BỘ GIÁO DỤC VÀ ĐÀO TẠO

TRƯỜNG ĐẠI HỌC QUẢN LÝ VÀ CÔNG NGHỆ HẢI PHÒNG

ĐỒ ÁN TỐT NGHIỆP

NGÀNH: CÔNG NGHỆ THÔNG TIN

Sinh viên : Phạm Thị Ngọc Anh

Giảng viên hướng dẫn: TS. Ngô Trường Giang

Hải Phòng -2023

BỘ GIÁO DỤC VÀ ĐÀO TẠO

TRƯỜNG ĐẠI HỌC QUẢN LÝ VÀ CÔNG NGHỆ HẢI PHÒNG

ĐỀ TÀI : PHÂN ĐOẠN NGỮ NGHĨA SỬ DỤNG MẠNG NƠ-RON TÍCH CHẬP

ĐỒ ÁN TỐT NGHIỆP ĐẠI HỌC HỆ CHÍNH QUY

NGÀNH : CÔNG NGHỆ THÔNG TIN

Sinh viên : Phạm Thị Ngọc Anh

Giảng viên hướng dẫn: TS. Ngô Trường Giang

Hải Phòng – 2023

BỘ GIÁO DỤC VÀ ĐÀO TẠO

TRƯỜNG ĐẠI HỌC QUẢN LÝ VÀ CÔNG NGHỆ HẢI PHÒNG

NHIỆM VỤ ĐỀ TÀI TỐT NGHIỆP

Sinh viên : Phạm Thị Ngọc Anh - **MSV** : 1912111012

Lớp : CT2301M

Ngành : Công nghệ thông tin

Tên đề tài : Phân đoạn ngữ nghĩa sử dụng mạng nơ-ron tích chập

LỜI CẢM ƠN

Trong quá trình làm đồ án vừa qua vì được sự chỉ dẫn nhiệt tình của thầy TS. Ngô Trường Giang – Trường Đại học Quản lý và Công nghệ Hải Phòng, em đã hoàn thành đồ án của mình. Mặc dù em đã cố gắng với sự tận tâm của thầy, nhưng vì thời gian và khả năng nên đồ án của em vẫn còn không tránh được những điều thiếu sót.

Em xin chân thành và bày tỏ lòng biết ơn sâu sắc đến thầy Ngô Trường Giang vì đã tận tình chỉ bảo, hướng dẫn và giành thời gian quý báu của mình cho em trong thời gian qua để em có thể hoàn thành đồ án của mình đúng thời hạn.

Em xin cảm ơn tất cả thầy cô giáo trong khoa Công nghệ thông tin vì đã truyền đạt cho em rất nhiều các kiến thức nền tảng, chuyên ngành, chuyên môn và chuyên sâu cực kì vững chắc trong những năm qua để em có thể hoàn thành được đồ án này.

Em xin cảm ơn Trường Đại học Quản lý và Công nghệ Hải Phòng vì không ngừng hỗ trợ và đào tạo những điều kiện tốt nhất trong những năm vừa qua để em có thể học và thực hiện tốt đồ án.

Em xin cảm ơn gia đình, bạn bè đã hỗ trợ và cổ vũ cho em trong suốt quá trình học tập cũng như làm đồ án để em có thể hoàn thành khoá học và đồ án theo quy định.

Em xin chân thành cảm ơn !

MỤC LỤC

DANH MỤC HÌNH VẼ

MỞ ĐẦU

Phân đoạn ngữ nghĩa (semantics segmentation) là bài toán gán nhãn đối tượng cho từng điểm ảnh và từ đó có thể phân biệt chính xác ảnh của đối tượng cần quan tâm so với ảnh của các đối tượng khác hoặc ảnh nền. Mục tiêu là phân tích một ảnh thành các đối tượng có ý nghĩa, để máy tính dễ dàng hiểu được các thông tin chứa trong ảnh.

CNN là từ viết tắt của cụm Convolutional Neural Network hay là mạng nơ ron tích chập. Đây là mô hình vô cùng tiên tiến được áp dụng nhiều trong lĩnh vực học sâu Deep learning. Mạng CNN cho phép người dùng xây dựng những hệ thống phân loại và dự đoán với độ chính xác cực cao. Hiện nay, mạng CNN được ứng dụng nhiều hơn trong xử lý ảnh, cụ thể là nhận diện đối tượng trong ảnh.

Tích chập là một khái niệm trong xử lý tín hiệu số nhằm biến đổi thông tin đầu vào thông qua một phép tích chập với bộ lọc để trả về đầu ra là một tín hiệu mới. Tín hiệu này sẽ làm giảm những đặc trưng mà bộ lọc không quan tâm và chỉ giữ những đặc trưng chính.

Mạng nơ ron tích chập chúng rất hữu ích trong việc phân loại hình ảnh vì chúng có thể trích xuất các đặc điểm liên quan từ hình ảnh, điều này có lợi cho việc phân loại và nhận dạng hình ảnh. Biểu mẫu mới dễ xử lý hơn mà không làm mất đi các đặc điểm quan trọng để đưa ra dự đoán chính xác.

Công nghệ CNN có khả năng phát triển mạnh mẽ trong tương lai. Đây là lý do em chọn đề tài “Phân đoạn ngữ nghĩa sử dụng mạng nơ ron tích chập ” để triển khai. Đề tài này sẽ tìm hiểu mô hình phân đoạn ngữ nghĩa ảnh sử dụng kiến trúc mạng nơ-ron tích chập sâu với các kỹ thuật liên quan và ứng dụng, nội dung của đồ án bao gồm:

Chương 1 : Tổng quan về phân loại ảnh

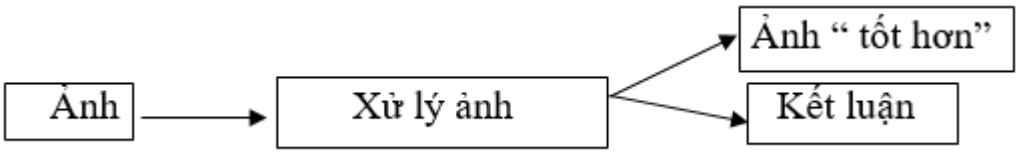
Chương 2 : Mạng nơ ron nhân chập

Chương 3 : Ứng dụng CNN cho phân đoạn ngữ nghĩa

CHƯƠNG 1 : TỔNG QUAN VỀ PHÂN ĐOẠN ẢNH

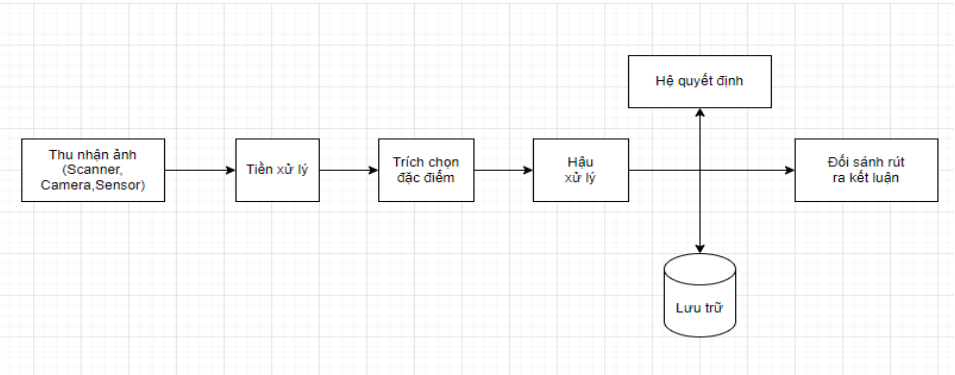
1.1 Xử lý ảnh là gì

Xử lý ảnh là quá trình khác thú vị khi biến đổi hình ảnh thông thường sang dạng kĩ thuật số, sau đó tận dụng để tạo ra những thông tin hữu ích. Để làm điều này, hệ thống xử lý ảnh thường xem mọi bức ảnh như một tín hiệu 2D và tiến hành các phép biến đổi dựa trên các kỹ thuật đã được xác định trước. Con người thu nhận thông tin qua các giác quan, trong đó thị giác đóng vai trò quan trọng nhất. Những năm trở lại đây với sự phát triển của phần cứng máy tính, xử lý ảnh và đồ hoạ đó phát triển một cách mạnh mẽ và có nhiều ứng dụng trong cuộc sống. Xử lý ảnh và đồ hoạ đóng một vai trò quan trọng trong tương tác người máy. Quá trình xử lý ảnh được xem như là quá trình thao tác ảnh đầu vào nhằm cho ra kết quả mong muốn. Kết quả đầu ra của một quá trình xử lý ảnh có thể là một ảnh "tốt hơn" hoặc một kết luận.



Hình 1.1 Quá trình xử lý ảnh

Ảnh có thể xem là tập hợp các điểm ảnh và mỗi điểm ảnh được xem như là đặc trưng cường độ sáng hay một dấu hiệu nào đó tại một vị trí nào đó của đối tượng trong không gian và nó có thể xem như một hàm n biến P(c1, c2,..., cn). Do đó, ảnh trong xử lý ảnh có thể xem như ảnh n chiều. Sơ đồ tổng quát của một hệ thống xử lý ảnh minh họa như sau:



Hình 1.2 Các bước cơ bản trong một hệ thống xử lý ảnh

1.2 Phân đoạn ảnh

Phân đoạn ảnh (hay còn gọi là phân vùng ảnh, tiếng Anh Image segmentation) là quá trình phân chia ảnh thành các vùng hoặc đối tượng có tính chất thoả mãn một tiêu chí xác định (có sự tương đồng về mức xám, kết cấu, màu sắc, v..v) Mức độ chi tiết của việc phân chia phụ thuộc vào từng bài toán cần giải quyết. Phân đoạn ảnh là một bài toán căn bản nhưng cũng rất phức tạp trong chuỗi xử lý và phân tích ảnh nói chung bởi sự đa dạng trong định nghĩa cũng như tính chất của vùng hoặc đối tượng quan tâm trong ảnh Thời kỳ đầu của thị giác máy tính, các giải thuật phân vùng ảnh chưa quan tâm nhiều đến ngữ nghĩa và thực thể đối tượng cần xem xét. Trong thời gian gần đây, phân vùng ảnh hướng đến phân tách ảnh thành các vùng, mỗi vùng có thể chứa một đối tượng hoặc các thực thể của một lớp đối tượng nào đó. Ba bài toán thường gặp trong quá trình phân vùng ảnh đó là:

- *Phát hiện đối tượng*: tìm bao đóng chứa các đối tượng quan tâm (vd. con người).
- *Phân vùng ngữ nghĩa (semantic segmentation)*: phân chia các điểm ảnh vào các lớp khác nhau. Các đối tượng cùng một lớp sẽ thuộc cùng một vùng (vd. mọi người trong ảnh thuộc một vùng).
- *Phân vùng thực thể (instance segmentation)*: phân chia các điểm ảnh vào các lớp nhưng các đối tượng của cùng một lớp thì có nhãn khác nhau. Như vậy mỗi thực thể của một lớp đối tượng sẽ được xác định là một vùng riêng biệt (vd. mỗi người trong ảnh là một vùng).

Phân vùng ảnh là bài toán được đề cập và giải quyết từ những năm 1970 trong các công bố của Brice và Fenema. Năm 1974 Watanabe đề xuất kỹ thuật phân vùng ảnh dựa trên *lấy ngưỡng*. Năm 1978 Jack Sklandsky đề xuất kỹ thuật phân vùng ảnh dựa trên *phát hiện biên*. Kỹ thuật *lan vùng* xác định trực tiếp vùng bằng cách lan vùng từ một vị trí trong ảnh cho đến khi nào tiêu chí vùng vẫn còn thỏa mãn do Brice và Fennema đề xuất năm 1970, sau đó được cải tiến

bởi Pavlidis và các cộng sự năm 1990, R. Adams and L. Bischof – 1994, Zugaj và cộng sự năm 1998; Hojjatoleslami, S. A. và Kittler, J – 1998.

Năm 1979, Coleman và Andrews giới thiệu kỹ thuật phân vùng ảnh dựa trên *phân cụm*. Kỹ thuật *Watershed* coi ảnh là một bề mặt topo, khi đó việc phân vùng được xem là thực hiện phép biến đổi watershed để tìm ra các vùng ảnh quan tâm. Kỹ thuật này lần đầu tiên được giới thiệu bởi Digabel và Lantu´ejoul vào năm 1978, tiếp tục được cải tiến bởi S Beucher 1992, V Grau và các cộng sự năm 2004.

Năm 2004, Rother và cộng sự đề xuất kỹ thuật *Graph cuts*. Năm 2006, Kato và Pong đề xuất kỹ thuật *Trường ngẫu nhiên markov có điều kiện*. Kỹ thuật này coi bài toán phân vùng là bài toán gán nhãn và đi tìm lời giải theo hướng tiếp cận xác suất.

Hiện nay, các giải thuật học sâu đã cho những kết quả ấn tượng trên rất nhiều bài toán liên quan đến phân tích và hiểu ảnh. Các giải thuật học sâu coi bài toán phân vùng ảnh là bài toán gán nhãn mức điểm ảnh và đưa ra các kết quả phân vùng thực thể ngữ nghĩa phục vụ cho nhiều bài toán sau đó. Năm 2015, Long và cộng sự đề xuất *Mạng tích chập đầy đủ (Fully Convolutional Networks)* cho bài toán phân vùng ảnh bằng cách thay đổi kiến trúc mạng VGG-16 và GoogleNet để xử lý các đầu vào và đầu ra có kích thước không cố định. Cũng vào năm này, No và cộng sự đề xuất Mô hình *Tự mã hóa – giải mã* cho phân vùng ảnh. Ren và cộng sự đề xuất mạng theo *Mô hình mạng tích chập vùng (R-CNN)* cho phân vùng thực thể.

Năm 2016, Visin và cộng sự đề xuất mạng theo *Mô hình hồi qui* (Recurrent Neural Network - RNN) cho phép mô hình hóa quan hệ phục thuộc ngắn/dài hạn giữa các điểm ảnh để cải thiện chất lượng phân vùng ảnh. Năm 2017, Lin và cộng sự đề xuất mạng FPN (Feature Pyramid Network) theo *Mô hình đa phân giải và cấu trúc kim tự tháp*, trong đó, cấu trúc phân cấp

đa tầng các mạng CNN được lồng ghép để tạo các đặc trưng đa phân giải nhằm phát hiện các đối tượng ở kích thước khác nhau trong ảnh.

Năm 2018, Marcos và cộng sự đã đề xuất *Mô hình mạng CNN kết hợp với mô hình biên động* (active contour).

Phân vùng ảnh được ứng dụng trong nhiều lĩnh vực khác nhau như phân tích ảnh y tế, xe tự hành, giám sát an ninh, thực tại tăng cường, tương tác người máy, v.v.

- Phân tích ảnh y tế: phân vùng các khối u, các vùng tổn thương trên ảnh nội soi, ảnh CT, MRI
- Xe tự hành: Phát hiện các biển báo giao thông, làn đường, xe cộ, trợ giúp lái cho xe tự hành
- Giám sát an ninh: Phát hiện người truy nhập trái phép trong tòa nhà, phân vùng và định vị bất thường
- Tương tác người máy: phân vùng bàn tay trong ảnh để điều khiển thiết bị trong phòng thông minh
- Đa dạng sinh học: phát hiện và phân vùng ảnh cây cối phục vụ cho việc nhận dạng tự động, trợ giúp trong việc lưu trữ, bảo tồn bảo tàng cây thuốc, cây gỗ quý
- Vệ tinh, viễn thám: khoanh vùng các vùng ảnh quan tâm (phân hoạch, giám sát diện rộng).

Phân vùng ảnh là một pha xử lý trung gian quan trọng trong chuỗi xử lý phân tích và hiểu ảnh. Bài toán phân vùng ảnh vẫn đối mặt với các thách thức cần phải giải quyết trong tương lai như sau: Các cơ sở dữ liệu thách thức hơn; Các mô hình học sâu có thể giải thích được; Các phương pháp học không giám sát hoặc ít giám sát; Mô hình thời gian thực cho các ứng dụng khác nhau; Phân vùng dựa trên đám mây điểm: hiện nay các thiết bị cho phép thu nhận dữ liệu

3D đang phổ biến. Việc phân vùng dựa trên đám mây điểm 3D cũng sẽ là một khuynh hướng phát triển trong tương lai gần.

