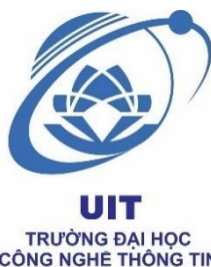


**ĐẠI HỌC QUỐC GIA THÀNH PHỐ HỒ CHÍ MINH
TRƯỜNG ĐẠI HỌC CÔNG NGHỆ THÔNG TIN**

----□&□----



BÁO CÁO ĐỒ ÁN MÁY HỌC

Đề tài:
**ỨNG DỤNG QUẢN LÝ HÌNH ẢNH
TRÊN THIẾT BỊ DI ĐỘNG**

Giảng Viên Hướng Dẫn:
Thầy Phạm Nguyễn Trường An
Thầy Lê Đình Duy

Sinh viên thực hiện:		
STT	Họ và tên	MSSV
1	Nguyễn Lộc Linh	19521754
2	Hoàng Anh Tú	19522449
3	Huỳnh Phạm Việt Pháp	19522571

MỤC LỤC

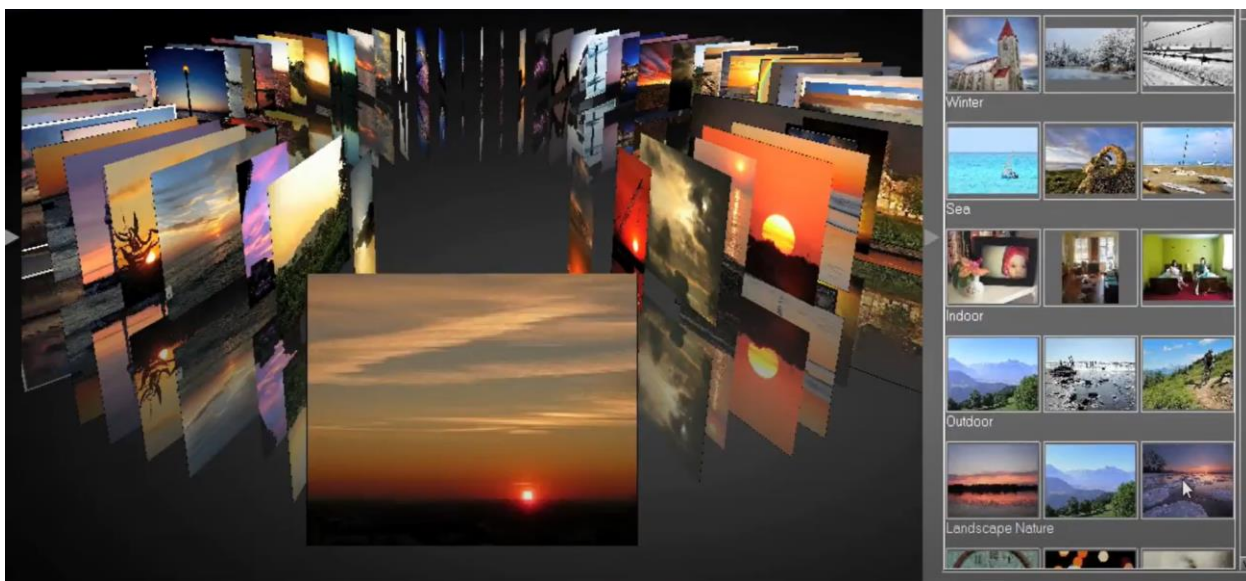
I. GIẢI TRÌNH CHỈNH SỬA SAU VẤN ĐÁP	3
II. GIỚI THIỆU	3
III. XÁC ĐỊNH BỐI CẢNH YÊU CẦU	4
1. Bài toán	4
2. Xác định yêu cầu bài toán	4
IV. THU THẬP VÀ PHÂN TÍCH DỮ LIỆU	5
1. Thu thập dữ liệu	5
2. Phân tích dữ liệu	14
V. ĐỀ XUẤT MÔ HÌNH	17
1. Mô hình sử dụng	17
2. Giới thiệu mô hình	17
VI. HUẤN LUYỆN MÔ HÌNH	20
VII. KẾT QUẢ THỰC NGHIỆM	24
VIII. TRIỂN KHAI MÔ HÌNH LÊN WEBSITE	30
IX. KẾT LUẬN	31
1. Ưu điểm & nhược điểm	31
2. Hướng phát triển	31
X. TÀI LIỆU THAM KHẢO	32

I. GIẢI TRÌNH CHỈNH SỬA SAU VẤN ĐÁP

- Chỉnh sửa và bổ sung cách đánh giá trong phần [KẾT QUẢ THỰC NGHIỆM](#). Bao gồm các thông tin Confusion Matrix của từng mô hình.
- Bổ sung một số hình ảnh áp dụng mô hình để dự đoán chủ đề hình ảnh trên trang web ở phần [TRIỂN KHAI MÔ HÌNH LÊN WEBSITE](#).

II. GIỚI THIỆU

Trong thời đại ngày nay điện thoại là một thiết bị không thể thiếu đối với mỗi người. Việc chụp ảnh hay selfie bằng điện thoại đã không phải là một việc quá xa lạ hiện nay. Nhu cầu chụp ảnh ngày càng phát triển, người ta càng chụp nhiều ảnh trong cuộc sống hơn. Do đó số lượng ảnh trong điện thoại ngày càng tăng cao, việc vài ngàn ảnh trong một chiếc điện thoại đã không phải là hiếm gặp. Nhưng khi ảnh quá nhiều thì sẽ phát sinh ra nhiều vấn đề. Mỗi khi muốn tìm lại một tấm ảnh nào đó thì sẽ khá mất thời gian vì phải duyệt tuần tự. Từ đó phát sinh ra nhu cầu cần một ứng dụng tự động phân loại các hình ảnh để giúp tìm kiếm ảnh nhanh hơn.



Trong thời đại học máy, Transfer Learning là một kỹ thuật cho phép người dùng sử dụng lại mô hình đã được đào tạo và sử dụng nó cho một nhiệm vụ khác. Phân loại hình ảnh là quá trình lấy một hình ảnh làm dữ liệu đầu vào và gán cho nó một lớp với một xác suất. Quá trình này sử dụng các mô hình học sâu là các mạng thần kinh

sâu, hoặc chi tiết hơn là Mạng thần kinh chuyển đổi (CNNs -Convolutional Neural Networks).

Trong bài báo cáo này nhóm chúng em sử dụng ba mô hình để thực hiện việc phân loại hình ảnh đó chính là VGG16, Xception và InceptionV3.

III. XÁC ĐỊNH BỐI CẢNH YÊU CẦU

1. Bài toán

Bài toán phân loại ảnh theo chủ đề.

Ngữ cảnh đưa ra là khi ảnh trong điện thoại quá nhiều, mỗi khi muốn tìm lại một tấm ảnh nào đó sẽ khá mất thời gian vì phải duyệt tuần tự tất cả ảnh. Do đó, cần có một ứng dụng tự động phân loại các hình ảnh để giúp tìm kiếm nhanh hơn.

Ví dụ, nếu các ảnh của tài liệu được gom thành một nhóm thì khi tìm tài liệu, chỉ cần truy cập vào nhóm đó mà thôi.

Nhóm đã phân loại sẵn 21 chủ đề thường hay xuất hiện trong ảnh điện thoại bao gồm: Áo quần, bầu trời, bìa sách, biển, cây cối, ảnh chụp màn hình, điện thoại, đồ ăn, đồi núi, giày, hoa, hoá đơn, sông suối hồ, tài liệu, tòa nhà, xe cộ, người, selfie, đồng ruộng, đường phố và thú cưng.

Hình ảnh đưa vào có thể thuộc 1 trong các chủ đề trên. Việc phân loại ấy chính là bài toán mà nhóm đang hướng đến trong bài báo cáo ngày hôm nay.

2. Xác định yêu cầu bài toán

Input: Một tấm ảnh chụp bằng điện thoại.

Output: Tên chủ đề của tấm ảnh dựa trên 21 chủ đề nhóm đã đưa ra.

IV. THU THẬP VÀ PHÂN TÍCH DỮ LIỆU

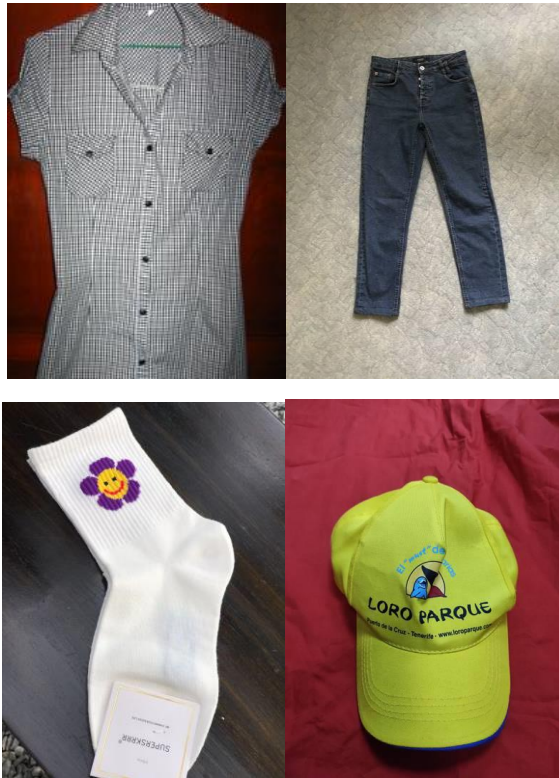
1. Thu thập dữ liệu

Dữ liệu của nhóm phần lớn là ảnh chụp từ điện thoại và một phần từ Internet. Dữ liệu được phân chia thành 21 chủ đề khác nhau: Áo quần, bầu trời, bìa sách, biển, cây cối, ảnh chụp màn hình, điện thoại, đồ ăn, đồi núi, giày, hoa, hoá đơn, sông suối hồ, tài liệu, tòa nhà, xe cộ, người, selfie, đồng ruộng, đường phố và thú cưng.

