**ĐẠI HỌC QUỐC GIA TP. HỒ CHÍ MINH**

**TRƯỜNG ĐẠI HỌC BÁCH KHOA**

**KHOA ĐIỆN – ĐIỆN TỬ**

**BỘ MÔN ĐIỀU KHIỂN – TỰ ĐỘNG**

**---------------o0o---------------**

****

**BÀI TẬP LỚN**

**CHUẨN ĐOÁN BỆNH UNG THƯ VÚ THÔNG QUA**

**ẢNH CHỤP X – QUANG SỬ DỤNG MÔ HÌNH CNN**

**GVHD: PGS. TS. Huỳnh Thái Hoàng**

**Sinh viên thực hiện:**

**1. Ngô Duy Bảo MSSV: 1710587**

**2.**

*TP. Hồ Chí Minh, tháng 6, năm 2021*

# Lời cảm ơn

Chúng tôi xin gửi lời cảm ơn chân thành nhất đến PGS. TS. Huỳnh Thái Hoàng, người đã giảng dạy chúng em trong suốt cả học kỳ để từ đó chúng em có được những kiến thức để thực hiện tốt bài tập lớn này. Trong suốt quá trình thực hiện, chúng em luôn cố gắng vận dụng thật tốt những kiến thức mà thầy đã truyền đạt cho chúng em. Chúng tôi sẽ không thể hoàn thành được bài tập này nếu không có những sự giúp đỡ của thầy.

Nhóm sinh viên

Ngô Duy Bảo

Someone

# Tóm tắt

Bài toán chuẩn đoán bệnh ung thư vú thông qua ảnh chụp X – quang đang được sự quan tâm của rất nhiều người nghiên cứu kỹ thuật trong lĩnh vực y sinh bởi sự ứng dụng rộng rãi của nó. Đối với những phương pháp xử lý ảnh thông thường thì chúng ta phải cố gắng tìm ra các đặc trưng của ảnh để phân loại chúng, việc làm này sẽ mất rất nhiều thời gian vì ta không thật sự là một chuyên gia trong ngành nên việc tìm được các đặc trưng là rất mất thời gian. Tuy nhiên, ngày nay với sự phát triển mạnh mẽ của mạng thần kinh mà đặc biệt là các bài toán phân loại hình ảnh thì đã có một công cụ rất mạnh đó chính là mô hình học sâu.Ngõ vào bài toán là một ảnh chụp X – quang bằng cách xây dựng một mô hình máy học chúng tôi cố gắng dự đoán thật tốt là người đó có bị ung thư hay không. Để giải quyết bài toán này chúng tôi quyết định tự xây dựng một mô mình mạng thần kinh tích chập (CNN: Convolution Neural Network) với bộ dữ liệu có sẵn đó là Inbreast, MIAS, DDSM những hình ảnh trong tập dữ liệu này đã được tăng cường. Kết quả với mô hình chúng tôi đạt được sai số 6%. Đây có thể là một kết quả chưa tốt nhưng đây chỉ là phiên bản đầu của chúng tôi. Chúng tôi sẽ cố gắng cải thiện mô hình trong những phiên bản kế tiếp.

# Chương 1: Giới thiệu

Trong chương đầu tiên của tiểu luận này, chúng tôi sẽ giới thiệu tổng quang về bài toán chuẩn đoán ung thư vú, nêu ra những vấn đề và mục tiêu chúng tôi cần giải quyết, cùng với những kết quả, đóng góp mà chúng tôi thực hiện được.

