BỘ GIÁO DỤC VÀ ĐÀO TẠO

ĐẠI HỌC QUỐC GIA THÀNH PHỐ HỒ CHÍ MINH
TRƯỜNG ĐẠI HỌC CÔNG NGHỆ THÔNG TIN
KHOA KHOA HỌC MÁY TÍNH
ĐỒ ÁN MÔN HỌC

CS331 – THỊ GIÁC MÁY TÍNH NÂNG CAO

ĐỀ TÀI

PHÂN LOẠI CẢM XÚC KHUÔN MẶT NGƯỜI

Giảng viên hướng dẫn:	TS. Mai Tiến Dũng
Năm học:	2022-2023
Học kỳ:	1
Nhóm sinh viên thực hiện:	Lê Quang Hùng
	Tôn Anh Trúc
	Nguyễn Hoàng Hải
Mã số sinh viên:	20521363
	20520944
	20521279
Email:	20521363@gm.uit.edu.vn
	20520944@gm.uit.edu.vn
	20521279@gm.uit.edu.vn
Lớp:	CS331.N12.KHCL

THÔNG TIN CHUNG

Đề tài:	Phân loại cảm xúc.
Môn học:	CS331 – Thị giác máy tính nâng cao.
Lớp:	CS331.N12.KHCL.
Giảng viên hướng dẫn:	TS. Mai Tiến Dũng.
Thời gian thực hiện:	Học kỳ 1. Năm học: 2022 – 2023.
Sinh viên thực hiện:	Lê Quang Hùng – 20521363. Tôn Anh Trúc – 20520944. Nguyễn Hoàng Hải – 20521279.
Nội dung đề tài:	Phân loại cảm xúc. Hay nói cách khác nhóm sẽ xác định cảm xúc của từng người trong hình với các loại cảm xúc: Angry, Disgust, Fear, Happy, Sad, Surprise, Neutral. Đưa ra kết quả thử nghiệm từ bộ dữ liệu mà nhóm đã thu thập. So sánh độ chính xác qua các phương pháp thực hiện khác nhau.
Kế hoạch thực hiện:	Tuần 1, 2: Thành lập nhóm. Chọn đề tài. Xây dựng kế hoạch thực hiện và phân công nhiệm vụ. Tuần 3, 4, 5, 6, 7: Khảo sát tình hình thực tế. Thu thập các thông tin liên quan và tìm nguồn tài liệu tham khảo. Tìm hiểu các kiến thức cần thiết cho quá trình thực hiện đề tài. Tuần 8, 9, 10, 11, 12: Tìm hiểu các công nghệ và công cụ có thể sử dụng trong đề tài. Thiết kế, xây dựng và viết chương trình cài đặt. Đánh giá và kiểm thử chương trình cài đặt. Tuần 13, 14: Chuẩn bị slide trình bày và tập duyệt báo cáo.

	Tuần 15: Báo cáo đồ án. Tuần Dự trữ: Chỉnh sửa báo cáo và chương trình cài đặt sau khi báo cáo.

LỜI CẢM ƠN

Nhóm xin chân thành gửi lời cảm ơn đến TS. Mai Tiến Dũng – Giảng viên khoa Khoa học máy tính, Trường Đại học Công nghệ thông tin, Đại học Quốc gia thành phố Hồ Chí Minh, đồng thời là giảng viên giảng dạy lớp CS331.N12.KHCL – Môn Thị giác máy tính nâng cao, trong thời gian qua đã tận tình hướng dẫn và định hướng cho nhóm trong suốt quá trình thực hiện và hoàn thành đồ án.

Trong quá trình thực hiện đồ án nhóm đã cố gắng rất nhiều để hoàn thành đồ án một cách tốt nhất và hoàn thiện nhất, song cũng sẽ không tránh khỏi được những sai sót ngoài ý muốn. Nhóm mong rằng sẽ nhận được những lời nhận xét và những lời góp ý chân thành từ quý thầy/cô và các bạn trong quá trình thực hiện chương trình của nhóm để chương trình ngày càng hoàn thiện hơn. Mọi thắc mắc cũng như mọi góp ý của mọi người xin gửi mail về một trong các địa chỉ email sau: 20520944@gm.uit.edu.vn (Tôn Anh Trúc), 20521279@gm.uit.edu.vn (Nguyễn Hoàng Hải), 20521363@gm.uit.edu.vn (Lê Quang Hùng). Mỗi ý kiến đóng góp sẽ là một nguồn động lực to lớn đối với nhóm để nhóm có thể cố gắng cải tiến chương trình ngày càng hoàn thiện và phát triển đồ án lên một mức cao hơn, nhóm cũng sẽ dựa vào đó để phát triển hơn những ưu điểm và cải thiện được phần nào đó những nhược điểm của chương trình. Hy vọng đề tài "Phân loại cảm xúc" do nhóm thực hiện sẽ trở thành một công cụ hữu ích và có thể ứng dụng được trong lĩnh vực Thị giác máy tính và cũng như ở đời thực.

Thành phố Hồ Chí Minh, tháng 2 năm 2023

Nhóm sinh viên thực hiện

Tôn Anh Trúc, Nguyễn Hoàng Hải, Lê Quang Hùng

Mục lục

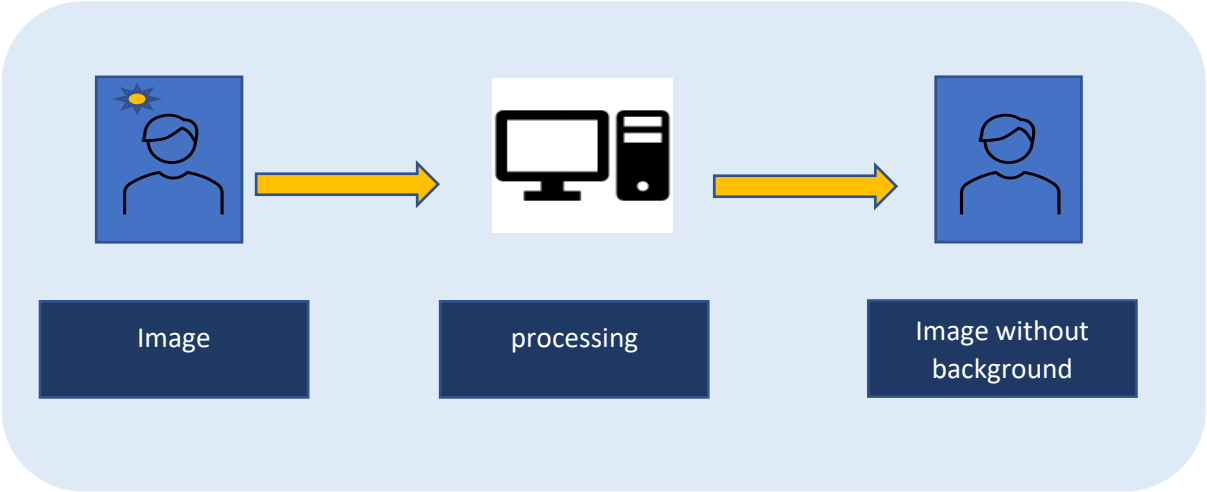
Chương 1. TỔNG QUAN

1. Giới thiệu

Khái niệm thị giác máy tính

Thị giác máy tính (Computer Vision) là một hình thức công nghệ mô tả khả năng của bộ máy có thể thu nhận và phân tích các dữ liệu trực quan, sau đó sẽ tiến hành đưa ra các quyết định về nó. Đơn giản thì thị giác máy tính là một công nghệ thuộc lĩnh vực khoa học máy tính và trí tuệ nhân tạo mà có tầm nhìn và khả năng xử lý nhận dạng như con người.

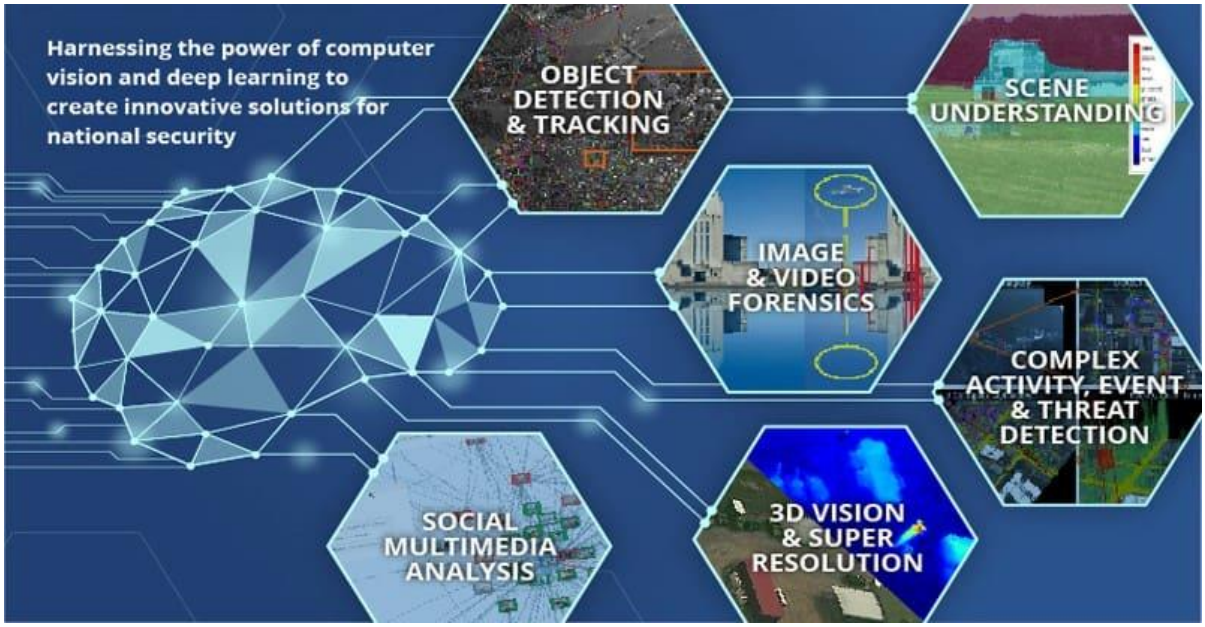
Mục đích của những người nghiên cứu về lĩnh vực này là tận dụng những kiến thức về thị giác của con người để phát triển những công cụ và kỹ thuật phù hợp vào hệ thống máy tính sao cho máy tính có thể giải quyết những vấn đề thực tế được đặt ra có liên qua đến thị giác máy tính. Tất cả có thể được thực hiện trên quy mô trên ảnh hay có thể là trên video.



Mô phỏng quá trình áp dụng thị giác máy tính trong việc xóa background của hình ảnh (remove background)

Các lĩnh vực con của thị giác máy tính bao gồm tái cấu trúc cảnh, dò tìm sự kiện, theo dõi video, nhận diện bố cục đối tượng, học, chỉ mục, đánh giá chuyển động và phục hồi ảnh.

Công nghệ này đề cập đến toàn bộ quá trình mô phỏng tầm nhìn trong một bộ máy phi sinh học của con người. Quá trình này bao gồm chụp ảnh, phát hiện và nhận dạng đối tượng, nhận biết bối cảnh tạm thời và sau đó nâng cấp sự hiểu biết về những sự kiện đã xảy ra trong một khoảng thời gian hợp lý.



Một số ứng dụng của thị giác máy tính hiện nay

Các cơ sở khoa học của thị giác máy tính

Thị giác: Là khả năng nhận và diễn giải thông tin từ ánh sáng đi vào mắt. Việc tri giác này còn được gọi là thị lực, sự nhìn. Những bộ phận khác nhau cấu thành thị giác được xem như là một tổng thể như là hệ thị giác và được tập trung nghiên cứu trong nhiều lĩnh vực khác nhau như tâm lý, khoa học nhận thức, khoa học thần kinh và sinh học phân tử.

Vấn đề chính của thị giác: Là những gì mà con người thấy được không phải chỉ là sự biến đổi của các kích thích ở võng mạc (tức là ảnh trên võng mạc). Vì vậy, những người có quan tâm đến dạng tri giác này đã cố gắng giải thích cách làm việc của tiếng trình xử lý hình ảnh để tạo ra cái mà chúng ta thực sự nhìn thấy.

Các nghiên cứu trước đây về thị giác: Có hai trường phái chính thời Hy Lạp cổ đưa ra các giải thích đầu tiên về cách hoạt động của việc nhìn trong cơ thể người.

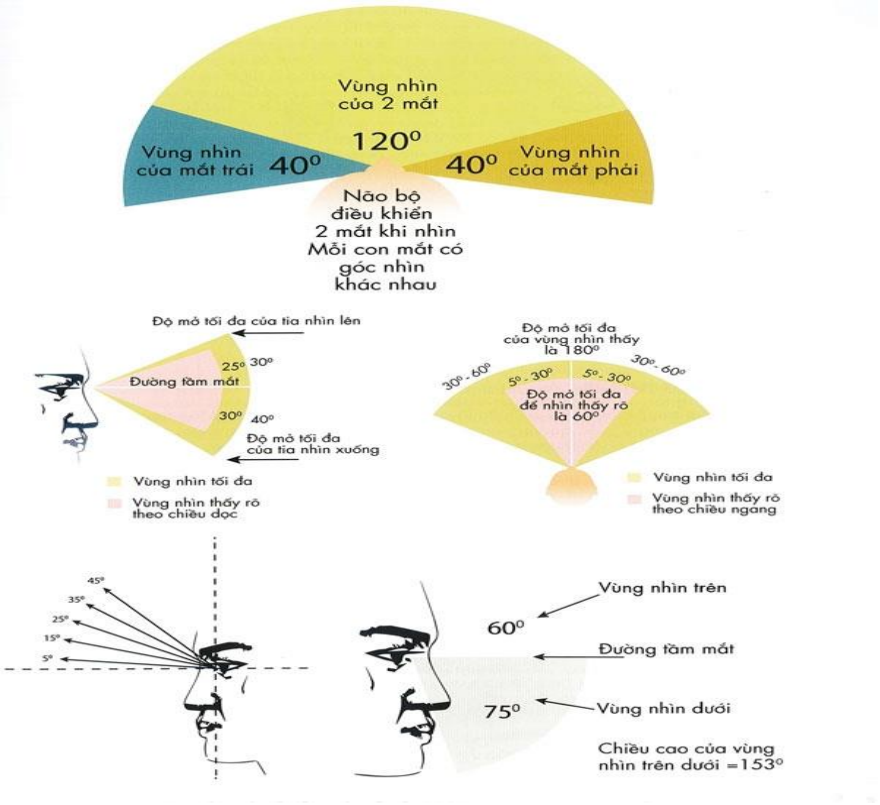
- Trường phái thứ nhất là "lý thuyết phát xạ": sự nhìn diễn ra khi các tia nhìn phát ra từ mắt bị chặn bởi các vật được nhìn thấy. Nếu nhìn một vật thì đó là do các tia nhìn đi từ mắt rơi vào vật đó. Tuy nhiên, một ảnh khúc xạ cũng được nhìn thấy bởi các tia nhìn. Lúc này, tia nhìn đi từ mắt, xuyên qua không khí và tơi vào vật được nhìn thấy sau khi bị khúc xạ như là kết quả của sự chuyển động của các tia nhìn phát ra từ mắt.
- Trường phái thứ hai chủ trường về cách tiếp cận gọi là "sự dựa vào", xem sự nhìn như là có một cái gì đó đại diện cho vật đi vào mắt. Với sự ủng hộ của các nhà truyền bá chính và những người kế tục, lý thuyết này dường như đã đạt đến một ít tri giác về bản chất của sự nhìn, nhưng ánh sáng chưa đóng vai trò gì trong lý thuyết này và nó chỉ là sự suy đoán mà không có các cơ sở thực nghiệm.

Suy luận vô thức:

- Nghiên cứu đầu tiên của Hermann von Helmholtz về thị giác trong thời hiện đại, ông khảo sát mắt người và kết luận rằng về mặt quan học thì mắt người khá kém cỏi. Theo ông thì tông tin thu được qua mắt kém chất lượng đến nổi khó mà thấy gì được. Vì thế ông kết luận rằng sự nhìn chỉ có thể là kết quả của một dạng suy luận vô thức: một vấn đề về việc thừa nhạn và kết luận từ những dữ liệu không đầy đủ, dựa trên những kinh nghiệm đã có.
- Sự suy luận này cần có những kinh nghiệm về thế giới. Các ví dụ đã được biết đến là:
  + Ánh sáng đến từ phía trên
  + Các vật thường không được nhìn từ phía dưới lên
  + Các khuôn mặt được nhìn thấy (và nhận ra) từ phía bên phải
- Nghiên cứu về ảo giác (trường hợp mà quá trình suy luận là sai) đã đem lại những hiểu biết sâu hơn về các dạng suy luận mà hệ thị giác thực hiện.

Lý thuyết Gestalt:

- Các nhà tâm lý học cấu trúc hình thức (Gestalt psychologists) đã đưa ra nhiều câu hỏi nghiên cứu mà các nhà khoa học về thị giác thời nay đang nghiên cứu.
- Các định luật cấu trúc hình thức về sự tổ chức đã dẫn đến nghiên cứu về cách thức mà con người nhận thức các thành phần thị giác như là các mẫu thức hoặc các toàn thể được tổ chức thay cho nhiều phần riêng biệt khác nhau. Theo thuyết này thì có 6 yếu tố chính để xác định xem chúng ta ghép nhóm các đồ vật theo sự nhìn: sự gần gũi về không gian, sự tương tự, sự đóng kín, sự đối xứng, sự quen thuộc và sự liên tục.



Các vấn đề liên quan đến thị giác

Các lĩnh vực của thị giác máy tính

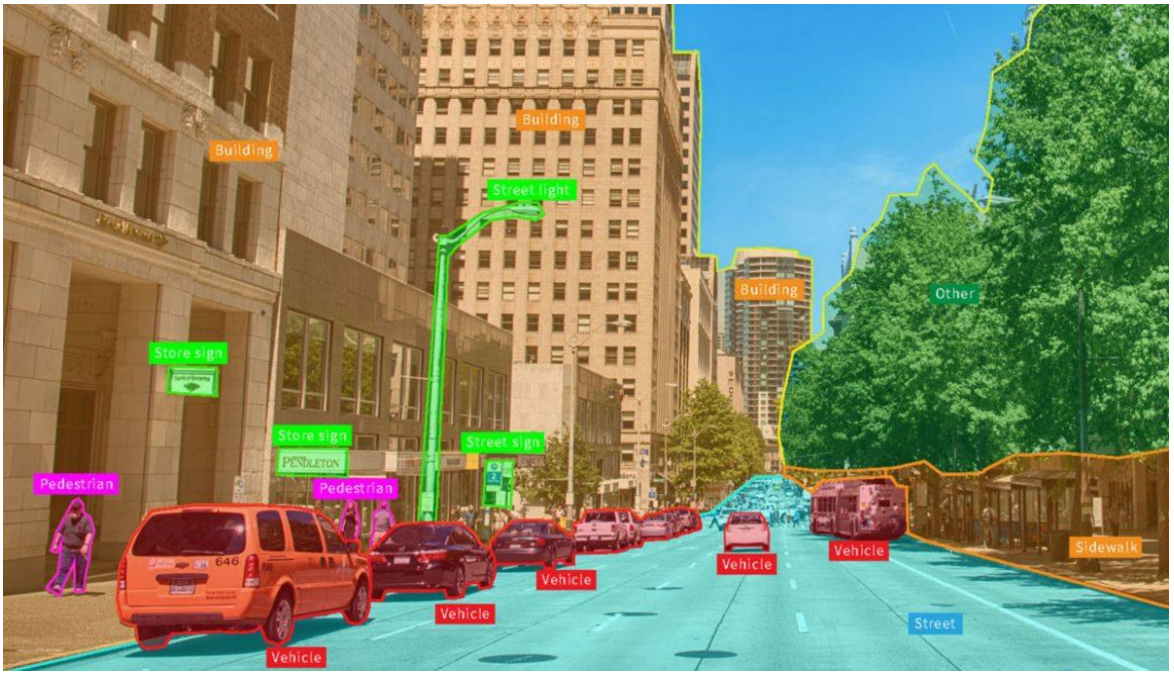
Xử lý hình ảnh:

- Đây là một trong những mảng quan trọng nhất trong kỹ thuật thị giác máy tính, làm tiền đề cho nhiều nghiên cứu sau này. Nó là một lĩnh vực mang tính khoa học và công nghệ. Xử lý ảnh là một ngành khoa học mới mẽ so với nhiều ngành khoa học khác nhưng tốc độ phát triển của nó rất mạnh mẽ, kích thích các trung tâm nghiên cứu.
- Hai nhiệm vụ cơ bản của xử lý ảnh là nâng cao chất lượng thông tin hình ảnh và xử lý số liệu cung cấp cho các quá trình khác trong đó có việc ứng dụng thị giác vào điều khiển. Xử lý ảnh trước đây chủ yếu được sử dụng làm nâng cao chất lượng ảnh (gia tăng chất lượng ảnh quang học trong mắt người quan sát). Thời gian gần đây, phạm vi ứng dụng xử lý ảnh mở rộng không ngừng, có thể nói hiện không có lĩnh vực khoa học nào không sử dụng các thành tựu của công nghệ xử lí ảnh kĩ thuật số.

