ĐẠI HỌC QUỐC GIA HÀ NỘI

TRƯỜNG ĐẠI HỌC CÔNG NGHỆ

Đinh Xuân Nhất

NGHIÊN CỨU CÁC THUẬT TOÁN NHẬN DẠNG CẢM XÚC KHUÔN MẶT TRÊN ẢNH 2D

KHOÁ LUẬN TỐT NGHIỆP ĐẠI HỌC HỆ CHÍNH QUY

Ngành: Công nghệ thông tin

HÀ NỘI – 2010

ĐẠI HỌC QUỐC GIA HÀ NỘI

TRƯỜNG ĐẠI HỌC CÔNG NGHỆ

Đinh Xuân Nhất

NGHIÊN CỨU CÁC THUẬT TOÁN NHẬN DẠNG CẢM XÚC KHUÔN MẶT TRÊN ẢNH 2D

KHOÁ LUẬN TỐT NGHIỆP ĐẠI HỌC HỆ CHÍNH QUY

Ngành: Công nghệ thông tin

Cán bộ hướng dẫn: PGS TS. Bùi Thế Duy

HÀ NỘI – 2010

LỜI CẢM ƠN

Lời đầu tiên em xin bày tỏ lòng biết ơn tới các thầy, cô giáo trong trường Đại học Công nghệ - Đại học Quốc gia Hà Nội. Các thầy cô đã dạy bảo, chỉ dẫn chúng em và luôn tạo điều kiện tốt nhất cho chúng em học tập trong suốt quá trình học đại học đặc biệt là trong thời gian làm khoá luận tốt nghiệp.

Em xin bày tỏ lòng biết ơn sâu sắc tới PGS TS. Bùi Thế Duy, thầy đã hướng dẫn em tận tình trong học kỳ vừa qua.

Tôi cũng xin cảm ơn những người bạn của mình, các bạn đã luôn ở bên tôi, giúp đỡ và cho tôi những ý kiến đóng góp quý báu trong học tập cũng như trong cuộc sống.

Cuối cùng con xin gửi tới bố mẹ và toàn thể gia đình lòng biết ơn và tình cảm yêu thương nhất. Con xin dành tặng bố mẹ kết quả mà con đã đạt được trong suốt bốn năm học đại học. Con cám ơn bố mẹ nhiều.

Hà nội, ngày 25/05/2010

Đinh Xuân Nhất

TÓM TẮT

Bài toàn nhận dạng cảm xúc đã bắt đầu được nghiên cứu từ những năm 1970 nhưng kết quả đạt được vẫn còn nhiều hạn chế. Hiện nay vấn đế này vẫn đang được rất nhiều người quan tâm bởi tính hấp dẫn cùng những vấn đề phức tạp của nó. Mục tiêu của khóa luận này là nghiên cứu và đánh giá về các phương pháp nhận dạng mặt người trong việc nhận dạng ra 5 cảm xúc cơ bản: Vui, buồn, ghê tởm, dận giữ và tự nhiên trên ảnh tĩnh, chính diện.

Từ khóa: Facial Expression Recognition, Principal Component Analysis, Neural Network, Decision Tree, Weka…

MỤC LỤC

DANH MỤC HÌNH ẢNH

Chương 1. GIỚI THIỆU

1.1 Cấu trúc của khóa luận

Với nội dung trình bày những lý thuyết cơ bản và cách áp dụng vào bài toán nhận dạng cảm xúc khuôn mặt, khóa luận được tổ chức theo cấu trúc như sau:

Chương 1: Giới thiệu

Giới thiệu sơ lược về các phương pháp nhận dạng cảm xúc, ứng dụng của nó trong cuộc sống hàng ngày, giới thiệu các phương pháp được sử dụng trong khóa luận này, mục tiêu và cấu trúc của khóa luận.

Chương 2: Một số lý thuyết cơ bản

Chương hai đi vào giới thiệu tổng quan về các lý thuyết cơ bản. Những kiến thức cơ bản này là tiền đề để người đọc hiểu được cách áp dụng vào bài toán nhận dạng cảm xúc và lớp các bài toán nhận dạng nói chung.

Chương 3: Các phương pháp nhận dạng cảm xúc

Chương này đi vào giới thiệu một số phương pháp nhận dạng cảm xúc sử dụng các lý thuyết cơ bản đã nêu ở chương hai

Chương 4: Thực nghiệm

Chương này phân tích về ưu, nhược điểm và so sánh, đánh giá giữa các phương pháp.

Chương 5: Kết luận

Chương này tổng kết lại những gì đã đạt được và chưa đạt được. Từ đó nêu lên những hướng nghiên cứu và phát triển tiếp theo.

1.2 Nhận dạng cảm xúc khuôn mặt và ứng dụng

Trong vài năm gần đây, cùng với sự phát triển về khoa học và công nghệ, tương tác người máy đã trở thành một lĩnh vực nổi bật nhằm cung cấp cho con người khả năng phục vụ của máy móc. Điều này bắt nguồn từ khả năng máy móc có thể tương tác được với con người. Máy móc cần các kỹ năng để trao đổi thông tin với con người và 1 trong những kỹ năng đó là khả năng hiểu được cảm xúc. Cách tốt nhất để một người biểu thị cảm xúc là qua khuôn mặt. Bài toàn nhận dạng cảm xúc khuôn mặt đã

được bắt đầu nghiên cứu từ những năm 1970 nhưng kết quả đạt được đến nay vẫn còn nhiều hạn chế.

Ứng dụng của nhận dạng cảm xúc trong cuộc sống hàng ngày là rất lớn, các hệ thống phát hiện trạng thái buồn ngủ dựa vào cảm xúc trên khuôn mặt được phát triển để cảnh báo cho người lái xe khi thấy dấu hiệu buồn ngủ, mệt mỏi. Các hệ thống kiểm tra tính đúng đắn của thông tin, các phần mềm điều khiển dựa vào cảm xúc, các thiết bị hỗ trợ người tàn tật,...

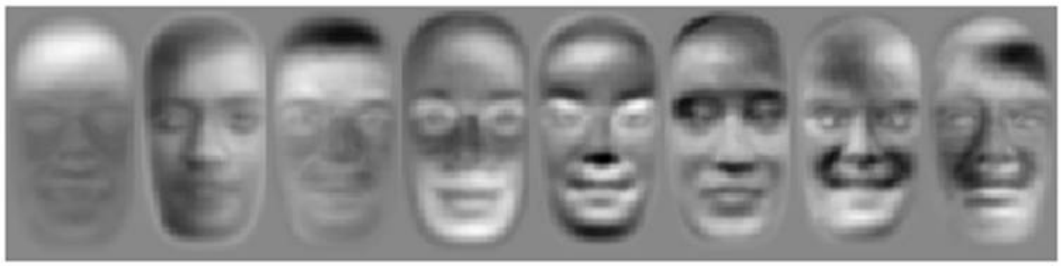
Mục tiêu của khóa luận này là nghiên cứu 1 số phương pháp nhận dạng cảm xúc khuôn mặt dựa trên ảnh hai chiều và trực diện

1.3 Một số phương pháp nhận dạng cảm xúc khuôn mặt

Có nhiều phương pháp đã được nghiên cứu để giải quết bài toán này, điển hình là một số phương pháp sau: Sử dụng các đơn vị vận động trên khuôn mặt (Action units – AU), sử dụng PCA, AAM kết hợp tương quan điểm, sử dụng các phương pháp học,… Mỗi phương pháp đều có ưu và nhược điểm riêng. Đối với các phương pháp sử dụng PCA kết hợp mạng nơron, cần một tập dữ liệu chuẩn để huấn luyện. Việc xây dựng các tập huấn luyện này cũng tương đối khó khăn và tốn kém vì cần nhiều người làm mẫu, những người này phải có khả năng diễn đạt cảm xúc tốt, ngoài ra còn cần sự đánh giá của các chuyên gia tâm lý. Hiện nay có một số tập huấn luyện chuẩn thường được dùng như JAFFE (Japanese Female Facial Expression) hay Cohn-kanade.

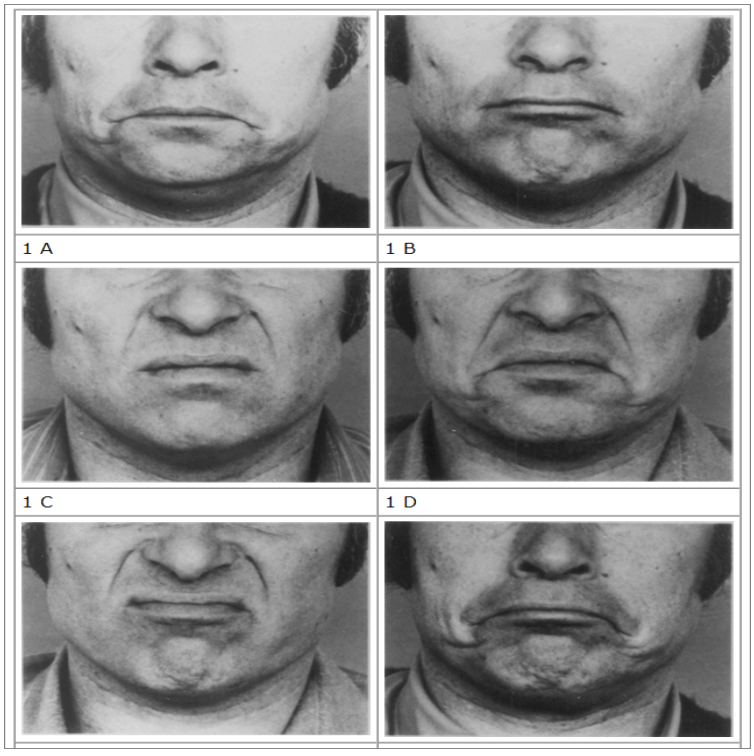
1.3.1 Các phương pháp dựa trên đặc trưng của ảnh

Các kỹ thuật sử dụng trong phương pháp này là phân tích thành phần chính PCA, sau đó huấn luyện bằng các thuật toán học. PCA được Karl Pearson tạo ra năm 1901. Đến những năm 80, Sirovich và Kirby đã phát triển kỹ thuật này để thể hiện khuôn mặt một cách hiệu quả. Đưa ra sự giống nhau giữa nhiều hình ảnh khuôn mặt khác nhau, kĩ thuật này tìm ra những thành phần cơ bản của sự phân bố trên khuôn mặt, thể hiện bằng các eigenvectors. Từng khuôn mặt trong một tập hợp các khuôn mặt sau đó có thể tính xấp xỉ bằng sự kết hợp tuyến tính giữa những eigenvector lớn nhất, được biết tới như eigenfaces.