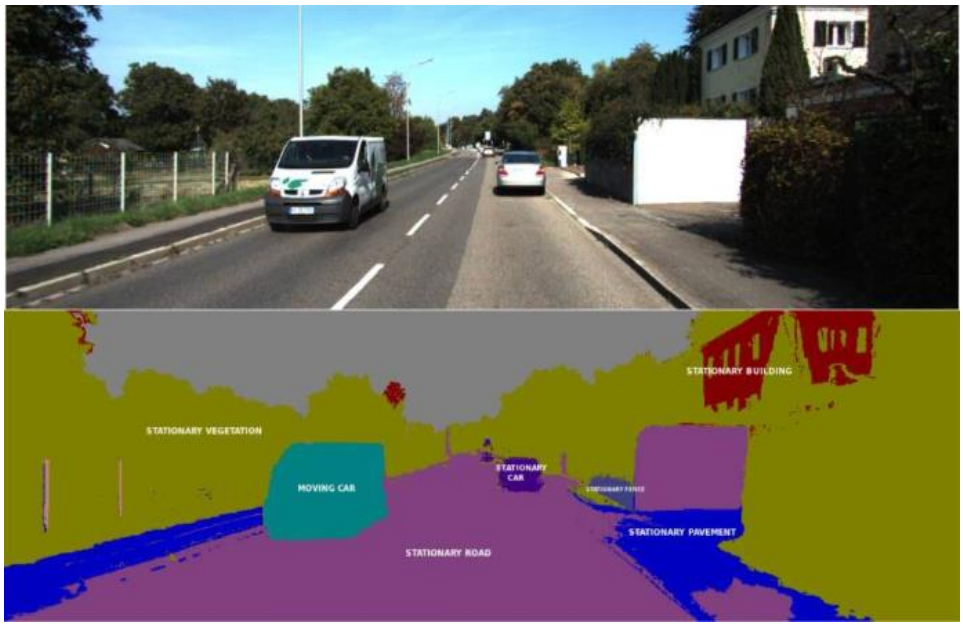
Hiện nay, tại Việt Nam, có nhiều phòng thí nghiệm, viện nghiên cứu, bộ môn hoặc các nhà nghiên cứu độc lập, các công ty đang theo đuổi và giải quyết các bài toán liên quan đến phân vùng ảnh phục vụ cho các ứng dụng khác nhau như phân vùng ảnh nội soi, ảnh X quang phục vụ các ứng dụng y tế (Viện MICA, Viện Điện tử Y Sinh – ĐHBK Hà Nội); phân vùng ảnh văn bản phục vụ tự động nhận dạng ký tự (Viện công nghệ thông tin thuộc Viện Hàn lâm KHCNVN), phân vùng ảnh vệ tinh (Viện công nghệ thông tin thuộc Viện Hàn lâm KHCNVN, Học viện Kỹ thuật Quân sự, Đại học Quốc gia Hà Nội); phân vùng ảnh giao thông trong điều hướng phân luồng (Công ty Biển Bạc), phân vùng ảnh người, tay trong ứng dụng giám sát hoặc tương tác người máy (Viện MICA, ĐHBK Hà Nội, Đại học Quốc gia HCM), v.v.

1.3 Phân đoạn ảnh ngữ nghĩa

1.3.1 Phân đoạn ngữ nghĩa là gì

Phân đoạn ngữ nghĩa là quá trình phân bổ nhãn ngữ nghĩa cho mọi pixel có trong hình ảnh. Ví dụ: tính năng phát hiện biển báo giao thông nhằm mục đích phân loại từng pixel thành ô tô, người đi bộ hoặc biển báo đường bộ.

Phân đoạn ngữ nghĩa nhằm mục đích giải quyết các vấn đề phân loại bằng thị giác máy tính thúc đẩy học sâu và mạng lưới thần kinh tích chập (CNN). Nó liên quan đến việc gắn nhãn cho từng pixel trong hình ảnh bằng nhãn danh mục hoặc lớp, chẳng hạn như “người”, “cây” hoặc “ô tô”.



Hình 1.3 Gán nhãn cho từng pixel trong ảnh

Phân đoạn theo ngữ nghĩa khác với các tác vụ phân đoạn tiêu chuẩn vốn gắn nhãn các pixel theo đặc tính vật lý của chúng (tức là màu sắc của chúng). Nó **nhằm mục đích chú thích từng pixel** bằng nhãn đối tượng của nó. Thuật toán xác định đối tượng hoạt động theo kiến trúc học tập của mạng lưới thần kinh.

Phân đoạn theo ngữ nghĩa gần đây đã thu hút được nhiều sự chú ý hơn trong lĩnh vực thị giác máy tính và học sâu. Điều này là do ứng dụng ngày càng tăng của nó trong các ngành công nghiệp khác nhau.

1.3.2 Các loại phân đoạn hình ảnh khác nhau

Phân đoạn hình ảnh là quá trình phân chia một hình ảnh kỹ thuật số (hình ảnh tĩnh và hình ảnh video) thành nhiều phân đoạn. Các phân khúc này là các khu vực đồng nhất về nội dung và đặc điểm. Phân đoạn được sử dụng trong các tác vụ xử lý hình ảnh và thị giác máy tính, chẳng hạn như nhận dạng đối tượng và hiểu cảnh.

Các loại phân đoạn hình ảnh bao gồm:

- Phân đoạn ngữ nghĩa
- Phân đoạn tức thì

Phân đoạn ngữ nghĩa

Phân đoạn ngữ nghĩa là một kỹ thuật học sâu phân loại các pixel hình ảnh thành các lớp được xác định trước. Mục tiêu là gán nhãn cho mọi pixel trong ảnh, cho phép phân biệt các đối tượng và ranh giới của chúng trong ảnh. Điều này rất hữu ích trong các ứng dụng khác nhau, chẳng hạn như xe tự hành, hình ảnh y tế và phân tích hình ảnh vệ tinh. Trong các phương tiện tự trị, phân đoạn theo ngữ nghĩa có thể được sử dụng để xác định ranh giới đường, biển báo giao thông và người đi bộ. Trong hình ảnh y tế, nó có thể được sử dụng để phân đoạn khối u và các cấu trúc quan trọng khác trong quá trình quét y tế. Phương pháp này sử dụng các mạng thần kinh tích chập (CNN) để phân tích một hình ảnh và tạo ra một bản đồ phân đoạn xác định đối tượng và ranh giới của chúng. Đầu ra là một hình ảnh được phân đoạn trong đó mỗi pixel được dán nhãn, hiểu rõ các đối tượng và mối quan hệ của chúng trong hình ảnh

Phân đoạn ngữ nghĩa là một trường con của thị giác máy tính và học máy liên quan đến việc xác định và gắn nhãn các đối tượng cũng như các thành phần ngữ nghĩa khác trong hình ảnh. Phân đoạn theo ngữ nghĩa rất quan trọng trong các nhiệm vụ như:

- Phát hiện đối tượng
- Phân đoạn
- Thu hồi hình ảnh
- Tìm kiếm hình ảnh
- Phân loại

Phân đoạn tức thì

Phân đoạn tức thì và phân đoạn ngữ nghĩa có mục tiêu khác nhau. Phân đoạn ngữ nghĩa nhằm gắn nhãn cho từng pixel, trong khi phân đoạn tức thì nhằm phân biệt giữa các phiên bản của cùng một lớp.

1.3.4 Một số ứng dụng

Nhận dạng khuôn mặt :

Hình 1.4 Nhận dạng khuôn mặt

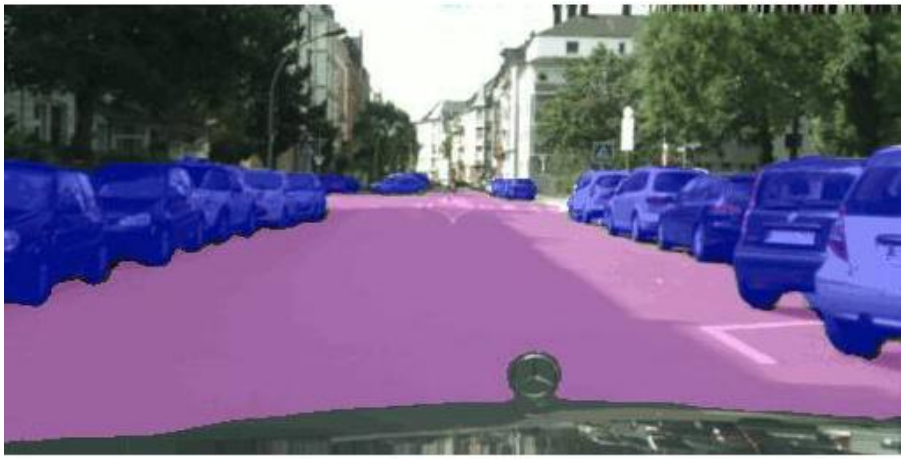
Phân đoạn ngữ nghĩa được sử dụng để nhận dạng khuôn mặt. Mô hình được đào tạo để dự đoán xem một hình ảnh có chứa khuôn mặt hay không. Hình ảnh đầu vào trước tiên được chuyển đổi thành biểu diễn vectơ mà bạn có thể sử dụng để so sánh với các vectơ khác.

Ý tưởng đằng sau ứng dụng này là bạn có thể phân chia khuôn mặt thành các vùng khác nhau, chẳng hạn như mắt, mũi và miệng. Sau đó, điều này có thể xác định người được đề cập bằng cách so sánh hình ảnh với tập dữ liệu về các khuôn mặt đã biết.

Dưới đây là bảng phân tích về quy trình nhận dạng khuôn mặt:

- Hệ thống xác minh trước tiên sẽ phân tích khuôn mặt.
- Tiếp theo, hệ thống sẽ loại bỏ mọi nhiễu và các yếu tố gây mất tập trung khác khỏi hình ảnh để mang lại hình ảnh chất lượng tốt hơn.
- Hệ thống tiến hành phân tích hình ảnh và xác định hình ảnh đó thuộc về chủ thể nào trong cơ sở dữ liệu.
- Đầu ra của chương trình này cho biết đối tượng trong ảnh đầu vào có trong cơ sở dữ liệu hay không.

Xe tự lái :

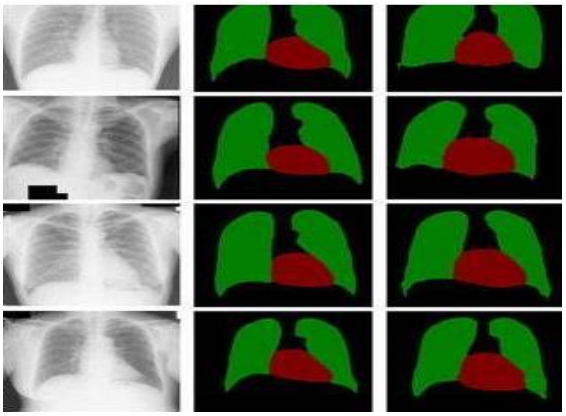


Hình 1.5 Xe tự lái

Phân đoạn theo ngữ nghĩa là bước đầu tiên hướng tới việc thiết lập hệ thống lái tự động. Để hiểu thế giới xung quanh, ô tô sử dụng kỹ thuật thị giác máy tính để xác định các vật thể và thuộc tính của chúng. Bước tiếp theo là đưa ra quyết định dựa trên những quan sát này.

Ví dụ, nếu một chiếc ô tô nhìn thấy một người đi bộ đang đi trên đường, nó cần biết người đi bộ này sẽ đi đâu và liệu người đó có băng qua đường hay không. Để làm được điều đó, chiếc ô tô cần hiểu những đồ vật trong khung cảnh đó là gì và chúng sẽ di chuyển như thế nào trong tương lai (tức là quỹ đạo của chúng). Đây là nơi phân đoạn ngữ nghĩa xuất hiện.

Chẩn đoán hình ảnh y tế :



Hình 1.6 Chuẩn đoán y tế

Chẩn đoán hình ảnh y tế tốn nhiều thời gian, chi phí và dễ xảy ra lỗi. Quá trình này bao gồm việc dán nhãn cho từng pixel của hình ảnh với cấu trúc giải

phẫu tương ứng. Quá trình này đòi hỏi nỗ lực đáng kể từ các chuyên gia được đào tạo, những người dành hàng giờ để chú thích cho mỗi hình ảnh.

Phân đoạn theo ngữ nghĩa giúp giảm thời gian cần thiết cho việc gắn nhãn thủ công bằng chú thích hình ảnh. Ngành y tế có thể tận dụng mô hình học máy tự động gắn nhãn cho từng pixel trong hình ảnh với cấu trúc giải phẫu tương ứng**.** Bằng cách này, các bác sĩ có thể tập trung vào chẩn đoán thực tế thay vì dành thời gian cho các nhiệm vụ có giá trị thấp như phân loại pixel trong hình ảnh.

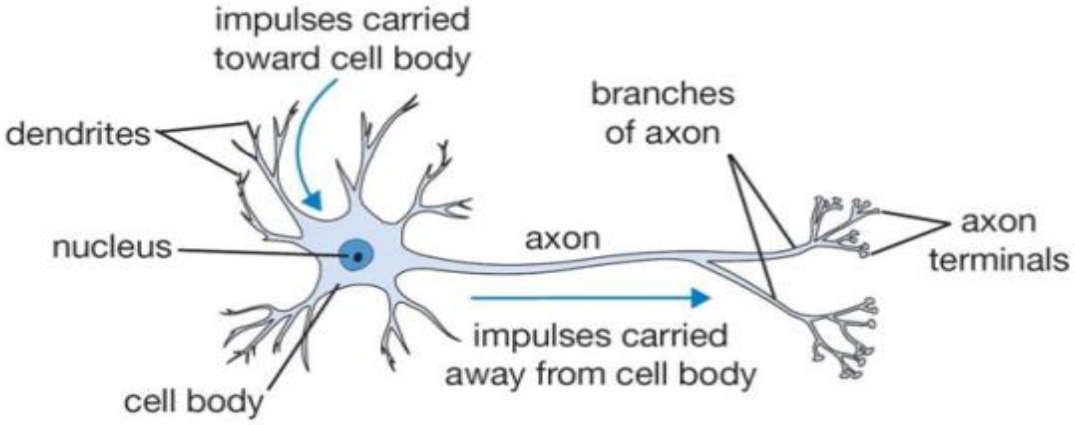
Tóm lại, phân đoạn ngữ nghĩa là một lĩnh vực quan trọng của thuật toán học sâu được tận dụng để tăng cường những tiến bộ trong thị giác máy tính. Phân đoạn theo ngữ nghĩa sẽ tiếp tục phát triển trong nhiều danh mục con liên quan, phát hiện đối tượng, phân loại và bản địa hóa.

CHƯƠNG 2: MẠNG NƠ RON NHÂN CHẬP

2.1 Mạng nơ ron

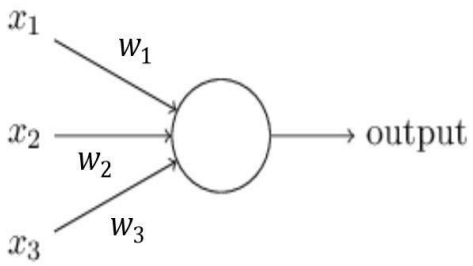
Mạng nơ ron (Neural Networks – NN) là một phương thức trong lĩnh vực trí tuệ nhân tạo được áp dụng để dạy máy tính xử lý dữ liệu, bằng cách lấy cảm hứng từ bộ não con người. Mạng nơ ron sử dụng các nút hoặc nơ ron liên kết với nhau trong một cấu trúc phân lớp tương tự như bộ não con người. Phương thức này tạo ra một hệ thống thích ứng cho phép máy tính sử dụng để học hỏi từ sai lầm của chúng và liên tục cải thiện qua thời gian. Vì vậy mạng nơ ron nhân tạo nhắm tới giải quyết các vấn đề phức tạp.