Xử lý hình ảnh (Image Processing)

Nhận diện mẫu: Giải thích các kỹ thuật khác nhau để phân loại mẫu.

Quang trắc: Liên quan đến việc thu thập các số đo chính xác từ hình ảnh.



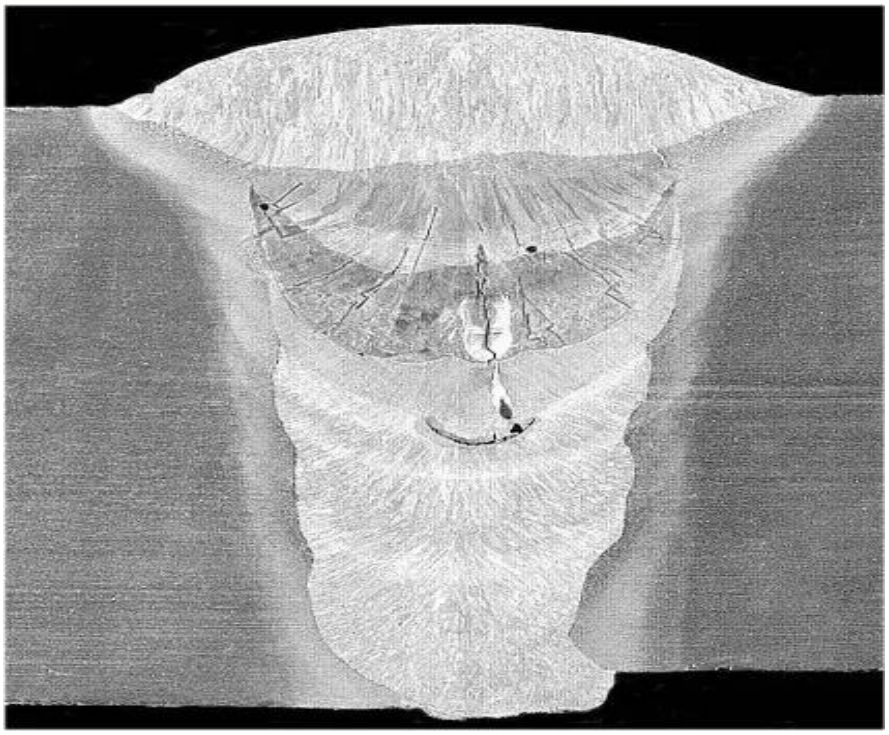
Một ví dụ trong lĩnh vực nhận diện mẫu (Pattern Recognition)

Ứng dụng của thị giác máy tính

Hiện nay công nghệ thị giác máy tính có rất nhiều ứng dụng vào trong đời sống thường ngày của chúng ta, chẳng hạn như máy chơi game có thể nhận ra cử chỉ của con người hoặc camera ở điện thoại di động có thể tự động tập trung vào các đối tượng chính ở trong bối cảnh mà chúng ta cần chụp hoặc là cần quay video. Nó đang tác động đến nhiều lĩnh vực trong cuộc sống của chúng ta.

Một số ứng dụng cơ bản nhất của thị giác máy tính hiện nay: Phát hiện các khiếm khuyết, trình học tự động, vận hành tự động, xử lí dữ liệu, lĩnh vực y tế, nhận diện khuôn mặt, ngân hàng, bán lẻ.

- Phát hiện các khiếm khuyết: Ứng dụng vào việc kiểm tra vết nứt kim loại, lỗi sơn, bản in xấu,…
  Với thị giác máy tính thì chúng ta có thể kiểm tra tất cả các lỗi nhỏ nhất có kích thước nhỏ hơn 0,05mm.



Kiểm tra vết nứt trong mối hàn

- Trình đọc tự động: Một trong những ứng dụng nỗi bật hiện nay là ứng dụng vào việc tự động dịch trong ứng dụng Google translate khi camera điện thoại được trỏ vào vùng văn bản của bất kỳ ngôn ngữ nào.
  Sử dụng thuật toán nhận dạng ký tự (OCR) để trích xuất thông tin, cụ thể là nhận dạng ký tự quang học, cho phép một bản dịch chính xác sau đó chuyển thành lớp phủ lên văn bản thực.

Dịch trực tiếp bằng hình ảnh

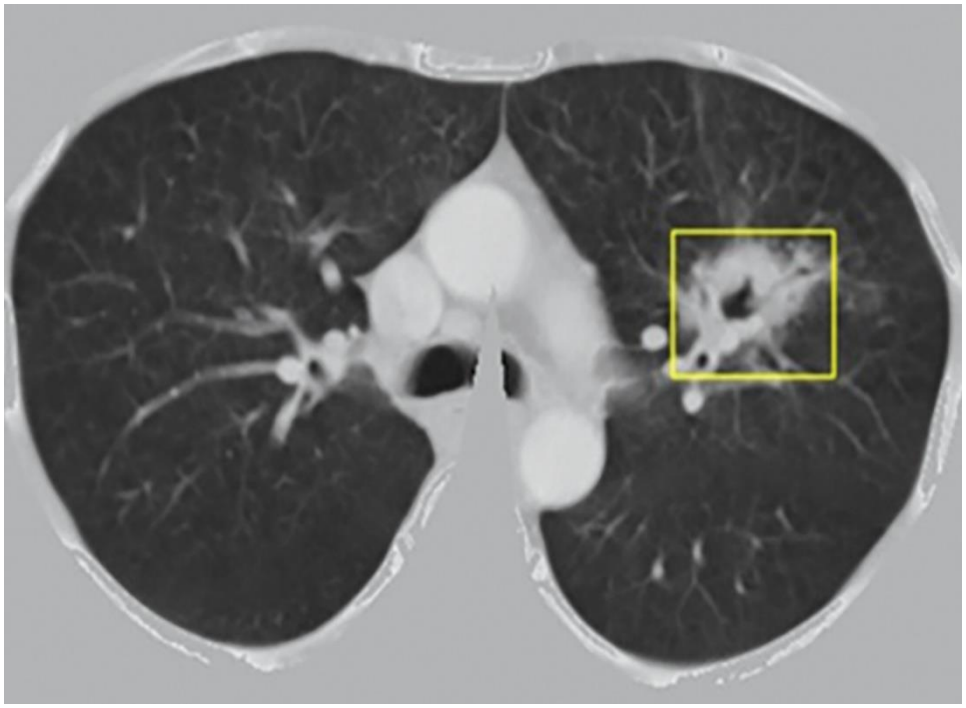
- Vận hành tự động: Xe không người lái, máy bay không người lái, robot,… Trong lĩnh vực này phụ thuộc rất nhiều vào Computer vision và Deep learning.
  Cụ thể như trong xe tự hành: Công nghệ AI phân tích dữ liệu thu thập được từ hàng triệu người lái xe, học hỏi từ hành vi lái xe để tự động tìm làn đường, ước tính độ cong đường, phát hiện các mốt nguy hiểm và giải thích các tín hiệu và tín hiệu giao thông.

Xe tự hành

- Xử lí dữ liệu: Nhận dạng và tổ chức thông tin.
  Các công cụ Computer vision và mô hình Deep learning đã được đưa vào nghiên cứu, đòi hỏi khối lượng dữ liệu lớn được gán nhãn. Khi các thuật toán Deep learning phát triển, chúng chủ yếu thay thế quy trình gắn thẻ thủ công thông qua một phương pháp tiếp cận được gọi là nghiên cứu dữ liệu đán đông - thu thập theo thời gian thực tự động và gắn thẻ dữ liệu do các chuyên gia tạo ra và từ đó máy học sẽ bắt đầu quy trình nhận dạng các đối tượng.

Hệ thống thông tin

- Lĩnh vực y tế: Nhận dạng mẫu và xử lý hình ảnh.
  Hình ảnh y khoa đã trở thành một phần thiết yếu trong cách thức làm việc của các chuyên gia trong lĩnh vực y tế, hướng đến các công cụ chẩn đoán tốt hơn và tăng đáng kể khả năng đưa ra các hành động hiệu quả hơn.



Chẩn đoán tế bào ung thư

- Nhận diện khuôn mặt:
  Khái niệm này xuất hiện từ năm 2011 khi Google chứng minh rằng có thể tạo ra một máy dò tìm khuôn mặt chỉ bằng những hình ảnh không được gắn nhãn.

Họ đã thiết kế một hệ thống có thể tự học để phát hiện hình ảnh con mèo mà không cần giải thích với hệ thống là con mèo trông như thế nào.
Ngày nay, điện thoại thông minh có thể sử dụng máy ảnh chất lượng cao để nhận dạng. Thị giác máy tính đang được sử dụng trong lĩnh vực an ninh để tìm kiếm tội phạm, dự đoán sự di chuyển khẩn cấp của đám đông,…

Nhận diện khuôn mặt

- Ngân hàng: Các ứng dụng nhận dạng hình ảnh sử dụng học máy để phân loại và trích xuất dữ liệu phục vụ cho việc giám sát quá trình xác thực các tài liệu như thẻ căn cước hoặc giấy phép lái xe có thể được sử dụng để cải thiện trải nghiệm của khách hàng từ xa và tăng cường bảo mật.

Nhận diện khách hàng thông qua việc quét hình ảnh

- Bán lẻ: Thị giác máy tính đang được sử dụng trong các cửa hàng ngày càng nhiều, đặc biệt là giúp cải thiện trải nghiệm của khách hàng, Pinterest Lens là một công cụ tìm kiếm sử dụng thị giác máy tính để phát hiện các đối tượng. Bằng cách sử dụng ứng dụng điện thoại thông minh trong các cửa hàng, chúng ta có thể hình dung một sản phẩm trông như thế nào và nhận được các sản phẩm khác liên quan đến nó.

Quét mã QR của sản phẩm

Những hạn chế của thị giác máy tính

Các hệ thống thị giác máy tính hiện tại thực hiện tốt việc phân loại hình ảnh và bản địa hóa các đối tượng trong ảnh, khi chúng được đào tạo đầy đủ với các ví dụ. Nhưng ở phần cốt lõi của chúng, các thuật toán học sâu cung cấp sức mạnh cho các ứng dụng thị giác máy tính chính là việc đối chiếu các mẫu pixel. Chúng không hiểu những gì đang diễn ra trong các hình ảnh.

Con người có thể khai thác kiến thức rộng lớn về thế giới của mình để lấp đầy những lỗ hổng khi họ đối mặt với một tình huống mà họ chưa từng thấy trước đây. Không giống như con người, các thuật toán thị giác máy tính cần phải được hướng dẫn kỹ lưỡng về các loại đối tượng mà chúng phải phát hiện. Ngay khi môi trường của chúng chứa những thứ đi chệch khỏi các ví dụ đã được đào tạo, chúng bắt đầu hành động theo những cách phi lý, chẳng hạn như không phát hiện ra các phương tiện khẩn cấp dừng đỗ ở những vị trí khác thường.

Hiện tại, giải pháp duy nhất để giải quyết những vấn đề này là đào tạo các thuật toán AI trên với ngày càng nhiều các ví dụ, với hi vọng lượng dữ liệu bổ sung sẽ bao quát mọi tình huống mà AI sẽ gặp phải. Nhưng những kinh nghiệm cho thấy, nếu không có sự nhận thức theo tình huống, sẽ luôn có những góc khuất trong những tình huống hiếm hoi làm rối loạn thuật toán AI.

Mặc dù thị giác máy tính có những hạn chế nhất định nhưng dường như trí thông minh thị giác không dễ tách rời khỏi phần còn lại của trí thông minh, đặc biệt là kiến thức chung, sự trừu tượng và kỹ năng ngôn ngữ.

2. Sơ lược về đề tài

Xã hội thời nay ngày càng có xu hướng tự động hóa, mọi công việc đều có thể được đảm nhận bởi máy móc. Tuy nhiên vẫn còn những hạn chế mà máy móc hiện đại vẫn chưa thể nào phát huy tối đa tác dụng như con người, ví dụ như công việc giao tiếp, dịch vụ, đánh giá mức độ hài lòng của khách hàng hay nói rộng hơn là phân loại cảm xúc của con người.

Với sự giúp đỡ của thị giác máy tính, ta hoàn toàn có thể chẩn đoán được cảm xúc của người mà ta cần phân loại.

Ở những nơi công cộng, lượng người đông, để nhận biết bằng sức người cảm xúc chung của mọi người rất khó khăn hoặc những loại hình dịch vụ không nhất thiết phải giao tiếp người - người, hoặc để phân loại cảm xúc của các bức ảnh, tranh nghệ thuật, sử dụng thị giác máy tính để phân loại ta sẽ được kết quả nhanh hơn, hiệu quả hơn.

3. Phát biểu bài toán

Đầu vào (Input)

Đối tượng cần để nhận diện là con người. Đầu vào là một bức hình có thể là một người, một vài người hoặc là rất nhiều người.

Đầu ra (Output)

Hình ảnh khuôn mặt của những người mà model có thể nhận diện được ở trong input đồng thời dự đoán cảm xúc của người đó.

Ví dụ minh họa

Để minh họa một cách trực quan mô hình của bài toán, chúng tôi xin đưa ra một ví dụ như sau:

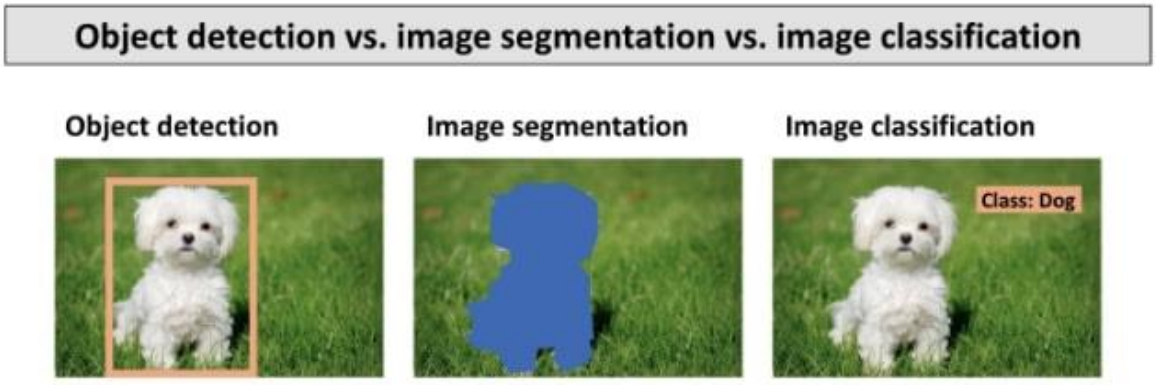
Input

output

4. Mô hình tổng quát

Classification là gì?

Phân loại là một trong ba cách phổ biến nhất để xử lý dữ liệu hình ảnh. Nếu tác vụ phát hiện đối tượng (Object detection) nhằm mục tiêu xác định vị trí vật thể bằng cách xây dựng hộp tọa độ (bounding box); phân đoạn ảnh (Image segmentation) cung cấp thông tin chi tiết hơn về kích thước và hình dạng vật thể thì phân loại ảnh (Image classification) giúp tìm ra câu trả lời: Vậy hình ảnh thuộc loại nào?



Sự khác biệt giữa phát hiện đối tượng, phân đoạn ảnh và phân loại ảnh (Nguồn ảnh: Levity)

Phân loại hình ảnh là gì?

Phân loại hình ảnh (Image classification) hay Nhận dạng hình ảnh (Image recognition) là một trong những tác vụ của thị giác máy tính, ở đó thuật toán xem xét và dán nhãn cho hình ảnh từ một tập danh mục được xác định và đào tạo trước.

Ví dụ, với một tập các hình ảnh, mỗi hình ảnh mô tả một con mèo hoặc một con chó, thuật toán sẽ "quan sát" toàn bộ dữ liệu và dựa trên hình dạng, màu sắc để hình thành giả thuyết liên quan đến nội dung của ảnh. Kết quả thu được là từ tập dữ liệu ban đầu, các hình ảnh chó/mèo đã được phân loại một cách tự động.

Thực tế, thị giác góp phần tạo nên 80-85% nhận thức của con người về thế giới. Hàng ngày, mỗi người phải thực hiện phân loại trên bất kỳ dữ liệu hình ảnh nào mà chúng ta bắt gặp.

Do đó, mô phỏng nhiệm vụ phân loại với sự trợ giúp của mạng nơ-ron là một trong những ứng dụng đầu tiên của thị giác máy tính mà các nhà nghiên cứu nghĩ đến.

Các kỹ thuật phân loại ảnh

Có nhiều thuật toán khác nhau được ứng dụng trong việc phân loại hình ảnh. Các thuật toán này được chia thành hai nhóm chính là Học có giám sát (supervised learning) và Học không giám sát (unsupervised learning).

Phân loại có giám sát

Trong học máy có giám sát, thuật toán được huấn luyện trên một tập hình ảnh đã được dán nhãn. Từ dữ liệu mẫu này, thuật toán có thể trích xuất thông tin, phục vụ phân loại ngay cả những hình ảnh chưa từng nhìn thấy trước đó.

Xuyên suốt quá trình đào tạo, đặc điểm của ma trận hình ảnh sẽ được trích xuất dưới dạng dữ liệu quan trọng để đưa vào xử lý. Các đặc điểm này đại diện cho hình ảnh trong không gian chiều thấp (lower-dimensional feature space) và là cơ sở để thuật toán tiến hành phân loại.

Trong quá trình đánh giá, các đặc điểm của ảnh thử nghiệm được thu thập và tái phân loại với sự hỗ trợ của mạng thần kinh nhân tạo. Hệ thống lúc này đã có thể nhận biết các đặc điểm điển hình của mọi lớp hình ảnh mà nó được đào tạo.

Các phương pháp phân loại phổ biến dựa trên học có giám sát bao gồm:

- Support Vector Machines
- Decision Trees
- K Nearest Neighbors

Các mạng nơ-ron thường được sử dụng để phân loại hình ảnh có giám sát bao gồm AlexNet, ResNet, DenseNet và Inception.

Đối với phân loại có giám sát, việc dán nhãn dữ liệu đóng vai trò quan trọng. Độ chính xác của dữ liệu được dán nhãn quyết định phần lớn hiệu suất của mô hình học

máy. Các thuật toán phân loại có giám sát có thể được chia thành hai mục nhỏ hơn dựa trên nhãn dữ liệu.

Phân loại nhãn đơn

Phân loại nhãn đơn (Single-label classification) là tác vụ phổ biến nhất trong phân loại ảnh có giám sát. Theo đó, mỗi hình ảnh được đại diện bởi một nhãn/chú thích (a single label or annotation). Mô hình xuất ra một giá trị hoặc dự đoán duy nhất cho mỗi hình ảnh mà nó xử lý.

Đầu ra từ mô hình là mã hóa One-hot (từng giá trị được biến đổi thành các đặc trưng nhị phân chỉ chứa giá trị 1 hoặc 0). Mã hóa One-hot có độ dài bằng số lớp và giá trị biểu thị xác suất hình ảnh thuộc về lớp này.

Hàm Softmax được sử dụng để đảm bảo các xác suất tổng bằng một và xác suất tối đa được chọn làm đầu ra của mô hình. Mặc dù Softmax không có giá trị về mặt dự đoán, nhưng nó giúp ràng buộc đầu ra giữa 1 và 0, nhờ vậy, có thể đánh giá độ tin cậy của mô hình từ điểm Softmax.

Một số ví dụ về bộ dữ liệu phân loại nhãn đơn bao gồm MNIST, SVHN, ImageNet, v.v.

Phân loại nhãn đơn có thể được xếp vào phân loại đa lớp (Multiclass classification) hoặc phân loại nhị phân (binary classification).

Phân loại đa nhãn

Phân loại đa nhãn là một tác vụ phân loại trong đó mỗi hình ảnh có thể chứa nhiều hơn một nhãn hoặc một số hình ảnh chứa đồng thời tất cả các nhãn.

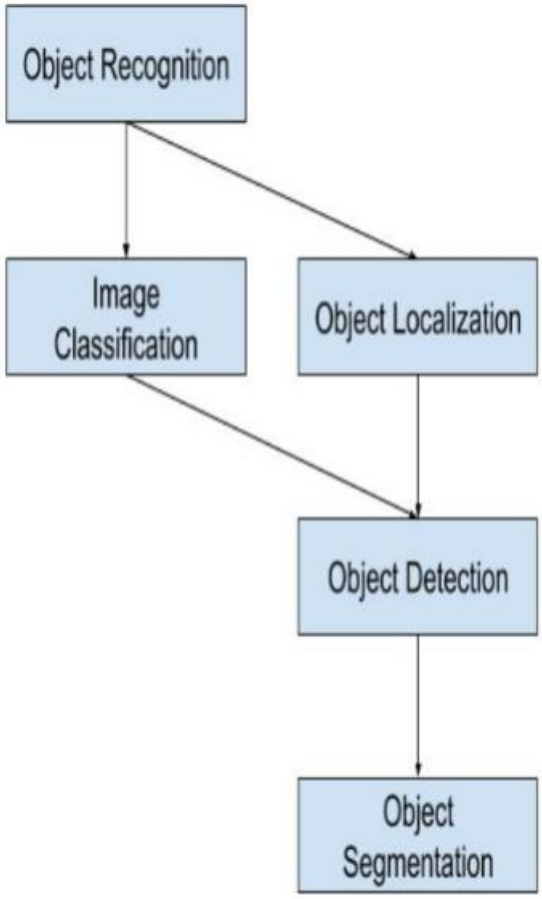
Phân loại đa nhãn xuất hiện phổ biến trong lĩnh vực xử lý hình ảnh y tế, khi một bệnh nhân có thể được chẩn đoán mắc nhiều bệnh dựa trên dữ liệu chụp X-quang.