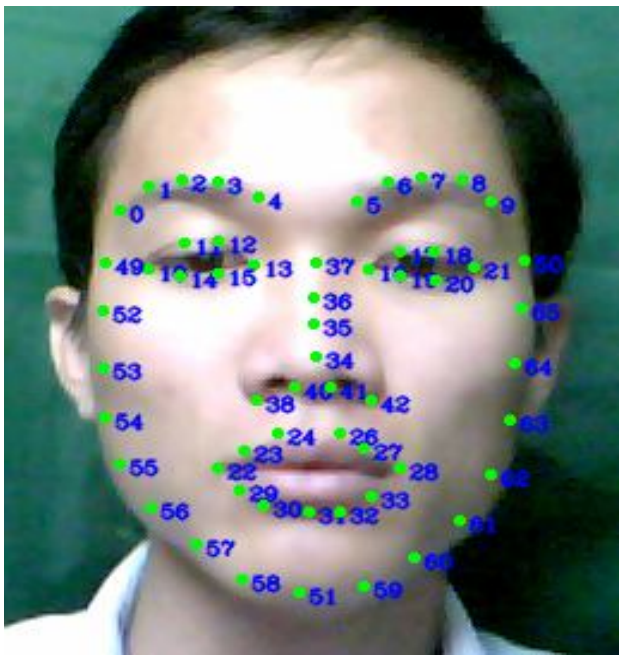
1.3.2 Phương pháp sử dụng Action Units

Phương pháp này nhận dạng cảm xúc dựa trên các đơn vị chuyển động của khuôn mặt (AU). Có tất cả 64 AU, mỗi AU là sự kết hợp của một số các cơ trên khuôn mặt. Cảm xúc được nhận dạng bằng cách phát hiện tại một thời điểm có bao nhiêu AU xuất hiện trên khuôn mặt và với các AU xuất hiện cùng nhau tướng ứng với 1 cảm xúc.

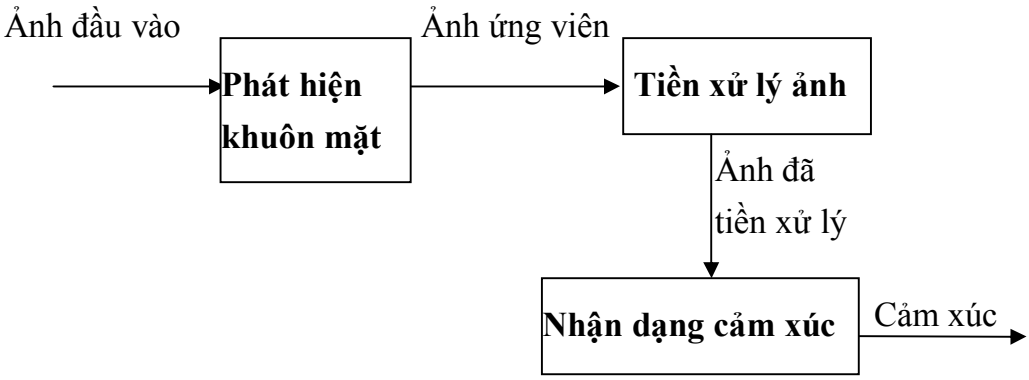


1.3.3 Phương pháp dùng mô hình AAM kết hợp tương quan điểm

Phương pháp này sử dụng mô hình AAM để phát hiện khuôn mặt. Sau đó dựa vào tỷ lệ giữa 2 mắt, lông mày, miệng, mũi, … để nhận dạng cảm xúc. Khó khăn của phương pháp này là việc xác định ngưỡng tỉ lệ để xác định cảm xúc. Tuy nhiên phương pháp này có ưu điểm về tốc độ, dó đó thường đươc ứng dụng trong nhận dạng cảm xúc thời gian thực.



1.3.4 Mô hình tổng quan



Hình 1: Mô hình nhận dạng cảm xúc

1.4 Các thách thức trong vấn đề nhận dạng cảm xúc khuôn mặt

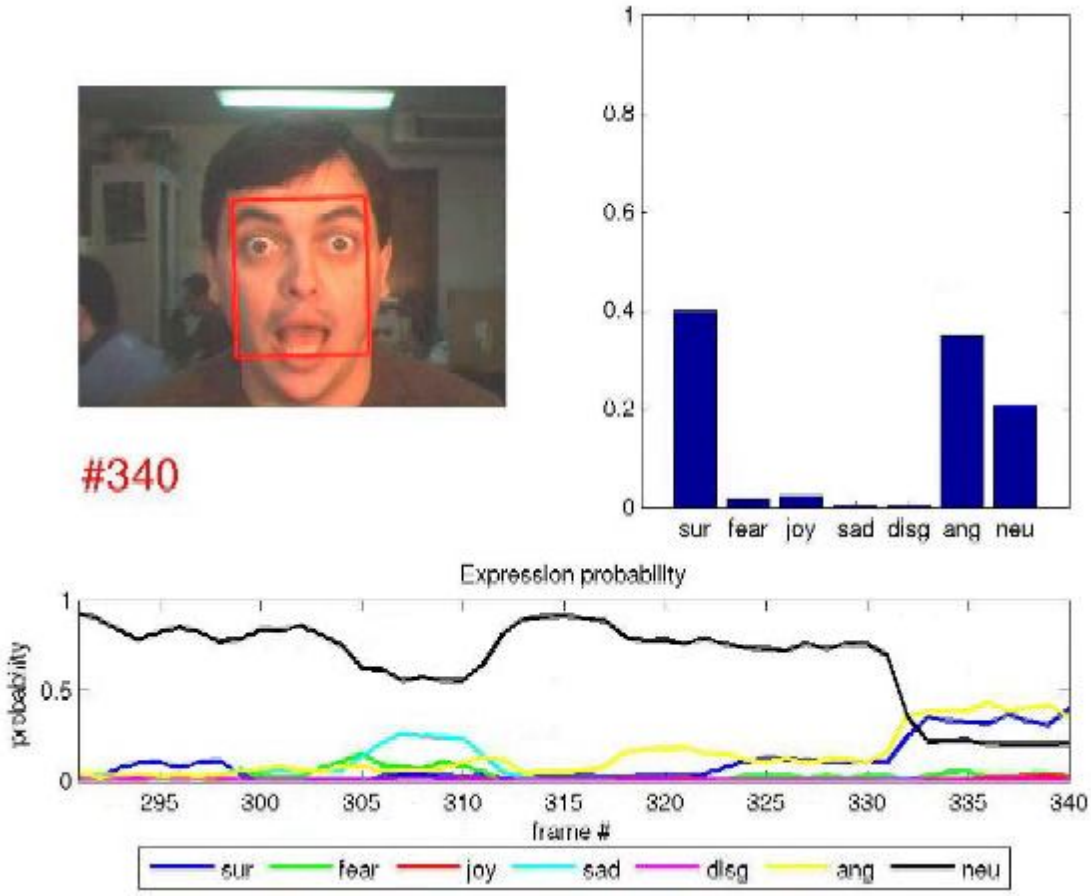
Xác định cảm xúc khuôn mặt là một bài toán khó bởi vì con người ngoài 7 cảm xúc cơ bản, còn rất nhiều cảm xúc đa dạng khác. Hơn nữa vì nhận dạng cảm xúc dựa trên các đặc điểm của khuôn mặt nên thực tế không thể biết được cảm xúc đó là đúng hay không. Về phương pháp nhận dạng, cũng gặp khó khăn khi ảnh khuôn mặt không chính diện, quá bé, hay trong điều kiện ánh sáng không tốt.

1.5 Các vấn đề liên quan

Bên cạnh việc nhận dạng cảm xúc trong không gian 2D còn có một số vấn đề liên quan mật thiết.

Nhận dạng cảm xúc trong không gian 3D[10]: Đây là vấn đề rất gần gũi với nhận dạng cảm xúc trong không gian 2D, tuy nhiên trong không gian 3D chúng ta có nhiều thông tin hơn, ngoài màu sắc, đặc trưng còn có hình dáng của khuôn mặt,…

Nhận dạng cảm xúc trong video: Vấn đề này dễ dàng hơn vì chúng ta có rất nhiều thông tin về khuôn mặt dựa vào các khung hình liên tiếp, và vấn đề này cũng thực tiễn hơn nhiều so với nhận dạng cảm xúc trong không gian 2D.



Chương 2. MỘT SỐ LÝ THUYẾT CƠ BẢN

2.1 Giới thiệu về mạng nơron[6]

Có thể nói, hiện nay, không có một định nghĩa chính thức nào cho mạng neural. Tuy nhiên phần lớn mọi người đều đồng tình rằng mạng neural là một mạng bao gồm rất nhiều bộ xử lý đơn giản (gọi là các unit), mỗi unit có vùng nhớ riêng của mình. Các unit được kết nối với nhau thông qua kênh thông tin (gọi là các connection), thường mang dữ liệu số (không phải là các ký hiệu), và được mã hóa theo một cách nào đấy. Các unit chỉ xử lý trên bộ dữ liệu của riêng nó và trên các đầu vào được đưa tới thông qua các liên kết. hạn chế của các phép xử lý cục bộ này là nó thường ở trạng thái nghỉ trong suốt quá trình học.

Một số mạng neural là các mô hình mạng neural sinh học, một số thì không, nhưng từ trước tới nay, thì tất cả các lĩnh vực của mạng neural đều được nghiên cứu xây dựng xuất phát từ các yêu cầu xây dựng các hệ thống nhận tạo rất phức tạp, hay các phép xử lý "thông minh", và những gì tương tự như bộ não con người.

Hầu hết các mạng neural đều có một vài quy tắc học nào đó mà thông qua đó các trọng số của các liên két được điều chỉnh dựa trên dữ liệu. Nói cách khác, các mạng neural "học" và các ví dụ và dựa trên các dữ liệu đó thì nó có khả năng tổng quát tri thức và đưa ra "nhận thức của mình".