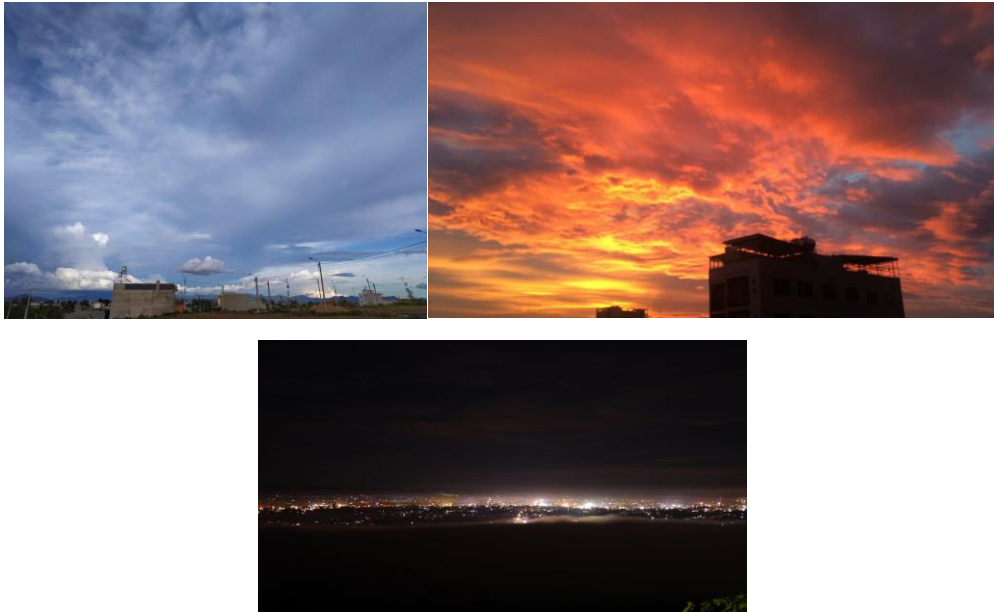
Ảnh chụp từ điện thoại gồm ảnh tự chụp và ảnh cung cấp bởi người thân và bạn bè. Ảnh lấy trên Internet từ nhiều nguồn khác nhau để bù vào những chủ đề có ít ảnh.

Một vài ví dụ cho dữ liệu của các chủ đề:

- Áo quần: gồm ảnh chỉ có áo và/hoặc quần và/hoặc có phụ kiện như mũ, vớ,... không bao gồm các chủ đề khác.



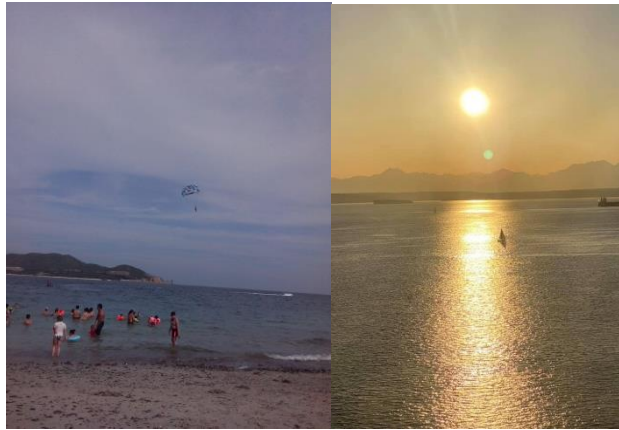
- Bầu trời: Ảnh phần lớn là bầu trời, nếu có chủ đề khác đồi núi, biển,... thì ưu tiên các chủ đề đó.



- Bìa sách: Gồm ảnh chụp đối diện, hoặc nghiêng không đáng kể của các bìa sách.



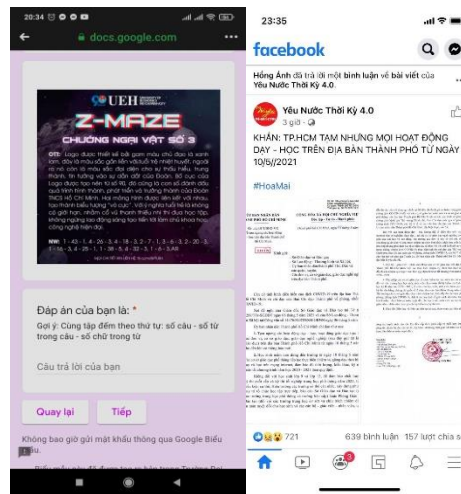
- Biển: ảnh chụp biển, bờ biển, nếu có người hoặc selfie trên chủ đề biển thì vẫn tính là biển.



- Cây cối: là ảnh gồm phần lớn là một hoặc nhiều cây, không có chủ đề khác hoặc các chủ đề khác chiếm quá ít.



- Ảnh chụp màn hình: các ảnh chụp màn hình, thường ảnh có biểu tượng các cột sóng điện thoại và pin, thời gian, ...



- Điện thoại: Ảnh chỉ có chủ đề là một hay nhiều chiếc điện thoại.



- Đồ ăn: Ảnh gồm đồ ăn và thức uống. Chủ đề khá phổ biến trên điện thoại hiện nay.



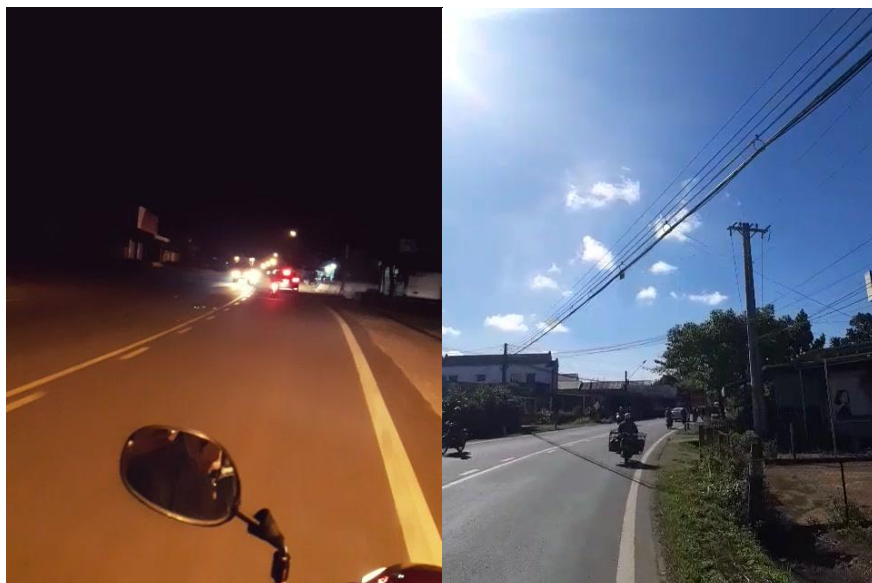
- Đồi núi: Ảnh có xuất hiện đồi núi có thể có cả cây cối, không ưu tiên ảnh có người và sông suối hồ.



- Đồng ruộng: Ảnh đồng ruộng phần lớn gồm cây lúa và cảnh quang của đồng ruộng.



- Đường phố: ảnh xuất hiện con đường.



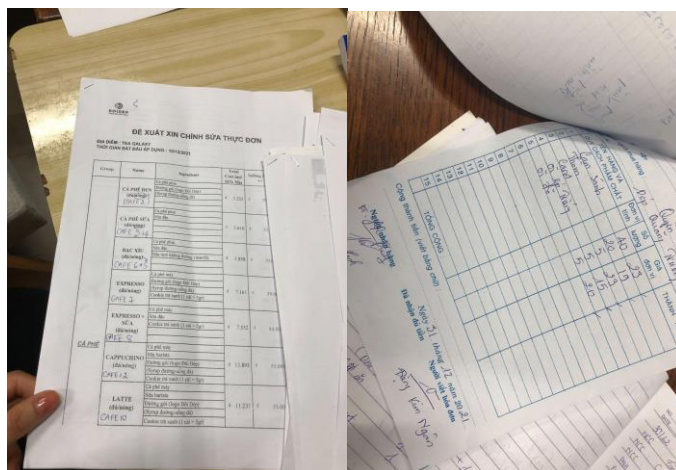
- Giày: Ảnh chụp các đôi giày từ nhiều góc độ và có thể chụp chân lúc đang mang giày.



- Hoa: Ảnh gồm nhiều loại hoa khác nhau và được chụp từ nhiều góc.



- Hoá đơn: gồm ảnh hoá đơn, phiếu nhập hàng, phiếu xuất hàng, giấy nộp tiền,...



