

**ĐẠI HỌC QUỐC GIA THÀNH PHỐ HỒ CHÍ MINH**  
**TRƯỜNG ĐẠI HỌC CÔNG NGHỆ THÔNG TIN**

**DEEP LEARNING TRONG KHOA HỌC DỮ LIỆU**  
**LAB 4**

**ỨNG DỤNG MẠNG NEURAL TÍCH CHẬP**

**Họ và tên : Lưu Quang Tiến Hoàng**

**MSSV : 20521342**

**Lớp : DS201.N11**

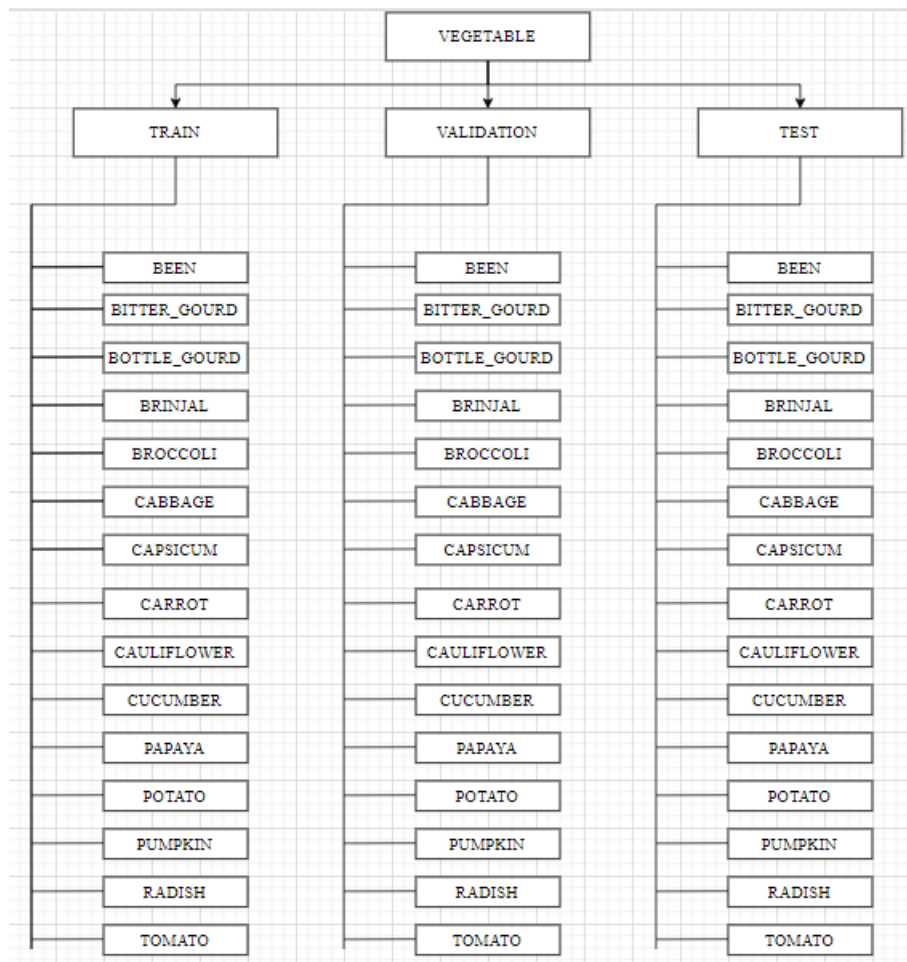
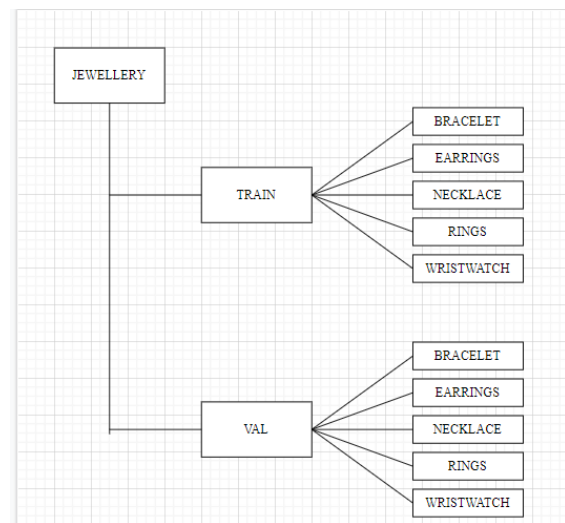
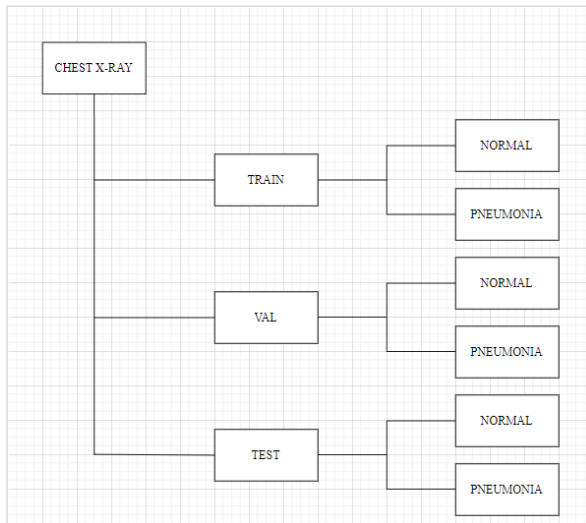
## Mục tiêu