Phân loại hình ảnh hoạt động như thế nào?

Máy tính xử lý một hình ảnh dưới dạng pixel. Theo đó, hình ảnh chỉ là một mảng ma trận, và kích thước của ma trận phụ thuộc vào độ phân giải hình ảnh.

Do đó, xử lý hình ảnh là tiến hành phân tích dữ liệu toán học với sự trợ giúp của các thuật toán. Các thuật toán này chia nhỏ hình ảnh thành một tập hợp các đặc điểm nổi bật, giúp giảm khối lượng công việc của bộ phân loại cuối cùng.

Quá trình trích xuất đặc điểm là bước quan trọng nhất trong việc phân loại hình ảnh. Phân loại, đặc biệt là phân loại có giám sát, phụ thuộc phần lớn vào dữ liệu được cung cấp cho thuật toán. Một bộ dữ liệu phân loại tốt phải đảm bảo các yêu cầu về sự cân bằng của dữ liệu, chất lượng của ảnh và chú giải kèm theo.



Sơ đồ các mối liên hệ giữa các tác vụ trong Computer Vision

5. Đối tượng nghiên cứu

Đối tượng nghiên cứu của đề tài là mô hình phân loại cảm xúc con người.

Đề tài sử dụng 4 hướng tiếp cận đó là sử dụng Support Vector Machine (SVM), Convolutional Neural Network (CNN) (tự build), VGG16, VGG19 nhằm mục đích so sánh hiệu quả của từng model khi sử dụng với bộ dữ liệu đã được train và với bộ dữ liệu nằm bên ngoài.

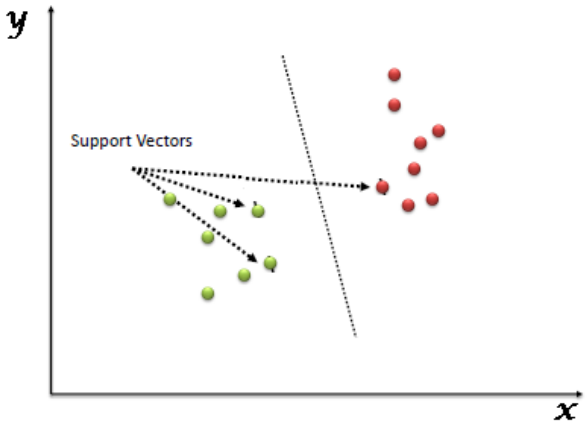
Chương 2. CƠ SỞ LÝ THUYẾT

1. Support Vector Machine (SVM)

Khái niệm

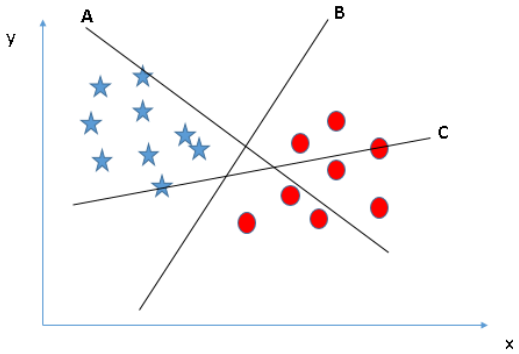
SVM là một thuật toán giám sát, nó có thể sử dụng cho cả việc phân loại hoặc đệ quy. Tuy nhiên nó được sử dụng chủ yếu cho việc phân loại. Trong thuật toán này, chúng ta vẽ đồ thị dữ liệu là các điểm trong n chiều (ở đây n là số lượng các tính năng có) với giá trị của mỗi tính năng sẽ là một phần liên kết. Sau đó chúng ta thực hiện tìm "đường bay" (hyper-plane) phân chia các lớp. Hyper-plane nó chỉ hiểu đơn giản là 1 đường thẳng có thể phân chia các lớp ra thành hai phần riêng biệt.

Support Vectors hiểu một cách đơn giản là các đối tượng trên đồ thị tọa độ quan sát, Support Vector Machine là một biên giới để chia hai lớp tốt nhất.



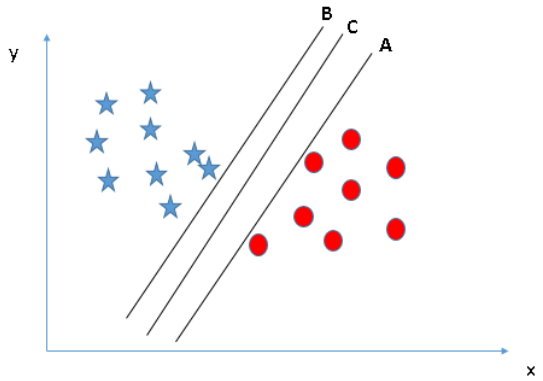
Xác định "Làm sao để vẽ-xác định đúng hyper-plane". Chúng ta sẽ theo các tiêu chí sau:

Có 3 đường hyper-lane (A, B và C). Bây giờ đường nào là hyper-lane đúng cho nhóm ngôi sao và hình tròn.



Quy tắc số một để chọn 1 hyper-lane, chọn một hyper-plane để phân chia hai lớp tốt nhất. Trong ví dụ này chính là đường B.

Quy tắc thứ hai chính là xác định khoảng cách lớn nhất từ điểu gần nhất của một lớp nào đó đến đường hyper-plane. Khoảng cách này được gọi là "Margin", Hãy nhìn hình bên dưới, trong đấy có thể nhìn thấy khoảng cách margin lớn nhất đấy là đường C. Cần nhớ nếu chọn lầm hyper-lane có margin thấp hơn thì sau này khi dữ liệu tăng lên thì sẽ



sinh ra nguy cơ cao về việc xác định nhầm lớp cho dữ liệu.

Margin là khoảng cách giữa siêu phẳng đến 2 điểm dữ liệu gần nhất tương ứng với các phân lớp. Trong ví dụ quả táo quả lê đặt trên mặt bán, margin chính là khoảng cách giữa cây que và hai quả táo và lê gần nó nhất. Điều quan trọng ở đây đó là phương pháp SVM luôn cố gắng cực đại hóa margin này, từ đó thu được một siêu phẳng tạo khoảng cách xa nhất so với 2 quả táo và lê. Nhờ vậy, SVM có thể giảm thiểu việc phân lớp sai (misclassification) đối với điểm dữ liệu mới đưa vào.

Cuối cùng, chúng tôi đã thử sử dụng Biểu đồ Gradients định hướng (HOG) để mô tả sự phân bố của gradient và hướng cạnh trong hình ảnh trước khi xử lý chúng. Ý tưởng đằng sau việc sử dụng HOG là những cảm xúc khác nhau sẽ có độ chuyển màu khác nhau và riêng biệt, đặc biệt là xung quanh vùng miệng và mắt. Mặc dù HOG không giúp ích đáng kể cho độ chính xác của SVM, nhưng nó đã nâng độ chính xác của bộ phân loại lên đến độ chính xác SVM cao nhất của chúng tôi là 53,81%.

HOG (histogram of oriented gradients) là một feature descriptor được sử dụng trong computer vision và xử lý hình ảnh, dùng để detec một đối tượng.

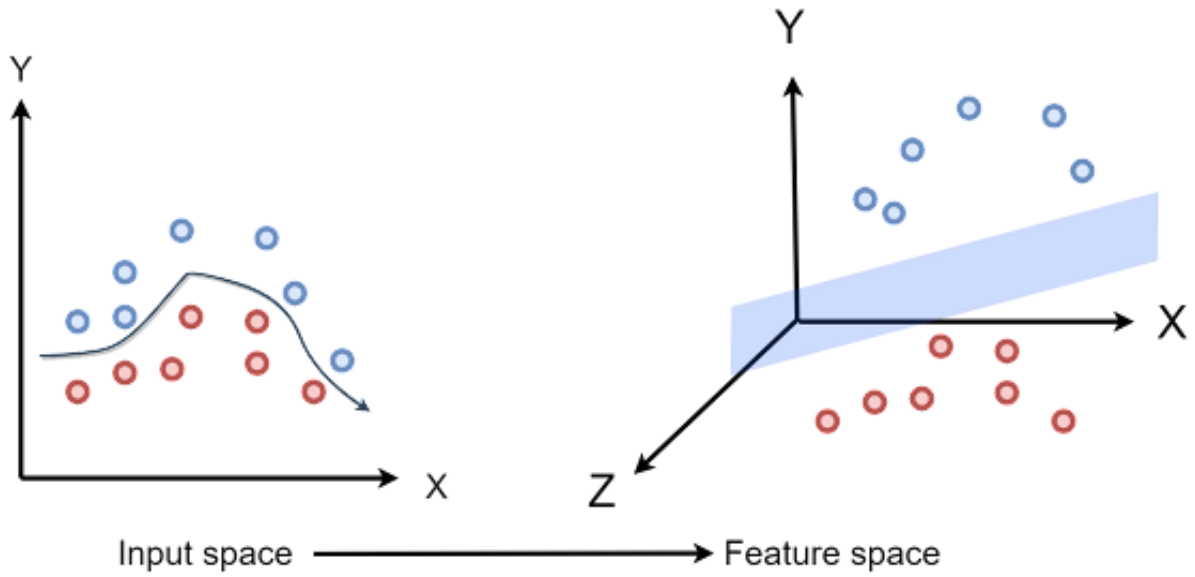
Hog được sử dụng chủ yếu để mô tả hình dạng và sự xuất hiện của một object trong ảnh. Bài toán tính toán Hog thường gồm 5 bước:

- Chuẩn hóa hình ảnh trước khi xử lý.
- Tính toán gradient theo cả hướng x và y.
- Lấy phiếu bầu cùng trọng số trong các cell.
- Chuẩn hóa các block.

- Thu thập tất cả các biểu đồ cường độ gradient định hướng để tạo ra feature vector cuối cùng.

Kernel trong SVM là gì?

SVM là một thuật toán đã cho thấy thành công lớn trong lĩnh vực phân loại. Nó phân tách dữ liệu thành các danh mục khác nhau bằng cách tìm hyperplane tốt nhất và tối đa hóa khoảng cách giữa các điểm. Để đạt được điều này, một kernel function sẽ được đề xuất để minh họa cách thức hoạt động của nó với support vector machines. Các kernel functions là một công cụ rất mạnh để khám phá các không gian nhiều chiều. Chúng cho phép chúng ta thực hiện phân biệt tuyến tính trên các manifolds phi tuyến, điều này có thể dẫn đến độ chính xác và độ bền cao hơn so với các mô hình tuyến tính truyền thống.



Kernel function chỉ là một hàm toán học sử dụng để chuyển đổi một không gian chiều thấp từ input thành một không gian có chiều cao hơn (nhiều chiều hơn). Điều này được thực hiện bằng cách ánh xạ data vào một không gian đặc trưng mới. Trong không gian này, data sẽ phân tách tuyến tính. Điều này có nghĩa rằng support vector space có thể được sử dụng để tìm một hyperplane để phân tách data.

Ví dụ: nếu input x có 2 chiều (không gian 2 chiều), kernel function sẽ ánh xạ vào một không gian 3 chiều. Trong không gian này, data sẽ phân tách tuyến tính (linearly separable).

Ngoài ra, chúng cung cấp nhiều đặc trưng hơn so với những thuật toán khác như nà neural networds hoặc tree ensembles trong một số vấn đề liên quan đến việc nhận dạng chữ viết tay, nhận diện khuôn mặt,…bởi vì chúng trích xuất các thuộc tính properties của các điểm data thông qua một kernel function.

Polynomial Kernel

Polynomial Kernel là một loại Kernel SVM sử dụng hàm đa thức để ánh xạ data vào không gian có chiều cao hơn. Nó thực hiện điều này bằng cách lấy tích vô hướng của các điểm dữ liệu trong không gian ban

đầu và hàm đa thức trong không gian mới. Trong polynomial kernel cho SVM, dữ liệu được ánh xạ vào không gian nhiều chiều hơn bằng cách sử dụng hàm đa thức. Tích vô hướng của các điểm dữ liệu trong không gian ban đầu và hàm đa thức trong không gian mới sau đó được lấy. Polynomial kernel thường được sử dụng trong các bài toán phân loại SVM khi dữ liệu không thể phân tách tuyến tính. Bằng cách ánh xạ dữ liệu vào một không gian nhiều chiều hơn, polynomial kernel đôi khi có thể tìm thấy một hyperplane phân tách các lớp. Polynomial kernel có một số tham số có thể được điều chỉnh để cải thiện hiệu suất của nó, bao gồm bậc của đa thức và hệ số của đa thức. Đối với đa thức bậc d, polynomial kernel được định nghĩa là:

$$K(x_1, x_2) = (x_1^T x_2 + c)^d$$

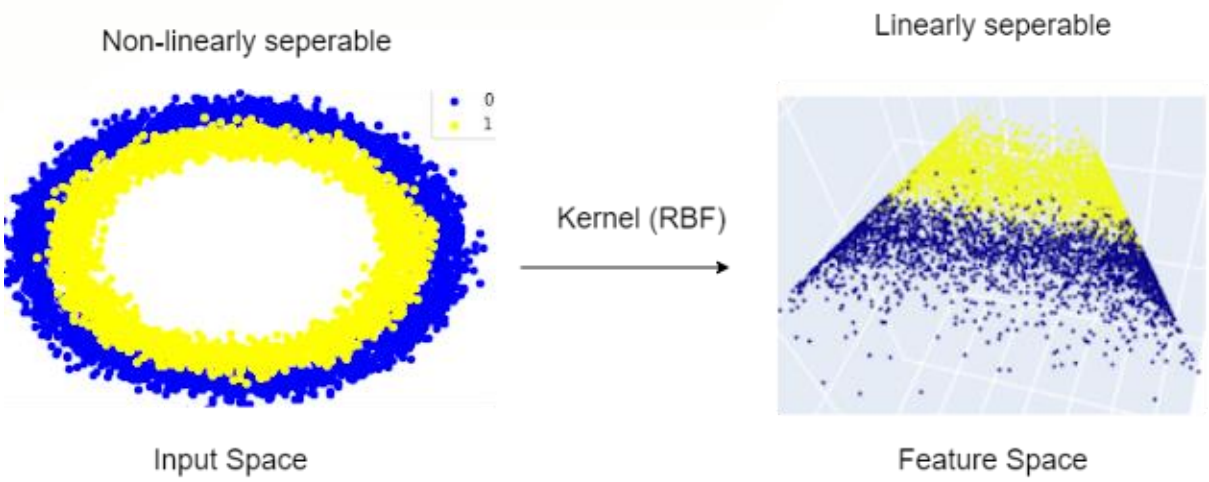
- Trong đó c là một hằng số và x1 và x2 là các vectơ trong không gian ban đầu. Tham số c có thể được sử dụng để kiểm soát sự đánh đổi giữa mức độ phù hợp của dữ liệu huấn luyện và kích thước của lề.
- Khi một tập dữ liệu được cung cấp có chứa các đặc trưng x1 và x2, phương trình có thể được chuyển thành:

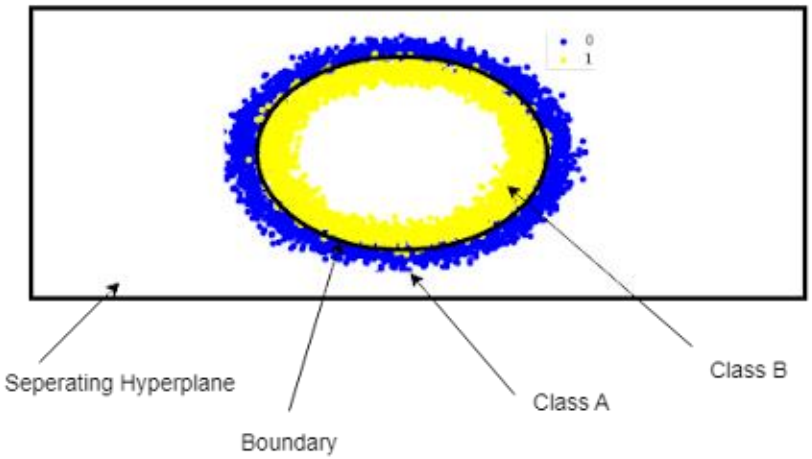
$$\begin{bmatrix} x_1 \\ x_2 \end{bmatrix} . \begin{bmatrix} x_1 & x_2 \end{bmatrix} = \begin{bmatrix} x_1^2 & x_1 x_2 \\ \\ x_1 x_2 & x_2^2 \end{bmatrix}$$

- Các term quan trọng chúng ta cần lưu lý là x1, x2, x1^2, x2^2, và x1*x2. Khi tìm thấy các terms mới này, tập dữ liệu phi tuyến tính được chuyển đổi sang một kích thước khác có các đặc trưng x1^2, x2^2, và x1*x2.

RBF kernel

RBF viết tắt của Radial Basis Function Kernel là một kernel rất mạnh được sử dụng trong SVM. Không giống như linear hay polynomial kernels, RBF phức tạp hơn và hiệu quả hơn đồng thời có thể kết hợp nhiều Polynomial Kernel nhiều lần ở các mức độ khác nhau để chiếu dữ liệu có thể phân tách phi tuyến tính vào không gian nhiều chiều hơn để có thể phân tách dữ liệu đó bằng hyperplane.





Kernel RBF hoạt động bằng cách ánh xạ dữ liệu vào không gian nhiều chiều bằng cách tìm các tích vô hướng và bình phương của tất cả các đối tượng trong tập dữ liệu, sau đó thực hiện phân loại bằng cách sử dụng ý tưởng cơ bản của SVM tuyến tính. Để chiếu dữ liệu vào một không gian nhiều chiều hơn, Kernel RBF sử dụng cái gọi là hàm cơ sở xuyên tâm có thể được viết là:

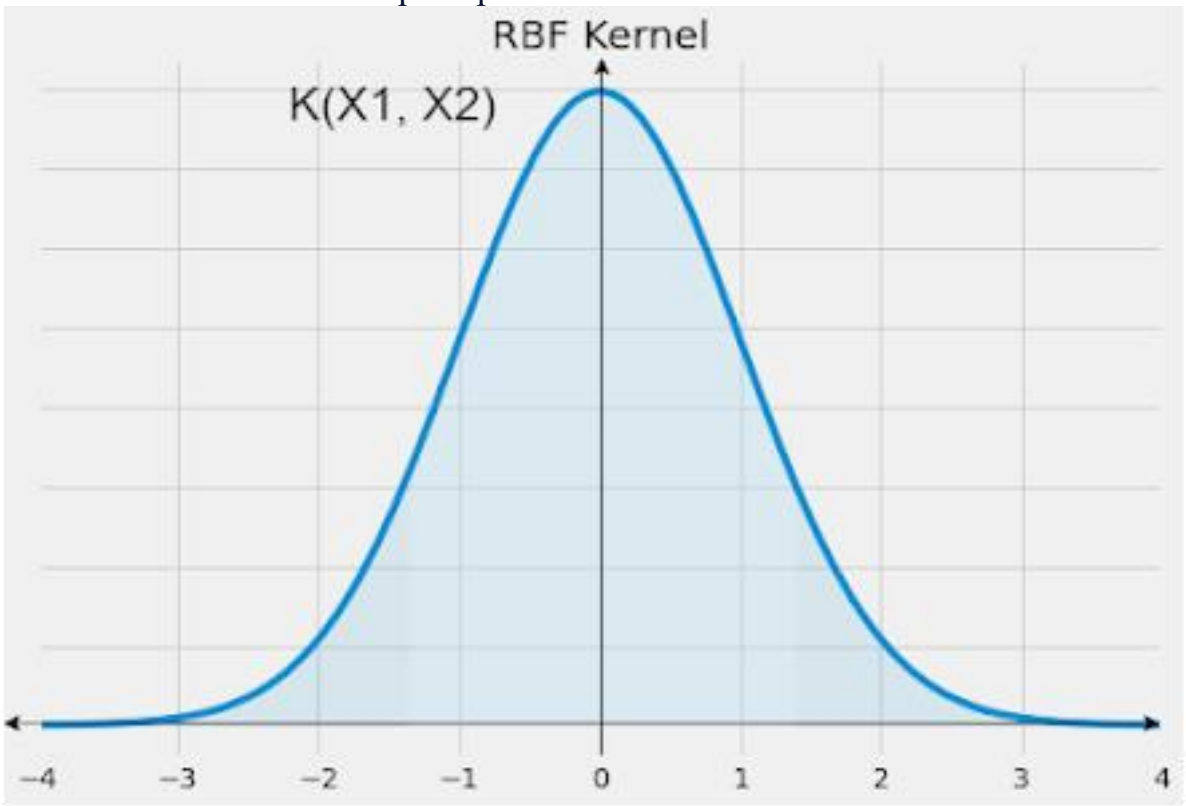
$$K(X_1, X_2) = \exp(-\frac{\|X_1 - X_2\|^2}{2\sigma^2})$$

Ở đây ||X1 - X2||^2 được gọi là Khoảng cách Euclide bình phương và σ là một tham số tự do có thể được sử dụng để điều chỉnh phương trình. Khi giới thiệu một tham số mới $\gamma = 1 / 2\sigma^2$, phương trình sẽ là

$$K(X_1, X_2) = \exp(-\gamma \|X_1 - X_2\|^2)$$

Khoảng cách Euclide bình phương được nhân với tham số gamma và sau đó tìm số mũ của tổng thể. Phương trình này có thể tìm thấy các sản phẩm bên trong được chuyển đổi để ánh xạ trực tiếp dữ liệu sang các chiều cao hơn mà không thực sự chuyển đổi toàn bộ tập dữ liệu dẫn đến không hiệu quả. Và đây là lý do tại sao nó được gọi là chức năng Kernel RBF.

