# MỤC LỤC

[MỤC LỤC 1](#_Toc32007686)

[DANH MỤC HÌNH ẢNH 2](#_Toc32007687)

[MỞ ĐẦU 3](#_Toc32007688)

[PHẦN I- TỔNG QUAN 6](#_Toc32007689)

[**1.** **Vấn đề** 6](#_Toc32007690)

[**2.** **Giải quyết vấn đề** 6](#_Toc32007691)

[**3.** **Quá trình thực hiện** 8](#_Toc32007692)

[PHẦN II- THU THẬP DỮ LIỆU 10](#_Toc32007693)

[1. **Phân tích yêu cầu** 10](#_Toc32007694)

[**2.** **Thu thập Dataset** 10](#_Toc32007695)

[**2.1.** **Tools mining images** 10](#_Toc32007696)

[**2.2.** **Dữ liệu thực tế** 11](#_Toc32007697)

[**3.** **Tiền xử lý dữ liệu** 12](#_Toc32007698)

[PHẦN III- TRAINED MODELS 13](#_Toc32007699)

[**1.** **Giới thiệu và Mô hình áp dụng** 13](#_Toc32007700)

[**2.** **Kiến trúc CNN** 14](#_Toc32007701)

[**3.** **Transfer Learning** 16](#_Toc32007702)

[**4.** **Keras** 20](#_Toc32007703)

[**5.** **Quá trình thực hiện** 21](#_Toc32007704)

[6. Xây dựng ứng dụng 26](#_Toc32007705)

[**6.1.** **Nạp môi trường Tensorflow và Python** 26](#_Toc32007706)

[**6.2.** **Nhúng mô hình vào ứng dụng** 27](#_Toc32007707)

[**6.3.** **Xử lý ảnh từ Internet và xuất kết quả** 28](#_Toc32007708)

[KẾT LUẬN 30](#_Toc32007709)

[TÀI LIỆU THAM KHẢO 32](#_Toc32007710)

# DANH MỤC HÌNH ẢNH

[Hình 1: Mối quan hệ giữa AI, Machine Learning và Deep Learning 3](file:///C:\Users\DELL\Desktop\DACNA-07\DACN_A.docx#_Toc32007711)

[Hình 2: Các thư viện của các hãng lớn 5](#_Toc32007712)

[Hình 3: Ảnh chó và mèo trong Image Classification 7](#_Toc32007713)

[Hình 4: Công cụ download image 11](#_Toc32007714)

[Hình 5: Trích xuất đặc trưng của ảnh 14](#_Toc32007715)

[Hình 6: Lấy max-pooling với độ trượt bằng 2 15](#_Toc32007716)

[Hình 7: Lớp Dense trong CNN 16](#_Toc32007717)

[Hình 8: Kiến trúc của AlexNet 17](#_Toc32007718)

[Hình 9: Kiến trúc ZFNet 17](#_Toc32007719)

[Hình 10: Kiến trúc của GoogleLeNet 18](#_Toc32007720)

[Hình 11: Kiến trúc của VGG16 18](#_Toc32007721)

[Hình 12: Huấn luyện Machine Learning truyền thống 19](#_Toc32007722)

[Hình 13: Hai miền dữ liệu khác nhau 20](#_Toc32007723)

[Hình 14: Transfer Learning 20](#_Toc32007724)

[Hình 15: Các framework nổi bật trong năm 2018 21](#_Toc32007725)

[Hình 16: Validation set 23](#_Toc32007726)

[Hình 18: Kết quả xuất ra của mô hình 24](#_Toc32007727)

[Hình 19: Overfitting trong mô hình huấn luyện 25](#_Toc32007728)

[Hình 20: Sử dụng Early Stopping để giảm Overfitting 25](#_Toc32007729)

[Hình 21: Kết quả sau khi sử dụng Early Stopping 26](#_Toc32007730)

[Hình 22: Các lệnh update / upgrade môi trường 26](#_Toc32007731)

[Hình 23: Cài đặt OpenCV 27](#_Toc32007732)

[Hình 24: Cài đặt Tensorflow và Keras 27](#_Toc32007733)

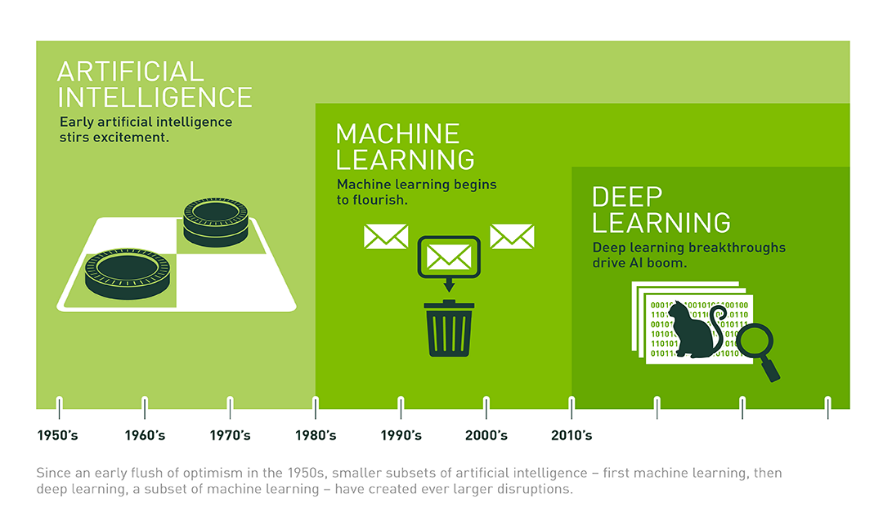
[Hình 25: Script để tải mô hình 28](#_Toc32007734)

[Hình 26: Xử lý kích thước ảnh trước khi dự đoán 29](#_Toc32007735)

[Hình 27: Kết quả sau khi dự đoán 29](#_Toc32007736)

[Hình 28: Object detection 31](#_Toc32007737)

# MỞ ĐẦU

Trong cuộc cách mạng công nghiệp lần thứ tư, Trí tuệ nhân tạo (AI) và Máy học (Machine Learning - ML) là những thuật ngữ nổi bật. Cách tiếp cận với AI cũng khác vì đòi hỏi quá trình từ việc nắm vững các kiến thức cấu trúc dữ liệu và giải thuật cơ bản cho đến các cách giải quyết với những bài toán đạo hàm, hồi quy, …

Hình 1: Mối quan hệ giữa AI, Machine Learning và Deep Learning

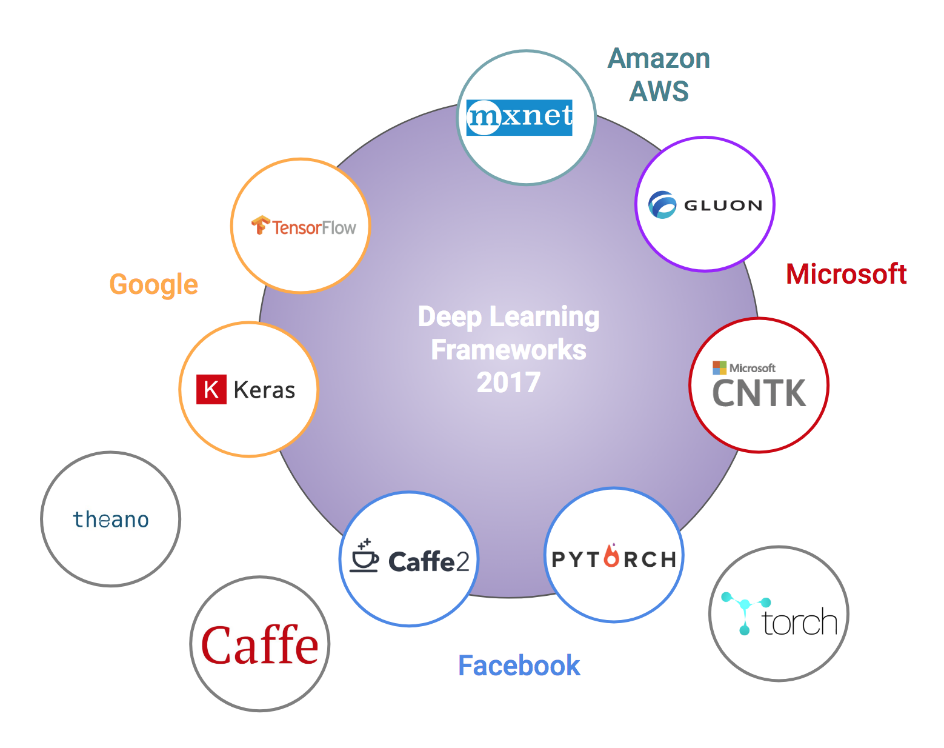
Trí tuệ nhân tạo (Artificial Intelligence) là một thuật ngữ gồm hai từ Artificial và Intelligence, cái tên gồm hai từ đã phần nào diễn giải được ý nghĩa của nó.