Một mạng nơ ron được cấu thành bởi các nơ ron đơn lẻ được gọi là các perceptron. Nên trước tiên ta tìm hiểu xem perceptron là gì rồi ta sẽ tiến tới kiến trúc của mạng nơ ron sau. Nơ ron nhân tạo được lấy cảm hứng từ nơ ron sinh học như hình mô tả bên dưới :



Hình 2.1 Mô tả mạng nơ ron sinh học

Như hình trên, ta có thể thấy một nơ ron có thể nhận nhiều đầu vào và cho ra một kết quả duy nhất. Mô hình của perceptron cũng tương tự như vậy:



Hình 2.2 Mô hình perceptron

Một perceptron sẽ nhận một hoặc nhiều đầu vào x và cho ra một kết quả o dạng nhị phân duy nhất. Giả sử perceptron có ba đầu vào x_1 x_2 x_3,,. Để tính toán đầu ra chúng ta sử dụng các trọng số số w_1, w_2, w_3 là các số thực tiễn tả độ quan trọng của đầu vào tương ứng. Đầu ra là 0 hay 1 được xác định thông qua tổng xích ma của các tích $w_i * x_i$ với một giá trị ngưỡng (threshold) theo biểu thức :

$$o = \begin{cases} 0 \; IF \; \sum_i w_i x_i \; \leq threshold \\ 1 \; IF \; \sum_i w_i x_i \; > threshold \end{cases} \qquad (2 - 1)$$

Đặt b = - threshold, ta có thể viết lại thành :

$$o = \begin{cases} 0 \; IF \; \sum_i w_i x_i + b \; \leq 0 \\ 1 \; IF \; \sum_i w_i x_i + b > 0 \end{cases} \qquad (2 - 2)$$

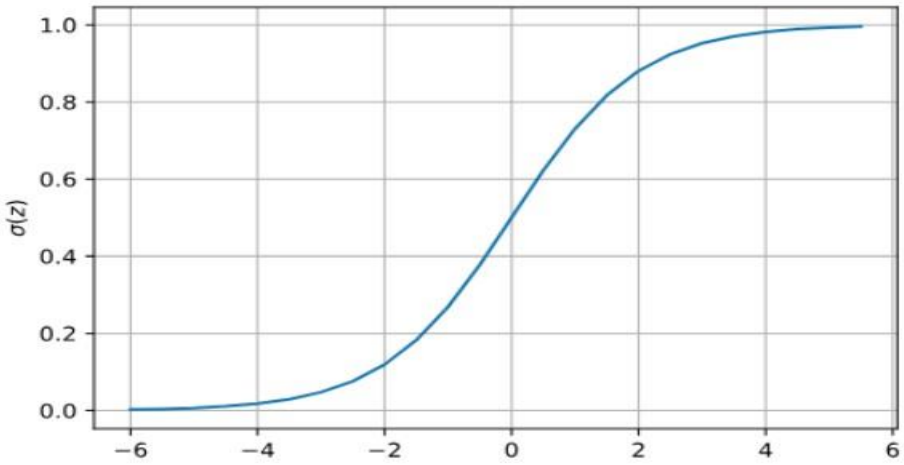
Chúng ta có thể thấy khi thay đổi các giá trị trọng số, hay ngưỡng chúng ta có thể tạo ra các đầu ra khác nhau. Hay nói cách khác là tạo ra các quyết định khác nhau. Có thể nhận thấy được nhược điểm của perceptron là khi ta thay đổi một lượng nhỏ các trọng số có thể gây ra sự thay đổi lớn về đầu ra. Đầu ra có thể lật từ 0 sang 1 hoặc 1 sang 0n. Để khắc phục điều này, sigmoid nơ ron tạo nra cho phép thay đổi nhỏ của trọng số mà chỉ tạo ra một thay đổi nhỏ ở đầu ra

Cách hoạt động của sigmoid nơ ron là sử dụng đầu vào số thực và tạo ra một đầu ra. Các trọng số vẫn giống perceptron nhưng đầu ra không phải là 0 hay 1. Thay vào đó đầu ra được quyết định bởi một hàm singmiod $\sigma(z)$, với σ là hàm singmiod được định nghĩa như sau :

$$\sigma(z) = \frac{1}{1+e^{-z}} \qquad (2 – 3)$$

Giả sử các đầu vào là là x_1, x_2,... các trọng số w_1, w_2,..., và B là một giá trị tín ngưỡng. Và ta đặt z= $\sum_i w_i x_i + b$ thì công thức lúc này sẽ có dạng :

$$\sigma(z) = \frac{1}{1+e^{-\sum_i w_i x_i + b}} \qquad (2 – 4)$$

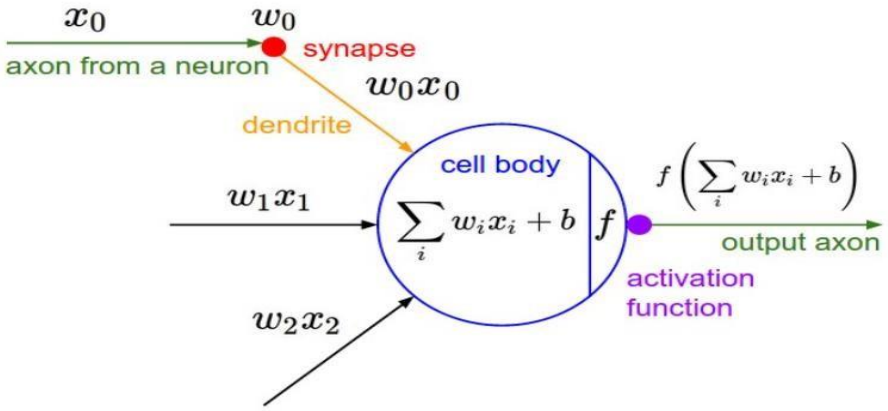


Hình 2.3 Đồ thị hàm singmoid

Ngoài hàm Singmoid ra, ta còn có thể có một số hàm kích hoạt khác như Tanh, ReLu để thay thế hàm Singmoid bởi dạng đồ thị của nó cũng tương tự như singmoid. Một cách tổng quát, hàm perceptron được biểu diễn qua một hàm kích hoạt (activation function) $f(z)$ như sau, trong đó : z $= \sum_i w_i x_i + b$

$$o = f(\sum_i w_i x_i + b) \qquad (2-5)$$

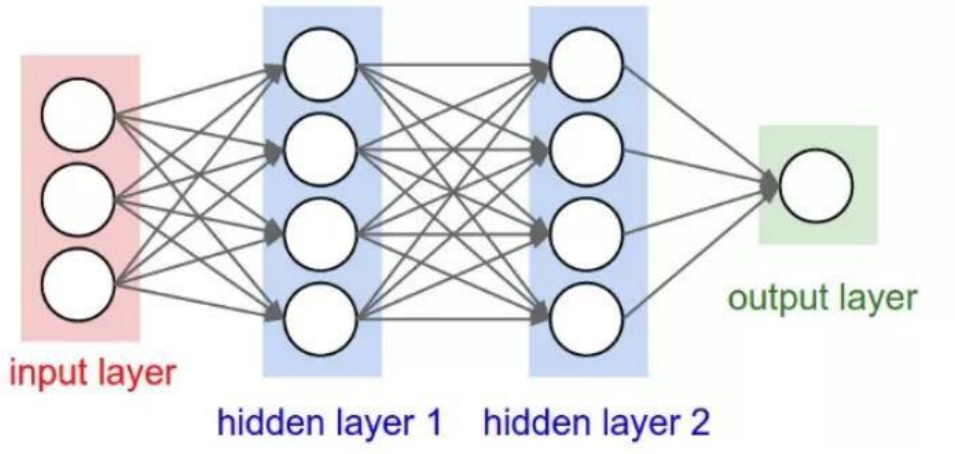
Bằng cách biểu diễn như vậy, ta có thể coi mô hình perceotron được thể hiện chi tiết như sau :



Hình 2.4 Mô hình perceptron chi tiết

2.1 Kiến trúc mạng nơ ron

Mạng nơ ron là sự kết hợp của các tầng perceptron hay còn được gọi là perceptron đa tầng (multilayer perceptron) như hình vẽ bên dưới :



Hình 2.5 Kiến trúc mạng nơ ron

- Tầng vào (input layer) : Là tầng bên trái cùng của mạng thể hiện cho các đầu vào của mạng.
- Tầng ra (output layer) : Là tầng bên phải cùng của mạng thể hiện cho các đầu ra của mạng
- Tầng ẩn (hidden layer) : Là tầng nằm giữa tầng vào và tầng ra thể hiện cho việc suy luận logic của mạng. Lưu ý rằng, một nơ ron chỉ có một tầng vào một tầng ra nhưng có thể có nhiều tầng ẩn.

Trong mạng nơ ron, mỗi nút mạng là một sigmoid nơ ron nhưng hàm kích hoạt của chúng có thể khác nhau.

Ở mỗi tầng, số lượng các nút dạng (nơ ron) có thể khách nhau tuỳ thuộc vào bài toán và cách giải quyết. Nhưng thường khi làm việc người ta để các tầng ẩn có số lượng nơ ron bằng nhau. Ngoài ra, các nơ ron ở các tầng thường được liên kết đôi một với nhau tạo thành mạng kết nối đầy đủ (Fully connected netwwork)

2.3 Mạng nơ ron tích chập (Convolutional Neural Netwwork – CNN)

2.3.1 Định nghĩa mạng nơ ron tích chập

CNN được viết tắt của Convolutional Neural Netwwork hay còn gọi là mạng nơ ron tích. Là một trong những mô hình Deep Learning cực kì tiên tiến,

bởi chúng mạng lại khả năng xây dựng những hệ thống phân loại có độ chính xác cao và thông minh. Nhờ khả năng đó CNN có rất nhiều ứng dụng đặc biệt là những bài toán cần phân loại đối tượng trong ảnh.

Mạng nơ ron tích chập là một mạng nơ ron nhân tạo với các toán tử tích chập. Nó có khả năng học một lượng lớn dữ liệu trong khoảng thời gian ngắn hơn đáng kể so với mạng nơ ron thông thường. Nguyên nhân là do nó giới hạn việc sử dụng ít trọng số hơn, trong khi vẫn duy trì độ chính xác chỉ thấp hơn một phần nhỏ so với kiến trúc truyền thống.

2.3.2 Các lớp cơ bản của mạng CNN

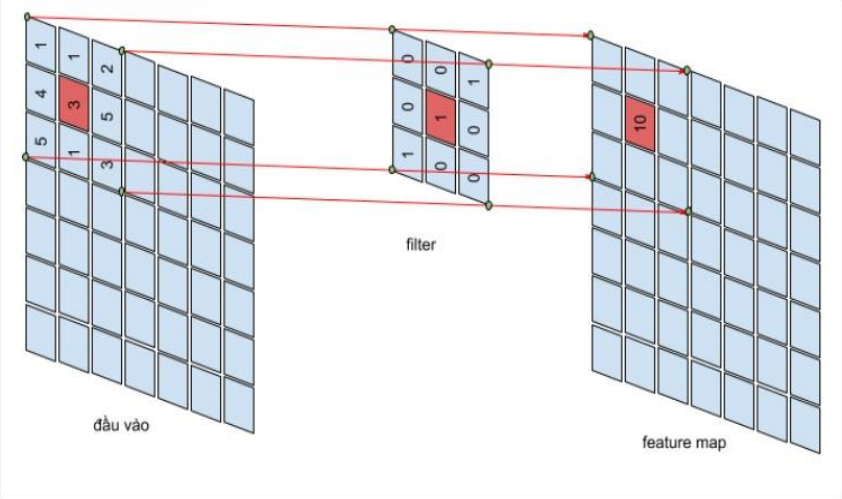
Mạng CNN bao gồm những lớp cơ bản sau :

- ***2.3.a Lớp tích chập (convolutional layer – CONV)***

Lớp tích chập là phần quan trọng nhất của toàn mạng CNN, nó có nhiệm vụ thực hiện phép toán tích chập trên ảnh đầu vào để trích xuất các đặc trưng của ảnh. Lớp này sẽ sử dụng một tập các bộ lọc (filter) có kích thước 5×5 hoặc 3×3 áp vào một vùng trong ảnh đầu vào và tiến hành tính tích chập giữa các trọng số của bộ lọc và các giá trị điểm ảnh trong vùng được quét. Bộ lọc sẽ lần lượt được dịch chuyển theo một giá trị bước trượt (stride) quét toàn bộ ảnh đầu vào. Các trọng số của filter ban đầu sẽ được khởi tạo ngẫu nhiên và sẽ được cập nhật dần trong quá trình huấn luyện mô hình. Các yếu tố quan trọng trong lớp tích chập là : padding, stride, feature map và filter map.

- Filter map : Sử dụng filter để áp dụng vào các vùng của ma trận ảnh đầu vào.
- Stride (trượt) : Tức là ta dịch chuyển filter map theo từng pixel dựa vào các giá trị từ trái qua phải.
- Padding (lề) : Thường giá trị lề xung quanh của các ma trận ảnh sẽ được gán các giá trị 0 để có thể tiến hành nhân tích chập mà không làm giảm kích thước ma trận ảnh ban đầu

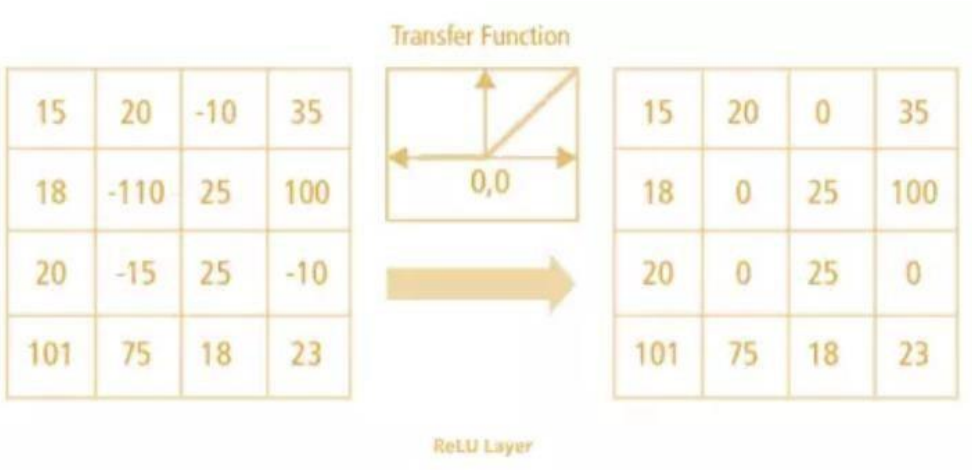
- Featura map : Biểu diễn kết quả của việc áp dụng một bộ lọc (filter) lên một ảnh đầu vào thông qua phép toán tích chập. Feature map thường chứa các thông tin về các đặc trưng (features) cụ thể của ảnh đầu vào.



Hình 2.6 Ví dụ về lớp tích chập

2.3.b Lớp ReLU (ReLU layer)

Lớp ReLU (Rectified Linear Unit) thường được đặt ngay sau lớp tích chập. Với mục đích là đưa ra tính phi tuyến trong mô hình có khả năng học được các biểu diễn phúc tạp từ dữ liệu. Lớp này sử dụng hàm kích hoạt ReLU với đầu ra là là $f(x) = \max(0, x)$, được giới thiệu bởi Geoffrey E.Hinton năm 2010. Hàm này có nhiệm vụ chuyển toàn bộ giá trị âm trong kết quả lấy từ lớp tích chập thành giá trị 0 và giữ nguyên giá trị đầu vào khi lớn hơn 0. Trước khi hàm ReLU được áp dụng thì những hàm như Sigmoid hay Tanh mới là những hàm được sử sụng phổ biến. Hàm ReLU được ưa chuộng vì tính toán đơn giản và cũng cho kết quả tốt hơn.

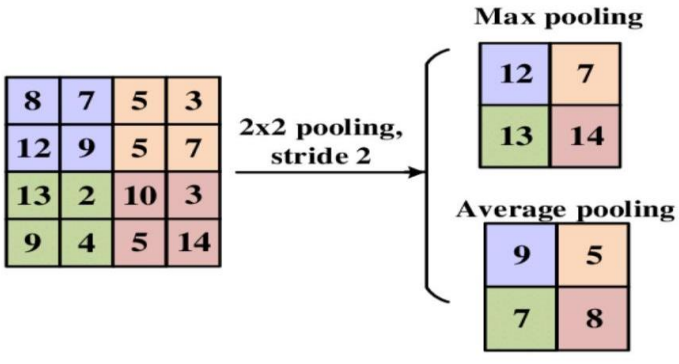


Hình 2.7 Ví dụ về hàm ReLU

2.3.c Lớp gộp (Pooling layer – POOL)

Sau lớp ReLU chúng ta sẽ sử dụng lớp gộp, với chức năng chính lag giảm kích thước dữ liệu của lớp trước đó nhưng vẫn giữ được các thuộc tính quan trọng. Kích thước dữ liệu giảm, giúp giảm việc tính toán trong mô hình. Lớp này sử dụng một cửa sổ trượt quét qua toàn bộ ảnh dữ liệu và trượt theo buóc trượt (stride) cho trước. Khác với lớp tích chập, lớp gộp không tính tích chập mà tiến hành lấy một giá trị được xem là giá trị đại diện cho thông tin ảnh tại vùng được quét. Hiện tại có 2 loại phép gộp được sử dụng phổ biến là max pooling và average pooling:

- Max pooling : Phép gộp chọn giá trị lớp nhất trong vùng mà nó đang được quét.
- Average pooling : Phép gộp tính trung bình các giá trị trong vùng mà nó dùng được quét.

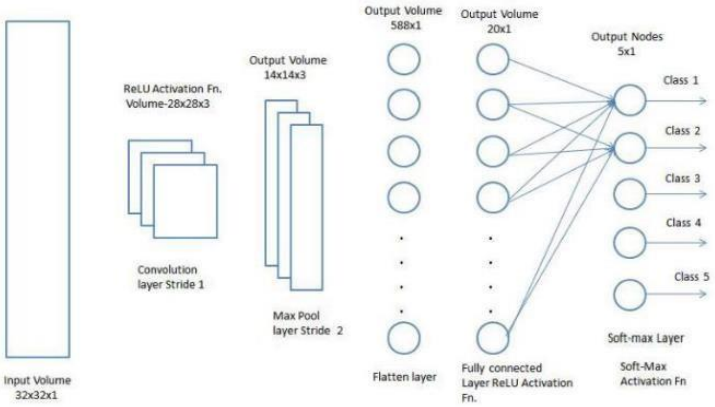


Hình 2.7 Max pooling và Average pooling

Mặc dù rất nhiều thông tin bị mất trong lớp gộp nhưng nó cũng mang lại một số lợi ích cho CNN như chúng giảm tốc độ phức tạp, nâng cao hiệu quả và hạn chế rủ ro bị thừa.