Biểu đồ phân phối của RBF Kernel sẽ như sau:



2. Convolutional Neural Network (CNN)

Giới thiệu

Trong deep learning, CNN hay Convolutional Neural Network là lớp mạng nơ-ron nhân tạo, được dùng phổ biến trong công việc phân tích ảnh. Mạng CNN là một tập hợp các lớp tích chập chồng lên nhau và sử dụng các hàm nonlinear activation như ReLU và tanh để kích hoạt các trọng số trong các node. Mỗi một lớp sau khi thông qua các hàm kích hoạt sẽ tạo ra các thông tin trừu tượng hơn cho các lớp tiếp theo.

Mỗi một lớp sau khi thông qua các hàm kích hoạt sẽ tạo ra các thông tin trừu tượng hơn cho các lớp tiếp theo. Trong mô hình mạng truyền ngược (feedforward neural network) thì mỗi neural đầu vào (input node) cho mỗi neural đầu ra trong các lớp tiếp theo. Mô hình này gọi là mạng kết nối đầy đủ (fully connected layer). Còn trong mô hình CNNs thì ngược lại. Các lớp liên kết được với nhau thông qua cơ chế tích chập. Lớp tiếp theo là kết quả phép tính tích chập từ lớp trước đó, nhờ vậy mà ta có được các kết nối cục bộ. Như

vậy mỗi neuron ở lớp kế tiếp sinh ra từ kết quả của filter áp đặt lên một vùng ảnh cục bộ của neuron trước đó. Mỗi một lớp được sử dụng các filter khác nhau thông thường có hàng trăm hàng nghìn filter như vậy và kết hợp kết quả của chúng lại. Ngoài ra có một số lớp khác như pooling/subsampling lớp dùng để chắt lọc lại các thông tin hữu ích hơn. Trong quá trình huấn luyện mạng (traning) CNN tự động học các giá trị qua các lớp filter. Ví dụ trong tác vụ phân lớp ảnh, CNNs sẽ cố gắng tìm ra thông số tối ưu cho các filter tương ứng theo thứ tự raw pixel > edges > shapes > facial > high-level features. Lớp cuối cùng được dùng để phân lớp ảnh.

Cấu trúc mạng CNN

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

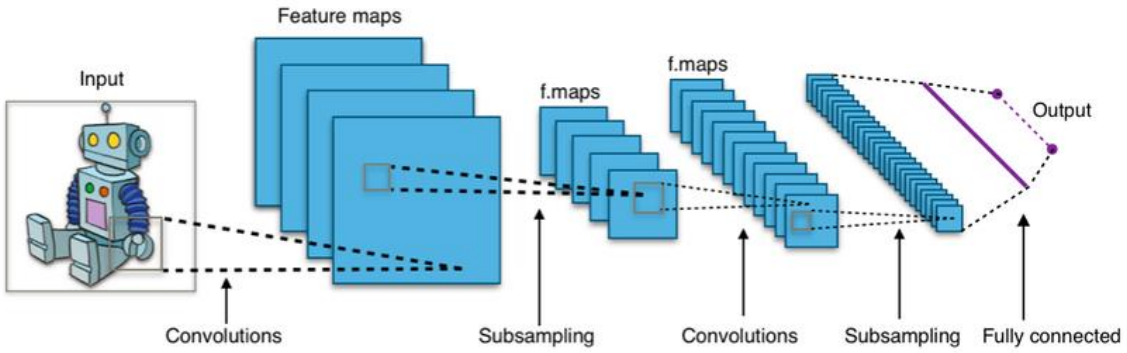
Mỗi một lớp sau khi thông qua các hàm kích hoạt sẽ tạo ra các thông tin trừu tượng hơn cho các lớp tiếp theo. Trong mô hình mạng truyền ngược (feedforward neural network) thì mỗi neural đầu vào (input node) cho mỗi neural đầu ra trong các lớp tiếp theo.

Mô hình này gọi là mạng kết nối đầy đủ (fully connected layer) hay mạng toàn vẹn (affine layer). Còn trong mô hình CNNs thì ngược lại. Các layer liên kết được với nhau thông qua cơ chế convolution.

Layer tiếp theo là kết quả convolution từ layer trước đó, nhờ vậy mà ta có được các kết nối cục bộ. Như vậy mỗi neuron ở lớp kế tiếp sinh ra từ kết quả của filter áp đặt lên một vùng ảnh cục bộ của neuron trước đó.

Mỗi một lớp được sử dụng các filter khác nhau thông thường có hàng trăm hàng nghìn filter như vậy và kết hợp kết quả của chúng lại. Ngoài ra có một số layer khác như pooling/subsampling layer dùng để chắt lọc lại các thông tin hữu ích hơn (loại bỏ các thông tin nhiễu).

Trong quá trình huấn luyện mạng (traning) CNN tự động học các giá trị qua các lớp filter dựa vào cách thức mà bạn thực hiện. Ví dụ trong tác vụ phân lớp ảnh, CNNs sẽ cố gắng tìm ra thông số tối ưu cho các filter tương ứng theo thứ tự raw pixel > edges > shapes > facial > high-level features. Layer cuối cùng được dùng để phân lớp ảnh.



Trong mô hình CNN có 2 khía cạnh cần quan tâm là tính bất biến (Location Invariance) và tính kết hợp (Compositionality). Với cùng một đối tượng, nếu đối tượng này được chiếu theo các gốc độ khác nhau (translation, rotation, scaling) thì độ chính xác của thuật toán sẽ bị ảnh hưởng đáng kể.

Pooling layer sẽ cho bạn tính bất biến đối với phép dịch chuyển (translation), phép quay (rotation) và phép co giãn (scaling). Tính kết hợp cục bộ cho ta các cấp độ biểu diễn thông tin từ mức độ thấp đến mức độ cao và trừu tượng hơn thông qua convolution từ các filter.

Đó là lý do tại sao CNNs cho ra mô hình với độ chính xác rất cao. Cũng giống như cách con người nhận biết các vật thể trong tự nhiên.

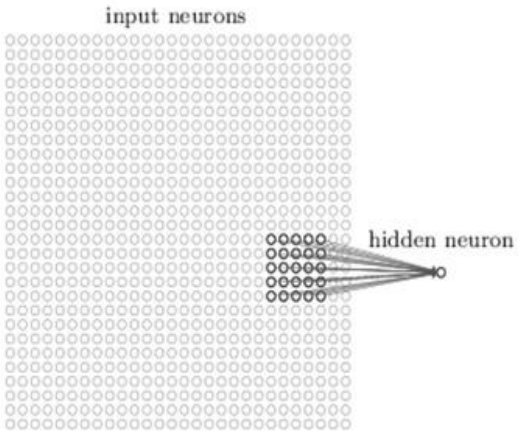
Mạng CNN sử dụng 3 ý tưởng cơ bản:

- các trường tiếp nhận cục bộ (local receptive field)
- trọng số chia sẻ (shared weights)
- tổng hợp (pooling).

Trường tiếp nhận cục bộ (local receptive field)

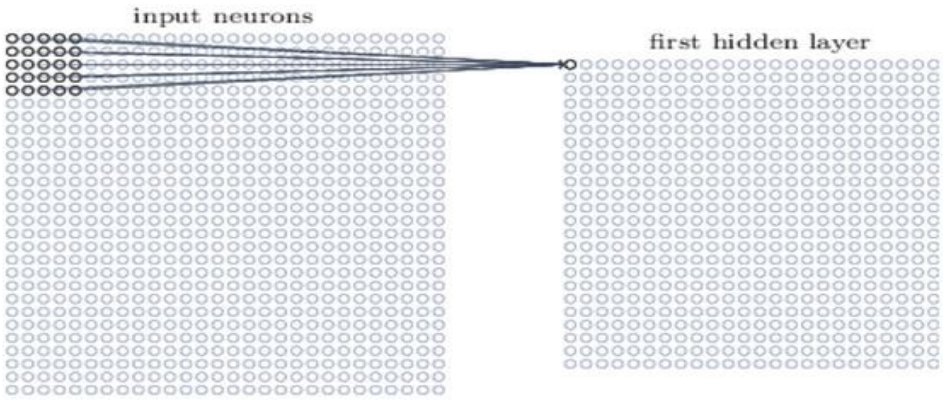
Đầu vào của mạng CNN là một ảnh. Ví dụ như ảnh có kích thước 28×28 thì tương ứng đầu vào là một ma trận có 28×28 và giá trị mỗi điểm ảnh là một ô trong ma trận. Trong mô hình mạng ANN truyền thống thì chúng ta sẽ kết nối các neuron đầu vào vào tầng ảnh.

Tuy nhiên trong CNN chúng ta không làm như vậy mà chúng ta chỉ kết nối trong một vùng nhỏ của các neuron đầu vào như một filter có kích thước 5×5 tương ứng (28- 5 + 1) 24 điểm ảnh đầu vào. Mỗi một kết nối sẽ học một trọng số và mỗi neuron ẩn sẽ học một bias. Mỗi một vùng 5×5 đấy gọi là một trường tiếp nhận cục bộ.

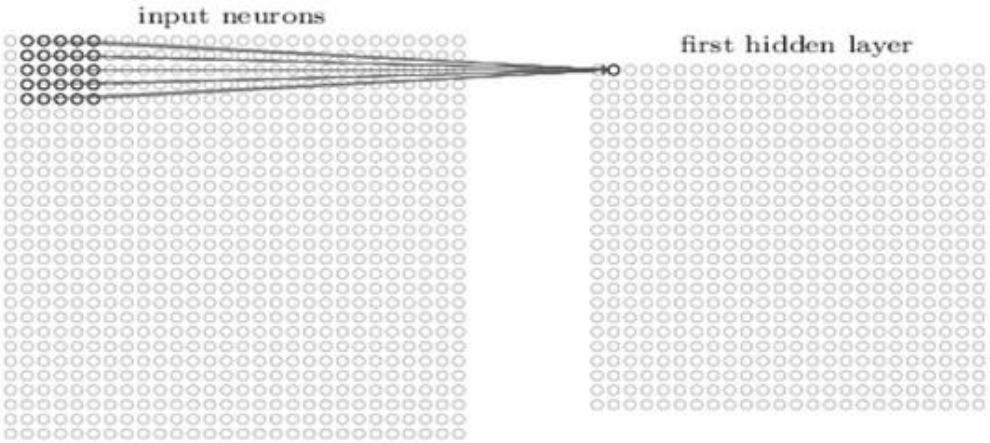


Một cách tổng quan, ta có thể tóm tắt các bước tạo ra 1 hidden layer bằng các cách sau:

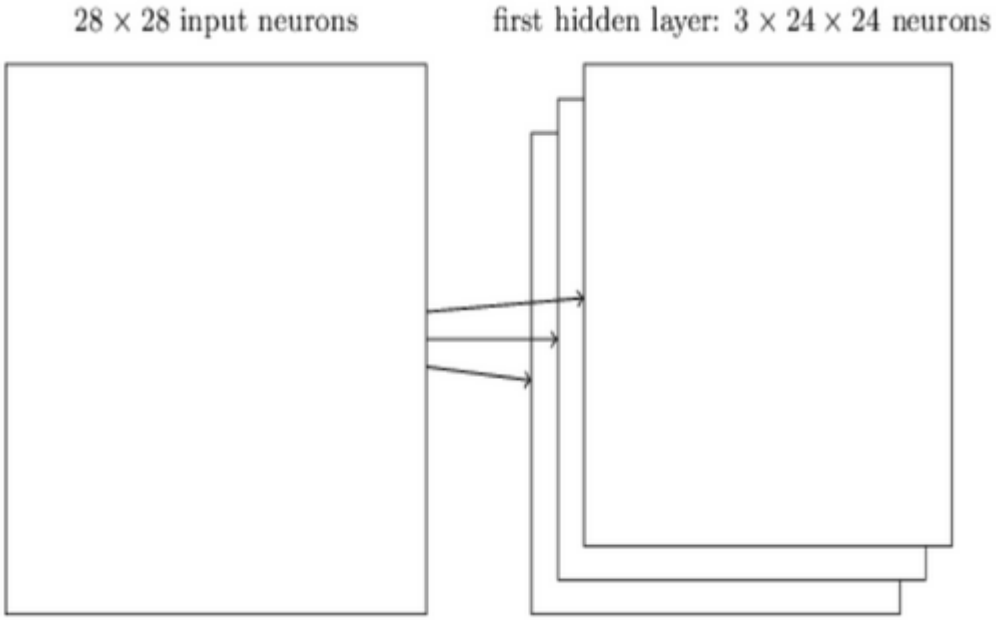
1. Tạo ra neuron ẩn đầu tiên trong lớp ần 1



2. Dịch filter qua bên phải một cột sẽ tạo được neuron ẩn thứ 2



Với bài toán nhận dạng ảnh người ta thường gọi ma trận lớp đầu vào là feature map, trọng số xác định các đặc trương là shared weight và độ lệch xác định một feature map là shared bias. Như vậy đơn giản nhất là qua các bước trên chúng ta chỉ có 1 feature map. Tuy nhiên trong nhận dạng ảnh chúng ta cần nhiều hơn một feature map.



Như vậy, local receptive field thích hợp cho việc phân tách dữ liệu ảnh, giúp chọn ra những vùng ảnh có giá trị nhất cho việc đánh giá phân lớp.

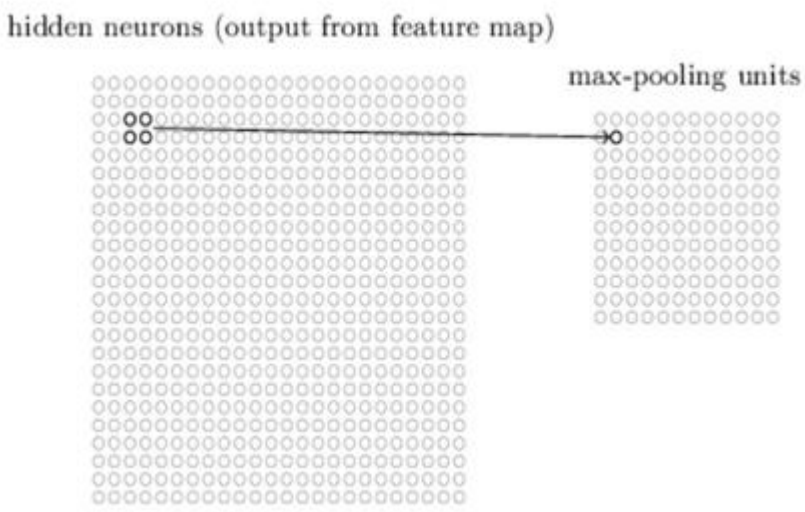
Trọng số chia sẻ (shared weight and bias)

Đầu tiên, các trọng số cho mỗi filter (kernel) phải giống nhau. Tất cả các nơ-ron trong lớp ẩn đầu sẽ phát hiện chính xác feature tương tự chỉ ở các vị trí khác nhau trong hình ảnh đầu vào. Chúng ta gọi việc map từ input layer sang hidden layer là một feature map. Mỗi một feature map giúp detect một vài feature trong bức ảnh. Lợi ích lớn nhất của trọng số chia sẻ là giảm tối đa số lượng tham số trong mạng CNN.

Lớp tổng hợp (pooling layer)

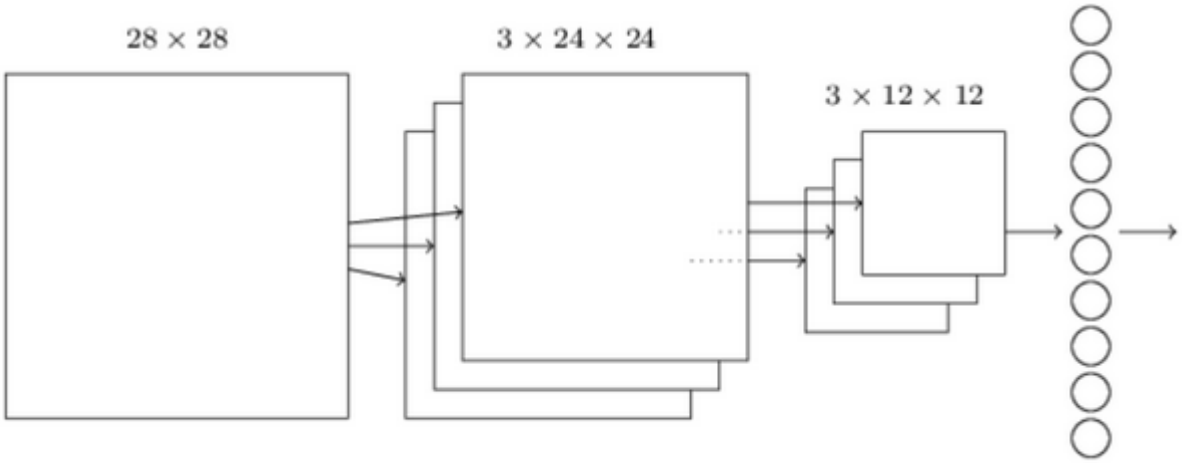
Lớp pooling thường được sử dụng ngay sau lớp convulational để đơn giản hóa thông tin đầu ra để giảm bớt số lượng neuron.

Thủ tục pooling phổ biến là max-pooling, thủ tục này chọn giá trị lớn nhất trong vùng đầu vào 2×2.



Như vậy qua lớp Max Pooling thì số lượng neuron giảm đi phân nửa. Trong một mạng CNN có nhiều Feature Map nên mỗi Feature Map chúng ta sẽ cho mỗi Max Pooling khác nhau. Chúng ta có thể thấy rằng Max Pooling là cách hỏi xem trong các đặc trưng này thì đặc trưng nào là đặc trưng nhất. Ngoài Max Pooling còn có L2 Pooling.

Cuối cùng ta đặt tất cả các lớp lại với nhau thành một CNN với đầu ra gồm các neuron với số lượng tùy bài toán.



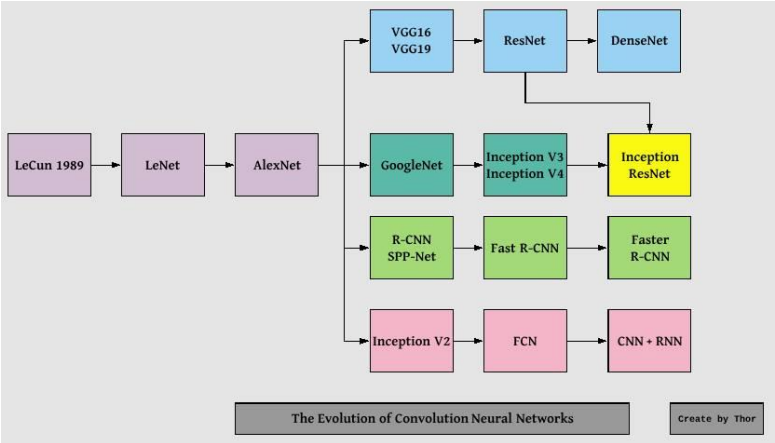
2 lớp cuối cùng của các kết nối trong mạng là một lớp đầy đủ kết nối (fully connected layer) . Lớp này nối mọi nơron từ lớp max pooled tới mọi nơron của tầng ra.