## Tổng quan

Điều khiển thông minh là một lĩnh vực lớn mà trong đó nổi bật là lĩnh vực thị giác máy tính. Thị giác máy tính là một lĩnh vực quan trọng trong đời sống hiện nay, bao gồm các phương pháp thu, nhận và xử lý ảnh số nhằm phân tích các thông tin có được trong ảnh, với rất nhiều ứng dụng trong quân sự, an ninh, kinh tế và y tế như nhận diện khuôn mặt, nhận diện chữ viết tay, phát hiện và bám đuổi vật thể, phân đoạn ảnh, phân loại ảnh …

Trong đề tài lần này, nhóm chúng tôi sẽ thực hiện một ứng dụng của thị giác máy tính, đó chính là sử dụng mô hình CNN để phân loại ảnh có chứa khối u lành tính và ác tính dựa trên những liệu có sẵn trên internet.

## Nhiệm vụ đề tài

Trong đề tài này, có các nhiệm vụ cần được nghiên cứu và hoàn thành là:

Nội dung 1: Tìm hiểu nguyên lý, lý thuyết về mô hình CNN.

Nội dung 2: Huấn luyện mô hình tự thiết kế.

Nội dung 3: Thực hiện đánh giá kết quả phân loại dựa trên mô hình xây dựng được.

# Chương 2: Lý thuyết

Ngay từ đầu chúng tôi quyết định giải quyết bài toán theo hướng sử dụng mô hình mạng neural theo những kiến thức đã được học trong học phần môn học. Với dạng bài toán phân loại hình ảnh, khi sử dụng mô hình mạng thần kinh thông thường đòi hỏi chúng ta phải sử dụng một lượng lớn các thông số do ảnh đầu vào đối với bài toán này có độ phân giải 227x227 pixels thì điểm dữ liệu đưa vào của mạng là giá trị của từng pixel trong ảnh do đó lúc này mô hình mạng của chúng ta sẽ chứa rất nhiều thông số điều này khiến cho việc sử lý sẽ rất lâu vì khối lượng tính toán rất lớn. Hiểu được điều này các nhà nghiên cứu họ đã tìm ra được một mô hình mạng khác được gọi là mạng thần kinh tích chập viết tắt là CNN (Convolution Neural Network). Để làm rõ vấn đề trên chúng ta sẽ tìm hiểu ở phần sau.

## 2.1 Giới thiệu ưu điểm của CNN.

# Chương 3: Huấn luyện mô hình

Với những lý thuyết đã trình bày ở chương trước của bài báo cáo này, chúng tôi quyết định sẽ xây dựng mô hình dựa trên những lý thuyết trên. Chúng tôi sẽ nói chi tiết trong phần kế tiếp.

## 3.1 Giới thiệu về ngôn ngữ lập trình Python.

Python là một ngôn ngữ lập trình thông dịch (interpreted), hướng đối tượng (object – oriented), và là một ngôn ngữ bậc cao (high – level) ngữ nghĩa động. Python hỗ trợ các module và các gói khuyến khích chương trình module hóa và tái sử dụng mã.

Sau đây là các đặc điểm của Python:

* Ngữ pháp đơn giản, dễ đọc.
* Vừa hướng thủ tục, vừa hướng đối tượng.
* Hỗ trợ module và hỗ trợ gói.
* Xử lý lỗi bằng ngoại lệ.
* Kiểu dữ liệu động ở mức cao.
* Có các bộ thư viện chuẩn và các module ngoài, đáp ứng tất cả các nhu cầu lập trình.
* Có khả năng tương tác với các module khác viết trên C/C++.
* Có thể nhúng vào ứng dụng như một giao tiếp kịch bản.



Hình 3.1: Biểu tượng của Python.

Chính vì những đặc điểm ưu việt của ngôn ngữ lập trình Python nên chúng tôi quyết định sẽ xây dựng mô hình học máy của chúng tôi dựa trên ngôn ngữ này.

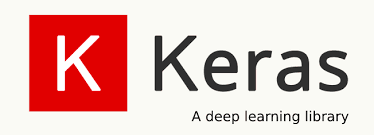
## 3.2 Giới thiệu về Keras.