Mạng neural là mô hình mạng ứng dụng các phương pháp xử lý song song và các thành phần mạng xử lý hoàn toàn đợc lập với nhau. Một vài người xem khả năng xử lý song song số lượng lớn và tính liên kết cao của mạng neural là các tính chất đặc trưugn của nó. Tuy nhiên với những yêu cầu như thế thì lại không có những mô hình đơn giản, ví dụ như mô hình hồi quy tuyến tính đơn giản, một mô hình được ứng dụng rất rộng rãi của mạng neural.

Mạng neural có thể được áp dụng trong mọi trường hợp khi tồn tại một mối liên hệ giữa các biến độc lập (inputs) và các biến phụ thuộc (outputs), thậm chí là ngay cả khi mối quan hệ đó phứuc tạp. Một số lĩnh vực mà mạng neural đã được áp dụng thành công như dự đoán triệu chứng y học, dự đoán thị trường chứng khoán, đánh giá độ tin cậy tài chính, điều chỉnh điều kiện của cơ cấu máy móc.

2.1.1 Mạng Perceptron nhiều tầng (MPL – Multi Perceptron Layer)

MPL là một loại mạng lan truyền tiến được huấn luyện theo kiểu học có giám sát. Mạng là một cấu trúc gồm nhiều lớp trọng số. Ở đây ta chỉ xét đến loại mạng lan truyền khả vi. Đây là loại mạng có thể áp dụng phương pháp tính toán khá hiệu quả và mạnh gọi là lan truyền ngược lỗi , để xác định đạo hàm hàm lỗi theo các trọng số và độ dốc trong mạng. Đây là một tính chất rất quan trọng của những mạng kiểu này bởi những đạo hàm này đóng vai trò trung tâm trong các giải thuật học của các mạng đa lớp. Vấn đề lan truyền ngược sẽ được ta xét tới trong một phần riêng sau này.

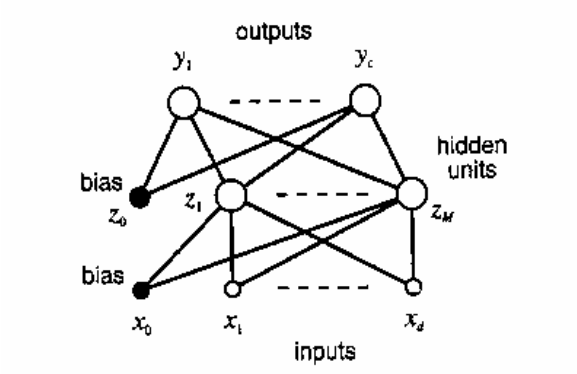
2.1.2 Ánh xạ mạng lan truyền tiến

Trong phần này ta sẽ nghiên cứu mô hình mạng neural lan truyền tiến như là một khung tổng quát đại diện cho các hàm ánh xạ phi tuyến giữa tập các biến đầu vào và tập các biến đầu ra.

2.1.2.1 Mạng phân lớp

Các mạng đơn lớp được xây dựng dựa trên sự kết hợp tuyến tính các biến đầu vào được chuyển đổi bởi một hàm truyền phi tuyến.

Ta có thể xây dựng được các hàm tổng quát hơn bằng cách nghiên cứu những mô hình mạng có các lớp các nút là liên tiếp, với các kết nối từ tất cả các nút thuộc một lớp tới tất cả các nút thuộc lớp kế tiếp, và không cho phép bất kỳ một loại kết nối nào khác. Những mạng phân lớp như thế này có thể dễ phân tích hơn các cấu trúc tổng quát khác, và cũng dễ được mô phỏng bởi phần mềm hơn.



Hình 2: Mô hình mạng lan truyền tiến

Các nút không phải là các nút nhập và nút xuất được gọi là các nút ẩn. Trong mô hình chúng ta nghiên cứu ở đây, có d nút nhập, M nút ẩn và c nút xuất.

Kết quả của nút ẩn thứ j được tính như sau:

$$a_j = \sum_{i=1}^{d} w_{ji}^{(1)} x_i + w_{j0}^{(1)} \qquad \textbf{(I.26)}$$

Trong đó là trọng số của lớp đầu tiên, từ nút nhập i đến nút ẩn j, và là trọng ngưỡng của nút ẩn j.

Giả sử đặt một biến cố định $x_0 = 1$. Từ đó công thức (I.26) có thể được viết lại:

$$a_j = \sum_{i=0}^{d} w_{ji}^{(1)} x_i \qquad \textbf{(I.27)}$$

Sau đó độ hoạt động z_k của nút ẩn j được tính toán bằng cách chuyển đổi tổng tuyến tính (I.27) sử dụng hàm truyền g(.), tức là: $z_k = g(a_j)$ **(I.28)**

Kết xuất của mạng được tính bằng cách chuyển đổi độ hoạt động của các nút ẩn sử dụng một lớp các nút thứ 2. Với mỗi nút xuất *k*, ta có:

$$a_k = \sum_{i=1}^{M} w_{kj}^{(2)} z_j + w_{k0}^{(2)} \qquad \textbf{(I.29)}$$

Đặt $z_0 = 1$ ta có:

$$a_k = \sum_{i=0}^{M} w_{kj}^{(2)} z_j \qquad \textbf{(I.30)}$$

Sau đó giá trị này được cho qua hàm truyền phi tuyến cho ta kết xuất đầu ra của nút xuất k: $y_k = \tilde{g}(a_k)$ **(I.31)**

Ở đây ta sử dụng kí hiệu để biểu diễn hàm truyền của các nút xuất nhằm chỉ ra rằng hàm này có thể không trùng với hàm đã được sử dụng trong lớp ẩn.

Kết hợp (I.27), (I.28), (I.30), (I.31) ta có công thức chung cho mô hình mạng trong hình trên:

$$y_k = \tilde{g}\left(\sum_{j=0}^{M} w_{kj}^{(2)} g\left(\sum_{i=0}^{d} w_{ji}^{(1)} x_i \right) \right) \qquad \textbf{(I.32)}$$

2.1.2.2 Kiến trúc mạng tổng quát

Ta có thể xây dựng được những ánh xạ mạng tổng quát hơn bằng cách nghiên cứu những sơ đồ mạng phức tạp hơn. Tuy nhiên ở đây thì ta chỉ giới hạn nghiên cứu trong phạm vi các mạng lan truyền tiến.

Mạng lan truyền tiến là mạng không có một kết nối quay lui nào trong mạng.

Theo Bishop (1995): OVề mặt tổng quát, một mạng được gọi là lan truyền tiến nếu nó có thể gán các số liên tục cho tất cả các nút nhập, tất cả các nút ẩn và nút xuất sao cho mỗi nút chỉ có thể nhận được các kết nối từ các nút nhập hoặc các nút được gán số bé hơn.Õ

Với những mạng có tính chất như thế, kết xuất của mạng là các hàm quyết định của các đầu vào, và vì thế toàn bộ mạng được gọi là một *ánh xạ hàm phi tuyến đa biến*.

Kết xuất của nút *k* tính được như sau:

$$z_k = g\left(\sum_j w_{kj} z_j \right) \qquad \textbf{(I.33)}$$

trong đó g(.) là một hàm truyền phi tuyến, và *j* thuộc tập tất cả các nút nhập và các nút gửi kết nối tới nút *k* (Tham số trọng ngưỡng cũng đã được bao hàm ở trong tổng này).

Với một tập cho trước các giá trị đầu vào, áp dụng liên tục công thức (I.33) sẽ cho phép các kích hoạt của tất cả các nút trong mạng được ước lượng, bao gồm cả các kích hoạt của các nút xuất. Quá trình này được gọi là lan truyền tiến các tín hiệu qua mạng.

Nếu như các hàm truyền của tất cả các nút ẩn trong mạng là tuyến tính, thì với những mạng như thế ta luôn luôn tìm được một mô hình mạng tương đương mà không có một nút ẩn nào. Những mạng này được gọi là mạng tuyến tính đa lớp và vì thế

không được đi sâu nghiên cứu, mà người ta chỉ chủ yếu nghiên cứu các mạng đa lớp với các hàm truyền của các nút ẩn là phi tuyến.

2.1.3 Hàm sigmoid

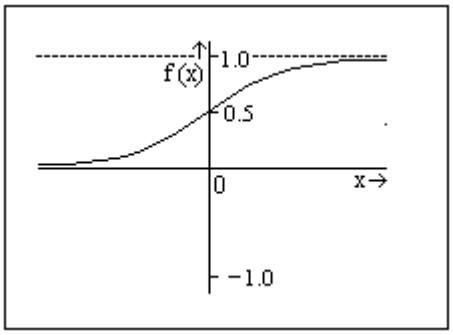
Bây giờ chúng ta sẽ xem xét hàm truyền logistic dạng S, trong đó các đầu ra của nó nằm trong khoảng (0,1), có phương trình như sau:

$$g(a) = \frac{1}{1 + \exp(-a)} \qquad \textbf{(I.34)}$$

Hình vẽ dưới đây biểu diễn một hàm truyền sigmoid cho các nút trong mạng. Đây là một hàm mũ có một đặc tính vô cùng quan trọng vì : khi x chạy từ vô cùng lớn đến vô cùng bé thì f(x) luôn chạy trong khoảng từ 0 đến 1. Giải thuật học ở đây sẽ điều chỉnh trọng số của các kết nối giữa các nút để hàm này ánh xạ giá trị của x sang dạng nhị phân, thông thường:

f(x) > 0.9 : f(x) = 1

f(x) < 0.1 : f(x) = 0.



Hình 3: Đồ thị hàm truyền sigmoid

Trong phần này chúng ta sẽ xem xét các mạng neural với nút xuất tuyến tính. Tuy nhiên điều này cũng chẳng hạn chế lớp các hàm mà mạng có thể xấp xỉ hoá. Việc sử dụng các hàm sigmoid tại các đầu ra sẽ giới hạn phạm vi có thể xảy ra của các nút

xuất thành phạm vi có thể đạt tới được của hàm sigmoid (giá trị kết xuất là từ 0 tới 1), và trong một số trường hợp thì điều này có thể là không mong muốn. Thậm chí ngay cả khi giá trị xuất mong muốn là nằm trong giới hạn của hàm sigmoid thì chúng ta vẫn phải chú ý rằng hàm sigmoid g(.) là một hàm đơn điệu tăng, do đó nó có thể lấy nghịch đảo được. Do vậy một giá trị xuất y mong muốn đối với mạng có nút xuất thuộc dạng sigmoid thì tương đương với một giá trị xuất $g^{-1}(y)$ đối với mạng có nút xuất tuyến tính.

