Báo cáo bài tập lớn giữa kì môn học Xử lý ảnh

INT3404\_3

***Đề tài: “Phân loại ảnh hóa đơn đơn giản - simpleOcr”***

I, Thành viên

* Trần Quang Trung – 18021331
* Phạm Ngọc Tú
* Thịnh Thành Vinh

II, Nội dung bài tập lớn

a. Giới thiệu

Trong phần bài tập này, bạn sẽ cài đặt các thuật toán xử lý ảnh cơ bản để nhận dạng loại hoá đơn, theo ảnh đầu vào. Có tất cả 3 loại hoá đơn chính là highlands, phuclong, starbucks, trong trường hợp không hải 3 loại hoá đơn trên thì trả về là others. Tập ảnh ví dụ (được thu thập từ công cụ tìm kiếm google) được để trong thư mục sampledata với nhãn được để trong tệp labels.csv Các ảnh đầu vào bị ảnh hưởng bởi cái điều kiện ánh sáng, góc chụp và có thể bị xoay.

Các bạn cần cài đặt thuật toán trong tệp tin ***simple\_ocr.py*** với 2 hàm chính:

* Hàm khởi tạo:
* Hàm ***find\_label***: trả về nhãn hóa đơn của 1 ảnh đầu vào trong 4 loại: *others, highlands, phuclong, starbucks*

Các bạn có thể sử dụng ***main.py*** để chấm điểm

Các nhóm được khuyến khích sử dụng mô hình học sâu hoặc các phương pháp xử lí ảnh thông thường để giải bài toán. Điểm của nhóm sẽ được cho theo:

* Độ chính xác trên tệp kiểm tra (5pts)
* Báo cáo (5pts)

b. Chấm điểm

* Mỗi nhóm có tối đa không quá 3 người
* Kết quả phần độ chính xác sẽ được kiểm tra theo tập ảnh mới, khác tập ảnh hiện tại
* Phần báo cáo: cách tiếp cận, kết quả chạy thử, tổ chức chương trình

c. Timeline

* Week 08: hoàn thiện danh sách nhóm
* Week 14: nộp code + báo cáo

d. Giới hạn

* Sử dụng thêm các thư viện ngoài là không được phép
* Thời gian khởi tạo mô hình không quá 60s, tổng thời giãn xử lí không quá 120s
* Số ảnh test không quá 30 ảnh, kích thước không quá 2048x2048
* Chỉ được sử dụng thư viện trong file ***requirements.txt***
* Mô hình sử dụng không quá 50mb

e. Các phần nâng cao

* Xử lý các bài toán con: cắt hóa đơn, xoay hóa đơn, cân bằng sáng
* Tự thu thập và kiểm tra trên tập dữ liệu thu thập được
* Mở rộng bài toán: cho nhiều loại hóa đơn, nhận dạng các thông tin khác như giá tiền

III, Cách tiếp cận bài toán

Cách tiếp cận: sử dụng học sâu để phân lớp ảnh hóa đơn với tập datasets bao gồm các ảnh hóa đơn của các cửa hàng: starbucks coffee, highlands, phuclong và hóa đơn của cửa hàng khác

* Phương pháp học sâu được sử dụng: sử dụng mô hình CNN (Convolution Neural Network) để phân lớp
* Mô hình CNN được sử dụng: DenseNet121
* Tập datasets: chi tiết ở link github bên dưới
* Sử dụng Pytorch để train và test

IV, Mạng DenseNet

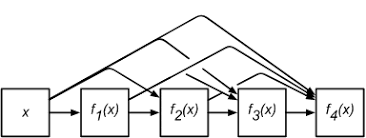
a. Ý tưởng

Ở ResNet chúng ta phân tách hàm số thành một hàm xác định và một hàm phi tuyến:

Cùng nhắc lại công thức khai triển Taylor tại :

Ta có thể thấy công thức của ResNet cũng gần tương tự như khai triển taylor tại đạo hàm bậc nhất,  tương ứng với thành phần số dư. Khai triển Taylor sẽ càng chuẩn xác nếu chúng ta phân rã được số dư thành nhiều đạo hàm bậc cao hơn.

Ý tưởng của DenseNet cũng như vậy, chúng ta sẽ sử dụng một mạng lưới các kết nối tắt dày đặc để liên kết các khối với nhau.