Khi ta xây dựng một mô hình mạng thần kinh hay là mạng thần kinh tích chập thì chúng ta có thể xây dựng chúng một cách hoàn chỉnh bằng Python. Tuy nhiên, nó chỉ thật sự ưu việt khi ta tính toán thuận, còn về phần tính toán ngược để cập nhật trọng số mô hình thì việc sử dụng Python trở nên phức tạp hơn rất nhiều dẫn đến khó thực hiện và tối ưu được tốc độ tính toán. Đây chính là lý do các framework về deeplearning ra đời với các đặc điểm:

* Người dùng chỉ cần định nghĩa mô hình và hàm mất mát (hay còn gọi là hàm mục tiêu), framework sẽ lo phần lan truyền ngược.
* Việc định nghĩa các lớp, hàm kích hoạt sẽ đơn giản hơn cho người dùng. Chúng ta chỉ cần khai báo số các lớp và hàm kích hoạt là gì thì tự khác framework sẽ tính toán cho chúng ta.

Thì Keras cũng là một framework của Deep learning đây được xem là một thư viện dễ sử dụng, thân thiện với người dùng và đủ khả năng giải quyết các vấn đề mà chúng tôi đã đặt ra từ trước.

Keras là một framework mã nguồn mở cho Deep learning được viết bằng Python. Nó có thể chạy trên nền của các framework deep learning khác như: tensorflow, theana, CNTK. Với các API (Application Programming Interface: giao diện lập trình ứng dụng) bậc cao, dễ sử dụng,dễ mở rộng, keras giúp người dùng xây dựng các mô hình Deep learning một cách đơn giản.