- Ôn tập kiến thức cơ bản về mô hình mạng neural tích chập.
- Biết cách áp dụng các mô hình mạng neural tích chập nổi tiếng được thư viện cung cấp để giải quyết một số bài toán phân loại ảnh cơ bản.

## 1. CÁC BỘ DỮ LIỆU

- **Bộ dữ liệu 1.** Bộ dữ liệu ảnh chụp X-quang phổi (Chest X-Ray).
  - Số lượng dữ liệu: 5216 ảnh train, 16 ảnh dev và 624 ảnh test.
  - Định dạng: .jpg.
  - Số lượng nhãn: 2 (phân lớp nhị phân), gồm 2 nhãn: Normal (bình thường) và Pneumonia (nhiễm bệnh viêm phổi).
  - Link: <https://www.kaggle.com/paultimothymooney/chest-xray-pneumonia>.
- **Bộ dữ liệu 2.** Bộ dữ liệu ảnh chụp trang sức (Jewellery).
  - Số lượng dữ liệu: 1566 ảnh train, 250 ảnh test.
  - Định dạng: .jpg.
  - Số lượng nhãn: 5 (phân lớp đa lớp), gồm 5 nhãn: Bracelet (vòng đeo tay), Earrings (hoa tai), Necklace (dây chuyền), Rings (nhẫn), Wristwatch (đồng hồ đeo tay).
  - Link: <https://github.com/princesegzy01/Jewellery-Classification>.
- **Bộ dữ liệu 3.** Bộ dữ liệu ảnh chụp rau củ (Vegetable).
  - Số lượng dữ liệu: 15000 ảnh train, 3000 ảnh dev, 3000 ảnh test.
  - Định dạng: .jpg.
  - Số lượng nhãn: 15 (phân lớp đa lớp), gồm 15 nhãn: Bean (đậu), Bitter\_Gourd (mướp đắng), Broccoli (bông cải xanh), Carrot (cà rốt), Papaya (đu đủ), Cucumber (dưa leo), Tomato (cà chua),...

(?) Vẽ cấu trúc tổ chức của ba bộ dữ liệu trên ?



## 2. ĐỌC DỮ LIỆU TỪ ẢNH

**(?)** Đọc dữ liệu từ các tập train, dev, test của hai bộ dữ liệu Chest X-Ray và Vegetable với các thông số như sau ?

directory	Đường dẫn đến thư mục tương ứng trong Drive
labels	'inferred'
label_mode	'binary' (phân lớp nhị phân) hoặc 'categorical' (phân lớp đa lớp)
class_names	Tập các nhãn của bộ dữ liệu
color_mode	'rgb'
batch_size	256
image_size	(227, 227)
interpolation	'bilinear'

- **CHEST X-RAY:**

```
▶ Chest_XRay_train_set= image_dataset_from_directory(  
    directory=Chest_XRay_PATH + '/train',  
    labels= 'inferred',  
    label_mode= 'binary',  
    class_names=Chest_XRay_CLASSNAMES,  
    color_mode='rgb',  
    batch_size=256,  
    image_size=(227,227),  
    interpolation= 'bilinear'  
)
```

📁 Found 5216 files belonging to 2 classes.

```
[7] Chest_XRay_test_set= image_dataset_from_directory(  
    directory=Chest_XRay_PATH + '/test',  
    labels= 'inferred',  
    label_mode= 'binary',  
    class_names=Chest_XRay_CLASSNAMES,  
    color_mode='rgb',  
    batch_size=256,  
    image_size=(227,227),  
    interpolation= 'bilinear'  
)
```

Found 624 files belonging to 2 classes.

```
[8] Chest_XRay_val_set= image_dataset_from_directory(  
    directory=Chest_XRay_PATH + '/val',  
    labels= 'inferred',  
    label_mode= 'binary',  
    class_names=Chest_XRay_CLASSNAMES,  
    color_mode='rgb',  
    batch_size=256,  
    image_size=(227,227),  
    interpolation= 'bilinear'  
)
```

Found 16 files belonging to 2 classes.

- **VEGETABLE:**

```
vegetable_train_set= image_dataset_from_directory(  
    directory=vegetable + '/train',  
    labels= 'inferred',  
    label_mode= 'categorical',  
    class_names=vegetable_CLASSNAMES,  
    color_mode='rgb',  
    batch_size=256,  
    image_size=(227,227),  
    interpolation= 'bilinear'  
)
```

Found 15000 files belonging to 15 classes.

```
[48] vegetable_test_set= image_dataset_from_directory(  
    directory=vegetable + '/test',  
    labels= 'inferred',  
    label_mode= 'categorical',  
    class_names=vegetable_CLASSNAMES,  
    color_mode='rgb',  
    batch_size=256,  
    image_size=(227,227),  
    interpolation= 'bilinear'  
)
```

Found 3000 files belonging to 15 classes.

```
[13] vegetable_val_set= image_dataset_from_directory(
    directory=vegetable + '/validation',
    labels= 'inferred',
    label_mode= 'categorical',
    class_names=vegetable_CLASSNAMES,
    color_mode='rgb',
    batch_size=256,
    image_size=(227,227),
    interpolation= 'bilinear'
)
```

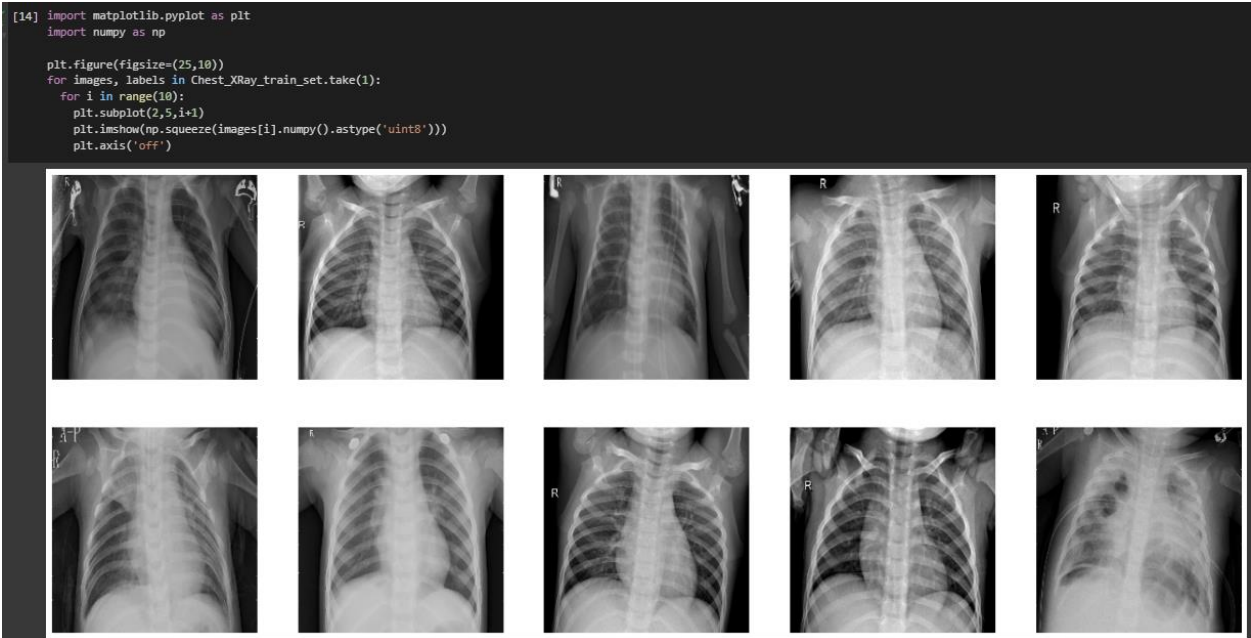
Found 3000 files belonging to 15 classes.

(?) Trực quan hóa một số ảnh (ngẫu nhiên) của hai bộ dữ liệu Chest X-Ray và Vegetable theo số lượng như sau ?

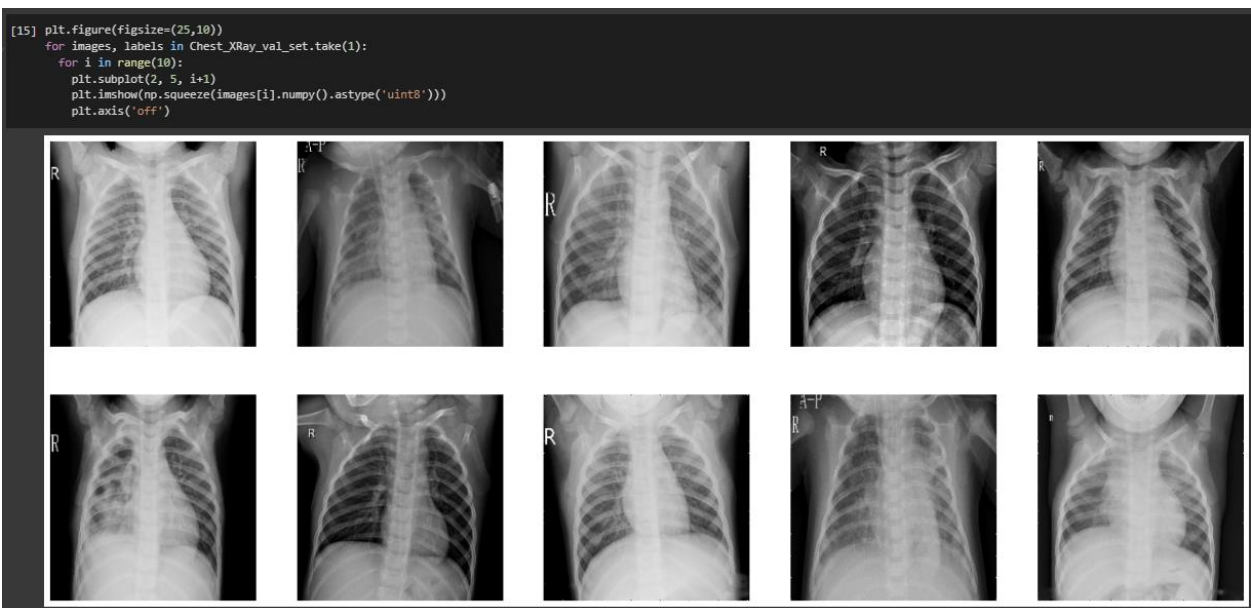
Bộ dữ liệu	Tập train	Tập dev	Tập test
Chest X-Ray	10 ảnh	10 ảnh	10 ảnh
Vegetable	12 ảnh	12 ảnh	12 ảnh

