**Grad-cam**

**Grad-cam hoạt động bằng cách:**

1. **Tìm lớp tích chập cuối cùng trong mạng(finding the final convolutional layer in the network)**
2. **Kiểm tra thông tin gradient( độ dốc) thông tin chảy vào lớp đó.( examining the gradient information flowing into that layer)**

**Đầu ra của Grad-CAM là một trực quan hóa bản đồ nhiệt cho một nhãn lớp nhất định** ( nhãn hàng đầu, nhãn dự đoán hoặc nhãn tùy ý mà chúng tôi chọn để gỡ lỗi). Chúng ta có thể sử dụng bản đồ nhiệt này để xác minh trực quan vị trí trong hình ảnh mà CNN đang tìm kiếm

Ưu điểm của phương pháp grad-cam:

* Dễ dàng thực hiện
* Hoạt động với gần như mọi kiến trúc mạng thần kinh chuyển đổi(Convolutional Neural Network architecture)
* Có thể sử dụng để gỡ lỗi trực quan(visually debug) nơi mạng đang tìm kiếmtrong một hình ảnh.

1. ***Thuật toán cnn***

Cấu trúc cnn:

CNN bao gồm tập hợp các lớp cơ bản bao gồm:

* Convolution layer + Nonlinear layer
* Pooling layer
* Fully connected layer.

Các lớp này liên kết với nhau theo một thứ tự nhất định. Thông thường:

1. Một ảnh sẽ được lan truyền qua tầng convolution layer + nonlinear layer đầu tiên
2. Sau đó các giá trị tính toán được sẽ lan truyền qua pooling layer, bộ ba convolution layer + nonlinear layer + pooling layer có thể được lặp lại nhiều lần trong network.
3. Và sau đó được lan truyền qua tầng fully connected layer và softmax để tính sác xuất ảnh đó chứa vật thế gì

* **Convolution layer**

Chức năng chính là phát hiện tính đặc trưng không gian hiệu quả. Các đặc trưng bao gồm đặc trưng cơ bản là góc,cạnh, màu sắc, hoặc đặc trưng phức tạp hơn như texture của ảnh.

Có 4 đối tượng ở tầng này: **ma trận dầu vào (1 ma trận 3 chiều)**, **bộ filters(** kích thước filter hầu hết đều là số lẻ ), **receptive field** và **feature map**. Bộ filters sẽ trượt qua qua từng vị trí (từ trái qua phải và từ trên xuống dưới) của trên bức ảnh để tính tích chập(convolution) giữa **bộ filter** và **phần tương ứng trên bức ảnh**, và phần tương ứng đó goi là **receptive field** ( có thể hiểu nó là vùng mà neuron có thể nhìn thấy để đưa ra quyết định). Ma trận đầu ra bởi quá trình này gọi là **featue map**. Do bộ filter trượt theo phương pháp trên nên ta có thể thấy *phép convolution bảo toàn thứ tự không gian của các điểm ảnh*.

Note: Khi các bạn áp dụng phép convolution thì ma trận đầu vào sẽ có nhỏ dần đi, do đó số layer của mô hình CNN sẽ bị giới hạn, và không thể xậy đựng deep nets mong muốn. Để giải quyết tình trạng này một cách đơn giản và phổ biến nhất để padding là sử dụng hàng số 0, ngoài ra các bạn cũng có thể sử dụng reflection padding hay là symmetric padding.

* **Nonlinear layer**

ReLU (Rectified Linear Units, f = max(0, x)) là hàm kích hoạt phổ biến nhất cho CNN. Hàm ReLU được ưa chuộng vì tính toán đơn giản, giúp hạn chế tình trạng vanishing gradient, và cũng cho kết quả tốt hơn. ReLU cũng như những hàm kích hoạt khác, được đặt ngay sau tầng convolution, ReLU sẽ gán những giá trị âm bằng 0 và giữ nguyên giá trị của đầu vào khi lớn hơn 0.

Note: Hàm này không có đạo hàm tại điểm 0, nên cẩn thận khi chọn learning rate khi sử dụng.

* **Pooling layer**

Một số loại pooling layer phổ biến như là max-pooling, average pooling, với chức năng chính là giảm chiều của tầng trước đó.

Ý tương đằng sau tầng pooling là vị trí tuyết đối của những đặc trưng trong không gian ảnh không còn cần cần thiết, thay vào đó vị trí tương đối giữ các đặc trưng đã đủ để phân loại đối tượng. Hơn nữa tầng pooling có khả năng giảm chiều cực kì nhiều, làm hạn chế overfit, và giảm thời gian huấn luyện tốt.