- Người: Ảnh gồm một hoặc nhiều người, dễ phát hiện người trong ảnh, không bao gồm chủ đề đường phố, selfie.



- Selfie: Ảnh chân dung từ 1 đến 3 người.



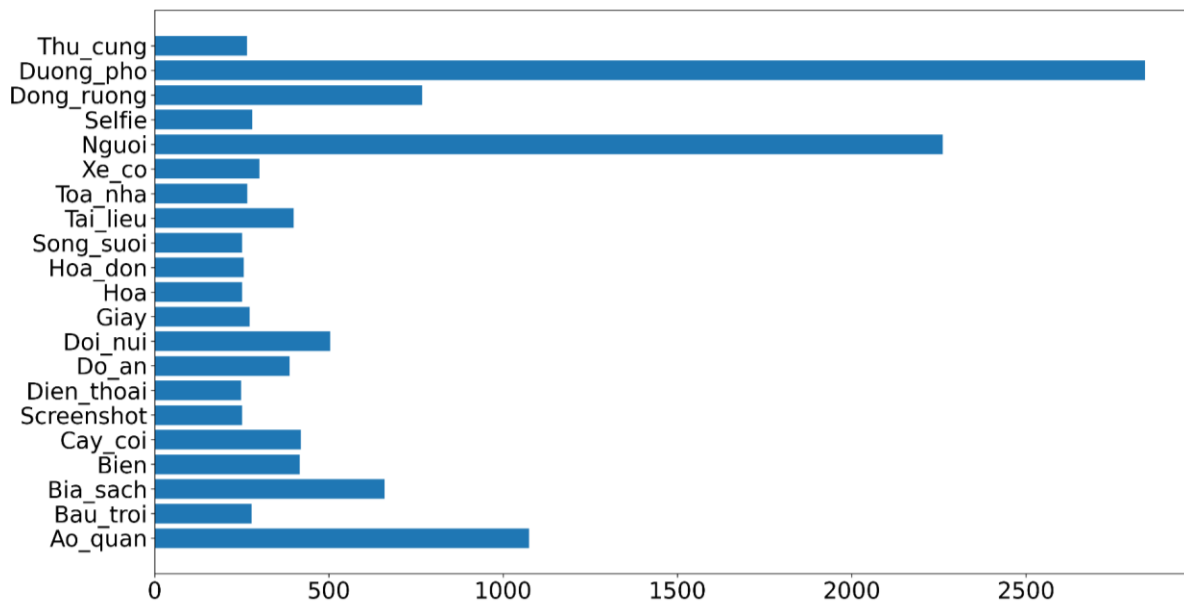
- Sông suối hồ: Ảnh xuất hiện sông suối hay hồ, ưu tiên chủ đề sông suối hồ hơn bầu trời và đồi núi.



- Xe cộ: chủ đề chính là xe, có thể là xe 2 bánh hoặc 4 bánh.



2. Phân tích dữ liệu

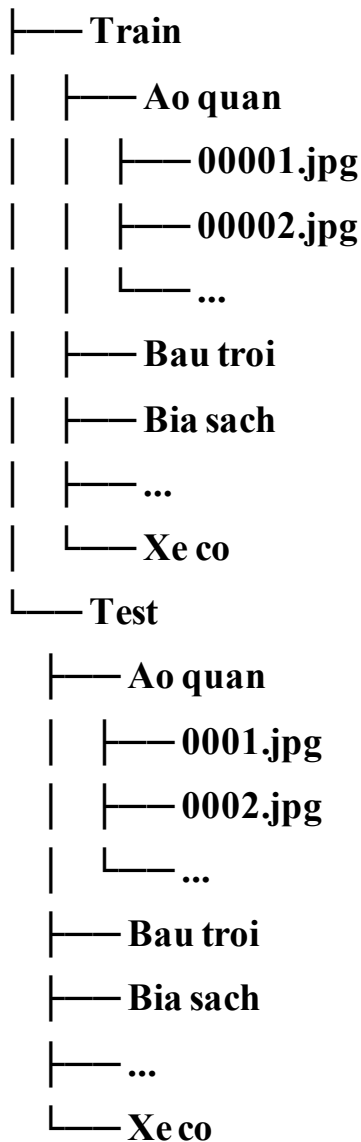


Biểu đồ thể hiện số lượng ảnh của mỗi chủ đề

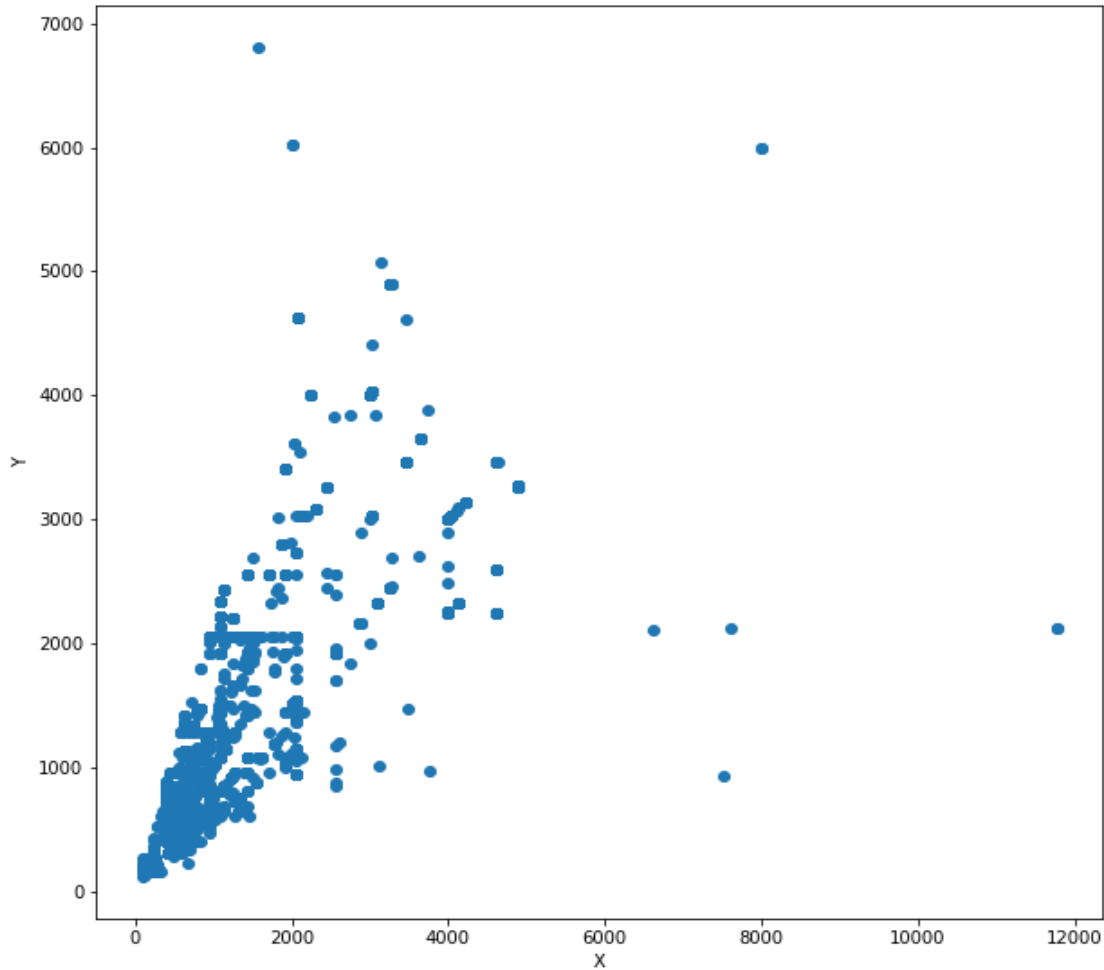
Mỗi chủ đề có số lượng ảnh dao động từ 251 đến 2841 ảnh. Các chủ đề như đường phố, người, áo quần có số lượng lớn hơn rất nhiều so với các chủ đề còn lại. Ảnh của chủ đề đường phố phần lớn được lấy từ video quay đường phố. Vì nhu cầu chụp ảnh ở hai chủ đề quần áo và người hiện nay khá phổ biến nên lượng dữ liệu thuộc hai chủ đề này khá cao.

Sơ đồ thư mục dữ liệu sẽ có dạng như sau:

Data



Dataset được chia làm 2 phần train và test, với phần test có số lượng ảnh là đúng 50 ảnh lấy ngẫu nhiên cho mỗi chủ đề. Như vậy số lượng ảnh tập train là 11600 ảnh, và số lượng ảnh của tập test là 1050 ảnh. Các ảnh được đánh số từ 1 cho đến 11600 đối với tập train và 1050 đối với tập test.



Phân phối kích thước ảnh theo chiều rộng và chiều cao

Phần lớn ảnh được chụp từ điện thoại và có kích thước phần lớn nằm trong khoảng dưới 2000x2000. Nhóm đã chọn lọc và loại bỏ các dữ liệu có kích thước quá lớn vì khi train nhóm đã gặp phải lỗi Decompression Bomb Warning khi ảnh vượt quá kích cỡ cho phép (trên 89 478 485 pixel).

Chiều ngang ảnh thấp nhất là 92.

Chiều ngang ảnh cao nhất là 11776.

Chiều dọc ảnh thấp nhất là 116.

