TRƯỜNG ĐẠI HỌC BÁCH KHOA HÀ NỘI

VIỆN TOÁN ỨNG DỤNG VÀ TIN HỌC

----- oOo -----

ĐỒ ÁN II

ĐỀ TÀI : ỨNG DỤNG MẠNG NEURAL TÍCH CHẬP TRÊN HỆ THỐNG NHÚNG RASPBERRY PI

Giảng viên hướng dẫn: Thầy Lê Chí Ngọc

Sinh viên thực hiện: Nguyễn Thị Hằng

MSSV: 20161384

Lớp: Toán Tin - K61

Hà Nội, ngày 29 tháng 09 năm 2019

NHẬN XÉT CỦA GIẢNG VIÊN HƯỚNG DẪN

1. Mục đích và nội dung của đồ án:

……

……

……

……

2. Kết quả đạt được:

……

……

……

……

3. Ý thức làm việc của sinh viên:

……

……

……

……

Hà Nội, ngày tháng năm

Giảng viên hướng dẫn

(Ký và ghi rõ họ tên)

Lời Nói Đầu

Nhận dạng, phân lớp là bài toán xuất hiện cách đây khá lâu và luôn thu hút được khá nhiều sự quan tâm của các nhà nghiên cứu. Đặc biệt trong thập niên gần đây, khi cách mạng 4.0 bùng nổ trên toàn cầu thì nhận dạng, phân lớp không chỉ được quan tâm tìm hiểu mà nó còn có ứng dụng rộng rãi phổ biến trong rất nhiều lĩnh vực thực tiễn. Các ứng dụng có ý nghĩa thực tiễn lớn như : phân loại biển số xe, phát hiện bệnh trong y học …

Những năm gần đây Deep Learning phát triển cực mạnh dựa trên lượng dữ liệu huấn luyện khổng lồ và khả năng tính toán ngày càng được cải tiến của các máy tính. Các kết quả cho bài toán phân loại, nhận diện giọng nói, xử lý ngôn ngữ tự nhiên… ngày càng được nâng cao. Convolution Neural Networks (CNNs-mạng neural tích chập) là một trong những mô hình Deep Learning tiên tiến giúp cho chúng ta xây dựng được những hệ thống thông minh có độ chính xác cao như hiện nay.

Chính vì sự phát triển nở rộ của Deep Learning trong khoa học kỹ thuật nên em đã lựa chọn đề tài ứng dụng của mạng neural tích chập trên bài toán phân lớp ảnh làm đồ án II với mong muốn phần nào áp dụng bài toán vào thực tế. Đề tài đặt ra một số yêu cầu cần giải bài toán như :

- Xử lý ảnh đầu vào
- Trích chọn đặc trưng của ảnh
- Tiến hành phân lớp sử dụng mạng neural tích chập

Với những yêu cầu đặt ra ở trên cấu trúc đồ án sẽ bao gồm :

- **CHƯƠNG 1 : TỔNG QUAN VỀ HỆ THỐNG NHÚNG VÀ HỆ THỐNG NHÚNG RASPBERRY PI VÀ BÀI TOÁN PHÂN LỚP ẢNH (IMAGE CLASSIFICATION)**

  Giới thiệu tổng quan về hệ thống nhúng raspberry pi, cấu trúc phần cứng và phần mềm của một hệ thống raspberry pi.

  Giới thiệu tổng quan về bài toán phân lớp ảnh, mô hình tổng quát của một hệ phân lớp ảnh, kỹ thuật deep learning áp dụng vào bài toán và phạm vi đề tài.

- **CHƯƠNG 2 : CƠ SỞ LÝ THUYẾT MẠNG NEURAL TÍCH CHẬP (CONVOLUTION NEURAL NETWORKS) VÀ ỨNG DỤNG TRONG BÀI TOÁN PHÂN LỚP ẢNH**

  Cơ sở lý thuyết về mạng neural và mạng neural tích chập.

  Giới hiệu mô hình hóa bài toán, thiết kế dữ liệu luyện mạng, phương pháp huấn luyện mạng, xây dựng chương trình.

- **CHƯƠNG 3 CÀI ĐẶT CHƯƠNG TRÌNH**

  Trình bày môi trường cài đặt, các yêu cầu liên quan, đánh giá kết quả, ưu điểm, khuyết điểm và đưa ra hướng phát triển tương lai.

Em xin chân thành cảm ơn thầy Lê Chí Ngọc đã tận tình dạy dỗ và truyền đạt cho em những kiến thức quý giá trong suốt quá trình thực hiện đồ án.

Trong quá trình hoàn thiện đồ án do hạn chế về mặt thời gian cũng như kiến thức nên đồ án II không tránh khỏi những thiếu sót. Vì vậy em rất mong nhận được sự đóng góp ý kiến của thầy hướng dẫn, các thầy cô phản biện đồ án để em có thể đạt kết quả tốt hơn trong tương lai.

Hà nội, tháng 09 năm 2019

Người thực hiện đồ án

Hằng

Nguyễn Thị Hằng

Mục Lục

CHƯƠNG 1 TỔNG QUAN VỀ HỆ THỐNG NHÚNG VÀ HỆ THỐNG NHÚNG RASPBERRY PI VÀ BÀI TOÁN PHÂN LỚP ẢNH (IMAGE CLASSIFICATION)

1.1 Giới thiệu tổng quan các đặc điểm của hệ thống nhúng

Hệ thống nhúng (*embedded system*) là một thuật ngữ để chỉ một hệ thống có khả năng hoạt động độc lập, được nhúng vào trong một môi trường hay một hệ thống mẹ. Đó là các hệ thống tích hợp cả phần cứng và phần mềm phục vụ các bài toán chuyên dụng trong nhiều lĩnh vực công nghiệp, tự động hoá điều khiển, quan trắc và truyền tin. Đặc điểm của các hệ thống nhúng là hoạt động ổn định và có tính năng tự động hoá cao.

Hệ thống nhúng thường được thiết kế để thực hiện một chức năng chuyên biệt nào đó. Khác với các máy tính đa chức năng, chẳng hạn như máy tính cá nhân, một hệ thống nhúng chỉ thực hiện một hoặc một vài chức năng nhất định, thường đi kèm với những yêu cầu cụ thể và bao gồm một số thiết bị máy móc và phần cứng chuyên dụng mà ta không tìm thấy trong một máy tính đa năng nói chung. Vì hệ thống chỉ được xây dựng cho một số nhiệm vụ nhất định nên các nhà thiết kế có thể tối ưu hóa nó nhằm giảm thiểu kích thước và chi phí sản xuất. Các hệ thống nhúng thường được sản xuất hàng loạt với số lượng lớn. Hệ thống nhúng rất đa dạng, phong phú về chủng loại. Đó có thể là những thiết bị cầm tay nhỏ gọn như đồng hồ kĩ thuật số và máy chơi nhạc MP3, hoặc những sản phẩm lớn như đèn giao thông, bộ kiểm soát trong nhà máy hoặc hệ thống kiểm soát các máy năng lượng hạt nhân. Xét về độ phức tạp, hệ thống nhúng có thể rất đơn giản với một vi điều khiển hoặc rất phức tạp với nhiều đơn vị, các thiết bị ngoại vi và mạng lưới được nằm gọn trong một lớp vỏ máy lớn.

Các thiết bị PDA hoặc máy tính cầm tay cũng có một số đặc điểm tương tự với hệ thống nhúng như các hệ điều hành hoặc vi xử lý điều khiển chúng nhưng các thiết bị này không phải là hệ thống nhúng thật sự bởi chúng là các

thiết bị đa năng, cho phép sử dụng nhiều ứng dụng và kết nối đến nhiều thiết bị ngoại vi. [1]

1.1.1 Các đặc điểm

Hệ thống nhúng thường có một số đặc điểm chung như sau:

Các hệ thống nhúng được thiết kế để thực hiện một số nhiệm vụ chuyên dụng chứ không phải đóng vai trò là các hệ thống máy tính đa chức năng. Một số hệ thống đòi hỏi ràng buộc về tính hoạt động thời gian thực để đảm bảo độ an toàn và tính ứng dụng; một số hệ thống không đòi hỏi hoặc ràng buộc chặt chẽ, cho phép đơn giản hóa hệ thống phần cứng để giảm thiểu chi phí sản xuất.

Một hệ thống nhúng thường không phải là một khối riêng biệt mà là một hệ thống phức tạp nằm trong thiết bị mà nó điều khiển.

Phần mềm được viết cho các hệ thống nhúng được gọi là firmware và được lưu trữ trong các chip bộ nhớ ROM hoặc bộ nhớ flash chứ không phải là trong một ổ đĩa. Phần mềm thường chạy với số tài nguyên phần cứng hạn chế: không có bàn phím, màn hình hoặc có nhưng với kích thước nhỏ, dung lượng bộ nhớ thấp Sau đây, ta sẽ đi sâu, xem xét cụ thể đặc điểm của các thành phần của hệ thống nhúng. [2]

Giao diện

Các hệ thống nhúng có thể không có giao diện (đối với những hệ thống đơn nhiệm) hoặc có đầy đủ giao diện giao tiếp với người dùng tương tự như các hệ điều hành trong các thiết bị để bàn. Đối với các hệ thống đơn giản, thiết bị nhúng sử dụng nút bấm, đèn LED và hiển thị chữ cỡ nhỏ hoặc chỉ hiển thị số, thường đi kèm với một hệ thống menu đơn giản.

Còn trong một hệ thống phức tạp hơn, một màn hình đồ họa, cảm ứng hoặc có các nút bấm ở lề màn hình cho phép thực hiện các thao tác phức tạp

mà tối thiểu hóa được khoảng không gian cần sử dụng; ý nghĩa của các nút bấm có thể thay đổi theo màn hình và các lựa chọn. Các hệ thống nhúng thường có một màn hình với một nút bấm dạng cần điều khiển (joystick button) [3]. Sự phát triển mạnh mẽ của mạng toàn cầu đã mang đến cho những nhà thiết kế hệ nhúng một lựa chọn mới là sử dụng một giao diện web thông qua việc kết nối mạng. Điều này có thể giúp tránh được chi phí cho những màn hình phức tạp nhưng đồng thời vẫn cung cấp khả năng hiển thị và nhập liệu phức tạp khi cần đến, thông qua một máy tính khác. Điều này là hết sức hữu dụng đối với các thiết bị điều khiển từ xa, cài đặt vĩnh viễn. Ví dụ, các router là các thiết bị đã ứng dụng tiện ích này.

Kiến trúc CPU

Các bộ xử lý trong hệ thống nhúng có thể được chia thành hai loại: vi xử lý và vi điều khiển. Các vi điều khiển thường có các thiết bị ngoại vi được tích hợp trên chip nhằm giảm kích thước của hệ thống. Có rất nhiều loại kiến trúc CPU được sử dụng trong thiết kế hệ nhúng như ARM, MIPS, Coldfire/68k, PowerPC, x86, PIC, 8051, Atmel AVR, Renesas H8, SH, V850, FR-V, M32R, Z80, Z8 … Điều này trái ngược với các loại máy tính để bàn, thường bị hạn chế với một vài kiến trúc máy tính nhất định. Các hệ thống nhúng có kích thước nhỏ và được thiết kế để hoạt động trong môi trường công nghiệp thường lựa chọn PC/104 và PC/104++ làm nền tảng. Những hệ thống này thường sử dụng DOS, Linux, NetBSD hoặc các hệ điều hành nhúng thời gian thực như QNX hay VxWorks. Còn các hệ thống nhúng có kích thước rất lớn thường sử dụng một cấu hình thông dụng là hệ thống on chip (System on a chip – SoC), một bảng mạch tích hợp cho một ứng dụng cụ thể (an application-specific integrated circuit – ASIC). Sau đó nhân CPU được mua và thêm vào như một phần của thiết kế chip. Một chiến lược tương tự là sử dụng FPGA (field-programmable

gate array) và lập trình cho nó với những thành phần nguyên lý thiết kế bao gồm cả CPU.

Thiết bị ngoại vi

Hệ thống nhúng giao tiếp với bên ngoài thông qua các thiết bị ngoại vi, ví dụ như:

- ✓ Serial Communication Interfaces (SCI): RS-232, RS-422, RS-485...
- ✓ Synchronous Serial Communication Interface: I2C, JTAG, SPI, SSC và ESSI
- ✓ Universal Serial Bus (USB)
- ✓ Networks: Controller Area Network, LonWorks...
- ✓ Bộ định thời: PLL(s), Capture/Compare và Time Processing Units
- ✓ Discrete IO: General Purpose Input/Output (GPIO)

Công cụ phát triển

Tương tự như các sản phẩm phần mềm khác, phần mềm hệ thống nhúng cũng được phát triển nhờ việc sử dụng các trình biên dịch (compilers), chương trình dịch hợp ngữ (assembler) hoặc các công cụ gỡ rối (debuggers) [4]. Tuy nhiên, các nhà thiết kế hệ thống nhúng có thể sử dụng một số công cụ chuyên dụng như :

- ✓ Bộ gỡ rối mạch hoặc các chương trình mô phỏng (emulator)
- ✓ Tiện ích để thêm các giá trị checksum hoặc CRC vào chương trình, giúp hệ thống nhúng có thể kiểm tra tính hợp lệ của chương trình đó.
- ✓ Đối với các hệ thống xử lý tín hiệu số, người phát triển hệ thống có thể sử dụng phần mềm workbench như MathCad hoặc Mathematica để mô phỏng các phép toán.

- ✓ Các trình biên dịch và trình liên kết (linker) chuyên dụng được sử dụng để tối ưu hóa một thiết bị phần cứng.
- ✓ Một hệ thống nhúng có thể có ngôn ngữ lập trình và công cụ thiết kế riêng của nó hoặc sử dụng và cải tiến từ một ngôn ngữ đã có sẵn.

Các công cụ phần mềm có thể được tạo ra bởi các công ty phần mềm chuyên dụng về hệ thống nhúng hoặc chuyển đổi từ các công cụ phát triển phần mềm GNU. Đôi khi, các công cụ phát triển dành cho máy tính cá nhân cũng được sử dụng nếu bộ xử lý của hệ thống nhúng đó gần giống với bộ xử lý của một máy PC thông dụng.