Một nút ẩn thuộc dạng sigmoid có thể xấp xỉ một nút ẩn tuyến tính bất kì một cách chính xác. Công việc này đạt được bằng cách thiết kế cho tất cả các trọng số các cung đầu vào của nút, cũng như các trọng ngưỡng, sao cho rất nhỏ để mà tổng của các giá trị nhập phải nằm trên phần tuyến tính của đường cong sigmoid, gần đúng với đường thẳng nguyên thuỷ. Trọng số trên cung xuất từ một nút đến tầng chứa các nút kế tiếp có thể tạo ra tương đối lớn để tái tỉ lệ với độ hoạt động (và với trọng ngưỡng để có được bước dịch chuyển phù hợp nếu cần thiết). Tương tự, một nút ẩn dạng sigmoid có thể được tạo ra nhằm xấp xỉ một hàm bậc thang (step) bằng việc đặt giá trị cho các trọng số và trọng ngưỡng rất lớn.

Bất kì một ánh xạ hàm liên tục nào đều có thể được trình bày với độ chính xác tuỳ ý bởi một mạng neural hai lớp trọng số sử dụng các nút ẩn dạng sigmoid (Bishop, 1995).

Do đó chúng ta biết được rằng những mạng neural với nhiều tầng nút xử lý cũng có khả năng xấp xỉ hoá bởi vì chúng đã chứa đựng trong nó mạng neural hai tầng như một trường hợp đặc biệt. Điều này cho phép các tầng còn lại được sắp xếp để thực hiện những biến đổi tuyến tính như đã thảo luận ở trên, và sự biến đổi đồng nhất chính là một trường hợp dặc biệt của một phép biến đổi tuyến tính (biết rằng có đủ số nút ẩn để không có sự giảm bớt về chiều xảy ra).

2.1.4 Thuật toán lan truyền ngược

Bây giờ chúng ta sẽ tập trung nghiên cứu một kĩ thuật rất phổ biến của mạng neural nhiều tầng. Chúng ta sẽ xem xét cách mà một mạng học một ánh xạ từ một tập dữ liệu cho trước.

Chúng ta đã biết việc học dựa trên định nghĩa của hàm lỗi, hàm lỗi này sau đó sẽ được tối thiểu hoá dựa vào các trọng số và các trọng ngưỡng trong mạng.

Trước tiên ta sẽ xem xét trường hợp mạng sử dụng hàm ngưỡng. Vấn đề cần bàn ở đây chính là cách để khởi tạo các trọng số cho mạng như thế nào. Công việc này thường được gọi là 'credit assignment problem'. nếu một nút đầu ra tạo ra một đáp số sai lệch thì chúng ta phải quyết định xem liệu nút ẩn nào phải chịu trách nhiệm cho sự sai lệch đó, cũng chính là việc quyết định trọng số nào cần phải điều chỉnh và điều chỉnh là bao nhiêu.

Để giải quyết vấn đề gán trọng số này, chúng ta hãy xem xét một mạng với các hàm truyền phân biệt ,do đó giá trị tổng trọng của các nút xuất sẽ trở thành một hàm phân biệt của các biến nhập và của trọng số và trọng ngưỡng. Nếu ta coi hàm lỗi, ví dụ có dạng sai số trung bình bình phương, là một hàm riêng biệt cho các giá trị xuất của mạng thì bản thân nó cũng chính là một hàm phân biệt của các trọng số.

Do đó chúng ta có thể tính toán được đạo hàm hàm lỗi theo các trọng số, và giá trị đạo hàm này lại có thể dùng để làm cực tiểu hoá hàm lỗi bằng cách sử dụng phương pháp giảm gradient (gradient descent) hoặc các phương pháp tối ưu hoá khác.

Giải thuật ước lượng đạo hàm hàm lỗi được biết đến với tên gọi *lan truyền ngược*, nó tương đương với việc lan truyền ngược lỗi trong mạng. Kĩ thuật về lan truyền ngược được biết đến rất rộng rãi và chi tiết qua các bài báo cũng như các cuốn sách của Rumelhart, Hinton và Williams (1986). Tuy nhiên gần đây một số ý tưởng tương tự cũng được một số nhà ngiên cứu phát triển bao gồm Werbos (1974) và Parker (1985).

Cần nói thêm rằng giải thuật lan truyền ngược được sử dụng trong mạng neural có ý nghĩa rất lớn. Ví dụ như, kiến trúc của mạng perceptron nhiều tầng cũng thường được gọi là mạng lan truyền ngược. Khái niệm *lan truyền ngược cũng thường được sử dụng để mô tả quá trình huấn luyện của mạng perceptron nhiều tầng sử dụng phương pháp gradient descent áp dụng trên hàm lỗi dạng sai số trung bình bình phương.* Để làm rõ hơn về thuật ngữ này chúng ta cần xem xét quá trình luyện mạng một cách kĩ càng. Phần lớn các giải thuật luyện mạng đều liên quan đến một thủ tục được lặp đi lặp lại nhằm làm tối thiểu hàm lỗi, bằng cách điều chỉnh trọng số trong một chuỗi các bước.

Tại mối bước như vậy, chúng ta có thể chia thành hai bước phân biệt.

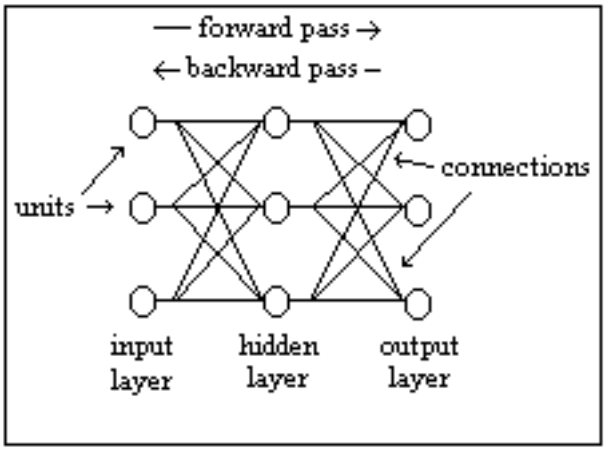
Tại bước thứ nhất, cần phải tính đạo hàm hàm lỗi theo các trọng số. Chúng ta đã biết rằng một đóng góp rất quan trọng của kĩ thuật lan truyền ngược đó là việc cung cấp một phương pháp hết sức hiệu quả về mặt tính toán trong việc đánh giá các đạo

hàm. Vì tại bước này lỗi sẽ được lan truyền ngược trở lại mạng nên chúng ta sẽ sử dụng khái niệm lan truyền ngược để đặc trưng riêng cho việc đánh giá đạo hàm này.

Tại bước thứ hai, các đạo hàm sẽ được sử dụng trong việc tính toán sự điều chỉnh đối với trọng số. Và kĩ thuật đơn giản nhất được sử dụng ở đây là kĩ thuật gradient descent, kĩ thuật này được Rumelhart et al. (1986) đưa ra lần đầu tiên.

Một điều hết sức quan trọng là phải nhận thức được rằng hai bước này là phân biệt với nhau. Do đó, quá trình xử lý đầu tiên , được biết đến là quá trình lan truyền ngược các lỗi vào trong mạng để đánh giá đạo hàm, có thể được áp dụng đối với rất nhiều laọi mạng khác nhau chứ không chỉ đối với riêng mạng perceptron nhiều tầng. Nó cũng có thể được áp dụng với các loại hàm lỗi khác chứ không chỉ là hàm tính sai số bình phương cực tiểu, và để đánh giá các đạo hàm khác này có thể sử dụng các phương pháp khác như phương pháp ma trận Jacobian và Hessian mà chúng ta sẽ xem xét ở phần sau. Và cũng tương tự như vậy thì tại bước thứ hai, việc điều chỉnh trọng số sử dụng các đạo hàm đã được tính trước đó có thể thực hiện với nhiều phương pháp tối ưu hoá khác nhau, và rất nhiều trong số các phương pháp đó cho kết quả tốt hơn phương pháp gradient descend.

2.1.4.1 Lan truyền ngược



Hình 4: Lan truyền ngược

Bây giờ chúng ta sẽ áp dụng giải thuật lan truyền ngược cho bất kì một mạng neural có cấu hình lan truyền tiến tuỳ ý, sử dụng các hàm truyền phi tuyến tuỳ ý, và cả

hàm lỗi có dạng tuỳ ý. Để minh hoạ chúng ta sẽ dùng một mạng có cấu trúc một tầng nút ẩn dạng sigmoid và hàm lỗi là hàm tính theo sai số trung bình bình phương.

Trong các mạng lan truyền tiến nói chung mỗi nút đều tình tổng trọng hoá các đầu vào của nó theo công thức:

$$a_j = \sum_i w_{ji} z_i \qquad \textbf{(I.35)}$$

Với z_i là giá trị nhập hoặc là giá trị xuất của một nút có cung kết nối với nút j và w_{ji} chính là trọng số của cung kết nối đó. Giá trị tổng này được tính trên tất cả các nút có kết nối trực tiếp với nút j. Chúng ta biết rằng, trọng ngưỡng của nút cũng được đưa vào trong tổng bằng cách tạo ra thêm một giá trị nhập cố định = 1. Tổng trong (I.35) lại được biến đổi thông qua một hàm truyền phi tuyến g(.) để đưa ra được gía trị xuất z_i của nút j theo công thức:

$$z_i = g(a_j) \textbf{ (I.36)}$$

Bây giờ chúng ta cần phải xác định giá trị của các trọng số trong mạng thông qua việc tối thiểu hoá hàm lỗi.

ở đây ta sẽ coi cá hàm lỗi được viết như một tổng của tất cả các lỗi tại mỗi mẫu riêng biệt.Tổng này sẽ được tính trên tất cả các mẫu của tập huấn luyện

$$E = \sum_n E^n \qquad \textbf{(I.37)}$$

Với n là nhãn của từng mẫu.

Chúng ta cũng giả định rằng lỗi E^n có thể được thể hiện như một hàm riêng của các biến đầu ra, có nghĩa là :

$$E^n = E^n(y_c, \ldots, y_c)$$

Mục đích của chúng ta ở đây chính là phải tìm ra một hàm nhằm để tính được đạo hàm của hàm lỗi theo các trọng số và trọng ngưỡng của mạng.