Cách chọn tham số cho CNN

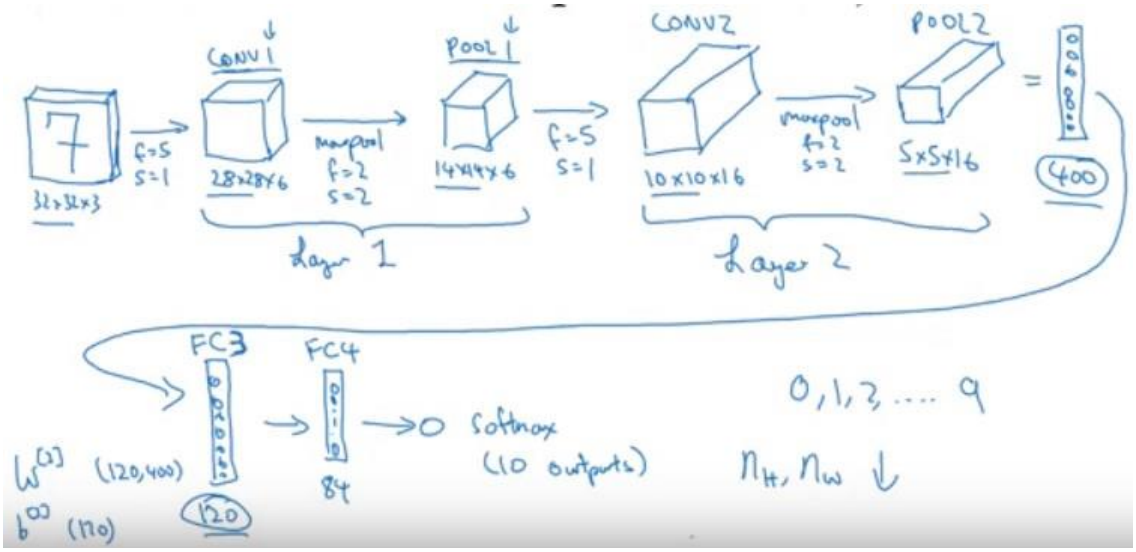
1. Số các convolution layer: càng nhiều các convolution layer thì performance càng được cải thiện. Sau khoảng 3 hoặc 4 layer, các tác động được giảm một cách đáng kể
2. Filter size: thường filter theo size 5×5 hoặc 3×3
3. Pooling size: thường là 2×2 hoặc 4×4 cho ảnh đầu vào lớn
4. Cách cuối cùng là thực hiện nhiều lần việc train test để chọn ra được param tốt nhất.

Lịch sử phát triển

1. LeNet(1988)



LeNet là một trong những mạng CNN lâu đời nổi tiếng nhất được Yann LeCUn phát triển vào những năm 1998s. Cấu trúc của LeNet gồm 2 layer (Convolution + maxpooling) và 2 layer fully connected layer và output là softmax layer . Chúng ta cùng tìm hiểu chi tiết architect của LeNet đối với dữ liệu mnist (accuracy lên đến 99%).



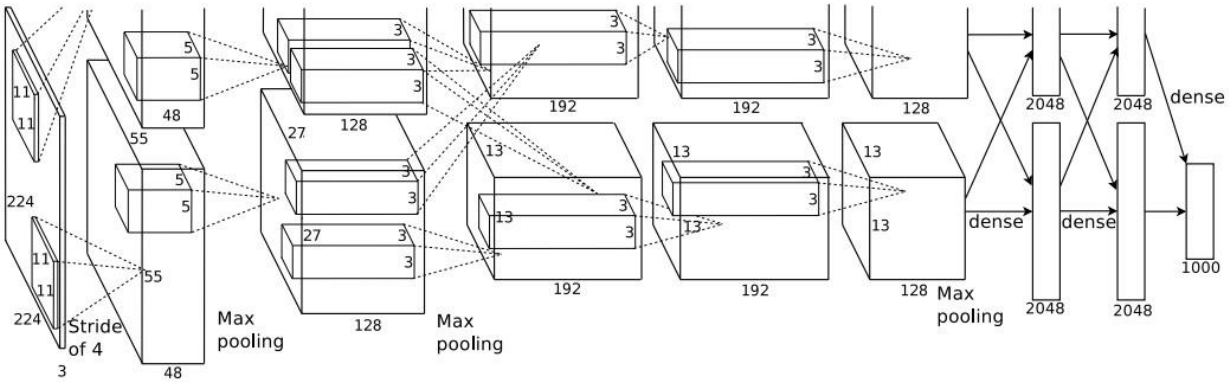
- Input shape 28x28x3

- Layer 1 :
- Convolution layer 1 : Kernel 5x5x3 , stride = 1,no padding, number filter = 6 ,output = 28x28x6.
- Maxpooling layer : pooling size 2x2,stride = 2,padding = “same”,output = 14x14x6.
- Layer 2 :
- Convolution layer 2 : kernel 5x5x6,stride = 1, no padding, number filter = 16,output = 10x10x16.
- Maxpooling layer : pooling size = 2x2, stride = 2, padding =”same”,output = 5x5x16.
- Flatten output = 5x5x16 = 400
- Fully connected 1 : output = 120
- Fully connected 2 : output = 84
- Softmax layer, output = 10 (10 digits). Nhược điểm của LeNet là mạng còn rất đơn giản và sử dụng sigmoid (or tanh) ở mỗi convolution layer mạng tính toán rất chậm.

2. Alexnet(2012).

AlexNet là một mạng CNN đã dành chiến thắng trong cuộc thi ImageNet LSVRC-2012 năm 2012 với large margin (15.3% VS 26.2% error rates). AlexNet là một mạng CNN traning với một số lượng parameter rất lớn (60 million) so với LeNet. Một số đặc điểm:

- Sử dụng relu thay cho sigmoid(or tanh) để xử lý với non-linearity. Tăng tốc độ tính toán lên 6 lần.
- Sử dụng dropout như một phương pháp regularization mới cho CNN. Dropout không những giúp mô hình tránh được overfitting mà còn làm giảm thời gian huấn luyện mô hình
- Overlap pooling để giảm size của network (Traditionally pooling regions không overlap).
- Sử dụng local response normalization để chuẩn hóa ở mỗi layer.
- Sử dụng kỹ thuật data augmentation để tạo them data training bằng cách translations, horizontal reflections.
- Alexnet training với 90 epochs trong 5 đến 6 ngày với 2 GTX 580 GPUs. Sử dụng SGD với learning rate 0.01, momentum 0.9 và weight decay 0.0005.



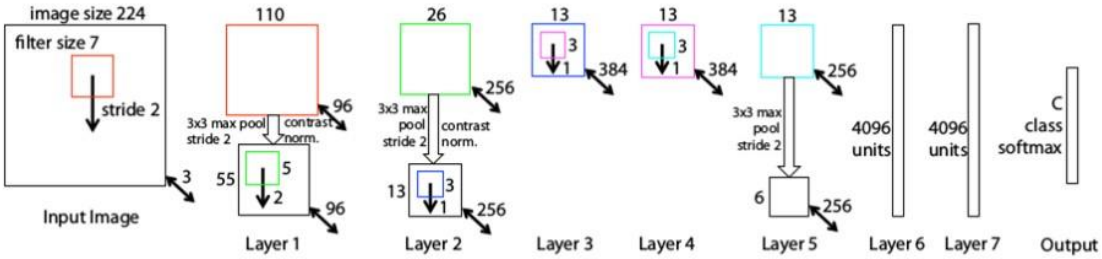
- Architect của Alexnet gồm 5 convolutional layer và 3 fully connection layer. Activation Relu được sử dụng sau mỗi convolution và fully connection layer. Detail architecter với dataset là imagenet size là 227x227x3 với 1000 class (khác với trong hình trên size là 224x224):
- Input shape 227x227x3.
- Layer 1 :
- Conv 1 : kernel : 11x11x3,stride = 4,no padding, number = 96,activation = relu,output = 55x55x96.
- Maxpooling layer : pooling size = 3x3,stride = 2,padding =”same” ,output = 27x27x96.
- Normalize layer.
- Layer 2 :
- Conv 2 : kernel :3x3x96,stride = 1, padding = “same”, number filter = 256,activation = relu,output = 27x27x256.
- Maxpooling layer : pooling size = 3x3,stride=2, padding =”same”,output = 13x13x256.
- Normalize layer.
- Layer 3:
- Conv 3 : kernel :3x3x256, stride = 1,padding=”same”, number filter = 384, activation = relu, output = 13x13x384.
- Layer 4:
- Conv 4 : kernel : 3x3x384 , stride = 1, padding = “same”, number filter = 384, activation= relu, output = 13x13x384
- Layer 5 :
- Conv 5 : kernel 3x3x384, stride = 1, padding = “same”, number filter = 256, activation = relu, output = 13x13x256.
- Pooling layer : pooling size = 3x3,stride =2,padding =”same”,output = 6x6x256.
- Flatten 256x6x6 = 9216
- Fully connected layer 1 : activation = relu , output = 4096 + dropout(0.5).
- Fully connected layer 2 : activation = relu , output = 4096 + dropout(0.5).

- Fully connected layer 3 : activation = softmax , output = 1000 (number class)

3. ZFNet(2013).

ZFNet là một mạng cnn thắng trong ILSVRC 2013 với top-5 error rate của 14.8% . ZFNet có cấu trúc rất giống với AlexNet với 5 layer convolution , 2 fully connected layer và 1 output softmax layer. Khác biệt ở chỗ kernel size ở mỗi Conv layer .Một số đặc điểm chính :

- Tương tự AlexNet nhưng có một số điều chỉnh nhỏ.
- Alexnet training trên 15m image trong khi ZF training chỉ có 1.3m image.
- Sử dụng kernel 7x7 ở first layer (alexnet 11x11).Lý do là sử dụng kernel nhỏ hơn để giữ lại nhiều thông tin trên image hơn.
- Tăng số lượng filter nhiều hơn so với alexnet
- Training trên GTX 580 GPU trong 20 ngày

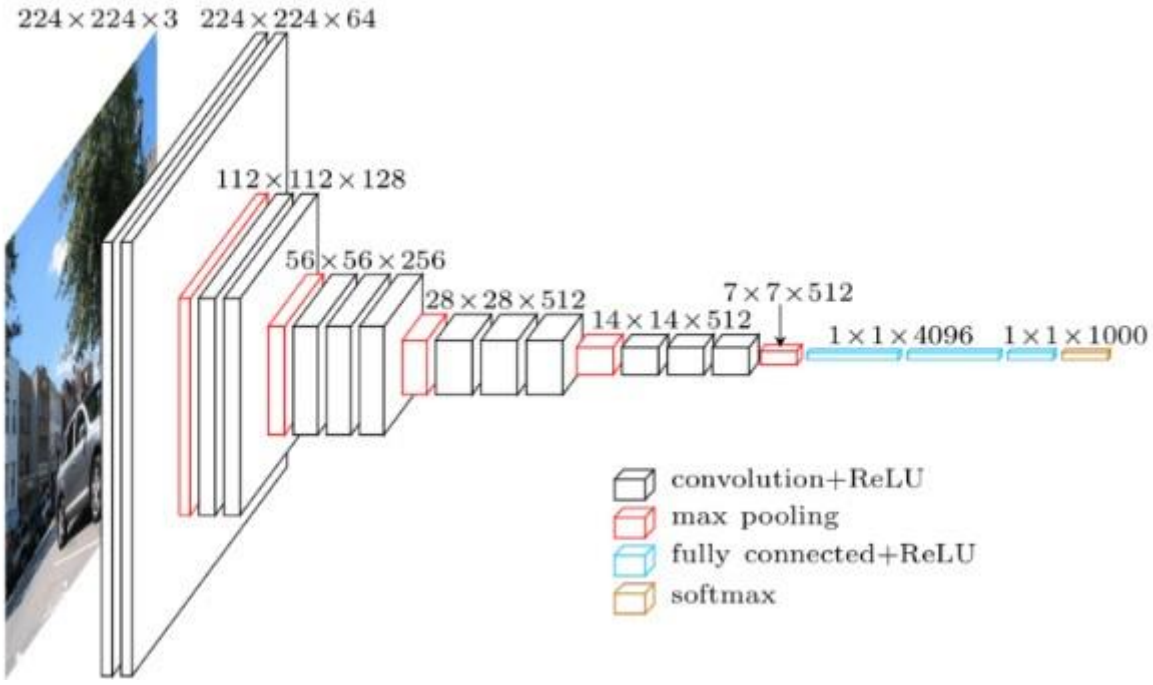


- Input shape 224x224x3 .
- Layer 1 :
- Conv 1 : kernel = 7x7x3, stride = 2, no padding, number filter = 96, output = 110x110x96.
- Maxpooling 1 : pooling size = 3x3,stride=2, padding = “same”,output = 55x55x96
- Normalize layer.
- Layer 2 :
- Conv 2 : kernel = 5x5x96, stride = 2, no padding, number filter = 256, output = 26x26x256.
- Maxpooling 2 : pooling size = 3x3, stride=2, padding = “same”,output = 13x13x256
- Normalize layer.
- Layer 3:
- Conv 3 : kernel = 3x3x256, stride=1, padding=”same”, number filter = 384,output = 13x13x384.
- Layer 4 :

- Conv 4 : kernel = 3x3x384, stride=1, padding=”same”, number filter = 384,output = 13x13x384.
- Layer 5 :
- Conv 5 : kernel = 3x3x384, stride=1, padding=”same”, number filter = 256,output = 13x13x256.
- Maxpooling : pooling size = 3x3,stride =2,padding =”same”,output = 6x6x256.
- Flatten 6x6x256 = 9216
- Fully connected 1 : activation = relu,output =4096
- Fully connected 2 : activation = relu,output =4096
- Softmax layer for classifier ouput = 1000

4. VGGNet(2014).

Sau AlexNet thì VGG ra đời với một số cải thiện hơn , trước tiên là model VGG sẽ deeper hơn, tiếp theo là thay đổi trong thứ tự conv. Từ LeNet đến AlexNet đều sử dụng Conv-maxpooling còn VGG thì sử dụng 1 chuỗi Conv liên tiếp Conv-Conv-Conv ở middle và end của architect VGG. Việc này sẽ làm cho việc tính toán trở nên lâu hơn nhưng những feature sẽ vẫn được giữ lại nhiều hơn so với việc sử dụng maxpooling sau mỗi Conv. Hơn nữa hiện nay với sự ra đời của GPU giúp tốc độ tính toán trở nên nhanh hơn rất nhiều lần thì vấn đề này không còn đáng lo ngại. VGG cho small error hơn AlexNet trong ImageNet Large Scale Visual Recognition Challenge (ILSVRC) năm 2014. VGG có 2 phiên bản là VGG16 và VGG19.



- Architect của VGG16 bao gồm 16 layer :13 layer Conv (2 layer conv-conv,3 layer conv-conv-conv) đều có kernel 3x3, sau mỗi layer

conv là maxpooling downsize xuống 0.5, và 3 layer fully connection. VGG19 tương tự như VGG16 nhưng có thêm 3 layer convolution ở 3 layer conv cuối (thành 4 conv stack với nhau).

- Detail parameter VGG16

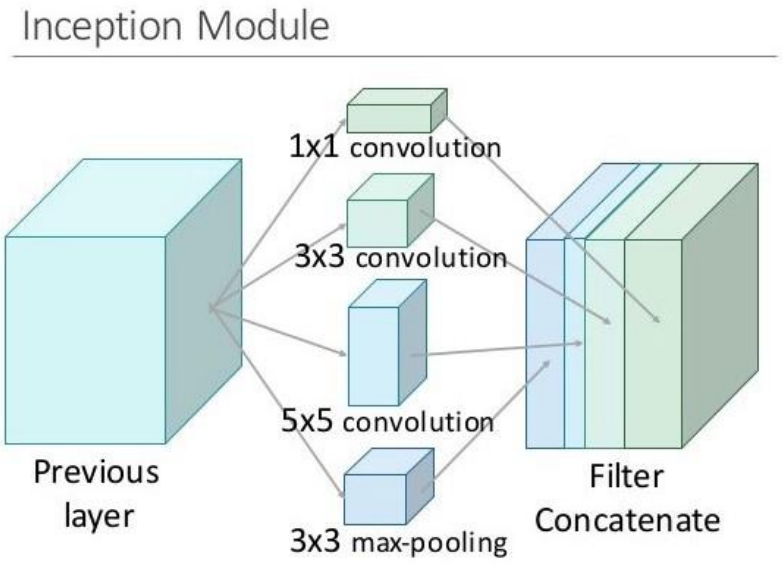
type	patch size/ stride	output size	depth	#1×1	#3×3 reduce	#3×3	#5×5 reduce	#5×5	pool proj	params	ops
convolution	7×7/2	112×112×64	1							2.7K	34M
max pool	3×3/2	56×56×64	0								
convolution	3×3/1	56×56×192	2		64	192				112K	360M
max pool	3×3/2	28×28×192	0								
inception (3a)		28×28×256	2	64	96	128	16	32	32	159K	128M
inception (3b)		28×28×480	2	128	128	192	32	96	64	380K	304M
max pool	3×3/2	14×14×480	0								
inception (4a)		14×14×512	2	192	96	208	16	48	64	364K	73M
inception (4b)		14×14×512	2	160	112	224	24	64	64	437K	88M
inception (4c)		14×14×512	2	128	128	256	24	64	64	463K	100M
inception (4d)		14×14×528	2	112	144	288	32	64	64	580K	119M
inception (4e)		14×14×832	2	256	160	320	32	128	128	840K	170M
max pool	3×3/2	7×7×832	0								
inception (5a)		7×7×832	2	256	160	320	32	128	128	1072K	54M
inception (5b)		7×7×1024	2	384	192	384	48	128	128	1388K	71M
avg pool	7×7/1	1×1×1024	0								
dropout (40%)		1×1×1024	0								
linear		1×1×1000	1							1000K	1M
softmax		1×1×1000	0								

- Sử dụng kernel 3x3 thay vì 11x11 ở alexnet(7x7 ZFNet). Kết hợp 2 conv 3x3 có hiểu quả hơn 1 cov 5x5 về receptive field giúp mạng deeper hơn lại giảm tham số tính toán cho model.
- 3 Conv 3x3 có receptive field same 1 conv 7x7.
- Input size giảm dần qua các conv nhưng tăng số chiều sâu.
- Làm việc rất tốt cho task classifier và localizer (rất hay được sử dụng trong object detection).
- Sử dụng relu sau mỗi conv và training bằng batch gradient descent.
- Có sử dụng data augmentation technique trong quá trình training.
- Training với 4 Nvidia Titan Black GPUs trong 2-3 tuần.

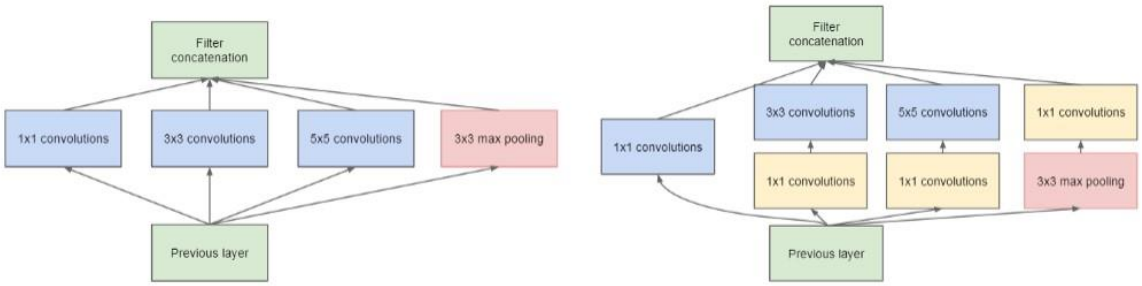
5. GoogleNet(2014).

- Năm 2014, google publish một mạng neural do nhóm research của họ phát triển có tên là googleNet. Nó performance tốt hơn VGG, googleNet 6.7% error rate trong khi VGG là 7.3% Ý tưởng chính là họ tạo ra một module mới có tên là inception giúp mạng traning sâu và nhanh hơn, chỉ có 5m tham số so với alexnet là 60m nhanh hơn gấp 12 lần.
- Inception module là một mạng CNN giúp training wider(thay vì them nhiều layer hơn vì rất dễ xảy ra overfitting + tăng parameter người ta nghĩ ra tăng deeper ở mỗi tầng layer) so với mạng CNN

bình thường. Mỗi layer trong CNN truyền thống sẽ extract các thông tin khác nhau. Output của 5x5 conv kernel sẽ khác với 3x3 kernel. Vậy để lấy những thông tin cần thiết cho bài toán của chúng ta thì nên dùng kernel size như thế nào ? Tại sao chúng sử dụng tất cả ta và sau đó để model tự chọn. Đó chính là ý tưởng của Inception module, nó tính toán các kernel size khác nhau từ một input sau đó concatenate nó lại thành output.