Độ tin cậy

Các hệ thống nhúng thường nằm trong các cỗ máy được kỳ vọng là sẽ chạy hàng năm trời liên tục mà không bị lỗi hoặc có thể khôi phục hệ thống khi gặp lỗi. Vì thế, các phần mềm hệ thống nhúng được phát triển và kiểm thử một cách cẩn thận hơn là phần mềm cho máy tính cá nhân. Ngoài ra, các thiết bị rời không đáng tin cậy như ổ đĩa, công tắc hoặc nút bấm thường bị hạn chế sử dụng. Việc khôi phục hệ thống khi gặp lỗi có thể được thực hiện bằng cách sử dụng các kỹ thuật như watchdog timer – nếu phần mềm không đều đặn nhận được các tín hiệu watchdog định kì thì hệ thống sẽ bị khởi động lại.

Một số vấn đề cụ thể về độ tin cậy như:

- ✓ Hệ thống không thể ngừng để sửa chữa một cách an toàn, ví dụ như ở các hệ thống không gian, hệ thống dây cáp dưới đáy biển, các đèn hiệu dẫn đường… Giải pháp đưa ra là chuyển sang sử dụng các hệ thống con dự trữ hoặc các phần mềm cung cấp một phần chức năng.

- ✓ Hệ thống phải được chạy liên tục vì tính an toàn, ví dụ như các thiết bị dẫn đường máy bay, thiết bị kiểm soát độ an toàn trong các nhà máy hóa chất… Giải pháp đưa ra là lựa chọn backup hệ thống.
- ✓ Nếu hệ thống ngừng hoạt động sẽ gây tổn thất rất nhiều tiền của ví dụ như các dịch vụ buôn bán tự động, hệ thống chuyển tiền, hệ thống kiểm soát trong các nhà máy …

1.1.2 Các kiến trúc phần mềm trên hệ thống nhúng

Một số loại kiến trúc phần mềm thông dụng trong các hệ thống nhúng như sau:

Vòng lặp kiểm soát đơn giản

Theo thiết kế này, phần mềm được tổ chức thành một vòng lặp đơn giản. Vòng lặp gọi đến các chương trình con, mỗi chương trình con quản lý một phần của hệ thống phần cứng hoặc phần mềm.

Hệ thống ngắt điều khiển

Các hệ thống nhúng thường được điều khiển bằng các ngắt. Có nghĩa là các tác vụ của hệ thống nhúng được kích hoạt bởi các loại sự kiện khác nhau. Ví dụ, một ngắt có thể được sinh ra bởi một bộ định thời sau một chu kỳ được định nghĩa trước, hoặc bởi sự kiện khi cổng nối tiếp nhận được một byte nào đó.

Loại kiến trúc này thường được sử dụng trong các hệ thống có bộ quản lý sự kiện đơn giản, ngắn gọn và cần độ trễ thấp. Hệ thống này thường thực hiện một tác vụ đơn giản trong một vòng lặp chính. Đôi khi, các tác vụ phức tạp hơn sẽ được thêm vào một cấu trúc hàng đợi trong bộ quản lý ngắt để được vòng lặp xử lý sau đó. Lúc này, hệ thống gần giống với kiểu nhân đa nhiệm với các tiến trình rời rạc.

Đa nhiệm tương tác

Một hệ thống đa nhiệm không ưu tiên cũng gần giống với kỹ thuật vòng lặp kiểm soát đơn giản ngoại trừ việc vòng lặp này được ẩn giấu thông qua một giao diện lập trình API. Các nhà lập trình định nghĩa một loạt các nhiệm vụ, mỗi nhiệm vụ chạy trong một môi trường riêng của nó. Khi không cần thực hiện nhiệm vụ đó thì nó gọi đến các tiến trình con tạm nghỉ (bằng cách gọi "pause", "wait", "yield" …). [2]

Ưu điểm và nhược điểm của loại kiến trúc này cũng giống với kiểm vòng lặp kiểm soát đơn giản. Tuy nhiên, việc thêm một phần mềm mới được thực hiện dễ dàng hơn bằng cách lập trình một tác vụ mới hoặc thêm vào hàng đợi thông dịch (queue-interpreter).

Đa nhiệm ưu tiên

Ở loại kiến trúc này, hệ thống thường có một đoạn mã ở mức thấp thực hiện việc chuyển đổi giữa các tác vụ khác nhau thông qua một bộ định thời. Đoạn mã này thường nằm ở mức mà hệ thống được coi là có một hệ điều hành và vì thế cũng gặp phải tất cả những phức tạp trong việc quản lý đa nhiệm.

Bất kỳ tác vụ nào có thể phá hủy dữ liệu của một tác vụ khác đều cần phải được tách biệt một cách chính xác. Việc truy cập tới các dữ liệu chia sẻ có thể được quản lý bằng một số kỹ thuật đồng bộ hóa như hàng đợi thông điệp (message queues), các phương thức truyền tin thị giác (semaphores)… Bởi những phức tạp nói trên, giải pháp thường được đưa ra đó là sử dụng một hệ điều hành thời gian thực. Lúc đó, các nhà lập trình có thể tập trung vào việc phát triển các chức năng của thiết bị chứ không cần quan tâm đến các dịch vụ của hệ điều hành nữa.

Vi nhân (Microkernel) và nhân ngoại (Exokernel)

Khái niệm vi nhân (microkernel) là một bước tiếp cận gần hơn tới khái niệm hệ điều hành thời gian thực. Lúc này, nhân hệ điều hành thực hiện việc cấp phát bộ nhớ và chuyển CPU cho các luồng thực thi. Còn các tiến trình người dùng sử dụng các chức năng chính như hệ thống file, giao diện mạng lưới,… Nói chung, kiến trúc này thường được áp dụng trong các hệ thống mà việc chuyển đổi và giao tiếp giữa các tác vụ là nhanh.

Còn nhân ngoại (exokernel) tiến hành giao tiếp hiệu quả bằng cách sử dụng các lời gọi chương trình con thông thường. Phần cứng và toàn bộ phần mềm trong hệ thống luôn đáp ứng và có thể được mở rộng bởi các ứng dụng.

Nhân khối (monolithic kernels)

Trong kiến trúc này, một nhân đầy đủ với các khả năng phức tạp được chuyển đổi để phù hợp với môi trường nhúng. Điều này giúp các nhà lập trình có được một môi trường giống với hệ điều hành trong các máy để bàn như Linux hay Microsoft Windows và vì thế rất thuận lợi cho việc phát triển. Tuy nhiên, nó lại đòi hỏi đáng kể các tài nguyên phần cứng làm tăng chi phí của hệ thống. Một số loại nhân khối thông dụng là Embedded Linux và Windows CE. Mặc dù chi phí phần cứng tăng lên nhưng loại hệ thống nhúng này đang tăng trưởng rất mạnh, đặc biệt là trong các thiết bị nhúng mạnh như Wireless router hoặc hệ thống định vị GPS. Lý do của điều này là:

- ✓ Hệ thống này có cổng để kết nối đến các chip nhúng thông dụng
- ✓ Hệ thống cho phép sử dụng lại các đoạn mã sẵn có phổ biến như các trình điều khiển thiết bị, Web Servers, Firewalls, …
- ✓ Việc phát triển hệ thống có thể được tiến hành với một tập nhiều loại đặc tính, chức năng còn sau đó lúc phân phối sản phẩm, hệ thống có

thể được cấu hình để loại bỏ một số chức năng không cần thiết. Điều này giúp tiết kiệm được những vùng nhớ mà các chức năng đó chiếm giữ.

- ✓ Hệ thống có chế độ người dùng để dễ dàng chạy các ứng dụng và gỡ rối. Nhờ đó, quy trình phát triển được thực hiện dễ dàng hơn và việc lập trình có tính linh động hơn.
- ✓ Có nhiều hệ thống nhúng thiếu các yêu cầu chặt chẽ về tính thời gian thực của hệ thống quản lý. Còn một hệ thống như Embedded Linux có tốc độ đủ nhanh để trả lời cho nhiều ứng dụng. Các chức năng cần đến sự phản ứng nhanh cũng có thể được đặt vào phần cứng.

1.2 Giới thiệu về hệ thống nhúng raspberry PI

Raspberry Pi là từ để chỉ các máy tính chỉ có một board mạch (hay còn gọi là máy tính nhúng) kích thước chỉ bằng một thẻ tín dụng, được phát triển tại Anh bởi Raspberry Pi Foundation với mục đích ban đầu là thúc đẩy việc giảng dạy về khoa học máy tính cơ bản trong các trường học và các nước đang phát triển. [5] [4]

Raspberry Pi gốc và Raspberry Pi 2 được sản xuất theo nhiều cấu hình khác nhau thông qua các thỏa thuận cấp phép sản xuất với Newark element14 (Premier Farnell), RS Components và Egoman. Các công ty này bán Raspberry Pi trực tuyến. Egoman sản xuất một phiên bản phân phối duy nhất tại Đài Loan, có thể được phân biệt với Pis khác bởi màu đỏ của chúng và thiếu dấu FCC/CE. Phần cứng là như nhau đối với tất cả các nhà sản xuất.

Raspberry Pi ban đầu được dựa trên hệ thống trên một vi mạch (SoC) BCM2835 của Broadcom, bao gồm một vi xử lý ARM1176JZF-S 700 MHz, VideoCore IV GPU, và ban đầu được xuất xưởng với 256 MB RAM, sau đó được nâng cấp (model B và B +) lên đến 512 MB. Board này cũng có

socket Secure Digital (SD) (model A và B) hoặc MicroSD (model A + và B +) dùng làm thiết bị khởi động và bộ lưu trữ liên tục. [6]

Trong năm 2014, Raspberry Pi Foundation đã phát hành *Mô-đun Compute*, đóng gói một BCM2835 với 512 MB RAM và một flash chip eMMC vào một module để sử dụng như một phần của hệ thống nhúng.

Tổ chức này cung cấp Debian và Arch Linux ARM để người dùng tải về. Các công cụ có sẵn cho Python như là ngôn ngữ lập trình chính, hỗ trợ cho BBC BASIC (thông qua RISC OS image hoặc Brandy Basic clone cho Linux), C, C++, Java, Perl và Ruby.

Tính đến ngày 08/06/2015, khoảng 5-6.000.000 mạch Raspberry Pi đã được bán. Khi đã trở thành máy tính cá nhân bán chạy nhanh nhất của Anh, Raspberry Pi là thiết bị được giao nhiều thứ hai sau Amstrad PCW, "Personal Computer Word-processor", bán được 8 triệu chiếc

Vào đầu tháng 2 năm 2015, thế hệ tiếp theo của Raspberry Pi, Raspberry Pi 2, đã được phát hành. Board máy tính mới này đầu tiên chỉ có một cấu hình (model B) và trang bị SoC Broadcom BCM2836, với một nhân ARM Cortex-A7 CPU 4 lõi và một VideoCore IV dual-core GPU; 1 GB bộ nhớ RAM với thông số kỹ thuật còn lại tương tự như của các thế hệ model B+ trước đó. Raspberry Pi 2 vẫn giữ nguyên giá $35 so với model B, với model A+ giá $20 vẫn còn được bán.

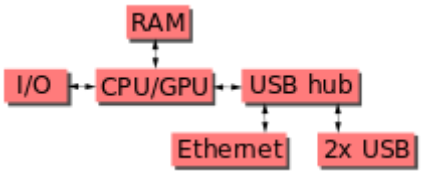
Hầu hết các mạch Pi được sản xuất tại một nhà máy Sony tại Pencoed, Wales; một số được sản xuất tại Trung Quốc hoặc Nhật Bản. [7]

1.2.1 Phần cứng

Phần cứng Raspberry Pi qua nhiều phiên bản được trang bị cấu hình khác nhau, về dung lượng bộ nhớ, vi xử lí, thiết bị ngoại vi …

Hình 1 0.1 Raspberry pi 3

Sơ đồ khối trên mô tả cấu tạo của các model A, B, A+, B+. Model A và A+ không có cổng Ethernet và USB. Bộ chuyển đổi Ethernet phải kết nối qua một cổng USB. Trong model A và A+, cổng USB kết nối trực tiếp đến SoC. Trên model B+, chip này có chứa một hub USB với 5 đầu ra, trong đó có 4 cổng, model B chỉ cung cấp 2 cổng.



Hình 1 0.2 Sơ đồ cấu tạo model trên raspberry pi

Bộ vi xử lý

SoC được sử dụng trong Raspberry Pi thế hệ đầu tiên khá giống với chip được sử dụng trong những chiếc điện thoại thông minh đời cũ (như iPhone 3G / 3GS). Raspberry Pi dựa trên SoC BCM2835 của Broadcom, trong đó gồm một bộ xử lý ARM1176JZF-S 700 MHz, GPU VideoCore IV và RAM. Nó có bộ nhớ cache cấp 1 16 KB và bộ nhớ cache cấp 2 128 KB. Cache cấp 2 này được sử dụng chủ yếu bởi GPU. SoC được xếp chồng lên nhau dưới chip RAM, vì vậy ta chỉ thấy được cạnh của nó. [2]

RAM

Trên các bảng mạch model B beta cũ hơn, 128 MB đã được phân bổ theo mặc định cho GPU, để lại 128 MB cho CPU.

Trên mô hình phát hành 256 MB đầu tiên (và mô hình A), có thể có ba phân chia khác nhau. Phân chia mặc định là 192 MB (RAM cho CPU), đủ để giải mã video 1080p độc lập hoặc cho 3D đơn giản, nhưng có lẽ không Đối với cả hai cùng nhau, 224 MB chỉ dành cho Linux, chỉ với bộ đệm khung hình 1080p và có khả năng thất bại đối với bất kỳ video hoặc 3D nào. 128 MB dành cho 3D nặng, cũng có thể là giải mã video (ví dụ XBMC). 128 MB cho Broadcom VideoCore IV. Đối với model B mới có RAM 512 MB ban đầu, có các tệp phân chia bộ nhớ tiêu chuẩn mới được phát hành (arm256_start.elf, arm384_start.elf, arm496_start.elf) cho RAM 256 MB, 384 MB và 496 MB (và RAM video 256 MB, 128 MB và 16 MB). Nhưng một tuần sau đó, RPF đã phát hành phiên bản mới của start.elf có thể đọc một mục mới trong config.txt (gpu_mem = xx) và có thể tự động gán một Lượng RAM (từ 16 đến 256 MB trong 8 bước) cho GPU, do đó phương pháp phân tách bộ nhớ cũ hơn đã trở nên lỗi thời và một start.elf hoạt động tương tự cho Raspberry Pi 256 và 512 MB [4].