Chiều dọc ảnh cao nhất là 6808.

Một số ảnh có kích thước chiều ngang chênh lệch nhiều so với chiều dọc hoặc ngược lại là hình chụp theo chế độ panorama trên điện thoại.

V. ĐỀ XUẤT MÔ HÌNH

1. Mô hình sử dụng

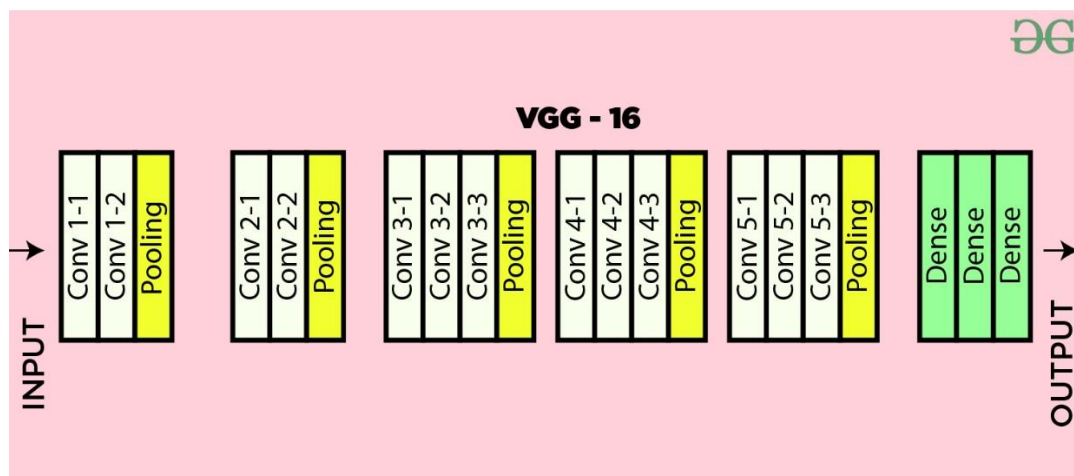
Nhóm sử dụng ba mô hình tiêu biểu của CNN gồm VGG16, Xception và InceptionV3.

2. Giới thiệu mô hình

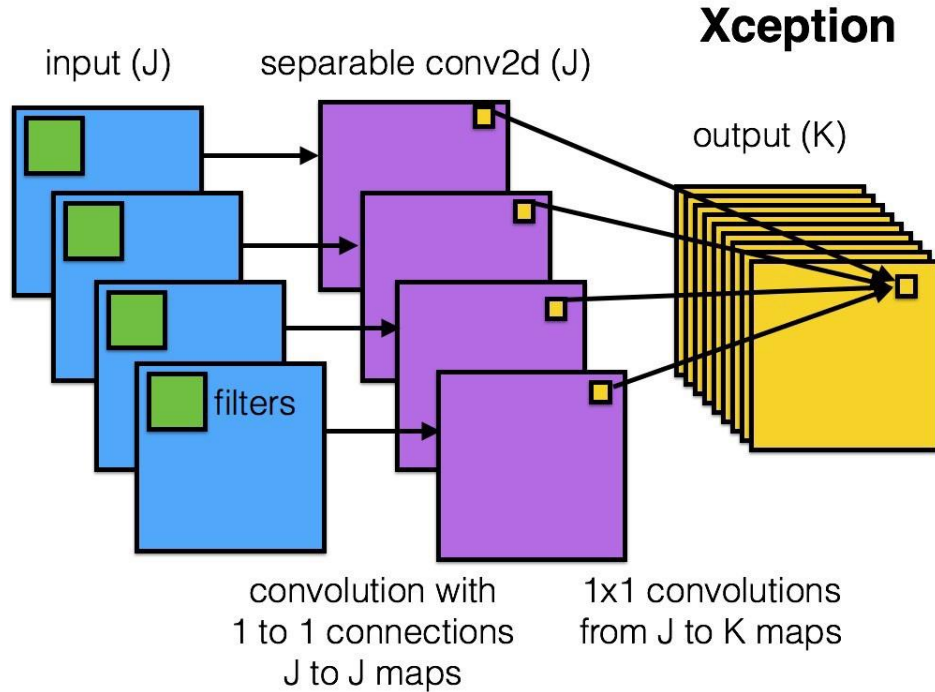
a. Mô hình VGG16

VGG16 là một Kiến trúc mạng nơ-ron liên kết (CNN) đơn giản và được sử dụng rộng rãi được sử dụng cho ImageNet.

Kiến trúc VGG16 được phát triển và giới thiệu bởi Karen Simonyan và Andrew Zisserman từ Đại học Oxford, vào năm 2014, thông qua bài báo của họ “Các mạng kết hợp rất sâu để nhận dạng hình ảnh quy mô lớn”. 'VGG' là tên viết tắt của Visual Geometry Group, là một nhóm các nhà nghiên cứu tại Đại học Oxford đã phát triển kiến trúc này.



b. Mô hình Xception



Mạng Xception là viết tắt của Extreme Inception, có thể được dịch như là phiên bản mạnh mẽ hơn của mô hình mạng Inception của Google.

Đây là một kiến trúc mạng nơ-ron tích chập hoàn toàn dựa trên các lớp tích chập phân tách theo chiều sâu.

Kiến trúc này được đề xuất bởi François Chollet (người tạo ra Keras) và điều duy nhất mà anh ấy mang đến cho Inception là anh ấy tạo ra sự phức tạp một cách tối ưu để chúng mất ít thời gian hơn. Điều này đạt được bằng cách tách các phần chập 2D thành 2 phần chập 1D.

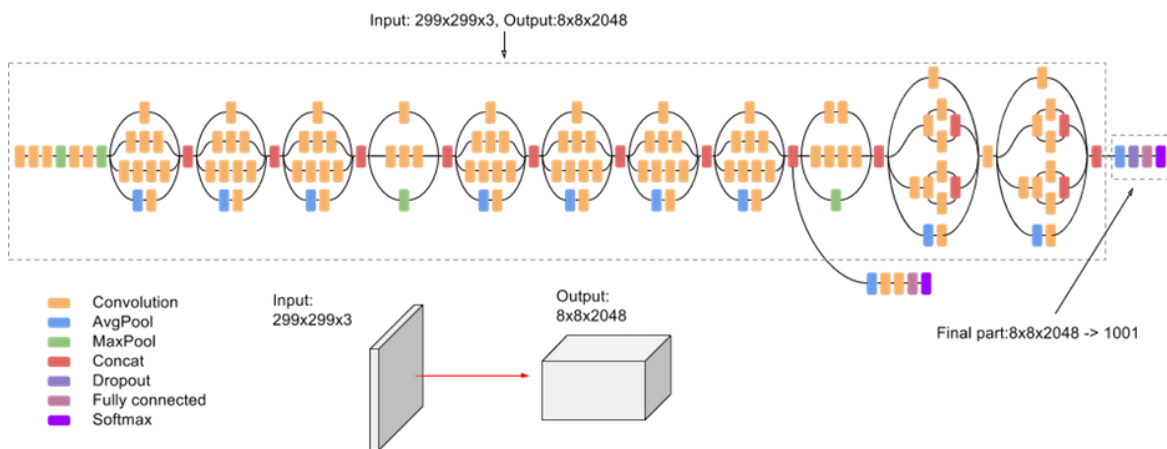
c. Mô hình InceptionV3

Vào năm 2014, mạng GoogLeNet (Inception-V1) đã giành chiến thắng ở cuộc thi ILSVRC (ImageNet Large-Scale Visual Recognition Challenge).

Việc cùng kết hợp đồng thời các bộ lọc có kích thước khác vào cùng một block có thể mang lại hiệu quả đó chính là kiến trúc khối Inception.

Inception-V3 là kế thừa của Inception-V1 bao gồm 24 triệu tham số. Toàn bộ các layer tích chập của Inception-V3 được theo sau bởi một layer batch normalization và một ReLU activation. Batch normalization là kỹ thuật chuẩn hóa đầu vào theo từng minibatch tại mỗi layer theo phân phối chuẩn hóa $N(0,1)$, giúp cho quá trình huấn luyện thuật toán nhanh hơn.

Inception-V3 giải quyết được vấn đề thắt cổ chai (representational bottlenecks). Tức là kích thước của các layers không bị giảm một cách đột ngột. Đồng thời Inception-V3 có một cách tính toán hiệu quả hơn nhờ sử dụng phương pháp nhân tố (factorisation methods).



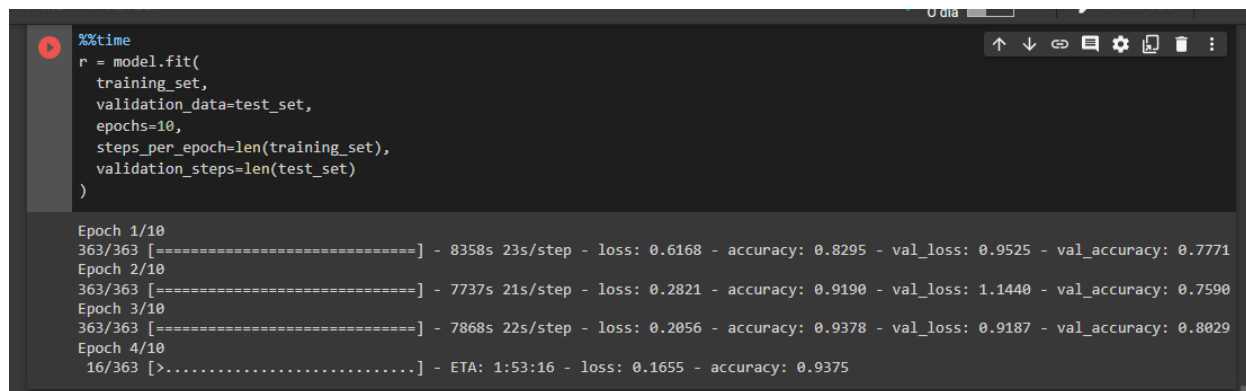
VI. HUẤN LUYỆN MÔ HÌNH

Nhóm sử dụng các mô hình CNN pre-trained được xây dựng bởi thư viện keras để tiến hành huấn luyện.

Một Epoch được tính là khi chúng ta đưa tất cả dữ liệu vào mạng neural network 1 lần.

Mô hình VGG16

Trong quá trình train mô hình khá lâu dẫn đến nhiều lúc chương trình bị lỗi giữa chừng, trọng số của mô hình cũng như các biến mất hết. Do đó, thay vì train một lúc 5 epoch thì em lưu lại mô hình mỗi khi đạt được kết quả tốt nhất, và chia nhỏ mỗi lần train trên 2 đến 3 epoch.



```
%%time
r = model.fit(
    training_set,
    validation_data=test_set,
    epochs=10,
    steps_per_epoch=len(training_set),
    validation_steps=len(test_set)
)
```

Epoch 1/10
363/363 [=====] - 8358s 23s/step - loss: 0.6168 - accuracy: 0.8295 - val_loss: 0.9525 - val_accuracy: 0.7771
Epoch 2/10
363/363 [=====] - 7737s 21s/step - loss: 0.2821 - accuracy: 0.9190 - val_loss: 1.1440 - val_accuracy: 0.7590
Epoch 3/10
363/363 [=====] - 7868s 22s/step - loss: 0.2056 - accuracy: 0.9378 - val_loss: 0.9187 - val_accuracy: 0.8029
Epoch 4/10
16/363 [>.....] - ETA: 1:53:16 - loss: 0.1655 - accuracy: 0.9375

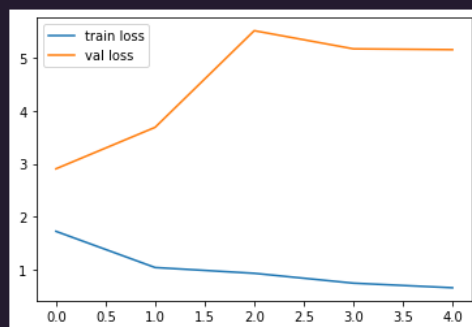
Mô hình Xception

Khi train dữ liệu với epochs=5, thu được các thông số như sau:

```
%%time
r = model.fit(
    training_set,
    validation_data=test_set,
    epochs=5,
    steps_per_epoch=len(training_set),
    validation_steps=len(test_set)
)

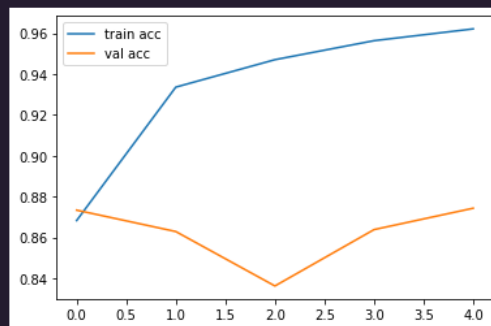
Epoch 1/5
363/363 [=====] - 5860s 16s/step - loss: 1.7257 - accuracy: 0.8683 - val_loss: 2.9034 - val_accuracy: 0.8733
Epoch 2/5
363/363 [=====] - 3285s 9s/step - loss: 1.0435 - accuracy: 0.9336 - val_loss: 3.6847 - val_accuracy: 0.8629
Epoch 3/5
363/363 [=====] - 3303s 9s/step - loss: 0.9346 - accuracy: 0.9471 - val_loss: 5.5055 - val_accuracy: 0.8362
Epoch 4/5
363/363 [=====] - 3273s 9s/step - loss: 0.7490 - accuracy: 0.9564 - val_loss: 5.1636 - val_accuracy: 0.8638
Epoch 5/5
363/363 [=====] - 3329s 9s/step - loss: 0.6622 - accuracy: 0.9621 - val_loss: 5.1481 - val_accuracy: 0.8743
CPU times: user 8h 14min 50s, sys: 16min 1s, total: 8h 30min 51s
Wall time: 5h 20min 12s
```

```
[ ] plt.plot(r.history['loss'], label='train loss')
plt.plot(r.history['val_loss'], label='val loss')
plt.legend()
plt.show()
plt.savefig('LossVal_loss')
```



<Figure size 432x288 with 0 Axes>

```
[ ] plt.plot(r.history['accuracy'], label='train acc')
plt.plot(r.history['val_accuracy'], label='val acc')
plt.legend()
plt.show()
plt.savefig('AccVal_acc')
```



<Figure size 432x288 with 0 Axes>

Mô hình Inception V3

Ban đầu với epochs=5, các thông số đã thu được như hình sau:

```
%%time
r = model.fit(
    training_set,
    validation_data=test_set,
    epochs=5,
    steps_per_epoch=len(training_set),
    validation_steps=len(test_set)
)
```

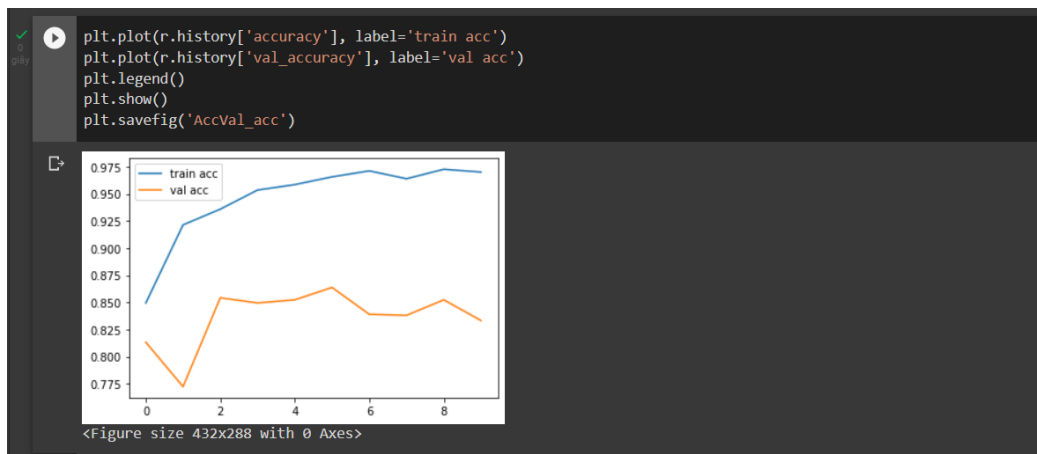
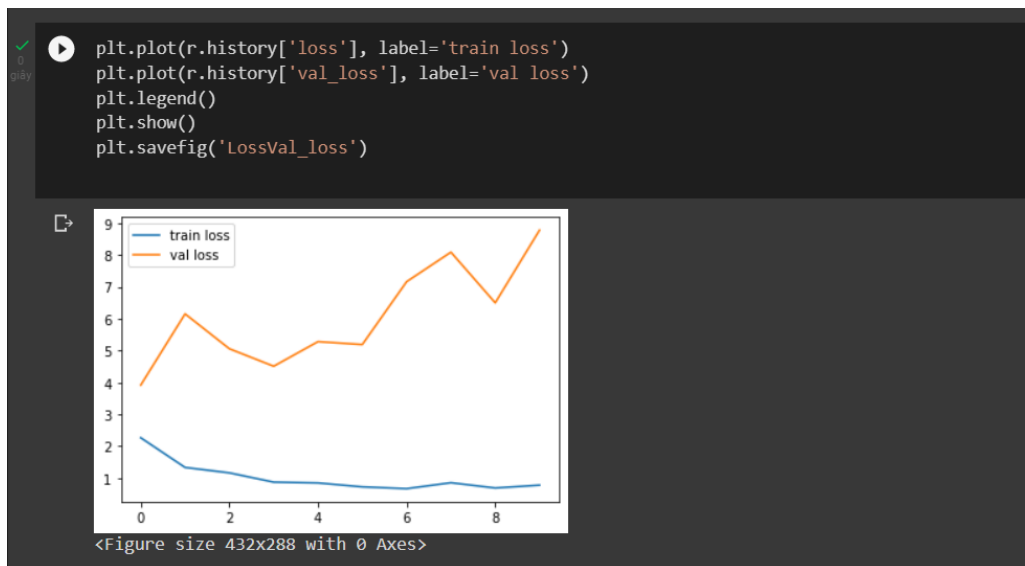
```
Epoch 1/5
3/363 [.....] - ETA: 1:53:28 - loss: 13.1782 - accuracy: 0.1979/usr/local/lib/python3.7/dist-packages/PIL/Image.py:288
DecompressionBombWarning,
363/363 [=====] - 6712s 18s/step - loss: 2.5330 - accuracy: 0.8493 - val_loss: 5.9463 - val_accuracy: 0.7571
Epoch 2/5
363/363 [=====] - 2192s 6s/step - loss: 1.3484 - accuracy: 0.9215 - val_loss: 3.9757 - val_accuracy: 0.8229
Epoch 3/5
363/363 [=====] - 2194s 6s/step - loss: 1.0492 - accuracy: 0.9429 - val_loss: 5.0710 - val_accuracy: 0.8333
Epoch 4/5
363/363 [=====] - 2216s 6s/step - loss: 1.0435 - accuracy: 0.9471 - val_loss: 5.4493 - val_accuracy: 0.8229
Epoch 5/5
363/363 [=====] - 2216s 6s/step - loss: 0.8079 - accuracy: 0.9550 - val_loss: 6.3948 - val_accuracy: 0.8457
CPU times: user 4h 59min 26s, sys: 10min 25s, total: 5h 9min 52s
Wall time: 4h 20min 8s
```