* Nhân tạo (Artificial), “nhân tạo” là do con người tạo ra, là cái gì đó không có thật, giống như là mô phỏng lại tư tưởng của con người. Một ví dụ đơn giản cho “nhân tạo” đó là sân cỏ nhân tạo. Có rất nhiều lý do trong việc sử dụng cỏ nhân tạo thay cho cỏ thật là vì tiết kiệm chi phí cắt cỏ, có được khả năng chống chịu cao hơn, sử dụng lâu hơn, … Tạo ra những thứ “nhân tạo” như thế để con người có thể áp dụng vào nhiều mục đích khác nhau mà theo tôi là liên quan đến kinh tế và con người đang trở nên “lười hóa”.
* Trí tuệ (Intelligence) là sự thông tuệ, thông minh. Là một thuật ngữ được cho là phức tạp, không phải là khó đọc mà không có thước đo nào có thể đo được sự thông minh. Có thể hiểu đó là sự hiểu biết, là sự sáng tạo, học tập, lao động, tư duy logic trong việc giải quyết vấn đề. Con người thông minh là do con người có quá trình học hỏi và thực hành dựa vào môi trường nơi con người nhận thức, nơi con người đã khám phá. Động vật cũng vậy, cũng có trí khôn riêng của từng loài nên sự so sánh trí thông minh giữa các loài là sự so sánh khập khiễng.

Ở đây, không phải là tôi đang đưa ra các lý thuyết về các khái niệm trên, mà tôi muốn nói rằng, trí thông minh của con người và động vật là trí thông minh tự nhiên (Natural Intelligence - NI). AI thuộc về mảng Khoa học máy tính, AI được vận hành bởi máy móc (cụ thể, laptop, máy trạm, máy bàn, …) và phần mềm. Được gọi là nhân tạo không phải là ngẫu nhiên, mà đó là sự bắt chước của máy với tư duy của con người truyền cho nó, dựa trên môi trường, quan sát, quá trình học tập.

Hình thành từ khoảng 100 năm về trước nhưng lúc đấy sự tiến bộ của về các thiết bị phần cứng lẫn phần mềm đều chưa đủ đáp ứng được. Từ 1943 - Mô hình toán học đầu tiên của mạng lưới nơ-ron ra đời với sự kết hợp của Walter Pitts và Warren McCulloch, hay là 1957 – Thiết lập nền tảng cho các mạng lưới nơ-ron sâu của Frank Rosenblatt, và sự tiến hóa không ngừng cho đến ngày hôm nay.

Thật may mắn là các tập đoàn công nghệ lớn đã để mắt đến AI và phát triển chúng đến tầm có thể nói là vượt xa trí tưởng tượng của con người (nhưng vẫn trong kiểm soát) – đó là hai robot Alice và Bob của Facebook đã tự giao tiếp với nhau theo “ngôn ngữ robot” nên đã buộc phải ngắt. Các frameworks không thể không nhắc đến là Tensorflow của Google, PyTorch của Facebook, Windows AI của Microsoft, … đó là các công cụ mã nguồn mở phổ biến nhất hiện nay nhằm phục vụ cho sự phát triển của AI.



Hình 2: Các thư viện của các hãng lớn

Nhận định được tầm quan trọng của “thế giới tự động hóa” đang tạo ra viễn cảnh trước mắt, tôi đã tham gia vào nghiên cứu Machine Learning với các công nghệ hỗ trợ từ dịch vụ của Google là Google Colabs, Google Cloud Platform; Keras – API mạng nơ-ron cấp cao.

# PHẦN I- TỔNG QUAN

1. **Vấn đề**

Chúng ta có thể nhận ra rằng khu vực mà thường tập trung đông dân cư nhất luôn là các thành phố đông đúc, náo nhiệt; trái lại đó là những vùng nông thôn yên tĩnh với ruộng lúa, cây cối, ... Sự phân bố dân cư không đều như vậy là một vấn đề nghiêm trọng có thể dẫn đến vấn đề về việc sử dụng đất như xây nhà ở, trồng trọt, kinh doanh, … vì nó ảnh hưởng trực tiếp đến đời sống sinh hoạt của người dân. Ví dụ như mảnh đất đang thuộc quyền sở hữu của C (tên đại diện một người) là thuộc loại đất trồng cây lâu năm, chưa được cấp chuyển sang đất ở đô thị, nếu như C cố ý hoặc vô tình xây nhà thì C đã phạm luật và sẽ bị cưỡng chế bằng việc tháo dỡ. Ngoài ra, khi Nhà nước muốn giải phóng mặt bằng để xây dựng một công trình nào, Sân bay Long Thành chẳng hạn, thì cũng phải tính đến chuyện giải phóng sao tối ưu nhất, phải xem rằng mảnh đất cần xây dựng là đất gì để tính kinh phí đền bù bằng việc liệt kê sẵn các mức giá tương ứng từng loại đất đó. Bên cạnh đó, còn nhiều vấn đề khác nữa liên quan đến quyền sử dụng đất đai ở Việt Nam.

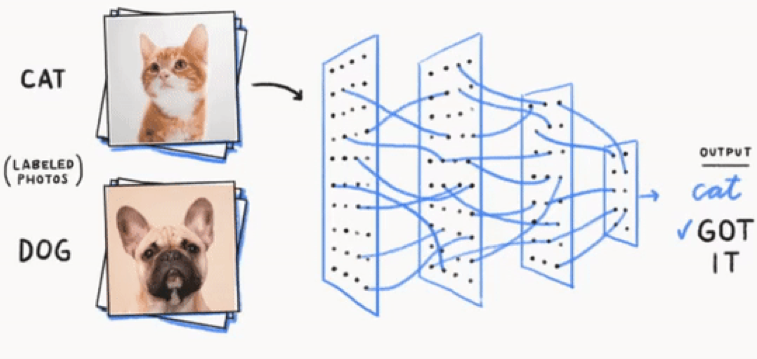
Trước các vấn đề như vậy thì cần có sự hỗ trợ của công nghệ cho việc xử lý các dữ liệu với khối lượng lớn và phức tạp. Công nghệ hỗ trợ việc phân tích, xử lý các dữ liệu lớn và đưa ra kết quả một cách nhanh chóng. Thế nên, trong đồ án chuyên ngành A này, tôi sẽ thực hiện một ứng dụng nhận diện, phân loại đất đai sử dụng ảnh chụp mặt đất. Cụ thể, tôi sẽ phân loại các ảnh chụp đất đai từng loại khác nhau theo quy định của Bộ Tài Nguyên và Môi Trường bằng việc áp dụng Machine Learning.

1. **Giải quyết vấn đề**

Như vấn đề đã trình bày, dựa vào bảng phân loại đất của Bộ Tài Nguyên và Môi Trường, tôi chọn ra 9 phân loại đất để áp dụng là

* Đất chưa sử dụng (BCS)
* Đất trồng cây hằng năm (CHN)
* Đất trồng cây lâu năm (CLN)
* Đất giao thông (DGT)
* Đất khu vui chơi, giải trí công cộng (DKV)
* Đất ở đô thị (ODT)
* Đất khu công nghiệp (SKK)
* Đất sông, ngòi, kênh, rạch, suối (SON)
* Đất nuôi trồng thủy sản nước ngọt (TSN)

Với việc sử dụng ảnh chụp mặt đất và các ảnh lấy được từ Internet thì áp dụng bài toán Phân loại ảnh (Image Classification) là thích hợp trong trường hợp này. Classification là việc gán nhãn cho một đối tượng. Ví dụ tôi có một bộ sưu tập hình gồm ảnh của con chó và con mèo, khi tôi rút một tấm ảnh bất kì từ bộ sưu tập, tôi sẽ phải viết lên tấm hình đó là chó hay là mèo để phân loại. Đây là việc tôi đang truyền tri thức của tôi vào bộ sưu tập bằng cách viết tên của con vật đó lên hình để cho những người không biết (khi xem bộ sưu tập của tôi) thì có thể nhờ vào chữ tôi đã viết, từ đó biết được con đó là con chó hay con mèo.



Hình 3: Ảnh chó và mèo trong Image Classification

Phương pháp giải các bài toán này đã được tìm ra và phát triển nhằm tối ưu việc nhận dạng ảnh một cách chính xác nhất thông qua các cuộc nghiên cứu, hackathon, ... Các mô hình được đánh giá là tốt nhất phải kể đến là

* Xception
* VGG16 - VGG19
* ResNet50
* InceptionV3
* DenseNet
* MobileNet
* …

Đây là các mô hình đã được huấn luyện sẵn thông qua việc thử nghiệm và sai nhiều lần để cho ra được mô hình như vậy. Tận dụng việc tối ưu từ các mô hình này đem lại, tôi áp dụng chúng vào để xử lý vấn đề của mình.

1. **Quá trình thực hiện**

Để hoàn thành đồ án cũng như đáp ứng đúng tiến độ thì tôi cũng đặt ra kế hoạch theo từng giai đoạn để có thể hoàn thành tốt đồ án này. Sau đây là bảng công việc của tôi.

|  |  |  |
| --- | --- | --- |
| Giai đoạn | Từ tuần – đến tuần | Công việc |
| 1 | 01  (09/09 – 15/09) | Tìm hiểu về vấn đề cần giải quyết   * Mục đích của bài toán * Công nghệ áp dụng cho bài toán * Cách lấy dữ liệu như thế nào * Các chuẩn cần lưu ý |
| 2 | 02-08  (16/09 – 27/10) | Thu thập dữ liệu và tiền xử lý dữ liệu |
| 3 | 08-13  (28/10 – 1/12) | * Tìm hiểu về CNN * Xây dựng thử nghiệm mô hình đơn giản * Huấn luyện mô hình. * Tìm hiểu mô hình Xception, Keras. * Tìm hiểu cách sử dụng Google Cloud Platform và cách deploy mô hình lên server. |
| 4 | 13-15  (2/12 – 16/12) | * Testing * Viết báo cáo * Chuẩn bị mô hình để phản biện |

# PHẦN II- THU THẬP DỮ LIỆU

1. **Phân tích yêu cầu**

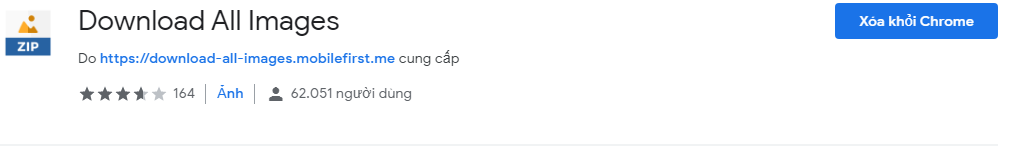
Với những gì đã trình bày bên trên, hiểu rõ mục tiêu là sử dụng bài toán phân loại hình ảnh và ngoài ra cần phải chuẩn bị tập dữ liệu đủ tốt. Vì các mô hình đã qua quá trình thử và sai rất nhiều để tối ưu cho nên tập dữ liệu chưa đủ tốt mà áp dụng mô hình quá phức tạp thì cũng sẽ không được kết quả cao. Khi đó việc giải quyết vấn đề sẽ không hiệu quả, thay vào đó phải thu thập càng nhiều và xử lý càng tốt tập dữ liệu thì sau khi áp dụng mô hình sẽ cho kết quả cao hơn. Vấn đề được nêu ra là giải quyết việc phân loại đất và tôi đã chuẩn bị một số lượng lớn hình ảnh tương ứng 9 phân loại đất tôi đã nêu ra. Những loại này là những loại đất tôi đánh giá độ phổ biến cao, gần gũi. Để cho tập dataset được tốt nhất, tôi phải chọn lọc ra được những bức ảnh có góc chụp ngang, mang tính thực tế và loại bỏ những ảnh chụp từ vệ tinh.

1. **Thu thập Dataset**

Quá trình thu thập dữ liệu, tôi đưa ra các phương án sau để thu thập được dữ liệu hiệu quả đó là kết hợp dữ liệu đi chụp thực tế, dữ liệu được thu thập trên Internet và ảnh được cung cấp từ cộng đồng.

* 1. **Tools mining images**

Trong quá trình thu thập ảnh, tôi đã tìm hiểu về các nguồn ảnh như ImageNet, GoolgeImage, Kaggle, … Sau đó tôi thực hiện việc download thủ công từng ảnh một về máy. Nhận định việc này là rất tốn thời gian nếu như cứ tiếp tục với số lượng ngày càng cần là rất lớn. Từ vấn đề phát sinh, tôi đã tìm hiểu về các công cụ, extensions để hỗ trợ trong việc thu thập ảnh Internet trở nên nhanh chóng. Các extensions tham khảo được trong cửa hàng Chorme đó là Image Downloader, Online Download Manager, Easy AliExpress Image Downloader, Download All Images, … Trong đó Download All Images là extensions có tính dễ sử dụng, nhiều người dùng và có độ tin cậy cao.



Hình 4: Công cụ download image

Sau khi tiện ích được thêm vào, tôi truy cập Google Images và tìm kiếm ảnh mà tôi cần bằng cách sử dụng từ khóa tiếng việt và tiếng anh để tìm kiếm, ví dụ như “bắp”, “corn”, … Sau đó, tôi nhấn vào icon để download ảnh, tiện ích sẽ thực hiện “crawl dữ liệu” các ảnh trên website hiện tại về và hiển thị dưới dạng file nén (.zip). Các ảnh trên sau khi giải nén thì có rất nhiều ảnh không đúng với mong muốn, ảnh bị hư không mở được do quá trình tải. Tiến hành lọc các ảnh không sử dụng như ảnh từ vệ tinh, ảnh không liên quan đến chủ đề đang cần, ảnh không có tính thực tế. Như vậy, tôi đã giải quyết được việc thu thập ảnh trên Internet nhờ vào tiện ích trên.

* 1. **Dữ liệu thực tế**

Việc thu thập dữ liệu thực tế để mang lại thực tế và chính xác của dữ liệu, trên Internet thì crowdsourcing về vấn đề này rất ít nên tôi đã cùng các đồng nghiệp đi thu thập ảnh thực tế bằng cách chụp lại các ảnh liên quan đến dữ liệu đang cần. Và hiện nay thì lượng ảnh đã được thống kê bao gồm ảnh thực tế và ảnh từ Internet như sau

|  |  |
| --- | --- |
| **Tên** | **Số lượng** |
| Đất chưa sử dụng (BCS) | 520 |
| Đất trồng cây hằng năm (CHN) | 1,801 |
| Đất trồng cây lâu năm (CLN) | 2,266 |
| Đất giao thông (DGT) | 1,725 |
| Đất khu vui chơi, giải trí công cộng (DKV) | 676 |
| Đất ở đô thị (ODT) | 717 |
| Đất khu công nghiệp (SKK) | 1,034 |
| Đất sông, ngòi, kênh, rạch, suối (SON) | 763 |
| Đất nuôi trồng thủy sản nước ngọt (TSN) | 636 |

1. **Tiền xử lý dữ liệu**

Tiền xử lý dữ liệu được cho là bước rất quan trọng trước khi đưa vào mô hình huấn luyện, nó quyết định đến tính đúng đắn của kết quả đầu ra cao. Hiểu được tính quan trọng của giai đoạn này nên tôi đã dành thời gian lọc lại tất cả các ảnh trong các thư mục trên. Mỗi thư mục tôi đặt tên trùng với tên phân loại đất tôi đã chọn. Ngoài ra, tôi tạo thêm một thư mục Deleted\_pics để lưu lại các bức ảnh tôi đã xóa để lọc lại xem có thể sử dụng được hay không vì có thể lúc đó tôi vì nguyên nhân nào đó mà xóa nhầm. Bên cạnh đó, để không bị lặp tên của hình khi gom chung lại một tệp, tôi đã dùng script để quy về thành một thể đặt tên thống nhất và cũng nhờ đó sẽ dễ xử lý về sau.

# PHẦN III- TRAINED MODELS

1. **Giới thiệu và Mô hình áp dụng**

Trong các vấn đề tôi đã đề cập, phân loại và ghi nhãn là quan trọng nhất cần được ưu tiên để xử lý trong dự án này. Dữ liệu đầu vào được định nghĩa là một bức ảnh chụp, kết quả là để xác định loại đất mà hình ảnh nói về (đất chưa sử dụng, đất cây hằng năm, cây lâu năm, ...). Vấn đề này tương tự như vấn đề phân loại hình ảnh. Do đó, giải pháp hiện tại là sử dụng các kỹ thuật để trích xuất các đặc trưng trên hình ảnh và phân loại chúng. Nên tôi sử dụng mạng nơ-ron tích chập (CNN) trong phân loại hình ảnh. CNN hay Mạng nơ-ron tích chập là một trong những mô hình Deep learning đang được ứng dụng phổ biến. CNN giúp xây dựng những hệ thống xử lý ảnh thông minh như là Facebook, Google, Amazon đã áp dụng cho các hệ thống của họ trong việc xử lý nhận diện khuôn mặt, hệ thống xe tự động, … Thấy được những tiện ích và xử lý mạnh mẽ về hình ảnh thì công việc tôi cần làm là nghiên cứu về CNN và cách để xây dựng CNN. Qua quá trình tìm hiểu thì Keras là thích hợp để xây dựng với CNN. Trước khi bắt đầu, tôi đã nghiên cứu các phương pháp của CNN và cách thức hoạt động của nó. Như sau:

Đối với mỗi hình ảnh đầu vào, CNN sẽ cố gắng trích xuất các đặc trưng của từng hình ảnh. Tính năng này được gọi là đặc trưng để sử dụng trong việc phân biệt với các ảnh khác. Các đặc trưng sẽ không được chỉ định cụ thể và CNN sẽ tự động thực hiện thông qua đào tạo. Sau khi lọc, các đặc trưng sẽ được chuyển qua các lớp mạng nơ-ron. Cuối cùng, CNN sẽ chỉ ra những gì đầu vào có khả năng là theo phân phối xác suất cho mỗi lớp.



Hình 5: Trích xuất đặc trưng của ảnh

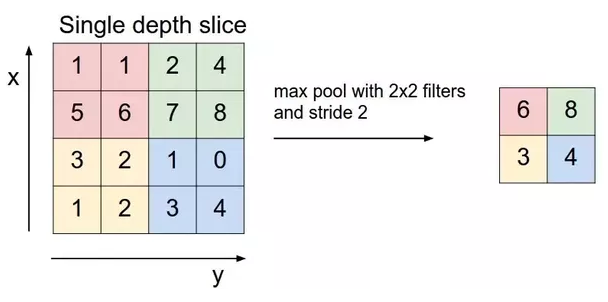
1. **Kiến trúc CNN**

CNN có hiệu quả cao trong bài toán phân loại ảnh vì thế nó được nâng cấp liên tục để đạt được độ chính xác cao. CNN có được kết quả như ngày hôm nay là do đã bắt chước thành công phương pháp xử lý thị giác của con người.

Tôi sẽ cung cấp một số khái niệm kỹ thuật mà tôi đã nghiên cứu về mô hình CNN. Trước hết, tôi sẽ trả lời câu hỏi thế nào là tích chập? Theo những gì tôi hiểu, về mặt toán học, tích chập là sự kết hợp của hai hàm (hoặc ma trận, ...) để tạo ra một hàm (hoặc ma trận, ...) thứ ba. Cũng có thể được gọi là sự hợp nhất của hai bộ thông tin. Phép tích chập khác với tương quan chéo ở chỗ nó cần lật kernel theo chiều ngang và dọc trước khi tính tổng của tích. Về mặt CNN, đó là sự kết hợp giữa đầu vào và kernel để trích xuất các đặc trưng (feature).

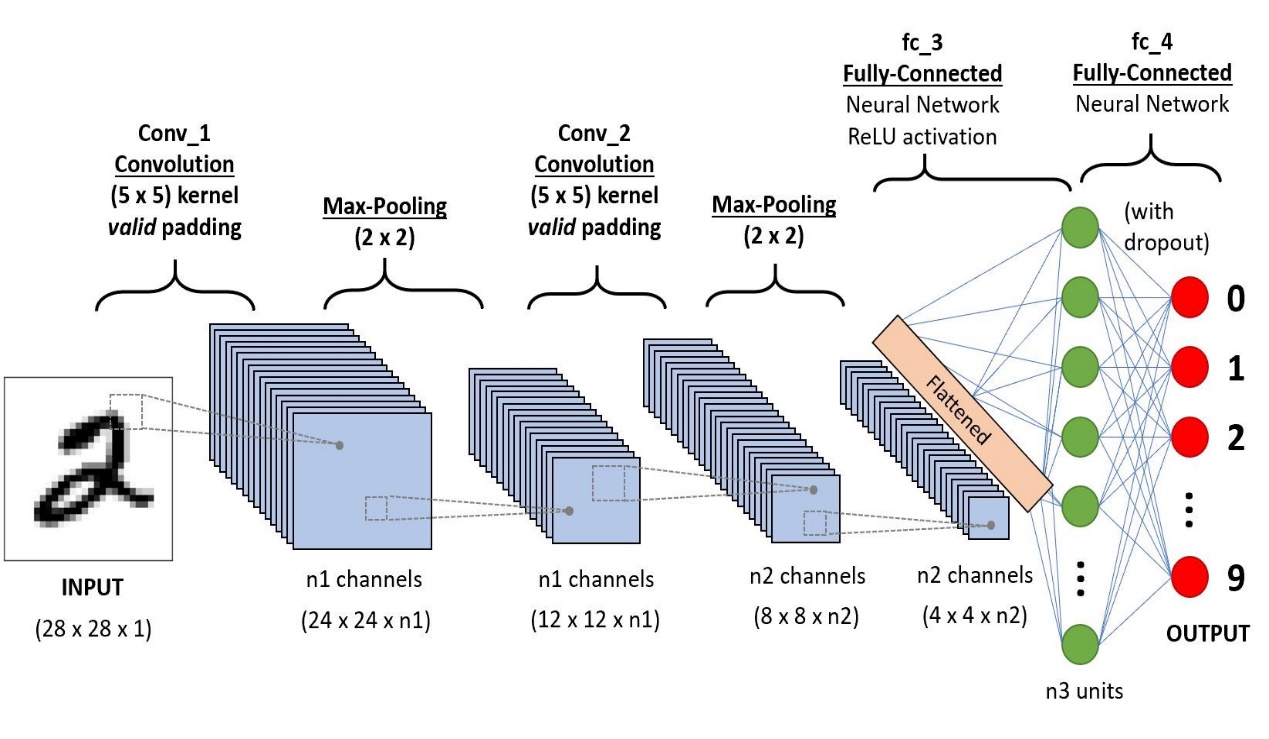
Thứ hai, nguyên tắc hoạt động của CNN là trượt kernel trên đầu vào để trích xuất các đặc trưng đầu ra của đầu vào. Từ đó, tạo ra một ma trận mới lưu tất cả những đặc trưng đó. Sau đó, công việc của pooling layer là chọn max pooling (điểm cao nhất trong mỗi feature) và tạo một ma trận mới để lưu trữ các giá trị max pooling đó. Hai quy trình này được lặp lại cho đến khi lớp được kết nối đầy đủ (fully connected layer) của mô hình được thỏa mãn. Vì vậy, các khái niệm được đề cập như feature, pooling và fully connected có ý nghĩa gì trong CNN?

* Feature: CNNs so sánh hình ảnh từng mảnh. Các mảnh mà nó tìm kiếm được gọi là các đặc trưng. Bằng cách tìm thấy các kết hợp đặc trưng thô ở các vị trí gần giống nhau trong hai hình ảnh, CNNs nhận thấy sự tương đồng tốt hơn nhiều so với các lược đồ khớp toàn bộ hình ảnh. Mỗi đặc trưng giống như một hình ảnh nhỏ một mảng giá trị hai chiều nhỏ. Các tính năng phù hợp với các khía cạnh phổ biến của hình ảnh. Lấy ví dụ về chó và mèo ở trên, thì khi một người nhìn vào một bức hình có thể xác định đó là chó hay mèo là vì họ đã biết được những đặc trưng của con vật đó.
* Pooling: Lớp pooling thường được sử dụng ngay sau các lớp chập để đơn giản hóa thông tin đầu ra để giảm số lượng nơ-ron. Quy trình pooling là max-pooling, quy trình này chọn giá trị lớn nhất trong vùng nhập (kernel). Thông qua lớp tổng hợp tối đa, số lượng nơ-ron giảm một nửa. Trên mạng CNN có nhiều Feature Maps, mỗi Feature Maps, tôi sẽ cung cấp cho mỗi max-pooling khác nhau. Có thể thấy rằng max-pooling là làm thế nào để hỏi những đặc trưng nào là đặc trưng nhất của các feature này.



Hình 6: Lấy max-pooling với độ trượt bằng 2

* Fully connected layer (Dense): Các mạng nơ-ron trong lớp này được kết nối đầy đủ với các lớp nơ-ron tiếp theo. Tên khác của Fully connected layer là lớp Dense. Đối với các mạng CNN, Keras cung cấp một loại lớp gọi là Dense cho phép xây dựng một lớp Dense với các tham số cơ bản như số lượng nơ-ron trong lớp đó, chức năng kích hoạt, khởi tạo tham số cho lần chạy đầu tiên và nhiều thông số khác



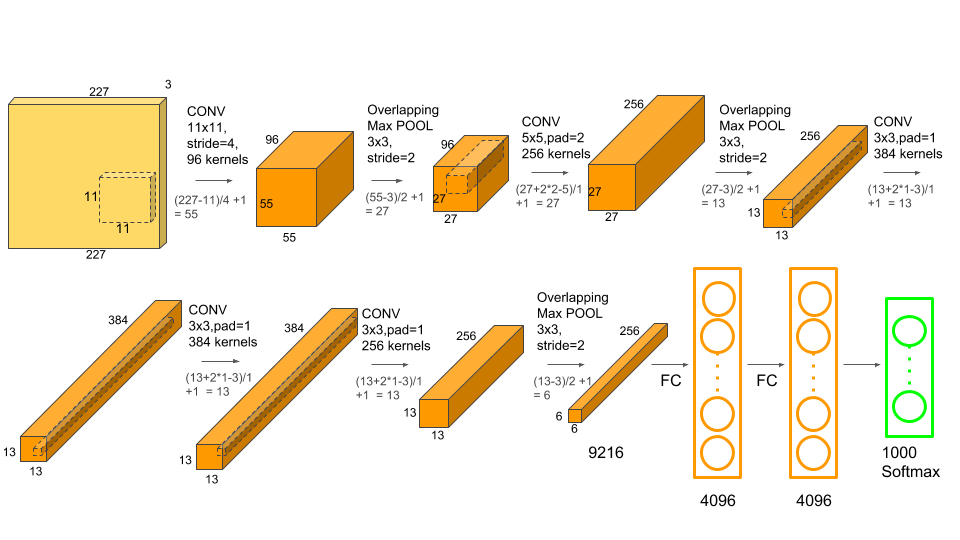
Hình 7: Lớp Dense trong CNN

Tóm lại thì CNN xây dựng dựa trên các thành phần đã đề cập bên trên một cách hợp lý và điều chỉnh các đặc trưng của mỗi lớp.