Mạng

Mặc dù model A và A+ không có một cổng 8P8C ("RJ45") Ethernet, chúng có thể được kết nối với một mạng sử dụng một bộ adapter USB Ethernet hoặc Wi-Fi ngoại vi do người dùng cung cấp. Trên model B và B+ cổng Ethernet được cung cấp bởi một adapter USB Ethernet có sẵn.

Thiết bị ngoại vi

Raspberry Pi có thể hoạt động với bất kỳ bàn phím máy tính và chuột thông qua kết nối với các cổng USB.

Video

Bộ điều khiển video có khả năng phân giải chuẩn truyền hình hiện đại, chẳng hạn như HD và Full HD, và các độ phân giải màn hình và cao hơn hoặc thấp hơn và độ phân giải TV CRT chuẩn cũ hơn. Khi vận chuyển (tức là không có tùy chỉnh ép xung) nó có các khả năng như sau: 640×350EGA ; 640×480VGA …

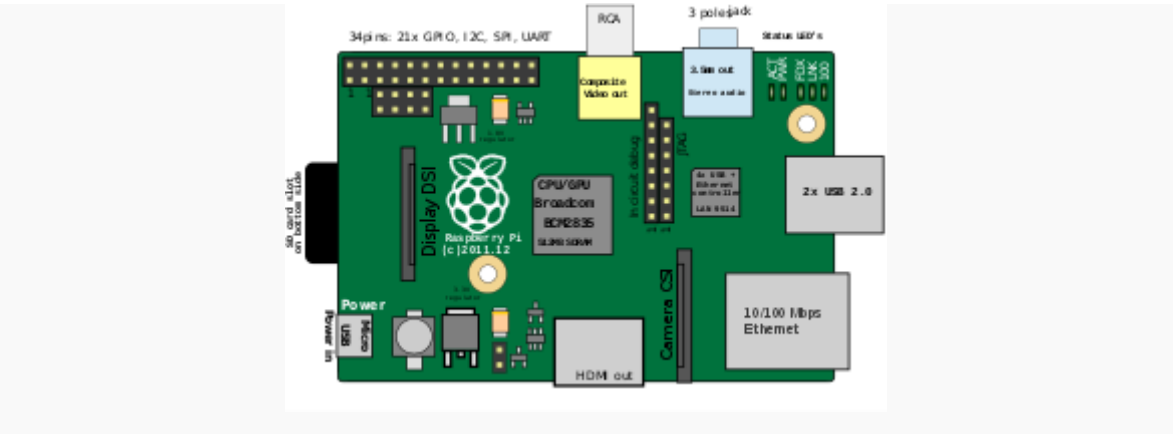
Đồng hồ thời gian thực

Raspberry Pi không được trang bị đồng hồ thời gian thực, có nghĩa là nó không thể theo dõi thời gian trong ngày, khi nó không hoạt động.

Thay vào đó, một chương trình chạy trên Pi có thể lấy thời gian từ một máy chủ thời gian mạng hoặc do người dùng nhập vào lúc khởi động.

Một đồng hồ thời gian thực (như DS1307) với pin dự phòng có thể được thêm vào (thường thông qua giao tiếp I^2C).

Các cổng kết nối



Hình 1 0.3 Sơ đồ cổng kết nối raspberry pi 2

Vị trí của các kết nối và các IC chính trên Raspberry Pi1 model B phiên bản 2

Cổng GPIO

Raspberry Pi A+, B+ và 2B GPIO J8 có 40-chân pinout.[21] Model A và B chỉ có 26 chân.

Model B rev 2 cũng có một pad (gọi là P5 trên board mạch và P6 trên sơ đồ) của 8 chân cung cấp truy cập đến một kết nối 4 GPIO bổ sung.

Model A và B quy định GPIO truy cập vào LED trạng thái ACT sử dụng GPIO 16. Model A+ và B+ và quy định GPIO truy cập vào các LED trạng thái ACT sử dụng GPIO 47, và LED trạng thái nguồn sử dụng GPIO 35.

Phụ kiện

Camera - Ngày 14 tháng 5 năm 2013, the Foundation và các nhà phân phối RS Components & Premier Farnell / Element 14 đã ra mắt board camera Raspberry Pi với một bản cập nhật firmware kèm theo. Board camera được vận chuyển đi kèm với một cáp phẳng linh hoạt để cắm vào đầu nối CSI nằm giữa cổng Ethernet và HDMI. Trong Raspbian, ta kích hoạt hệ thống sử dụng board

camera này bằng cách cài đặt hoặc nâng cấp lên phiên bản mới nhất của hệ điều hành (OS) và sau đó chạy Raspi-config và chọn tùy chọn camera. Giá của mô-đun camera này là €20 ở châu Âu (09 Tháng 9 năm 2013)

Gertboard – một thiết bị được Raspberry Pi Foundation khuyến khích, được thiết kế cho mục đích giáo dục, nó sẽ giúp mở rộng các chân GPIO của Raspberry Pi để cho phép giao tiếp với và điều khiển các đèn LED, tiếp điểm, tín hiệu analog, cảm biến và các thiết bị khác. Nó cũng bao gồm một trình điều khiển tương thích với Arduino theo tùy chọn để giao tiếp với Pi.

Infrared Camera – Vào tháng 10 năm 2013, tổ chức này đã tuyên bố rằng họ sẽ bắt đầu sản xuất một module camera không có bộ lọc hồng ngoại, được gọi là Pi NoIR.

Các board mở rộng HAT (Hardware Attached on Top-Phần cứng đính kèm ở mặt trên) – Cùng với model B+, được lấy cảm hứng bởi các board Arduino shield, giao diện cho các board HAT được trang bị bởi Raspberry Pi Foundation. Mỗi board HAT mang theo một EEPROM nhỏ (điển hình là một CAT24C32WI-GT3) chứa các chi tiết có liên quan tới board này, do đó hệ điều hành của Raspberry Pi được sẽ được thông báo về HAT, và chi tiết kỹ thuật của nó, liên quan tới hệ điều hành sử dụng HAT. Board HAT có 4 lỗ định vị ở 4 góc hình chữ nhật của nó.

1.2.2 Phần mềm

Các hệ điều hành Raspberry Pi chủ yếu sử dụng các hệ điều hành dựa trên nhân Linux.

Chip ARM11 tại trung tâm của Pi (mô hình thế hệ đầu tiên) được dựa trên phiên bản 6 của ARM. Các phiên bản hiện tại của một số phân nhánh phổ biến của Linux, bao gồm Ubuntu, sẽ không chạy trên ARM11. Không thể chạy Windows trên Raspberry Pi gốc, mặc dù Raspberry Pi 2 mới có thể chạy trên hệ điều hành Windows 10 IoT Core. Raspberry Pi 2 hiện tại chỉ hỗ trợ Ubuntu Snappy Core, Raspbian, OpenELEC và RISC OS.

Trình quản lý cài đặt cho Raspberry Pi là NOOBS. Các hệ điều hành đi kèm với NOOOBS là:

- ✓ Arch Linux ARM
- ✓ OpenELEC
- ✓ Pidora (biến thể của Fedora)
- ✓ Puppy Linux
- ✓ Raspbmc và trung tâm truyền thông số mã nguồn mở XBMC[37]
- ✓ RISC OS – là hệ điều hành của máy tính dựa trên nền tảng ARM đầu tiên.
- ✓ Raspbian (được đề xuất dùng cho Raspberry Pi 1) – được bảo trì độc lập bởi the Foundation; dựa trên cổng kiến trúc Debian ARM hard-float (armhf) được thiết kế ban đầu cho ARMv7 và các bộ xử lý kế tiếp (với Jazelle RCT/ThumbEE và VFPv3), biên dịch cho các tập lệnh hạn chế hơn ARMv6 của Raspberry Pi 1. Một thẻ SD có kích thước tối thiểu là 4 GB là cần thiết cho những Raspbian image được cung cấp bởi Raspberry Pi Foundation. Có một Pi Store (Kho) dùng để trao đổi chương trình.

- ✓ Raspbian Server Edition là một phiên bản rút gọn với các gói phần mềm đi kèm ít hơn so với phiên bản Raspbian dành cho máy tính để bàn thông thường.
- ✓ Giao thức máy chủ hiển thị Wayland cho phép sử dụng hiệu quả GPU để tăng tốc phần cứng chức năng vẽ GUI. Vào ngay 16 Tháng 4 năm 2014, một GUI shell dành cho Weston gọi Maynard đã được phát hành.
- ✓ PiBang Linux – là một biến thể từ Raspbian.
- ✓ Raspbian for Robots – là một biến thể của Raspbian dành cho các dự án robot với LEGO, Grove, và Arduino.

1.3 Giới thiệu tổng quan về bài toán phân lớp ảnh

Bài toán phân lớp ảnh (**Image Classification**) là một trong những nhiệm vụ phổ biến trong Computer Vision đã và đang được nhiều nhà nghiên cứu quan tâm và có nhiều phương pháp được đề xuất để giải quyết bài toán này với các mục tiêu nâng cao hiệu quả phân lớp .Một trong những cách tiếp cận phổ biến hiện nay cho bài toán này là phân lớp ảnh dựa trên đa đặc trưng ảnh. Trước tiên, chiếu mẫu cần phân lớp về nhiều không gian biểu diễn khác nhau. Trong giai đoạn này vấn đề đặt ra cần lựa chọn phép biến đổi ảnh thích hợp để trích chọn đặc trưng ảnh phù hợp với mục tiêu ứng dụng. Sau đó, sử dụng các kỹ thuật tính toán thông minh đánh giá phân loại ảnh theo các không gian này về các lớp tương ứng. Cuối cùng, hợp nhất các kết quả đánh giá để đưa ra kết luận. Cách tiếp cận này phản ánh đầy đủ tính đa dạng của mẫu cần phân lớp.

1.4 Bài toán phân lớp (Classification)

Đây là một bài toán phân lớp tương đối tiêu chuẩn, đã được nghiên cứu trong một thời gian khá dài. Một hệ thống nhận diện cảm xúc khuôn mặt thường được triển khai gồm 3 bước :

- ✓ Nhận ảnh và tiền xử lý: Ảnh khuôn mặt được lấy từ nguồn dữ liệu tĩnh (chẳng hạn như từ file, database), hoặc động (từ livestream, webcam, camera,…), nguồn dữ liệu này có thể trải qua một số bước tiền xử lý nhằm tăng chất lượng hình ảnh để giúp việc phát hiện cảm xúc trở nên hiệu quả hơn.
- ✓ Trích xuất các đặc trưng: Bước rất quan trọng, đặc biệt với các phương pháp truyền thống, các đặc trưng khuôn mặt được tính toán dựa trên các thuật toán có sẵn, kết quả thường là một vector đặc trưng làm đầu vào cho bước sau.

- ✓ Phân lớp : Đây là một bài toán phân lớp điển hình, rất nhiều các thuật toán có thể áp dụng trong bước này như KNN, SVM, LDA, HMM,…

Một vấn đề lớn đối với bài toán phân lớp ảnh là thiếu sót các dataset tiêu chuẩn đủ lớn. Các dataset cũng có nhiều khác biệt nhau về số lượng và cách phân loại, cũng như cách tính hiệu suất của các phương pháp phân loại.

1.4.1 Mô hình bài toán phân lớp tổng quát

Một đối tượng được biểu diễn bằng tập hợp n tính năng hoặc thuộc tính, được xem như một vector đặc trưng n chiều. Một mô hình phân lớp tổng quát thường sẽ có hai pha: đào tạo (Training), phận lớp (Classification), chính vì vaayjtoanf bộ dữ liệu được cung cấp được chia làm hai loại ,một là được dung để đào tạo mô hình (tập huấn luyện), hai được dung để kiểm tra mô hình (tập kiểm thử) ngoài ra ta có thể tạo một tập validation dùng để điều chỉnh trọng số mô hình tránh overfiting.

Dữ liệu được thu thập thông qua các thiết bị image sensor : Camera…

Dữ liệu sau khi thu thập thông qua thiết bị image sensor, sau đó đi qua pha training (analysisa/description) của mô hình phân lớp, ở đây (features extraction) sẽ trích rút đặ tính của đối tượng , các đặc tính sau khi được trích rút sẽ được lựa chọn (features selection) dựa trên các mục đích của mô hình phận lớp , việc làm này nhằm mục đích giảm thiểu các đặc tính tính cung cấp cho quá trình phân lớp classification (recogition). Các đặc tính đủ tốt sẽ được chuyển vào pha phân lớp. một đặc tính của đối tượng được coi là đủ tốt nếu đảm bảo các yếu tố sau :

- ✓ Tính bất biến dịch : Đặc tính được chọn phải như nhau ở mọi vị trí của đối tượng.