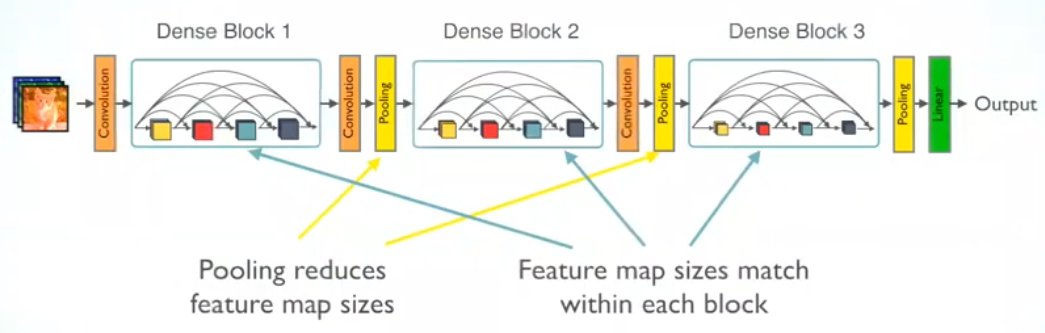


Hình 5.14: Ý tưởng mạng DenseNet

Từ đầu vào x ta sẽ áp dụng liên tiếp một chuỗi các ánh xạ liên tiếp với cấp độ phức tạp tăng dần:

b. Translation Layers

DenseNet sẽ khác so với ResNet đó là chúng ta không cộng trực tiếp vào mà thay vào đó, các đầu ra của từng phép ánh xạ có cùng kích thước dài và rộng sẽ được concatenate với nhau thành một khối theo chiều sâu. Sau đó để giảm chiều dữ liệu chúng ta áp dụng tầng chuyển tiếp (translation layer). Tầng này dùng một tầng tích chập 1×11×1 để giảm số lượng kênh, theo sau là một max pooling giúp giảm kích thước dài và rộng. Các bạn sẽ dễ dàng hình dung hơn qua hình vẽ bên dưới:

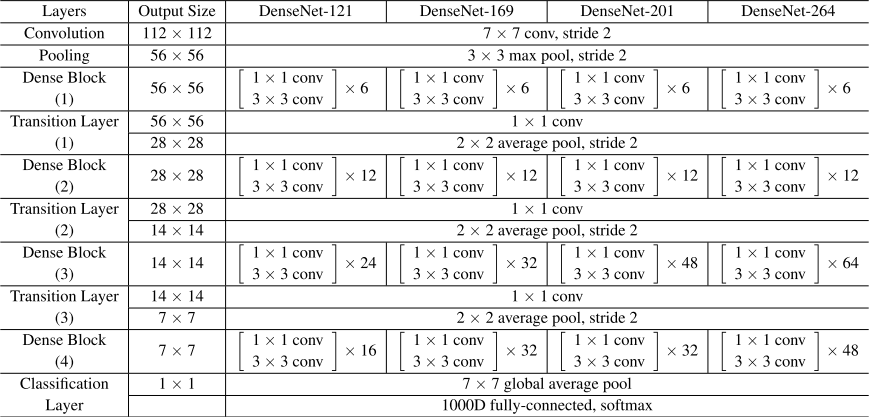


c. Kiến trúc Densenet

* Connectivity: layer thứ sẽ nhận tất cả các bản đồ đặc trưng của các layer trước: ... ,
* DenseBlock: gồm các lớp conv và được lặp lại

D =

* Growth Rate: Giả sử sau mỗi lần qua 1 dense layer thì bản đồ đặc trưng được bổ sung thêm đặc trưng chính là growth rate
* Bottleneck Layers: trong mỗi denseblock thì lớp nhân chập có tác dụng giảm tham số đầu vào cho lớp để làm tăng hiệu suất tính toán lớp chính là Bottleneck Layer



d. Lợi ích của DenseNet

* Khắc phục được lỗi Vanishing Gradient
* So với ResNet thì DenseNet có số lượng tham số giảm đáng kể và có so với connection của CNN truyền thống như VGG,
* Do mỗi layer trong DenseNet nhận tất cả các layer trước làm đầu vào nên DenseNet có các tính năng đa dạng và có xu hướng có mô hình phong phú hơn
* Trong DenseNet, classifier sử dụng các tính năng của tất cả các cấp độ phức tạp nên DenseNet hoạt động tốt khi dữ liệu đào tạo không đủ

V, Triển khai mô hình vào bài toán

a. Tiền xử lý

* Bước 1: Chuyển ảnh đầu vào thành ảnh màu xám
* Bước 2: Sử dụng Histogram Equalize để cân bằng sáng
* Bước 3: Sử dụng GaussianBlur để loại bớt nhiễu
* Bước 4: Sử dụng cv2.getStructuringElement để detect các vùng trắng
* Bước 5: Trước khi binarization, cần phải hiệu chỉnh độ chiếu sáng không đồng đều của nền
* Bước 6: Tìm contours
* Bước 7: Lấy ra contour lớn nhất
* Bước 8: Crop ảnh theo contour lớn nhất đã detect ở bước trên
* Qua 8 bước thu được ảnh crop theo hóa đơn; tuy nhiên 1 số ảnh việc cắt vẫn chưa thực sự tốt