- **CHEST X-RAY:**

**Train:**



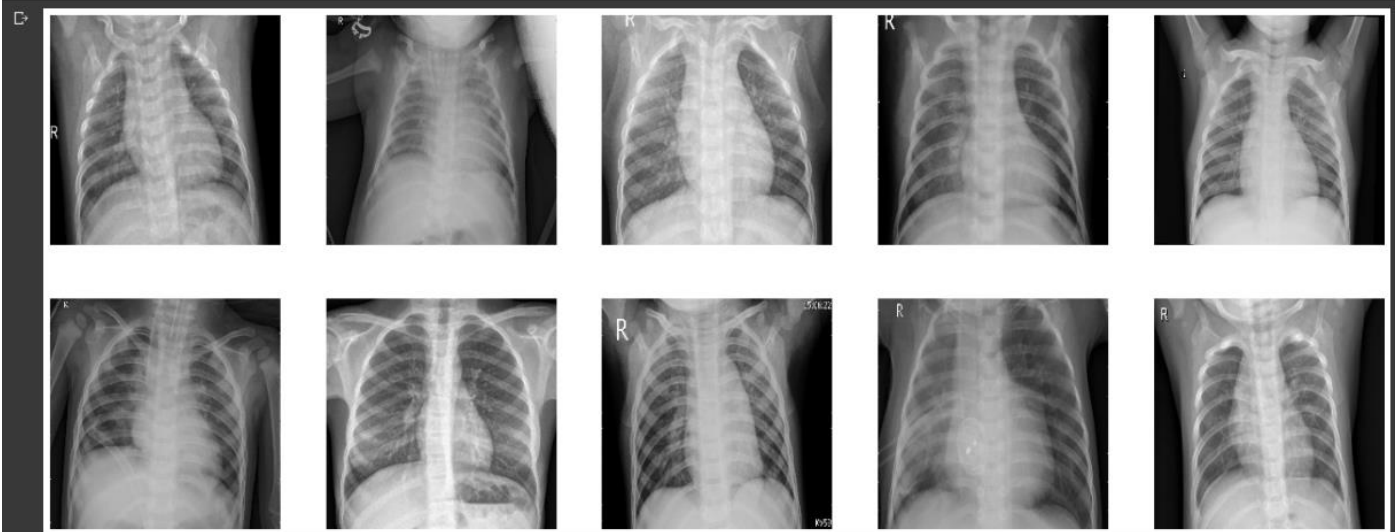
**Val:**





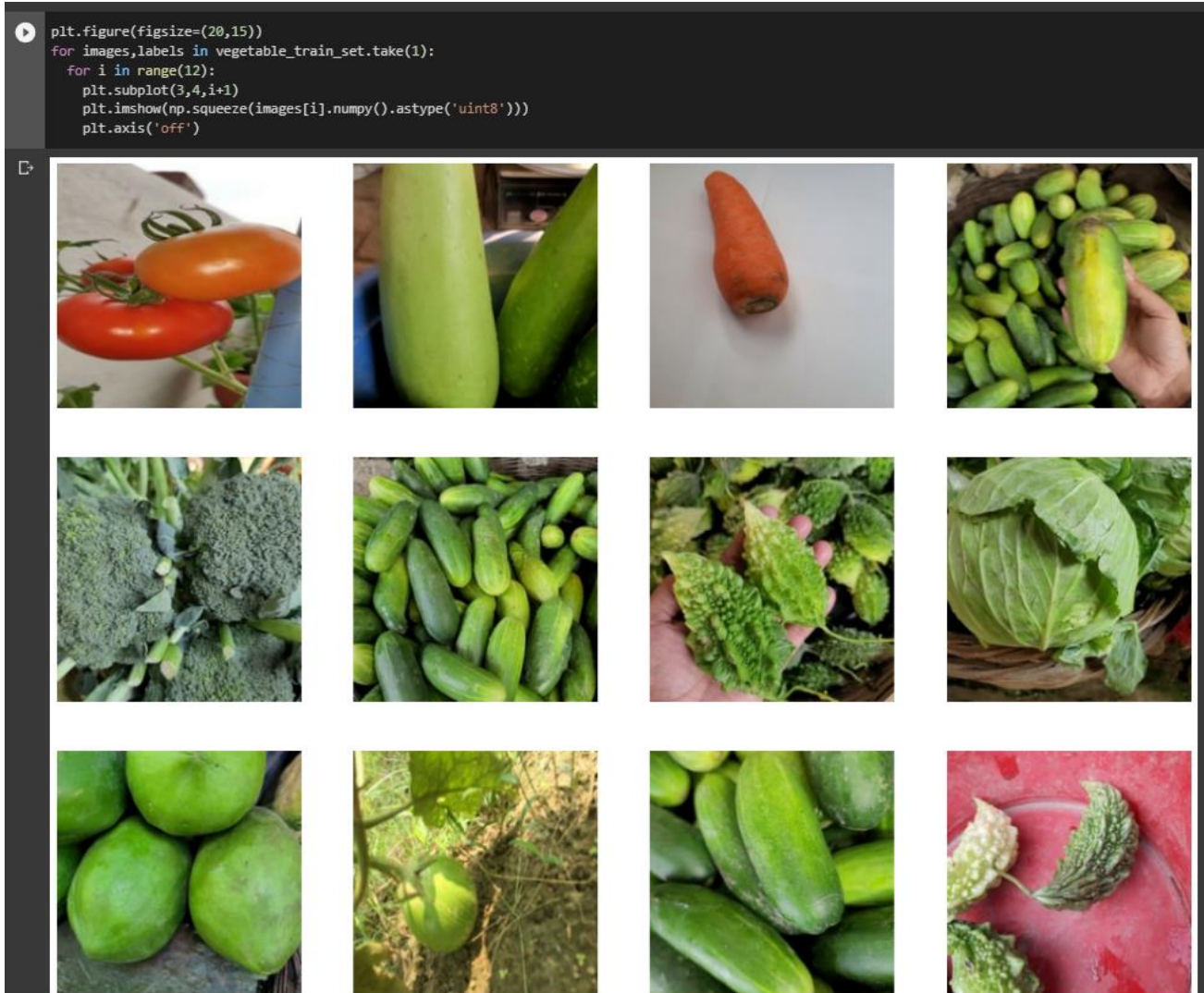
## Test:

```
plt.figure(figsize=(25,10))
for images, labels in Chest_XRay_test_set.take(1):
    for i in range(10):
        plt.subplot(2, 5, i+1)
        plt.imshow(np.squeeze(images[i].numpy().astype('uint8'))))
        plt.axis('off')
```