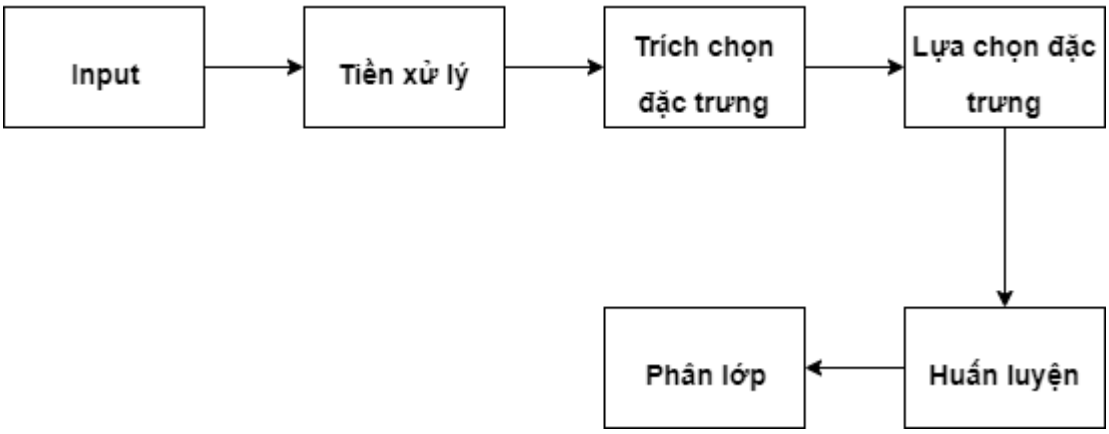
- ✓ Tính bất biến xoay : Các đặc tính được chọn không được thay đổi khi xoay đối tượng.
- ✓ Tính bất biến tỷ lệ: Việc thay đổi tỷ lệ của đối tượng không ảnh hưởng nhiều đến đặc tính.
- ✓ Tính chống nhiễu : Các đặc tính được chọn ra phải có khả năng chống lại các yếu tố gây nhiễu ,các đặc tính không bị thay đổi kể cả khi có sự tác động của các yếu tố gây nhiễu.
- ✓ Nhỏ gọn : Số lượng các đặc tính được chọn ra không được quá lớn.
- ✓ Độ tin cậy : Dù có trải qua các quá trình tác động đặc tính được chọn của đối tượng tượng phải được giữ nguyên.

Trong quá trình phân loại, hệ thống sẽ sử dụng các đặc tính đã được chọn để tiến hành so sánh. Để so sánh hai đối tượng mô hình sử dụng một số liệu đo khoảng cách (mức độ tương tự hoặc mức độ khác nhau của đối tượng) để đánh giá. Đó là biểu thức tính khoảng cách giữa các điểm đối diện cho các đối tưởng trong không gian đặc tính.

Mô hình phân lớp truyền thống :

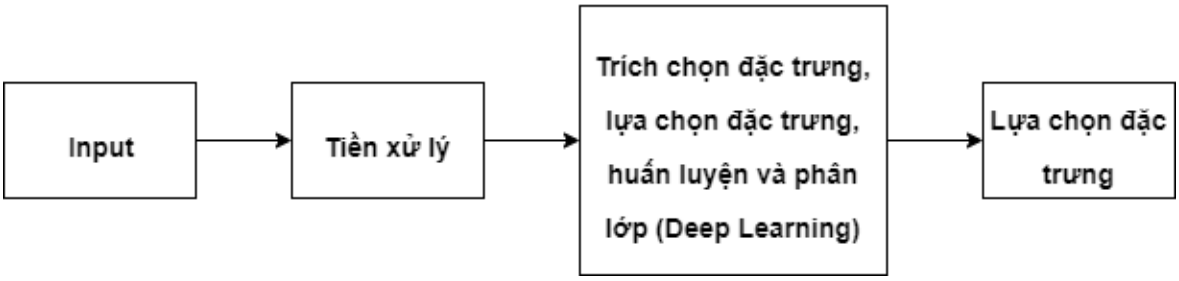
Sự ra đời của Deep Learning mang theo những phát triển vượt bậc trong công nghệ nhận dạng, Deep Learning khiến việc nhận dạng trở nên dễ dàng và thông minh hơn. Có thể hình dung sự thay đổi của mô hình nhận dạng khi sử dụng Deep Learning trong so sánh dưới đây :

Thông thường một mô hình nhận dạng, phân lớp sẽ phải bao gồm các giai đoạn như sơ đồ dưới đây :



Hình 1 0.4 Mô hình nhận dạng, phân lớp truyền thống

Tuy nhiên khi sử dụng Deep Learning việc nhận dạng của bạn sẽ thông minh hơn rất nhiều, về cơ bản khối lượng công việc cần xử lý sẽ được giảm đi đáng kể :



Hình 1 0.5 Mô hình phân lớp sử dụng Deep Learning

Và khi nhắc đến Deep Learning trong ứng dụng nhận dạng, phân lớp thì Convolution Neural Networks (CNNs – mạng neural tích chập) là một trong những mô hình Deep Learning tiên tiến và độ chính xác cao. Mỗi CNNs gồm có hoặc nhiều hơn các lớp tích chập (Convolution) với các lớp đầy đủ két nối đến các đỉnh. So với các mô hình khác CNNs được sử dụng chung trọng số của Convolution nên mô hình tương đối nhẹ, giảm thiểu khả năng tính toán, quá trình huấn luyện nhanh và cho kết quả tương đối chính xác. Chúng cũng có thể được huấn luyện với tiêu chuẩn lan truyền ngược

1.5 Phạm vi đề tài

Như đã nêu các phần trên, bài toán nhận dạng, phân lớp có rất nhiều ứng dụng trong thực tế. Trong đồ án II em sử dụng mô hình mạng neural tích chập (Convolution Neural Networks – CNNs) để xây dựng chương trình nhận dạng, phân lớp trên hệ thống nhúng raspberry pi. Tuy nhiên mức độ nghiên cứu của đồ án II này em mới chỉ dừng ở phần tìm hiểu lý thuyết và xây dựng demo chương trình trên laptop cá nhân.

CHƯƠNG 2 CƠ SỞ LÝ THUYẾT MẠNG NEURAL TÍCH CHẬP (CONVOLUTION NEURAL NETWORKS) VÀ ỨNG DỤNG TRONG BÀI TOÁN PHÂN LỚP ẢNH

2.1 Lý thuyết về mạng neural tích chập (Convolutional Neural Networks)

2.1.1 Các định nghĩa liên quan

Đối tượng (Parttern): Là một thực thể có tính áng chừng mà chúng ta có thể gán cho nó một cái tên. Ví dụ : màu sắc trên đồ vật, biển số xe, khuôn mặt người, tín hiệu giọng nói…

Lớp (Class or category) : Tập các đối tượng có cùng các tính chất đặc trưng được nhóm thành một lớp. Ví dụ : lớp đồ vật cứng, lớp động vật ăn thịt, lớp chất khí …

Đặc trưng (Features) : Sự mô tả phần tử bất kỳ được lấy làm đại diện cho những phần tử khác trong cùng một lớp mà được đồng nhất với phần tử đại diện bởi các tính chất chung. Một số dạng đối tượng thường gặp: cố định, tạm thời, tín hiệu …

Nhận dạng (Recognition or Classificcation) : nhận dạng là việc phân lớp đối tượng (được thể hiện bằng các thuộc tính của nó) vào một trong các lớp cố định cho trước, theo quy tắc giải quyết nhất định, tương đồng của các đối tượng cùng loại. Ví dụ : Nhận dạng giọng nói, nhận dạng mặt người, nhận dạng tài liệu …

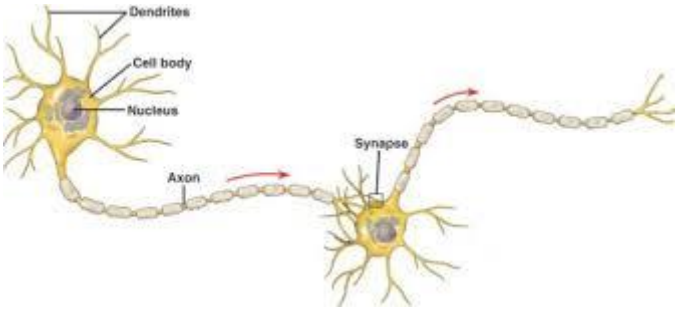
2.1.2 Giới thiệu mạng Neural

Một mạng Neural được cấu thành bởi các Neural đơn lẻ được gọi là các perceptron. Nên trước tiên ta tìm hiểu xem perceptron là gì đã rồi tiến tới mô hình của mạng Neural sau. Neural nhận tạo được lấy cảm hứng từ mạng Neural sinh học.

2.1.2.1 Neural sinh học và mạng Neural sinh học

Một Neural sinh học là một tế bào xử lý và truyền thông tin bằng các tín hiệu hóa học qua một khớp thần kinh tới các tế bào khác. Hệ thần kinh con người có khoảng khoảng 10^{10} tế bào Neural thần kinh và mỗi tế bào Neural này có thể liên kết với 10^{4} tế bào Neural khác thông qua các khớp nốt (các xung thần kinh).

Mỗi neural sinh học sẽ bao gồm ba phần đó là : thân tế bào, các hình cây và một sợi trục như trong hình dưới đây



Hình 2 0.1 Mạng neural sinh học

Thân tế bào có nhiệm vụ tiếp nhận hay phát ra các xung thần kinh, bên trong có nhân (soma) ,hệ thống dây thần kinh vào (dendrites – còn gọi là các nhánh thụ giác) để dẫn truyền các xung thần kinh. Các đầu day thần kinh vào nhận tín hiệu từ các neural khác, nhân sẽ sinh ra tín hiệu ở đầu ra của neural và truyền tới các neural khác được nối với đầu ra qua trục.

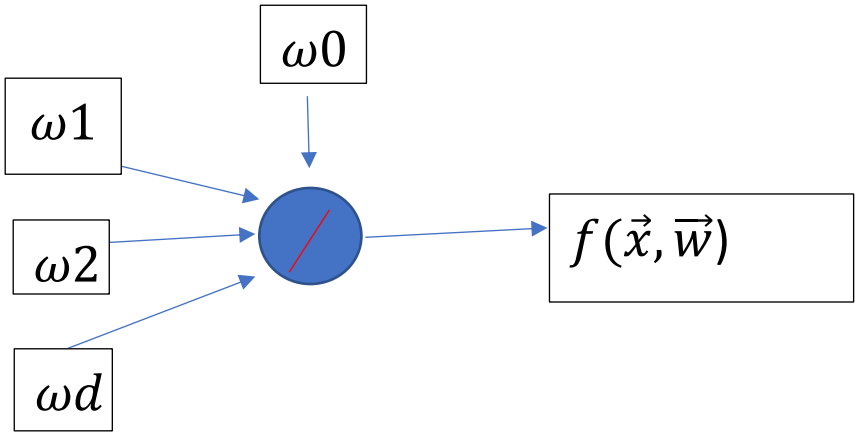
Mỗi neural liên kết với các neural khác tạo thành mạng neural sinh học. Theo các nhà nghiên cứu, chức năng của hệ thần kinh không phụ thuộc nhiều vào vai trò của từng neural đơn lẻ mà phụ thuộc vào cách mà toàn bộ các neural được nối với nhau, tức mạng neural. Từ các đặc điểm của mạng neural sinh học trên được vận dụng để xây dựng một mạng neural nhận tạo có thể giải quyết các bài toán đặt ra [8]

2.1.2.2 Mạng neural nhân tạo

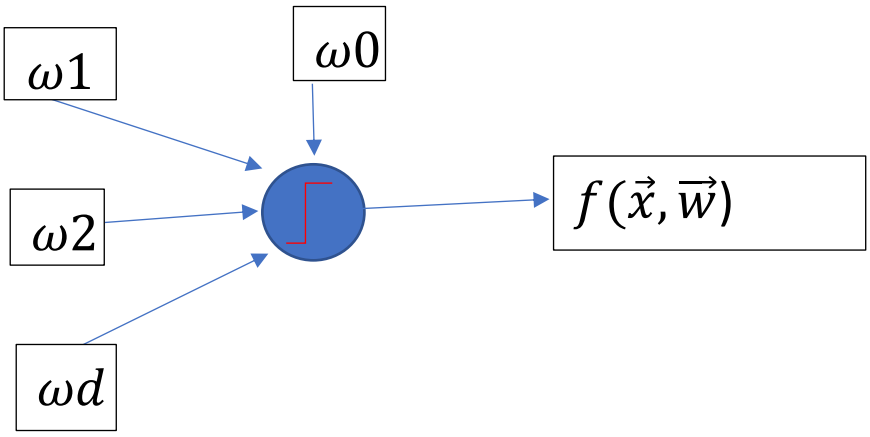
Mạng neural nhân tạo hay thường gọi ngắn gọn mạng neural (tiếng Anh là Artificial Neural network - ANN hay Neural Network) là một mô hình toán học hay mô hình tính toán được xây dựng dựa trên các mạng neural sinh học. Nó gồm có một nhóm các neural nhân tạo (nút) nối với nhau thông qua các liên kết (trọng số liên kết) và làm việc như một thể thống nhất để giải quyết một vấn đề cụ thể nào đó. Việc xử lý thông tin bằng cách truyền theo các kết nối và tính giá trị mới tại các nút (cách tiếp cận connectionism đối với tính toán). Trong nhiều trường hợp, mạng neural nhân tạo là một hệ thống thích ứng (*adaptive system*) tự thay đổi cấu trúc của mình dựa trên các thông tin bên ngoài hay bên trong chảy qua mạng trong quá trình học. [8]

Trong thực tế sử dụng, nhiều mạng neural là các công cụ mô hình hóa dữ liệu thống kê phi tuyến. Chúng có thể được dùng để mô hình hóa các mối quan hệ phức tạp giữa dữ liệu vào và kết quả hoặc để tìm kiếm các dạng/mẫu trong dữ liệu.