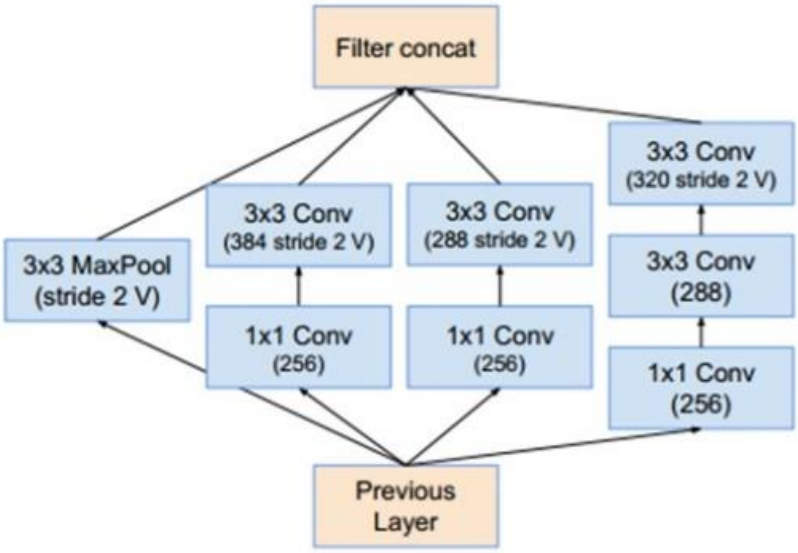


- Trong inception người ta dùng conv kernel 1x1 với 2 mục đích là giảm tham số tính toán và dimensionality reduction . Dimensionality reduction có thể hiểu làm giảm depth của input (vd iput 28x28x100 qua kernel 1x1 với filter = 10 sẽ giảm depth về còn 28x28x10). Giảm chi phí tính toán có thể hiểu qua ví dụ sau :
    - Input shape 28x28x192 qua kernel 5x5 với 32 thì ouput là 28x28x32(padding same) thì tham số tính toán là (5x5x192)*(28x28x32)=120 million
    - Input shape 28x28x192 qua kernel 1x1x192 filter = 16 , output = 28x28x16 tiếp tục với kernel 5x5x32 filter = 16 đươch output = 28x28x32. Tổng tham số tính toán : (28x28x16)*192 + (28x28x32)*(5x5x16) = 2.4 + 10 = 12.4 million. Ta thấy với cùng output là 28x28x32 thì nếu dùng kernel 5x5x192 với 32 filter thì sẽ có tham số gấp 10 lần so với sử dụng kernel 1x1x192 sau đó dùng tiếp 1 kernel 5x5x16 với filter 32. Inception hiện giờ có 4 version , ta sẽ cùng tìm hiểu sơ qua các version:
- Inception v1 : có 2 dạng là naïve và dimension reduction. Khác biệt chính đó là version dimension reduction nó dùng conv 1x1 ở mỗi layer để giảm depth của input giúp model có ít tham số hơn. Inception naïve có architect gồm 1x1 conv,3x3 conv, 5x5 conv và 3x3 maxpooling.

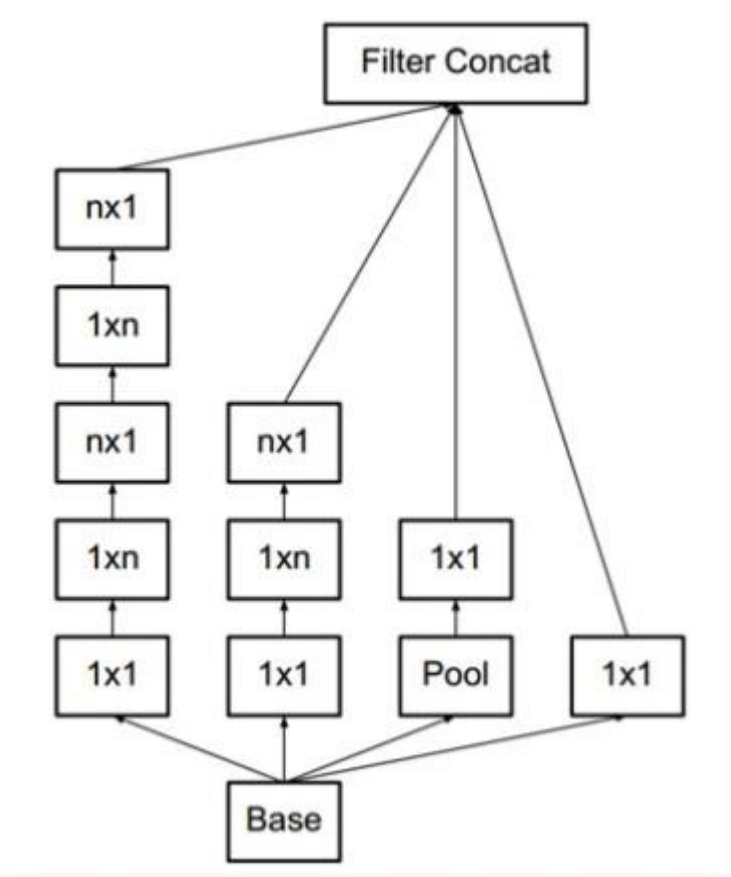


(a) Inception module, naïve version (b) Inception module with dimension reductions

Inception v2 : Cải thiện version 1, thêm layer batchnormalize và giảm Internal Covariate Shift. Ouput của mỗi layer sẽ được normalize về Gaussian N(0,1). Conv 5x5 sẽ được thay thế bằng 2 conv 3x3 để giảm computation cost.



Inception v3 : Điểm đáng chú ý ở version này là Factorization. Conv 7x7 sẽ được giảm về conv 1 dimesion là (1x7),(7x1). Tương tự conv 3x3 (3x1,1x3). Tăng tốc độ tính toán. Khi tách ra 2 conv thì làm model sâu hơn.



Inception v4 : là sự kết hợp inception và resnet. Detail googleNet architect

type	patch size/ stride	output size	depth	#1×1	#3×3 reduce	#3×3	#5×5 reduce	#5×5	pool proj	params	ops
convolution	7×7/2	112×112×64	1							2.7K	34M
max pool	3×3/2	56×56×64	0								
convolution	3×3/1	56×56×192	2		64	192				112K	360M
max pool	3×3/2	28×28×192	0								
inception (3a)		28×28×256	2	64	96	128	16	32	32	159K	128M
inception (3b)		28×28×480	2	128	128	192	32	96	64	380K	304M
max pool	3×3/2	14×14×480	0								
inception (4a)		14×14×512	2	192	96	208	16	48	64	364K	73M
inception (4b)		14×14×512	2	160	112	224	24	64	64	437K	88M
inception (4c)		14×14×512	2	128	128	256	24	64	64	463K	100M
inception (4d)		14×14×528	2	112	144	288	32	64	64	580K	119M
inception (4e)		14×14×832	2	256	160	320	32	128	128	840K	170M
max pool	3×3/2	7×7×832	0								
inception (5a)		7×7×832	2	256	160	320	32	128	128	1072K	54M
inception (5b)		7×7×1024	2	384	192	384	48	128	128	1388K	71M
avg pool	7×7/1	1×1×1024	0								
dropout (40%)		1×1×1024	0								
linear		1×1×1000	1							1000K	1M
softmax		1×1×1000	0								

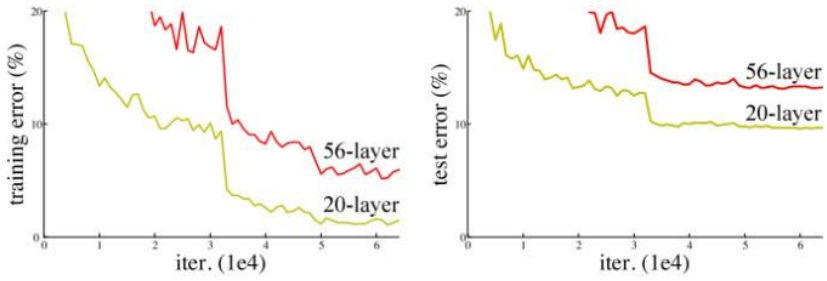
GoogleNet gồm 22 layer, khởi đầu vẫn là những simple convolution layer, tiếp theo là những block của inception module với maxpooling theo sau mỗi block. Một số đặc điểm chính:

- Sử dụng 9 Inception module trên toàn bộ architect. Làm model deeper hơn rất nhiều.

- Không sử dụng fully connection layer mà thay vào đó là average pooling từ 7x7x1024 volume thành 1x1x1024 volume giảm thiểu được rất nhiều parameter.
- Ít hơn 12x parameter so với Alexnet.
- Auxiliary Loss được add vào total loss(weight =0.3). Nhưng được loại bỏ khi test.

6. ResNets(2015).

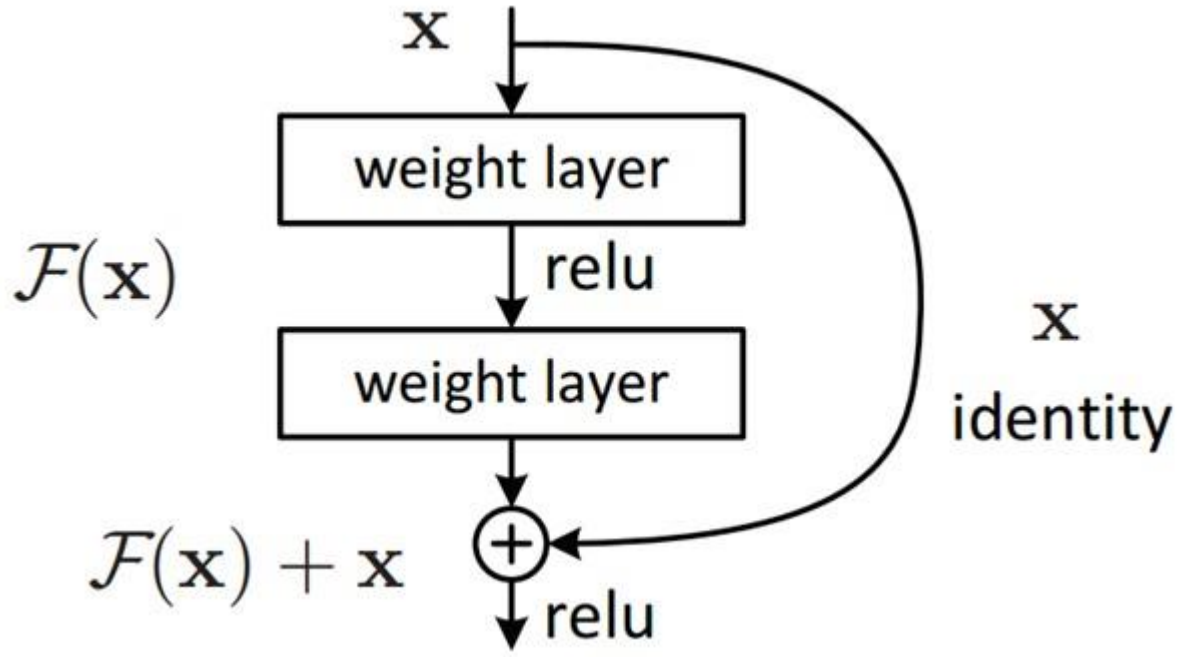
ResNet được phát triển bởi microsoft năm 2015 với paper " Deep residual learning for image recognition". ResNet winer ImageNet ILSVRC competition 2015 với error rate 3.57% ,ResNet có cấu trúc gần giống VGG với nhiều stack layer làm cho model deeper hơn. Không giống VGG, resNet có depth sâu hơn như 34,55,101 và 151 . Resnet giải quyết được vấn đề của deep learning truyền thống , nó có thể dễ dàng training model với hàng trăm layer. Để hiểu ResNet chúng ta cần hiểu vấn đề khi stack nhiều layer khi training, vấn đề đầu tiên khi tăng model deeper hơn gradient sẽ bị vanishing/explodes. Vấn đề này có thể giải quyết bằng cách thêm Batch Normalization nó giúp normalize output giúp các hệ số trở nên cân bằng hơn không quá nhỏ hoặc quá lớn nên sẽ giúp model dễ hội tụ hơn. Vấn đề thứ 2 là degradation, Khi model deeper accuracy bắt đầu bão hòa(saturated) thậm chí là giảm. Như hình vẽ bên dưới khi stack nhiều layer hơn thì training error lại cao hơn ít layer như vậy vấn đề không phải là do overfitting. Vấn đề này là do model không dễ training khó học hơn, thử tượng tượng một training một shallow model, sau đó chúng ta stack thêm nhiều layer , các layer sau khi thêm vào sẽ không học thêm được gì cả (identity mapping) nên accuracy sẽ tương tự như shallow model mà không tăng. Resnet được ra đời để giải quyết vấn đề degradation này.



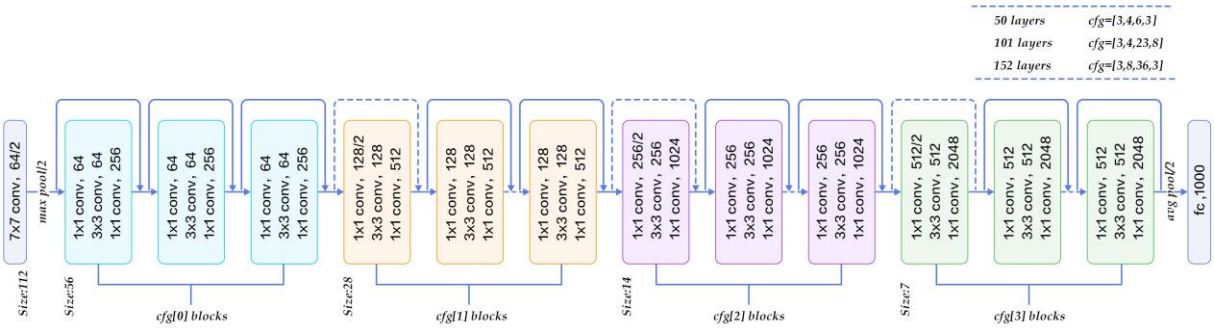
increasing network depth leads to worse performance

ResNet có architecture gồm nhiều residual block, ý tưởng chính là skip layer bằng cách add connection với layer trước. Ý tưởng của residual block là feed foword x(input) qua một số layer conv-max-conv,

ta thu được F(x) sau đó add thêm x vào H(x) = F(x) + x . Model sẽ dễ học hơn khi chúng ta thêm feature từ layer trước vào.



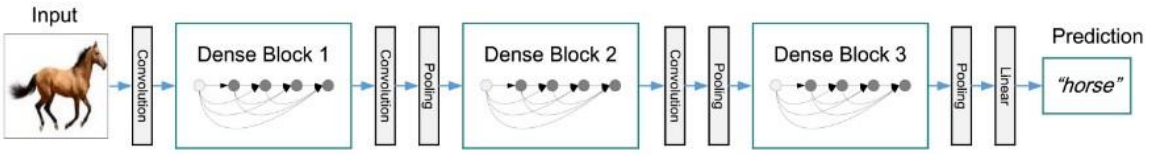
- Sử dụng batch Normalization sau mỗi Conv layer.
- Initialization Xavier/2
- Training với SGD + momentum(0.9)
- Learning rate 0.1, giảm 10 lần nếu error ko giảm
- Mini batch size 256
- Weight decay 10^-5
- Không sử dụng dropout



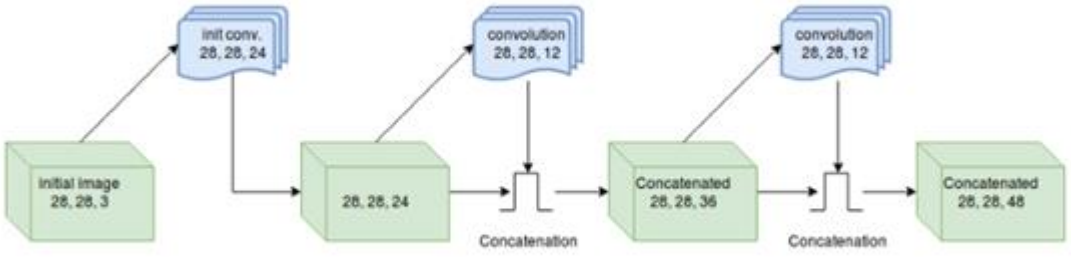
7. Densenet(2016)

Densenet(Dense connected convolutional network) là một trong những netwok mới nhất cho visual object recognition. Nó cũng gần giống Resnet nhưng có một vài điểm khác biệt. Densenet có cấu trúc gồm các dense block và các transition layers. Được stack dense block-transition layers-dense block- transition layers như hình vẽ. Với CNN

truyền thống nếu chúng ta có L layer thì sẽ có L connection, còn trong densenet sẽ có L(L+1)/2 connection.



Hãy tưởng tượng ban đầu ta có 1 image size (28,28,3). Đầu tiên ta khởi tạo feature layer bằng Conv tạo ra 1 layer size (28,28,24). Sau mỗi layer tiếp theo (Trong dense block) nó sẽ tạo thêm K= 12 feature giữa nguyên width và height. Khi đó output tiếp theo sẽ là (28,28,24 +12),(28,28,24 +12+12). Ở mỗi dense block sẽ có normalization, nonlinearity và dropout. Để giảm size và depth của feature thì transition layer được đặt giữa các dense block, nó gồm Conv kernel size =1, average pooling (2x2) với stride = 2 nó sẽ giảm output thành (14,14,48)



Block of convolution layers with results concatenated

Detail parameter :

Layers	Output Size	DenseNet-121	DenseNet-169	DenseNet-201	DenseNet-264
Convolution	112 × 112	7 × 7 conv, stride 2			
Pooling	56 × 56	3 × 3 max pool, stride 2			
Dense Block (1)	56 × 56	[1 × 1 conv, 3 × 3 conv] × 6	[1 × 1 conv, 3 × 3 conv] × 6	[1 × 1 conv, 3 × 3 conv] × 6	[1 × 1 conv, 3 × 3 conv] × 6
Transition Layer (1)	56 × 56	1 × 1 conv			
	28 × 28	2 × 2 average pool, stride 2			
Dense Block (2)	28 × 28	[1 × 1 conv, 3 × 3 conv] × 12	[1 × 1 conv, 3 × 3 conv] × 12	[1 × 1 conv, 3 × 3 conv] × 12	[1 × 1 conv, 3 × 3 conv] × 12
Transition Layer (2)	28 × 28	1 × 1 conv			
	14 × 14	2 × 2 average pool, stride 2			
Dense Block (3)	14 × 14	[1 × 1 conv, 3 × 3 conv] × 24	[1 × 1 conv, 3 × 3 conv] × 32	[1 × 1 conv, 3 × 3 conv] × 48	[1 × 1 conv, 3 × 3 conv] × 64
Transition Layer (3)	14 × 14	1 × 1 conv			
	7 × 7	2 × 2 average pool, stride 2			
Dense Block (4)	7 × 7	[1 × 1 conv, 3 × 3 conv] × 16	[1 × 1 conv, 3 × 3 conv] × 32	[1 × 1 conv, 3 × 3 conv] × 32	[1 × 1 conv, 3 × 3 conv] × 48
Classification Layer	1 × 1	7 × 7 global average pool			
		1000D fully-connected, softmax			

Một số ưu điểm của Densenet:

- Accuracy : Densenet training tham số ít hơn 1 nửa so với Resnet nhưng có same accuracy so trên ImageNet classification dataset.
- Overfitting : DenseNet resistance overfitting rất hiệu quả.
- Giảm được vashing gradient.
- Sử dụng lại feature hiệu quả hơn.

Ứng dụng của CNN

Mặc dù CNN chủ yếu được sử dụng cho các vấn đề về computer vision, nhưng điều quan trọng là đề cập đến khả năng giải quyết các vấn đề học tập khác của họ, chủ yếu liên quan đến tích chuỗi dữ liệu. Ví dụ: CNN đã được biết là hoạt động tốt trên chuỗi văn bản, âm thanh và video, đôi khi kết hợp với các mạng khác quả cầu kiến trúc hoặc bằng cách chuyển đổi các chuỗi thành hình ảnh có thể được xử lý của CNN. Một số vấn đề dữ liệu cụ thể có thể được giải quyết bằng cách sử dụng CNN với chuỗi dữ liệu là các bản dịch văn bản bằng máy, xử lý ngôn ngữ tự nhiên và gắn thẻ khung video, trong số nhiều người khác.

Classification: Đây là nhiệm vụ được biết đến nhiều nhất trong computer vision. Ý tưởng chính là phân loại nội dung chung của hình ảnh thành một tập hợp các danh mục, được gọi là nhãn. Ví dụ: phân loại có thể xác định xem một hình ảnh có phải là của một con chó, một con mèo hay bất kỳ động vật khác. Việc phân loại này được thực hiện bằng cách xuất ra xác suất của hình ảnh thuộc từng lớp.

Localization: Mục đích chính của localization là tạo ra một bounding box mô tả vị trí của đối tượng trong hình ảnh. Đầu ra bao gồm một nhãn lớp và một bounding box. Tác vụ này có thể được sử dụng trong cảm biến để xác định xem một đối tượng ở bên trái hay bên phải của màn hình.

Detection: Nhiệm vụ này bao gồm thực hiện localization trên tất cả các đối tượng trong ảnh. Các đầu ra bao gồm nhiều bounding box, cũng như nhiều nhãn lớp (một cho mỗi hộp). Nhiệm vụ này được sử dụng trong việc chế tạo ô tô tự lái, với mục tiêu là có thể xác định vị trí các biển báo giao thông, đường, ô tô khác, người đi bộ và bất kỳ đối tượng nào khác có thể phù hợp để đảm bảo trải nghiệm lái xe an toàn.

Segmentation: Nhiệm vụ ở đây là xuất ra cả nhãn lớp và đường viền của mỗi đối tượng hiện diện trong hình ảnh. Điều này chủ yếu được sử dụng để đánh dấu các đối tượng quan trọng của hình ảnh cho phân tích sâu hơn. Ví dụ: tác vụ này có thể được sử dụng để phân định rõ ràng khu vực tương ứng với khối u trong hình ảnh phổi của bệnh nhân. Hình sau mô tả cách vật thể quan tâm được phác thảo và gán nhãn.

Chương 3. GIẢI QUYẾT BÀI TOÁN

1. Hướng tiếp cận

Để nhằm mục đích đánh giá về tính hiệu quả trong thực tế của một số model khi sử dụng trong việc phân loại cảm xúc, nhóm quyết định tiếp cận vài toán theo 4 hướng là sử dụng SVM, tự build model CNN, VGG16 và VGG19.