Đối với từng mẫu, ta sẽ coi như đã cung cấp một vector nhập tương ứng là đầu vàovà đã tính được các giá trị xuất của các nút ẩn cũng như nút xuất theo các công thức (I.35), (I.36). Quá trình này thường được gọi là quá trình lan truyền tiến trong mạng.

Bây giờ hãy xem xét việc tính đạo hàm của E^n theo cá trọng số w_{ji}. Giá trị xuất của các nút sẽ phụ thuộc vào từng mẫu nhập n nào. Tuy nhiên để dễ nhìn, ta quy ước sẽ bỏ qua việc viết kí tự n trên các biến nhập và xuất. Trước tiên ta cần chú ý rằng E^n phụ thuộc vào trọng số w_{ji} thông qua tổng giá trị nhập a_i của nút j. Do đó ta có thể đưa ra công thức tính các đạo hàm riêng như sau:

$$\frac{\partial E^n}{\partial w_{ji}} = \frac{\partial E^n}{\partial a_j} * \frac{\partial a_j}{\partial w_{ji}} \qquad \textbf{(I.38)}$$

Từ (I.35) ta có:

$$\frac{\partial a_j}{\partial w_{ji}} = z_i \qquad \textbf{(I.39)}$$

Như vậy suy ra:

$$\frac{\partial E^n}{\partial w_{ji}} = \delta_j z_i \qquad \textbf{(I.40)}$$

Trong đó $\delta_j \equiv \frac{\partial E^n}{\partial a_j}$

Từ công thức (I.40) ta thấy rằng để tính được đạo hàm chúng ta chỉ cần tính giá trị cho mỗi nút ẩn và nút xuất trong mạng và sau đó áp dụng công thức (I.40).

Với các nút xuất thì việc tính δ_k là hết sức đơn giản.

Ta có:

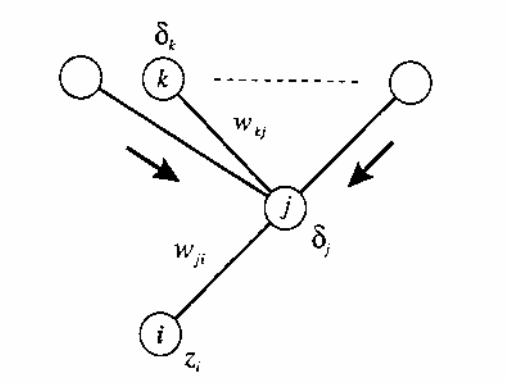
$$\delta_k \equiv \frac{\partial E^n}{\partial a_k} = g'(a_k)\frac{\partial E^n}{\partial y_k} \qquad \textbf{(I.41)}$$

Để tính ra (I.41) ta cần tìm ra công thức tính g'(a) và $\frac{\partial E^n}{\partial y}$.

Để tính được δ cho cá nút ẩn, ta cần sử dụng công thức tính đạo hàm riêng:

$$\delta_j \equiv \frac{\partial E^n}{\partial a_j} = \sum_k \frac{\partial E^n}{\partial a_k}\frac{\partial a_k}{\partial a_j} \qquad \textbf{(I.42)}$$

Trong đó giá trị tổng được tính trên các nút k mà nút j kết nối đến. Việc sắp xếp các nút cũng như các trọng số được minh hoạ trong Hình 6.



Hình 5: Minh họa việc tính δ_j cho việc tính nút ẩn j

Chú ý rằng các nút có nhãn k này có thể bao gồm cả nút nhập và nút xuất.

Bây giờ chúng ta có công thức lan truyền ngược như sau:

$$\delta_j \equiv g'(a_j)\sum_k w_{kj}\delta_k \qquad \textbf{(I.43)}$$

Công thức này nói lên rằng giá trị của δ đối với một nút ẩn có thể đươc tính từ việc lan truyền ngược các giá trị δ của các nút ẩn cao hơn trong mạng, như được minh hoạ trong hình 5. Bởi vì chúng ta đã biết đươc các giá trị δ của các nút xuất nên ta có thể áp dụng (I.43) một cách đệ quy nhằm tính ra các giá trị δ cho tất cả các nút ẩn trong mạng, mà không quan tâm đến cấu hình của nó.

Chúng ta có thể tổng kết lại giải thuật lan truyền ngược nhằm tính đạo hàm hàm lỗi E^n theo các trọng số trong 4 bước:

- Đưa vector nhập x^n vào mạng và lan truyền tiến nó trong mạng sử dụng và để tìm ra giá trị xuất cho tất cả các nút ẩn cũng như nút xuất.

- Tính δ cho tất cả các nút xuất sử dụng công thức

- Lan truyền ngược các d bằng công thức để thu được δ cho mỗi nút ẩn trong mạng.

- áp dụng $\frac{\partial E^n}{\partial w_{ji}} = \delta_j z_i$ để tính các đạo hàm.

Đạo hàm của lỗi tổng E có thể thu được bằng cách lặp đi lặp lại các bước trên đối với trừng mẫu trong tập huấn luyện và sau đó tính tổng trên tất cả các lỗi.

Trong quá trình tính đạo hàm trên chúng ta đã giả định rằng mỗi nút ẩn cũng như xuất đếu có chung một hàm truyền g(.). Tuy nhiên điều này hoàn toàn có thể tính được với trường hợp mỗi nút khác nhau đếu có các hàm truyền riêng, đơn giản bằng cách đánh dấu dạng của hàm g(.) ứng với từng nút.

2.1.4.2 Hiệu quả của lan truyền ngược

Một trong những đặc tính quan trọng nhất của lan truyền ngược chính là ở khả năng tính toàn hiệu quả của nó.

Đặt w là tổng số các trọng số và trọng ngưỡng. Do đó một phép tính hàm lỗi (cho một mẫu nhập nào đó) cần O*(w)* thao tác với w đủ lớn. Điều này cho phép số lượng trọng số có thể lớn hơn số lượng nút, trừ những mạng có quá ít kết nối. Do vậy, hiệu quả của việc tính toán trong lan truyền ngược sẽ liên quan đến việc tính giá trị của tổng trong công thức (I.35), còn việc tính toán các hàm truyền thì tổng phí khá nhỏ. Mỗi lượt tính tổng trong (I.35) cần đến một phép nhân và một phép cộng, dẫn đến chi phí tính toán toàn bộ sẽ bằng O*(w)*.

Với tất cả w trọng số thì sẽ có w đạo hàm cần tính toán. Với mỗi lần tính đạo hàm như vậy cần phải thực hiện tìm biểu thức hàm lỗi, xác định công thức tính đạo

hàm và sau đó tính toán chúng theo giải thuật lan truyền ngược, mỗi công việc đó sẽ đòi hỏi O*(w)* thao tác. Như vậy toàn bộ quá trình tính toán tất cả các đạo hàm sẽ tỉ lệ với $O(w^2)$. Giải thật lan truyền ngược cho phép các đạo hàm được tính trong O*(w)* thao tác. Điều này cũng dẫn đến rằng cả hai pha lan truyền ngược và lan truyền tiến đều cần O*(w)* thao tác, việc tính đạo hàm theo công thức (I.43) cũng cần O*(w)* thao tác.Như vậy giải thuật lan truyền ngược đã làm giảm độ phức tạp tính toán từ $O(w^2)$ đến O*(w)* đối với mỗi vector nhập. Vì quá trình luyện mạng, dù có sử dụng lan truyền ngược, có thể cần rất nhiều thời gian, nên việc đạt được hiệu quả như vậy là hết sức quan trọng.Với tổng số N mẫu luyện, số lượng các bước tính toán để đánh giá hàm lỗi trên toàn bộ tập dữ liệu sẽ là N lần bước tính toán của một mẫu.

2.2 Giới thiệu về PCA

Phần này giúp người đọc hiểu được phép phân tích thành phần chính (PCA). PCA là một kỹ thuật hữu ích trong các ứng dụng nhận dạng mặt và nén ảnh, và là một kỹ thuật phổ biến để tìm mẫu trong các dữ liệu nhiều chiều[4].

Trước khi đi vào tìm hiểu PCA, tôi xin giới thiệu về các khái niệm toán học sẽ được sử dụng trong PCA. Các khái niệm đó bao gồm: Độ lệch chuẩn (Standard deviation), phương sai (variance), hiệp phương sai (covariance), vec tơ riêng (eigenvector), giá trị riêng (eigenvalue).

2.2.1 Một số khái niệm toán học

2.2.1.1 Độ lệch chuẩn

Để hiểu độ lệch chuẩn, chúng ta cần một tập dữ liệu. Giả sử ta có tập

X = [1 2 4 6 12 15 25 45 68 67 65 98]

X là ký hiệu đại diện cho tập số, mỗi số riêng biệt được ký hiệu X_i (Ví dụ X_3 = 4). Phần tử đầu tiên là X_1 và n là số lượng phần tử của tập hợp. Khi đó trung bình của mẫu có công thức:

$$\bar{X} = \frac{\sum_{i=1}^{n} X_i}{n}$$

$\bar{X}$ Là ký hiệu trung bình của mẫu, tuy nhiên trung bình mẫu không nói lên được nhiều điều ngoại trừ cho ta biết nó là một điểm giữa. Ví dụ với 2 tập dữ liệu

[0 8 12 20] và [8 9 11 12]

có trung bình mẫu bằng nhau nhưng lại khá khác nhau. Sự khác biệt ở đây chính là khoảng cách của dữ liệu. Và độ lệch chuẩn là đại lượng để đo khoảng cách này. Ta có thể hiêu độ lệch chuẩn là khoảng cách trung bình từ trung bình mẫu đến các điểm của dữ liệu. Ta có công thức:

$$s = \sqrt{\frac{\sum_{i=1}^{n}(X_i - \bar{X})^2}{(n-1)}}$$

Tập hợp 1

X	$(X - \bar{X})$	$(X - \bar{X})^2$
0	-10	100
8	-2	4
12	2	4
20	10	100
Total		208
Divided by (n-1)		69.333
Square Root		8.3266

Tập hợp 2

X_i	$(X_i - \bar{X})$	$(X_i - \bar{X})^2$
8	-2	4
9	-1	1
11	1	1
12	2	4
Total		10
Divided by (n-1)		3.333
Square Root		1.8257

Ta có thể dễ dàng nhận thấy tập dữ liệu 1 có độ lệch chuẩn lớn hơn có khoảng cách lớn hơn tập dữ liệu 2.