Một mạng neural nhân tạo được cấu hình cho một ứng dụng cụ thể (nhận dạng mẫu, phân loại dữ liệu…) thông qua một quá trình học từ tập các mẫu huấn luyện. Một mạng neural nhân tạo thường được kết hợp từ ba thành phần : đơn vị xử lý, hàm kết hợp, hàm kích hoạt. Ví dụ một số mạng neural cơ bản:



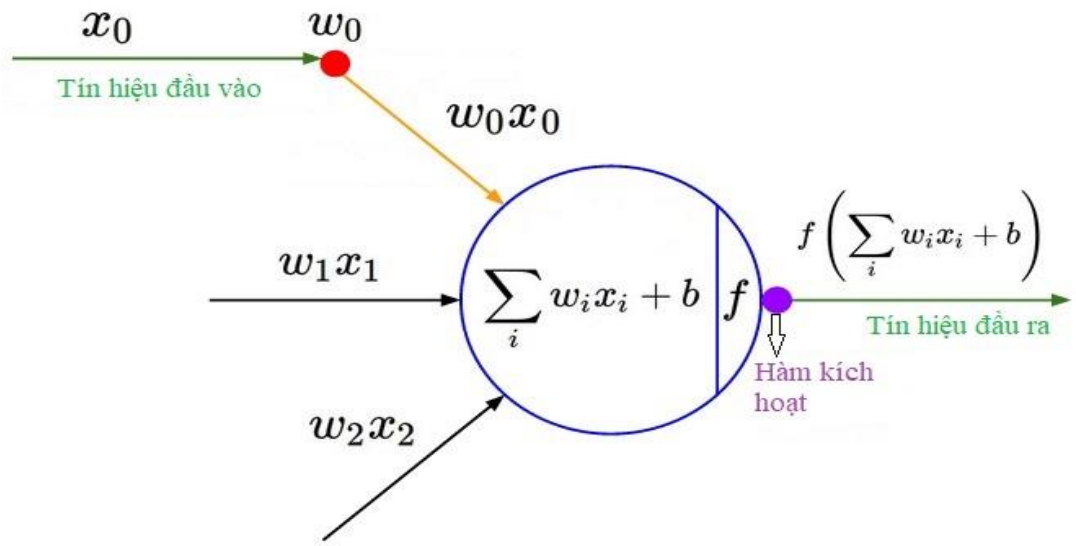
Hình 2 0.2 Mô hình mạng neural tuyến tính



Hình 2 0.3 Mô hình mạng neural sigmoid

Đơn vị xử lý :

Còn được gọi là một neural hay một nút (node), thực hiện một công việc rất đơn giản : nó nhận tín hiệu vào từ các đơn vị phía trước hay một nguồn bên ngoài và sử dụng chúng để tính tín hiệu ra sẽ được làn truyền sang một đơn vị khác cấu trúc của một neural được minh họa như sau :



Hình 2 0.4 Mô hình chung của mạng neural

Trong đó :

- ✓ X_i là các tín hiệu đầu vào

- ✓ W_i là các trọng số. Tương ứng với X_i có trọng số W_iX_i
- ✓ b là hệ số bias
- ✓ $a_i = \sum_i w_i x_i + b$ là đầu vào mạng (net input)
- ✓ $z_i = f(\sum_i w_i x_i + b)$ là đầu ra của neural
- ✓ f hàm kích hoạt

Một mạng neural có ba kiểu đơn vị :

- ✓ Các đơn vị đầu vào (input units), nhận tín hiệu từ bên ngoài
- ✓ Các đơn vị đầu ra (output units), gửi tín hiệu ra bên ngoài
- ✓ Các đơn vị ẩn (hidden units) các tín hiệu vào (input) và ra (output) của nó nằm trong mạng

Mỗi đơn vị *i* có thể có một hoặc nhiều đầu vào nhưng chỉ có một đầu ra. Một đầu vào tới một đơn vị có thể là dữ liệu từ bên ngoài mạng, hoặc là đầu ra của một đơn vị khác, hoặc là đầu ra của chinhd nó.

Hàm kết hợp :

Mỗi một đơn vị trong mạng kết hợp các giá trị đưa vào nó thông qua các liên kết với đơn vị khác, sinh ra một giá trị gọi là net input. Hàm thực hiện nhiệm vụ này gọi là hàm kết hợp (combination function), được định nghĩa bởi luật lan truyền cụ thể. Trong phần lớn các mạng neural, chúng ta giả sử rằng mỗi một đơn vị cung cấp một bộ cộng như là đầu vào cho đơn vị mà nó có liên kết. tổng đầu vào đơn vị *i* đơn giản chỉ là trọng số của các đầu ra riêng lẻ từ các đơn vị kết nối cộng thêm ngưỡng hay độ lệch *b :*

$$a_i = \sum_i w_i x_i + b$$

Trường hợp $w_i > 0$, neural được coi là đang ở trong trạng thái kích thích. tương tự nếu như $w_i < 0$, neural ở trạng thái kiềm chế.

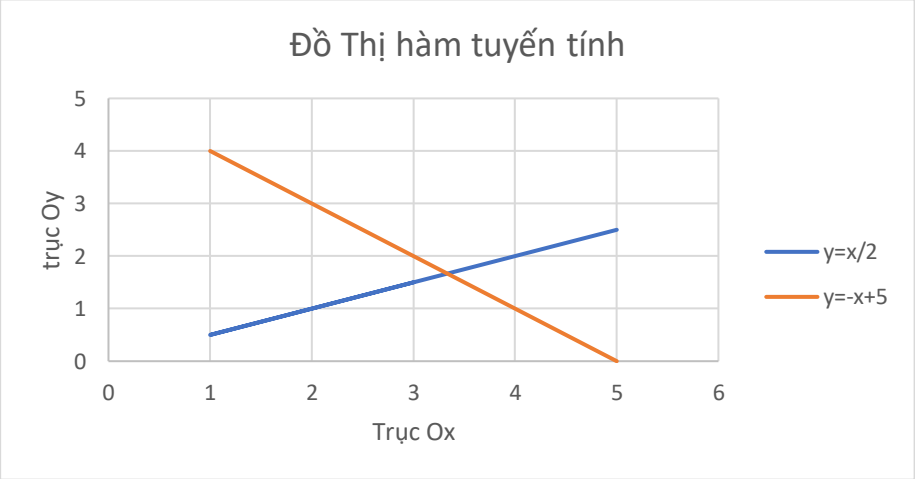
Hàm kích hoạt :

Phần lớn các đơn vị trong mạng neural chuyển net input bằng cách sử dụng một hàm vô hướng (scalar-to-scalar function) gọi là hàm kích hoạt [9], kết quả của hàm này là một giá trị gọi là mức kích hoạt của đơn vị (unit's activation). Loại trừ khả năng đơn vị đó thuộc lớp ra, giá trị kích hoạt được đưa vào một hay nhiều đơn vị khác. Các hàm kích hoạt thường bị ép vào một khoảng giá trị xác định, do đó thường được gọi là các hàm bẹp (squashing). Các hàm kích hoạt được sử dụng là :

Hàm đồng nhất (Linear function , Identity function) :

$$g(x)=x$$

Nếu coi các đầu vào là một đơn vị thì chúng sẽ sử dụng hàm này. Đôi khi một hằng số được nhân với net-input để tạo ra một hàm đồng nhất.



Hình 2 0.5 Đồ thị hàm tuyến tính

Công thức Loss function :

$$J = \frac{1}{2} * \frac{1}{N} * \left(\sum_{i=1}^{N} (\hat{y}_i - y_i) \right)$$

$$\hat{y}_i = \omega_i * x_i + \omega_0$$

- ✓ J không âm
- ✓ J càng nhỏ thì đường thẳng càng gần điểm dữ liệu. Nếu j=0 thì đường thẳng đi qua tất cả các điểm dữ liệu

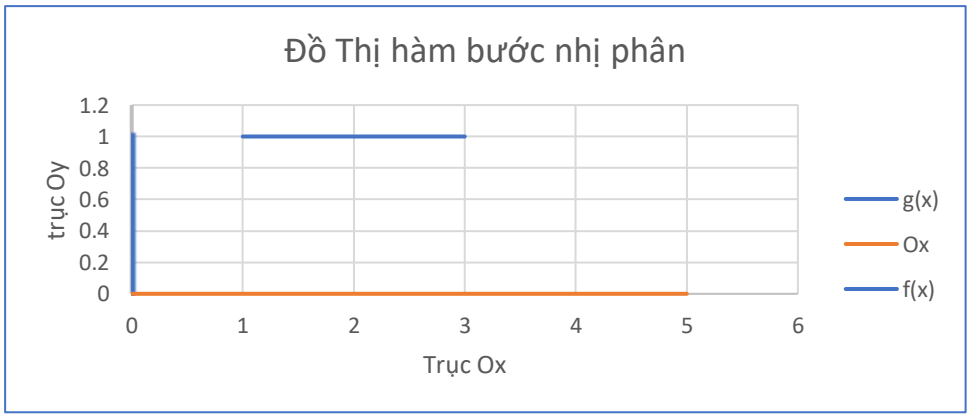
  => bài toán đặt ra là cực tiểu hàm loss function để thu được kết quả tốt nhất có thể

Hàm bước nhị phân (Binary step function, Hard limit function) :

Hàm này cũng được biết đến với tên "Hàm ngưỡng" (Threshold function hay Heaviside function) [10]. Đầu ra của hàm này được giới hạn vào một trong hai giá trị:

$$g(x) = \begin{cases} 1\,, nếu\ (x > \theta) \\ 0\,, nếu\ (x < \theta\ ) \end{cases}$$

Dạng hàm này được sử dụng trong các mạng chỉ có một lớp. Trong hình vẽ sau, *b* được chọn bằng 1.



Hình 2 0.6 Đồ thị hàm bước nhị phân

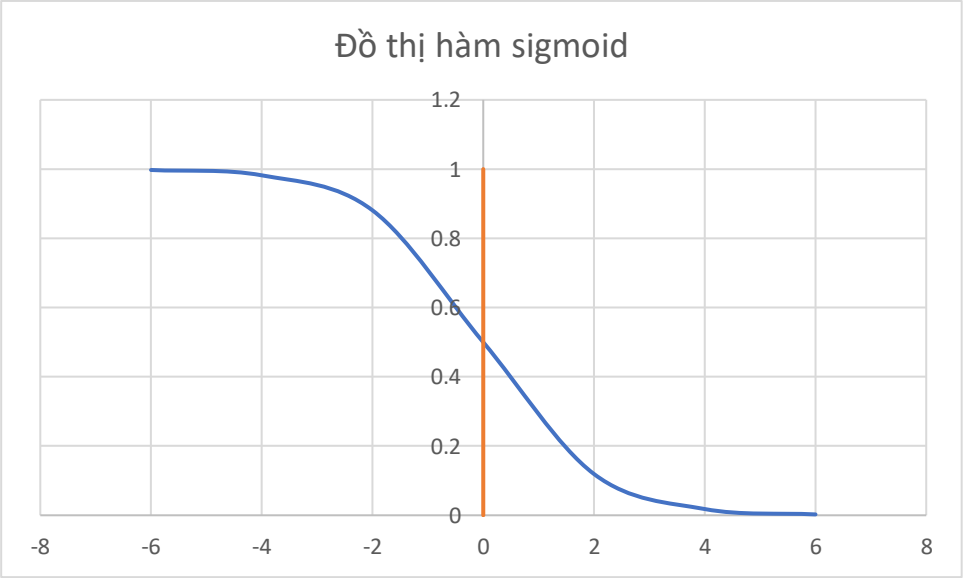
Hàm sigmoid (Sigmoid function (logsig)):

$$g(x) = \frac{1}{1 + e^{-x}}$$

Quan hệ linear function với logistic function :

$$\hat{y} = \sigma\left(\omega_0 + \omega_1 * x_1^{(i)} + \omega_2 * x_2^{(i)}\right) = \frac{1}{1 + e^{-(\omega_0 + \omega_1 * x_1^{(i)} + \omega_2 * x_2^{(i)})}}$$

Hàm này đặc biệt thuận lợi khi sử dụng cho các mạng được huấn luyện (trained) bởi thuật toán Lan truyền ngược (back-propagation), bởi vì nó dễ lấy đạo hàm, do đó có thể giảm đáng kể tính toán trong quá trình huấn luyện. Hàm này được ứng dụng cho các chương trình ứng dụng mà các đầu ra mong muốn rơi vào khoảng [0,1].



Hình 2 0.7 Đồ thị hàm Sigmoid

Đạo hàm logistic function :

Hàm logistic có đạo hàm tại mọi điểm và dễ tính toán :

$$\frac{d}{dx}f(x) = f(x)(1 - f(x))$$

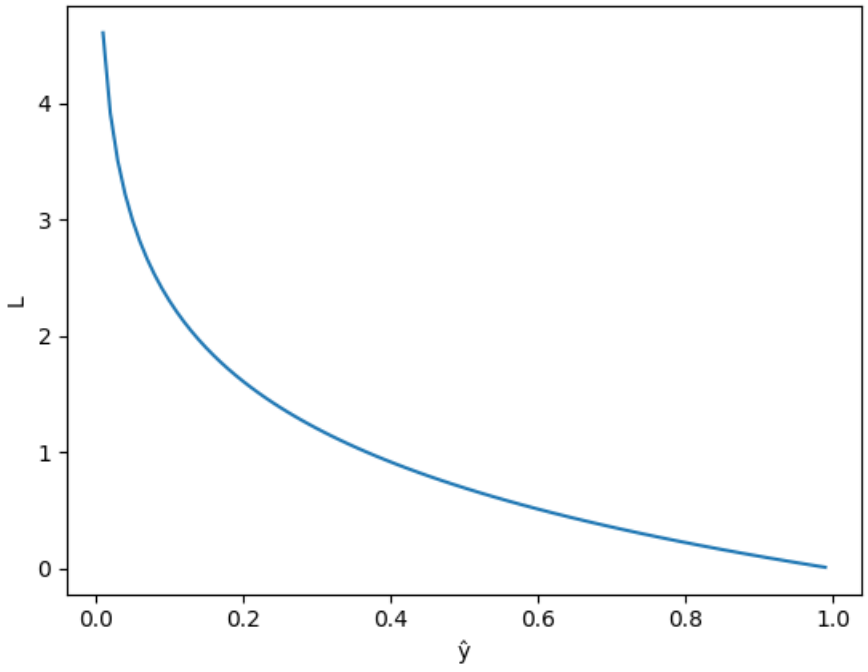
Công thức Loss function :