- **VEGETABLE:**

**Train:**



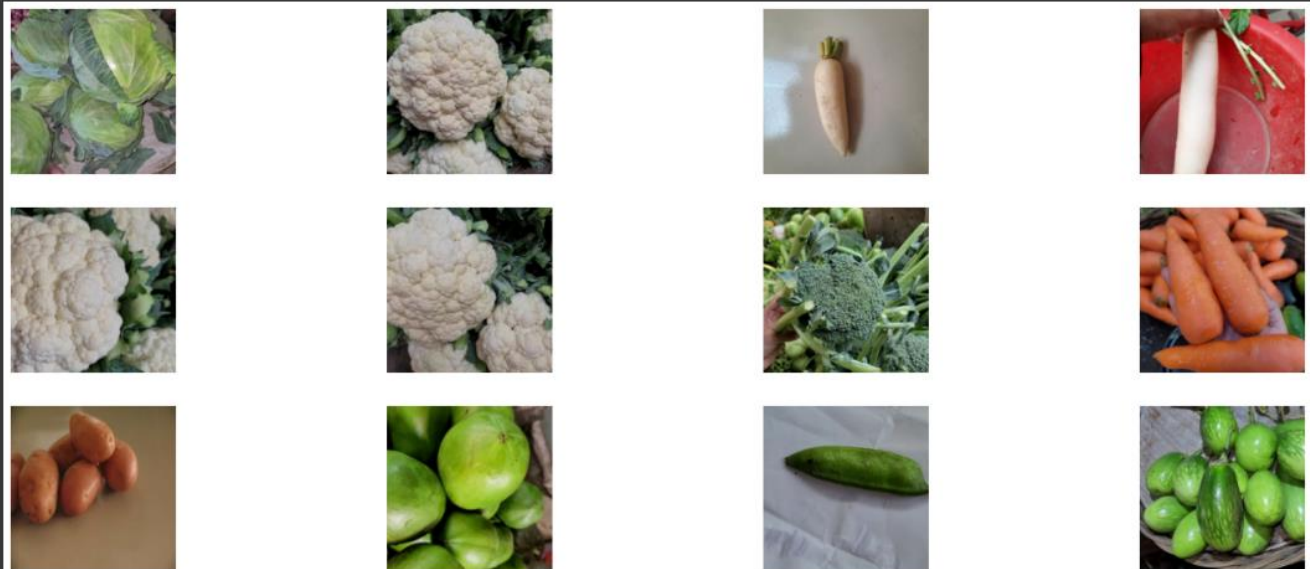
## Val:

```
plt.figure(figsize=(25,10))
for images, labels in vegetable_val_set.take(1):
    for i in range(12):
        plt.subplot(3, 4, i+1)
        plt.imshow(np.squeeze(images[i].numpy().astype('uint8'))))
        plt.axis('off')
```