1. **Transfer Learning**

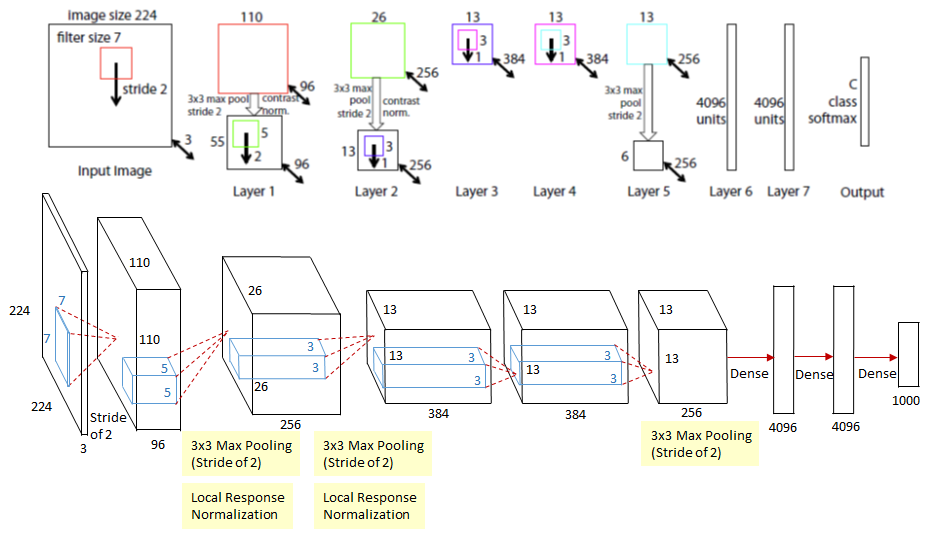
Những năm gần đây, Deep learning phát triển cực nhanh với lượng dữ liệu training khổng lồ kết hợp với khả năng tính toán của máy tính được cải tiến thì kết quả của các bài toán phân loại ảnh ngày được nâng cấp. Bộ dữ liệu ImageNet với hơn 1000 classes được cho là dùng để training với Deep learning nhiều nhất. Có thể kể đến một vài tên như

* AlexNet - Kiến trúc AlexNet lớn hơn nhiều so với các kiến trúc CNNs được sử dụng trong thị giác máy tính trước kia (trước năm 2010), vd kiến trúc LeNet của Yann LeCun năm 1998. Nó có 60 triệu tham số và 650000 nơ-ron và tốn khoảng từ năm đến sáu ngày huấn luyện trên hai GPU GTX 580 3GB.



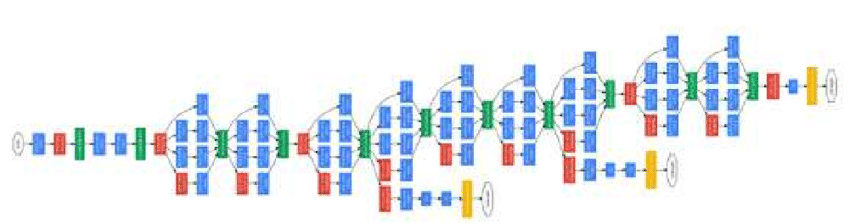
Hình 8: Kiến trúc của AlexNet

* ZFNet - giành chiến thắng trong các cuộc thi ILSVRC (ImageNet Large Scale Visual Recognition Challenge) năm 2013, top 5 với error rate khoảng 14.8%



Hình 9: Kiến trúc ZFNet

* GoogLeNet (Inception v1) - giành chiến thắng trong các cuộc thi ILSVRC (ImageNet Large Scale Visual Recognition Challenge) năm 2014, top 5 với error rate khoảng 6.67%.



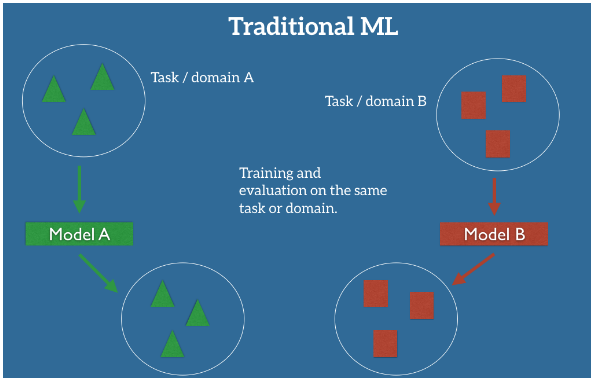
Hình 10: Kiến trúc của GoogleLeNet

* VGG – “Á hậu” tại cuộc thi ILSVRC 2014 được cộng đồng mệnh danh là VGGNet và được phát triển bởi Simonyan và Zisserman, bao gồm 138 triệu tham số, có thể hơi khó xử lý.



Hình 11: Kiến trúc của VGG16

Nhìn chung thì các mô hình này đều rất nhiều layers và cũng chính vì nhiều layers nên khiến cho các mô hình có thể đạt được kết quả cao nhưng cũng đồng nghĩa với việc training tốn kém nhiều thời gian, ví dụ bộ ImageNet cũng khoảng 1 tuần. Nếu với bài toán tương tự nhưng dựa trên tập dữ liệu khác thì việc train đi train lại không phải là một cách giải quyết hay và rất ít người làm điều đó vì cơ sở dữ liệu có kích thước lớn rất ít.



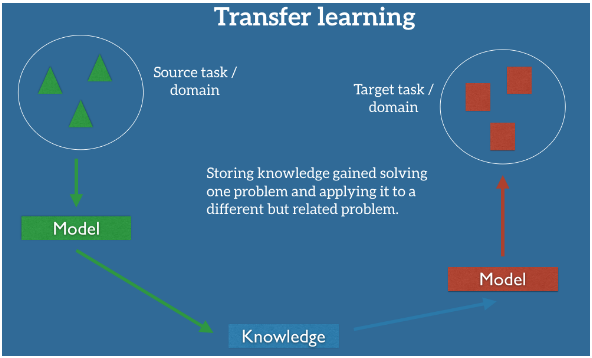
Hình 12: Huấn luyện Machine Learning truyền thống

Một phương pháp thay thế, sử dụng từ các mô hình (bên trên) mà đã được train trước thường được dùng và một số kỹ thuật khác để giải quyết sẽ hiệu quả hơn. Phương pháp thay thế này được gọi là “Transfer Learning”. Nếu tôi muốn train một model để phát hiện người đi bộ trên hình ảnh vào ban đêm, tôi có thể áp dụng một mô hình đã được train trên một domain tương tự, ví dụ: vào hình ảnh ban ngày. Tuy nhiên, trên thực tế, chúng ta thường trải qua sự suy giảm hoặc sụp đổ về hiệu suất do model đã thừa hưởng sự sai lệch của dữ liệu train và không biết cách khái quát hóa cho domain mới. Nếu muốn train một model để thực hiện một nhiệm vụ mới, chẳng hạn như phát hiện người đi xe đạp, thậm chí không thể sử dụng lại một mô hình hiện có, vì các nhãn giữa các nhiệm vụ khác nhau.



Hình 13: Hai miền dữ liệu khác nhau

Transfer learning cho phép xử lý các tình huống này bằng cách tận dụng dữ liệu được gắn nhãn hiện có của một số tác vụ hoặc domain liên quan. Người tạo cố gắng lưu trữ kiến thức thu được trong việc giải quyết nhiệm vụ nguồn trong miền nguồn và áp dụng nó cho vấn đề người khác quan tâm như có thể thấy.