Đây là hàm đánh giá độ tốt của model. Như vậy $\hat{y}$ càng gần y càng tốt :

$$L = -(y_i * \log(\hat{y_i}) + (1 - y_i) * \log(1 - \hat{y_i}))$$

Đánh giá L nếu :

$$y_i = 1 => L = -\log(\hat{y_i})$$



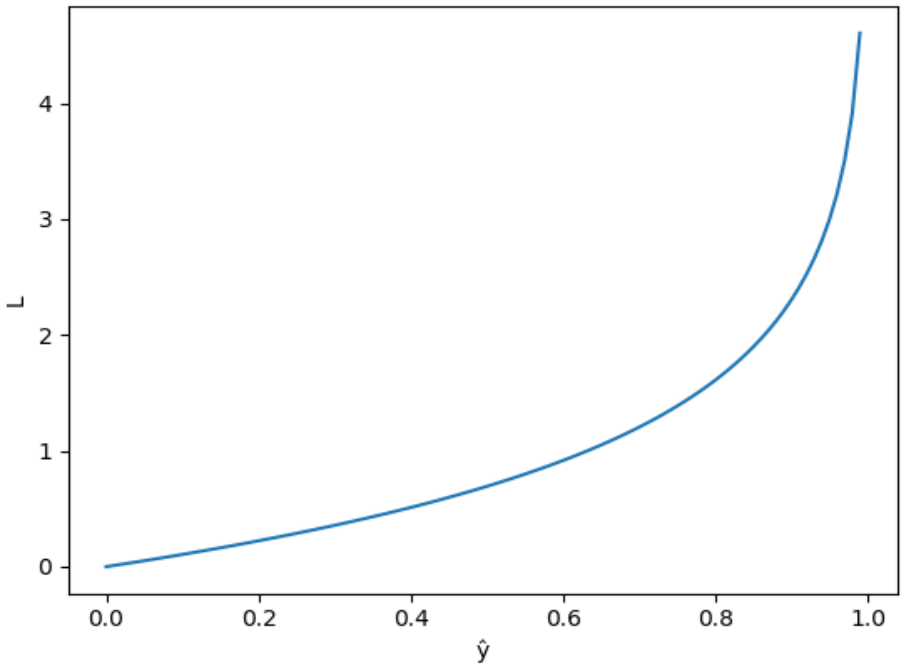
Hình 2 0.8 Đồ thị loss function – logistic (trường hợp : $y_i = 1$)

- ✓ Hàm L giảm dần từ 0 đến 1.
- ✓ Khi model dự đoán $\hat{y_i}$ gần 1, tức giá trị dự đoán gần với giá trị thật y_i thì L nhỏ, xấp xỉ 0.

- ✓ Khi model dự đoán $\widehat{y_i}$ gần 0, tức giá trị dự đoán ngược lại giá trị thật y_i thì L rất lớn.

Ngược lại nếu :

$$y_i = 0 => L = -\log(1 - \widehat{y_i})$$



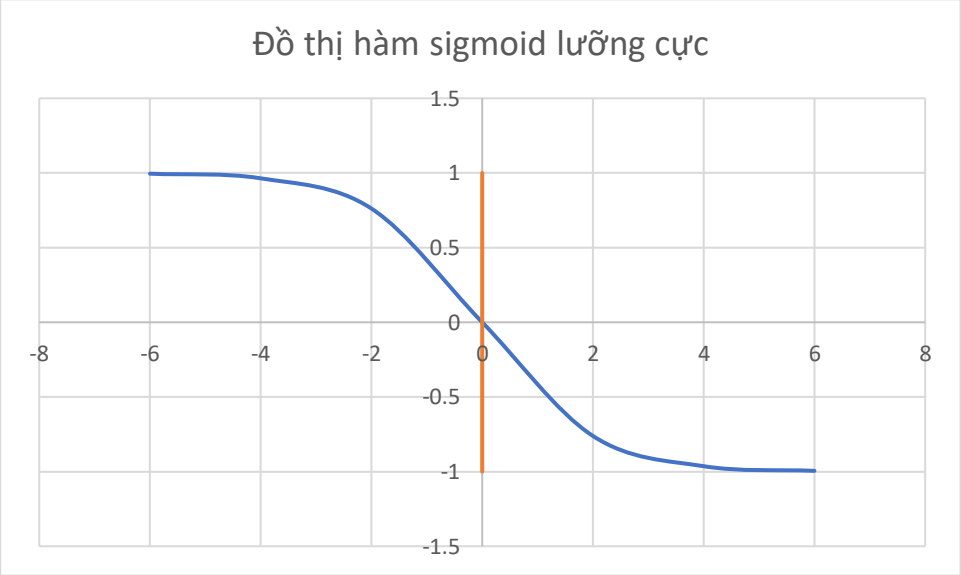
Hình 2 0.9 Đồ thị loss function - logistic (trường hợp $y_i = 0$)

- ✓ Hàm L tăng dần từ 0 đến 1.
- ✓ Khi model dự đoán $\widehat{y_i}$ gần 0, tức giá trị dự đoán gần với giá trị thật y_i thì L nhỏ, xấp xỉ 0.
- ✓ Khi model dự đoán $\widehat{y_i}$ gần 1, tức giá trị dự đoán ngược lại giá trị thật y_i thì L rất lớn.

Hàm sigmoid lưỡng cực (Bipolar sigmoid function (tansig)) :

$$g(x) = \frac{1 - e^{-x}}{1 + e^{-x}}$$

Hàm này có các thuộc tính tương tự hàm sigmoid. Nó làm việc tốt đối với các ứng dụng có đầu ra yêu cầu trong khoảng [-1,1].



Hình 2 0.10 Đồ thị hàm sigmoid lưỡng cực

Các hàm chuyển của các đơn vị ẩn (hidden units) là cần thiết để biểu diễn sự phi tuyến vào trong mạng. Lý do là hợp thành của các hàm đồng nhất là một hàm đồng nhất. Mặc dù vậy nhưng nó mang tính chất phi tuyến (nghĩa là, khả năng biểu diễn các hàm phi tuyến) làm cho các mạng nhiều tầng có khả năng rất tốt trong biểu diễn các ánh xạ phi tuyến. Tuy nhiên, đối với luật học lan truyền ngược, hàm phải khả vi (differentiable) và sẽ có ích nếu như hàm được gắn trong một khoảng nào đó. Do vậy, hàm sigmoid là lựa chọn thông dụng nhất.

Đối với các đơn vị đầu ra (output units), các hàm chuyển cần được chọn sao cho phù hợp với sự phân phối của các giá trị đích mong muốn. Chúng ta đã

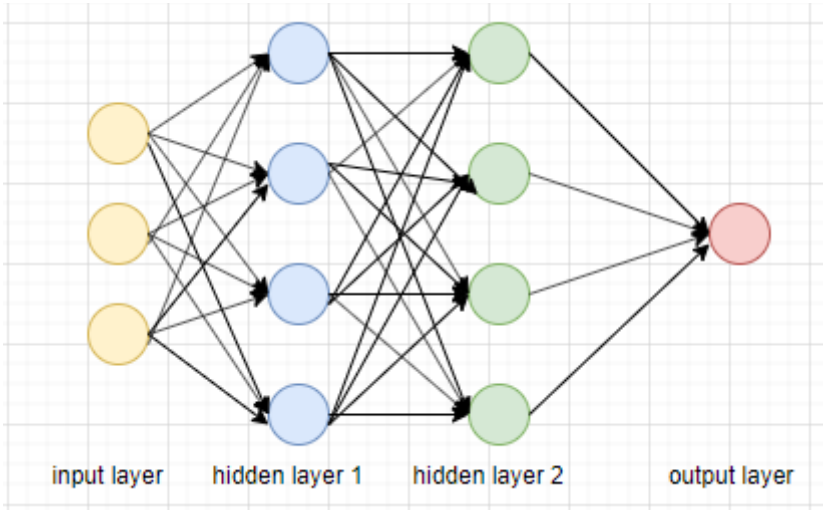
thấy rằng đối với các giá trị ra trong khoảng [0,1], hàm sigmoid là có ích; đối với các giá trị đích mong muốn là liên tục trong khoảng đó thì hàm này cũng vẫn có ích, nó có thể cho ta các giá trị ra hay giá trị đích được căn trong một khoảng của hàm kích hoạt đầu ra. Nhưng nếu các giá trị đích không được biết trước khoảng xác định thì hàm hay được sử dụng nhất là hàm đồng nhất (identity function). Nếu giá trị mong muốn là dương nhưng không biết cận trên thì nên sử dụng một hàm kích hoạt dạng mũ (exponential output activation function).

2.1.2.3 Phân loại mạng neural

Phân loại theo cách truyền tín hiệu

Dựa vào cách truyền tín hiệu trong mạng neural ta có thể phân loại thành một số cách truyền tín hiệu :

Mạng lan truyền tiến : Gồm các mạng perceptron một lớp , mạng perceptron nhiều tầng và mạng RBF.



Hình 2 0.11 Mô hình mạng lan truyền tiến

Như bạn thấy thì tất cả các nốt mạng (neural) được kết hợp đôi một với nhau theo một chiều duy nhất từ tầng vào tới tầng ra. Tức là mỗi nốt ở một tầng nào đó sẽ

nhận đầu vào là tất cả các nốt ở tầng trước đó mà không suy luận ngược lại. Hay nói cách khác, việc suy luận trong mạng neural network là **suy luận tiến** (feedforward) :

$$z_i^{(l+1)} = \sum_{j=1}^{n^l} w_{ij}^{(l+1)} a_j^{(l)} + b_i^{(l+1)}$$

$$a_i^{(l+1)} = f\left(z_i^{(l+1)}\right)$$

Trong đó, $n^{(l)}$ số lượng nút ở tầng *l* tương ứng và $a_j^{(l)}$ là nút mạng thứ j của tầng *l*. Còn $w_{ij}^{(l+1)}$ là tham số trọng lượng của đầu vào $a_j^{(l)}$ đối với nút mạng thứ *i* của tầng *l+1* và $b_i^{(l+1)}$ là độ lệch (*bias*) của nút mạng thứ *i* của tầng *l+1*. Đầu ra của nút mạng này được biểu diễn bằng $a_j^{(l+1)}$ ứng với hàm kích hoạt $f(z_i)$ tương ứng.

Riêng với tầng vào, thông thường $a^{(l)}$ cũng chính là các đầu vào X tương ứng của mạng.

Để tiện tính toán, ta coi $a_0^{(l)}$ là một đầu vào và $w_{i0}^{(l+1)} = b_i^{(l+1)}$ là tham số trọng lượng của đầu vào này. Lúc đó ta có thể viết lại công thức trên dưới dạng véc-tơ:

$$z_i^{(l+1)} = w_i^{(l+1)} . a^{(l)}$$

$$a_i^{(l+1)} = f(z_i^{(l+1)})$$

Nếu nhóm các tham số của mỗi tầng thành một ma trận có các cột tương ứng với tham số mỗi nút mạng thì ta có thể tính toán cho toàn bộ các nút trong một tầng bằng véc-tơ:

$$z^{(l+1)} = w^{(l+1)}.a^{(l)}$$

$$a^{(l+1)} = f(z^{(l+1)})$$

Mạng lan truyền ngược :

Mạng lan truyền ngược hay còn được gọi là mạng phản hồi (truy hồi) được sử dụng khá phổ biến trong các model của AI hiện nay như DeepID-X hay CNNs và đã được ứng dụng trong thực tế như: dùng làm bộ nhớ địa chỉ hóa nội dung; dùng làm các bộ tối ưu …

2.1.3 Mạng neural tích chập

Convolutional Neural Networks (CNNs – Mạng nơ-ron tích chập) là một trong những mô hình Deep Learning tiên tiến. Nó giúp cho chúng ta xây dựng được những hệ thống thông minh với độ chính xác cao như hiện nay. Như hệ thống xử lý ảnh lớn như Facebook, Google hay Amazon đã đưa vào sản phẩm của mình những chức năng thông minh như nhận diện khuôn mặt người dùng, phát triển xe hơi tự lái hay drone giao hàng tự động.

CNNs được sử dụng nhiều trong các bài toán nhận dạng, phân lớp các object trong ảnh. Để tìm hiểu tại sao thuật toán này được sử dụng rộng rãi cho việc nhận dạng (detection).

2.1.3.1 Định nghĩa mạng neural tích chập

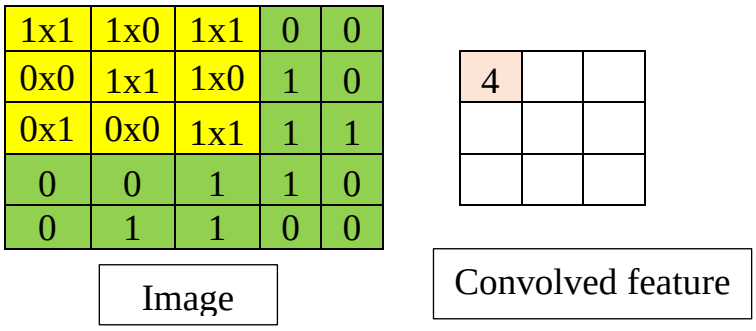
Mô hình mạng neural nhân tạo truyền thẳng ra đời đã được áp dụng rất nhiều trong các bài toán nhận dạng Tuy nhiên mạng neural truyền thẳng không thực hiện tốt lắm với các bài toán dữ liệu nhận dạng hình ảnh. Chính vì sự quá

đầy đủ tạo nên những hạn chế cho mô hình. Ví dụ như một hình ảnh có kích thước 32x32 pixel sẽ cho ra vector đặc trưng có 1024 chiều, còn đối với ảnh màu có cùng kích thước sẽ là 3072 chiều. Điều này có nghĩa là ta sẽ cần đến 3072 trọng số nối giữa lớp ẩn kế tiếp. Như vậy với một bức ảnh nhỏ (32x32) thì ta cũng cần đến một mô hình đồ sộ, điều này càng khó khan hơn khi thao tác với các ảnh kích thước lớn hơn nữa. Chính vì vậy mạng Convolution Neural Netwworks (CNNs- mạng neural tích chập) ra đời. Mạng neural tích chập (CNNs hoặc ConvNet) là một trong thuật toán học hình ảnh và video phổ biến nhất. Giống như các mạng neural khác, CNNs bao gồm một lớp đầu vào, một lớp đầu ra và nhiều lớp ẩn ở giữa [9].