## Test:

```
plt.figure(figsize=(25,10))
for images, labels in vegetable_test_set.take(1):
    for i in range(12):
        plt.subplot(3, 4, i+1)
        plt.imshow(np.squeeze(images[i].numpy().astype('uint8'))))
        plt.axis('off')
```

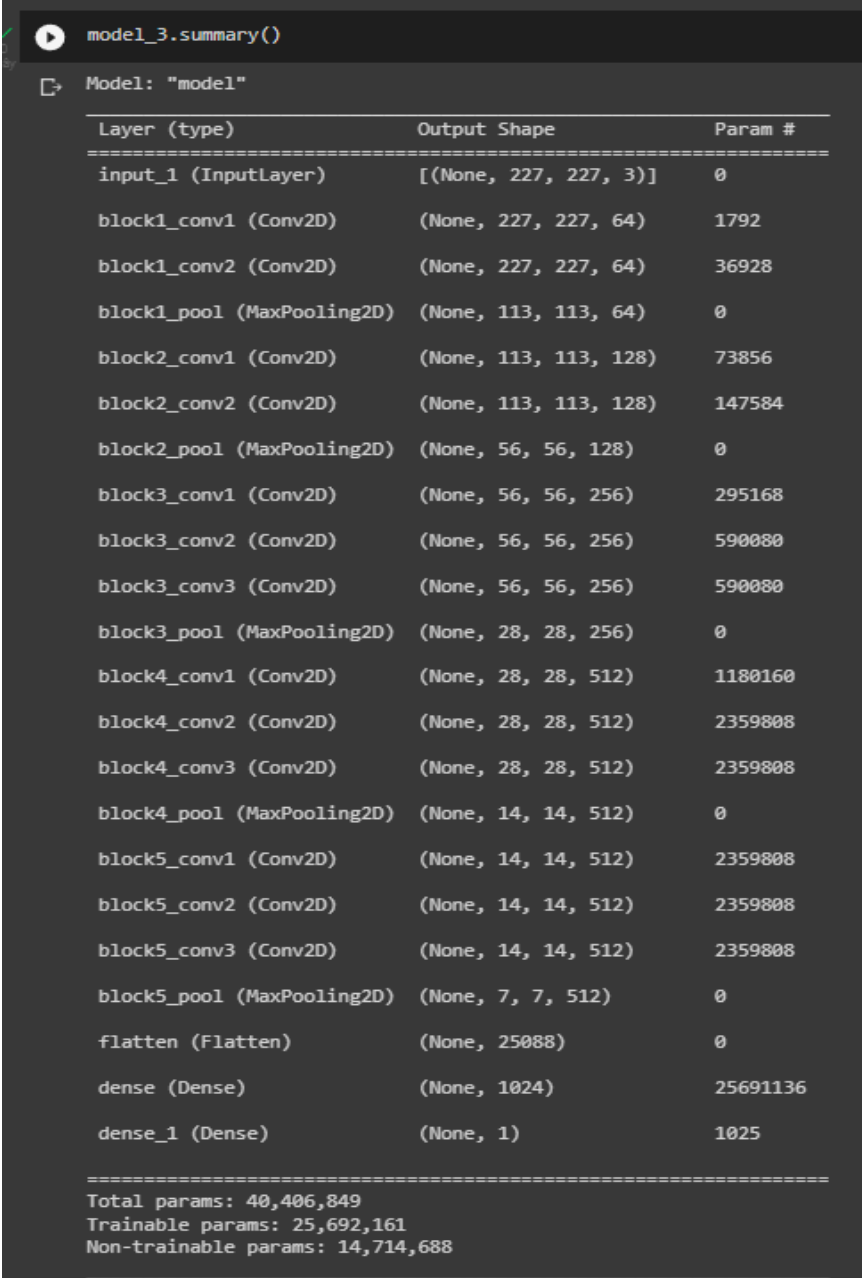


### 3. ỨNG DỤNG MÔ HÌNH

(?) Xem cấu trúc của mô hình trả lời các câu hỏi sau:

- Kết quả chạy lệnh summary ?
- Trong mô hình có những loại layer nào ?
- Nêu tổng số tham số và số lượng tham số cần huấn luyện ?

**Kết quả chạy lệnh summary:**



```
model_3.summary()
```

Model: "model"

Layer (type)	Output Shape	Param #
input_1 (InputLayer)	[None, 227, 227, 3]	0
block1_conv1 (Conv2D)	(None, 227, 227, 64)	1792
block1_conv2 (Conv2D)	(None, 227, 227, 64)	36928
block1_pool (MaxPooling2D)	(None, 113, 113, 64)	0
block2_conv1 (Conv2D)	(None, 113, 113, 128)	73856
block2_conv2 (Conv2D)	(None, 113, 113, 128)	147584
block2_pool (MaxPooling2D)	(None, 56, 56, 128)	0
block3_conv1 (Conv2D)	(None, 56, 56, 256)	295168
block3_conv2 (Conv2D)	(None, 56, 56, 256)	590880
block3_conv3 (Conv2D)	(None, 56, 56, 256)	590880
block3_pool (MaxPooling2D)	(None, 28, 28, 256)	0
block4_conv1 (Conv2D)	(None, 28, 28, 512)	1180160
block4_conv2 (Conv2D)	(None, 28, 28, 512)	2359808
block4_conv3 (Conv2D)	(None, 28, 28, 512)	2359808
block4_pool (MaxPooling2D)	(None, 14, 14, 512)	0
block5_conv1 (Conv2D)	(None, 14, 14, 512)	2359808
block5_conv2 (Conv2D)	(None, 14, 14, 512)	2359808
block5_conv3 (Conv2D)	(None, 14, 14, 512)	2359808
block5_pool (MaxPooling2D)	(None, 7, 7, 512)	0
flatten (Flatten)	(None, 25088)	0
dense (Dense)	(None, 1024)	25691136
dense_1 (Dense)	(None, 1)	1025

=====  
Total params: 40,406,849  
Trainable params: 25,692,161  
Non-trainable params: 14,714,688

Trong mô hình có các loại layer là: Input layer, Conv2D, MaxPooling2D, Flatten, Dense.

Tổng tham số: 40406849

Tham số huấn luyện: 25692161

**(?)** Với những mẫu dữ liệu bị dự đoán sai, in ra nhãn đúng và nhãn dự đoán để so sánh ?

```
[30] for i in range(len(y_true)):
      if y_pred_total[i] != y_true[i]:
          print(i, '(', y_true[i], ') - ', y_pred_total[i])
```

2 ( 0.0 ) - 1  
13 ( 0.0 ) - 1  
16 ( 0.0 ) - 1  
25 ( 0.0 ) - 1  
26 ( 0.0 ) - 1  
33 ( 0.0 ) - 1  
37 ( 0.0 ) - 1  
45 ( 0.0 ) - 1  
54 ( 0.0 ) - 1  
58 ( 0.0 ) - 1  
59 ( 0.0 ) - 1  
63 ( 0.0 ) - 1  
67 ( 0.0 ) - 1  
70 ( 0.0 ) - 1  
86 ( 0.0 ) - 1  
91 ( 0.0 ) - 1  
94 ( 0.0 ) - 1  
113 ( 0.0 ) - 1  
114 ( 0.0 ) - 1  
123 ( 0.0 ) - 1  
124 ( 0.0 ) - 1  
127 ( 0.0 ) - 1  
128 ( 0.0 ) - 1  
142 ( 0.0 ) - 1  
144 ( 0.0 ) - 1  
146 ( 0.0 ) - 1  
150 ( 0.0 ) - 1

**(?)** Tính độ chính xác của mô hình dự đoán bằng các độ đo đánh giá như Accuracy, Precision, Recall và F1-score ?

```
[31] from sklearn.metrics import accuracy_score, f1_score, precision_score, recall_score
```

```
[32] print('Accuracy: ',accuracy_score(y_pred_total, y_true)*100)
```

```
Accuracy: 80.12820512820514
```

```
[33] print('F1_score : ',f1_score(y_pred_total, y_true)*100)
```

```
F1_score : 86.28318584070797
```

```
[34] print('Precision :', precision_score(y_pred_total, y_true)*100)
```

```
Precision : 100.0
```

```
[35] print('Recall', recall_score(y_pred_total, y_true)*100)
```

```
Recall 75.87548638132296
```

**Trả lời:**

Accuracy: 80.128

F1\_score: 86.283

Precision: 100

Recall: 75.875