoma by gene and condition. In *AIP Conference Proceedings*, volume 2023, page 020221. AIP Publishing LLC, 2018.

[7] Sumit Chopra, Raia Hadsell, and Yann LeCun. Learning a similarity metric discriminatively, with application to face verification. In *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*, volume 1, pages 539–546. IEEE, 2005.

[8] Joon Son Chung, Jaesung Huh, Seongkyu Mun, Minjae Lee, Hee Soo Heo, Soyeon Choe, Chiheon Ham, Sunghwan Jung, Bong-Jin Lee, and Icksang Han. In defence of metric learning for speaker recognition. *arXiv preprint arXiv:2003.11982*, 2020.

[9] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee, 2009.

[10] Jiankang Deng, Jia Guo, Niannan Xue, and Stefanos Zafeiriou. Arcface: Additive angular margin loss for deep face recognition. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4690–4699, 2019.

[11] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[12] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

[13] Syed Fawad Hussain, Gilles Bisson, and Clément Grimal. An improved co-similarity measure for document clustering. In *2010 Ninth International Conference on Machine Learning and Applications*, pages 190–197. IEEE, 2010.

[14] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[15] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. *Advances in neural information processing systems*, 25:1097–1105, 2012.

[16] Yann LeCun, Bernhard Boser, John S Denker, Donnie Henderson, Richard E Howard, Wayne Hubbard, and Lawrence D Jackel. Backpropagation applied to handwritten zip code recognition. *Neural computation*, 1(4):541–551, 1989.

[17] Weiyang Liu, Yandong Wen, Zhiding Yu, Ming Li, Bhiksha Raj, and Le Song. Sphereface: Deep hypersphere embedding for face recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 212–220, 2017.

[18] Tom M Mitchell et al. Machine learning. 1997.

[19] Arsha Nagrani, Joon Son Chung, and Andrew Zisserman. Voxceleb: a large-scale speaker identification dataset. *arXiv preprint arXiv:1706.08612*, 2017.

[20] Vinod Nair and Geoffrey E Hinton. Rectified linear units improve restricted boltzmann machines. In *Icml*, 2010.

[21] Koji Okabe, Takafumi Koshinaka, and Koichi Shinoda. Attentive statistics pooling for deep speaker embedding. *arXiv preprint arXiv:1803.10963*, 2018.

[22] FA Rezaur rahman Chowdhury, Quan Wang, Ignacio Lopez Moreno, and Li Wan. Attention-based models for text-dependent speaker verification. In *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 5359–5363. IEEE, 2018.

[23] Mirco Ravanelli and Yoshua Bengio. Speaker recognition from raw waveform with sincnet. In *2018 IEEE Spoken Language Technology Workshop (SLT)*, pages 1021–1028. IEEE, 2018.

[24] Dipanjan Sarkar, Raghav Bali, and Tamoghna Ghosh. *Hands-On Transfer Learning with Python: Implement advanced deep learning and neural network models using TensorFlow and Keras*. Packt Publishing Ltd, 2018.

[25] Florian Schroff, Dmitry Kalenichenko, and James Philbin. Facenet: A unified embedding for face recognition and clustering. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 815–823, 2015.

[26] Diego F Silva, Chin-Chia M Yeh, Yan Zhu, Gustavo EAPA Batista, and Eamonn Keogh. Fast similarity matrix profile for music analysis and exploration. *IEEE Transactions on Multimedia*, 21(1):29–38, 2018.

[27] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

[28] David Snyder, Daniel Garcia-Romero, Daniel Povey, and Sanjeev Khudanpur. Deep neural network embeddings for text-independent speaker verification. In *Interspeech*, pages 999–1003, 2017.

[29] David Snyder, Daniel Garcia-Romero, Gregory Sell, Alan McCree, Daniel Povey, and Sanjeev Khudanpur. Speaker recognition for multi-speaker conversations using x-vectors. In *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 5796–5800. IEEE, 2019.

[30] David Snyder, Daniel Garcia-Romero, Gregory Sell, Daniel Povey, and Sanjeev Khudanpur. X-vectors: Robust dnn embeddings for speaker recognition. In *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 5329–5333. IEEE, 2018.

[31] David Snyder, Jesús Villalba, Nanxin Chen, Daniel Povey, Gregory Sell, Najim Dehak, and Sanjeev Khudanpur. The jhu speaker recognition system for the voices 2019 challenge. In *INTERSPEECH*, pages 2468–2472, 2019.

[32] Ehsan Variani, Xin Lei, Erik McDermott, Ignacio Lopez Moreno, and Javier Gonzalez-Dominguez. Deep neural networks for small footprint text-dependent speaker verification. In *2014 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 4052–4056. IEEE, 2014.

[33] Jesús Villalba, Nanxin Chen, David Snyder, Daniel Garcia-Romero, Alan McCree, Gregory Sell, Jonas Borgstrom, Fred Richardson, Suwon Shon, François Grondin, et al. State-of-the-art speaker recognition for telephone and video speech: The jhu-mit submission for nist sre18. In *Interspeech*, pages 1488–1492, 2019.

[34] Alex Waibel, Toshiyuki Hanazawa, Geoffrey Hinton, Kiyohiro Shikano, and Kevin J Lang. Phoneme recognition using time-delay neural networks. *IEEE transactions on acoustics, speech, and signal processing*, 37(3):328–339, 1989.

[35] Feng Wang, Jian Cheng, Weiyang Liu, and Haijun Liu. Additive margin softmax for face verification. *IEEE Signal Processing Letters*, 25(7):926–930, 2018.

[36] Jian Wang, Feng Zhou, Shilei Wen, Xiao Liu, and Yuanqing Lin. Deep metric learning with angular loss. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2593–2601, 2017.

[37] Zhuoyi Wang, Hemeng Tao, Zelun Kong, Swarup Chandra, and Latifur Khan. Metric learning based framework for streaming classification with concept evolution. In *2019 International Joint Conference on Neural Networks (IJCNN)*, pages 1–8. IEEE, 2019.

[38] Weidi Xie, Arsha Nagrani, Joon Son Chung, and Andrew Zisserman. Utterance-level aggregation for speaker recognition in the wild. In *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 5791–5795. IEEE, 2019.

[39] Jiawei Yang, Susanto Rahardja, and Pasi Fränti. Outlier detection: how to threshold outlier scores? In *Proceedings of the international conference on artificial intelligence, information processing and cloud computing*, pages 1–6, 2019.

[40] Matthew D Zeiler. Adadelta: an adaptive learning rate method. *arXiv preprint arXiv:1212.5701*, 2012.

[41] Chunlei Zhang, Kazuhito Koishida, and John HL Hansen. Text-independent speaker verification based on triplet convolutional neural network embeddings. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 26(9):1633–1644, 2018.

[42] Yingke Zhu, Tom Ko, David Snyder, Brian Mak, and Daniel Povey. Self-attentive speaker embeddings for text-independent speaker verification. In *Interspeech*, volume 2018, pages 3573–3577, 2018.