Dưới đây là một vài hình ảnh kết quả sau khi preprocess: bên trái ảnh đã cắt, bên phải ảnh chưa cắt

|  |  |
| --- | --- |
|  |  |
|  |  |
|  |  |
|  |  |

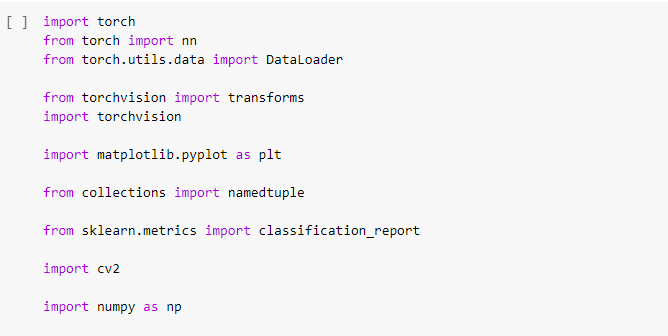
b. Thu thập dữ liệu datasets

Tập dữ liệu dataset bao gồm:

* Tập train: HighLands gồm 146 ảnh, Other gồm 78 ảnh, PhucLong gồm 80 ảnh, StarBuck gồm 164 ảnh
* Tập test: HighLands gồm 59 ảnh, Other gồm 78 ảnh, PhucLong gồm 31 ảnh, StarBuck gồm 30 ảnh

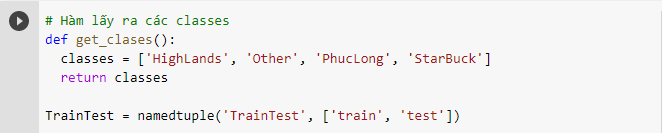
c. Triển khai mô hình

*Khai báo thư viện*



*Chuẩn bị dữ liệu*

* Các nhãn: HighLands, PhucLong, Starbucks, Other
* Kích cỡ ảnh train và test: 224x224
* Data Augmentation: RandomCrop, RandomHorizontalFlip, RandomVerticalFlip
* Đưa vào batch để tận dụng khả năng xử lý song song của GPU
* Folder train và test lần lượt trong: traindir và testdir

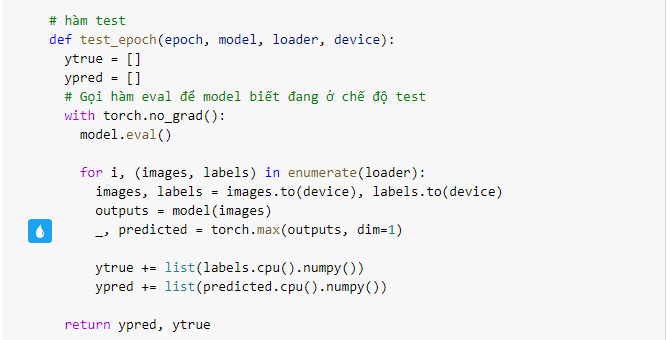
****

****

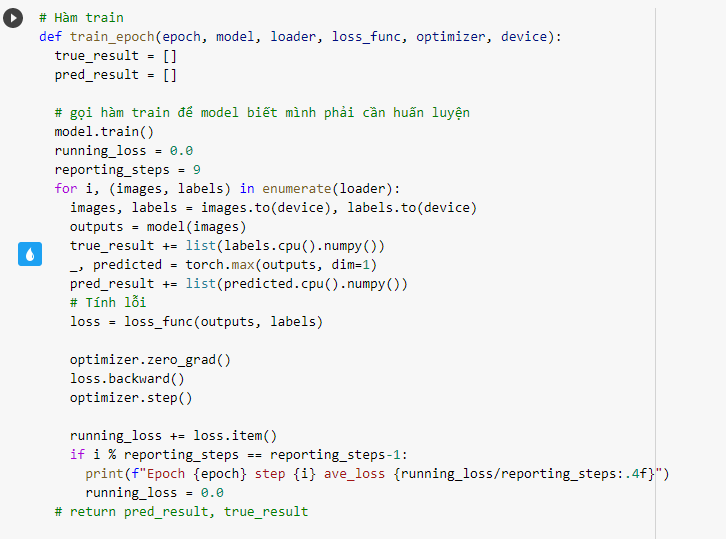
*Hàm train và test*

* BatchSize = 88
* Report sau mỗi 99 epoch
* Train: cho đi qua model và tính lỗi bằng loss\_func sau đó cập nhật tham số mới
* Test: khai báo chế độ eval (đánh giá - evaluate) và trả về nhãn dự đoán và nhãn thực tế

Hàm test:



Hàm train:

**

*Hàm main*

* model\_in = 'desnet': tên mạng Cnn đùng để huấn luyện
* PATH: đường dẫn lưu trọng số sau khi huấn luyện
* Đặc trưng ra = 4 vì có 4 lớp HighLands, PhucLong, Starbucks, Other
* Hàm lỗi được sử dụng: Cross-Entropy
* Hàm tối ưu đước sử dụng: SGD - Stochastic Gradient Descent