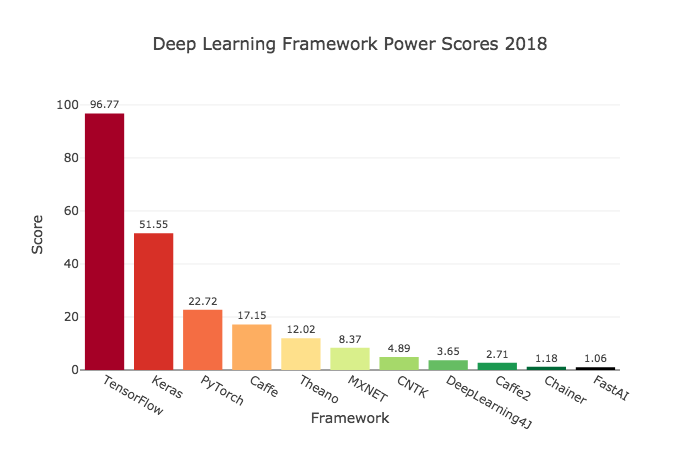


Hình 14: Transfer Learning

1. **Keras**

Keras là một API mạng nơ-ron cấp cao, thiết kế gần gũi với ngôn ngữ con người, được viết bằng Python và có khả năng chạy trên môi trường của TensorFlow, CNTK hoặc Theano. Về phía back-end thì hiện nay Keras đang khuyến khích người dùng cài đặt trên Tensorflow. Nó được phát triển với trọng tâm là cho phép thử nghiệm nhanh. Với việc cung cấp các cú pháp đơn giản, dễ hiểu, có thể đi từ ý tưởng đến kết quả với độ trễ ít nhất, có thể là “chìa khóa” để thực hiện nghiên cứu tốt.

* Cho phép tạo prototyping dễ dàng và nhanh chóng (thông qua sự thân thiện với người dùng, tính mô đun và khả năng mở rộng).
* Hỗ trợ cả mạng tích chập (CNNs) và mạng recurrent, cũng như kết hợp cả hai.
* Chạy tốt trên CPU và GPU.



Hình 15: Các framework nổi bật trong năm 2018

Kết hợp những đặc điểm của Keras và đặc biệt là sự kiện sát nhập giữa Tensorflow với Keras (năm 2019) thì tôi tự tin chọn Keras để sử dụng trong đồ án này.

1. **Quá trình thực hiện**

Trước khi trình bày về mô hình sẽ sử dụng tôi sẽ cung cấp một số khái niệm sẽ sử dụng trong quá trình train.

* Huấn luyện (training)

Huấn luyện mô hình là một quá trình phân tích dữ liệu đầu vào (input) nhằm tìm ra quy luật nào đó thông qua việc trích xuất các đặc trưng của đầu vào. Việc training sẽ đưa dữ liệu đầu vào đi qua từng bước xử lý để trả về kết quả mong muốn. Các bước xử lý này được gọi là một lớp mô hình (model layer). Mỗi layer sẽ có những cách xử lý dữ liệu khác nhau và nối tiếp nhau tạo thành một chuỗi (hoặc một bộ), các output của layer này sẽ là input của layer tiếp theo, cho nên một mô hình luôn có nhiều layer để tăng thêm độ chính xác.

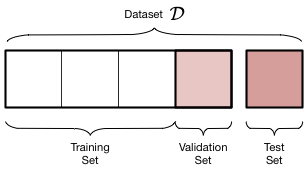
Khái niệm về kiến trúc của một mô hình bắt nguồn từ việc sắp xếp các layer cũng như điều chỉnh các hyperparameter (siêu tham số như epoch – số lần lặp, learning-rate – tốc độ học, …) cho phù hợp. Các mô hình khi mới “sinh ra”, tuy bao gồm rất nhiều layer nhưng sẽ chưa được đánh giá là tốt hay không, mà phải trải qua quá trình thử và sai nhiều lần để điều chỉnh lại các hyperparameter cho phù hợp. Nghĩa là mỗi người có thể tạo ra cho mình một mô hình tùy ý định dạng, không cần có một công thức nào, miễn mô hình đó đáp ứng tốt được vấn đề cần giải quyết.

* Tập train (training set)

Tập huấn luyện là tài liệu mà qua đó máy tính học cách xử lý thông tin. “Máy học” sử dụng các thuật toán - nó bắt chước các khả năng của bộ não con người để tiếp nhận các đầu vào đa dạng và xem xét chúng… Tập huấn luyện thường được viết thủ công và mô hình sẽ tuân theo chính xác các quy tắc và định nghĩa được đưa ra trong tập huấn luyện. Thông thường thì tập huấn luyện được tách ngẫu nhiên theo tỷ lệ nhất định (70%) trên tập dữ liệu gốc. Để cho đạt được kết quả tốt thì tập huấn luyện nên được chiếm tỷ lệ 60% trở lên trên tập dữ liệu gốc (dataset).

* Tập validation (validation set)

Tập validation cũng gần giống như tập training set. Tập validation là một tập dữ liệu của ví dụ sử dụng để điều chỉnh các hyperparameter của một phân loại (classifier). Đôi khi nó còn được gọi là tập phát triển hoặc "tập hợp dev". Tập này thường chiếm từ 40% trên dữ liệu gốc. Nói nom na thì tập train là những bài giảng hàng ngày của giáo viên còn tập validation là các bài kiểm tra 15 phút của giáo viên trong lớp để kiểm tra độ hiểu bài của các sinh viên.



Hình 16: Validation set

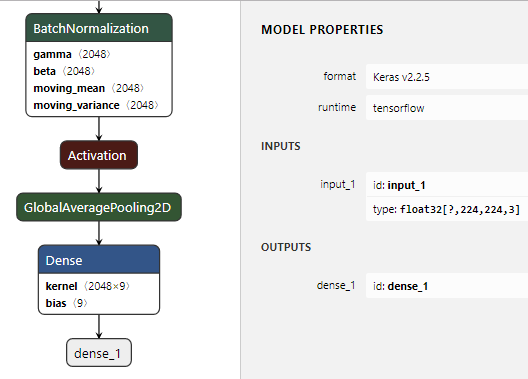
Lưu ý là tập validation cần được độc lập với tập train để tránh ảnh hưởng đến kết quả đầu ra, nghĩa là không lấy dữ liệu trong tập train cho việc kiểm tra và ngược lại vì các ảnh cần sự dự đoán để đưa ra giá trị đúng (ở đây là cần đúng tên nhãn tương ứng với tấm hình đầu vào).

* Suy luận từ model (Inference)

Suy luận (hay Inference) là một quá trình bắt đầu từ việc nhận 1 tấm hình đầu vào (input), sau khi qua các lớp xử lý của mô hình và cho ra được kết quả đầu ra (output). Đó biểu thị cho việc xuất một kết quả từ mô hình đã dự đoán ra được từ input.

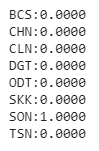
Sau khi tìm hiểu về cách xây dựng kiến trúc CNN cũng như Keras, tôi kế thừa việc tạo ra kiến trúc bằng Transfer learning, thông qua nhiều lần chỉnh sửa các hyperparameter cho phù hợp và đã cho ra nhiều kết quả khác nhau.

Đầu vào (input) là một bức ảnh sẽ được tôi resize lại cho phù hợp để đưa vào model. Cụ thể là một bức ảnh RGB với kích thước dài rộng bằng nhau và bằng 224 được biểu diễn dưới dạng 1 tensor (1x224x224x3). Sau khi qua tất cả các layers thì đầu ra (output) cuối cùng là 1 vector với độ dài là 9, nói cách khác là mảng gồm 9 số thực, có thể thấy được ở loại mà lớp dense biểu diễn là float32.