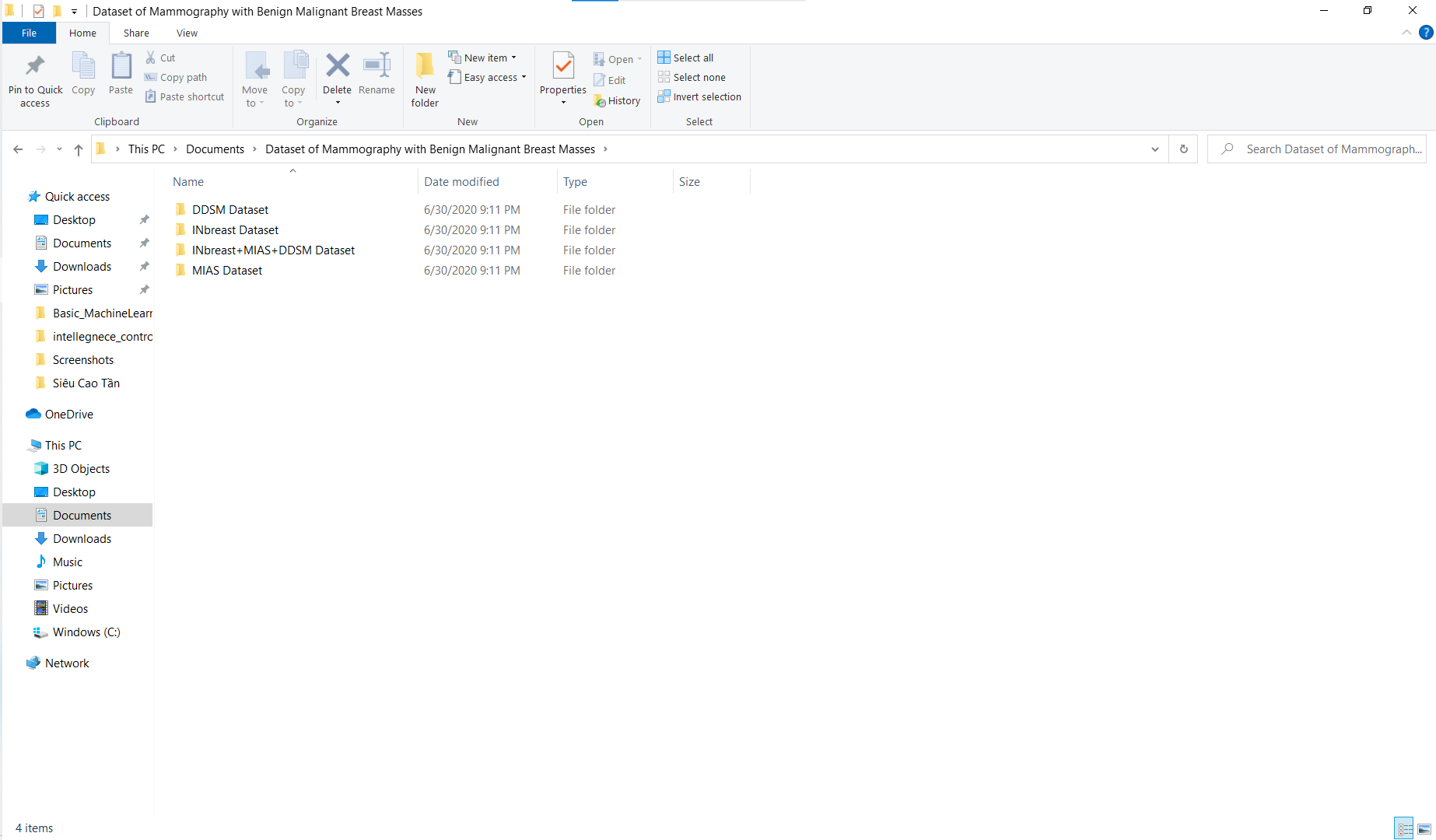


Hình 3.2: Biểu tượng thư viện Keras.

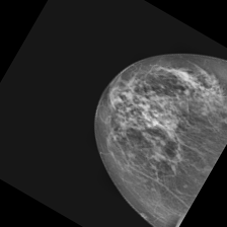
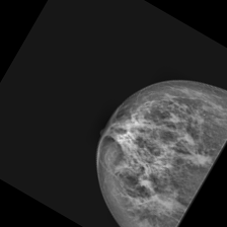
## 3.3 Tiền xử lý và tăng cường dữ liệu.

Chúng tôi tiến hành xây dựng mô hình dựa trên bộ dữ liệu có tên là Dataset of Breast mammography images with Masses đã được xây dựng bởi Ting Lin và Mei-Ling Huang.

Tập dữ liệu chứa các nhũ ảnh với các khối u lành tính và ác tính. Những ảnh trong tập dữ liệu đầu tiên được trích xuất từ 106 ảnh khối u từ tập dữ liệu Inbreast, 53 ảnh khối u từ tập dữ liệu MIAS và 2188 ảnh khối u từ tập dữ liệu DDSM. Sau đó chúng tôi sẽ tiến hành tăng cường dữ liệu này và tiến hành cân bằng histogram thích nghi giới hạn độ tương phản để tiền xử lý các hình ảnh này. Sau khi tăng cường dữ liệu, tập dữ liệu Inbreast có 7632 ảnh, tập dữ liệu MIAS có 3816 ảnh và tập dữ liệu DDSM có 13128 ảnh. Ngoài ra, chúng tôi sẽ tiến hành gộp những dữ liệu này với nhau để huấn luyện và đánh giá mô hình. Tất cả hình ảnh đều được đưa về độ phân giải 227x227 pixels.



Hình 3.3: Thư mục chứa dữ liệu.



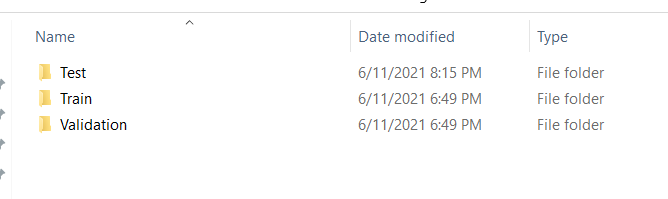
Hình 3.4: Hình ảnh khối u lành tính. Hình 3.5: Hình ảnh khối u ác tính.