2.3.d Lớp kết nối đầy đủ (Fully connected layer – FC)

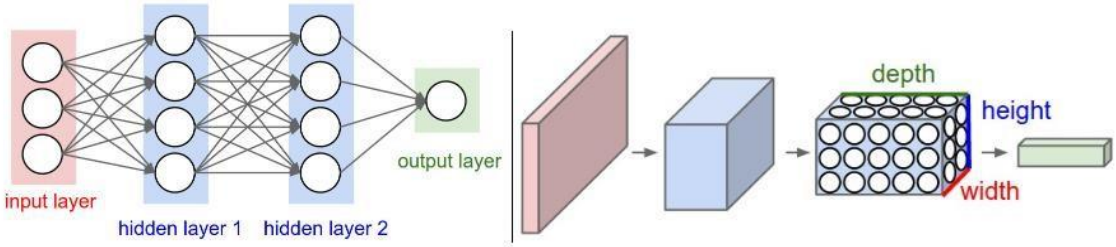
Lớp kết nối đầy đủ là lớp cuối cùng trong CNN. Lớp này thực hiện nhiệm vụ phân loại dựa trên các đặc trưng được trích xuất thông qua các lớp trước đó. Lớp FC thường tận dụng hàm kích hoạt softmax để phân loại đầu vào một cách thích hợp



Hình 2.8 Lớp kết nối đầy đủ

2.4 Kiến trúc mạng CNN

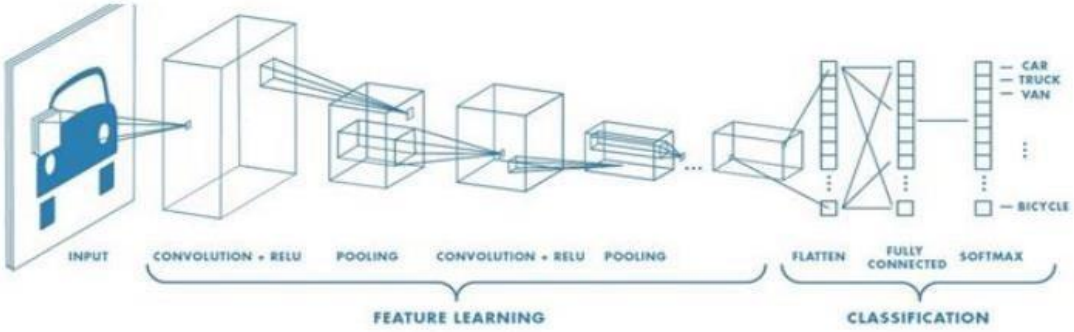
Mạng nơ ron tích chập có kiến trúc khác với mạng nơ ron thông thường. Mạng nơ ron bình thường chuyển đổi đầu vào thông qua hàng loạt các tầng ẩn. Mỗi tầng là một tập các nơ ron và các tầng được liên kết đầy đủ và các nơ ron ở tầng trước đó.Ở tầng cuối cùng sẽ là tầng kết quả đại diện cho dự đoán của mạng. Đầu tiên, mạng nơ ron tích chập được chia thành 3 chiều : rộng, cao và sâu. Đây là một thiết kế rất phù hợp cho bài toàn phân loại dữ liệu ảnh có kích thước chiều cao x chiều rộng x 3 kênh màu (đỏ - lục - lam). Tiếp theo các nơ ron trong mạng không kết nối hoàn toàn với tất cả nơ ron kể tiếp mà chỉ liên kết với một phần nào của chúng. Cuối cùng một tầng đầu ra được chuyển thành vectơ của giá trị xác suất



Hình 2.9 Mạng nơ ron thông thường (trái) và CNN (phải)

Kiến trúc CNN gồm 2 phần :

- Phần trích xuất đặc trưng : Trong phần này, mạng sẽ tiến hành tính toán hàng loạt phép tích chập và ghép gộp (pooding) để phát hiện các đặc trưng.
- Phần phân lớp : Tại phần này, một lớp với các liên kết đầy đủ sẽ đóng vai trò như một bộ phân lớp các đặc trưng đã rút trích được trước đó. Tầng này sẽ đưa ra được xác suất của một đối tượng.



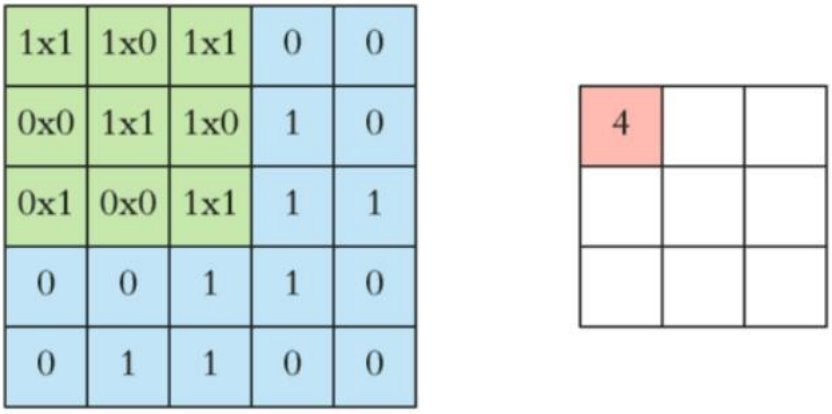
Hình 2.10 Kiến trúc mạng CNN

2.4.a Trích xuất đặc trưng

Đầu tiên ảnh đầu vào sẽ được đưa qua lớp tích chập để thực hiên tính toán tích chập giữa ảnh đầu vào và bộ lọc (filter map) để tạo ra một ma trận đặc trưng (featura map)

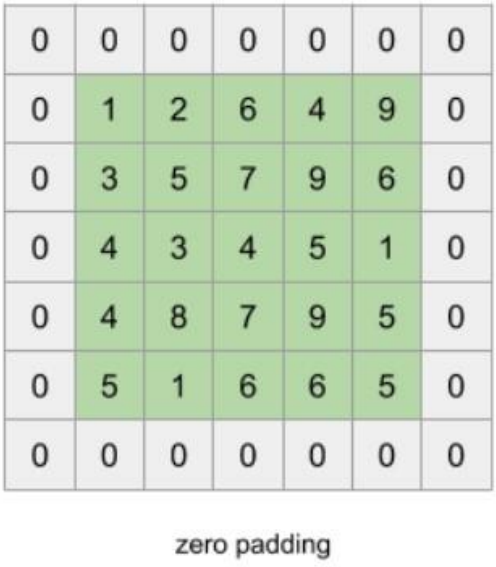
Ta thực hiện phép tích chập bằng cách trượt filter theo dữ liệu đầu vào. Tại mỗi vị trí, ta tiến hành phép nhân ma trận và tính tổng các giá trị để đưa vào ma trận đặc trưng (feature map).

Ví dụ về phép tích chập như hình dưới đây, trong đó thành phần filter kích thước là 3×3 (màu xanh lá) trượt trên ma trận ảnh đầu vào (màu xanh dương) và kết quả được trả về bản đồ đặc trưng (màu hồng)



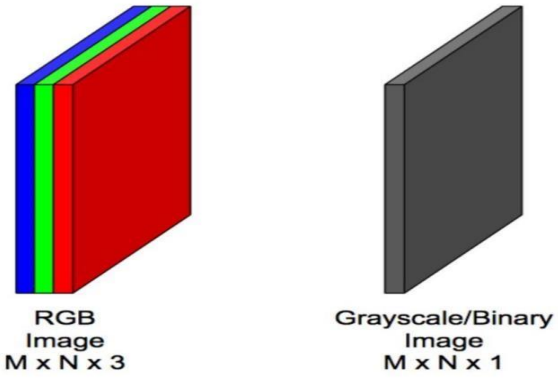
Hình 2.11 Ví dụ về phép tích chập

Bởi vì kích thước đầu ra luôn nhỏ hơn đầu vào nên ta cần một phép xử lý đầu vào để đầu ra không bị thay đổi. Đơn giản ta chỉ cần thêm một lể (padding) vào đầu vào. Một lề (padding) với giá trị 0 sẽ được thêm vào xung quanh ma trận đầu vào trước khi thực hiện phép tích chập.



Hình 2.12 Hình ảnh minh hoạ ma trận đầu vào sau khi thêm padding = 1 với giá trị bằng 0

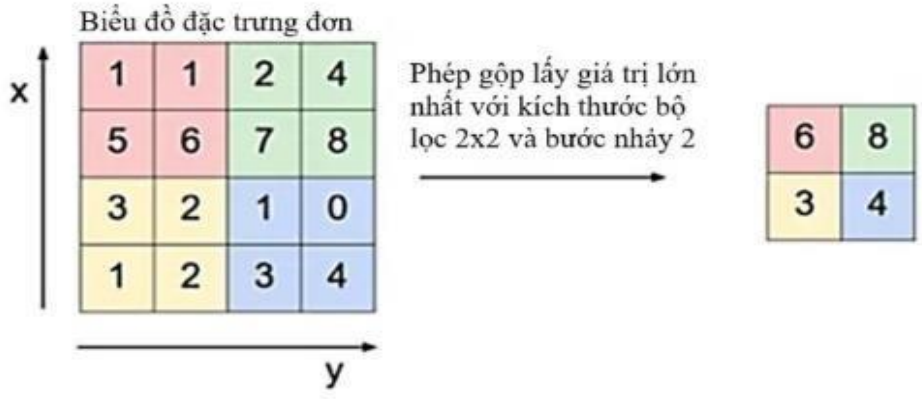
Trong thực tế phép tích chập được thực hiện trên không gian 3 chiều, vì mỗi hình ảnh được biểu diễn dưới dạng ba chiều : chiều rộng, chiều cao và chiều sâu. Chiều sâu ở đây tương ứng với giá trị màu sắc của hình ảnh, thường được biểu diễn dưới dạng RGB



Hình 2.13 Hình ảnh RGB và ảnh xám

Sau đó đầu ra của lớp tích chập sẽ đưa qua lớp ReLU, lớp này sử dụng hàm kích hoạt ReLU để có đầu ra dưới dạng phi thuyền và chuyển toàn bộ giá trị âm trong kết quả lấy từ lớp tích chập thành giá trị 0 và giữ nguyên giá trị đầu vào khi lớn hơn 0.

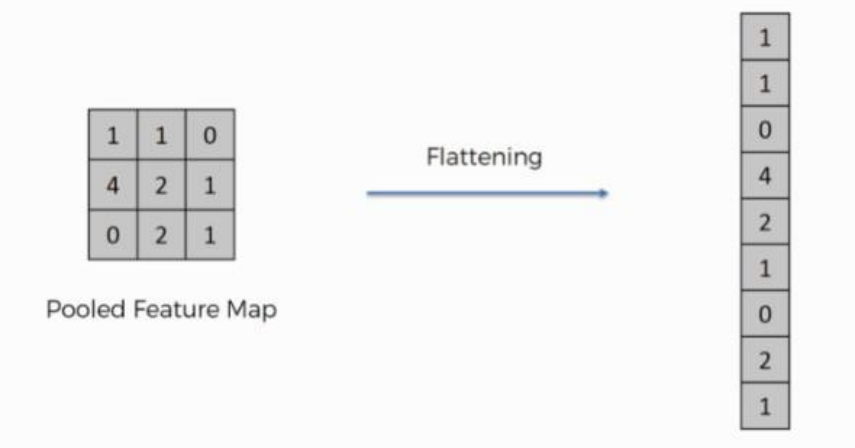
Sau lớp ReLU kết quả sẽ đi qua lớp gộp (pooling layer). Việc này giúp giảm kích thước dữ liệu, giảm thời gian học và hạn chế việc overfitting. Overfitting là một hiện tượng khi mô hình máy học được huấn luyện quá mức trên dữ liệu huấn luyện, đến mức độ học thuộc lòng dữ liệu đó mà không tổng quá hoá được cho dữ liệu mới chưa nhìn thấy. Trong trường hợp này, mô hình có thể thích ứng quá mức với các chi tiết không quan trọng trong dữ liệu huấn luyện, nên làm giảm khả năng dự đoán chính xác trên dữ liệu mới. Một phép gộp đơn giản thường được dùng đó là phép gộp lấy giá trị lớn nhất (max pooling), phép gộp này sẽ lấy giá trị lớn nhất của một vùng để đại diện cho vùng đó.



Hình 2.14 Phép gộp lấy giá trị lớn nhất

2.4.b Phân loại

Sau lớp tích chập và pooling, vì đầu vào của mạng liên kết đầy đủ là một chiều nên feature map được làm phẳng (flatten) trước khi đưa vào lớp kết nối đầy đủ (Fully Connected layer)



Hình 2.15 Ví dụ minh hoạ khi làm phẳng feature map để đưa vào lớp kết nối đầy đủ

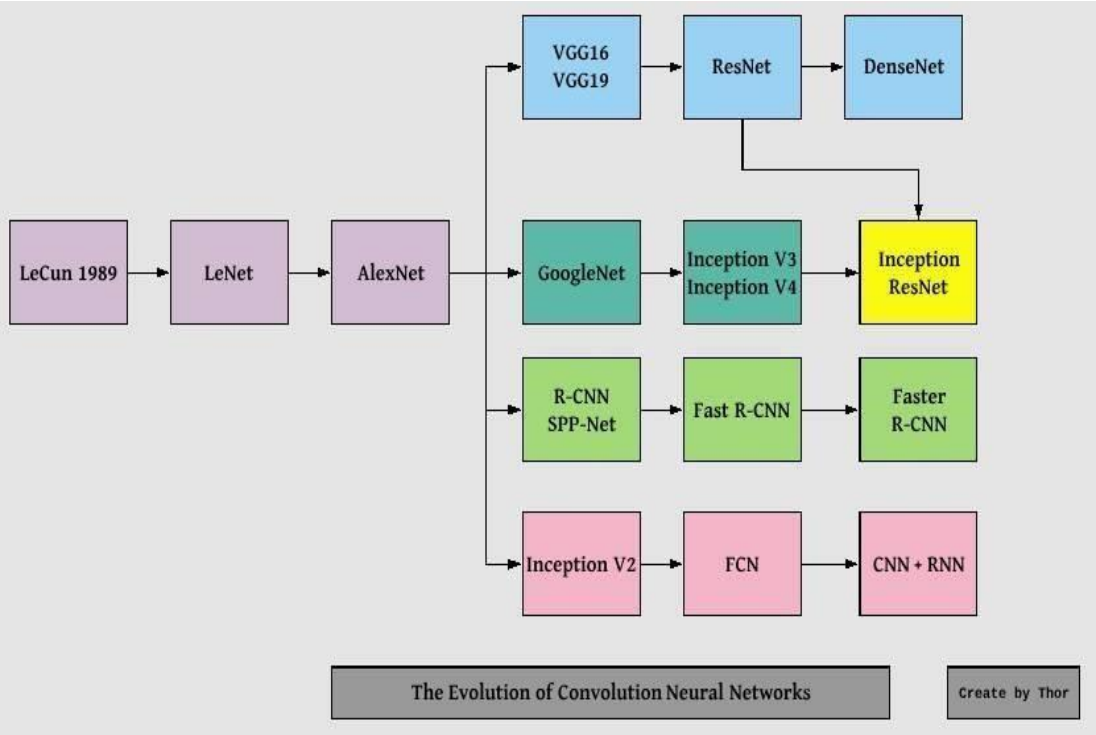
Lớp kết nối đầy đủ (Fully Connected layer) này hoạt động như một mạng nơ ron truyền thống, kết nối các đặc trưng đã học từ lớp trước để thực hiện phân loại và cuối cùng sử dụng hàm kích hoạt softmax để phân loại, Hàm softmax sẽ chuyển đổ giá trị đầu ra thành các giá trị xác suất tương ứng với các lớp đối tượng. Nó đảm bảo rằng tổng xác suất của tất cả các lớp bằng 1. Trong đó lớp có xác suất cao nhất được coi là lớp dự đoán cho đối tượng hình ảnh.

2.5 Một số cấu trúc mạng CNN

Hiện nay có nhiều mạng nơ ron tích chập nổi tiếng và phổ biến. Kết quả thử nghiệm có hiệu suất tốt. Cho nên thay vì tự thiết kế mạng chúng ta sử dụng các thiết kế có sẵn. Hầu hết các mạng CNN đều được thiết kế theo nguyên tắc chung:

- Sử dụng nhiều convolution layer chồng lên nhau
- Giảm dần kích thước output mỗi lớp
- Tăng dần số lượng feature map

Trong khi các mạng CNN trước đây hầu hết theo dạng chỉ đơn giản thiết kế theo lối nhiều lớp Convolution xếp chồng lên nhau thì nhiều mạng mới đây đã thiết kế sáng tạo hơn và cho kết quả hiệu quả hơn. Cac kiến trúc mạng này đóng vai trò rất quan trọng và áp dụng cho những nhiệm vụ cụ thể trong Computer Vision. Các mạng này được dùng nhiều trong việc trích xuất các feature để phục vụ cho các nhiệm vụ chuyên biệt.



Hình 2.16 Sự phát triển của mạng nơ-ron tích chập

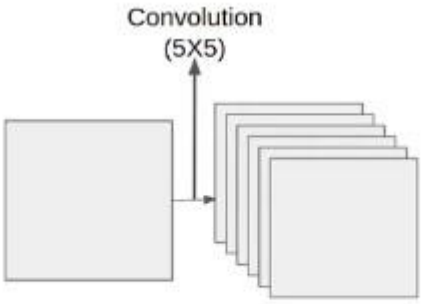
2.5.1 Kiến trúc LeNet-5

LeNet-5 là một trong những mô hình được đào tạo trước sớm nhất, được Yann LeCun và những người khác đề xuất vào năm 1998. Họ sử dụng kiến trúc này để nhận dạng các kí tự viết tay và in bằng máy. Lý do chính đằng sau sự phổ biến của mô hình này là kiến trúc đơn giản và dễ hiểu của nó. Nó là một mạng lưới nơ ron tích chập nhiều lớp để phân loại hình ảnh.

Mạng có 5 lớp với các tham số có thể học được và do đó được đặt tên là Lenet-5. Nó có ba bộ lớp chập với sự kết hợp của việc gộp trung bình. Sau lớp tích chập và lớp gộp trung bình, chúng ta có hai lớp được kết nối đầy đủ. Cuối cùng, bộ phân loại Softmax sẽ phân loại hình ảnh thành lớp tương ứng.

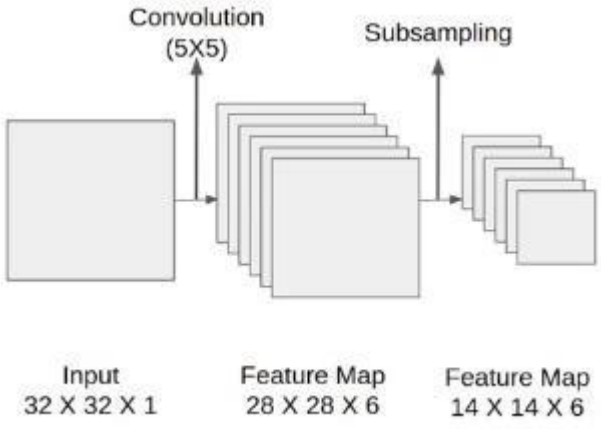
Input
32 X 32 X 1

Đầu vào của mô hình này là hình ảnh thang độ xám 32 x 32 do đó số lượng kênh là một

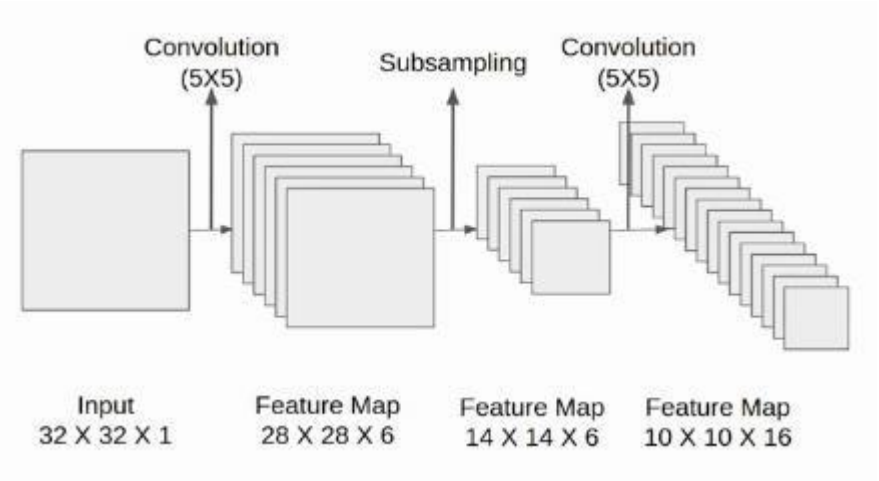


Output shape = ((32-5+1) X (32-5+1) X 6)
= (28 X 28 X 6)

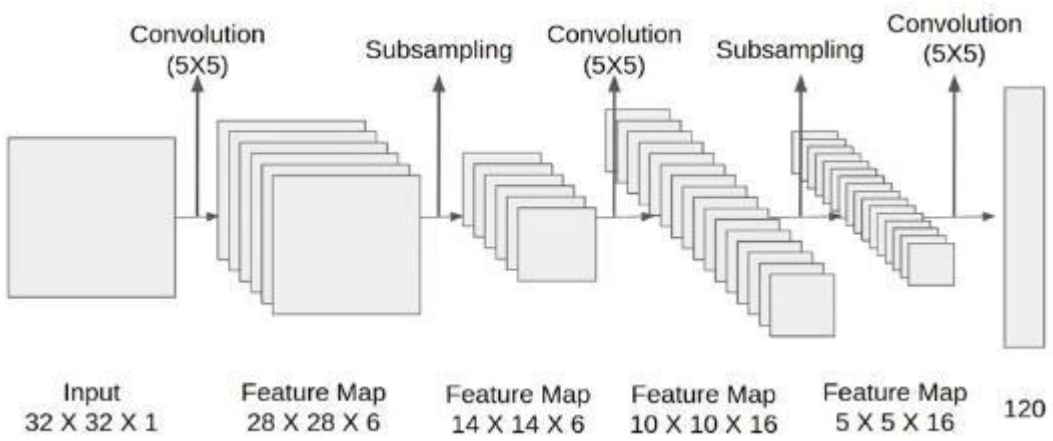
Sau đó, ta thao tác tích chập đầu tiên với kích thước bộ lọc 5xxx5 và ta sẽ có 6 bộ lọc như vậy. Kết quả là chúng ta có được bản đồ đặc trưng có kích thước 28x28x6. Ở đây số lượng kênh bằng số lượng bộ lọc được áp dụng.



Sau thao tác gộp đầu tiên, chúng ta áp dụng phép gộp trung bình và kích thước của bản đồ đối tượng địa lý giảm đi một nửa. Lưu ý rằng, số lượng kênh được giữ nguyên.



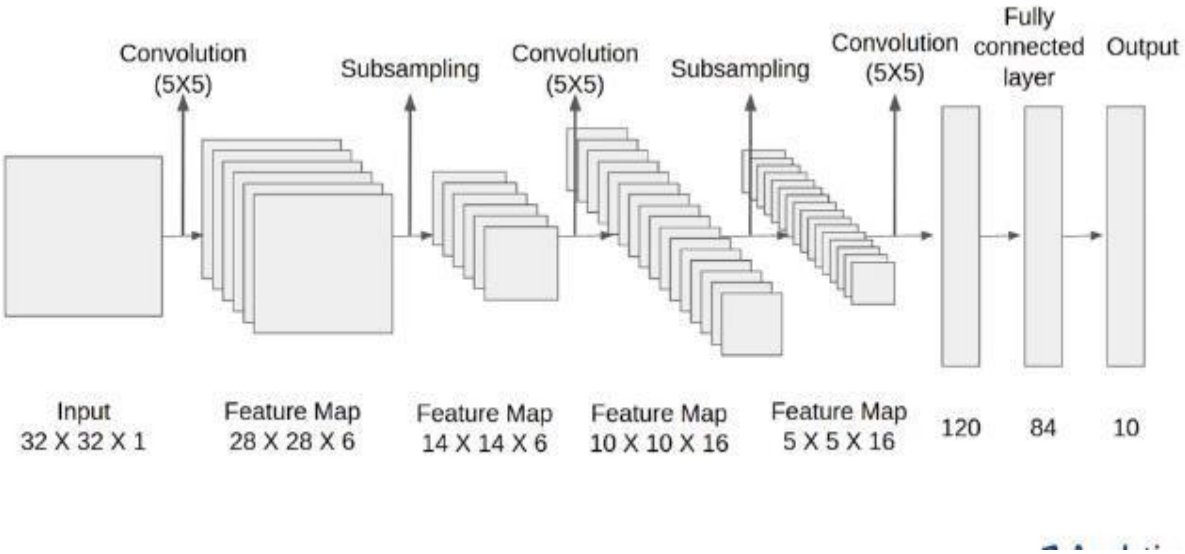
Tiếp theo, chúng ta có một lớp tích chập với mười sáu bộ lọc có kích thước 5x5. Một lần nữa bản đồ tính năng đã thay đổi nó là 10x10x16. Kích thước đầu ra được tính theo cách tương tự. Sau đó, lại áp dụng lớp tổng hợp hoặc lấy mẫu con trung bình, điều này một lần nữa làm giảm kích thước của bản đồ đặc trưng xuống một nửa, tức là 5x5x16.



Tiếp đến, chúng ta có lớp tích chập cuối cùng có kích thước 5x5 với 120 bộ lọc. Như thể hiện trong hình ảnh trên. Để lại kích thước bản đồ đặc trưng 1x1x120. Sau đó làm phẳng kết quả là 120 giá trị.

Sau các lớp tích chập này, chúng ta có một lớp được kết nối đầy đủ với 84 nơ ron. Cuối cùng, có một lớp đầu ra với 10 nơ ron vì dữ liệu có 10 lớp.

Đây là kiến trúc cuối cùng của mô hình Lenet-5



Hình 2.17 Mô hình kiến trúc LeNet-5

2.5.2 Kiến trúc AlexNet

AlexNet là một kiến trúc deep leaning và đại diện cho một biến thể của mạng nơ-ron tích chập. Ban đầu nó được đề xuất bởi Alex Krizhevsky trong quá trình nghiên cứu của ông, dưới sự hướng dẫn của Geoffrey E.Hinton, một nhân vật nổi bật trong lĩnh vực nghiên cứu deep leaning. Năm 2012, Alex Krizhevsky đã tham gia thử thách nhận dạng hình ảnh quy mô lớn ImageNet (ILSVRC2012) và sử dụng mô hình AlexNet, đạt tỉ lệ lỗi ấn tượng trong top 5 là 15,3%, vượt qua người về nhì hơn 10,8 điểm phần trăm.

Kiến trúc AlexNet bao gồm tổng cộng 8 lớp

Năm lớp đầu tiên là các lớp chập. Kích thước của các bộ lọc tích chập là 11x11, 5x5, 3x3, 3x3 và 3x3 cho các lớp tích chập tương ứng. Một số lớp tích chập được theo sau bởi các lớp tổng hợp tối đa, giúp giảm kích thước không gian trong khi vẫn giữ được các tính năng quan trọng. Hàm kích hoạt được sử dụng trong mạng là đơn vị tuyến tính chỉnh lưu (ReLu), được biến đến với hiệu suất vượt trội so với hàm sigmoid và tanh.

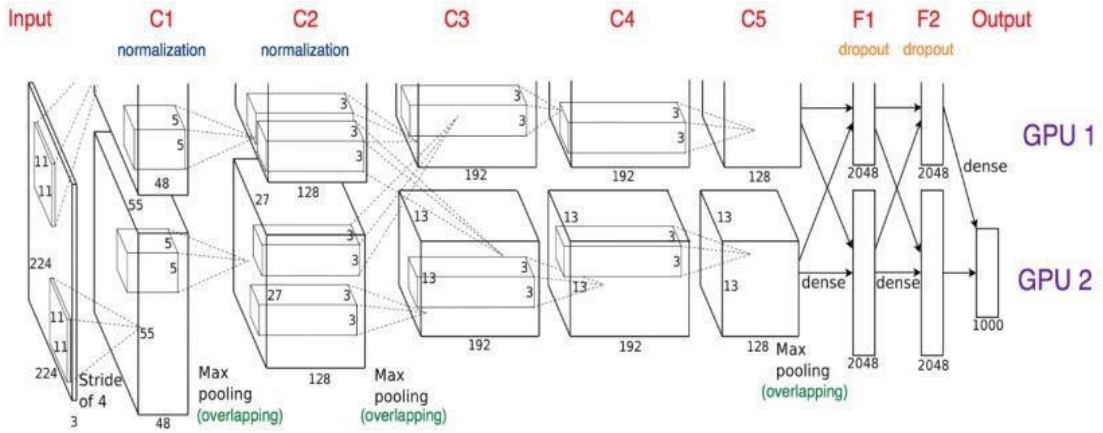
Sau các lớp chập, có ba lớp được kết nối đầy đủ. Các thông số của mạng có thể được điều chỉnh dựa trên hiệu suất huấn luyện. AlexNet có thể được sử

dụng với phương pháp học chuyển giao, sử dụng các trọng số được đào tạo trước trên bộ dữ liệu ImageNet để đạt được hiệu suất vượt trội.

Các thành phần chính của kiến trúc AlexNet :

- *Thành phần 1 - Mạng nơ ron chuyển đổi (CNN)* : Alexnet là kiến trúc mạng nơ ron chuyển đổi (CNN) sâu được thiết kế cho các nhiệm vụ phân loại hình ảnh. CNN đặc biệt phù hợp cho các nhiệm vụ nhận dạng hình ảnh, tận dụng các lớp tích chập để tìm hiểu các tính năng từ hình ảnh theo thứ bậc.
- *Thành phần 2 – Kiến trúc :* AlexNet bao gồm tám lớp, trong đó năm lớp đầu tiên là lớp tích chập và ba lớp cuối cùng là các lớp được kết nối đầy đủ. Các lớp tích chập được thiết kế để trích xuất các mẫu và đặc điểm có liên quan từ hình ảnh đầu vào, trong khi các lớp được kết nối đầy đủ thực hiện phân loại dựa trên các đặc điểm đó.
- *Thành phần 3 – Kích hoạt ReLu :* Các hàm kích hoạt đơn vị tuyến tính chỉnh lưu (ReLU) được sử dụng sau mỗi lớp tích chập và được kết nối đầy đủ. ReLU giới thiệu tính phi tuyến tính, cho phép mạng mô hình hoá các mối quan hệ phức tạp hơn trong dữ liệu.
- *Thành phần 4 – Max Pooling :* Các lớp gộp tối đa được áp dụng sau các lớp chập nhất định để giảm kích thước không gian trong khi vẫn giữ được các tính năng thiết yếu. Quá trình lấy mẫu xuống này giúp giảm tính toán và kiểm soát quá mức.
- *Thành phần 5 – Chuẩn hoá phản hồi cục bộ :* Chuẩn hoá phản hồi cục bộ (LRL) được triển khai để tăng cường khả năng tổng quát hoá bằng cách chuẩn hoá đầu ra của một nơ-ron so với các nơ-ron lân cận. Điều này tạo ra một dạng ức chế ngang, làm cho mạng trở lên mạnh mẽ hơn trước những biến đổi của dữ liệu đầu vào.

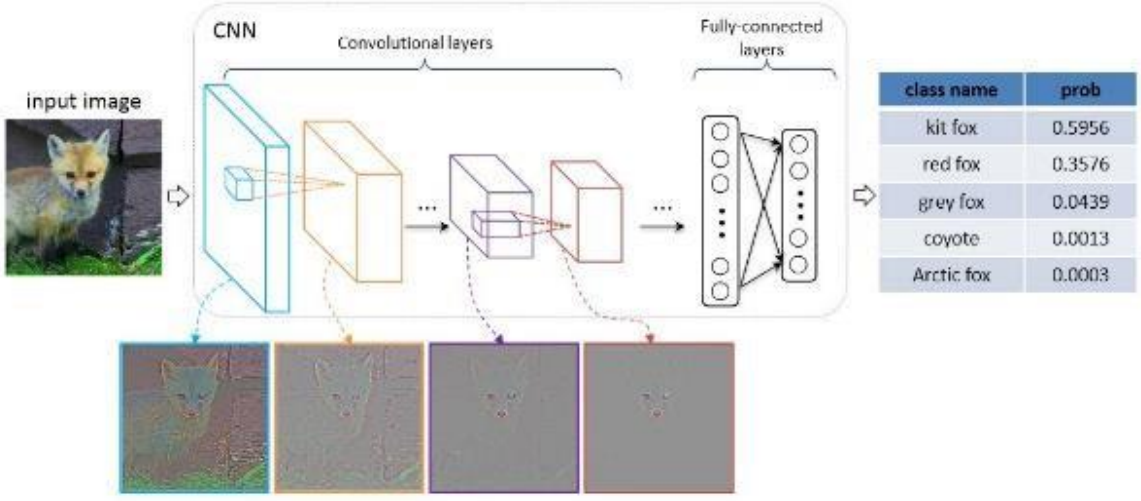
- *Thành phần 6 – Bỏ học :* AlexNet sử dụng chính quy bỏ học trong quá trình đào tạo, trong đó các nơ-ron ngẫu nhiên bị loại bỏ trong quá trình chuyển tiếp và lùi. Kỹ thuật này ngăn chặn việc trang bị quá mức và cải thiện hiệu suất khái quát hoá mô hình.
- *Thành phần 7 – Chuẩn hoá hàng loạt :* Chuẩn hoá hàng loạt được áp dụng để chuẩn hoá đầu ra của từng lớp trong một lô nhỏ trong quá trình đào tạo. Nó ổn định và tăng tốc quá trình đào tạo, cho phép tốc độ học tập cao hơn và kiến trúc sâu hơn.
- *Thành phần 8* – Kích hoạt Softmax : Lớp cuối cùng của AlexNet sử dụng chức năng kích hoạt Softmax để chuyển đổi đầu ra thô của mô hình thành xác xuất của lớp. Điều này cho phép mạng cung cấp phân bố xác suất trên các lớp có thể có cho mỗi hình ảnh đầu vào.
- *Thành phần 9 – Đào tạo rồi tối ưu hoá :* AlexNet được đào tạo bằng cách sử dụng phương pháp giảm độ dốc ngẫu nhiên có động lượng. Tốc độ học được điều chỉnh trong quá trình đào tạo và các kỹ thuật tăng cường dữ liệu được áp dụng để tăng tính đa dạng của tập dữ liệu huấn luyện.
- *Thành phần 10 – Cuộc thi ImageNet :* AlexNet đã đạt được thành công đáng kể khi tham gia thử thách nhận dạng hình ảnh quy mô lớn ImageNet (ILSVRC) vào năm 2012. Hiệu suất vượt trội của nó đã giúp phổ biến deep leaning và CNN cho các nhiệm vụ phân loại hình ảnh.



Hình 2.18 Mô hình kiến trúc AlexNet

2.5.3 Kiến trúc VGG-16

VGG-16Net dựa trên các tính năng thiết yếu nhất của mạng nơ-ron tích chập (CNN)



Hình 2.19 Hình mô tả cách thức hoạt động của VGG-16

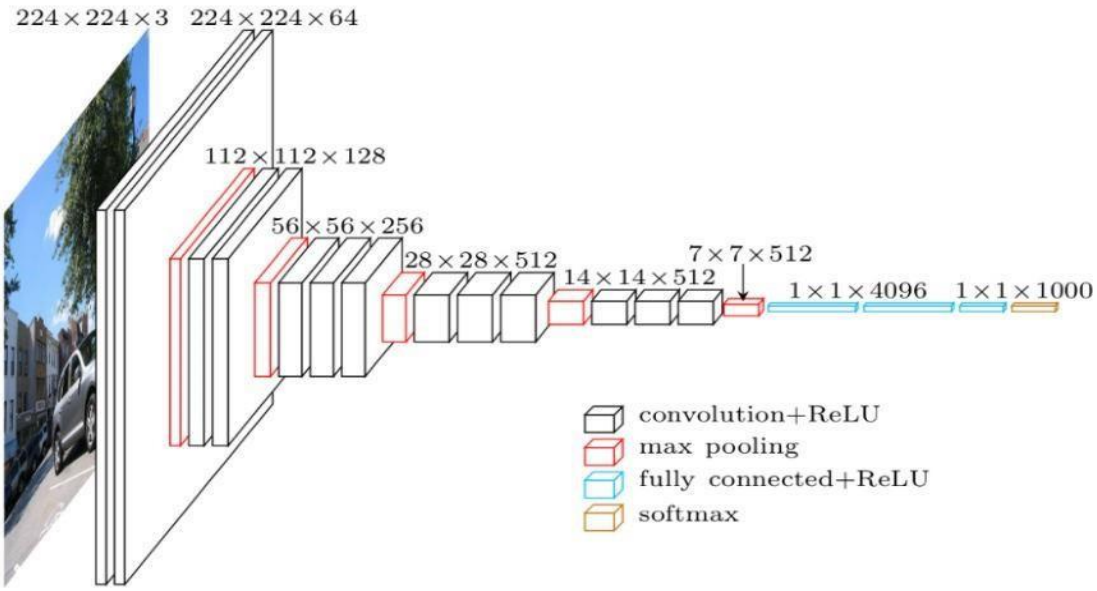
Mạng VGG được xây dựng với các bộ lọc tích chập rất nhỏ. VGG-16 bao gồm 13 lớp chập và ba lớp được kết nối đầy đủ.

Chúng ta hãy nhìn sơ qua về kiến trúc của VGG :

- Đầu vào : VGGNet có kích thước đầu vào hình ảnh là 224x224. Đối với cuộc thi ImageNet, những người tạo ra mô hình đã cắt bỏ phần trung tâm 224x224 trong mỗi hình ảnh để giữ kích thước đầu vào của hình ảnh nhất quán.
- Lớp tích chập : Các lớp tích chập của VGG tận dụng trường tiếp nhận tối thiểu, tức là 3x3 kích thước nhỏ nhất có thể mà vẫn ghi được lên/xuống và trái/phải. Hơn nữa, còn có các bộ lọc tích chập 1x1 hoạt động như một phép biến đổi tuyến tính của đầu vào. Tiếp theo là đơn vị ReLU, đây là một cải tiến lớn của AlexNet giúp giảm thời gian đào tạo. ReLU là viết tắt của hàm kích hoạt đơn vị tuyến tính được chỉnh lưu, nó là một hàm tuyến tính từng phần sẽ xuất ra đầu vào nếu dương. Mặt khác, đầu ra bằng không bước tích chập được cố định ở 1 pixel để giữ nguyên độ phân giải không

gian sau khi tích chập (sải bước là số lần dịch chuyển pixel trên ma trận đầu vào).

- Lớp ẩn : Tất cả các lớp ẩn trong mạng VGG đều sử dụng ReLU. VGG thường không tận dụng chuẩn hoá phản hồi cục bộ (LRN) vì nó làm tăng mức tiêu thụ bộ nhớ và thời gian đào tạo. Hơn nữa, nó không cải thiện độ chính xác tổng thể.
- Các lớp được kết nối đầy đủ : VGGNet có ba lớp được kết nối đầy đủ. Trong số ba lớp, hai lớp đầu tiên có 4096 kênh, mỗi lớp và lớp thứ ba có 1000 kênh, 1 kênh cho mỗi lớp



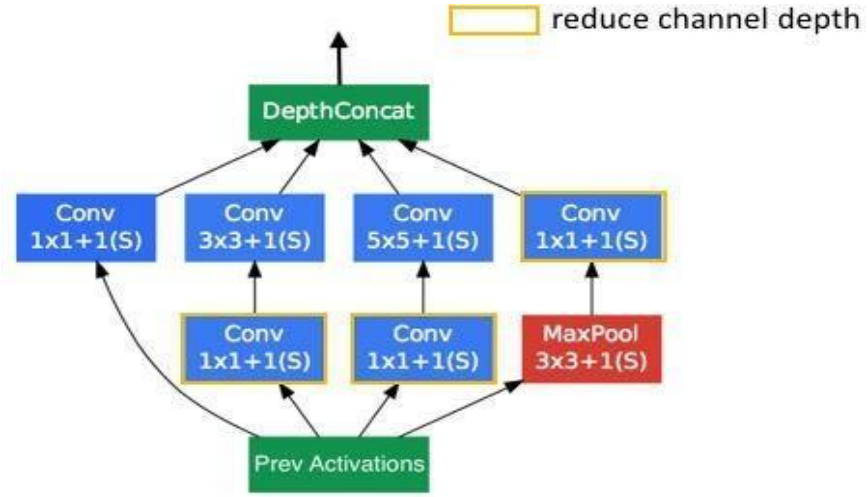
Hình 2.20 Mô hình kiến trúc AGG-16

2.5.4 Kiến trúc Inception (GoogLeNet)

Kiến trúc Inception lần đầu tiên được các nhà nghiên cứu tại google giới thiệu vào năm 2014 như một giải pháp cho vấn đề hạn chế tài nguyên tính toán cho mạng lưới thần kinh sâu. Kiến trúc Inceptron ban đầu còn được gọi là GoogLeNet, được thiết kế để cải thiện hiệu quả của mạng lưới thần kinh sâu bằng cách giảm số lượng tham số mà không làm giảm độ chính xác.

Kiến trúc Inception kết hợp song song các bộ lọc tích chập 1x1, 3x3, và 5x5 để trích xuất các đặc điểm từ hình ảnh đầu vào ở các tỉ lệ khác nhau. Đầu

ra của mỗi bộ lọc sau đó được nối và chuyển sang lớp tiếp theo, nơi quá trình được lặp lại. Cách tiếp cận này cho phép mạng nắm bắt được các đặc điểm cục bộ và toàn cục của hình ảnh đầu vào đồng thời giảm thiểu số lượng tham số. Kể từ khi được giới thiệu, kiến trúc Inception đã trải qua nhiều lần sửa đổi và cải tiến. Vào năm 2015, các nhà nghiên cứu đã giới thiệu kiến trúc Inception – v2 thay thế các bộ lọc tích chập 5x5 bằng hai bộ lọc 3x3 nối tiếp, tạo ra một kiến trúc hiệu quả hơn với ít tham số hơn



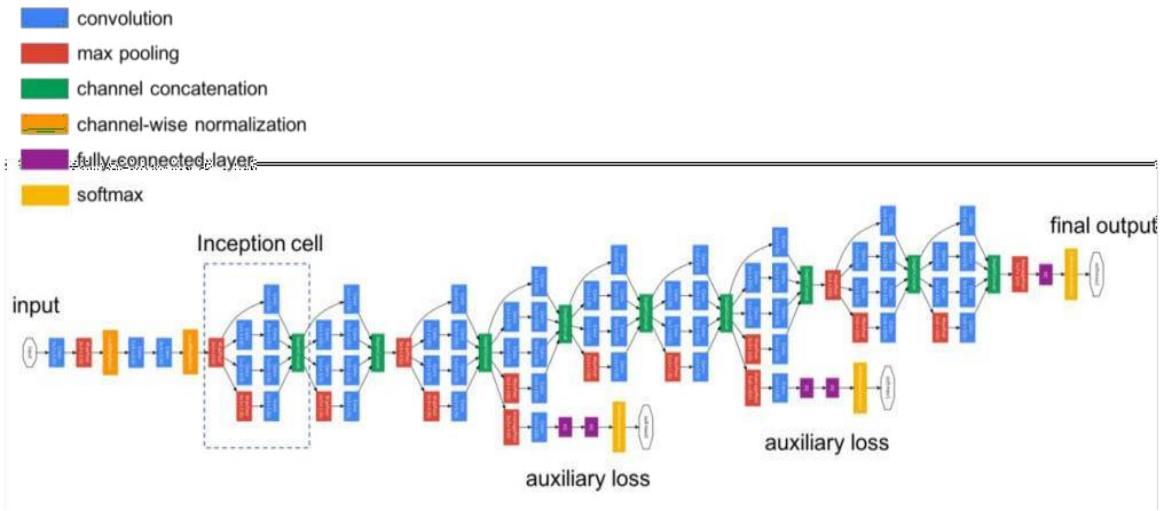
Hình 2.21 Mô hình dạng cell của kiến trúc Inception

Năm 2016, kiến trúc Inception-v3 đã được giới thiệu, kết hợp với một số cải tiến bao gồm chuẩn hoá hàng loạt và giảm các bộ lọc được sử dụng trong mỗi lớp. Kiến trúc này đã đạt được hiệu suất tiên tiến trên bộ dữ liệu ImageNet, một chuẩn mực cho các tác vụ phân loại hình ảnh.

Gần đây, kiến trúc Inception –v4 đã được giới thiệu, cải tiến hơn nữa kiến trúc Inception-v3 bằng cách kết hợp các kết nối còn lại, giống như các kết nối được sử dụng trong kiến trúc ResNet. Kiến trúc này thậm chí còn đạt được hiệu suất tốt hơn trên tập dữ liệu ImageNet, đồng thời hiệu quả hơn các phiên bản trước của kiến trúc Inception. Kiến trúc khởi đầu đã tác động đáng kể đến thị giác máy tính và học sâu về mặt hiệu quả và hiệu suất của nó đối với các nhiệm vụ phân loại hình ảnh đầy thách thức. Sự phát triển của nó theo thời gian đã dẫn đến các phiên bản kiến trúc ngày càng hiệu quả và hiệu quả với phiên

bản mới nhất kết hợp với một số kĩ thuật tiên tiến nhất trong nghiên cứu học sâu.

Dưới đây là kiến trúc mạng Inception, mạng được xây dựng từ việc ghép các inception cell lại với nhau



Hình 2.22 Mô hình kiến trúc Inception

Inception có một đặc điểm khá hay là có thêm 2 output phụ. Người ta tin rằng hai output phụ này không quá ảnh hưởng tới chất lượng của mạng trong khi train những epoch đầu. Nó giúp cho việc train diễn ra nhanh hơn khi tối ưu những layer đầu dựa vào các output phụ (trong những epoch đầu).Có thể nghĩ đơn giản rằng trong những epoch đầu, các layer càng gần cuối càng ít được tối ưu ngay, do đó chưa cần thiết phải tối ưu ngay. Sau một thời gian tối ưu các layer đầu rồi mới tối ưu các layer tiếp theo dựa vào final input. Việc này cải thiện khả năng tính toán và tốc độ train khá nhiều.

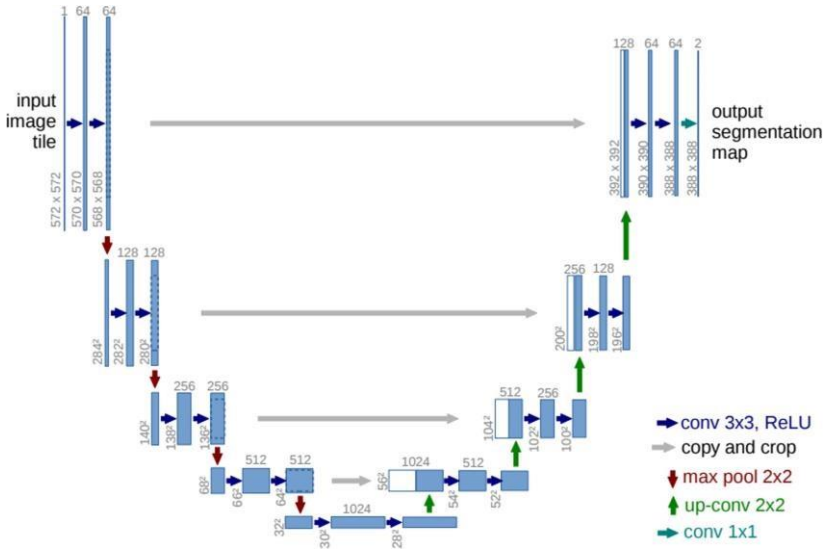
Hiện nay, Inception liên tục được cải tiến và đã cho ra nhiều phiên bản, Inception V1 (5 triệu tham số), Inception V3 (23 triệu tham số), Inception V4.

Ngoài ra còn có những kiến trúc ResNet-50 (2015), sử dụng kết nối tắt để ánh xạ các đầu vào từ những layer trước đó tới những layer sau. Là kiến trúc mạng rất sâu nhưng có số tham số nhỏ hơn nhờ kế thừa những kĩ thuật từ GoogLeNet. Kiến trúc mạng DenseNet (2016), là bước phát triển tiếp theo của

ResNet khi kế thừa kiến trúc khối và phát triển kết nối tắt theo một mạng lưới dày đặc.

Trên đây là điểm qua một số mạng CNN phổ biến được sử dụng trong bài toán Image Classification. Trong thực tế, việc sử dụng kết hợp nhiều phương pháp khác nhau thường sẽ cho hiệu quả tốt hơn.

2.5.2 Kiến trúc U-Net



Hình 2.23 Kiến trúc mô hình U-Net

Kiến trúc mạng bao gồm contracting path (bên trái) và expansive path (bên phải). Trong đó, contracting path tuân theo kiến trúc điển hình của convolutional network. Bao gồm việc lặp lại 2 lần các lớp tích chập 3x3 không padding, theo sau là hàm ReLU (rectified linear unit) và một lớp max pooling 2x2 với stride =2 để downsampling.

Với mỗi lần downsampling, ta nhân đôi số lượng feature channels. Tóm lại, contracting path bao gồm convolutional layers và max pooling layers để downsample input image và trích xuất features. Về expansive path, mỗi bước bao gồm upampling feature map với lớp tích chập 2x2, giảm nửa số lượng feature channels và nối với cropped feature map tương ứng từ contracting path, 2 lớp tích chập 3x3 theo sau bởi hàm ReLU. Cropping là cần thiết do sự mất mát các pixels biên (border pixel) qua mỗi lần tích chập.

Ngoài ra, các “ skip connections” liên kết với các layer tương ứng với contracting path cho phép tích hợp thông tin từ hình ảnh đầu vào mô hình, tăng độ chính xác của segmetation map. Layer cuối cùng sử dụng conv 1x1 áp dụng softmax để cho ra output xác suất pixrl x rơi vào class k. Tóm lại, expansive bao gồm các lớp tích chập và các lớp upsampling, upsample feature maps từ contracting path và kết hợp chúng với feature từ ảnh đầu vào để tạo ra bản đồ phân loại cuối cùng.

Trong hình minh hoạ, mô hình có 23 lớp tích chập và mô hình U-Net hoạt động bằng cách trích xuất các tính năng từ hình ảnh đầu vào và sử dụng contracting path, kết hợp các features này với các tính năng từ ảnh đầu vào sử dụng expanding path và skip connections, và tạo ra bản đồ phân đoạn cuối cùng bằng cách sử dụng các lớp tích chập ở expanding path.

CHƯƠNG 3: ỨNG DỤNG CNN CHO PHÂN ĐOẠN NGỮ NGHĨA

3.1 Môi trường và cài đặt

3.1.1 Google colad

- Google Colab với T4 GPU
- Python 3

Google Colab là một phiên bản lưu trữ trên đám mây giống với Jupyter Notebook do Google Research phát triển. Bằng cách sử dụng Google Colab, ta không cần cài đặt và nâng cấp phần cứng máy tính cá nhân mà vẫn có thể xử lý các công việc nặng về CPU/GPU trong Python. Colab cung cấp miễn phí quyền truy cập vào các hạ tầng điện toán đám mây như bộ lưu trữ, bộ nhớ, GPU, CPU và TPU.

Hình 3.1 GoogleColab

Ưu điểm:

- Miễn phí và dễ sử dụng: Colab là một dịch vụ miễn phí, giúp người dùng có thể thực hiện các dự án máy học mà không cần phải lo lắng về việc cài đặt và quản lý môi trường làm việc.
- Tích hợp với Google Drive: Colab tích hợp chặt chẽ với Google Drive, giúp lưu trữ và chia sẻ dễ dàng dự án của bạn.

- GPU và TPU hỗ trợ: Colab cung cấp truy cập vào GPU (Graphics Processing Unit) và TPU (Tensor Processing Unit) miễn phí, giúp gia tăng tốc độ huấn luyện mô hình machine learning.
- Thư viện và frameworks đã được cài đặt: Nhiều thư viện và frameworks phổ biến như TensorFlow, PyTorch, và OpenCV đã được cài sẵn, giúp giảm bớt bước đầu cài đặt môi trường làm việc.
- Hỗ trợ jupyter notebook: Colab sử dụng định dạng Jupyter Notebook, giúp người dùng tạo, chia sẻ và lưu trữ dễ dàng các tệp notebook của mình.

Nhược điểm:

- Giới hạn tài nguyên: Người dùng miễn phí có giới hạn thời gian sử dụng GPU và TPU, và không thể sử dụng các GPU hàng đợi để giảm thời gian chờ đợi.
- Không đảm bảo ổn định và duy trì dài hạn: Do là dịch vụ miễn phí, không có sự đảm bảo về sự ổn định và tính khả dụng so với dịch vụ trả phí

3.1.2 Cài đặt môi trường Google Colab

Huấn luyện (hay còn gọi là train) một mô hình Deep Learning, cần xử lý lượng phép tính lớn hơn nhiều so với các mô hình Machine Learning khác. Để cải thiện tốc độ tính toán, người ta dùng GPU (Graphics Processing Unit) thay cho CPU (Central Processing Unit) vì với 1 GPU cho phép xử lý nhiều phép tính song song với rất nhiều core sẽ nhanh hơn nhiều so với CPU. Tuy nhiên giá của GPU thì khá đắt đỏ để mua hoặc thuê server có GPU. Thế nên Google đã cung cấp Google Colab miễn phí có GPU để chạy code python (deep learning) cho mục đích nghiên cứu.

Sau khi đăng nhập tài khoản Google, tạo notebook thông qua trang web của Google Colab, thay đổi từ CPU sang GPU thông qua Runtime → Change runtime type → T4 GPU → Save.

3.1.3 Các thư viện sử dụng

OS: Là thư viện để thao tác tệp và thư mục.

Numpy và Pandas: Là thư viện quan trọng trong xử lý dữ liệu và tính toán số học. Numpy hỗ trợ các phép toán số học trên mảng, pandas được sử dụng để làm việc với dữ liệu dạng bảng (DataFrame).

Matplotlib: Là thư viện trực quan hóa dữ liệu. Matplotlib thường được sử dụng cho các biểu đồ cơ bản.

CV2: Là thư viện xủ lý hình ảnh của OpenCV.

PIL: Là thư viện xử lý hình ảnh của Python.

TensorFlow / Keras: Là các thư viện mã nguồn mở được phát triển bởi Google, được sử dụng để xây dựng các mô hình học máy và mạng nơ-ron.

Sequential: Là một trong các phương thức từ keras để xây dựng mô hình. Sequential được sử dụng để xây dựng mô hình theo kiểu tuần tự.

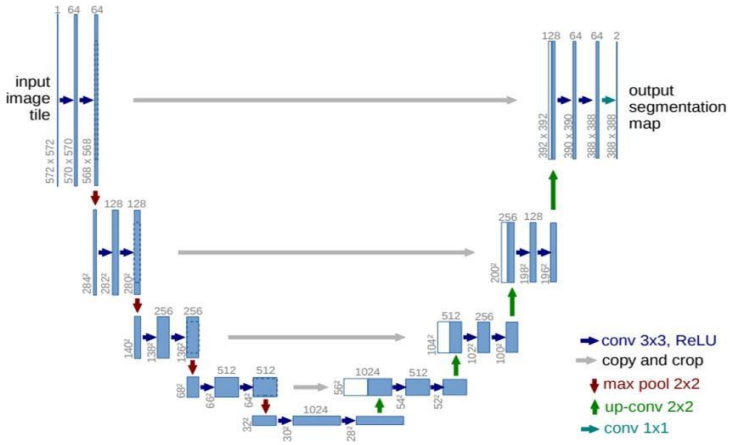
ImageDataGenerator: Là một phương pháp trong keras giúp tạo ra các biến thể của ảnh từ dữ liệu huấn luyện, giúp tăng cường hình ảnh (Image augmentation) và tránh overfitting.

Conv2D, MaxPooling2D, BatchNormalization, Dense,DropOut, Flatten: Là các lớp liên quan đến việc xây dựng mạng CNN.

ReduceLROnPlateau / EarlyStopping: Là các "Callback". Callback trong Keras là các hàm có thể thực hiện các hành động cụ thể tại các giai đoạn khác nhau trong quá trình huấn luyện. Nếu hiệu suất của model không cải thiện, bạn có thể giảm tỷ lệ học hoặc dừng huấn luyện sớm để tránh overfitting.

3.2 Lựa chọn mô hình thử nghiệm

Khi so sánh về số lượng tham số cũng như độ phức tạp cài đặt giữa các phương pháp, đồ án chọn cài đặt thử nghiệm trên mô hình U-Net (trình bày trong phần 2.5.5)



Hình 2.25 Kiến trúc mô hình U-Net

3.2.1 Xây dựng tập dữ liệu thử nghiệm

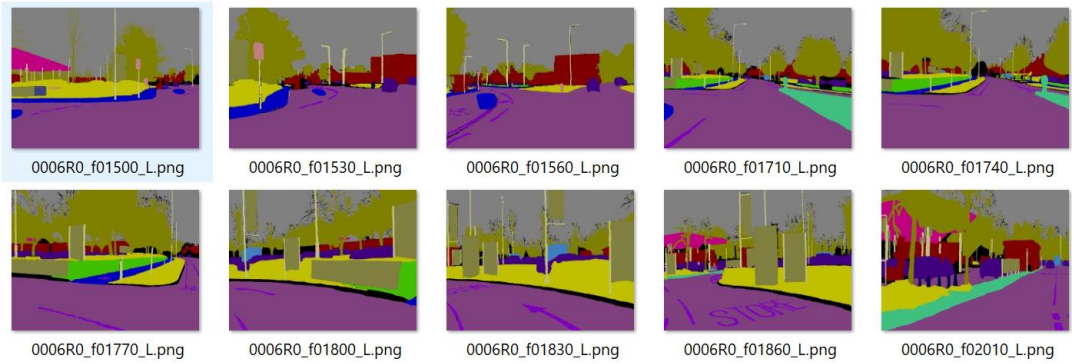
Bộ dữ liệu được sử dụng được lấy từ Cambridge-driving Labeled Video Database (CamVid) gồm 700 ảnh có các nhãn ngữ nghĩa về lớp đối tượng, kèm theo dữ liệu mô tả chi tiết. Cơ sở dữ liệu cung cấp nhãn đúng cho từng pixel, liên kết với một trong 32 lớp ngữ nghĩa.

Cơ sở dữ liệu này đáp ứng nhu cầu về dữ liệu thực nghiệm để đánh giá một cách định lượng các thuật toán liên quan đến phân đoạn ngữ nghĩa. Dữ liệu này được thu từ góc nhìn của một chiếc ô tô đang di chuyển. Tình huống lái xe tăng cường số lượng và đa dạng của các lớp đối tượng được quan sát. Tập dữ liệu 3 phần:

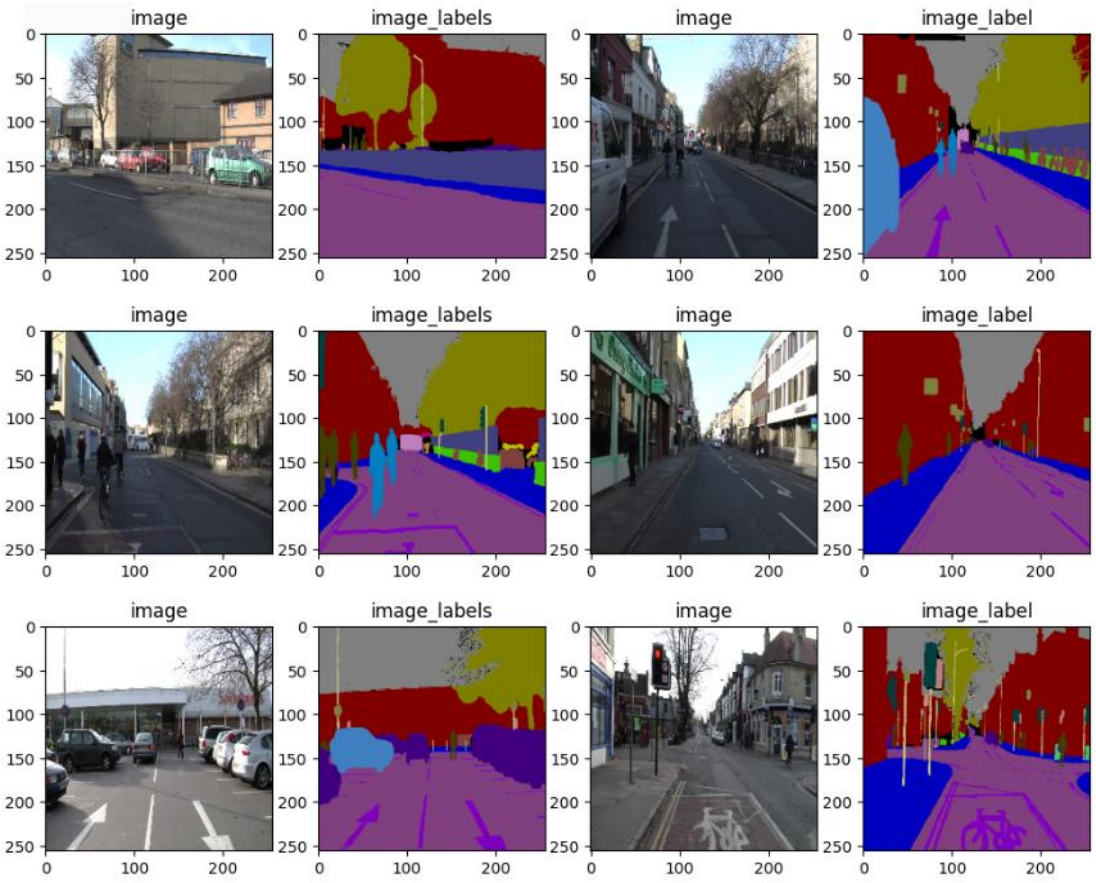
- Dữ liệu cho bước huấn luyện: 300 ảnh (150 ảnh gốc và kèm theo 150 ảnh được gán nhãn)
- Dữ liệu cho bước đánh giá: 100 ảnh (100 ảnh gốc và kèm theo 100 ảnh được gán nhãn)
- 231 ảnh cho bước kiểm thử (231 ảnh gốc và kèm theo 231 ảnh được gán nhãn)

Hình sau minh họa một số ảnh trong tập huấn luyện

3.2 Ảnh gốc



3.3 Ảnh được gán nhãn



3.4 Một số cặp ảnh cùng nhãn

Tập dữ liệu thử nghiệm có đặc điểm mỗi ảnh chứa các hình ảnh của thành phố từ góc nhìn của một ô tô. Tập huấn luyện là tập dữ liệu được sử dụng để huấn luyện mô hình. Tập đánh giá và kiểm thử bao gồm các ảnh không trùng với ảnh trong tập huấn luyện.

Tiền xử lý ảnh đầu vào: Dữ liệu đưa vào huấn luyện là 2 tập ảnh *train* và *train_labels*, trong đó *train* là tập ảnh đầu vào, *train_labels* là tập nhãn. Ảnh ban đầu sẽ có kích thước 960x720, nên để thuận lợi trong quá trình huấn luyện và so sánh với tập nhãn thì các ảnh sẽ được thay đổi kích thước xuống 256x256.

3.2.2 Bước huấn luyện và lưu mô hình

Bộ tham số khởi đầu: Trong mô hình thử nghiệm này, đồ án sử dụng bộ tham số khởi đầu của mạng U-Net được dùng trong huấn luyện mạng cho bài toán phân loại.

```
def GiveMeUnet(inputImage, numFilters = 16, droupouts = 0.1,
doBatchNorm = True):
    # Định nghĩa phần encoder
    c1 = Conv2dBlock(inputImage, numFilters * 1, kernelSize = 3,
doBatchNorm = doBatchNorm)
    p1 = tf.keras.layers.MaxPooling2D((2,2))(c1)
    p1 = tf.keras.layers.Dropout(droupouts)(p1)

    c2 = Conv2dBlock(p1, numFilters * 2, kernelSize = 3, doBatchNorm
= doBatchNorm)
    p2 = tf.keras.layers.MaxPooling2D((2,2))(c2)
    p2 = tf.keras.layers.Dropout(droupouts)(p2)

    c3 = Conv2dBlock(p2, numFilters * 4, kernelSize = 3, doBatchNorm
= doBatchNorm)
    p3 = tf.keras.layers.MaxPooling2D((2,2))(c3)
    p3 = tf.keras.layers.Dropout(droupouts)(p3)

    c4 = Conv2dBlock(p3, numFilters * 8, kernelSize = 3, doBatchNorm
= doBatchNorm)
    p4 = tf.keras.layers.MaxPooling2D((2,2))(c4)
    p4 = tf.keras.layers.Dropout(droupouts)(p4)

    c5 = Conv2dBlock(p4, numFilters * 16, kernelSize = 3, doBatchNorm
= doBatchNorm)

    # Định nghĩa phần  decoder
```

```
    u6 = tf.keras.layers.Conv2DTranspose(numFilters*8, (3, 3),
strides = (2, 2), padding = 'same')(c5)
    u6 = tf.keras.layers.concatenate([u6, c4])
    u6 = tf.keras.layers.Dropout(droupouts)(u6)
    c6 = Conv2dBlock(u6, numFilters * 8, kernelSize = 3, doBatchNorm
= doBatchNorm)

    u7 = tf.keras.layers.Conv2DTranspose(numFilters*4, (3, 3),
strides = (2, 2), padding = 'same')(c6)

    u7 = tf.keras.layers.concatenate([u7, c3])
    u7 = tf.keras.layers.Dropout(droupouts)(u7)
    c7 = Conv2dBlock(u7, numFilters * 4, kernelSize = 3, doBatchNorm
= doBatchNorm)

    u8 = tf.keras.layers.Conv2DTranspose(numFilters*2, (3, 3),
strides = (2, 2), padding = 'same')(c7)
    u8 = tf.keras.layers.concatenate([u8, c2])
    u8 = tf.keras.layers.Dropout(droupouts)(u8)
    c8 = Conv2dBlock(u8, numFilters * 2, kernelSize = 3, doBatchNorm
= doBatchNorm)

    u9 = tf.keras.layers.Conv2DTranspose(numFilters*1, (3, 3),
strides = (2, 2), padding = 'same')(c8)
    u9 = tf.keras.layers.concatenate([u9, c1])
    u9 = tf.keras.layers.Dropout(droupouts)(u9)
    c9 = Conv2dBlock(u9, numFilters * 1, kernelSize = 3, doBatchNorm
= doBatchNorm)

    output = tf.keras.layers.Conv2D(3, (1, 1), activation =
'sigmoid')(c9)
    model = tf.keras.Model(inputs = [inputImage], outputs = [output])
    return model
```

Các tham số cho mô hình

```
#Các tham số
num_epochs=10
batch=32
l_rate=0.01
# Đầu vào của mô hình
inputs = tf.keras.layers.Input((256, 256, 3))
myTransformer = GiveMeUnet(inputs, droupouts=0.07)

#Tham số cho tối ưu
optimizer = tf.keras.optimizers.Adam(learning_rate=l_rate)
# Birn dịch mô hình
myTransformer.compile(optimizer='Adam', loss='binary_crossentropy',
metrics=['accuracy'])
```

Phương pháp học: Sử dụng thuật toán tối ưu hóa Adam để điều chỉnh trọng số của mô hình dựa trên độ dốc của hàm mất mát. Thuật toán Adam là một phương pháp tối ưu hóa gradient dựa trên sự kết hợp của hai kỹ thuật: Momentum (động lượng) và RMSprop (Root Mean Square Propagation).