Khi nhóm quyết định nâng epochs lên 10 để thử cải thiện kết quả training thì thu được kết quả như sau:

```
+ Mã + Văn bản
validation_steps=len(test_set)
)

Epoch 1/10
100/363 [====>.....] - ETA: 54:48 - loss: 4.1808 - accuracy: 0.7412/usr/local/lib/python3.7/dist-packages/PIL/Image.py:2800: DecompressionBombWarning: Image size (1
DecompressionBombWarning,
363/363 [=====] - 5051s 14s/step - loss: 2.2641 - accuracy: 0.8495 - val_loss: 3.9190 - val_accuracy: 0.8133
Epoch 2/10
363/363 [=====] - 2266s 6s/step - loss: 1.3328 - accuracy: 0.9215 - val_loss: 6.1537 - val_accuracy: 0.7724
Epoch 3/10
363/363 [=====] - 2396s 7s/step - loss: 1.1632 - accuracy: 0.9359 - val_loss: 5.0582 - val_accuracy: 0.8543
Epoch 4/10
363/363 [=====] - 2292s 6s/step - loss: 0.8720 - accuracy: 0.9535 - val_loss: 4.5098 - val_accuracy: 0.8495
Epoch 5/10
363/363 [=====] - 2386s 7s/step - loss: 0.8465 - accuracy: 0.9585 - val_loss: 5.2819 - val_accuracy: 0.8524
Epoch 6/10
363/363 [=====] - 2445s 7s/step - loss: 0.7231 - accuracy: 0.9657 - val_loss: 5.1904 - val_accuracy: 0.8638
Epoch 7/10
363/363 [=====] - 2356s 6s/step - loss: 0.6671 - accuracy: 0.9712 - val_loss: 7.1635 - val_accuracy: 0.8390
Epoch 8/10
363/363 [=====] - 2268s 6s/step - loss: 0.8531 - accuracy: 0.9640 - val_loss: 8.0900 - val_accuracy: 0.8381
Epoch 9/10
363/363 [=====] - 2309s 6s/step - loss: 0.6876 - accuracy: 0.9727 - val_loss: 6.5000 - val_accuracy: 0.8524
Epoch 10/10
363/363 [=====] - 2282s 6s/step - loss: 0.7773 - accuracy: 0.9701 - val_loss: 8.7810 - val_accuracy: 0.8333
CPU times: user 10h 52min 49s, sys: 16min 33s, total: 11h 9min 22s
Wall time: 7h 16min 42s
```



VII. KẾT QUẢ THỰC NGHIỆM

	Thực tế (có)	Thực tế (không)	
Dự đoán (có)	TP	FP	Precision
Dự đoán (không)	FN	TN	
Recall			

Confusion Matrix là một phương pháp đánh giá kết quả của những bài toán phân loại với việc xem xét cả những chỉ số về độ chính xác và độ bao quát của các dự đoán cho từng lớp. Một confusion matrix gồm 4 chỉ số sau đối với mỗi lớp phân loại:

- **True Positive (TP):** là những hình mà ta đoán đúng lớp thực sự của nó.
- **True Negative (TN):** là những hình mà thực không thuộc lớp nào đó, và ta đoán nó không thuộc lớp đó.
- **False Positive (FP):** là những hình mà ta đoán sai lớp thực sự của nó.
- **False Negative (FN):** là những hình thuộc một lớp nào đó mà ta đoán nó không thuộc lớp đó.

Accuracy được tính bằng số lượng ảnh được dự đoán đúng trên tổng số ảnh trong tập test. Accuracy được tính bằng công thức sau:

$$\text{accuracy} = \frac{TP + TN}{TP + TN + FP + FN}$$

Precision trả lời cho câu hỏi trong các trường hợp được dự báo là positive thì có bao nhiêu trường hợp là đúng? Và tất nhiên precision càng cao thì mô hình của chúng ta càng tốt trong việc phân loại hồ sơ BAD (Nhóm positive).

$$\text{precision} = \frac{TP}{TP + FP}$$

Recall đo lường tỷ lệ dự báo chính xác các trường hợp positive trên toàn bộ các mẫu thuộc nhóm positive. Công thức của recall như sau:

$$\text{Recall} = \frac{TP}{TP + FN}$$

Tuy nhiên, chỉ có Precision hay chỉ có Recall thì không đánh giá được chất lượng mô hình.

Chỉ dùng Precision, mô hình chỉ đưa ra dự đoán cho một điểm mà nó chắc chắn nhất. Khi đó Precision = 1, tuy nhiên ta không thể nói là mô hình này tốt.

Chỉ dùng Recall, nếu mô hình dự đoán tất cả các điểm đều là positive. Khi đó Recall = 1, tuy nhiên ta cũng không thể nói đây là mô hình tốt.

Khi đó F1-score được sử dụng. F1-score là trung bình điều hòa (harmonic mean) của precision và recall (giả sử hai đại lượng này khác 0). F1-score được tính theo công thức:

$$F_1 = \left(\frac{\text{recall}^{-1} + \text{precision}^{-1}}{2} \right)^{-1} = 2 \times \frac{\text{precision} \cdot \text{recall}}{\text{precision} + \text{recall}}.$$

Mô hình VGG16

	precision	recall	f1-score	support
Ao quan	0.92	0.90	0.91	50
Bau troi	1.00	0.26	0.41	50
Bia sach	0.98	0.94	0.96	50
Bien	0.64	0.74	0.69	50
Cay coi	0.75	0.12	0.21	50
Chup man hinh	0.91	1.00	0.95	50
Dien thoai	0.96	0.94	0.95	50
Do an	0.43	0.98	0.60	50
Doi nui	0.77	0.48	0.59	50
Dong ruong	1.00	0.86	0.92	50
Duong pho	0.79	0.98	0.87	50
Giay	0.98	0.84	0.90	50
Hoa	0.68	0.88	0.77	50
Hoa don	0.84	0.92	0.88	50
Nguai	0.35	0.98	0.52	50
Selfie	0.95	0.40	0.56	50
Song suoi ho	0.90	0.38	0.54	50
Tai lieu	1.00	0.84	0.91	50
Thu cung	0.94	0.32	0.48	50
Toa nha	0.53	0.88	0.66	50
Xe co	1.00	0.70	0.82	50
accuracy			0.73	1050
macro avg	0.82	0.73	0.72	1050
weighted avg	0.82	0.73	0.72	1050

Kết quả dự đoán của mô hình VGG16 trên tập test được thể hiện ở hai hình bên.

Precision của chủ đề Áo quần được tính như sau:

$$precision_{Aoquan} = \frac{45}{45 + 2 + 1 + 1} \approx 0.92$$

Recall của chủ đề Áo quần là:

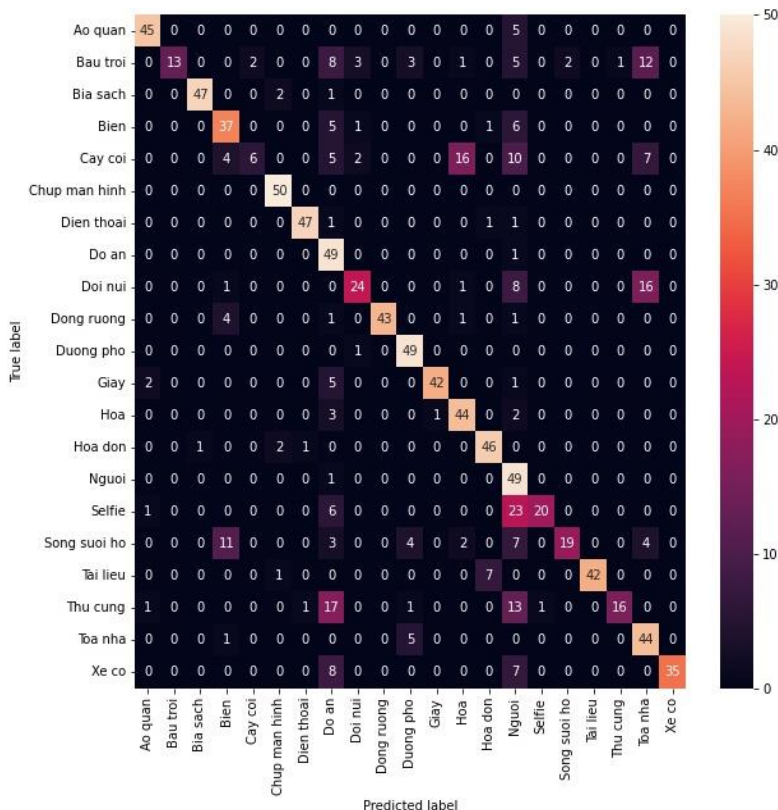
$$recall_{Aoquan} = \frac{45}{45 + 5} = 0.90$$

Và F1-score của chủ đề Áo quần là:

$$F1_{Aoquan} = 2 * \frac{0.92 * 0.9}{0.92 + 0.9} \approx 0.91$$

Như vậy kết quả tương ứng với số liệu trong hình bên.

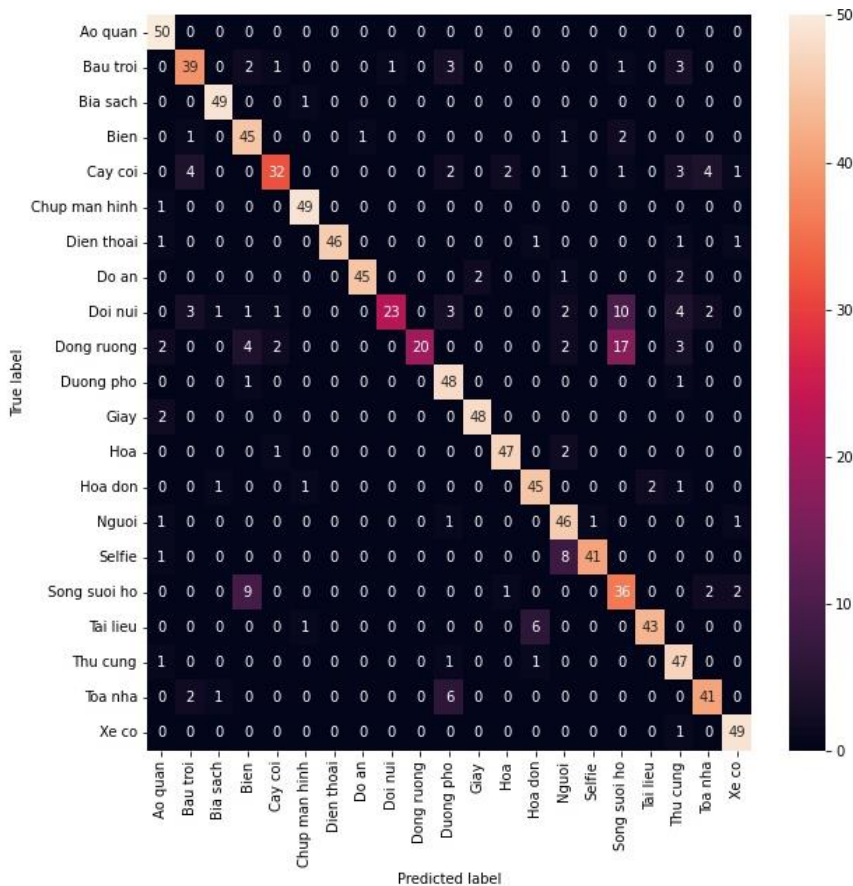
Nhìn tổng quát Confusion matrix bên, có thể nhận thấy mô hình khó dự đoán đúng các chủ đề thuộc Bầu trời, Biển, Cây cối, Đồi núi, Selfie, Sông suối hồ, Thú cưng, Xe cộ. Khi dự đoán chủ đề bầu trời thì dễ nhầm qua chủ đề Đồ ăn, Người và Tòa nhà. Dễ nhầm lẫn giữa Selfie và Người,... Tuy vậy vẫn dự đoán tốt các chủ đề Chụp màn hình, Đường phố và Người.



Mô hình Xception

	precision	recall	f1-score	support
Ao quan	0.85	1.00	0.92	50
Bau troi	0.80	0.78	0.79	50
Bia sach	0.94	0.98	0.96	50
Bien	0.73	0.90	0.80	50
Cay coi	0.86	0.64	0.74	50
Chup man hinh	0.94	0.98	0.96	50
Dien thoai	1.00	0.92	0.96	50
Do an	0.98	0.90	0.94	50
Doi nui	0.96	0.46	0.62	50
Dong ruong	1.00	0.40	0.57	50
Duong pho	0.75	0.96	0.84	50
Giay	0.96	0.96	0.96	50
Hoa	0.94	0.94	0.94	50
Hoa don	0.85	0.90	0.87	50
Nguai	0.73	0.92	0.81	50
Selfie	0.98	0.82	0.89	50
Song suoi ho	0.54	0.72	0.62	50
Tai lieu	0.96	0.86	0.91	50
Thu cung	0.71	0.94	0.81	50
Toa nha	0.84	0.82	0.83	50
Xe co	0.91	0.98	0.94	50
accuracy			0.85	1050
macro avg	0.87	0.85	0.84	1050
weighted avg	0.87	0.85	0.84	1050

Đối với chủ đề dễ nhầm lẫn như Hóa đơn, theo công thức ta tính được các chỉ số Precision = 0.85, Recall = 0.90 và F1-score = 0.87.



Cụ thể ở chủ đề Hóa đơn

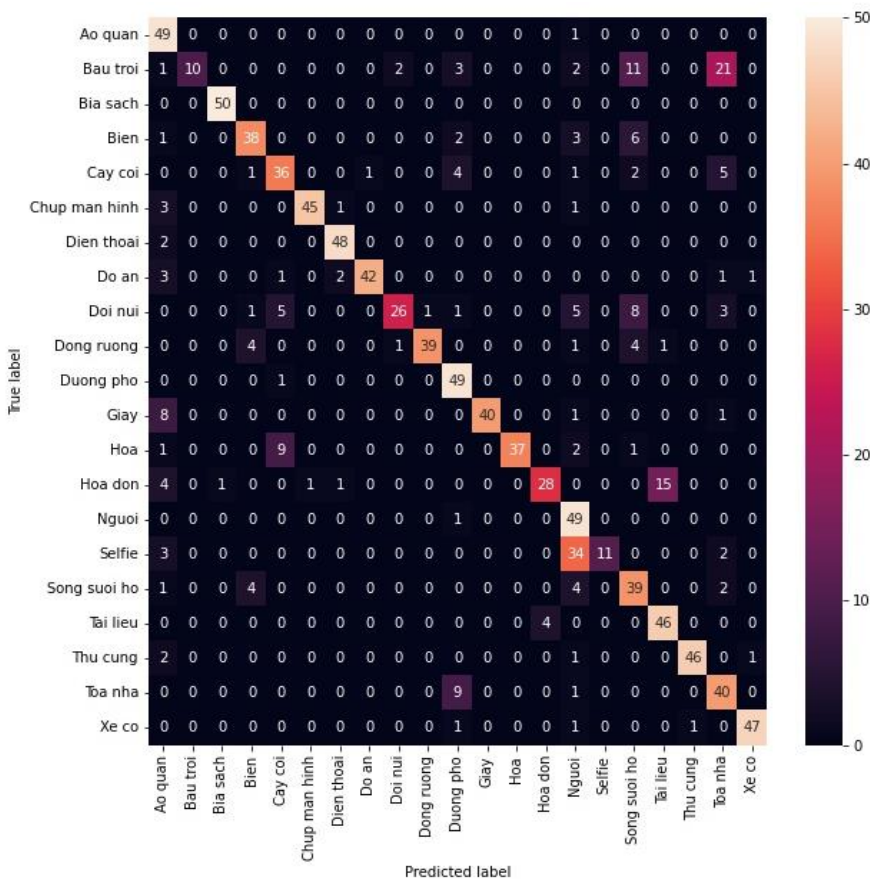
- FP = 8, trong đó 1 (Điện thoại), 1 (Thú cưng), 6 (Tài liệu).
- FN = 5, trong đó 1 (Bìa sách), 1 (Chụp màn hình), 2 (Tài liệu), 1 Thú cưng.

Qua đó thấy được mô hình này dự đoán khá tốt các chủ đề dễ nhầm lẫn như Hóa đơn và Tài liệu hoặc Người và Selfie,... Số lượng sai lệch khá ít.

Mô hình InceptionV3

	precision	recall	f1-score	support
Ao quan	0.63	0.98	0.77	50
Bau troi	1.00	0.20	0.33	50
Bia sach	0.98	1.00	0.99	50
Bien	0.79	0.76	0.78	50
Cay coi	0.69	0.72	0.71	50
Chup man hinh	0.98	0.90	0.94	50
Dien thoai	0.92	0.96	0.94	50
Do an	0.98	0.84	0.90	50
Doi nui	0.90	0.52	0.66	50
Dong ruong	0.97	0.78	0.87	50
Duong pho	0.70	0.98	0.82	50
Giay	1.00	0.80	0.89	50
Hoa	1.00	0.74	0.85	50
Hoa don	0.88	0.56	0.68	50
Nguai	0.46	0.98	0.62	50
Selfie	1.00	0.22	0.36	50
Song suoi ho	0.55	0.78	0.64	50
Tai lieu	0.74	0.92	0.82	50
Thu cung	0.98	0.92	0.95	50
Toa nha	0.53	0.80	0.64	50
Xe co	0.96	0.94	0.95	50
accuracy			0.78	1050
macro avg	0.84	0.78	0.77	1050
weighted avg	0.84	0.78	0.77	1050

Ở mô hình InceptionV3, mô hình dễ dự đoán nhầm chủ đề Bầu trời sang chủ đề Sông suối hồ và Tòa nhà. Chủ đề hoá đơn thì dễ nhầm lẫn sang chủ đề tài liệu. Dễ dự đoán chủ đề đồi núi nhầm sang chủ đề Cây cối, Người, Sông suối hồ, Tòa nhà.



Nhận xét chung thì chỉ số accuracy của mô hình Xception là cao nhất là 0.85, tiếp đến là mô hình InceptionV3 với accuracy 0.78, thấp nhất là VGG16 với accuracy 0.73. Các chủ đề dễ dự đoán sai nhất là chủ đề Bầu trời, Đồi núi, Đồng ruộng, Selfie, ...

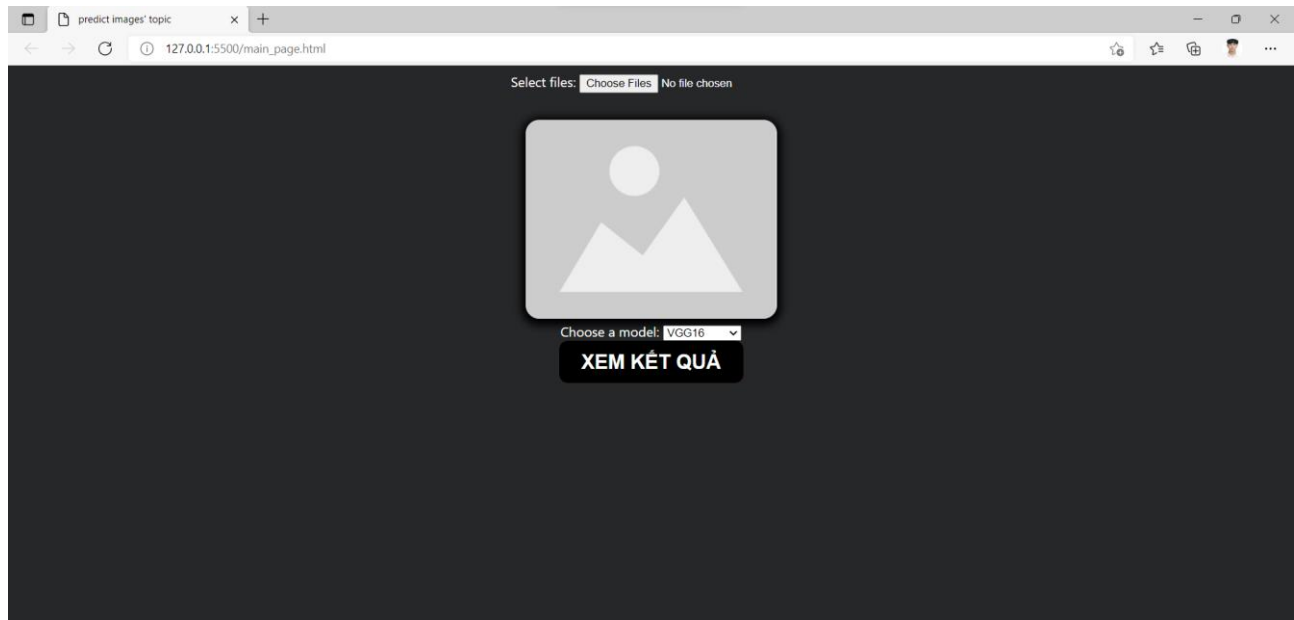
Thời gian train của từng mô hình như bảng sau:

Mô hình	Thời gian train (h)
VGG16	~10.5
Xception	~5.3
InceptionV3	~4.3

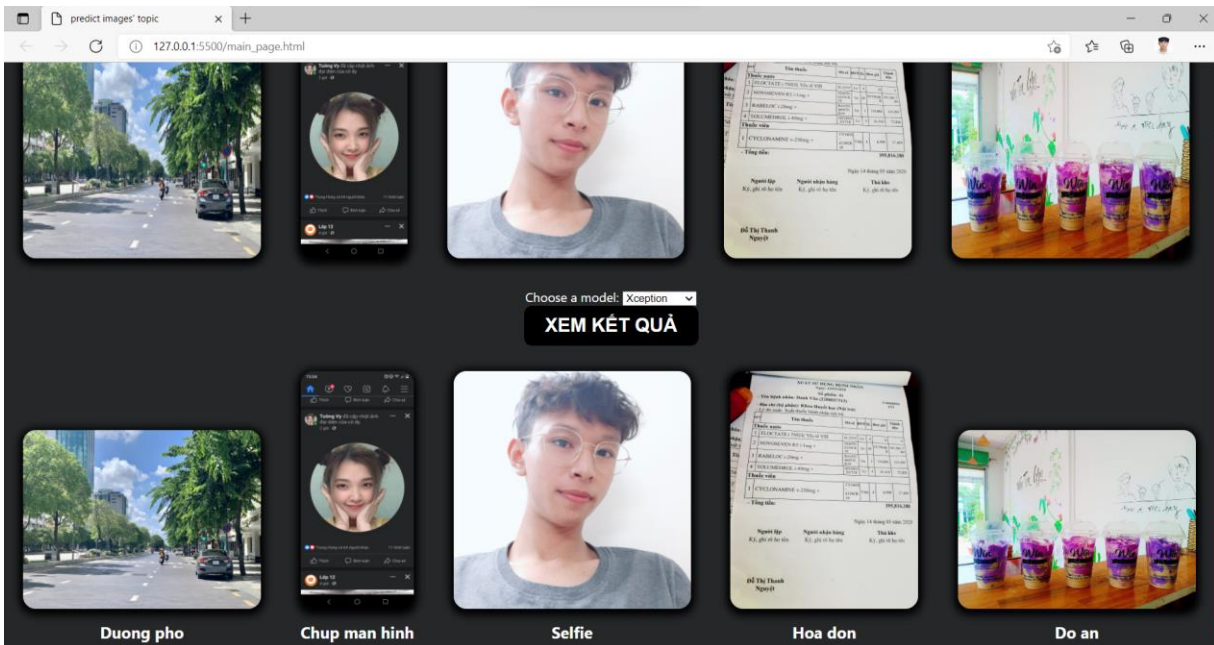
Nhận xét

Độ chính xác trên tập test VGG16 là nhỏ nhất trong khi thời gian train lâu nhất lên đến hơn 10 tiếng đồng hồ. Độ chính xác của mô hình InceptionV3 cao hơn so với mô hình VGG16 nhưng thời gian train nhỏ nhất trong cả 3 mô hình. Độ chính xác của mô hình Xception là cao nhất trong khi thời gian train chỉ bằng 1 nửa thời gian train của mô hình VGG16.

VIII. TRIỂN KHAI MÔ HÌNH LÊN WEBSITE



Ban đầu trang web cho phép người dùng chọn ảnh để dự đoán, người dùng có thể chọn một hoặc nhiều ảnh tùy thích. Ngoài ra có thể tùy ý chọn model dự đoán gồm (VGG16, Xception và InceptionV3). Khi hoàn thành chọn ảnh và model thì chỉ cần nhấn vào nút “XEM KẾT QUẢ”. Sau đó chúng ta sẽ nhận được kết quả dự đoán như sau:



IX. KẾT LUẬN

1. Ưu điểm & nhược điểm

Ưu điểm	Nhược điểm
Tự thu thập được 12600 ảnh để làm dữ liệu huấn luyện cho các mô hình.	Số lượng các chủ đề còn chưa đa dạng. Số lượng ảnh giữa các chủ đề còn khá chênh lệch.
Tìm hiểu áp dụng được 3 mô hình CNN để train dữ liệu.	Kiến thức về các mô hình còn hạn chế. Chưa tuning các siêu tham số trong các mô hình.
Tìm hiểu và tạo được một trang web đơn giản để dự đoán chủ đề ảnh.	Chưa tạo được ứng dụng app được trên điện thoại. Thiếu tính năng cho ứng dụng web.

2. Hướng phát triển

Thiết kế thành ứng dụng hoàn chỉnh để có thể sử dụng trên điện thoại.

Thu thập thêm đa dạng dữ liệu, làm giàu dữ liệu để cải thiện mô hình.

Tiền xử lý dữ liệu trước khi đưa vào mô hình.

Tìm hiểu kỹ hơn các mô hình đã được sử dụng, tuning tham số của các mô hình.

Thêm đa dạng chủ đề hơn, có thể cho phép người dùng tự phân loại chủ đề theo ý muốn.

Thiết kế giao diện web cho dễ sử dụng, tăng tốc độ chạy nếu có thể, bổ sung các tính năng khác.

X. TÀI LIỆU THAM KHẢO

- [1] [Khoa học dữ liệu \(phamdinhhkhanh.github.io\)](http://phamdinhhkhanh.github.io)
- [2] <https://keras.io/api/applications/inceptionv3/>
- [3] <https://keras.io/api/applications/xception/>
- [4] <https://keras.io/api/applications/vgg/#vgg16-function>
- [5] <https://keras.io/api/applications/>
- [6] <https://math2it.com/hieu-confusion-matrix/>