VGG

AlexNet được ra đời vào năm 2012 và nó được cải tiến từ các convolutional neural network truyền thống, vì vậy chúng ta có thể hiểu VGG là sản phẩm kế thừa của AlexNet nhưng nó được tạo nên bởi một group khác với tên gọi là Visual Geometry Group ở Oxford, do đó nó có tên VGG. Nó mang và sử dụng một số ý tưởng từ những sản phẩm tiền nhiệm của nó và cải thiện chúng đồng thời nó sử dụng deep convolutional neural layers để cải thiện độ chính xác.

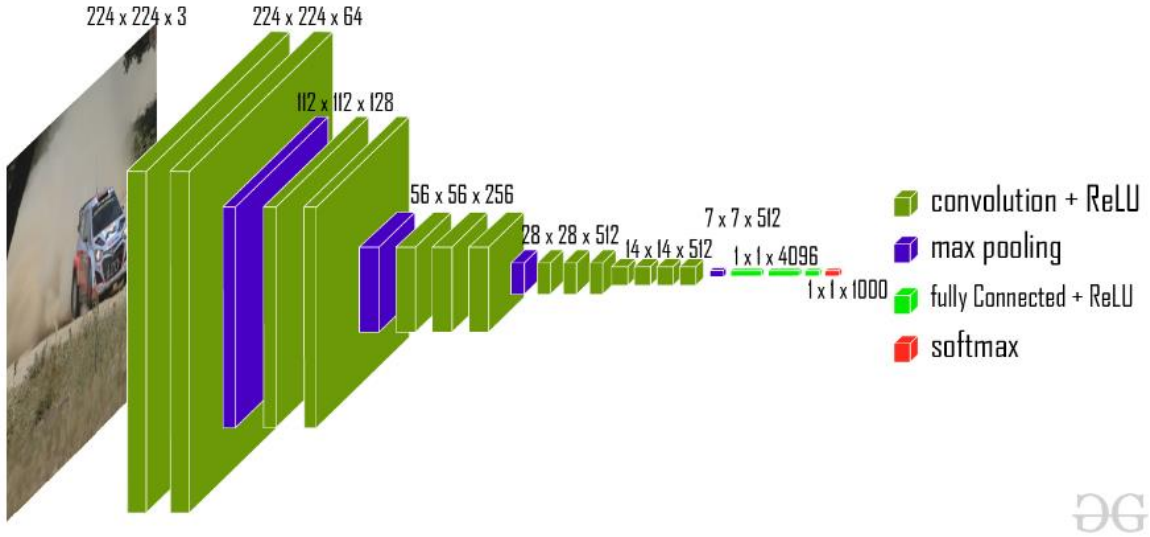
Cấu trúc VGG

Đầu vào của mạng là hình ảnh có kích thước (224,224,3). Hai lớp đầu tiên có 64 channels với bộ lọc có kích thước 3*3 và cùng một padding. Sau đó một lớp max pool với stride (2,2), hai convolutinal layers có 128 bộ lọc với kích thước (3,3). Tiếp theo là lớp max-pooling với stride (2,2) giống như lớp trước đó. Sau đó, có 2 convolutional layers với kích thước bộ lọc (3,3) và 256 bộ lọc. Sau đó có 2 bộ với 3 convolutional layers và max pool layer. Mỗi bộ lọc trong 512 bộ lọc có kích thước (3,3) với cùng một padding. Hình ảnh này sau đó được chuyển đến stack của convolutional layer. Trong các convolutional layer và max-pooling layer, các bộ lọc được sử dụng với kích thước 3*3 thay vì 11*11 như trong AlexNet và 7*7 trong ZF-Net. Trong một số lớp, nó cũng sử dụng pixel 1*1, được sử dụng để thao tác với số channels đầu vào. Có một padding với 1 pixel (same padding) được thực hiện sau mỗi convolutional layer để ngăn những đặc trưng không gian của hình ảnh.

VGG-16

ImageNet Large Scale Visual Recognition Challenge (ILSVRC) là một cuộc thi hàng năm của lĩnh vực thị giác máy tính. Mỗi năm, các đội cạnh tranh dựa trên 2 nhiệm vụ. Nhiệm vụ thứ nhất là nhận diện các đối tượng trong một hình ảnh với 200 class, được gọi là object localization. Nhiệm vụ thứ 2 là phân lớp ảnh, mỗi hình ảnh được gán nhãn một trong 1000 loại nhãn, được gọi là image classification. VGG-16 là mạng CNN được đề xuất bởi Karen Simonyan và Andrew Zisserman của Visual Goemetry Group Lab của trường đại học Oxford vào năm 2014 trong tờ báo có tiêu đề “VERY DEEP CONVOLUTIONAL NETWORKS FOR LARGE-SCALE IMAGE

RECOGNITION". Model này đã đạt được hạng nhất và hạng hai trong các danh mục của cuộc thi ILSVRC năm 2014.



Cấu trúc VGG-16

Model sau khi train bởi mạng VGG-16 đạt độ chính xác 92.7% top-5 test trong bộ dữ liệu ImageNet gồm 14 triệu hình ảnh thuộc 1000 lớp khác nhau.

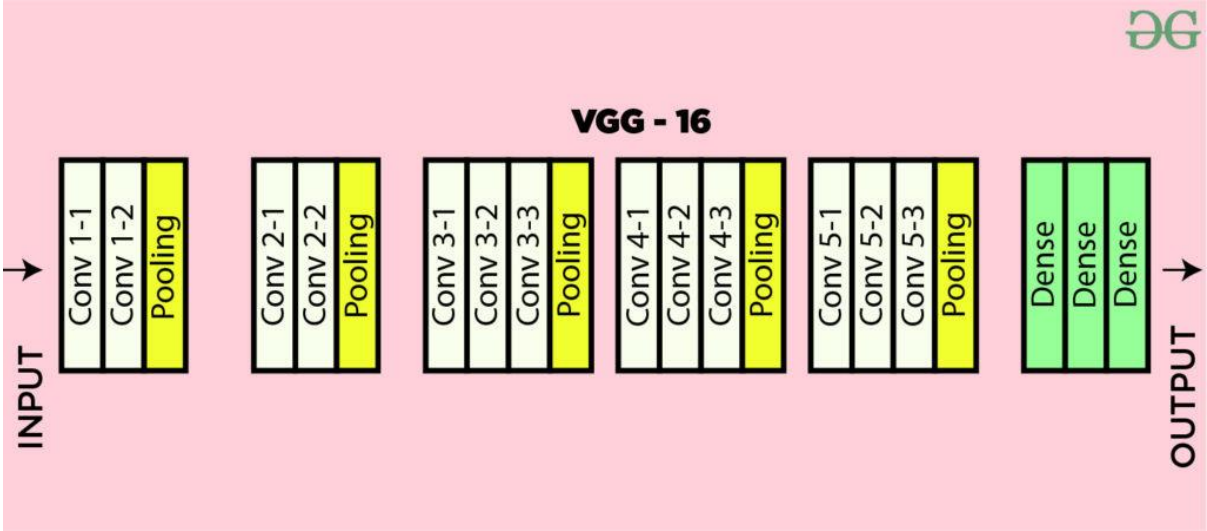
Cấu trúc VGG-16

conv: convolutional layer, pool: pooling layer, fc: fully connected layer

Phân tích:

- Convolutional layer: kích thước 3*3, padding=1, stride=1. Tại sao không ghi stride, padding mà vẫn biết? Vì mặc định sẽ là stride=1 và padding để cho output cùng width và height với input.
- Pool/2 : max pooling layer với size 2*2
- 3*3 conv, 64: thì 64 là số kernel áp dụng trong layer đấy, hay depth của output của layer đấy.
- Càng các convolutional layer sau thì kích thước width, height càng giảm nhưng depth càng tăng.

- Sau khá nhiều convolutional layer và pooling layer thì dữ liệu được flatten và cho vào fully connected layer.



Biểu đồ cấu trúc VGG-16

Cấu trúc VGG-16

- 16 trong VGG16 đề cập đến 16 lớp có trọng số. trong VGG16 có 13 convolutional layers, 5 max pooling layers và 3 dến layers, đúng ra nó có tất cả 21 lớp nhưng chỉ có 16 lớp trọng số, tức là lớp các tham số có thể học được.
- VGG16 lấy kích thước tensor đầu vào 224,224 với 3 RGB channel.
- Điều đặc biệt nhất của VGG16 là thay vì có một số lượng lớn các siêu tham số thì nó tập trung vào việc có các convolution layers của bộ lọc 3*3 với stride = 1 và luôn luôn sử dụng same padding và maxpool layer của bộ lọc 2*2 với stride = 2.
- Convolution và max pool layers được sắp xếp nhất quán trong toàn bộ cấu trúc.
- Conv-1 layer có 64 bộ lọc, Conv-2 có 128 bộ lọc, Conv-3 có 256 bộ lọc, Conv-4 và Còn-5 có 512 bộ lọc.
- 3 fully connected layer tuân theo một stack của convolutional layer: 2 lớp đầu tiên mỗi lớp có 4096 chanels, lớp thứ 3 thực hiện phân loại ILSVRC 1000 chiều vì vậy chứa 1000 chanels (một channel cho một class). Lớp cuối cùng là soft-max layer.

Cấu hình

Bảng bên dưới liệt kê các cấu trúc VGG khác nhau. Chúng ta có thể thấy rằng có 2 phiên bản VGG-16 (C và D). Không có nhiều điểm khác biệt giữa chúng ngoại trừ một số convolutional layer, convolutional sử dụng bộ lọc có kích thước (3,3) thay vì (1,1). Hai cái này chứa lần lượt 134 triệu và 138 triệu tham số tương ứng.

ConvNet Configuration					
A	A-LRN	B	C	D	E
11 weight layers	11 weight layers	13 weight layers	16 weight layers	16 weight layers	19 weight layers
input (224 × 224 RGB image)					
conv3-64	conv3-64 **LRN**	conv3-64 **conv3-64**	conv3-64 conv3-64	conv3-64 conv3-64	conv3-64 conv3-64
maxpool					
conv3-128	conv3-128	conv3-128 **conv3-128**	conv3-128 conv3-128	conv3-128 conv3-128	conv3-128 conv3-128
maxpool					
conv3-256 conv3-256	conv3-256 conv3-256	conv3-256 conv3-256	conv3-256 conv3-256 **conv1-256**	conv3-256 conv3-256 **conv3-256**	conv3-256 conv3-256 conv3-256 **conv3-256**
maxpool					
conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512 **conv1-512**	conv3-512 conv3-512 **conv3-512**	conv3-512 conv3-512 conv3-512 **conv3-512**
maxpool					
conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512 **conv1-512**	conv3-512 conv3-512 **conv3-512**	conv3-512 conv3-512 conv3-512 **conv3-512**
maxpool					
FC-4096					
FC-4096					
FC-1000					
soft-max					

Các cấu hình VGG

Hạn chế của VGG-16

- Quá trình train rất chậm (model VGG ban đầu được train trên Nvidia Titan GPU trong 2-3 tuần).
- Kích thước file ImageNet Weight dùng để train cho VGG-16 có dung lượng lên đến 528MB. Vì vậy nó chiếm khá nhiều dung lượng ổ đĩa và băng thông khiến nó kém hiệu quả.
- 138 triệu tham số dẫn đến sự cố exploding gradients.

VGG-19

VGG19 được phát triển dựa trên VGG16, bao gồm 16 convolutional neural networks layers và 3 fully connected layers và lớp cuối cùng là hàm softmax, fully connected layers và lớp cuối cùng sẽ giữ nguyên cho tất cả các cấu trúc mạng.

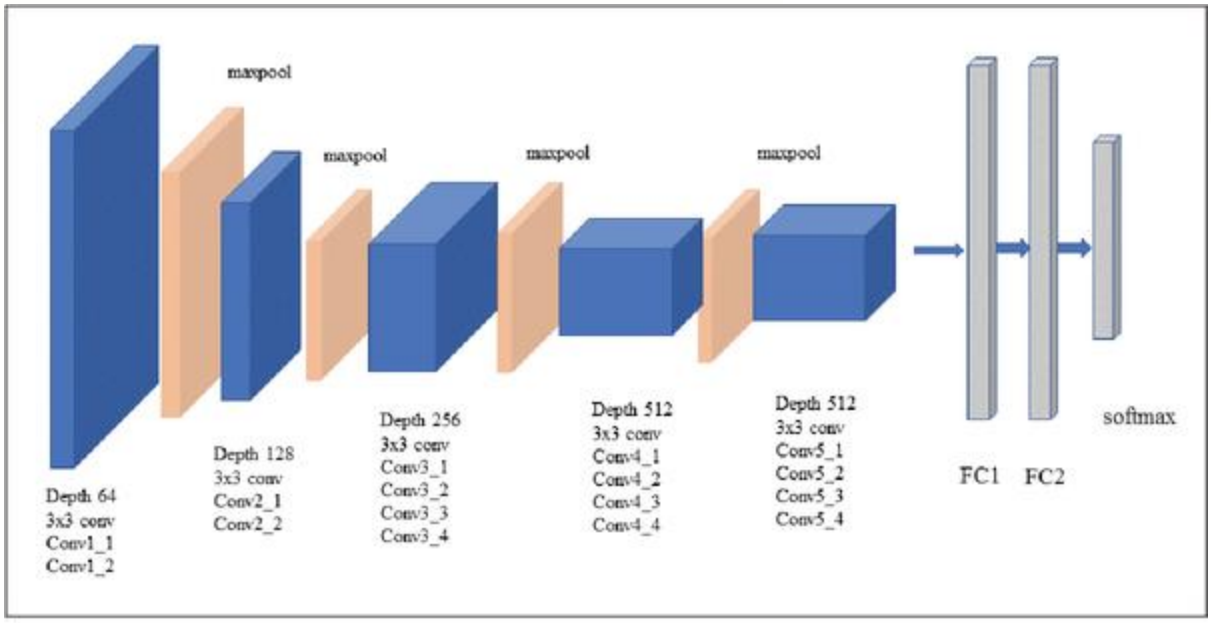


Fig. 3. VGG-19 network architecture

Cấu trúc VGG19

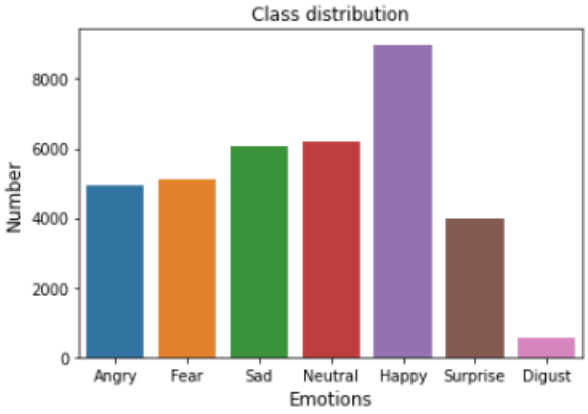
Cấu trúc VGG-19

- Hình ảnh RGB có kích thước cố định (224*224) được cung cấp làm đầu vào cho mạng này, điều đó cũng có ý nghĩa rằng ma trận có hình dạng (224,224,3).
- Quá trình tiền xử lý duy nhất đã được thực hiện là chúng đã trừ giá trị trung bình RBG từ mỗi pixel, được tính toán trên toàn bộ tập huấn luyện.
- Sử dụng kernel có kích thước (3*3) với một stride có kích thước 1 pixel, điều này cho phép chúng bao phủ toàn bộ hình ảnh.
- Spatial padding được sử dụng để duy trì spatial resolution của hình ảnh.
- Max pooling được thực hiện trên cửa sổ pixel 2*2 với stride = 2.
- Điều này được tuân theo Rectified linear unit (RELU) để giới thiệu tính phi tuyến tính để làm cho model phân lớp tốt hơn và cải thiện thời gian tính toán trước vì các model trước đó đã sử dụng hàm tanh hoặc hàm sigmoid, điều này cho thấy tốt hơn nhiều so với các mô hình đó.
- Để thực hiện 3 fullu connected layers, trong đó 2 lớp đầu tiên có kích thước 4096 và lớp sau đó có 1000 channels để phân loại ILSVRC 1000 chiều và lớp cuối cùng là một hàm softmax.

2. Dataset

Nhóm sẽ sử dụng bộ data FER2013 được lấy từ kaggle.
Bộ data gồm có 35000+ ảnh với 7 class.

- Angry: 4953 ảnh
- Fear: 5121 ảnh
- Sad 6077 ảnh
- Neutral: 6198 ảnh
- Happy: 8989 ảnh
- Surprise: 4002 ảnh
- Disgust: 547 ảnh



Sơ đồ số lượng ảnh trong mỗi class

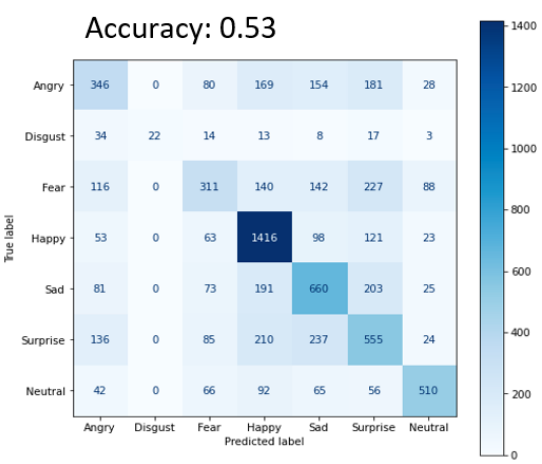
Một số hình ảnh trong bộ dataset và class của chúng

Scan me!

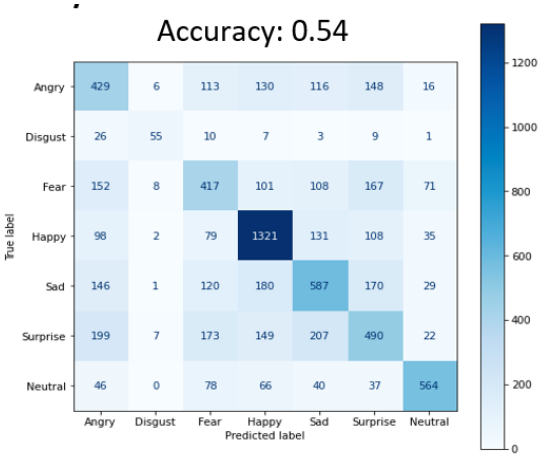
Đường dẫn đến dataset

3. Đánh giá kết quả train + test

SVM



RBF



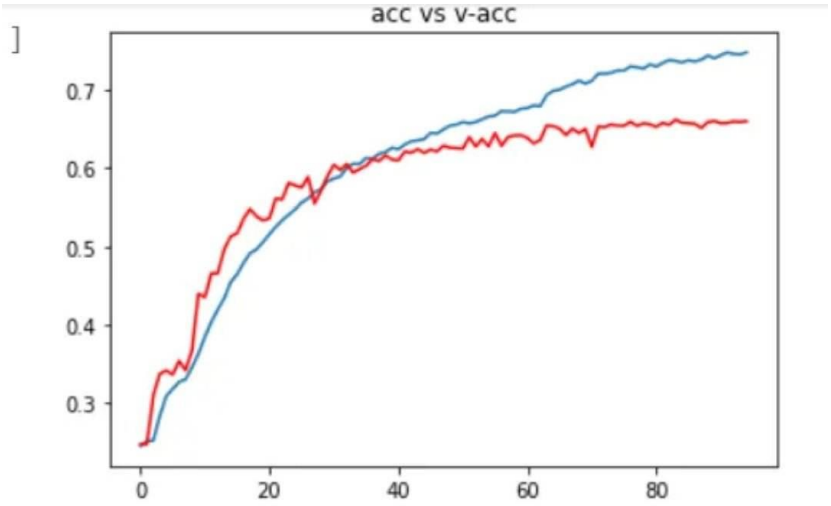
Poly

Qua confusion matrix trên so với số lượng bức ảnh có trong mỗi class với kết quả dự đoán được trong mỗi class thì có thể thấy rằng class happy và class neutral là 2 class được predict tốt nhất. Mặc dù class happy có số lượng ảnh trong bộ dữ liệu nhất đồng thời nó là class dự đoán đúng nhất nhưng nó phải là class được dự đoán ra nhiều nhất từ những class khác, ở những kết quả dự đoán sai của các class khác thiên về 2 class happy, sad và surprise là nhiều nhất. Còn đối với class disgust là class có số lượng ảnh trong bộ dữ liệu ít nhất ngoài những hình ảnh được dự

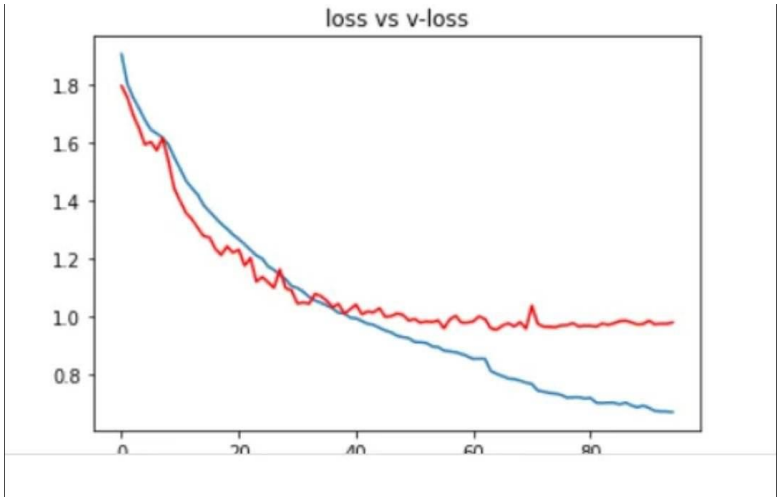
đoán đúng thì những hình ảnh dự đoán sai được dự đoán thành class angry rất nhiều, sở dĩ điều này xảy ra cũng có thể trong bộ dữ liệu những hình ảnh của class disgust và class angry khá giống nhau. Đối với khi sử dụng RBF, với true class là disgust, số lượng class disgust được dự đoán đúng không chiếm ưu thế mà số lượng class angry lại chiếm ưu thế.

CNN

Train

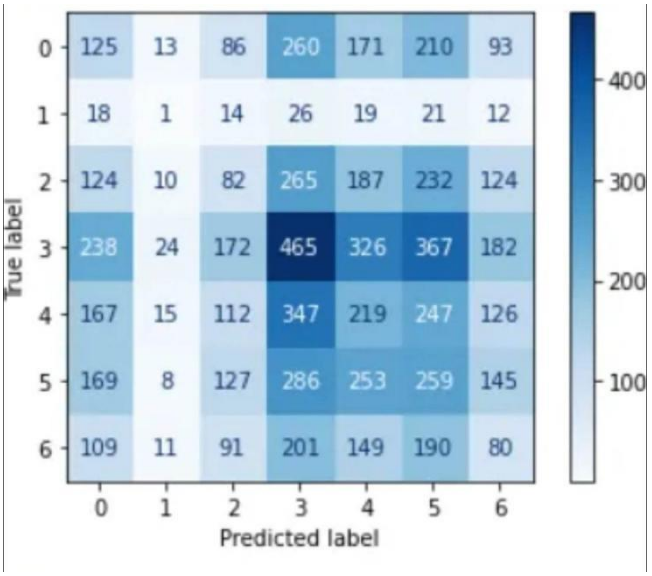


Khi cho mô hình train với epoch bằng 500 (để mô hình có thể train tới khi nào đạt được kết quả tốt nhất và bị dừng bởi early stopping của Keras nhằm tránh bị Overfitting)với bộ dữ liệu fer2013 (file ảnh), model cho ra kết quả train tương đối tốt với accuracy xấp xỉ 75% và val-accuracy xấp xỉ 65%.



Tương tự với train-loss và val-loss cho thấy giá giảm dần đều theo thời gian. Nhưng so với train-loss thì val-loss cao hơn và đồng thời có xu hướng tăng lên nếu quá trình train vẫn tiếp tục được thực hiện.

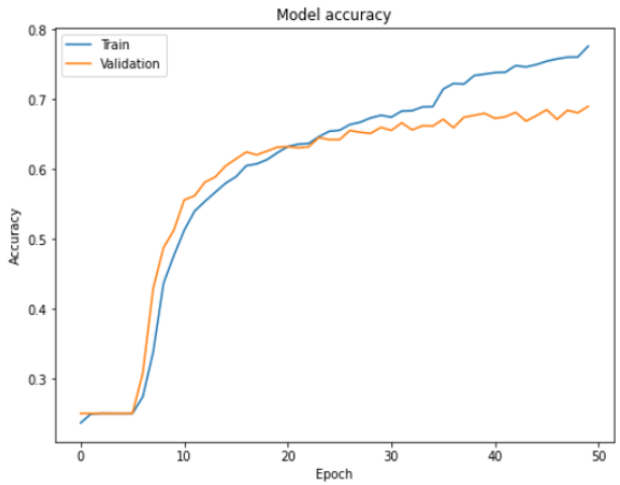
Test



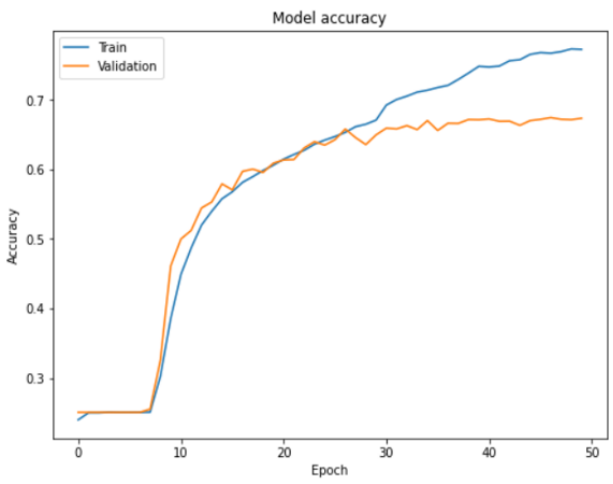
Qua confusion matrix đã cho thấy rằng class happy được dự đoán đúng nhiều nhất. Ngược lại class surprise và fear lại bị nhầm lẫn nhiều nhất, tiếp sau đó là happy và sad. Ngoài ra, các dữ kiện bị nhầm lẫn khá nhiều, các cảm xúc có hình ảnh gần giống nhau sẽ bị model hiểu lầm và phân bổ sai. Từ đó dẫn đến confusion matrix không được đẹp và có cảm giác rối rắm.

VGG

Train

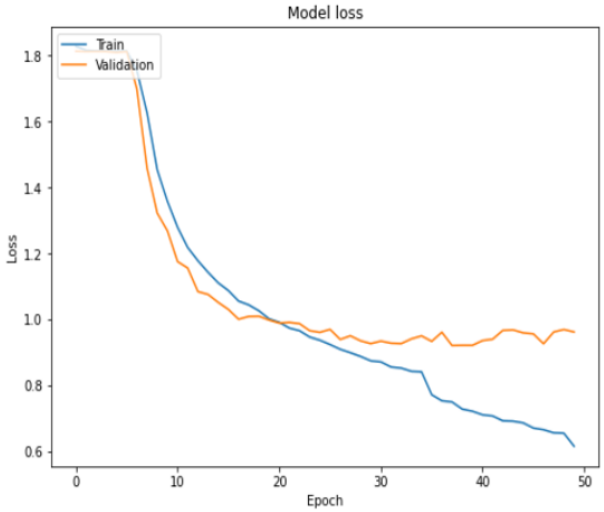


VGG16

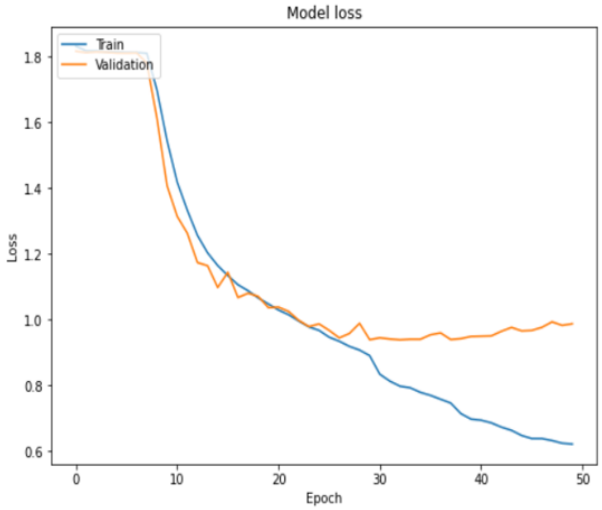


VGG19

Khi train 2 model VGG16 và VGG19 đồng thời với 50 epoch với bộ dữ liệu fer2013 (file .csv), cả 2 model có độ chính xác (accuracy) trên tập train tương đối cao (xấp xỉ 80%) và độ chính xác trên tập val thấp hơn (xấp xỉ 80 %). Nhưng khi so sánh độ chính xác khi train model trên 2 tập dữ liệu train và val thì thấy rằng model VGG16 tốt hơn so với VGG19 mặc dù không đáng kể, đồng thời trên tập val, độ chính xác khi sử dụng model VGG16 tăng ổn định hơn so với model VGG19, ở model VGG19 độ chính xác trên tập val có một số lần giảm nhiều hơn rất nhiều so với model VGG16. Qua đó khi đánh giá model với độ chính xác khi train thì có thể thấy rằng cả 2 model đều khá tốt.



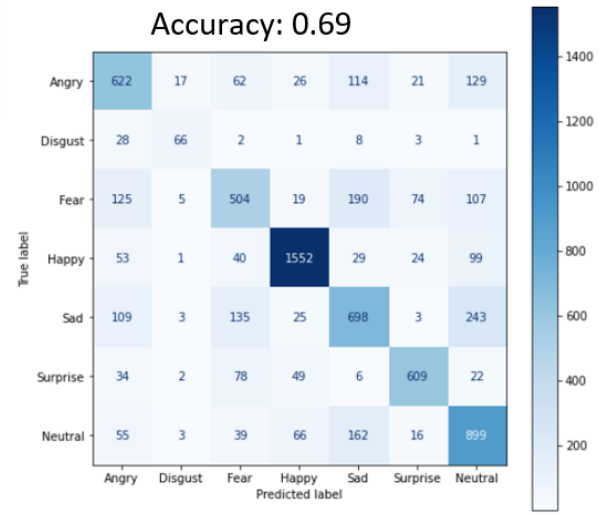
VGG16



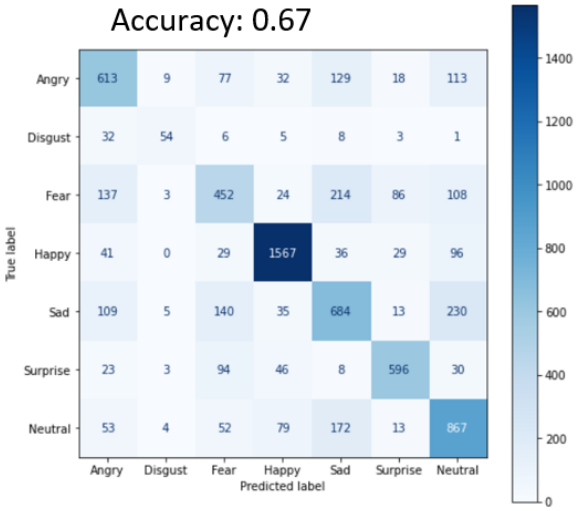
VGG19

Đối với giá trị loss trên tập train và tập val của cả 2 model, 2 hình trên cho thấy rằng giá trị loss trên tập train (xấp xỉ 0.6) tốt hơn so với trên tập val (xấp xỉ 1) đồng thời với cả 2 model, loss trên tập train vẫn còn có xu hướng giảm nhưng ở trên tập val thì ngược lại, loss hầu như không giảm mà còn có xu hướng tăng lên.

Test



VGG16



VGG19

Qua confusion matrix trên so với số lượng bức ảnh có trong mỗi class với kết quả dự đoán được trong mỗi class thì có thể thấy rằng class happy và class surprise là 2 class được predict tốt nhất. Mặc dù class happy có số lượng ảnh trong bộ dữ liệu nhất đồng thời nó là class dự đoán đúng nhất nhưng nó không phải là class được dự đoán ra nhiều nhất từ những class khác, ở những kết quả dự đoán sai của các class khác thiên về 2 class sad và neutral là nhiều nhất. Còn đối với class disgust là class có số lượng ảnh trong bộ dữ liệu ít nhất ngoài những hình ảnh được dự đoán đúng thì những hình ảnh dự đoán sai được dự đoán thành class angry rất nhiều, sở dĩ điều này xảy ra cũng có thể trong bộ dữ liệu những hình ảnh của class disgust và class angry khá giống nhau.

Đồng thời qua độ chính xác (Accuracy) thì dễ thấy trong bài toán này model VGG16 tốt hơn so với model VGG19.

4. Đánh giá mô hình với bộ dữ liệu nằm ngoài bộ dữ liệu đã được train

	True_labels	Predict_rbf_labels(SVM)	Predict_poly_labels(SVM)	Predict_labels_CNN	Predict_labels_VGG16	Predict_labels_VGG19
0	Angry	Angry	Angry	Angry	Angry	Angry
1	Angry	Angry	Angry	Angry	Angry	Neutral
2	Angry	Angry	Fear	Angry	Angry	Sad
3	Disgust	Fear	Fear	Fear	Fear	Neutral
4	Disgust	Neutral	Neutral	Disgust	Neutral	Sad
5	Disgust	Neutral	Disgust	Disgust	Angry	Happy
6	Fear	Sad	Disgust	Fear	Fear	Sad
7	Fear	Sad	Disgust	Fear	Fear	Fear
8	Fear	Sad	Disgust	Fear	Fear	Fear
9	Happy	Sad	Sad	Happy	Disgust	Happy
10	Happy	Happy	Happy	Happy	Happy	Happy
11	Happy	Happy	Happy	Happy	Angry	Happy
12	Neutral	Neutral	Neutral	Sad	Surprise	Neutral
13	Neutral	Neutral	Neutral	Sad	Neutral	Neutral
14	Neutral	Happy	Happy	Surprise	Happy	Happy
15	Sad	Sad	Happy	Surprise	Fear	Sad
16	Sad	Surprise	Happy	Surprise	Fear	Happy
17	Sad	Fear	Happy	Happy	Happy	Happy
18	Surprise	Surprise	Surprise	Neutral	Fear	Fear
19	Surprise	Surprise	Surprise	Neutral	Surprise	Surprise
20	Surprise	Surprise	Surprise	Neutral	Surprise	Surprise

Kết quả thử nghiệm mô hình

	RBF(SVM)	Poly(SVM)	VNN	VGG16	VGG19
Accuracy với bộ datatest	0.53	0.54	0.65	0.69	0.67
Số nhãn đoán đúng khi sử dụng với bộ dataset bên ngoài	11	10	11	10	11
Accuracy với bộ data bên ngoài	0.52	0.48	0.52	0.48	0.52

Qua bảng trên có thể thấy kết quả dự đoán của mô hình khi sử dụng mạng neural chính xác hơn so với khi sử dụng SVM. Cao nhất là khi sử dung VGG16 và thấp nhất là khi sử dụng RBF của SVM. Nhưng khi sử dụng để dự đoán dữ liệu thực tế (dữ liệu nằm ngoài bộ dataset sử dụng để train) thì kết quả hầu như ngang nhau, bên cạnh đó kết quả mô hình dự đoán được khi sử dụng dữ liệu thực tế thấp hơn khá nhiều so với kết quả thử nghiệm của mô hình với bộ dữ liệu test. Sở dĩ kết quả khi test trên dữ liệu thực tế thấp hơn khi test trên bộ dữ liệu test nhiều đến như vậy một phần cũng có thể do bộ dữ liệu dùng để train có điểm tương đồng lớn trong các class, đồng thời cảm xúc trên khuôn mặt của con người đối với chúng ta cũng không phải dễ phân biệt cho nên lúc train xong, khả năng phân biệt các đặc trưng trên khuôn mặt của các class cũng sẽ không cao nên khi nhận diện luôn những hình ảnh đã có sẵn trong bộ dữ liệu sở dĩ kết quả cũng không khả quan cho lắm thì lúc test trên dữ liệu thực tế, kết quả chắc chắn sẽ không khả quan cho lắm, cụ thể ở bài toán lần này kết quả khi test trên dữ liệu thực tế thấp hơn khi test trên tập test. Nhưng nói về độ chênh lệch giữa 2 kết quả trên thì khi sử dụng mạng neural chênh lệch nhiều hơn so với khi sử dụng SVM, cũng có thể khẳng định trong trường hợp này sử dụng SVM trên thực tế hiệu quả hơn khi sử dụng mạng neural. Tại vì SVM sẽ dự đoán nhãn của hình ảnh input bằng cách thức là so sánh đặc trưng của input với đặc trưng của kết quả train để dự đoán lớp của hình ảnh, nhưng đối với mạng neural thì nó sẽ sử dụng kiến thức đã được học từ tập train để dự đoán kết quả cho hình ảnh mới.

Chương 4: KẾT LUẬN

1. Nhận xét về mô hình

- Support vector machine (SVM) sử dụng một bộ phân lớp tuyến tính để phân loại các điểm dữ liệu, trong trường hợp này là hình ảnh, bằng cách so sánh đặc trưng của chúng với đặc trưng của các điểm dữ liệu huấn luyện được sử dụng để xây dựng mô hình SVM. Khi một hình ảnh mới được đưa vào, SVM sẽ tính toán đặc trưng của nó và sử dụng mô hình đã được huấn luyện để dự đoán lớp tương ứng. Vì vậy độ chính xác khi test mô hình với bộ datatest được chia ra từ bộ dữ liệu trong quá trình huấn luyện mô hình so với khi test mô hình với bộ data hoàn toàn nằm ngoài bộ dữ liệu ban đầu dùng để huấn luyện mô hình sẽ không chênh lệch nhiều.
- Convolution neural network (CNN) là một kiến trúc mạng neural nhân tạo được thiết kế đặc biệt để xử lý dữ liệu hình ảnh. CNN sử dụng một loạt các lớp, bao gồm lớp tích chập, lớp phi tuyến và lớp gộp, để học các đặc trưng từ hình ảnh và dự đoán lớp tương ứng. Trong quá trình huấn luyện, CNN sẽ tự động học cách trích xuất đặc trưng từ các hình ảnh trong tập dữ liệu huấn luyện. Sau đó, khi một hình ảnh mới được đưa vào, CNN sẽ tính toán đặc trưng của nó và sử dụng mô hình đã được huấn luyện để dự đoán lớp tương ứng. Vì vậy độ chính xác khi test mô hình với bộ datatest được chia ra từ bộ dữ liệu trong quá trình huấn luyện mô hình so với khi test mô hình với bộ data hoàn toàn nằm ngoài bộ dữ liệu ban đầu dùng để huấn luyện mô hình sẽ ít chênh lệch nếu các lớp của dữ liệu có đặc trưng khắc nhau nhiều hoặc là hoàn toàn khác nhau, lúc đó mô hình đưa ra sử dụng ở thực tế sẽ tốt hơn, còn nếu đặc trưng của các lớp trong dữ liệu tương đương nhau, không khác nhau nhiều thì lúc đó test mô hình với bộ data hoàn toàn nằm ngoài bộ dữ liệu ban đầu sẽ chênh lệch nhiều so với khi test với bộ datatest được chia ra từ bộ dữ liệu huấn luyện ban đầu, lúc này nếu đưa mô hình ra sử dụng ở ngoài thực tế sẽ kém hiệu quả hơn.
- Cả mô hình dự đoán lớp cho hình ảnh khi sử dụng SVM hay sử dụng CNN đều phụ thuộc rất nhiều vào bộ dữ liệu dùng để huấn luyện, nếu các lớp có trong bộ dữ liệu dùng để huấn luyện có đặc trưng khác nhau hoặc có thể là hoàn toàn khác nhau thì lúc đó mô hình sẽ dự đoán đúng hơn và hiệu quả hơn.

2. Bài học kinh nghiệm

Sau khi hoàn thành xong đề tài này, các thành viên trong nhóm đã đều có khả năng trả lời các câu hỏi liên quan tới mô hình phân loại cảm xúc trên khuôn mặt người sử dụng SVM hoặc CNN. Bài báo cáo đi sát với toàn bộ nội dung thành viên đã thực hiện cũng như sưu tầm được những kiến thức cần phải phân tích rõ ràng. Trong quá trình nghiên cứu và tìm hiểu, nhóm đã thu được nững kết quả thực nghiệm hữu ích thông qua việc sử dụng nhiều mô hình để áp dụng cho đề tài này. Qua việc đánh giá kết quả thực nghiệm, nhóm cũng đã thu được một số kinh nghiệm quý báu từ mô hình.

Tài liệu tham khảo

Tiếng Việt

1. https://bkaii.com.vn/tin-tuc/795-thi-giac-may-tinh-la-gi-su-khac-nhau-giua-thi-giac-may-va-thi-giac-may-tinh
2. https://bkaii.com.vn/tin-tuc/798-nhung-ung-dung-thuc-tien-cua-thi-giac-may-tinh
3. https://thigiacmay.com/nhung-ung-dung-hieu-qua-cua-thi-giac-may-tinh-computer-vision-vao-thuc-tien/
4. https://www.thegioimaychu.vn/blog/ai-hpc/computer-vision-thi-giac-may-tinh-la-gi-p5951/#:~:text=Nh%E1%BB%AFng%20h%E1%BA%A1n%20ch%E1%BA%BF%20c%E1%BB%A7a%20Th%E1%BB%8B%20gi%C3%A1c%20M%C3%A1y%20t%C3%ADnh&text=Ch%C3%BAng%20kh%C3%B4ng%20hi%E1%BB%83u%20nh%E1%BB%AFng%20g%C3%AC,%E2%80%8Bth%E1%BB%A9c%20c%C6%A1%20b%E1%BA%A3n%20chung
5. https://vinbigdata.com/phan-loai-hinh-anh-trong-thi-giac-may-tinh/
6. https://dlapplications.github.io/2018-07-06-CNN/

Tiếng Anh

1. https://www.geeksforgeeks.org/vgg-16-cnn-model/
2. https://www.pycodemates.com/2022/10/svm-kernels-polynomial-kernel.html
3. https://www.pycodemates.com/2022/10/the-rbf-kernel-in-svm-complete-guide.html