2.2.1.2 Phương sai

Phương sai là một đại lượng khác dùng để đo khoảng cách của dữ liệu. Ta có công thức:

$$s^2 = \frac{\sum_{i=1}^{n}(X_i - \bar{X})^2}{(n-1)}$$

Dễ thấy phương sai chính là bình phương độ lệch chuẩn.

2.2.1.3 Hiệp phương sai

Ta thấy rằng 2 đại lượng độ lệch chuẩn và phương sai chỉ sử dụng được trong 1 chiều. Trong thực tế dữ liệu có thể có rất nhiều chiều. Một ví dụ đơn giản ta có dữ liệu về cân nặng và điểm số của toàn bộ sinh viên trong lớp K51-KHMT. Đối với dữ liệu này, độ lệch chuẩn và phương sai chỉ tính được trên từng chiều riêng biệt và ta không thấy được mối liên hệ giữa 2 chiều này.

Tương tự phương sai, hiệp phương sai là đại lượng đo sự biến thiên giữa 2 chiều. Nếu tính hiệp phương sai giữa 1 chiều với chính nó ta được phương sai của chiều đó. Nếu tập dữ liệu có 3 chiều x, y, z ta có thể tính hiệp phương sai của từng cặp chiều (x, y), (y, z), (z, x). Công thức của hiệp phương sai tương tự công thức của phương sai. Công thức của phương sai được khai triển như sau:

$$var(X) = \frac{\sum_{i=1}^{n}(X_i - \bar{X})(X_i - \bar{X})}{(n-1)}$$

Và công thức của hiệp phương sai:

$$cov(X,Y) = \frac{\sum_{i=1}^{n}(X_i - \bar{X})(Y_i - \bar{Y})}{(n-1)}$$

Từ công thức hiệp phương sai ta thấy, nếu $cov(X,Y)$ dương thì X, Y đồng biến, $cov(X,Y)$ âm thì X, Y nghịch biến, nếu bằng 0 thì X, Y độc lập.

2.2.1.4 Ma trận hiệp phương sai

Hiệp phương sai đó sự biến thiên giữa 2 chiều, do đó đối với tập dữ liệu có n chiều ta có $\frac{n!}{(n-2)!*2}$ giá trị hiệp phương sai khác nhau. Và để thuận tiện cho việc tính toán ta biểu diễn các giá trị này thông qua một ma trận gọi là ma trận hiệp phương sai. Định nghĩa của ma trận như sau:

$$C^{n\times n} = (c_{i,j},\ c_{i,j} = cov(Dim_i, Dim_j)),$$

Trong đó $C^{n\times n}$ là 1 ma trận với n hàng, n cột và Dim_x là chiều thứ x. Ví dụ ma trận hiệp phương sai của 1 tập dữ liệu có 3 chiều x, y, z:

$$C = \begin{pmatrix} cov(x,x) & cov(x,y) & cov(x,z) \\ cov(y,x) & cov(y,y) & cov(y,z) \\ cov(z,x) & cov(z,y) & cov(z,z) \end{pmatrix}$$

2.2.2 Ma trận đại số

Phần này giới thiệu về 2 khái niệm là nền tảng được sử dụng trong PCA đó là vectơ riêng (eigenvector) và giá trị riêng (eigenvalue).

$$\begin{pmatrix} 2 & 3 \\ 2 & 1 \end{pmatrix} \times \begin{pmatrix} 1 \\ 3 \end{pmatrix} = \begin{pmatrix} 11 \\ 5 \end{pmatrix}$$

$$\begin{pmatrix} 2 & 3 \\ 2 & 1 \end{pmatrix} \times \begin{pmatrix} 3 \\ 2 \end{pmatrix} = \begin{pmatrix} 12 \\ 8 \end{pmatrix} = 4 \times \begin{pmatrix} 3 \\ 2 \end{pmatrix}$$

Hình 6: Ví dụ về 1 non-eigenvector và 1 eigenvector

$$2 \times \begin{pmatrix} 3 \\ 2 \end{pmatrix} = \begin{pmatrix} 6 \\ 4 \end{pmatrix}$$

$$\begin{pmatrix} 2 & 3 \\ 2 & 1 \end{pmatrix} \times \begin{pmatrix} 6 \\ 4 \end{pmatrix} = \begin{pmatrix} 24 \\ 16 \end{pmatrix} = 4 \times \begin{pmatrix} 6 \\ 4 \end{pmatrix}$$

Hình 7: Ví dụ về 1 eigenvector có tỉ lệ khác vẫn 1 là eigenvector

2.2.3 Eigenvector (Vectơ riêng)

Ta có thể nhân 2 ma trận với điều kiện kích cỡ phù hợp và eigenvector là 1 trường hợp đặc biệt của phép nhân này. Quan sát 2 phép nhân ma trận với vector trên hình 3.1. Ở ví dụ thứ nhất vectơ kết quả không phải là một bội số của vectơ gốc trong khi ở ví dụ thứ 2 vectơ kết quả bằng 4 lần vectơ gốc. Ta thấy rằng vectơ $\begin{pmatrix} 3 \\ 2 \end{pmatrix}$ (trong ví dụ 2) biểu diễn 1 mũi tên từ điểm (0, 0) đến điểm (3, 2) và ma trận còn lại được hiểu là ma trận chuyển đổi. Nếu ta nhân ma trận này về bên trái của vectơ thì vectơ mới nhận được chính là vectơ cũ bị tịnh tiến đi 1 lượng. Đó là tính biến đổi của vectơ riêng.

Các tính chất của vectơ riêng:

- Chỉ các ma trận vuông (n x n) mới có vectơ riêng.
- Không phải mọi ma trận vuông đều có vectơ riêng.
- Nếu 1 ma trận vuông (n x n) có vectơ riêng thì sẽ có n vectơ riêng.
- Nếu nhân vectơ riêng với 1 số thì kết quả sau khi nhân với ma trận chuyển đổi, vectơ kết quả vẫn là vectơ ban đầu
- Tất cả các vectơ riêng của 1 ma trận đều trực giao với nhau

2.2.4 Eigenvalue (Giá trị riêng)

Giá trị riêng là một khái niệm liên quan chặt chẽ đến vectơ riêng. Thực tế chúng ta đã thấy 1 giá trị riêng trong hình 3.1. Chú ý trong cả 2 ví dụ trên, số được nhân với 2 vectơ riêng bằng nhau và bằng 4. 4 được gọi là giá trị riêng ứng với 1 vectơ riêng (2

vectơ riêng trong 2 ví dụ trên là tương đương nhau). Giá trị riêng và vectơ riêng luôn đi với nhau thành 1 cặp.

2.2.5 Phân tích thành phần chính (PCA)

PCA là 1 phương pháp để nhận dạng các mẫu trong dữ liệu và biểu diễn dữ liệu bằng cách làm nổi bật sự giống và khác nhau. Khi các mẫu trong dữ liệu rất khó nhận ra trong không gian nhiều chiều thì PCA là một công cụ mạnh để phân tích chúng.

Các bước cơ bản trong PCA:

Bước1: Lấy dữ liệu (Get data)

Bước2: Trừ trung bình mẫu.

Với mỗi chiều dữ liệu giả sử ở chiều x, ta đều có 1 trung bình mẫu, công việc trong bước này là trừ tất cả giá trị trong chiều x cho trung bình mẫu x. Kết thúc bước này ta sẽ có trung bình mẫu ở tất cả các chiều là 0.

Bước 3: Tính ma trận hiệp phương sai

Bước 4: Tính các vectơ riêng và giá trị riêng của ma trận hiệp phương sai.

Bước 5: Chọn các thành phần chính

Đây là bước cuối cùng trong PCA. Trong bước này, tùy thuộc vào số lượng thành phần chính cần lấy, ta lấy lần lượt các thành phần (vectơ riêng) tương ứng với các giá trị riêng cao nhất.

Chương 3. CÁC PHƯƠNG PHÁP NHẬN DẠNG CẢM XÚC KHUÔN MẶT

Trong khuôn khổ luận văn này các phương pháp nhận dạng cảm xúc chỉ thực hiện trên ảnh khuôn mặt mẫu 2D.

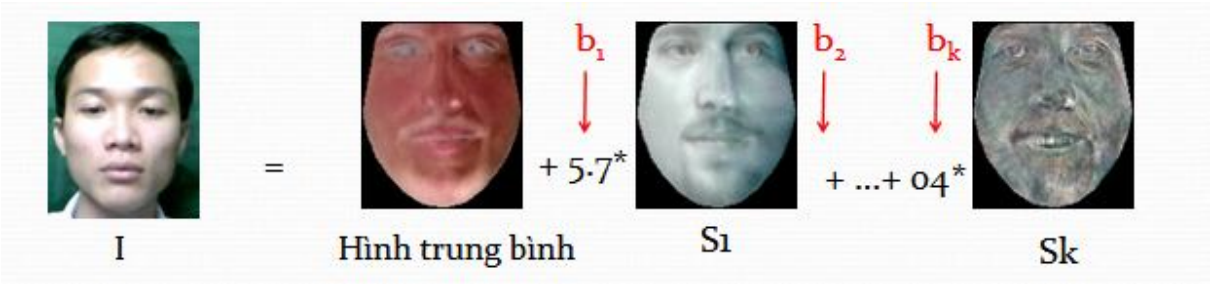
3.1 Nhận dạng cảm xúc dựa trên PCA truyền thống

3.1.1 Trích chọn đặc trưng

Xây dựng một tập các vectơ đặc trưng (S_1, S_2,…S_k) cho mỗi hình huấn luyện sử dụng phép phân tích PCA.

Hình 8: Ví dụ về trích chọn đặc trưng bằng PCA

Ứng với mỗi vectơ đặc trưng riêng có 1 giá trị riêng. Như vậy mỗi hình huấn luyện được đại diện bởi một tập các giá trị riêng.



Mỗi cảm xúc bao gồm 1 tập ảnh huấn luyện

Ví dụ cảm xúc vui

$$\mathbf{I_{(Happy\ 1)} = (b_{Happy\ 1\ 1}, b_{Happy\ 1\ 2}, b_{Happy\ 1\ 3}\dots b_{Happy\ 1\ n})}$$

$$\mathbf{I_{(Happy\ 2)} = (b_{Happy\ 2\ 1}, b_{Happy\ 2\ 2}, b_{Happy\ 2\ 3}\dots b_{Happy\ 2\ n})}$$

:

$$I_{(Happy\ m)} = (b_{Happy\ m\ 1}, b_{Happy\ m\ 2}, b_{Happy\ m\ 3} \dots b_{Happy\ m\ n})$$

Cảm xúc buồn

$$I_{(Sad\ 1)} = (b_{Sad\ 1\ 1}, b_{Sad\ 1\ 2}, b_{Sad\ 1\ 3} \dots b_{Sad\ 1\ n})$$

$$I_{(Sad\ 2)} = (b_{Sad\ 2\ 1}, b_{Sad\ 2\ 2}, b_{Sad\ 2\ 3} \dots b_{Sad\ 2\ n})$$

:

$$I_{(Sad\ m)} = (b_{Sad\ m\ 1}, b_{Sad\ m\ 2}, b_{Sad\ m\ 3} \dots b_{Sad\ m\ n})$$

Với 1 hình ảnh cần nhận dạng cảm xúc, sử dụng PCA ta được 1 tập các giá trị riêng.

$$I_{(Nhan\_dang)} = (b_{Nhan\_dang\ 1}, b_{Nhan\_dang\ 2}, b_{Nhan\_dang\ 3} \dots b_{Nhan\_dang\ n})$$

3.1.2 Quá trình nhận dạng

Lần lượt tính khoảng cách Euclid từ ảnh cần nhận dạng đến mỗi ảnh trong tập huấn luyện

$$S_{(Happy\ 1)} = (S_{Happy\ 1,1} - b_{Nhan\_dang\ 1})^2 + (S_{Happy\ 1,2} - b_{Nhan\_dang\ 2})^2 + \dots + (S_{Happy\ 1,n} - b_{Nhan\_dang\ n})^2$$

$$S_{(Happy\ 2)} = (S_{Happy\ 2,1} - b_{Nhan\_dang\ 2})^2 + (S_{Happy\ 2,2} - b_{Nhan\_dang\ 2})^2 + \dots + (S_{Happy\ 2,n} - b_{Nhan\_dang\ n})^2$$

:

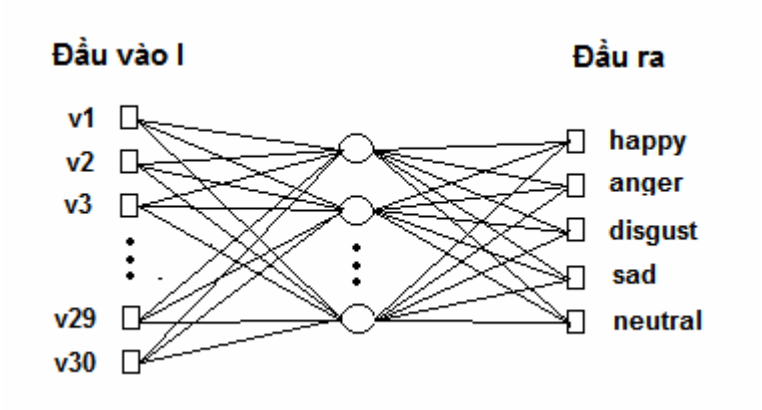
$$S_{(Happy\ m)} = (S_{Happy\ m,1} - b_{Nhan\_dang\ 1})^2 + (S_{Happy\ m,2} - b_{Nhan\_dang\ 2})^2 + \dots + (S_{Happy\ m,n} - b_{Nhan\_dang\ n})^2$$

Khi đó cảm xúc của ảnh cần nhận dạng sẽ được xác định bằng cảm xúc của ảnh trong tập huấn luyện mà khoảng cách Euclid từ ảnh đó đến ảnh cần nhận dạng là bé nhất.

3.2 Nhận dạng cảm xúc dựa trên PCA kết hợp các thuật toán học

3.2.1 Mạng nơron

Mô hình:



Hình 9: Mô hình mạng nơron

Hình vẽ trên cho ta mô hình của mạng nơ ron sử dụng trong khóa luận này. Đây là mạng MLP (MultiLayer Perceptron) bao gồm 3 lớp. Lớp đầu vào gồm 30 nút là 30 giá trị riêng của 1 ảnh sau khi dùng PCA để trích chọn đặc trưng. Lớp ẩn và lớp đầu ra gồm 5 nút là 5 cảm xúc.

Trong mô hình mạng neural MPL này, chúng ta sẽ sử dụng thuật toán lan truyền ngược (Backprobagation) để tiến hành học mạng, phương pháp giảm lỗi được sử dụng là phương pháp giảm gradient với hàm truyền hay hàm kích hoạt là hàm sigmoid. Toàn bộ thuật toán và lý thuyết về vấn đề này đã được đề cập đến trong chương I của đồ án.

3.2.2 Cây quyết định

Trong lĩnh vực học máy, cây quyết định là một kiểu mô hình dự báo (*predictive model*), nghĩa là một ánh xạ từ các quan sát về một sự vật/hiện tượng tới các kết luận về giá trị mục tiêu của sự vật/hiện tượng. Mỗi một nút trong (*internal node*) tương ứng với một biến; đường nối giữa nó với nút con của nó thể hiện một giá trị cụ thể cho biến đó. Mỗi nút lá đại diện cho giá trị dự đoán của biến mục tiêu, cho trước các giá trị của các biến được biểu diễn bởi đường đi từ nút gốc tới nút lá đó. Kỹ thuật học máy dùng trong cây quyết định được gọi là học bằng cây quyết định, hay chỉ gọi với cái tên ngắn gọn là cây quyết định.

Học bằng cây quyết định cũng là một phương pháp thông dụng trong khai phá dữ liệu. Khi đó, cây quyết định mô tả một cấu trúc cây, trong đó, các lá đại diện cho các phân loại còn cành đại diện cho các kết hợp của các thuộc tính dẫn tới phân loại đó. Một cây quyết định có thể được học bằng cách chia tập hợp nguồn thành các tập con dựa theo một kiểm tra giá trị thuộc tính. Quá trình này được lặp lại một cách đệ qui cho mỗi tập con dẫn xuất. Quá trình đệ qui hoàn thành khi không thể tiếp tục thực hiện việc chia tách được nữa, hay khi một phân loại đơn có thể áp dụng cho từng phần tử của tập con dẫn xuất. Một bộ phân loại rừng ngẫu nhiên (*random forest*) sử dụng một số cây quyết định để có thể cải thiện tỉ lệ phân loại.

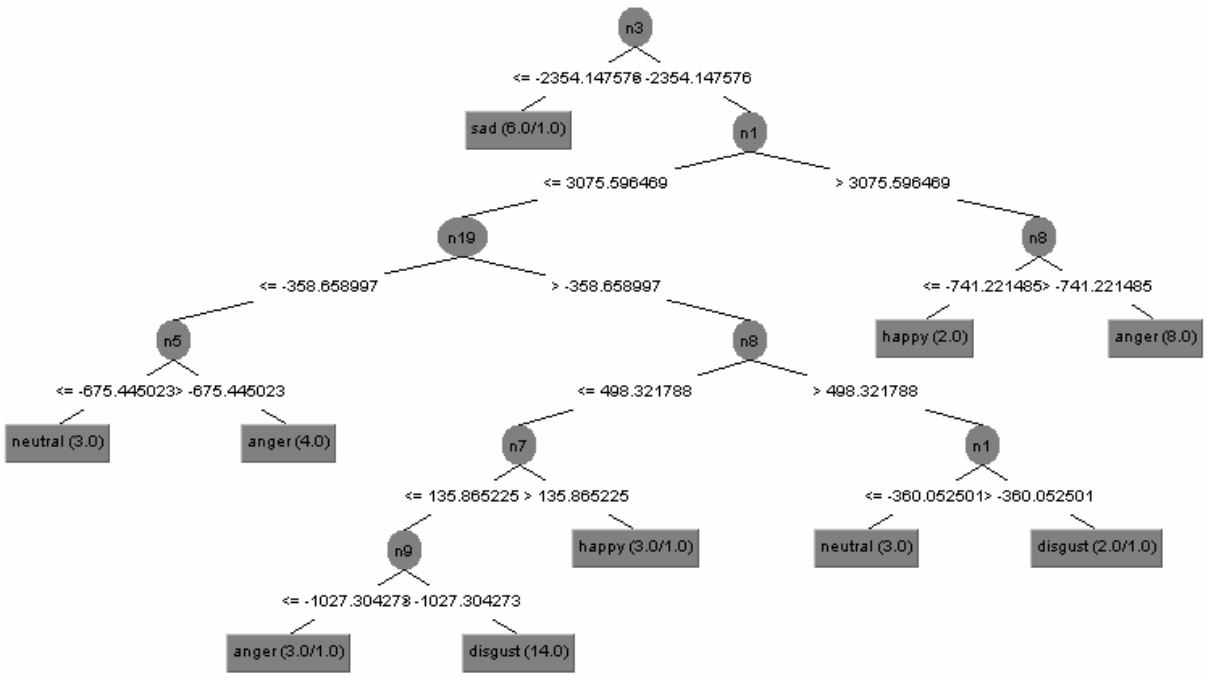
Cây quyết định cũng là một phương tiện có tính mô tả dành cho việc tính toán các xác suất có điều kiện.

Cây quyết định có thể được mô tả như là sự kết hợp của các kỹ thuật toán học và tính toán nhằm hỗ trợ việc mô tả, phân loại và tổng quát hóa một tập dữ liệu cho trước.

Dữ liệu được cho dưới dạng các bản ghi có dạng:

$(x, y) = (x_1, x_2, x_3..., x_k, y)$

Biến phụ thuộc (*dependant variable*) y là biến mà chúng ta cần tìm hiểu, phân loại hay tổng quát hóa. x_1, x_2, x_3 ... là các biến sẽ giúp ta thực hiện công việc đó.



Hình 10: Cây quyết định

Chương 4. THỰC NGHIỆM

4.1 Môi trường thực nghiệm

Chương trình chạy giải thuật PCA được viết bằng ngôn ngữ Matlab, chạy trên nền hệ điều hành Windows 7 Professional, Laptop có tốc độ CPU 2.0 Ghz, bộ nhớ ram 2Gb.

Matlab là 1 phần mềm nổi tiếng của công ty MathWorks, là một ngôn ngữ hiệu năng cao cho tính toán kỹ thuật. Nó tích hợp tính toán, hiển thị và lập trình trong một môi trường dễ sử dụng. Một số ứng dụng tiêu biểu của Matlab như: Hỗ trợ toán học và tính toán, mô phỏng, phân tích, khảo sát và hiển thị số liệu, phát triển ứng dụng với các giao diện đồ họa. Matlab đầu tiên được viết bằng Fortran để cung cấp truy nhập dễ dàng tới phần mềm ma trận được phát triển bởi các dự án Linpack và Eispack. Sau đó nó được viết bằng ngôn ngữ C trên cơ sở các thư viện nêu trên và phát triển thêm nhiều lĩnh vực của tính toán khoa học và các ứng dụng kỹ thuật. Ngoài các tính năng cơ bản, phần mềm MATLAB còn được trang bị thêm các ToolBox – các gói chương trình (thư viện) cho các lĩnh vực ứng dụng rất đa dạng như xử lý tín hiệu, nhận dạng hệ thống, xử lý ảnh, mạng nơ ron, logic mờ, tài chính, tối ưu hóa, phương trình đạo hàm riêng, tin sinh học.

Mạng MultiLayer Perceptron được cung cấp bởi phần mềm Weka.

4.2 Dữ liệu đầu vào

Gồm có 75 ảnh khuôn mặt mẫu, độ phân giải 600 x 800 điểm ảnh, tất cả các ảnh đều là khuôn mặt của một người và có độ sáng đồng đều nhau. Cảm xúc thể hiện trong mỗi ảnh khá rõ ràng. Tập dữ liệu này chỉ có 5 cảm xúc chính là: Vui, buồn, ghê tởm, giận dữ và bình thường.

4.3 Khảo sát và đánh giá

Trong 75 ảnh khuôn mặt mẫu, 40 ảnh bất kỳ được chọn làm dữ liệu huấn luyện và 35 ảnh còn lại làm dữ liệu test.

4.3.1 Phương pháp PCA truyền thống

Với phương pháp này, kết quả nhận dạng được như sau:

- Vui: 80%
- Ghê tởm: 70%
- Giận dữ: 86%
- Buồn: 55%
- Bình thường: 84%
- Trung bình: 65%

Dựa vào kết quả, ta có thể thấy khả năng nhận dạng của phương pháp này không cao. Tuy hiên đây là kết quả với dữ liệu huấn luyện bé. Nếu tập dữ liệu huấn luyện lớn hơn chắc chắn khả năng nhận dạng sẽ tăng.

Đây là một phương pháp đơn giản và dễ hiểu để nhận dạng cảm xúc khuôn mặt tuy nhiên nhược điểm lớn nhất của nó là tốc độ xử lý chậm. Khi tập huấn luyện lớn, bao gồm hàng nghìn ảnh, khi đó với mỗi ảnh cần nhận dạng, ta phải so khớp với lần lượt từng ảnh trong tập huấn luyện. Vì tốc độ chậm nên phương pháp này thường không được ứng dụng nhiều trong thực tế. Bên cạnh đó phương pháp này cũng gặp khó khăn khi ảnh cần nhận dạng không có độ sáng tốt, hoặc khi khuôn mặt không chính diện.

4.3.2 Phương pháp sử dụng mạng nơron

Với phương pháp huấn luyện bằng mạng nơron, chúng ta sẽ sử dụng giải thuật Multilayer Perceptron được cung cấp trong công cụ Weka với các đặc trưng được trích chọn bằng PCA, kết quả nhận dạng được như sau:

- Vui: 100 %
- Ghê tởm: 100%
- Giận dữ: 67%
- Buồn: 50%
- Bình thường: 80%
- Kết quả trung bình: 87%

Khi thay đổi số tầng ẩn lên lớn hơn 5 hoặc bé hơn 5, kết quả trung bình giảm xuống 83,3%. Như vậy khả năng phân loại của mạng nơron không tăng khi số lượng tầng ẩn tăng.

4.3.3 Phương pháp sử dụng cây quyết định

chúng ta sẽ sử dụng giải thuật cây quyết định J48-Decision Tree được cung cấp trong Weka.

Kết quả nhận dạng được

- Vui: 60 %
- Ghê tởm: 14,3%
- Giận dữ: 16,7%
- Buồn: 0%
- Bình thường: 60%
- Kết quả trung bình: 36,7%

Trong giải thuật cây quyết định J48 được cung cấp bởi Weka có 3 tham số quan trọng

- confidenceFactor: Nhân tố sử dụng cho việc cắt tỉa (Nếu giá trị này càng nhỏ thì cây sinh ra sẽ được cắt càng nhiều).
- minNumObj: Số thể hiện tối thiểu trên một nút lá trong cây.
- unPruned: nếu là True thì cây sinh ra sẽ được cắt tỉa và ngược lại.

Sau khi điều chỉnh các tham số, kết quả tốt nhất thu được:

- confidenceFactor: 0.25
- minNumObj: 2
- unPruned: False

Giải thuật cây quyết định J48 cho kết quả nhận dạng rất thấp, nguyên nhân có thể do tập ảnh huấn luyện quá ít (45 ảnh).

4.4 Tổng kết

Chương này mô tả thực nghiệm và kết quả của 3 phương pháp nhận dạng cảm xúc. Phương pháp thứ nhất là dùng PCA và tính khoảng cách Euclid, phương pháp này khả năng nhận dạng trung bình, tốc độ chậm. Phương pháp thứ 2 sử dụng mạng nơron (MLP). Phương pháp này có khả năng nhận dạng tốt, tốc độ nhanh. Phương pháp thứ 3 là phương pháp dùng cây quyết định để phân lớp. Phương pháp này đạt kết quả rất thấp tuy nhiên do tập ảnh huấn luyện quá ít (40 ảnh) nên chưa đánh giá hết khả năng nhận dạng của phương pháp này.

Trong cả 3 phương pháp cảm xúc vui luôn đạt kết quả cao nhất do có số lượng ảnh huấn luyện nhiều nhất. Cảm xúc buồn có số lượng ảnh huấn luyện ít nhất nên đạt kết quả thấp nhất. Những cảm xúc còn lại đều đạt kết quả tương đối.

Cây quyết định là một giải thuật phân lớp nhưng nó chỉ đạt hiệu quả cao khi số lượng lớp là 2. Với số lượng lớn hơn 2 tính hiệu quả của giải thuật giảm đi. Trong khóa luận này vì thời gian chuẩn bị ngắn nên không đi sâu vào phân tích giải thuật này mà chỉ dùng để xem như một phương pháp tham khảo thêm.

Chương 5. KẾT LUẬN

Qua thời gian nghiên cứu về các phương pháp nhận dạng cảm xúc khuôn mặt, đặc biệt là qua quá trình thực hiện khóa luận tốt nghiệp, em đã tìm hiểu được một số thuật toán học và áp dụng các thuật toán này cho bài toán phân lớp để nhận dạng cảm xúc. Những kết quả chính mà khóa luận đã đạt được có thể được tổng kết như sau:

- Giới thiệu chi tiết về phương pháp trích chọn đặc trưng (PCA) và Mạng nơron nhiều tầng truyền thẳng (Multilayer Perceptron), đồng thời giới thiệu sơ lược về 1 giải thuật phân lớp khác là cây quyết định.
- Áp dụng các giải thuật này cho bài toán nhận dạng cảm xúc.
- Nhận xét và đánh giá những kết quả đạt được của các giải thuật trong bài toán nhận dạng cảm xúc.

Bên cạnh những kết quả đã đạt được, còn có những vấn đề mà thời điểm hiện tại khóa luận chưa giải quyết được.

- Xây dựng tập huấn luyện lớn để đạt kết quả chính xác hơn.
- Nghiên cứu về một số các giải thuật trích chọn đặc trưng và phân lớp dữ liệu khác
- Xây dựng một chương trình hoàn chỉnh có giao diện tương tác với người sử dụng

PHỤ LỤC - MỘT SỐ THUẬT NGỮ ANH – VIỆT

Thuật ngữ	Giải nghĩa
Back propagation algorithm	Thuật toán lan truyền ngược sai số
Cross validation	Một cách chọn mẫu trong tập train và tập test để tránh hiện tượng overfitting
Feed forward	Lan truyền xuôi
Input/hidden/output layer	Lớp đầu vào/ẩn/ đầu ra
Mean squared error	Sai số bình phương trung bình
MLP (MultiLayer Perceptrons)	Mạng neuron nhiều tầng truyền thẳng
Transformation/activation function	Hàm truyền/hàm kích hoạt
Unsupervised learning	Học không có giám sát
Validation set	Tập mẫu xác nhận mạng

TÀI LIỆU THAM KHẢO

1. G.Zhao, M.Pietikäinen. Dynamic texture recognition using local binary patterns with an application to facial expressions. IEEE Transactions on Pattern Analysis and Machine Intelligence. 2007.

2. Y.L.Tian, T.Kanade, J.Cohn. Recognizing action units for facial expression analysis. IEEE Transactions on Pattern Analysis and Machine Intelligence. 2001.

3. Z.Wen, T. Huang. Capturing Subtle Facial Motions in 3D Face Tracking. International Conference on Computer Vision. 2003.

4. Y.Zhang, Q.Ji. Active and dynamic information fusion for facial expression understanding from image sequence. IEEE Transactions on Pattern Analysis and Machine Intelligence. 2005.

5. M.S.Bartlett, J.C.Hager, P.Ekman, T.J.Sejnowski. Measuring facial expressions by computer image analysis. Psychophysiology. 1999.

6. Z.Zhang, M.Lyons, M.Schuster, S.Akamatsu. Comparison Between Geometry-Based and Gabor-Wavelets-Based Facial Expression Recognition Using Multi-Layer Perceptron. IEEE International Conference on Automatic Face and Gesture Recognition. 1998.

7. M.Pantic, I.Patras. Dynamics of facial expression: Recognition of facial actions and their temporal segments from face profile image sequences. IEEE Transactions on Systems, Man and Cybernetics. 2006.

8. E.Holden, R.Owens. Automatic Facial Point Detection, Asian Conference on Computer Vision. 2002.

9. D.Vukadinovic, M.Pantic. Fully Automatic Facial Feature Point Detection Using Gabor Feature Based Boosted Classifiers. IEEE International Conference on Systems, Man and Cybernetics. 2005.

10. L.Chen, L.Zhang, H.Zhang, M.Abdel-Mottaleb. 3D Shape Constraint for Facial Feature Localization using Probabilistic-like Output. IEEE International Workshop Analysis and Modeling of Faces and Gestures. 2004.