Huấn luyện và lưu mô hình

```
retVal = myTransformer.fit(
    np.array(framObjTrain['img']),
    np.array(framObjTrain['mask']),
    epochs=num_epochs,
    batch_size=batch,
    validation_data=(np.array(framObjValidation['img']),
np.array(framObjValidation['mask'])),  # Add validation data
    verbose=1  # Set verbose to 1 to see training progress
)
myTransformer.save("/content/drive/MyDrive/CamVid/model.h5")
```

3.3 Đánh giá mô hình

Mô hình được đánh giá sử dụng phương thức evaluate của mô hình được xây dựng trong thư viện Keras của tensorflow. Keras tự động thực hiện tính toán độ chính xác trong quá trình lặp huấn luyện, cung cấp một cách thuận tiện để theo dõi và đánh giá hiệu suất của mô hình trên cả tập dữ liệu huấn luyện và kiểm tra cho các nhiệm vụ phân loại nhị phân như phân đoạn hình ảnh.

Giá trị hàm lỗi trên bộ dữ liệu huấn luyện và đánh giá giảm và duy trì ổn định trong quá trình huấn luyện, tuy nhiên số epoch còn hạn chế, do đó giá trị loss có thể giảm hơn nữa nếu tăng số epoch.

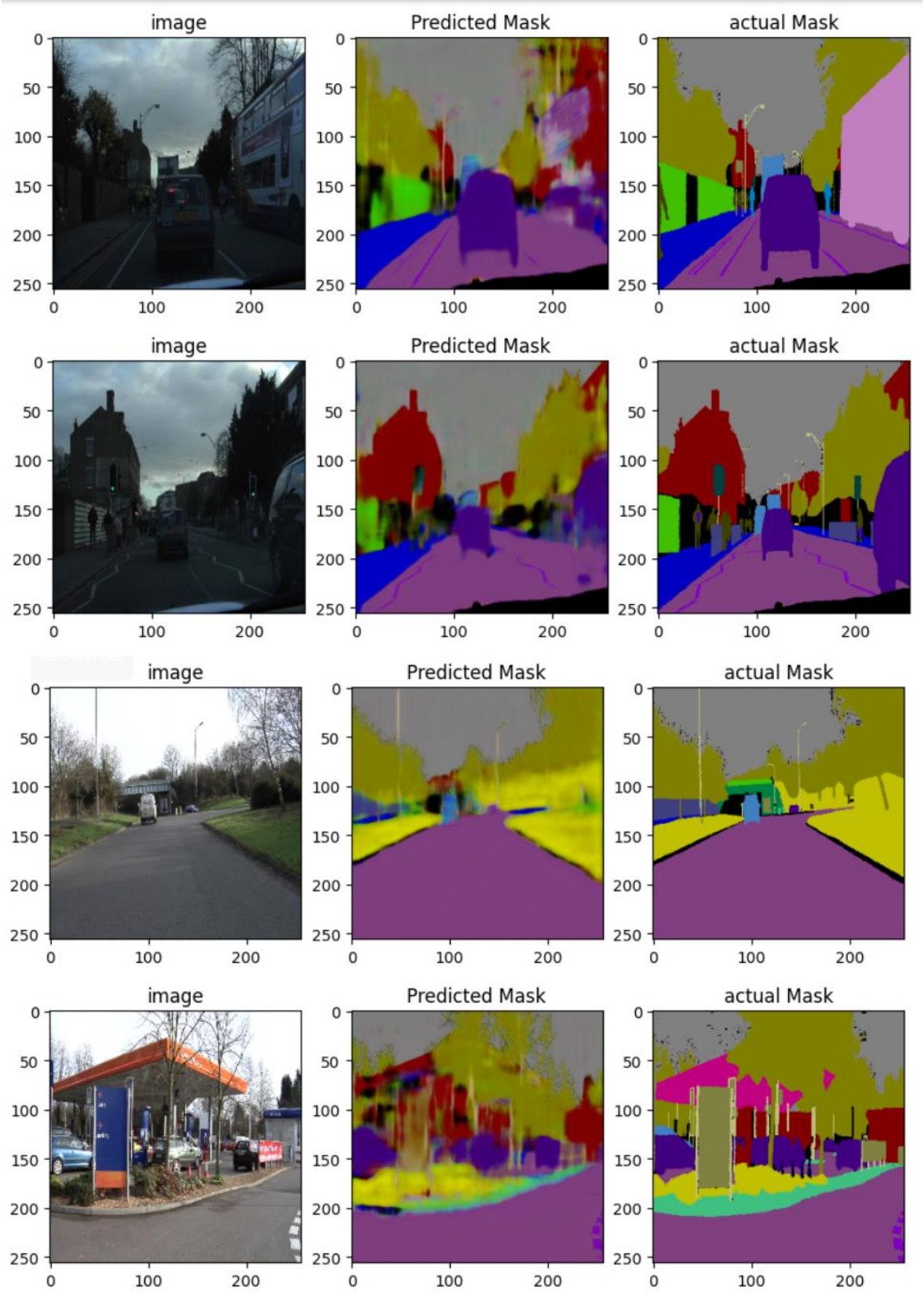
```
#Đánh giá độ chính xác của mô hình
loss,acc=model.evaluate(np.array(framObjValidation['img']),
    np.array(framObjValidation['mask']),batch_size=32, verbose=0)
print('The accuracy of the model for testing data is:',acc*100)
print('The Loss of the model for testing data is:',loss)
```

```
The accuracy of the model for testing data is: 73.97396564483643
The Loss of the model for testing data is: 0.5288674831390381
```

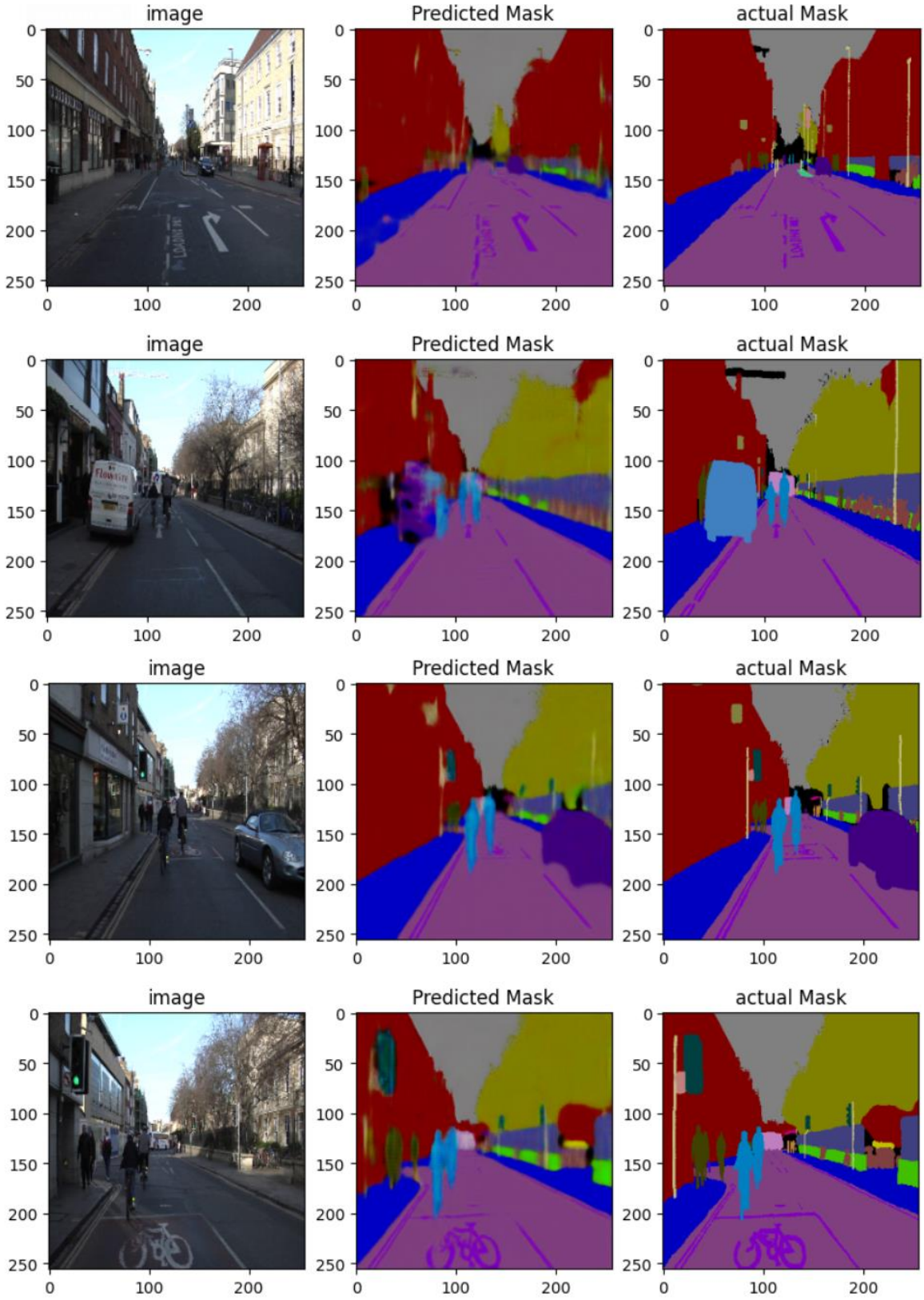
Kết quả: Bộ tham số huấn luyện trên mô hình U-Net thu được kết quả độ chính xác là 73,97%. Tỷ lệ này chưa thật sự tốt trên bộ dữ liệu đánh giá do sự hạn chế về mặt cấu hình và số lượng dữ liệu huấn luyện.

3.4 Một số kết quả thử nghiệm

3.4.1 Kết quả kiểm thử trong tập dữ liệu test



3.5 Kết quả test1



3.6 Kết quả test 2

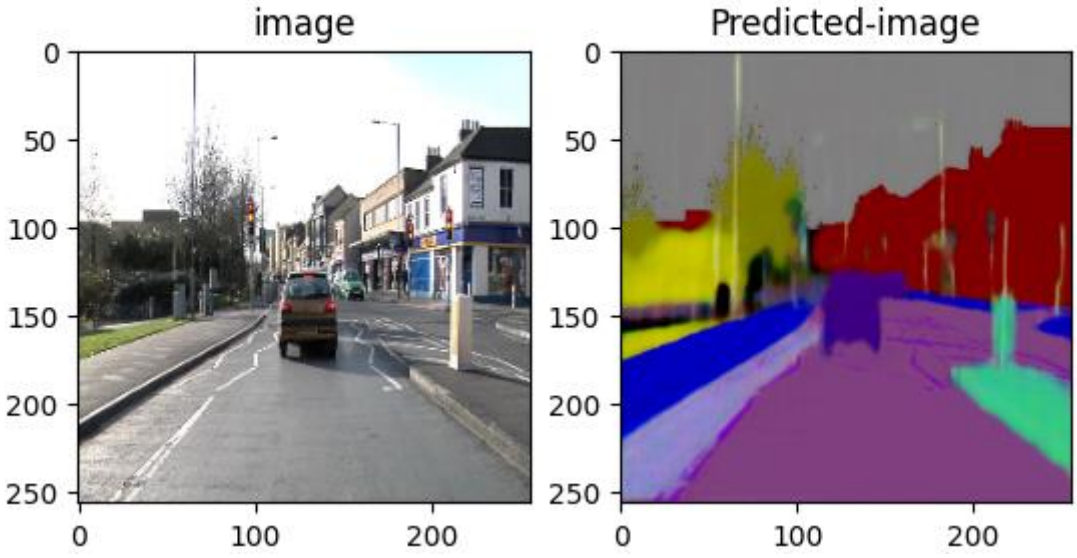
3.4.2 Kiểm thử trên một ảnh

```
# Dự đoán 1 ảnh bất kỳ
imgPath = '/content/drive/MyDrive/CamVid/train/0016E5_01230.png'
```

```
shape=256
framObjTrain_single = {'img' : [],
           'mask' : []
          }
img = plt.imread(imgPath)
img = cv2.resize(img, (shape, shape))
framObjTrain_single['img'].append(img)
img = framObjTrain_single['img']
imgProc =np.array(img[0:1])
predictions1 = model.predict(imgProc)
for i in range(len(predictions1)):
    predictions1[i] =
cv2.merge((predictions1[i,:,:,0],predictions1[i,:,:,1],predictions1[i
,:,:,2]))

plt.subplot(1,2,1)
plt.imshow(img[0])
plt.title('image')
plt.subplot(1,2,2)
plt.imshow(predictions1[0])
plt.title('Predicted-image')
```

```
1/1 [==============================] - 0s 172ms/step
Text(0.5, 1.0, 'Predicted-image')
```



Hình 3.7 Kết quả test 1 ảnh riêng

KẾT LUẬN

Thông qua đồ án tốt nghiệp với đề tài "Phân đoạn ngữ nghĩa sử dụng mạng nơ-ron tích chập" em đã có được một số kết quả và có một số hạn chế như sau:

Kết quả đạt được:

- Tìm hiểu tổng quan về Machine learning và Deep learning.
- Tìm hiểu về ngôn ngữ lập trình Python, chương trình chạy phụ trợ TensorFlow.
- Tìm hiểu về mạng CNN, bài toán phân đoạn ảnh nghữ nghĩa
- Xây dựng mô hình CNN ứng dụng trong phân đoạn ngữ nghĩa
- Đánh giá độ chính xác của mô hình.
- Thử nghiệm mô hình phân đoạn ngữ nghĩa

Một số hạn chế:

- Số lượng ảnh huấn luyện còn ít, bên cạnh đó số lần lặp trong quá trình huấn luyện bị giới hạn do năng lực tính toán của máy tính hạn chế không đáp ứng được. Do vậy độ chính xác còn thấp.
- Mới chỉ dừng lại thử nghiệm xây dựng mô hình UNET đơn giản, chưa so thử nghiệm và sánh với các mô hình hiện đại để tìm ra mô hình phù hợp nhất cho tập dữ liệu này.

Hướng phát triển

- Thử nghiệm và đánh giá với nhiều kiến trúc CNN khác nhau để tìm mô hình phù hợp nhất.
- Tăng số lượng ảnh huấn luyện và sử dụng máy tính có năng lực tính toán cao hơn để tăng độ chính xác của mô hình

Mặc dù cố gắng trong quá trình thực hiện ĐATN để hoàn thiện đồ án một cách tốt nhất nhưng do năng lực, kiến thức và trình độ còn nhiều hạn chế nên em còn nhiều thiết sót. Em rất mong thầy cô có thể giúp đỡ để em hoàn thiện hơn. Em đã cố gắng trong quá trình thực hiện ĐATN để hoàn thiện đồ án một cách tốt nhất nhưng do năng lực, kiến thức và trình độ còn nhiều hạn chế nên em còn nhiều thiết sót.

Em rất mong thầy cô có thể giúp đỡ để em hoàn thiện hơn

TÀI LIỆU THAM KHẢO

[1.] Tuấn, Nguyễn Thanh. *Deep Learning Cơ Bản.* Hà Nội : NXB Thanh Niên, August 2020.

[2.] *What is machine learning?* IBM. New York IBM https://www.ibm.com/topics/machine-learning, 2023.

[3.] *Tổng quan về Artificial Neural Network.* Dương, Đỗ. Hà Nội :

[4.] https://viblo.asia/p/tong-quan-ve-artificial-neural-network-1VgZvwYrlAw, 2018.

[5.] *Phân Nhóm Các Thuật Toán Học Máy.* Tùng, Phạm Duy.

[6.] https://www.phamduytung.com/blog/2019-04-19-deep-learning-view/, Hồ Chí Minh : Website, 4/2019.

[7.] Https://www.mathworks.com/help/vision/ug/semantic-segmentation-using-deeplearning.html

[8.] Https://www.mathworks.com/help/vision/ug/getting-started-with-semanticsegmentation-using-deep-learning.html

[9.] Https://phamdinhkhanh.github.io/2020/06/10/ImageSegmention.html

[10.] Https://viso.ai/deep-learning/image-segmentation-using-deep-learning/