Hình 17: Biểu diễn dạng số thực của vector output

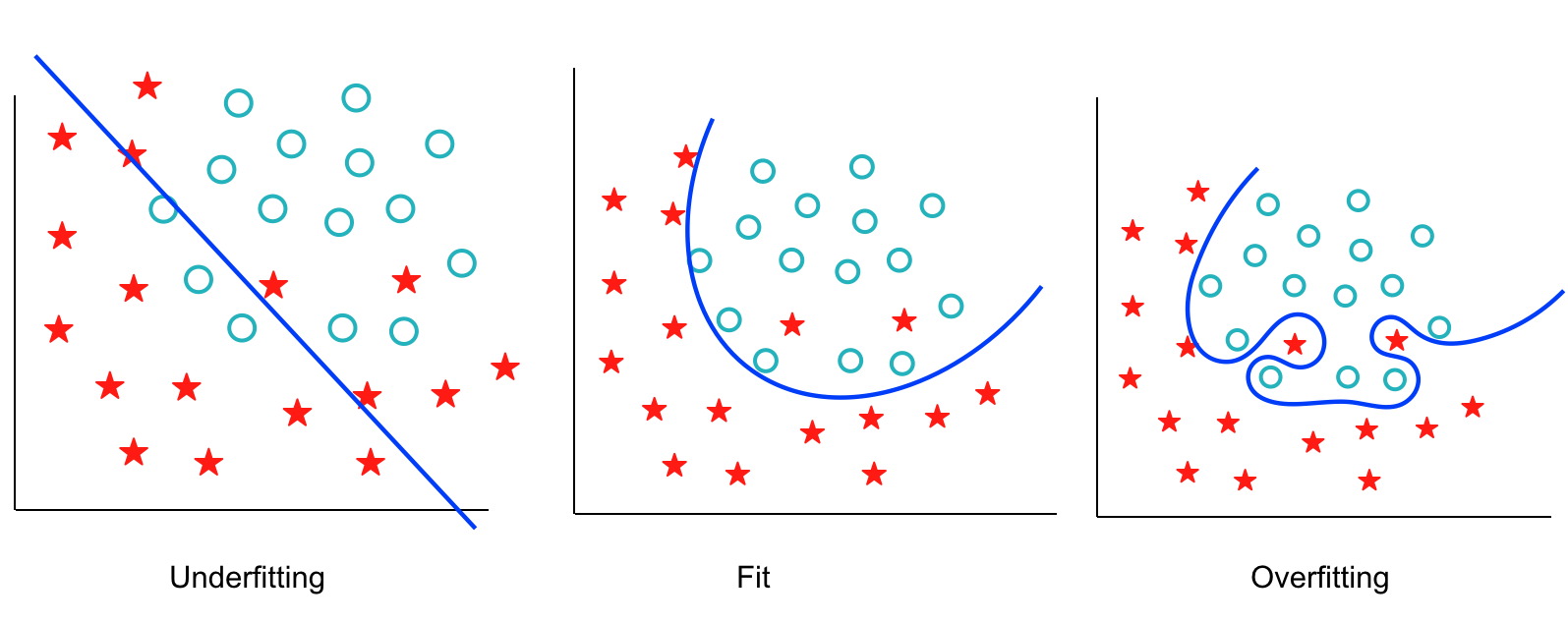
Ý nghĩa mà đầu ra (output) diễn tả là những phân loại đất mà tôi muốn ở đầu vào (input) gồm 9 phân loại đất (BCS, CHN, CLN, …). Các kết quả ở output sẽ có giá trị số thực nằm trong đoạn [0,1] và tổng của 9 loại đất là 1.



Hình 18: Kết quả xuất ra của mô hình

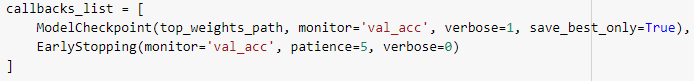
Trong tất cả model được train sẵn thì tôi nhận thấy Xception là một mô hình tốt để sử dụng với độ chính xác từ 79% - 94%. Với việc sử dụng model này, trong quá trình đầu tôi thử nghiệm huấn luyện với số lần lặp (epoch) là 10, tôi nhận thấy được kết quả chính xác (accuracy) trong tập train và chính xác (validation\_accuracy) trong tập kiểm tra có độ chênh lệch khá cao, gần đến 25%. Việc này cũng tương tự với kết quả mất mát (loss và validation\_loss). Loss là giá trị của hàm mất mát, hàm này mô tả sự mất mát dữ liệu trong quá trình huấn luyện. Nói nom na là giống như hình thức thưởng phạt đối với model, model sẽ đóng phạt nếu mỗi lần nó dự đoán sai, và số mức phạt tỉ lệ thuận với độ trầm trọng của sai sót. Thì một model tốt tức là model dự đoán đúng với nhãn đã gán, độ chính xác sẽ tiến về 1 và độ sai sót sẽ tiến về 0.

Việc xảy ra hiện tượng chênh lệch như vậy được gọi là hiện tượng “Overfitting”, giống như hiện tượng một học sinh trong lớp kiểm tra thì điểm rất cao nhưng khi đi thi thì điểm lại thấp. Đây là một điều mà các model khi training cần lưu ý.



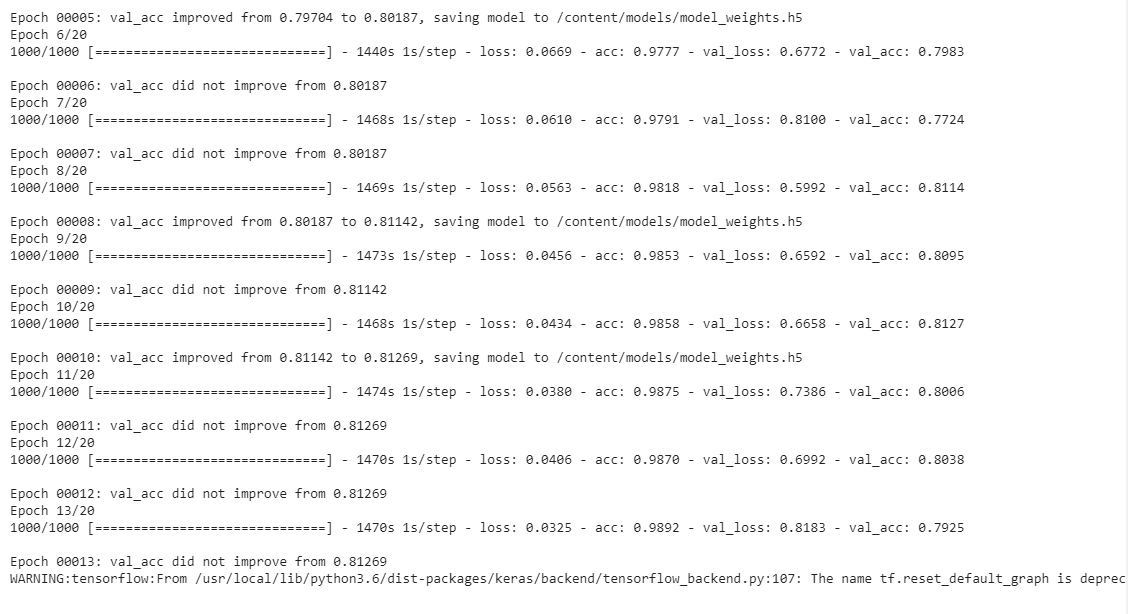
Hình 19: Overfitting trong mô hình huấn luyện

Có nhiều kỹ thuật để tránh trường hợp xảy ra Overfitting như: tăng độ phức tạo model, tăng lượng dữ liệu, kết thúc việc training đúng lúc (Early Stopping), … Chỉnh lại số epoch là 20 kết hợp áp dụng Early Stopping vào model, với việc nó sẽ “giám sát” cho qua các lần sai, nếu như quá số lần tôi thiết lập thì việc training sẽ dừng lại để tránh bị Overfitting lần nữa.



Hình 20: Sử dụng Early Stopping để giảm Overfitting

Sau đó, tôi tiến hành thử chạy lại và kết quả được như sau



Hình 21: Kết quả sau khi sử dụng Early Stopping

Kết quả khả quan hơn với độ chính xác là 81,27% và do có áp dụng kỹ thuật Early Stopping nên khi nhận thấy model không thể cải thiện thêm nữa và đang có đà đi xuống thì sẽ tự dừng lại và lưu lại model.

## Xây dựng ứng dụng

Sau quá trình tìm hiểu, xây dựng và training model thì tôi thực hiện xây dựng một máy ảo trên GCP để suy luận ra kết quả từ model. Các bước để thực hiện việc xây dựng như sau

* 1. **Nạp môi trường Tensorflow và Python**

Đầu tiên, tạo một VM instance theo các bước đã được hướng dẫn, sau đó nhấn SSH rồi update / upgrade môi trường theo các lệnh



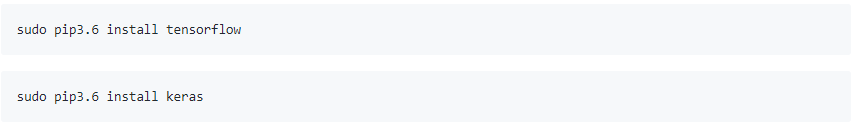
Hình 22: Các lệnh update / upgrade môi trường

Vì làm việc thông qua Python nên tôi cần cài đặt trình quản lý packages của Python, là pip. Do cài đặt python phiên bản mới (3.6) nên tôi cũng phải cài đặt pip3.6 cho tương tích.

Tiếp đến, cài đặt opencv với Python (OpenCV là một thư viện mã nguồn mở cho Computer Vision, hỗ trợ việc xử lý ảnh và máy học, và các tính năng tăng tốc GPU trong hoạt động thời gian thực), tensorflow và keras vào pip3.6.



Hình 23: Cài đặt OpenCV



Hình 24: Cài đặt Tensorflow và Keras

* 1. **Nhúng mô hình vào ứng dụng**

Chuẩn bị môi trường xong thì tôi chuẩn bị một đoạn script để upload model lên thì tôi cần nhúng nó vào để truy xuất được kết quả. Model ở đây là sau khi được train ở Google Colab xong thì tôi có lưu lại với tên “top\_model\_weight.h5” và dùng lệnh “models.load\_model” để load lên.



Hình 25: Script để tải mô hình

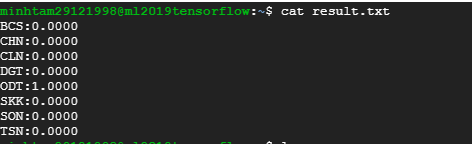
* 1. **Xử lý ảnh từ Internet và xuất kết quả**

Vì kết quả đầu vào đã trình bày là một ảnh RGB với kích thước dài và rộng là 224, nên khi ảnh được lấy để kiểm tra thì phải được “resize” lại đúng kích thước bằng cách sử dụng thư viện xử lý ảnh của python (opencv) để xử lý.



Hình 26: Xử lý kích thước ảnh trước khi dự đoán

Sau đó gán tấm ảnh vào model và dùng hàm predict để xuất kết quả. Kết quả sẽ được trả về file “result.txt” và dùng lệnh đọc file đó ra.



Hình 27: Kết quả sau khi dự đoán

# KẾT LUẬN

Trong suốt quá trình 15 tuần để thực hiện đồ án, tôi đã hoàn thành được các yêu cầu đề ra như

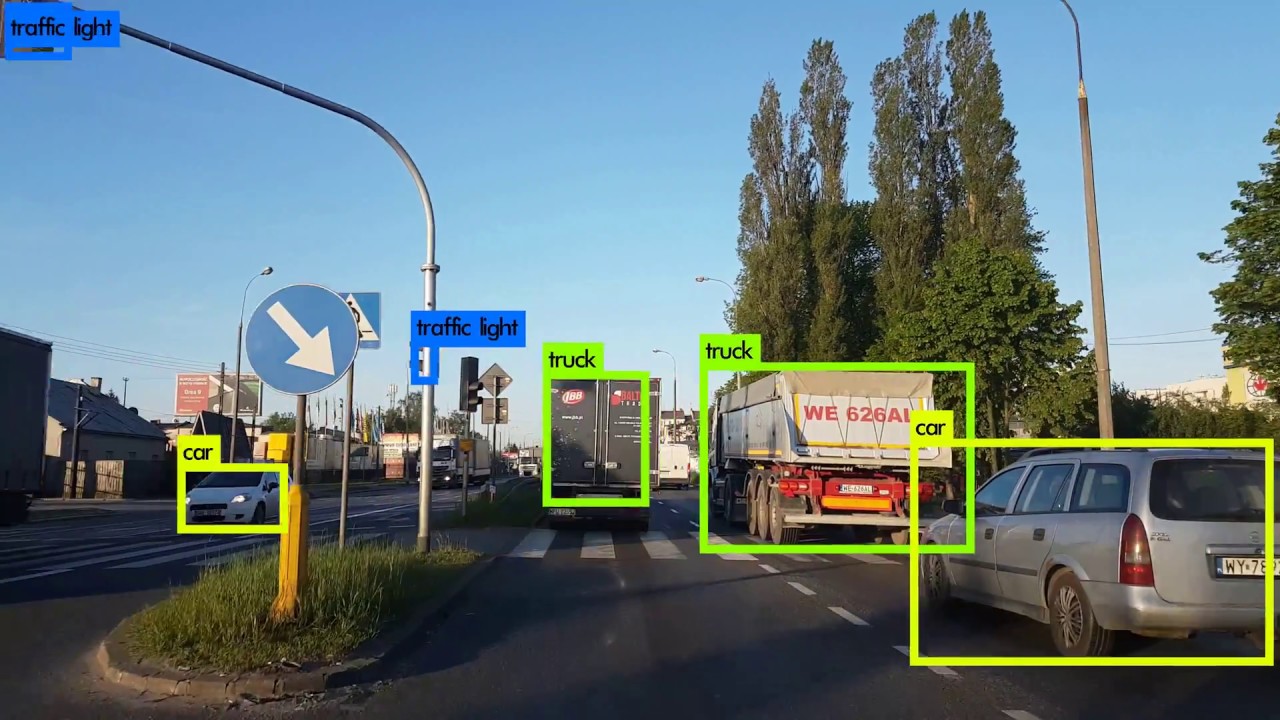
1. Thu thập dữ liệu ảnh từ Internet bằng các công cụ tiện ích hỗ trợ và ảnh thực tế bằng việc chụp ảnh ở các khu vực trong TPHCM.
2. Xử lý được dữ liệu đầu vào cho đúng định dạng tên cũng như nội dung của ảnh.
3. Tìm hiểu về mô hình CNN với Keras để xây dựng được mô hình huấn luyện.
4. Ứng dụng các kỹ thuật và môi trường trong việc huấn luyện mô hình như Transfer learning và giảm thiểu việc Overfitting trong quá trình huấn luyện bằng việc áp dụng kỹ thuật Early Stopping.
5. Xuất kết quả bằng việc nhúng mô hình vào máy ảo dựa trên nền tảng điện toán đám mây do Google Cloud cung cấp.

Qua quá trình làm việc này tôi đã nâng cao được các kiến thức về cách sử dụng các công nghệ mới, các kiến thức mới về Machine learning, Deep learning cũng như các kỹ năng về lập trình trong việc ứng dụng các script để giải quyết tiền xử lý ảnh. Bên cạnh những mục tiêu đạt được, vẫn còn đó một số điểm cần phải cải thiện vì chưa đạt được yêu cầu như

* Độ chính xác chưa cao, chỉ đạt ở mức chấp nhận được.
* Dữ liệu trong tập dataset cần được cải thiện bằng cách cập nhật thêm ảnh để mặt bằng chung các tập được cân bằng.
* Xử lý những ảnh chưa đúng định dạng (sRGB) về RGB.

Do đó để phát triển đồ án thì tôi cần đặt ra những hướng phát triển và phải cải thiện một số việc như

1. Thu thập thêm các ảnh nhằm gia tăng số lượng ảnh cho dataset bằng các ảnh thực tế để nâng cao tính thực tế của mô hình. Giải pháp cho việc thu thập ảnh là xây dựng một ứng dụng trên điện thoại và cơ sở dữ liệu cho phép người khác chụp và gửi ảnh về cơ sở dữ liệu. Từ đó download ảnh về và xử lý.
2. Nâng cao độ chính xác của mô hình bằng cách sử dụng thêm một số kỹ thuật mới và chỉnh lại các hyperarameter cho phù hợp. Các mô hình khác để có thể đồng giải quyết vấn đề là Object Detection. Tôi sẽ cắt ra những vật thể (object) như cây, cỏ, nhà, sông, suối, … sau đó xây dựng mô hình huấn luyện. Kết quả từ Object Detection trả về sẽ là các vật thể được dự đoán tên với tỷ lệ chính xác tương ứng.



Hình 28: Object detection

Từ đó khi suy luận từ mô hình, căn cứ số lượng xuất hiện của các vật thể kết hợp kết quả từ mô hình CNN, tôi sẽ đưa ra được kết luận loại đất nào là phù hợp.

1. Xây dựng giao diện ứng dụng cho người dùng. Giao diện sẽ là một trang web cho phép người dùng tải ảnh lên và dự đoán ảnh của mình. Sau khi mô hình dự đoán xong và trả kết quả, đồng thời kết quả sẽ kèm theo thông tin về loại đất đã được dự đoán.

# TÀI LIỆU THAM KHẢO

[1] Andrew Fogg, A History of Machine Learning and Deep Learning, <https://www.import.io/post/history-of-deep-learning>

[2] Siby Jose Plathottam, Comparing GPU and TPU training performance on Google Colaboratory, <https://medium.com/datadriveninvestor/comparing-gpu-and-tpu-training-performance-on-google-colaboratory-c1e54e26993f>

[3] Tensorflow, <https://www.tensorflow.org/>

[4]Anaconda, <https://docs.anaconda.com/>

[5] Ayyüce Kızrak, Step-by-Step Use of Google Colab’s Free TPU, <https://heartbeat.fritz.ai/step-by-step-use-of-google-colab-free-tpu-75f8629492b3>

[6] Google Cloud, <https://cloud.google.com/docs/overview/>, <https://cloud.google.com/free/docs/gcp-free-tier>

[7] Deep Learning, <https://www.mathworks.com/solutions/deep-learning/convolutional-neural-network.html>

[8] Purit Punyawiwat, Interns Explain CNN, <https://blog.datawow.io/interns-explain-cnn-8a669d053f8b>

[9] Sebastian Ruder, Transfer Learning - Machine Learning's Next Frontier, <https://ruder.io/transfer-learning/index.html#whatistransferlearning>

[10] Vũ Hữu Tiệp, <https://machinelearningcoban.com/>