Sau khi hoàn thành quá trình xử lý và tăng cường dữ liệu chúng tôi tiến hành huấn luyện và đánh giá mô hình bằng ngôn ngữ lập trình Python.

## 3.4 Huấn luyện và đánh giá mô hình.

Trước khi tiến hành huấn luyện chúng tôi muốn chia tập dữ liệu của mình ra làm ba phần, một phần dành cho tập train (tập huấn luyện), một phần dành cho tập validation (tập xác thực) và còn lại là tập thử nghiệm (tập test). Chúng tôi sẽ tiến hành chia với trọng số chia là 60% cho tập dữ liệu huấn luyện, 20% cho tập dữ liệu xác thực và 20% còn lại cho tập dữ liệu thử nghiệm.

Mô hình sẽ được huấn luyện dựa trên tập dữ liệu huấn luyện và dữ liệu xác thực, tức là mô hình sẽ được học những dữ liệu có trong tập huấn luyện sau đó sẽ được tính sai số mô hình dựa trên tập dữ liệu xác thực. Chú ý rằng trong quá trình huấn luyện mô hình phải đảm bảo là mô hình chúng ta chưa bao giờ được học những dữ liệu của tập thử nghiệm để đảm bảo độ tin cậy của mô hình cũng như tính chính xác trong học thuật.



Hình 3.6: Thư mục chứa dữ liệu.

Sau khi thực hiện xong bước huấn luyện mô hình chúng ta sẽ đánh giá mô hình vừa mới huấn luyện được dựa trên tập dữ liệu thử nghiệm mà ta đã chia trước đó.

### 3.4.1 Các bước chuẩn bị cần thiết cho quá trình huấn luyện:

Bước 1: Viết chương trình để tải dữ liệu để huấn luyện và đặt tên là Data\_Generate.py.

Đoạn chương trình:

*# Khai báo các thư viện*

*import tensorflow as tf*

*from tensorflow import keras*

*from tensorflow.keras import layers*

*import pandas as pd*

*from keras.preprocessing.image import ImageDataGenerator, load\_img, img\_to\_array*

*direct\_path="Data/Train" # đặt tên đường dẫn*

*# Viết hàm tải dữ liệu lên.*

*def LoadData(direct\_path, image\_size, batch\_size, seed):*

*# Train dataset*

*train\_ds = tf.keras.preprocessing.image\_dataset\_from\_directory(*

*direct\_path,*

*validation\_split=0.2,*

*subset="training",*

*seed=seed,*

*image\_size=image\_size,*

*batch\_size=batch\_size,*

*) # shuffle = True, batch\_size = 32*

*# Validate dataset*

*val\_ds = tf.keras.preprocessing.image\_dataset\_from\_directory(*

*direct\_path,*

*validation\_split=0.2,*

*subset="validation",*

*seed=seed,*

*image\_size=image\_size,*

*batch\_size=batch\_size,*

*) # shuffle = True, batch\_size = 32*

*return (train\_ds, val\_ds)*

Bước 2: Tiếp đến sẽ viết một chương trình có tên là LOADdata\_Aug.py dùng để tăng cường những dữ liệu vừa tải lên sử dụng thư viện Keras.

Đoạn chương trình: đã đính kèm theo bài nộp.

Việc tăng cường dữ liệu này sẽ giúp cho mô hình sẽ học được nhiều hơn và giúp tăng độ chính xác hơn.

Bước 3: Tiến hành viết chương trình huấn luyện mô hình có tên là Model.py.

Đoạn chương trình: đã đính kèm theo bài nộp.