* **Fully connected layer**

Tầng này có chức năng chuyển ma trận đặc trưng ở tầng trước thành vector chứa xác suất của các đối tượng cần được dự đoán. Ví dụ, trong bài toán phân loại số viết tay MNIST có 10 lớp tương ứng 10 số từ 0-1, tầng fully connected layer sẽ chuyển ma trận đặc trưng của tầng trước thành vector có 10 chiều thể hiện xác suất của 10 lớp tương ứng.

Và cuối cùng, quá trình huấn luyện mô hình CNN cho bài toán phân loại ảnh cũng tương tự như huấn luyện các mô hình khác. Chúng ta cần có hàm độ lỗi để tính sai số giữ dự đoán của mô hình và nhãn chính xác, cũng như sử dụng thuật toán backpropagation cho quá trình cập nhật trọng số.

Tài liệu tham khảo:

<https://www.pyimagesearch.com/2020/03/09/grad-cam-visualize-class-activation-maps-with-keras-tensorflow-and-deep-learning/>

<https://pbcquoc.github.io/cnn/>

Chuyển vào thư muc cần chạy

**Install các package sau:**

pip install opencv-python

pip install --upgrade tensorflow==2.0.0-rc1

pip install imutils

**run:**

python apply\_gradcam.py --image images/space\_shuttle.jpg

Phân tích code trong file **apply\_gradcam.py:**

argparse.ArgumentParser(): hàm này dung để phân tích( parse ) các tham số và tùy chọn trên command line.

Cấu trúc ArgumentParser.add\_argument(name or flags...[, action][, nargs][, const][, default][, type][, choices][, required][, help][, metavar][, dest])

* name or flags: tên hoặc 1 danh sách các tên tùy chọn đó. Ví dụ: foo, -f, -foo…
* action: tên của action sẽ được thực hiện khi tham số được nhập. Có các action sau:
  + store: Lưu lại giá trị người dùng đã nhập vào name or flag. Action này là action mặc định.
  + store\_const: Lưu giá trị nhận được vào tham số const
  + store\_true và store\_false: Một trường hợp đặc biệt của store\_const giúp bạn lưu giá trị boolean vào name or flag

Tài liệu tham khảo: <https://viblo.asia/p/pymotm-argparse-l5XRBVeVRqPe#_vi-du-2>.

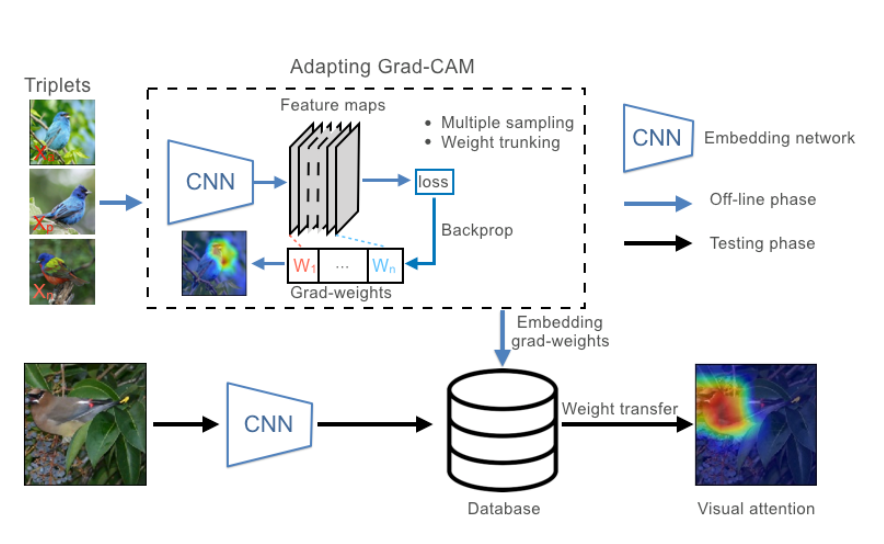
<https://github.com/insikk/Grad-CAM-tensorflow>

<https://keras.io/examples/vision/grad_cam/>

pip install pydicom

pip install dicom

Phương pháp grad-cam



Đầu vào của phương pháp của chúng tôi là một mạng được đào tạo trước và một hình ảnh. Đầu ra là một bản đồ chú ý trực quan theo phong cách CAM (bản đồ nhiệt).

<https://arxiv.org/pdf/1808.08114.pdf>

<https://www.arxiv-vanity.com/papers/2001.06538/>