2.1.3.2 Các lớp trong mô hình neural tích chập

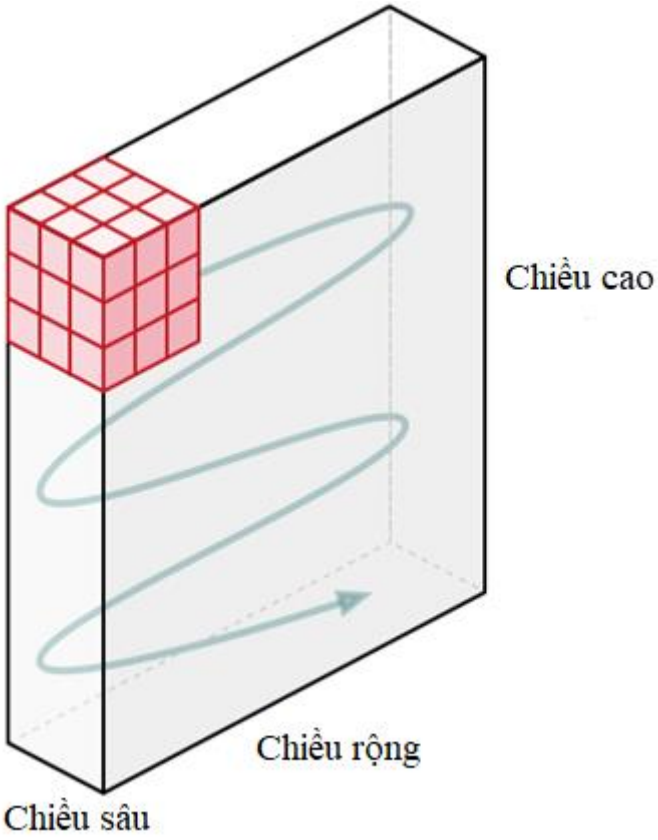
Feature Dertection Layers: Khối này bao gồm 3 lớp được sử dụng kết hợp với nhau nhằm mục đích trích xuất đặc trưng, và lựa chọn đặc trưng.

Convolution : Đây là lớp quan trọng nhất trong cấu trúc của CNNs, đây chính là nơi trích xuất các đặc trưng thể hiện tư tưởng ban đầu của mạng neural tích chập [8]. Thay vì kết nối toàn bộ điểm, lớp này sẽ sử dụng một bộ kernel có kích thước nhỏ hơn so với ảnh (thường là 2x2 cho đến 5x5) áp vào một vùng trong ảnh và tiến hành tích chập giữa bộ kernel này và giá trị điểm ảnh trong vùng cục bộ đó. Kernel sẽ lần lượt được dịch chuyển theo một giá trị bị trượt (stride) chạy dọc theo ảnh và quét toàn bộ ảnh.



Hình 2 0.12 Mô phỏng hoạt động của Convolution

Tuy nhiên ảnh màu có tới 3 channels red, green, blue nên khi biểu diễn ảnh dưới dạng tensor 3 chiều. Nên ta cũng sẽ định nghĩa kernel là 1 tensor 3 chiều kích thước k*k*3.

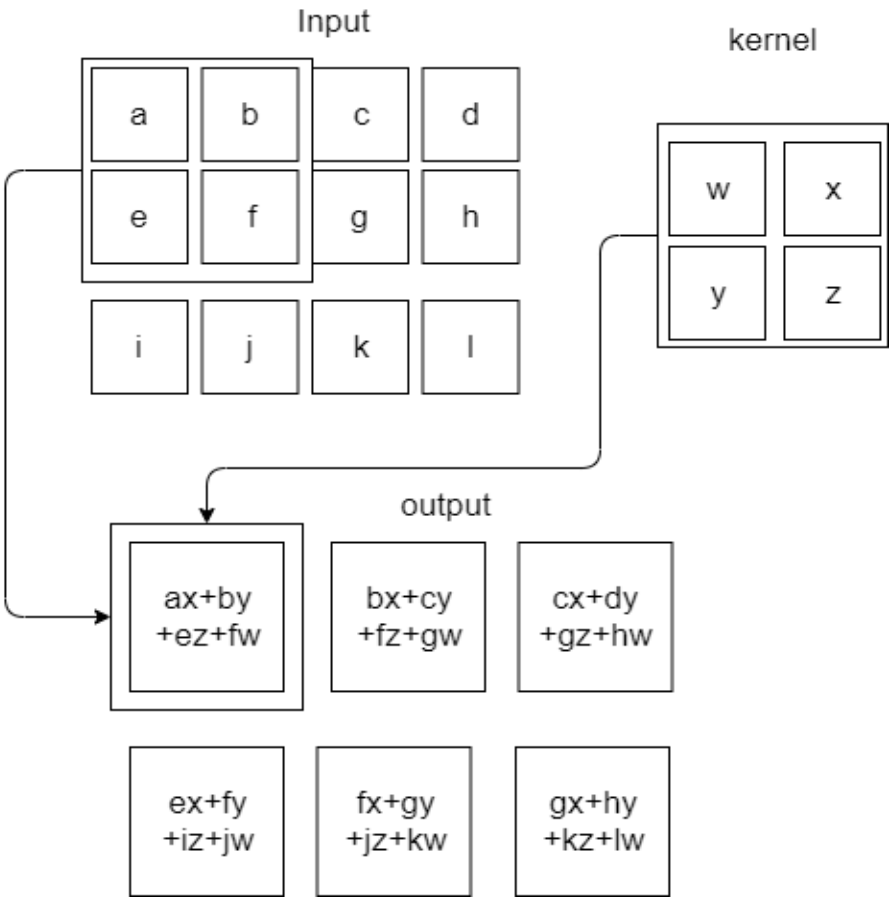


Hình 2 0.13 Mô tả kernel

Phép tính convolution trên ảnh màu với k=3. Như vậy sử dụng Convolution sẽ có những ưu điểm sau :

Giảm số lượng tham số : Ở ANN truyền thống , các neural ở lớp trước sẽ kết nối tất cả các neural ở lớp sau (full connected) gây nên tình trạng quá nhiều tham số cần học. Đây là nguyên nhân chính gây nên tình trạng overfiting cũng như tang thời gian huấn luyện. Việc sử dụng Convolution trong đó cho phép chia sẻ sử dụng local receptive fields giúp giảm tham số.

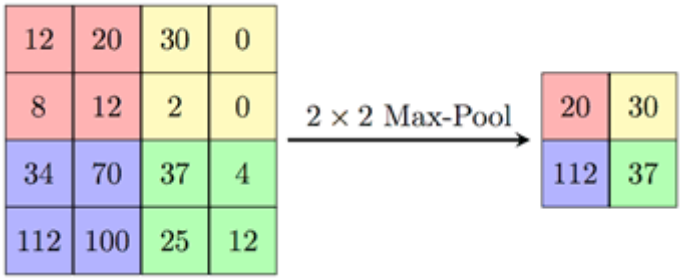
Các tham số trong quá trình sử dụng conv hay giá trị của các filter-kernel sẽ được học trong quá trình huấn luyện. Như giới thiệu ở phần trên các thông tin này biểu thị thông tin giúp rút trích ra được các đặc trưng như góc, cạnh, đốm màu trong ảnh … như vậy việc sử dụng Conv sẽ giúp xây dựng mô hình tự học rất đặc trưng.



Hình 2 0.14 Mô phỏng quá trình hoạt động của convolution

Lớp pooling sử dụng một cửa sổ trượt quét qua toàn bộ ảnh dữ liệu, mỗi lần trượt theo một bước trượt (stride) cho trước. Tuy nhiên, khác với lớp Convlution, lớp pooling không tính tích chập mà tiến hành lấy mẫu (subsampling) [9]. Khi cửa sổ trượt trên ảnh chỉ có một giá trị được xem là giá trị đại diện cho thông tin ản tại vùng đó (giá trị mẫu) được giữ lại. Các phương thức lấy phổ biến trong lớp pooling là maxpooling (lấy giá trị lớp nhất),

minpooling (lấy giá trị nhỏ nhất) và averagepooling (lấy giá trị trung bình). Xét một ảnh có kích thước 32x32 và lớp pooling sử dụng là filter có kích thước 2x2 và bước trượt stride = 2, phương pháp sử dụng maxpooling. Filter sẽ lần lượt duyệt qua ảnh, với mỗi lần duyệt chỉ có giá trị lớn nhất trong bốn giá trị nằm trong vùng cửa sổ 2x2 của filter được giữ lại và đưa ra đầu ra. Như vậy sau khi qua pooling, ảnh sẽ giảm kích thước xuống 16x16 (mỗi chiều giảm 2 lần).



Hình 2 0.15 Max - Pooling

Lớp pooling có vai trò làm giảm kích thước dữ liệu. Với mỗi bức ảnh kích thước lớn qua nhiều lớp pooling sẽ được thu nhỏ lại tuy nhiên vẫn giữ được những đặc cần cho việc nhận dạng (thông tin cách lấy mẫu). Việc giảm kích thước dữ liệu sẽ làm giảm lượng tham số, tăng hiệu quả tính toán và góp phần kiểm soát hiện tượng quá khớp (overfitting) [11].

Lớp ReLU – Rectified Linear Unit : về cơ bản Convolution là một phép biến đổi tuyến tính. Nếu tất cả các neural được tổng hợp bởi các phép biến đổi tuyến tính thì một mạng neural đều có thể đưa về dạng một hàm tuyến tính. Khi đó mạng ANN sẽ đưa các bài toán về logistic regression. Do đó tại mỗi neural cần có một hàm truyền dưới dạng phi tuyến [11].

Lớp này sử dụng hàm kích hoạt $f(x) = \max(0, x)$. Nói một cách đơn giản, lớp này có nhiệm vụ chuyển toàn bộ giá trị âm trong kết quả lấy từ lớp Convolution thành giá trị 0. Ý nghĩa của cách cài sặt này chính là tạo nên tính phi tuyến cho mô hình. Tương tự như trong mạng truyền thẳng, việc xây dựng

đa tầng đa lớp trở nên vô nghĩa. Có rất nhiều cách để khiến mô hình trở nên phi tuyến như sử dụng các hàm kích hoạt sigmoid, tanh….

Nhưng hàm $f(x) = \max(0, x)$. dễ cài đặt, tính toán nhanh và vẫn hiệu quả.

Classification Layers: Sau khi phát hiện kiến trúc của CNNs chuyển sang quá trình phân loại.

FC-Fully connected : lớp này tương tự với lớp trong mạng neural truyền thẳng, các giá trị ảnh được liên kết đầy đủ vào node trong lớp tiếp theo. Sau khi ảnh được xử lý và rút trích đặc trưng từ lớp trước đó, dữ liệu ảnh sẽ không còn quá lớn so với mô hình truyền thẳng nên ta có thể sử dụng mô hình truyền thẳng để tiến hành nhận dạng. Tóm lại, lớp fully-connected đóng vai trò như một mô hình phân lớp và tiến hành dựa trên dữ liệu đã được xử lý ở các lớp trước đó.

2.1.3.3 Phương thức hoạt động của mạng neural tích chập

Một mạng neural tích chập được hình thành bằng cách ghép các lớp trình bày trên lại với nhau. Mô hình bắt đầu với lớp convolution. Lớp ReLU thường luôn được cài đặt ngay sau lớp Convolution hay Pooling tùy theo kiến trúc mà ta muốn xây dựng. Cuối cùng sẽ là lớp fully-connected để tiến hành phân lớp.

2.1.3.4 Ứng dụng của mạng neural tích chập (CNNs)

Mô hình mạng neural tích chập được áp dụng khá nhiều trong các bài toán nhận dạng như nhận dạng vật thể trong ảnh, nhận dạng chữ viết tay, chữ số viết tay, nhận dạng vật thể 3D, xử lý tiếng nói, xử lý ngôn ngữ tự nhiên …

Đã có rất nhiều các bài báo khoa học đề xuất ra nhiều kiến trúc mạng neural tích chập khác nhau cho các bài toán khác nhau như : LeNet, VGGNet, GoogleNet…. Tuy nhiên, sự phát triển của mạng neural tích chập đến một

ngưỡng nào đó bị chậm dần do những hạn chế của mô hình. Kích thước và độ sâu của mạng neural khiến mạng neural trở nên không linh hoạt. Giả sử ta đã huấn luyện thành công mô hình nhận diện 10 đối tượng khác nhau, một nhu cầu nhận diện đối tượng thứ 11 nảy sinh và để có thể nhận diện đối tượng thứ 11 này phải ta phải xây dựng một kiến trúc khác và huấn luyện lại từ đầu. Dù vậy, trong thực tế ta có thể chọn giải quyết các bài toán cụ thể mà nhu cầu mở rộng không quá lớn và không quá đòi hỏi độ linh hoạt cao. Ví dụ nhận diện các nhân viên cấp cao có quyền truy cập các hồ sơ quan trọng trong công ty, số lượng nhân viên cấp cao, không quá nhiều và cũng không phải là thường xuyên thay đổi, hay như bài toán nhận dạng chữ số (từ 0 đến 9) và không thay đổi theo thời gian. Với những bài toán như vậy thì mạng neural tích chập vẫn là một mô hình hiệu quả.

2.2 Giới thiệu mô hình hóa bài toán

Từ những cơ sở lý thuyết nhận dạng và mạng neural tích chập được trình bày ở trên, dưới đây em sẽ áp dụng nó vào việc xây dựng chương trình giải quyết bài toán phân lớp. /chương trình mà em xây dựng sẽ có nhiệm vụ nhận diện và phân biệt ảnh vào một nhóm mà do người dùng tự thiết lập tập dataset.

Input : Dữ liệu được đem huấn luyện là bộ dữ liệu người dùng tự tạo thông qua thiết bị sensor : Camera. Bộ dữ liệu được lấy làm 4 lớp, mỗi lớp có khoảng 30 ảnh.

Output : Model huấn luyện dùng để phân lớp ảnh (được lưu dưới dạng .json, .h5). Chương trình demo nhận dạng phân lớp xây dựng trên laptop cá nhân.

2.3 Thiết kế dữ liệu luyện mạng

Dữ liệu dùng để huấn luyện là bộ ảnh được chụp từ camera của laptop cá nhân. Bộ dữ liệu được chụp theo 4 lớp, mỗi lớp sẽ có 30 ảnh, mỗi ảnh có kích thước 224x224x3 pixel (ảnh rgb), tổng kích thước là 150528 pixel. Mỗi pixel có giá trị khác nhau đại diện cho độ sáng tối, màu sắc, giá trị nằm trong khoảng 0 đến 255.

2.4 Phương pháp huấn luyện mạng, và xây dựng mô hình

Các mô hình CNN : VGG16, VGG19, ResNET … tuy có độ chính xác cao, nhưng chúng đều có một điểm hạn chế chung đó là không phù hợp với các ứng dụng trên mobile hay các hệ thống nhúng có khả năng tính toán thấp. Nếu muốn deploy các mô hình trên cho các ứng dụng real time, ta cần phải có cấu hình cực kì mạnh mẽ (GPU / TPU) còn đối với các hệ thống nhúng (Raspberry Pi, Nano pc, etc) hay các ứng dụng chạy trên smart phone, ta cần có một mô hình "nhẹ" hơn, vì vậy em lựa chọn mô hình MobileNetV2 có độ chính xác không hề thua kém các mô hình khác như VGG16, VGG19 trong khi lượng parameters chỉ vỏn vẹn 3.5M (khoảng 1/40 số tham số của VGG16).

Yếu tố chính giúp MobileNetV2 có được độ chính xác cao trong khi thời gian tính toán thấp nằm ở sự cải tiến Conv layer bình thường. Trong MobileNet có 2 Covn layer được sử dụng là: SeparableConv và DepthwiseConv. Thay vì thực hiện phép tích chập như thông thường, SeparableConv sẽ tiến hành phép tích chập depthwise spatial, sau đó là phép tích chập pointwise. Còn DepthwiseConv sẽ chỉ thực hiện phép tích chập depthwise spatial (không tính pointwise convolution). Việc chia phép tích chập ra như vậy giúp giảm đáng kể khối lượng tinh toán và số lượng tham số của mạng. Với sự thay đổi này, MobileNetV2 có thể hoạt động một cách mượt mà ngay cả trên phần cứng cấu hình thấp.

Em sử dụng pretrainmodel MobilenetV2 để huấn luyện mạng. Với cấu trúc model như sau :

```
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
input (InputLayer)           (None, 96, 96, 3)         0
_________________________________________________________________
mobilenetv2_1.00_224 (Model) multiple                  2257984
_________________________________________________________________
global_average_pooling2d_6 ( (None, 1280)              0
_________________________________________________________________
dense_6 (Dense)              (None, 512)               655872
_________________________________________________________________
batch_normalization_6 (Batch (None, 512)               2048
_________________________________________________________________
re_lu_1 (ReLU)               (None, 512)               0
_________________________________________________________________
dense_7 (Dense)              (None, 4)                 2052
_________________________________________________________________
activation_1 (Activation)    (None, 4)                 0
=================================================================
Total params: 2,917,956
Trainable params: 2,882,820
Non-trainable params: 35,136
```

CHƯƠNG 3 CÀI ĐẶT CHƯƠNG TRÌNH

3.1 Môi trường cài đặt thuật toán và các vấn đề liên quan

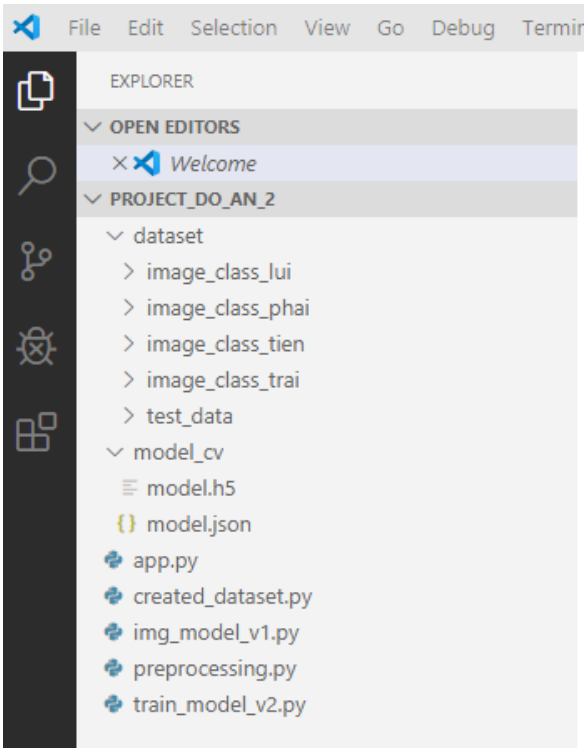
Chương trình được cài đặt trên ngôn ngữ python (phiên bản 3.7) và được thử nghiệm trên hệ điều hành Windows 10, 64 bit, máy tính laptop lenovo ThinkPad T450s, sử dụng chip Intel® Core™ i7-5600U CPU @ 2.60GHz 2.59GHz, RAM 8.00 GB.

Dữ liệu được thu thập từ sensor : camera của laptop Lenovo ThinkPad T450s.

IDE sử dụng : Visual studio code. Các thư viện Python sử dụng : tensorflow, pandas, numpy, matplotlib, cv2, keras.

3.2 Xây dựng chương trình

Chương trình được xây dựng với cấu trúc như sau :



Hình 3 0.1 Cấu trúc folder chương trình

Trong đó chứ năng chính của các class như sau :

created_dataset.py : Đây là class có chức năng tạo tập dữ liệu huấn luyện và sau khi thu thập ảnh thông qua camera sẽ được lưu vào 4 folder tương ứng với 4 nhãn trong quá trình huấn luyện. 4 folder tương ứng là :

- ✓ dataset/image_class_lui
- ✓ dataset/image_class_tien
- ✓ dataset/image_class_trai
- ✓ dataset/image_class_phai

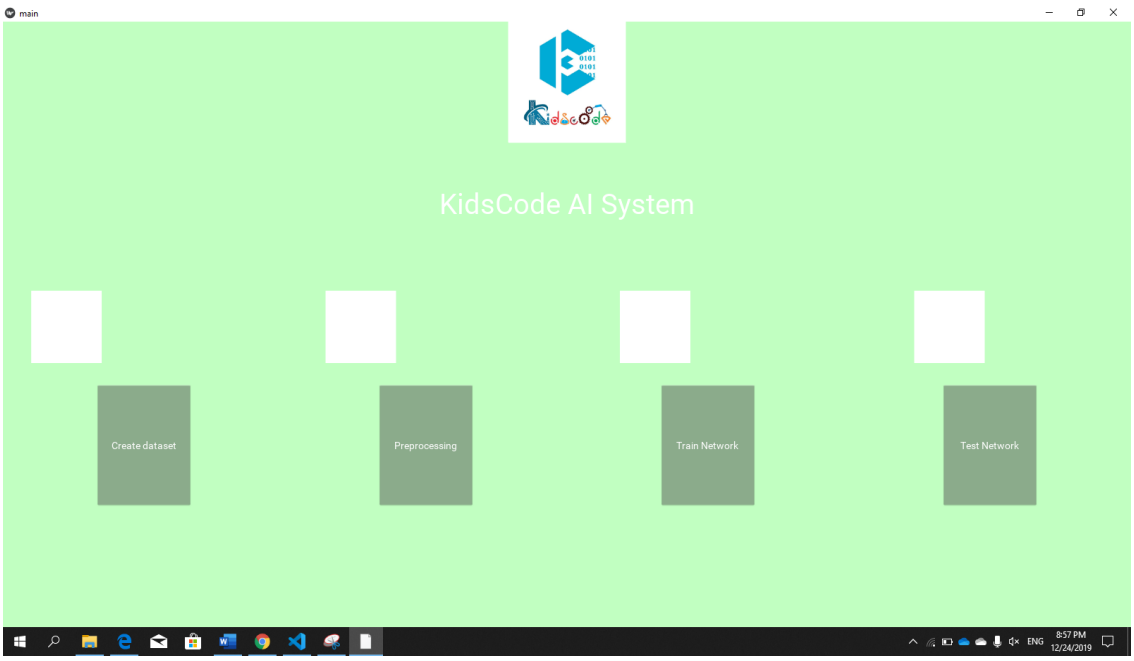
preprocessing.py : Đây là class có chức năng tiền xử lý ảnh đầu vào trước khi đem qua mô hình huấn luyện.

train_model_v2.py : Đây là class có trức năng huấn luyện mạng và model sau khi huấn luyện sẽ được lưu vào folder model_cv : model.h5, model.json.

img_model_v1.py : Đây là class sử dụng model để phân lớp ảnh đầu ra của class sẽ là giá trị (1, 2, 3, 4) tương ứng với 4 nhãn được huấn luyện.

app.py : Một số demo giao diện cơ bản của chương trình như sau :

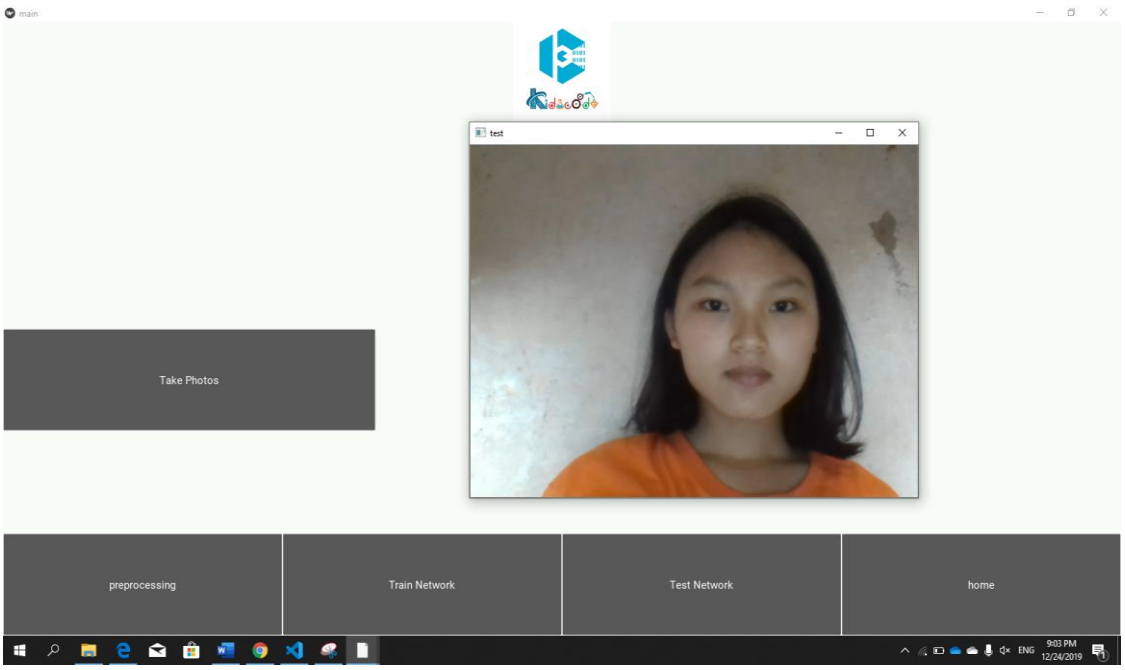
Giao diện trang chủ của chương trình :



Hình 3 0.2 Giao diện trang chủ của chương trình

Tương ứng với 4 class nêu ở trên : created_dataset.py, preprocessing.py, train_model_v2.py, img_model_v1.py

Giao diện của created_dataset :



Hình 3 0.3 Giao diện Created_dataset của chương trình

3.3 Kết quả nhận dạng, phân lớp

Vì đầu vào dữ liệu huấn luyện do chính người dùng chụp nên quá trình huấn luyện và đánh giá thực nghiệm thu được kết quả tương đối tốt dưới đây là kết quả huấn luyện :

```
20/20 [==============================] - 0s 1ms/step
Test loss: 0.019753485918045044
Test accuracy: 1.0
20/20 [==============================] - 0s 1ms/step
Test loss: 0.011278676800429821
Test accuracy: 1.0
20/20 [==============================] - 1s 29ms/step
acc: 100.00%
```

3.4 Định hướng phát triển trong tương lai

Trong đồ án II, có khá nhiều khuyết điểm, trong tương lai em sẽ cố gắng biến nó thành một chương trình có ứng dụng thực tế chứ không dừng lại ở mức độ nghiên cứu.

Hướng phát triển :

- ✓ Xây dựng được giao diện đẹp hơn, dễ dàng sử dụng, tích hợp phát triển ngoài nhận dạng phân lớp ảnh, em sẽ cố gắng nghiên cứu về nhận dạng âm thanh, giọng nói cho chương trình.
- ✓ Tích hợp chương trình lên hệ thống nhúng raspberry pi dùng để điều khiển hoạt động của raspberry pi tự động.

DANH SÁCH HÌNH VẼ

INDEX

TÀI LIỆU THAM KHẢO

[1] D. Geer, Trends in Cyber Security and Embedded Systems, November 2013.

[2] T. Cox, Raspberry Pi Cookbook.

[3] dexterind, Raspbian For Robots by Dexter Industries, 2014-01-15.

[4] "Raspberry Pi Foundation-About us," [Online]. Available: https://www.raspberrypi.org/about/.

[5] Peter, "Price," 03-06-2011.

[6] "Embedded Systems Week (ESWEEK)," 13 - 18 OCTOBER 2019 | NEW YORK CITY, NY. [Online].

[7] W. W. (. Bledsoe, "“The Model Method in Facial Recognition”, Technical Report PRI 15. Panoramic Research, Inc., Palo Alto, California.".

[8] Prabhu, "Understanding of Convolutional Neural Network (CNN) — Deep Learning," 4 Mar 2018. [Online]. Available: https://medium.com/@RaghavPrabhu/understanding-of-convolutional-neural-network-cnn-deep-learning-99760835f148.

[9] A. C. M. a. S. Guido, Introduction to Machine Learning with Python, FILL IN PRODUCTION EDI-, June 2016.

[10] Keras-team, "Keras Documentation," [Online]. Available: https://keras.io/layers/convolutional/.

[11] P. Punyawiwat, "Interns Explain CNN," 8 Feb 2018. [Online]. Available: https://blog.datawow.io/interns-explain-cnn-8a669d053f8b.