Tóm tắt đoạn chương trình:

Sau khi tải dữ liệu và tăng cường dữ liệu chúng tôi sẽ xây dựng một mạng thần kinh tích chập với ngõ vào là ảnh 3 chiều có kích thước 32x32x3 (32x32 chính là độ phân giải của ảnh còn 3 chính là số kênh màu của ảnh cụ thể là RGB, trong quá trình tiền xử lý chúng tôi đã resize về từ 227x227 thành 32x32).

Sau đó chúng tôi xây dựng các lớp tích chập đầu tiền để trích xuất các đặc trưng trong ảnh bằng câu lệnh:

*model.add(layers.Conv2D(512,(3,3),activation='relu', input\_shape=input\_shape))*

Ở lớp đầu tiên này chúng tôi sẽ dùng 512 kernel có kích thước 3x3 để trích xuất các đặc trưng trong ảnh. Hàm kích hoạt chúng tôi sẽ chọn làm hàm ‘relu’ việc dụng hàm kích hoạt này với mục đích mong muốn là các trọng số của chúng tôi sẽ luôn được cập nhật trong quá trình train.

Tiếp đến trong lớp tích chập là lớp MaxPooling và BatchNormalization

*model.add(layers.MaxPooling2D((2, 2)))*

*model.add(layers.BatchNormalization())*

Mục đích của lớp MaxPooling chính là làm giảm kích thước của feature maps nhưng vẫn giữ được những thông tin qua trọng của đặc trưng.

Sau khi qua lớp MaxPooling thì các giá trị pixels trong feature maps vừa tính được sẽ bị phân tán trong khoảng từ [-1;1] điều này là không muốn vì khi điểm dữ liệu của chúng ta phân tán như thế tức là sự thay đổi không ổn định điều này dẫn đến đạo hàm của hàm mục tiêu thay đổi không ổn định làm cho việc tối ưu sẽ gặp nhiều vấn đề. Vì vậy chúng ta sẽ dùng một kỹ thuật gọi là BatchNormalization, kỹ thuật này sẽ làm cho điểm dữ liệu của feature map sẽ có giá trị trung bình là 0 và phương sai là 1 điều này sẽ giải quyết được vấn đề nêu trên.

Tương tự ở những lớp tích chập khác cũng như thế và mô hình của chúng tôi sẽ sử dụng ba lớp tích chập.

Cuối cùng chúng tôi được một ngõ ra chứa tất cả các thành phần đặc trưng của ảnh đầu vào với độ phân giải được tối giản rất nhiều. Chúng tôi sẽ thêm một lớp fully-connected với ba lớp với lớp đầu tiên gồm 512 nút mạng lớp thứ 2 gồn 256 nút mạng, lớp thứ 3 gồm 128 nút mạng. Cuối cùng chúng tôi sẽ dùng 2 nút mạng để phân loại lành tính hay là ác tính.

Đoạn chương trình:

*model.add(layers.Flatten())*

*model.add(layers.Dense(512, activation='relu'))*

*model.add(layers.BatchNormalization())*

*model.add(layers.Dense(256, activation='relu'))*

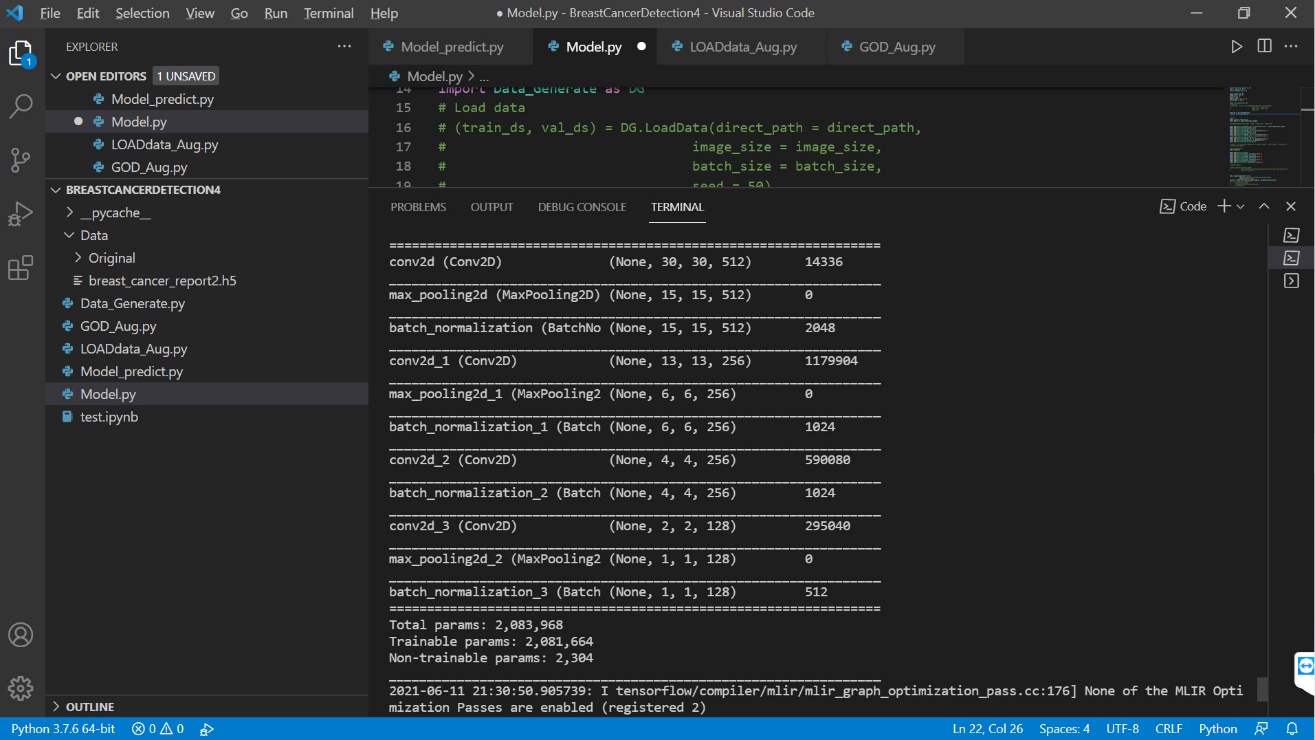
*model.add(layers.BatchNormalization())*

*model.add(layers.Dense(128, activation='relu'))*

*model.add(layers.BatchNormalization())*

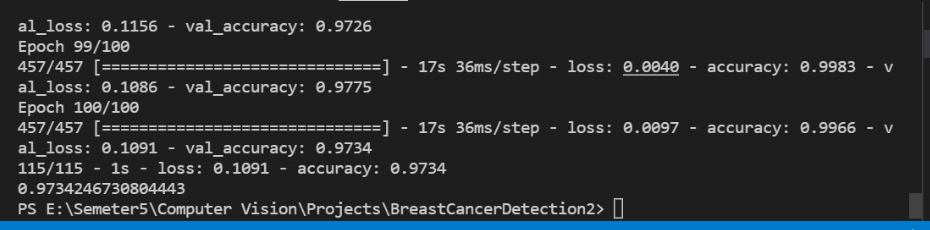
*model.add(layers.Dense(2, activation='sigmoid'))*

Như vậy là chúng tôi đã xây dựng xong một mô hình mạng phân loại với số lượng 2083968 thông số.



Hình 3.7: Tổng kết thông số mô hình.

Chúng tôi tiến hành huấn luyện với 100 epochs và được kết quả ở tập dữ liệu huấn luyện như sau:



### 3.4.2 Đánh giá mô hình vừa huấn luyện trên tập thử nghiệm.

Chúng tôi sẽ viết một đoạn chương trình có tên là Model\_predict.py để tính toán kết quả đánh giá ở tập thử nghiệm.

Và kết quả đạt được là: